# DL7100/DL7200

## Digital Oscilloscope
## Communication Interface

# USER'S MANUAL

YOKOGAWA ◆
Yokogawa Electric Corporation

**IM 701410-11E**
3rd Edition

## Introduction

Thank you for purchasing YOKOGAWA's DL7100/DL7200 Digital Oscilloscope.
This Communication Interface User's Manual describes the functions and commands of the GP-IB and serial (RS-232) interfaces. To ensure proper use of the GP-IB/serial (RS-232) interfaces, please read this manual thoroughly.
Keep the manual in a safe place for quick reference whenever a question arises.
Three manuals are provided with the DL7100/DL7200 including this Communication Interface User's Manual.

| Manual Name | Manual No. | Description |
|---|---|---|
| DL7100/DL7200 User's Manual | IM 701410-01E | Describes all functions except for the communications functions and operation procedures of the instrument. |
| DL7100/DL7200 Communication User's Manual | IM 701410-11E | Describes the communications functions of the GP-IB/serial (RS-232) interface. |
| DL7100/DL7200 Operation Guide | IM 701410-02E | Explains basic operations only. |

## Note

- The contents of this manual are subject to change without prior notice as a result of improvements in instrument's performance and functions.
- 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 representative listed on the back cover of this manual.
- Copying or reproduction of all or any part of the contents of this manual without YOKOGAWA's permission is strictly prohibited.

## Trademarks

- HP-GL is a registered trademark of Hewlett-Packard Company.
- IBM PC/AT is a registered trademark of International Business Machines Corporation.
- Other product names are trademarks or registered trademarks of their respective holders.

## Revisions

1st Edition: August 1999
2nd Edition: August 2000
3rd Edition: July 2001

Disk No. DL27
3rd Edition: July 2001 (YK)

# How to Use this Manual

## Structure of this Manual

This User's Manual consists of the following sections.

**Chapter 1** **Overview of the GP-IB Interface**
Describes the functions and specifications of GP-IB.

**Chapter 2** **Overview of the Serial (RS-232) Interface**
Describes the functions and specifications of the serial interface (RS-232).

**Chapter 3** **Network Interface**
Describes the functions and specifications of the network interface.

**Chapter 4** **Before Programming**
Describes formats used when sending a command.

**Chapter 5** **Command**
Describes each command.

**Chapter 6** **Status Report**
Describes the status byte, various registers and queues.

**Chapter 7** **Sample Programs**
Sample programs, written in Visual BASIC, for MS-DOS/V machines equipped with the following GP-IB board: AT-GPIB/TNT IEEE-488.2, from National Instruments.

**Appendix**
Contains references including the ASCII character code table.

**Index**
Provides an alphabetically ordered index.

## Conventions Used in this Manual

### Symbols used for Notes and Keys

| Type | Symbol | Description | |
|---|---|---|---|
| Unit | k<br>K | 1000<br>1024 | e.g.: 100 kS/s (sample rate)<br>e.g.: 640 KB (floppy disk memory capacity) |
| Note | ***Note*** | Provides information that is necessary for proper operation of the instrument. | |
| Key | "Probe" | Refers to a soft key displayed on the screen. | |

### Symbols used in syntax descriptions

Symbols which are used in the syntax descriptions in Chapter 5 are shown below. These symbols are referred to as BNF notation (Backus-Naur Form). For detailed information, refer to page 4-5.

| Symbol | Description | Example | Example of Input |
|---|---|---|---|
| <> | Defined value | `CHANnel <x> <x>=1 to 4` | →`CHANNEL2` |
| {}<br>\| | One of the options in {} is selected.<br>Exclusive OR | `COUPling {AC\|DC\|DC50\|GND}` | →`COUPLING AC` |
| [] | Abbreviated | `TRIGger [:SIMPle]:SLOPe` | →`TRIGger:SLOPe` |

# Contents

## Chapter 6 Status Report

## Chapter 7 Sample Program

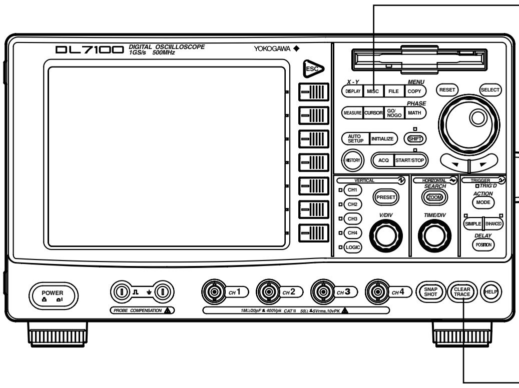
## Appendix

## Index

# Chapter 1 Overview of the GP-IB Interface

## 1.1 Names of the Parts and Their Function

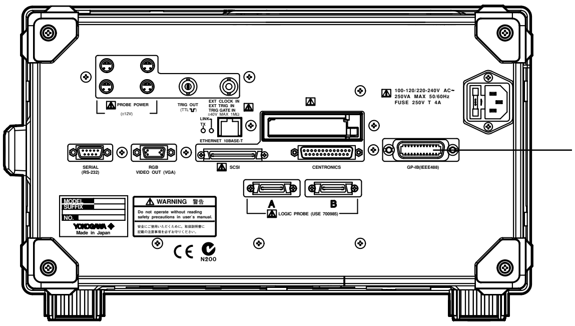
### Front Panel



**MISC key (page 1-6)**
Press to enter the communication settings such as the address and the timeout.

**SHIFT+CLEAR TRACE key**
Press to switch from remote mode to local mode which allows key operation.
However, this is not possible if Local Lockout has been set by the controller (refer to page 1-7).

### Rear Panel



**GP-IB connector**
This connector is for connecting the controller (such as a PC) with the GP-IB cable. For information on how to connect the GP-IB cable, refer to the following page.

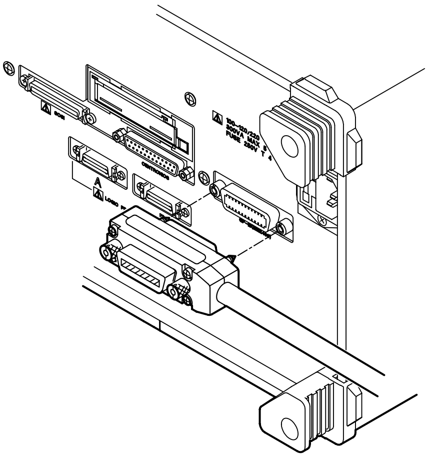
## 1.2 Connecting the GP-IB Cable

### GP-IB Cable

The GP-IB connector on the side panel of the DL716 is a 24-pin connector that conforms to IEEE Standard 488-1978. Use a GP-IB cable that also conforms to IEEE Standard 488-1978.

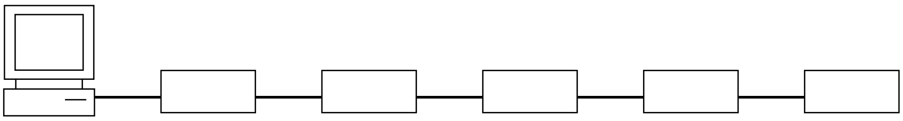
### Connection Method

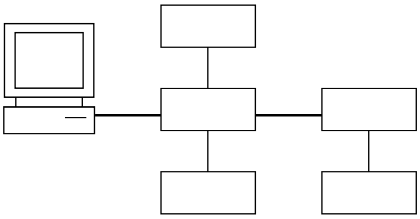
Connect the GP-IB cable as shown below.



### Connection Precautions

- Be sure to tighten the screws on the GP-IB cable connector firmly.
- The instrument can be connected to more than one item of equipment (e.g. a personal computer) if more than one GP-IB cable is used. However, it is not possible to connect more than 15 items of equipment (including the controller) to a single bus.
- If you connect the instrument to more than one item of equipment, make sure that a different address is used for each item.
- Each connecting cable must be 2 m or less in length.
- The total length of all the cables must not exceed 20 m.
- While communications are in progress, more than two-thirds of the connected equipment items must be turned ON.
- When connecting more than one item of equipment, connect them so that the connection route forms a star or linear configuration. Loop or parallel wiring is not allowed.





### CAUTION

Be sure to switch off power to both your PC and the oscilloscope before connecting or disconnecting cables. Failure to switch power off may cause internal circuit failure or improper operation.

# 1.3 GP-IB Interface Functions

## GP-IB Interface Functions

### Listener function

- Allows you to make the settings which you can make using the panel keys on the instrument, except for the power ON/OFF and GP-IB communications settings.
- Receives commands from a controller requesting output of set-up and waveform data. Also receives status report commands.

### Talker function

Outputs set-up and waveform data.

***Note***

The listen-only, talk-only and controller functions are not available on this instrument.

## Switching between Remote and Local Modes

### When switched from Local to Remote Mode

Remote mode is activated when a REN (Remote Enable) message is received from a controller while local mode is active.

- REMOTE is displayed on the lower right of the screen.
- All front panel keys except the  SHIFT+CLEAR TRACE key can no longer be operated any more.
- Settings entered in local mode are retained.

### When switched from Remote to Local Mode

Pressing the SHIFT+CLEAR TRACE key in remote mode puts the instrument in local mode.  However, this is not possible if Local Lockout has been set by the controller (page 1-7).

- The REMOTE indicator on the lower right of the screen is turned off.
- All front panel keys are operative.
- Settings entered in remote mode are retained.

# 1.4 GP-IB Interface Specifications

## GP-IB Interface Specifications

| | |
|---|---|
| Electrical and mechanical specifications | : Conforms to IEEE Standard 488-1978. |
| Interface functions | : Refer to the table below. |
| Protocol | : Conforms to IEEE Standard 488.2-1992. |
| Code | : ISO (ASCII) code |
| Mode | : Addressable mode / Talk-only mode (switched automatically) |
| Address setting | : Addresses 0 to 30 can be selected from the GP-IB setting screen, displayed when you press the MISC key. |
| Remote mode clear | : Remote mode can be cleared by pressing the SHIF+CLEAR TRACE key. However, this is not possible if Local Lockout has been set by the controller. |

**Interface functions**

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

## Data Transfer Rate

The table below shows approximate response times for output of waveform data, assuming the following configuration.

| | |
|---|---|
| Model | : 701410 |
| Controller | : DELL GXMT 5133 |
| GB-IB Board | : NI AT-GPIB/TNT(PNP) |
| Programming language | : Visual C++ |

| Data volume | Byte format | Word format | ASCII format |
|---|---|---|---|
| 1000 | about 25 ms | about 28 ms | about 650 ms |
| 10000 | about 60 ms | about 100 ms | about 6.4 s |
| 100000 | about 410 ms | about 950 ms | about 63 s |
| 1000000 | about 4 s | about 8.5 s | about 622 s |

# 1.5 Setting  Up the Instrument (GP-IB)

## *Before You Begin*

When you make settings which can be made using the front panel keys of the instrument or when you output set-up data or waveform data using the controller, the following settings must be made.

**Setting the address**

This function allows you to set the instrument's address for addressable mode within the range of 0 to 30. Each item of equipment connected via a GP-IB interface has its own address, by which it can be identified. Care must be taken to ensure that all interconnected devices are assigned unique addresses.

***Note***

Do not change the address while the GP-IB interface is being used by the controller.

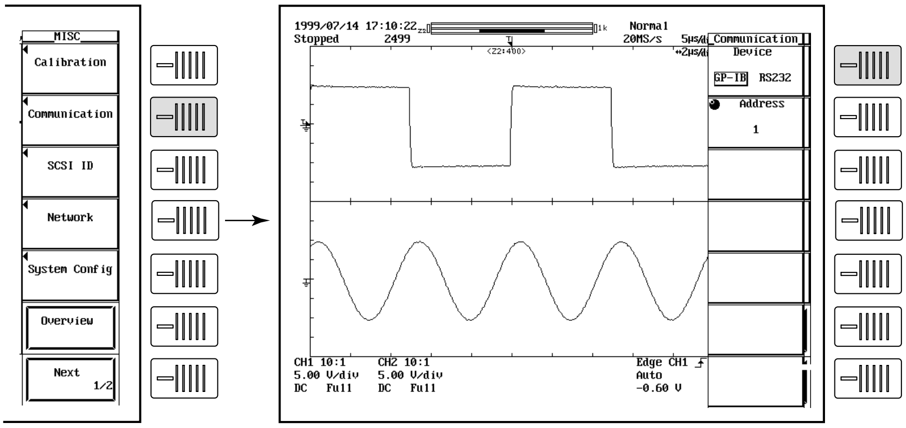
## *Operationg Procedure*

**Displaying the GP-IB menu**

1. Press the MISC key.
2. Press the "Communication" soft key.
3. Press the "Device" soft key to select "GP-IB."

**Setting the address**

4. Turn the jog shuttle to set the desired address.



```
1999/07/14 17:10:22            Normal
Stopped        2499            20MS/s   5µs/div
Communication
Device
GP-IB  RS232
Address
1
CH1 10:1     CH2 10:1          Edge CH1
5.00 V/div   5.00 V/div        Auto
DC   Full    DC   Full         -0.60 V
```

***For DL7100s prior to firmware version 2.01, “Graphic Color” (instead of “Network”) is displayed when you press the MISC key.**

# 1.6 Response to Interface Messages

## Response to Interface Message

**Response to a uni-line message**

**IFC (Interface Clear)**

Clears the talker and listener. Stops output if data is being output.

**REN (Remote Enable)**

Switches between remote and local modes.

IDY (Identify) is not supported.

**Response to a multi-line message (address command)**

**GTL (Go To Local)**

Switches to local mode.

**SDC (Selected Device Clear)**

Clears the program message (command) which is currently being output. Also clears the output queue (page 6-5).
`*OPC` and `*OPC?` will be disabled if they are currently being executed.
`*WAI` and `COMMunicate:WAIT` will be stopped immediately.

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

**Response to a multi-line message (universal command)**

**LLO (Local Lockout)**

Invalidates the SHIFT+CLEAR key on the front panel to disable switching to local mode.

**DCL (Device Clear)**

Same as SDC.

**SPE (Serial Poll Enable)**

Sets the talker function to serial poll mode for all equipment connected to the communications bus. The controller performs polling on equipment sequentially.

**SPD (Serial Poll Disable)**

Clears serial poll mode as the talker function for all equipment connected to the communications bus.

PPU (Parallel Poll Unconfigure) is not supported.

## What is an Interface Message?

An interface message is also called an interface command or bus command, and is issued by the controller. Interface messages are classified as follows.

**Uni-line messages**

Messages are transferred through a single control line. The following three types of uni-line message are available.
IFC (Interface Clear)
REN (Remote Enable)
IDY (Identify)

### Multi-line message

Eight data lines are used to transmit a message. Multi-line messages are classified as follows.

### Address commands

Valid when the equipment is designated as a listener or a talker. The following five address commands are available.

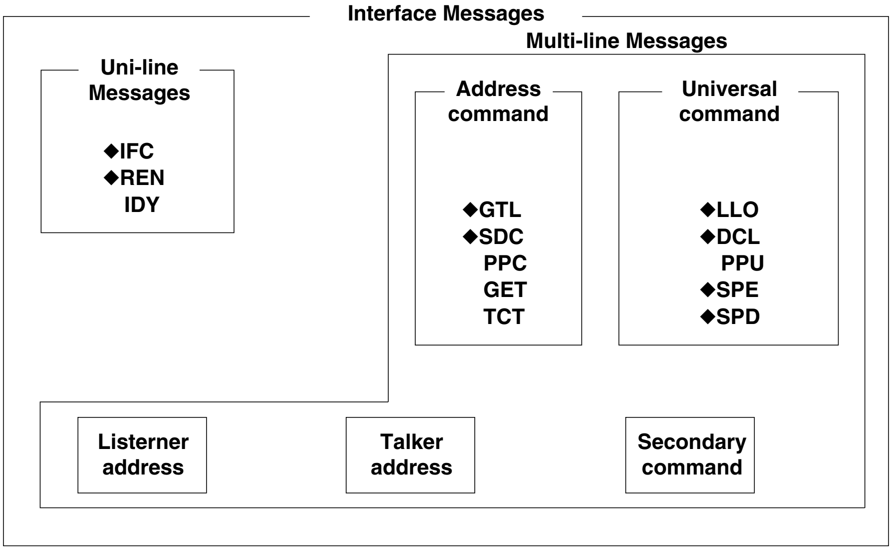
- Commands valid for equipment designated as a listener;
  GTL (Go To Local)
  SDC (Selected Device Clear)
  PPC (Parallel Poll Configure)
  GET (Group Execute Trigger)
- Command valid for equipment designated as a talker;
  TCT (Take Control)

### Universal commands

Valid for any item of equipment, irrespective of whether the item is designated as a listener or a talker. The following five universal commands are available.

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

In addition to the above commands, a listener address, talker address on secondary command can be sent in an interface message.



Messages marked with a "◆" are interface messages supported by the DL716.

***Note***

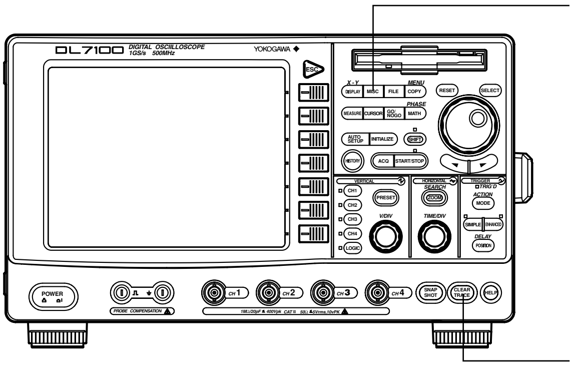
**Differences between SDC and DCL**

The SDC command is an address command and requires that both the talker and listener be designated; however DCL is a universal command and does not require that the talker and listener be designated. Therefore, SDC is used for particular items of equipment, while DCL can be used for any equipment connected to the communications bus.

# Chapter 2 Overview of the Serial (RS-232) Interface

## 2.1 Names of the Parts and Their Function

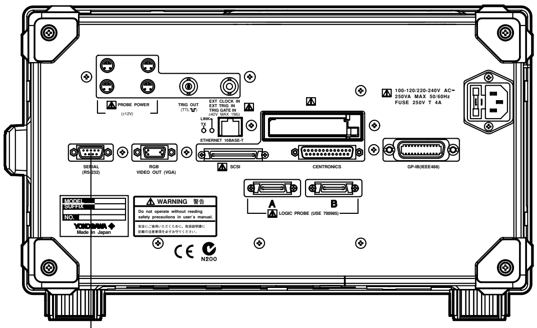
### Front Panel



**MISC key (page 2-8)**
Press to enter the communication settings such as the baud rate, data format, and the handshaking method.

**SHIFT+CLEAR TRACE key**
Press to switch from remote mode to local mode which allows key operation.
However, this is not possible if Local Lockout has been set by the controller (refer to page 2-2).

### Rear Panel



**SERIAL (RS-232) connector**
This connector is for connecting the controller (such as a PC) with the RS-232 cable. For information on how to connect the RS-232 cable, refer to the following page.

## 2.2 Serial (RS-232) Interface Functions and Specifications

### Receiving Function

It is possible to make the same settings via the serial (RS-232) interface as can be made using the front panel keys.

Measured/computed data, panel set-up information and error codes can be received.

### Sending Function

Measured/computed data can be output.

Panel set-up information and the status byte can be output.

Error codes which have occurred can be output.

### RS-232 Interface Specifications

| | |
|---|---|
| Electrical characteristics | : Conforms to EIA RS-232. |
| Connection | : Point-to-point |
| Communications | : Full-duplex |
| Synchronization | : Start-stop system |
| Baud rate | : 1200, 2400, 4800, 9600, 19200, 38400, 57600 |
| Start bit | : 1 bit (fixed) |
| Data Length | : 7 or 8 bits |
| Parity | : Even, odd or no parity |
| Stop Bit | : 1 or 2 bits |
| Connector | : DELC-J9PAF-13L6 (JAE or equivalent) |
| Hardware handshaking | : User can select whether CA or CB signals will always be True, or will be used for control. |
| Software Handshaking | : User can control both transmission and reception using X-on and X-off signals.<br>X-on (ASCII 11H)<br>X-off (ASCII 13H) |
| Receive buffer size | : 256 bytes |

### Switching between Remote and Local Modes

**When switched from Local to Remote Mode**

Remote mode is activated when the "`COMMunicate:REMote ON`" command is received from a controller while local mode is active.

- REMOTE is displayed on.
- All front panel keys except the SHIFT+CLEAR TRACE key can no longer be operated any more.
- Settings entered in local mode are retained.

**When switched from Remote to Local Mode**

Pressing the SHIFT+CLEAR TRACE key in remote mode puts the instrument in local mode. However, this is not possible of Local Lockout (when the "`COMMunicate:LOCKout ON`" command is received) has been set by the controller.

Local mode is activated when the "`COMMunicate:REMote OFF`" command regardless of Local Lockout.

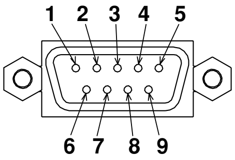
- The REMOTE indicator is turned off.
- All front panel keys are operative.
- Settings entered in remote mode are retained.

## 2.3 Connecting the Serial (RS-232) Interface Cable

When connecting this instrument to a computer, make sure that the handshaking method, data transmission rate and data format selected for the instrument match those selected for the computer.

For details, refer to the following pages. Also make sure that the correct interface cable is used.

### Connector and Signal Names



2. RD (Received Data) : Data received from personal computer.
Signal direction...Input.

3. SD (Send Data) : Data transmitted to a personal computer.
Signal direction...Output.

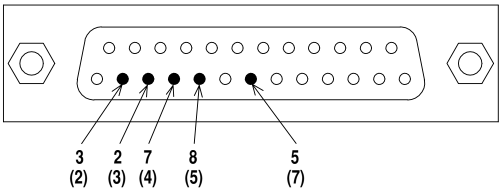
5. SG (Signal Ground) : Ground for signals.

7. RS (Request to Send) : Signal used for handshaking when receiving data from a personal computer.
Signal direction...Output.

8. CS (Clear to Send) : Signal used for handshaking when transmitting data to a personal computer.
Signal direction...Input.

Pin Nos. 1, 4, 6, and 9 are not used.

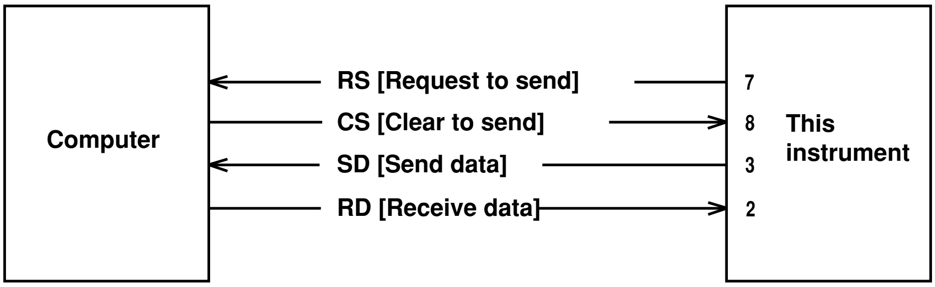
### 9-25 Pin Connector



The number between brackets refer to the pin Nos. of the 25-pin connector.

### Signal Direction

The figure below shows the direction of the signals used by the serial (RS-232) interface.



## Table of RS-232 Standard Signals and their JIS and CCITT Abbreviations

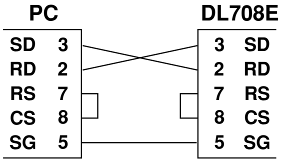
| Pin No.<br>(9-pin connector) | Abbreviation | | | Description |
|---|---|---|---|---|
| | RS-232 | CCITT | JIS | |
| 5 | AB (GND) | 102 | SG | Signal ground |
| 3 | BA (TXD) | 103 | SD | Transmitted data |
| 2 | BB (RXD) | 104 | RD | Received data |
| 7 | CA (RTS) | 105 | RS | Request to send |
| 8 | CB (CTS) | 106 | CS | Clear to send |

## Signal line connection examples

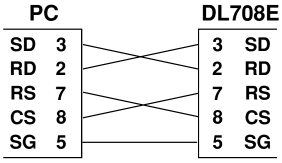
The pin numbers shown are that of 9-pin connectors.

In general, use a cross cable.

• OFF-OFF / XON-XON



• CTS-RTS(CS-RS)



## 2.4 Handshaking

To use an serial (RS-232) interface for transferring data between this instrument and a computer, it is necessary to use certain procedures by mutual agreement to ensure the proper transfer of data. These procedures are called "handshaking." Various handshaking systems are available depending on the computer to be used; the same handshaking system must be used for both the computer and this instrument.
This instrument allows you to choose any handshaking mode from the following three modes.

**Handshake format Descriptions**→ ○

| **Handshake Method** | | **Data Sending Control (control method when sending data to a computer)** | | | **Data Receiving Control (control method when receiving data from a computer)** | | |
|---|---|---|---|---|---|---|---|
| | | **Software Handshake** | **Hardware Handshake** | No handshake | **Software Handshake** | **Hardware Handshake** | No handshake |
| | **The menu of this instrument** | Sending stops when X-off is received, and sending is resumed when X-on is received. | Sending stops when CB(CTS) is False, and sending is resumed when CB is True. | | X-off is sent when received data buffer becomes 3/4-full, and X-on is sent when the received data buffer is only 1/4-full. | CA (RTS) is set to False when received data buffer is only 3/4-full, and is set to True when received data buffer is only 1/4-full. | |
| OFF-OFF | NO-NO | | | ○ | | | ○ |
| XON-XON | XON-XON | ○ | | | ○ | | |
| CS-RS | CTS-RTS | | ○ | | | ○ | |

### OFF-OFF

- **Transmission data control**
  There is no handshake status between the instrument and host computer. The X-OFF and X-ON signal from the host computer is processed as data, and the CS signal is ignored.
- **Reception data control**
  There is no handshake status between the recorder and host computer. When the recorder reception buffer becomes full, the excess data is discarded. RS = True (fixed).

### XON-XON

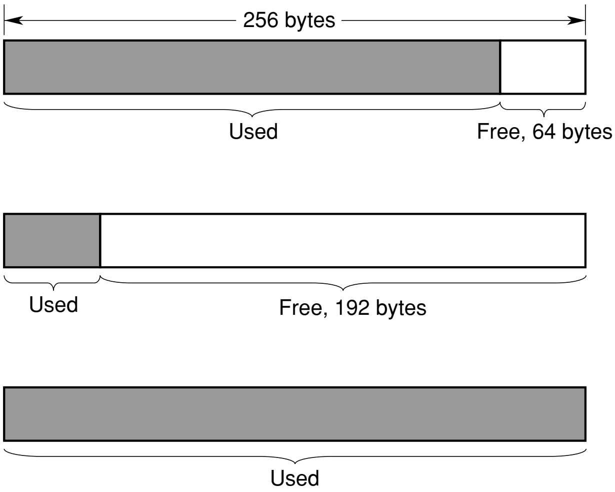
- **Transmission data control**
  A software handshake status is established between the instrument and host computer. The instrument will stop a data transmission when an X-OFF signal is received from the host computer, and will resume transmission when the next X-ON signal is received. A CS signal from the host computer is ignored.
- **Reception data control**
  A software handshake status is established between the instrument and host computer. When the intstruments reception buffer vacancy reaches 64bytes, the X-OFF signal will be sent to the host computer. When the reception buffer vacancy reaches 192 bytes, the X-ON signal will be sent. RS = True (fixed).

