User's Manual

DL6000/DLM6000 Series Digital Oscilloscope/ Mixed Signal Oscilloscope Serial Bus Signal Triggering and Analysis Function (The Trigger/analysis features of I^2C bus signal, CAN bus signal, LIN bus signal, SPI bus signal, and UART signal)

YOKOGAWA
Yokogawa Electric Corporation

IM DLM6054--51EN
1st Edition

Thank you for purchasing a DL6000 Series Digital Oscilloscope with the Serial Bus Signal Analysis Function or DLM6000 Series Mixed Signal Oscilloscope with the Serial Bus Signal Analysis Function. This user's manual describes the serial bus signal analysis feature.[1]

1 Analyzable signal types vary depending on the installed options.

/F3 option	I^2C bus signals, SPI bus signals, and UART signals
/F4 option	CAN bus signals, LIN bus signals, and UART signals

For information about other features, operating procedures, and handling precautions of the DL6000 and DLM6000 Series, see the following manuals.

Manual Title	Manual No.	Description
DL6000/DLM6000 Series Digital Oscilloscope Mixed Signal Oscilloscope User's Manual	IM DLM6054-01EN	Explains all features and procedures of the DL6000/DLM6000 Series excluding the communication features.
DL6000/DLM6000 Series Digital Oscilloscope Mixed Signal Oscilloscope Communication Interface User's Manual (in CD)	IM DLM6054-17EN	Explains the communication interface features of the DL6000/DLM6000 Series.
DL6000/DLM6000 Series Digital Oscilloscope Mixed Signal Oscilloscope Power Supply Analysis Function User's Manual	IM DLM6054-61EN	Explains the operating procedures of the optional power supply analysis feature.

Notes

- To upgrade to the latest firmware version, go to the following Web page, and then browse to the download page.
  http://tmi.yokogawa.com/service-support/downloads/
- The contents of this manual are subject to change without prior notice as a result of continuing improvements to the instrument's performance and features. The figures given in this manual may differ from the actual 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 Electric Corporation is strictly prohibited.

1st Edition : Octoberl 2009 (YK)

Trademarks

- DLM is pending trademark of Yokogawa Electric Corporation.
- Adobe, Acrobat, and PostScript are trademarks of Adobe Systems Incorporated.
- For purposes of this manual, the TM and ® symbols do not accompany their respective trademark names or registered trademark names.
- Other company and product names are trademarks or registered trademarks of their respective companies.

Revisions

- 1st Edition: October 2009

Symbols and Notations Used in This Manual

Safety Markings

The following markings are used in this manual.

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

Notation Used in the Procedural Explanations

On pages that describe the operating procedures in chapters 2 through 4, the following notations are used to distinguish the procedures from their explanations.

Procedure Carry out the procedure according to the step numbers. All procedures are written with inexperienced users in mind; experienced users may not need to carry out all the steps.

Explanation This section describes the setup items and the limitations regarding the procedures.

Notation of User Controls

Panel/Soft Key Names and Menu Items Set in Boldface

Boldface type indicates the names of user-controlled panel keys, and soft key items and menu items displayed on screen.

SHIFT+Panel Key

The SHIFT+Panel key means you will press the SHIFT key to turn ON the indicator of SHIFT key and then press the panel key. The menu marked in purple below the pressed key appears on the screen.

Unit

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

Communication Command Notation

In the detailed explanations of the communication commands in chapter 7, DLM6000-specific commands are written in blue italics.
These commands are not available on the DL6000.

Contents

1.1 I²C Bus Signal

I²C Bus is an abbreviation for Integrated Circuit Bus. It is a bidirectional bus for connecting ICs. By using this feature, you will be able to analyze data while displaying I²C Bus signal waveforms. The I²C bus signal analysis feature consists of the following three main features.

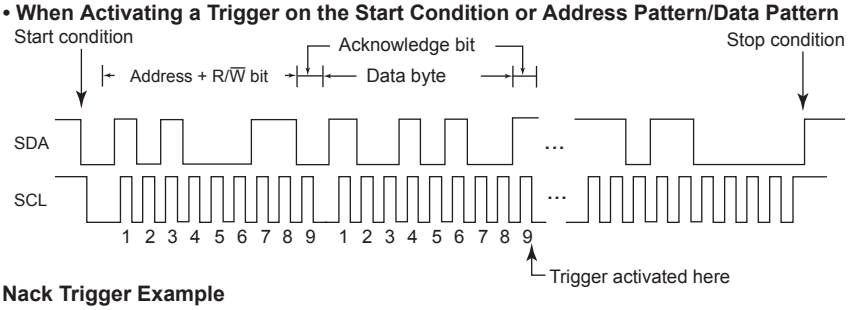
Trigger ►For the procedure, see section 3.1

A trigger can be activated under the following conditions.

- When a start condition is detected.
- When a Nack is detected.
- When the specified address pattern (7-bit address, 7-bit address + sub address, or 10-bit address) is met.
- When the data pattern is met or not met.
- When a specified general call address is detected.

Address & Data Trigger Example

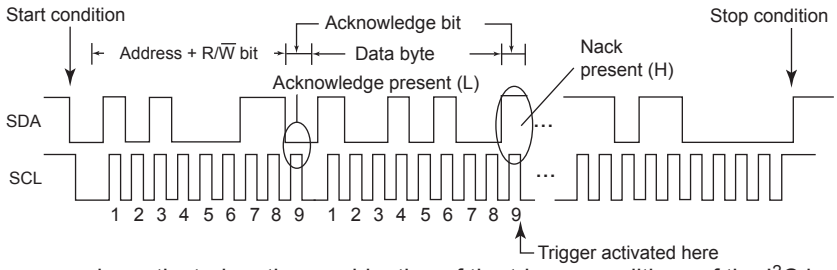
• When Activating a Trigger on the Start Condition or Address Pattern/Data Pattern



Nack Trigger Example

• When Activating a Trigger When the Acknowledge Bit Is Not Present

(When the SDA Signal is high)



A trigger can be activated on the combination of the trigger conditions of the I²C bus signal and analog signal (event interval trigger or B trigger). For details on the event interval trigger or B trigger, see section 6.11 or section 6.12 in the *User's Manual IM DLM6054-01EN*.

Analysis ►For the procedure, see section 4.2

This feature analyzes the I²C bus signal data and shows a list of the analysis results. There are two types of analysis result lists: simple and detail. The simple list displays the analysis number, start and stop conditions, analysis data, address and data types, read/write signal, and the status of the Acknowledge bit for each byte. The detail list displays the time from the trigger position and data information in addition to the items displayed by the simple list. The data of the detail list can be saved to an arbitrary storage medium in CSV format. In addition, you can select an arbitrary byte in the analysis result list and move the zoom position (the center of the zoom box) to the head of that byte.

Search ►For the procedure, see section 5.2

This feature searches for data that matches a specific address pattern, data pattern, or Acknowledge bit status in the I²C bus signal data. When the search is executed, the zoom box (ZOOM1 or ZOOM2) moves to the data position where the conditions are met, and the data is displayed expanded in the zoom window.

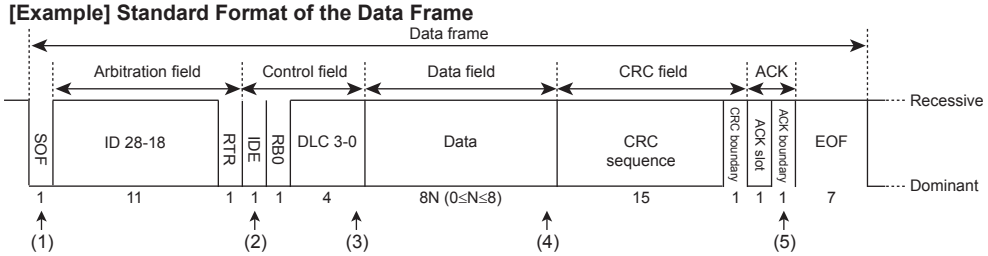
1.2 CAN Bus Signal

CAN stands for Controller Area Network. It is a serial communication protocol standardized internationally by the ISO (International Organization for Standardization).
By using this feature, you will be able to analyze data while displaying signal waveforms on the CAN bus as analog waveforms.
This CAN bus signal analysis feature consists of the following three main features.

Trigger ►For the procedure, see section 3.2

By setting the ID bit pattern, DLC, Data, and ACK slot status of the CAN bus, a trigger can be activated on a specific data frame or remote frame. Up to four ID/Data conditions can be specified allowing triggers to be activated on their OR conditions. In addition, the SOF (Start of Frame) or error frame can be used as a trigger condition.

[Example] Standard Format of the Data Frame



(1) Trigger point when the trigger condition is set to SOF
(2) Trigger point when the trigger condition is set only on the ID bit pattern
(3) Trigger point when the trigger condition is set on the ID bit pattern and DLC
(4) Trigger point when the trigger condition is set on the ID bit pattern and Data bit pattern
(5) Trigger point when the trigger condition is set to the ACK slot state

A trigger can be activated on the combination of the trigger conditions of the CAN bus signal and analog signal (event interval trigger or B trigger). For details on the event interval trigger or B trigger, see section 6.11 or section 6.12 in the *User's Manual IM DLM6054-01EN*.

Analysis ►For the procedure, see section 4.3

This feature analyzes the CAN bus signal data and shows a list of the analysis results. There are two types of analysis result lists: simple and detail. The simple list displays the analysis number, the type of analyzed frame, ID, Data, ACK slot status for each frame. The detail list displays the time from the trigger position, DLC, and CRC sequence in addition to the items displayed by the simple list. The data of the analysis result can be saved to an arbitrary storage medium in CSV format.
You can select an arbitrary frame in the analysis results list and automatically display the CAN bus signal for that frame (zoom link). The zoom position (the center of the zoom box) can be moved to the head of a specified field of the frame (field jump).

Search ►For the procedure, see section 5.3

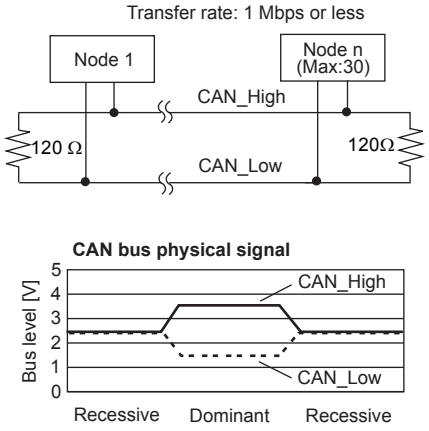
A specific frame or field can be searched on the CAN bus signal data. When the search is executed, the zoom box moves to the data position where the conditions are met, and the data is displayed expanded in the zoom window (Zoom1 or Zoom2).

High-speed CAN (ISO11898) and Low-speed CAN (ISO11519-2)

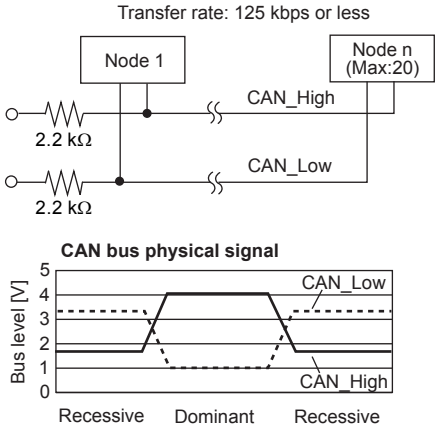
Representative standards for the CAN physical layer are High-speed CAN (ISO 11898) and Low-speed CAN (ISO 11519-2).

As shown in the following figure, the bus level is determined by the potential difference between two buses, CAN_High and CAN_Low, in either standard.

High speed CAN (ISO11898)



Low speed CAN (ISO11519-2)



Connecting the Probe

Probe to Be Used

A differential probe is used when measuring CAN bus signals.

Compatible differential probes: 701920, 701922, and 701924 by Yokogawa

When displaying the recessive voltage level higher than the dominant voltage level (Recessive: H)

For a two wire system (differential)

Connect the differential probe negative (–) to CAN_High, and the probe positive (+) to CAN_Low.

For a one wire system (single-ended)

Connect the differential probe negative (–) to CAN_High, and probe positive (+) to GND (ground potential).

When displaying the recessive voltage level less than the dominant voltage level (Recessive: L)

For a two wire system (differential)

Connect the differential probe negative (–) to CAN_Low, and the probe positive (+) to CAN_High.

For a one wire system (single-ended)*

Connect the differential probe negative (–) to GND (ground potential), and probe positive (+) to CAN_High.

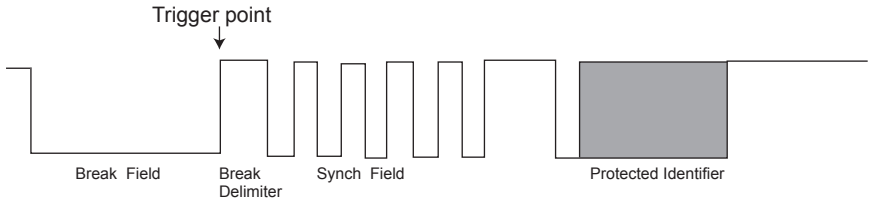
* In this case, the passive probe (model 701939) can be connected to CAN_High.

1.3 LIN Bus Signal

LIN stands for Local Interconnect Network. It is a serial communication protocol used mainly for automobiles and other vehicles.
By using this feature, you will be able to analyze data while displaying signal waveforms on the LIN bus as analog waveforms.
The LIN bus signal analysis feature consists of the following three main features.

Trigger ►For the procedure, see section 3.3

The trigger activates on the rising edge of the break delimiter. One of the following can be selected for the bit rate: 19200 bps, 9600 bps, 4800 bps, 2400 bps, 1200 bps, or User.



A trigger can be activated on the combination of the trigger conditions of the LIN bus signal and CAN bus signal, or of the LIN bus signal and analog signal (event interval trigger or B trigger). For details on the event interval trigger or B trigger, see section 6.11 or section 6.12 in the *User's Manual IM DLM6054-01EN*.

Analysis ►For the procedure, see section 4.4

This feature analyzes the LIN bus signal data and shows a list of the analysis results. There are two types of analysis result lists: simple and detail. The simple list displays the analysis number, ID, Data, and Checksum status. The detail list displays the time from the trigger position, ID field, ID parity error, and Checksum error in addition to the items displayed by the simple list. The data of the analysis result can be saved to an arbitrary storage medium in CSV format. You can select an arbitrary field in the analysis results list and automatically display the LIN bus signal for that field (zoom link).

Search ►For the procedure, see section 5.4

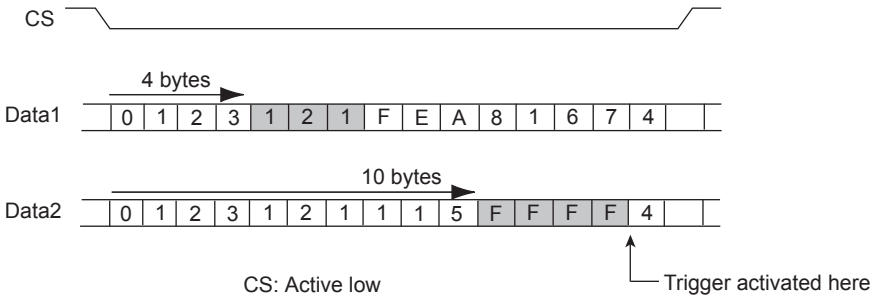
You can search for a specific field on the LIN bus signal data. When the search is executed, the zoom box moves to the data position where the conditions are met, and the data is expanded in the zoom window (Zoom1 or Zoom2).

1.4 SPI Bus Signal

The SPI (Serial Peripheral Interface) Bus is a synchronized serial bus that is widely used for inter-IC communications and data communications.
By using this feature, you will be able to analyze data while displaying the SPI Bus signal waveform.
The SPI Bus signal analysis feature consists of the following three main features.

Trigger ►For the procedure, see section 3.4

Acquires SPI Bus signals by comparing the specified conditions with the input signals at the byte level (8 bits).
The data position to be compared can be specified in terms of the number of bytes from the assertion of the chip select signal (CS). You can set two data patterns (Data 1 and Data 2) for the four-wire SPI and one data pattern for the three-wire SPI. For Data 1 and Data 2, a trigger is activated at the position where the latter data pattern matches.
An example is given below for the case when comparing Data 1 (3 bytes) from the 4th byte after the assertion of the CS and comparing Data 2 (4 bytes) from the 10th byte after the assertion of the CS and activating the trigger when both patterns match



CS: Active low

Data1	Condition:	True
	Data Size:	3
	Pattern:	121
	Data Position:	4
Data2	Condition:	True
	Data Size:	4
	Pattern:	FFFF
	Data Position:	10

A trigger can be activated on the combination of the trigger conditions of the SPI bus signal and analog signal (event interval trigger or B trigger). For details on the event interval trigger or B trigger, see section 6.11 or section 6.12 in the *User's Manual IM DLM6054-01EN*.

Analysis ►For the procedure, see section 4.5

This feature analyzes the SPI bus signal data and shows a list of the analysis results. Analysis occurs in sync with the clock signal (Clock), in segments whose size is determined by the field size (Field Size) and the enabled bit range (Enable MSB/LSB). There are two types of analysis result lists: simple and detail. The simple list displays the analysis number, Data 1/Data 2 (in hexadecimal notation), and CS status for each byte. The detail list displays the time from the trigger position, the start and end positions of the active period and Data 1/Data 2 (in binary notation) in addition to the items displayed by the simple list. The data of the detail list can be saved to an arbitrary storage medium in CSV format.

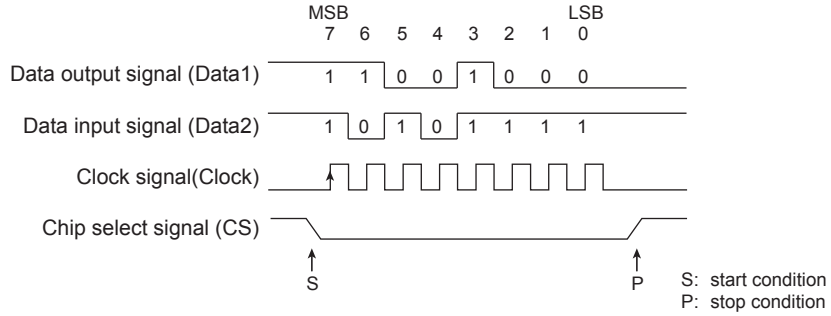
In addition, you can select an arbitrary byte in the analysis result list and move the zoom position (the center of the zoom box) to the head of that byte.

Search ►For the procedure, see section 5.5

This feature searches for data that matches a specific data pattern in the SPI bus signal data. When the search is executed, the zoom box moves to the data position where the conditions are met, and the data is displayed expanded in the zoom window (Zoom1 or Zoom2).

Analysis and Search Example

The table below indicates how the DL6000/DLM6000 analyzes or searches the signal shown in the figure below differently depending on the Field Size and Enable MSB/LSB settings.



Analysis and Search Conditions

- Data: Eight-bit Segment
- Bit order: MSB First

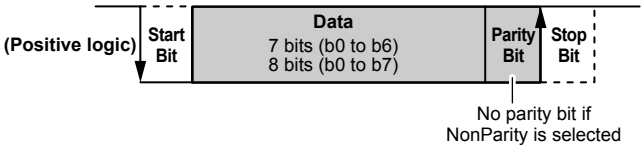
Analysis Conditions		Analysis Results		
Field Size	Enable MSB/LSB	Data1	Data2	CS
4bit	3 to 0 (4bit)	C, 8	A, F	L
		Analysis: The data is analyzed in two four-bit segments. Search: The DL6000/DLM6000 searches for four bits from the comparison start field.		
6bit	5 to 0 (6bit)	3, 2	2, B	L
		Analysis: The data is analyzed in one six-bit segment. (The bits are split into one two-bit segment and one four-bit segment.) Search: The DL6000/DLM6000 searches for six bits from the comparison start field.		
8bit	5 to 0 (6bit)	0, 8	2, F	L
		Analysis: The six least significant bits of one eight-bit segment of the data are analyzed. (The bits are split into one two-bit segment and one four-bit segment.) Search: The DL6000/DLM6000 searches for the least significant six bits in an eight-bit segment from the comparison start field.		
12bit		Analysis and searching do not take place.		

1.5 UART Signal

Universal Asynchronous Receiver Transmitter (UART) is an integrated circuit that performs serial-to-parallel conversion and parallel-to-serial conversion. UART is generally used in inter-device communication such as with EIA RS-232.
By using this feature, you will be able to analyze data while displaying UART signal waveforms.
The UART signal analysis feature consists of the following three main features.

Trigger ►For the procedure, see section 3.5

The trigger activates on the stop bit of all data frames. One of the following can be selected for the bit rate: 115200bps, 57600bps, 38400bps, 19200bps, 9600bps, 4800bps, 2400bps, 1200bps, or User.



The DL6000/DLM6000 Series digital oscilloscopes cannot trigger on the combination of UART signals and other signals.

Analysis ►For the procedure, see section 4.6

This feature analyzes the UART signal data and shows a list of the analysis results. There are two types of analysis result lists: simple and detail. The simple list displays the analysis number, Data, and error status. The detail list displays the time from the trigger position in addition to the items displayed by the simple list. The data of the analysis result can be saved to an arbitrary storage medium in CSV format. You can select an arbitrary data in the analysis results list and automatically display the UART signal for that data (zoom link).

Search ►For the procedure, see section 5.6

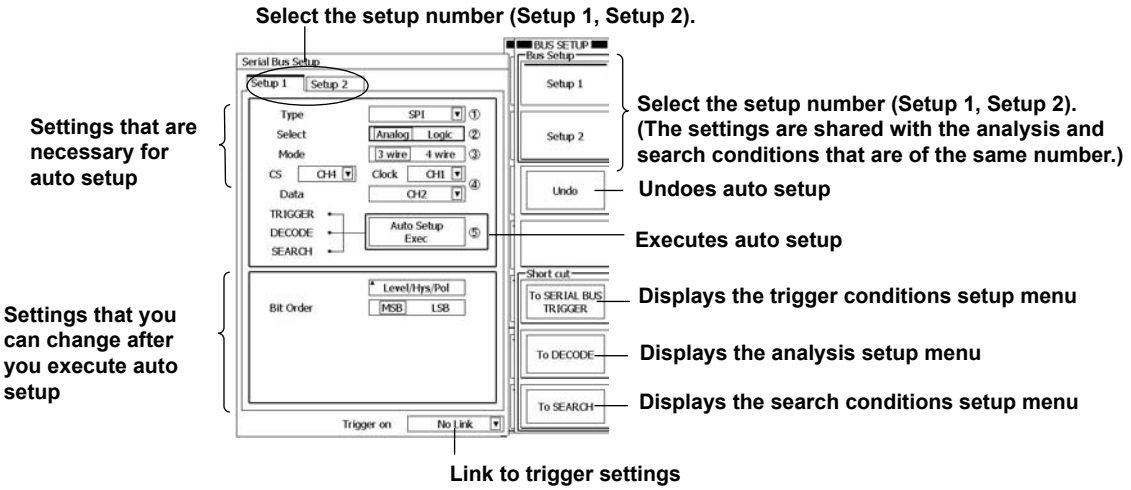
This feature searches for data that matches a specific data pattern or error status in the UART signal data. When the search is executed, the zoom box moves to the data position where the conditions are met, and the data is displayed expanded in the zoom window (Zoom1 or Zoom2).

2.1 Executing Serial Bus Signal Auto Setup

Procedure

SETUP Serial Bus Setup Menu

Press **SETUP** and then the **Serial Bus Setup** soft key to display the following menu.



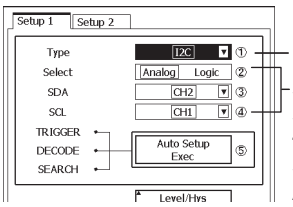
Note

If you set "Trigger on" to Setup 1 or Setup 2, the trigger settings change to the serial bus trigger settings that were specified on Setup 1 or Setup 2.

Settings That Are Necessary for Auto Setup

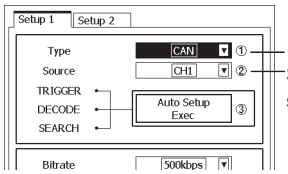
The settings differ depending on the type of serial bus signal.

I2C



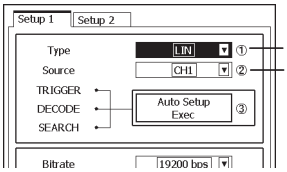
Set Type to I2C.
Select the SDA and SCL sources. (On the DL6000 series, Select is fixed to Analog, and this setting is not displayed.)

CAN



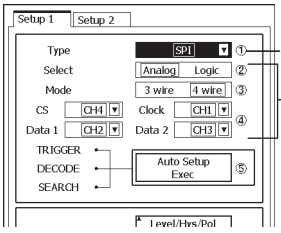
Set Type to CAN.
Select the source.

LIN



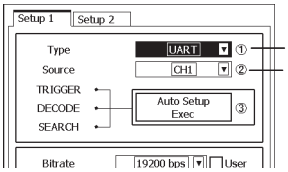
Set Type to LIN.
Select the source.

SPI



Set Type to SPI.
Select the mode and the CS, clock, and data sources. (On the DL6000 series, Select is fixed to Analog, and this setting is not displayed.)

UART



Set Type to UART.
Select the source.

Note

You cannot execute auto setup if Source is set to a waveform from M1 to M4 (the math waveforms).
Trigger, analysis, and search conditions use the same settings.

Executing Auto Setup

Select **Auto Setup Exec**, and press **SET**.

Note

You cannot execute auto setup when the serial bus signal type is set to SPI and the CS source is set to None.

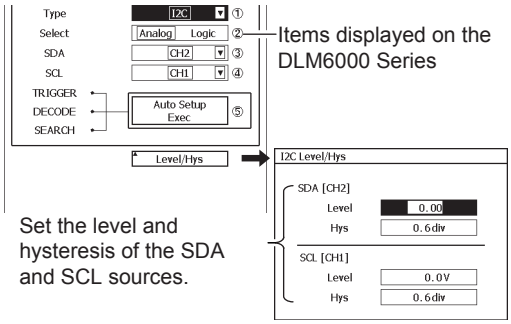
Undoing Auto Setup

Press the **Undo** soft key to set the settings back to the original values.

Adjusting the Settings after Auto Setup

The adjustable items vary depending on the serial bus signal type.

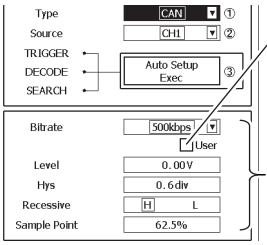
I2C



Items displayed on the DLM6000 Series

Set the level and hysteresis of the SDA and SCL sources.

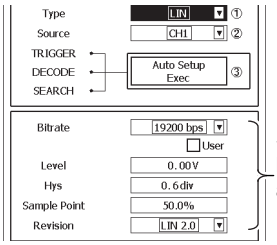
CAN



Select this check box to set any bit rate in the given range.

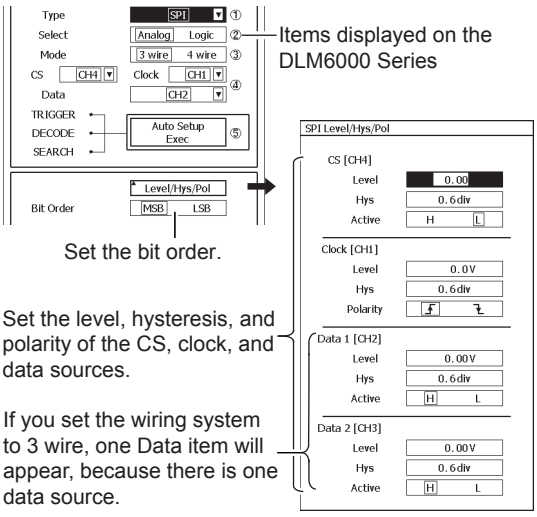
Set the bit rate, level, hysteresis, recessive level, and sample point.

LIN



Set the bit rate, level, hysteresis, sample point, and revision.

SPI



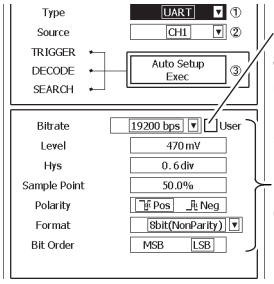
Items displayed on the DLM6000 Series

Set the bit order.

Set the level, hysteresis, and polarity of the CS, clock, and data sources.

If you set the wiring system to 3 wire, one Data item will appear, because there is one data source.

UART



Select this check box to set any bit rate in the given range.

Set the bit rate, level, hysteresis, sample point, polarity, format, and bit order.

Note

When you execute auto setup, all the settings other than those that are necessary for auto setup may be set to either their optimum values as determined by the DL6000/DLM6000 or their default values. Check the settings after you execute auto setup.

Explanation

Some of the trigger, analysis, and search settings of the I^2C, CAN, LIN, SPI, and UART serial bus signals can be automatically set up.
If you execute auto setup and the DL6000 or DLM6000 detects a serial bus signal, the trigger, analysis, and search settings will automatically be set to values appropriate for the input signal.

Setup Number (Setup 1 and Setup 2)

You can specify two sets of setup conditions. The setup conditions for Setup 1 are applied to the analysis conditions for Analysis 1 and the search conditions for Search 1. The same is true for Setup 2, Analysis 2 and Search 2. For details, see section 2.2.
You can also apply either set of setup conditions to the trigger conditions.

Settings Necessary for Auto Setup

Source

Select source signals* on which to perform auto setup according to the serial bus signal type.

I^2C	Select an SDA (serial data) source and an SCL (serial clock) source. If you set the Select box to Analog, select from CH1 to CH4. If you set the Select box to Logic, select from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from A0 to A7 or from C0 to C7 on the 16 bits model).
CAN	Select a source from CH1 to CH4.
LIN	Select a source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on the 16 bits model).
SPI	Select a CS (chip select) source, a clock source, and a data source. If you set the Select box to Analog, select from CH1 to CH4. If you set the Select box to Logic, select from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from A0 to A7 or from C0 to C7 on the 16 bits model).
UART	Select a source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on the 16 bits model).

* If you select a source from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7, set the threshold level. For the setup procedure, see section 5.2 in the *User's Manual IM DLM6054-01EN*.
If you select a source from M1 to M4, you will not be able to execute auto setup.

Wiring System

Select the wiring system only in the case of an SPI serial bus signal.

3 wire	One data line
4 wire	Two data lines

Note

Even if you do not execute auto setup, the settings that you have specified are applied to the analysis and search conditions. For details, see section 2.2.

Linking to the Trigger Settings (Trigger on)

You can link the settings in Setup 1 or Setup 2 to the trigger settings. The trigger settings change as soon as you set "Trigger on" to Setup 1 or Setup 2.
If you change the setup conditions, the changes are also applied to the trigger settings.
If you execute auto setup, "Trigger on" is set to the setup number that was executed, and the setup results are applied to the trigger settings.

Note

- If you change the trigger settings directly, "Trigger on" is set to No Link, and the link between the setup conditions and the trigger settings is cleared.
- Even if the trigger is set to Edge, if you set "Trigger on" to Setup 1 or Setup 2, the trigger setting will change to Enhanced.

Executing Auto Setup

You can execute auto setup to make the DL6000/DLM6000 change its settings based on the input signals.

The setup results are applied to the analysis and search conditions. For details, see section 2.2.

If you execute auto setup, “Trigger on” is set to the setup number that was executed, and the setup results are applied to the trigger settings. If the serial bus signal cannot be detected, an error message is displayed.

Signals That Auto Setup Can Be Used

Auto setup is possible on a serial bus signal when the following conditions are met.

Voltage	Amplitude greater than or equal to 200 mV (when the probe attenuation is set to 1:1)
Bit rate	Greater than or equal to 1200 bps
Frames	At least 5 frames over 10 seconds

Note

- Measurement will only be correct if the probe attenuation ratio is set properly. Be sure to set the probe attenuation ratio properly before executing auto setup. For the setup procedure, see section 6.6 in the *User’s Manual IM DLM6054-01EN*.
- If you select a source from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7, the voltage amplitude must exceed the threshold level. For operating instructions, see section 5.2 in the *User’s Manual IM DLM6054-01EN*.

Center Position after Auto Setup

The center position after auto setup will be 0 V.

Waveforms That Were Displayed before Auto Setup

If you execute auto setup, data in the acquisition memory will be overwritten, and waveforms that were displayed before auto setup will be cleared.

Undoing Auto Setup

You can revert to the settings before auto setup by pressing the Undo soft key. However, you cannot undo auto setup if you turn OFF the power, because the settings before auto setup will be discarded.

Items That Are Set to Default Values and Items That Are Set to Detected Values

When you execute auto setup, the items are set to default values or set to values that are detected from the signal as shown in the following table. Items that are not in the table maintain their current values.

I²C		
Items set to default values	Mode	Every Start
	Hysteresis	0.6div
	Qualification	Don't care
Items that are set to detected values	SDA and SCL source levels	
CAN		
Items set to default values	Mode	SOF
	Hysteresis	0.6div
	Sample point	62.5%
Items that are set to detected values	Bit rate	
	Source level	
	Recessive level	
LIN		
Items set to default values	Mode	Break
	Hysteresis	0.6div
	Sample point	50.0%
Items that are set to detected values	Bit rate	
	Source level	
	Revision	
SPI		
Items set to default values	Hysteresis	0.6div
Items that are set to detected values	CS, clock, and data source levels	
UART		
Items set to default values	Mode	Every Data
	Hysteresis	0.6div
	Sample point	50.0%
Items that are set to detected values	Bit rate	
	Source level	
	Polarity	

Adjusting Settings after Auto Setup

You can adjust the items according to the serial bus signal type. For the selectable ranges, see the referenced sections.

I²C	SDA and SCL source levels and hysteresis See "Explanation" in section 4.2.
CAN	Source signal's bit rate, level, hysteresis, recessive level, and sample point See "Explanation" in section 4.3.
LIN	Source signal's bit rate, level, hysteresis, sample point, and revision See "Explanation" in section 4.4.
SPI	Bit order; CS, clock, and data source levels, hysteresis, and polarities See "Explanation" in section 4.5.
UART	Source signal's bit rate, level, hysteresis, sample point, polarity, format, and bit order See "Explanation" in section 4.6.

Trigger, Analysis, and Search Features

For the procedure on how to use the trigger, analysis, and search features, see the referenced sections below.

	Trigger	Analysis	Search
I²C	Section 3.1	Section 4.2	Section 5.2
CAN	Section 3.2	Section 4.3	Section 5.3
LIN	Section 3.3	Section 4.4	Section 5.4
SPI	Section 3.4	Section 4.5	Section 5.5
UART	Section 3.5	Section 4.6	Section 5.6

2.2 Sharing of the Serial Bus Signal's Trigger, Analysis, and Search Settings

The DL6000/DLM6000 shares the trigger, analysis, and search settings. If you change a setting in one feature, the corresponding setting will also change in the other features.

How Setup Settings Affect Trigger, Analysis, and Search Settings

How Setup Settings Affect Analysis and Search Settings

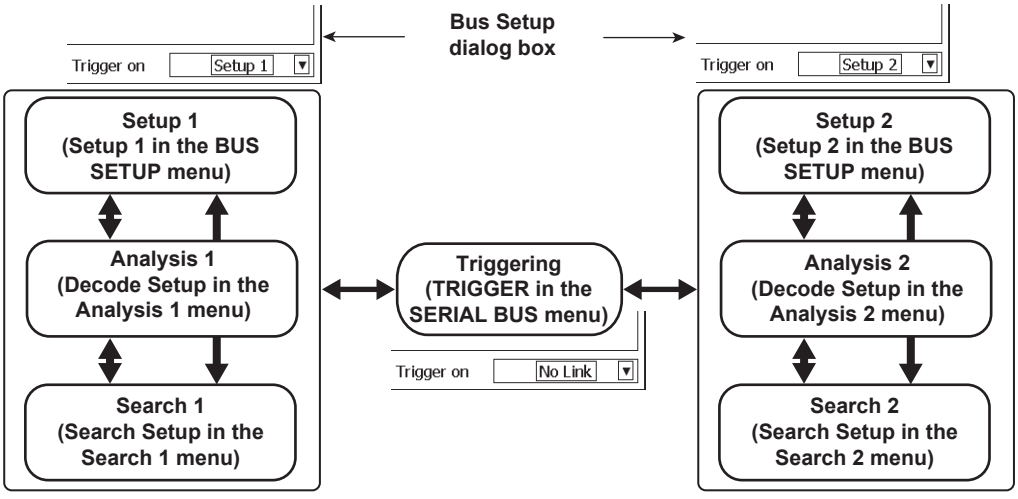
The setup (Setup 1 and Setup 2), analysis (Analysis 1 and Analysis 2), and search (Search 1 and Search 2) settings are linked together based on their number. If you execute auto setup, the setup results are applied to the analysis and search settings that have the same number.

Note

Even if you do not execute auto setup, the settings that you specified on the Serial Bus Setup menu are applied to the analysis and search conditions.

How Setup Settings Affect Trigger Settings

- If you set “Trigger on” to Setup 1 on the Serial Bus Setup menu, the Setup 1 settings are applied to the trigger settings. Likewise, if you select Setup 2, the Setup 2 settings are applied to the trigger settings.
- When you execute auto setup, the specified serial bus signal's type, source, and detected values (see page 2-5) are applied to the trigger settings (SERIAL BUS menu under TRIGGER). At the same time, the setup number (Setup 1 or Setup 2) that was executed during auto setup is displayed in the “Trigger on” box.



How Analysis and Search Settings Affect Setup and Trigger Settings

- **How Analysis and Search Settings Affect Setup Settings**

  If you change settings for Analysis 1 or Search 1, the corresponding settings in auto setup 1 will change. The same holds true for Analysis 2 and Search 2.

- **How Analysis and Search Settings Affect Trigger Settings**
- If “Trigger on” is set to Setup 1 and you change the Analysis 1 or Search 1 settings, the corresponding trigger settings will change. Changing the Analysis 2 or Search 2 settings will not affect the trigger settings. Likewise, if “Trigger on” is set to Setup 2 and you change the Analysis 2 or Search 2 settings, the corresponding trigger settings will change, but changing the Analysis 1 or Search 1 settings will not affect the trigger settings.
- If “Trigger on” is set to No Link, changing the Analysis 1 or 2 settings or Search 1 or 2 settings will not affect the trigger settings.

How Trigger Settings Affect Setup, Analysis, and Search Settings

- If "Trigger on" is set to Setup 1 and you change the trigger settings, the corresponding Setup 1, Analysis 1, and Search 1 settings will change. The Setup 2, Analysis 2, and Search 2 settings will not be affected. Likewise, if you set "Trigger on" to Setup 2 and you change the trigger settings, the corresponding Setup 2, Analysis 2, and Search 2 settings will be changed, but the Setup 1, Analysis 1, and Search 1 settings will not be affected.
- If you set "Trigger on" to No Link, changing the trigger settings will not affect any of the setup, analysis, or search settings.

Note

Even if "Trigger on" is set to Setup 1 or Setup 2, it will change to No Link if you select a trigger type other than I2C, CAN, LIN, SPI or UART. In particular, "Trigger on" will change to No Link if:

- You select TV or Serial in the ENHANCED menu under TRIGGER.
- You press EDGE/STATE, WIDTH, or EVENT INTERVAL on the front panel.

Note

Even if "Trigger on" is set to Setup 1 or Setup 2, it will change to No Link if you select a trigger type other than I^2C, CAN, LIN, SPI or UART. In particular, "Trigger on" will change to No Link if:

- You select TV or Serial in the ENHANCED menu under TRIGGER.
- You press EDGE/STATE, WIDTH, or EVENT INTERVAL on the front panel.

Common Items

The table below indicates the shared serial bus signal items by trigger type.

Trigger Type	I^2C	CAN	LIN	SPI	UART
Trigger type (Type)	Changes to the selected trigger type menu.				
Source	Yes	Yes	Yes	Yes	Yes
Bit rate (Bitrate)	No	Yes	Yes	No	Yes
Level	Yes	Yes	Yes	Yes	Yes
Hysteresis (Hys)	Yes	Yes	Yes	Yes	Yes
Sample point	No	Yes	Yes*	No	Yes*
Recessive level	No	Yes	No	No	No
Revision	No	No	Yes*	No	No
Polarity/active	No	No	No	Yes	Yes
Bit order	No	No	No	Yes	Yes*
Wiring system (Mode)	No	No	No	Yes	No
Format	No	No	No	No	Yes
Parity	No	No	No	No	Yes*

Yes: Shared item; No: Item not available

* The trigger feature does not have a setup menu.

Hysteresis

The hysteresis values that are applied from the trigger settings to the setup, analysis, and search settings are shown in the following table.

Trigger Setting	Setup, Analysis, and Search Setting
	0.6 div
	1.0 div

The hysteresis values that are applied from the setup, analysis, and search settings to the trigger settings are shown in the following table.

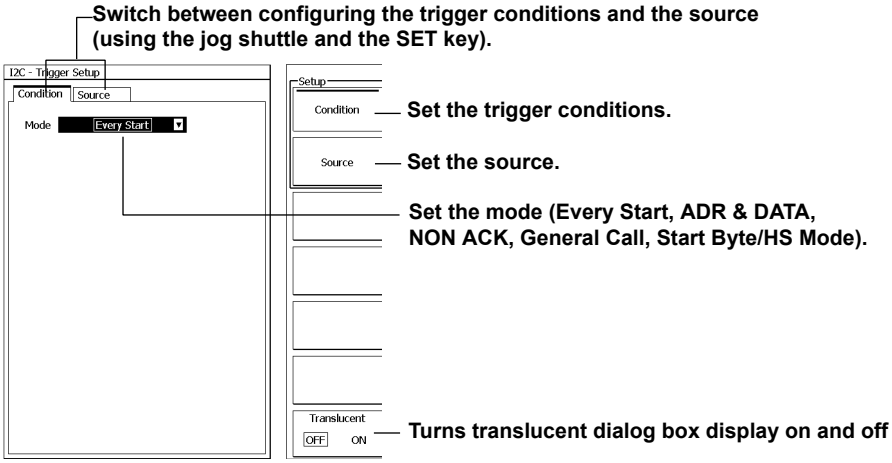
Setup, Analysis, and Search Setting	Trigger Setting
0.9 div or less	
1.0 div or greater	

3.1 Triggering on an I²C Bus Signal

Procedure

Setup Menu

Press **ENHANCED**, the **Type** soft key, the **Serial Bus** soft key, the **I2C** soft key, and then the Setup soft key to display the following menu.

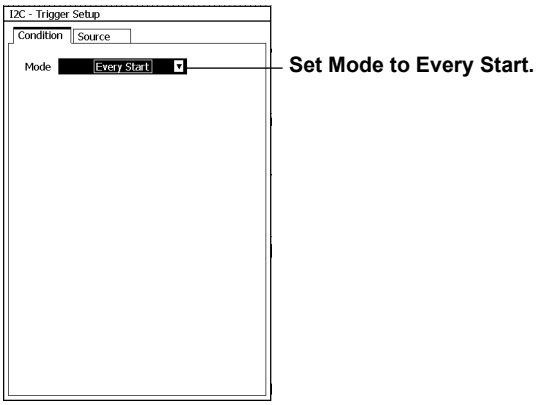


Setting Trigger Conditions (Condition)

Depending on the address to compare, there are five modes:

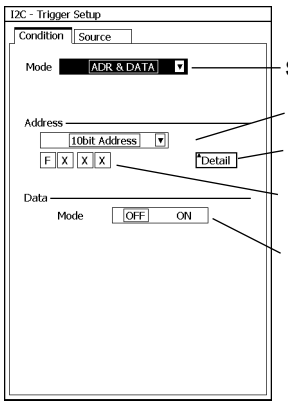
- Every Start
- ADR & DATA
- NON ACK
- General Call
- Start Byte/HS Mode

When Triggering in Every Start Mode



When Triggering in ADR & DATA Mode

When 10bit Address Is Selected



Set Mode to ADR & DATA.

Set the address.

Press to configure the address pattern details.

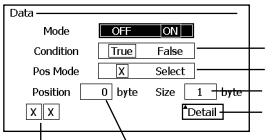
Set the address pattern.

Set whether or not to make the data pattern a trigger condition.

When 7bit Address Is Selected

When 7bit + Sub Address Is Selected

Making a Data Pattern a Trigger Condition



Set the comparison condition.

Set the comparison method.

Set the data length.

Press to configure the data pattern details.

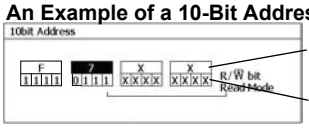
Set the data pattern.

Set the comparison start position for when Pos Mode is set to Select.

Detailed Address Pattern and Data Pattern Configuration

Press Detail to display a dialog box, and then use the jog shuttle and the soft keys to set the address or data pattern.

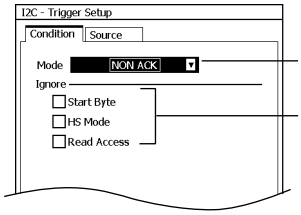
An Example of a 10-Bit Address Pattern



Hexadecimal

Binary

When Triggering in NON ACK Mode

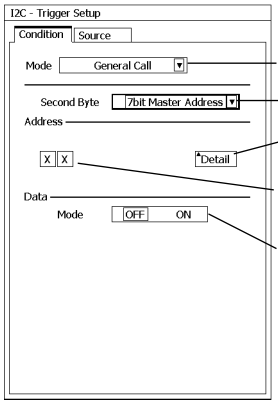


Set Mode to NON ACK.

Select the target acknowledge bits.

When Triggering in General Call Mode

When 7bit Master Address Is Selected



Set Mode to General Call.

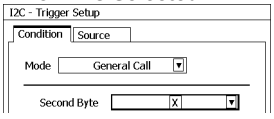
Set the second byte.

Press to configure the address pattern details.

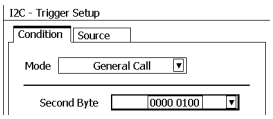
Set the address pattern.

Set whether or not to make the data pattern a trigger condition.

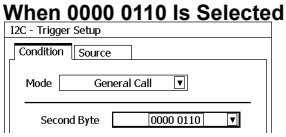
When X Is Selected



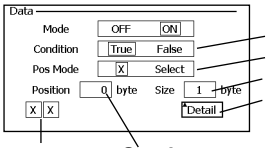
When 0000 0100 Is Selected



When 0000 0110 Is Selected



Making a Data Pattern a Trigger Condition



Set the comparison condition.

Set the comparison method.

Set the data length.

Press to configure the data pattern details.

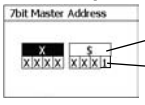
Set the data pattern.

Set the comparison start position for when Pos Mode is set to Select.

Detailed Address Pattern and Data Pattern Configuration

Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the address or data pattern.

An Example of a 7-Bit Master Address Pattern



Hexadecimal

Binary

When Triggering in Start Byte/HS Mode

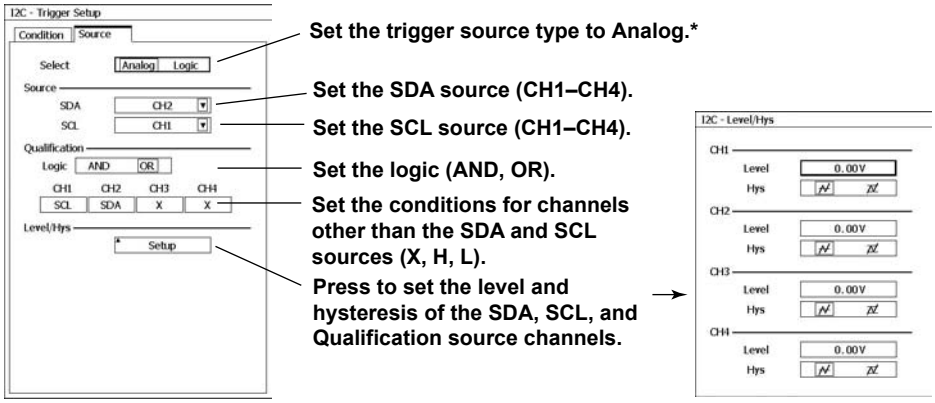
Set Mode to Start Byte/HS Mode.
Select Start Byte or HS Mode.

Setting the Trigger Source (Source)

Specify the trigger source to compare with the data pattern and the comparison conditions.

Select the **Source** tab, or press the **Source** soft key to display the following menu.

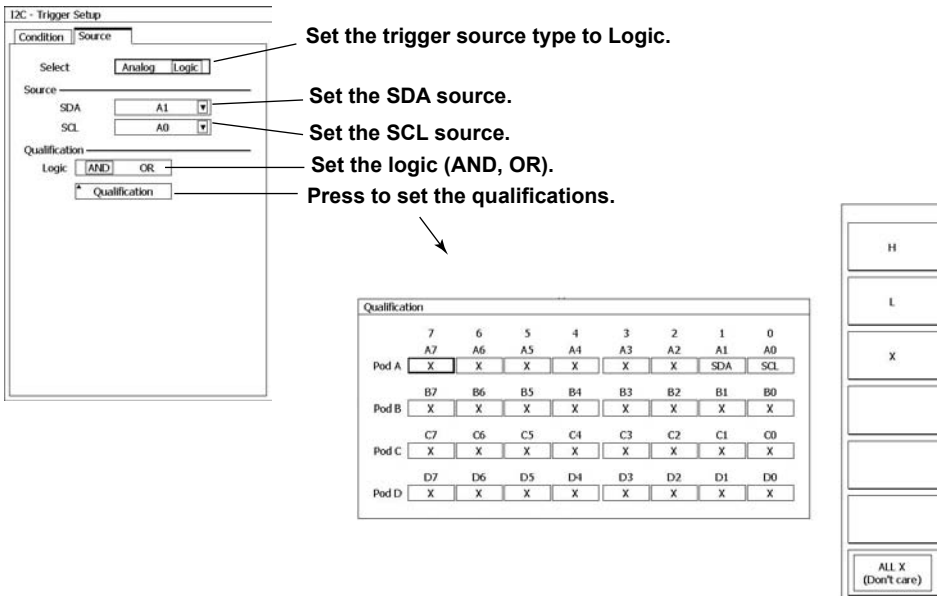
For Analog Sources



Set the trigger source type to Analog.*
Set the SDA source (CH1–CH4).
Set the SCL source (CH1–CH4).
Set the logic (AND, OR).
Set the conditions for channels other than the SDA and SCL sources (X, H, L).
Press to set the level and hysteresis of the SDA, SCL, and Qualification source channels.

*** This is fixed to Analog on the DL6000 series, and the setting is not displayed.**

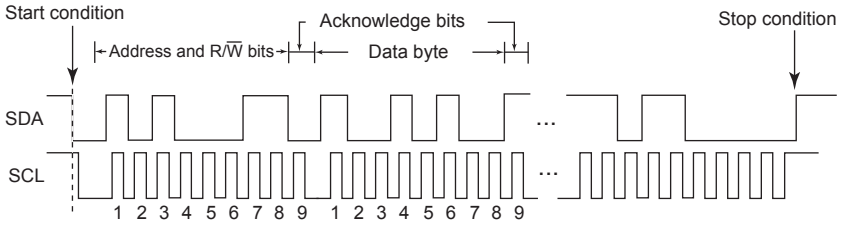
For Logic Sources (DLM6000)



Set the trigger source type to Logic.
Set the SDA source.
Set the SCL source.
Set the logic (AND, OR).
Press to set the qualifications.

Explanation

This feature triggers on I²C bus signals. The following figure shows the data format of I²C bus signals.

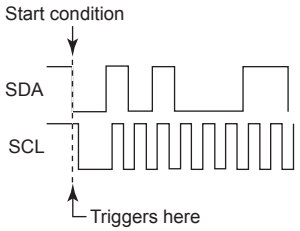


Mode

Select the I²C trigger mode from Every Start, ADR & DATA, NON ACK, General Call, and Start Byte/HS Mode.

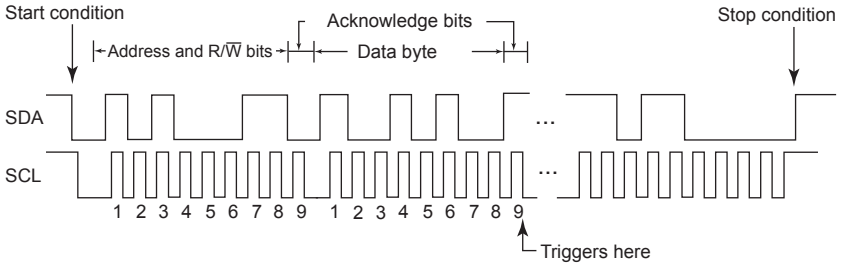
Every Start Mode

When a start condition is detected, the DL6000/DLM6000 triggers on the falling edge of the SDA signal.



ADR & DATA Mode

When the address and data values match, the DL6000/DLM6000 triggers on the falling edge of the 9th SCL signal clock.



- **Address**
  - You can set the address type to 7bit Address, 7bit + Sub Address, or 10bit Address.
  - Set the address pattern in hexadecimal or binary notation. The address trigger condition is met when the specified address pattern matches the input signal address pattern.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

- **Data**

  You can select whether or not to use the data pattern as a trigger condition.

  - Comparison Condition

    The data trigger condition is met when the result of comparing the input signal pattern with the specified pattern meets the selected comparison condition.

True	When the patterns match
False	When the patterns don't match

  - Comparison Start Position

    In the Pos Mode setting, you can set the comparison start point to the specified point (Select) or don't care (X). If you select Select, the DL6000/DLM6000 skips the specified number of bytes and starts comparing from the next data byte.

    Selectable range: 0 to 9999 bytes
  - Data Size

    Set how many consecutive data bytes you want to compare.

    Selectable range: 1 to 4 bytes
  - Data Pattern

    Set the data pattern for the specified size in hexadecimal or binary notation.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be "$."

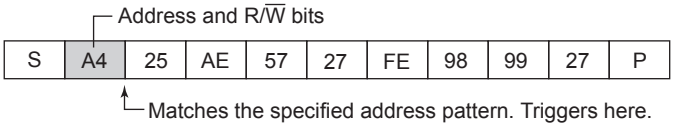
- **Example**

  This example displays the data sequence at the byte level (hexadecimal notation) and indicates the trigger position. The following notations are used in the figure.

  S: Start condition, P: Stop condition, Shading: Compared pattern

  Trigger only on the address pattern

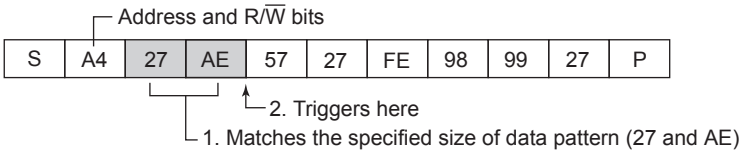
Mode	ADR & DATA
Address	7bit address, A4
Data	Mode: OFF



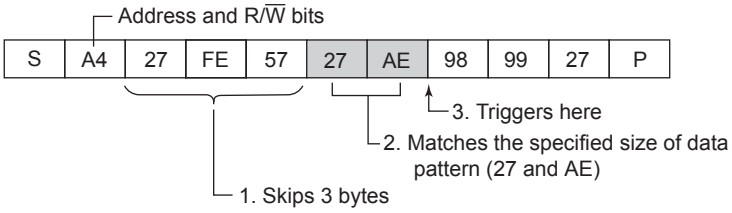
  Trigger only on the data pattern

Mode	ADR & DATA
Address	Don't care
Data	Mode: ON, Condition: True, Size: 2 bytes, Data pattern: 27 and AE

  < Pos Mode: X >

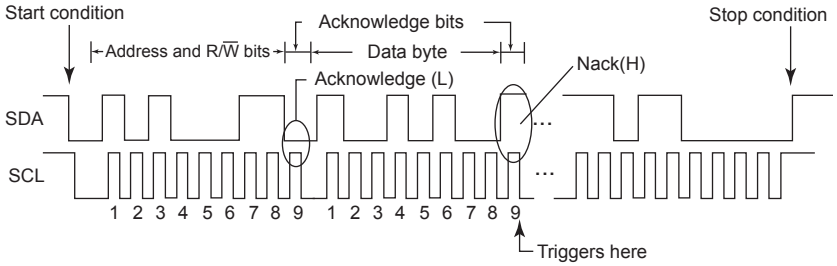


  < Pos Mode: Select, Position: 3 >



NON ACK Mode

The DL6000/DLM6000 triggers when the acknowledge bit is Nack (when the SDA signal is high). You can select whether use or ignore the following acknowledge bits for triggering: start byte, HS mode master code, and read access byte.



General Call Mode

The DL6000/DLM6000 triggers on the general call address (0000 0000).

- **Second Byte**

  You can use the second byte address pattern (the byte after the general call address) as a trigger condition. The second byte trigger condition is met when the specified pattern matches the input signal pattern.

X	Not used as a trigger condition
0000 0100	When the input signal pattern matches the pattern 0000 0100 (0x04)
0000 0110	When the input signal pattern matches the pattern 0000 0110 (0x06)
7bit Master Address	When the input signal pattern matches the specified pattern If you select 7bit Master Address, you can use the data pattern as a trigger condition as described in the next section.

- **Data**

  The conditions and settings are the same as those explained on page 3-6. See the respective item for details.

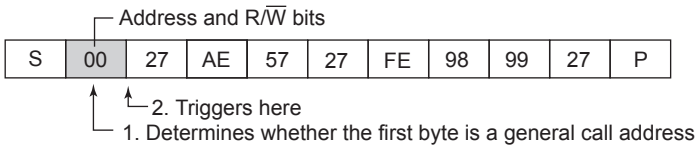
- **Example**

  This example displays the data sequence at the byte level (hexadecimal notation) and indicates the trigger position. The following notations are used in the figure.

  S: Start condition, P: Stop condition, Shading: Compared pattern

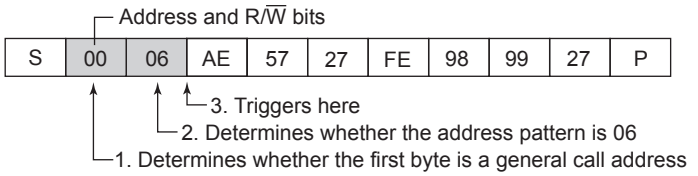
  Trigger only on the general call address

Mode	General Call
Second Byte	X



  Trigger when the second byte address is 06

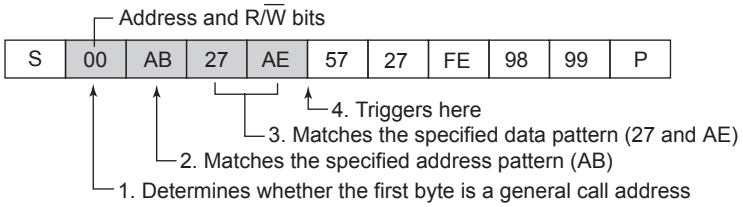
Mode	General Call
Second Byte	0000 0110



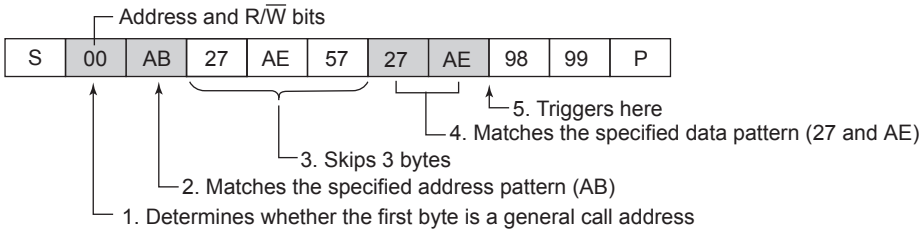
Trigger when the second byte address matches the specified pattern

Mode	General Call
Second Byte	7bit Master Address, address pattern: 1010 1011 (0xAB)
Data	Mode: ON, Condition: True, Size: 2 bytes, Data pattern: 27 and AE

< Pos Mode: X >



< Pos Mode: Select, Position: 3 >

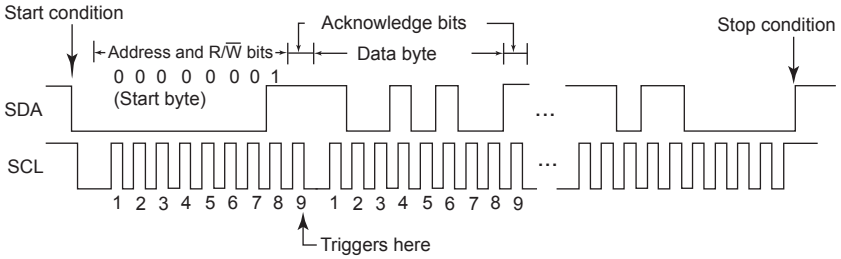


Start Byte/HS Mode

The DL6000/DLM6000 triggers on the start byte or the HS mode master code.

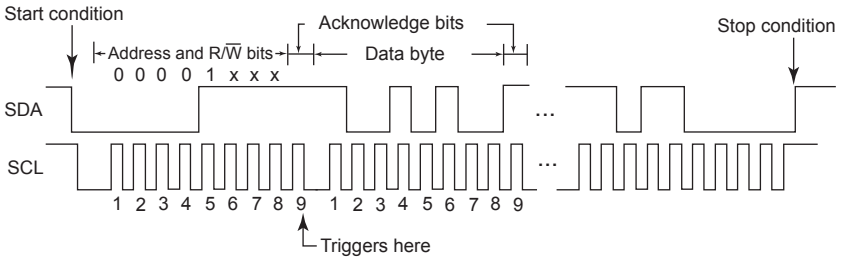
- **Start Byte**

  The DL6000/DLM6000 triggers on a start byte (pattern: 0000 0001).



- **HS Mode**

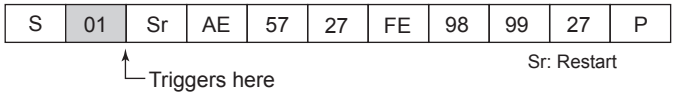
  The DL6000/DLM6000 triggers on the HS (high-speed) mode master code (pattern: 0000 1XXX).



- **Example**

  This example displays the data sequence at the byte level (hexadecimal notation) and indicates the trigger position. The following notations are used in the figure.
  S: Start condition, P: Stop condition, Shading: Compared pattern
  Trigger on a start byte

Mode	Start Byte/HS Mode
Type	Start byte



SDA, SCL, and Qualification

SDA and SCL Sources

You can select the serial data (SDA) and serial clock (SCL) sources.

- You can select the sources from CH1 to CH4.
- On the DLM6000, select the sources from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).

Trigger Level

You can set the I^2C bus signal trigger level for CH1 to CH4 separately.

- The selectable range is 8 divisions within the screen. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.
- You can reset the trigger level to the current offset voltage by pressing RESET.
- If the source is set to a signal from A0 to D7, the trigger level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IMDLM6054-01EN*.

Hysteresis

Hysteresis specifies a voltage range above and below the trigger level to prevent the DL6000/DLM6000 from triggering on minute voltage fluctuations.

	Specifies a hysteresis of approximately 0.3 divisions around the trigger level.*
	Specifies a hysteresis of approximately 1 division around the trigger level.*

* The values above are typical. They are not strictly warranted.

For the relationship between this hysteresis and the setup, analysis, and search hysteresis, see page 2-7.

Qualification and Logic

- **Qualification**

  Set the state of signals other than those selected for the SDA and SCL to H, L, or X. This trigger requirement is called qualification requirement. The qualification requirement is met when the selected state matches the input signal state.

H	When the input signal is high
L	When the input signal is low
X	Not used as a trigger condition (Don't care)

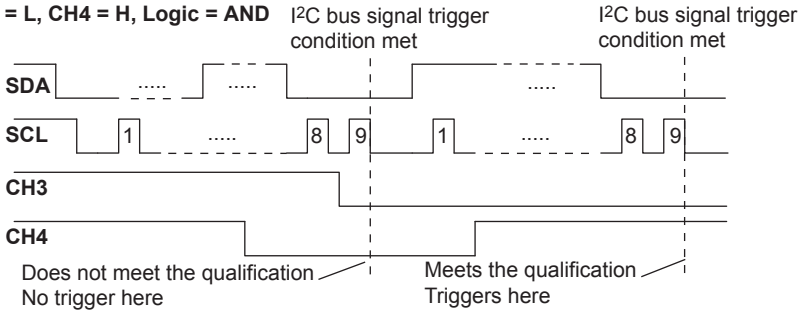
* The level for determining high or low is the trigger level that you set above when you set the signal to a channel from CH1 to CH4. If the source is set to a signal from A0 to D7, the trigger level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IMDLM6054-01EN*.

- **Logical Condition**

  You can select the logical condition for the qualification and the trigger condition for the I^2C bus signal that you set in each mode. When the logical condition is met, the DL6000/DLM6000 triggers.

AND	When the qualification and the I^2C bus signal trigger condition are both met
OR	When either the qualification or the I^2C bus signal trigger condition is met

CH3 = L, CH4 = H, Logic = AND



Note

To trigger only on the I^2C bus signal trigger condition (SDA and SCL signals), specify the settings as follows:

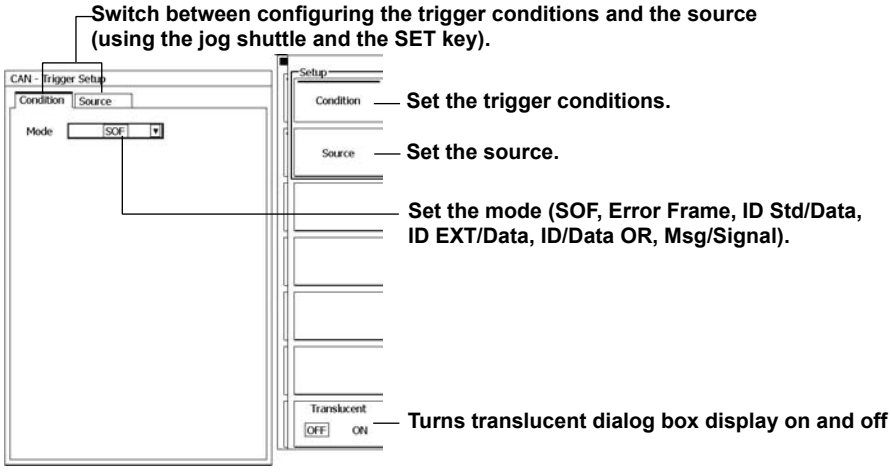
- The state of signals other than those selected for the SDA and SCL: X (don't care)
- Logic: AND

3.2 Triggering on a CAN Bus Signal

Procedure

Setup Menu

Press **ENHANCED**, the **Type** soft key, the **Serial Bus** soft key, the **CAN** soft key, and then the Setup soft key to display the following menu.

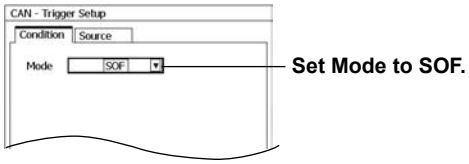


Setting Trigger Conditions (Condition)

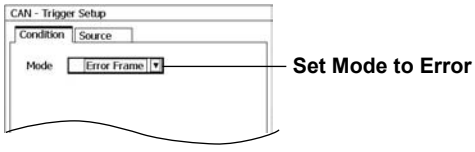
Depending on the address to compare, there are six modes:

- SOF
- Error Frame
- ID Std/Data
- ID EXT/Data
- ID/Data OR
- Msg/Signal

When Triggering in SOF Mode

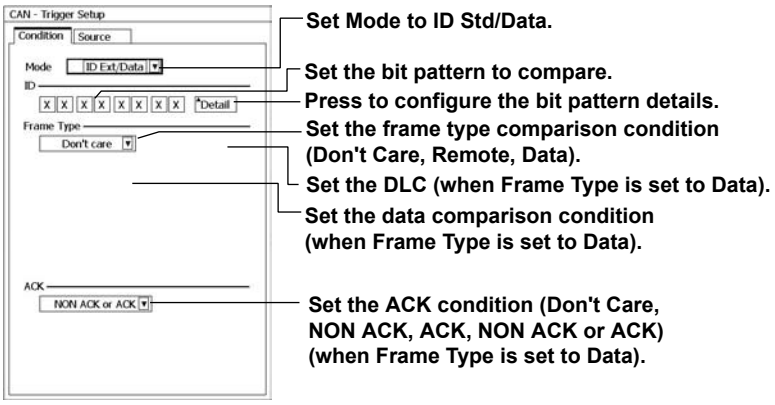


When Triggering in Error Frame Mode



When Triggering in ID Std/Data or ID Ext/Data Mode

ID Std/Data

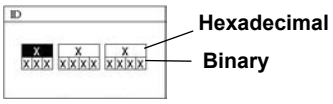


ID Ext/Data

Detailed Bit Pattern Configuration

Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the bit pattern.

An Example of an ID Std/Data ID Bit Pattern

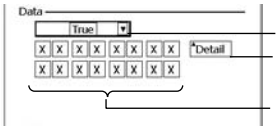


Making Frame Type's Data a Trigger Condition

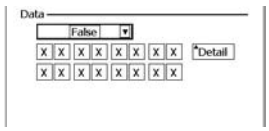
The settings vary depending on the Data comparison conditions.