## CS-RS

- **Transmission data control**
  A software handshake status is established between the instrument and host computer. The instrument will stop a data transmission if a "CS = False" status is established, and will resume the transmission when a "CS = True" status is established. The X-OFF and X-ON signals from the host computer are processed as data.
- **Reception data control**
  A hardware handshake status is established between the instrument and host computer. When the intstruments reception buffer vacancy reaches 64bytes, an "RS=False" status will be established. When the reception buffer vacancy reaches 192 bytes, an "RS=True" status will be established.

## Precautions Regarding Data Receiving Control

When handshaking is used to control the reception of data, data may still be sent from the computer even if the free space in the receive buffer drops below 64 bytes. In this case, after the receive buffer becomes full, the excess data will be lost, whether handshaking is in effect or not. Data storage to the buffer will begin again when there is free space in the buffer.



When handshaking is in use, reception of data will stop when the free space in the buffer drops to 64 bytes since data cannot be passed to the main program fast enough to keep up with the transmission.

After reception of data stops, data continues to be passed to the internal program. Reception of data starts again when the free space in the buffer increases to 192 bytes.

Whether handshaking is in use or not, if the buffer becomes full, any additional data received is no longer stored and is lost.

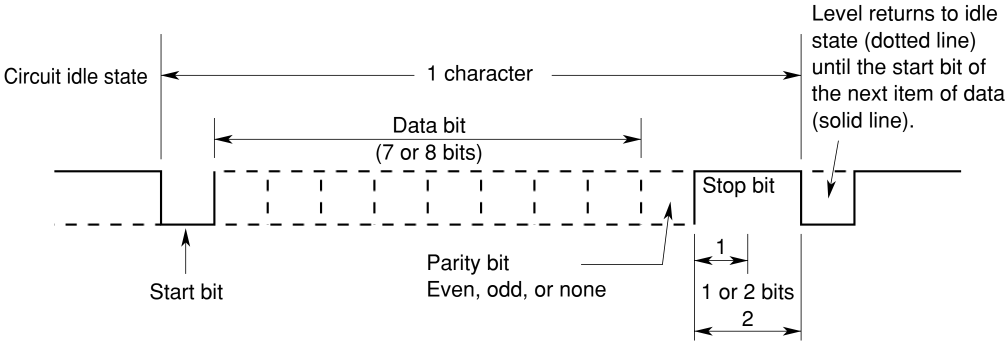
Data Receiving Control using Handshaking

***Note***

It is necessary to create a host computer program which prevents the buffers of both the intrument and the computer from becoming full.

## 2.5 Matching the Data Format

The serial (RS-232) interface of this instrument performs communications using start-stop synchronization. In start-stop synchronization, one character is transmitted at a time. Each character consists of a start bit, data bits, a parity bit, and a stop bit. Refer to the figure below.



# 2.6 Setting Up the Instrument (Serial)

## *Before You Begin*

When using the controller to set the items which can be set locally using the keys on the instrument, or when outputting the setup information or the waveform data to the controller, set the following items.

**Baud rate**

Select from the following choices.

1200, 2400, 4800, 9600, 19200, 38400, 57600

**Data format**

Select the combination of the data length and the stop bit from the following choices.

8-NO-1, 7-EVEN-1, 7-ODD-1, 7-NO-2

**Handshaking method**

Select the transmit data control and the receive data control from the following choices.

NO-NO, XON-XON, CTS-RTS

**Terminator**

Select from the following choices. The terminator used when sending the data from this instrument is selected on the menu. Use either "LF" or "CR+LF" for the terminator in receiving the data.

CR, LF, CR+LF

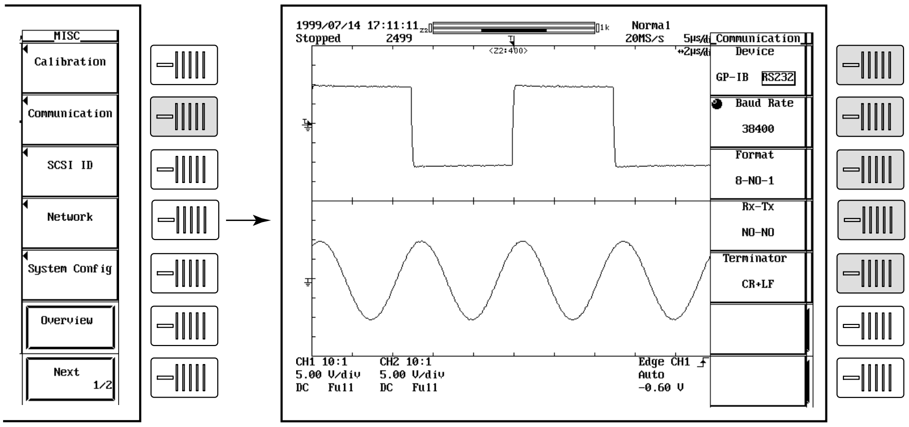
## *Operating Procedure*

**Displaying the serial (RS-232) menu**

1. Press the MISC key.
2. Press the "Communication" soft key.
3. Press the "Device" soft key to select "RS232."

**Selecting the baud rate, the data format and etc.**

4. Press the "BaudRate" (baud rate), "Format" (data format), "Rx-Tx" (handshaking method), and the "Terminator" (terminator) soft keys individually, and turn the jog shuttle to set each item.



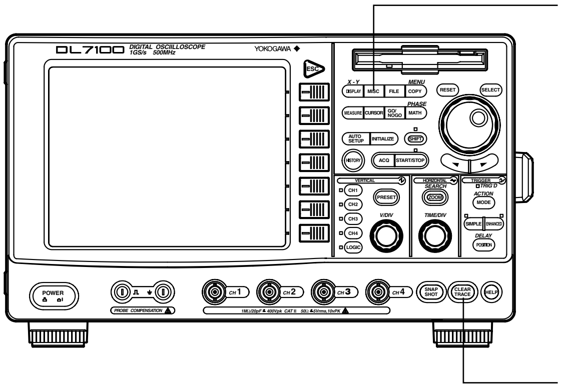
Device GP-IB RS232
Baud Rate 38400
Format 8-NO-1
Rx-Tx NO-NO
Terminator CR+LF

***For DL7100s prior to firmware version 2.01, “Graphic Color” (instead of “Network”) is displayed when you press the MISC key.**

# Chapter 3 Overview of the Network Interface

## 3.1 Names of the Parts and Their Function

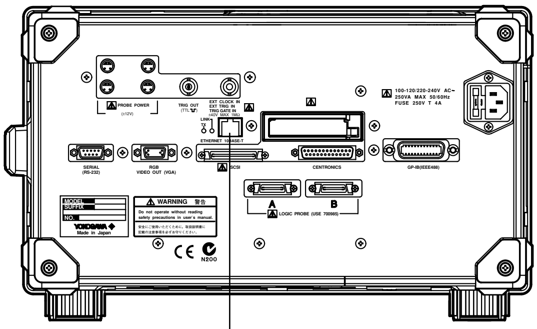
### Front Panel



**MISC key (page 3-6)**
Press to enter the user name and password for user verification.

**SHIFT+CLEAR TRACE key**
Press to switch from remote mode to local mode which allows key operation.
However, this is not possible if Local Lockout has been set by the controller (refer to page 3-2).

### Rear Panel



**Ethernet interface connector**
Connector used to connect the DL7100/DL7200 to the controller (PC) using an Ethernet cable. For details on how to connect the cable, see page 3-4.

## 3.2 Network Interface Functions and Specifications

### Receiving Function

You can specify the same settings as those specified by front panel key operations.
Receives output requests for measured and computed data, setting parameters of the panel, and error codes.

### Sending Function

Outputs measured and computed data.
Outputs setting parameters of the panel and the status byte.
Outputs error codes that are generated.

### Network Interface Specifications

| | |
|---|---|
| Electrical and mechanical specifications | : Conforms to IEEE802.3 |
| Number of simultaneous connections | : 1 |
| Port number | : 10001/tcp |

### Data Transfer Rate

The response time when outputting waveform data is shown below as a reference.

| | |
|---|---|
| Target model | : 701440 main unit |
| Controller | : EPSON DIRECT MT-4000 (Pentium III, 1 GHz) |
| Network adapter | : 3Com 3C920 |
| Language used | : VisualC++ |

| Number of Data Points | Byte Data | Word Data | ASCII Data |
|---|---|---|---|
| 1000 | 7.4 ms | 9.7 ms | 344 ms |
| 10000 | 33.3 ms | 56.4 ms | 3.54 s |
| 100000 | 370 ms | 582 ms | 34.14 s |
| 1000000 | 2834 ms | 4950 ms | 342.0 s |
| 4000000 | 11.21 s | 19.86 s | 1344.5 s |
| 16000000 | 44.8 s | 79.4 s | 5280.7 s |

### Switching between Remote and Local Modes

**When switching from local to remote mode**

If the DL7100/DL7200 receives a “`:COMMunicate:REMote ON`” command from the PC when it is in the local mode, it switches to the remote mode.

- REMOTE is displayed at the upper left corner of the screen.
- All keys except SHIFT+CLEAR TRACE are disabled.
- The settings that existed in the local mode are maintained even when the DL7100/DL7200 switches to the remote mode.

**When switching from remote to local mode**

Pressing SHIFT+CLEAR TRACE when the instrument is in the remote mode causes the instrument to switch to the local mode. However, this is void when the DL7100/DL7200 has received a “`:COMMunicate:LOCKout ON`” command from the PC (local lockout condition). When the DL7100/DL7200 receives a “`:COMMunicate:REMote OFF`” command from the PC, the DL7100/DL7200 switches to the local mode regardless of the local lock condition.

- REMOTE indication at the upper right corner disappears.
- Key operations are enabled.
- The settings that existed in the remote mode are maintained even when the instrument switches to the local mode.

***Note***

The network interface cannot be used simultaneously with another communication interface (GP-IB, serial (RS-232), or USB).

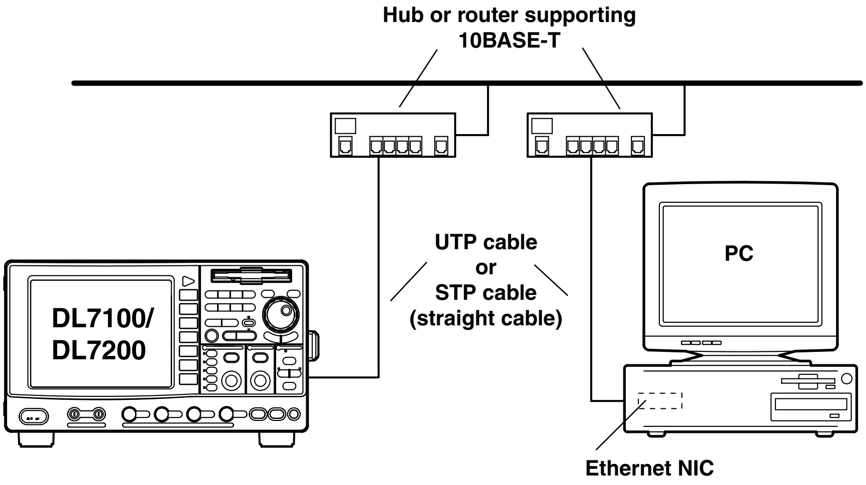
## User Verification Function

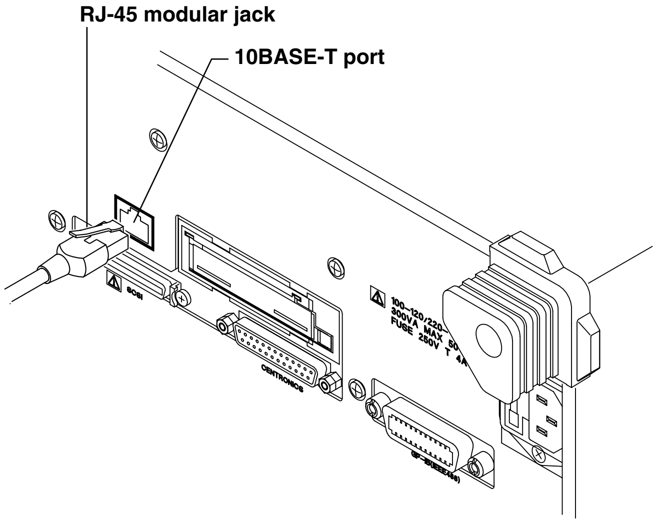
The network interface has a user verification function. A user name and password for the DL7100/DL7200 are set in advance. The user name and password are set on the Communication setup screen under the MISC menu of the DL7100/DL7200. For details, see section 3.4, “Setting the DL7100/DL7200 (Network).”

# 3.3 Connection via the Network Interface

## Connection Method

Connect a UTP (Unshielded Twisted-Pair) cable or an STP (Shielded Twisted-Pair) cable that is connected to a hub, for example, to the 10BASE-T port on the rear panel of the DL7100/DL7200.





## Connection Precautions

To connect the DL7100/DL7200 and a PC, use a straight cable via a hub.

# 3.4 Setting Up the Instrument (Network)

## Function

Carry out the following settings when using a controller to set information that can be specified through key operation on the DL7100/DL7200 or when outputting setting parameters or output waveform data to the controller.

### Setting the User Name and Password

The network interface has a user verification function. A user name and password for the DL7100/DL7200 are set in advance.

- Setting the user name
  Enter the user name using up to 15 characters. The default setting is “anonymous.”
- Setting the password
  Enter the password using up to 15 characters.

***Note***

If the user verification fails, connection to the DL7100/DL7200 is dropped.
Password is not required if the user name is “anonymous.”
If you change the user name, password, and time out, power cycle the instrument.

### Setting Time Out

The network connection is automatically dropped, if there is no access to the DL7100/DL7200 within the time specified here.

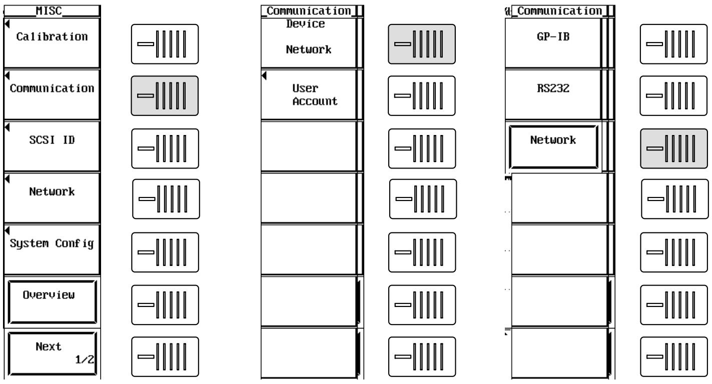
### Setting the TCP/IP

For details, see section 13.2, “Setting the Ethernet Interface (TCP/IP)” in the *DL7100/DL7200 User’s Manual IM701410-01E*.

## Operating Procedure

### Displaying the Network Menu

1. Press the MISC key.
2. Press the “Communication” soft key.
3. Press the “Device” soft key and select “Network.”

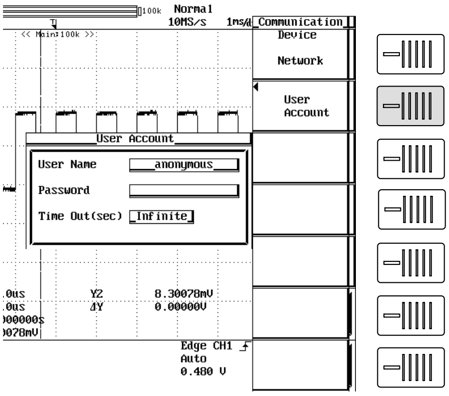


***Note***

Only the communication interface selected by Device is enabled. The DL7100/DL7200 does not accept commands that are transmitted to other unselected communication interfaces.

### Setting the User Name, Password, and Time Out

4. Press the “User Account” soft key to display the User Account dialog box.



User Account

User Name anonymous

Password

Time Out(sec) Infinite

5. Turn the jog shuttle to move the cursor to “User Name” and press the “SELECT” key.
6. Enter the user name. Specify anonymous if you wish to allow access to all users. To restrict access, enter the user name using up to 15 characters.
7. Turn the jog shuttle to move the cursor to “Password” and press the “SELECT” key.
8. Enter the password using up to 15 characters. Enter the password again for confirmation.
9. Turn the jog shuttle to move the cursor to “Time Out” and press the “SELECT” key.
10. Turn the jog shuttle to set the timeout time. Press the “SELECT” key to confirm. Pressing the “RESET” key will reset the value to “Infinite.”

***Note***

For details on the keyboard (software keyboard) operation, see section 4.1, “Entering Values and Character Strings” in the *DL7100/DL7200 User’s Manual IM701410-01E*.

User name and password are case-sensitive.

### Setting the TCP/IP

To use the network interface function, the following TCP/IP settings must be entered.

- IP address
- Subnet mask
- Default gateway

For details on how to enter these parameters, see section 13.2, “Setting the Ethernet Interface (TCP/IP)” in the *DL7100/DL7200 User’s Manual IM701410-01E*.

# Chapter 4 Before Programming

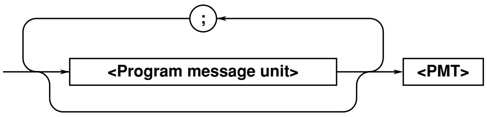
## 4.1 Messages

Blocks of message data are transferred between the controller and this instrument during communications. Messages sent from the controller to this instrument are called program messages, and messages sent back from this instrument to the controller are called response messages.

If a program message contains a query command, i.e. a command which requests a response, this instrument returns a response message. A single response message is always returned in reply to a program message.

### Program Messages

The format of a program message is shown below.



### <Program message unit>

A program message consists of zero or more program message units; each unit corresponds to one command. This instrument executes commands one by one according to the order in which they are received.

Program message units are delimited by a ";."

For a description of the format of the program message unit, refer to the explanation given further below.

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

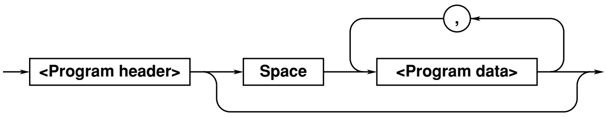
### <PMT>

PMT is a terminator used to terminate each program message. The following three types of terminator are available.

| | |
|---|---|
| NL (New Line) | : Same as LF (Line Feed). ASCII code "0AH" is used. |
| ^END | : END message defined in IEEE488.1. (EOI signal) (The data byte sent with an END message will be the final item of the program message unit.) |
| NL^END | : NL with an END message attached (NL is not included in the program message unit.) |

### Program message unit format

The format of a program message unit is shown below.



### <Program header>

A program header is used to indicate the command type. For details, refer to page 4-3.

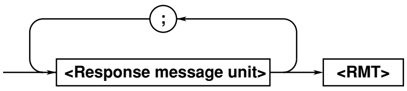
### <Program data>

If certain conditions are required for the execution of a command, program data must be added. Program data must be separated from the header by a space (ASCII code "20H"). If multiple items of program data are included, they must be separated by a "," (comma). For details, refer to page 4-5.

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

### Response Messages

The format of a response message is shown below.



### <Response message units>

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

Response message units are delimited by a ";."

For the response message format, refer to the next page.

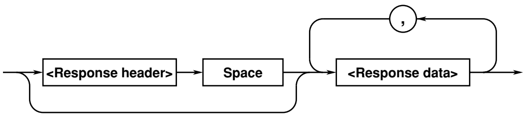
Example $\underbrace{\texttt{:ACQUIRE:MODE NORMAL}}_{\text{Unit}}\texttt{;}\underbrace{\texttt{COUNT 1}}_{\text{Unit}}\texttt{<RMT>}$

### <RMT>

RMT is the terminator used for every response message. Only one type of response message is available; NL^END.

### Response message unit format

The format of a program message unit is shown below.



### <Response header>

A response header sometimes precedes the response data. Response data must be separated from the header by a space. For details, refer to page 4-5.

### <Response data>

Response data is used to define a response. If multiple items of response data are used, they must be separated by a "," (comma). For details, refer to page 4-5.

Example $\underbrace{\texttt{1.25E-02<RMT>}}_{\text{Data}}$ $\underbrace{\texttt{:ACQUIRE:MODE}}_{\text{Header}}$ $\underbrace{\texttt{NORMAL<RMT>}}_{\text{Data}}$

If a program message contains more than one query, responses are made in the same order as the queries. Normally, each query returns only one response message unit, but there are some queries which return more than one response message unit. The first response message unit always responds to the first query, but it is not always true that the 'n'th unit always responds to the 'n'th query. Therefore, if you want to make sure that a response is made to each query, the program message must be divided up into individual messages.

### Points to Note concerning Message Transmission

- It is always possible to send a program message if the previous message which was sent did not contain any queries.
- If the previous message contained a query, it is not possible to send another program message until a response message has been received. An error will occur if a program message is sent before a response message has been received in its entirety. A response message which has not been received will be discarded.
- If an attempt is made by the controller to receive a response message, even if there it no response message, an error will occur. An error will also occur if the controller makes an attempt to receive a response message before transmission of a program message has been completed.

- If a program message of more than one unit is sent and some of the units are incomplete, this instrument receives program message units which the instrument thinks complete and attempts to execute them. However, these attempts may not always be successful and a response may not always be returned, even if the program message contains queries.

### Dead Lock

This instrument has a buffer memory in which both program and response messages of 1024 bytes or more can be stored. (The number of bytes available will vary depending on the operating state of the instrument.) If both buffer memories become full at the same time, this instrument becomes inoperative. This state is called deadlock. In this case, operation can be resumed by discarding the response message.
No dead lock will occur, if the size of the program message including the PMT is kept below 1024 bytes. Furthermore, no deadlock will occur if the program message does not contain a query.

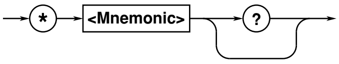
## 4.2 Commands

There are three types of command (program header) which can be sent from the controller to this instrument. They differ in the format of their program headers.

They are
- Common command header
- Compound header
- Simple header

### Common Command Header

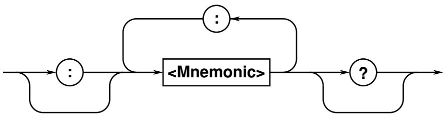
Commands defined in IEEE 488.2-1987 are called common commands. The header format of a common command is shown below. An asterisk (*) must always be attached to the beginning of a command.



An example of a common command
`*CLS`

### Compound Header

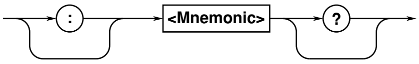
Commands designed to be used only with this instrument are classified and arranged in a hierarchy according to their function. The format of a compound header is illustrated below. A colon (:) must be used when specifying a lower-level header.



An example of a compound header
`:ACQuire:MODE`

### Simple Header

These commands (headers) are functionally independent of each other and are not arranged hierarchically. The format of a simple header is shown below.



An example of a simple header
`:STARt`

***Note***

A mnemonic is a character string made up of alphanumeric characters.

### When Concatenating Commands

#### Command Group

A command group is a group of commands which have the same compound header. A command group may contain sub-groups.

Example Commands relating to acquisition settings

```
:ACQuire:AVERage:COUNt     :ACQuire:MODE
:ACQuire:AVERage:EWEight   :ACQuire:REPetitive
:ACQuire:CLOCK             :ACQuire:RLENgth
:ACQuire:COUNt
:ACQuire:INTerleave
```

#### When Concatenating Commands of the Same Group

This instrument stores the hierarchical level of the command which is currently being executed, and performs analysis on the assumption that the next command to be sent will also belong to the same level. Therefore, it is possible to omit the header if the commands belong to the same group.

Example `:ACQuire:MODE NORMal;COUNt 1<PMT>`

#### When Concatenating Commands of Different Groups

A colon (:) must be included before the header of a command, if the command does not belong to the same group as the preceding command. It is impossible to omit a colon (:).

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

#### When Concatenating Simple Headers

When you type in a simple header after another command, you must include a colon (:) before the simple header. It is impossible to omit a colon (:).

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

#### When Concatenating Common Commands

Common commands defined in IEEE 488.2-1987 are independent of hierarchical level. Thus, it is not necessary to add a colon (:) before a common command.

Example `:ACQuire:MODE NORMal;*CLS;`
`COUNt 1<PMT>`

### When Separating Commands with <PMT>

If a terminator is used to separate two commands, each command is a separate message. Therefore, the common header must be typed in for each command even when commands of the same command group are being concatenated.

Example
```
:ACQuire:MODE NORMal<PMT>:
ACQuire:COUNt 1<PMT>
```

### Upper-level Query

An upper-level query is a compound header to which a question mark is appended. Execution of an upper-level query allows all a group's settings to be output at once. Some query groups comprising more than three hierarchical levels can output all their lower level settings.

Example
```
:CHANnel1?<PMT>→:CHANNEL1:
DISPLAY ON;LABEL "CH1    ";
COUPLING DC;POSITION 0.00;PROBE 10;
VDIV 50.0E+00;BWIDTH FULL;
OFFSET 0.0E+00;LSCALE:MODE 0
```

In reply to a query, a response can be returned as a program message to this instrument. Transmitting a response can restore the settings made when the query was executed. However, some upper-level queries will not return set-up data which is not currently in use. Note that not all a group's information will necessarily be sent out as a response.

### Header Interpretation Rules

This instrument interprets the header received according to the following rules.

- Mnemonics are not case sensitive.
  Example "CURSor" can also be written as "cursor" or "Cursor."
- The lower-case part of a header can be omitted.
  Example "CURSor" can also be written as "CURSO" or "CURS."
- If the header ends with a question mark, the command is a query. It is not possible to omit the question mark.
  Example "CURSor?" cannot be abbreviated to anything shorter than "CURS?."
- If the "x" at the end of a mnemonic is omitted, it is assumed to be "1."
  Example If "CHANnel<x>" is written as "CHAN," this represents "CHANnel1."
- Any part of a command enclosed by [ ] can be omitted.
  Example "TRIGger[:SIMPLle]:LEVel" can be written as "TRIG:LEV."
- However, a part enclosed by [ ] cannot be omitted if it is located at the end of an upper-level query.
  Example "TRIGger?" and "TRIGger:SIMPle?" belong to different upper-level query levels.

## 4.3 Response

On receiving a query from the controller, this instrument returns a response message to the controller. A response message is sent in one of the following two forms.

- Response consisting of a header and data
  If the query can be used as a program message without any change, a command header is attached to the query, which is then returned.
  Example `:ACQUire:MODE?<PMT>→:ACQUire:MODE NORMAL<RMT>`

- Response consisting of data only
  If the query cannot be used as a program message unless changes are made to it (i.e. it is a query-only command), no header is attached and only the data is returned. Some query-only commands can be returned after a header is attached to them.
  Example `:MEASure:CHANnel1:PTOPeak:VALue?<PMT>→10.0E+00<RMT>`

### When returning a response without a header

It is possible to remove the header from a response consisting of a header and data. The "`COMMunicate:HEADer`" command is used to do this.

### Abbreviated form

Normally, the lower-case part is removed from a response header before the response is returned to the controller. Naturally, the full form of the header can also be used. For this, the "`COMMunicate:VERBose`" command is used. The part enclosed by [ ] is also omitted in the abbreviated form.