If the Data Comparison Condition Is True or False

True

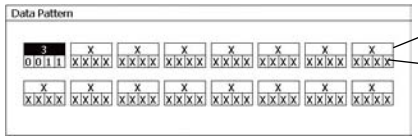


False



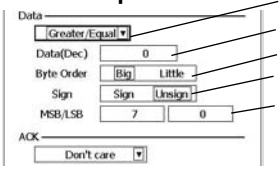
Detailed Data Pattern Configuration

Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the data pattern.

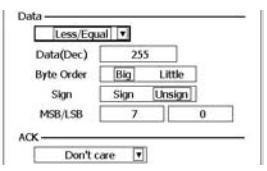


If the Data Comparison Condition is Greater/Equal or Less/Equal

Greater/Equal

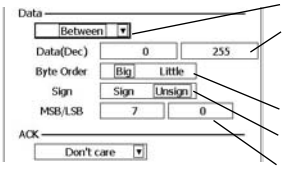


Less/Equal

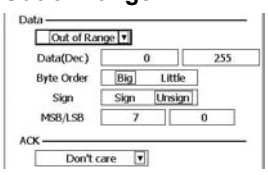


If the Data Comparison Condition is Between or Out of Range

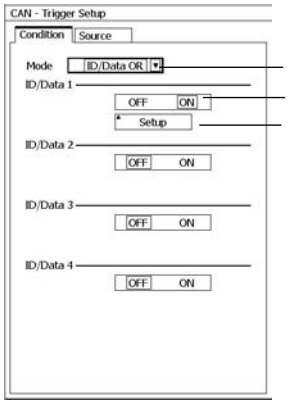
Between



Out of Range



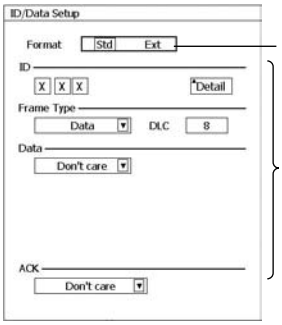
When Triggering in ID/Data OR Mode



Set Mode to ID/Data OR.

Select whether to make this a trigger condition (ON) or not (OFF).

Press to configure ID Std/Data or ID Ext/Data.

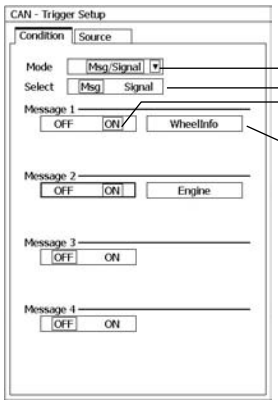


Select Std (ID Std/Data) or Ext (ID Ext/Data).

Set the frame type and data conditions. (See "When Triggering in ID Std/Data or ID Ext/Data Mode.")

When Triggering in Msg/Signal Mode

Making a Message a Trigger Condition

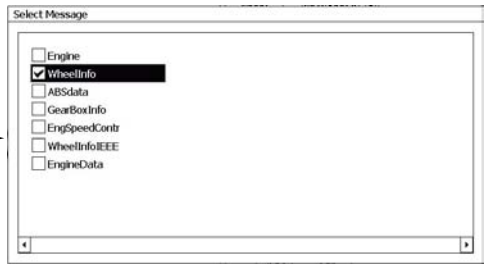


Set Mode to Msg/Signal.

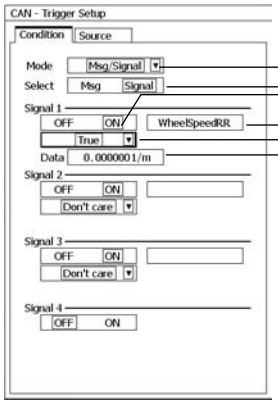
Set Select to Msg.

Turns the message on and off

Press to select a message (when the message is set to ON).



Making a Signal a Trigger Condition



Set Mode to Msg/Signal.

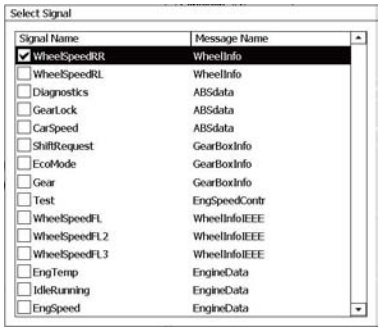
Set Select to Signal.

Turns the signal on and off

Press to select a signal (when the signal is set to ON).

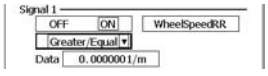
Set the data comparison condi-

Set the data (when the comparison condition is a value other than Don't care).

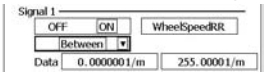


Set the data comparison condition.

When True, False, Greater/Equal, or Less/Equal Is Selected

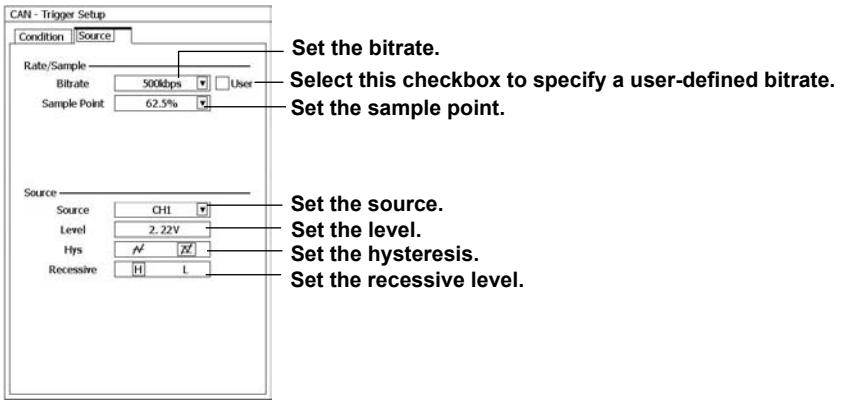


When Between or Out of Range Is Selected



Setting the Trigger Source (Source)

Select the **Source** tab, or press the **Source** soft key to display the following menu.



Explanation

This feature triggers on CAN bus signals. For details on the CAN bus signal frame format, see page 3-18.

Mode

Set the CAN trigger mode to SOF, Error Frame, ID Std/Data, ID Ext/Data, or ID/Data OR.

SOF Mode

Triggers on the start of a CAN bus signal frame.

SOF: Start of Frame

Error Frame Mode

The DL6000/DLM6000 triggers when the error frame's error flag is active.

ID Std/Data and ID Ext/Data Modes

ID Std/Data mode is used to trigger on the data frame or remote frame in standard format.
ID Ext/Data mode is used to trigger on the data frame or remote frame in extended format.
The DL6000/DLM6000 triggers on the AND logic of ID, Frame Type, Data, and ACK conditions.
The settings in ID Std/Data mode are shared with the settings in ID Ext/Data mode.

ID

Set the ID bit pattern in hexadecimal or binary notation. The ID bit pattern is 11 bits in standard format and 29 bits in extended format. The ID trigger condition is met when the specified bit pattern matches the input signal ID bit pattern.

- If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
- If a binary pattern contains any Xs, the corresponding hexadecimal display will be "$."

Frame Type

The DL6000/DLM6000 can be configured to trigger on the remote frame or data frame.

- Selecting the Frame

  A CAN bus signal frame contains a Remote Transmission Request (RTR) bit that indicates whether the frame is a remote frame or a data frame. Select the frame that the DL6000/DLM6000 will trigger on.

Don't care	The DL6000/DLM6000 will trigger on both remote frames and data frames.
Remote	The DL6000/DLM6000 will trigger on remote frames.
Data Frame	The DL6000/DLM6000 will trigger on data frames.

If you select Don't care or Remote, the DLC and Data trigger conditions in the next section will be ignored.

- DLC (Data Length Code)
  Set the data field length. The DLC trigger condition is met when the input signal DLC value matches the reference value. Set this value only when the frame type is set to Data Frame.
  Selectable range: 0 to 8 bytes
  If you set this value to zero, the data trigger conditions in the next section will be ignored.

Data

You can use the Data Field value as a trigger condition. Set this value only when the frame type is set to Data Frame.

- Comparison Condition
  The data trigger condition is met when the result of comparing the input signal Data Field value with the reference value meets the selected comparison condition.

Don't care	Not used as a trigger condition
True	When the input signal value meets the reference value
False	When the input signal value does not match the reference value
Greater/Equal	When the input signal value is greater than or equal to the reference value
Less/Equal	When the input signal value is less than or equal to the reference value
Between	When the input signal value is within the reference range that includes the boundary reference values
Out of Range	When the input signal value is outside the reference range that excludes the boundary reference values

- Data Pattern
  Set the data pattern for the length specified by DLC in hexadecimal or binary notation. The data pattern is valid only when the comparison condition is set to True or False.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be "$."

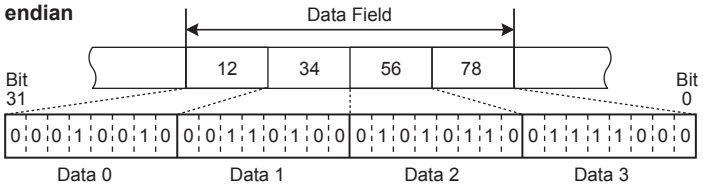
- Reference Value Data(Dec)
  - If you set the comparison condition to Greater/Equal or Less/Equal, set one reference value.
  - If you select Between or Out of Range, set two reference values to define a reference range. The values are automatically adjusted so that the lower limit is less than or equal to the upper limit.
  - If the comparison condition is True or False, the data pattern is used as the reference value.
  - Selectable range
    Set the selectable range in decimal notation.

Unsigned	0 to 9E+18 The selectable maximum value is limited by the data length and bit position that are determined by the DLC and MSB/LSB settings, respectively.
Signed	–9E+18 to 9E+18 The selectable minimum and maximum values are limited by the data length and bit position that are determined by the DLC and MSB/LSB settings, respectively.

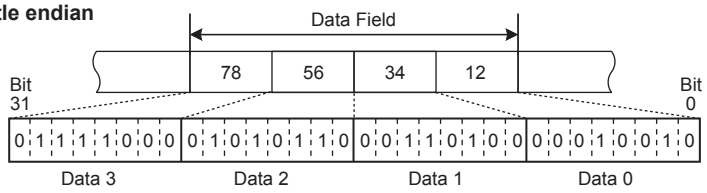
The value is displayed in exponential notation when it exceeds 7 digits (example: 1234567E+10).

- Byte Order
  Set the data byte order to big endian or little endian. For example, the following figure shows a 4-byte data stream on the bus (12345678 in hexadecimal notation).

Big endian



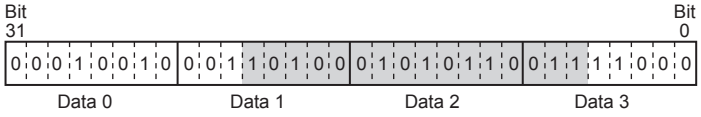
Little endian



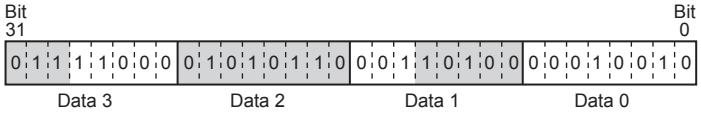
- Sign
  Select whether or not to add a sign to the data.
  The selectable range for the data reference value varies depending on this setting.

- MSB/LSB
  Set the MSB and LSB positions in the data to compare. For example, to compare bits 5 to 20 in a 4-byte data stream (12345678 in hexadecimal notation), set the MSB to 20 and the LSB to 5. The shaded sections in the following figure indicate the bits that will be compared depending on the byte order setting.
  Selectable range: 0 to the data size bytes × 8 – 1. The maximum value is 63.

Big endian



Little endian



ACK

You can use the ACK slot status as a trigger condition. The ACK trigger condition is met when the selected status matches the input signal ACK slot status.

Don't care	Not used as a trigger condition
NON ACK	When the status is recessive
ACK	When the status is dominant
NON ACK or ACK	When the status is recessive or dominant

ID/Data OR Mode

The DL6000/DLM6000 triggers on the OR logic of multiple ID Std/Data conditions or multiple ID Ext/Data conditions. You can set up to four ID Data conditions. The ID Std/Data settings are shared with the ID Ext/Data settings.

- You can select whether or not to use each ID/Data condition as a trigger condition.
- The trigger conditions and settings of each ID/Data condition are the same as those described on pages 3-14 to 3-16. See the respective page for details.

Note

When using the ID/Data OR mode, set conditions so that the trigger point will be the same. If you don't, the DL6000/DLM6000 may not trigger at the correct position.

Msg/Signal Mode

In this mode, the messages and symbols in the physical value/symbol definition file (.sbl)* that has been loaded into the DL6000/DLM6000 are used as trigger conditions.

* Physical value/symbol definition files (.sbl) are derived by converting CANdb files (.dbc). For instructions on how to load files, see section 13.9 in the User's Manual IM DLM6054-01EN.

The trigger condition settings in Msg/Signal mode are as follows:

Msg Mode		
ID		Loaded from the .sbl file
Other conditions		Not used as trigger conditions
Signal Mode		
ID		Loaded from the .sbl file
Frame type		Fixed to Data Frame
Data		
	Comparison condition	The selected condition
	Reference value	The specified value
	Byte order	Loaded from the .sbl file
	Sign	Loaded from the .sbl file
	MSB/LSB	Loaded from the .sbl file
	ACK	Not used as a trigger condition

Source Bit Rate, Sample Point, Trigger Level, Hysteresis, and Recessive Level

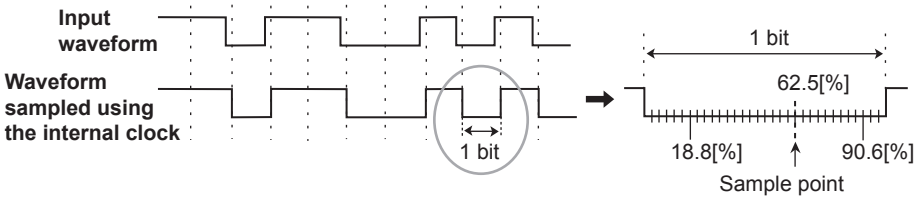
Bit Rate

You can select the CAN bus signal transfer rate from the following:
1 Mbps, 500 kbps, 250 kbps, 125 kbps, 83.3 kbps, and 33.3 kbps
If you select the User check box, you can set the transfer rate from 10.0 kbps to 1.000 Mbps in 0.1-kbps steps.

Sample Point

You can set the point for determining the bus level (recessive or dominant) from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 CAN bus signal trigger circuit samples the input CAN bus signal using the internal clock and detects the point of change from recessive to dominant. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate.

If the sample point is set to 62.5%



Trigger Level

You can set the CAN bus signal trigger level for CH1 to CH4 separately.

- The selectable range is 8 divisions within the screen. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.
- You can reset the trigger level to the current offset voltage by pressing RESET.

Hysteresis

Hysteresis specifies a voltage range above and below the trigger level to prevent the DL6000/DLM6000 from triggering on minute voltage fluctuations.

	Specifies a hysteresis of approximately 0.3 divisions around the trigger level.*
	Specifies a hysteresis of approximately 1 division around the trigger level.*

* The values above are typical. They are not strictly warranted.

For the relationship between this hysteresis and the setup, analysis, and search hysteresis, see page 2-7.

Recessive Level

Set the recessive level to high (H) or low (L). The logical value of the recessive level is 1 and that of the dominant level is 0 in either setting.

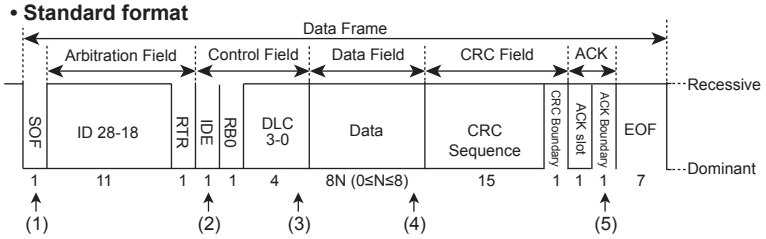
H	The recessive level is higher than the dominant level.
L	The recessive level is less than the dominant level.

Frame Format and Trigger Point

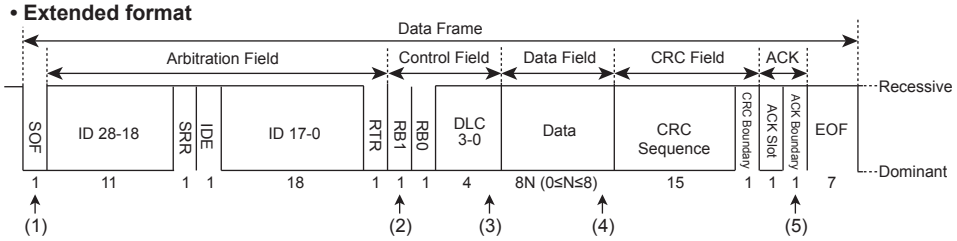
The following figure shows the frame and trigger point of each frame.

Data Frame

• Standard format



• Extended format

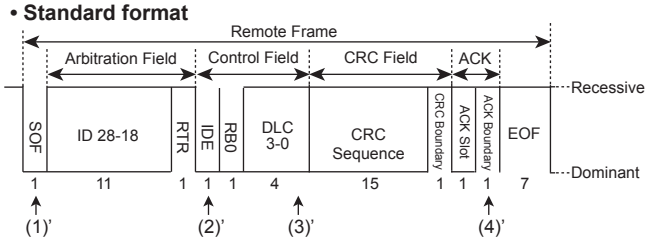


Positions (1) to (5) above are trigger points for the following conditions.
(1) Mode: SOF
(2) Mode: ID X*, Frame (RTR): Don't care, ACK: Don't care
(3) Mode: ID X*, Frame (RTR): Data, Data Field: Don't care, ACK: Don't care
(4) Mode: ID X*, Frame (RTR): Data, Data Field: Not Don't care, ACK: Don't care
(5) ACK: Not Don't care

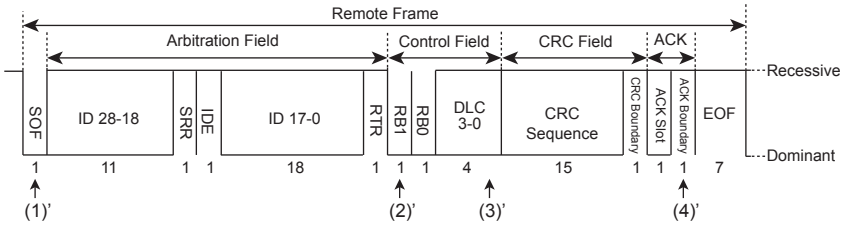
* ID X: ID Std/Data, ID Ext/Data, or ID/Data OR

Remote Frame

• Standard format



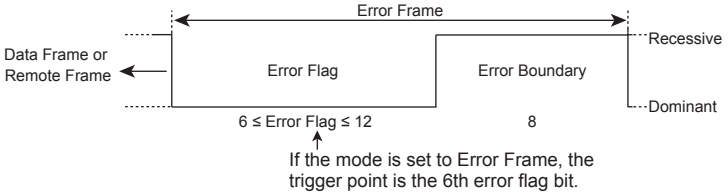
• Extended format



Positions (1)' to (5)' above are trigger points for the following conditions.
(1)' Mode: SOF
(2)' Mode: ID X*, Frame(RTR): Don't care, ACK: Don't care
(3)' Mode: ID X*, Frame(RTR): Remote, ACK: Don't care
(4)' ACK: Not Don't care

* ID X: ID Std/Data, ID Ext/Data, or ID/Data OR

Error Frame



If the mode is set to Error Frame, the trigger point is the 6th error flag bit.

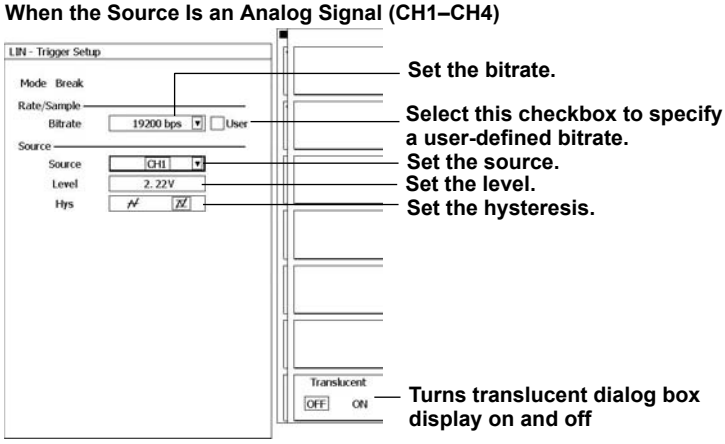
3.3 Triggering on a LIN Bus Signal

Procedure

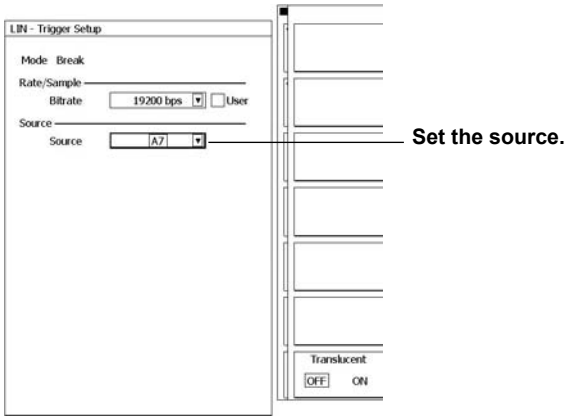
Setup Menu

Press **ENHANCED,** the **Type** soft key, the **Serial Bus** soft key, the **LIN** soft key, and then the Setup soft key to display the following menu.

When the Source Is an Analog Signal (CH1–CH4)



When the Source Is a Logic Signal (A0–D7)



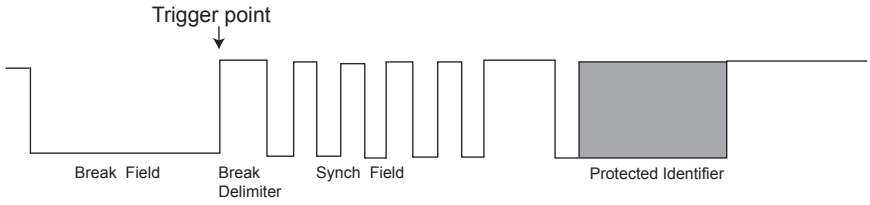
Explanation

This feature triggers the rising edge of the Break delimiter of the LIN bus signal.

Mode

The mode is fixed to Break.

The trigger activates on the rising edge of the Break delimiter of the LIN bus signal.



Source

You can select the source.

- On the DL6000, select the source from CH1 to CH4.
- On the DLM6000, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).

Bit Rate

You can select the LIN bus signal transfer rate from the following:

19200 bps, 9600 bps, 4800 bps, 2400 bps, 1200 bps

If you select the User check box, you can set the transfer rate from 1000 bps to 20000 bps in 10-bps steps.

Trigger Level

You can set the LIN bus signal trigger level for CH1 to CH4 separately.

- The selectable range is 8 divisions within the screen. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.
- You can reset the trigger level to the current offset voltage by pressing RESET.
- If the source is set to a signal from A0 to D7, the trigger level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-01EN*.

Hysteresis

Hysteresis specifies a voltage range above and below the trigger level to prevent the DL6000/DLM6000 from triggering on minute voltage fluctuations.

	Specifies a hysteresis of approximately 0.3 divisions around the trigger level.*
	Specifies a hysteresis of approximately 1 division around the trigger level.*

* The values above are typical. They are not strictly warranted.

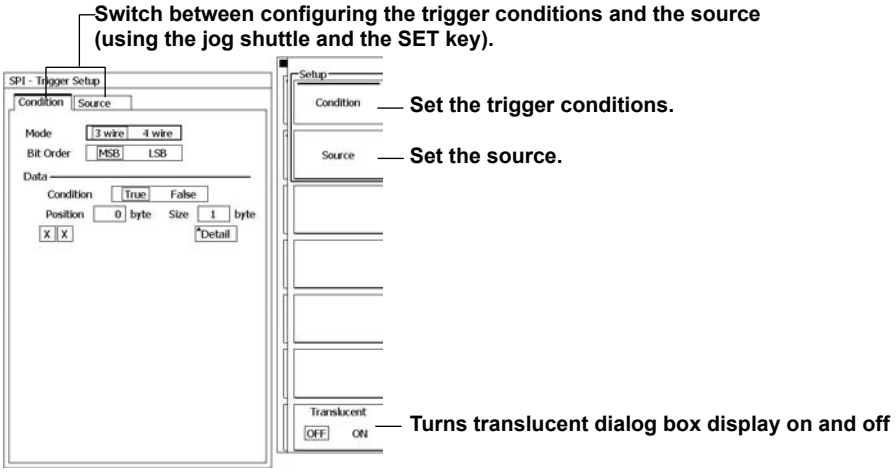
For the relationship between this hysteresis and the setup, analysis, and search hysteresis, see page 2-7.

3.4 Triggering on a SPI Bus Signal

Procedure

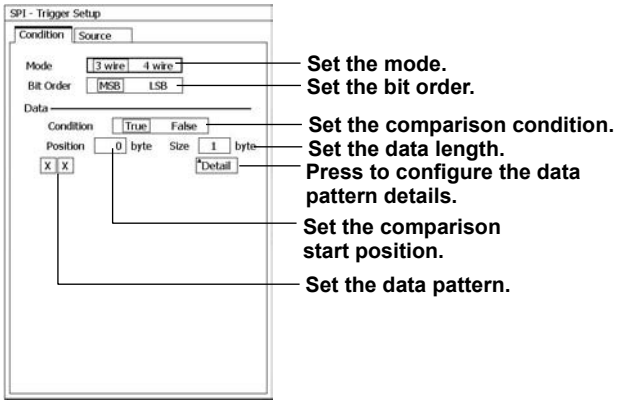
Setup Menu

Press **ENHANCED,** the **Type** soft key, the **Serial Bus** soft key, the **SPI** soft key, and then the Setup soft key to display the following menu.

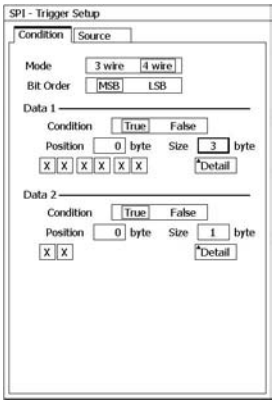


Setting Trigger Conditions (Condition)

When Mode Is Set to 3Wire

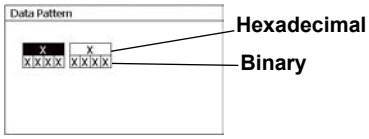


When Mode Is Set to 4Wire



Detailed Data Pattern Configuration

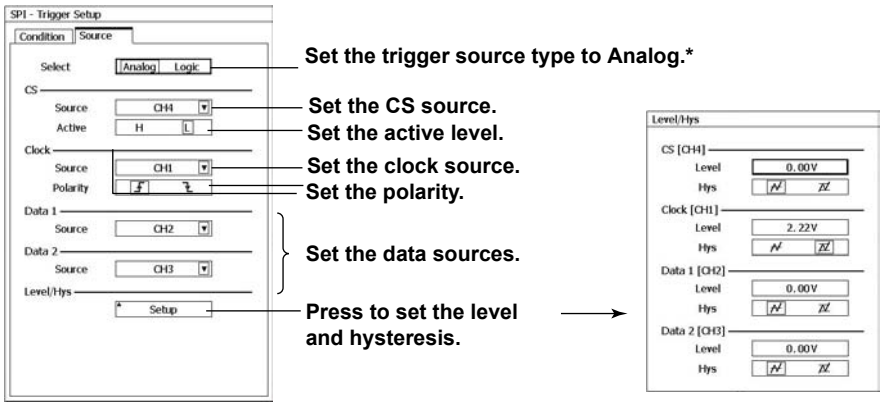
Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the data pattern.



Setting the Trigger Source (Source)

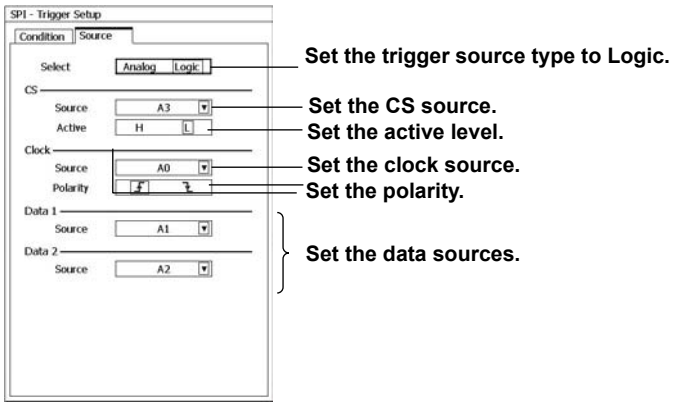
Select the **Source** tab, or press the **Source** soft key to display the following menu.

For Analog Sources



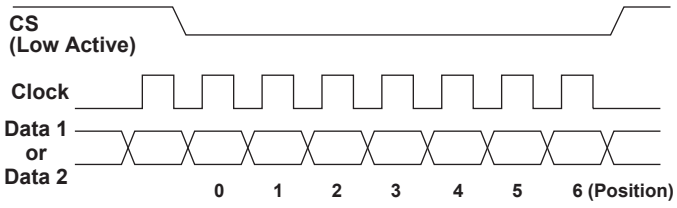
*** This is fixed to Analog on the DL6000 series, and the setting is not displayed.**

For Logic Sources (DLM6000)



Explanation

This feature triggers on SPI bus signals. The following figure shows the SPI bus signal timing chart.



Wiring System

Select the wiring system from the following:

Three-wire	The DL6000/DLM6000 triggers on the data pattern condition of one data line.
Four-wire	The DL6000/DLM6000 triggers on the data pattern conditions of Data 1 and Data 2 lines. You can also use one of the two data lines as a trigger condition.

Bit Order

You can select the bit order based on the data stream.

- If you are setting the data in binary notation, set the pattern in the order of the data stream, regardless of the bit order setting.
- If you are setting the data in hexadecimal notation, set the pattern in 4-bit segments according to the bit order setting.

MSB	When the data stream is MSB first
LSB	When the data stream is LSB first

Data

You can use a data pattern as a trigger condition.

- Comparison Condition
  The data trigger condition is met when the result of comparing the input signal pattern with the specified pattern meets the selected comparison condition.

True	When the patterns match
False	When the patterns don’t match

- Comparison Start Position
  Set the comparison start position. For example, to start comparing from the first data byte after the CS signal is activated, specify zero.
  Selectable range: 0 to 9999 bytes
- Data Size
  Set how many consecutive data bytes you want to compare.
  Selectable range: 1 to 4 bytes
- Data Pattern
  Set the data pattern for the specified size in hexadecimal or binary notation.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

CS, Clock, and Data

You can select the chip select (CS), clock, and data sources.

- On the DL6000, select the sources from CH1 to CH4.
- On the DLM6000, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).

CS

You can select the CS level for activating the data.

H	When the signal is high
L	When the signal is low

Clock

You can select the clock edge that specifies when the data patterns are compared.

↑	On the rising edge
↓	On the falling edge

Trigger Level

When the CS, clock, or data* is set to analog (CH1 to CH4), you can set the trigger level for each source.

- The selectable range is 8 divisions within the screen. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.
- You can reset the trigger level to the current offset voltage by pressing RESET.

* If the source is set to a logic signal from A0 to D7, the trigger level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-01EN*.

Hysteresis

Hysteresis specifies a voltage range above and below the trigger level to prevent the DL6000/DLM6000 from triggering on minute voltage fluctuations.

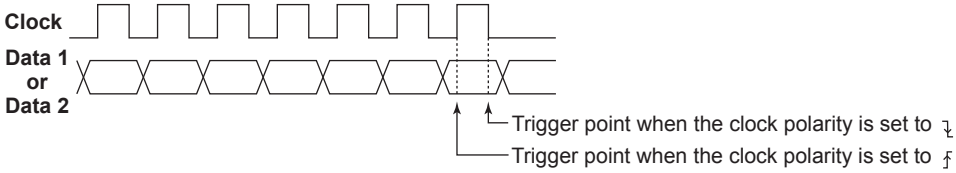
	Specifies a hysteresis of approximately 0.3 divisions around the trigger level.*
	Specifies a hysteresis of approximately 1 division around the trigger level.*

* The values above are typical. They are not strictly warranted.

For the relationship between this hysteresis and the setup, analysis, and search hysteresis, see page 2-7.

Trigger Point

The trigger point is determined by the clock polarity setting as follows:



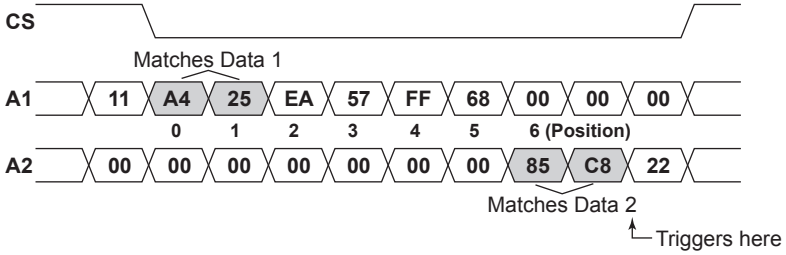
Example

This example displays the data sequence at the byte level (hexadecimal notation) and indicates the trigger position.

The Data 1 and Data 2 pattern references are set to A1 and A2, respectively.

Shading: Pattern to compare

CS	Active: L
Data 1	Condition: True, Position: 0, Size: 2 bytes, data pattern: A4 and 25
Data 2	Condition: True, Position: 6, Size: 2 bytes, data pattern: 85 and C8



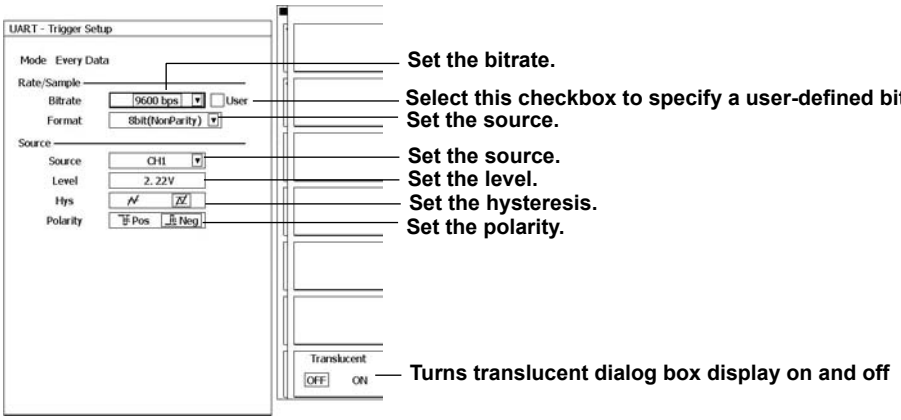
3.5 Triggering on a UART Signal

Procedure

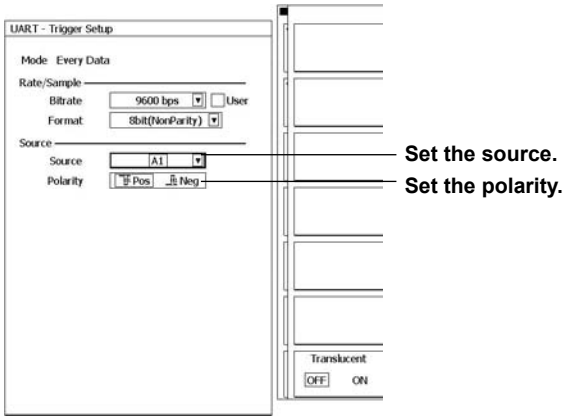
Setup Menu

Press **ENHANCED,** the **Type** soft key, the **Serial Bus** soft key, the **UART** soft key, and then the Setup soft key to display the following menu.

When the Source Is an Analog Signal (CH1–CH4)



When the Source Is a Logic Signal (A0–D7)

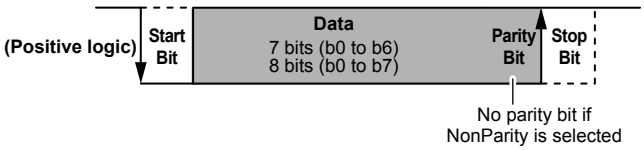


Explanation

This feature triggers on UART bus signals.

Mode

The mode is fixed to Every Data. This feature triggers on the stop bit of all data frames.



Source

You can select the source.

- On the DL6000, select the source from CH1 to CH4.
- On the DLM6000, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).

Bit Rate

You can select the UART bus signal transfer rate from the following:
115200 bps, 57600 bps, 38400 bps, 19200 bps, 9600 bps, 4800 bps, 2400 bps, and 1200 bps

If you select the User check box, you can set the transfer rate from 1000 bps to 200000 bps in 100-bps steps.

Format

You can select the format from the following:

8bit + Parity	8-bit data + parity bit
7bit + Parity	7-bit data + parity bit
8bit(NonParity)	8-bit data with no parity bit

Trigger Level

You can set the UART signal trigger level for CH1 to CH4 separately.

- The selectable range is 8 divisions within the screen. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.
- You can reset the trigger level to the current offset voltage by pressing RESET.
- If the source is set to a signal from A0 to D7, the trigger level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-01EN*.

Hysteresis

Hysteresis specifies a voltage range above and below the trigger level to prevent the DL6000/DLM6000 from triggering on minute voltage fluctuations.

	Specifies a hysteresis of approximately 0.3 divisions around the trigger level.*
	Specifies a hysteresis of approximately 1 division around the trigger level.*

* The values above are typical. They are not strictly warranted.

For the relationship between this hysteresis and the setup, analysis, and search hysteresis, see page 2-7.

Polarity

You can select the bit state that will be considered logical 1.

Pos	Positive logic
Neg	Negative logic

Note

You cannot set the hold-off time when the UART signal trigger feature is enabled. For details on the hold-off time, see section 6.1 in the *User's Manual IM DLM6054-01EN*.

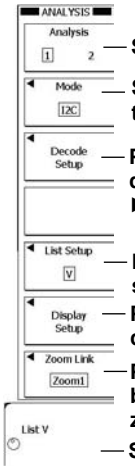
4.1 Selecting the Serial Bus Signal and Displaying and Saving Analysis Results

Procedure

ANALYSIS Menu

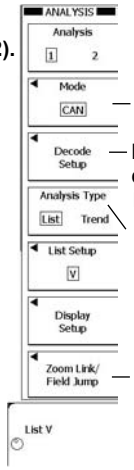
Press **ANALYSIS,** the **Mode** soft key, the **Serial Bus** soft key, and then the soft key for a serial bus to display one of the following menus.

When I2C Is Selected



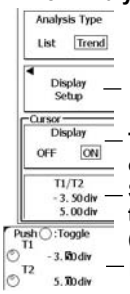
Set the analysis number (1, 2).
Set the analysis mode to I2C.
Press to configure the decode settings. ► section 4.2
Press to configure the list settings.
Press to configure the display settings.
Press to set the link between the list and a zoom window.
Scrolls the list

When CAN Is Selected



Set the analysis mode to CAN.
Press to configure the decode settings. ► section 4.3
Set the analysis result display method (List, Trend).
Press to set the link between the list and a zoom window, and to set which field to jump to.

When Analysis Type Is Set to Trend



Press to configure the display settings.
Turns the cursor display on and off
Switches the value that the jog shuttle adjusts (T1, T2, T1&T2)
Moves the cursors

When LIN Is Selected

Set the analysis mode to LIN.
Press to configure the decode settings. ► section 4.4

When SPI Is Selected

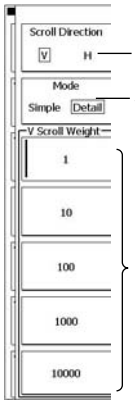
Set the analysis mode to SPI.
Press to configure the decode settings. ► section 4.5

When UART Is Selected

Set the analysis mode to UART.
Press to configure the decode settings. ► section 4.6

Configuring the List Settings (List Setup)

Press the List Setup soft key to display the following menu.



Set the scroll direction (V, H).

Set the display mode (Simple, Detail).

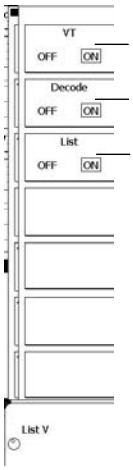
Set the number of entries to scroll.

Note

With CAN bus signals, this feature does not work if AnalysisType is set to Trend.

Setting the Display (Display Setup)

Press the Display Setup soft key to display the following menu.

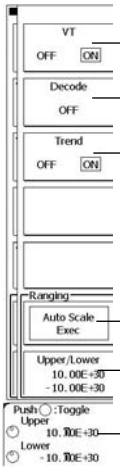


Turns the VT waveform display on and off

Turns the decode display on and off

Turns the list display on and off

CAN Bus Signal Analysis When Analysis Type Is Set to Trend



Turns the VT waveform display on and off

Turns the decode display on and off

Turns the trend display on and off

Executes auto scaling of the trend display

Switches the value that the jog shuttle adjusts (Upper, Lower)

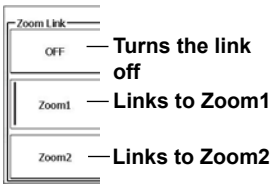
Sets the upper and lower limits of the trend display.

Setting the Zoom Link and Field Jump (CAN/LIN)

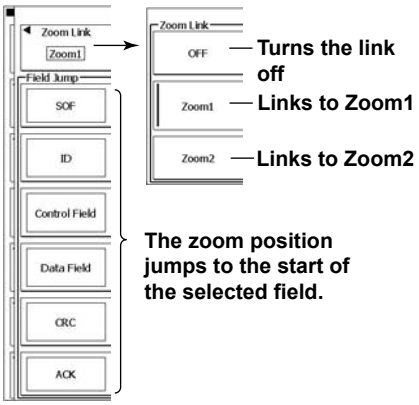
If the serial bus signal is I2C, SPI, or UART, press the Zoom Link soft key.

If the serial bus signal is CAN or LIN, press the Zoom Link/Field Jump soft key. One of the following menus appears.

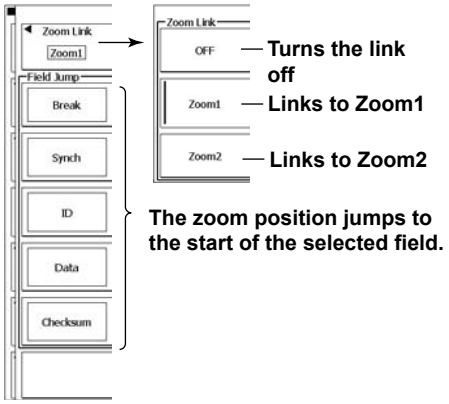
When I2C, SPI, or UART Is Selected



When CAN Is Selected



When LIN Is Selected



Note

With CAN bus signals, this feature does not work if AnalysisType is set to Trend.

Saving the Analysis Results

To save the serial bus list for a signal such as I2C, CAN, LIN, SPI, or UART, set Data Type to Serial Bus as explained in section 13.6 in the User's Manual IM DLM6054-01EN.

Explanation

Analysis Type

This manual describes I^2C, CAN, LIN, SPI, and UART serial bus signal analysis features. For information about other analysis features, see the *User's Manual IM DLM6054-01EN.*

Data Analyzed

The DL6000/DLM6000 can analyze the following data displayed on the screen.

- Waveform data
  The DL6000/DLM6000 can analyze data any time regardless of whether or not it is acquiring data. If acquiring signals, the DL6000/DLM6000 updates the analysis results in sync with the displayed waveform.
  The DL6000/DLM6000 can also analyze waveform data saved to the history memory (the waveform data at the record number selected using HISTORY menu > Select).
- Loaded acquisition data (ACQ data)

Displaying Analysis Results

The following pages describe the contents of the I^2C, CAN, LIN, SPI, and UART analysis results.

I^2C

- **Number of Analyzable Data Values**

Up to 40000 bytes (20000 bytes before and after the analysis reference point)

- **Simple Display**

No.	Analysis Number.
S/P	Display the data condition. S: Start condition, P: Stop condition
Hex	Data in hexadecimal notation
Form	Address or data
R/W	Signal type
ACK	Acknowledge bit state

For a description of these items, see "Detail Display."

- **Detail Display**

No.	Analysis Number. Negative numbers are assigned to frames before the analysis reference point, and positive numbers are assigned to frames after the reference point. The DL6000/DLM6000 can display the analysis result numbers from –19999 to 20000 (up to 40000 data bytes). Pressing the RESET key highlights data number zero.
S/P	Display the data condition. S: Start condition, P: Stop condition
Time(ms)	Displays the time from the trigger position to the head of the byte in milliseconds.
Binary	Displays data in binary notation.
Hex	Displays data in hexadecimal notation.
Form	Indicates "A" for address and "D" for data.
R/W	Displays "R" for a read signal and "W" for a write signal.
ACK	Displays "1" when an acknowledge bit is detected and "0" when it is not.
Info	Display the data type.

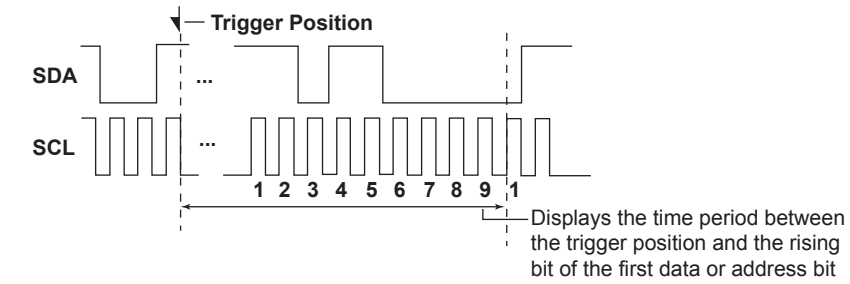
Simple display

No.	S/P	Hex	Form	R/W	ACK
9	P	4B	D		1
10	S	F8	A	W	0
11		EB	D		0
12	P	21	D		0
13	S	9D	A	R	0
14		66	D		0
15	P	E2	D		1
16	S	38	A	W	0
17		9B	D		0
18	P	2C	D		0
19	S	6D	A	R	0
20		D7	D		0
21	P	4E	D		1

Detail display

No.	S/P	Time(ms)	Binary	Hex	Form	R/W	ACK	Info
1	S	-2.46700	01110101	75	A	R	0	7-bit
2		-2.43100	10111100	BC	D		0	
3	P	-2.39500	11101111	EF	D		1	
4	S	-1.32304	00111000	38	A	W	0	7-bit
5		-1.28704	01010011	53	D		0	
6	P	-1.25104	10101001	A9	D		0	
7	S	-1.13904	10011101	9D	A	R	0	7-bit
8		-1.10304	00010000	10	D		0	
9	P	-1.06704	01001011	4B	D		1	
10	S	0.00496	11111000	F8	A	W	0	7-bit
11		0.04096	11101011	EB	D		0	
12	P	0.07696	00100001	21	D		0	
13	S	0.18896	10011101	9D	A	R	0	7-bit

Time (ms) Display



- **Decoded Field Display**

Decodes each data value and displays the value in color. This feature can be used when the source signal is set to a channel from CH1 to CH4 or from M1 to M4.

Adr	Hexadecimal value in light green
Data	Hexadecimal value in cyan
R/W	Pink
Ack	Yellow
General Call	Green
Start Byte	Orange
HS Mode	Orange

CAN

- **Number of Analyzable Frames**

Up to 3000 frames (1500 frames before and after the analysis reference point)

- **Frames Analyzed**

Remote frame	Detects whether or not the ID, data, CRC, and ACK values are present.
Data frame	Detects whether or not the ID, CRC, and ACK values are present.
Error frame	Detects error flags.
Overload frame	Detects an overload flag.

- **Simple Display**

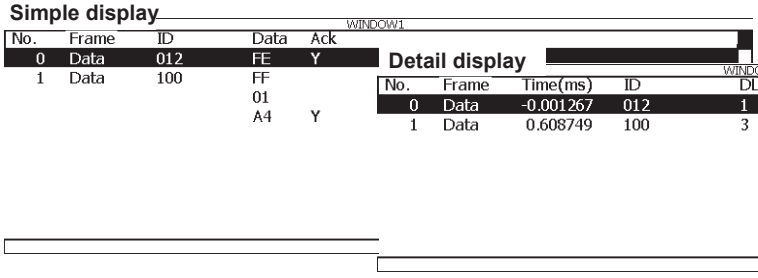
No.	Analysis Number
Frame	Frame type
ID	Hexadecimal ID display
Data	Data in hexadecimal notation display
Ack	ACK slot state

For a description of these items, see “Detail Display.”

- **Detail Display**

No.	Analysis Number. Negative numbers are assigned to frames before the analysis reference point, and positive numbers are assigned to frames after the reference point. The DL6000/DLM6000 can display the analysis result numbers from –1499 to 1500 (up to 3000 frames). Pressing the RESET key highlights frame number zero.
Frame	Displays the frame type. The DL6000/DLM6000 can analyze the following four frame types: data frame, remote frame, error frame, and overload frame.
Time(ms)	Displays the time from the trigger position to the head of the frame in milliseconds.
ID	Displays the 11-bit standard format ID value or the 29-bit extended format ID value in hexadecimal notation.
DLC	Displays the effective number of bytes in hexadecimal notation.
Data(Bin)	Displays data in binary notation when the frame type is data. Each byte is displayed in a separate line.
Data	Displays data in hexadecimal notation when the frame type is data. Each byte is displayed in a separate line.
CRC	Displays the CRC sequence in hexadecimal notation when the frame type is data or remote.
Ack	Displays “Y” when an ACK bit is detected and “N” when it is not.

Simple display



Detail display

- **Decoded Field Display**

Decodes each field value and displays the value in color.

ID	Light green
DLC	Pink
Data	Cyan
CRC sequence	Light blue
Alarm frame	Red
Overload frame	Green
Frame background	Gray
Stuff bit	Gray fill

LIN

- **Number of Analyzable Frames**

Up to 3000 frames (1500 frames before and after the analysis reference point)

- **Fields Analyzed**

Break, Synch, ID, Data, Checksum

- **Simple Display**

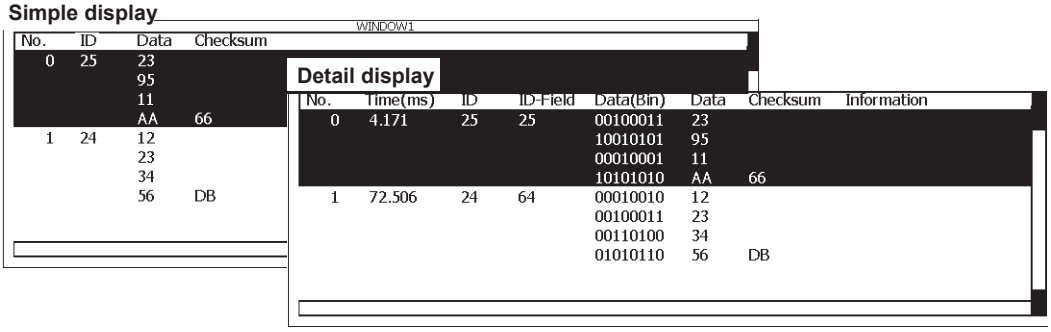
No.	Analysis Number
ID	ID in hexadecimal notation
Data	Data in hexadecimal notation
Checksum	Checksum in hexadecimal notation

For a description of these items, see “Detail Display.”

- **Detail Display**

No.	Analysis Number. Negative numbers are assigned to frames before the analysis reference point, and positive numbers are assigned to frames after the reference point. The DL6000/DLM6000 can display the analysis result numbers from –1499 to 1500 (up to 3000 frames). Pressing the RESET key highlights frame number zero.
Time(ms)	Displays the time from the trigger position to the head of the frame in milliseconds.
ID	Displays the ID value in hexadecimal notation.
ID-Field	Displays the ID value including the two parity bits in hexadecimal notation.
Data(Bin)	Displays data in binary notation. Each byte is displayed in a separate line.
Data	Displays data in hexadecimal notation. Each byte is displayed in a separate line.
Checksum	Displays the checksum value in hexadecimal notation.
Information	Detects and displays the following words. If a WakeUp signal is detected, WakeUp appears. If multiple errors are detected in one frame, the error with the highest precedence in the list below appears. Timeout Error, Framing Error, Checksum Error, Synch Error, Parity Error

Simple display



- **Decoded Field Display**

Decodes each field value and displays the value in color.

Break	Orange	
Synch	Pink	
ID	Light green	
Data	Cyan	
Checksum	Light blue	
WakeUp	Green	
Start Bit	Gray fill	
Stop Bit	Gray fill	
Error	Red	
	• Timeout Error	Displays a thick red link line in the area that errors occurred.
	• Framing Error	Displays “Framing Error” using black characters on a red background in the field in which an error occurs. It is displayed with a higher precedence than Checksum Error, Synch Error, or Parity Error.
	• Checksum Error, Synch Error, Parity Error	Displays the characters of the synch, ID, or checksum field in which an error occurs using black characters on red background.

SPI

- **Number of Analyzable Data Values**

Up to 40000 bytes (20000 bytes before and after the analysis reference point)

- **Simple Display**

No.	Analysis Number.
Data 1(H)	Data 1 in hexadecimal notation
Data 2(H)	Data 2 in hexadecimal notation
CS	Displays the CS status.

For a description of these items, see “Detail Display.”

- **Detail Display**

No.	Analysis Number. Negative numbers are assigned to frames before the analysis reference point, and positive numbers are assigned to frames after the reference point. The DL6000/DLM6000 can display the analysis result numbers from –19999 to 20000 (up to 40000 data bytes). Pressing the RESET key highlights data number zero.
Time(ms)	Displays the time from the trigger position to the head of the byte in milliseconds.
Data 1(B)	Displays data 1 in binary notation.
Data 1(H)	Displays data 1 in hexadecimal notation.
Data 2(B)	Displays data 2 in binary notation.
Data 2(H)	Displays data 2 in hexadecimal notation.
CS	Displays the CS status.
S/P	Indicates the active period by displaying “S” for the start position and “P” for the stop position.

Simple display

No.	Data1(H)	Data2(H)	CS
80	00	--	H
81	00	--	H
82	00	--	H
83	00	--	H
84	00	--	H
85	00	--	H
86	00	--	H
87	C4	--	H
88	A9	--	H
89	3B	--	H
90	00	--	H
91	00	--	H
92	00	--	H

Detail display

No.	Time(ms)	Data1(B)	Data1(H)	Data2(B)	Data2(H)	CS	S/P
80	0.336344	00000000	00	--------	--	H	
81	0.344344	00000000	00	--------	--	H	
82	0.352344	00000000	00	--------	--	H	
83	0.360344	00000000	00	--------	--	H	
84	0.368344	00000000	00	--------	--	H	
85	0.376344	00000000	00	--------	--	H	
86	0.384344	00000000	00	--------	--	H	P
87	0.400344	11000100	C4	--------	--	H	S P
88	0.424344	10101001	A9	--------	--	H	S
89	0.432344	00111011	3B	--------	--	H	P
90	0.472344	00000000	00	--------	--	H	S
91	0.480344	00000000	00	--------	--	H	
92	0.488344	00000000	00	--------	--	H	

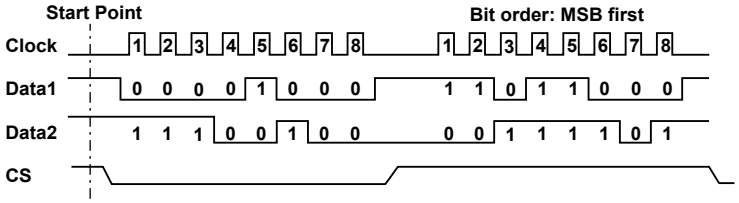
- **Decoded Field Display**

Decodes each data value and displays the value in color. This feature can be used when the source signal is set to a channel from CH1 to CH4 or from M1 to M4.

Data	Hexadecimal value in cyan
Group background	Gray

- **Display Example**

Examples for two analysis conditions are given below.



Analysis Conditions when clock (CH1) = ↓ and CS (CH4) = L	
Item	Display
Analysis number (No.)	0
Data 1 in hexadecimal notation	08
Data 2 in hexadecimal notation	E4
CS signal status	
Analysis Conditions when clock (CH1) = ↓ and CS (CH4) = H	
Item	Display
Analysis number (No.)	0
Data 1 in hexadecimal notation	D8
Data 2 in hexadecimal notation	3D
CS signal status	H

Note

If there is no change in the CS signal from high to low or low to high, the DL6000/DLM6000 will not analyze the I/O data.

UART

- **Number of Analyzable Data Values**

Up to 3000 bytes (1500 bytes before and after the analysis reference point)

- **Fields Analyzed**

  Data

- **Simple Display**

No.	Analysis number
Data/ASCII	Hexadecimal data or ASCII display
Information	Displays errors

For a description of these items, see “Detail Display.”

- **Detail Display**

No.	Analysis Number. Negative numbers are assigned to frames before the analysis reference point, and positive numbers are assigned to frames after the reference point. The DL6000/DLM6000 can display the analysis result numbers from –1499 to 1500 (up to 3000 data bytes). Pressing the RESET key highlights data number zero.
Time(ms)	Displays the time from the trigger position to the head of the byte in milliseconds.
Size	Displays the number of data bytes only when display mode’s Grouping feature is set to ON.
Data(Bin)	Displays data in binary notation.
Data/ASCII	Displays data in hexadecimal notation. If you set the display mode to ASCII, data is displayed using ASCII codes.
Information	Displays the following errors. If multiple errors are detected in one data byte, the error with the highest precedence in the list below appears. Framing Error, Parity Error

Simple display

No.	Data	Information
0	30	
1	53	
2	74	
3	61	
4	72	

Simple display - Grouping ON

WINDOW1

No.	Data(Hex)
0	47 50 53 32 30 30
1	53 74 61 72 74 5F 41
2	33 36 3A 31 36 35
3	31 34 32 3A 35 33 31
4	47 50 53 32
5	53 74 61 72

Simple display - ASCII

WINDOW1

No.	ASCII	Information
0	0	
1	S	
2	t	
3	a	
4	r	
5	t	
6	_	
7	A	
8	3	
9	6	

Detail display

No.	Time(ms)	Data(Bin)	Data
0	-0.4944	00110000	30
1	1.0168	01010011	53
2	1.5872	01110100	74
3	2.1616	01100001	61
4	2.7348	01110010	72
5	3.3072	00110100	74

Detail display - Grouping ON

WINDOW1

No.	Time(ms)	Size	Data(Hex)	Information
0	-80.912	6	47 50 53 32 30 30	
1	-80.339	7	53 74 61 72 74 5F 41	
2	-79.765	6	33 36 3A 31 36 35	
3	-79.192	7	31 34 32 3A 35 33 31	
4	-78.621	6	47 50 53 32 30 30	
5	-77.110	7	53 74 61 72 74 5F 41	

Detail display - ASCII

WINDOW1

No.	Time(ms)	ASCII	Information
0	-0.4944	0	
1	1.0168	S	
2	1.5872	t	
3	2.1616	a	
4	2.7348	r	
5	3.3072	t	
6	3.8792	_	
7	4.4520	A	
8	5.9620	3	
9	6.5376	6	

- **Decoded Field Display**

Decodes each field value and displays the value in color.

Data	Cyan
Parity	Yellow
Start Bit	Gray fill
Stop Bit	Gray fill
Error	Red • Framing Error: Displays "Framing Error" using black characters on a red background in the field in which an error occurs. It is displayed with higher precedence than Parity Error. • Parity Error: Displays the characters of the field in which an error occurs using black characters on red background.

Zoom Link

Select the zoom link from the following:

OFF	Disables the zoom link feature.
Zoom1	Links to Zoom1.
Zoom2	Links to Zoom2.

The default setting for WINDOW 1 is Zoom1 and WINDOW 2 is Zoom2. If Zoom1 or Zoom2 is selected and you select (highlight) any byte in the analysis result list, the zoom position (the center of the zoom box) will move to the head of the byte. On the contrary, if you move the zoom position, the corresponding byte in the analysis result list in the Zoom1 or Zoom2 box will be highlighted.

Field Jump

If the analysis type is FlexRay, CAN, or LIN and the zoom link feature is enabled, you can move the zoom position to the head of the specified field in the highlighted frame in the analysis result list.

Saving Analysis Results

You can save analysis results (simple and detail) in CSV format to an external storage medium. The extension is .csv.
The list will be saved in the appropriate format.

Data Saved

In the menu that appears when you set the data type to Serial Bus in the procedure described in section 13.8 in the *User's Manual IM DLM6054-01EN*, select whether to save Ana1 or Ana2.

Ana1	Saves the analysis results that are determined under the conditions specified using the menu that appears when the WINDOW 1 key is pressed.
Ana2	Saves the analysis results that are determined under the conditions specified using the menu that appears when the WINDOW 2 key is pressed.

Data Size

I^2C	(The number of analyzed bytes + 4) × 65 [bytes]
CAN	(The number of analyzed frames + 4) × 155 [bytes]
LIN	(The number of analyzed frames + 4) × 170 [bytes]
SPI	(The number of analyzed bytes + 4) × 79 [bytes]
UART	(The number of analyzed frames + 4) × 40 [bytes]

* The data sizes are reference values. They are not strictly warranted. Use them as a guideline when you save data.

4.2 Analyzing an I²C Signal

Procedure

ANALYSIS Menu

Press **ANALYSIS**, the **Mode** soft key, the **Serial Bus** soft key, and then the **I2C** soft key to display the following menu.

Set the analysis number (1, 2).

Set the analysis mode to I2C.

Press to configure the decode settings.

Press to configure the list settings. ► section 4.1

Press to configure the display settings. ► section 4.1

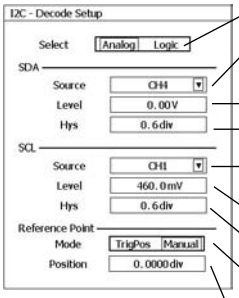
Press to set the link between the list and a zoom window. ► section 4.1

Scrolls the list. ► section 4.1

Configuring Decode Settings (Decode Setup)

Press the **Decode Setup** soft key to display one of the following menus.

Analog Signal



Set Select to Analog.*

Set the SDA source (CH1–CH4, M1–M4).

Set the level.

Set the hysteresis.

Set the SCL source (CH1–CH4, M1–M4).

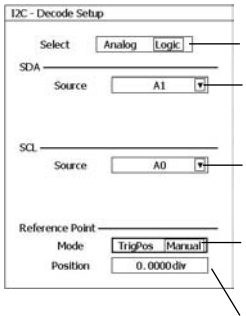
Set the level.

Set the hysteresis.

Set the analysis reference point mode (TrigPos, Manual).

When Mode is set to Manual, set the reference point (±5.00 div).

Logic Signal (DLM6000)



Set Select to Logic.*

Set the SDA source.

Set the SCL source.

Set the analysis reference point mode (TrigPos, Manual).

When Mode is set to Manual, set the reference point (±5.00 div).

* This is fixed to Analog on the DL6000 series, and the setting is not displayed.

Explanation

SDA and SCL Sources

You can select the serial data (SDA) and serial clock (SCL) sources.

- On the DL6000, select the sources from CH1 to CH4 or from M1 to M4.
- On the DLM6000, select the source from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).

Level

Set the level for determining whether the signal level is 0 or 1 for CH1 to CH4 and M1 to M4.*

The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

* If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IIM DLM6054*-01EN.

Hysteresis

Set the hysteresis for CH1 to CH4 and M1 to M4.

The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.

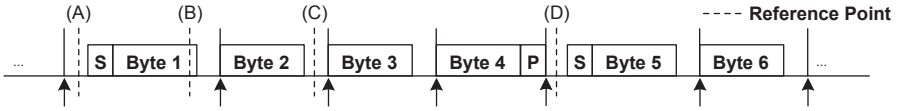
For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Analysis Reference Point

Select the analysis reference point from the following:

Trig Pos	Sets the analysis reference position to the trigger position.
Manual	Set the analysis reference position manually. The selectable range is ±5.00 divisions, and the resolution is 0.01 divisions.

Byte No. 0 in the analysis result list varies depending on the reference point position as follows:



(A): Byte No. 0 → Byte 1 (Byte 2 is No. 1, byte 3 is No. 2, and so on)
(B): Byte No. 0 → Byte 1 (Byte 2 is No. 1, byte 3 is No. 2, and so on)
(C): Byte No. 0 → Byte 2 (byte1 is No. -1, byte 3 is No. 1, and byte 4 is No. 2, and so on)
(D): Byte No. 0 → Byte 5 (byte1 is No. -4, ..., byte 4 is No. -1, and byte 6 is No. 1, and so on)

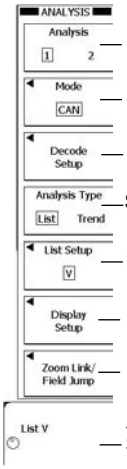
S: Start condition, **P:** Stop condition

4.3 Analyzing a CAN Bus Signal

Procedure

ANALYSIS Menu

Press **ANALYSIS**, the **Mode** soft key, the **Serial Bus** soft key, and then the **CAN** soft key to display the following menu.



Set the analysis number (1, 2).

Set the analysis mode to CAN.

Press to configure the decode settings.

Set the analysis result display method. ► section 4.1

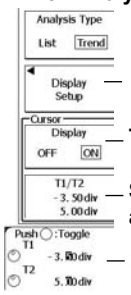
Press to configure the list settings. ► section 4.1

Press to configure the display settings. ► section 4.1

Press to set the link between the list and a zoom window, and to set which field to jump to. ► section 4.1

Scrolls the list. ► section 4.1

When Analysis Type Is Set to Trend



Press to configure the display settings. ► section 4.1

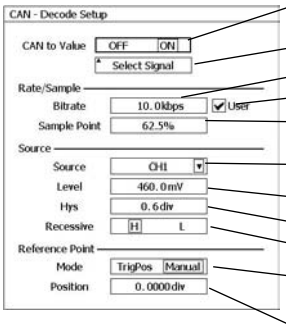
Turns the cursor display on and off. ► section 4.1

Switches the value that the jog shuttle adjusts (T1, T2, T1&T2). ► section 4.1

Moves the cursors. ► section 4.1

Configuring Decode Settings (Decode Setup)

Press the **Decode Setup** soft key to display the following menu.



Set whether to compare the signal (ON) or not (OFF).

Press to select the signal.

Set the bitrate.

Select this checkbox to specify a user-defined bitrate.

Set the sample point.

Set the source (CH1–CH4, M1–M4).

Set the level.

Set the hysteresis.

Set the reference condition (H, L).

Set the analysis reference point mode (TrigPos, Manual).

When Mode is set to Manual, set the reference point (±5.00 div).

Explanation

Signal Symbol Display (CAN to Value)

If you set “Can to Value” to ON, you can specify the physical value/symbol definition file (.sbl)* that has been loaded into the DL6000/DLM6000. If the input signal matches the signal of the specified physical value/symbol definition file, the DL6000/DLM6000 can display the symbol of that signal.
Integer and floating-point data types (32-bit and 16-bit) are supported.

* Physical value/symbol definition files are derived by converting CANdb files (.dbc). You can use YOKOGAWA's free Symbol Editor software to perform this conversion. You can obtain Symbol Editor from the YOKOGAWA website.

Bit Rate

You can select the CAN bus signal transfer rate from the following:
1 Mbps, 500 kbps, 250 kbps, 125 kbps, 83.3 kbps, and 33.3 kbps
If you select the User check box, you can set the transfer rate from 10.0 kbps to 1.000 Mbps in 0.1-kbps steps.

Sample Point

You can set the point for determining the bus level (recessive or dominant) from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 CAN bus signal trigger circuit samples the input CAN bus signal using the internal clock and detects the point of change from recessive to dominant. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate. See the illustration on page 3-19.

Source

Select the source waveform from CH1 to CH4 or from M1 to M4.

Level

Set the level for determining whether the signal level is 0 or 1.
The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

Hysteresis

The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Recessive Level

Set the recessive level to high (H) or low (L). The logical value of the recessive level is 1 and that of the dominant level is 0 in either setting.

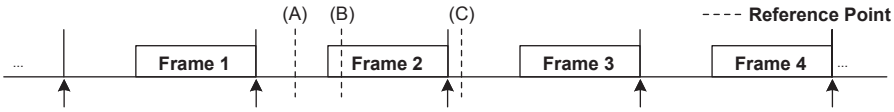
H	The recessive level is higher than the dominant level.
L	The recessive level is less than the dominant level.

Analysis Reference Point

Select the analysis reference point from the following:

Trig Pos	Sets the analysis reference position to the trigger position.
Manual	Set the analysis reference position manually. The selectable range is ±5.00 divisions, and the resolution is 0.01 divisions.

Frame No. 0 in the analysis result list varies depending on the reference point position as follows:



(A): Frame No. 0 → Frame 2 (frame 1 is No. -1, frame 3 is No. 1, and frame 4 is No. 2)
(B): Frame No. 0 → Frame 2 (frame 1 is No. -1, frame 3 is No. 1, and frame 4 is No. 2)
(C): Frame No. 0 → Frame 3 (frame 1 is No. -2, frame 2 is No. -1, and frame 4 is No. 1)

4.4 Analyzing a LIN Bus Signal

Analysis

ANALYSIS Menu

Press **ANALYSIS**, the **Mode** soft key, the **Serial Bus** soft key, and then the **LIN** soft key to display the following menu.

Set the analysis number (1, 2).

Set the analysis mode to LIN.

Press to configure the decode settings.

Press to configure the list settings. ► section 4.1

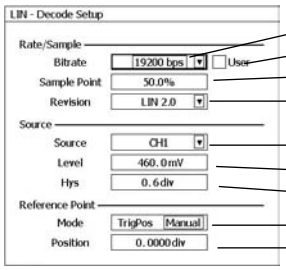
Press to configure the display settings. ► section 4.1

Press to set the link between the list and a zoom window, and to set which field to jump to. ► section 4.1

Scrolls the list. ► section 4.1

Configuring Decode Settings (Decode Setup)

Press the **Decode Setup** soft key to display the following menu.



Set the bitrate.

Select this checkbox to specify a user-defined bitrate.

Set the sample point.

Set the revision (LIN 2.0, LIN 1.3, Both).

Set the source (CH1–CH4, M1–M4, A0–D7).

Set the level.

Set the hysteresis.

Set the analysis reference point mode (TrigPos, Manual).

When Mode is set to Manual, set the reference point (±5.00 div).

Explanation

Bit Rate

You can select the LIN bus signal transfer rate from the following:
19200 bps, 9600 bps, 4800 bps, 2400 bps, 1200 bps
If you select the User check box, you can set the transfer rate from 1000 bps to 20000 bps in 10-bps steps.

Sample Point

You can set the point for determining the bus level from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 LIN bus signal trigger circuit samples the input LIN bus signal using the internal clock and detects the point of level change. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate. See the illustration on page 3-19.

Revision

You can select revision 2.0 or 1.3. Select whether to detect enhanced checksum or classic checksum errors. For details on errors, see page 5.4.

LIN 2.0	Detects errors in the enhanced checksum that includes the protection ID. (However, if the ID is a value from 60 (0x3c) to 63 (0x3f), classic checksum errors will be detected.)
LIN 1.3	Detects classic checksum errors only in the data field.
Both	An error occurs when both LIN 2.0 and LIN 1.3 checksum errors are detected. No error occurs, if only one of the errors is detected.

Source

You can select the source.

- On the DL6000, select the source from CH1 to CH4 or from M1 to M4.
- On the DLM6000, select the source from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).

Level

Set the level for determining whether the signal level is 0 or 1.*
The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

* If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IIM DLM6054*-01EN.

Hysteresis

The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Analysis Reference Point

Select the analysis reference point from the following:

Trig Pos	Sets the analysis reference position to the trigger position.
Manual	Set the analysis reference position manually. The selectable range is ±5.00 divisions, and the resolution is 0.01 divisions.

For details on the analysis reference point and the numbers in the analysis result list, see page 4-16.

4.5 Analyzing a SPI Bus Signal

Procedure

ANALYSIS Menu

Press **ANALYSIS**, the **Mode** soft key, the **Serial Bus** soft key, and then the **SPI** soft key to display the following menu.

Set the analysis number (1, 2).

Set the analysis mode to SPI.

Press to configure the decode settings.

Press to configure the list settings. ► section 4.1

Press to configure the display settings. ► section 4.1

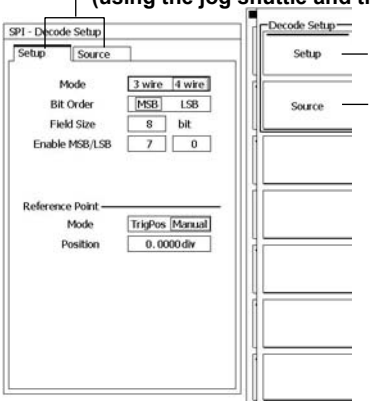
Press to set the link between the list and a zoom window. ► section 4.1

Scrolls the list. ► section 4.1

Configuring Decode Settings (Decode Setup)

Press the Decode Setup soft key to display the following menu.

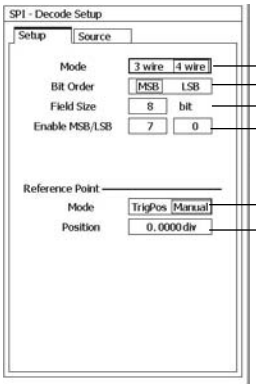
Switch between configuring the analysis settings and the source (using the jog shuttle and the SET key).



Press to configure the analysis settings.

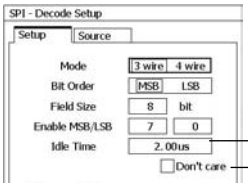
Press to configure the source.

Configuring Analysis Settings (Setup)



Set the mode (3 wire, 4 wire).
Set the bit order (MSB, LSB).
Set the field size (4–32 bits).
Set the range of effective bits (the range depends on the field size setting).
Set the analysis reference point mode (TrigPos, Manual).
When Mode is set to Manual, set the reference point (±5.00 div).

When CS Source in the Source Settings Is Set to None

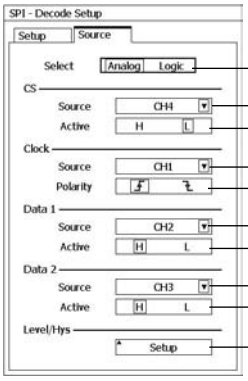


Set the clock's idle time.
Select this checkbox when you want to ignore the idle time setting and analyze from the left edge of the screen.

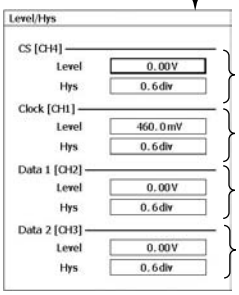
Configuring the Source (Source)

Select the **Source** tab, or press the **Source** soft key to display the following menu.

When Analog Is Selected

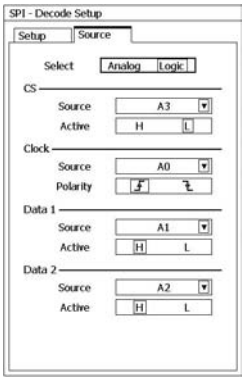


Set the source type to Analog.*
Set the CS source (None, CH1–CH4, M1–M4).
Set the active level (H, L).
Set the Clock source (CH1–CH4, M1–M4).
Set the polarity.
Set the Data 1 source (None, CH1–CH4, M1–M4).
Set the active level (H, L).
Set the Data 2 source (None, CH1–CH4, M1–M4).
Set the active level (H, L).
Press to set the level and hysteresis.



Set the CS source's level and hysteresis.
Set the Clock source's level and hysteresis.
Set the Data 1 source's level and hysteresis.
Set the Data 2 source's level and hysteresis.

When Logic Is Selected (DLM6000)



* This is fixed to Analog on the DL6000 series, and the setting is not displayed.

Explanation

Wiring System

Select the wiring system from the following:

Three-wire	Analyzes the data on a single data line.
Four-wire	Analyzes the data on Data 1 and Data 2 lines.

Bit Order

You can set the bit order to MSB or LSB based on the data stream.

Field Size

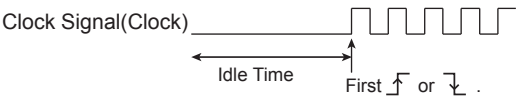
You can set the field size.
It can be set to a value from 4 to 32 bits.

Enabled Bit Range

You can specify the range of bits to enable within the field. Enter the MSB in the box on the left and the LSB in the box on the right.
Only the enabled bits are analyzed.

Clock Idle Time

If you set the CS source to None, the DL6000/DLM6000 will treat the first rising or falling edge of the clock signal after the specified idle time has elapsed as the data start position. If you select the Don't care check box, analysis will start from the left of the screen, regardless of the set idle time value.



CS, Clock, and Data

You can select tthe CS (chip select), clock, and data sources.

- On the DL6000, select the sources from None*1, from CH1 to CH4, or from M1 to M4.
- On the DLM6000, select the sources from None*, from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from None*1, from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).
  * None can only be selected for CS. If you set CS to None, the DL6000/DLM6000 uses the Idle Time setting to determine the data start position.

CS

You can select on which CS level to analyze the data.

H	When the signal is high
L	When the signal is low

Clock

You can select on which clock edge to determine the data status.

⎍	On the rising edge
⎎	On the falling edge

Data 1 and Data 2

You can select which data status to assign to 1 (active) or 0.

H	Set to 1 when the data status is greater than or equal to the specified level or 0 otherwise.
L	Set to 1 when the data status is less than or equal to the specified level or 0 otherwise.

Level

Set the reference level for CH1 to CH4 and M1 to M4.*

The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

* If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IIM DLM6054*-01EN.

Hysteresis

Set the hysteresis for CH1 to CH4 and M1 to M4.

The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.

For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Analysis Reference Point

Select the analysis reference point from the following:

Trig Pos	Sets the analysis reference position to the trigger position.
Manual	Set the analysis reference position manually. The selectable range is ±5.00 divisions, and the resolution is 0.01 divisions.

For details on the analysis reference point and the numbers in the analysis result list, see page 4-12.

4.6 Analyzing an UART Signal

Procedure

ANALYSIS Menu

Press **ANALYSIS**, the **Mode** soft key, the **Serial Bus** soft key, and then the **UART** soft key to display the following menu.

Set the analysis number (1, 2).

Set the analysis mode to UART.

Press to configure the decode settings.

Press to configure the list settings. ► section 4.1

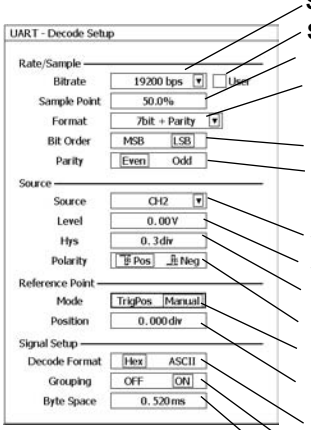
Press to configure the display settings. ► section 4.1

Press to set the link between the list and a zoom window. ► section 4.1

Scrolls the list. ► section 4.1

Configuring Decode Settings (Decode Setup)

Press the **Decode Setup** soft key to display the following menu.



Set the bitrate.

Select this checkbox to specify a user-defined bitrate.

Set the sample point.

Set the format of the analysis data (8bit + Parity, 7bit + Parity, 8bit(NonParity)).

Set the bit order (MSB, LSB).

Set the parity (Even, Odd) (when Format is set to 8bit + Parity or 7bit + Parity).

Set the source (CH1–CH4, M1–M4, A0–D7).

Set the level (when Source is set to CH1–CH4 or M1–M4).

Set the hysteresis (when Source is set to CH1–CH4 or M1–M4).

Set the polarity.

Set the analysis reference point mode (TrigPos, Manual).

When Mode is set to Manual, set the reference point (±5.00 div).

Set the analysis result display method (Hex, ASCII).

Turns grouping on and off.

Set the grouping time (when Grouping is set to ON).

Explanation

Bit Rate

You can set the CAN bus signal transfer rate to
115200 bps, 57600 bps, 38400 bps, 19200 bps, 9600 bps, 4800 bps, 2400 bps, or 1200 bps
If you select the User check box, you can set the transfer rate from 1000 bps to 200000 bps in 100-bps steps.

Sample Point

You can set the point for determining the signal level from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 UART signal trigger circuit samples the input UART signal using the internal clock and detects the point of level change. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate. See the illustration on page 3-19.

Format

You can select the format from the following:

8bit + Parity	8-bit data + parity bit
7bit + Parity	7-bit data + parity bit
8bit(NonParity)	8-bit data with no parity bit

Bit Order

Select the input signal bit order.

MSB	Reads the data pattern MSB first.
LSB	Reads the data pattern LSB first.

Parity

Set the parity bit to even or odd.

Source

You can select the source.

- On the DL6000 Series, select the source from CH1 to CH4 or from M1 to M4.
- On the DLM6000 Series, select the source from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).

Level

Set the level for determining whether the signal level is 0 or 1.*
The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

* If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IIM DLM6054*-01EN.

Hysteresis

The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
For details on the analysis reference point and the numbers in the analysis result list, see page 4-12.

Polarity

You can select the bit state that will be considered logical 1.

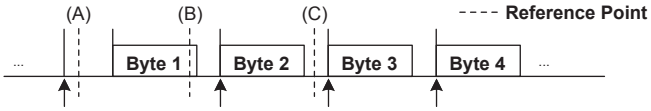
Pos	Positive logic
Neg	Negative logic

Analysis Reference Point

Select the analysis reference point from the following:

Trig Pos	Sets the analysis reference position to the trigger position.
Manual	Set the analysis reference position manually. The selectable range is ±5.00 divisions, and the resolution is 0.01 divisions.

Byte No. 0 in the analysis result list varies depending on the reference point position as follows:



(A): Byte No. 0 → Byte 1 (Byte 2 is No. 1, byte 3 is No. 2, and so on)
(B): Byte No. 0 → Byte 1 (Byte 2 is No. 1, byte 3 is No. 2, and so on)
(C): Byte No. 0 → Byte 2 (byte1 is No. -1, byte 3 is No. 1, and byte 4 is No. 2, and so on)

Analysis Result Display Mode

- You can set the data display format to hexadecimal or ASCII.
  - For ASCII, code names appear for control codes such as LF.
  - Data values that are greater than or equal to 7F appear in hexadecimal even if you select ASCII.
- You can select whether or not to display data that are shorter than the specified byte space as one consolidated group data.
- Byte space time

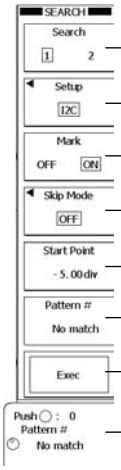
Selectable range	From the time corresponding to the sum of the number of bits in the UART signal data format and 2 bits to 100 ms The 2 bits above correspond to the start and stop bits. For example, if the data format is 8 bits + a parity bit, the time is equal to: Data (8) + Parity Bit (1) + Start Bit (1) + Stop Bit (1) = 11 bits
Resolution	Time corresponding to 1 bps
Default value	The time corresponding to the sum of the number of bits in the UART signal data format and 2 bits

5.1 Selecting the Serial Bus Signal and Skip Mode, Executing the Search, and Displaying the Results

Procedure

SEARCH Menu

Press **SEARCH** to display the following menu.



Set the search condition number (1, 2).

Press to configure the search conditions. ► sections 5.2–5.6

Turns the display of detected point markers on and off

Set the skip mode (OFF, Hold Off, Decimation).

Set the search start point (using the jog shuttle).

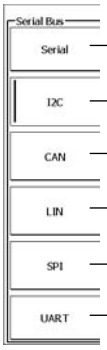
Specify the detected point that you want to display (using the jog shuttle).

Executes a search.

Set the detected point number and the search start point.

Setting the Serial Bus Type

Press the **Setup** soft key, the Type soft key, and then the **Serial Bus** soft key to display the following menu.



Serial (see section 10.9 in IM DLM6054-01EN)

I2C ► section 5.2

CAN ► section 5.3

LIN ► section 5.4

SPI ► section 5.5

UART ► section 5.6

Explanation

Search Type

This manual describes I^2C, CAN, LIN, SPI, and UART serial bus signal search features. The DL6000/DLM6000 searches for sections in the target signal that meet specified conditions and zooms in on the sections. For information about other search features, see the *User's Manual IM DLM6054-01EN*.

Detected Point Mark (Mark)

Select whether or not to display marks on the detected points. When this feature is turned on, detected point marks appear at the top section of the main window.

ON: Displays the marks

OFF: Does not display the marks

Skip Mode

After finding a point that meets the search conditions, the DL6000/DLM6000 skips searching for the specified time or count.

OFF	Searches all found points.
Hold Off	Skips searching for the specified time. The selectable range is 0.1 ns to 1.00000 s (six significant digits). The resolution is 0.1 ns.
Decimation	Skips searching for the specified count. The selectable range is 1 to 9999.

Search Start Point

The selectable range is ±5.00 divisions. The resolution is 0.01 divisions.

Displaying Search Results

Numbers are assigned to the points that are found. Zero is assigned to the first found point, one is assigned to the second found point, and so on.

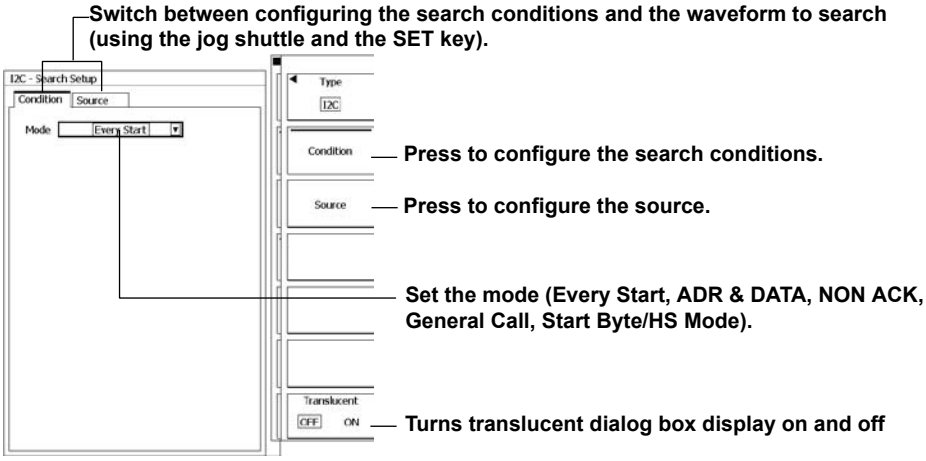
- The maximum found-point number is 4999.
- You can display the waveform that corresponds to the selected found-point number in the zoom waveform area.

5.2 Searching I²C Bus Signals

Procedure

Setup Menu

Press **SEARCH,** the **Setup** soft key, the **Type** soft key, the **Serial Bus** soft key, and then the **I2C** soft key to display the following menu.

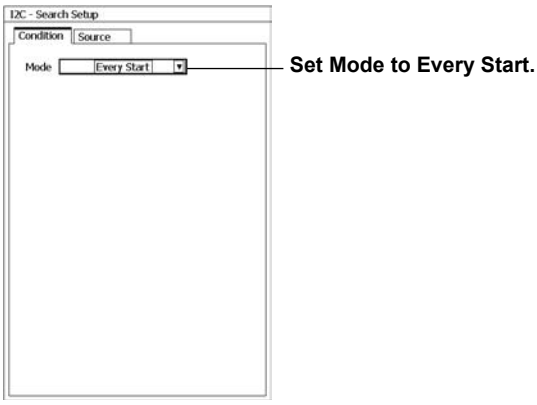


Setting Search Conditions (Condition)

Depending on the address to compare, there are five modes:

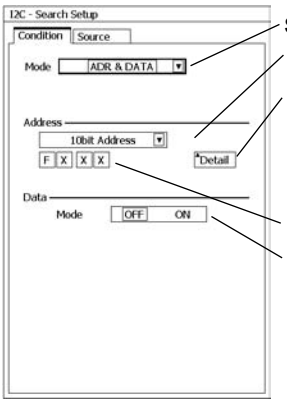
- Every Start
- ADR & DATA
- NON ACK
- General Call
- Start Byte/HS Mode

When Searching in Every Start Mode



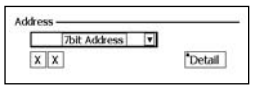
When Searching in ADR & DATA Mode

When 10bit Address Is Selected

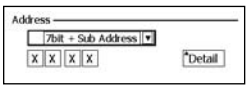


Set Mode to ADR & DATA.
Set the address.
Press to configure the address pattern details. (The configuration is the same as is explained in the section on triggering.)
Set the address pattern.
Set whether or not to make the data pattern a trigger condition.

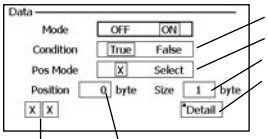
When 7bit Address Is Selected



When 7bit + Sub Address Is Selected

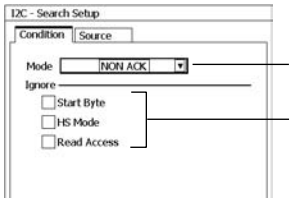


Making a Data Pattern a Trigger Condition



Set the comparison condition.
Set the comparison method.
Set the data length.
Press to configure the data pattern details. (The configuration is the same as is explained in the section on triggering.)
Set the data pattern.
Set the comparison start position for when Pos Mode is set to Select.

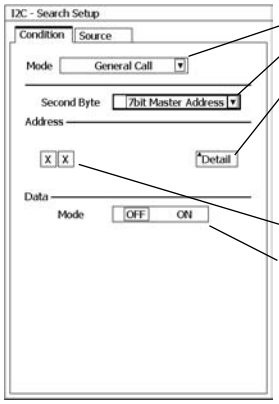
When Searching in NON ACK Mode



Set Mode to NON ACK.
Select the target acknowledge bits.

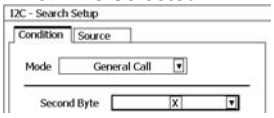
When Searching in General Call Mode

When 7bit Master Address Is Selected

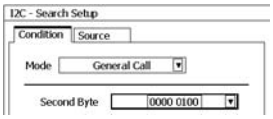


Set Mode to General Call.
Set the second byte.
Press to configure the address pattern details. (The configuration is the same as is explained in the section on triggering.)
Set the address pattern.
Set whether or not to make the data pattern a search condition.

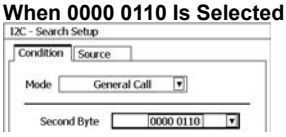
When X Is Selected



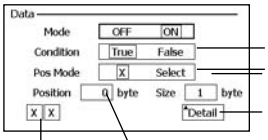
When 0000 0100 Is Selected



When 0000 0110 Is Selected



Making a Data Pattern a Search Condition



Set the comparison condition.
Set the comparison method.
Set the data length.
Press to configure the data pattern details. (The configuration is the same as is explained in the section on triggering.)
Set the data pattern.
Set the comparison start position for when Pos Mode is set to Select.

When Searching in Start Byte/HS Mode

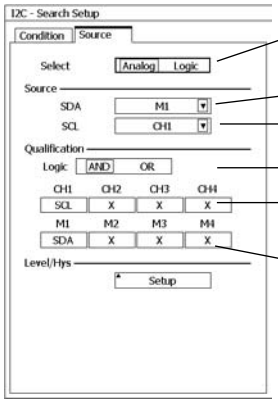
Set Mode to Start Byte/HS Mode.
Select Start Byte or HS Mode.

Setting the Waveform to Search (Source)

Specify the waveform to compare with the data pattern and the comparison conditions.

Select the **Source** tab, or press the **Source** soft key to display the following menu.

For Analog Sources



Set the type of the waveform to search to Analog.*

Set the SDA source (CH1–CH4, M1–M4).

Set the SCL source (CH1–CH4, M1–M4).

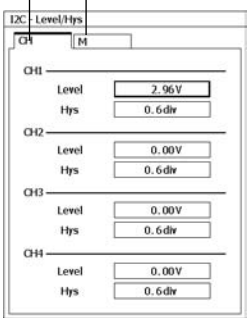
Set the logic (AND, OR).

Set the conditions for channels other than the SDA and SCL sources (X, H, L).

Press to set the level and hysteresis of the SDA, SCL, and Qualification source channels.

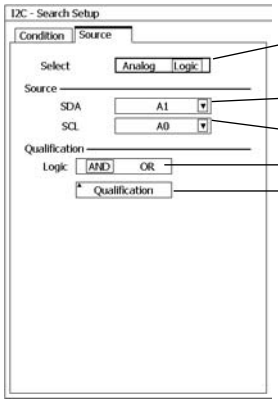
Channel waveforms

Math waveforms



* This is fixed to Analog on the DL6000 series, and the setting is not displayed.

For Logic Sources (DLM6000)



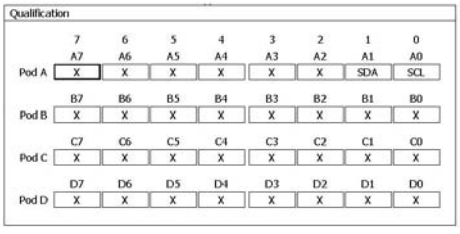
Set the type of the waveform to search to Logic.

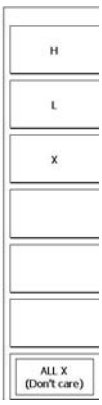
Set the SDA source.

Set the SCL source.

Set the logic.

Press to set the qualifications.





Executing the Search

To execute the search, follow the instructions given in section 5.1.

Explanation

This feature searches I^2C bus signals. For details on the I^2C bus signal data format, see “Explanation” in section 3.1.

Modes

Set the I^2C search mode to Every Start, ADR & DATA, NON ACK, General Call, or Start Byte/HS Mode.

Address

- You can set the address type to 7bit Address, 7bit + Sub Address, or 10bit Address.
- Set the address pattern in hexadecimal or binary notation. The address search condition is met when the specified address pattern matches the input signal address pattern.
- If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
- If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

Data

You can select whether or not to use the data pattern as a search condition.

- **Comparison Condition**

  The data search condition is met when the result of comparing the input signal pattern with the specified pattern meets the selected comparison condition.

True	When the patterns match
False	When the patterns don’t match

- **Comparison Start Position**

  In the Pos Mode setting, you can set the comparison start point to the specified point (Select) or don’t care (X). If you select Select, the DL6000/DLM6000 skips the specified number of bytes and starts comparing from the next data byte.

  Selectable range: 0 to 9999

- **Data Size**

  Set how many consecutive data bytes you want to compare.

  Selectable range: 1 to 4

- **Data Pattern**

  Set the data pattern for the specified size in hexadecimal or binary notation.

  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

SDA, SCL, and Qualification

SDA and SCL Sources

You can select the serial data (SDA) and serial clock (SCL) sources.

- On the DL6000, select the sources from CH1 to CH4 or from M1 to M4.
- On the DLM6000, select the sources from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).

- **Level**

  Set the level for determining whether the signal level is 0 or 1 for CH1 to CH4 and M1 to M4.*
  The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions.
  For example, if the T/div setting is 2 mV/division, the resolution is 0.02 mV.

  * If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-0*1EN

- **Hysteresis**

  Set the hysteresis for CH1 to CH4 and M1 to M4.
  The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
  For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Qualification and Logic

- **Qualification**

  Set the state of signals other than those selected for the SDA and SCL to H, L, or X. This search requirement is called qualification requirement. The qualification requirement is met when the selected state matches the input signal state.

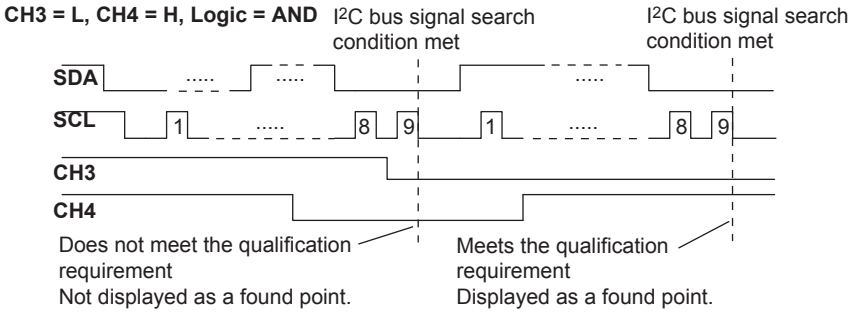
H	When the signal is high
L	When the signal is low
X	Not used as a search condition (Don't care)

  * The level for determining high or low is the level that you set above when you set the signal to a channel from CH1 to CH4 or from M1 to M4. If the source is set to a signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-0*1EN

- **Logical Condition**

  You can select the logical condition for the qualification and the search condition for the I^2C bus signal that you set in each mode. The DL6000/DLM6000 searches for points where the logic condition is met.

AND	When the qualification requirement and the I^2C bus signal search condition are both met
OR	When either the qualification requirement or the I^2C bus signal search condition is met



Note

To search using only the I^2C bus signal search condition (SDA and SCL signals), specify the settings as follows:

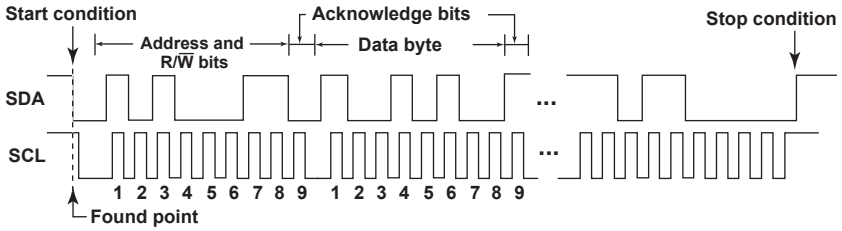
- The state of signals other than those selected for the SDA and SCL: X (don't care)
- Logic: AND

Found Point

The points that the DL6000/DLM6000 finds vary depending on the mode as follows:

- **Every Start mode**

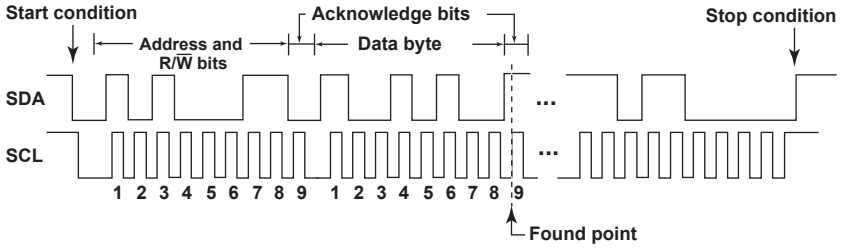
  Like the trigger point, the found point will be at the falling edge of the SDA signal.



- **A mode other than Every Start**

  The found point will be at the rising edge of the acknowledge bit of SCL after the spe condition is met.

  The following example is for the case when the mode is ADR & DATA, but it applies to other search modes as well.

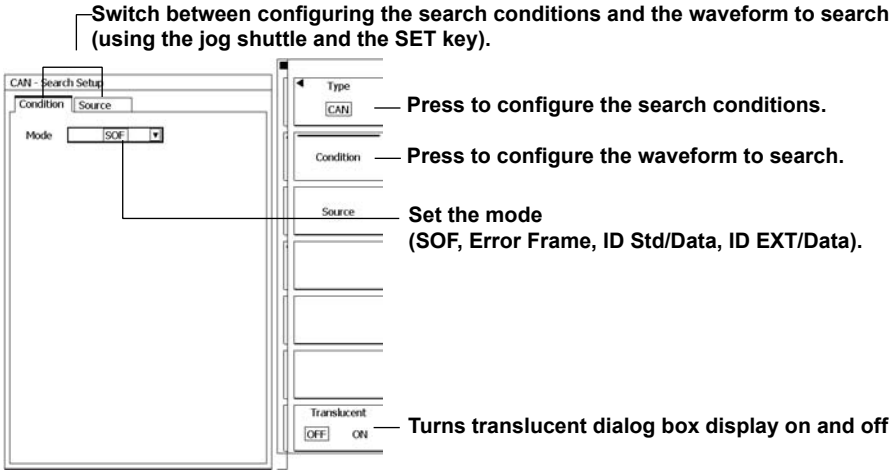


5.3 Searching CAN Bus Signals

Procedure

Setup Menu

Press **SEARCH**, the **Setup** soft key, the **Type** soft key, the **Serial Bus** soft key, and then the **CAN** soft key to display the following menu.

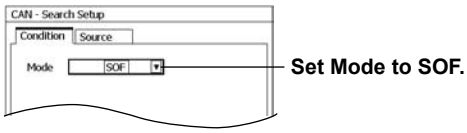


Setting Search Conditions (Condition)

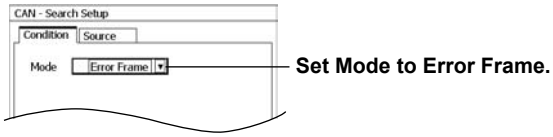
Depending on the address to compare, there are five modes:

- SOF
- Error Frame
- ID Std/Data
- ID EXT/Data
- Msg/Signal

When Searching in SOF Mode

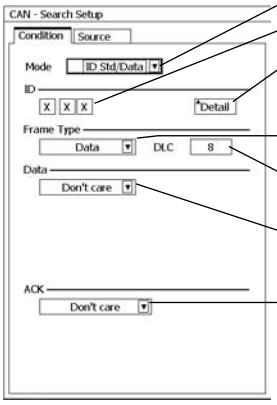


When Searching in Error Frame Mode



When Searching in ID Std/Data or ID Ext/Data Mode

ID Std/Data



Set Mode to ID Std/Data.
Set the bit pattern to compare.
Press to configure the bit pattern details. (The configuration is the same as is explained in the section on triggering.)
Set the frame type comparison condition (Don't Care, Remote, Data).
Set the DLC (when Frame Type is set to Data).
Set the data comparison condition (when Frame Type is set to Data).
Set the ACK condition (Don't Care, NON ACK, ACK, NON ACK or ACK) (when Frame Type is set to Data).

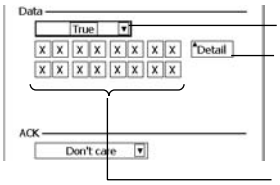
ID Ext/Data

Making Frame Type's Data a Search Condition

The settings vary depending on the Data comparison conditions.

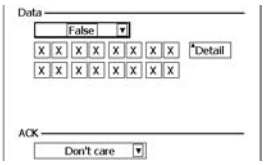
If the Data Comparison Condition Is True or False

True



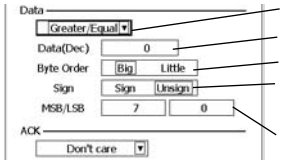
Select True.
Press to configure the data pattern details. (The configuration is the same as is explained in the section on triggering.)
Set the data pattern.

False



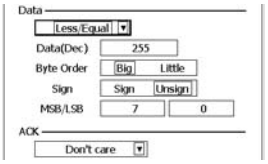
If the Data Comparison Condition is Greater/Equal or Less/Equal

Greater/Equal



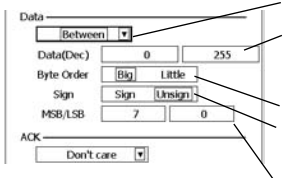
Select Greater/Equal.
Set the reference value.
Set the byte order (endian).
Set whether data is signed or unsigned.
Set the position of the most significant bit (left text box) and least significant bit (right text box).

Less/Equal



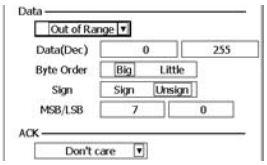
If the Data Comparison Condition is Between or Out of Range

Between



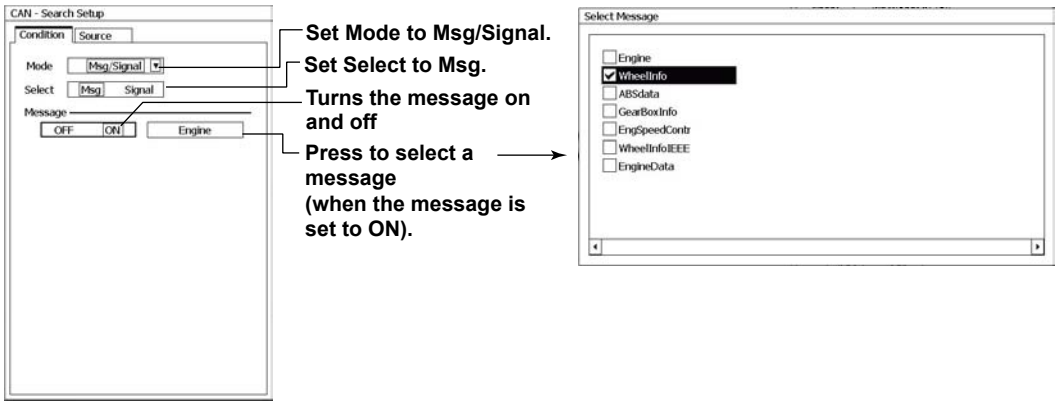
Select Between.
Set the lower limit (left text box) and upper limit (right text box) of the reference range.
Set the byte order (endian).
Set whether data is signed or unsigned.
Set the position of the most significant bit (left text box) and least significant bit (right text box).

Out of Range



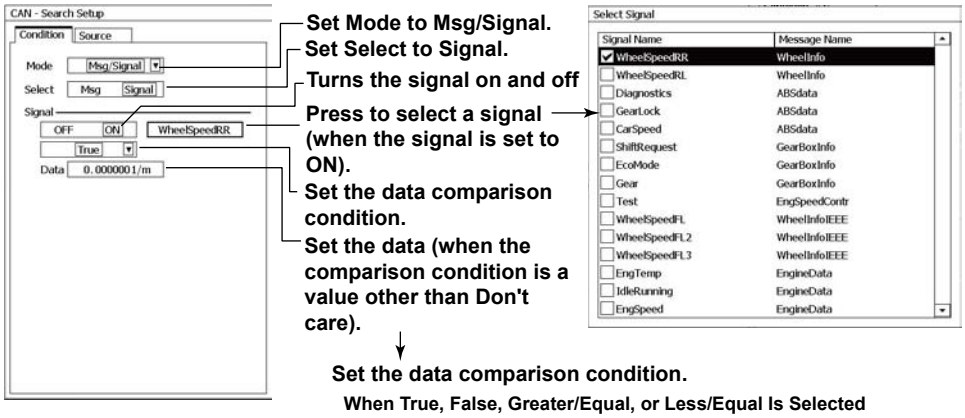
When Searching in Msg/Signal Mode

Making a Message a Search Condition



Set Mode to Msg/Signal.
Set Select to Msg.
Turns the message on and off
Press to select a message (when the message is set to ON).

Making a Signal a Search Condition



Set Mode to Msg/Signal.
Set Select to Signal.
Turns the signal on and off
Press to select a signal (when the signal is set to ON).
Set the data comparison condition.
Set the data (when the comparison condition is a value other than Don't care).

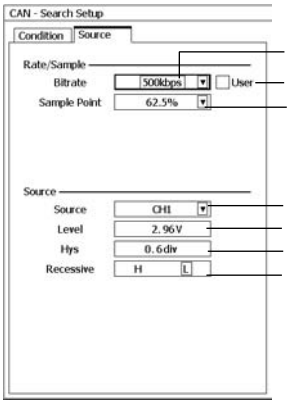
Set the data comparison condition.

When True, False, Greater/Equal, or Less/Equal Is Selected

When Between or Out of Range Is Selected

Setting the Waveform to Search (Source)

Select the **Source** tab, or press the **Source** soft key to display the following menu.



Set the bitrate.
Select this checkbox to specify a user-defined bitrate.
Set the sample point.
Set the waveform to search.
Set the level.
Set the hysteresis.
Set the recessive level.

Executing the Search

To execute the search, follow the instructions given in section 5.1.

Explanation

This feature searches CAN bus signals. For details on the CAN bus signal frame format, see "Explanation" in section 3.2.

Modes

Set the CAN search mode to SOF, Error Frame, ID Std/Data, and ID Ext/Data.

SOF Mode

Searches for the frame start point of a CAN bus signal.
SOF: Start of Frame

Error Frame Mode

The DL6000/DLM6000 searches for the point of error when the error frame's error flag is active.

ID Std/Data and ID Ext/Data Modes

ID Std/Data mode is used to search the data frame or remote frame in standard format.
ID Ext/Data mode is used to search the data frame or remote frame in extended format.
The DL6000/DLM6000 searches using the AND logic of ID, Frame Type, Data, and ACK conditions.
The settings in ID Std/Data mode are shared with the settings in ID Ext/Data mode.

- **ID**

  Set the ID bit pattern in hexadecimal or binary notation. The ID bit pattern is 11 bits in standard format and 29 bits in extended format. The ID search condition is met when the specified bit pattern matches the input signal ID bit pattern.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be "$."

- **Frame Type**

  The DL6000/DLM6000 can be configured to search the remote frame or data frame.
  Selecting the Frame
  A CAN bus signal frame contains a Remote Transmission Request (RTR) bit that indicates whether the frame is a remote frame or a data frame. Select the frame to search.

Don't care	Searches both remote frames and data frames.
Remote	Searches remote frames.
Data Frame	Searches data frames.

  If you select Don't care or Remote, the DLC and Data search conditions in the next section will be ignored.
  DLC (Data Length Code)
  Set the data field length. The DLC search condition is met when the input signal DLC value matches the reference value. Set this value only when the frame type is set to Data Frame.
  Selectable range: 0 to 8
  If you set this value to zero, the data search conditions in the next section will be ignored.

- **Data**

  You can use the Data Field value as a search condition. Set this value only when the frame type is set to Data Frame.

  Comparison Condition

  The data search condition is met when the result of comparing the input signal data field values with the reference values meets the selected comparison condition.

Don't care	Not used as a search condition
True	When the input signal value meets the reference value
False	When the input signal value does not match the reference value
Greater/Equal	When the input signal value is greater than or equal to the reference value
Less/Equal	When the input signal value is less than or equal to the reference value
Between	When the input signal value is within the reference range that includes the boundary reference values
Out of Range	When the input signal value is outside the reference range that excludes the boundary reference values

  Data Pattern

  Set the data pattern for the length specified by DLC in hexadecimal or binary notation. The data pattern is valid only when the comparison condition is set to true or false.

  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be "$."

  Reference Value Data(Dec)

  - If you set the comparison condition to Greater/Equal or Less/Equal, set one reference value.
  - If you select Between or Out of Range, set two reference values to define a reference range. The values are automatically adjusted so that the lower limit is less than or equal to the upper limit.
  - If the comparison condition is True or False, the data pattern is used as the reference value.
  - Selectable range

    Set the selectable range in decimal notation.

Unsigned	0 to 9E+18 The selectable maximum value is limited by the data length and bit position that are determined by the DLC and MSB/LSB settings, respectively.
Signed	–9E+18 to 9E+18 The selectable minimum and maximum values are limited by the data length and bit position that are determined by the DLC and MSB/LSB settings, respectively.

    The value is displayed in exponential notation when it exceeds 7 digits (example: 1234567E+10).

  Byte Order (Byte Order)

  Set the data byte order to big endian (Big) or little endian (Little). For an example, see page 3-15.

  Sign

  Select whether or not to add a sign to the data.

  The selectable range for the data reference value varies depending on this setting.

  MSB/LSB

  Set the MSB and LSB positions in the data to compare. For an example, see page 3-18.

  Selectable range: 0 to the data size bytes × 8 – 1. The maximum value is 63.

- **ACK**

  You can use the ACK slot state as a search condition. The ACK search condition is met when the selected state matches the input signal ACK slot state.

Don't care	Not used as a search condition
NON ACK	When the status is recessive
ACK	When the status is dominant
NON ACK or ACK	When the status is recessive or dominant

Bit Rate, Sample Point, Level, Hysteresis, and Recessive Level

Bit Rate

You can select the CAN bus signal transfer rate from the following:
1 Mbps, 500 kbps, 250 kbps, 125 kbps, 83.3 kbps, and 33.3 kbps
If you select the User check box, you can set the transfer rate from 10.0 kbps to 1.000 Mbps in 0.1-kbps steps.

Sample Point

You can set the point for determining the bus level (recessive or dominant) from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 CAN bus signal trigger circuit samples the input CAN bus signal using the internal clock and detects the point of change from recessive to dominant. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate. See the illustration on page 3-19.

Source

Select the source from CH1 to CH4 or from M1 to M4.

- **Level**

  Set the level for determining whether the signal level is 0 or 1.
  The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions.
  For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

- **Hysteresis**

  The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
  For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

- **Recessive Level**

  Set the recessive level to high (H) or low (L). The logical value of the recessive level is 1 and that of the dominant level is 0 in either setting.

H	The recessive level is higher than the dominant level.
L	The recessive level is less than the dominant level.

Found Point

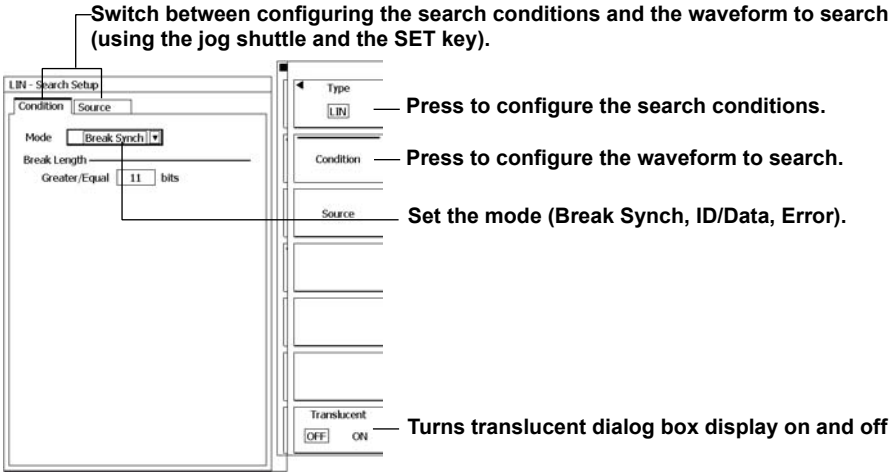
The found-point position is the same as the trigger point position. For a description of the trigger point, see "Explanation" in section 3.2.

5.4 Searching LIN Bus Signals

Procedure

Setup Menu

Press **SEARCH**, the **Setup** soft key, the **Type** soft key, the **Serial Bus** soft key, and then the **LIN** soft key to display the following menu.

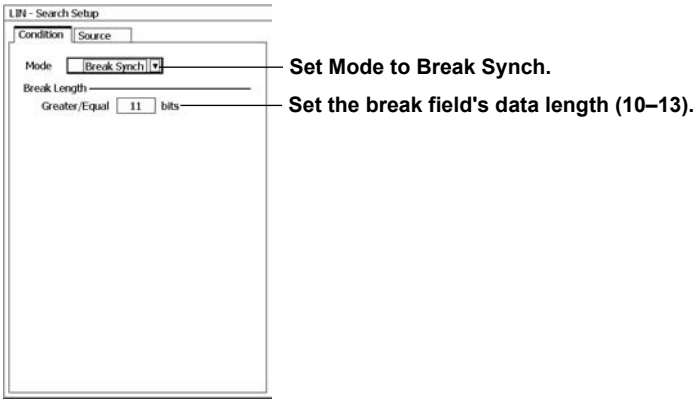


Setting Search Conditions (Condition)

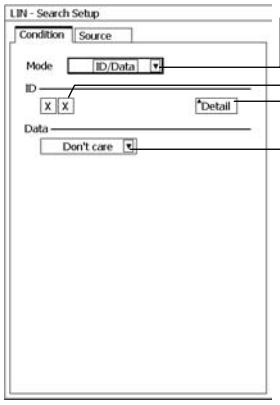
Depending on the address to compare, there are three modes:

- Break Synch
- ID/Data
- Error

When Searching in Break Synch Mode



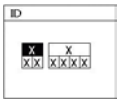
When Searching in ID/Data Mode



Set Mode to ID/Data.

Set the bit pattern to compare.

Press to configure the bit pattern details.

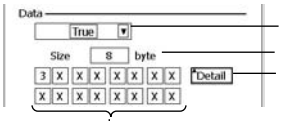


Set the data comparison condition.

Detailed Bit Pattern Configuration
Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the bit pattern.

If the Data Comparison Condition Is True or False

True



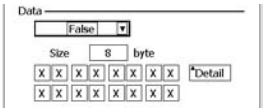
Select True.

Set the length of data to compare.

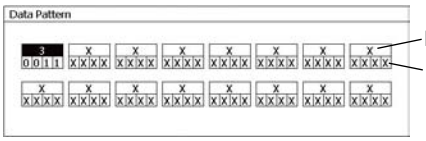
Press to configure the data pattern details.

Set the data pattern.

False



Detailed Data Pattern Configuration
Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the data pattern.

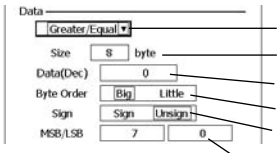


Hexadecimal

Binary

If the Data Comparison Condition is Greater/Equal or Less/Equal

Greater/Equal



Select Greater/Equal.

Set the length of data to compare.

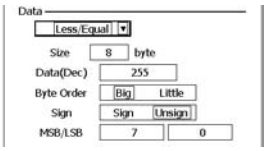
Set the reference value.

Set the byte order (endian).

Set whether data is signed or unsigned.

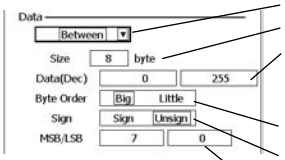
Set the position of the most significant bit (left text box) and least significant bit (right text box).

Less/Equal



If the Data Comparison Condition is Between or Out of Range

Between



Select Between.

Set the length of data to compare.

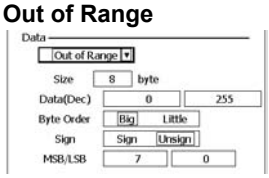
Set the lower limit (left text box) and upper limit (right text box) of the reference range.

Set the byte order (endian).

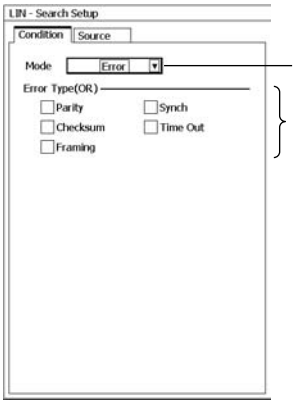
Set whether data is signed or unsigned.

Set the position of the most significant bit (left text box) and least significant bit (right text box).

Out of Range



When Searching in Error Mode

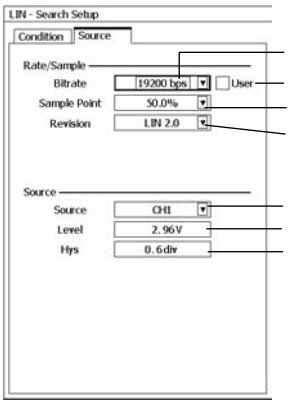


Set Mode to Error.

Set the error types.

Setting the Waveform to Search (Source)

Select the **Source** tab, or press the **Source** soft key to display the following menu.



Set the bitrate.

Select this checkbox to specify a user-defined bitrate.

Set the sample point.

Set the revision (LIN 2.0, LIN 1.3, Both).

Set the waveform to search.

Set the level.

Set the hysteresis.

Executing the Search

To execute the search, follow the instructions given in section 5.1.

Explanation

This feature searches LIN bus signals.

Modes

Set the LIN search mode to Break Synch, ID/Data, and Error.

Break Synch Mode

The DL6000/DLM6000 searches for points where break field + synch field are detected.
Select the break field data length from the following:
Greater then equal to 10, 11, 12, or 13

ID/Data Mode

The DL6000/DLM6000 searches using the AND logic of ID and Data conditions.

- **ID**

  Set the 6-bit protected ID (ID0 to ID5) bit pattern in the protected identifier field in hexadecimal or binary notation. The ID search condition is met when the specified bit pattern matches the input signal ID bit pattern.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

- **Data**

  You can use the Data 1 to Data 8 values as a search condition.
  Comparison Condition
  The data search condition is met when the result of comparing the input signal data values with the reference values meets the selected comparison condition.

Don’t care	Not used as a search condition
True	When the input signal value meets the reference value
False	When the input signal value does not match the reference value
Greater/Equal	When the input signal value is greater than or equal to the reference value
Less/Equal	When the input signal value is less than or equal to the reference value
Between	When the input signal value is within the reference range that includes the boundary reference values
Out of Range	When the input signal value is outside the reference range that excludes the boundary reference values

  Data Size
  Set the data length to search.
  Selectable range: 1 to 8 bytes
  Data Pattern
  Set the data pattern for the specified size in hexadecimal or binary notation. The data pattern is valid only when the comparison condition is set to true or false.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

Reference Value Data(Dec)

- If you set the comparison condition to Greater/Equal or Less/Equal, set one reference value.
- If you select Between or Out of Range, set two reference values to define a reference range. The values are automatically adjusted so that the lower limit is less than or equal to the upper limit.
- If the comparison condition is True or False, the data pattern is used as the reference value.
- Selectable range

  Set the selectable range in decimal notation.

Unsigned	0 to 9E+18 The selectable maximum value is limited by the data length and bit position that are determined by the Size and MSB/LSB settings, respectively.
Signed	–9E+18 to 9E+18 The selectable minimum and maximum values are limited by the data length and bit position that are determined by the Data Size and MSB/LSB settings, respectively.

  The value is displayed in exponential notation when it exceeds 7 digits (example: 1234567E+10).

Byte Order

Set the data byte order to big endian or little endian. For an example, see page 3-17.

Sign

Select whether or not to add a sign to the data.

The selectable range for the data reference value varies depending on this setting.

MSB/LSB

Set the MSB and LSB positions in the data to compare. For an example, see page 3-18.

Selectable range: 0 to the data size bytes × 8 – 1. The maximum value is 63.

Error Mode

Searches for points where errors occurred.

You can select the type of errors to detect from the table below.

- You can select multiple error types.
- The DL6000/DLM6000 will trigger if any of the selected errors occurs.

Parity	The DL6000/DLM6000 calculates the parity of the protected identifier field. If the result does not meet the following equations, an error occurs. • Even parity check: ID0 xor ID1 xor ID2 xor ID4 xor P0 = 0 P0 = ID0 xor ID1 xor ID2 xor ID4 • Odd parity check: ID1 xor ID3 xor ID4 xor ID5 xor P1 = 1 P1 = ¬(ID1 xor ID3 xor ID4 xor ID5)
Checksum	Revision LIN 2.0 (enhanced checksum) If the total value*1 of the protected identifier field, all data fields, and checksum is not 0xFF, an error occurs. However, if the protected identifier field ID is from 0x60 to 0x63, the DL6000/DLM6000 checks based on the calculated result of the classic checksum. Revision LIN 1.3 (classic checksum) If the calculated result of all data fields and checksum is not 0xFF, an error occurs.
Synch	If the synch field is not 0x55, an error occurs. Even if the synch field is 0x55, if the input signal bit rate is not within –5.6% to 6.3% of the specified bit rate (see the next section for details), an error occurs.
Timeout	• Slave Not Responding Error If the frame has not ended by the time defined by the following equation elapses after a break detection, an error occurs. $1.4 \times (T_{Header}{}^{*2} + T_{Response}{}^{*3})$ • Header Timeout Error If the header has not ended by the time defined by the following equation elapses after a break detection, an error occurs. $1.4 \times T_{Header}{}^{*2}$ • Response Timeout Error If the response has not ended by the time defined by the following equation elapses after a break detection, an error occurs. $1.4 \times T_{Response}{}^{*3}$ where 34 is the header data length, n is the number of data points, and 1 is the checksum.
Framing	When the DL6000/DLM6000 detects that the field, data, or stop bit is at low level, an error occurs. If it detects break field + synch field in the middle of a frame, an error occurs.

*1 If the value exceeds 255, it is carried over.

*2 Nominal header length $T_{Header} = 34 \times T_{BIT}{}^{*4}$

*3 Nominal response length $T_{Response} = 10 \times (N + 1) \times T_{BIT}{}^{*4}$ (where N is the data length)

*4 Nominal time needed to transmit one bit defined in the physical layer.

Note

If the bus contains a frame with a different data length and you set the error type to Checksum, Timeout, or Framing, the DL6000/DLM6000 may not be able to search the correct position.

Bit Rate, Sample Point, Revision, Level, and Hysteresis

Bit Rate

You can select the LIN bus signal transfer rate from the following:

19200 bps, 9600 bps, 4800 bps, 2400 bps, 1200 bps

If you select the User check box, you can set the transfer rate from 1000 bps to 20000 bps in 10-bps steps.

Sample Point

You can set the point for determining the bus level from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 LIN bus signal trigger circuit samples the input LIN bus signal using the internal clock and detects the point of level change. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate. See the illustration on page 3-19.

Revision

You can select revision 2.0 or 1.3. If the search mode is set to Error and the error type is set to Checksum, select whether to detect enhanced checksum or classic checksum errors.

LIN 2.0	Detects errors in the enhanced checksum that includes the protection ID. (However, if the ID is a value from 60 (0x3c) to 63 (0x3f), classic checksum errors will be detected.)
LIN 1.3	Detects classic checksum errors only in the data field.
Both	An error occurs when both LIN 2.0 and LIN 1.3 checksum errors are detected. No error occurs, if only one of the errors is detected.

Source

You can select the source.

- On the DL6000, select the source from CH1 to CH4 or from M1 to M4.
- On the DLM6000, select the source from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).

- **Level**

  Set the level for determining whether the signal level is 0 or 1.*
  The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions.
  For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.
  * If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-0*1EN

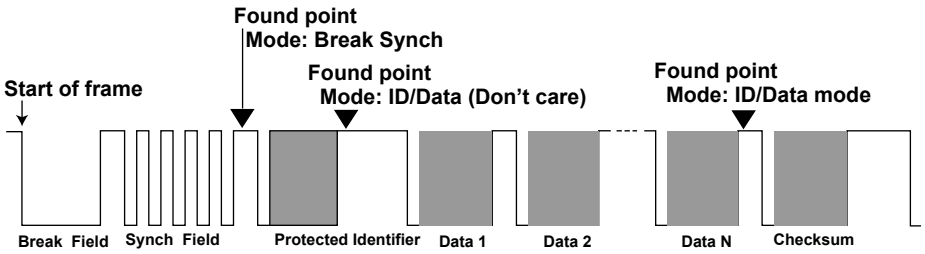
- **Hysteresis**

  The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
  For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Found Point

Break Synch, ID, and Data Modes

Below is an example.



Error Mode

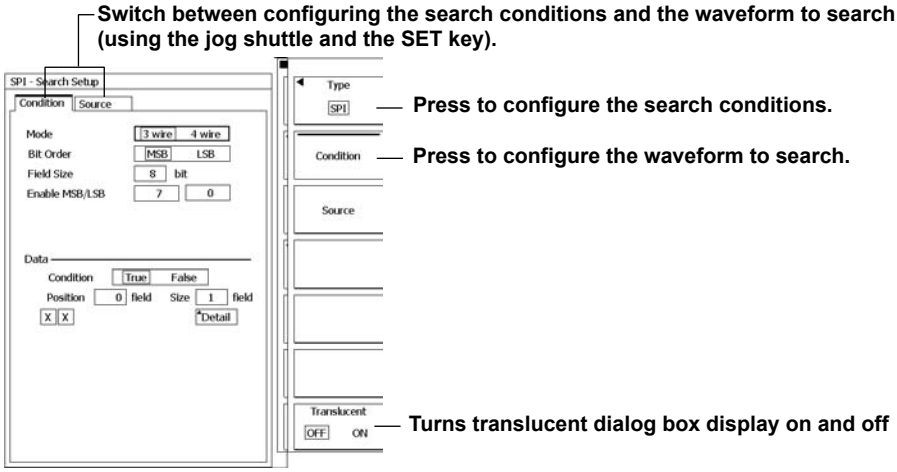
The search start position is where an error occurs.

5.5 Searching SPI Bus Signals

Procedure

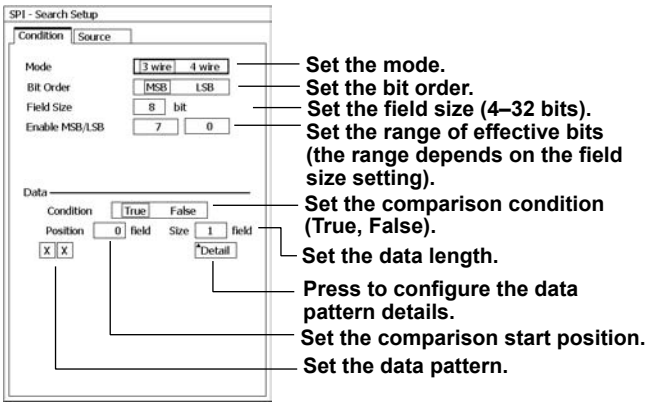
Setup Menu

Press **SEARCH**, the **Setup** soft key, the **Type** soft key, the **Serial Bus** soft key, and then the **SPI** soft key to display the following menu.

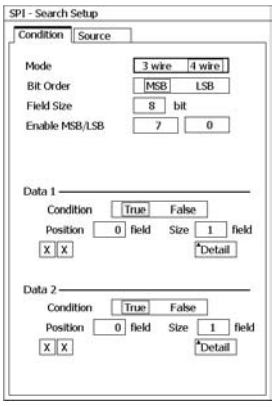


Setting Search Conditions (Condition)

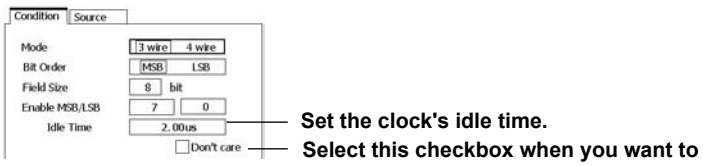
When Mode Is Set to 3Wire



When Mode Is Set to 4Wire



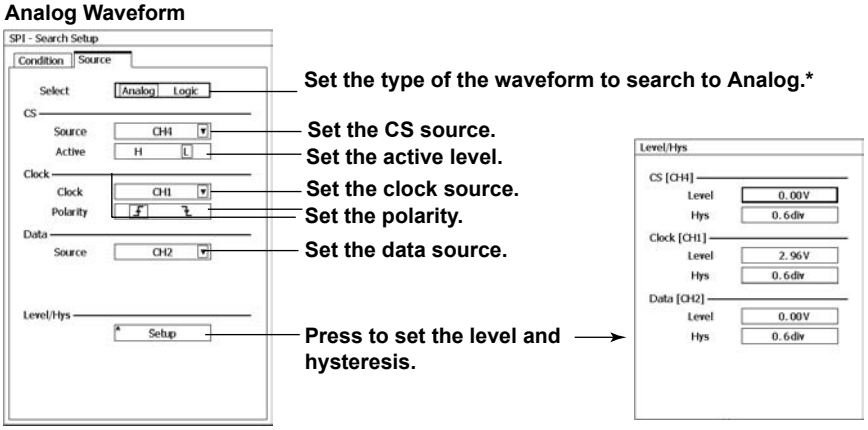
When CS Source in the Source Settings Is Set to None



Setting the Waveform to Search (Source)

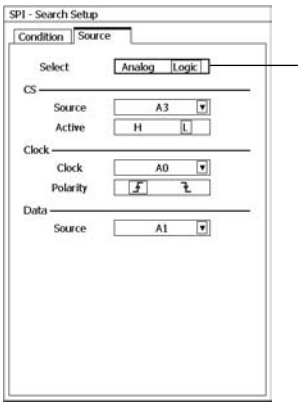
Select the **Source** tab, or press the **Source** soft key to display the following menu.

Analog Waveform



* This is fixed to Analog on the DL6000 series, and the setting is not displayed.

Logic Waveform (DLM6000)



Executing the Search

To execute the search, follow the instructions given in section 5.1.

Explanation

This feature searches SPI bus signals. For the SPI bus signal time chart, see “Explanation” in section 3.4.

Wiring System

Select from the following modes.

Three-wire	The DL6000/DLM6000 searches using the data pattern condition of one data line.
Four-wire	The DL6000/DLM6000 searches using the data pattern conditions of Data 1 and Data 2 lines. You can also use one of the two data lines as a search condition.

Bit Order

You can set the bit order to MSB or LSB based on the data stream.

- If you are setting the data in binary notation, set the pattern in the order of the data stream, regardless of the bit order setting.
- If you are setting the data in hexadecimal notation, set the pattern in 4-bit segments according to the bit order setting.

MSB	When the data stream is MSB first
LSB	When the data stream is LSB first

Field Size

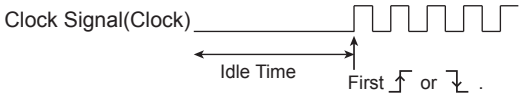
You can set the field size.
It can be set to a value from 4 to 32 bits.

Enabled Bit Range

You can specify the range of bits to enable within the field. Enter the MSB in the box on the left and the LSB in the box on the right.
Only the enabled bits are searched.

Clock Idle Time

If you set the CS source to None, the DL6000/DLM6000 will treat the first rising or falling edge of the clock signal after the specified idle time has elapsed as the data start position. If you select the Don't care check box, search will start from the left of the screen, regardless of the set idle time value.



Data

You can use a data pattern as a search condition.

Comparison Condition

The data search condition is met when the result of comparing the input signal pattern with the specified pattern meets the selected comparison condition.

True	When the patterns match
False	When the patterns don't match

Comparison Start Position

Set the comparison start position. For example, to start comparing from the first data byte after the data start position.

Selectable range: 0 to 9999

Data Size

Set how many consecutive data fields you want to compare.

Selectable range: 1 to 4

Specify how many fields of consecutive data to compare.

If you set the field size to 32 bits and the data length to 4 fields, you can compare 128 bits of data.

Data Pattern

Set the data pattern for the specified size in hexadecimal or binary notation.

- If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
- If a binary pattern contains any Xs, the corresponding hexadecimal display will be "$."

CS, Clock, and Data

You can select the CS (chip select), clock, and data sources.

- On the DL6000, select the sources from None*1, from CH1 to CH4, or from M1 to M4.
- On the DLM6000, select the sources from None*1, from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from None*1, from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).

*1 None can only be selected for CS. If you set CS to None, the DL6000/DLM6000 uses the Idle Time setting to determine the data start position.

CS

You can select the CS level for activating the data.

H	When the signal is high
L	When the signal is low

Clock

You can select the clock edge that specifies when the data patterns are compared.

↑	On the rising edge
↓	On the falling edge

Level

Set the reference level for CH1 to CH4 and M1 to M4.*

The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions. For example, if the T/div setting is 2 V/division, the resolution is 0.02 V.

* If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-0*1EN

Hysteresis

Set the hysteresis for CH1 to CH4 and M1 to M4.

The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.

For the relationship between this hysteresis and the trigger hysteresis, see page 2-7.

Found Point

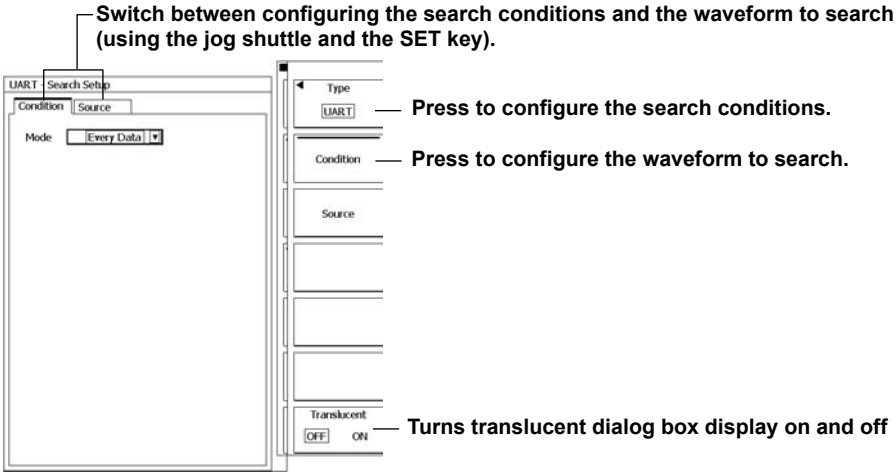
The found-point position is the same as the trigger point position. For a description of the trigger point, see "Explanation" in section 3.4.

5.6 Searching UART Signals

Procedure

Setup Menu

Press **SEARCH**, the **Setup soft** key, the **Type** soft key, the **Serial Bus** soft key, and then the **UART** soft key to display the following menu.

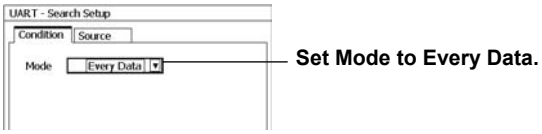


Setting Search Conditions (Condition)

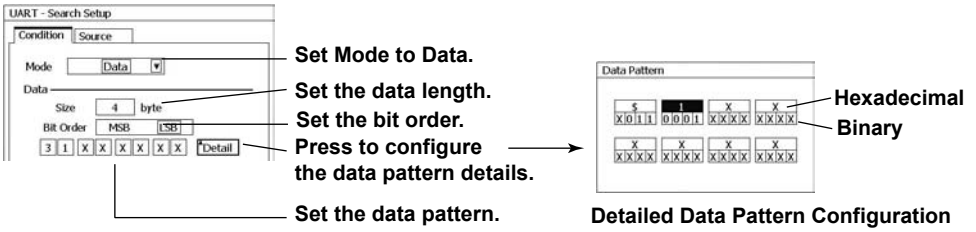
Depending on the address to compare, there are three modes:

- Every Data
- Data
- Error

When Searching in Every Data Mode



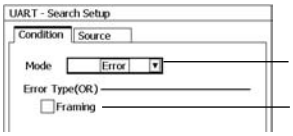
When Searching in Data Mode



Detailed Data Pattern Configuration
Press Detail to display a dialog box, and then use the jog shuttle, the SET key, and the soft keys to set the data pattern.