## 4.4 Data

### Data

A data section comes after the header. A space must be included between the header and the data. The data contains conditions and values. Data is classified as below.

| Data | Description |
|---|---|
| <Decimal> | Value expressed as a decimal number<br>(Example: CH2's probe attenuation<br>→`CHANnel2:PROBe 100`) |
| <Voltage><Time><Frequency> | Physical value<br>(Example: Time axis range→`TIMebase:TDIV 1US`) |
| <Register> | Register value expressed as either binary, octal, decimal or hexadecimal<br>(Example: Extended event register value<br>→`STATus:EESE #HFE`) |
| <Character data> | Specified character string (mnemonic).<br>Can be selected from { }<br>(Example: CH2 input coupling<br>→`CHANnel2:COUPling {AC\|DC\|DC50\|GND}`) |
| <Boolean> | Indicates ON/OFF. Set to ON, OFF or value<br>(Example: CH2 display ON<br>→`CHANnel2:DISPlay ON`) |
| <Character string data> | Arbitrary character string<br>(Example: Comment on screen-data output<br>→`HCOPy:COMMent "ABCDEF"`) |
| <Filename> | Gives the name of a file.<br>(Example: Name of file to be saved<br>→`FILE:SAVE:WAVeform:NAME "CASE1"`) |
| <Block data> | Arbitrary 8-bit data<br>(Example: Response to acquired waveform data<br>→`#800000010ABCDEFGHIJ`) |

### <Decimal>

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

| Symbol | Description | Example |
|---|---|---|
| <NR1> | Integer | 125 –1 +1000 |
| <NR2> | Fixed point number | 125.0 –.90 +001. |
| <NR3> | Floating point number | 125.0E+0 –9E–1 +.1E4 |
| <NRf> | Any of the forms <NR1> to <NR3> is allowed. | |

- Decimal values which are sent from the controller to this instrument can be sent in any of the forms to <NR3>. In this case, <NRf> appears.
- For response messages which are returned from this instrument to the controller, the form (<NR1> to <NR3> to be used) is determined by the query. The same form is used, irrespective of whether the value is large or small.
- In the case of <NR3>, the "+" after the "E" can be omitted, but the "–" cannot.
- If a value outside the setting range is entered, the value will be normalized so that it is just inside the range.
- If the value has more than the significant number of digits, the value will be rounded.

### <Voltage>, <Time>, <Frequency>

<Voltage>, <Time>, and <Frequency> indicate decimal values which have physical significance. <Multiplier> or <Unit> can be attached to <NRf>. They can be entered in any of the following forms.

| Form | Example |
|---|---|
| <NRf><Multiplier><Unit> | 5MV |
| <NRf><Unit> | 5E-3V |
| <NRf><Multiplier> | 5M |
| <NRf> | 5E-3 |

### <Multiplier>

Multipliers which can be used are shown below.

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

### <Unit>

Units which can be used are shown below.

| Symbol | Word | Description |
|---|---|---|
| V | Volt | Voltage |
| S | Second | Time |
| HZ | Hertz | Frequency |
| MHZ | Megahertz | Frequency |

- <Multiplier> and <Unit> are not case sensitive.
- "U" is used to indicate "μ."
- "MA" is used for Mega (M) to distinguish it from Mili, except for in the case of Megahertz, which is expressed as "MHZ." Hence, it is not permissible to use "M" (Mili) for Hertz.
- If both <Multiplier> and <Unit> are omitted, the default unit will be used.
- Response messages are always expressed in <NR3> form. Neither <Multiplier> nor <Unit> is used, therefore the default unit is used.

### <Register>

<Register> indicates an integer, and can be expressed in hexadecimal, octal, or binary as well as as a decimal number. <Register> is used when each bit of a value has a particular meaning. <Register> is expressed in one of the following forms.

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

- <Register> is not case sensitive.
- Response messages are always expressed as <NR1>.

### <Character Data>

<Character data> is a specified string of character data (a mnemonic). It is mainly used to indicate options, and is chosen from the character strings given in { }. For interpretation rules, refer to "Header Interpretation Rules" on page 4-4.

| Form | Example |
|---|---|
| {AC\|DC\|DC50\|GND} | AC |

- As with a header, the "COMMunicate:VERBose" command can be used to return a response message in its full form. Alternatively, the abbreviated form can be used.
- The "COMMunicate:HEADer" command does not affect <character data>.

### <Boolean>

<Boolean> is data which indicates ON or OFF, and is expressed in one of the following forms.

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

- When <Boolean> is expressed in <NRf> form, OFF is selected if the rounded integer value is "0" and ON is selected if the rounded integer is "Not 0."
- A response message is always "1" if the value is ON and "0" if it is OFF.

### <Character String Data>

<Character string data> is not a specified character string like <Character data>. It is an arbitrary character string. A character string must be enclosed in single quotation marks (') or double quotation marks (").

| Form | Example |
|---|---|
| <Character string data> | 'ABC' "IEEE488.2-1987" |

- Response messages are always enclosed in double quotation marks.
- If a character string contains a double quotation mark ("), the double quotation mark will be replaced by two concatenated double quotation marks (""). This rule also applies to a single quotation mark within a character string.
- <Character string data> is an arbitrary character string, therefore this instrument assumes that the remaining program message units are part of the character string if no single (') or double quotation mark (") is encountered. As a result, no error will be detected if a quotation mark is omitted.

### <Filename>

Gives the name of a file. The format is as follows.

| Form | Example |
|---|---|
| {<NRf>\|<Character data>\|<Character string>} | 1 CASE "CASE" |

- If you input an <NRf> value, the system converts the value (after rounding to the nearest integer) to the corresponding 8-character ASCII string. (If you set the value to 1, the name becomes "00000001".) Note that negative values are not allowed.
- If you enter a <character data> or <character string> argument that is longer than eight characters, only the first eight characters are used.
- Response messages always return filenames as <character string> arguments.

### <Block data>

<Block data> is arbitrary 8-bit data. <Block data> is only used for response messages. Response messages are expressed in the following form.

| Form | Example |
|---|---|
| #N<N-digit decimal value><Data byte string> | #800000010ABCDEFGHIJ |

- #N
  Indicates that the data is <Block data>. "N" is an ASCII character string number (digits) which indicates the number of data bytes that follow.
- **<N-digits decimal value>**
  Indicates the number of bytes of data. (Example: 00000010=10 bytes)
- **<Data byte string>**
  The actual data. (Example: ABCDEFGHIJ)
- Data is comprised of 8-bit values (0 to 255). This means that the ASCII code "0AH," which stands for "NL," can also be a code used for data. Hence, care must be taken when programming the controller.

## 4.5 Synchronization with the Controller

### Overlap Commands and Sequential Commands

There are two kinds of command; overlap commands and sequential commands. Execution of an overlap command may start before execution of the previously sent command is completed.

The "CHANnel1:VDIV" command, for example, is a sequential command. Assume that you set a new V/div value and immediately request return of the new value, as follows:

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

In this case, the oscilloscope always returns the newest setting ("5V"). This is because it always completes processing of the current sequential command (in this case, "VDIV 5V") before moving on to the next command ("VDIV?").

In contrast, assume that you begin a file load and then immediately query the V/div value:

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

Because "FILE:LOAD:SETup:EXECute "CASE1"" is an overlapped command, the oscilloscope will advance to the "CHANNel1:VDIV?" command before it finishes the load. The returned V/div value will not show the newest setting, but will rather show the setting in use before the setup was changed.

Obviously, use of overlapped commands may in some cases produce inappropriate results. Where necessary, you can avoid such problems as described below.

### Synchronization with an Overlap Command

#### Using the *WAI command

The *WAI command causes the commands which follow it to wait until an overlap command has been executed.

Example

:COMMunicate:OPSE #0040;:FILE:LOAD:SETup:
EXECute "CASE1";*WAI;:CHANnel1:VDIV?<PMT>

The "COMMunicate:OPSE" command is used to designate which commands are to be subject to the *WAI command. In the above example, only auto set-up is designated.

Since a *WAI command is executed just before "CHANnel1:VDIV?," "CHANnel1:VDIV?" will not be executed until auto set-up has been completed.

#### Using the COMMunicate:OVERlap command

The "COMMunicate:OVERlap" command is used to enable or disable overlap operation.

Example

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

The "COMMunicate:OVERlap #HFFBF" command disables overlapped operation of the medium access command, while enabling all other overlap-type operations. The oscilloscope will therefore handle "FILE:LOAD:SETup:EXECute "CASE1"" as sequential command, ensuring that the "CHANnel1:VIDV?" command (in the above example) will not execute until file loading is completed.

#### Using the *OPC command

The *OPC command causes the OPC bit (bit 0) of the standard event register (page 6-3) to be set to "1" when an overlap operation has been completed.

Example

:COMMunicate:OPSE #H0040;*ESE 1;*ESR?;
*SRE 32;:FILE:LOAD:SETup:EXECute "CASE1";
*OPC<PMT>
(Response to *ESR? is decoded.)
(Service request is awaited.)
CHANnel1:VDIV?<PMT>

The "COMMunicate:OPSE" command is used to designate which commands are to be subject to the *OPC command. In the above example, only medium access commands are designated.

*ESE 1 and *SRE 32 stipulate that a service request is generated only when the OPC bit is set to "1."

*ESR? is used to clear the standard event register.

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

### Using the *OPC? query

The *OPC? query generates a response when an overlap operation has been completed.

Example

```
:COMMunicate:OPSE #H0040;:FILE:LOAD:SETup:
EXECute "CASE1";*OPC?<PMT>
```

(Response to *OPC? is decoded.)

```
:CHANnel1:VDIV?<PMT>
```

The "COMMunicate:OPSE" command is used to designate which commands are to be subject to the *OPC? command. In the above example, only medium access commands are designated.
Since *OPC? does not generate a response until an overlap operation is completed, file loading will have been completed when a response to *OPC? is read.

***Note***

Most commands are sequential commands. Commands used in Chapter 5 are sequential commands unless otherwise specified.

### Synchronization with Non-Overlap Commands

Synchronization is sometimes required for reasons other than communications-related reasons, such as the activation of a trigger, even if a sequential command is used.
For example, if a "next program" message is transmitted to make an inquiry about the waveform data which has been acquired using single mode as the trigger mode, the "WAVeform:SEND?" command is sometimes executed whether acquisition has been completed or not, causing a command execution error.

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

In this case, synchronization with the time at which acquisition is completed must be accomplished, as shown next.

### Using STATus:CONDition? query

A "STATus:CONDition?" query is used to make an query about the contents of the condition register (page 6-4). It is possible to judge whether acquisition is in progress or not by reading bit 0 of the condition register. Bit 0 is "1" if acquisition is in progress, and "0" if acquisition is stopped.

Example

```
:TRIGger:MODE SINGle;:STARt<PMT>
:STATus:CONDition?<PMT>
```

(Returns to the previous status if bit 0 is found to be "1" when the response is decoded.)

```
:WAVeform:SEND?<PMT>
```

A "WAVeform:SEND?" query will not be executed until bit 0 of the condition register has been set to "0."

### Using the extended event register

Changes in the condition register are reflected in the extended event register (page 6-4).

Example

```
:STATus:FILTer1 FALL;:STATus:EESE 1;EESR?;
*SRE 8;:TRIGger:MODE SINGle;:STARt<PMT>
```

(Response to "STATus:EESR?" is decoded.)
(Service request is awaited.)

```
:WAVeform:SEND?<PMT>
```

The "STATus:FILTer1 FALL" command sets the transition filter such that Bit 0 (FILTer1) of the Extended Event Register sets to 1 when Bit 0 of the Condition Register changes from 1 to 0.
"STATus:EESE 1" is a command used only to reflect the status of bit 0 of the extended event register in the status byte.
"STATus:EESR?" is used to clear the extended event register.
The *SRE command is used to generate a service request caused solely by the extended event register.
"WAVeform:SEND?" will not be executed until a service request is generated.

**Using the COMMunicate:WAIT command**

The "COMMunicate:WAIT" command halts communications until a specific event is generated.

Example

```
:STATus:FILTer1
FALL;:STATus:EESR?;:TRIGger:MODE SINGle;:
STARt<PMT>
```

(Response to "STATus:EESR?" is decoded.)

```
:COMMunicate:WAIT 1;:WAVeform:SEND?<PMT>
```

For a description of "STATus:FILTer1 FALL" and "STATus:EESR?," refer to "Using the extended event register" on this page.
"COMMunicate:WAIT 1" means that communications is halted until bit 0 of the extended event register is set to "1."

# Chapter 5 Commands

## 5.1 A List of Commands

| Command | Function | Page |
|---|---|---|
| **ACQuire Group** | | |
| :ACQuire? | Queries all settings related to waveform acquisition. | 5-15 |
| :ACQuire:AVERage? | Queries all settings related to averaging. | 5-15 |
| :ACQuire:AVERage:COUNt | Sets the number of waveform acquisitions during averaging or queries the current setting. | 5-15 |
| :ACQuire:AVERage:EWEight | Sets the attenuation constant for exponential averaging or queries the current setting. | 5-16 |
| :ACQuire:CLOCk | Sets the time base or queries the current setting. | 5-16 |
| :ACQuire:COUNt | Sets the number of waveform acquisitions during normal mode, average mode, and box average or queries the current setting. | 5-16 |
| :ACQuire:INTerleave | Turns the interleave mode ON/OFF or queries the current setting. | 5-16 |
| :ACQuire:MODE | Sets the waveform acquisition mode or queries the current setting. | 5-16 |
| :ACQuire:REPetitive | Turns the repetitive sampling mode ON/OFF or queries the current setting. | 5-16 |
| :ACQuire:RLENgth | Sets the record length or queries the current setting. | 5-16 |
| **ASETup Group** | | |
| :ASETup:EXECute | Executes auto setup. | 5-16 |
| :ASETup:UNDO | Cancels the previous auto setup operation. | 5-16 |
| **CALibrate Group** | | |
| :CALibrate? | Queries all settings related to calibration. | 5-17 |
| :CALibrate:DESKew? | Queries all settings related to skew compensation. | 5-17 |
| :CALibrate:DESKew:MODE | Turns the skew compensation ON/OFF or queries the current setting. | 5-17 |
| :CALibrate:DESKew:TARGet | Sets the channel on which skew compensation is to be performed or queries the current setting. | 5-17 |
| :CALibrate:DESKew:TIME | Sets the skew adjustment time that are common to all channels or queries the current setting. | 5-17 |
| :CALibrate[:EXECute] | Executes the calibration. | 5-17 |
| :CALibrate:MODE | Turns auto calibration ON/OFF or queries the current setting. | 5-17 |
| **CHANnel Group** | | |
| :CHANnel<x>? | Queries all settings related to the vertical axis of each channel. | 5-19 |
| :CHANnel<x>:BWIDth | Sets the input filter of each channel or queries the current setting. | 5-19 |
| :CHANnel<x>:COUPling | Sets the input coupling of each channel or queries the current setting. | 5-19 |
| :CHANnel<x>:DISPlay | Turns the display of each channel ON/OFF or queries the current setting. | 5-19 |
| :CHANnel<x>:LABel | Sets a waveform label on each channel or queries the current setting. | 5-19 |
| :CHANnel<x>:LSCale? | Queries all settings related to the linear scaling of each channel. | 5-19 |
| :CHANnel<x>:LSCale:AVALue | Sets scaling constant A or queries the current setting. | 5-19 |
| :CHANnel<x>:LSCale:BVALue | Sets offset value B or queries the current setting. | 5-19 |
| :CHANnel<x>:LSCale:MODE | Turns linear scaling ON/OFF or queries the current setting. | 5-19 |
| :CHANnel<x>:LSCale:UNIT | Sets the measurement unit to be added to the result of the linear scaling or queries the current setting. | 5-19 |
| :CHANnel<x>:MODE | Turns the display of each channel ON/OFF or queries the current setting. | 5-20 |
| :CHANnel<x>:OFFSet | Sets the offset voltage of each channel or queries the current setting. | 5-20 |
| :CHANnel<x>:POSition | Sets the vertical position of each channel or queries the current setting. | 5-20 |
| :CHANnel<x>:PROBe | Sets the probe attenuation of each channel or queries the current setting. | 5-20 |
| :CHANnel<x>:VARiable | Sets the voltage sensitivity in units of 0.01 of the V/div setting for each channel or queries the current setting. | 5-20 |
| :CHANnel<x>:VDIV | Sets the V/div value of each channel or queries the current setting. | 5-20 |
| **CLEar Group** | | |
| :CLEar | Clears the trace. | 5-20 |

| Command | Function | Page |
|---|---|---|
| **COMMunicate Group** | | |
| :COMMunicate? | Queries all settings related to communications. | 5-21 |
| :COMMunicate:HEADer | Sets whether or not to add a header to the response data or queries the current setting. | 5-21 |
| :COMMunicate:LOCKout | Sets or clears the local lockout. | 5-21 |
| :COMMunicate:OPSE | Sets the overlap commands that are subject to *OPC, *OPC?, and *WAI or queries the current setting. | 5-22 |
| :COMMunicate:OPSR? | Queries the operation pending status register value. | 5-22 |
| :COMMunicate:OVERlap | Sets the commands that will operate as overlap commands or queries the current setting. | 5-22 |
| :COMMunicate:REMote | Switches to remote/local mode. | 5-22 |
| :COMMunicate:STATus? | Queries the line status. | 5-22 |
| :COMMunicate:VERBose | Sets the response messages to full form or abbreviated form or queries the current setting. | 5-22 |
| :COMMunicate:WAIT | Waits for a specified extended event. | 5-22 |
| :COMMunicate:WAIT? | Creates a response that is returned when the specified event occurs. | 5-23 |
| **CURSor Group** | | |
| :CURSor? | Queries all settings related to cursor measurements. | 5-27 |
| :CURSor:TY? | Queries all settings related to the cursor in the T-Y display. | 5-27 |
| :CURSor:TY:DEGRee? | Queries all settings related to angle cursors. | 5-27 |
| :CURSor:TY:DEGRee:DX? | Queries the angle difference, ΔD, between angle cursors. | 5-27 |
| :CURSor:TY:DEGRee:DY? | Queries the angle difference, ΔV, between angle cursors. | 5-27 |
| :CURSor:TY:DEGRee:JUMP | Moves the angle cursor to the zoomed waveform. | 5-27 |
| :CURSor:TY:DEGRee:POSition<x> | Sets the position of the angle cursor or queries the current setting. | 5-27 |
| :CURSor:TY:DEGRee:REFerence<x> | Sets the position of the angle reference start point (Reference1) or the end point (Reference2) or queries the current setting. | 5-27 |
| :CURsor:TY:DEGRee:RVALue | Sets the angle reference width or queries the current setting. | 5-28 |
| :CURSor:TY:DEGRee:TRACe | Sets the waveform on which to use the angle cursor or queries the current setting. | 5-28 |
| :CURSor:TY:DEGRee:X<x>? | Queries the angle at the angle cursor. | 5-28 |
| :CURSor:TY:DEGRee:Y<x>? | Queries the voltage value at the angle cursor. | 5-28 |
| :CURSor:TY:HORizontal? | Queries all settings related to H cursors. | 5-28 |
| :CURSor:TY:HORizontal:DY? | Queries the Y-axis value between the H cursors. | 5-28 |
| :CURSor:TY:HORizontal:POSition<x> | Sets the H cursor position or queries the current setting. | 5-28 |
| :CURSor:TY:HORizontal:TRACe | Sets the waveform on which to use the H cursor or queries the current setting. | 5-28 |
| :CURSor:TY:HORizontal:Y<x>? | Queries the Y-axis value of the H cursor. | 5-28 |
| :CURSor:TY:LOGic? | Sets/queries all settings related to the cursor in the logic T-Y display. | 5-28 |
| :CURSor:TY:LOGic:FORMat | Sets/queries the data format in the logic T-Y display. | 5-29 |
| :CURSor:TY:LOGic:ORDer | Queries the bit order of logic in the T-Y display. | 5-29 |
| :CURSor:TY:MARKer? | Queries all settings related to marker cursors. | 5-29 |
| :CURSor:TY:MARKer:DX<x>? | Queries the X-axis value between the marker cursors. | 5-29 |
| :CURSor:TY:MARKer:DY<x>? | Queries the physical value of the Y-axis difference between the marker cursors. | 5-29 |
| :CURSor:TY:MARKer:M<x>? | Queries all settings related to the specified marker cursor. | 5-29 |
| :CURSor:TY:MARKer:M<x>:JUMP | Moves the marker cursor to the zoomed waveform. | 5-29 |
| :CURSor:TY:MARKer:M<x>:POSition<x> | Sets the X-axis position of the marker cursor or queries the current setting. | 5-29 |
| :CURSor:TY:MARKer:M<x>:TRACe | Sets the waveform on which to use the marker cursor or queries the current setting. | 5-29 |
| :CURSor:TY:MARKer:X<x>? | Queries the X-axis position of the marker cursor. | 5-30 |
| :CURSor:TY:MARKer:Y<x>? | Queries the Y-axis position of the marker cursor. | 5-30 |
| :CURSor:TY:TYPE | Sets the cursor type or queries the current setting. | 5-30 |
| :CURSor:TY:VERTical? | Queries all settings related to V cursors. | 5-30 |
| :CURSor:TY:VERTical:DX? | Queries the X-axis value between V cursors. | 5-30 |
| :CURSor:TY:VERTical:DY? | Queries the Y-axis value between V cursors. | 5-30 |
| :CURSor:TY:VERTical:JUMP | Moves the V cursor to the zoomed waveform. | 5-30 |
| :CURSor:TY:VERTical:PERDt? | Queries the 1/ΔT value between V cursors. | 5-30 |
| :CURSor:TY:VERTical:POSition<x> | Sets the V cursor position or queries the current setting. | 5-31 |
| :CURSor:TY:VERTical:TRACe | Sets the waveform on which to use the V cursor or queries the current setting. | 5-31 |

| Command | Function | Page |
|---|---|---|
| :CURSor:TY:VERTical:X<x>? | Queries the X-axis position of the V cursor. | 5-31 |
| :CURSor:TY:VERTical:Y<x>? | Queries the Y-axis position of the V cursor. | 5-31 |
| :CURSor:XY? | Queries all settings related to the cursors in the X-Y display. | 5-31 |
| :CURSor:XY:HORizontal? | Queries all settings related to the H cursors in the X-Y display. | 5-31 |
| :CURSor:XY:HORizontal:DY? | Queries the Y-axis value of the H cursor of the X-Y display. | 5-31 |
| :CURSor:XY:HORizontal:POSition<x> | Sets the H cursor position of the X-Y display or queries the current setting. | 5-31 |
| :CURSor:XY:HORizontal:TRACe | Sets the waveform on which to use the H cursor in the X-Y display or queries the current setting. | 5-31 |
| :CURSor:XY:HORizontal:Y? | Queries the Y-axis value of the H cursor of the X-Y display. | 5-32 |
| :CURSor:XY:MARKer? | Queries all settings related to the marker cursors in the X-Y display. | 5-32 |
| :CURSor:XY:MARKer:POSition | Sets the X-axis position of the marker cursor in the X-Y display or queries the current setting. | 5-32 |
| :CURSor:XY:MARKer:T? | Queries the time value of the marker cursor in the X-Y display. | 5-32 |
| :CURSor:XY:MARKer:TRACe | Sets the waveform on which to use the marker cursor in the X-Y display or queries the current setting. | 5-32 |
| :CURSor:XY:MARKer:X? | Queries the X-axis position of the marker cursor in the X-Y display. | 5-32 |
| :CURSor:XY:MARKer:Y? | Queries the Y-axis position of the marker cursor in the X-Y display. | 5-32 |
| :CURSor:XY:TYPE | Sets the cursor type of the X-Y display or queries the current setting. | 5-32 |
| :CURSor:XY:VERTical? | Queries all settings related to V cursors in the X-Y display. | 5-32 |
| :CURSor:XY:VERTical:DX? | Queries the X-axis value between the V cursors in the X-Y display. | 5-32 |
| :CURSor:XY:VERTical:POSition<x> | Sets the V cursor position of the X-Y display or queries the current setting. | 5-32 |
| :CURSor:XY:VERTical:TRACe | Sets the waveform on which to use the V cursor in the X-Y display or queries the current setting. | 5-32 |
| :CURSor:XY:VERTical:X<x>? | Queries the X-axis position of the V cursor in the X-Y display. | 5-33 |
| **DISPlay Group** | | |
| :DISPlay? | Queries all settings related to the display. | 5-36 |
| :DISPlay:ACCumulate? | Queries all settings related to the accumulation display. | 5-36 |
| :DISPlay:ACCumulate:COLor | Sets the color grade width or queries the current setting. | 5-36 |
| :DISPlay:ACCumulate:MODE | Sets the accumulation mode or queries the current setting. | 5-36 |
| :DISPlay:ACCumulate:PERSistence | Sets the accumulation time or queries the current setting. | 5-36 |
| :DISPlay:COLor | Sets whether to change the screen color using RGB or brightness or queries the current setting. | 5-36 |
| :DISPlay:RGB? | Queries all settings related to the screen color. | 5-36 |
| :DISPlay:RGB:MENU? | Queries all settings related to colors used in the menu. | 5-36 |
| :DISPlay:RGB:MENU:{BACK\|DIALog\|FOCus\|FORE\|LINE\|SELect\|SHADow} | Sets the color of the characters, background, dialog, and lines of the menu screen or queries the current setting. | 5-37 |
| :DISPlay:RGB:OTHers? | Queries all settings related to the colors of other items. | 5-37 |
| :DISPlay:RGB:OTHers:{CURSor\|GRID\|MARKer\|SNAP\|ZONE<x>} | Sets the color for cursors, grid, markers, and snapshot waveforms or queries the current setting. | 5-37 |
| :DISPlay:RGB:WAVeform? | Queries all settings related to the waveform colors. | 5-37 |
| :DISPlay:RGB:WAVeform:{CHANnel<x>\|BACK\|MATH<x>\|PODA\|PODB} | Sets the colors for waveform and computation channels or queries the current setting. | 5-37 |
| :DISPlay:FORMat | Sets the display format or queries the current setting. | 5-37 |
| :DISPlay:GRATicule | Sets the graticule type or queries the current setting. | 5-37 |
| :DISPlay:INTENsity? | Queries all settings related to the brightness of the displayed items. | 5-37 |
| :DISPlay:INTENsity:{CURSor\|MENU\|GRID\|SNAP\|WAVeform} | Sets the brightness of each display item or queries the current setting. | 5-37 |
| :DISPlay:INTERpolate | Sets the interpolation method or queries the current setting. | 5-38 |
| :DISPlay:MAPPing | Sets the mapping mode used to allocate the waveforms to the divided windows or queries the current setting. | 5-38 |
| :DISPlay:SMAPping? | Queries the mapping of all waveforms to the divided windows. | 5-38 |
| :DISPlay:SMAPping:{CHANnel<x>\|MATH<x>} | Maps waveforms to the divided windows or queries the current setting. | 5-38 |