When Searching in Error Mode

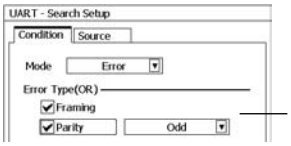
When Format in the Source Settings Is Set to



Set Mode to Error.

Set the error types.

When Format in the Source Settings Is Set to 8bit + Parity or 7bit + Parity

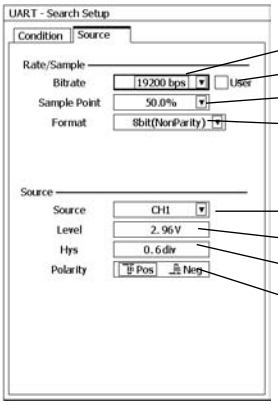


Set the error types.

Setting the Waveform to Search (Source)

Select the **Source** tab, or press the **Source** soft key to display the following menu.

Analog Waveform



Set the bitrate.

Select this checkbox to specify a user-defined bitrate.

Set the sample point.

Set the format of the analysis data (8bit + Parity, 7bit + Parity, 8bit(NonParity)).

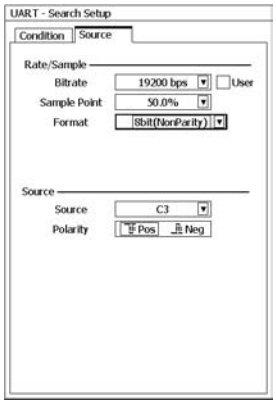
Set the waveform to search (CH1–CH4, M1–M4, A0–D7).

Set the level (when Source is set to CH1–CH4 or M1–M4).

Set the hysteresis (when Source is set to CH1–CH4 or M1–M4).

Set the polarity.

Logic Waveform



Executing the Search

To execute the search, follow the instructions given in section 5.1.

Explanation

This feature searches UART signals. For details on the UART signal data format, see “Explanation” in section 3.5.

Modes

Set the UART search mode to Every Data, Data, and Error.

Every Data Mode

The DL6000/DLM6000 searches for the stop bit of all data frames.

Data Mode

Searches for a data pattern.

- Data Size
  Set how many consecutive data bytes you want to compare.
  Selectable range: 1 to 4
- Bit Order
  Select the bit order used to read the data pattern when comparing the input signal data pattern to the specified data pattern.

MSB	Reads the data pattern MSB first.
LSB	Reads the data pattern LSB first.

- Data Pattern
  Set the data pattern for the specified size in hexadecimal or binary notation.
  - If you specify X, the condition is assumed to be met regardless of the corresponding bit status.
  - If a binary pattern contains any Xs, the corresponding hexadecimal display will be “$.”

Error Mode

Searches for points where errors occurred.

You can select the type of errors to detect from the table below.

- You can select multiple error types.
- The DL6000/DLM6000 will search for all selected errors.

Framing	Searches for a position where the logic value of the stop bit is zero.
Parity	When the DL6000/DLM6000 detects a parity error in a received character, the DL6000/DLM6000 searches for the stop bit position. • You can select which parity to check, odd or even. • Errors will not occur if the parity bit is set to none.

Bit Rate, Sample Point, Format, Level, Hysteresis, and Polarity

Bit Rate

You can set the UART signal transfer rate to
115200 bps, 57600 bps, 38400 bps, 19200 bps, 9600 bps, 4800 bps, 2400 bps, or 1200 bps
If you select the User check box, you can set the transfer rate from 1000 bps to 200000 bps in 100-bps steps.

Sample Point

You can set the point for determining the signal level from 18.8 to 90.6% in 3.1% steps.
The DL6000/DLM6000 UART signal trigger circuit samples the input UART signal using the internal clock and detects the point of level change. Taking the detected point of change to be 0% and the point that is bit time after the point of change to be 100%, you set the sample point in percentage. The bit time is the reciprocal of the set bit rate. See the illustration on page 3-19.

Format

You can select the format from the following:

8bit + Parity	8-bit data + parity bit
7bit + Parity	7-bit data + parity bit
8bit(NonParity)	8-bit data with no parity bit

Source

You can select the source.

- On the DL6000, select the source from CH1 to CH4 or from M1 to M4.
- On the DLM6000, select the source from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).
- **Level**
  Set the level for determining whether the signal level is 0 or 1.*
  The selectable range is ±10 divisions from the vertical position. The resolution is 0.01 divisions.
  For example, if the T/div setting is 2 mV/division, the resolution is 0.02 mV.
  * If the source is set to a logic signal from A0 to D7, the level is the threshold level that you set according to the instructions in section 5.2 of the *User's Manual IM DLM6054-0*1EN
- **Hysteresis**
  The selectable range is from 0.0 to 4.0 divisions. The resolution is 0.1 divisions.
  Trigger hysteresis settings  and  correspond to 0.6 divisions and 1.0 division.
- **Polarity**
  You can select the bit state that will be considered logical 1.

Pos	Positive logic
Neg	Negative logic

Found Point

The found-point position is the same as the trigger point position. See "Explanation" in section 3.5 for details.

6.1 Messages

Messages may appear on the screen during operation, and this section explains these messages, and corrective action to take. This section contains a list of messages that only pertain to the serial bus signal analysis and search features. There are other messages that pertain to the digital oscilloscope itself and communications. For a description of these messages, see the respective user's manual that is listed on page i.

You can select the message language. For instructions on how to select the message language, see section 16.1 in the *User's Manual IM DLM6054-01EN*.
If the corrective action requires servicing, contact your nearest YOKOGAWA dealer for repairs.

Code	Message	Corrective Action	Section
58	Search execution is completed, but no record was found that matched the pattern.	–	Chapter 5
69	Any serial bus signal can not be detected.	–	2.1
70	Serial bus automatic setting was aborted.	–	2.1
73	Check the input voltage level and attenuation ratio.	–	2.1
506	Save data do not exist. Check the content to be saved.	Display the analysis result, and then execute save operation again.	4.1
670	The corresponding field was not found.	–	–
675	Serial bus automatic setting is in progress. Please wait.	–	–

7.1 List of Commands

Command	Function	Page
ANALysis Group		
:ANALysis:LSBus<x>?	Queries all settings related to the logic serial bus signal feature.	7-24
:ANALysis:LSBus<x>[:ANALyze]?	Queries all settings related to the logic serial bus signal.	7-24
:ANALysis:LSBus<x>[:ANALyze]:I2CBus?	Queries all settings related to the logic I^2C bus signal analysis.	7-24
:ANALysis:LSBus<x>[:ANALyze]:I2CBus:CLOCk	Sets the clock channel of the logic I^2C bus signal analysis or queries the current setting.	7-24
:ANALysis:LSBus<x>[:ANALyze]:I2CBus:DTRace	Sets the data channel of the logic I^2C bus signal analysis or queries the current setting.	7-24
:ANALysis:LSBus<x>[:ANALyze]:LINBus?	Queries all settings related to the logic LIN bus signal analysis.	7-24
:ANALysis:LSBus<x>[:ANALyze]:LINBus:BRATe	Sets the bit rate (data transfer rate) of the logic LIN bus signal analysis or queries the current setting.	7-24
:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:BREak	Executes a field jump to the Break Field in the results of the logic LIN bus signal analysis.	7-24
:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:CSUM	Executes a field jump to the Checksum Field in the results of the logic LIN bus signal analysis.	7-24
:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:DATA	Executes a field jump to the Data Field in the results of the logic LIN bus signal analysis.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:IDENtifier	Executes a field jump to the Identifier Field in the results of the logic LIN bus signal analysis.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:SYNCh	Executes a field jump to the Synch Field in the results of the logic LIN bus signal analysis.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LINBus:REVision	Sets the revision (1.3, 2.0, or Both) of the logic LIN bus signal analysis or queries the current setting.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LINBus:SPOint	Sets the logic LIN bus signal analysis sample point or queries the current setting.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LINBus:TRACe	Sets the trace of the logic LIN bus signal analysis or queries the current setting.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LIST?	Queries all settings related to the analysis result list of the logic serial bus signal analysis.	7-25
:ANALysis:LSBus<x>[:ANALyze]:LIST:DISPlay	Turns ON/OFF the analysis result list of the logic serial bus signal analysis or queries the current setting.	7-26
:ANALysis:LSBus<x>[:ANALyze]:LIST:ITEM?	Queries all items displayed on the analysis result list of the logic serial bus signal analysis.	7-26
:ANALysis:LSBus<x>[:ANALyze]:LIST:MODE	Sets the mode of the analysis result list of the logic serial bus signal analysis or queries the current setting.	7-26
:ANALysis:LSBus<x>[:ANALyze]:LIST:SCRoll	Sets the scroll method of the analysis result list of the logic serial bus signal analysis or queries the current setting.	7-26
:ANALysis:LSBus<x>[:ANALyze]:LIST:VALue?	Queries the automated measured value of the specified analysis number in the analysis result list of the logic serial bus signal analysis.	7-26
:ANALysis:LSBus<x>[:ANALyze]:MODE	Sets the logic serial bus signal analysis mode or queries the current setting.	7-26
:ANALysis:LSBus<x>[:ANALyze]:RPOint	Sets the analysis reference point of the logic serial bus signal analysis or queries the current setting.	7-27
:ANALysis:LSBus<x>[:ANALyze]:SPIBus?	Queries all settings related to the logic SPI bus signal analysis.	7-27
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CLOCk?	Queries all settings related to the clock signal channel of the logic SPI bus signal analysis.	7-27
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CLOCk:POLarity	Sets the polarity of the clock signal channel of the logic SPI bus signal analysis or queries the current setting.	7-27
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CLOCk:SOURce	Sets the clock signal channel of the logic SPI bus signal analysis or queries the current setting.	7-27
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CS?	Queries all settings related to the chip select signal channel of the logic SPI bus signal analysis.	7-27
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CS:ACTive	Sets the active level of the chip select signal channel of the logic SPI bus signal analysis or queries the current setting.	7-27

Command	Function	Page
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CS:TRACe	Sets the chip select signal channel of the logic SPI bus signal analysis or queries the current setting.	7-28
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:DATA<x>?	Queries all settings related to each data of the logic SPI bus signal analysis.	7-28
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:DATA<x>:ACTive	Sets the active level of each data of the logic SPI bus signal analysis or queries the current setting.	7-28
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:DATA<x>:TRACe	Sets the data channel of the logic SPI bus signal analysis or queries the current setting.	7-28
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]?	Queries all settings related to the setup of the logic SPI bus signal analysis.	7-28
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:BITorder	Sets the bit order of the logic SPI bus signal analysis or queries the current setting.	7-28
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:EMSBLSB	Sets the enabled range of the field used for logic SPI bus signal analysis or queries the current setting.	7-29
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:FSIZe	Sets the field size used for logic SPI bus signal analysis or queries the current setting.	7-29
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:ITIMe	Sets the idle time used in logic SPI bus signal analysis or queries the current setting.	7-29
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:MODE	Sets the wiring system of the logic SPI bus signal analysis (three-wire or four-wire) or queries the current setting.	7-29
:ANALysis:LSBus<x>[:ANALyze]:UART?	Queries all settings related to the logic UART bus signal analysis.	7-29
:ANALysis:LSBus<x>[:ANALyze]:UART:BITorder	Sets the logic UART bus signal analysis bit order or queries the current setting.	7-30
:ANALysis:LSBus<x>[:ANALyze]:UART:BRATe	Sets the logic UART bus signal analysis bit rate (data transfer rate) or queries the current setting.	7-30
:ANALysis:LSBus<x>[:ANALyze]:UART:BSPace	Sets the byte space for grouping data that is used in logic UART signal analysis or queries the current setting.	7-30
:ANALysis:LSBus<x>[:ANALyze]:UART:DFORmat	Sets the decoded character display format for logic UART signal analysis or queries the current setting.	7-30
:ANALysis:LSBus<x>[:ANALyze]:UART:FORMat	Sets the logic UART bus signal analysis data format or queries the current setting.	7-30
:ANALysis:LSBus<x>[:ANALyze]:UART:GROuping	Turns on or off the grouping feature for logic UART signal analysis or queries the current setting.	7-30
:ANALysis:LSBus<x>[:ANALyze]:UART:PMODe	Sets the logic UART bus signal analysis parity mode or queries the current setting.	7-31
:ANALysis:LSBus<x>[:ANALyze]:UART:POLarity	Sets the logic UART bus signal analysis parity or queries the current setting.	7-31
:ANALysis:LSBus<x>[:ANALyze]:UART:SPOint	Sets the logic UART bus signal analysis sample point or queries the current setting.	7-31
:ANALysis:LSBus<x>[:ANALyze]:UART:TRACe	Sets the logic UART bus signal analysis trace or queries the current setting.	7-31
:ANALysis:LSBus<x>:ZLINkage	Sets the zoom link of the logic serial bus signal analysis or queries the current setting.	7-31
:ANALysis:SBUS<x>?	Queries all settings related to the serial bus signal analysis feature.	7-32
:ANALysis:SBUS<x>:ANALyze?	Queries all settings related to the serial bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus?	Queries all settings related to the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:BRATe	Sets the bit rate (data transfer rate) of the CAN bus signal analysis or queries the current setting.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:ACK	Executes a field jump to the ACK Field in the results of the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:CONTrol	Executes a field jump to the Control Field in the results of the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:CRC	Executes a field jump to the CRC Field in the results of the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:DATA	Executes a field jump to the Data Field in the results of the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:IDENtifier	Executes a field jump to the Identifier Field in the results of the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:SOF	Executes a field jump to the SOF Field in the results of the CAN bus signal analysis.	7-32
:ANALysis:SBUS<x>[:ANALyze]:CANBus:RECessive	Sets the recessive level (bus level) of the CAN bus signal analysis or queries the current setting.	7-33

Command	Function	Page
:ANALysis:SBUS<x>[:ANALyze]:CANBus:SPOint	Sets the sample point of the CAN bus signal analysis or queries the current setting.	7-33
:ANALysis:SBUS<x>[:ANALyze]:CANBus:TRACe	Sets the trace of the CAN bus signal analysis or queries the current setting.	7-33
:ANALysis:SBUS<x>[:ANALyze]:DECode	Turns the serial bus signal analysis decoding display ON/OFF or queries the current status.	7-33
:ANALysis:SBUS<x>[:ANALyze]:I2CBus?	Queries all settings related to the I^2C bus signal analysis.	7-33
:ANALysis:SBUS<x>[:ANALyze]:I2CBus:CLOCk	Sets the clock channel of the I^2C bus signal analysis or queries the current setting.	7-33
:ANALysis:SBUS<x>[:ANALyze]:I2CBus:DTRace	Sets the data channel of the I^2C bus signal analysis or queries the current setting.	7-33
:ANALysis:SBUS<x>[:ANALyze]:LINBus?	Queries all settings related to the LIN bus signal analysis.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:BRATe	Sets the LIN bus signal analysis bitrate (data transfer rate) or queries the current setting.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:BREak	Executes a field jump to the Break Field in the results of the LIN bus signal analysis.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:CSUM	Executes a field jump to the Checksum Field in the results of the LIN bus signal analysis.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:DATA	Executes a field jump to the Data Field in the results of the LIN bus signal analysis.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:IDENtifier	Executes a field jump to the Identifier Field in the results of the LIN bus signal analysis.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:SYNCh	Executes a field jump to the Synch Field in the results of the LIN bus signal analysis.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:REVision	Sets the LIN bus signal analysis revision (1.3, 2.0, or Both) or queries the current setting.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:SPOint	Sets the LIN bus signal analysis sample point or queries the current setting.	7-34
:ANALysis:SBUS<x>[:ANALyze]:LINBus:TRACe	Sets the LIN bus signal analysis trace or queries the current setting.	7-35
:ANALysis:SBUS<x>[:ANALyze]:LIST?	Queries all settings related to the list display of the serial bus signal analysis.	7-35
:ANALysis:SBUS<x>[:ANALyze]:LIST:DISPlay	Turns the serial bus signal analysis list display ON/OFF or queries the current status.	7-35
:ANALysis:SBUS<x>[:ANALyze]:LIST:ITEM?	Queries the item in the list display of the serial bus signal analysis.	7-35
:ANALysis:SBUS<x>[:ANALyze]:LIST:MODE	Sets the mode of the list display of the serial bus signal analysis or queries the current setting.	7-35
:ANALysis:SBUS<x>[:ANALyze]:LIST:SCRoll	Sets the scroll method of the list display of the serial bus signal analysis or queries the current setting.	7-35
ANALysis:SBUS<x>[:ANALyze]:LIST:VALue?	Queries the automated measured value of the specified analysis number in the analysis result list of the serial bus signal analysis.	7-35
:ANALysis:SBUS<x>[:ANALyze]:MODE	Sets the serial bus signal analysis mode or queries the current setting.	7-36
:ANALysis:SBUS<x>[:ANALyze]:RPOint	Sets the analysis reference point of the serial bus signal analysis or queries the current setting.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus?	Queries all settings related to the SPI bus signal analysis.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CLOCk?	Queries all settings related to the clock channel of the SPI bus signal analysis.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CLOCk:POLarity	Sets the polarity of the clock channel of the SPI bus signal analysis or queries the current setting.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CLOCk:SOURce	Sets the clock channel of the SPI bus signal analysis or queries the current setting.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS?	Queries all settings related to the chip select channel of the SPI bus signal analysis.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS:ACTive	Sets the active level of the chip select channel of the SPI bus signal analysis or queries the current setting.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS:TRACe	Sets the chip select channel of the SPI bus signal analysis or queries the current setting.	7-36
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>?	Queries all settings related to the data of the SPI bus signal analysis.	7-37

Command	Function	Page
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:ACTive	Sets the active level of the data of the SPI bus signal analysis or queries the current setting.	7-37
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:TRACe	Sets the data channel of the SPI bus signal analysis or queries the current setting.	7-37
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:SETup?	Queries all settings related to the SPI bus signal analysis setup.	7-37
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:BITorder	Sets the bit order of the SPI bus signal analysis or queries the current setting.	7-37
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:EMSBLSB	Sets the enabled range of the field used for SPI bus signal analysis or queries the current setting.	7-37
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:FSIZe	Sets the field size used for SPI bus signal analysis or queries the current setting.	7-37
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:ITIMe	Sets the idle time used in SPI bus signal analysis or queries the current setting.	7-38
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:MODE	Sets the wiring system of the SPI bus signal analysis (three-wire or four-wire) or queries the current setting.	7-38
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>?	Queries all settings related to the threshold level of the source channel of the serial bus signal analysis.	7-38
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:HYSTeresis	Sets the hysteresis of the threshold level of the source channel of the serial bus signal analysis or queries the current setting.	7-38
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:LEVel	Sets the level of the threshold level of the source channel of the serial bus signal analysis or queries the current setting.	7-38
:ANALysis:SBUS<x>[:ANALyze]:UART?	Queries all settings related to the UART bus signal analysis.	7-38
:ANALysis:SBUS<x>[:ANALyze]:UART:BITorder	Sets the UART bus signal analysis bit order or queries the current setting.	7-39
:ANALysis:SBUS<x>[:ANALyze]:UART:BRATe	Sets the UART bus signal analysis bit rate (data transfer rate) or queries the current setting.	7-39
:ANALysis:SBUS<x>[:ANALyze]:UART:BSPace	Sets the byte space for grouping data that is used in UART signal analysis or queries the current setting.	7-39
:ANALysis:SBUS<x>[:ANALyze]:UART:DFORmat	Sets the decoded character display format for UART signal analysis or queries the current setting.	7-39
:ANALysis:SBUS<x>[:ANALyze]:UART:FORMat	Sets the UART bus signal analysis data format or queries the current setting.	7-39
:ANALysis:SBUS<x>[:ANALyze]:UART:GROuping	Turns on or off the grouping feature for UART signal analysis or queries the current setting.	7-39
:ANALysis:SBUS<x>[:ANALyze]:UART:PMODe	Sets the UART bus signal analysis parity mode or queries the current setting.	7-40
:ANALysis:SBUS<x>[:ANALyze]:UART:POLarity	Sets the UART bus signal analysis polarity or queries the current setting.	7-40
:ANALysis:SBUS<x>[:ANALyze]:UART:SPOint	Sets the UART bus signal analysis sample point or queries the current setting.	7-40
:ANALysis:SBUS<x>[:ANALyze]:UART:TRACe	Sets the UART bus signal analysis trace or queries the current setting.	7-40
:ANALysis:SBUS<x>:ZLINkage	Sets the zoom link of the serial bus signal analysis or queries the current setting.	7-40
:ANALysis:TYPE<x>	Sets the analysis feature type or queries the current setting.	7-40

Command	Function	Page
SEARch Group		
SEARch<x>:CANBus?	Queries all settings related to the CAN bus signal search.	7-41
:SEARch<x>:CANBus:SETup?	Queries all settings related to the CAN bus signal search setup.	7-41
:SEARch<x>:CANBus[:SETup]:ACK	Sets the ACK condition of the CAN bus signal search or queries the current setting.	7-41
:SEARch<x>:CANBus[:SETup]:BRATe	Sets the bit rate (data transfer rate) of the CAN bus signal search or queries the current setting.	7-41
:SEARch<x>:CANBus[:SETup]:DATA?	Queries all settings related to the CAN bus signal search data.	7-41
:SEARch<x>:CANBus[:SETup]:DATA:BORDer	Sets the byte order of the CAN bus signal search data or queries the current setting.	7-41
:SEARch<x>:CANBus[:SETup]:DATA:CONDition	Sets the data condition of the CAN bus signal search or queries the current setting.	7-41
:SEARch<x>:CANBus[:SETup]:DATA:DATA<x>	Sets the comparison data of the CAN bus signal search data or queries the current setting.	7-42
:SEARch<x>:CANBus[:SETup]:DATA:DLC	Sets the number of valid bytes (DLC) of the CAN bus signal search data or queries the current setting.	7-42
:SEARch<x>:CANBus[:SETup]:DATA:HEXA	Sets the CAN bus signal search data in hexadecimal notation.	7-42
:SEARch<x>:CANBus[:SETup]:DATA:MSBLsb	Sets the MSB and LSB bits of the CAN bus signal search data or queries the current setting.	7-42
:SEARch<x>:CANBus[:SETup]:DATA:PATTern	Sets the CAN bus signal search data in binary notation or queries the current setting.	7-42
:SEARch<x>:CANBus[:SETup]:DATA:SIGN	Sets the sign of the CAN bus signal search data or queries the current setting.	7-42
:SEARch<x>:CANBus[:SETup]:IDEXt?	Queries all settings related to the ID of the extended format of the CAN bus signal search.	7-43
:SEARch<x>:CANBus[:SETup]:IDEXt:HEXA	Sets the ID of the extended format of the CAN bus signal search in hexadecimal notation.	7-43
:SEARch<x>:CANBus[:SETup]:IDEXt:PATTern	Sets the ID of the extended format of the CAN bus signal search in binary notation or queries the current setting.	7-43
:SEARch<x>:CANBus[:SETup]:IDSTd?	Queries all settings related to the ID of the standard format of the CAN bus signal search.	7-43
:SEARch<x>:CANBus[:SETup]:IDSTd:HEXA	Sets the ID of the standard format of the CAN bus signal search in hexadecimal notation.	7-43
:SEARch<x>:CANBus[:SETup]:IDSTd:PATTern	Sets the ID of the standard format of the CAN bus signal search in binary notation or queries the current setting.	7-43
:SEARch<x>:CANBus[:SETup]:MODE	Sets the CAN bus signal search mode or queries the current setting.	7-43
:SEARch<x>:CANBus[:SETup]:RECessive	Sets the recessive level (bus level) of the CAN bus signal search or queries the current setting.	7-43
:SEARch<x>:CANBus[:SETup]:RTR	Sets the RTR of the CAN bus signal search or queries the current setting.	7-43
:SEARch<x>:CANBus[:SETup]:SPOint	Sets the sample point of the CAN bus signal search or queries the current setting.	7-44
:SEARch<x>:CANBus[:SETup]:TRACe	Sets the trace of the CAN bus signal search or queries the current setting.	7-44
SEARch<x>:I2CBus?	Queries all settings related to the I^2C bus signal search.	7-44
:SEARch<x>:I2CBus:CLOCk?	Queries all settings related to the clock of the I^2C bus signal search.	7-44
:SEARch<x>:I2CBus:CLOCk:SOURce	Sets the clock trace of the I^2C bus signal search or queries the current setting.	7-44
:SEARch<x>:I2CBus:SETup?	Queries all settings related to the I^2C bus signal search setup.	7-44
:SEARch<x>:I2CBus[:SETup]:ADATa?	Queries all settings related to the address of the I^2C bus signal search.	7-44
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT10address?	Queries all settings related to the 10-bit address of the I^2C bus signal search.	7-44
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT10address:HEXA	Sets the 10-bit address of the I^2C bus signal search in hexadecimal notation.	7-44
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT10address:PATTern	Sets the 10-bit address of the I^2C bus signal search in binary notation or queries the current setting.	7-44
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress?	Queries all settings related to the 7-bit address of the I^2C bus signal search.	7-44
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:HEXA	Sets the 7-bit address of the I^2C bus signal search in hexadecimal notation.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:PATTern	Sets the 7-bit address of the I^2C bus signal search in binary notation or queries the current setting.	7-45

Command	Function	Page
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub?	Queries all settings related to the 7-bit + Sub address of the I²C bus signal search.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + Sub address of the I²C bus signal search.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:HEXA	Sets the 7-bit address of the 7-bit + Sub address of the I²C bus signal search in hexadecimal notation.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:PATTern	Sets the 7-bit address of the 7-bit + Sub address of the I²C bus signal search in binary notation or queries the current setting.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress?	Queries all settings related to the Sub address of the 7-bit + Sub address of the I²C bus signal search.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:HEXA	Sets the Sub address of the 7-bit + Sub address of the I²C bus signal search in hexadecimal notation.	7-45
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:PATTern	Sets the Sub address of the 7-bit + Sub address of the I²C bus signal search in binary notation or queries the current setting.	7-46
:SEARch<x>:I2CBus[:SETup]:ADATa:TYPE	Sets the address type of the I²C bus signal search or queries the current setting.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA?	Queries all settings related to the data of the I²C bus signal search.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA:BYTE	Sets the number of data bytes of the I²C bus signal search or queries the current setting.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA:CONDition	Sets the determination method (match or not match) of the data of the I²C bus signal search or queries the current setting.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA:DPOSition	Sets the position for comparing the data pattern of the I²C bus signal search or queries the current setting.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA:HEXA<x>	Sets the data of the I²C bus signal search in hexadecimal notation.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA:MODE	Enables/Disables the data conditions of the I²C bus signal search or queries the current setting.	7-46
:SEARch<x>:I2CBus[:SETup]:DATA:PATTern<x>	Sets the data of the I²C bus signal search in binary notation or queries the current setting.	7-47
:SEARch<x>:I2CBus[:SETup]:DATA:PMODe	Sets the pattern comparison start position mode of the data of the I²C bus signal search or queries the current setting.	7-47
:SEARch<x>:I2CBus[:SETup]:DATA:TRACe	Sets the trace of the data of the I²C bus signal search or queries the current setting.	7-47
:SEARch<x>:I2CBus[:SETup]:GCALl?	Queries all settings related to the general call of the I²C bus signal search.	7-47
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress?	Queries all settings related to the 7-bit master address of the general call of the I²C bus signal search.	7-47
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:HEXA	Sets the 7-bit master address of the general call of the I²C bus signal search in hexadecimal notation.	7-47
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:PATTern	Sets the 7-bit master address of the general call of the I²C bus signal search in binary notation or queries the current setting.	7-47
:SEARch<x>:I2CBus[:SETup]:GCALl:SBYTe (Second Byte)	Sets the second byte type of the general call of the I²C bus signal search or queries the current setting.	7-47
:SEARch<x>:I2CBus[:SETup]:MODE	Sets the search mode of the I²C bus signal search or queries the current setting.	7-48
:SEARch<x>:I2CBus[:SETup]:NAIGnore?	Queries all settings related to the NON ACK ignore mode of the I²C bus signal search.	7-48
:SEARch<x>:I2CBus[:SETup]:NAIGnore:HSMode	Sets whether to ignore NON ACK in high speed mode of the I²C bus signal search or queries the current setting.	7-48
:SEARch<x>:I2CBus[:SETup]:NAIGnore:RACCess	Sets whether to ignore NON ACK in read access mode of the I²C bus signal search or queries the current setting.	7-48
:SEARch<x>:I2CBus[:SETup]:NAIGnore:SBYTe(Start Byte)	Sets whether to ignore NON ACK in the start byte of the I²C bus signal search or queries the current setting.	7-48
:SEARch<x>:I2CBus[:SETup]:SBHSmode?	Queries all settings related to the start byte and high speed mode of the I²C bus signal search.	7-48
:SEARch<x>:I2CBus[:SETup]:SBHSmode:TYPE	Sets the type of the start byte or high speed mode of the I²C bus signal search or queries the current setting.	7-48
:SEARch<x>:LINBus?	Queries all settings related to the LIN bus signal search or queries the current setting.	7-48
:SEARch<x>:LINBus[:SETup]?	Queries all settings related to setup of the LIN bus signal search or queries the current setting.	7-48
:SEARch<x>:LINBus[:SETup]:BLENgth	Sets the LIN bus signal search break length or queries the current setting.	7-49

Command	Function	Page
:SEARch<x>:LINBus[:SETup]:BRATe	Sets the LIN bus signal search bitrate (data transfer rate) or queries the current setting.	7-49
:SEARch<x>:LINBus[:SETup]:DATA?	Queries all settings related to data of the LIN bus signal search or queries the current setting.	7-49
:SEARch<x>:LINBus[:SETup]:DATA:BNUM	Sets the number of LIN bus signal search data bytes or queries the current setting.	7-49
:SEARch<x>:LINBus[:SETup]:DATA:BORDer	Sets the data byte order of the LIN bus signal search or queries the current setting.	7-49
:SEARch<x>:LINBus[:SETup]:DATA:CONDition	Sets the LIN bus signal search data or queries the current setting.	7-49
:SEARch<x>:LINBus[:SETup]:DATA:DATA<x>	Sets the comparison data of the LIN bus signal search data or queries the current setting.	7-49
:SEARch<x>:LINBus[:SETup]:DATA:HEXA	Sets the LIN bus signal search data in hexadecimal.	7-49
:SEARch<x>:LINBus[:SETup]:DATA:MSBLsb	Sets the MSB/LSB bit of the LIN bus signal search or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:DATA:PATTern	Sets the LIN bus signal search data in binary or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:DATA:SIGN	Sets the sign order of the LIN bus signal search or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:ERRor?	Queries all settings related to the LIN bus signal search error.	7-50
:SEARch<x>:LINBus[:SETup]:ERRor:CHECksum	Sets the LIN bus signal search Checksum error or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:ERRor:FRAMing	Sets the LIN bus signal search Framing error or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:ERRor:PARity	Sets the LIN bus signal search Parity error or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:ERRor:SYNCh	Sets the LIN bus signal search Synch error or queries the current setting.	7-50
:SEARch<x>:LINBus[:SETup]:ERRor:TOUT	Sets the LIN bus signal search Timeout error or queries the current setting.	7-51
:SEARch<x>:LINBus[:SETup]:ID?	Queries all settings related to ID of the LIN bus signal search or queries the current setting.	7-51
:SEARch<x>:LINBus[:SETup]:ID:HEXA	Sets the LIN bus signal search ID in hexadecimal.	7-51
:SEARch<x>:LINBus[:SETup]:ID:PATTern	Sets the LIN bus signal search ID in binary or queries the current setting.	7-51
:SEARch<x>:LINBus[:SETup]:MODE	Sets the LIN bus signal search mode or queries the current setting.	7-51
:SEARch<x>:LINBus[:SETup]:REVision	Sets the LIN bus signal search revision (1.3, 2.0, or Both) or queries the current setting.	7-51
:SEARch<x>:LINBus[:SETup]:SPOint	Sets the LIN bus signal search sampling point or queries the current setting.	7-51
:SEARch<x>:LINBus[:SETup]:TRACe	Sets the LIN bus signal search trace or queries the current setting.	7-51
:SEARch<x>:SLOGic:I2CBus?	Queries all settings related to the logic I^2C bus signal search.	7-52
:SEARch<x>:SLOGic:I2CBus:CLOCk?	Queries all settings related to the clock channel of the logic I^2C bus signal search.	7-52
:SEARch<x>:SLOGic:I2CBus:CLOCk:SOURce	Sets the clock channel of the logic I^2C bus signal search or queries the current setting.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]?	Queries all settings related to the setup of the logic I^2C bus signal search.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa?	Queries all settings related to the address of the logic I^2C bus signal search.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT10address?	Queries all settings related to the 10-bit address of the logic I^2C bus signal search.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT10address:HEXA	Sets the 10-bit address of the logic I^2C bus signal search in hexadecimal notation.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT10address:PATTern	Sets the 10-bit address of the logic I^2C bus signal search in binary notation or queries the current setting.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7ADdress?	Queries all settings related to the 7-bit address of the logic I^2C bus signal search.	7-52
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7ADdress:HEXA	Sets the 7-bit address of the logic I^2C bus signal search in hexadecimal notation.	7-53

Command	Function	Page
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7ADdress:PATTern	Sets the 7-bit address of the logic I²C bus signal search in binary notation or queries the current setting.	7-53
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub?	Queries all settings related to the 7-bit address + Sub address of the logic I²C bus signal search.	7-53
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit address + Sub address of the logic I²C bus signal search.	7-53
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:HEXA	Queries all settings related to the 7-bit address of the 7-bit address + Sub address of the logic I²C bus signal search.	7-53
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:PATTern	Sets the 7-bit address of the 7-bit address + Sub address of the logic I²C bus signal search in binary notation or queries the current setting.	7-53
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress?	Queries all settings related to the Sub address of the 7-bit address + Sub address of the logic I²C bus signal search.	7-53
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:HEXA	Queries all settings related to the Sub address of the 7-bit address + Sub address of the logic I²C bus signal search.	7-54
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:PATTern	Sets the Sub address of the 7-bit address + Sub address of the logic I²C bus signal search in binary notation or queries the current setting.	7-54
:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:TYPE	Sets the address type of the logic I²C bus signal search or queries the current setting.	7-54
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA?	Queries all settings related to the data of the logic I²C bus signal search.	7-54
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:BYTE	Sets the number of setup data bytes of the logic I²C bus signal search or queries the current setting.	7-54
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:CONDition	Sets the determination method (match or not match) of the data of the logic I²C bus signal search or queries the current setting.	7-54
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:DPOSition	Sets the position for comparing the data pattern of the logic I²C bus signal search or queries the current setting.	7-55
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:HEXA<x>	Sets the data of the logic I²C bus signal search in hexadecimal notation.	7-55
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:MODE	Enables/disables the data conditions of the logic I²C bus signal search or queries the current setting.	7-55
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PATTern<x>	Sets the data of the logic I²C bus signal search in binary notation or queries the current setting.	7-55
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PMODe	Sets the pattern comparison start position mode of the logic I²C bus signal search or queries the current setting.	7-55
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:TRACe	Sets the data trace of the logic I²C bus signal search or queries the current setting.	7-55
:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl?	Queries all settings related to the general call of the logic I²C bus signal search.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:BIT7maddress?	Queries all settings related to the 7-bit master address of the general code of the logic I²C bus signal search.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:BIT7maddress:HEXA	Sets the 7-bit master address of the general call of the logic I²C bus signal search in hexadecimal notation.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:BIT7maddress:PATTern	Sets the 7-bit master address of the general call of the logic I²C bus signal search in binary notation or queries the current setting.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:SBYTe (Second Byte)	Sets the second byte type of the general call of the logic I²C bus signal search or queries the current setting.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:MODE	Sets the search mode of the logic I²C bus signal search or queries the current setting.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore?	Queries all settings related to the NON ACK ignore mode of the logic I²C bus signal search.	7-56
:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore:HSMode	Sets whether to ignore NON ACK in high speed mode of the logic I²C bus signal search or queries the current setting.	7-57
:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore:RACCess	Sets whether to ignore NON ACK in read access mode of the logic I²C bus signal search or queries the current setting.	7-57
:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore:SBYTe (Start Byte)	Sets whether to ignore NON ACK in the start byte of the I²C bus trigger or queries the current setting.	7-57

Command	Function	Page
:SEARch<x>:SLOGic:I2CBus[:SETup]:SBHSmode?	Queries all settings related to the start byte and high speed mode of the logic I^2C bus signal search.	7-57
:SEARch<x>:SLOGic:I2CBus[:SETup]:SBHSmode:TYPE	Sets the type of start byte and high speed mode of the logic I^2C bus signal search or queries the current setting.	7-57
:SEARch<x>:SLOGic:LINBus?	Queries all settings related to the logic LIN bus signal search.	7-57
:SEARch<x>:SLOGic:LINBus[:SETup]?	Queries all settings related to the setup of the logic LIN bus signal search.	7-57
:SEARch<x>:SLOGic:LINBus[:SETup]:BLENgth	Sets the logic LIN bus signal search break length or queries the current setting.	7-58
:SEARch<x>:SLOGic:LINBus[:SETup]:BRATe	Sets the bit rate (data transfer rate) of the logic LIN bus signal search or queries the current setting.	7-58
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA?	Queries all settings related to the data of the logic LIN bus signal search.	7-58
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:BNUM	Sets the number of bytes of the logic LIN bus signal search or queries the current setting.	7-58
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:BORDer	Sets the data byte order of the logic LIN bus signal search or queries the current setting.	7-58
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:CONDition	Sets the data condition of the logic LIN bus signal search or queries the current setting.	7-58
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA<x>	Sets the comparison data of the logic LIN bus signal search data or queries the current setting.	7-59
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:HEXA	Sets the data of the logic LIN bus signal search in hexadecimal notation.	7-59
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:MSBLsb	Sets the MSB/LSB bit of the logic LIN bus signal search or queries the current setting.	7-59
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:PATTern	Sets the data of the logic LIN bus signal search in binary notation or queries the current setting.	7-59
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:SIGN	Sets the data sign of the logic LIN bus signal search or queries the current setting.	7-59
:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor?	Queries all settings related to the logic LIN bus signal search error.	7-60
:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:CHECksum	Sets the logic LIN bus signal search Checksum error or queries the current setting.	7-60
:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:FRAMing	Sets the logic LIN bus signal search Framing error or queries the current setting.	7-60
:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:PARity	Sets the logic LIN bus signal search Parity error or queries the current setting.	7-60
:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:SYNCh	Sets the logic LIN bus signal search Synch error or queries the current setting.	7-60
:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:TOUT	Sets the logic LIN bus signal search Timeout error or queries the current setting.	7-60
:SEARch<x>:SLOGic:LINBus[:SETup]:ID?	Queries all settings related to the ID of the logic LIN bus signal search.	7-61
:SEARch<x>:SLOGic:LINBus[:SETup]:ID:HEXA	Sets the ID of the logic LIN bus signal search in hexadecimal notation.	7-61
:SEARch<x>:SLOGic:LINBus[:SETup]:ID:PATTern	Sets the ID of the logic LIN bus signal search in binary notation or queries the current setting.	7-61
:SEARch<x>:SLOGic:LINBus[:SETup]:MODE	Sets the logic LIN bus signal search mode or queries the current setting.	7-61
:SEARch<x>:SLOGic:LINBus[:SETup]:REVision	Sets the logic LIN bus signal search revision (1.3, 2.0, or Both) or queries the current setting.	7-61
:SEARch<x>:SLOGic:LINBus[:SETup]:SPOint	Sets the logic LIN bus signal search sampling point or queries the current setting.	7-61
:SEARch<x>:SLOGic:LINBus[:SETup]:TRACe	Sets the trace of the logic LIN bus signal search or queries the current setting.	7-61
:SEARch<x>:SLOGic:SPIBus?	Queries all settings related to the logic SPI bus signal search.	7-62
:SEARch<x>:SLOGic:SPIBus:CLOCk?	Queries all settings related to the clock signal channel of the logic SPI bus signal search.	7-62
:SEARch<x>:SLOGic:SPIBus:CLOCk:POLarity	Sets the polarity of the clock signal channel of the logic SPI bus signal search or queries the current setting.	7-62
:SEARch<x>:SLOGic:SPIBus:CLOCk:SOURce	Sets the clock signal channel of the logic SPI bus signal search or queries the current setting.	7-62
:SEARch<x>:SLOGic:SPIBus:CS?	Queries all settings related to the chip select signal channel of the logic SPI bus signal search.	7-62

Command	Function	Page
:SEARch<x>:SLOGic:SPIBus:CS:ACTive	Sets the active level of the chip select signal channel of the logic SPI bus signal search or queries the current setting.	7-62
:SEARch<x>:SLOGic:SPIBus:CS:TRACe	Sets the chip select signal channel of the logic SPI bus signal search or queries the current setting.	7-62
:SEARch<x>:SLOGic:SPIBus[:SETup]?	Queries all settings related to the setup of the logic SPI bus signal search.	7-62
:SEARch<x>:SLOGic:SPIBus[:SETup]:BITorder	Sets the bit order of the logic SPI bus signal search or queries the current setting.	7-62
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>?	Queries all settings related to each data of the logic SPI bus signal search.	7-63
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:BYTE	Sets the data size (in bytes) of each data of the logic SPI bus signal search or queries the current setting.	7-63
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:CONDition	Sets the determination method (match/mismatch) of the data of the logic SPI bus signal search or queries the current setting.	7-63
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:DPOSition	Sets the pattern comparison start position of the logic SPI bus signal search or queries the current setting.	7-63
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:DSIZe	Sets the number of fields in the data used for logic SPI bus signal search or queries the current setting.	7-63
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:HEXA<x>	Sets the data of the logic SPI bus signal search in hexadecimal notation.	7-63
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:PATTern<x>	Sets the data of the logic SPI bus signal search in binary notation or queries the current setting.	7-64
:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:TRACe	Sets the source channel of each data of the logic SPI bus signal search or queries the current setting.	7-64
:SEARch<x>:SLOGic:SPIBus[:SETup]:EMSBLSB	Sets the enabled range of the field used for logic SPI bus signal search or queries the current setting.	7-64
:SEARch<x>:SLOGic:SPIBus[:SETup]:FSIZe	Sets the field size used for logic SPI bus signal search or queries the current setting.	7-64
:SEARch<x>:SLOGic:SPIBus[:SETup]:ITIMe	Sets the idle time used in logic SPI bus signal search or queries the current setting.	7-64
:SEARch<x>:SLOGic:SPIBus[:SETup]:MODE	Sets the wiring system of the logic SPI bus signal search (three-wire or four-wire) or queries the current setting.	7-64
:SEARch<x>:SLOGic:UART?	Queries all settings related to the logic UART bus signal search.	7-65
:SEARch<x>:SLOGic:UART:BRATe	Sets the logic UART bus signal search bit rate (data transfer rate) or queries the current setting.	7-65
:SEARch<x>:SLOGic:UART:DATA?	Queries all settings related to data of the logic UART bus signal search.	7-65
:SEARch<x>:SLOGic:UART:DATA:BITorder	Sets the data bit order of the logic UART bus signal search or queries the current setting.	7-65
:SEARch<x>:SLOGic:UART:DATA:DSIZe	Sets the number of data bytes of the logic UART bus signal search or queries the current setting.	7-65
:SEARch<x>:SLOGic:UART:DATA:HEXA	Sets the logic UART bus signal search data in hexadecimal.	7-65
:SEARch<x>:SLOGic:UART:DATA:PATTern	Sets the data of the logic UART bus signal search in binary or queries the current setting.	7-65
:SEARch<x>:SLOGic:UART:ERRor?	Queries all settings related to the logic UART bus signal search error.	7-65
:SEARch<x>:SLOGic:UART:ERRor:FRAMing	Sets the logic UART bus signal search Framing error or queries the current setting.	7-65
:SEARch<x>:SLOGic:UART:ERRor:PARity	Sets the logic UART bus signal search Parity error or queries the current setting.	7-66
:SEARch<x>:SLOGic:UART:ERRor:PMODe	Sets the logic UART bus signal search Parity mode or queries the current setting.	7-66
:SEARch<x>:SLOGic:UART:FORMat	Sets the logic UART bus signal search format or queries the current setting.	7-66
:SEARch<x>:SLOGic:UART:MODE	Sets the logic UART bus signal search mode or queries the current setting.	7-66
:SEARch<x>:SLOGic:UART:POLarity	Sets the logic UART bus signal search polarity or queries the current setting.	7-66
:SEARch<x>:SLOGic:UART:SPOint	Sets the logic UART bus signal search sampling point or queries the current setting.	7-66
:SEARch<x>:SLOGic:UART:TRACe	Sets the logic UART bus signal search trace or queries the current setting.	7-66
:SEARch<x>:SPIBus?	Queries all settings related to the SPI bus signal search.	7-67
:SEARch<x>:SPIBus:CLOCk	Queries all settings related to the clock channel of the SPI bus signal search.	7-67
:SEARch<x>:SPIBus:CLOCk:POLarity	Sets the polarity of the clock channel of the SPI bus signal search or queries the current setting.	7-67
:SEARch<x>:SPIBus:CLOCk:SOURce	Sets the clock channel of the SPI bus signal search or queries the current setting.	7-67

Command	Function	Page
:SEARch<x>:SPIBus:CS?	Queries all settings related to the chip select channel of the SPI bus signal search.	7-67
:SEARch<x>:SPIBus:CS:ACTive	Sets the active level of the chip select channel of the SPI bus signal search or queries the current setting.	7-67
:SEARch<x>:SPIBus:CS:TRACe	Sets the chip select channel of the SPI bus signal search or queries the current setting.	7-67
:SEARch<x>:SPIBus:SETup?	Queries all settings related to the SPI bus signal search setup.	7-67
:SEARch<x>:SPIBus[:SETup]:BITorder	Sets the bit order of the SPI bus signal search or queries the current setting.	7-67
:SEARch<x>:SPIBus[:SETup]:DATA<x>?	Queries all settings related to the data of the SPI bus signal search.	7-67
:SEARch<x>:SPIBus[:SETup]:DATA<x>:BYTE	Sets the number of bytes of the data of the SPI bus signal search or queries the current setting.	7-67
:SEARch<x>:SPIBus[:SETup]:DATA<x>:CONDition	Sets the determination method (match or not match) of the data of the SPI bus signal search or queries the current setting.	7-68
:SEARch<x>:SPIBus[:SETup]:DATA<x>:DPOSition	Sets the pattern comparison start position of the data of the SPI bus signal search or queries the current setting.	7-68
:SEARch<x>:SPIBus[:SETup]:DATA<x>:DSIZe	Sets the number of fields in the data used for SPI bus signal search or queries the current setting.	7-68
:SEARch<x>:SPIBus[:SETup]:DATA<x>:HEXA<x>	Sets the data of the SPI bus signal search in hexadecimal notation.	7-68
:SEARch<x>:SPIBus[:SETup]:DATA<x>:PATTern<x>	Sets the data of the SPI bus signal search in binary notation or queries the current setting.	7-68
:SEARch<x>:SPIBus[:SETup]:DATA<x>:TRACe	Sets the source channel of the data of the SPI bus signal search or queries the current setting.	7-68
:SEARch<x>:SPIBus[:SETup]:EMSBLSB	Sets the enabled range of the field used for SPI bus signal search or queries the current setting.	7-69
:SEARch<x>:SPIBus[:SETup]:FSIZe	Sets the field size used for SPI bus signal search or queries the current setting.	7-69
:SEARch<x>:SPIBus[:SETup]:ITIMe	Sets the idle time used in SPI bus signal search or queries the current setting.	7-69
:SEARch<x>:SPIBus[:SETup]:MODE	Sets the wiring system of the SPI bus signal search (three-wire or four-wire) or queries the current setting.	7-69
:SEARch<x>:TRACe<x>:LEVel	Sets the threshold level of the trace or queries the current setting.	7-69
:SEARch<x>:TYPE	Sets the search type or queries the current setting.	7-69
:SEARch<x>:UART?	Queries all settings related to the UART bus signal search.	7-69
:SEARch<x>:UART:BRATe	Sets the UART bus signal search bit rate (data transfer rate) or queries the current setting.	7-70
:SEARch<x>:UART:DATA?	Queries all settings related to data of the UART bus signal search	7-70
:SEARch<x>:UART:DATA:BITorder	Sets the data bit order of the UART bus signal search or queries the current setting.	7-70
:SEARch<x>:UART:DATA:DSIZe	Sets the number of data bytes of the UART bus signal search or queries the current setting.	7-70
:SEARch<x>:UART:DATA:HEXA	Sets the UART bus signal search data in hexadecimal.	7-70
:SEARch<x>:UART:DATA:PATTern	Sets the data of the UART bus signal search in binary or queries the current setting.	7-70
:SEARch<x>:UART:ERRor?	Queries all settings related to the UART bus signal search error.	7-70
:SEARch<x>:UART:ERRor:FRAMing	Sets the UART bus signal search Framing error or queries the current setting.	7-70
:SEARch<x>:UART:ERRor:PARity	Sets the UART bus signal search Parity error or queries the current setting.	7-70
:SEARch<x>:UART:ERRor:PMODe	Sets the UART bus signal search Parity mode or queries the current setting.	7-70
:SEARch<x>:UART:FORMat	Sets the UART bus signal search format or queries the current setting.	7-71
:SEARch<x>:UART:MODE	Sets the UART bus signal search mode or queries the current setting.	7-71
:SEARch<x>:UART:POLarity	Sets the UART bus signal search polarity or queries the current setting.	7-71
:SEARch<x>:UART:SPOint	Sets the UART bus signal search sample point or queries the current setting.	7-71
:SEARch<x>:UART:TRACe	Sets the UART bus signal search trace or queries the current setting.	7-71

Command	Function	Page
SERialbus Group		
:SERialbus?	Queries all settings related to the serial bus setup.	7-72
:SERialbus:SETup<x>?	Queries all settings related to each setup of the serial bus setup.	7-72
:SERialbus:SETup<x>:ASETup:ABORt	Cancels auto setup of the serial bus setup.	7-72
:SERialbus:SETup<x>:ASETup:EXECute	Executes auto setup of the serial bus setup.	7-72
:SERialbus:SETup<x>:ASETup:UNDO	Undoes the executed auto setup of the serial bus setup.	7-72
:SERialbus:SETup<x>:CANBus?	Queries all settings related to the CAN bus setup.	7-72
:SERialbus:SETup<x>:CANBus:BRATe	Sets the CAN bus setup bit rate (data transfer rate) or queries the current setting.	7-72
:SERialbus:SETup<x>:CANBus:RECessive	Sets the CAN bus setup recessive level (bus level) or queries the current setting.	7-72
:SERialbus:SETup<x>:CANBus:SPOint	Sets the CAN bus setup sample point or queries the current setting.	7-72
:SERialbus:SETup<x>:CANBus:TRACe	Sets the CAN bus setup trace or queries the current setting.	7-72
:SERialbus:SETup<x>:I2CBus?	Queries all settings related to the I2C bus setup.	7-73
:SERialbus:SETup<x>:I2CBus:CLOCk	Sets the I2C bus setup clock channel or queries the current setting.	7-73
:SERialbus:SETup<x>:I2CBus:DTRace	Sets the I2C bus signal analysis data channel or queries the current setting.	7-73
:SERialbus:SETup<x>:LINBus?	Queries all settings related to the LIN bus setup.	7-73
:SERialbus:SETup<x>:LINBus:BRATe	Sets the LIN bus setup bit rate (data transfer rate) or queries the current setting.	7-73
:SERialbus:SETup<x>:LINBus:REVision	Sets the LIN bus setup revision (1.3, 2.0, or Both) or queries the current setting.	7-73
:SERialbus:SETup<x>:LINBus:SPOint	Sets the LIN bus setup sample point or queries the current setting.	7-73
:SERialbus:SETup<x>:LINBus:TRACe	Sets the LIN bus setup trace or queries the current setting.	7-73
:SERialbus:SETup<x>:SPIBus?	Queries all settings related to the SPI bus setup.	7-73
:SERialbus:SETup<x>:SPIBus:BITorder	Sets the SPI bus setup bit order or queries the current setting.	7-74
:SERialbus:SETup<x>:SPIBus:CLOCk?	Queries all settings related to the channel of the clock signal of the SPI bus setup.	7-74
:SERialbus:SETup<x>:SPIBus:CLOCk:POLarity	Sets the polarity of the channel of the clock signal of the SPI bus setup.	7-74
:SERialbus:SETup<x>:SPIBus:CLOCk:TRACe	Sets the channel of the clock signal of the SPI bus setup or queries the current setting.	7-74
:SERialbus:SETup<x>:SPIBus:CS?	Queries all settings related to the channel of the chip select signal of the SPI bus setup.	7-74
:SERialbus:SETup<x>:SPIBus:CS:ACTive	Sets the active level of the channel of the chip select signal of the SPI bus setup or queries the current setting.	7-74
:SERialbus:SETup<x>:SPIBus:CS:TRACe	Sets the channel of the chip select signal of the SPI bus setup or queries the current setting.	7-74
:SERialbus:SETup<x>:SPIBus:DATA<x>?	Queries all settings related to each data of the SPI bus setup.	7-74
:SERialbus:SETup<x>:SPIBus:DATA<x>:ACTive	Sets the active level of each data of the SPI bus setup or queries the current setting.	7-75
:SERialbus:SETup<x>:SPIBus:DATA<x>:TRACe	Sets each data channel of the SPI bus setup or queries the current setting.	7-75
:SERialbus:SETup<x>:SPIBus:ITIMe	Sets the idle time used in SPI bus setup or queries the current setting.	7-75
:SERialbus:SETup<x>:SPIBus:MODE	Sets the wiring method (3-wire/4-wire) of the SPI bus setup or queries the current setting.	7-75
:SERialbus:SETup<x>:TRACe<x>?	Queries all settings related to each trace.	7-75
:SERialbus:SETup<x>:TRACe<x>:HYSTeresis	Sets the hysteresis of the threshold level of each trace or queries the current setting.	7-75
:SERialbus:SETup<x>:TRACe<x>:LEVel	Sets the threshold level of each trace or queries the current setting.	7-75
:SERialbus:SETup<x>:TYPE	Sets the serial bus setup type or queries the current setting.	7-76

Command	Function	Page
:SERialbus:SETup<x>:UART?	Queries all settings related to the UART bus setup.	7-76
:SERialbus:SETup<x>:UART:BITorder	Sets the UART bus setup bit order or queries the current setting.	7-76
:SERialbus:SETup<x>:UART:BRATe	Sets the UART bus setup bit rate (data transfer rate) or queries the current setting.	7-76
:SERialbus:SETup<x>:UART:FORMat	Sets the UART bus setup data format or queries the current setting.	7-76
:SERialbus:SETup<x>:UART:PMODe	Sets the UART bus setup Parity mode or queries the current setting.	7-76
:SERialbus:SETup<x>:UART:POLarity	Sets the UART bus setup polarity or queries the current setting.	7-76
:SERialbus:SETup<x>:UART:SPOint	Sets the UART bus setup sample point or queries the current setting.	7-76
:SERialbus:SETup<x>:UART:TRACe	Sets the UART bus setup trace or queries the current setting.	7-76
:SERialbus:TLINk	Sets the serial bus setup trigger link or queries the current setting.	7-77

TRIGger Group

:TRIGger:EINTerval:EVENt<x>:CANBus?	Queries all settings related to the CAN bus signal trigger of the event.	7-78
:TRIGger:EINTerval:EVENt<x>:CANBus:ACK	Sets the ACK condition of the CAN bus signal trigger or queries the current setting.	7-78
:TRIGger:EINTerval:EVENt<x>:CANBus:BRATe	Sets the bit rate (data transfer rate) of the CAN bus signal trigger or queries the current setting.	7-78
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA?	Queries all settings related to the CAN bus signal trigger data.	7-78
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:BORDer	Sets the byte order of the CAN bus signal trigger data or queries the current setting.	7-78
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:CONDition	Sets the data condition of the CAN bus signal trigger or queries the current setting.	7-78
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA<x>	Sets the comparison data of the CAN bus signal trigger data or queries the current setting.	7-79
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DLC	Sets the number of valid bytes (DLC) of the CAN bus signal trigger data or queries the current setting.	7-79
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:HEXA	Sets the CAN bus signal trigger data in hexadecimal notation.	7-79
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:MSBLsb	Sets the MSB and LSB bits of the CAN bus signal trigger data or queries the current setting.	7-79
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:PATTern	Sets the CAN bus signal trigger data in binary notation or queries the current setting.	7-79
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:SIGN	Sets the sign of the CAN bus signal trigger data or queries the current setting.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt?	Queries all settings related to the ID of the extended format of the CAN bus signal trigger.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:HEXA	Sets the ID of the extended format of the CAN bus signal trigger in hexadecimal notation.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:PATTern	Sets the ID of the extended format of the CAN bus signal trigger in binary notation or queries the current setting.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR?	Queries all settings related to the OR condition of the CAN bus signal trigger.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>?	Queries all settings related to each ID of the OR condition of the CAN bus signal trigger.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:ACK	Sets each ACK condition of the OR condition of the CAN bus signal trigger or queries the current setting.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA?	Queries all settings related to each data of the OR condition of the CAN bus signal trigger.	7-80
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:BORDer	Sets byte order of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-81
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition	Sets each data condition of the OR condition of the CAN bus signal trigger or queries the current setting.	7-81
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA<x>	Sets comparison data of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-81
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DLC	Sets the number of valid bytes (DLC) of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-81
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:HEXA	Sets each data of the OR condition of the CAN bus signal trigger in hexadecimal notation.	7-82

Command	Function	Page
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:MSBLsb	Sets the MSB and LSB bits of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-82
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:PATTern	Sets each data of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.	7-82
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:SIGN	Sets sign of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-82
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:FORMat	Sets each message format (standard or extended) of the OR condition of the CAN bus signal trigger or queries the current setting.	7-82
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt?	Queries all settings related to the ID of each extended format of the OR condition of the CAN bus signal trigger.	7-82
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt:HEXA	Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in hexadecimal notation.	7-82
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt:PATTern	Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.	7-83
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd?	Queries all settings related to the ID of each standard format of the OR condition of the CAN bus signal trigger.	7-83
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:HEXA	Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in hexadecimal notation.	7-83
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:PATTern	Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.	7-83
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:MODE	Enables or disables each condition of the OR condition of the CAN bus signal trigger or queries the current setting.	7-83
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:RTR	Sets each RTR of the OR condition of the CAN bus signal trigger or queries the current setting.	7-83
:TRIGger:EINTerval:EVENt<x>:CANBus:IDSTd?	Queries all settings related to the ID of the standard format of the CAN bus signal trigger.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:IDSTd:HEXA	Sets the ID of the standard format of the CAN bus signal trigger in hexadecimal notation.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:IDSTd:PATTern	Sets the ID of the standard format of the CAN bus signal trigger in binary notation or queries the current setting.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:MODE	Sets the CAN bus signal trigger mode or queries the current setting.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:RECessive	Sets the recessive level (bus level) of the CAN bus signal trigger or queries the current setting.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:RTR	Sets the RTR of the CAN bus signal trigger or queries the current setting.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:SOURce	Sets the trigger source of the CAN bus signal trigger or queries the current setting.	7-84
:TRIGger:EINTerval:EVENt<x>:CANBus:SPOint	Sets the sample point of the CAN bus signal trigger or queries the current setting.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus?	Queries all settings related to the I^2C bus trigger of the event.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa?	Queries all settings related to the address of the I^2C bus trigger.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address?	Queries all settings related to the 10-bit address of the I^2C bus trigger.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:HEXA	Sets the 10-bit address of the I^2C bus trigger in hexadecimal notation.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:PATTern	Sets the 10-bit address of the I^2C bus trigger in binary notation or queries the current setting.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress?	Queries all settings related to the 7-bit address of the I^2C bus trigger.	7-85
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress:HEXA	Sets the 7-bit address of the I^2C bus trigger in hexadecimal notation.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress:PATTern	Sets the 7-bit address of the I^2C bus trigger in binary notation or queries the current setting.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub?	Queries all settings related to the 7-bit + Sub address of the I^2C bus trigger.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + Sub address of the I^2C bus trigger.	7-86

Command	Function	Page
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA	Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern	Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:SADDress?	Queries all settings related to the Sub address of the 7-bit + Sub address of the I²C bus trigger.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:SADDress:HEXA	Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.	7-86
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:SADDress:PATTern	Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:TYPE	Sets the address type of the I²C bus trigger or queries the current setting.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:CLOCk?	Queries all settings related to the clock of the I²C bus trigger.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:CLOCk:SOURce	Sets the clock trace of the I²C bus trigger or queries the current setting.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA?	Queries all settings related to the data of the I²C bus trigger.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:BYTE	Sets the number of data bytes of the I²C bus trigger or queries the current setting.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:CONDition	Sets the determination method (match or not match) of the data of the I²C bus trigger or queries the current setting.	7-87
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:DPOSition	Sets the position for comparing the data pattern of the I²C bus trigger or queries the current setting.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:HEXA<x>	Sets the data of the I²C bus trigger in hexadecimal notation.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:MODE	Enables/Disables the data conditions of the I²C bus trigger or queries the current setting.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PATTern<x>	Sets the data of the I²C bus trigger in binary notation or queries the current setting.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PMODe	Sets the pattern comparison start position mode of the data of the I²C bus trigger or queries the current setting.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:SOURce	Sets the data trace of the I²C bus trigger or queries the current setting.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl?	Queries all settings related to the general call of the I²C bus trigger.	7-88
:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:BIT7maddress?	Queries all settings related to the 7-bit master address of the general call of the I²C bus trigger.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:BIT7maddress:HEXA	Sets the 7-bit master address of the general call of the I²C bus trigger in hexadecimal notation.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:BIT7maddress:PATTern	Sets the 7-bit master address of the general call of the I²C bus trigger in binary notation or queries the current setting.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:SBYTe (Second Byte)	Sets the second byte type of the general call of the I²C bus trigger or queries the current setting.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:MODE	Sets the trigger mode of the I²C bus trigger or queries the current setting.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore?	Queries all settings related to the NON ACK ignore mode of the I²C bus trigger.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore:HSMode	Sets whether to ignore NON ACK in high speed mode of the I²C bus trigger or queries the current setting.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore:RACCess	Sets whether to ignore NON ACK in read access mode of the I²C bus trigger or queries the current setting.	7-89
:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore:SBYTe (Start Byte)	Sets whether to ignore NON ACK in the start byte of the I²C bus trigger or queries the current setting.	7-90
:TRIGger:EINTerval:EVENt<x>:I2CBus:SBHSmode?	Queries all settings related to the start byte and high speed mode of the I²C bus trigger.	7-90
:TRIGger:EINTerval:EVENt<x>:I2CBus:SBHSmode:TYPE	Sets the type of the start byte or high speed mode of the I²C bus trigger or queries the current setting.	7-90

Command	Function	Page
:TRIGger:EINTerval:EVENt<x>:LINBus?	Queries all settings related to LIN bus signal triggers of each event.	7-90
:TRIGger:EINTerval:EVENt<x>:LINBus:BRATe	Sets the LIN bus signal trigger bitrate (data transfer rate) or queries the current setting.	7-90
:TRIGger:EINTerval:EVENt<x>:LINBus:SOURce	Sets the LIN bus signal trigger source or queries the current setting.	7-90
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus?	Queries all settings related to the logic I²C bus trigger for each event.	7-90
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa?	Queries all settings related to the address of the logic I²C bus trigger.	7-90
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT10address?	Queries all settings related to the 10-bit address of the logic I²C bus trigger.	7-90
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT10address:HEXA	Sets the 10-bit address of the logic I²C bus trigger in hexadecimal notation.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT10address:PATTern	Sets the 10-bit address of the logic I²C bus trigger in binary notation or queries the current setting.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7ADdress?	Queries all settings related to the 7-bit address of the logic I²C bus trigger.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7ADdress:HEXA	Sets the 7-bit address of the logic I²C bus trigger in hexadecimal notation.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7ADdress:PATTern	Sets the 7-bit address of the logic I²C bus trigger in binary notation or queries the current setting.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub?	Queries all settings related to the 7-bit + Sub address of the logic I²C bus trigger.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA	Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.	7-91
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern	Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.	7-92
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:SADDress?	Queries all settings related to the sub address of the 7-bit + Sub address of the logic I²C bus trigger.	7-92
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:HEXA	Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.	7-92
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:PATTern	Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.	7-92
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:TYPE	Sets the address type of the logic I²C bus trigger or queries the current setting.	7-92
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:CLOCk?	Queries all settings related to the clock of the logic I²C bus trigger.	7-92
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:CLOCk:SOURce	Sets the clock trace for the logic I²C bus trigger or queries the current setting.	7-93
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA?	Queries all settings related to the data of the logic I²C bus trigger.	7-93
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:BYTE	Sets the number of settings for the logic I²C bus trigger or queries the current setting.	7-93
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:CONDition	Sets the determination method for the data of the logic I²C bus trigger (match / no match) or queries the current setting.	7-93
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:DPOSition	Sets the pattern comparison position for the data of the logic I²C bus trigger or queries the current setting.	7-93
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:HEXA<x>	Sets the data of the logic I²C bus trigger in hexadecimal notation.	7-93
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:MODE	Enables/disables the data conditions of the logic I²C bus trigger or queries the current setting.	7-94

Command	Function	Page
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:PATTern<x>	Sets the data for the logic I²C bus trigger in binary notation or queries the current setting.	7-94
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:PMODe	Sets the pattern comparison start position for the data of the logic I²C bus trigger or queries the current setting.	7-94
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:SOURce	Sets the data trace for the logic I²C bus trigger or queries the current setting.	7-94
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl?	Queries all settings related to the general call of the logic I²C bus trigger.	7-94
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:BIT7maddress?	Queries all settings related to the 7-bit master address of the general call of the logic I²C bus trigger.	7-94
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:BIT7maddress:HEXA	Sets the 7-bit master address of the general call of the logic I²C bus trigger in hexadecimal notation.	7-94
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:BIT7maddress:PATTern	Sets the 7-bit master address of the general call of the logic I²C bus trigger in binary notation or queries the current setting.	7-95
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:SBYTe (Second Byte)	Sets the type of the second byte of the general call of the logic I²C bus trigger or queries the current setting.	7-95
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:MODE	Sets the trigger mode for the logic I²C bus trigger or queries the current setting.	7-95
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore?	Queries all settings related to the NON-ACK Ignore mode of the logic I²C bus trigger.	7-95
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore:HSMode	Sets whether to ignore NON ACK in high speed mode of the logic I²C bus trigger or queries the current setting.	7-95
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore:RACCess	Sets whether to ignore NON ACK in read access mode of the logic I²C bus trigger or queries the current setting.	7-95
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore:SBYTe (Start Byte)	Sets whether to ignore NON ACK in the start byte of the logic I²C bus trigger or queries the current setting.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:SBHSmode?	Queries all settings related to the start byte/high speed mode of the logic I²C bus trigger.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:SBHSmode:TYPE	Sets the type of the start byte/high speed mode of the logic I²C bus trigger or queries the current setting.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:LINBus?	Queries all settings related to the logic LIN bus signal trigger of each event.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:LINBus:BRATe	Sets the bit rate (data transfer rate) of the logic LIN bus signal trigger or queries the current setting.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:LINBus:SOURce	Sets the trigger source of the logic LIN bus signal trigger or queries the current setting.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus?	Queries all settings related to the logic SPI bus trigger for each event.	7-96
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:BITorder	Sets the bit order for the logic SPI bus trigger or queries the current setting.	7-97
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk?	Queries all settings related to the clock of the logic SPI bus trigger.	7-97
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:POLarity	Sets the polarity of the clock trace for the logic SPI bus trigger or queries the current setting.	7-97
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:SOURce	Sets the clock trace for the logic SPI bus trigger or queries the current setting.	7-97
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS?	Queries all settings related to the chip select of the logic SPI bus trigger.	7-97
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:ACTive	Sets the active level of the chip select for the logic SPI bus trigger or queries the current setting.	7-97
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:SOURce	Sets the chip select trace for the logic SPI bus trigger or queries the current setting.	7-98
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>?	Queries all settings related to each data of the logic SPI bus trigger.	7-98
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:BYTE	Sets the number of settings for each data of the logic SPI bus trigger or queries the current setting.	7-98
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:CONDition	Sets the determination method for the data of the logic SPI bus trigger (match / no match) or queries the current setting.	7-98
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:DPOSition	Sets the pattern comparison start position for the data of the logic SPI bus trigger or queries the current setting.	7-98

Command	Function	Page
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:HEXA<x>	Sets the data of the logic SPI bus trigger in hexadecimal notation.	7-98
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:PATTern<x>	Sets each data of the logic SPI bus trigger in binary notation or queries the current setting.	7-99
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:SOURce	Sets the trace of each data of the logic SPI bus trigger or queries the current setting.	7-99
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:MODE	Sets the wiring method (3-wire/4-wire) of the logic SPI bus trigger or queries the current setting.	7-99
:TRIGger:EINTerval:EVENt<x>:SPIBus?	Queries all settings related to the SPI bus trigger of the event.	7-99
:TRIGger:EINTerval:EVENt<x>:SPIBus:BITorder	Sets the bit order of the SPI bus trigger or queries the current setting.	7-99
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk?	Queries all settings related to the clock of the SPI bus trigger.	7-99
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:POLarity	Sets the polarity of the clock trace of the SPI bus trigger or queries the current setting.	7-99
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:SOURce	Sets the clock trace of the SPI bus trigger or queries the current setting.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS?	Queries all settings related to the chip select of the SPI bus trigger.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:ACTive	Sets the active level of the chip select of the SPI bus trigger or queries the current setting.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:SOURce	Sets the chip select trace of the SPI bus trigger or queries the current setting.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>?	Queries all settings related to the data of the SPI bus trigger.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:BYTE	Sets the number of bytes of the data of the SPI bus trigger or queries the current setting.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:CONDition	Sets the determination method (match or not match) of the data of the SPI bus trigger or queries the current setting.	7-100
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:DPOSition	Sets the pattern comparison start position of the data of the SPI bus trigger or queries the current setting.	7-101
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:HEXA<x>	Sets the data of the SPI bus trigger in hexadecimal notation.	7-101
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:PATTern<x>	Sets the data of the SPI bus trigger in binary notation or queries the current setting.	7-101
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:SOURce	Sets the trace of the data of the SPI bus trigger or queries the current setting.	7-101
:TRIGger:EINTerval:EVENt<x>:SPIBus:MODE	Sets the wiring system of the SPI bus trigger (three-wire or four-wire) or queries the current setting.	7-101
:TRIGger:EINTerval:EVENt<x>:STATe:CHANnel<x>	Sets the condition to be satisfied of the channel or queries the current setting.	7-101
:TRIGger:EINTerval:EVENt<x>:TYPE	Sets the trigger type of the event or queries the current setting.	7-102
:TRIGger:ENHanced:CANBus?	Queries all settings related to the CAN bus signal trigger.	7-102
:TRIGger:ENHanced:CANBus:ACK	Sets the ACK condition of the CAN bus signal trigger or queries the current setting.	7-102
:TRIGger:ENHanced:CANBus:BRATe	Sets the bit rate (data transfer rate) of the CAN bus signal trigger or queries the current setting.	7-102
:TRIGger:ENHanced:CANBus:DATA?	Queries all settings related to the CAN bus signal trigger data.	7-102
:TRIGger:ENHanced:CANBus:DATA:BORDer	Sets the byte order of the CAN bus signal trigger data or queries the current setting.	7-102
:TRIGger:ENHanced:CANBus:DATA:CONDition	Sets the data condition of the CAN bus signal trigger or queries the current setting.	7-102
:TRIGger:ENHanced:CANBus:DATA:DATA<x>	Sets the comparison data of the CAN bus signal trigger data or queries the current setting.	7-102
:TRIGger:ENHanced:CANBus:DATA:DLC	Sets the number of valid bytes (DLC) of the CAN bus signal trigger data or queries the current setting.	7-103
:TRIGger:ENHanced:CANBus:DATA:HEXA	Sets the CAN bus signal trigger data in hexadecimal notation.	7-103
:TRIGger:ENHanced:CANBus:DATA:MSBLsb	Sets the MSB and LSB bits of the CAN bus signal trigger data or queries the current setting.	7-103
:TRIGger:ENHanced:CANBus:DATA:PATTern	Sets the CAN bus signal trigger data in binary notation or queries the current setting.	7-103

Command	Function	Page
:TRIGger:ENHanced:CANBus:DATA:SIGN	Sets the sign of the CAN bus signal trigger data or queries the current setting.	7-103
:TRIGger:ENHanced:CANBus:IDEXt?	Queries all settings related to the ID of the extended format of the CAN bus signal trigger.	7-103
:TRIGger:ENHanced:CANBus:IDEXt:HEXA	Sets the ID of the extended format of the CAN bus signal trigger in hexadecimal notation.	7-103
:TRIGger:ENHanced:CANBus:IDEXt:PATTern	Sets the ID of the extended format of the CAN bus signal trigger in binary notation or queries the current setting.	7-103
:TRIGger:ENHanced:CANBus:IDOR?	Queries all settings related to the OR condition of the CAN bus signal trigger.	7-103
:TRIGger:ENHanced:CANBus:IDOR:ID<x>?	Queries all settings related to each ID of the OR condition of the CAN bus signal trigger.	7-104
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:ACK	Sets each ACK condition of the OR condition of the CAN bus signal trigger or queries the current setting.	7-104
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA?	Queries all settings related to each data of the OR condition of the CAN bus signal trigger.	7-104
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:BORDer	Sets byte order of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-104
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:CONDition	Sets each data condition of the OR condition of the CAN bus signal trigger or queries the current setting.	7-104
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:DATA<x>	Sets comparison data of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-104
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:DLC	Sets the number of valid bytes (DLC) of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-105
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:HEXA	Sets each data of the OR condition of the CAN bus signal trigger in hexadecimal notation.	7-105
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:MSBLsb	Sets the MSB and LSB bits of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-105
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:PATTern	Sets each data of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.	7-105
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:SIGN	Sets sign of each data of the OR condition of the CAN bus signal trigger or queries the current setting.	7-105
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:FORMat	Sets each message format (standard or extended) of the OR condition of the CAN bus signal trigger or queries the current setting.	7-105
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt?	Queries all settings related to the ID of each extended format of the OR condition of the CAN bus signal trigger.	7-106
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:HEXA	Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in hexadecimal notation.	7-106
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:PATTern	Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.	7-106
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd?	Queries all settings related to the ID of each standard format of the OR condition of the CAN bus signal trigger.	7-106
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:HEXA	Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in hexadecimal notation.	7-106
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:PATTern	Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.	7-106
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:MODE	Enables or disables each condition of the OR condition of the CAN bus signal trigger or queries the current setting.	7-107
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:RTR	Sets each RTR of the OR condition of the CAN bus signal trigger or queries the current setting.	7-107
:TRIGger:ENHanced:CANBus:IDSTd?	Queries all settings related to the ID of the standard format of the CAN bus signal trigger.	7-107
:TRIGger:ENHanced:CANBus:IDSTd:HEXA	Sets the ID of the standard format of the CAN bus signal trigger in hexadecimal notation.	7-107
:TRIGger:ENHanced:CANBus:IDSTd:PATTern	Sets the ID of the standard format of the CAN bus signal trigger in binary notation or queries the current setting.	7-107
:TRIGger:ENHanced:CANBus:MODE	Sets the CAN bus signal trigger mode or queries the current setting.	7-107
:TRIGger:ENHanced:CANBus:RECessive	Sets the recessive level (bus level) of the CAN bus signal trigger or queries the current setting.	7-107
:TRIGger:ENHanced:CANBus:RTR	Sets the RTR of the CAN bus signal trigger or queries the current setting.	7-107
:TRIGger:ENHanced:CANBus:SOURce	Sets the trigger source of the CAN bus signal trigger or queries the current setting.	7-108
:TRIGger:ENHanced:CANBus:SPOint	Sets the sample point of the CAN bus signal trigger or queries the current setting.	7-108
:TRIGger:ENHanced:I2CBus?	Queries all settings related to the I^2C bus trigger.	7-108

Command	Function	Page
:TRIGger:ENHanced:I2CBus:ADATa?	Queries all settings related to the address of the I²C bus trigger.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT10address?	Queries all settings related to the 10-bit address of the I²C bus trigger.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT10address:HEXA	Sets the 10-bit address of the I²C bus trigger in hexadecimal notation.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT10address:PATTern	Sets the 10-bit address of the I²C bus trigger in binary notation or queries the current setting.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT7ADdress?	Queries all settings related to the 7-bit address of the I²C bus trigger.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT7ADdress:HEXA	Sets the 7-bit address of the I²C bus trigger in hexadecimal notation.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT7ADdress:PATTern	Sets the 7-bit address of the I²C bus trigger in binary notation or queries the current setting.	7-108
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub?	Queries all settings related to the 7-bit + Sub address of the I²C bus trigger.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + Sub address of the I²C bus trigger.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA	Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern	Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:SADDress?	Queries all settings related to the Sub address of the 7-bit + Sub address of the I²C bus trigger.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:SADDress:HEXA	Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:SADDress:PATTern	Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.	7-109
:TRIGger:ENHanced:I2CBus:ADATa:TYPE	Sets the address type of the I²C bus trigger or queries the current setting.	7-109
:TRIGger:ENHanced:I2CBus:CLOCk?	Queries all settings related to the clock of the I²C bus trigger.	7-109
:TRIGger:ENHanced:I2CBus:CLOCk:SOURce	Sets the clock trace of the I²C bus trigger or queries the current setting.	7-110
:TRIGger:ENHanced:I2CBus:DATA?	Queries all settings related to the data of the I²C bus trigger.	7-110
:TRIGger:ENHanced:I2CBus:DATA:BYTE	Sets the number of data bytes of the I²C bus trigger or queries the current setting.	7-110
:TRIGger:ENHanced:I2CBus:DATA:CONDition	Sets the determination method (match or not match) of the data of the I²C bus trigger or queries the current setting.	7-110
:TRIGger:ENHanced:I2CBus:DATA:DPOSition	Sets the position for comparing the data pattern of the I²C bus trigger or queries the current setting.	7-110
:TRIGger:ENHanced:I2CBus:DATA:HEXA<x>	Sets the data of the I²C bus trigger in hexadecimal notation.	7-110
:TRIGger:ENHanced:I2CBus:DATA:MODE	Enables/Disables the data conditions of the I²C bus trigger or queries the current setting.	7-110
:TRIGger:ENHanced:I2CBus:DATA:PATTern<x>	Sets the data of the I²C bus trigger in binary notation or queries the current setting.	7-110
:TRIGger:ENHanced:I2CBus:DATA:PMODe	Sets the pattern comparison start position mode of the data of the I²C bus trigger or queries the current setting.	7-111
:TRIGger:ENHanced:I2CBus:DATA:SOURce	Sets the data trace of the I²C bus trigger or queries the current setting.	7-111
:TRIGger:ENHanced:I2CBus:GCALl?	Queries all settings related to the general call of the I²C bus trigger.	7-111
:TRIGger:ENHanced:I2CBus:GCALl:BIT7maddress?	Queries all settings related to the 7-bit master address of the general call of the I²C bus trigger.	7-111
:TRIGger:ENHanced:I2CBus:GCALl:BIT7maddress:HEXA	Sets the 7-bit master address of the general call of the I²C bus trigger in hexadecimal notation.	7-111
:TRIGger:ENHanced:I2CBus:GCALl:BIT7maddress:PATTern	Sets the 7-bit master address of the general call of the I²C bus trigger in binary notation or queries the current setting.	7-111
:TRIGger:ENHanced:I2CBus:GCALl:SBYTe (Second Byte)	Sets the second byte type of the general call of the I²C bus trigger or queries the current setting.	7-111
:TRIGger:ENHanced:I2CBus:MODE	Sets the trigger mode of the I²C bus trigger or queries the current setting.	7-111
:TRIGger:ENHanced:I2CBus:NAIGnore?	Queries all settings related to the NON ACK ignore mode of the I²C bus trigger.	7-111
:TRIGger:ENHanced:I2CBus:NAIGnore:HSMode	Sets whether to ignore NON ACK in high speed mode of the I²C bus trigger or queries the current setting.	7-112

Command	Function	Page
:TRIGger:ENHanced:I2CBus:NAIGnore:RACCess	Sets whether to ignore NON ACK in read access mode of the I²C bus trigger or queries the current setting.	7-112
:TRIGger:ENHanced:I2CBus:NAIGnore:SBYTe (Start Byte)	Sets whether to ignore NON ACK in the start byte of the I²C bus trigger or queries the current setting.	7-112
:TRIGger:ENHanced:I2CBus:SBHSmode?	Queries all settings related to the start byte and high speed mode of the I²C bus trigger.	7-112
:TRIGger:ENHanced:I2CBus:SBHSmode:TYPE	Sets the type of the start byte or high speed mode of the I²C bus trigger or queries the current setting.	7-112
:TRIGger:ENHanced:LINBus?	Queries all settings related to the LIN bus trigger or queries the current setting.	7-112
:TRIGger:ENHanced:LINBus:BRATe	Sets the LIN bus signal trigger bitrate (data transfer rate) or queries the current setting.	7-112
:TRIGger:ENHanced:LINBus:SOURce	Sets the LIN bus signal trigger source or queries the current setting.	7-112
:TRIGger:ENHanced:SPIBus?	Queries all settings related to the SPI bus trigger.	7-112
:TRIGger:ENHanced:SPIBus:BITorder	Sets the bit order of the SPI bus trigger or queries the current setting.	7-113
:TRIGger:ENHanced:SPIBus:CLOCk?	Queries all settings related to the clock of the SPI bus trigger.	7-113
:TRIGger:ENHanced:SPIBus:CLOCk:POLarity	Sets the polarity of the clock trace of the SPI bus trigger or queries the current setting.	7-113
:TRIGger:ENHanced:SPIBus:CLOCk:SOURce	Sets the clock trace of the SPI bus trigger or queries the current setting.	7-113
:TRIGger:ENHanced:SPIBus:CS?	Queries all settings related to the chip select of the SPI bus trigger.	7-113
:TRIGger:ENHanced:SPIBus:CS:ACTive	Sets the active level of the chip select of the SPI bus trigger or queries the current setting.	7-113
:TRIGger:ENHanced:SPIBus:CS:SOURce	Sets the chip select trace of the SPI bus trigger or queries the current setting.	7-113
:TRIGger:ENHanced:SPIBus:DATA<x>?	Queries all settings related to the data of the SPI bus trigger.	7-113
:TRIGger:ENHanced:SPIBus:DATA<x>:BYTE	Sets the number of bytes of the data of the SPI bus trigger or queries the current setting.	7-113
:TRIGger:ENHanced:SPIBus:DATA<x>:CONDition	Sets the determination method (match or not match) of the data of the SPI bus trigger or queries the current setting.	7-114
:TRIGger:ENHanced:SPIBus:DATA<x>:DPOSition	Sets the pattern comparison start position of the data of the SPI bus trigger or queries the current setting.	7-114
:TRIGger:ENHanced:SPIBus:DATA<x>:HEXA<x>	Sets the data of the SPI bus trigger in hexadecimal notation.	7-114
:TRIGger:ENHanced:SPIBus:DATA<x>:PATTern<x>	Sets the data of the SPI bus trigger in binary notation or queries the current setting.	7-114
:TRIGger:ENHanced:SPIBus:DATA<x>:SOURce	Sets the trace of the data of the SPI bus trigger or queries the current setting.	7-114
:TRIGger:ENHanced:SPIBus:MODE	Sets the wiring system of the SPI bus trigger (three-wire or four-wire) or queries the current setting.	7-114
:TRIGger:ENHanced:UART?	Queries all settings related to the UART signal trigger.	7-114
:TRIGger:ENHanced:UART:BRATe	Sets the UART signal trigger bit rate (data transfer rate) or queries the current setting.	7-115
:TRIGger:ENHanced:UART:FORMat	Sets the UART signal trigger format or queries the current setting.	7-115
:TRIGger:ENHanced:UART:POLarity	Sets the UART signal trigger polarity or queries the current setting.	7-115
:TRIGger:ENHanced:UART:SOURce	Sets the UART signal trigger source or queries the current setting.	7-115
:TRIGger:ENHanced:UART:SPOint	Sets the UART signal trigger sample point or queries the current setting.	7-115
:TRIGger:LOGic:I2CBus?	Queries all settings related to the logic I²C bus trigger.	7-115
:TRIGger:LOGic:I2CBus:ADATa?	Queries all settings related to the address of the logic I²C bus trigger.	7-115
:TRIGger:LOGic:I2CBus:ADATa:BIT10address?	Queries all settings related to the 10-bit address of the logic I²C bus trigger.	7-115
:TRIGger:LOGic:I2CBus:ADATa:BIT10address:HEXA	Sets the 10-bit address of the logic I²C bus trigger in hexadecimal notation.	7-115
:TRIGger:LOGic:I2CBus:ADATa:BIT10address:PATTern	Sets the 10-bit address of the logic I²C bus trigger in binary notation or queries the current setting.	7-115
:TRIGger:LOGic:I2CBus:ADATa:BIT7ADdress?	Queries all settings related to the 7-bit address of the logic I²C bus trigger.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7ADdress:HEXA	Sets the 7-bit address of the logic I²C bus trigger in hexadecimal notation.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7ADdress:PATTern	Sets the 7-bit address of the logic I²C bus trigger in binary notation or queries the current setting.	7-116

Command	Function	Page
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub?	Queries all settings related to the 7-bit + Sub address of the logic I²C bus trigger.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess?	Queries all settings related to the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA	Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern	Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:SADDress?	Queries all settings related to the sub address of the 7-bit + Sub address of the logic I²C bus trigger.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:HEXA	Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.	7-116
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:PATTern	Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.	7-117
:TRIGger:LOGic:I2CBus:ADATa:TYPE	Sets the address type of the logic I²C bus trigger or queries the current setting.	7-117
:TRIGger:LOGic:I2CBus:CLOCk?	Queries all settings related to the clock of the logic I²C bus trigger.	7-117
:TRIGger:LOGic:I2CBus:CLOCk:SOURce	Sets the clock trace for the logic I²C bus trigger or queries the current setting.	7-117
:TRIGger:LOGic:I2CBus:DATA?	Queries all settings related to the data of the logic I²C bus trigger.	7-117
:TRIGger:LOGic:I2CBus:DATA:BYTE	Sets the number of settings for the logic I²C bus trigger or queries the current setting.	7-117
:TRIGger:LOGic:I2CBus:DATA:CONDition	Sets the determination method for the data of the logic I²C bus trigger (match / no match) or queries the current setting.	7-117
:TRIGger:LOGic:I2CBus:DATA:DPOSition	Sets the pattern comparison position for the data of the logic I²C bus trigger or queries the current setting.	7-117
:TRIGger:LOGic:I2CBus:DATA:HEXA<x>	Sets the data of the logic I²C bus trigger in hexadecimal notation.	7-117
:TRIGger:LOGic:I2CBus:DATA:MODE	Enables/disables the data conditions of the logic I²C bus trigger or queries the current setting.	7-118
:TRIGger:LOGic:I2CBus:DATA:PATTern<x>	Sets the data for the logic I²C bus trigger in binary notation or queries the current setting.	7-118
:TRIGger:LOGic:I2CBus:DATA:PMODe	Sets the pattern comparison start position for the data of the logic I²C bus trigger or queries the current setting.	7-118
:TRIGger:LOGic:I2CBus:DATA:SOURce	Sets the data trace for the logic I²C bus trigger or queries the current setting.	7-118
:TRIGger:LOGic:I2CBus:GCALl?	Queries all settings related to the general call of the logic I²C bus trigger.	7-118
:TRIGger:LOGic:I2CBus:GCALl:BIT7maddress?	Queries all settings related to the 7-bit master address of the general call of the logic I²C bus trigger.	7-118
:TRIGger:LOGic:I2CBus:GCALl:BIT7maddress:HEXA	Sets the 7-bit master address of the general call of the logic I²C bus trigger in hexadecimal notation.	7-118
:TRIGger:LOGic:I2CBus:GCALl:BIT7maddress:PATTern	Sets the 7-bit master address of the general call of the logic I²C bus trigger in binary notation or queries the current setting.	7-118
:TRIGger:LOGic:I2CBus:GCALl:SBYTe (Second Byte)	Sets the type of the second byte of the general call of the logic I²C bus trigger or queries the current setting.	7-118
:TRIGger:LOGic:I2CBus:MODE	Sets the trigger mode for the logic I²C bus trigger or queries the current setting.	7-119
:TRIGger:LOGic:I2CBus:NAIGnore?	Queries all settings related to the NON-ACK Ignore mode of the logic I²C bus trigger.	7-119
:TRIGger:LOGic:I2CBus:NAIGnore:HSMode	Sets whether to ignore NON ACK in high speed mode of the logic I²C bus trigger or queries the current setting.	7-119
:TRIGger:LOGic:I2CBus:NAIGnore:RACCess	Sets whether to ignore NON ACK in read access mode of the logic I²C bus trigger or queries the current setting.	7-119
:TRIGger:LOGic:I2CBus:NAIGnore:SBYTe (Start Byte)	Sets whether to ignore NON ACK in the start byte of the logic I²C bus trigger or queries the current setting.	7-119
:TRIGger:LOGic:I2CBus:SBHSmode?	Queries all settings related to the start byte/high speed mode of the logic I²C bus trigger.	7-119
:TRIGger:LOGic:I2CBus:SBHSmode:TYPE	Sets the type of the start byte/high speed mode of the logic I²C bus trigger or queries the current setting.	7-119
:TRIGger:LOGic:LINBus?	Queries all settings related to the logic LIN bus signal triggers.	7-119
:TRIGger:LOGic:LINBus:BRATe	Sets the bit rate (data transfer rate) of the logic LIN bus signal trigger or queries the current setting.	7-119
:TRIGger:LOGic:LINBus:SOURce	Sets the trigger source of the logic LIN bus signal trigger or queries the current setting.	7-120

Command	Function	Page
:TRIGger:LOGic:SPIBus?	Queries all settings related to the logic SPI bus trigger.	7-120
:TRIGger:LOGic:SPIBus:BITorder	Sets the bit order for the logic SPI bus trigger or queries the current setting.	7-120
:TRIGger:LOGic:SPIBus:CLOCk?	Queries all settings related to the clock of the logic SPI bus trigger.	7-120
:TRIGger:LOGic:SPIBus:CLOCk:POLarity	Sets the polarity of the clock trace for the logic SPI bus trigger or queries the current setting.	7-120
:TRIGger:LOGic:SPIBus:CLOCk:SOURce	Sets the clock trace for the logic SPI bus trigger or queries the current setting.	7-120
:TRIGger:LOGic:SPIBus:CS?	Queries all settings related to the chip select of the logic SPI bus trigger.	7-120
:TRIGger:LOGic:SPIBus:CS:ACTive	Sets the active level of the chip select for the logic SPI bus trigger or queries the current setting.	7-120
:TRIGger:LOGic:SPIBus:CS:SOURce	Sets the chip select trace for the logic SPI bus trigger or queries the current setting.	7-120
:TRIGger:LOGic:SPIBus:DATA<x>?	Queries all settings related to each data of the logic SPI bus trigger.	7-120
:TRIGger:LOGic:SPIBus:DATA<x>:BYTE	Sets the number of settings for each data of the logic SPI bus trigger or queries the current setting.	7-121
:TRIGger:LOGic:SPIBus:DATA<x>:CONDition	Sets the determination method for the data of the logic SPI bus trigger (match / no match) or queries the current setting.	7-121
:TRIGger:LOGic:SPIBus:DATA<x>:DPOSition	Sets the pattern comparison start position for the data of the logic SPI bus trigger or queries the current setting.	7-121
:TRIGger:LOGic:SPIBus:DATA<x>:HEXA<x>	Sets the data of the logic SPI bus trigger in hexadecimal notation.	7-121
:TRIGger:LOGic:SPIBus:DATA<x>:PATTern<x>	Sets each data of the logic SPI bus trigger in binary notation or queries the current setting.	7-121
:TRIGger:LOGic:SPIBus:DATA<x>:SOURce	Sets the trace of each data of the logic SPI bus trigger or queries the current setting.	7-121
:TRIGger:LOGic:SPIBus:MODE	Sets the wiring method (3-wire/4-wire) of the logic SPI bus trigger or queries the current setting.	7-121
:TRIGger:LOGic:UART?	Queries all settings related to the logic UART signal trigger.	7-121
:TRIGger:LOGic:UART:BRATe	Sets the logic UART signal trigger bit rate (data transfer rate) or queries the current setting.	7-122
:TRIGger:LOGic:UART:FORMat	Sets the logic UART signal trigger format or queries the current setting.	7-122
:TRIGger:LOGic:UART:POLarity	Sets the logic UART signal trigger polarity or queries the current setting.	7-122
:TRIGger:LOGic:UART:SOURce	Sets the logic UART signal trigger source or queries the current setting.	7-122
:TRIGger:LOGic:UART:SPOint	Sets the logic UART signal trigger sample point or queries the current setting.	7-122
:TRIGger:SOURce:CHANnel<x>:LEVel	Sets the trigger level of the channel or queries the current setting.	7-122
:TRIGger:SOURce:CHANnel<x>:STATe	Sets the condition to be satisfied of the channel or queries the current setting.	7-122
:TRIGger:TYPE	Sets the trigger type or queries the current setting.	7-123

7.2 ANALysis Group

:ANALysis:LSBus<x>?

Function Queries all settings related to the logic serial bus signal feature.

Syntax `:ANALysis:LSBus<x>?`

<x> = 1 or 2

:ANALysis:LSBus<x>[:ANALyze]?

Function Queries all settings related to the logic serial bus signal.

Syntax `:ANALysis:LSBus<x>[:ANALyze]?`

<x> = 1 or 2

:ANALysis:LSBus<x>[:ANALyze]:I2CBus?

Function Queries all settings related to the logic I^2C bus signal analysis.

Syntax `:ANALysis:LSBus<x>[:ANALyze]:I2CBus?`

<x> = 1 or 2

:ANALysis:LSBus<x>[:ANALyze]:I2CBus:CLOCk

Function Sets the clock channel of the logic I^2C bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:I2CBus:
CLOCk {A<y>|B<y>|C<y>|D<y>}
:ANALysis:LSBus<x>[:ANALyze]:I2CBus:
CLOCk?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:I2CBUS:
CLOCK A0
:ANALYSIS:LSBUS1:ANALYZE:I2CBUS:CLOCK?
-> :ANALYSIS:LSBUS1:ANALYZE:I2CBUS:
CLOCK A0
```
Description On 16-bit models, you can only select {A<x>|C<x>}.

:ANALysis:LSBus<x>[:ANALyze]:I2CBus:DTRace

Function Sets the data channel of the logic I^2C bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:I2CBus:
DTRace {A<y>|B<y>|C<y>|D<y>}
:ANALysis:LSBus<x>[:ANALyze]:I2CBus:
DTRace?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:I2CBUS:
DTRACE A0
:ANALYSIS:LSBUS1:ANALYZE:I2CBUS:
DTRACE? -> :ANALYSIS:LSBUS1:ANALYZE:
I2CBUS:DTRACE A0
```
Description On 16-bit models, you can only select {A<x>|C<x>}.

:ANALysis:LSBus<x>[:ANALyze]:LINBus?

Function Queries all settings related to the logic LIN bus signal analysis.

Syntax `:ANALysis:LSBus<x>[:ANALyze]:LINBus?`

<x> = 1 or 2

:ANALysis:LSBus<x>[:ANALyze]:LINBus:BRATe

Function Sets the bit rate (data transfer rate) of the logic LIN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
BRATe {<NRf>|USER,<NRf>}
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
BRATe?
```
<x> = 1 or 2

<NRf> = 1200, 2400, 4800, 9600, or 19200

<NRf> of USER = See section 4.4.

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
BRATE 19200
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
BRATE? -> :ANALYSIS:LSBUS1:ANALYZE:
LINBUS:BRATE 19200
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:BREak

Function Executes a field jump to the Break Field in the results of the logic LIN bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
FJUMp:BREak
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:FJUMP:
BREAK
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:CSUM

Function Executes a field jump to the Checksum Field in the results of the logic LIN bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
FJUMp:CSUM
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:FJUMP:
CSUM
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:DATA

Function Executes a field jump to the Data Field in the results of the logic LIN bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
FJUMp:DATA
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:FJUMP:
DATA
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:IDENtifier

Function Executes a field jump to the Identifier Field in the results of the logic LIN bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
FJUMp:IDENtifier
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:FJUMP:
IDENTIFIER
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:FJUMp:SYNCh

Function Executes a field jump to the Synch Field in the results of the logic LIN bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
FJUMp:SYNCh
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:FJUMP:
SYNCH
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:REVision

Function Sets the revision (1.3, 2.0, or Both) of the logic LIN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
REVision {BOTH|LIN1_3|LIN2_0}
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
REVision?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
REVISION LIN1_3
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
REVISION? -> :ANALYSIS:LSBUS1:ANALYZE:
LINBUS:REVISION LIN1_3
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:SPOint

Function Sets the logic LIN bus signal analysis sample point or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
SPOint {<NRf>}
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
SPOint?
```
<x> = 1 or 2

<NRf> = 18.8 to 90.6(%)

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
SPOINT 18.8
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
SPOINT?
-> :ANALYSIS:LSBUS1:ANALYZE:LINBUS:
SPOINT 18.8E+00
```

:ANALysis:LSBus<x>[:ANALyze]:LINBus:TRACe

Function Sets the trace of the logic LIN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
TRACe {A<y>|B<y>|C<y>|D<y>}
:ANALysis:LSBus<x>[:ANALyze]:LINBus:
TRACe?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:
TRACE A0
:ANALYSIS:LSBUS1:ANALYZE:LINBUS:TRACE?
-> :ANALYSIS:LSBUS1:ANALYZE:LINBUS:
TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}..

:ANALysis:LSBus<x>[:ANALyze]:LIST?

Function Queries all settings related to the analysis result list of the logic serial bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LIST?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LIST?
-> :ANALYSIS:LSBUS1:ANALYZE:LIST:
DISPLAY 1;MODE DETAIL;SCROLL HORIZONTAL
```

:ANALysis:LSBus<x>[:ANALyze]:LIST:DISPlay

Function Turns ON/OFF the analysis result list of the logic serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LIST:
DISPlay {<Boolean>}
:ANALysis:LSBus<x>[:ANALyze]:LIST:
DISPlay?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LIST:
DISPLAY ON
:ANALYSIS:LSBUS1:ANALYZE:LIST:
DISPLAY? -> :ANALYSIS:LSBUS1:ANALYZE:
LIST:DISPLAY 1
```

:ANALysis:LSBus<x>[:ANALyze]:LIST:ITEM?

Function Queries all items displayed on the analysis result list of the logic serial bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LIST:ITEM?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LIST:ITEM?
-> :ANALYSIS:LSBUS1:ANALYZE:LIST:
ITEM" No.,S/P,Hex,Form,R/W,ACK,"
```

:ANALysis:LSBus<x>[:ANALyze]:LIST:MODE

Function Sets the mode of the analysis result list of the logic serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LIST:
MODE {DETail|SIMPle}
:ANALysis:LSBus<x>[:ANALyze]:LIST:
MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LIST:
MODE DETAIL
:ANALYSIS:LSBUS1:ANALYZE:LIST:MODE?
-> :ANALYSIS:LSBUS1:ANALYZE:LIST:
MODE DETAIL
```

:ANALysis:LSBus<x>[:ANALyze]:LIST:SCRoll

Function Sets the scroll method of the analysis result list of the logic serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LIST:
SCRoll {HORizontal|VERTical}
:ANALysis:LSBus<x>[:ANALyze]:LIST:
SCRoll?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LIST:
SCROLL HORIZONTAL
:ANALYSIS:LSBUS1:ANALYZE:LIST:SCROLL?
-> :ANALYSIS:LSBUS1:ANALYZE:LIST:
SCROLL HORIZONTAL
```

:ANALysis:LSBus<x>[:ANALyze]:LIST:VALue?

Function Queries the automated measured value of the specified analysis number in the analysis result list of the logic serial bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:LIST:
VALue? {<NRf>|MAXimum|MINimum}
```
<x> = 1 or 2
<NRf> = -40000 to 40000
(<NRf> = -2999 to 2999 for :ANALysis:SBUS<x>[:ANALyze]:MODE CANBus.)

Example
```
:ANALYSIS:LSBUS1:ANALYZE:LIST:VALUE? 1
-> :ANALYSIS:LSBUS1:ANALYZE:LIST:
VALUE" 1, P, 00, A, , 0,"
```

Description Set the data to MAXimum or MINimum to specify the maximum list display number or the minimum list display number.

:ANALysis:LSBus<x>[:ANALyze]:MODE

Function Sets the logic serial bus signal analysis mode or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
MODE {I2CBus|LINBus|SPIBus|UART}
:ANALysis:LSBus<x>[:ANALyze]:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:MODE I2CBUS
:ANALYSIS:LSBUS1:ANALYZE:MODE?
-> :ANALYSIS:LSBUS1:ANALYZE:
MODE I2CBUS
```

:ANALysis:LSBus<x>[:ANALyze]:RPOint

Function Sets the analysis reference point of the logic serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
RPOint {<NRf>,MANual|TRIGger}
:ANALysis:LSBus<x>[:ANALyze]:RPOint?
```
<x> = 1 or 2
<NRf> = -5 to 5(div)

Example
```
:ANALYSIS:LSBUS1:ANALYZE:
RPOINT MANUAL,1
:ANALYSIS:LSBUS1:ANALYZE:RPOINT?
-> :ANALYSIS:LSBUS1:ANALYZE:
RPOINT MANUAL, 1.00000E+00
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus?

Function Queries all settings related to the logic SPI bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus?
```
<x> = 1 or 2

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CLOCk?

Function Queries all settings related to the clock signal channel of the logic SPI bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CLOCk?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CLOCK?
-> :ANALYSIS:LSBUS1:ANALYZE:SPIBUS:
CLOCK:POLARITY FALL;SOURCE A0
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock signal channel of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CLOCk:POLarity {FALL|RISE}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CLOCk:POLarity?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CLOCK:
POLARITY FALL
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CLOCK:
POLARITY? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:CLOCK:POLARITY FALL
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CLOCk:SOURce

Function Sets the clock signal channel of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CLOCk:SOURce {A<y>|B<y>|C<y>|D<y>}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CLOCk:SOURce?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CLOCK:
SOURCE A0
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CLOCK:
SOURCE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:CLOCK:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CS?

Function Queries all settings related to the chip select signal channel of the logic SPI bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CS?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CS?
-> :ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CS:
ACTIVE HIGH;TRACE A0
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CS:ACTive

Function Sets the active level of the chip select signal channel of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CS:
ACTive {HIGH|LOW}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CS:
ACTive?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CS:
ACTIVE HIGH
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CS:
ACTIVE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:CS:ACTIVE HIGH
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:CS:TRACe

Function Sets the chip select signal channel of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CS:
TRACe {A<y>|B<y>|C<y>|D<y>|NONE}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
CS:
TRACe?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CS:
TRACE A0
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:CS:
TRACE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:CS:TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:DATA<x>?

Function Queries all settings related to each data of the logic SPI bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
DATA<x>?
```
<x> of LSBus<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:DATA1?
-> :ANALYSIS:LSBUS1:ANALYZE:SPIBUS:
DATA1:ACTIVE HIGH;TRACE A0
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:DATA<x>:ACTive

Function Sets the active level of each data of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
DATA<x>:ACTive {HIGH|LOW}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
DATA<x>:ACTive?
```
<x> of LSBus<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:DATA1:
ACTIVE HIGH
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:DATA1:
ACTIVE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:DATA1:ACTIVE HIGH
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus:DATA<x>:TRACe

Function Sets the data channel of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
DATA<x>:TRACe {A<y>|B<y>|C<y>|D<y>}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus:
DATA<x>:TRACe?
```
<x> of LSBus<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:DATA1:
TRACE A0
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:DATA1:
TRACE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:DATA1:TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]?

Function Queries all settings related to the setup of the logic SPI bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP?
-> :ANALYSIS:LSBUS1:ANALYZE:SPIBUS:
SETUP:BITORDER LSBFIRST;EMSBLSB 1,7;
FSIZE 4;ITIME 10.0000E-09;MODE WIRE3
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:BITorder

Function Sets the bit order of the logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:
BITorder {LSBFirst|MSBFirst}
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:BITorder?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
BITORDER LSBFIRST
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
BITORDER? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:SETUP:BITORDER LSBFIRST
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:EMSBLSB

Function Sets the enabled range of the field used for logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:EMSBLSB {<NRf>,<NRf>}
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:EMSBLSB?
```
<x> = 1 or 2
<NRf> = See section 4.5.

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
EMSBLSB 1,7
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
EMSBLSB? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:SETUP:EMSBLSB 1,7
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:FSIZe

Function Sets the field size used for logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:FSIZe {<NRf>}
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:FSIZe?
```
<x> = 1 or 2
<NRf> = 4 to 32

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
FSIZE 4
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
FSIZE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:SETUP:FSIZE 4
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:ITIMe

Function Sets the idle time used in logic SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:ITIMe {<Time>|DONTcare}
:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:
SETup]:ITIMe?
```
<x> = 1 or 2
<Time> = 10ns to 1ms in 10-ns steps

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
ITIME 10NS
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
ITIME? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:SETUP:ITIME 10.0000E-09
```

:ANALysis:LSBus<x>[:ANALyze]:SPIBus[:SETup]:MODE

Function Sets the wiring system of the logic SPI bus signal analysis (three-wire or four-wire) or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:MODE {WIRe3|WIRe4}
:ANALysis:LSBus<x>[:ANALyze]:
SPIBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
MODE WIRE3
:ANALYSIS:LSBUS1:ANALYZE:SPIBUS:SETUP:
MODE? -> :ANALYSIS:LSBUS1:ANALYZE:
SPIBUS:SETUP:MODE WIRE3
```

:ANALysis:LSBus<x>[:ANALyze]:UART?

Function Queries all settings related to the logic UART bus signal analysis.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:
BITORDER LSBFIRST;BRATE 19200;
BSPACE 10.00E-03;DFORMAT HEXA;
FORMAT BIT7PARITY;GROUPING 1;
PMODE EVEN;POLARITY NEGATIVE;
SPOINT 18.8E+00;TRACE A0
```

:ANALysis:LSBus<x>[:ANALyze]:UART:BITorder

Function Sets the logic UART bus signal analysis bit order or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:
BITorder {LSBFirst|MSBFirst}
:ANALysis:LSBus<x>[:ANALyze]:UART:
BITorder?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:
BITORDER LSBFIRST
:ANALYSIS:LSBUS1:ANALYZE:UART:
BITORDER?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:
BITORDER LSBFIRST
```

:ANALysis:LSBus<x>[:ANALyze]:UART:BRATe

Function Sets the logic UART bus signal analysis bit rate (data transfer rate) or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:BRATe
{<NRf>|USER,<NRf>}
:ANALysis:LSBus<x>[:ANALyze]:UART:
BRATe?
```
<x> = 1 or 2

<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, or 115200

<NRf> of USER = See section 4.6.

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:BRATE
19200
:ANALYSIS:LSBUS1:ANALYZE:UART:BRATE?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:BRATE
19200
```

:ANALysis:LSBus<x>[:ANALyze]:UART:BSPace

Function Sets the byte space for grouping data that is used in logic UART signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:
BSPace {<Time>}
:ANALysis:LSBus<x>[:ANALyze]:UART:
BSPace?
```
<x> = 1 or 2

<Time> = See section 4.6.

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:
BSPACE 10ms
:ANALYSIS:LSBUS1:ANALYZE:UART:BSPACE?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:BSPACE
10.00E-03
```

:ANALysis:LSBus<x>[:ANALyze]:UART:DFORmat

Function Sets the decoded character display format for logic UART signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:
DFORmat {ASCii|HEXA}
:ANALysis:LSBus<x>[:ANALyze]:UART:
DFORmat?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:
DFORMAT ASCII
:ANALYSIS:LSBUS1:ANALYZE:UART:DFORMAT?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:
DFORMAT ASCII
```

:ANALysis:LSBus<x>[:ANALyze]:UART:FORMat

Function Sets the logic UART bus signal analysis data format or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:
FORMat {BIT7parity|BIT8Noparity|
BIT8Parity}
:ANALysis:LSBus<x>[:ANALyze]:UART:
FORMat?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:
FORMAT BIT7PARITY
:ANALYSIS:LSBUS1:ANALYZE:UART:FORMAT?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:
FORMAT BIT7PARITY
```

:ANALysis:LSBus<x>[:ANALyze]:UART:GROuping

Function Turns on or off the grouping feature for logic UART signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:
GROuping {<Boolean>}
:ANALysis:LSBus<x>[:ANALyze]:UART:
GROuping?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:GROUPING
ON
:ANALYSIS:LSBUS1:ANALYZE:UART:GROUPING?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:
GROUPING 1
```

:ANALysis:LSBus<x>[:ANALyze]:UART:PMODe

Function Sets the logic UART bus signal analysis parity mode or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:PMODe {EVEN|ODD}
:ANALysis:LSBus<x>[:ANALyze]:UART:PMODe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:PMODE EVEN
:ANALYSIS:LSBUS1:ANALYZE:UART:PMODE?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:PMODE EVEN
```

:ANALysis:LSBus<x>[:ANALyze]:UART:POLarity

Function Sets the logic UART bus signal analysis parity or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:POLarity {NEGative|POSitive}
:ANALysis:LSBus<x>[:ANALyze]:UART:POLarity?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:POLARITY NEGATIVE
:ANALYSIS:LSBUS1:ANALYZE:UART:POLARITY?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:POLARITY NEGATIVE
```

:ANALysis:LSBus<x>[:ANALyze]:UART:SPOint

Function Sets the logic UART bus signal analysis sample point or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:SPOint {<NRf>}
:ANALysis:LSBus<x>[:ANALyze]:UART:SPOint?
```
<x> = 1 or 2

<NRf> = 18.8 to 90.6(%)

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:SPOINT 18.8
:ANALYSIS:LSBUS1:ANALYZE:UART:SPOINT?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:SPOINT 18.8E+00
```

:ANALysis:LSBus<x>[:ANALyze]:UART:TRACe

Function Sets the logic UART bus signal analysis trace or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>[:ANALyze]:UART:TRACe {A<y>|B<y>|C<y>|D<y>}
:ANALysis:LSBus<x>[:ANALyze]:UART:TRACe?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:ANALYSIS:LSBUS1:ANALYZE:UART:TRACE A0
:ANALYSIS:LSBUS1:ANALYZE:UART:TRACE?
-> :ANALYSIS:LSBUS1:ANALYZE:UART:TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:ANALysis:LSBus<x>:ZLINkage

Function Sets the zoom link of the logic serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:LSBus<x>:ZLINkage {OFF|Z1|Z2}
:ANALysis:LSBus<x>:ZLINkage?
```
<x> = 1 or 2

Example
```
:ANALYSIS:LSBUS1:ZLINKAGE OFF
:ANALYSIS:LSBUS1:ZLINKAGE?
-> :ANALYSIS:LSBUS1:ZLINKAGE OFF
```

:ANALysis:SBUS<x>?

Function Queries all settings related to the serial bus signal analysis feature.

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

:ANALysis:SBUS<x>:ANALyze?

Function Queries all settings related to the serial bus signal analysis.

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

:ANALysis:SBUS<x>[:ANALyze]:CANBus?

Function Queries all settings related to the CAN bus signal analysis.

Syntax `:ANALysis:SBUS<x>[:ANALyze]:CANBus?`
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS?
-> :ANALYSIS:SBUS1:ANALYZE:CANBUS:
BRATE 1000000;RECESSIVE HIGH;
SPOINT 62.5E+00;TRACE 1
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:BRATe

Function Sets the bit rate (data transfer rate) of the CAN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
BRATe {<NRf>|USER,<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
BRATe?
```
<x> = 1 or 2
<NRf> = 33300, 83300, 125000, 250000, 500000, 1000000
<NRf> of USER = See section 4.3.

:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:ACK

Function Executes a field jump to the ACK Field in the results of the CAN bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
FJUMp:ACK
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:FJUMP:
ACK
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:CONTrol

Function Executes a field jump to the Control Field in the results of the CAN bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
FJUMp:CONTrol
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:FJUMP:
CONTROL
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:CRC

Function Executes a field jump to the CRC Field in the results of the CAN bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
FJUMp:CRC
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:FJUMP:
CRC
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:DATA

Function Executes a field jump to the Data Field in the results of the CAN bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
FJUMp:DATA
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:FJUMP:
DATA
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:IDENtifier

Function Executes a field jump to the Identifier Field in the results of the CAN bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
FJUMp:IDENtifier
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:FJUMP:
IDENTIFIER
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:FJUMp:SOF

Function Executes a field jump to the SOF Field in the results of the CAN bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
FJUMp:SOF
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:FJUMP:
SOF
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:RECessive

Function Sets the recessive level (bus level) of the CAN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
RECessive {HIGH|LOW}
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
RECessive?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:
RECESSIVE HIGH
:ANALYSIS:SBUS1:ANALYZE:CANBUS:
RECESSIVE?
-> :ANALYSIS:SBUS1:ANALYZE:CANBUS:
RECESSIVE HIGH
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:SPOint

Function Sets the sample point of the CAN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
SPOint {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:
SPOINT 18.8
:ANALYSIS:SBUS1:ANALYZE:CANBUS:SPOINT?
-> :ANALYSIS:SBUS1:ANALYZE:CANBUS:
SPOINT 18.8E+00
```

:ANALysis:SBUS<x>[:ANALyze]:CANBus:TRACe

Function Sets the trace of the CAN bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
TRACe {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:CANBus:
TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:CANBUS:TRACE 1
:ANALYSIS:SBUS1:ANALYZE:CANBUS:TRACE?
-> :ANALYSIS:SBUS1:ANALYZE:CANBUS:
TRACE 1
```

:ANALysis:SBUS<x>[:ANALyze]:DECode

Function Turns the serial bus signal analysis decoding display ON/OFF or queries the current status.

Syntax
```
ANALysis:SBUS<x>[:ANALyze]:
DECode {<Boolean>}
:ANALysis:SBUS<x>[:ANALyze]:DECode?
```
<x>=1 or 2

Example
```
ANALYSIS:SBUS1:ANALYZE:DECODE ON
:ANALYSIS:SBUS1:ANALYZE:DECODE?
-> :ANALYSIS:SBUS1:ANALYZE:DECODE 1
```

:ANALysis:SBUS<x>[:ANALyze]:I2CBus?

Function Queries all settings related to the I^2C bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:I2CBus?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:I2CBus:CLOCk

Function Sets the clock channel of the I^2C bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:I2CBus:
CLOCk {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:I2CBus:
CLOCk?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:I2CBUS:CLOCK 1
:ANALYSIS:SBUS1:ANALYZE:I2CBUS:CLOCK?
-> :ANALYSIS:SBUS1:ANALYZE:I2CBUS:
CLOCK 1
```

:ANALysis:SBUS<x>[:ANALyze]:I2CBus:DTRace

Function Sets the data channel of the I^2C bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:I2CBus:
DTRace {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:I2CBus:
DTRace?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:I2CBUS:DTRACE 1
:ANALYSIS:SBUS1:ANALYZE:I2CBUS:DTRACE?
-> :ANALYSIS:SBUS1:ANALYZE:I2CBUS:
DTRACE 1
```

:ANALysis:SBUS<x>[:ANALyze]:LINBus?

Function Queries all settings related to the LIN bus signal analysis.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus?

<x>=1 or 2

:ANALysis:SBUS<x>[:ANALyze]:LINBus:BRATe

Function Sets the LIN bus signal analysis bitrate (data transfer rate) or queries the current setting.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:BRATe {<NRf>|USER,<NRf>}

:ANALysis:SBUS<x>[:ANALyze]:LINBus:BRATe?

<x>=1 or 2

<NRf>=1200, 2400, 4800, 9600, 19200

USER <NRf>=See section 4.4.

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:BRATE 19200

:ANALYSIS:SBUS1:ANALYZE:LINBUS:BRATE?

-> :ANALYSIS:SBUS1:ANALYZE:LINBUS:BRATE 19200

:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:BREak

Function Executes a field jump to the Break Field in the results of the LIN bus signal analysis.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:BREak

<x> = 1 or 2

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:FJUMP:BREAK

:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:CSUM

Function Executes a field jump to the Checksum Field in the results of the LIN bus signal analysis.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:CSUM

<x> = 1 or 2

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:FJUMP:CSUM

:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:DATA

Function Executes a field jump to the Data Field in the results of the LIN bus signal analysis.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:DATA

<x> = 1 or 2

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:FJUMP:DATA

:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:IDENtifier

Function Executes a field jump to the Identifier Field in the results of the LIN bus signal analysis.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:IDENtifier

<x> = 1 or 2

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:FJUMP:IDENTIFIER

:ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:SYNCh

Function Executes a field jump to the Synch Field in the results of the LIN bus signal analysis.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:FJUMp:SYNCh

<x> = 1 or 2

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:FJUMP:SYNCH

:ANALysis:SBUS<x>[:ANALyze]:LINBus:REVision

Function Sets the LIN bus signal analysis revision (1.3, 2.0, or Both) or queries the current setting.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:REVision {BOTH|LIN1_3|LIN2_0}

:ANALysis:SBUS<x>[:ANALyze]:LINBus:REVision?

<x>=1 or 2

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:REVISION LIN1_3

:ANALYSIS:SBUS1:ANALYZE:LINBUS:REVISION? -> :ANALYSIS:SBUS1:ANALYZE:LINBUS:REVISION LIN1_3

:ANALysis:SBUS<x>[:ANALyze]:LINBus:SPOint

Function Sets the LIN bus signal analysis sample point or queries the current setting.

Syntax :ANALysis:SBUS<x>[:ANALyze]:LINBus:SPOint {<NRf>}

:ANALysis:SBUS<x>[:ANALyze]:LINBus:SPOint?

<x> = 1 or 2

<NRf> = 18.8 to 90.6(%)

Example :ANALYSIS:SBUS1:ANALYZE:LINBUS:SPOINT 18.8

:ANALYSIS:SBUS1:ANALYZE:LINBUS:SPOINT?

-> :ANALYSIS:SBUS1:ANALYZE:LINBUS:SPOINT 18.8E+00

:ANALysis:SBUS<x>[:ANALyze]:LINBus:TRACe

Function Sets the LIN bus signal analysis trace or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:LINBus:
TRACe {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:LINBus:
TRACe?
```
<x>=1 or 2
<NRf>=1–8

Example
```
:ANALYSIS:SBUS1:ANALYZE:LINBUS:TRACE 1
:ANALYSIS:SBUS1:ANALYZE:LINBUS:TRACE?
-> :ANALYSIS:SBUS1:ANALYZE:LINBUS:
TRACE 1
```

:ANALysis:SBUS<x>[:ANALyze]:LIST?

Function Queries all settings related to the list display of the serial bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:LIST?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:LIST:DISPlay

Function Turns the serial bus signal analysis list display ON/OFF or queries the current status.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:LIST:
DISPlay {<Boolean>}
:ANALysis:SBUS<x>[:ANALyze]:LIST:
DISPlay?
```
<x>=1 or 2

:ANALysis:SBUS<x>[:ANALyze]:LIST:ITEM?

Function Queries the item in the list display of the serial bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:LIST:ITEM?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:LIST:ITEM?
-> :ANALYSIS:SBUS1:ANALYZE:LIST:
ITEM" No.,S/P,Hex,Form,R/W,ACK,"
```

:ANALysis:SBUS<x>[:ANALyze]:LIST:MODE

Function Sets the mode of the list display of the serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:LIST:
MODE {DETail|SIMPle}
:ANALysis:SBUS<x>[:ANALyze]:LIST:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:LIST:
MODE DETAIL
:ANALYSIS:SBUS1:ANALYZE:LIST:MODE?
-> :ANALYSIS:SBUS1:ANALYZE:LIST:
MODE DETAIL
```

:ANALysis:SBUS<x>[:ANALyze]:LIST:SCRoll

Function Sets the scroll method of the list display of the serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:
LIST:SCRoll {HORizontal|VERTical}
:ANALysis:SBUS<x>[:ANALyze]:
LIST:SCRoll?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:LIST:
SCROLL HORIZONTAL
:ANALYSIS:SBUS1:ANALYZE:LIST:SCROLL?
-> :ANALYSIS:SBUS1:ANALYZE:LIST:
SCROLL HORIZONTAL
```

ANALysis:SBUS<x>[:ANALyze]:LIST:VALue?

Function Queries the automated measured value of the specified analysis number in the analysis result list of the serial bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:LIST:VALue?
{<NRf>|MAXimum|MINimum}
```
<x> = 1 or 2
<NRf> = –40000 to 40000
(<NRf> = –2999 to 2999 for :ANALysis:SBUS<x>[:ANALyze]:MODE CANBus)

Example
```
:ANALYSIS:SBUS1:ANALYZE:LIST:
VALUE? 1
-> :ANALYSIS:SBUS1:ANALYZE:LIST:
VALUE "1, P, 00, A, , 0,"
```

Description Set the data to MAXimum or MINimum to specify the maximum list display number or the minimum list display number.

:ANALysis:SBUS<x>[:ANALyze]:MODE

Function Sets the serial bus signal analysis mode or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:
MODE {CANBus|I2CBus|LIN|SPIBus|UART}
:ANALysis:SBUS<x>[:ANALyze]:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:MODE I2CBUS
:ANALYSIS:SBUS1:ANALYZE:MODE?
-> :ANALYSIS:SBUS1:ANALYZE:MODE I2CBUS
```

:ANALysis:SBUS<x>[:ANALyze]:RPOint

Function Sets the analysis reference point of the serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:
RPOint {MANual,<NRf>|TRIGger}
:ANALysis:SBUS<x>[:ANALyze]:RPOint?
```
<x> = 1 or 2
<NRf> = –5 to 5 (div)

Example
```
:ANALYSIS:SBUS1:ANALYZE:RPOINT MANAUL,1
:ANALYSIS:SBUS1:ANALYZE:RPOINT?
-> :ANALYSIS:SBUS1:ANALYZE:
RPOINT MANUAL,1.00000E+00
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus?

Function Queries all settings related to the SPI bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CLOCk?

Function Queries all settings related to the clock channel of the SPI bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CLOCk?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock channel of the SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CLOCk:POLarity {FALL|RISE}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CLOCk:POLarity?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:
CLOCK:POLARITY FALL
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CLOCK:
POLARITY? -> :ANALYSIS:SBUS1:ANALYZE:
SPIBUS:CLOCK:POLARITY FALL
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CLOCk:SOURce

Function Sets the clock channel of the SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CLOCk:SOURce {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CLOCk:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CLOCK:
SOURCE 1
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CLOCK:
SOURCE? -> :ANALYSIS:SBUS1:ANALYZE:
SPIBUS:CLOCK:SOURCE 1
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS?

Function Queries all settings related to the chip select channel of the SPI bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS:ACTive

Function Sets the active level of the chip select channel of the SPI bus signal analysis or queries the current setting.

Syntax:
```
ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS:
ACTive {HIGH|LOW}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS:
ACTive?
```
<x> = 1 or 2

Example:
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CS:
ACTIVE HIGH
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CS:
ACTIVE? -> :ANALYSIS:SBUS1:ANALYZE:
SPIBUS:CS:ACTIVE HIGH
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:CS:TRACe

Function Sets the chip select channel of the SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CS:TRACe {<NRf>|NONE}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:
CS:TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CS:
TRACE 1
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:CS:
TRACE? -> :ANALYSIS:SBUS1:ANALYZE:
SPIBUS:CS:TRACE 1
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>?

Function Queries all settings related to the data of the SPI bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>?
```
<x> of SBUS<x> = 1 or 2
<x> of DATA<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:ACTive

Function Sets the active level of the data of the SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:ACTive {HIGH|LOW}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:ACTive?
```
<x> of SBUS<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:DATA1:ACTIVE HIGH
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:DATA1:ACTIVE? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:DATA1:ACTIVE HIGH
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:TRACe

Function Sets the data channel of the SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:TRACe {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:DATA<x>:TRACe?
```
<x> of SBUS<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:DATA1:TRACE 1
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:DATA1:TRACE? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:DATA1:TRACE 1
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus:SETup?

Function Queries all settings related to the SPI bus signal analysis setup.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus:SETup?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:BITorder

Function Sets the bit order of the SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:BITorder {LSBFirst|MSBFirst}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:BITorder?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:BITORDER LSBFIRST
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:BITORDER? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:BITORDER LSBFIRST
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:EMSBLSB

Function Sets the enabled range of the field used for SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:EMSBLSB {<NRf>,<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:EMSBLSB?
```
<x> = 1 or 2
<NRf> = See section 4.5.

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:EMSBLSB 1,7
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:EMSBLSB? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:EMSBLSB 1,7
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:FSIZe

Function Sets the field size used for SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:FSIZe {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:FSIZe?
```
<x> = 1 or 2
<NRf> = 4 to 32

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:FSIZE 4
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:FSIZE? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:FSIZE 4
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:ITIMe

Function Sets the idle time used in SPI bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:ITIMe {<Time>|DONTcare}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:ITIMe?
```
<x> = 1 or 2
<Time> = 10ns to 1ms in 10-ns steps

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:ITIME 10NS
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:ITIME? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:ITIME 10.0000E-09
```

:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:MODE

Function Sets the wiring system of the SPI bus signal analysis (three-wire or four-wire) or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:MODE {WIRE3|WIRE4}
:ANALysis:SBUS<x>[:ANALyze]:SPIBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:MODE WIRE3
:ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:MODE? -> :ANALYSIS:SBUS1:ANALYZE:SPIBUS:SETUP:MODE WIRE3
```

:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>?

Function Queries all settings related to the threshold level of the source channel of the serial bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>?
```
<x> of SBUS<x> = 1 or 2
<x> of TRACe<x> = 1 to 8

:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:HYSTeresis

Function Sets the hysteresis of the threshold level of the source channel of the serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:HYSTeresis {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:HYSTeresis?
```
<x> of SBUS<x> = 1 or 2
<x> of TRACe<x> = 1 to 8
<NRf> = 0 to 4 (div)

Example
```
:ANALYSIS:SBUS1:ANALYZE:TRACE1:HYSTERESIS 1
:ANALYSIS:SBUS1:ANALYZE:TRACE1:HYSTERESIS? -> :ANALYSIS:SBUS1:ANALYZE:TRACE1:HYSTERESIS 1.000E+00
```

:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:LEVel

Function Sets the level of the threshold level of the source channel of the serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:LEVel {<NRf>|<Voltage>|<current>}
:ANALysis:SBUS<x>[:ANALyze]:TRACe<x>:LEVel?
```
<x> of SBUS<x> = 1 or 2
<x> of TRACe<x> = 1 to 8
<NRf>, <Voltage>, and <Current> = See sections 4.2 to 4.6.

Example
```
:ANALYSIS:SBUS1:ANALYZE:TRACE1:LEVEL 1V
:ANALYSIS:SBUS1:ANALYZE:TRACE1:LEVEL? -> :ANALYSIS:SBUS1:ANALYZE:TRACE1:LEVEL 1.000E+00
```

:ANALysis:SBUS<x>[:ANALyze]:UART?

Function Queries all settings related to the UART bus signal analysis.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART?
```
<x> = 1 or 2

:ANALysis:SBUS<x>[:ANALyze]:UART:BITorder

Function Sets the UART bus signal analysis bit order or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
BITorder {LSBFirst|MSBFirst}
:ANALysis:SBUS<x>[:ANALyze]:UART:
BITorder?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
BITORDER LSBFIRST
:ANALYSIS:SBUS1:ANALYZE:UART:BITORDER?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
BITORDER LSBFIRST
```

:ANALysis:SBUS<x>[:ANALyze]:UART:BRATe

Function Sets the UART bus signal analysis bit rate (data transfer rate) or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
BRATe {<NRf>|USER,<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:UART:
BRATe?
```
<x> = 1 or 2

<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, or 115200

<NRf> of USER =See section 4.6.

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
BRATE 19200
:ANALYSIS:SBUS1:ANALYZE:UART:BRATE?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
BRATE 19200
```

:ANALysis:SBUS<x>[:ANALyze]:UART:BSPace

Function Sets the byte space for grouping data that is used in UART signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:BSPace
{<Time>}
:ANALysis:SBUS<x>[:ANALyze]:UART:
BSPace?
```
<x> = 1 or 2

<Time> = See section 4.6.

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
BSPACE 10ms
:ANALYSIS:SBUS1:ANALYZE:UART:BSPACE?
-> :ANALYSIS:SBUS1:ANALYZE:UART:BSPACE
10.00E-03
```

:ANALysis:SBUS<x>[:ANALyze]:UART:DFORmat

Function Sets the decoded character display format for UART signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
DFORmat {ASCii|HEXA}
:ANALysis:SBUS<x>[:ANALyze]:UART:
DFORmat?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
DFORMAT ASCII
:ANALYSIS:SBUS1:ANALYZE:UART:DFORMAT?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
DFORMAT ASCII
```

:ANALysis:SBUS<x>[:ANALyze]:UART:FORMat

Function Sets the UART bus signal analysis data format or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
FORMat {BIT7parity|BIT8Noparity|
BIT8Parity}
:ANALysis:SBUS<x>[:ANALyze]:UART:
FORMat?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
FORMAT BIT7PARITY
:ANALYSIS:SBUS1:ANALYZE:UART:FORMAT?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
FORMAT BIT7PARITY
```

:ANALysis:SBUS<x>[:ANALyze]:UART:GROuping

Function Turns on or off the grouping feature for UART signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
GROuping {<Boolean>}
:ANALysis:SBUS<x>[:ANALyze]:UART:
GROuping?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
GROUPING ON
:ANALYSIS:SBUS1:ANALYZE:UART:GROUPING?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
GROUPING 1
```

:ANALysis:SBUS<x>[:ANALyze]:UART:PMODe

Function Sets the UART bus signal analysis parity mode or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
PMODe {EVEN|ODD}
:ANALysis:SBUS<x>[:ANALyze]:UART:
PMODe?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:PMODE EVEN
:ANALYSIS:SBUS1:ANALYZE:UART:PMODE?
-> :ANALYSIS:SBUS1:ANALYZE:UART:PMODE
EVEN
```

:ANALysis:SBUS<x>[:ANALyze]:UART:POLarity

Function Sets the UART bus signal analysis polarity or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
POLarity {NEGative|POSitive}
:ANALysis:SBUS<x>[:ANALyze]:UART:
POLarity?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
POLARITY NEGATIVE
:ANALYSIS:SBUS1:ANALYZE:UART:POLARITY?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
POLARITY NEGATIVE
```

:ANALysis:SBUS<x>[:ANALyze]:UART:SPOint

Function Sets the UART bus signal analysis sample point or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
SPOint {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:UART:
SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:
SPOINT 18.8
:ANALYSIS:SBUS1:ANALYZE:UART:SPOINT?
-> :ANALYSIS:SBUS1:ANALYZE:UART:
SPOINT 18.8E+00
```

:ANALysis:SBUS<x>[:ANALyze]:UART:TRACe

Function Sets the UART bus signal analysis trace or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>[:ANALyze]:UART:
TRACe {<NRf>}
:ANALysis:SBUS<x>[:ANALyze]:UART:
TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:ANALYSIS:SBUS1:ANALYZE:UART:TRACE 1
:ANALYSIS:SBUS1:ANALYZE:UART:TRACE?
-> :ANALYSIS:SBUS1:ANALYZE:UART:TRACE 1
```

:ANALysis:SBUS<x>:ZLINkage

Function Sets the zoom link of the serial bus signal analysis or queries the current setting.

Syntax
```
:ANALysis:SBUS<x>:ZLINkage {OFF|Z1|Z2}
:ANALysis:SBUS<x>:ZLINkage?
```
<x> = 1 or 2

Example
```
:ANALYSIS:SBUS1:ZLINKAGE OFF
:ANALYSIS:SBUS1:ZLINKAGE?
-> :ANALYSIS:SBUS1:ZLINKAGE OFF
```

:ANALysis:TYPE<x>

Function Sets the analysis feature type or queries the current setting.

Syntax
```
:ANALysis:TYPE<x> {AHIStogram|CANBus|FF
T|HARMonics|I2CBus|LINBus|SPIBus|UART|
WPARameter|XY|OFF}
:ANALysis:TYPE<x>?
```
<x> = 1 or 2

Example
```
:ANALYSIS:TYPE1 AHISTOGRAM
:ANALYSIS:TYPE1?
-> :ANALYSIS:TYPE1 AHISTOGRAM
```

Description {HARMonics|LSBus} can be applied to DLM6000.

7.3 SEARch Group

SEARch<x>:CANBus?

Function Queries all settings related to the CAN bus signal search.

Syntax `:SEARch<x>:CANBus?`

<x> = 1 or 2

:SEARch<x>:CANBus:SETup?

Function Queries all settings related to the CAN bus signal search setup.

Syntax `:SEARch<x>:CANBus:SETup?`

<x> = 1 or 2

:SEARch<x>:CANBus[:SETup]:ACK

Function Sets the ACK condition of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:
ACK {ACK|ACKBoth|DONTcare|NONack}
:SEARch<x>:CANBus[:SETup]:ACK?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:ACK ACK
:SEARCH1:CANBUS:SETUP:ACK?
-> :SEARCH1:CANBUS:SETUP:ACK ACK
```

:SEARch<x>:CANBus[:SETup]:BRATe

Function Sets the bit rate (data transfer rate) of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:
BRATe {<NRf>|USER,<NRf>}
:SEARch<x>:CANBus[:SETup]:BRATe?
```
<x> = 1 or 2

<NRf> =33300, 83300, 125000, 250000, 500000, 1000000

<NRf> of USER = See section 5.3.

Example
```
:SEARCH1:CANBUS:SETUP:BRATE 83300
:SEARCH1:CANBUS:SETUP:BRATE?
-> :SEARCH1:CANBUS:SETUP:BRATE 83300
```

:SEARch<x>:CANBus[:SETup]:DATA?

Function Queries all settings related to the CAN bus signal search data.

Syntax `:SEARch<x>:CANBus[:SETup]:DATA?`

<x> = 1 or 2

:SEARch<x>:CANBus[:SETup]:DATA:BORDer

Function Sets the byte order of the CAN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
BORDer {BIG|LITTle}
:SEARch<x>:CANBus[:SETup]:DATA:BORDer?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:DATA:BORDER BIG
:SEARCH1:CANBUS:SETUP:DATA:BORDER?
-> :SEARCH1:CANBUS:SETUP:DATA:
BORDER BIG
```

:SEARch<x>:CANBus[:SETup]:DATA:CONDition

Function Sets the data condition of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
CONDition {BETWeen|DONTcare|FALSe|
GTHan|LTHan|ORANge|TRUE}
:SEARch<x>:CANBus[:SETup]:DATA:
CONDition?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:DATA:
CONDITION BETWEEN
:SEARCH1:CANBUS:SETUP:DATA:CONDITION?
-> :SEARCH1:CANBUS:SETUP:DATA:
CONDITION BETWEEN
```

:SEARch<x>:CANBus[:SETup]:DATA:DATA<x>

Function Sets the comparison data of the CAN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
DATA<x> {<NRf>}
:SEARch<x>:CANBus[:SETup]:DATA:
DATA<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = See section 5.3.

Example
```
:SEARCH1:CANBUS:SETUP:DATA:DATA1 1
:SEARCH1:CANBUS:SETUP:DATA:DATA1?
-> :SEARCH1:CANBUS:SETUP:DATA:
DATA1 1.0000000E+00
```

Description
- Use `:SEARch<x>:CANBus[:SETup]:DATA:DATA1` when `:SEARch<x>:CANBus[:SETup]:DATA:CONDition GTHan` is specified.
- Use `:SEARch<x>:CANBus[:SETup]:DATA:DATA2` when `:SEARch<x>:CANBus[:SETup]:DATA:CONDition LTHan` is specified.
- Use `:SEARch<x>:CANBus[:SETup]:DATA:DATA1` to set the smaller value and `:SEARch<x>:CANBus[:SETup]:DATA:DATA2` to set the larger value when `:SEARch<x>:CANBus[:SETup]:DATA:CONDition BETWeen|ORANge` is specified.

:SEARch<x>:CANBus[:SETup]:DATA:DLC

Function Sets the number of valid bytes (DLC) of the CAN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
DLC {<NRf>}
:SEARch<x>:CANBus[:SETup]:DATA:DLC?
```
<x> = 1 or 2
<NRf> = 0 to 8

Example
```
:SEARCH1:CANBUS:SETUP:DATA:DLC 0
:SEARCH1:CANBUS:SETUP:DATA:DLC?
-> :SEARCH1:CANBUS:SETUP:DATA:DLC 0
```

:SEARch<x>:CANBus[:SETup]:DATA:HEXA

Function Sets the CAN bus signal search data in hexadecimal notation.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
HEXA {<String>}
```
<x> = 1 or 2
<String> = Up to 16 characters by combining '0' to 'F' and 'X' (in one-byte unit)

Example
```
:SEARCH1:CANBUS:SETUP:DATA:HEXA "A9"
```

:SEARch<x>:CANBus[:SETup]:DATA:MSBLsb

Function Sets the MSB and LSB bits of the CAN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
MSBLsb {<NRf>,<NRf>}
:SEARch<x>:CANBus[:SETup]:DATA:MSBLsb?
```
<x> = 1 or 2
<NRf> = See section 5.3.