| Command | Function | Page |
|---|---|---|
| :DISPlay:SVALue | Turns the scale value display ON/OFF or queries the current setting. | 5-38 |
| :DISPlay:TLABel | Turns the display of waveform labels ON/OFF or queries the current setting. | 5-38 |
| :DISPlay:TRANslucent | Turns the popup menu's transparent mode ON/OFF or queries the current setting. | 5-38 |
| **FILE Group** | | |
| :FILE? | Queries all settings related to the specified medium. | 5-42 |
| :FILE:DELete:{ASCii\|BINary\|BMP\|FLOat\|MEASure\|POSTscript\|SETup\|SNAP\|TIFF}:[EXECute] | Deletes various types of data files. | 5-42 |
| :FILE:DIRectory? | Queries all settings related to the current medium. | 5-42 |
| :FILE[:DIRectory]:CDIRectory | Changes the current directory. | 5-42 |
| :FILE[:DIRectory]:DRIVe | Selects the current medium. | 5-42 |
| :FILE[:DIRectory]:FILTer | Sets a filter or queries the current setting. | 5-42 |
| :FILE[:DIRectory]:FREE? | Queries the free space (number of bytes) on the current medium. | 5-42 |
| :FILE[:DIRectory]:MDIRectory | Creates a directory in the current directory. | 5-42 |
| :FILE[:DIRectory]:PATH? | Queries the current path. | 5-42 |
| :FILE:FORMat? | Queries all settings related to the formatting of the floppy disk. | 5-42 |
| :FILE:FORMat:EXECute | Formats the floppy disk. | 5-43 |
| :FILE:FORMat:TYPE | Sets the format type for the floppy disk or queries the current setting. | 5-43 |
| :FILE:LOAD? | Queries all settings related to file loading. | 5-43 |
| :FILE:LOAD:{BINary\|SETup\|SNAP}:ABORt | Aborts loading the data. | 5-43 |
| :FILE:LOAD:{BINary\|SETup\|SNAP}[:EXECute] | Loads the data. | 5-43 |
| :FILE:LOAD:PLANe | Sets the plane number of the snapshot waveform data or queries the current setting. | 5-43 |
| :FILE:LOAD:SNAP? | Queries all settings related to the loading of the snapshot waveform data. | 5-43 |
| :FILE:SAVE? | Queries all settings related to file saving. | 5-43 |
| :FILE:SAVE:ANAMing | Turns the auto naming function of files ON/OFF or queries the current setting. | 5-43 |
| :FILE:SAVE:{ASCii\|BINary\|FLOat}? | Queries all settings related to various types of data. | 5-43 |
| :FILE:SAVE:{ASCii\|BINary\|FLOat\|MEASure\|SETup\|SNAP}:ABORt | Aborts saving the data. | 5-43 |
| :FILE:SAVE:{ASCii\|BINary\|FLOat\|MEASure\|SETup\|SNAP}[:EXECute] | Saves the file. | 5-43 |
| :FILE:SAVE:{ASCii\|BINary\|FLOat}:HISTory | Sets the number of histories to save or queries the current setting. | 5-43 |
| :FILE:SAVE:{ASCii\|BINary\|FLOat}:RANGe | Sets the range of data to be saved or queries the current setting. | 5-44 |
| :FILE:SAVE:{ASCii\|BINary\|FLOat}:TRACe | Sets the waveform to be saved or queries the current setting. | 5-44 |
| :FILE:SAVE:BINary:COMPression | Sets data compression on the waveform to be saved or queries the current setting. | 5-44 |
| :FILE:SAVE:COMMent | Sets comments or queries the current setting. | 5-44 |
| :FILE:SAVE:NAME | Sets the name of the file to be saved or queries the current setting. | 5-44 |
| :FILE:UNLoad? | Queries all settings related to canceling the loaded data. | 5-44 |
| :FILE:UNLoad:{BINary\|SNAP}? | Queries all settings related to canceling of loaded data. | 5-44 |
| :FILE:UNLoad:{BINary\|SNAP}[:EXECute] | Clears the loaded data. | 5-44 |
| :FILE:UNLoad:BINary:TRACe | Sets the waveform to be cleared from the loaded data or queries the current setting. | 5-44 |
| :FILE:UNLoad:SNAP:PLANe | Sets the plane number to be cleared from the loaded data or queries the current setting. | 5-44 |
| **GONogo Group** | | |
| :GONogo? | Queries all settings related to GO/NO-GO determination. | 5-46 |
| :GONogo:ABORt | Aborts GO/NO-GO determination. | 5-46 |
| :GONogo:ACTion? | Queries all settings related to the action to be performed when the result of the determination is NO-GO and also the determination criteria. | 5-46 |
| :GONogo:ACTion:BUZZer | Sets whether or not to sound an alarm when the determination is NO-GO or queries the current setting. | 5-46 |
| :GONogo:ACTion:HCOPy | Sets whether or not to make a hard copy to the built-in printer (option) when the determination results in NO-GO or queries the current setting. | 5-47 |
| :GONogo:ACTion:MAIL? | Queries all setting values for mail reports sent when NO-GO is determined. | 5-47 |

| Command | Function | Page |
|---|---|---|
| :GONogo:ACTion:MAIL:COUNt | Sets/queries the maximum number of e-mail is to be sent when the determination is NO-GO. | 5-47 |
| :GONogo:ACTion:MAIL:MODE | Sets/queries whether and e-mail report is sent when the determination is NO-GO. | 5-47 |
| :GONogo:ACTion:SAVE | Sets whether or not to save the NO-GO result to the medium or queries the current setting. | 5-47 |
| :GONogo:COUNt? | Queries the GO/NO-GO determination count. | 5-47 |
| :GONogo:EXECute | Executes GO/NO-GO determination. | 5-47 |
| :GONogo:LOGic | Sets the GO/NO-GO determination logic or queries the current setting. | 5-47 |
| :GONogo:MODE | Sets the GO/NO-GO determination type or queries the current setting. | 5-47 |
| :GONogo:NGCount? | Queries the number of NO-GO results. | 5-47 |
| :GONogo:PARameter? | Queries all settings related to the parameter determination. | 5-47 |
| :GONogo:PARameter:ITEM<x>? | Queries all settings related to waveform parameters. | 5-48 |
| :GONogo:PARameter:ITEM<x>:CAUSe? | Queries whether or not the waveform parameter is the cause of the NO-GO result. | 5-48 |
| :GONogo:PARameter:ITEM<x>:CONDition | Sets the criteria for each waveform parameter or queries the current setting. | 5-48 |
| :GONogo:PARameter:ITEM<x>:MODE | Turns the waveform parameter ON/OFF or queries the current setting. | 5-48 |
| :GONogo:PARameter:ITEM<x>:TRACe | Sets the waveform on which to use the waveform parameter or queries the current setting. | 5-48 |
| :GONogo:PARameter:ITEM<x>:TYPE? | Queries the waveform parameter and the upper and lower limits. | 5-48 |
| :GONogo:PARameter:ITEM<x>:TYPE:<parameter> | Sets the upper and lower limits of each waveform parameter or queries the current setting. | 5-48 |
| :GONogo:PARameter:ITEM<x>:VALue? | Queries the measured value of each waveform parameter. | 5-48 |
| :GONogo:SEQuence | Selects single or continuous action or queries the current setting. | 5-49 |
| :GONogo:STOPcount | Sets the number of waveform acquisitions on which to perform GO/NO-GO determination or queries the current setting. | 5-49 |
| :GONogo:TRANge | Sets the range over which to perform GO/NO-GO determination or queries the current setting. | 5-49 |
| :GONogo:WAIT? | Waits for the GO/NO-GO determination to complete with a timeout. | 5-49 |
| :GONogo:ZONE? | Queries all settings related to zone determination. | 5-49 |
| :GONogo:ZONE:EDIT<x>:EXIT | Exits from the edit menu of the zone for zone determination. | 5-49 |
| :GONogo:ZONE:EDIT<x>:PART | Edits a portion of the zone for zone determination. | 5-49 |
| :GONogo:ZONE:EDIT<x>:WHOLe | Edits the zone for zone determination. | 5-49 |
| :GONogo:ZONE:ZONE<x>? | Queries all settings related to zone determination of the specified zone. | 5-50 |
| :GONogo:ZONE:ZONE<x>:CAUSe? | Queries whether or not the specified zone for zone determination is the cause of the NO-GO result. | 5-50 |
| :GONogo:ZONE:ZONE<x>:CONDition | Sets the criteria for the zone for zone determination or queries the current setting. | 5-50 |
| :GONogo:ZONE:ZONE<x>:MODE | Turns ON/OFF the determination of the zone or queries the current setting. | 5-50 |
| :GONogo:ZONE:ZONE<x>:SEND | Sets the zone data specified for the determination zone or queries the current setting. | 5-50 |
| **HCOPy Group** | | |
| :HCOPy? | Queries all settings related to the hard copy of the screen data. | 5-53 |
| :HCOPy:ABORt | Aborts data output or paper feeding. | 5-53 |
| :HCOPy:{BMP\|TIFF}? | Queries all settings related to the BMP or TIFF format. | 5-53 |
| :HCOPy:BMP:COMPression | Enables compression on BMP format image data or queries the current setting. | 5-53 |
| :HCOPy:{BMP\|TIFF}:TONE | Sets the number of colors to use for the BMP or TIFF format or queries the current setting. | 5-53 |
| :HCOPy:CENTronics? | Queries all settings related to the output to the external printer. | 5-53 |
| :HCOPy:CENTronics:TONE | Sets the half tone on the external printer output or queries the current setting. | 5-53 |
| :HCOPy:CENTronics:TYPE | Sets the type of output commands to send to the external printer or queries the current setting. | 5-53 |
| :HCOPy:COMMent | Sets comments at the upper section of the screen or queries the current setting. | 5-54 |
| :HCOPy:DIRection | Sets the data output destination or queries the current setting. | 5-54 |
| :HCOPy:EXECute | Executes data output. | 5-54 |

| Command | Function | Page |
|---|---|---|
| :HCOPy:FORMat | Sets the format of the image that is to be saved to the file or queries the current setting. | 5-54 |
| :HCOPy:NETPrint? | Queries all setting values involving output to network printers. | 5-54 |
| :HCOPy:NETPrint:TONE | Sets/queries halftones for output to network printers. | 5-54 |
| :HCOPy:NETPrint:TYPE | Sets/queries command types for output to network printers. | 5-54 |
| :HCOPy:PRINter? | Queries all settings related to the output to the built-in printer. | 5-54 |
| :HCOPy:PRINter:MAG | Sets the zoom rate to use during long copy or queries the current setting. | 5-54 |
| :HCOPy:PRINter:RANGe | Sets the output range during long copy or queries the current setting. | 5-54 |
| :HCOPy:PRINter:REPort | Sets whether or not to output additional information to the built-in printer or queries the current setting. | 5-54 |
| :HCOPy:PRINter:TYPE | Selects short copy or long copy or queries the current setting. | 5-55 |
| :HCOPy:SAVE? | Queries all settings related to file output. | 5-55 |
| :HCOPy:SAVE:ANAMing | Turns the auto naming function of file names ON/OFF or queries the current setting. | 5-55 |
| :HCOPy:SAVE:CDIRectory | Changes the directory in which to create the file. | 5-55 |
| :HCOPy:SAVE:DRIVe | Selects the medium on which to create the file. | 5-55 |
| :HCOPy:SAVE:NAME | Sets the name of the file to be created or queries the current setting. | 5-55 |
| **HISTory Group** | | |
| :HISTory? | Queries all settings related to the history memory function. | 5-57 |
| :HISTory:ABORt | Aborts the search. | 5-57 |
| :HISTory:DISPlay | Sets the start/end number of the display record or queries the current setting. | 5-57 |
| :HISTory:EXECute | Executes zone search. | 5-57 |
| :HISTory:SMODe | Sets the history memory's search mode or queries the current setting. | 5-57 |
| :HISTory:DMODe | Sets whether to display one record or all records or queries the current setting. | 5-57 |
| :HISTory:PARameter? | Queries all settings related to the parameter search. | 5-58 |
| :HISTory:PARameter:ITEM<x>? | Queries all settings related to the specified parameter of the parameter search. | 5-58 |
| :HISTory:PARameter:ITEM<x>:CONDition | Sets the criteria for the specified parameter of the parameter search or queries the current setting. | 5-58 |
| :HISTory:PARameter:ITEM<x>:SOURce | Sets the trace on which to perform the parameter search of the specified parameter or queries the current setting. | 5-58 |
| :HISTory:PARameter:ITEM<x>:TYPE:<parameter> | Sets the upper and lower limits for the specified parameter of the parameter search or queries the current setting. | 5-58 |
| :HISTory:PARameter:ITEM<x>:VALue? | Queries the measured value of the specified parameter of the parameter search or queries the current setting. | 5-58 |
| :HISTory:PARameter:LOGic | Sets the determination logic of the parameter search or queries the current setting. | 5-58 |
| :HISTory:PARameter:TRANge | Sets the range over which to perform the parameter search or queries the current setting. | 5-58 |
| :HISTory:RECord | Sets the target record or queries the current setting. | 5-59 |
| :HISTory:RECord? MINimum | Queries the minimum record number. | 5-59 |
| :HISTory:TIME? | Queries the time of the specified record number. | 5-59 |
| :HISTory:ZONE? | Queries all settings related to zone search. | 5-59 |
| :HISTory:ZONE:EDIT<x> | Queries all settings related to all zone cursors. | 5-59 |
| :HISTory:ZONE:EDIT<x>:CONDition | Sets the search condition of the zone cursor or queries the current setting. | 5-59 |
| :HISTory:ZONE:EDIT<x>:HORizontal | Sets the vertical position of the zone cursor or queries the current setting. | 5-59 |
| :HISTory:ZONE:EDIT<x>:SOURce | Sets the waveform on which to use the zone cursor or queries the current setting. | 5-59 |
| :HISTory:ZONE:EDIT<x>:VERTical | Sets the horizontal position of the zone cursor or queries the current setting. | 5-60 |
| :HISTory:ZONE:LOGic | Sets the search logic between zone cursors or queries the current setting. | 5-60 |
| **IMAGe Group** | | |
| :IMAGe? | Queries all settings related to the screen image data output. | 5-61 |
| :IMAGe:FORMat | Sets the output format of the screen image or queries the current setting. | 5-61 |
| :IMAGe:SEND? | Queries the screen image data. | 5-61 |
| :IMAGe:TONE | Sets the color tone of the output screen image data or queries the current setting. | 5-61 |

| Command | Function | Page |
|---|---|---|
| **INITialize Group** | | |
| :INITialize:EXECute | Executes initialization. | 5-61 |
| :INITialize:UNDO | Cancels the initialization that was executed. | 5-61 |
| **LOGic Group (option)** | | |
| :LOGic? | Queries all settings related to the logic probe. | 5-64 |
| :LOGic:MODE | Turns the logic mode ON/OFF or queries the current setting. | 5-64 |
| :LOGic:{PODA\|PODB}? | Queries all settings related to the specified logic probe. | 5-64 |
| :LOGic:{PODA\|PODB}:ALL:DISPlay | Turns the display of all the bits of the specified logic probe ON/OFF. | 5-64 |
| :LOGic:{PODA\|PODB}:{BIT0\|BIT1\|BIT2\|BIT3\|BIT4\|BIT5\|BIT6\|BIT7}:DISPlay | Displays the bits of the specified logic probe or queries the current setting. | 5-64 |
| :LOGic:{PODA\|PODB}:{BIT0\|BIT1\|BIT2\|BIT3\|BIT4\|BIT5\|BIT6\|BIT7}:LABel | Sets a label to the bits of the specified logic probe or queries the current setting. | 5-64 |
| :LOGic:{PODA\|PODB}:THReshold? | Queries all settings related to the threshold level. | 5-64 |
| :LOGic:{PODA\|PODB}:THReshold:TYPE | Executes the threshold level preset or queries the current setting. | 5-65 |
| :LOGic:{PODA\|PODB}:THReshold:USERlevel | Sets the user-defined threshold level or queries the current setting. | 5-65 |
| **MATH Group** | | |
| :MATH<x>? | Queries all settings related to computations. | 5-67 |
| :MATH<x>:BINary? | Queries all settings related to binary computations. | 5-67 |
| :MATH<x>:BINary:{CHANnel<x>\|MATH<x>} | Sets the threshold level for each channel of the binary computation or queries the current setting. | 5-67 |
| :MATH<x>:DISPlay | Sets whether or not to display computation waveforms or queries the current setting. | 5-67 |
| :MATH<x>:FFT? | Queries all settings related to the FFT. | 5-67 |
| :MATH<x>:FFT:POINt | Sets the number of points for the FFT or queries the current setting. | 5-67 |
| :MATH<x>:FFT:SPOint | Sets the start point of the FFT or queries the current setting. | 5-67 |
| :MATH<x>:FFT:WINDow | Sets the window function of the FFT or queries the current setting. | 5-67 |
| :MATH<x>:INTegral? | Queries all settings related to integration. | 5-67 |
| :MATH<x>:INTegral:SPOint | Sets the start point of the integration or queries the current setting. | 5-68 |
| :MATH<x>:LABel | Sets/queries the math's label. | 5-68 |
| :MATH<x>:OPERation | Sets the type of computation or queries the current setting. | 5-68 |
| :MATH<x>:SCALe? | Queries all settings related to scaling. | 5-68 |
| :MATH<x>:SCALe:MODE | Sets the scaling method or queries the current setting. | 5-68 |
| :MATH<x>:SCALe:VALue | Sets the upper and lower limits during manual scaling or queries the current setting. | 5-68 |
| :MATH<x>:SMOothing | Turns smoothing ON/OFF or queries the current setting. | 5-68 |
| :MATH<x>:UNIT | Sets the unit to be added to the computed result or queries the current setting. | 5-68 |
| **MEASure Group** | | |
| :MEASure? | Queries all settings related to the automated measurement of waveform parameters. | 5-71 |
| :MEASure:{CHANnel<x>\|MATH<x>}? | Queries the ON/OFF state of every parameter of the waveform. | 5-73 |
| :MEASure:{CHANnel<x>\|MATH<x>}:ALL | Turns all the measurement parameters of the waveform ON/OFF at once. | 5-73 |
| :MEASure:{CHANnel<x>\|MATH<x>}:COPY | Copies the ON/OFF state of all measurement parameters of one waveform to all other waveforms. | 5-73 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay? | Queries all settings related to the delay between waveform channels. | 5-73 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:MEASure? | Queries all settings related to the measured waveform when measuring the delay between channels. | 5-73 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:MEASure:COUNt | Sets the edge count of the measured waveform when measuring the delay between channels or queries the current setting. | 5-73 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:MEASure:SLOPe | Sets the slope of the measured waveform when measuring the delay between channels or queries the current setting. | 5-74 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:REFerence? | Queries all settings related to the reference waveform when measuring the delay between channels. | 5-74 |

| Command | Function | Page |
|---|---|---|
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:REFerence:COUNt | Sets the edge count of the reference waveform when measuring the delay between channels or queries the current setting. | 5-74 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:REFerence:SLOPe | Sets the slope of the reference waveform when measuring the delay between channels or queries the current setting. | 5-74 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:REFerence:SOURce | Sets whether or not to set the reference to the trigger point or the waveform when measuring the delay between channels or queries the current setting. | 5-74 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:REFerence:TRACe | Sets the reference trace when measuring the delay between channels or queries the current setting. | 5-74 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DELay:STATe | Turns the delay parameter ON/OFF and sets the display format or queries the current setting. | 5-75 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DPRoximal? | Queries all settings related to distal, mesial, and proximal values. | 5-75 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DPRoximal:MODE | Sets the distal, mesial, and proximal mode or queries the current setting. | 5-75 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DPRoximal:PERCent | Sets the distal, mesial, and proximal points or queries the current setting. (%) | 5-75 |
| :MEASure:{CHANnel<x>\|MATH<x>}:DPRoximal:UNIT | Sets the distal, mesial, and proximal points or queries the current setting. (voltage) | 5-75 |
| :MEASure:{CHANnel<x>\|MATH<x>}:METHod | Sets the High and Low points or queries the current setting. | 5-75 |
| :MEASure:{CHANnel<x>\|MATH<x>}:<parameter>? | Queries all settings related to the parameters of each waveform. | 5-76 |
| :MEASure:{CHANnel<x>\|MATH<x>}:<parameter>:COUNt? | Queries the number of statistical processing of each parameter. | 5-76 |
| :MEASure:{CHANnel<x>\|MATH<x>}:<parameter>:{MAXimum\|MEAN\|MINimum\|SDEViation}? | Queries the statistics of each parameter. | 5-76 |
| :MEASure:{CHANnel<x>\|MATH<x>}:<parameter>:STATe | Turns the parameter ON/OFF or queries the current setting. (one by one) | 5-76 |
| :MEASure:{CHANnel<x>\|MATH<x>}:<parameter>:VALue? | Queries the value of the automated measurement of each parameter. | 5-77 |
| :MEASure:CYCLe? | Queries all settings related to the periodic statistical processing of the automated measurement of waveform parameters. | 5-77 |
| :MEASure:CYCLe:ABORt | Aborts the periodic statistical processing of the automated measurement of waveform parameters. | 5-77 |
| :MEASure:CYCLe:EXECute | Executes the periodic statistical processing of the automated measurement of waveform parameters. | 5-77 |
| :MEASure:CYCLe:TRACe | Sets the waveform used to determine the period for the periodic statistical processing of automated measurement of waveform parameters or queries the current setting. | 5-77 |
| :MEASure:HISTory:ABORt | Aborts the statistical processing within the historical data of the automated measurement of waveform parameters. | 5-77 |
| :MEASure:HISTory:EXECute | Executes the statistical processing within the historical data of the automated measurement of waveform parameters. | 5-77 |
| :MEASure:MODE | Turns the automated measurement of waveform parameters ON/OFF and sets the statistical processing or queries the current setting. | 5-77 |
| :MEASure:TRANge | Sets the measurement range or queries the current setting. | 5-77 |
| :MEASure:WAIT? | Waits for the completion of the automated measurement with a timeout. | 5-78 |
| **PHASe Group** | | |
| :PHASe? | Queries all settings related to the phase shift function. | 5-78 |
| :PHASe:CHANnel<x> | Sets the amount of phase shift of the waveform or queries the current setting. | 5-78 |
| :PHASe:MODE | Turns the phase shift function ON/OFF or queries the current setting. | 5-78 |

| Command | Function | Page |
|---|---|---|
| **PRESet Group** | | |
| :PRESet? | Queries all settings related to the preset function. | 5-80 |
| :PRESet:ALL:PROBe | Sets the preset values of the probe attenuation of all channels. | 5-80 |
| :PRESet:ALL:TYPE | Sets preset values to all channels. | 5-80 |
| :PRESet:ALL:USERdefine:LEVel | Sets the preset trigger level of all channels. | 5-80 |
| :PRESet:ALL:USERdefine:OFFSet | Sets the preset offset voltage of all channels. | 5-80 |
| :PRESet:ALL:USERdefine:VDIV | Sets the preset V/div value of all channels. | 5-80 |
| :PRESet:CHANnel<x>? | Queries the preset information of an arbitrary channel. | 5-80 |
| :PRESet:CHANnel<x>:PROBe | Sets the preset probe attenuation of an arbitrary channel or queries the current setting. | 5-80 |
| :PRESet:CHANnel<x>:TYPE | Sets the preset type to the channel or queries the current setting. | 5-80 |
| :PRESet:CHANnel<x>:USERdefine? | Queries all preset values when using User preset. | 5-80 |
| :PRESet:CHANnel<x>:USERdefine:LEVel | Sets the preset trigger level or queries the current setting. | 5-80 |
| :PRESet:CHANnel<x>:USERdefine:OFFSet | Sets the preset offset voltage or queries the current setting. | 5-81 |
| :PRESet:CHANnel<x>:USERdefine:VDIV | Sets the preset V/div value or queries the current setting. | 5-81 |
| :PRESet:EXECute | Executes preset. | 5-81 |
| :PRESet:SELectch | Sets the channel to be preset or queries the current setting. | 5-81 |
| **SEARch Group** | | |
| :SEARch? | Queries all settings related to the search function. | 5-86 |
| :SEARch:ASCRoll? | Queries all settings related to the auto scroll function. | 5-86 |
| :SEARch:ASCRoll:DIRection | Sets the direction for moving the zoom box of the auto scroll or queries the current setting. | 5-86 |
| :SEARch:ASCRoll:SPEed | Sets the speed for moving the zoom box of the auto scroll or queries the current setting. | 5-86 |
| :SEARch:EDGE? | Queries all settings related to the edge search. | 5-86 |
| :SEARch:EDGE:COUNt | Sets the edge search count or queries the current setting. | 5-86 |
| :SEARch:EDGE:HYSTeresis | Sets the hysteresis on the edge search level or queries the current setting. | 5-86 |
| :SEARch:EDGE:LEVEl | Sets the edge search level or queries the current setting. | 5-87 |
| :SEARch:EDGE:SLOPe | Sets the slope or queries the current setting. | 5-87 |
| :SEARch:EDGE:SOURce | Sets the waveform on which to perform the edge search or queries the current setting. | 5-87 |
| :SEARch:EXECute | Executes the search. | 5-87 |
| :SEARch:MLINkage | Turns the link ON/OFF when changing the zoom rate or queries the current setting. | 5-87 |
| :SEARch:MAG<x> | Sets the zoom rate of the zoom window or queries the current setting. | 5-87 |
| :SEARch:PLINkage | Turns the link ON/OFF when moving the zoom box or queries the current setting. | 5-87 |
| :SEARch:POSition<x> | Sets the zoom box position or queries the current setting. | 5-87 |
| :SEARch:PPATtern? | Queries all settings related to the parallel pattern search. | 5-88 |
| :SEARch:PPATtern:{CHANnel<x>\|MATH<x>}? | Queries all settings related to the waveform of the parallel pattern search. | 5-88 |
| :SEARch:PPATtern:{CHANnel<x>\|MATH<x>}:HYSTeresis | Sets the hysteresis on the waveform on which to perform the parallel pattern search or queries the current setting. | 5-88 |
| :SEARch:PPATtern:{CHANnel<x>\|MATH<x>}:LEVel | Sets the search level on the waveform on which to perform the parallel pattern search or queries the current setting. | 5-88 |
| :SEARch:PPATtern:{CHANnel<x>\|MATH<x>}:PATTern | Sets the search pattern on the waveform on which to perform the parallel pattern search or queries the current setting. | 5-88 |
| :SEARch:PPATtern:CLOCk? | Queries all settings related to the clock channel of the parallel pattern search. | 5-88 |
| :SEARch:PPATtern:CLOCk:HYSTeresis | Sets the hysteresis on the clock channel of the parallel pattern search or queries the current setting. | 5-89 |
| :SEARch:PPATtern:CLOCk:LEVel | Sets the search level of the clock channel of the parallel pattern search or queries the current setting. | 5-89 |
| :SEARch:PPATtern:CLOCk:SLOPe | Sets the slope of the clock channel of the parallel pattern search or queries the current setting. | 5-89 |

| Command | Function | Page |
|---|---|---|
| :SEARch:PPATtern:CLOCk:SOURce | Sets the waveform used as the clock channel of the parallel pattern search or queries the current setting. | 5-89 |
| :SEARch:PPATtern:LOGic? | Queries all settings related to the logic channel of the parallel pattern search. | 5-89 |
| :SEARch:PPATtern:LOGic:{A0\|A1\|A2\|A3\|A4\|A5\|A6\|A7\|B0\|B1\|B2\|B3\|B4\|B5\|B6\|B7} | Sets the bit condition of the clock channel of the parallel pattern search or queries the current setting. | 5-89 |
| :SEARch:SELect | Queries the storage number of the found position and the position that is stored in the specified storage number. | 5-89 |
| :SEARch:SELect? MAXimum | Queries the maximum storage number of the found positions. | 5-89 |
| :SEARch:SPATtern? | Queries all settings related to the serial pattern search. | 5-90 |
| :SEARch:SPATtern:BIT | Sets the bit of the logic waveform to be searched or queries the current setting. | 5-90 |
| :SEARch:SPATtern:CLEar | Clears all search patterns (Don't care). | 5-90 |
| :SEARch:SPATtern:CLOCk? | Queries all settings related to the clock channel. | 5-90 |
| :SEARch:SPATtern:CLOCk:HYSTeresis | Sets the hysteresis of the clock channel or queries the current setting. | 5-90 |
| :SEARch:SPATtern:CLOCk:LEVEl | Sets the level of the clock channel or queries the current setting. | 5-90 |
| :SEARch:SPATtern:CLOCk:SLOPe | Sets the slope of the clock channel or queries the current setting. | 5-90 |
| :SEARch:SPATtern:CLOCk:SOURce | Sets the clock channel source waveform or queries the current setting. | 5-90 |
| :SEARch:SPATtern:INTerval | Sets the search interval of the waveform to be searched or queries the current setting. | 5-90 |
| :SEARch:SPATtern:LEVel | Sets the upper and lower limits of the search level of the waveform to be searched or queries the current setting. | 5-91 |
| :SEARch:SPATtern:PATTern<x> | Sets the search pattern or queries the current setting. | 5-91 |
| :SEARch:SPATtern:PSELect | Selects the pattern number to which the search pattern is to be set or queries the current setting. | 5-91 |
| :SEARch:SPATtern:SOURce | Sets the waveform on which to perform the serial pattern search or queries the current setting. | 5-91 |
| :SEARch:SPOint | Sets the start position of the search or queries the current setting. | 5-91 |
| :SEARch:TWINdow | Sets the zoom window in which to display the sections that are found or queries the current setting. | 5-91 |
| :SEARch:TYPE | Sets the search type or queries the current setting. | 5-91 |
| :SEARch:WIDTh? | Queries all settings related to the pulse width search. | 5-92 |
| :SEARch:WIDTh:HYSTeresis | Sets the hysteresis on the pulse width search level or queries the current setting. | 5-92 |
| :SEARch:WIDTh:LEVel | Sets the pulse width search level or queries the current setting. | 5-92 |
| :SEARch:WIDTh:POLarity | Sets the pulse width search level's polarity or queries the current setting. | 5-92 |
| :SEARch:WIDTh:SOURce | Sets the search source for the pulse width search or queries the current setting. | 5-92 |
| :SEARch:WIDTh:TIME<x> | Sets the pulse width when using the pulse width search or queries the current setting. | 5-92 |
| :SEARch:WIDTh:TYPE | Sets the search type when using the pulse width search or queries the current setting. | 5-92 |