Example
```
:SEARCH1:CANBUS:SETUP:DATA:MSBLSB 1,0
:SEARCH1:CANBUS:SETUP:DATA:MSBLSB?
-> :SEARCH1:CANBUS:SETUP:DATA:
MSBLSB 1,0
```

:SEARch<x>:CANBus[:SETup]:DATA:PATTern

Function Sets the CAN bus signal search data in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
PATTern {<String>}
:SEARch<x>:CANBus[:SETup]:DATA:
PATTern?
```
<x> = 1 or 2
<String> = Up to 64 characters by combining '0,' '1,' and 'X' (in one-byte unit)

Example
```
:SEARCH1:CANBUS:SETUP:DATA:
PATTERN "11011111"
:SEARCH1:CANBUS:SETUP:DATA:PATTERN?
-> :SEARCH1:CANBUS:SETUP:DATA:
PATTERN "11011111"
```

:SEARch<x>:CANBus[:SETup]:DATA:SIGN

Function Sets the sign of the CAN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:DATA:
SIGN {SIGN|UNSign}
:SEARch<x>:CANBus[:SETup]:DATA:SIGN?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:DATA:SIGN SIGN
:SEARCH1:CANBUS:SETUP:DATA:SIGN?
-> :SEARCH1:CANBUS:SETUP:DATA:SIGN SIGN
```

:SEARch<x>:CANBus[:SETup]:IDEXt?

Function Queries all settings related to the ID of the extended format of the CAN bus signal search.

Syntax
```
:SEARch<x>:CANBus[:SETup]:IDEXt?
```
<x> = 1 or 2

:SEARch<x>:CANBus[:SETup]:IDEXt:HEXA

Function Sets the ID of the extended format of the CAN bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:CANBus[:SETup]:IDEXt:
HEXA {<String>}
```
<x> = 1 or 2
<String> = 8 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:CANBUS:SETUP:IDEXT:
HEXA "1AEF5906"
```

:SEARch<x>:CANBus[:SETup]:IDEXt:PATTern

Function Sets the ID of the extended format of the CAN bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:IDEXt:
PATTern {<String>}
:SEARch<x>:CANBus[:SETup]:IDEXt:
PATTern?
```
<x> = 1 or 2
<String> = 29 characters by combining '0,' '1,' and 'X'

Example
```
:SEARCH1:CANBUS:SETUP:IDEXT:
PATTERN "11001011011100001110111011111"
:SEARCH1:CANBUS:SETUP:IDEXT:PATTERN?
-> :SEARCH1:CANBUS:SETUP:IDEXT:
PATTERN "11001011011100001110111011111"
```

:SEARch<x>:CANBus[:SETup]:IDSTd?

Function Queries all settings related to the ID of the standard format of the CAN bus signal search.

Syntax
```
:SEARch<x>:CANBus[:SETup]:IDSTd?
```
<x> = 1 or 2

:SEARch<x>:CANBus[:SETup]:IDSTd:HEXA

Function Sets the ID of the standard format of the CAN bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:CANBus[:SETup]:IDSTd:
HEXA {<String>}
```
<x> = 1 or 2
<String> = 3 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:CANBUS:SETUP:IDSTD:HEXA "5DF"
```

:SEARch<x>:CANBus[:SETup]:IDSTd:PATTern

Function Sets the ID of the standard format of the CAN bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:IDSTd:
PATTern {<String>}
:SEARch<x>:CANBus[:SETup]:IDSTd:
PATTern?
```
<x> = 1 or 2
<String> = 11 characters by combining '0,' '1,' and 'X'

Example
```
:SEARCH1:CANBUS:SETUP:IDSTD:
PATTERN "10111011111"
:SEARCH1:CANBUS:SETUP:IDSTD:PATTERN?
-> :SEARCH1:CANBUS:SETUP:IDSTD:
PATTERN "10111011111"
```

:SEARch<x>:CANBus[:SETup]:MODE

Function Sets the CAN bus signal search mode or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:
MODE {EFRame|IDEXt|IDSTd|SOF}
:SEARch<x>:CANBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:MODE EFRAME
:SEARCH1:CANBUS:SETUP:MODE?
-> :SEARCH1:CANBUS:SETUP:MODE EFRAME
```

:SEARch<x>:CANBus[:SETup]:RECessive

Function Sets the recessive level (bus level) of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:
RECessive {HIGH|LOW}
:SEARch<x>:CANBus[:SETup]:RECessive?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:RECESSIVE HIGH
:SEARCH1:CANBUS:SETUP:RECESSIVE?
-> :SEARCH1:CANBUS:SETUP:RECESSIVE HIGH
```

:SEARch<x>:CANBus[:SETup]:RTR

Function Sets the RTR of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:
RTR {DATA|DONTcare|REMote}
:SEARch<x>:CANBus[:SETup]:RTR?
```
<x> = 1 or 2

Example
```
:SEARCH1:CANBUS:SETUP:RTR DATA
:SEARCH1:CANBUS:SETUP:RTR?
-> :SEARCH1:CANBUS:SETUP:RTR DATA
```

:SEARch<x>:CANBus[:SETup]:SPOint

Function Sets the sample point of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:
SPOint {<NRf>}
:SEARch<x>:CANBus[:SETup]:SPOint?
```
<x> = 1 or 2
<NRf>=18.8 to 90.6(%)

Example
```
:SEARCH1:CANBUS:SETUP:SPOINT 18.8
:SEARCH1:CANBUS:SETUP:SPOINT?
-> :SEARCH1:CANBUS:SETUP:
SPOINT 18.8E+00
```

:SEARch<x>:CANBus[:SETup]:TRACe

Function Sets the trace of the CAN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:CANBus[:SETup]:TRACe {<NRf>}
:SEARch<x>:CANBus[:SETup]:TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:SEARCH1:CANBUS:SETUP:TRACE 1
:SEARCH1:CANBUS:SETUP:TRACE?
-> :SEARCH1:CANBUS:SETUP:TRACE 1
```

SEARch<x>:I2CBus?

Function Queries all settings related to the I^2C bus signal search.

Syntax
```
:SEARch<x>:I2CBus?
```
<x> = 1 or 2

:SEARch<x>:I2CBus:CLOCk?

Function Queries all settings related to the clock of the I^2C bus signal search.

Syntax
```
:SEARch<x>:I2CBus:CLOCk?
```
<x> = 1 or 2

:SEARch<x>:I2CBus:CLOCk:SOURce

Function Sets the clock trace of the I^2C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus:CLOCk:SOURce {<NRf>}
:SEARch<x>:I2CBus:CLOCk:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:SEARCH1:I2CBUS:CLOCK:SOURCE 1
:SEARCH1:I2CBUS:CLOCK:SOURCE?
-> :SEARCH1:I2CBUS:CLOCK:SOURCE 1
```

:SEARch<x>:I2CBus:SETup?

Function Queries all settings related to the I^2C bus signal search setup.

Syntax
```
:SEARch<x>:I2CBus:SETup?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa?

Function Queries all settings related to the address of the I^2C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT10address?

Function Queries all settings related to the 10-bit address of the I^2C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT10address?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT10address:HEXA

Function Sets the 10-bit address of the I^2C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT10address:HEXA {<String>}
```
<x> = 1 or 2
<String> = 3 characters by combining '0' to 'F' and 'X' (bit 8 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:HEXA" 5DF"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT10address:PATTern

Function Sets the 10-bit address of the I^2C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT10address:PATTern {<String>}
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT10address:PATTern?
```
<x> = 1 or 2
<String> = 11 characters by combining '0', '1', and 'X' (bit 8 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:PATTERN" 10111011111"
:SEARCH1:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:PATTERN?
-> :SEARCH1:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:PATTERN" 10111011111"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress?

Function Queries all settings related to the 7-bit address of the I^2C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT7ADdress?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:HEXA

Function Sets the 7-bit address of the I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7ADDRESS:HEXA " DE"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:PATTern

Function Sets the 7-bit address of the I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:PATTern {<String>}
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7ADdress:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0', '1', and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7ADDRESS:PATTERN " 11011110"
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7ADDRESS:PATTERN?
-> :SEARCH1:I2CBUS:SETUP:ADATA:BIT7ADDRESS:PATTERN " 11011110"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub?

Function Queries all settings related to the 7-bit + Sub address of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + Sub address of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:HEXA

Function Sets the 7-bit address of the 7-bit + Sub address of the I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:ADDRESS:HEXA " CD"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:PATTern

Function Sets the 7-bit address of the 7-bit + Sub address of the I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:PATTern {<String>}
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0', '1', and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:ADDRESS:PATTERN " 11001101"
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:ADDRESS:PATTERN?
-> :SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:ADDRESS:PATTERN " 11001101"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress?

Function Queries all settings related to the Sub address of the 7-bit + Sub address of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:HEXA

Function Sets the Sub address of the 7-bit + Sub address of the I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:SADDRESS:HEXA " EF"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:PATTern

Function Sets the Sub address of the 7-bit + Sub address of the I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT7APsub:SADDress:PATTern {<String>}
:SEARch<x>:I2CBus[:SETup]:ADATa:
BIT7APsub:SADDress:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:
SADDRESS:PATTERN " 11101111"
:SEARCH1:I2CBUS:SETUP:ADATA:BIT7APSUB:
SADDRESS:PATTERN?
-> :SEARCH1:I2CBUS:SETUP:ADATA:
BIT7APSUB:SADDRESS:PATTERN " 11101111"
```

:SEARch<x>:I2CBus[:SETup]:ADATa:TYPE

Function Sets the address type of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:ADATa:
TYPE {BIT10address|BIT7ADdress|
BIT7APsub}
:SEARch<x>:I2CBus[:SETup]:ADATa:TYPE?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:ADATA:
TYPE BIT10ADDRESS
:SEARCH1:I2CBUS:SETUP:ADATA:TYPE?
-> :SEARCH1:I2CBUS:SETUP:ADATA:
TYPE BIT10ADDRESS
```

:SEARch<x>:I2CBus[:SETup]:DATA?

Function Queries all settings related to the data of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:DATA:BYTE

Function Sets the number of data bytes of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:
BYTE {<NRf>}
:SEARch<x>:I2CBus[:SETup]:DATA:BYTE?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:BYTE 1
:SEARCH1:I2CBUS:SETUP:DATA:BYTE?
-> :SEARCH1:I2CBUS:SETUP:DATA:BYTE 1
```

:SEARch<x>:I2CBus[:SETup]:DATA:CONDition

Function Sets the determination method (match or not match) of the data of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:
CONDition {FALSe|TRUE}
:SEARch<x>:I2CBus[:SETup]:DATA:
CONDition?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:
CONDITION TRUE
:SEARCH1:I2CBUS:SETUP:DATA:
CONDITION?
-> :SEARCH1:I2CBUS:SETUP:DATA:
CONDITION TRUE
```

:SEARch<x>:I2CBus[:SETup]:DATA:DPOSition

Function Sets the position for comparing the data pattern of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:
DPOSition {<NRf>}
:SEARch<x>:I2CBus[:SETup]:DATA:
DPOSition?
```
<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:DPOSITION 1
:SEARCH1:I2CBUS:SETUP:DATA:
DPOSITION?
-> :SEARCH1:I2CBUS:SETUP:DATA:
DPOSITION 1
```

:SEARch<x>:I2CBus[:SETup]:DATA:HEXA<x>

Function Sets the data of the I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:
HEXA<x> {<String>}
```
<x> of SEARch<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:HEXA1 " AB"
```

:SEARch<x>:I2CBus[:SETup]:DATA:MODE

Function Enables/Disables the data conditions of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:
MODE {<Boolean>}
:SEARch<x>:I2CBus[:SETup]:DATA:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:MODE ON
:SEARCH1:I2CBUS:SETUP:DATA:MODE?
-> :SEARCH1:I2CBUS:SETUP:DATA:MODE 1
```

:SEARch<x>:I2CBus[:SETup]:DATA:PATTern<x>

Function Sets the data of the I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:PATTern<x> {<String>}
:SEARch<x>:I2CBus[:SETup]:DATA:PATTern<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of <PATTern x> = 1 to 4
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:PATTERN1 " 10101011"
:SEARCH1:I2CBUS:SETUP:DATA:PATTERN1?
-> :SEARCH1:I2CBUS:SETUP:DATA:PATTERN1 " 10101011"
```

:SEARch<x>:I2CBus[:SETup]:DATA:PMODe

Function Sets the pattern comparison start position mode of the data of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:PMODe {DONTcare|SELect}
:SEARch<x>:I2CBus[:SETup]:DATA:PMODe?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:DATA:PMODE DONTCARE
:SEARCH1:I2CBUS:SETUP:DATA:PMODE?
-> :SEARCH1:I2CBUS:SETUP:DATA:PMODE DONTCARE
```

:SEARch<x>:I2CBus[:SETup]:DATA:TRACe

Function Sets the trace of the data of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:DATA:TRACe {<NRf>}
:SEARch<x>:I2CBus[:SETup]:DATA:TRACe?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:DATA::TRACe 1
:SEARCH1:I2CBUS:SETUP:DATA:TRACe?
-> :SEARCH1:I2CBUS:SETUP:DATA::TRACe 1
```

:SEARch<x>:I2CBus[:SETup]:GCALl?

Function Queries all settings related to the general call of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:GCALl?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress?

Function Queries all settings related to the 7-bit master address of the general call of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:HEXA

Function Sets the 7-bit master address of the general call of the I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is fixed 1)

Example
```
:SEARCH1:I2CBUS:SETUP:GCALL:BIT7MADDRESS:HEXA " BA"
```

:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:PATTern

Function Sets the 7-bit master address of the general call of the I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:PATTern {<String>}
:SEARch<x>:I2CBus[:SETup]:GCALl:BIT7maddress:PATTern?
```
<x> = 1 or 2
<String> = 7 characters by combining '0,' '1,' and 'X'

Example
```
:SEARCH1:I2CBUS:SETUP:GCALL:BIT7MADDRESS:PATTERN " 1010101"
:SEARCH1:I2CBUS:SETUP:GCALL:BIT7MADDRESS:PATTERN?
-> :SEARCH1:I2CBUS:SETUP:GCALL:BIT7MADDRESS:PATTERN " 1010101"
```

:SEARch<x>:I2CBus[:SETup]:GCALl:SBYTe (Second Byte)

Function Sets the second byte type of the general call of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:GCALl:SBYTe {BIT7maddress|DONTcare|H04|H06}
:SEARch<x>:I2CBus[:SETup]:GCALl:SBYTe?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:GCALL:SBYTE BIT7MADDRESS
:SEARCH1:I2CBUS:SETUP:GCALL:SBYTE?
-> :SEARCH1:I2CBUS:SETUP:GCALL:SBYTE BIT7MADDRESS
```

:SEARch<x>:I2CBus[:SETup]:MODE

Function Sets the search mode of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:
MODE {ADATa|ESTart|GCALl|NAIGnore|
SBHSmode}
:SEARch<x>:I2CBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:MODE ADATA
:SEARCH1:I2CBUS:SETUP:MODE?
-> :SEARCH1:I2CBUS:SETUP:MODE ADATA
```

:SEARch<x>:I2CBus[:SETup]:NAIGnore?

Function Queries all settings related to the NON ACK ignore mode of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:NAIGnore?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:NAIGnore:HSMode

Function Sets whether to ignore NON ACK in high speed mode of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:NAIGnore:
HSMode {<Boolean>}
:SEARch<x>:I2CBus[:SETup]:NAIGnore:
HSMode?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:NAIGNORE:
HSMODE ON
:SEARCH1:I2CBUS:SETUP:NAIGNORE:HSMODE?
-> :SEARCH1:I2CBUS:SETUP:NAIGNORE:
HSMODE 1
```

:SEARch<x>:I2CBus[:SETup]:NAIGnore:RACCess

Function Sets whether to ignore NON ACK in read access mode of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:NAIGnore:
RACCess {<Boolean>}
:SEARch<x>:I2CBus[:SETup]:NAIGnore:
RACCess?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:NAIGNORE:
RACCESS ON
:SEARCH1:I2CBUS:SETUP:NAIGNORE:
RACCESS?
-> :SEARCH1:I2CBUS:SETUP:NAIGNORE:
RACCESS 1
```

:SEARch<x>:I2CBus[:SETup]:NAIGnore:SBYTe(Start Byte)

Function Sets whether to ignore NON ACK in the start byte of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:NAIGnore:
SBYTe {<Boolean>}
:SEARch<x>:I2CBus[:SETup]:NAIGnore:
SBYTe?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:NAIGNORE:SBYTE ON
:SEARCH1:I2CBUS:SETUP:NAIGNORE:
SBYTE?
-> :SEARCH1:I2CBUS:SETUP:NAIGNORE:
SBYTE 1
```

:SEARch<x>:I2CBus[:SETup]:SBHSmode?

Function Queries all settings related to the start byte and high speed mode of the I²C bus signal search.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:SBHSmode?
```
<x> = 1 or 2

:SEARch<x>:I2CBus[:SETup]:SBHSmode:TYPE

Function Sets the type of the start byte or high speed mode of the I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:I2CBus[:SETup]:SBHSmode:
TYPE {HSMode|SBYTe}
:SEARch<x>:I2CBus[:SETup]:SBHSmode:
TYPE?
```
<x> = 1 or 2

Example
```
:SEARCH1:I2CBUS:SETUP:SBHSMODE:
TYPE HSMODE
:SEARCH1:I2CBUS:SETUP:SBHSMODE:TYPE?
-> :SEARCH1:I2CBUS:SETUP:SBHSMODE:
TYPE HSMODE
```

:SEARch<x>:LINBus?

Function Queries all settings related to the LIN bus signal search or queries the current setting.

Syntax
```
SEARch<x>:LINBus?
```
<x>=1 or 2

:SEARch<x>:LINBus[:SETup]?

Function Queries all settings related to setup of the LIN bus signal search or queries the current setting.

Syntax
```
SEARch<x>:LINBus[:SETup]?
```
<x>=1 or 2

:SEARch<x>:LINBus[:SETup]:BLENgth

Function Sets the LIN bus signal search break length or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:
BLENgth {<NRf>}
:SEARch<x>:LINBus[:SETup]:BLENgth?
```
<x> = 1 or 2
<NRf> = 10 to 13

Example
```
:SEARCH1:LINBUS:SETUP:BLENGTH 10
:SEARCH1:LINBUS:SETUP:BLENGTH?
-> :SEARCH1:LINBUS:SETUP:BLENGTH 1
```

:SEARch<x>:LINBus[:SETup]:BRATe

Function Sets the LIN bus signal search bitrate (data transfer rate) or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:BRATe {<NRf>|
USER,<NRf>}
:SEARch<x>:LINBus[:SETup]:BRATe?
```
<x>=1 or 2
<NRf>=1200, 2400, 4800, 9600, 19200
USER <NRf>=See section 5.4.

Example
```
:SEARCH1:LINBUS:SETUP:BRATE 19200
:SEARCH1:LINBUS:SETUP:BRATE?
-> :SEARCH1:LINBUS:SETUP:BRATE 19200
```

:SEARch<x>:LINBus[:SETup]:DATA?

Function Queries all settings related to data of the LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA?
```
<x>=1 or 2

:SEARch<x>:LINBus[:SETup]:DATA:BNUM

Function Sets the number of LIN bus signal search data bytes or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
BNUM {<NRf>}
:SEARch<x>:LINBus[:SETup]:DATA:BNUM?
```
<x>=1 or 2
<NRf>=1–8

Example
```
:SEARCH1:LINBUS:SETUP:DATA:BNUM 1
:SEARCH1:LINBUS:SETUP:DATA:BNUM?
-> :SEARCH1:LINBUS:SETUP:DATA:BNUM 1
```

:SEARch<x>:LINBus[:SETup]:DATA:BORDer

Function Sets the data byte order of the LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
BORDer {BIG|LITTle}
:SEARch<x>:LINBus[:SETup]:DATA:BORDer?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:DATA:BORDER BIG
:SEARCH1:LINBUS:SETUP:DATA:BORDER? ->
:SEARCH1:LINBUS:SETUP:DATA:BORDER BIG
```

:SEARch<x>:LINBus[:SETup]:DATA:CONDition

Function Sets the LIN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
CONDition {BETWeen|DONTcare|FALSe|
GTHan|LTHan|ORANge|TRUE}
:SEARch<x>:LINBus[:SETup]:DATA:
CONDition?
```
<x>=1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:DATA:
CONDITION DONTCARE
:SEARCH1:LINBUS:SETUP:DATA:CONDITION?
-> :SEARCH1:LINBUS:SETUP:DATA:
CONDITION DONTCARE
```

:SEARch<x>:LINBus[:SETup]:DATA:DATA<x>

Function Sets the comparison data of the LIN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
DATA<x> {<NRf>}
:SEARch<x>:LINBus[:SETup]:DATA:
DATA<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = See section 5.4.

Example
```
:SEARCH1:LINBUS:SETUP:DATA:DATA1 1
:SEARCH1:LINBUS:SETUP:DATA:DATA1?
-> :SEARCH1:LINBUS:SETUP:DATA:
DATA1 1.0000000E+00
```

Description
- For :SEARch<x>:LINBus[:SETup]:DATA:CONDition GTHan, set using: SEARch<x>:LINBus[:SETup]:DATA:DATA1.
- For :SEARch<x>:LINBus[:SETup]:DATA:CONDition LTHan, set using: SEARch<x>:LINBus[:SETup]:DATA:DATA2.
- For :SEARch<x>:LINBus[:SETup]:DATA:CONDition BETWeen|ORANge, set small values with: SEARch<x>:LINBus[:SETup]:DATA:DATA1, and large values with: SEARch<x>:LINBus[:SETup]:DATA:DATA2.

:SEARch<x>:LINBus[:SETup]:DATA:HEXA

Function Sets the LIN bus signal search data in hexadecimal.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
HEXA {<string>}
```
<x>=1 or 2
<string>=Combination of up to 16 hex characters (‘0’ – ‘F’ and ‘X’) (changed with the BNUM setting)

Example
```
:SEARCH1:LINBUS:SETUP:DATA:HEXA "3B"
```

:SEARch<x>:LINBus[:SETup]:DATA:MSBLsb

Function Sets the MSB/LSB bit of the LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
MSBLsb {<NRf>,<NRf>}
:SEARch<x>:LINBus[:SETup]:DATA:MSBLsb?
```
<x> = 1 or 2
<NRf> = See section 5.4.

Example
```
:SEARCH1:LINBUS:SETUP:DATA:MSBLSB 1,0
:SEARCH1:LINBUS:SETUP:DATA:MSBLSB? ->
:SEARCH1:LINBUS:SETUP:DATA:MSBLSB 1,0
```

:SEARch<x>:LINBus[:SETup]:DATA:PATTern

Function Sets the LIN bus signal search data in binary or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
PATTern {<string>}
:SEARch<x>:LINBus[:SETup]:DATA:
PATTern?
```
<x>=1 or 2
<string>=Combination of up to 64 characters (‘0,’ ‘1,’ and ‘X’) (changed with the BNUM setting)

Example
```
:SEARCH1:LINBUS:SETUP:DATA:
PATTERN "11011111"
:SEARCH1:LINBUS:SETUP:DATA:PATTERN?
-> :SEARCH1:LINBUS:SETUP:DATA:
PATTERN "11011111"
```

:SEARch<x>:LINBus[:SETup]:DATA:SIGN

Function Sets the sign order of the LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:DATA:
SIGN {SIGN|UNSign}
:SEARch<x>:LINBus[:SETup]:DATA:SIGN?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:DATA:SIGN SIGN
:SEARCH1:LINBUS:SETUP:DATA:SIGN? ->
:SEARCH1:LINBUS:SETUP:DATA:SIGN SIGN
```

:SEARch<x>:LINBus[:SETup]:ERRor?

Function Queries all settings related to the LIN bus signal search error.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ERRor?
```
<x> = 1 or 2

:SEARch<x>:LINBus[:SETup]:ERRor:CHECksum

Function Sets the LIN bus signal search Checksum error or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ERRor:
CHECksum {<Boolean>}
:SEARch<x>:LINBus[:SETup]:ERRor:
CHECksum?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:ERROR:CHECKSUM ON
:SEARCH1:LINBUS:SETUP:ERROR:CHECKSUM?
-> :SEARCH1:LINBUS:SETUP:ERROR:
CHECKSUM 1
```

:SEARch<x>:LINBus[:SETup]:ERRor:FRAMing

Function Sets the LIN bus signal search Framing error or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ERRor:
FRAMing {<Boolean>}
:SEARch<x>:LINBus[:SETup]:ERRor:
FRAMing?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:ERROR:FRAMING ON
:SEARCH1:LINBUS:SETUP:ERROR:FRAMING? ->
:SEARCH1:LINBUS:SETUP:ERROR:FRAMING 1
```

:SEARch<x>:LINBus[:SETup]:ERRor:PARity

Function Sets the LIN bus signal search Parity error or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ERRor:
PARity {<Boolean>}
:SEARch<x>:LINBus[:SETup]:ERRor:
PARity?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:ERROR:PARITY ON
:SEARCH1:LINBUS:SETUP:ERROR:PARITY? ->
:SEARCH1:LINBUS:SETUP:ERROR:PARITY 1
```

:SEARch<x>:LINBus[:SETup]:ERRor:SYNCh

Function Sets the LIN bus signal search Synch error or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ERRor:
SYNCh {<Boolean>}
:SEARch<x>:LINBus[:SETup]:ERRor:SYNCh?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:ERROR:SYNCH ON
:SEARCH1:LINBUS:SETUP:ERROR:SYNCH? ->
:SEARCH1:LINBUS:SETUP:ERROR:SYNCH 1
```

:SEARch<x>:LINBus[:SETup]:ERRor:TOUT

Function Sets the LIN bus signal search Timeout error or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ERRor:
TOUT {<Boolean>}
:SEARch<x>:LINBus[:SETup]:ERRor:TOUT?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:ERROR:TOUT ON
:SEARCH1:LINBUS:SETUP:ERROR:TOUT? ->
:SEARCH1:LINBUS:SETUP:ERROR:TOUT 1
```

:SEARch<x>:LINBus[:SETup]:ID?

Function Queries all settings related to ID of the LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ID?
```
<x>=1 or 2

:SEARch<x>:LINBus[:SETup]:ID:HEXA

Function Sets the LIN bus signal search ID in hexadecimal.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ID:
HEXA {<string>}
```
<x>=1 or 2
<string>=Combination of up to 2 characters (‘0’ –‘F’ and ‘X’)

Example
```
:SEARCH1:LINBUS:SETUP:ID:HEXA "2A"
```

:SEARch<x>:LINBus[:SETup]:ID:PATTern

Function Sets the LIN bus signal search ID in binary or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:ID:
PATTern {<string>}
:SEARch<x>:LINBus[:SETup]:ID:PATTern?
```
<x>=1 or 2
<string>=Combination of up to 6 characters (‘0,’ ‘1,’ and ‘X’)

Example
```
:SEARCH1:LINBUS:SETUP:ID:
PATTERN "101111"
:SEARCH1:LINBUS:SETUP:ID:PATTERN?
-> :SEARCH1:LINBUS:SETUP:ID:
PATTERN "101111"
```

:SEARch<x>:LINBus[:SETup]:MODE

Function Sets the LIN bus signal search mode or queries the current setting.

Syntax
```
SEARch<x>:LINBus[:SETup]:MODE {IDData|
SYNCh}
:SEARch<x>:LINBus[:SETup]:MODE?
```
<x>=1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:MODE IDDATA
:SEARCH1:LINBUS:SETUP:MODE?
-> :SEARCH1:LINBUS:SETUP:MODE IDDATA
```

:SEARch<x>:LINBus[:SETup]:REVision

Function Sets the LIN bus signal search revision (1.3, 2.0, or Both) or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:
REVision {BOTH|LIN1_3|LIN2_0}
:SEARch<x>:LINBus[:SETup]:REVision?
```
<x> = 1 or 2

Example
```
:SEARCH1:LINBUS:SETUP:REVISION LIN1_3
:SEARCH1:LINBUS:SETUP:REVISION? ->
:SEARCH1:LINBUS:SETUP:REVISION LIN1_3
```

:SEARch<x>:LINBus[:SETup]:SPOint

Function Sets the LIN bus signal search sampling point or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:
SPOint {<NRf>}
:SEARch<x>:LINBus[:SETup]:SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:SEARCH1:LINBUS:SETUP:SPOINT 18.8
:SEARCH1:LINBUS:SETUP:SPOINT? ->
:SEARCH1:LINBUS:SETUP:SPOINT 18.8E+00
```

:SEARch<x>:LINBus[:SETup]:TRACe

Function Sets the LIN bus signal search trace or queries the current setting.

Syntax
```
:SEARch<x>:LINBus[:SETup]:TRACe {<NRf>}
:SEARch<x>:LINBus[:SETup]:TRACe?
```
<x>=1 or 2
<NRf>=1–8

Example
```
:SEARCH1:LINBUS:SETUP:TRACE 1
:SEARCH1:LINBUS:SETUP:TRACE?
-> :SEARCH1:LINBUS:SETUP:TRACE 1
```

:SEARch<x>:SLOGic:I2CBus?

Function Queries all settings related to the logic I²C bus signal search.

Syntax `:SEARch<x>:SLOGic:I2CBus?`

<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus:CLOCk?

Function Queries all settings related to the clock channel of the logic I²C bus signal search.

Syntax `:SEARch<x>:SLOGic:I2CBus:CLOCk?`

<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus:CLOCk:SOURce

Function Sets the clock channel of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus:CLOCk:SOURce
{A<y>|B<y>|C<y>|D<y>}
:SEARch<x>:SLOGic:I2CBus:CLOCk:SOURce?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:I2CBUS:CLOCK:SOURCE A0
:SEARCH1:SLOGIC:I2CBUS:CLOCK:
SOURCE? -> :SEARCH1:SLOGIC:I2CBUS:
CLOCK:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:SEARch<x>:SLOGic:I2CBus[:SETup]?

Function Queries all settings related to the setup of the logic I²C bus signal search.

Syntax `:SEARch<x>:SLOGic:I2CBus[:SETup]?`

<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa?

Function Queries all settings related to the address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT10address?

Function Queries all settings related to the 10-bit address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT10address?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT10address:HEXA

Function Sets the 10-bit address of the logic I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT10address:HEXA {<String>}
```
<x> = 1 or 2

<String> = 3 characters by combining '0' to 'F' and 'X' (bit 8 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:HEXA " 5DF"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT10address:PATTern

Function Sets the 10-bit address of the logic I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT10address:PATTern {<String>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT10address:PATTern?
```
<x> = 1 or 2

<String> = 11 characters by combining '0' to '1' and 'X' (bit 8 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:PATTERN " 10111011111"
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:PATTERN? -> :SEARCH1:
SLOGIC:I2CBUS:SETUP:ADATA:
BIT10ADDRESS:
PATTERN " 10111011111"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7ADdress?

Function Queries all settings related to the 7-bit address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:BIT7ADdress?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7ADdress:HEXA

Function Sets the 7-bit address of the logic I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7ADdress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7ADDRESS:HEXA" DE"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7ADdress:PATTern

Function Sets the 7-bit address of the logic I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7ADdress:PATTern {<String>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7ADdress:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0' to '1' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7ADDRESS:PATTERN" 11011110"
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7ADDRESS:PATTERN? -> :SEARCH1:
SLOGIC:I2CBUS:SETUP:ADATA:BIT7ADDRESS:
PATTERN" 11011110"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub?

Function Queries all settings related to the 7-bit address + Sub address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit address + Sub address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:ADDRess?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:HEXA

Function Queries all settings related to the 7-bit address of the 7-bit address + Sub address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:ADDRess:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7APSUB:ADDRESS:HEXA" CD"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:ADDRess:PATTern

Function Sets the 7-bit address of the 7-bit address + Sub address of the logic I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:ADDRess:PATTern {<String>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:ADDRess:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0' to '1' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7APSUB:ADDRESS:PATTERN" 11001101"
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7APSUB:ADDRESS:PATTERN? -> :SEARCH1:
SLOGIC:I2CBUS:SETUP:ADATA:BIT7APSUB:
ADDRESS:PATTERN" 11001101"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress?

Function Queries all settings related to the Sub address of the 7-bit address + Sub address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:SADDress?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:HEXA

Function Queries all settings related to the Sub address of the 7-bit address + Sub address of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:SADDress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7APSUB:SADDRESS:HEXA " EF"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:BIT7APsub:SADDress:PATTern

Function Sets the Sub address of the 7-bit address + Sub address of the logic I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:SADDress:PATTern {<String>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
BIT7APsub:SADDress:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0' to '1' and 'X'

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7APSUB:SADDRESS:PATTERN " 11101111"
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
BIT7APSUB:SADDRESS:
PATTERN? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:ADATA:BIT7APSUB:SADDRESS:
PATTERN " 11101111"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:ADATa:TYPE

Function Sets the address type of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
TYPE {BIT10address|BIT7ADdress|
BIT7APsub}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
ADATa:
TYPE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
TYPE BIT10ADDRESS
:SEARCH1:SLOGIC:I2CBUS:SETUP:ADATA:
TYPE? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:
ADATA:TYPE BIT10ADDRESS
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA?

Function Queries all settings related to the data of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:BYTE

Function Sets the number of setup data bytes of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:
BYTE {<NRf>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:
BYTE?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:
BYTE 1
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:
BYTE? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:
DATA:BYTE 1
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:CONDition

Function Sets the determination method (match or not match) of the data of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:
CONDition {FALSe|TRUE}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:
CONDition?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:
CONDITION FALSE
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:
CONDITION? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:DATA:CONDITION FALSE
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:DPOSition

Function Sets the position for comparing the data pattern of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:DPOSition {<NRf>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:DPOSition?
```
<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:DPOSITION 1
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:DPOSITION? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:DPOSITION 1
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:HEXA<x>

Function Sets the data of the logic I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:HEXA<x> {<String>}
```
<x> of SEARch<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:HEXA1" AB"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:MODE

Function Enables/disables the data conditions of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:MODE {<Boolean>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:MODE ON
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:MODE? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:MODE 1
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PATTern<x>

Function Sets the data of the logic I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PATTern<x> {<String>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PATTern<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<String> = 8 characters by combining '0' to '1' and 'X'

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:PATTERN1" 10101011"
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:PATTERN1? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:PATTERN1" 10101011"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PMODe

Function Sets the pattern comparison start position mode of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PMODe {DONTcare|SELect}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:PMODe?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:PMODE DONTCARE
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:PMODE? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:PMODE DONTCARE
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:TRACe

Function Sets the data trace of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:TRACe {A<y>|B<y>|C<y>|D<y>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:DATA:TRACe?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:TRACE A0
:SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:TRACE? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:DATA:TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}..

:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl?

Function Queries all settings related to the general call of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:BIT7maddress?

Function Queries all settings related to the 7-bit master address of the general code of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl:
BIT7maddress?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:BIT7maddress:HEXA

Function Sets the 7-bit master address of the general call of the logic I²C bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl:
BIT7maddress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is fixed 1)

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:GCALL:
BIT7MADDRESS:HEXA " BA"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:BIT7maddress:PATTern

Function Sets the 7-bit master address of the general call of the logic I²C bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl:
BIT7maddress:PATTern {<String>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl:
BIT7maddress:PATTern?
```
<x> = 1 or 2
<String> = 7 characters by combining '0' to '1' and 'X'

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:GCALL:
BIT7MADDRESS:PATTERN " 1010101"
:SEARCH1:SLOGIC:I2CBUS:SETUP:GCALL:
BIT7MADDRESS:PATTERN? -> :SEARCH1:
SLOGIC:I2CBUS:SETUP:GCALL:
BIT7MADDRESS:
PATTERN " 1010101"
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:GCALl:SBYTe (Second Byte)

Function Sets the second byte type of the general call of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl:
SBYTe {BIT7maddress|DONTcare|H04|H06}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
GCALl:
SBYTe?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:GCALL:
SBYTE BIT7MADDRESS
:SEARCH1:SLOGIC:I2CBUS:SETUP:GCALL:
SBYTE? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:GCALL:SBYTE BIT7MADDRESS
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:MODE

Function Sets the search mode of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:MODE
{ADATa|ESTart|GCALl|NAIGnore|SBHSmode}
:SEARch<x>:SLOGic:I2CBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:MODE ADATA
:SEARCH1:SLOGIC:I2CBUS:SETUP:
MODE? -> :SEARCH1:SLOGIC:I2CBUS:SETUP:
MODE ADATA
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore?

Function Queries all settings related to the NON ACK ignore mode of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore:HSMode

Function Sets whether to ignore NON ACK in high speed mode of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore:HSMode {<Boolean>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore:HSMode?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:NAIGNORE:
HSMODE ON
:SEARCH1:SLOGIC:I2CBUS:SETUP:NAIGNORE:
HSMODE? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:NAIGNORE:HSMODE 1
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore:RACCess

Function Sets whether to ignore NON ACK in read access mode of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore:RACCess {<Boolean>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore:RACCess?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:NAIGNORE:
RACCESS ON
:SEARCH1:SLOGIC:I2CBUS:SETUP:NAIGNORE:
RACCESS? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:NAIGNORE:RACCESS 1
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:NAIGnore:SBYTe (Start Byte)

Function Sets whether to ignore NON ACK in the start byte of the I²C bus trigger or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore:SBYTe {<Boolean>}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
NAIGnore:SBYTe?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:NAIGNORE:
SBYTE ON
:SEARCH1:SLOGIC:I2CBUS:SETUP:NAIGNORE:
SBYTE? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:NAIGNORE:SBYTE 1
```

:SEARch<x>:SLOGic:I2CBus[:SETup]:SBHSmode?

Function Queries all settings related to the start byte and high speed mode of the logic I²C bus signal search.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
SBHSmode?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:I2CBus[:SETup]:SBHSmode:TYPE

Function Sets the type of start byte and high speed mode of the logic I²C bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:I2CBus[:SETup]:
SBHSmode:TYPE {HSMode|SBYTe}
:SEARch<x>:SLOGic:I2CBus[:SETup]:
SBHSmode:TYPE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:I2CBUS:SETUP:SBHSMODE:
TYPE HSMODE
:SEARCH1:SLOGIC:I2CBUS:SETUP:SBHSMODE:
TYPE? -> :SEARCH1:SLOGIC:I2CBUS:
SETUP:SBHSMODE:TYPE HSMODE
```

:SEARch<x>:SLOGic:LINBus?

Function Queries all settings related to the logic LIN bus signal search.

Syntax
```
:SEARch<x>:SLOGic:LINBus?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:LINBus[:SETup]?

Function Queries all settings related to the setup of the logic LIN bus signal search.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:LINBus[:SETup]:BLENgth

Function Sets the logic LIN bus signal search break length or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
BLENgth {<NRf>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
BLENgth?
```
<x> = 1 or 2
<NRf> = 10 to 13

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:BLENGTH 10
:SEARCH1:SLOGIC:LINBUS:SETUP:BLENGTH?
-> :SEARCH1:SLOGIC:LINBUS:SETUP:
BLENGTH 10
```

:SEARch<x>:SLOGic:LINBus[:SETup]:BRATe

Function Sets the bit rate (data transfer rate) of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:BRATe
{<NRf>|USER,<NRf>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
BRATe?
```
<x> = 1 or 2
<NRf> = 1200, 2400, 4800, 9600, or 19200
<NRf> of USER = See section 5.4.

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:
BRATE 19200
:SEARCH1:SLOGIC:LINBUS:SETUP:
BRATE? -> :SEARCH1:SLOGIC:LINBUS:SETUP:
BRATE 19200
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA?

Function Queries all settings related to the data of the logic LIN bus signal search.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:BNUM

Function Sets the number of bytes of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:
BNUM {<NRf>}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:
BNUM?
```
<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:
BNUM 1
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:
BNUM? -> :SEARCH1:SLOGIC:LINBUS:SETUP:
DATA:BNUM 1
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:BORDer

Function Sets the data byte order of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:
BORDer {BIG|LITTle}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:
BORDer?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:
BORDER BIG
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:
BORDER? -> :SEARCH1:SLOGIC:LINBUS:
SETUP:DATA:BORDER BIG
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:CONDition

Function Sets the data condition of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:
CONDition {BETWeen|DONTcare|
FALSe|GTHan|LTHan|ORANge|TRUE}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:
CONDition?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:
CONDITION BETWEEN
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:
CONDITION? -> :SEARCH1:SLOGIC:LINBUS:
SETUP:DATA:CONDITION BETWEEN
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA<x>

Function Sets the comparison data of the logic LIN bus signal search data or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA<x> {<NRf>}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = See section 5.4.

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:DATA1 1
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:DATA1? -> :SEARCH1:SLOGIC:LINBUS:SETUP:DATA:DATA1 1.0000000E+00
```

Description
- For :SEARch<x>:SLOGic:LINBus[:SETup]:DATA:CONDition GTHan, set using: SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA1.
- For :SEARch<x>:SLOGic:LINBus[:SETup]:DATA:CONDition LTHan, set using: SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA2.
- For :SEARch<x>:SLOGic:LINBus[:SETup]:DATA:CONDition BETWeen|ORANge, set small values with: SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA1, and large values with: SEARch<x>:SLOGic:LINBus[:SETup]:DATA:DATA2.

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:HEXA

Function Sets the data of the logic LIN bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:HEXA {<String>}
```
<x> = 1 or 2
<String> = Up to 16 characters by combining ‘0’ to ‘F’ and ‘X’ (varies depending on the BNUM setting)

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:HEXA" 3B"
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:MSBLsb

Function Sets the MSB/LSB bit of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:MSBLsb {<NRf>,<NRf>}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:MSBLsb?
```
<x> = 1 or 2
<NRf> = See section 5.4.

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:MSBLSB 1,0
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:MSBLSB? -> :SEARCH1:SLOGIC:LINBUS:SETUP:DATA:MSBLSB 1,0
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:PATTern

Function Sets the data of the logic LIN bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:PATTern {<String>}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:PATTern?
```
<x> = 1 or 2
<String> = Up to 64 characters by combining ‘0’ to ‘1’ and ‘X’ (varies depending on the BNUM setting)

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:PATTERN" 11011111"
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:PATTERN? -> :SEARCH1:SLOGIC:LINBUS:SETUP:DATA:PATTERN" 11011111"
```

:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:SIGN

Function Sets the data sign of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:SIGN {SIGN|UNSign}
:SEARch<x>:SLOGic:LINBus[:SETup]:DATA:SIGN?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:SIGN SIGN
:SEARCH1:SLOGIC:LINBUS:SETUP:DATA:SIGN?
-> :SEARCH1:SLOGIC:LINBUS:SETUP:DATA:SIGN SIGN
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor?

Function Queries all settings related to the logic LIN bus signal search error.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:CHECksum

Function Sets the logic LIN bus signal search Checksum error or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
CHECksum {<Boolean>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
CHECksum?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
CHECKSUM ON
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
CHECKSUM? -> :SEARCH1:SLOGIC:LINBUS:
SETUP:ERROR:CHECKSUM 1
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:FRAMing

Function Sets the logic LIN bus signal search Framing error or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
FRAMing {<Boolean>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
FRAMing?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
FRAMING ON
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
FRAMING? -> :SEARCH1:SLOGIC:LINBUS:
SETUP:ERROR:FRAMING 1
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:PARity

Function Sets the logic LIN bus signal search Parity error or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
PARity {<Boolean>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
PARity?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
PARITY ON
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
PARITY? -> :SEARCH1:SLOGIC:LINBUS:
SETUP:ERROR:PARITY 1
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:SYNCh

Function Sets the logic LIN bus signal search Synch error or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
SYNCh {<Boolean>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
SYNCh?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
SYNCH ON
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
SYNCH? -> :SEARCH1:SLOGIC:LINBUS:SETUP:
ERROR:SYNCH 1
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ERRor:TOUT

Function Sets the logic LIN bus signal search Timeout error or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
TOUT {<Boolean>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
ERRor:
TOUT?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
TOUT ON
:SEARCH1:SLOGIC:LINBUS:SETUP:ERROR:
TOUT? -> :SEARCH1:SLOGIC:LINBUS:SETUP:
ERROR:TOUT 1
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ID?

Function Queries all settings related to the ID of the logic LIN bus signal search.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:ID?
```
<x> = 1 or 2

:SEARch<x>:SLOGic:LINBus[:SETup]:ID:HEXA

Function Sets the ID of the logic LIN bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:ID:
HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ID:
HEXA " 2A"
```

:SEARch<x>:SLOGic:LINBus[:SETup]:ID:PATTern

Function Sets the ID of the logic LIN bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:ID:
PATTern {<String>}
:SEARch<x>:SLOGic:LINBus[:SETup]:ID:
PATTern?
```
<x> = 1 or 2
<String> = 6 characters by combining '0' to '1' and 'X'

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:ID:
PATTERN " 101111"
:SEARCH1:SLOGIC:LINBUS:SETUP:ID:
PATTERN? -> :SEARCH1:SLOGIC:LINBUS:
SETUP:ID:PATTERN " 101111"
```

:SEARch<x>:SLOGic:LINBus[:SETup]:MODE

Function Sets the logic LIN bus signal search mode or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
MODE {IDData|SYNCh}
:SEARch<x>:SLOGic:LINBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:MODE
IDDATA
:SEARCH1:SLOGIC:LINBUS:SETUP:
MODE? -> :SEARCH1:SLOGIC:LINBUS:SETUP:
MODE IDDATA
```

:SEARch<x>:SLOGic:LINBus[:SETup]:REVision

Function Sets the logic LIN bus signal search revision (1.3, 2.0, or Both) or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
REVision {BOTH|LIN1_3|LIN2_0}
:SEARch<x>:SLOGic:LINBus[:SETup]:
REVision?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:
REVISION LIN1_3
:SEARCH1:SLOGIC:LINBUS:SETUP:REVISION?
-> :SEARCH1:SLOGIC:LINBUS:SETUP:
REVISION LIN1_3
```

:SEARch<x>:SLOGic:LINBus[:SETup]:SPOint

Function Sets the logic LIN bus signal search sampling point or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
SPOint {<NRf>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:
SPOINT 18.8
:SEARCH1:SLOGIC:LINBUS:SETUP:SPOINT?
-> :SEARCH1:SLOGIC:LINBUS:SETUP:
SPOINT 18.8E+00
```

:SEARch<x>:SLOGic:LINBus[:SETup]:TRACe

Function Sets the trace of the logic LIN bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:LINBus[:SETup]:
TRACe {A<y>|B<y>|C<y>|D<y>}
:SEARch<x>:SLOGic:LINBus[:SETup]:
TRACe?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:LINBUS:SETUP:TRACE A0
:SEARCH1:SLOGIC:LINBUS:SETUP:
TRACE? -> :SEARCH1:SLOGIC:LINBUS:SETUP:
TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}..

:SEARch<x>:SLOGic:SPIBus?

Function Queries all settings related to the logic SPI bus signal search.

Syntax `:SEARch<x>:SLOGic:SPIBus?`

<x> = 1 or 2

:SEARch<x>:SLOGic:SPIBus:CLOCk?

Function Queries all settings related to the clock signal channel of the logic SPI bus signal search.

Syntax `:SEARch<x>:SLOGic:SPIBus:CLOCk?`

<x> = 1 or 2

:SEARch<x>:SLOGic:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock signal channel of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus:CLOCk:
POLarity {FALL|RISE}
:SEARch<x>:SLOGic:SPIBus:CLOCk:
POLarity?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:SPIBUS:CLOCK:
POLARITY FALL
:SEARCH1:SLOGIC:SPIBUS:CLOCK:
POLARITY? -> :SEARCH1:SLOGIC:SPIBUS:
CLOCK:POLARITY FALL
```

:SEARch<x>:SLOGic:SPIBus:CLOCk:SOURce

Function Sets the clock signal channel of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus:CLOCk:
SOURce {A<y>|B<y>|C<y>|D<y>}
:SEARch<x>:SLOGic:SPIBus:CLOCk:SOURce?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:SPIBUS:CLOCK:SOURCE A0
:SEARCH1:SLOGIC:SPIBUS:CLOCK:
SOURCE? -> :SEARCH1:SLOGIC:SPIBUS:
CLOCK:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:SEARch<x>:SLOGic:SPIBus:CS?

Function Queries all settings related to the chip select signal channel of the logic SPI bus signal search.

Syntax `:SEARch<x>:SLOGic:SPIBus:CS?`

<x> = 1 or 2

:SEARch<x>:SLOGic:SPIBus:CS:ACTive

Function Sets the active level of the chip select signal channel of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus:CS:
ACTive {HIGH|LOW}
:SEARch<x>:SLOGic:SPIBus:CS:ACTive?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:SPIBUS:CS:ACTIVE HIGH
:SEARCH1:SLOGIC:SPIBUS:CS:
ACTIVE? -> :SEARCH1:SLOGIC:SPIBUS:CS:
ACTIVE HIGH
```

:SEARch<x>:SLOGic:SPIBus:CS:TRACe

Function Sets the chip select signal channel of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus:CS:
TRACe {A<y>|B<y>|C<y>|D<y>|NONE}
:SEARch<x>:SLOGic:SPIBus:CS:TRACe?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:SPIBUS:CS:TRACE A0
:SEARCH1:SLOGIC:SPIBUS:CS:
TRACE? -> :SEARCH1:SLOGIC:SPIBUS:CS:
TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:SEARch<x>:SLOGic:SPIBus[:SETup]?

Function Queries all settings related to the setup of the logic SPI bus signal search.

Syntax `:SEARch<x>:SLOGic:SPIBus[:SETup]?`

<x> = 1 or 2

:SEARch<x>:SLOGic:SPIBus[:SETup]:BITorder

Function Sets the bit order of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
BITorder {LSBFirst|MSBFirst}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
BITorder?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:
BITORDER LSBFIRST
:SEARCH1:SLOGIC:SPIBUS:SETUP:
BITORDER? -> :SEARCH1:SLOGIC:SPIBUS:
SETUP:BITORDER LSBFIRST
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>?

Function Queries all settings related to each data of the logic SPI bus signal search.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Description DATA2 is valid when: `SEARch<x>:SLOGic:SPIBus[:SETup]:MODE WIRE4` is specified.

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:BYTE

Function Sets the data size (in bytes) of each data of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:BYTE {<NRf>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:BYTE?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
BYTE 1
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
BYTE? -> :SEARCH1:SLOGIC:SPIBUS:SETUP:
DATA1:BYTE 1
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:CONDition

Function Sets the determination method (match/mismatch) of the data of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:CONDition {FALSe|TRUE}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:CONDition?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
CONDITION FALSE
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
CONDITION? -> :SEARCH1:SLOGIC:SPIBUS:
SETUP:DATA1:CONDITION FALSE
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:DPOSition

Function Sets the pattern comparison start position of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:DPOSition {<NRf>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:DPOSition?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
DPOSITION 1
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
DPOSITION? -> :SEARCH1:SLOGIC:SPIBUS:
SETUP:DATA1:DPOSITION 1
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:DSIZe

Function Sets the number of fields in the data used for logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:DSIZe {<NRf>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:DSIZe?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
DSIZE 1
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
DSIZE? -> :SEARCH1:SLOGIC:SPIBUS:SETUP:
DATA1:DSIZE 1
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:HEXA<x>

Function Sets the data of the logic SPI bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:HEXA<x> {<String>}
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<String> = Up to 8 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
HEXA1" EF"
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:PATTern<x>

Function Sets the data of the logic SPI bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:PATTern<x> {<String>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:PATTern<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<String> = Up to 32 characters by combining '0' to '1' and 'X'

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
PATTERN1 " 11101111"
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
PATTERN1? -> :SEARCH1:SLOGIC:SPIBUS:
SETUP:DATA1:PATTERN1 " 11101111"
```
.

:SEARch<x>:SLOGic:SPIBus[:SETup]:DATA<x>:TRACe

Function Sets the source channel of each data of the logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:TRACe {A<y>|B<y>|C<y>|D<y>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
DATA<x>:TRACe?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
TRACE A0
:SEARCH1:SLOGIC:SPIBUS:SETUP:DATA1:
TRACE? -> :SEARCH1:SLOGIC:SPIBUS:
SETUP:DATA1:TRACE A0
```
Description On 16-bit models, you can only select {A<x>|C<x>}.

:SEARch<x>:SLOGic:SPIBus[:SETup]:EMSBLSB

Function Sets the enabled range of the field used for logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
EMSBLSB {<NRf>,<NRf>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
EMSBLSB?
```
<x> = 1 or 2
<NRf> = See section 5.5.

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:
EMSBLSB 1,7
:SEARCH1:SLOGIC:SPIBUS:SETUP:
EMSBLSB? -> :SEARCH1:SLOGIC:SPIBUS:
SETUP:EMSBLSB 1,7
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:FSIZe

Function Sets the field size used for logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
FSIZe {<NRf>}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
FSIZe?
```
<x> = 1 or 2
<NRf> = 4 to 32

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:FSIZE 4
:SEARCH1:SLOGIC:SPIBUS:SETUP:FSIZE? ->
:SEARCH1:SLOGIC:SPIBUS:SETUP:FSIZE 4
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:ITIMe

Function Sets the idle time used in logic SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
ITIMe {<Time>|DONTcare}
:SEARch<x>:SLOGic:SPIBus[:SETup]:
ITIMe?
```
<x> = 1 or 2
<Time> = 10ns to 1ms in 10-ns steps

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:ITIME 10NS
:SEARCH1:SLOGIC:SPIBUS:SETUP:
ITIME? -> :SEARCH1:SLOGIC:SPIBUS:SETUP:
ITIME 10.0000E-09
```

:SEARch<x>:SLOGic:SPIBus[:SETup]:MODE

Function Sets the wiring system of the logic SPI bus signal search (three-wire or four-wire) or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:SPIBus[:SETup]:
MODE {WIRE3|WIRE4}
:SEARch<x>:SLOGic:SPIBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:SPIBUS:SETUP:MODE WIRE3
:SEARCH1:SLOGIC:SPIBUS:SETUP:
MODE? -> :SEARCH1:SLOGIC:SPIBUS:SETUP:
MODE WIRE3
```

:SEARch<x>:SLOGic:UART?

Function Queries all settings related to the logic UART bus signal search.

Syntax `:SEARch<x>:SLOGic:UART?`
<x> = 1 or 2

:SEARch<x>:SLOGic:UART:BRATe

Function Sets the logic UART bus signal search bit rate (data transfer rate) or queries the current setting.

Syntax `:SEARch<x>:SLOGic:UART:BRATe {<NRf>|USER,<NRf>}`
`:SEARch<x>:SLOGic:UART:BRATe?`
<x> = 1 or 2
<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
<NRf> of USER = See section 5.6.

Example
```
:SEARCH1:SLOGIC:UART:BRATE 19200
:SEARCH1:SLOGIC:UART:BRATE? ->
:SEARCH1:SLOGIC:UART:BRATE 19200
```

:SEARch<x>:SLOGic:UART:DATA?

Function Queries all settings related to data of the logic UART bus signal search.

Syntax `:SEARch<x>:SLOGic:UART:DATA?`
<x> = 1 or 2

:SEARch<x>:SLOGic:UART:DATA:BITorder

Function Sets the data bit order of the logic UART bus signal search or queries the current setting.

Syntax `:SEARch<x>:SLOGic:UART:DATA:BITorder {LSBFirst|MSBFirst}`
`:SEARch<x>:SLOGic:UART:DATA:BITorder?`
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:DATA:
BITORDER LSBFIRST
:SEARCH1:SLOGIC:UART:DATA:BITORDER?
-> :SEARCH1:SLOGIC:UART:DATA:
BITORDER LSBFIRST
```

:SEARch<x>:SLOGic:UART:DATA:DSIZe

Function Sets the number of data bytes of the logic UART bus signal search or queries the current setting.

Syntax `:SEARch<x>:SLOGic:UART:DATA:DSIZe {<NRf>}`
`:SEARch<x>:SLOGic:UART:DATA:DSIZe?`
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:SLOGIC:UART:DATA:DSIZE 1
:SEARCH1:SLOGIC:UART:DATA:DSIZE? ->
:SEARCH1:SLOGIC:UART:DATA:DSIZE 1
```

:SEARch<x>:SLOGic:UART:DATA:HEXA

Function Sets the logic UART bus signal search data in hexadecimal.

Syntax `:SEARch<x>:SLOGic:UART:DATA:HEXA {<String>}`
<x> = 1 or 2
<String> = Up to 8 characters by combining '0' to 'F' and 'X,' units of 1 byte

Example `:SEARCH1:SLOGIC:UART:DATA:HEXA "A9".`

:SEARch<x>:SLOGic:UART:DATA:PATTern

Function Sets the data of the logic UART bus signal search in binary or queries the current setting.

Syntax `:SEARch<x>:SLOGic:UART:DATA:PATTern {<String>}`
`:SEARch<x>:SLOGic:UART:DATA:PATTern?`
<x> = 1 or 2
<String> = Up to 32 characters by combining '0,' '1,' and 'X,' units of 1 byte

Example
```
:SEARCH1:SLOGIC:UART:DATA:
PATTERN "11011111"
:SEARCH1:SLOGIC:UART:DATA:PATTERN?
-> :SEARCH1:SLOGIC:UART:DATA:
PATTERN "11011111"
```

:SEARch<x>:SLOGic:UART:ERRor?

Function Queries all settings related to the logic UART bus signal search error.

Syntax `:SEARch<x>:SLOGic:UART:ERRor?`
<x> = 1 or 2

:SEARch<x>:SLOGic:UART:ERRor:FRAMing

Function Sets the logic UART bus signal search Framing error or queries the current setting.

Syntax `:SEARch<x>:SLOGic:UART:ERRor:FRAMing {<Boolean>}`
`:SEARch<x>:SLOGic:UART:ERRor:FRAMing?`
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:ERROR:FRAMING ON
:SEARCH1:SLOGIC:UART:ERROR:FRAMING? ->
:SEARCH1:SLOGIC:UART:ERROR:FRAMING 1
```

:SEARch<x>:SLOGic:UART:ERRor:PARity

Function Sets the logic UART bus signal search Parity error or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:ERRor:
PARity {<Boolean>}
:SEARch<x>:SLOGic:UART:ERRor:PARity?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:ERROR:PARITY ON
:SEARCH1:SLOGIC:UART:ERROR:PARITY? ->
:SEARCH1:SLOGIC:UART:ERROR:PARITY 1
```

:SEARch<x>:SLOGic:UART:ERRor:PMODe

Function Sets the logic UART bus signal search Parity mode or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:ERRor:
PMODe {EVEN|ODD}
:SEARch<x>:SLOGic:UART:ERRor:PMODe?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:ERROR:PMODE EVEN
:SEARCH1:SLOGIC:UART:ERROR:PMODE? ->
:SEARCH1:SLOGIC:UART:ERROR:PMODE EVEN
```

:SEARch<x>:SLOGic:UART:FORMat

Function Sets the logic UART bus signal search format or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:
FORMat {BIT7parity|BIT8Noparity|
BIT8Parity}
:SEARch<x>:SLOGic:UART:FORMat?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:FORMAT BIT7PARITY
:SEARCH1:SLOGIC:UART:FORMAT? ->
:SEARCH1:SLOGIC:UART:FORMAT BIT7PARITY
```

:SEARch<x>:SLOGic:UART:MODE

Function Sets the logic UART bus signal search mode or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:
MODE {DATA|ERRor}
:SEARch<x>:SLOGic:UART:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:MODE DATA
:SEARCH1:SLOGIC:UART:MODE? -> :SEARCH1:
SLOGIC:UART:MODE DATA
```

:SEARch<x>:SLOGic:UART:POLarity

Function Sets the logic UART bus signal search polarity or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:
POLarity {NEGative|POSitive}
:SEARch<x>:SLOGic:UART:POLarity?
```
<x> = 1 or 2

Example
```
:SEARCH1:SLOGIC:UART:POLARITY NEGATIVE
:SEARCH1:SLOGIC:UART:POLARITY? ->
:SEARCH1:SLOGIC:UART:POLARITY NEGATIVE
```

:SEARch<x>:SLOGic:UART:SPOint

Function Sets the logic UART bus signal search sampling point or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:SPOint {<NRf>}
:SEARch<x>:SLOGic:UART:SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:SEARCH1:SLOGIC:UART:SPOINT 18.8
:SEARCH1:SLOGIC:UART:SPOINT? ->
:SEARCH1:SLOGIC:UART:SPOINT 18.8E+00
```

:SEARch<x>:SLOGic:UART:TRACe

Function Sets the logic UART bus signal search trace or queries the current setting.

Syntax
```
:SEARch<x>:SLOGic:UART:TRACe
{A<y>|B<y>|C<y>|D<y>}
:SEARch<x>:SLOGic:UART:TRACe?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:SEARCH1:SLOGIC:UART:TRACE A0
:SEARCH1:SLOGIC:UART:TRACE? ->
:SEARCH1:SLOGIC:UART:TRACE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:SEARch<x>:SPIBus?
Function Queries all settings related to the SPI bus signal search.
Syntax :SEARch<x>:SPIBus?
<x> = 1 or 2

:SEARch<x>:SPIBus:CLOCk
Function Queries all settings related to the clock channel of the SPI bus signal search.
Syntax :SEARch<x>:SPIBus:CLOCk?
<x> = 1 or 2

:SEARch<x>:SPIBus:CLOCk:POLarity
Function Sets the polarity of the clock channel of the SPI bus signal search or queries the current setting.
Syntax :SEARch<x>:SPIBus:CLOCk:
POLarity {FALL|RISE}
:SEARch<x>:SPIBus:CLOCk:POLarity?
<x> = 1 or 2
Example :SEARCH1:SPIBUS:CLOCK:POLARITY FALL
:SEARCH1:SPIBUS:CLOCK:POLARITY?
-> :SEARCH1:SPIBUS:CLOCK:POLARITY FALL

:SEARch<x>:SPIBus:CLOCk:SOURce
Function Sets the clock channel of the SPI bus signal search or queries the current setting.
Syntax :SEARch<x>:SPIBus:CLOCk:SOURce {<NRf>}
:SEARch<x>:SPIBus:CLOCk:SOURce?
<x> = 1 or 2
<NRf> = 1 to 8
Example :SEARCH1:SPIBUS:CLOCK:SOURCE 1
:SEARCH1:SPIBUS:CLOCK:SOURCE?
-> :SEARCH1:SPIBUS:CLOCK:SOURCE 1

:SEARch<x>:SPIBus:CS?
Function Queries all settings related to the chip select channel of the SPI bus signal search.
Syntax :SEARch<x>:SPIBus:CS?
<x> = 1 or 2
Example :SEARCH1:SPIBUS:CS?
-> :SEARCH1:SPIBUS:CS:ACTIVE HIGH;
TRACE 1

:SEARch<x>:SPIBus:CS:ACTive
Function Sets the active level of the chip select channel of the SPI bus signal search or queries the current setting.
Syntax :SEARch<x>:SPIBus:CS:ACTive {HIGH|LOW}
:SEARch<x>:SPIBus:CS:ACTive?
<x> = 1 or 2
Example :SEARCH1:SPIBUS:CS:ACTIVE HIGH
:SEARCH1:SPIBUS:CS:ACTIVE?
-> :SEARCH1:SPIBUS:CS:ACTIVE HIGH

:SEARch<x>:SPIBus:CS:TRACe
Function Sets the chip select channel of the SPI bus signal search or queries the current setting.
Syntax :SEARch<x>:SPIBus:CS:TRACe {<NRf>|NONE}
:SEARch<x>:SPIBus:CS:TRACe?
<x> = 1 or 2
<NRf> = 1 to 8
Example :SEARCH1:SPIBUS:CS:TRACE 1
:SEARCH1:SPIBUS:CS:TRACE?
-> :SEARCH1:SPIBUS:CS:TRACE 1

:SEARch<x>:SPIBus:SETup?
Function Queries all settings related to the SPI bus signal search setup.
Syntax :SEARch<x>:SPIBus:SETup?
<x> = 1 or 2

:SEARch<x>:SPIBus[:SETup]:BITorder
Function Sets the bit order of the SPI bus signal search or queries the current setting.
Syntax :SEARch<x>:SPIBus[:SETup]:
BITorder {LSBFirst|MSBFirst}
:SEARch<x>:SPIBus[:SETup]:BITorder?
<x> = 1 or 2
Example :SEARCH1:SPIBUS:SETUP:BITORDER LSBFIRST
:SEARCH1:SPIBUS:SETUP:BITORDER?
-> :SEARCH1:SPIBUS:SETUP:
BITORDER LSBFIRST

:SEARch<x>:SPIBus[:SETup]:DATA<x>?
Function Queries all settings related to the data of the SPI bus signal search.
Syntax :SEARch<x>:SPIBus[:SETup]:DATA<x>?
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
Description DATA2 is valid when :SEARch<x>:
SPIBus[:SETup]:MODE WIRE4 is specified.

:SEARch<x>:SPIBus[:SETup]:DATA<x>:BYTE
Function Sets the number of bytes of the data of the SPI bus signal search or queries the current setting.
Syntax :SEARch<x>:SPIBus[:SETup]:DATA<x>:
BYTE {<NRf>}
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
BYTE?
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4
Example :SEARCH1:SPIBUS:SETUP:DATA1:BYTE 1
:SEARCH1:SPIBUS:SETUP:DATA1:BYTE?
-> :SEARCH1:SPIBUS:SETUP:DATA1:BYTE 1

:SEARch<x>:SPIBus[:SETup]:DATA<x>:CONDition

Function Sets the determination method (match or not match) of the data of the SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
CONDition {FALSe|TRUE}
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
CONDition?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:SEARCH1:SPIBUS:SETUP:DATA1:
CONDITION TRUE
:SEARCH1:SPIBUS:SETUP:DATA1:
CONDITION?
-> :SEARCH1:SPIBUS:SETUP:DATA1:
CONDITION TRUE
```

:SEARch<x>:SPIBus[:SETup]:DATA<x>:DPOSition

Function Sets the pattern comparison start position of the data of the SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
DPOSition {<NRf>}
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
DPOSition?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:SEARCH1:SPIBUS:SETUP:DATA1:DPOSITION 1
:SEARCH1:SPIBUS:SETUP:DATA1:
DPOSITION?
-> :SEARCH1:SPIBUS:SETUP:DATA1:
DPOSITION 1
```

:SEARch<x>:SPIBus[:SETup]:DATA<x>:DSIZe

Function Sets the number of fields in the data used for SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
DSIZe {<NRf>}
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
DSIZe?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:SPIBUS:SETUP:DATA1:DSIZE 1
:SEARCH1:SPIBUS:SETUP:DATA1:DSIZE?
-> :SEARCH1:SPIBUS:SETUP:DATA1:DSIZE 1
```

:SEARch<x>:SPIBus[:SETup]:DATA<x>:HEXA<x>

Function Sets the data of the SPI bus signal search in hexadecimal notation.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
HEXA<x> {<String>}
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<String> = Up to 8 characters by combining '0' to 'F' and 'X'

Example
```
:SEARCH1:SPIBUS:SETUP:DATA1:HEXA1 " EF"
```

:SEARch<x>:SPIBus[:SETup]:DATA<x>:PATTern<x>

Function Sets the data of the SPI bus signal search in binary notation or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
PATTern<x> {<String>}
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
PATTern<x>?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of <PATTern x> = 1 to 4
<String> = Up to 32 characters by combining '0,' '1,' and 'X'

Example
```
:SEARCH1:SPIBUS:SETUP:DATA1:
PATTERN1 " 11101111"
:SEARCH1:SPIBUS:SETUP:DATA1:
PATTERN1?
-> :SEARCH1:SPIBUS:SETUP:DATA1:
PATTERN1 " 11101111"
```

:SEARch<x>:SPIBus[:SETup]:DATA<x>:TRACe

Function Sets the source channel of the data of the SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
TRACe {<NRf>}
:SEARch<x>:SPIBus[:SETup]:DATA<x>:
TRACe?
```
<x> of SEARch<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 8

Example
```
:SEARCH1:SPIBUS:SETUP:DATA1:TRACE 1
:SEARCH1:SPIBUS:SETUP:DATA1:TRACE?
-> :SEARCH1:SPIBUS:SETUP:DATA1:TRACE 1
```

:SEARch<x>:SPIBus[:SETup]:EMSBLSB

Function Sets the enabled range of the field used for SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:
EMSBLSB {<NRf>,<NRf>}
:SEARch<x>:SPIBus[:SETup]:EMSBLSB?
```
<x> = 1 or 2
<NRf> = See section 5.5.

Example
```
:SEARCH1:SPIBUS:SETUP:EMSBLSB 1,7
:SEARCH1:SPIBUS:SETUP:EMSBLSB?
-> :SEARCH1:SPIBUS:SETUP:EMSBLSB 1,7
```

:SEARch<x>:SPIBus[:SETup]:FSIZe

Function Sets the field size used for SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:FSIZe {<NRf>}
:SEARch<x>:SPIBus[:SETup]:FSIZe?
```
<x> = 1 or 2
<NRf> = 4 to 32

Example
```
:SEARCH1:SPIBUS:SETUP:FSIZE 4
:SEARCH1:SPIBUS:SETUP:FSIZE?
-> :SEARCH1:SPIBUS:SETUP:FSIZE 4
```

:SEARch<x>:SPIBus[:SETup]:ITIMe

Function Sets the idle time used in SPI bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:
ITIMe {<Time>|DONTcare}
:SEARch<x>:SPIBus[:SETup]:ITIMe?
```
<x> = 1 or 2
<Time> = 10ns to 1ms in 10-ns steps

Example
```
:SEARCH1:SPIBUS:SETUP:ITIME 10NS
:SEARCH1:SPIBUS:SETUP:ITIME?
-> :SEARCH1:SPIBUS:SETUP:ITIME
10.0000E-09
```

:SEARch<x>:SPIBus[:SETup]:MODE

Function Sets the wiring system of the SPI bus signal search (three-wire or four-wire) or queries the current setting.

Syntax
```
:SEARch<x>:SPIBus[:SETup]:
MODE {WIRE3|WIRE4}
:SEARch<x>:SPIBus[:SETup]:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:SPIBUS:SETUP:MODE WIRE3
:SEARCH1:SPIBUS:SETUP:MODE?
-> :SEARCH1:SPIBUS:SETUP:MODE WIRE3
```

:SEARch<x>:TRACe<x>:LEVel

Function Sets the threshold level of the trace or queries the current setting.

Syntax
```
:SEARch<x>:TRACe<x>:LEVel {<NRf>|<Voltage>|<Current>}
:SEARch<x>:TRACe<x>:LEVel?
```
<x> of SEARch<x> = 1 or 2
<x> of TRACe<x> = 1 to 8
<NRf>, <Voltage>, and <Current> = See sections 5.2 to 5.6.

Example
```
:SEARCH1:TRACE1:LEVEL 0
:SEARCH1:TRACE1:LEVEL?
-> :SEARCH1:TRACE1:LEVEL 0.000E+00
```

Description This command applies to the channel corresponding to the source specified by the following commands.
- :SEARch<x>:I2CBus:CLOCk:SOURce
- :SEARch<x>:STRace
- :SEARch<x>:SPIBus:CLOCk:SOURce
- :SEARch<x>:SPIBus:CS:TRACe
- :SEARch<x>:SPIBus:DATA[1-2]:TRACe

:SEARch<x>:TYPE

Function Sets the search type or queries the current setting.

Syntax
```
:SEARch<x>:TYPE {CANBus|EDGE|EQUalify|
I2CBus|LEDGe|LI2Cbus|LINBus|LLINbus|
LQUalify|LSPAttern|LSPIbus|LSTate|
LUARt|LWIDth|SPATtern|SPIBus|STATe|
UART|WIDTh}
:SEARch<x>:TYPE?
```
<x> = 1 or 2

Example
```
:SEARCH1:TYPE CANBus
:SEARCH1:TYPE? -> :SEARCH1:TYPE CANBus
```

Description {LEDGe|LI2Cbus|LLINbus|LQUalify|LSPAttern|LSPIbus|LSTate|LUARt|LWIDth} can be applied to DLM6000.

:SEARch<x>:UART?

Function Queries all settings related to the UART bus signal search.

Syntax
```
:SEARch<x>:UART?
```
<x> = 1 or 2

:SEARch<x>:UART:BRATe

Function Sets the UART bus signal search bit rate (data transfer rate) or queries the current setting.

Syntax
```
:SEARch<x>:UART:
BRATe {<NRf>|USER,<NRf>}
:SEARch<x>:UART:BRATe?
```
<x> = 1 or 2
<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
<NRf> of USER = See section 5.6.

Example
```
:SEARCH1:UART:BRATE 19200
:SEARCH1:UART:BRATE? -> :SEARCH1:UART:
BRATE 19200
```

:SEARch<x>:UART:DATA?

Function Queries all settings related to data of the UART bus signal search

Syntax
```
:SEARch<x>:UART:DATA?
```
<x> = 1 or 2

:SEARch<x>:UART:DATA:BITorder

Function Sets the data bit order of the UART bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:UART:DATA:
BITorder {LSBFirst|MSBFirst}
:SEARch<x>:UART:DATA:BITorder?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:DATA:BITORDER LSBFIRST
:SEARCH1:UART:DATA:BITORDER? ->
:SEARCH1:UART:DATA:BITORDER LSBFIRST
```

:SEARch<x>:UART:DATA:DSIZe

Function Sets the number of data bytes of the UART bus signal search or queries the current setting.

Syntax
```
:SEARch<x>:UART:DATA:DSIZe {<NRf>}
:SEARch<x>:UART:DATA:DSIZe?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:SEARCH1:UART:DATA:DSIZE 1
:SEARCH1:UART:DATA:DSIZE? ->
:SEARCH1:UART:DATA:DSIZE 1
```

:SEARch<x>:UART:DATA:HEXA

Function Sets the UART bus signal search data in hexadecimal.

Syntax
```
:SEARch<x>:UART:DATA:HEXA {<String>}
```
<x> = 1 or 2
<String> = Up to 8 characters by combining '0' to 'F' and 'X,' units of 1 byte

Example
```
:SEARCH1:UART:DATA:HEXA "A9"
```

:SEARch<x>:UART:DATA:PATTern

Function Sets the data of the UART bus signal search in binary or queries the current setting.

Syntax
```
:SEARch<x>:UART:DATA:PATTern {<String>}
:SEARch<x>:UART:DATA:PATTern?
```
<x> = 1 or 2
<String> = Up to 32 characters by combining '0,' '1,' and 'X,' units of 1 byte

Example
```
:SEARCH1:UART:DATA:PATTERN "11011111"
:SEARCH1:UART:DATA:PATTERN? ->
:SEARCH1:UART:DATA:PATTERN "11011111"
```

:SEARch<x>:UART:ERRor?

Function Queries all settings related to the UART bus signal search error.

Syntax
```
:SEARch<x>:UART:ERRor?
```
<x> = 1 or 2

:SEARch<x>:UART:ERRor:FRAMing

Function Sets the UART bus signal search Framing error or queries the current setting.

Syntax
```
:SEARch<x>:UART:ERRor:
FRAMing {<Boolean>}
:SEARch<x>:UART:ERRor:FRAMing?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:ERROR:FRAMING ON
:SEARCH1:UART:ERROR:FRAMING? ->
:SEARCH1:UART:ERROR:FRAMING 1
```

:SEARch<x>:UART:ERRor:PARity

Function Sets the UART bus signal search Parity error or queries the current setting.

Syntax
```
:SEARch<x>:UART:ERRor:
PARity {<Boolean>}
:SEARch<x>:UART:ERRor:PARity?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:ERROR:PARITY ON
:SEARCH1:UART:ERROR:PARITY? ->
:SEARCH1:UART:ERROR:PARITY 1
```

:SEARch<x>:UART:ERRor:PMODe

Function Sets the UART bus signal search Parity mode or queries the current setting.

Syntax
```
:SEARch<x>:UART:ERRor:PMODe {EVEN|ODD}
:SEARch<x>:UART:ERRor:PMODe?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:ERROR:PMODE EVEN
:SEARCH1:UART:ERROR:PMODE? -> :SEARCH1:
UART:ERROR:PMODE EVEN
```

:SEARch<x>:UART:FORMat

Function Sets the UART bus signal search format or queries the current setting.

Syntax
```
:SEARch<x>:UART:FORMat {BIT7parity|
BIT8Noparity|BIT8Parity}
:SEARch<x>:UART:FORMat?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:FORMAT BIT7PARITY
:SEARCH1:UART:FORMAT? -> :SEARCH1:UART:
FORMAT BIT7PARITY
```

:SEARch<x>:UART:MODE

Function Sets the UART bus signal search mode or queries the current setting.

Syntax
```
:SEARch<x>:UART:MODE {DATA|ERRor}
:SEARch<x>:UART:MODE?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:MODE DATA
:SEARCH1:UART:MODE? -> :SEARCH1:UART:
MODE DATA
```

:SEARch<x>:UART:POLarity

Function Sets the UART bus signal search polarity or queries the current setting.

Syntax
```
:SEARch<x>:UART:
POLarity {NEGative|POSitive}
:SEARch<x>:UART:POLarity?
```
<x> = 1 or 2

Example
```
:SEARCH1:UART:POLARITY NEGATIVE
:SEARCH1:UART:POLARITY? -> :SEARCH1:
UART:POLARITY NEGATIVE
```

:SEARch<x>:UART:SPOint

Function Sets the UART bus signal search sample point or queries the current setting.

Syntax
```
:SEARch<x>:UART:SPOint {<NRf>}
:SEARch<x>:UART:SPOint?
```
<x> = 1 or 2

<NRf> = 18.8 to 90.6(%)

Example
```
:SEARCH1:UART:SPOINT 18.8
:SEARCH1:UART:SPOINT? -> :SEARCH1:UART:
SPOINT 18.8E+00
```

:SEARch<x>:UART:TRACe

Function Sets the UART bus signal search trace or queries the current setting.

Syntax
```
:SEARch<x>:UART:TRACe {<NRf>}
:SEARch<x>:UART:TRACe?
```
<x> = 1 or 2

<NRf> = 1 to 8

Example
```
:SEARCH1:UART:TRACE 1
:SEARCH1:UART:TRACE? -> :SEARCH1:UART:
TRACE 1
```

7.4 SERialbus Group

:SERialbus?
Function Queries all settings related to the serial bus setup.
Syntax :SERiabus?

:SERialbus:SETup<x>?
Function Queries all settings related to each setup of the serial bus setup.
Syntax :SERialbus:SETup<x>?
<x> = 1 or 2

:SERialbus:SETup<x>:ASETup:ABORt
Function Cancels auto setup of the serial bus setup.
Syntax :SERialbus:SETup<x>:ASETup:ABORt
<x> = 1 or 2
Example :SERIALBUS:SETUP1:ASETUP:ABORT

:SERialbus:SETup<x>:ASETup:EXECute
Function Executes auto setup of the serial bus setup.
Syntax :SERialbus:SETup<x>:ASETup:EXECute
<x> = 1 or 2
Example :SERIALBUS:SETUP1:ASETUP:EXECUTE

:SERialbus:SETup<x>:ASETup:UNDO
Function Undoes the executed auto setup of the serial bus setup.
Syntax :SERialbus:SETup<x>:ASETup:UNDO
<x> = 1 or 2
Example :SERIALBUS:SETUP1:ASETUP:UNDO

:SERialbus:SETup<x>:CANBus?
Function Queries all settings related to the CAN bus setup.
Syntax :SERialbus:SETup<x>:CANBus?
<x> = 1 or 2

:SERialbus:SETup<x>:CANBus:BRATe
Function Sets the CAN bus setup bit rate (data transfer rate) or queries the current setting.
Syntax :SERialbus:SETup<x>:CANBus:
BRATe {<NRf>|USER,<NRf>}
:SERialbus:SETup<x>:CANBus:BRATe?
<x> = 1 or 2
<NRf> = 33300, 83300, 125000, 250000, 500000, 1000000
<NRf> of USER = See section 5.3.
Example :SERIALBUS:SETUP1:CANBUS:BRATE 83300
:SERIALBUS:SETUP1:CANBUS:BRATE?
-> :SERIALBUS:SETUP1:CANBUS:BRATE 83300

:SERialbus:SETup<x>:CANBus:RECessive
Function Sets the CAN bus setup recessive level (bus level) or queries the current setting.
Syntax :SERialbus:SETup<x>:CANBus:
RECessive {HIGH|LOW}
:SERialbus:SETup<x>:CANBus:
RECessive?
<x> = 1 or 2
Example :SERIALBUS:SETUP1:CANBUS:RECESSIVE HIGH
:SERIALBUS:SETUP1:CANBUS:RECESSIVE?
-> :SERIALBUS:SETUP1:CANBUS:
RECESSIVE HIGH

:SERialbus:SETup<x>:CANBus:SPOint
Function Sets the CAN bus setup sample point or queries the current setting.
Syntax :SERialbus:SETup<x>:CANBus:
SPOint {<NRf>}
:SERialbus:SETup<x>:CANBus:SPOint?
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)
Example :SERIALBUS:SETUP1:CANBUS:SPOINT 18.8
:SERIALBUS:SETUP1:CANBUS:SPOINT?
-> :SERIALBUS:SETUP1:CANBUS:
SPOINT 18.8E+00

:SERialbus:SETup<x>:CANBus:TRACe
Function Sets the CAN bus setup trace or queries the current setting.
Syntax :SERialbus:SETup<x>:CANBus:
TRACe {<NRf>}
:SERialbus:SETup<x>:CANBus:TRACe?
<x> = 1 or 2
<NRf> = 1 to 8
Example :SERIALBUS:SETUP1:CANBUS:TRACE 1
:SERIALBUS:SETUP1:CANBUS:TRACE?
-> :SERIALBUS:SETUP1:CANBUS:TRACE 1

:SERialbus:SETup<x>:I2CBus?

Function Queries all settings related to the I2C bus setup.

Syntax
```
:SERialbus:SETup<x>:I2CBus?
```
<x> = 1 or 2

:SERialbus:SETup<x>:I2CBus:CLOCk

Function Sets the I2C bus setup clock channel or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:I2CBus:
CLOCk {<NRf>|A<y>|B<y>|C<y>|D<y>}
:SERialbus:SETup<x>:I2CBus:CLOCk?
```
<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:I2CBUS:CLOCK 1
:SERIALBUS:SETUP1:I2CBUS:CLOCK?
-> :SERIALBUS:SETUP1:I2CBUS:CLOCK 1
```

Description On 16-bit models of DLM6000, you can only select {A<x>|C<x>}..

:SERialbus:SETup<x>:I2CBus:DTRace

Function Sets the I2C bus signal analysis data channel or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:I2CBus:
DTRace {<NRf>|A<y>|B<y>|C<y>|D<y>}
:SERialbus:SETup<x>:I2CBus:DTRace?
```
<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:I2CBUS:DTRACE 1
:SERIALBUS:SETUP1:I2CBUS:DTRACE?
-> :SERIALBUS:SETUP1:I2CBUS:DTRACE 1
```

Description On 16-bit models of DLM6000, you can only select {A<x>|C<x>}..

:SERialbus:SETup<x>:LINBus?

Function Queries all settings related to the LIN bus setup.

Syntax
```
:SERialbus:SETup<x>:LINBus?
```
<x> = 1 or 2

:SERialbus:SETup<x>:LINBus:BRATe

Function Sets the LIN bus setup bit rate (data transfer rate) or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:LINBus:
BRATe {<NRf>|USER,<NRf>}
:SERialbus:SETup<x>:LINBus:BRATe?
```
<x> = 1 or 2
<NRf> = 1200, 2400, 4800, 9600, 19200
<NRf> of USER = See section 5.4.

Example
```
:SERIALBUS:SETUP1:LINBUS:BRATE 19200
:SERIALBUS:SETUP1:LINBUS:BRATE?
-> :SERIALBUS:SETUP1:LINBUS:BRATE 19200
```

:SERialbus:SETup<x>:LINBus:REVision

Function Sets the LIN bus setup revision (1.3, 2.0, or Both) or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:LINBus:
REVision {BOTH|LIN1_3|LIN2_0}
:SERialbus:SETup<x>:LINBus:REVision?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:LINBUS:
REVISION LIN1_3
:SERIALBUS:SETUP1:LINBUS:REVISION?
-> :SERIALBUS:SETUP1:LINBUS:
REVISION LIN1_3
```

:SERialbus:SETup<x>:LINBus:SPOint

Function Sets the LIN bus setup sample point or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:LINBus:
SPOint {<NRf>}
:SERialbus:SETup<x>:LINBus:SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:SERIALBUS:SETUP1:LINBUS:SPOINT 18.8
:SERIALBUS:SETUP1:LINBUS:SPOINT?
-> :SERIALBUS:SETUP1:LINBUS:
SPOINT 18.8E+00
```

:SERialbus:SETup<x>:LINBus:TRACe

Function Sets the LIN bus setup trace or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:LINBus:
TRACe {<NRf>|A<y>|B<y>|C<y>|D<y>}
:SERialbus:SETup<x>:LINBus:TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:LINBUS:TRACE 1
:SERIALBUS:SETUP1:LINBUS:TRACE?
-> :SERIALBUS:SETUP1:LINBUS:TRACE 1
```

Description On 16-bit models, you can only select {A<x>|C<x>}..

:SERialbus:SETup<x>:SPIBus?

Function Queries all settings related to the SPI bus setup.

Syntax
```
:SERialbus:SETup<x>:SPIBus?
```
<x> = 1 or 2

:SERialbus:SETup<x>:SPIBus:BITorder

Function Sets the SPI bus setup bit order or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:
BITorder {LSBFirst|MSBFirst}
:SERialbus:SETup<x>:SPIBus:BITorder?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:SPIBUS:
BITORDER LSBFIRST
:SERIALBUS:SETUP1:SPIBUS:BITORDER?
-> :SERIALBUS:SETUP1:SPIBUS:
BITORDER LSBFIRST
```

:SERialbus:SETup<x>:SPIBus:CLOCk?

Function Queries all settings related to the channel of the clock signal of the SPI bus setup.

Syntax
```
:SERialbus:SETup<x>:SPIBus:CLOCk?
```
<x> = 1 or 2

:SERialbus:SETup<x>:SPIBus:CLOCk:POLarity

Function Sets the polarity of the channel of the clock signal of the SPI bus setup.

Syntax
```
:SERialbus:SETup<x>:SPIBus:CLOCk:
POLarity {FALL|RISE}
:SERialbus:SETup<x>:SPIBus:CLOCk:
POLarity?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:SPIBUS:CLOCK:
POLARITY FALL
:SERIALBUS:SETUP1:SPIBUS:CLOCK:
POLARITY? -> :SERIALBUS:SETUP1:SPIBUS:
CLOCK:POLARITY FALL
```

:SERialbus:SETup<x>:SPIBus:CLOCk:TRACe

Function Sets the channel of the clock signal of the SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:CLOCk:
TRACe {<NRf>|A<y>|B<y>|C<y>|D<y>}
:SERialbus:SETup<x>:SPIBus:CLOCk:
TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:SPIBUS:CLOCK:
TRACE 1
:SERIALBUS:SETUP1:SPIBUS:CLOCK:TRACE?
-> :SERIALBUS:SETUP1:SPIBUS:CLOCK:
TRACE 1
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:SERialbus:SETup<x>:SPIBus:CS?

Function Queries all settings related to the channel of the chip select signal of the SPI bus setup.

Syntax
```
:SERialbus:SETup<x>:SPIBus:CS?
```
<x> = 1 or 2

:SERialbus:SETup<x>:SPIBus:CS:ACTive

Function Sets the active level of the channel of the chip select signal of the SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:CS:
ACTive {HIGH|LOW}
:SERialbus:SETup<x>:SPIBus:CS:ACTive?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:SPIBUS:CS:ACTIVE HIGH
:SERIALBUS:SETUP1:SPIBUS:CS:ACTIVE?
 -> :SERIALBUS:SETUP1:SPIBUS:CS:
ACTIVE HIGH
```

:SERialbus:SETup<x>:SPIBus:CS:TRACe

Function Sets the channel of the chip select signal of the SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:CS:
TRACe {<NRf>|A<y>|B<y>|C<y>|D<y>|ANONe
|LNONe}
:SERialbus:SETup<x>:SPIBus:CS:TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:SPIBUS:CS:TRACE 1
:SERIALBUS:SETUP1:SPIBUS:CS:TRACE?
->:SERIALBUS:SETUP1:SPIBUS:CS:TRACE 1
```

Description
- On 16-bit models, you can only select {A<x>|C<x>|ANONe>}.
- Specify ANONe when you want to use the idle time to control the analog SPI bus data start position.
- Specify LNONe when you want to use the idle time to control the logic SPI bus data start position.

:SERialbus:SETup<x>:SPIBus:DATA<x>?

Function Queries all settings related to each data of the SPI bus setup.

Syntax
```
:SERialbus:SETup<x>:SPIBus:DATA<x>?
```
<x> of SETup<x> = 1 or 2
<x> of DATA<x> = 1 or 2

:SERialbus:SETup<x>:SPIBus:DATA<x>:ACTive

Function Sets the active level of each data of the SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:DATA<x>:
ACTive {HIGH|LOW}
:SERialbus:SETup<x>:SPIBus:DATA<x>:
ACTive?
```
<x> of SETup<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:SPIBUS:DATA1:
ACTIVE HIGH
:SERIALBUS:SETUP1:SPIBUS:DATA1:ACTIVE?
-> :SERIALBUS:SETUP1:SPIBUS:DATA1:
ACTIVE HIGH
```

:SERialbus:SETup<x>:SPIBus:DATA<x>:TRACe

Function Sets each data channel of the SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:DATA<x>:
TRACe {<NRf>|A<y>|B<y>|C<y>|D<y>}
:SERialbus:SETup<x>:SPIBus:DATA<x>:
TRACe?
```
<x> of SETup<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:SPIBUS:DATA1:
TRACE 1
:SERIALBUS:SETUP1:SPIBUS:DATA1:TRACE?
-> :SERIALBUS:SETUP1:SPIBUS:DATA1:
TRACE 1
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:SERialbus:SETup<x>:SPIBus:ITIMe

Function Sets the idle time used in SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:
ITIMe {<Time>}
:SERialbus:SETup<x>:SPIBus:ITIMe?
```
<x> = 1 or 2
<Time> = 10ns to 1ms in 10-ns steps

Example
```
:SERIALBUS:SETUP1:SPIBUS:ITIME 10NS
:SERIALBUS:SETUP1:SPIBUS:ITIME?
-> :SERIALBUS:SETUP1:SPIBUS:
ITIME 10.0000E-09
```

:SERialbus:SETup<x>:SPIBus:MODE

Function Sets the wiring method (3-wire/4-wire) of the SPI bus setup or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:SPIBus:
MODE {WIRE3|WIRE4}
:SERialbus:SETup<x>:SPIBus:MODE?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:SPIBUS:MODE WIRE3
:SERIALBUS:SETUP1:SPIBUS:MODE?
-> :SERIALBUS:SETUP1:SPIBUS:MODE WIRE3
```

:SERialbus:SETup<x>:TRACe<x>?

Function Queries all settings related to each trace.

Syntax
```
:SERialbus:SETup<x>:TRACe<x>?
```
<x> of SETup<x> = 1 or 2
<x> of TRACe<x> = 1 to 8

:SERialbus:SETup<x>:TRACe<x>:HYSTeresis

Function Sets the hysteresis of the threshold level of each trace or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:TRACe<x>:
HYSTeresis {<NRf>}
:SERialbus:SETup<x>:TRACe<x>:
HYSTeresis?
```
<x> of SETup<x> = 1 or 2
<x> of TRACe<x> = 1 to 8
<NRf> = 0 to 4(div, in 0.1-div steps)

Example
```
:SERIALBUS:SETUP1:TRACE1:HYSTERESIS 1
:SERIALBUS:SETUP1:TRACE1:HYSTERESIS?
-> :SERIALBUS:SETUP1:TRACE1:
HYSTERESIS 1.000E+00
```

:SERialbus:SETup<x>:TRACe<x>:LEVel

Function Sets the threshold level of each trace or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:TRACe<x>:
LEVel {<NRf>|<Voltage>|<Current>}
:SERialbus:SETup<x>:TRACe<x>:LEVel?
```
<x> of SETup<x> = 1 or 2
<x> of TRACe<x> = 1 to 8
<NRf>, <Voltage>, <Current> = See sections 5.2 to 5.6.

Example
```
:SERIALBUS:SETUP1:TRACE1:LEVEL 0
:SERIALBUS:SETUP1:TRACE1:LEVEL?
-> :SERIALBUS:SETUP1:TRACE1:
LEVEL 0.000E+00
```

:SERialbus:SETup<x>:TYPE

Function Sets the serial bus setup type or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:TYPE {CANBus|
I2CBus|LINBus|SPIBus|UART}
:SERialbus:SETup<x>:TYPE?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:TYPE CANBUS
:SERIALBUS:SETUP1:TYPE?
-> :SERIALBUS:SETUP1:TYPE CANBUS
```

:SERialbus:SETup<x>:UART?

Function Queries all settings related to the UART bus setup.

Syntax
```
:SERialbus:SETup<x>:UART?
```
<x> = 1 or 2

:SERialbus:SETup<x>:UART:BITorder

Function Sets the UART bus setup bit order or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:
BITorder {LSBFirst|MSBFirst}
:SERialbus:SETup<x>:UART:BITorder?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:UART:BITORDER
LSBFIRST
:SERIALBUS:SETUP1:UART:BITORDER?
-> :SERIALBUS:SETUP1:UART:
BITORDER LSBFIRST
```

:SERialbus:SETup<x>:UART:BRATe

Function Sets the UART bus setup bit rate (data transfer rate) or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:
BRATe {<NRf>|USER,<NRf>}
:SERialbus:SETup<x>:UART:BRATe?
```
<x> = 1 or 2
<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

Example
```
:SERIALBUS:SETUP1:UART:BRATE 19200
:SERIALBUS:SETUP1:UART:BRATE?
-> :SERIALBUS:SETUP1:UART:BRATE 19200
```

:SERialbus:SETup<x>:UART:FORMat

Function Sets the UART bus setup data format or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:
FORMat {BIT7parity|BIT8Noparity|
BIT8Parity}
:SERialbus:SETup<x>:UART:FORMat?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:UART:
FORMAT BIT7PARITY
:SERIALBUS:SETUP1:UART:FORMAT?
-> :SERIALBUS:SETUP1:UART:
FORMAT BIT7PARITY
```

:SERialbus:SETup<x>:UART:PMODe

Function Sets the UART bus setup Parity mode or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:
PMODe {EVEN|ODD}
:SERialbus:SETup<x>:UART:PMODe?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:UART:PMODE EVEN
:SERIALBUS:SETUP1:UART:PMODE?
-> :SERIALBUS:SETUP1:UART:PMODE EVEN
```

:SERialbus:SETup<x>:UART:POLarity

Function Sets the UART bus setup polarity or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:
POLarity {NEGative|POSitive}
:SERialbus:SETup<x>:UART:POLarity?
```
<x> = 1 or 2

Example
```
:SERIALBUS:SETUP1:UART:POLARITY
NEGATIVE
:SERIALBUS:SETUP1:UART:POLARITY?
-> :SERIALBUS:SETUP1:UART:
POLARITY NEGATIVE
```

:SERialbus:SETup<x>:UART:SPOint

Function Sets the UART bus setup sample point or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:SPOint {<NRf>}
:SERialbus:SETup<x>:UART:SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:SERIALBUS:SETUP1:UART:SPOINT 18.8
:SERIALBUS:SETUP1:UART:SPOINT?
-> :SERIALBUS:SETUP1:UART:
SPOINT 18.8E+00
```

:SERialbus:SETup<x>:UART:TRACe

Function Sets the UART bus setup trace or queries the current setting.

Syntax
```
:SERialbus:SETup<x>:UART:
TRACe {<NRf>|A<y>|B<y>|C<y>|D<y>}
:SERialbus:SETup<x>:UART:TRACe?
```
<x> = 1 or 2
<NRf> = 1 to 8
<y> = 0 to 7

Example
```
:SERIALBUS:SETUP1:UART:TRACE 1
:SERIALBUS:SETUP1:UART:TRACE?
-> :SERIALBUS:SETUP1:UART:TRACE 1
```

Description On 16-bit models, you can only select {A<x>|C<x>}

:SERialbus:TLINk

Function Sets the serial bus setup trigger link or queries the current setting.

Syntax
```
:SERialbus:TLINk {OFF|SETUP1|SETUP2}
:SERialbus:TLINk?
```

Example
```
:SERIALBUS:TLINK OFF
:SERIALBUS:TLINK?
 -> :SERIALBUS:TLINK OFF
```

7.5 TRIGger Group

:TRIGger:EINTerval:EVENt<x>:CANBus?

Function Queries all settings related to the CAN bus signal trigger of the event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus?
<x> = 1 or 2
```

:TRIGger:EINTerval:EVENt<x>:CANBus:ACK

Function Sets the ACK condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
ACK {ACK|ACKBoth|DONTcare|NONack}
:TRIGger:EINTerval:EVENt<x>:CANBus:
ACK?
<x> = 1 or 2
```

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
ACK ACK
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
ACK? -> :TRIGGER:EINTERVAL:EVENT1:
CANBUS:ACK ACK
```

:TRIGger:EINTerval:EVENt<x>:CANBus:BRATe

Function Sets the bit rate (data transfer rate) of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
BRATe {<NRf>|USER,<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:
BRATe?
```
<x> = 1 or 2
<NRf> = 33300, 83300, 125000, 250000, 500000, 1000000
<NRf> of USER = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
BRATE 83300
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
BRATE? -> :TRIGGER:EINTERVAL:EVENT1:
CANBUS:BRATE 83300
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA?

Function Queries all settings related to the CAN bus signal trigger data.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
DATA?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:BORDer

Function Sets the byte order of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
DATA:BORDer {BIG|LITTle}
:TRIGger:EINTerval:EVENt<x>:CANBus:
DATA:BORDer?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
DATA:BORDER BIG:
TRIGGER:EINTERVAL:EVENT1:CANBUS:
DATA:BORDER? -> :TRIGGER:EINTERVAL:
EVENT1:CANBUS:DATA:BORDER BIG
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:CONDition

Function Sets the data condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
DATA:CONDition {BETWeen|DONTcare|
FALSe|GTHan|LTHan|ORANge|TRUE}
:TRIGger:EINTerval:EVENt<x>:CANBus:
DATA:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
DATA:CONDITION BETWEEN
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
DATA:CONDITION? -> TRIGGER:
EINTERVAL:EVENT1:CANBUS:DATA:
CONDITION BETWEEN
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA<x>

Function Sets the comparison data of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA<x> {<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:DATA1 1
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:DATA1? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:DATA1 1.0000000E+00
```

Description
- Use `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA1` when `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:CONDition GTHan` is specified.
- Use `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA2` when `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:CONDition LTHan` is specified.
- Use `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA1` to set the smaller value and `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DATA2` to set the larger value when `:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:CONDition BETWeen|ORANge` is specified.

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DLC

Function Sets the number of valid bytes (DLC) of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DLC {<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:DLC?
```
<x> = 1 or 2
<NRf> = 0 to 8

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:DLC 0
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:DLC? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:DLC 0
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:HEXA

Function Sets the CAN bus signal trigger data in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:HEXA {<String>}
```
<x> = 1 or 2
<String> = Up to 16 characters by combining '0' to 'F' and 'X' (in one-byte unit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:HEXA "A9"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:MSBLsb

Function Sets the MSB and LSB bits of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:MSBLsb?
```
<x> = 1 or 2
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:MSBLSB 1,0
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:MSBLSB? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:MSBLSB 1,0
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:PATTern

Function Sets the CAN bus signal trigger data in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:PATTern?
```
<x> = 1 or 2
<String> = Up to 64 characters by combining '0,' '1,' and 'X' (in one-byte unit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:PATTERN "11011111"
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:PATTERN? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:PATTERN "11011111"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:SIGN

Function Sets the sign of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:SIGN {SIGN|UNSign}
:TRIGger:EINTerval:EVENt<x>:CANBus:DATA:SIGN?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:SIGN SIGN
:TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:SIGN? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:DATA:SIGN SIGN
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt?

Function Queries all settings related to the ID of the extended format of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:HEXA

Function Sets the ID of the extended format of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:HEXA {<String>}
```
<x> = 1 or 2

<String> = 8 characters by combining ‘0’ to ‘F’ and ‘X’

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDEXT:HEXA "1AEF5906"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:PATTern

Function Sets the ID of the extended format of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDEXt:PATTern?
```
<x> = 1 or 2

<String> = 29 characters by combining ‘0,’ ‘1,’ and ‘X’

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDEXT:PATTERN "11001011011100001111011011111"
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDEXT:PATTERN? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDEXT:PATTERN "11001011011100001110111011111"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR?

Function Queries all settings related to the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>?

Function Queries all settings related to each ID of the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>?
```
<x> of EVENt<x> = 1 or 2

<x> of ID<x> = 1 to 4

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:ACK

Function Sets each ACK condition of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:ACK {ACK|ACKBoth|DONTcare|NONack}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:ACK?
```
<x> of EVENt<x> = 1 or 2

<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:ACK ACK
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:ACK? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:ACK ACK
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA?

Function Queries all settings related to each data of the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA?
```
<x> of EVENt<x> = 1 or 2

<x> of ID<x> = 1 to 4

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:BORDer

Function Sets byte order of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:BORDer {BIG|LITTle}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:BORDer?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:BORDER BIG
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:BORDER? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:BORDER BIG
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition

Function Sets each data condition of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition {BETWeen|DONTcare|FALSe|GTHan|LTHan|ORANge|TRUE}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:CONDITION BETWEEN
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:CONDITION? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:CONDITION BETWEEN
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA<x>

Function Sets comparison data of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA<x> {<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<x> of DATA<x> = 1 or 2
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:DATA1 1
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:DATA1? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:DATA1 1.0000000E+00
```

Description
- Use `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA1` when `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition GTHan` is specified.
- Use `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA2` when `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition LTHan` is specified.
- Use `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA1` to set the smaller value and `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DATA2` to set the larger value when `:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:CONDition BETWeen|ORANge` is specified.

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DLC

Function Sets the number of valid bytes (DLC) of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DLC {<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:DLC?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<NRf> = 0 to 8

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:DLC 0
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:DLC? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:DATA:DLC 0
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:HEXA

Function Sets each data of the OR condition of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:HEXA {<String>}
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<String> = Up to 16 characters by combining '0' to 'F' and 'X' (in one-byte unit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:HEXA "A9"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:MSBLsb

Function Sets the MSB and LSB bits of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:MSBLsb?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:MSBLSB 1,0
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:MSBLSB? -> :TRIGGER:
EINTERVAL:EVENT1:CANBUS:IDOR:ID1:
DATA:MSBLSB 1,0
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:PATTern

Function Sets each data of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:PATTern?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<String> = Up to 64 characters by combining '0,' '1,' and 'X' (in one-byte unit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:PATTERN "11011111"
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:PATTERN? -> :TRIGGER:
EINTERVAL:EVENT1:CANBUS:IDOR:ID1:
DATA:PATTERN "11011111"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:DATA:SIGN

Function Sets sign of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:SIGN {SIGN|UNSign}
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:DATA:SIGN?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:SIGN SIGN
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:DATA:SIGN? -> :TRIGGER:
EINTERVAL:EVENT1:CANBUS:IDOR:ID1:
DATA:SIGN SIGN
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:FORMat

Function Sets each message format (standard or extended) of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:FORMat {STD|EXT}
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:FORMat?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:FORMAT STD
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:FORMAT? -> :TRIGGER:
EINTERVAL:EVENT1:CANBUS:IDOR:ID1:
FORMAT STD
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt?

Function Queries all settings related to the ID of each extended format of the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:IDEXt?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt:HEXA

Function Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDOR:ID<x>:IDEXt:HEXA {<String>}
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<String> = 8 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDOR:ID1:IDEXT:HEXA "1AEF5906"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt:PATTern

Function Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDEXt:PATTern?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<String> = 29 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDEXT:PATTERN "11001011011100001110111011111"
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDEXT:PATTERN? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDEXT:PATTERN "11001011011100001110111011111"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd?

Function Queries all settings related to the ID of each standard format of the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:HEXA

Function Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:HEXA {<String>}
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<String> = 3 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDSTD:HEXA "5DF"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:PATTern

Function Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:IDSTd:PATTern?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4
<String> = 11 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDSTD:PATTERN "10111011111"
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDSTD:PATTERN? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:IDSTD:PATTERN "10111011111"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:MODE

Function Enables or disables each condition of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:MODE {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:MODE?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:MODE ON
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:MODE? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:MODE 1
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:RTR

Function Sets each RTR of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:RTR {DATA|DONTcare|REMote}
:TRIGger:EINTerval:EVENt<x>:CANBus:IDOR:ID<x>:RTR?
```
<x> of EVENt<x> = 1 or 2
<x> of ID<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:RTR DATA
:TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:RTR? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:IDOR:ID1:RTR DATA
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDSTd?

Function Queries all settings related to the ID of the standard format of the CAN bus signal trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDSTd?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:CANBus:IDSTd:HEXA

Function Sets the ID of the standard format of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDSTd:HEXA {<String>}
```
<x> = 1 or 2
<String> = 8 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDSTD:HEXA "5DF"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:IDSTd:PATTern

Function Sets the ID of the standard format of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDSTd:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:CANBus:
IDSTd:PATTern?
```
<x> = 1 or 2
<String> = 11 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDSTD:PATTERN "10111011111"
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
IDSTD:PATTERN? -> :TRIGGER:EINTERVAL:
EVENT1:CANBUS:IDSTD:
PATTERN "10111011111"
```

:TRIGger:EINTerval:EVENt<x>:CANBus:MODE

Function Sets the CAN bus signal trigger mode or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
MODE {EFRame|IDEXt|IDOR|IDSTd|SOF}
:TRIGger:EINTerval:EVENt<x>:CANBus:
MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
MODE EFRAME
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
MODE? -> :TRIGGER:EINTERVAL:EVENT1:
CANBUS:MODE EFRAME
```

:TRIGger:EINTerval:EVENt<x>:CANBus:RECessive

Function Sets the recessive level (bus level) of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
RECessive {HIGH|LOW}
:TRIGger:EINTerval:EVENt<x>:CANBus:
RECessive?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
RECESSIVE HIGH
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
RECESSIVE? -> :TRIGGER:EINTERVAL:
EVENT1:CANBUS:RECESSIVE HIGH
```

:TRIGger:EINTerval:EVENt<x>:CANBus:RTR

Function Sets the RTR of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
RTR {DATA|DONTcare|REMote}
:TRIGger:EINTerval:EVENt<x>:CANBus:
RTR?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
RTR DATA
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
RTR? -> :TRIGGER:EINTERVAL:EVENT1:
CANBUS:RTR DATA
```

:TRIGger:EINTerval:EVENt<x>:CANBus:SOURce

Function Sets the trigger source of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:
SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:
SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:CANBUS:
SOURCE? -> :TRIGGER:EINTERVAL:EVENT1:
CANBUS:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:CANBus:SPOint

Function Sets the sample point of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:CANBus:SPOint {<NRf>}
:TRIGger:EINTerval:EVENt<x>:CANBus:SPOint?
```
<x> = 1 or 2
<NRf> = 18.8 to 90.6(%)

Example
```
:TRIGGER:EINTERVAL:EVENT1:CANBUS:SPOINT 18.8
:TRIGGER:EINTERVAL:EVENT1:CANBUS:SPOINT? -> :TRIGGER:EINTERVAL:EVENT1:CANBUS:SPOINT 18.8E+00
```

:TRIGger:EINTerval:EVENt<x>:I2CBus?

Function Queries all settings related to the I^2C bus trigger of the event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa?

Function Queries all settings related to the address of the I^2C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address?

Function Queries all settings related to the 10-bit address of the I^2C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:HEXA

Function Sets the 10-bit address of the I^2C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:HEXA {<String>}
```
<x> = 1 or 2
<String> = 3 characters by combining '0' to 'F' and 'X' (bit 8 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:ADATA:
BIT10ADDRESS:HEXA "7AB"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:PATTern

Function Sets the 10-bit address of the I^2C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT10address:PATTern?
```
<x> = 1 or 2
<String> = 11 characters by combining '0', '1', and 'X' (bit 8 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:ADATA:BIT10ADDRESS:
PATTERN "10111011111"
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:ADATA:BIT10ADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:ADATA:BIT10ADDRESS:
PATTERN "10111011111"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress?

Function Queries all settings related to the 7-bit address of the I^2C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress:HEXA

Function Sets the 7-bit address of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7ADdress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7ADDRESS:HEXA "DE"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7ADdress:PATTern

Function Sets the 7-bit address of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7ADdress:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7ADdress:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0', '1', and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7ADDRESS:PATTERN "11011110"
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7ADDRESS:PATTERN? -> :TRIGGER:
EINTERVAL:EVENT1:I2CBUS:ADATA:
BIT7ADDRESS:PATTERN "11011110"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub?

Function Queries all settings related to the 7-bit + Sub address of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + Sub address of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:ADDRess?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA

Function Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:ADDRess:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:BIT7APSUB:ADDRESS:HEXA "AB"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern

Function Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:ADDRess:PATTern
{<String>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:ADDRess:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0', '1', and 'X' (bit 0 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7APSUB:ADDRESS:PATTERN "10101011"
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7APSUB:ADDRESS:PATTERN? -> :TRIGGER:
EINTERVAL:EVENT1:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:PATTERN "10101011"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:SADDress?

Function Queries all settings related to the Sub address of the 7-bit + Sub address of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:SADDress?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:SADDress:HEXA

Function Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:SADDress:HEXA
{<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7APSUB:SADDRESS:HEXA "EF"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:BIT7APsub:SADDress:PATTern

Function Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:SADDress:PATTern
{<String>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:BIT7APsub:SADDress:PATTern?
```
<x> = 1 or 2
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7APSUB:SADDRESS:PATTERN "10101011"
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
BIT7APSUB:SADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:BIT7APSUB:SADDRESS:
PATTERN "10101011"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:ADATa:TYPE

Function Sets the address type of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:TYPE {BIT10address|BIT7ADdress|
BIT7APsub}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
ADATa:TYPE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:TYPE BIT10ADDRESS
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
ADATA:
TYPE? -> :TRIGGER:EINTERVAL:EVENT1:
I2CBUS:ADATA:TYPE BIT10ADDRESS
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:CLOCk?

Function Queries all settings related to the clock of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
CLOCk?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:CLOCk:SOURce

Function Sets the clock trace of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
CLOCk:SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
CLOCk:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
CLOCK:
SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
CLOCK:
SOURCE? -> :TRIGGER:EINTERVAL:EVENT1:
I2CBUS:CLOCK:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA?

Function Queries all settings related to the data of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
DATA?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:BYTE

Function Sets the number of data bytes of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
DATA:BYTE {<NRf>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
DATA:BYTE?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:
BYTE 1
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:
BYTE? -> :TRIGGER:EINTERVAL:EVENT1:
I2CBUS:DATA:BYTE 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:CONDition

Function Sets the determination method (match or not match) of the data of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
DATA:CONDition {FALSe|TRUE}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
DATA:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:
CONDITION TRUE
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:
CONDITION? -> :TRIGGER:EINTERVAL:
EVENT1:I2CBUS:DATA:CONDITION TRUE
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:DPOSition

Function Sets the position for comparing the data pattern of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:DPOSition {<NRf>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:DPOSition?
```
<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:DPOSITION 1
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:DPOSITION? -> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:DPOSITION 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:HEXA<x>

Function Sets the data of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:HEXA<x> {<String>}
```
<x> of EVENt<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:HEXA1 "AB"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:MODE

Function Enables/Disables the data conditions of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:MODE {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:MODE ON
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:MODE? -> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:MODE 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PATTern<x>

Function Sets the data of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PATTern<x> {<String>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PATTern<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:PATTERN1 "10101011"
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:PATTERN1? -> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:PATTERN1 "10101011"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PMODe

Function Sets the pattern comparison start position mode of the data of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PMODe {DONTcare|SELect}
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:PMODe?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:PMODE SELECT
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:PMODE? -> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:PMODE SELECT
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:SOURce

Function Sets the data trace of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:DATA:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:SOURCE? -> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:DATA:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl?

Function Queries all settings related to the general call of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:BIT7maddress?

Function Queries all settings related to the 7-bit master address of the general call of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
GCALl:BIT7maddress?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:BIT7maddress:HEXA

Function Sets the 7-bit master address of the general call of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
GCALl:BIT7maddress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is fixed 1)

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
GCALL:
BIT7MADDRESS:HEXA "AB"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:BIT7maddress:PATTern

Function Sets the 7-bit master address of the general call of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
GCALl:BIT7maddress:PATTern {<String>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
GCALl:BIT7maddress:PATTern?
```
<x> = 1 or 2
<String> = 7 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
GCALL:BIT7MADDRESS:PATTERN "1010101"
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
GCALL:BIT7MADDRESS:PATTERN?
 -> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:
GCALL:BIT7MADDRESS:PATTERN "1010101"
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:GCALl:SBYTe (Second Byte)

Function Sets the second byte type of the general call of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
GCALl:SBYTe {BIT7maddress|DONTcare|H04|
H06}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
GCALl:SBYTe?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
GCALL:SBYTE BIT7MADDRESS
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
GCALL:SBYTE?
-> :TRIGGER:EINTERVAL:EVENT1:
I2CBUS:GCALL:SBYTE BIT7MADDRESS
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:MODE

Function Sets the trigger mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
MODE {ADATa|ESTart|GCALl|NAIGnore|
SBHSmode}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
MODE ADATA
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:MODE?
-> :TRIGGER:EINTERVAL:EVENT1:I2CBUS:
MODE ADATA
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore?

Function Queries all settings related to the NON ACK ignore mode of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore:HSMode

Function Sets whether to ignore NON ACK in high speed mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore:HSMode {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore:HSMode?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
NAIGNORE:HSMODE ON
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
NAIGNORE:HSMODE? -> :TRIGGER:EINTERVAL:
EVENT1:I2CBUS:NAIGNORE:HSMODE 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore:RACCess

Function Sets whether to ignore NON ACK in read access mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore:RACCess {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore:RACCess?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
NAIGNORE:RACCESS ON
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
NAIGNORE:RACCESS? -> :TRIGGER:
EINTERVAL:
EVENT1:I2CBUS:NAIGNORE:RACCESS 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:NAIGnore:SBYTe (Start Byte)

Function Sets whether to ignore NON ACK in the start byte of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore:SBYTe {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
NAIGnore:SBYTe?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
NAIGNORE:SBYTE ON
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
NAIGNORE:SBYTE? -> :TRIGGER:EINTERVAL:
EVENT1:I2CBUS:NAIGNORE:SBYTE 1
```

:TRIGger:EINTerval:EVENt<x>:I2CBus:SBHSmode?

Function Queries all settings related to the start byte and high speed mode of the I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
SBHSmode?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:I2CBus:SBHSmode:TYPE

Function Sets the type of the start byte or high speed mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:I2CBus:
SBHSmode:TYPE {HSMode|SBYTe}
:TRIGger:EINTerval:EVENt<x>:I2CBus:
SBHSmode:TYPE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
SBHSMODE:TYPE HSMODE
:TRIGGER:EINTERVAL:EVENT1:I2CBUS:
SBHSMODE:TYPE? -> :TRIGGER:EINTERVAL:
EVENT1:I2CBUS:SBHSMODE:TYPE HSMODE
```

:TRIGger:EINTerval:EVENt<x>:LINBus?

Function Queries all settings related to LIN bus signal triggers of each event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LINBus?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LINBus:BRATe

Function Sets the LIN bus signal trigger bitrate (data transfer rate) or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LINBus:
BRATe {<NRf>|USER,<NRf>}
:TRIGger:EINTerval:EVENt<x>:LINBus:
BRATe?
```
<x> = 1 or 2
<NRf> = 1200, 2400, 4800, 9600, 19200
<NRf> for USER = See the main unit User's Manual.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LINBUS:
BRATE 19200
:TRIGGER:EINTERVAL:EVENT1:LINBUS:BRATE?
-> :TRIGGER:EINTERVAL:EVENT1:LINBUS:
BRATE 19200
```

:TRIGger:EINTerval:EVENt<x>:LINBus:SOURce

Function Sets the LIN bus signal trigger source or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LINBus:
SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:LINBus:
SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:LINBUS:
SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:LINBUS:
SOURCE? -> :TRIGGER:EINTERVAL:EVENT1:
LINBUS:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus?

Function Queries all settings related to the logic I²C bus trigger for each event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa?

Function Queries all settings related to the address of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT10address?

Function Queries all settings related to the 10-bit address of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT10address?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT10address:HEXA

Function Sets the 10-bit address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT10address:HEXA
{<string>}
```
<x> = 1 or 2
<string> = combination of 3 characters (0-F, and X), where bit 8 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT10ADDRESS:HEXA "7AB"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT10address:PATTern

Function Sets the 10-bit address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT10address:PATTern
{<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT10address:PATTern?
```
<x> = 1 or 2
<string> = combination of 11 characters (0, 1, and X), where bit 8 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT10ADDRESS:
PATTERN "10111011111"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT10ADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT10ADDRESS:
PATTERN "10111011111"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7ADdress?

Function Queries all settings related to the 7-bit address of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7ADdress?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7ADdress:HEXA

Function Sets the 7-bit address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7ADdress:HEXA
{<string>}
```
<x> = 1 or 2
<string> = combination of 2 characters (0-F, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7ADDRESS:HEXA "DE"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7ADdress:PATTern

Function Sets the 7-bit address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7ADdress:
PATTern {<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7ADdress:PATTern?
```
<x> = 1 or 2
<string> = combination of 8 characters (0, 1, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7ADDRESS:PATTERN
"11011110"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7ADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7ADDRESS:
PATTERN "11011110"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub?

Function Queries all settings related to the 7-bit + Sub address of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:ADDRess?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA

Function Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:ADDRess:
HEXA {<string>}
```
<x> = 1 or 2
<string> = combination of 2 characters (0-F, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:
ADATA:BIT7APSUB:ADDRESS:HEXA "AB"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern

Function Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:ADDRess:
PATTern {<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:ADDRess:
PATTern?
```
<x> = 1 or 2
<string> = combination of 8 characters (0, 1, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7APSUB:ADDRESS:
PATTERN "10101011"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:
ADATA:BIT7APSUB:ADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7APSUB:ADDRESS:
PATTERN "10101011"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:SADDress?

Function Queries all settings related to the sub address of the 7-bit + Sub address of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:SADDress?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:HEXA

Function Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:SADDress:HEXA
{<string>}
```
<x> = 1 or 2
<string> = Combination of up to 2 characters (0-F and X)

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7APSUB:SADDRESS:
HEXA "EF"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:PATTern

Function Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:SADDress:
PATTern {<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:BIT7APsub:SADDress:
PATTern?
```
<x> = 1 or 2
<string> = combination of 8 characters (0, 1, and X).

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7APSUB:SADDRESS:
PATTERN "10101011"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:
ADATA:BIT7APSUB:SADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:BIT7APSUB:SADDRESS:
PATTERN "10101011"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:ADATa:TYPE

Function Sets the address type of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:TYPE {BIT10address|
BIT7ADdress|BIT7APsub}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:ADATa:TYPE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:TYPE BIT10ADDRESS
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:TYPE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:ADATA:TYPE BIT10ADDRESS
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:CLOCk?

Function Queries all settings related to the clock of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:CLOCk?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:CLOCk:SOURce

Function Sets the clock trace for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:CLOCk:SOURce {A<y>|B<y>|C<y>|D<y>}
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:CLOCk:SOURce?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:CLOCK:SOURCE A0
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:CLOCK:SOURCE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:CLOCK:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA?

Function Queries all settings related to the data of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:BYTE

Function Sets the number of settings for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:BYTE {<NRf>}
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:BYTE?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:BYTE 1
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:BYTE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:BYTE 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:CONDition

Function Sets the determination method for the data of the logic I²C bus trigger (match / no match) or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:CONDition {FALSe|TRUE}
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:CONDITION FALSE
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:CONDITION?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:CONDITION FALSE
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:DPOSition

Function Sets the pattern comparison position for the data of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:DPOSition {<NRf>}
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:DPOSition?
```
<x> = 1 or 2
<NRf>=0 to 9999

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:DPOSITION 1
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:DPOSITION?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:DATA:DPOSITION 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:HEXA<x>

Function Sets the data of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:HEXA<x> {<string>}
```
<x> of EVENt<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<string> = Combination of up to 2 characters (0-F and X)

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:I2CBUS:
DATA:HEXA1 "AB"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:MODE

Function Enables/disables the data conditions of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:MODE {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:MODE ON
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:MODE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:MODE 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:PATTern<x>

Function Sets the data for the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:PATTern<x> {<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:PATTern<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<string> = combination of 8 characters (0, 1, and X).

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:PATTERN1 "10101011"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:PATTERN1?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:PATTERN1 "10101011"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:PMODe

Function Sets the pattern comparison start position for the data of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:PMODe {DONTcare|SELect}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:PMODe?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:PMODE DONTCARE
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:PMODE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:PMODE DONTCARE
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:DATA:SOURce

Function Sets the data trace for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:SOURce {A<y>|B<y>|C<y>|
D<y>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:DATA:SOURce?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:SOURCE A0
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:SOURCE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:DATA:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl?

Function Queries all settings related to the general call of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:BIT7maddress?

Function Queries all settings related to the 7-bit master address of the general call of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl:BIT7maddress?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:BIT7maddress:HEXA

Function Sets the 7-bit master address of the general call of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl:BIT7maddress:
HEXA {<string>}
```
<x> = 1 or 2
<string> = combination of 2 characters (0-F and X), where bit 0 is fixed to '1.'

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:BIT7MADDRESS:HEXA "AB"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:BIT7maddress:PATTern

Function Sets the 7-bit master address of the general call of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl:BIT7maddress:PATTern
{<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl:BIT7maddress:PATTern?
```
<x> = 1 or 2
<string> = combination of 7 characters (0, 1, and X).

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:BIT7MADDRESS:PATTERN
"1010101"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:BIT7MADDRESS:PATTERN?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:BIT7MADDRESS:
PATTERN "1010101"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:GCALl:SBYTe (Second Byte)

Function Sets the type of the second byte of the general call of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl:SBYTe {BIT7maddress|
DONTcare|H04|H06}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:GCALl:SBYTe?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:SBYTE BIT7MADDRESS
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:SBYTE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:GCALL:SBYTE BIT7MADDRESS
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:MODE

Function Sets the trigger mode for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:MODE {ADATa|ESTart|GCALl|
NAIGnore|SBHSmode}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:MODE ADATA
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:MODE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:MODE ADATA
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore?

Function Queries all settings related to the NON-ACK Ignore mode of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore:HSMode

Function Sets whether to ignore NON ACK in high speed mode of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore:HSMode {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore:HSMode?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:HSMODE ON
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:HSMODE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:HSMODE 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore:RACCess

Function Sets whether to ignore NON ACK in read access mode of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore:RACCess {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore:RACCess?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:RACCESS ON
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:RACCESS?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:RACCESS 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:NAIGnore:SBYTe (Start Byte)

Function Sets whether to ignore NON ACK in the start byte of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore:SBYTe {<Boolean>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:NAIGnore:SBYTe?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:SBYTE ON
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:SBYTE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:NAIGNORE:SBYTE 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:SBHSmode?

Function Queries all settings related to the start byte/high speed mode of the logic I²C bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:SBHSmode?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:I2CBus:SBHSmode:TYPE

Function Sets the type of the start byte/high speed mode of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:SBHSmode:TYPE {HSMode|SBYTe}
:TRIGger:EINTerval:EVENt<x>:LOGic:
I2CBus:SBHSmode:TYPE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:SBHSMODE:TYPE HSMODE
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:SBHSMODE:TYPE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
I2CBUS:SBHSMODE:TYPE HSMODE
```

:TRIGger:EINTerval:EVENt<x>:LOGic:LINBus?

Function Queries all settings related to the logic LIN bus signal trigger of each event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
LINBus?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:LINBus:BRATe

Function Sets the bit rate (data transfer rate) of the logic LIN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
LINBus:BRATe {<NRf>|USER,<NRf>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
LINBus:BRATe?
```
<x> = 1 or 2

<NRf>=1200,2400,4800,9600,19200

<NRf> for USER = See the main unit user's manual.

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
LINBUS:BRATE 19200
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
LINBUS:BRATE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
LINBUS:BRATE 19200
```

:TRIGger:EINTerval:EVENt<x>:LOGic:LINBus:SOURce

Function Sets the trigger source of the logic LIN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
LINBus:SOURce {A<y>|B<y>|C<y>|D<y>}
:TRIGger:EINTerval:EVENt<x>:LOGic:
LINBus:SOURce?
```
<x> = 1 or 2

<y> = 0 to 7

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
LINBUS:SOURCE A0
:TRIGGER:EINTERVAL:EVENT1:LOGIC:
LINBUS:SOURCE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:
LINBUS:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus?

Function Queries all settings related to the logic SPI bus trigger for each event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:
SPIBus?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:BITorder

Function Sets the bit order for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:BITorder {LSBFirst|MSBFirst}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:BITorder?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:BITORDER LSBFIRST
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:BITORDER?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:BITORDER LSBFIRST
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk?

Function Queries all settings related to the clock of the logic SPI bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock trace for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:POLarity {FALL|RISE}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:POLarity?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CLOCK:POLARITY FALL
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CLOCK:POLARITY?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CLOCK:POLARITY FALL
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:SOURce

Function Sets the clock trace for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:SOURce {A<y>|B<y>|C<y>|D<y>}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CLOCk:SOURce?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CLOCK:SOURCE A0
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CLOCK:SOURCE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CLOCK:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS?

Function Queries all settings related to the chip select of the logic SPI bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:ACTive

Function Sets the active level of the chip select for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:ACTive {HIGH|LOW}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:ACTive?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CS:ACTIVE HIGH
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CS:ACTIVE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CS:ACTIVE HIGH
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:SOURce

Function Sets the chip select trace for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:SOURce {A<y>|B<y>|C<y>|D<y>}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:CS:SOURce?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CS:SOURCE A0
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CS:SOURCE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:CS:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>?

Function Queries all settings related to each data of the logic SPI bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Description DATA2 is only valid when `:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:MODE WIRE4`

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:BYTE

Function Sets the number of settings for each data of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:BYTE {<NRf>}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:BYTE?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:BYTE 1
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:BYTE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:BYTE 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:CONDition

Function Sets the determination method for the data of the logic SPI bus trigger (match / no match) or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:CONDition {FALSe|TRUE}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:CONDition?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:CONDITION FALSE
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:CONDITION?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:CONDITION FALSE
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:DPOSition

Function Sets the pattern comparison start position for the data of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:DPOSition {<NRf>}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:DPOSition?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:DPOSITION 1
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:DPOSITION?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:DPOSITION 1
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:HEXA<x>

Function Sets the data of the logic SPI bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:HEXA<x> {<string>}
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<string> = Combination of up to 2 characters (0-F and X)

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:
DATA1:HEXA1 "AB"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:PATTern<x>

Function Sets each data of the logic SPI bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:PATTern<x> {<string>}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:PATTern<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<string> = combination of 8 characters (0, 1, and X).

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:PATTERN1 "10101011"
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:PATTERN1?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:PATTERN1 "10101011"
```

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:SOURce

Function Sets the trace of each data of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:SOURce {A<y>|B<y>|C<y>|D<y>}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:DATA<x>:SOURce?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<y> = 0 to 7

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:SOURCE A0
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:SOURCE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:DATA1:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:MODE

Function Sets the wiring method (3-wire/4-wire) of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:MODE {WIRE3|WIRE4}
:TRIGger:EINTerval:EVENt<x>:LOGic:SPIBus:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:MODE WIRE3
:TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:MODE?
-> :TRIGGER:EINTERVAL:EVENT1:LOGIC:SPIBUS:MODE WIRE3
```

:TRIGger:EINTerval:EVENt<x>:SPIBus?

Function Queries all settings related to the SPI bus trigger of the event.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:SPIBus:BITorder

Function Sets the bit order of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:BITorder {LSBFirst|MSBFirst}
:TRIGger:EINTerval:EVENt<x>:SPIBus:BITorder?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:BITORDER LSBFIRST
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:BITORDER? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:BITORDER LSBFIRST
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk?

Function Queries all settings related to the clock of the SPI bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock trace of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:POLarity {FALL|RISE}
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:POLarity?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CLOCK:
POLARITY FALL
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CLOCK:
POLARITY? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:CLOCK:POLARITY FALL
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:SOURce

Function Sets the clock trace of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:SPIBus:CLOCk:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CLOCK:SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CLOCK:SOURCE? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:CLOCK:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:CS?

Function Queries all settings related to the chip select of the SPI bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS?
```
<x> = 1 or 2

:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:ACTive

Function Sets the active level of the chip select of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:ACTive {HIGH|LOW}
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:ACTive?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CS:ACTIVE HIGH
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CS:ACTIVE? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:CS:ACTIVE HIGH
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:SOURce

Function Sets the chip select trace of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:SPIBus:CS:SOURce?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CS:SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:CS:SOURCE? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:CS:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>?

Function Queries all settings related to the data of the SPI bus trigger.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Description DATA2 is valid when `:TRIGger:EINTerval:EVENt<x>:SPIBus:MODE WIRE4`.

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:BYTE

Function Sets the number of bytes of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:BYTE {<NRf>}
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:BYTE?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:
BYTE 1
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:
BYTE? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:BYTE 1
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:CONDition

Function Sets the determination method (match or not match) of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:CONDition {FALSe|TRUE}
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:CONDition?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:CONDITION TRUE
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:CONDITION? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:CONDITION TRUE
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:DPOSition

Function Sets the pattern comparison start position of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:DPOSition {<NRf>}
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:DPOSition?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:DPOSITION 1
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:DPOSITION? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:DPOSITION 1
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:HEXA<x>

Function Sets the data of the SPI bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:HEXA<x> {<String>}
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:
HEXA1 "AB"
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:PATTern<x>

Function Sets the data of the SPI bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:PATTern<x> {<String>}
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:PATTern<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:PATTERN1 "10101011"
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:PATTERN1?
 -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:PATTERN1 "10101011"
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:SOURce

Function Sets the trace of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:SOURce {<NRf>}
:TRIGger:EINTerval:EVENt<x>:SPIBus:DATA<x>:SOURce?
```
<x> of EVENt<x> = 1 or 2
<x> of DATA<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:SOURCE 1
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:SOURCE? ->
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:DATA1:SOURCE 1
```

:TRIGger:EINTerval:EVENt<x>:SPIBus:MODE

Function Sets the wiring system of the SPI bus trigger (three-wire or four-wire) or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:SPIBus:MODE {WIRE3|WIRE4}
:TRIGger:EINTerval:EVENt<x>:SPIBus:MODE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:MODE WIRE3
:TRIGGER:EINTERVAL:EVENT1:SPIBUS:MODE? -> :TRIGGER:EINTERVAL:EVENT1:SPIBUS:MODE WIRE3
```

:TRIGger:EINTerval:EVENt<x>:STATe:CHANnel<x>

Function Sets the condition to be satisfied of the channel or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:STATe:CHANnel<x> {DONTcare|HIGH|IN|LOW|OUT}
:TRIGger:EINTerval:EVENt<x>:STATe:CHANnel<x>?
```
<x> of EVENt<x> = 1 or 2
<x> of CHANnel<x> = 1 to 4

Example
```
:TRIGGER:EINTERVAL:EVENT1:STATE:CHANNEL1 HIGH
:TRIGGER:EINTERVAL:EVENT1:STATE:CHANNEL1? -> :TRIGGER:EINTERVAL:EVENT1:STATE:CHANNEL1 HIGH
```

Description
- This command is valid when `:TRIGger:EINTerval:EVENt<x>:TYPE I2CBus`.

:TRIGger:EINTerval:EVENt<x>:TYPE

Function Sets the trigger type of the event or queries the current setting.

Syntax
```
:TRIGger:EINTerval:EVENt<x>:
TYPE {CANBus|EDGE|EQUalify|I2CBus|
LINBus|LI2Cbus|LLINbus|LSPAttern|
LSPIbus|LPSTate|LPULse|LQUalify|LSTate|
PQUalify|PSTate|PULSe|SPATtern|SPIBus|
STATe}
:TRIGger:EINTerval:EVENt<x>:TYPE?
```
<x> = 1 or 2

Example
```
:TRIGGER:EINTERVAL:EVENT1:TYPE CANBUS
:TRIGGER:EINTERVAL:EVENT1:TYPE?
-> :TRIGGER:EINTERVAL:EVENT1:
TYPE CANBUS
```

Description {LI2Cbus|LLINbus|LSPAttern|LSPIbus|LPSTate|LPULse|LQUalify|LSTate} are only available on the DLM6000.

:TRIGger:ENHanced:CANBus?

Function Queries all settings related to the CAN bus signal trigger.

Syntax
```
:TRIGger:ENHanced:CANBus?
```

:TRIGger:ENHanced:CANBus:ACK

Function Sets the ACK condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:ACK {ACK|
ACKBoth|DONTcare|NONack}
:TRIGger:ENHanced:CANBus:ACK?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:ACK ACK
:TRIGGER:ENHANCED:CANBUS:ACK?
-> :TRIGGER:ENHANCED:CANBUS:ACK ACK
```

:TRIGger:ENHanced:CANBus:BRATe

Function Sets the bit rate (data transfer rate) of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:BRATe
{<NRf>|USER,<NRf>}
:TRIGger:ENHanced:CANBus:BRATe?
```
<NRf> = 33300, 83300, 125000, 250000, 500000, 1000000
<NRf> of USER = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:ENHANCED:CANBUS:BRATE 83300
:TRIGGER:ENHANCED:CANBUS:BRATE?
-> :TRIGGER:ENHANCED:CANBUS:BRATE 83300
```

:TRIGger:ENHanced:CANBus:DATA?

Function Queries all settings related to the CAN bus signal trigger data.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA?
```

:TRIGger:ENHanced:CANBus:DATA:BORDer

Function Sets the byte order of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:BORDer
{BIG|LITTle}
:TRIGger:ENHanced:CANBus:DATA:BORDer?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:
BORDER BIG
:TRIGGER:ENHANCED:CANBUS:DATA:
BORDER? -> :TRIGGER:ENHANCED:CANBUS:
DATA:BORDER BIG
```

:TRIGger:ENHanced:CANBus:DATA:CONDition

Function Sets the data condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:
CONDition {BETWeen|DONTcare|FALSe|
GTHan|LTHan|ORANge|TRUE}
:TRIGger:ENHanced:CANBus:DATA:
CONDition?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:
CONDITION BETWEEN
:TRIGGER:ENHANCED:CANBUS:DATA:
CONDITION? -> :TRIGGER:ENHANCED:
CANBUS:DATA:CONDITION BETWEEN
```

:TRIGger:ENHanced:CANBus:DATA:DATA<x>

Function Sets the comparison data of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:
DATA<x> {<NRf>}
:TRIGger:ENHanced:CANBus:DATA:DATA<x>?
```
<x> = 1 or 2
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:DATA1 1
:TRIGGER:ENHANCED:CANBUS:DATA:
DATA1? -> :TRIGGER:ENHANCED:CANBUS:
DATA:DATA1 1.0000000E+00
```

Description
- Use `:TRIGger:ENHANCED:CANBus:DATA:DATA1` when `:TRIGger:ENHANCED:CANBus:DATA:CONDition GTHan` is specified.
- Use `:TRIGger:ENHANCED:CANBus:DATA:DATA2` when `:TRIGger:ENHANCED:CANBus:DATA:CONDition LTHan` is specified.
- Use `:TRIGger:ENHANCED:CANBus:DATA:DATA1` to set the smaller value and `:TRIGger:ENHANCED:CANBus:DATA:DATA2` to set the larger value when `:TRIGger:ENHANCED:CANBus:DATA:CONDition BETWeen|ORANge` is specified.