### SNAP Group

| | | |
|---|---|---|
| :SNAP | Executes snap shot. | 5-93 |

### SSTARt Group

| | | |
|---|---|---|
| :SSTARt? | Starts the data acquisition with the trigger mode set to single. | 5-93 |

### STARt Group

| | | |
|---|---|---|
| :STARt | Starts waveform acquisition. | 5-93 |

### STATus Group

| | | |
|---|---|---|
| :STATus? | Queries all settings related to the communication status function. | 5-94 |
| :STATus:CONDition? | Queries the status register. | 5-94 |
| :STATus:EESE | Sets the extended event enable register or queries the current setting. | 5-94 |
| :STATus:EESR? | Queries the extended event register and clears the register. | 5-94 |
| :STATus:ERRor? | Queries the error code and the details of the error. | 5-95 |
| :STATus:FILTer<x> | Sets the transition filter or queries the current setting. | 5-95 |
| :STATus:QENable | Sets whether or not to store messages other than errors in the error queue or queries the current setting. | 5-95 |

| Command | Function | Page |
|---|---|---|
| :STATus:QMESsage | Sets whether or not to attach message information to the response to the ":STATus:ERRor?" query or queries the current setting. | 5-95 |
| :STATus:SPOLl? | Executes serial polling. | 5-95 |
| **STOP Group** | | |
| :STOP | Stops the waveform acquisition. | 5-95 |
| **SYSTem Group** | | |
| :SYSTem? | Queries all settings related to the system. | 5-97 |
| :SYSTem:BATTery? | Queries the lithium battery level. | 5-97 |
| :SYSTem:CLICk | Turns the click sound ON/OFF or queries the current setting. | 5-97 |
| :SYSTem:CLOCk? | Queries all settings related to date and time. | 5-97 |
| :SYSTem:CLOCk:DATE | Sets the date or queries the current setting. | 5-97 |
| :SYSTem:CLOCk:MODE | Turns the date and time display ON/OFF or queries the current setting. | 5-97 |
| :SYSTem:CLOCk:TIME | Sets the time or queries the current setting. | 5-97 |
| :SYSTem:LANGuage | Sets the message language or queries the current setting. | 5-97 |
| :SYSTem:LCD? | Queries all settings related to the LCD monitor. | 5-97 |
| :SYSTem:LCD:AUTO? | Queries all settings related to the LCD backlight auto-off function. | 5-97 |
| :SYSTem:LCD:AUTO:MODE | Turns the LCD backlight ON/OFF or queries the current setting. | 5-97 |
| :SYSTem:LCD:AUTO:TIME | Sets the time for the LCD backlight auto-off function or queries the current setting. | 5-97 |
| :SYSTem:LCD:BRIGhtness | Sets the brightness of the LCD monitor or queries the current setting. | 5-98 |
| :SYSTem:LCD:MODE | Turns the LCD backlight ON/OFF or queries the current setting. | 5-98 |
| :SYSTem:OCANcel(Offset Cancel) | Sets whether or not to reflect the offset voltage to the measured results or the computed results or queries the current setting. | 5-98 |
| :SYSTem:OVERview | Displays the system information. | 5-98 |
| :SYSTem:SCSI? | Queries all settings related to the SCSI-ID. | 5-98 |
| :SYSTem:SCSI:INITialize | Initializes SCSI-related settings. | 5-98 |
| :SYSTem:SCSI:OWNid | Sets the SCSI-ID of the instrument or queries the current setting. | 5-98 |
| :SYSTem:TGAte | Sets the external trigger gate or queries the current setting. | 5-98 |
| **TIMebase Group** | | |
| :TIMebase? | Queries all settings related to the time base. | 5-99 |
| :TIMebase:SOURce | Sets the time base or queries the current setting. | 5-99 |
| :TIMebase:SRATe? | Queries the sample rate. | 5-99 |
| :TIMebase:TDIV | Sets the T/div value or queries the current setting. | 5-99 |
| **TRIGger Group** | | |
| :TRIGger? | Queries all settings related to the trigger. | 5-104 |
| :TRIGger:ABN? | Queries all settings related to the A→B (n) trigger. | 5-104 |
| :TRIGger:ABN:COUNt | Sets the number of times condition B is to be met or queries the current setting. | 5-104 |
| :TRIGger:ACTion? | Queries all settings related to the action-on-trigger. | 5-104 |
| :TRIGger:ACTion:ACQCount | Sets the number of times to perform the action for the action-on-trigger or queries the current setting. | 5-104 |
| :TRIGger:ACTion:BUZZer | Sets whether or not to sound an alarm when the action is enabled or queries the current setting. | 5-105 |
| :TRIGger:ACTion:HCOPy | Sets whether or not to make a hard copy of the screen image data when the action is enabled or queries the current setting. | 5-105 |
| :TRIGger:ACTion:MAIL? | Queries all the setting values for mail reports sent as an ACTion. | 5-105 |
| :TRIGger:ACTion:MAIL:COUNt | Sets/queries the limit for the number of reports that are sent as an ACTion. | 5-105 |
| :TRIGger:ACTion:MAIL:MODE | Sets/queries whether mail reports are sent as an ACTion. | 5-105 |
| :TRIGger:ACTion:SAVE | Sets whether or not to save the waveform data when the action is enabled or queries the current setting. | 5-105 |
| :TRIGger:ACTion:STARt | Starts action-on-trigger. | 5-105 |
| :TRIGger:ACTion:STOP | Aborts action-on-trigger. | 5-105 |
| :TRIGger:ADB? | Queries all settings related to the A Delay B trigger. | 5-105 |
| :TRIGger:ADB:DELay | Sets the delay time of condition B or queries the current setting. | 5-105 |
| :TRIGger:ATRigger? | Queries all settings related to condition A. | 5-105 |

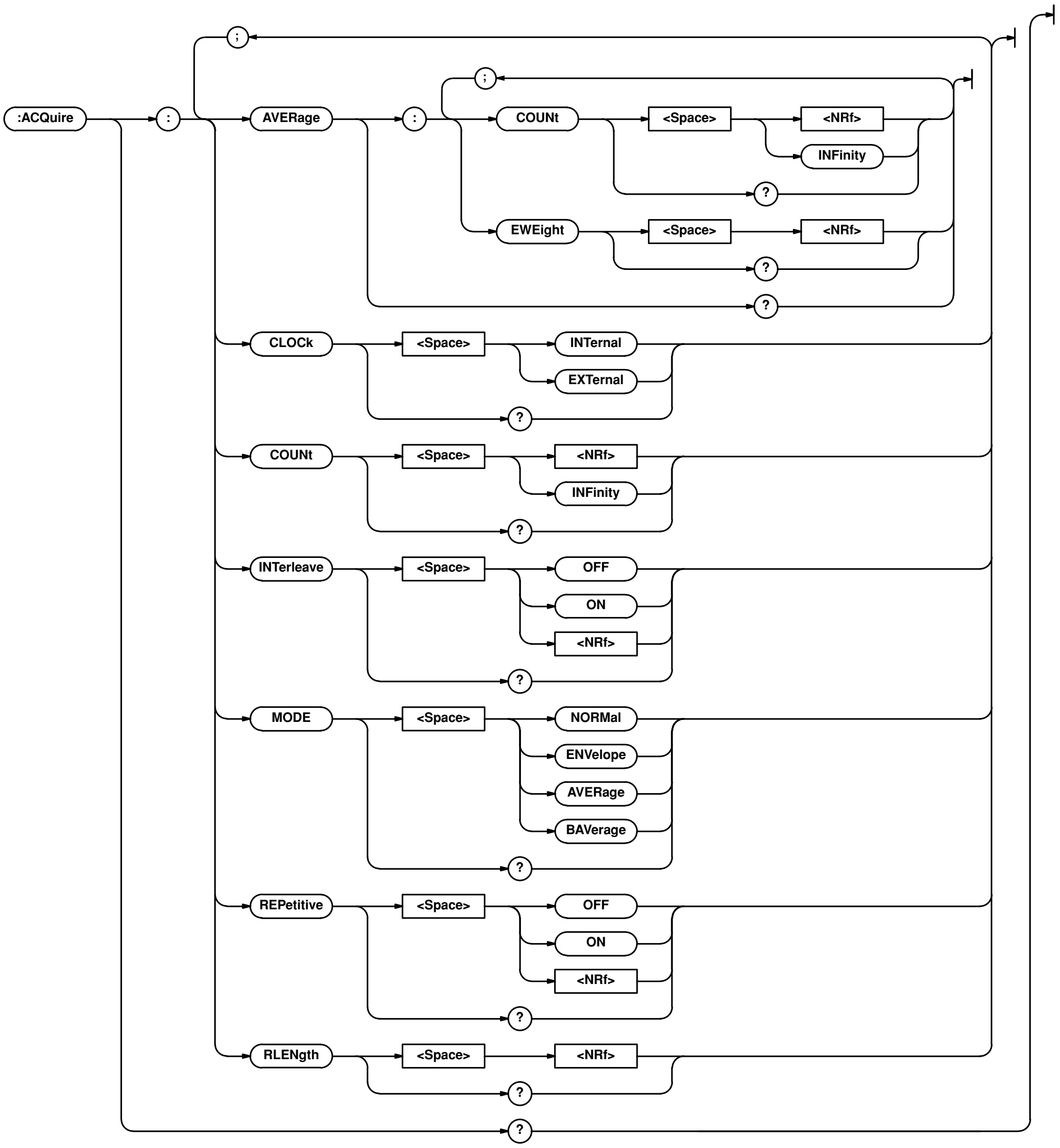
| Command | Function | Page |
|---|---|---|
| :TRIGger:ATRigger:CHANnel<x> | Sets the condition of each channel for condition A or queries the current setting. | 5-106 |
| :TRIGger:ATRigger:CONDition | Sets the conditions for condition A or queries the current setting. | 5-106 |
| :TRIGger:BTRigger? | Queries all settings related to condition B. | 5-106 |
| :TRIGger:BTRigger:CHANnel<x> | Sets the condition of each channel for condition B or queries the current setting. | 5-106 |
| :TRIGger:BTRigger:CONDition | Sets the conditions for condition B or queries the current setting. | 5-106 |
| :TRIGger:DELay | Sets the delay or queries the current setting. | 5-106 |
| :TRIGger:HOLDoff? | Queries all settings related to the hold off. | 5-106 |
| :TRIGger:HOLDoff:TIME | Sets the hold off time or queries the current setting. | 5-106 |
| :TRIGger:LOGic? | Queries all settings related to logic trigger edge. | 5-106 |
| :TRIGger:LOGic:{A0\|A1\|A2\|A3\|A4\|A5\|A6\|A7\|B0\|B1\|B2\|B3\|B4\|B5\|B6\|B7} | Sets the condition of each bit or queries the current setting. | 5-107 |
| :TRIGger:LOGic:CLOCk? | Queries all settings related to the clock channel. | 5-107 |
| :TRIGger:LOGic:CLOCk:SLOPe | Sets the slope of the clock channel or queries the current setting. | 5-107 |
| :TRIGger:LOGic:CLOCk:SOURce | Sets the clock channel or queries the current setting. | 5-107 |
| :TRIGger:LOGic:CONDition | Sets the trigger condition or queries the current setting. | 5-107 |
| :TRIGger:MODE | Sets the trigger mode or queries the current setting. | 5-107 |
| :TRIGger:OR? | Queries all settings related to the OR trigger. | 5-107 |
| :TRIGger:OR:CHANnel<x> | Sets the edge of each channel or queries the current setting. | 5-107 |
| :TRIGger:OR:WINDow | Turns the window ON/OFF or queries the current setting. | 5-108 |
| :TRIGger:PATTern? | Queries all settings related to the pattern trigger. | 5-108 |
| :TRIGger:PATTern:CHANnel<x> | Sets the state of each channel or queries the current setting. | 5-108 |
| :TRIGger:PATTern:CLOCk? | Queries all settings related to the clock channel. | 5-108 |
| :TRIGger:PATTern:CLOCk:SLOPe | Sets the slope of the clock channel or queries the current setting. | 5-108 |
| :TRIGger:PATTern:CLOCk:SOURce | Sets the clock channel or queries the current setting. | 5-108 |
| :TRIGger:PATTern:CONDition | Sets the trigger condition or queries the current setting. | 5-108 |
| :TRIGger:POSition | Sets the trigger position or queries the current setting. | 5-108 |
| :TRIGger:SCOunt | Sets the number of times the trigger is activated when the trigger mode is set to Single (N) or queries the current setting. | 5-108 |
| :TRIGger[:SIMPle]? | Queries all settings related to the simple trigger. | 5-108 |
| :TRIGger[:SIMPle]:COUPling | Sets the trigger coupling or queries the current setting. | 5-109 |
| :TRIGger[:SIMPle]:HFRejection | Turns the low-pass filter (HF rejection) ON/OFF or queries the current setting. | 5-109 |
| :TRIGger[:SIMPle]:HYSTeresis | Sets the hysteresis on the search level or queries the current setting. | 5-109 |
| :TRIGger[:SIMPle]:LEVel | Sets the trigger level of the specified channel or queries the current setting. | 5-109 |
| :TRIGger[:SIMPle]:PROBe | Sets the probe when the trigger source is set to "EXTernal" or queries the current setting. | 5-109 |
| :TRIGger[:SIMPle]:SLOPe | Sets the trigger slope of the specified channel or queries the current setting. | 5-109 |
| :TRIGger[:SIMPle]:SOURce | Sets the trigger source or queries the current setting. | 5-109 |
| :TRIGger:SOURce? | Queries all settings related to the trigger source for the enhanced trigger. | 5-110 |
| :TRIGger:SOURce:CHANnel<x>? | Queries all settings related to the trigger source of the specified channel. | 5-110 |
| :TRIGger:SOURce:CHANnel<x>:COUPling | Sets the trigger coupling of each channel or queries the current setting. | 5-110 |
| :TRIGger:SOURce:CHANnel<x>:HFRejection | Sets the low-pass filter or queries the current setting. | 5-110 |
| :TRIGger:SOURce:CHANnel<x>:HYSTeresis | Sets the hysteresis on the search level or queries the current setting. | 5-110 |
| :TRIGger:SOURce:CHANnel<x>:LEVel | Sets the trigger level of each channel or queries the current setting. | 5-110 |
| :TRIGger:SOURce:CHANnel<x>:WIDTh | Sets the width of the window trigger or queries the current setting. | 5-110 |
| :TRIGger:TV? | Queries all settings related to the TV trigger. | 5-111 |
| :TRIGger:TV:FIELd | Sets the field or queries the current setting. | 5-111 |
| :TRIGger:TV:FRAMe | Sets the frame skip function or queries the current setting. | 5-111 |
| :TRIGger:TV:LEVel | Sets the trigger level or queries the current setting. | 5-111 |
| :TRIGger:TV:LINE | Sets the line or queries the current setting. | 5-111 |
| :TRIGger:TV:POLarity | Sets the input polarity or queries the current setting. | 5-111 |
| :TRIGger:TV:TYPE | Sets the trigger input type or queries the current setting. | 5-111 |
| :TRIGger:TYPE | Sets the trigger type or queries the current setting. | 5-111 |
| :TRIGger:WIDTh? | Queries all settings related to the pulse width trigger. | 5-111 |
| :TRIGger:WIDTh:CHANnel<x> | Sets the trigger condition of each channel or queries the current setting. | 5-111 |
| :TRIGger:WIDTh:CONDition | Sets the trigger condition or queries the current setting. | 5-112 |

| Command | Function | Page |
|---|---|---|
| :TRIGger:WIDTh:TIME<x> | Sets the pulse width or queries the current setting. | 5-112 |
| :TRIGger:WIDTh:TLINkage | Turns the link ON/OFF when changing the pulse width for the pulse width trigger or queries the current setting. | 5-112 |
| :TRIGger:WIDTh:TYPE | Sets the trigger type or queries the current setting. | 5-112 |
| :TRIGger:WIDTh:WINDow | Turns the window ON/OFF or queries the current setting. | 5-112 |
| **WAVeform Group** | | |
| :WAVeform? | Queries all information related to the waveform data. | 5-113 |
| :WAVeform:BITS? | Queries the bit length of the specified waveform data. | 5-113 |
| :WAVeform:BYTeorder | Sets the transmission byte order or queries the current setting. | 5-114 |
| :WAVeform:DATaselect | Sets whether to use the ACQ data or P-P data when querying the waveform data or queries the current setting. | 5-114 |
| :WAVeform:END | Sets which point of the specified waveform is to be the last data point or queries the current setting. | 5-114 |
| :WAVeform:FORMat | Sets the transmission data format or queries the current setting. | 5-114 |
| :WAVeform:LENGth? | Queries the total data length of the specified waveform. | 5-114 |
| :WAVeform:OFFSet? | Queries the offset voltage of the specified waveform. | 5-114 |
| :WAVeform:RANGe? | Queries the range that is used when converting the waveform specified by the ":WAVeform:TRACe" command to physical values. | 5-114 |
| :WAVeform:RECord | Sets the target record number of the waveform group or queries the current setting. | 5-114 |
| :WAVeform:RECord? MINimum | Queries the smallest record number. | 5-114 |
| :WAVeform:SEND? | Queries the specified waveform data. | 5-115 |
| :WAVeform:SIGN? | Queries the existence of a sign. | 5-115 |
| :WAVeform:SRATe? | Queries the sample rate of the specified record. | 5-115 |
| :WAVeform:STARt | Sets which point of the specified waveform is to be the start data point or queries the current setting. | 5-115 |
| :WAVeform:TRACe | Sets the target waveform of the WAVeform group or queries the current setting. | 5-115 |
| :WAVeform:TRIGger? | Queries the trigger position of the specified record. | 5-115 |
| :WAVeform:TYPE? | Queries the acquisition mode of the specified waveform. | 5-115 |
| **XY Group** | | |
| :XY<x>? | Queries all settings related to the X-Y display. | 5-116 |
| :XY<x>:DISPlay | Sets whether or not to display the X-Y waveform or queries the current setting. | 5-116 |
| :XY<x>:MODE | Sets the display mode or queries the current setting. | 5-116 |
| :XY:TRANge | Sets the range of the T-Y waveform to be used in the X-Y display or queries the current setting. | 5-116 |
| :XY<x>:XAXis | Sets whether or not use the same channel for the X-axis for XY1 and XY2. | 5-117 |
| :XY<x>:XTRace | Sets the channel that is to be assigned to the X-axis or queries the current setting. | 5-117 |
| :XY<x>:YTRace | Sets the channel that is to be assigned to the Y-axis or queries the current setting. | 5-117 |
| **ZOOM Group** | | |
| :ZOOM? | Queries all settings related to the zooming of the waveform. | 5-119 |
| :ZOOM:ALLOcation? | Queries all settings related to the waveform being zoomed. | 5-119 |
| :ZOOM:ALLOcation:ALL | Selects all waveforms to be zoomed. | 5-119 |
| :ZOOM:ALLOcation:{CHANnel<x>\|MATH<x>} | Sets the waveform to be zoomed or queries the current setting. | 5-119 |
| :ZOOM:ALLOcation:{PODA\|PODB} | Sets the pod to be zoomed or queries the current setting. | 5-119 |
| :ZOOM:FITMeasure | Changes the range of the automated measurement of waveform parameters to the ends of the zoom box. | 5-119 |
| :ZOOM:FORMat | Sets the zoom waveform display format or queries the current setting. | 5-119 |
| :ZOOM:MLINkage | Turns the zoom rate link ON/OFF or queries the current setting. | 5-119 |
| :ZOOM:MAG<x> | Sets the zoom rate of the zoomed waveform display or queries the current setting. | 5-119 |
| :ZOOM:MODE | Sets the zoomed waveform display mode or queries the current setting. | 5-119 |
| :ZOOM:PLINkage | Turns the link ON/OFF when moving the zoom box or queries the current setting. | 5-119 |
| :ZOOM:POSition<x> | Sets the zoom box position or queries the current setting. | 5-119 |

| Command | Function | Page |
|---|---|---|
| **Common Command Group** | | |
| *CAL? | Executes calibration and queries the result. | 5-120 |
| *CLS | Clears the standard event register, extended event register, and error queue. | 5-120 |
| *ESE | Sets the standard event enable register or queries the current setting. | 5-120 |
| *ESR? | Queries the standard event register and clears the register. | 5-121 |
| *IDN? | Queries the instrument model. | 5-121 |
| *LRN? | Queries all current settings of the command group. | 5-121 |
| *OPC | Sets the OPC event after the completion of the specified overlap command. | 5-122 |
| *OPC? | Creates a response upon the completion of the specified overlap command. | 5-122 |
| *OPT? | Queries the installed options. | 5-122 |
| *PSC | Sets whether or not to clear the registers at power up or queries the current setting. | 5-122 |
| *RST | Initializes the command group's settings. | 5-122 |
| *SRE | Sets the service request enable register or queries the current setting. | 5-122 |
| *STB? | Queries the status byte register. | 5-122 |
| *TST? | Executes self test and queries the result. | 5-123 |
| *WAI | Holds the subsequent command until the completion of the specified overlap operation. | 5-123 |

## 5.2 ACQuire Group

The commands in the ACQuire group are used for making settings and queries about acquisition. This allows you to make the same settings that you can make using the ACQ key on the front panel.



### :ACQuire?

Function Quiries all the acquisition settings.

Syntax :ACQuire?

Example :ACQUIRE?→:ACQUIRE:RLENGTH 1000; MODE NORMAL;AVERAGE:COUNT INFINITY; EWEIGHT 16;:ACQUIRE:COUNT INFINITY: INTERLEAVE 0;REPETITIVE 0;CLOCK INTERNAL

### :ACQuire:AVERage?

Function Quiries all the averaging and acquisition count settings.

Syntax :ACQuire:AVERage?

Example :ACQUIRE:AVERAGE?→:ACQUIRE:AVERAGE: COUNT INFINITY;EWEIGHT 16

### :ACQuire:AVERage:COUNt

Function Sets/queries the acquisition count in averaging mode.

Syntax :ACQuire:AVERage:COUNt {<NRf>|INFinity}
:ACQuire:AVERage:COUNt?
<NRf>=2 to 65536 (in steps of $2^n$)

Example :ACQUIRE:AVERAGE:COUNT INFINITY
:ACQUIRE:AVERAGE:COUNT?→:ACQUIRE: AVERAGE:COUNT INFINITY

### :ACQuire:AVERage:EWEight (Exponent WEIGHT)

Function Sets/queries the average weight for infinite averaging mode.

Syntax :ACQuire:AVERage:EWEight {<NRf>}
:ACQuire:AVERage:EWEight?
<NRf>=2 to 256 (in steps of $2^n$)

Example :ACQUIRE:AVERAGE:EWEIGHT 16
:ACQUIRE:AVERAGE:EWEIGHT?→:ACQUIRE:AVERAGE:EWEIGHT 16

### :ACQuire:CLOCk

Function Sets/queries the time base.

Syntax :ACQuire:{INTernal|EXTernal}
:ACQuire:CLOCk?

Example :ACQUIRE:CLOCK INTERNAL
:ACQUIRE:CLOCK?→:ACQUIRE:CLOCK INTERNAL

### :ACQuire:COUNt

Function Sets the number of waveform acquisitions during normal mode, envelope mode, and box average or queries the current setting.

Syntax :ACQuire:COUNt {<NRf>|INFinity}
:ACQuire:COUNt?
<NRf>=2 to 65536

Example :ACQUIRE:COUNT INFINITY
:ACQUIRE:COUNT?→:ACQUIRE:COUNT INFINITY

### :ACQuire:INTerleave

Function Turns the interleave mode ON/OFF or queries the current setting.

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

Example :ACQUIRE:INTERLEAVE ON
:ACQUIRE:INTERLEAVE?→:ACQUIRE:INTERLEAVE ON

### :ACQuire:MODE

Function Sets/queries the acquisition mode.

Syntax :ACQuire:MODE{AVERage|BAVerage|ENVelope|NORMal}
:ACQuire:MODE?

Example :ACQUIRE:MODE NORMAL
:ACQUIRE:MODE?→:ACQUIRE:MODE NORMAL

### :ACQuire:REPetitive

Function Turns the repetitive sampling mode ON/OFF or queries the current setting.

Syntax :ACQuire:REPetitive {<Boolean>}
:ACQuire:REPetitive?

Example :ACQUIRE:REPETITIVE OFF
:ACQUIRE:REPETITIVE?→:ACQUIRE:REPETITIVE 0

### :ACQuire:RLENgth

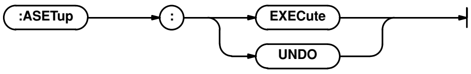
Function Sets/queries the record length.

Syntax :ACQuire:RLENgth {<NRf>}
:ACQuire:RLENgth?
<NRf>= 1000 to 8000000 (DL7100)
1000 to 16000000 (DL7200)

Example :ACQUIRE:RLENGTH 1000
:ACQUIRE:RLENGTH?→:ACQUIRE:RLENGTH 1000

## 5.3 ASETup Group

The commands in the ASETup group are used for setting and querying auto-setup. This allows you to make the same settings and queries that you can make using the AUTO SETUP key on the front panel.



### :ASETup:EXECute

Function Executes auto-setup.