:TRIGger:ENHanced:CANBus:DATA:DLC

Function Sets the number of valid bytes (DLC) of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:DLC {<NRf>}
:TRIGger:ENHanced:CANBus:DATA:DLC?
```
<NRf> = 0 to 8

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:DLC 0
:TRIGGER:ENHANCED:CANBUS:DATA:DLC?
-> :TRIGGER:ENHANCED:CANBUS:DATA:DLC 0
```

:TRIGger:ENHanced:CANBus:DATA:HEXA

Function Sets the CAN bus signal trigger data in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:HEXA {<String>}
```
<String> = Up to 16 characters by combining '0' to 'F' and 'X' (in one-byte unit)

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:HEXA "A9"
```

:TRIGger:ENHanced:CANBus:DATA:MSBLsb

Function Sets the MSB and LSB bits of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger:ENHanced:CANBus:DATA:MSBLsb?
```
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:MSBLSB 1,0
:TRIGGER:ENHANCED:CANBUS:DATA:MSBLSB? -> :TRIGGER:ENHANCED:CANBUS:DATA:MSBLSB 1,0
```

:TRIGger:ENHanced:CANBus:DATA:PATTern

Function Sets the CAN bus signal trigger data in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:PATTern {<String>}
:TRIGger:ENHanced:CANBus:DATA:PATTern?
```
<String> = Up to 64 characters by combining '0,' '1,' and 'X' (in one-byte unit)

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:PATTERN "11011111"
:TRIGGER:ENHANCED:CANBUS:DATA:PATTERN? -> :TRIGGER:ENHANCED:CANBUS:DATA:PATTERN "11011111"
```

:TRIGger:ENHanced:CANBus:DATA:SIGN

Function Sets the sign of the CAN bus signal trigger data or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:DATA:SIGN {SIGN|UNSign}
:TRIGger:ENHanced:CANBus:DATA:SIGN?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:DATA:SIGN SIGN
:TRIGGER:ENHANCED:CANBUS:DATA:SIGN? -> :TRIGGER:ENHANCED:CANBUS:DATA:SIGN SIGN
```

:TRIGger:ENHanced:CANBus:IDEXt?

Function Queries all settings related to the ID of the extended format of the CAN bus signal trigger.

Syntax
```
:TRIGger:ENHanced:CANBus:IDEXt?
```

:TRIGger:ENHanced:CANBus:IDEXt:HEXA

Function Sets the ID of the extended format of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:CANBus:IDEXt:HEXA {<String>}
```
<String> = 8 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:ENHANCED:CANBUS:IDEXT:HEXA "1AEF5906"
```

:TRIGger:ENHanced:CANBus:IDEXt:PATTern

Function Sets the ID of the extended format of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDEXt:PATTern {<String>}
:TRIGger:ENHanced:CANBus:IDEXt:PATTern?
```
<String> = 29 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:CANBUS:IDEXT:PATTERN "11001011011100001110111011111"
:TRIGGER:ENHANCED:CANBUS:IDEXT:PATTERN? -> :TRIGGER:ENHANCED:CANBUS:IDEXT:PATTERN "11001011011100001110111011111"
```

:TRIGger:ENHanced:CANBus:IDOR?

Function Queries all settings related to the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR?
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>?

Function Queries all settings related to each ID of the OR condition of the CAN bus signal trigger.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>?`
<x> = 1 to 4

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:ACK

Function Sets each ACK condition of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
ACK {ACK|ACKBoth|DONTcare|NONack}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
ACK?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
ACK ACK
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
ACK? -> :TRIGGER:ENHANCED:CANBUS:
IDOR:ID1:ACK ACK
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA?

Function Queries all settings related to each data of the OR condition of the CAN bus signal trigger.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA?
```
<x> = 1 to 4

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:BORDer

Function Sets byte order of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA:BORDer {BIG|LITTle}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA:BORDer?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
DATA:BORDER BIG
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
DATA:BORDER? -> :TRIGGER:ENHANCED:
CANBUS:IDOR:ID1:DATA:BORDER BIG
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:CONDition

Function Sets each data condition of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA:CONDition {BETWeen|DONTcare|
FALSe|GTHan|LTHan|ORANge|TRUE}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA:CONDition?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
DATA:CONDITION BETWEEN
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
DATA:CONDITION? -> :TRIGGER:ENHANCED:
CANBUS:IDOR:ID1:DATA:
CONDITION BETWEEN
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:DATA<x>

Function Sets comparison data of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA:DATA<x> {<NRf>}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
DATA:DATA<x>?
```
<x> of ID<x> = 1 to 4
<x> of DATA<x> = 1 or 2
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
DATA:DATA1 1
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
DATA:DATA1? -> :TRIGGER:ENHANCED:
CANBUS:IDOR:ID1:DATA:
DATA1 1.0000000E+00
```

Description
- Use `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:DATA1` when `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:CONDition GTHan` is specified.
- Use `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:DATA2` when `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:CONDition LTHan` is specified.
- Use `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:DATA1` to set the smaller value and `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:DATA2` to set the larger value when `:TRIGger:ENHANCED:CANBus:IDOR:ID<x>:DATA:CONDition BETWeen|ORANge` is specified.

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:DLC

Function Sets the number of valid bytes (DLC) of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:DLC {<NRf>}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:DLC?
```
<x> = 1 to 4
<NRf> = 0 to 8

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:DLC 0
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:DLC? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:DLC 0
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:HEXA

Function Sets each data of the OR condition of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:HEXA {<String>}
```
<x> = 1 to 4
<String> = Up to 16 characters by combining '0' to 'F' and 'X' (in one-byte unit)

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:HEXA "A9"
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:MSBLsb

Function Sets the MSB and LSB bits of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:MSBLsb {<NRf>,<NRf>}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:MSBLsb?
```
<x> = 1 to 4
<NRf> = See the User's Manual (IM DLM6054-01EN).

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:MSBLSB 1,0
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:MSBLSB? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:MSBLSB 1,0
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:PATTern

Function Sets each data of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:PATTern {<String>}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:PATTern?
```
<x> = 1 to 4
<String> = Up to 64 characters by combining '0,' '1,' and 'X' (in one-byte unit)

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:PATTERN "11011111"
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:PATTERN? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:PATTERN "11011111"
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:SIGN

Function Sets sign of each data of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:SIGN {SIGN|UNSign}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:DATA:SIGN?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:SIGN SIGN
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:SIGN? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:DATA:SIGN SIGN
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:FORMat

Function Sets each message format (standard or extended) of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:FORMat {STD|EXT}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:FORMat?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:FORMAT STD
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:FORMAT? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:FORMAT STD
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt?

Function Queries all settings related to the ID of each extended format of the OR condition of the CAN bus signal trigger.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt?`
<x> = 1 to 4

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:HEXA

Function Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in hexadecimal notation.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:HEXA {<String>}`
<x> = 1 to 4
<String> = 8 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDEXT:HEXA "1AEF5906"
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:PATTern

Function Sets the ID of each extended format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:PATTern {<String>}`
`:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDEXt:PATTern?`
<x> = 1 to 4
<String> = 29 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDEXT:PATTERN "11001011011100001110111011111"
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDEXT:PATTERN? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDEXT:PATTERN "11001011011100001110111011111"
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd?

Function Queries all settings related to the ID of each standard format of the OR condition of the CAN bus signal trigger.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd?`
<x> = 1 to 4

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:HEXA

Function Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in hexadecimal notation.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:HEXA {<String>}`
<x> = 1 to 4
<String> = 3 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDSTD:HEXA "5DF"
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:PATTern

Function Sets the ID of each standard format of the OR condition of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax `:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:PATTern {<String>}`
`:TRIGger:ENHanced:CANBus:IDOR:ID<x>:IDSTd:PATTern?`
<x> = 1 to 4
<String> = 11 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDSTD:PATTERN "10111011111"
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDSTD:PATTERN? -> :TRIGGER:ENHANCED:CANBUS:IDOR:ID1:IDSTD:PATTERN "10111011111"
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:MODE

Function Enables or disables each condition of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
MODE {<Boolean>}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
MODE?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
MODE ON
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
MODE? -> :TRIGGER:ENHANCED:CANBUS:
IDOR:ID1:MODE 1
```

:TRIGger:ENHanced:CANBus:IDOR:ID<x>:RTR

Function Sets each RTR of the OR condition of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
RTR {DATA|DONTcare|REMote}
:TRIGger:ENHanced:CANBus:IDOR:ID<x>:
RTR?
```
<x> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
RTR DATA
:TRIGGER:ENHANCED:CANBUS:IDOR:ID1:
RTR? -> :TRIGGER:ENHANCED:CANBUS:
IDOR:ID1:RTR DATA
```

:TRIGger:ENHanced:CANBus:IDSTd?

Function Queries all settings related to the ID of the standard format of the CAN bus signal trigger.

Syntax
```
:TRIGger:ENHanced:CANBus:IDSTd?
```

:TRIGger:ENHanced:CANBus:IDSTd:HEXA

Function Sets the ID of the standard format of the CAN bus signal trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:CANBus:IDSTd:HEXA
{<String>}
```
<String> = 3 characters by combining ‘0’ to ‘F’ and ‘X’

Example
```
:TRIGGER:ENHANCED:CANBUS:IDSTD:
HEXA "5DF"
```

:TRIGger:ENHanced:CANBus:IDSTd:PATTern

Function Sets the ID of the standard format of the CAN bus signal trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:IDSTd:
PATTern {<String>}
:TRIGger:ENHanced:CANBus:IDSTd:
PATTern?
```
<String> = 11 characters by combining ‘0,’ ‘1,’ and ‘X’

Example
```
:TRIGGER:ENHANCED:CANBUS:IDSTD:
PATTERN "10111011111"
:TRIGGER:ENHANCED:CANBUS:IDSTD:
PATTERN? -> :TRIGGER:ENHANCED:CANBUS:
IDSTD:PATTERN "10111011111"
```

:TRIGger:ENHanced:CANBus:MODE

Function Sets the CAN bus signal trigger mode or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:MODE {EFRame|I
DEXt|IDOR|IDSTd|SOF}
:TRIGger:ENHanced:CANBus:MODE?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:MODE EFRAME
:TRIGGER:ENHANCED:CANBUS:MODE?
-> :TRIGGER:ENHANCED:CANBUS:MODE EFRAME
```

:TRIGger:ENHanced:CANBus:RECessive

Function Sets the recessive level (bus level) of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:
RECessive {HIGH|LOW}
:TRIGger:ENHanced:CANBus:RECessive?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:
RECESSIVE HIGH
:TRIGGER:ENHANCED:CANBUS:RECESSIVE?
-> :TRIGGER:ENHANCED:CANBUS:
RECESSIVE HIGH
```

:TRIGger:ENHanced:CANBus:RTR

Function Sets the RTR of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:
RTR {DATA|DONTcare|REMote}
:TRIGger:ENHanced:CANBus:RTR?
```

Example
```
:TRIGGER:ENHANCED:CANBUS:RTR DATA
:TRIGGER:ENHANCED:CANBUS:RTR?
-> :TRIGGER:ENHANCED:CANBUS:RTR DATA
```

:TRIGger:ENHanced:CANBus:SOURce

Function Sets the trigger source of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:SOURce {<NRf>}
:TRIGger:ENHanced:CANBus:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:CANBUS:SOURCE 1
:TRIGGER:ENHANCED:CANBUS:SOURCE?
-> :TRIGGER:ENHANCED:CANBUS:SOURCE 1
```

:TRIGger:ENHanced:CANBus:SPOint

Function Sets the sample point of the CAN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:CANBus:SPOint {<NRf>}
:TRIGger:ENHanced:CANBus:SPOint?
```
<NRf> = 18.8 to 90.6(%)

Example
```
:TRIGGER:ENHANCED:CANBUS:SPOINT 18.8
:TRIGGER:ENHANCED:CANBUS:
SPOINT? -> :TRIGGER:ENHANCED:CANBUS:
SPOINT 18.8E+00
```

:TRIGger:ENHanced:I2CBus?

Function Queries all settings related to the I^2C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus?
```

:TRIGger:ENHanced:I2CBus:ADATa?

Function Queries all settings related to the address of the I^2C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa?
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT10address?

Function Queries all settings related to the 10-bit address of the I^2C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT10address?
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT10address:HEXA

Function Sets the 10-bit address of the I^2C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT10address:HEXA {<String>}
```
<String> = 3 characters by combining ‘0’ to ‘F’ and ‘X’ (bit 8 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT10ADDRESS:HEXA "7AB"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT10address:PATTern

Function Sets the 10-bit address of the I^2C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT10address:PATTern {<String>}
:TRIGger:ENHanced:I2CBus:ADATa:
BIT10address:PATTern?
```
<String> = 11 characters by combining ‘0’, ‘1’, and ‘X’ (bit 8 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT10ADDRESS:PATTERN "10111011111"
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT10ADDRESS:PATTERN? -> :TRIGGER:
ENHANCED:I2CBUS:ADATA:
BIT10ADDRESS:PATTERN "10111011111"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7ADdress?

Function Queries all settings related to the 7-bit address of the I^2C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7ADdress?
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7ADdress:HEXA

Function Sets the 7-bit address of the I^2C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7ADdress:HEXA {<String>}
```
<String> = 2 characters by combining ‘0’ to ‘F’ and ‘X’ (bit 0 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7ADDRESS:HEXA "DE"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7ADdress:PATTern

Function Sets the 7-bit address of the I^2C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7ADdress:PATTern {<String>}
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7ADdress:PATTern?
```
<String> = 8 characters by combining ‘0’, ‘1’, and ‘X’ (bit 0 is the R/$\overline{W}$ bit)

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7ADDRESS:PATTERN "11011110"
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7ADDRESS:PATTERN? -> :TRIGGER:
ENHANCED:I2CBUS:ADATA:BIT7ADDRESS:
PATTERN "11011110"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub?

Function Queries all settings related to the 7-bit + Sub address of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub?
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + Sub address of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:ADDRess?
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA

Function Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:ADDRess:HEXA {String>}
```
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is the R/W_ bit)

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:HEXA "AB"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern

Function Sets the 7-bit address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:ADDRess:PATTern {<String>}
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:ADDRess:PATTern?
```
<String> = 8 characters by combining '0', '1', and 'X' (bit 0 is the R/W_ bit)

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:PATTERN "10101011"
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:PATTERN? -> :TRIGGER:
ENHANCED:I2CBUS:ADATA:BIT7APSUB:
ADDRESS:PATTERN "10101011"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:SADDress?

Function Queries all settings related to the Sub address of the 7-bit + Sub address of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:SADDress?
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:SADDress:HEXA

Function Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:SADDress:HEXA {<String>}
```
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:HEXA "EF"
```

:TRIGger:ENHanced:I2CBus:ADATa:BIT7APsub:SADDress:PATTern

Function Sets the Sub address of the 7-bit + Sub address of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:SADDress:PATTern {<String>}
:TRIGger:ENHanced:I2CBus:ADATa:
BIT7APsub:SADDress:PATTern?
```
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:PATTERN "10101011"
:TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:PATTERN?
-> :TRIGGER:ENHANCED:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:PATTERN "10101011"
```

:TRIGger:ENHanced:I2CBus:ADATa:TYPE

Function Sets the address type of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:ADATa:
TYPE {BIT10address|BIT7ADdress|
BIT7APsub}
:TRIGger:ENHanced:I2CBus:ADATa:TYPE?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:ADATA:
TYPE BIT10ADDRESS
:TRIGGER:ENHANCED:I2CBUS:ADATA:TYPE?
-> :TRIGGER:ENHANCED:I2CBUS:ADATA:
TYPE BIT10ADDRESS
```

:TRIGger:ENHanced:I2CBus:CLOCk?

Function Queries all settings related to the clock of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:CLOCk?
```

:TRIGger:ENHanced:I2CBus:CLOCk:SOURce

Function Sets the clock trace of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:CLOCk:
SOURce {<NRf>}
:TRIGger:ENHanced:I2CBus:CLOCk:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:I2CBUS:CLOCK:SOURCE 1
:TRIGGER:ENHANCED:I2CBUS:CLOCK:SOURCE?
-> :TRIGGER:ENHANCED:I2CBUS:CLOCK:
SOURCE 1
```

:TRIGger:ENHanced:I2CBus:DATA?

Function Queries all settings related to the data of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA?
```
<x> = 1 or 2

:TRIGger:ENHanced:I2CBus:DATA:BYTE

Function Sets the number of data bytes of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
BYTE {<NRf>}
:TRIGger:ENHanced:I2CBus:DATA:BYTE?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:BYTE 1
:TRIGGER:ENHANCED:I2CBUS:DATA:BYTE?
-> :TRIGGER:ENHANCED:I2CBUS:DATA:BYTE 1
```

:TRIGger:ENHanced:I2CBus:DATA:CONDition

Function Sets the determination method (match or not match) of the data of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
CONDition {FALSe|TRUE}
:TRIGger:ENHanced:I2CBus:DATA:
CONDition?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:
CONDITION TRUE
:TRIGGER:ENHANCED:I2CBUS:DATA:
CONDITION? -> :TRIGGER:ENHANCED:I2CBUS:
DATA:CONDITION TRUE
```

:TRIGger:ENHanced:I2CBus:DATA:DPOSition

Function Sets the position for comparing the data pattern of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
DPOSition {<NRf>}
:TRIGger:ENHanced:I2CBus:DATA:
DPOSition?
```
<NRf> = 0 to 9999

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:
DPOSITION 1
:TRIGGER:ENHANCED:I2CBUS:DATA:
DPOSITION? -> :TRIGGER:ENHANCED:I2CBUS:
DATA:DPOSITION 1
```

:TRIGger:ENHanced:I2CBus:DATA:HEXA<x>

Function Sets the data of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
HEXA<x> {<String>}
```
<x> = 1 to 4
<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:
HEXA1 "AB"
```

:TRIGger:ENHanced:I2CBus:DATA:MODE

Function Enables/Disables the data conditions of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
MODE {<Boolean>}
:TRIGger:ENHanced:I2CBus:DATA:MODE?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:MODE ON
:TRIGGER:ENHANCED:I2CBUS:DATA:MODE?
-> :TRIGGER:ENHANCED:I2CBUS:DATA:MODE 1
```

:TRIGger:ENHanced:I2CBus:DATA:PATTern<x>

Function Sets the data of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
PATTern<x> {<String>}
:TRIGger:ENHanced:I2CBus:DATA:
PATTern<x>?
```
<x> = 1 to 4
<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:
PATTERN1 "10101011"
:TRIGGER:ENHANCED:I2CBUS:DATA:PATTERN1?
-> :TRIGGER:ENHANCED:I2CBUS:DATA:
PATTERN1 "10101011"
```

:TRIGger:ENHanced:I2CBus:DATA:PMODe

Function Sets the pattern comparison start position mode of the data of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
PMODe {DONTcare|SELect}
:TRIGger:ENHanced:I2CBus:DATA:PMODe?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:
PMODE SELECT
:TRIGGER:ENHANCED:I2CBUS:DATA:PMODE?
-> :TRIGGER:ENHANCED:I2CBUS:DATA:
PMODE SELECT
```

:TRIGger:ENHanced:I2CBus:DATA:SOURce

Function Sets the data trace of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:DATA:
SOURce {<NRf>}
:TRIGger:ENHanced:I2CBus:DATA:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:I2CBUS:DATA:SOURCE 1
:TRIGGER:ENHANCED:I2CBUS:DATA:SOURCE?
-> :TRIGGER:ENHANCED:I2CBUS:DATA:
SOURCE 1
```

:TRIGger:ENHanced:I2CBus:GCALl?

Function Queries all settings related to the general call of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:GCALl?
```
<x> = 1 or 2

:TRIGger:ENHanced:I2CBus:GCALl:BIT7maddress?

Function Queries all settings related to the 7-bit master address of the general call of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:GCALl:
BIT7maddress?
```
<x> = 1 or 2

:TRIGger:ENHanced:I2CBus:GCALl:BIT7maddress:HEXA

Function Sets the 7-bit master address of the general call of the I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:I2CBus:GCALl:
BIT7maddress:HEXA {<String>}
```
<x> = 1 or 2
<String> = 2 characters by combining '0' to 'F' and 'X' (bit 0 is fixed 1)

Example
```
:TRIGGER:ENHANCED:I2CBUS:GCALL:
BIT7MADDRESS:HEXA "AB"
```

:TRIGger:ENHanced:I2CBus:GCALl:BIT7maddress:PATTern

Function Sets the 7-bit master address of the general call of the I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:GCALl:
BIT7maddress:PATTern {<String>}
:TRIGger:ENHanced:I2CBus:GCALl:
BIT7maddress:PATTern?
```
<x> = 1 or 2
<String> = 7 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:I2CBUS:GCALL:
BIT7MADDRESS:PATTERN "1010101"
:TRIGGER:ENHANCED:I2CBUS:GCALL:
BIT7MADDRESS:PATTERN? -> :TRIGGER:
ENHANCED:I2CBUS:GCALL:BIT7MADDRESS:
PATTERN "1010101"
```

:TRIGger:ENHanced:I2CBus:GCALl:SBYTe (Second Byte)

Function Sets the second byte type of the general call of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:GCALl:
SBYTe {BIT7maddress|DONTcare|H04|H06}
:TRIGger:ENHanced:I2CBus:GCALl:SBYTe?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:GCALL:
SBYTE BIT7MADDRESS
:TRIGGER:ENHANCED:I2CBUS:GCALL:SBYTE?
-> :TRIGGER:ENHANCED:I2CBUS:GCALL:
SBYTE BIT7MADDRESS
```

:TRIGger:ENHanced:I2CBus:MODE

Function Sets the trigger mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:MODE {ADATa|
ESTart|GCALl|NAIGnore|SBHSmode}
:TRIGger:ENHanced:I2CBus:MODE?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:MODE ADATA
:TRIGGER:ENHANCED:I2CBUS:MODE?
-> :TRIGGER:ENHANCED:I2CBUS:MODE ADATA
```

:TRIGger:ENHanced:I2CBus:NAIGnore?

Function Queries all settings related to the NON ACK ignore mode of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:NAIGnore?
```

:TRIGger:ENHanced:I2CBus:NAIGnore:HSMode

Function Sets whether to ignore NON ACK in high speed mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:NAIGnore:HSMode {<Boolean>}
:TRIGger:ENHanced:I2CBus:NAIGnore:HSMode?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:NAIGNORE:HSMODE ON
:TRIGGER:ENHANCED:I2CBUS:NAIGNORE:HSMODE? -> :TRIGGER:ENHANCED:I2CBUS:NAIGNORE:HSMODE 1
```

:TRIGger:ENHanced:I2CBus:NAIGnore:RACCess

Function Sets whether to ignore NON ACK in read access mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:NAIGnore:RACCess {<Boolean>}
:TRIGger:ENHanced:I2CBus:NAIGnore:RACCess?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:NAIGNORE:RACCESS ON
:TRIGGER:ENHANCED:I2CBUS:NAIGNORE:RACCESS? -> :TRIGGER:ENHANCED:I2CBUS:NAIGNORE:RACCESS 1
```

:TRIGger:ENHanced:I2CBus:NAIGnore:SBYTe (Start Byte)

Function Sets whether to ignore NON ACK in the start byte of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:NAIGnore:SBYTe {<Boolean>}
:TRIGger:ENHanced:I2CBus:NAIGnore:SBYTe?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:NAIGNORE:SBYTE ON
:TRIGGER:ENHANCED:I2CBUS:NAIGNORE:SBYTE? -> :TRIGGER:ENHANCED:I2CBUS:NAIGNORE:SBYTE 1
```

:TRIGger:ENHanced:I2CBus:SBHSmode?

Function Queries all settings related to the start byte and high speed mode of the I²C bus trigger.

Syntax
```
:TRIGger:ENHanced:I2CBus:SBHSmode?
```

:TRIGger:ENHanced:I2CBus:SBHSmode:TYPE

Function Sets the type of the start byte or high speed mode of the I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:I2CBus:SBHSmode:TYPE {HSMode|SBYTe}
:TRIGger:ENHanced:I2CBus:SBHSmode:TYPE?
```

Example
```
:TRIGGER:ENHANCED:I2CBUS:SBHSMODE:TYPE HSMODE
:TRIGGER:ENHANCED:I2CBUS:SBHSMODE:TYPE?
-> :TRIGGER:ENHANCED:I2CBUS:SBHSMODE:TYPE HSMODE
```

:TRIGger:ENHanced:LINBus?

Function Queries all settings related to the LIN bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:LINBus?
```

:TRIGger:ENHanced:LINBus:BRATe

Function Sets the LIN bus signal trigger bitrate (data transfer rate) or queries the current setting.

Syntax
```
:TRIGger:ENHanced:LINBus:BRATe {<NRf>|USER,<NRf>}
:TRIGger:ENHanced:LINBus:BRATe?
```
<NRf> = 1200, 2400, 4800, 9600, 19200
<NRf> for USER = See the main unit User's Manual.

Example
```
:TRIGGER:ENHANCED:LINBUS:BRATE 19200
:TRIGGER:ENHANCED:LINBUS:BRATE?
-> :TRIGGER:ENHANCED:LINBUS:BRATE 19200
```

:TRIGger:ENHanced:LINBus:SOURce

Function Sets the LIN bus signal trigger source or queries the current setting.

Syntax
```
:TRIGger:ENHanced:LINBus:SOURce {<NRf>}
:TRIGger:ENHanced:LINBus:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:LINBUS:SOURCE 1
:TRIGGER:ENHANCED:LINBUS:SOURCE?
-> :TRIGGER:ENHANCED:LINBUS:SOURCE 1
```

:TRIGger:ENHanced:SPIBus?

Function Queries all settings related to the SPI bus trigger.

Syntax
```
:TRIGger:ENHanced:SPIBus?
```

:TRIGger:ENHanced:SPIBus:BITorder

Function Sets the bit order of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:
BITorder {LSBFirst|MSBFirst}
:TRIGger:ENHanced:SPIBus:BITorder?
```

Example
```
:TRIGGER:ENHANCED:SPIBUS:
BITORDER LSBFIRST
:TRIGGER:ENHANCED:SPIBUS:BITORDER?
-> :TRIGGER:ENHANCED:SPIBUS:
BITORDER LSBFIRST
```

:TRIGger:ENHanced:SPIBus:CLOCk?

Function Queries all settings related to the clock of the SPI bus trigger.

Syntax
```
:TRIGger:ENHanced:SPIBus:CLOCk?
```

:TRIGger:ENHanced:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock trace of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:CLOCk:
POLarity {FALL|RISE}
:TRIGger:ENHanced:SPIBus:CLOCk:
POLarity?
```

Example
```
:TRIGGER:ENHANCED:SPIBUS:CLOCK:
POLARITY FALL
:TRIGGER:ENHANCED:SPIBUS:CLOCK:
POLARITY? -> :TRIGGER:ENHANCED:SPIBUS:
CLOCK:POLARITY FALL
```

:TRIGger:ENHanced:SPIBus:CLOCk:SOURce

Function Sets the clock trace of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:CLOCk:
SOURce {<NRf>}
:TRIGger:ENHanced:SPIBus:CLOCk:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:SPIBUS:CLOCK:SOURCE 1
:TRIGGER:ENHANCED:SPIBUS:CLOCK:SOURCE?
-> :TRIGGER:ENHANCED:SPIBUS:CLOCK:
SOURCE 1
```

:TRIGger:ENHanced:SPIBus:CS?

Function Queries all settings related to the chip select of the SPI bus trigger.

Syntax
```
:TRIGger:ENHanced:SPIBus:CS?
```

:TRIGger:ENHanced:SPIBus:CS:ACTive

Function Sets the active level of the chip select of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:CS:
ACTive {HIGH|LOW}
:TRIGger:ENHanced:SPIBus:CS:ACTive?
```

Example
```
:TRIGGER:ENHANCED:SPIBUS:CS:ACTIVE HIGH
:TRIGGER:ENHANCED:SPIBUS:CS:ACTIVE?
-> :TRIGGER:ENHANCED:SPIBUS:CS:
ACTIVE HIGH
```

:TRIGger:ENHanced:SPIBus:CS:SOURce

Function Sets the chip select trace of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:CS:
SOURce {<NRf>}
:TRIGger:ENHanced:SPIBus:CS:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:SPIBUS:CS:SOURCE 1
:TRIGGER:ENHANCED:SPIBUS:CS:SOURCE?
-> :TRIGGER:ENHANCED:SPIBUS:CS:SOURCE 1
```

:TRIGger:ENHanced:SPIBus:DATA<x>?

Function Queries all settings related to the data of the SPI bus trigger.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>?
```
<x> = 1 or 2

Description DATA2 is valid when `:TRIGger:ENHanced:SPIBus:MODE WIRE4` is specified.

:TRIGger:ENHanced:SPIBus:DATA<x>:BYTE

Function Sets the number of bytes of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>:
BYTE {<NRf>}
:TRIGger:ENHanced:SPIBus:DATA<x>:BYTE?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:SPIBUS:DATA1:BYTE 1
:TRIGGER:ENHANCED:SPIBUS:DATA1:BYTE?
-> :TRIGGER:ENHANCED:SPIBUS:DATA1:
BYTE 1
```

:TRIGger:ENHanced:SPIBus:DATA<x>:CONDition

Function Sets the determination method (match or not match) of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>:CONDition {FALSe|TRUE}
:TRIGger:ENHanced:SPIBus:DATA<x>:CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:ENHANCED:SPIBUS:DATA1:CONDITION TRUE
:TRIGGER:ENHANCED:SPIBUS:DATA1:CONDITION? -> :TRIGGER:ENHANCED:SPIBUS:DATA1:CONDITION TRUE
```

:TRIGger:ENHanced:SPIBus:DATA<x>:DPOSition

Function Sets the pattern comparison start position of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>:DPOSition {<NRf>}
:TRIGger:ENHanced:SPIBus:DATA<x>:DPOSition?
```
<x> = 1 or 2

<NRf> = 0 to 9999

Example
```
:TRIGGER:ENHANCED:SPIBUS:DATA1:DPOSITION 1
:TRIGGER:ENHANCED:SPIBUS:DATA1:DPOSITION? -> :TRIGGER:ENHANCED:SPIBUS:DATA1:DPOSITION 1
```

:TRIGger:ENHanced:SPIBus:DATA<x>:HEXA<x>

Function Sets the data of the SPI bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>:HEXA<x> {<String>}
```
<x> of DATA<x> = 1 or 2

<x> of HEXA<x> = 1 to 4

<String> = 2 characters by combining '0' to 'F' and 'X'

Example
```
:TRIGGER:ENHANCED:SPIBUS:DATA1:HEXA1 "AB"
```

:TRIGger:ENHanced:SPIBus:DATA<x>:PATTern<x>

Function Sets the data of the SPI bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>:PATTern<x> {<String>}
:TRIGger:ENHanced:SPIBus:DATA<x>:PATTern<x>?
```
<x> of DATA<x> = 1 or 2

<x> of PATTern<x> = 1 to 4

<String> = 8 characters by combining '0,' '1,' and 'X'

Example
```
:TRIGGER:ENHANCED:SPIBUS:DATA1:PATTERN1 "10101011"
:TRIGGER:ENHANCED:SPIBUS:DATA1:PATTERN1? -> :TRIGGER:ENHANCED:SPIBUS:DATA1:PATTERN1 "10101011"
```

:TRIGger:ENHanced:SPIBus:DATA<x>:SOURce

Function Sets the trace of the data of the SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:DATA<x>:SOURce {<NRf>}
:TRIGger:ENHanced:SPIBus:DATA<x>:SOURce?
```
<x> = 1 or 2

<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:SPIBUS:DATA1:SOURCE 1
:TRIGGER:ENHANCED:SPIBUS:DATA1:SOURCE? -> :TRIGGER:ENHANCED:SPIBUS:DATA1:SOURCE 1
```

:TRIGger:ENHanced:SPIBus:MODE

Function Sets the wiring system of the SPI bus trigger (three-wire or four-wire) or queries the current setting.

Syntax
```
:TRIGger:ENHanced:SPIBus:MODE {WIRE3|WIRE4}
:TRIGger:ENHanced:SPIBus:MODE?
```

Example
```
:TRIGGER:ENHANCED:SPIBUS:MODE WIRE3
:TRIGGER:ENHANCED:SPIBUS:MODE? -> :TRIGGER:ENHANCED:SPIBUS:MODE WIRE3
```

:TRIGger:ENHanced:UART?

Function Queries all settings related to the UART signal trigger.

Syntax
```
:TRIGger:ENHanced:UART?
```

:TRIGger:ENHanced:UART:BRATe

Function Sets the UART signal trigger bit rate (data transfer rate) or queries the current setting.

Syntax
```
:TRIGger:ENHanced:UART:
BRATe {<NRf>|USER, <NRf>}
:TRIGger:ENHanced:UART:BRATe?
```
<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
<NRf> of USER = See the DL6000/DLM6000 User's Manual

Example
```
:TRIGGER:ENHANCED:UART:BRATE 19200
:TRIGGER:ENHANCED:UART:BRATE? ->
:TRIGGER:ENHANCED:UART:BRATE 19200
```

:TRIGger:ENHanced:UART:FORMat

Function Sets the UART signal trigger format or queries the current setting.

Syntax
```
:TRIGger:ENHanced:UART:
FORMat {BIT7parity|BIT8Noparity|
BIT8Parity}
:TRIGger:ENHanced:UART:FORMat?
```

Example
```
:TRIGGER:ENHANCED:UART:
FORMAT BIT7PARITY
:TRIGGER:ENHANCED:UART:FORMAT? ->
:TRIGGER:ENHANCED:UART:
FORMAT BIT7PARITY
```

:TRIGger:ENHanced:UART:POLarity

Function Sets the UART signal trigger polarity or queries the current setting.

Syntax
```
:TRIGger:ENHanced:UART:
POLarity {NEGative|POSitive}
:TRIGger:ENHanced:UART:POLarity?
```

Example
```
:TRIGGER:ENHANCED:UART:
POLARITY NEGATIVE
:TRIGGER:ENHANCED:UART:POLARITY? ->
:TRIGGER:ENHANCED:UART:
POLARITY NEGATIVE
```

:TRIGger:ENHanced:UART:SOURce

Function Sets the UART signal trigger source or queries the current setting.

Syntax
```
:TRIGger:ENHanced:UART:SOURce {<NRf>}
:TRIGger:ENHanced:UART:SOURce?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:ENHANCED:UART:SOURCE 1
:TRIGGER:ENHANCED:UART:SOURCE? ->
:TRIGGER:ENHANCED:UART:SOURCE 1
```

:TRIGger:ENHanced:UART:SPOint

Function Sets the UART signal trigger sample point or queries the current setting.

Syntax
```
:TRIGger:ENHanced:UART:SPOint {<NRf>}
:TRIGger:ENHanced:UART:SPOint?
```
<NRf> = 18.8 to 90.6(%)

Example
```
:TRIGGER:ENHANCED:UART:SPOINT 18.8
:TRIGGER:ENHANCED:UART:SPOINT? ->
:TRIGGER:ENHANCED:UART:SPOINT 18.8E+00
```

:TRIGger:LOGic:I2CBus?

Function Queries all settings related to the logic I^2C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus?
```

:TRIGger:LOGic:I2CBus:ADATa?

Function Queries all settings related to the address of the logic I^2C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa?
```

:TRIGger:LOGic:I2CBus:ADATa:BIT10address?

Function Queries all settings related to the 10-bit address of the logic I^2C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT10address?
```

:TRIGger:LOGic:I2CBus:ADATa:BIT10address:HEXA

Function Sets the 10-bit address of the logic I^2C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT10address:HEXA {<string>}
```
<string> = combination of 3 characters (0-F, and X), where bit 8 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT10ADDRESS:HEXA "7AB"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT10address:PATTern

Function Sets the 10-bit address of the logic I^2C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT10address:PATTern {<string>}
:TRIGger:LOGic:I2CBus:ADATa:
BIT10address:PATTern?
```
<string> = combination of 11 characters (0, 1, and X), where bit 8 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT10ADDRESS:PATTERN "10111011111"
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT10ADDRESS:PATTERN?
->:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT10ADDRESS:PATTERN "10111011111"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7ADdress?

Function Queries all settings related to the 7-bit address of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7ADdress?
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7ADdress:HEXA

Function Sets the 7-bit address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7ADdress:HEXA {<string>}
```
<string> = combination of 2 characters (0-F, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7ADDRESS:HEXA "DE"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7ADdress:PATTern

Function Sets the 7-bit address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7ADdress:PATTern {<string>}
:TRIGger:LOGic:I2CBus:ADATa:
BIT7ADdress:PATTern?
```
<string> = combination of 8 characters (0, 1, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7ADDRESS:PATTERN "11011110"
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7ADDRESS:PATTERN?
-> :TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7ADDRESS:PATTERN "11011110"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub?

Function Queries all settings related to the 7-bit + Sub address of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub?
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess?

Function Queries all settings related to the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:ADDRess?
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:HEXA

Function Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:ADDRess:HEXA {<string>}
```
<string> = combination of 2 characters (0-F, and X), where bit 0 is R/$\overline{W}$ bit.

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:HEXA "AB"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:ADDRess:PATTern

Function Sets the 7-bit address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:ADDRess:PATTern {<string>}
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:ADDRess:PATTern?
```
<string> = combination of 8 characters (0, 1, and X), where bit 0 is R/W_ bit.

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:PATTERN "10101011"
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:PATTERN?
-> :TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:ADDRESS:PATTERN "10101011"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:SADDress?

Function Queries all settings related to the sub address of the 7-bit + Sub address of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:SADDress?
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:HEXA

Function Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:SADDress:HEXA {<string>}
```
<string> = Combination of up to 2 characters (0-F and X)

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:HEXA "EF"
```

:TRIGger:LOGic:I2CBus:ADATa:BIT7APsub:SADDress:PATTern

Function Sets the sub address of the 7-bit + Sub address of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:SADDress:PATTern {<string>}
:TRIGger:LOGic:I2CBus:ADATa:
BIT7APsub:SADDress:PATTern?
```
<string> = combination of 8 characters (0, 1, and X).

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:PATTERN "10101011"
:TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:PATTERN?
-> :TRIGGER:LOGIC:I2CBUS:ADATA:
BIT7APSUB:SADDRESS:PATTERN "10101011"
```

:TRIGger:LOGic:I2CBus:ADATa:TYPE

Function Sets the address type of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:ADATa:
TYPE {BIT10address|BIT7ADdress|
BIT7APsub}
:TRIGger:LOGic:I2CBus:ADATa:TYPE?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:ADATA:
TYPE BIT10ADDRESS
:TRIGGER:LOGIC:I2CBUS:ADATA:TYPE?
-> :TRIGGER:LOGIC:I2CBUS:ADATA:
TYPE BIT10ADDRESS
```

:TRIGger:LOGic:I2CBus:CLOCk?

Function Queries all settings related to the clock of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:CLOCk?
```

:TRIGger:LOGic:I2CBus:CLOCk:SOURce

Function Sets the clock trace for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:CLOCk:
SOURce {A<x>|B<x>|C<x>|D<x>}
:TRIGger:LOGic:I2CBus:CLOCk:SOURce?
```
<x>= 0 to 7

Example
```
:TRIGGER:LOGIC:I2CBUS:CLOCK:SOURCE A0
:TRIGGER:LOGIC:I2CBUS:CLOCK:SOURCE?
-> :TRIGGER:LOGIC:I2CBUS:CLOCK:
SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:I2CBus:DATA?

Function Queries all settings related to the data of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA?
```

:TRIGger:LOGic:I2CBus:DATA:BYTE

Function Sets the number of settings for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:BYTE {<NRf>}
:TRIGger:LOGic:I2CBus:DATA:BYTE?
```
<NRf> = 1 to 4

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:BYTE 1
:TRIGGER:LOGIC:I2CBUS:DATA:BYTE?
-> :TRIGGER:LOGIC:I2CBUS:DATA:BYTE 1
```

:TRIGger:LOGic:I2CBus:DATA:CONDition

Function Sets the determination method for the data of the logic I²C bus trigger (match / no match) or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:
CONDition {FALSe|TRUE}
:TRIGger:LOGic:I2CBus:DATA:CONDition?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:
CONDITION FALSE
:TRIGGER:LOGIC:I2CBUS:DATA:CONDITION?
-> :TRIGGER:LOGIC:I2CBUS:DATA:
CONDITION FALSE
```

:TRIGger:LOGic:I2CBus:DATA:DPOSition

Function Sets the pattern comparison position for the data of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:
DPOSition {<NRf>}
:TRIGger:LOGic:I2CBus:DATA:DPOSition?
```
<NRf> = 0 to 9999

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:DPOSITION 1
:TRIGGER:LOGIC:I2CBUS:DATA:DPOSITION?
-> :TRIGGER:LOGIC:I2CBUS:DATA:
DPOSITION 1
```

:TRIGger:LOGic:I2CBus:DATA:HEXA<x>

Function Sets the data of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:
HEXA<x> {<string>}
<x> of HEXA<x> = 1 to 4
```
<string> = Combination of up to 2 characters (0-F and X)

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:HEXA1 "AB"
```

:TRIGger:LOGic:I2CBus:DATA:MODE

Function Enables/disables the data conditions of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:
MODE {<Boolean>}
:TRIGger:LOGic:I2CBus:DATA:MODE?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:MODE ON
:TRIGGER:LOGIC:I2CBUS:DATA:MODE?
-> :TRIGGER:LOGIC:I2CBUS:DATA:MODE 1
```

:TRIGger:LOGic:I2CBus:DATA:PATTern<x>

Function Sets the data for the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:
PATTern<x> {<string>}
:TRIGger:LOGic:I2CBus:DATA:PATTern<x>?
```
<x> = 1 to 4
<string> = combination of 8 characters (0, 1, and X).

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:
PATTERN1 "10101011"
:TRIGGER:LOGIC:I2CBUS:DATA:PATTERN1?
-> :TRIGGER:LOGIC:I2CBUS:DATA:
PATTERN1 "10101011"
```

:TRIGger:LOGic:I2CBus:DATA:PMODe

Function Sets the pattern comparison start position for the data of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:PMODe
{DONTcare|SELect}
:TRIGger:LOGic:I2CBus:DATA:PMODe?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:
PMODE DONTCARE
:TRIGGER:LOGIC:I2CBUS:DATA:PMODE?
-> :TRIGGER:LOGIC:I2CBUS:DATA:
PMODE DONTCARE
```

:TRIGger:LOGic:I2CBus:DATA:SOURce

Function Sets the data trace for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:DATA:
SOURce {A<x>|B<x>|C<x>|D<x>}
:TRIGger:LOGic:I2CBus:DATA:SOURce?
```
<x> = 0 to 7

Example
```
:TRIGGER:LOGIC:I2CBUS:DATA:SOURCE A0
:TRIGGER:LOGIC:I2CBUS:DATA:SOURCE?
-> :TRIGGER:LOGIC:I2CBUS:DATA:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:I2CBus:GCALl?

Function Queries all settings related to the general call of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:GCALl?
```

:TRIGger:LOGic:I2CBus:GCALl:BIT7maddress?

Function Queries all settings related to the 7-bit master address of the general call of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:GCALl:
BIT7maddress?
```

:TRIGger:LOGic:I2CBus:GCALl:BIT7maddress:HEXA

Function Sets the 7-bit master address of the general call of the logic I²C bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:I2CBus:GCALl:
BIT7maddress:HEXA {<string>}
```
<string> = combination of 2 characters (0-F and X), where bit 0 is fixed to '1.'

Example
```
:TRIGGER:LOGIC:I2CBUS:GCALL:
BIT7MADDRESS:HEXA "AB"
```

:TRIGger:LOGic:I2CBus:GCALl:BIT7maddress:PATTern

Function Sets the 7-bit master address of the general call of the logic I²C bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:GCALl:
BIT7maddress:PATTern {<string>}
:TRIGger:LOGic:I2CBus:GCALl:
BIT7maddress:PATTern?
```
<string> = combination of 7 characters (0, 1, and X).

Example
```
:TRIGGER:LOGIC:I2CBUS:GCALL:
BIT7MADDRESS:PATTERN "1010101"
:TRIGGER:LOGIC:I2CBUS:GCALL:
BIT7MADDRESS:PATTERN?
-> :TRIGGER:LOGIC:I2CBUS:GCALL:
BIT7MADDRESS:PATTERN "1010101"
```

:TRIGger:LOGic:I2CBus:GCALl:SBYTe (Second Byte)

Function Sets the type of the second byte of the general call of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:GCALl:
SBYTe {BIT7maddress|DONTcare|H04|H06}
:TRIGger:LOGic:I2CBus:GCALl:SBYTe?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:GCALL:
SBYTE BIT7MADDRESS
:TRIGGER:LOGIC:I2CBUS:GCALL:SBYTE?
-> :TRIGGER:LOGIC:I2CBUS:GCALL:
SBYTE BIT7MADDRESS
```

:TRIGger:LOGic:I2CBus:MODE

Function Sets the trigger mode for the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:MODE {ADATa|ESTart|GCALl|NAIGnore|SBHSmode}
:TRIGger:LOGic:I2CBus:MODE?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:MODE ADATA
:TRIGGER:LOGIC:I2CBUS:MODE?
-> :TRIGGER:LOGIC:I2CBUS:MODE ADATA
```

:TRIGger:LOGic:I2CBus:NAIGnore?

Function Queries all settings related to the NON-ACK Ignore mode of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:NAIGnore?
```

:TRIGger:LOGic:I2CBus:NAIGnore:HSMode

Function Sets whether to ignore NON ACK in high speed mode of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:NAIGnore:HSMode {<Boolean>}
:TRIGger:LOGic:I2CBus:NAIGnore:HSMode?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:NAIGNORE:HSMODE ON
:TRIGGER:LOGIC:I2CBUS:NAIGNORE:HSMODE?
-> :TRIGGER:LOGIC:I2CBUS:NAIGNORE:HSMODE 1
```

:TRIGger:LOGic:I2CBus:NAIGnore:RACCess

Function Sets whether to ignore NON ACK in read access mode of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:NAIGnore:RACCess {<Boolean>}
:TRIGger:LOGic:I2CBus:NAIGnore:RACCess?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:NAIGNORE:RACCESS ON
:TRIGGER:LOGIC:I2CBUS:NAIGNORE:RACCESS?
-> :TRIGGER:LOGIC:I2CBUS:NAIGNORE:RACCESS 1
```

:TRIGger:LOGic:I2CBus:NAIGnore:SBYTe (Start Byte)

Function Sets whether to ignore NON ACK in the start byte of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:NAIGnore:SBYTe {<Boolean>}
:TRIGger:LOGic:I2CBus:NAIGnore:SBYTe?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:NAIGNORE:SBYTE ON
:TRIGGER:LOGIC:I2CBUS:NAIGNORE:SBYTE?
-> :TRIGGER:LOGIC:I2CBUS:NAIGNORE:SBYTE 1
```

:TRIGger:LOGic:I2CBus:SBHSmode?

Function Queries all settings related to the start byte/high speed mode of the logic I²C bus trigger.

Syntax
```
:TRIGger:LOGic:I2CBus:SBHSmode?
```

:TRIGger:LOGic:I2CBus:SBHSmode:TYPE

Function Sets the type of the start byte/high speed mode of the logic I²C bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:I2CBus:SBHSmode:TYPE {HSMode|SBYTe}
:TRIGger:LOGic:I2CBus:SBHSmode:TYPE?
```

Example
```
:TRIGGER:LOGIC:I2CBUS:SBHSMODE:TYPE HSMODE
:TRIGGER:LOGIC:I2CBUS:SBHSMODE:TYPE?
-> :TRIGGER:LOGIC:I2CBUS:SBHSMODE:TYPE HSMODE
```

:TRIGger:LOGic:LINBus?

Function Queries all settings related to the logic LIN bus signal triggers.

Syntax
```
:TRIGger:LOGic:LINBus?
```

:TRIGger:LOGic:LINBus:BRATe

Function Sets the bit rate (data transfer rate) of the logic LIN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:LINBus:BRATe {<NRf>|USER,<NRf>}
:TRIGger:LOGic:LINBus:BRATe?
```
<NRf>=1200,2400,4800,9600,19200
<NRf> for USER = See the main unit user's manual.

Example
```
:TRIGGER:LOGIC:LINBUS:BRATE 19200
:TRIGGER:LOGIC:LINBUS:BRATE?
-> :TRIGGER:LOGIC:LINBUS:BRATE 19200
```

:TRIGger:LOGic:LINBus:SOURce

Function Sets the trigger source of the logic LIN bus signal trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:LINBus:SOURce {A<x>|B<x>|C<x>|D<x>}
:TRIGger:LOGic:LINBus:SOURce?
```
<x>= 0 to 7

Example
```
:TRIGGER:LOGIC:LINBUS:SOURCE A0
:TRIGGER:LOGIC:LINBUS:SOURCE?
-> :TRIGGER:LOGIC:LINBUS:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:SPIBus?

Function Queries all settings related to the logic SPI bus trigger.

Syntax
```
:TRIGger:LOGic:SPIBus?
```

:TRIGger:LOGic:SPIBus:BITorder

Function Sets the bit order for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:BITorder {LSBFirst|MSBFirst}
:TRIGger:LOGic:SPIBus:BITorder?
```

Example
```
:TRIGGER:LOGIC:SPIBUS:BITORDER LSBFIRST
:TRIGGER:LOGIC:SPIBUS:BITORDER?
-> :TRIGGER:LOGIC:SPIBUS:BITORDER LSBFIRST
```

:TRIGger:LOGic:SPIBus:CLOCk?

Function Queries all settings related to the clock of the logic SPI bus trigger.

Syntax
```
:TRIGger:LOGic:SPIBus:CLOCk?
```

Example
```
:TRIGGER:LOGIC:SPIBUS:CLOCK?
-> :TRIGGER:LOGIC:SPIBUS:CLOCK:POLARITY FALL;SOURCE A0
```

:TRIGger:LOGic:SPIBus:CLOCk:POLarity

Function Sets the polarity of the clock trace for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:CLOCk:POLarity {FALL|RISE}
:TRIGger:LOGic:SPIBus:CLOCk:POLarity?
```

Example
```
:TRIGGER:LOGIC:SPIBUS:CLOCK:POLARITY FALL
:TRIGGER:LOGIC:SPIBUS:CLOCK:POLARITY?
-> :TRIGGER:LOGIC:SPIBUS:CLOCK:POLARITY FALL
```

:TRIGger:LOGic:SPIBus:CLOCk:SOURce

Function Sets the clock trace for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:CLOCk:SOURce {A<x>|B<x>|C<x>|D<x>}
:TRIGger:LOGic:SPIBus:CLOCk:SOURce?
```
<x>= 0 to 7

Example
```
:TRIGGER:LOGIC:SPIBUS:CLOCK:SOURCE A0
:TRIGGER:LOGIC:SPIBUS:CLOCK:SOURCE?
-> :TRIGGER:LOGIC:SPIBUS:CLOCK:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:SPIBus:CS?

Function Queries all settings related to the chip select of the logic SPI bus trigger.

Syntax
```
:TRIGger:LOGic:SPIBus:CS?
```

:TRIGger:LOGic:SPIBus:CS:ACTive

Function Sets the active level of the chip select for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:CS:ACTive {HIGH|LOW}
:TRIGger:LOGic:SPIBus:CS:ACTive?
```

Example
```
:TRIGGER:LOGIC:SPIBUS:CS:ACTIVE HIGH
:TRIGGER:LOGIC:SPIBUS:CS:ACTIVE?
-> :TRIGGER:LOGIC:SPIBUS:CS:ACTIVE HIGH
```

:TRIGger:LOGic:SPIBus:CS:SOURce

Function Sets the chip select trace for the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:CS:SOURce {A<x>|B<x>|C<x>|D<x>}
:TRIGger:LOGic:SPIBus:CS:SOURce?
```
<x> = 0 to 7

Example
```
:TRIGGER:LOGIC:SPIBUS:CS:SOURCE A0
:TRIGGER:LOGIC:SPIBUS:CS:SOURCE?
-> :TRIGGER:LOGIC:SPIBUS:CS:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:SPIBus:DATA<x>?

Function Queries all settings related to each data of the logic SPI bus trigger.

Syntax
```
:TRIGger:LOGic:SPIBus:DATA<x>?
```
<x> = 1 or 2

:TRIGger:LOGic:SPIBus:DATA<x>:BYTE

Function Sets the number of settings for each data of the logic SPI bus trigger or queries the current setting.

Syntax
```
: RIGger:LOGic:SPIBus:DATA<x>:
BYTE {<NRf>}
:TRIGger:LOGic:SPIBus:DATA<x>:BYTE?
```
<x> = 1 or 2
<NRf> = 1 to 4

Example
```
:TRIGGER:LOGIC:SPIBUS:DATA1:BYTE 1
:TRIGGER:LOGIC:SPIBUS:DATA1:BYTE?
-> :TRIGGER:LOGIC:SPIBUS:DATA1:BYTE 1
```

:TRIGger:LOGic:SPIBus:DATA<x>:CONDition

Function Sets the determination method for the data of the logic SPI bus trigger (match / no match) or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:DATA<x>:
CONDition {FALSe|TRUE}
:TRIGger:LOGic:SPIBus:DATA<x>:
CONDition?
```
<x> = 1 or 2

Example
```
:TRIGGER:LOGIC:SPIBUS:DATA1:
CONDITION FALSE
:TRIGGER:LOGIC:SPIBUS:DATA1:CONDITION?
-> :TRIGGER:LOGIC:SPIBUS:DATA1:
CONDITION FALSE
```

:TRIGger:LOGic:SPIBus:DATA<x>:DPOSition

Function Sets the pattern comparison start position for the data of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:DATA<x>:
DPOSition {<NRf>}
:TRIGger:LOGic:SPIBus:DATA<x>:
DPOSition?
```
<x> = 1 or 2
<NRf> = 0 to 9999

Example
```
:TRIGGER:LOGIC:SPIBUS:DATA1:DPOSITION 1
:TRIGGER:LOGIC:SPIBUS:DATA1:DPOSITION?
-> :TRIGGER:LOGIC:SPIBUS:DATA1:
DPOSITION 1
```

:TRIGger:LOGic:SPIBus:DATA<x>:HEXA<x>

Function Sets the data of the logic SPI bus trigger in hexadecimal notation.

Syntax
```
:TRIGger:LOGic:SPIBus:DATA<x>:
HEXA<x> {<string>}
```
<x> of DATA<x> = 1 or 2
<x> of HEXA<x> = 1 to 4
<string> = Combination of up to 2 characters (0-F and X)

Example
```
:TRIGGER:LOGIC:SPIBUS:DATA1:HEXA1 "AB"
```

:TRIGger:LOGic:SPIBus:DATA<x>:PATTern<x>

Function Sets each data of the logic SPI bus trigger in binary notation or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:DATA<x>:
PATTern<x> {<string>}
:TRIGger:LOGic:SPIBus:DATA<x>:
PATTern<x>?
```
<x> of DATA<x> = 1 or 2
<x> of PATTern<x> = 1 to 4
<string> = combination of 8 characters (0, 1, and X).

Example
```
:TRIGGER:LOGIC:SPIBUS:DATA1:
PATTERN1 "10101011"
:TRIGGER:LOGIC:SPIBUS:DATA1:PATTERN1?
-> :TRIGGER:LOGIC:SPIBUS:DATA1:
PATTERN1 "10101011"
```

:TRIGger:LOGic:SPIBus:DATA<x>:SOURce

Function Sets the trace of each data of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:DATA<x>:
SOURce {A<y>|B<y>|C<y>|D<y>}
:TRIGger:LOGic:SPIBus:DATA<x>:SOURce?
```
<x> = 1 or 2
<y> = 0 to 7

Example
```
:TRIGGER:LOGIC:SPIBUS:DATA1:SOURCE A0
:TRIGGER:LOGIC:SPIBUS:DATA1:SOURCE?
-> :TRIGGER:LOGIC:SPIBUS:DATA1:
SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:SPIBus:MODE

Function Sets the wiring method (3-wire/4-wire) of the logic SPI bus trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:SPIBus:MODE {WIRE3|
WIRE4}
:TRIGger:LOGic:SPIBus:MODE?
```

Example
```
:TRIGGER:LOGIC:SPIBUS:MODE WIRE3
:TRIGGER:LOGIC:SPIBUS:MODE?
-> :TRIGGER:LOGIC:SPIBUS:MODE WIRE3
```

:TRIGger:LOGic:UART?

Function Queries all settings related to the logic UART signal trigger.

Syntax
```
:TRIGger:LOGic:UART?
```

:TRIGger:LOGic:UART:BRATe

Function Sets the logic UART signal trigger bit rate (data transfer rate) or queries the current setting.

Syntax
```
:TRIGger:LOGic:UART:
BRATe {<NRf>|USER, <NRf>}
:TRIGger:LOGic:UART:BRATe?
```
<NRf> = 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
<NRf> of USER = See the main DL6000/DLM6000 User's Manual

Example
```
:TRIGGER:LOGIC:UART:BRATE 19200
:TRIGGER:LOGIC:UART:BRATE? -> :TRIGGER:
LOGIC:UART:BRATE 19200
```

:TRIGger:LOGic:UART:FORMat

Function Sets the logic UART signal trigger format or queries the current setting.

Syntax
```
:TRIGger:LOGic:UART:FORMat {BIT7parity|
BIT8Noparity|BIT8Parity}
:TRIGger:LOGic:UART:FORMat?
```

Example
```
:TRIGGER:LOGIC:UART:FORMAT BIT7PARITY
:TRIGGER:LOGIC:UART:FORMAT? ->
:TRIGGER:LOGIC:UART:FORMAT BIT7PARITY
```

:TRIGger:LOGic:UART:POLarity

Function Sets the logic UART signal trigger polarity or queries the current setting.

Syntax
```
:TRIGger:LOGic:UART:POLarity
{NEGative|POSitive}
:TRIGger:LOGic:UART:POLarity?
```

Example
```
:TRIGGER:LOGIC:UART:POLARITY NEGATIVE
:TRIGGER:LOGIC:UART:POLARITY? ->
:TRIGGER:LOGIC:UART:POLARITY NEGATIVE
```

:TRIGger:LOGic:UART:SOURce

Function Sets the logic UART signal trigger source or queries the current setting.

Syntax
```
:TRIGger:LOGic:UART:SOURce
{A<x>|B<x>|C<x>|D<x>}
:TRIGger:LOGic:UART:SOURce?
```
<x> = 0 to 7

Example
```
:TRIGGER:LOGIC:UART:SOURCE A0
:TRIGGER:LOGIC:UART:SOURCE? ->
:TRIGGER:LOGIC:UART:SOURCE A0
```

Description On 16-bit models, you can only select {A<x>|C<x>}.

:TRIGger:LOGic:UART:SPOint

Function Sets the logic UART signal trigger sample point or queries the current setting.

Syntax
```
:TRIGger:LOGic:UART:SPOint {<NRf>}
:TRIGger:LOGic:UART:SPOint?
```
<NRf> = 18.8 to 90.6(%)

Example
```
:TRIGGER:LOGIC:UART:SPOINT 18.8
:TRIGGER:LOGIC:UART:SPOINT? ->
:TRIGGER:LOGIC:UART:SPOINT 18.8E+00
```

:TRIGger:SOURce:CHANnel<x>:LEVel

Function Sets the trigger level of the channel or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:
LEVel {<Voltage>|<Current>}
:TRIGger:SOURce:CHANnel<x>:LEVel?
```
<x> = 1 to 4
<Voltage> and <Current> = See the DL6000/DLM6000 User's Manual.

Example
```
:TRIGGER:SOURCE:CHANNEL1:LEVEL 1V
:TRIGGER:SOURCE:CHANNEL1:LEVEL?
-> :TRIGGER:SOURCE:CHANNEL1:
LEVEL 1.000E+00
```

:TRIGger:SOURce:CHANnel<x>:STATe

Function Sets the condition to be satisfied of the channel or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:
STATe {DONTcare|HIGH|IN|LOW|OUT}
:TRIGger:SOURce:CHANnel<x>:STATe?
```
<x> = 1 to 4

Example
```
:TRIGGER:SOURCE:CHANNEL1:STATE HIGH
:TRIGGER:SOURCE:CHANNEL1:STATE?
-> :TRIGGER:SOURCE:CHANNEL1:STATE HIGH
```

Description
- This command is valid when `:TRIGger:TYPE EQUalify|I2CBus|PQUalify|PSTate|SPATtern|STATe`.
- {HIGH|LOW} is valid when `:TRIGger:TYPE I2CBus|SPATtern`.
- {IN|OUT} is valid when `:TRIGger:TYPE EQUalify|PQUalify|PSTate|STATe` and `:TRIGger:SOURce:CHANnel<x>:WINDow ON`. {HIGH|LOW} is valid when `:TRIGger:TYPE EQUalify|PQUalify|PSTate|STATe` and `:TRIGger:SOURce:CHANnel<x>:WINDow OFF`.

:TRIGger:TYPE

Function Sets the trigger type or queries the current setting.

Syntax
```
:TRIGger:TYPE {CANBus|EDGE|EICYcle|
EIDelay|EISequence|EOR|EQUalify|
I2CBus|LINBus|LI2Cbus|
LLINbus|LSPAttern|LSPIbus|LPSTate|
LPULse|LQUalify|LSTate|LUARt|
PQUalify|PSTate|PULSe|SPATtern|
SPIBus|STATe|TV|UART}
:TRIGger:TYPE?
```

Example
```
:TRIGGER:TYPE CANBUS:TRIGGER:TYPE?
-> :TRIGGER:TYPE CANBUS
```

Description {LI2Cbus|LLINbus|LSPAttern|LSPIbus|LPSTate|LPULse|LQUalify|LSTate|LUARt} are only available on the DLM6000.

8.1 I²C Bus Signal Analysis

Item	Specifications
Bus transfer rate	Up to 3.4 Mbit/s
Address mode	7 bits or 10 bits
Triggering	
Trigger source	On the DL6000 Series, select the source from CH1 to CH4.
	On the DLM6000 Series, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).
Mode	Every Start: Triggers on a start condition
	ADR&DATA: Triggers by comparing with the specified address or data Address type • 7-bit address • 7-bit + sub address • 10-bit address
	NON ACK: Triggers on a Nack bit
	General Call: Triggers by comparing with the second byte pattern of the general call address
	Start Byte/HS Mode: Triggers on a start byte or an HS mode master address
Analysis	
Signal	On the DL6000 Series, select the signal from CH1 to CH4 or from M1 to M4.
	On the DLM6000 Series, select the signal from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).
Analyzable data bytes	Up to 40000 bytes (20000 bytes before and after the analysis reference point)
Analysis result display	Simple display Analysis number (No.), start condition/stop condition (S/P), hexadecimal data display, address/data (form), read/write signal (R/W), acknowledge bit
	Detail display Analysis number (No.), start condition/stop condition (S/P), time from the trigger position (ms), binary data display, hexadecimal data display, address/data (form), read/write signal (R/W), acknowledge bit, data info
Zoom link	Zoom position (the center of the zoom box) movable to the highlighted byte in the analysis result list. Changing the zoom position also changes the highlighted frame in the analysis result list.

Item	Specifications
Data storage of the analysis result list	Saves the simple or detail display data of the analysis result list in CSV format (.csv extension).
Data search	Searches waveforms for specified address pattern, data pattern, or acknowledge bit state. When a waveform that meets the conditions is found, the zoom box moves to that point, and the DL6000/DLM6000 displays the specified waveform in the zoom waveform area.

8.2 CAN Bus Signal Analysis

Item	Specifications	
Version	CAN Version 2.0B	
Bit rate	Settable to1 M, 500 k, 250 k, 125 k, 83.3 k, 33.3 kbps or to any bit rate from 10 k to 1 Mbps in 0.1-kbps steps. Supports High-speed CAN (ISO11898) and Low-speed CAN (ISO11519-2).	
Triggering		
Trigger source	Selectable from CH1 to CH4.	
Mode	SOF:	Triggers on SOF (Start of Frame)
	Error Frame:	Triggers on an error frame
	ID Std/Data:	Triggers on a data frame or remote frame (ID: standard format)
	ID Ext/Data:	Triggers on a data frame or remote frame (ID: extended format)
	ID/Data OR:	Triggers on the OR logic of four data frame and remote frame types Standard or extended format selectable for each ID
Analysis		
Signal	Select from CH1 to CH4 or from M1 to M4.	
Analyzable frames	Up to 3000 frames (1500 frames before and after the analysis reference point)	
Analyzed frames	Data frame, remote frame, error frame, and overload frame	
Analysis result display	Simple display Analysis number (No.), frame type, hexadecimal ID display, hexadecimal data display, ACK slot state Detail display Analysis number (No.), frame type, time from the trigger position (ms), hexadecimal ID display, hexadecimal DLC display, binary data display, hexadecimal data display, hexadecimal CRC sequence display, ACK slot state	
Zoom link	Zoom link Zoom position (the center of the zoom box) movable to the start of the highlighted frame in the analysis result list. Changing the zoom position also changes the highlighted frame in the analysis result list. Field jump When zoom link is enabled, the zoom position can be moved to the start of the specified field of the highlighted frame in the analysis result list. Selectable fields are SOF, ID, control field, data field, CRC, and ACK.	
Data storage of the analysis result list	Saves the simple or detail display data of the analysis result list in CSV format (.csv extension).	
Data search	Searches waveforms for specified field or frame conditions. When a waveform that meets the conditions is found, the zoom box moves to that point, and the DL6000/DLM6000 displays the specified waveform in the zoom waveform area.	
Stuff bit computation	Extracts stuff bits from the CAN bus signal waveform and displays them as a MATH waveform (from MATH1 to MATH4). Supports bit stuffing. Decoded display available.	

8.3 LIN Bus Signal Analysis

Item	Specifications
Revision	LIN 1.3 or LIN 2.0
Bit rate	Settable to 1200 bps, 2400 bps, 4800 bps, 9600 bps, 19200 bps or to any bit rate from 1 k to 20 kbps in 10-bps steps.
Triggering	
Trigger source	On the DL6000 Series, select the source from CH1 to CH4. On the DLM6000 Series, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).
Mode	Break: Triggers on the break delimiter
Analysis	
Signal	On the DL6000 Series, select the signal from CH1 to CH4 or from M1 to M4. On the DLM6000 Series, select the signal from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).
Analyzable frames	Up to 3000 frames (1500 frames before and after the analysis reference point)
Analyzed fields	Break, Synch, ID, Data, Checksum
Analyzable revisions	LIN 2.0, LIN 1.3, or Both
Analysis result display	Simple display Analysis number (No.), hexadecimal ID display, hexadecimal data display, hexadecimal checksum display Detail display Analysis number (No.), time from the trigger position (ms), hexadecimal ID display, hexadecimal ID-Field display, binary data display, hexadecimal data display, hexadecimal checksum display, additional information
Zoom link	Zoom link Zoom position (the center of the zoom box) movable to the start of the highlighted frame in the analysis result list. Changing the zoom position also changes the highlighted frame in the analysis result list. Field jump When zoom link is enabled, the zoom position can be moved to the start of the specified field of the highlighted frame in the analysis result list. Selectable fields are break, synch, ID, data, and checksum.
Data storage of the analysis result list	Saves the simple or detail display data of the analysis result list in CSV format (.csv extension).
Data search	Searches waveforms for specified field or frame conditions. When a waveform that meets the conditions is found, the zoom box moves to that point, and the DL6000/DLM6000 displays the specified waveform in the zoom waveform area.

8.4 SPI Bus Signal Analysis

Item	Specifications
Triggering	
Trigger source	On the DL6000 Series, select the source from CH1 to CH4. On the DLM6000 Series, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).
Mode	Three-wire or four-wire Triggers by comparing the data the specified number of bytes after the CS assertion. Data length to be compared is selectable from 1 to 4 bytes.
Analysis	
Signal	On the DL6000 Series, select the signal from CH1 to CH4 or from M1 to M4. On the DLM6000 Series, select the signal from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).
Analyzable data bytes	Up to 40000 bytes (20000 bytes before and after the analysis reference point)
Analyzed frames	Data
Analysis result display	Simple display Analysis number (No.), hexadecimal data 1 display, hexadecimal data 2 display, CS status Detail display Analysis number (No.), time from the trigger position to the start bit of each data byte (ms), binary data 1 display, hexadecimal data 1 display, binary data 2 display, hexadecimal data 2 display, CS status, start position/stop position of the active period (S/P)
Zoom link	Zoom position (the center of the zoom box) movable to the highlighted byte in the analysis result list. Changing the zoom position also changes the highlighted frame in the analysis result list.
Data storage of the analysis result list	Saves the simple or detail display data of the analysis result list in CSV format (.csv extension).
Data search	Searches waveforms for the specified data pattern. When a waveform that meets the pattern is found, the zoom box moves to that point, and the DL6000/DLM6000 displays the specified waveform in the zoom waveform area.

8.5 UART Signal Analysis

Item	Specifications
Bit rate	Settable to 1200 bps, 2400 bps, 4800 bps, 9600 bps, 19200 bps, 38400 bps, 57600 bps, 115200 bps or to any bit rate from 1 k to 200 kbps in 100-bps steps.
Data format	8-bit data (no parity bit) 7-bit data + parity bit (selectable only for error trigger) 8-bit data + parity bit (selectable only for error trigger)
Triggering	
Trigger source	On the DL6000 Series, select the source from CH1 to CH4. On the DLM6000 Series, select the source from CH1 to CH4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from A0 to A7, or from C0 to C7 on 16-bit models).
Mode	Every Data: Triggers on the stop bit of all data frames.
Analysis	
Signal	On the DL6000 Series, select the signal from CH1 to CH4 or from M1 to M4. On the DLM6000 Series, select the signal from CH1 to CH4, from M1 to M4, from A0 to A7, from B0 to B7, from C0 to C7, or from D0 to D7 (from CH1 to CH4, from M1 to M4, from A0 to A7, or from C0 to C7 on 16-bit models).
Analyzable frames	Up to 3000 bytes (1500 bytes before and after the analysis reference point)
Analyzed fields	Data, additional information (parity error and framing error)
Analysis result display	Simple display Analysis number (No.) and hexadecimal data display[1] Detail display Analysis number (No.), time from the trigger position (ms), binary data display[1], hexadecimal data display[1], additional information Display Mode ASCII display of hexadecimal values, grouping of data
Zoom link	Zoom position (the center of the zoom box) movable to the start of the highlighted frame in the analysis result list. Changing the zoom position also changes the highlighted frame in the analysis result list.
Data storage of the analysis result list	Saves the simple or detail display data of the analysis result list in CSV format (.csv extension).
Data search	Searches waveforms for specified field or frame conditions. When a waveform that meets the conditions is found, the zoom box moves to that point, and the DL6000/DLM6000 displays the specified waveform in the zoom waveform area.

1 If you set the display mode to ASCII, data is displayed using ASCII codes.

Index

R

S

T

U

W

Z