Syntax :ASETup:EXECute

Example :ASETUP:EXECUTE

### :ASETup:UNDO

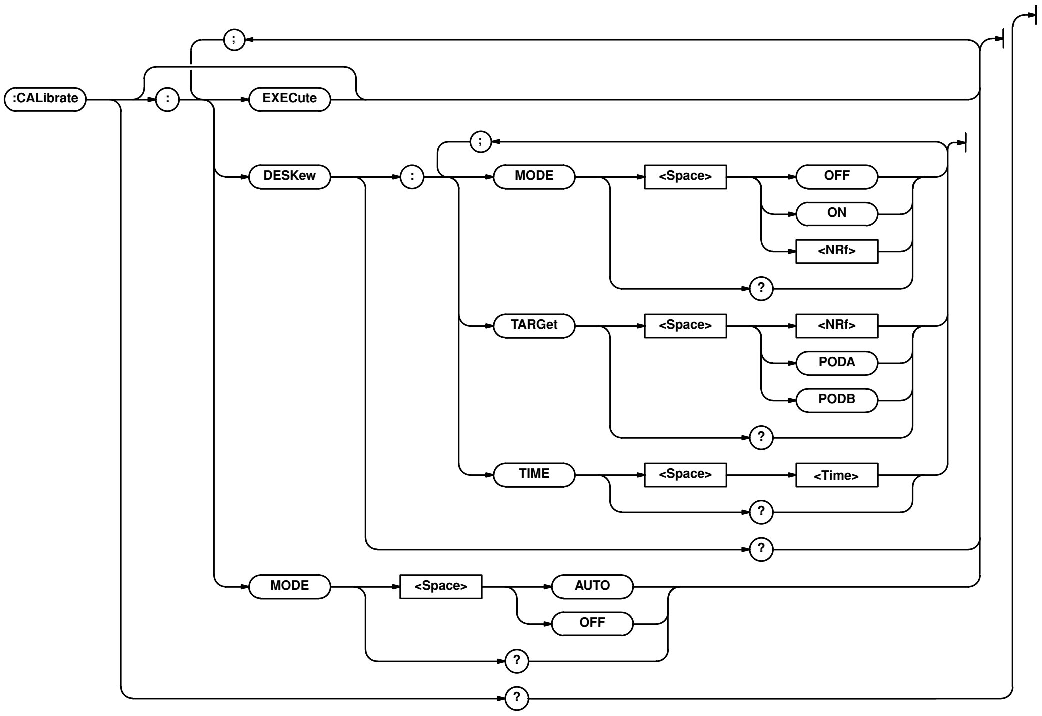
Function Cancels auto set-up settings.

Syntax :ASETup:UNDO

Example :ASETUP:UNDO

## 5.4 CALibrate Group

The commands in the CALibrate group are used for setting and querying the calibration. This allows you to make the same settings that you can make using the Calibration menu of the MISC key on the front panel.



### :CALibrate?

Function Queries all calibration settings.

Syntax `:CALibrate?`

Example `:CALIBRATE?→:CALIBRATE:MODE OFF;DESKEW:MODE 0;TARGET 1;TIME 0.0E+00`

### :CALibrate:DESKew?

Function Queries all settings related to skew compensation.

Syntax `:CALibrate:DESKew?`

Example `:CALIBRATE:DESKEW?→:CALIBRATE:DESKEW:MODE 0;TARGET 1;TIME 0.0E+00`

### :CALibrate:DESKew:MODE

Function Turns the skew compensation ON/OFF or queries the current setting.

Syntax
```
:CALibrate:DESKew:MODE {<Boolean>}
:CALibrate:DESKew:MODE?
```

Example
```
:CALIBRATE:DESKEW:NODE ON
:CALIBRATE:DESKEW:MODE?→:CALIBRATE:
DESKEW:MODE 0
```

### :CALibrate:DESKew:TARGet

Function Sets the channel on which skew compensation is to be performed or queries the current setting.

Syntax
```
CALibrate:DESKew:TARGet {<NRf>|PODA|PODB}
:CALibrate:DESKew:TARGet?
<NRf>=1 to 4
```

Example
```
:CALIBRATE:DESKEW:TARGET 1
:CALIBRATE:DESKEW:TARGET?→:CALIBRATE:
DESKEW:TRAGET 1
```

### :CALibrate:DESKew:TIME

Function Sets the skew adjustment time that are common to all channels or queries the current setting.

Syntax
```
:CALibrate:DESKew:TIME {<time>}
:CALibrate:DESKew:TIME?
<time>=–100 ns to 100 ns
```

Example
```
:CALIBRATE:DESKEW:TIME 1NS
:CALIBRATE:DESKEW:TIME?→:CALIBRATE:
DESKEW:TIME 1.000E-09
```

### :CALibrate[:EXECute]

Function Executes calibration.

Syntax `:CALibrate[:EXECute]`

Example `:CALIBRATE:EXECUTE`

### :CALibrate:MODE

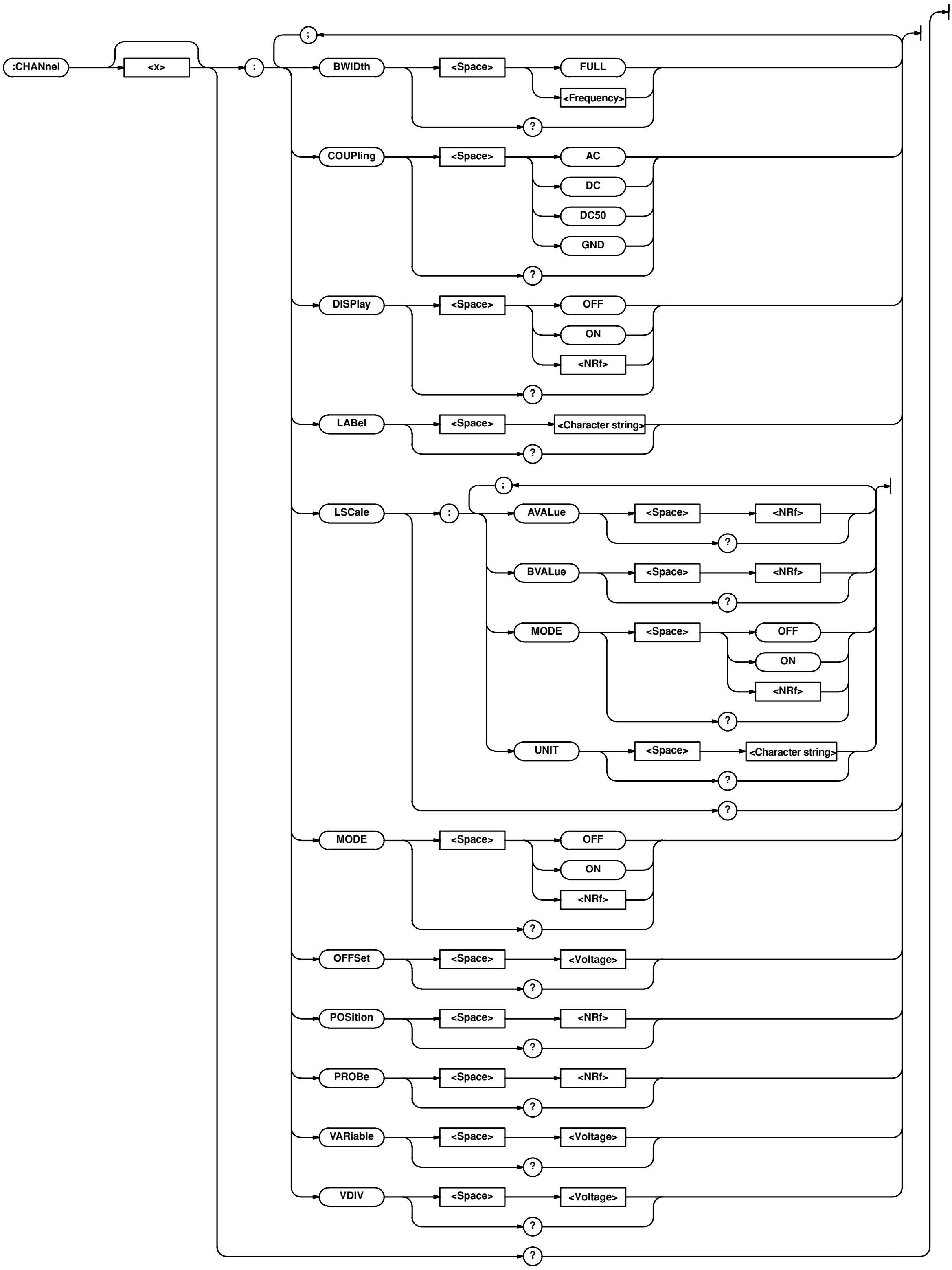
Function Sets/queries the ON/OFF of the auto calibration.

Syntax
```
:CALibrate:MODE {AUTO|OFF}
:CALibrate?
```

Example
```
:CALIBRATE:MODE AUTO
:CALIBRATE?→:CALIBRATE:MODE AUTO
```

## 5.5 CHANnel Group

The commands in the CHANnel group are used to make settings and queries about the vertical axis of the specified channel. This allows you to make the same settings that you can make using the VERTICAL keys (CH key, V/DIV key).



## :CHANnel<x>?

Function Queries all vertical axis settings for the specified channel.

Syntax :CHANnel<x>?
<x>=1 to 4

Example :CHANNEL?→:CHANNEL1:DISPLAY 1;
VDIV 50.000E+00;POSITION 0.00;
COUPLING DC;PROBE 10;OFFSET 0.0E+00;
BWIDTH FULL;VARIABLE 50.000E+00;LSCALE:
MODE 0;AVALUE 1.00000E+00;
BVALUE 0.00000E+00;UNIT "V";:CHANNEL1:
LABEL "CH1"

## :CHANnel<x>:BWIDth

Function Sets the input filter of each channel or queries the current setting.

Syntax :CHANnel<x>:BWIDth {FULL|<frequency>}
:CHANnel<x>:BWIDth?
<x>=1 to 4
{<frequency>}=20 MHz, 100 MHz

Example :CHANNEL1:BWIDTH FULL
:CHANNEL1:BWIDTH?→:CHANNEL1:BWIDTH FULL

## :CHANnel<x>:COUPling

Function Sets the input coupling of each channel or queries the current setting.

Syntax :CHANnel<x>:COUPling {AC|DC|DC50|GND}
:CHANnel<x>:COUPling?
<x>=1 to 4

Example :CHANNEL1:COUPLING DC
:CHANNEL1:COUPLING?→:CHANNEL1:
COUPLING DC

## :CHANnel<x>:DISPlay

Function Sets/queries display ON/OFF for the specified channel.

Syntax :CHANnel<x>:DISPlay {<Boolean>}
:CHANnel<x>:DISPlay?
<x>=1 to 4

Example :CHANNEL1:DISPLAY ON
:CHANNEL1:DISPLAY?→:CHANNEL1:DISPLAY 1

Description The same function can be performed using "CHANnel<x>:MODE."

## :CHANnel<x>:LABel

Function Sets/queries the channel's waveform label.

Syntax :CHANnel<x>:LABel {<Character string>}
:CHANnel<x>:LABel?
<x>=1 to 4
<Character string>=Within 8 characters.

Example :CHANNEL1:LABEL "CH1"
:CHANNEL1:LABEL?→:CHANNEL1:LABEL "CH1"

Description
- Only characters and symbols on the displayed keyboard are available for use in the label.
- ASCII codes for "Ω" and "μ" are 1EH and 1FH, respectively.

## :CHANnel<x>:LSCale?

Function Queries all linear scaling settings.

Syntax :CHANnel<x>:LSCale?
<x>=1 to 4

Example :CHANNEL1:LSCALE:MODE 1;AVALUE 1.00000E+00;
BVALUE 0.00000E+00;UNIT "V"

## :CHANnel<x>:LSCale:AVALue

Function Sets/queries the constant (coefficient) A of linear scaling.

Syntax :CHANnel<x>:LSCale:AVALue {<NRf>}
:CHANnel<x>:LSCale:AVALue?
<x>=1 to 4
<NRf>=−9.9999E+30 to 9.9999E+30

Example :CHANNEL1:LSCALE:AVALUE 10
:CHANNEL1:LSCALE:AVALUE?→:CHANNEL:
LSCALE:AVALUE 10.0000E+00

## :CHANnel<x>:LSCale:BVALue

Function Sets/queries the offset value B of linear scaling.

Syntax :CHANnel<x>:LSCale:BVALue {<NRf>}
:CHANnel<x>:LSCale:BVALue?
<x>=1 to 4
<NRf>=−9.9999E+30 to 9.9999E+30

Example :CHANNEL1:LSCALE:BVALUE 10
:CHANNEL1:LSCALE:BVALUE?→:CHANNEL:
LSCALE:BVALUE 10.0000E+00

## :CHANnel<x>:LSCale:MODE

Function Sets/queries the ON/OFF condition of linear scaling.

Syntax :CHANnel<x>:LSCale:MODE {<Boolean>}
:CHANnel<x>:LSCale:MODE?
<x>=1 to 4

Example :CHANNEL1:LSCALE:MODE ON
:CHANNEL1:LSCALE:MODE?→:CHANNEL1:
LSCALE:MODE 1

## :CHANnel<x>:LSCale:UNIT

Function Sets/queries the dimensional unit which is appended to the linear scaling result.

Syntax :CHANnel<x>:LSCale:UNIT
{<Character string>}
:CHANnel<x>:LSCale:UNIT?
<x>=1 to 4
<Character string>=Within to 4 characters.

Example :CHANNEL1:LSCALE:UNIT "EU"
:CHANNEL1:LSCALE:UNIT?→:CHANNEL1:
LSCALE:UNIT "EU"

Description
- Only the characters and symbols on the keyboard that appear on the instrument's screen can be used.
- ASCII codes for "Ω" and "μ" are 1EH and 1FH, respectively.
- This unit will not affect the scaled value in any way.

### :CHANnel<x>:MODE

Function Sets/queries display ON/OFF for the specified channel.

```
Syntax      :CHANnel<x>:MODE {<Boolean>}
            :CHANnel<x>:MODE?
            <x>=1 to 4
Example     :CHANNEL1:MODE ON
            :CHANNEL1:MODE?→:CHANNEL1:MODE 1
```

Description The same function can be performed using "CHANnel<x>:DISPlay."

### :CHANnel<x>:OFFSet

Function Sets/queries the offset voltage, when the voltage module is installed at the channel (slot).

```
Syntax      :CHANnel<x>:OFFSet {<Voltage>}
            :CHANnel<x>:OFFSet?
            <x>=1 to 4
            <Voltage>= See the IM701410-01E User's
                       Manual.
Example     :CHANNEL1:OFFSET 0V
            :CHANNEL1:OFFSET?→:CHANNEL1:
            OFFSET 0.000E+00
```

### :CHANnel<x>:POSition

Function Sets/queries the vertical position, when the voltage module is installed at the channel (slot).

```
Syntax      :CHANnel<x>:POSition {<NRf>}
            :CHANnel<x>:POSition?
            <x>=1 to 4
            <NRf>=−4.00 to 4.00div (in 1/100 steps)
Example     :CHANNEL1:POSITION 1
            :CHANNEL1:POSITION?→:CHANNEL1:
            POSITION 1.00E+00
```

### :CHANnel<x>:PROBe

Function Sets/queries the probe attenuation.

```
Syntax      :CHANnel<x>:PROBe {<NRf>}
            :CHANnel<x>:PROBe?
            <x>=1 to 4
            <NRf>= 1,10,100,1000 (See the IM701410-
                   01E User's Manual)
Example     :CHANNEL1:PROBE 1
            :CHANNEL1:PROBE?→:CHANNEL1:PROBE 1
```

### :CHANnel<x>:VARiable

Function Sets the voltage sensitivity in units of 0.01 of the V/div setting for each channel or queries the current setting.

```
Syntax      :CHANnel<x>:VARiable {<voltage>}
            :CHANnel<x>:VARiable?
            <x>=1 to 4
            <voltage>= See the IM701410-01E User's
            Manual.
Example     :CHANNEL1:VARIABLE 1V
            :CHANNEL1:VARIABLE?→:CHANNEL1:
            VARIABLE 1.000E+00
```

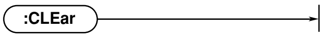
### :CHANnel<x>:VDIV

Function Sets/queries the voltage sensitivity (V/div).

```
Syntax      :CHANnel<x>:VDIV {<Voltage>}
            :CHANnel<x>:VDIV?
            <x>=1 to 4
            <Voltage>=2 mV to 10 V
            (Probe attenuation 1:1, See the
            IM701410-01E User's Manual)
Example     :CHANNEL1:VDIV 2V
            :CHANNEL1:VDIV?→:CHANNEL1:
            VDIV 2.000E+00
```

## 5.6 CLEar Group

The CLEar command is used to clear the trace. This allows you to perform the same operation that you can perform using the CLEAR TRACE key on the front panel.



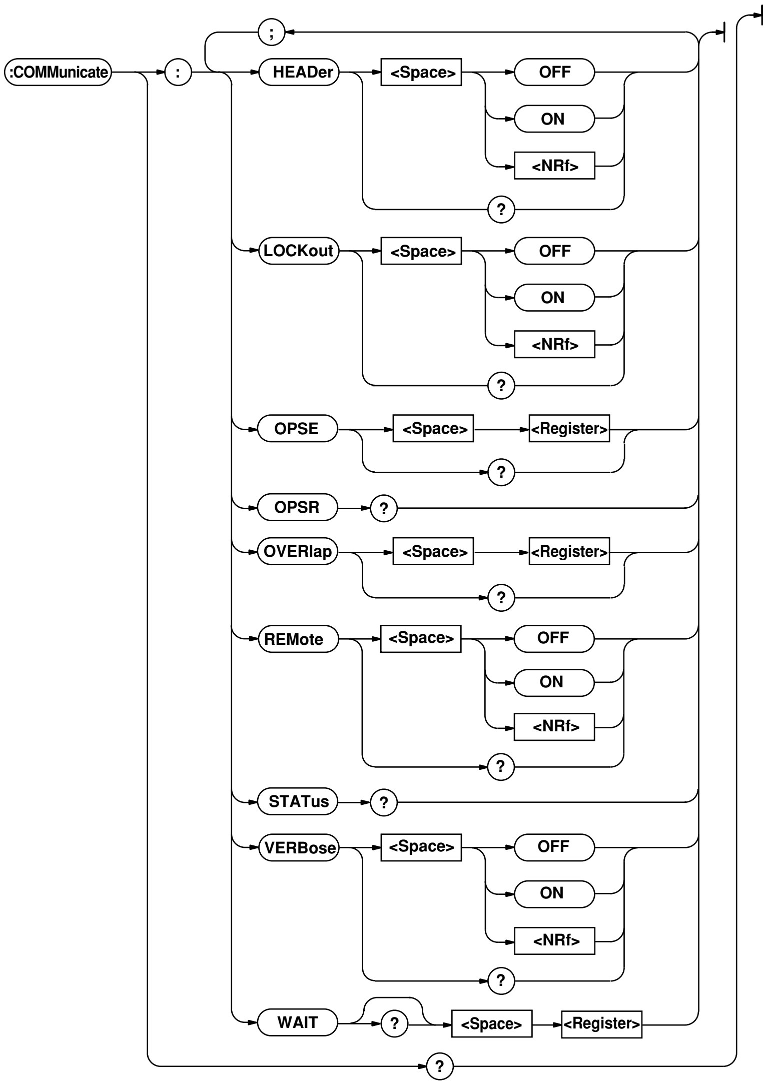
### :CLEar

Function Clears trace.

```
Syntax      :CLEar
Example     :CLEAR
```

## 5.7 COMMunicate Group

The commands in the COMMunicate group are used to make settings and queries about communications. There is no front panel key with this function.



### :COMMunicate?

Function Queries all communication settings.

Syntax `:COMMunicate?`

Example `:COMMUNICATE?→:COMMUNICATE:HEADER 1; OPSE 352;OVERLAP 352;VERBOSE 1`

### :COMMunicate:HEADer

Function Determines whether a header is to be added (for example: `CHANNEL1:VOLTAGE:PROBE 10`) or not (example: `10`) when sending a response to a query; or queries the current setting.

Syntax `:COMMunicate:HEADer {<Boolean>}`
`:COMMunicate:HEADer?`

Example `:COMMUNICATE:HEADER ON`
`:COMMUNICATE:HEADER?→:COMMUNICATE: HEADER 1`

### :COMMunicate:LOCKout

Function Sets/releases the local lockout.

Syntax `:COMMunicate:LOCKout {<Boolean>}`
`:COMMunicate:LOCKout?`

Example `:COMMUNICATE:LOCKOUT ON`
`:COMMUNICATE:LOCKOUT?→:COMMUNICATE: LOCKOUT 1`

Description This is an exclusive command for the serial (RS-232) interface.

## :COMMunicate:OPSE (Operation Pending Status Enable register)

Function Sets the overlap command to be used with *OPC, *OPC?, and *WAI, or queries the current setting.

Syntax
```
:COMMunicate:OPSE <Register>
:COMMunicate:OPSE?
<Register>= 0 to 65535, refer to the
            figure on the next page.
```

Example
```
:COMMUNICATE:OPSE 65535
:COMMUNICATE:OPSE?→:COMMUNICATE:
OPSE 352
```

Description In the above example, all bits are masked to "1" so that all overlap commands can be used by this command. However, bits that are fixed as "0" cannot be set to "1," so only bits 5, 6 and 8 are actually set to "1" and appears as "1" when a query is made.

## :COMMunicate:OPSR? (Operation Pending Status Register)

Function Inquires about the value in the operation pending status register.

Syntax
```
:COMMunicate:OPSR?
```

Example
```
:COMMUNICATE:OPSR?→0
```

Description For a description of the operation pending status register, refer to the figure on the next page.

## :COMMunicate:OVERlap

Function Selects/queries the commands enabled for overlap.

Syntax
```
:COMMunicate:OVERlap <Register>
:COMMunicate:OVERlap?
<Register>=0 to 65535, refer to the
            figure on the next page.
```

Example
```
:COMMUNICATE:OVERLAP 65535
:COMMUNICATE:OVERLAP?→:COMMUNICATE:
OVERLAP 352
```

Description
- In the above example, all bits are set to "1" so that all overlap commands can be used by this command. However, bits that are fixed as "0" cannot be set to "1," so only bits 5, 6 and 8 are actually set to "1" and appears as "1" when a query is made.
- COMMunicate: For the synchronization method using "OVERlap," refer to page 4-8.
- In the above example, bits 5, 6 and 8 are set to "1" so that all overlap commands can be used by this command. (Refer to the figure on the next page.)

## :COMMunicate:REMote

Function Sets remote/local. It is in remote mode when it is set to ON.

Syntax
```
:COMMunicate:REMote {<Boolean>}
:COMMunicate:REMote?
```

Example
```
:COMMUNICATE:REMOTE ON
:COMMUNICATE:REMOTE?→:COMMUNICATE:
REMOTE 1
```

Description This is an exclusive command for the serial (RS-232) interface.

## :COMMunicate:STATus?

Function Queries the circuit status.

Syntax
```
:COMMunicate:STATus?
```

Example
```
:COMMUNICATE:STATUS?→:COMMUNICATE:
STATUS 0
```

Description Status-bit meanings are as follows.

| Bit | GP-IB | Serial (RS-232) |
|---|---|---|
| 0 | Unrecoverable transmission error | Parity error |
| 1 | Always 0 | Framing error |
| 2 | Always 0 | Break character detected |
| 3 to | Always 0 | Always 0 |

Status bit sets when cause occurs, and clears when read.

## :COMMunicate:VERBose

Function Determines whether a response to a query is to be returned in full form (for example: CHANNEL1:VOLTAGE:PROBE 10) or in abbreviated form (for example: CHAN:PROB 10), or queries the current setting.

Syntax
```
:COMMunicate:VERBose {<Boolean>}
:COMMunicate:VERBose?
```

Example
```
:COMMUNICATE:VERBOSE ON
:COMMUNICATE:VERBOSE?→:COMMUNICATE:
VERBOSE 1
```

## :COMMunicate:WAIT

Function Waits until one of the specified extended events occurs.

Syntax
```
:COMMunicate:WAIT <Register>
<Register>=0 to 65535 (Extended event
            register; refer to page 6-4.)
```

Example
```
:COMMUNICATE:WAIT 65535
```

Description COMMunicate: For a description of the synchronizing method using "WAIT," refer to page 4-8.

## :COMMunicate:WAIT?

Function Generates a response when one of the specified extended events occurs.

Syntax :COMMunicate:WAIT? <Register>

<Register>=0 to 65535 (Extended event register; refer to page 6-4.)

Example :COMMUNICATE:WAIT? 65535→1

Operation pending status register/overlap enable register

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | HST | 0 | ACS | PRN | 0 | 0 | 0 | 0 | 0 |

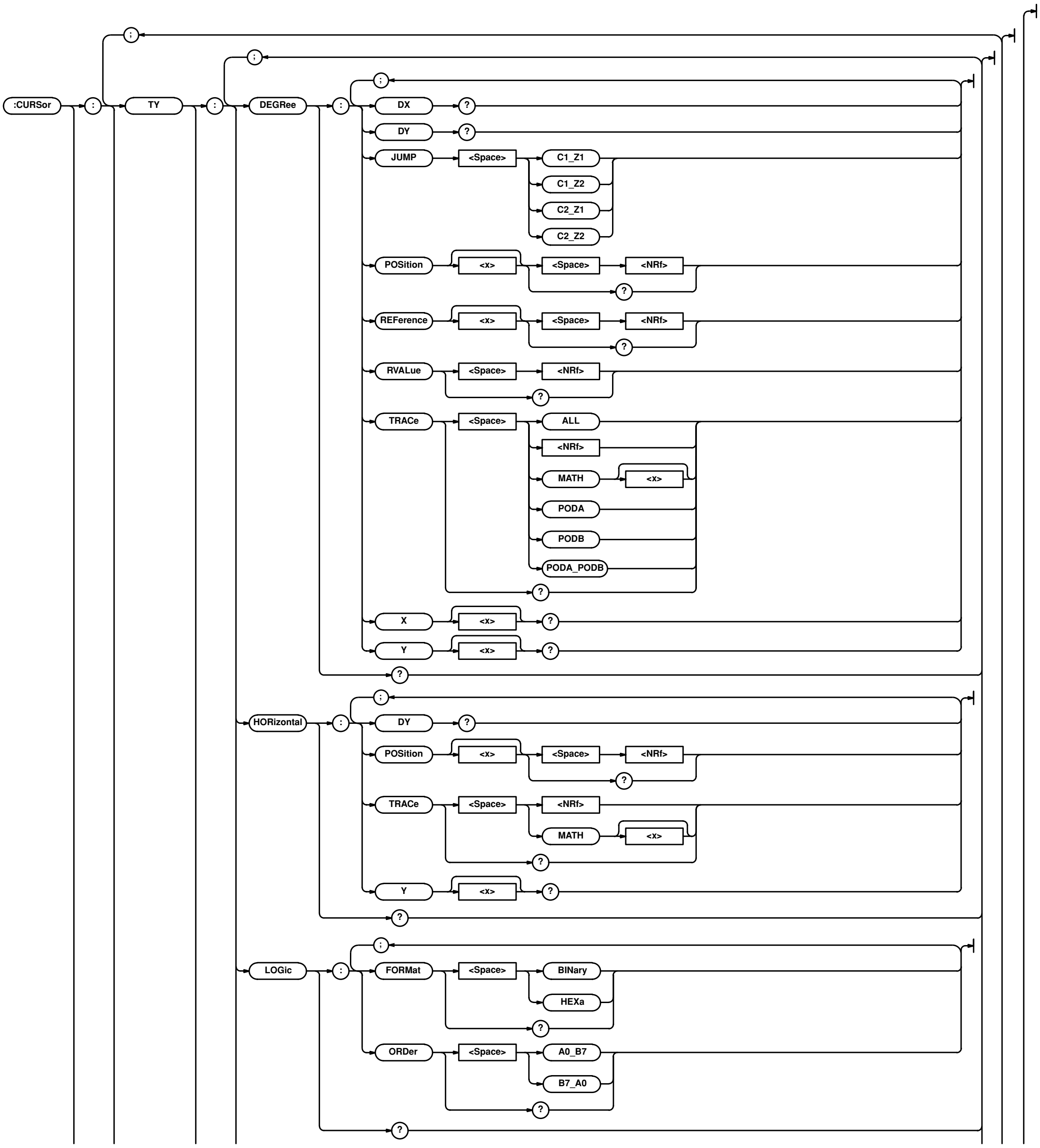
When bit 5 (PRN) = 1:
Printer operation not completed

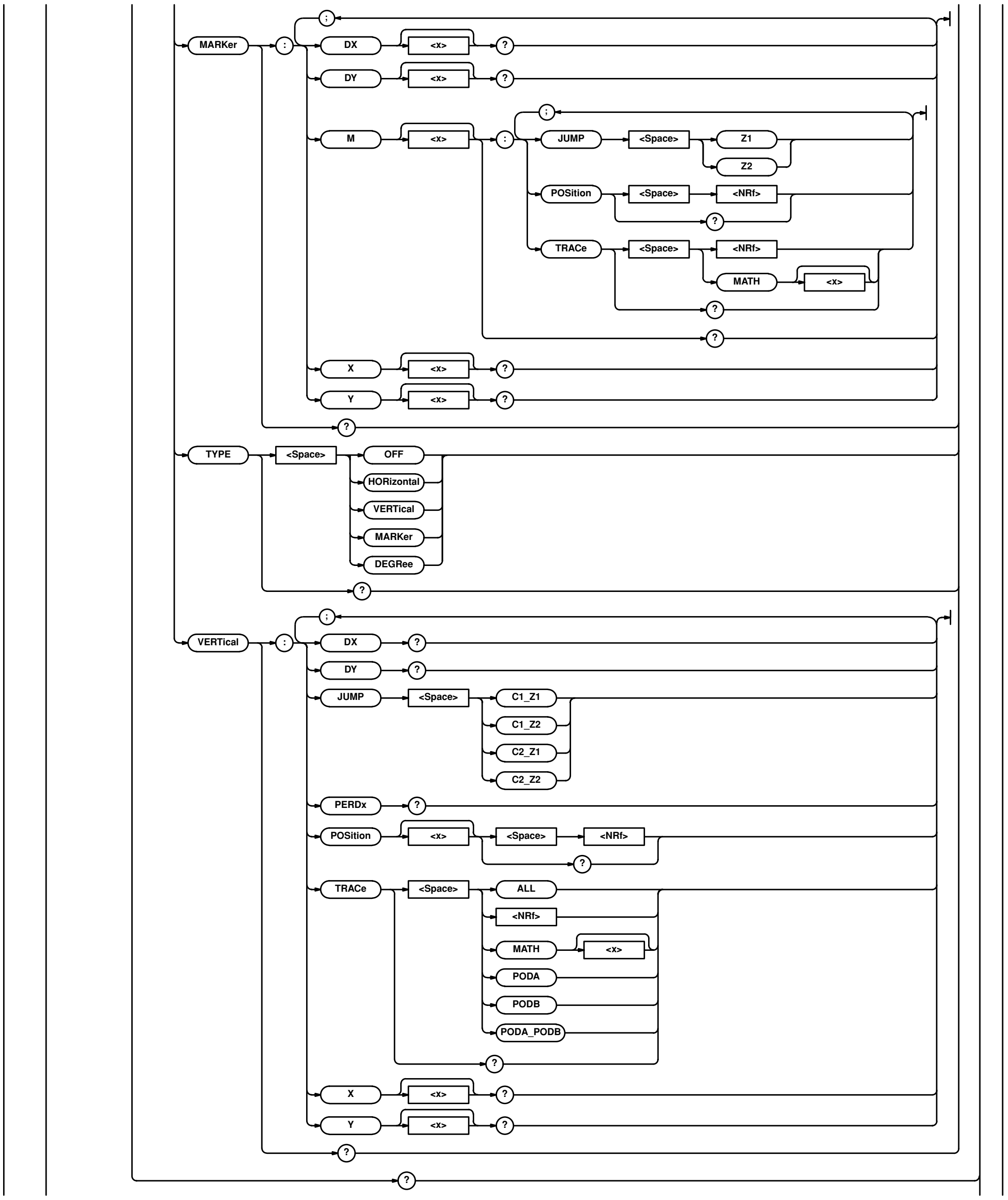
When bit 6 (ACS) = 1:
Medium not accessed

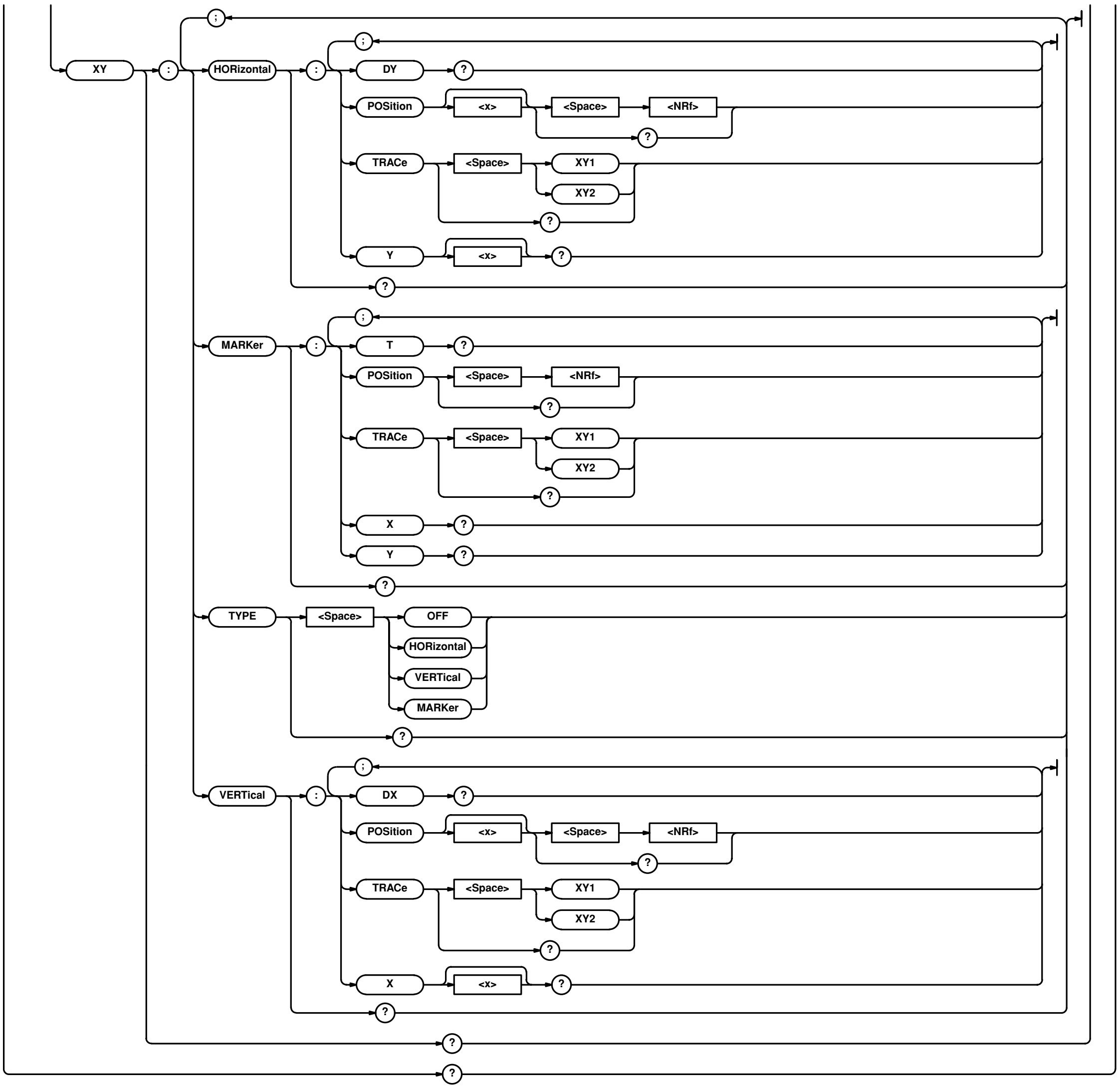
When bit 8 (HST) = 1:
History search not completed

## 5.8 CURSor Group

The commands in the CURSor group are used to make cursor measurement settings and queries. This allows you to make the same settings that you can make using the CURSOR key on the front panel.




MARKer
:
;
DX
<x>
?
DY
<x>
?
M
<x>
:
;
JUMP
<Space>
Z1
Z2
POSition
<Space>
<NRf>
?
TRACe
<Space>
<NRf>
MATH
<x>
?
?
X
<x>
?
Y
<x>
?
?
TYPE
<Space>
OFF
HORizontal
VERTical
MARKer
DEGRee
?
VERTical
:
;
DX
?
DY
?
JUMP
<Space>
C1_Z1
C1_Z2
C2_Z1
C2_Z2
PERDx
?
POSition
<x>
<Space>
<NRf>
?
TRACe
<Space>
ALL
<NRf>
MATH
<x>
PODA
PODB
PODA_PODB
?
X
<x>
?
Y
<x>
?
?
?


;
XY
:
HORizontal
:
;
DY
?
POSition
<x>
<Space>
<NRf>
?
TRACe
<Space>
XY1
XY2
?
Y
<x>
?
?
MARKer
:
;
T
?
POSition
<Space>
<NRf>
?
TRACe
<Space>
XY1
XY2
?
X
?
Y
?
?
TYPE
<Space>
OFF
HORizontal
VERTical
MARKer
?
VERTical
:
;
DX
?
POSition
<x>
<Space>
<NRf>
?
TRACe
<Space>
XY1
XY2
?
X
<x>
?
?
?
?

## :CURSor?

Function Queries all settings related to cursor measurements.

Syntax
```
:CURSor?
```

Example
```
:CURSOR?→:CURSOR:TY:TYPE OFF;
HORIZONTAL:TRACE 1;POSITION1 3.00;
POSITION2 -3.00;:CURSOR:TY:VERTICAL:
TRACE 1;POSITION1 -4.0000000;
POSITION2 4.0000000;:CURSOR:TY:MARKER:
M1:TRACE 1;POSITION -3.0000000;:CURSOR:
TY:MARKER:M2:TRACE 2;
POSITION -1.0000000;:CURSOR:TY:MARKER:
M3:TRACE 3;POSITION 1.0000000;:CURSOR:
TY:MARKER:M4:TRACE 4;
POSITION 3.0000000;:CURSOR:TY:DEGREE:
TRACE 1;POSITION1 -4.0000000;
POSITION2 4.0000000;
REFERENCE1 -2.0000000;
REFERENCE2 2.0000000;RVALUE 90;:CURSOR:
TY:LOGIC:FORMAT BINARY;ORDER A0_B7;:
CURSOR:XY:TYPE OFF;HORIZONTAL:TRACE XY1;
POSITION1 3.00;POSITION2 -3.00;:
CURSOR:XY:VERTICAL:TRACE XY1;
POSITION1 -3.00;POSITION2 3.00;:CURSOR:
XY:MARKER:TRACE XY1;POSITION -3.0000000
```

## :CURSor:TY?

Function Queries all settings related to the cursor in the T-Y display.

Syntax
```
:CURSor:TY?
```

Example
```
:CURSOR:TY?→:CURSOR:TY:TYPE OFF;
HORIZONTAL:TRACE 1;POSITION1 3.00;
POSITION2 -3.00;:CURSOR:TY:VERTICAL:
TRACE 1;POSITION1 -4.0000000;
POSITION2 4.0000000;:CURSOR:TY:MARKER:
M1:TRACE 1;POSITION -3.0000000;:CURSOR:
TY:MARKER:M2:TRACE 2;
POSITION -1.0000000;:CURSOR:TY:MARKER:
M3:TRACE 3;POSITION 1.0000000;:CURSOR:
TY:MARKER:M4:TRACE 4;
POSITION 3.0000000;:CURSOR:TY:DEGREE:
TRACE 1;POSITION1 -4.0000000;
POSITION2 4.0000000;
REFERENCE1 -2.0000000;
REFERENCE2 2.0000000;RVALUE 90;:CURSOR:
TY:LOGIC:FORMAT BINARY;ORDER A0_B7
```

## :CURSor:TY:DEGRee?

Function Queries all settings related to angle cursors in the T-Y display.

Syntax
```
:CURSor:TY:DEGree?
```

Example
```
:CURSOR:TY:DEGREE?→:CURSOR:TY:DEGREE:
TRACE 1;POSITION1 -4.0000000;
POSITION2 4.0000000;
REFERENCE1 -2.0000000;
REFERENCE2 2.0000000;RVALUE 90
```

## :CURSor:TY:DEGRee:DX?

Function Queries the angle difference, ΔD, between angle cursors in the T-Y display.

Syntax
```
:CURSor:TY:DEGRee:DX?
```

Example
```
:CURSOR:TY:DEGREE:DX?→:CURSOR:TY:
DEGREE:DX 180.00000E+00
```

## :CURSor:TY:DEGRee:DY?

Function Queries the angle difference, ΔV, between angle cursors in the T-Y display.

Syntax
```
:CURSor:TY:DEGRee:DY?
```

Example
```
:CURSOR:TY:DEGREE:DY?→:CURSOR:TY:
DEGREE:DY 6.2500000E-03
```

## :CURSor:TY:DEGRee:JUMP

Function Moves the angle cursor in the T-Y display to the specified zoomed waveform.

Syntax
```
:CURSor:TY:DEGRee:JUMP {C1_Z1|C1_Z2|
C2_Z1|C2_Z2}
```

Example
```
:CURSOR:TY:DEGREE:JUMP C1_Z1
```

Description The cursor jumps to the center position of the zoomed waveform.

C1 and C2 represent Cursor1 and Cursor2, respectively.

## :CURSor:TY:DEGRee:POSition<x>

Function Sets the position of the angle cursor in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:DEGRee:POSition<x> {<NRf>}
:CURSor:TY:DEGRee:POSition<x>?
<x>=1 or 2
<NRf>=−5 to 5 (10 div/displayed record
      length steps)
```

Example
```
:CURSOR:TY:DEGREE:POSITION1 2
:CURSOR:TY:DEGREE:POSITION1?→:CURSOR:
TY:DEGREE:POSITION1 2.0000000
```

## :CURSor:TY:DEGRee:REFerence<x>

Function Sets the position of the angle reference start point (Reference1) or the end point (Reference2) in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:DEGRee:REFerence<x> {<NRf>}
:CURSor:TY:DEGRee:REFerence<x>?
<x>=1 or 2
<NRf>=–5 to 5 (10 div/displayed record
      length steps)
```

Example
```
:CURSOR:TY:DEGREE:REFERENCE1 -1
:CURSOR:TY:DEGREE:REFERENCE1?→:CURSOR:
TY:DEGREE:REFERENCE1 -1.0000000
```

### :CURsor:TY:DEGRee:RVALue

Function Sets the angle reference width or queries the current setting.

Syntax
```
:CURSor:TY:DEGRee:RVALue {<NRf>}
:CURSor:TY:DEGRee:RVALue?
<NRf>=1 to 720
```

Example
```
:CURSOR:TY:DEGREE:RVALUE 180
:CURSOR:TY:DEGREE:RVALUE?→:CURSOR:TY:
DEGREE:RVALUE 180
```

### :CURSor:TY:DEGRee:TRACe

Function Sets the waveform on which to use the angle cursor in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:DEGRee:TRACe {<NRf>|MATH<x>|
PODA|PODB|PODA_PODB|ALL}
:CURSor:TY:DEGRee:TRACe?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:CURSOR:TY:DEGREE:TRACE 1
:CURSOR:TY:DEGREE:TRACE?→:CURSOR:TY:
DEGREE:TRACE 1
```

### :CURSor:TY:DEGRee:X<x>?

Function Queries the angle at the angle cursor in the T-Y display.

Syntax
```
:CURSor:TY:DEGRee:X<x>?
<x>=1 or 2
```

Example
```
(This is an example in which Cursor1 is
queried.)
:CURSOR:TY:DEGREE:X1?→:CURSOR:TY:
DEGREE:X1 -120.00000E+00
```

### :CURSor:TY:DEGRee:Y<x>?

Function Queries the voltage value at the angle cursor in the T-Y display.

Syntax
```
:CURSor:TY:DEGRee:Y<x>?
<x>=1 or 2
```

Example
```
(This is an example in which Cursor1 is
queried.)
:CURSOR:TY:DEGREE:Y1?→:CURSOR:TY:
DEGREE:Y1 10.000000E-03
```

Description When the specified trace is a logic waveform (option), this query returns a decimal value in which the MSB is bit0 (CH1) and the LSB is bit7 (CH8).

### :CURSor:TY:HORizontal?

Function Queries all settings related to H cursors in the T-Y display.

Syntax
```
:CURSor:TY:HORizontal?
```

Example
```
:CURSOR:TY:HORIZONTAL?→:CURSOR:TY:
HORIZONTAL:TRACE 1;POSITION1 3.00;
POSITION2 -3.00
```

### :CURSor:TY:HORizontal:DY?

Function Queries the Y-axis value between the H cursors in the T-Y display.

Syntax
```
:CURSor:TY:HORizontal:DY?
```

Example
```
:CURSOR:TY:HORIZONTAL:DY?→:CURSOR:TY:
HORIZONTAL:DY 3.0000000E+00
```

Description When the linear scaling is ON, the scaled value is queried.

### :CURSor:TY:HORizontal:POSition<x>

Function Sets the H cursor position in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:HORizontal:POSition<x>
{<NRf>}
:CURSor:TY:HORizontal:POSition<x>?
<x>=1 or 2
<NRf>=-4 to 4div (1/100 steps)
```

Example
```
:CURSOR:TY:HORIZONTAL:POSITION1 2
:CURSOR:TY:HORIZONTAL:POSITION1?→
:CURSOR:TY:HORIZONTAL:POSITION1 2.00
```

Description The <NRf> value is accurate to two digits beyond the decimal.

### :CURSor:TY:HORizontal:TRACe

Function Sets the waveform on which to use the H cursor in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:HORizontal:TRACe {<NRf>|
MATH<x>}
:CURSor:TY:HORizontal:TRACe?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:CURSOR:TY:HORIZONTAL:TRACE 1
:CURSOR:TY:HORIZONTAL:TRACE?→:CURSOR:
TY:HORIZONTAL:TRACE 1
```

### :CURSor:TY:HORizontal:Y<x>?

Function Queries the Y-axis value of the H cursor in the T-Y display.

Syntax
```
:CURSor:TY:HORizontal:Y<x>?
<x>=1 or 2
```

Example
```
:CURSOR:TY:HORIZONTAL:Y1?→:CURSOR:TY:
HORIZONTAL:Y1 -1.5000000E+00
```

Description When the linear scaling is ON, the scaled value is queried.

### :CURSor:TY:LOGic?

Function Sets/queries all settings related to the cursor in the logic T-Y display.

Syntax
```
:CURSor:TY:LOGic?
```

Example
```
:CURSOR:TY:LOGIC:?→:CURSOR:TY:LOGIC:
FORMAT BINARY;ORDER A0_B7
```

## :CURSor:TY:LOGic:FORMat

Function Sets/queries the data format in the logic T-Y display.

Syntax `:CURSor:TY:LOGic:FORMat {BINary|HEXa}`
`:CURSor:TY:LOGic:FORMat?`

Example `:CURSOR:TY:LOGIC:FORMAT BINARY`
`:CURSOR:TY:LOGIC:FORMAT?→:CURSOR:TY:LOGIC:FORMAT BINARY`

## :CURSor:TY:LOGic:ORDer

Function Queries the bit order of logic in the T-Y display.

Syntax `:CURSor:TY:LOGic:ORDer {A0_B7|B7_A0}`
`:CURSor:TY:LOGic:ORDer?`

Example `:CURSOR:TY:LOGIC:ORDER A0_B7`
`:CURSOR:TY:LOGIC:ORDER?→:CURSOR:TY:LOGIC:ORDER A0_B7`

## :CURSor:TY:MARKer?

Function Queries all settings related to marker cursors in the T-Y display.

Syntax `:CURSor:TY:MARKer?`

Example `:CURSOR:TY:MARKER?→:CURSOR:TY:MARKER:M1:TRACE 1;POSITION -3.0000000;:CURSOR:TY:MARKER:M2:TRACE 2;POSITION -1.0000000;:CURSOR:TY:MARKER:M3:TRACE 3;POSITION 1.0000000;:CURSOR:TY:MARKER:M4:TRACE 4;POSITION 3.0000000`

## :CURSor:TY:MARKer:DX<x>?

Function Queries the X-axis value between the marker cursors in the T-Y display.

Syntax `:CURSor:TY:MARKer:DX<x>?`
`<x>=1 to 4`

Example `:CURSOR:TY:MARKER:DX1?→:CURSOR:TY:MARKER:DX1 0.0000000E+00`

Description
- **When the time base is set to internal clock**
  The query is for the time between the markers in the time domain and the frequency between the markers in the frequency domain.
- **When the time base is set to external clock**
  The number of points between the marker cursors is queried.

## :CURSor:TY:MARKer:DY<x>?

Function Queries the physical value of the Y-axis difference between the marker cursors in the T-Y display.

Syntax `:CURSor:TY:MARKer:DY<x>?`
`<x>=1 to 4`

Example `:CURSOR:TY:MARKER:DY2?→:CURSOR:TY:MARKER:DY2 500.00000E-03`

Description When the linear scaling is ON, the scaled value is queried.

## :CURSor:TY:MARKer:M<x>?

Function Queries all settings related to the specified marker cursor.

Syntax `:CURSor:TY:MARKer:M<x>?`
`<x>=1 to 4`

Example `:CURSOR:TY:MARKER:M1?→:CURSOR:TY:MARKER:M1:TRACE 1:POSITION -3.0000000`

## :CURSor:TY:MARKer:M<x>:JUMP

Function Moves the marker cursor in the T-Y display to the specified zoomed waveform.

Syntax `:CURSor:MARKer:M<x>:JUMP {Z1|Z2}`
`<x>=1 to 4`

Example `:CURSOR:TY:MARKER:M1:JUMP Z1`

Description The cursor jumps to the center position of the zoomed waveform.

## :CURSor:TY:MARKer:M<x>:POSition

Function Sets the X-axis position of the marker cursor in the T-Y display or queries the current setting.

Syntax `:CURSor:TY:MARKer:M<x>:POSition {<NRf>}`
`:CURSor:TY:MARKer:M<x>:POSition?`
`<x>=1 to 4`
`<NRf>=–5 to 5 div (10 div/displayed record length steps)`

Example `:CURSOR:TY:MARKER:M1:POSITION -1`
`:CURSOR:TY:MARKER:M1:POSITION?→:CURSOR:TY:MARKER:M1:POSITION -1.0000000`

## :CURSor:TY:MARKer:M<x>:TRACe

Function Sets the waveform on which to use the marker cursor in the T-Y display or queries the current setting.

Syntax `:CURSor:TY:MARKer:M<x>:TRACe {<NRf>|MATH<x>}`
`:CURSor:TY:MARKer:M<x>:TRACe?`
`<x>=1 to 4`
`<NRf>=1 to 4`
`The <x> in MATH<x>=1 or 2`

Example `:CURSOR:TY:MARKER:M1:TRACE1 1`
`:CURSOR:TY:MARKER:M1:TRACE1?→:CURSOR:TY:MARKER:M1:TRACE1 1`

## :CURSor:TY:MARKer:X<x>?

Function Queries the X-axis position of the marker cursor in the T-Y display.

Syntax `:CURSor:TY:MARKer:X<x>?`
`<x>=1 to 4`

Example `:CURSOR:TY:MARKER:X1?→:CURSOR:TY:MARKER:X1 -4.5000E-03`

Description
- **When the time base is set to internal clock**
  The query is for the time in the time domain and frequency in the frequency domain.
- **When the time base is set to external clock**
  The query is for the number of points with respect to delay 0 in the time domain and the number of points from the beginning of the memory in the frequency domain.

## :CURSor:TY:MARKer:Y<x>?

Function Queries the Y-axis position of the marker cursor in the T-Y display.

Syntax `:CURSor:TY:MARKer:Y<x>?`
`<x>=1 to 4`

Example `:CURSOR:TY:MARKER:Y1?→:CURSOR:TY:MARKER:Y1 1.5000E-03`

Description When the linear scaling is ON, the scaled value is queried.

## :CURSor:TY:TYPE

Function Sets the type of cursor in the T-Y display or queries the current setting.

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

Example `:CURSOR:TY:TYPE HORIZONTAL`
`:CURSor:TY:TYPE?→:CURSOR:TY:TYPE HORIZONTAL`

## :CURSor:TY:VERTical?

Function Queries all settings related to V cursors in the T-Y display.

Syntax `:CURSor:TY:VERTical?`

Example `:CURSOR:TY:VERTICAL→:CURSOR:TY:VERTICAL:TRACE 1;POSITION1 -4.0000000;POSITION2 4.0000000`

## :CURSor:TY:VERTical:DX?

Function Queries the X-axis value between V cursors in the T-Y display.

Syntax `:CURSor:TY:VERTical:DX?`

Example `:CURSOR:TY:VERTICAL:DX?→:CURSOR:TY:VERTICAL:DX 2.50E-06`

Description
- **When the time base is set to internal clock**
  The query is for the time between the markers in the time domain and the frequency between the markers in the frequency domain.
- **When the time base is set to external clock**
  The number of points between the marker cursors is queried.

## :CURSor:TY:VERTical:DY?

Function Queries the Y-axis value between V cursors in the T-Y display.

Syntax `:CURSor:TY:VERTical:DY?`

Example `:CURSOR:TY:VERTICAL:DY?→:CURSOR:TY:VERTICAL:DY 1.50E+03`

Description
- When the linear scaling of the Y trace is ON, the scaled value is queried.
- When the specified trace is a logic waveform (option), "NAN(Not A Number)" is returned.

## :CURSor:TY:VERTical:JUMP

Function Moves the V cursor in the T-Y display to the specified zoomed waveform.

Syntax `:CURSor:TY:VERTical:JUMP {C1_Z1|C1_Z2|C2_Z1|C2_Z2}`

Example `:CURSOR:TY:VERTICAL:JUMP C1_Z1`

Description The cursor jumps to the center position of the zoomed waveform.
C1 and C2 represent V cursors, Cursor1 and Cursor2, respectively.

## :CURSor:TY:VERTical:PERDt?

Function Queries the 1/ΔT value between V cursors in the T-Y display.

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

Example `:CURSOR:TY:VERTICAL:PERDT?→:CURSOR:TY:VERTICAL:PERDT 2.50E+06`

Description When the specified trace is in the frequency domain or the trace is a logic waveform (option), "NAN(Not A Number)" is returned.

## :CURSor:TY:VERTical:POSition<x>

Function Sets the V cursor position in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:VERTical:POSition<x> {<NRf>}
:CURSor:TY:VERTical:POSition<x>?
<x>=1 or 2
<NRf>=–5 to 5 div (10 div/displayed
      record length steps)
```

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

## :CURSor:TY:VERTical:TRACe

Function Sets the waveform on which to use the V cursor in the T-Y display or queries the current setting.

Syntax
```
:CURSor:TY:VERTical:TRACe {<NRf>|
MATH<x>|ALL|PODA|PODB|PODA_PODB}
:CURSor:TY:VERTical:TRACe?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:CURSOR:TY:VERTICAL:TRACE 1
:CURSOR:TY:VERTICAL:TRACE?→:CURSOR:TY:
VERTICAL:TRACE 1
```

## :CURSor:TY:VERTical:X<x>?

Function Queries the X-axis position of the V cursor in the T-Y display.

Syntax
```
:CURSor:TY:VERTical:X<x>?
<x>=1 or 2
```

Example
```
:CURSOR:TY:VERTICAL:X1?TY::CURSOR:TY:
VERTICAL:X1 -2.50E-06
```

Description
- **When the time base is set to internal clock**
  The query is for the time in the time domain and frequency in the frequency domain.
- **When the time base is set to external clock**
  The query is for the number of points with respect to delay 0 in the time domain and the number of points from the beginning of the memory in the frequency domain.
- When the specified trace is a logic waveform (option), "NAN(Not A Number)" is returned.

## :CURSor:TY:VERTical:Y<x>?

Function Queries the Y-axis position of the V cursor in the T-Y display.

Syntax
```
:CURSor:TY:VERTical:Y<x>?
<x>=1 or 2
```

Example
```
:CURSOR:TY:VERTICAL:Y1?→:CURSOR:TY:
VERTICAL:Y1 2.50E+03
```

Description When the specified trace is a logic waveform (option), this query returns a decimal value in which the MSB is bit0 (CH1) and the LSB is bit7 (CH8).

## :CURSor:XY?

Function Queries all settings related to the cursors in the X-Y display.

Syntax
```
CURSor:XY?
```

Example
```
CURSOR:XY?→:CURSOR:XY:TYPE OFF;
HORIZONTAL:TRACE XY1;POSITION1 3.00;
POSITION2 -3.00;:CURSOR:XY:VERTICAL:
TRACE XY1;POSITION1 -3.00;
POSITION2 3.00;:CURSOR:XY:MARKER:
TRACE XY1;POSITION -3.0000000
```

## :CURSor:XY:HORizontal?

Function Queries all settings related to the H cursors in the X-Y display.

Syntax
```
:CURSor:XY:HORizontal?
```

Example
```
:CURSOR:XY:HORIZONTAL?→:CURSOR:XY:
HORIZONTAL:TRACE XY1;POSITION1 3.00;
POSITION2 -3.00
```

## :CURSor:XY:HORizontal:DY?

Function Queries the Y-axis value of the H cursor of the X-Y display.

Syntax
```
:CURSor:XY:HORizontal:DY?
```

Example
```
:CURSOR:XY:HORIZONTAL:DY?→:CURSOR:XY:
HORIZONTAL:DY 3.0000000E+00
```

Description When the linear scaling is ON, the scaled value is queried.

## :CURSor:XY:HORizontal:POSition<x>

Function Sets the H cursor position of the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:HORizontal:POSition<x>
{<NRf>}
:CURSor:XY:HORizontal:POSition<x>?
<x>=1 or 2
<NRf>=–4 to 4div (1/100 steps)
```

Example
```
:CURSOR:XY:HORIZONTAL:POSITION1 2
:CURSOR:XY:HORIZONTAL:POSITION1?→
:CURSOR:XY:HORIZONTAL:POSITION1 2.00
```

## :CURSor:XY:HORizontal:TRACe

Function Sets the waveform on which to use the H cursor in the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:HORizontal:TRACe {XY1|XY2}
:CURSor:XY:HORizontal:TRACe?
```

Example
```
:CURSOR:XY:HORIZONTAL:TRACE XY1
:CURSOR:XY:HORIZONTAL:TRACE?→:CURSOR:
XY:HORIZONTAL:TRACE XY1
```

### :CURSor:XY:HORizontal:Y?

Function Queries the Y-axis value of the H cursor of the X-Y display.

Syntax `:CURSor:XY:HORizontal:Y?`

Example
```
:CURSOR:TY:HORIZONTAL:Y?→:CURSOR:TY:
HORIZONTAL:Y -1.5000000E+00
```

Description When the linear scaling is ON, the scaled value is queried.

### :CURSor:XY:MARKer?

Function Queries all settings related to the marker cursors in the X-Y display.

Syntax `:CURSor:XY:MARKer?`

Example
```
:CURSOR:XY:MARKER?→:CURSOR:XY:MARKER:
DISPLAY 1;TRACE XY1;POSITION -3.0000000
```

### :CURSor:XY:MARKer:POSition

Function Sets the X-axis position of the marker cursor in the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:MARKer:POSition {<NRf>}
:CURSor:XY:MARKer:POSition?
<NRf>=–5 to 5 div (10 div/displayed
      record length steps)
```

Example
```
:CURSOR:XY:MARKER:POSITION -1
:CURSOR:XY:MARKER:POSITION?→:CURSOR:XY:
MARKER:POSITION -1.0000000
```

### :CURSor:XY:MARKer:T?

Function Queries the time value of the marker cursor in the X-Y display.

Syntax `CURsor:XY:MARKer:T?`

Example
```
:CURSOR:XY:MARKER:T?→:CURSOR:XY:MARKER:
T 100.00E-03
```

### :CURSor:XY:MARKer:TRACe

Function Sets the waveform on which to use the marker cursor in the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:MARKer:TRACe {XY1|XY2}
:CURSor:XY:MARKer:TRACe?
```

Example
```
:CURSOR:XY:MARKER:TRACE XY1
:CURSOR:XY:MARKER:TRACE?→:CURSOR:XY:
MARKER:TRACE XY1
```

### :CURSor:XY:MARKer:X?

Function Queries the X-axis position of the marker cursor in the X-Y display.

Syntax `:CURSor:XY:MARKer:X?`

Example
```
:CURSOR:XY:MARKER:X1?→:CURSOR:XY:
MARKER:X -4.5000E-03
```

Description When the linear scaling is ON, the scaled value is queried.

### :CURSor:XY:MARKer:Y?

Function Queries the Y-axis position of the marker cursor in the X-Y display.

Syntax `:CURSor:XY:MARKer:Y?`

Example
```
:CURSOR:XY:MARKER:Y?→:CURSOR:XY:
MARKER:Y 1.5000E-03
```

Description When the linear scaling is ON, the scaled value is queried.

### :CURSor:XY:TYPE

Function Sets the cursor type of the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:TYPE {OFF|HORizontal|
VERTical|MARKer}
:CURSor:XY:TYPE?
```

Example
```
:CURSOR:XY:TYPE VERTICAL
:CURSor:XY:TYPE?→:CURSOR:XY:
TYPE VERTICAL
```

### :CURSor:XY:VERTical?

Function Queries all settings related to V cursors in the X-Y display.

Syntax `:CURSor:XY:VERTical?`

Example
```
:CURSOR:XY:VERTICAL→:CURSOR:XY:
VERTICAL:TRACE XY1;POSITION1 -3.00;
POSITION2 3.00
```

### :CURSor:XY:VERTical:DX?

Function Queries the X-axis value between the V cursors in the X-Y display.

Syntax `:CURSor:XY:VERTical:DX?`

Example
```
:CURSOR:XY:VERTICAL:DX?→:CURSOR:XY:
VERTICAL:DX 2.50E-06
```

Description When the linear scaling of the X trace is ON, the scaled value is queried.

### :CURSor:XY:VERTical:POSition<x>

Function Sets the V cursor position of the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:VERTical:POSition<x> {<NRf>}
:CURSor:XY:VERTical:POSition<x>?
<x>=1 or 2
<NRf>=–4 to 4div (1/100 steps)
```

Example
```
:CURSOR:XY:VERTICAL:POSITION1 2
:CURSOR:XY:VERTICAL:POSITION1?→
:CURSOR:XY:VERTICAL:POSITION1 2.00E+00
```

### :CURSor:XY:VERTical:TRACe

Function Sets the waveform on which to use the V cursor in the X-Y display or queries the current setting.

Syntax
```
:CURSor:XY:VERTical:TRACe {XY1|XY2}
:CURSor:XY:VERTical:TRACe?
```

Example
```
:CURSOR:XY:VERTICAL:TRACE
XY1:CURSOR:XY:VERTICAL:TRACE?→:CURSOR:
XY:VERTICAL:TRACE XY1
```

### :CURSor:XY:VERTical:X<x>?

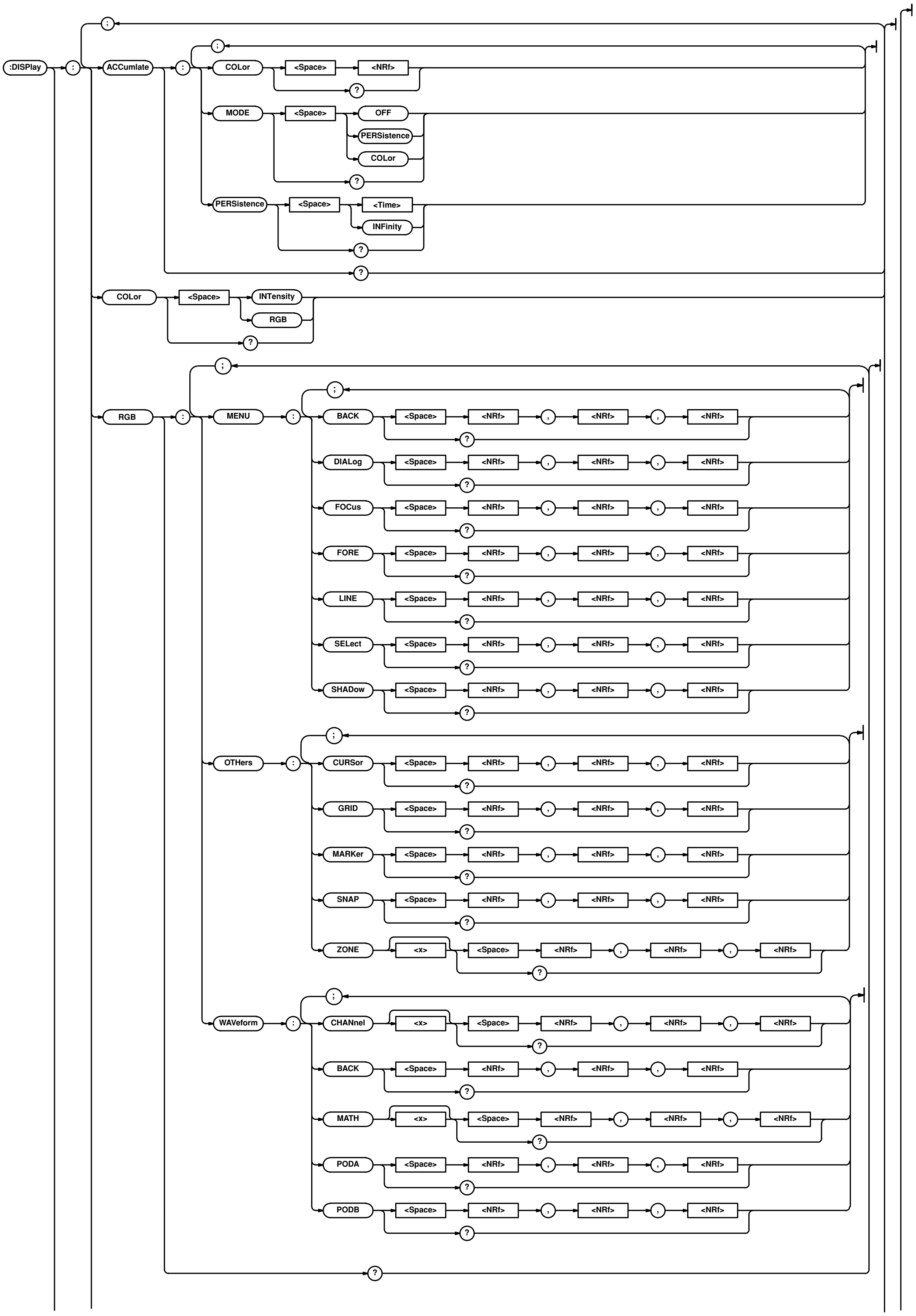
Function Queries the X-axis position of the V cursor in the X-Y display.

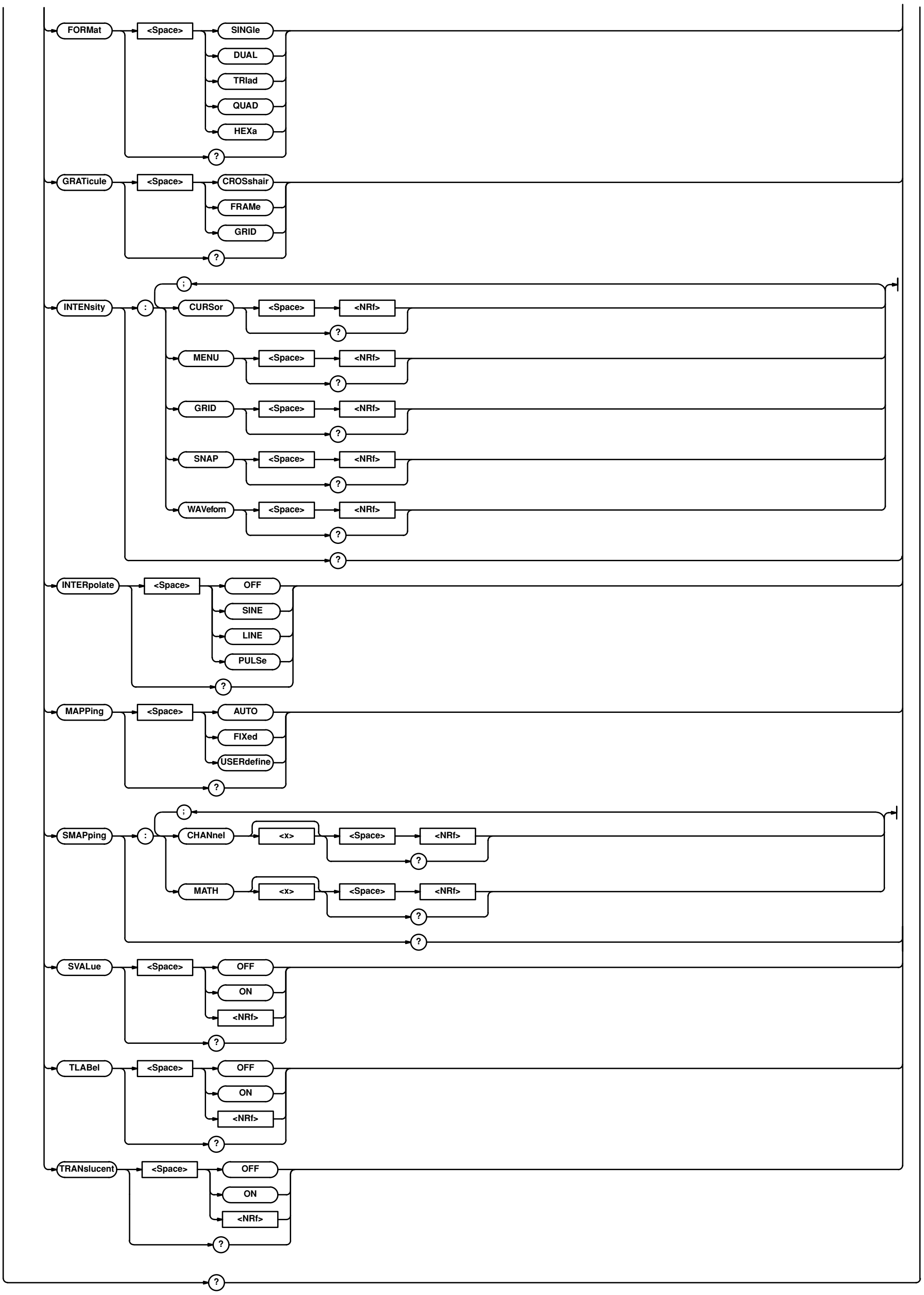
Syntax
```
:CURSor:XY:VERTical:X<x>?
<x>=1 or 2
```

Example
```
:CURSOR:XY:VERTICAL:X1?→:CURSOR:XY:
VERTICAL:X1 -2.50E-06
```

## 5.9 DISPlay Group

The commands in the DISPlay group are used to set or query the display parameters. This allows you to make the same settings that can be made using the DISPLAY and MISC key.




FORMat
<Space>
SINGle
DUAL
TRIad
QUAD
HEXa
?
GRATicule
<Space>
CROSshair
FRAMe
GRID
?
INTENsity
:
;
CURSor
<Space>
<NRf>
?
MENU
<Space>
<NRf>
?
GRID
<Space>
<NRf>
?
SNAP
<Space>
<NRf>
?
WAVeform
<Space>
<NRf>
?
?
INTERpolate
<Space>
OFF
SINE
LINE
PULSe
?
MAPPing
<Space>
AUTO
FIXed
USERdefine
?
SMAPping
:
;
CHANnel
<x>
<Space>
<NRf>
?
MATH
<x>
<Space>
<NRf>
?
?
SVALue
<Space>
OFF
ON
<NRf>
?
TLABel
<Space>
OFF
ON
<NRf>
?
TRANslucent
<Space>
OFF
ON
<NRf>
?
?

## :DISPlay?

Function Queries all display settings.

Syntax
```
:DISPlay?
```
Example
```
:DISPLAY?→:DISPLAY:FORMAT QUAD;
INTERPOLATE SINE;GRATICULE GRID;
SVALUE 0;TLABEL 0;ACCUMLATE:MODE OFF;
PERSISTENCE 100.0E-03;COLOR 16;:
DISPLAY:TRANSLUCENT 0;MAPPING AUTO;
SMAPPING:CHANNEL1 0;CHANNEL2 1;
CHANNEL3 2;CHANNEL4 3;MATH1 4;MATH2 5;:
DISPLAY:COLOR INTENSITY;INTENSITY:
MENU 10;WAVEFORM 10;SNAP 10;GRID 10;
CURSOR 10;:DISPLAY:RGB:MENU:
FORE 15,15,15;BACK 0,0,7;FOCUS 0,0,7;
SELECT 0,7,15;DIALOG 7,7,7;SHADOW 0,0,0;
LINE 15,15,15;:DISPLAY:RGB:WAVEFORM:
CHANNEL1 15,15,0;CHANNEL2 0,15,0;
CHANNEL3 15,0,15;CHANNEL4 0,15,15;
MATH1 15,0,0;MATH2 15,7,0;PODA 5,5,15;
PODB 11,4,15;BACK 0,0,0;:DISPLAY:RGB:
OTHERS:SNAP 15,15,15;ZONE1 0,0,15;
ZONE2 15,4,11;ZONE3 11,15,4;
ZONE4 0,7,15;GRID 8,8,8;CURSOR 13,13,13;
MARKER 15,15,15
```

## :DISPlay:ACCumulate?

Function Queries all accumulation settings.

Syntax
```
:DISPlay:ACCumulate?
```
Example
```
:DISPLAY:ACCUMULATE?→:DISPLAY:
ACCUMULATE:MODE OFF;
PERSISTENCE 100.0E-03;COLOR 16
```

## :DISPlay:ACCumulate:COLor

Function Sets/queries the color grading width.

Syntax
```
:DISPlay:ACCumulate:COLor {<NRf>}
:DISPlay:ACCumulate:COLor?
```
<NRf>=2 to 2048 (in steps of $2^n$)

Example
```
:DISPLAY:ACCUMULATE:COLOR 16
:DISPLAY:ACCUMULATE:COLOR?→:DISPLAY:
ACCUMULATE:COLOR 16
```

## :DISPlay:ACCumulate:MODE

Function Selects/queries the accumulation mode.

Syntax
```
:DISPlay:ACCumulate:MODE {COLor|OFF|
PERSistence}
:DISPlay:ACCumulate:MODE?
```
Example
```
:DISPLAY:ACCUMULATE:MODE PERSISTENCE
:DISPLAY:ACCUMULATE:MODE?→:DISPLAY:
ACCUMULATE:MODE PERSISTENCE
```

## :DISPlay:ACCumulate:PERSistence

Function Sets/queries the accumulation time.

Syntax
```
:DISPlay:ACCumulate:PERSistence {<Time>|
INFinity}
:DISPlay:ACCumulate:PERSistence?
<Time>=100 ms to 50 s (in 1-2-5 steps)
```
Example
```
:DISPLAY:ACCUMULATE:PERSISTENCE 100MS
:DISPLAY:ACCUMULATE:PERSISTENCE?→
:DISPLAY:ACCUMULATE:
PERSISTENCE 100.0E-03
```

## :DISPlay:COLor

Function Sets whether to change the screen color using RGB or brightness or queries the current setting.

Syntax
```
:DISPlay:COLor {INTensity|RGB}
:DISPlay:COLor?
```
Example
```
:DISPLAY:COLOR INTENSITY
:DISPLAY:COLOR?→:DISPLAY:
COLOR INTENSITY
```

## :DISPlay:RGB?

Function Queries all settings related to the screen color.

Syntax
```
:DISPlay:RGB?
```
Example
```
:DISPLAY:RGB?→:DISPLAY:RGB:MENU:
FORE 15,15,15;BACK 0,0,7;FOCUS 0,0,7;
SELECT 0,7,15;DIALOG 7,7,7;SHADOW 0,0,0;
LINE 15,15,15;:DISPLAY:RGB:WAVEFORM:
CHANNEL1 15,15,0;CHANNEL2 0,15,0;
CHANNEL3 15,0,15;CHANNEL4 0,15,15;
MATH1 15,0,0;MATH2 15,7,0;PODA 5,5,15;
PODB 11,4,15;BACK 0,0,0;:DISPLAY:RGB:
OTHERS:SNAP 15,15,15;ZONE1 0,0,15;
ZONE2 15,4,11;ZONE3 11,15,4;
ZONE4 0,7,15;GRID 8,8,8;CURSOR 13,13,13;
MARKER 15,15,15
```

## :DISPlay:RGB:MENU?

Function Queries all settings related to colors used in the menu.

Syntax
```
:DISPlay:RGB:MENU?
```
Example
```
:DISPLAY:RGB:MENU?→:WAVEFORM::DISPLAY:
RGB:MENU:FORE 15,15,15;BACK 0,0,7;
FOCUS 0,0,7;SELECT 0,7,15;DIALOG 7,7,7;
SHADOW 0,0,0;LINE 15,15,15
```

### :DISPlay:RGB:MENU:{BACK|DIALog|FOCus|FORE|LINE|SELect|SHADow}

Function Sets the color of the characters, background, focus, select, dialog, shade, and lines of the menu screen or queries the current setting.

Syntax
```
:DISPlay:RGB:MENU:{BACK|DIALog|FOCus|
FORE|LINE|SELect|SHADow} {<NRf>,<NRf>,
<NRf>}
:DISPlay:RGB:MENU:{BACK|DIALog|FOCus|
FORE|LINE|SELect|SHADow}?
<NRf>=0 to 15 (R, G, B, in that order)
```

Example
```
:DISPlay:RGB:MENU:BACK 15,15,15
:DISPLAY:RGB:MENU:BACK?→:DISPLAY:
RGB:MENU:BACK 15,15,15
```

### :DISPlay:RGB:OTHers?

Function Queries all settings related to the colors of other items.

Syntax
```
:DISPlay:RGB:OTHers?
```

Example
```
:DISPLAY:RGB:OTHERS?→:DISPLAY:RGB:
OTHERS:SNAP 15,15,15;ZONE1 0,0,15;
ZONE2 15,4,11;ZONE3 11,15,4;
ZONE4 0,7,15;GRID 8,8,8;CURSOR 13,13,13;
MARKER 15,15,15
```

### :DISPlay:RGB:OTHers:{CURSor|GRID|MARKer|SNAP|ZONE<x>}

Function Sets the color for cursors, grid, markers, and snapshot waveforms or queries the current setting.

Syntax
```
:DISPlay:RGB:OTHers:{CURSor|GRID|MARKer|
SNAP|ZONE<x>} {<NRf>,<NRf>,<NRf>}
:DISPlay:RGB:OTHers:{CURSor|GRID|MARKer|
SNAP|ZONE<x>}?
<x>=1 or 2
<NRf>=0 to 15 (R, G, B, in that order)
```

Example
```
:DISPLAY:RGB:OTHERS:CURSOR 0,0,0
:DISPLAY:RGB:OTHERS:CURSOR?→:DISPLAY:
RGB:OTHERS:CURSOR 0,0,0
```

### :DISPlay:RGB:WAVeform?

Function Queries all settings related to the waveform colors.

Syntax
```
:DISPlay:RGB:WAVeform?
```

Example
```
:DISPLAY:RGB:WAVEFORM?→:DISPLAY:RGB:
WAVEFORM:CHANNEL1 15,15,0;
CHANNEL2 0,15,0;CHANNEL3 15,0,15;
CHANNEL4 0,15,15;MATH1 15,0,0;
MATH2 15,7,0;PODA 5,5,15;PODB 11,4,15;
BACK 0,0,0
```

### :DISPlay:RGB:WAVeform:{CHANnel<x>|BACK|MATH<x>|PODA|PODB}

Function Sets the colors for waveform, computation, and logic channels and the background or queries the current setting.

Syntax
```
:DISPlay:RGB:WAVeform:{CHANnel<x>|BACK|
MATH<x>|PODA|PODB} {<NRf>,<NRf>,<NRf>}
:DISPlay:RGB:WAVeform:{CHANnel<x>|BACK|
MATH<x>|PODA|PODB}
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<NRf>=0 to 15 (R, G, B, in that order)
```

Example
```
:DISPLAY:RGB:WAVEFORM:CHANNEL1 0,0,0
:DISPLAY:RGB:WAVEFORM:CHANNEL1?→
:DISPLAY:RGB:WAVEFORM:CHANNEL1 0,0,0
```

### :DISPlay:FORMat

Function Sets/queries the display format.

Syntax
```
:DISPlay:FORMat {SINGle|DUAL|TRIad|QUAD|
HEXa}
:DISPlay:FORMat?
```

Example
```
:DISPLAY:FORMAT QUAD
:DISPLAY:FORMAT?→:DISPLAY:FORMAT QUAD
```

### :DISPlay:GRATicule

Function Sets/queries the graticule.

Syntax
```
:DISPlay:GRATicule {CROSshair|FRAMe|
GRID}
:DISPlay:GRATicule?
```

Example
```
:DISPLAY:GRATICULE GRID
:DISPLAY:GRATICULE?→:DISPLAY:
GRATICULE GRID
```

### :DISPlay:INTENsity?

Function Queries all settings related to the brightness of the displayed items.

Syntax
```
:DISPlay:INTENsity?
```

Example
```
:DISPLAY:INTENSITY?→:DISPLAY:INTENSITY:
MENU 10;WAVeform 10;SNAP 10;GRID 10;
CURSOR 10
```

### :DISPlay:INTENsity:{CURSor|MENU|GRID|SNAP|WAVeform}

Function Sets the brightness of each display item or queries the current setting.

Syntax
```
:DISPlay:INTENsity:{CURSor|MENU|GRID|
SNAP|WAVeform} {<NRf>}
:DISPlay:INTENsity:{CURSor|MENU|GRID|
SNAP|WAVeform}?
<NRf>=1 to 15
```

Example
```
(The following example is for the
cursor.)
:DISPLAY:INTENSITY:CURSOR 6
:DISPLAY:INTENSITY:CURSOR?→:DISPLAY:
INTENSITY:CURSOR 6
```

## :DISPlay:INTERpolate

Function Selects/queries the interpolation method.

Syntax
```
:DISPlay:INTerpolate {LINE|OFF|PULSe|
SINE}
:DISPlay:INTerpolate?
```
Example
```
:DISPLAY:INTERPOLATE LINE
:DISPLAY:INTERPOLATE?→:DISPLAY:
INTERPOLATE LINE
```

## :DISPlay:MAPPing

Function Sets/queries the mapping mode.

Syntax
```
:DISPlay:MAPPing {AUTO|FIXed|USERdefine}
:DISPlay:MAPPing?
```
Example
```
:DISPLAY:MAPPING AUTO
:DISPLAY:MAPPING?→:DISPLAY:MAPPING AUTO
```

## :DISPlay:SMAPping?

Function Queries all settings relating to the assignment of the waveforms to the split window.

Syntax
```
:DISPlay:SMAPping?
```
Example
```
:DISPLAY:SMAPPING?→:DISPLAY:SMAPPING:
CHANNEL1 0;CHANNEL2 1;CHANNEL3 2;
CHANNEL4 3;MATH1 4;MATH2 5
```

## :DISPlay:SMAPping:{CHANnel<x>|MATH<x>} (Set Mapping)

Function Sets/queries the assignment of the waveforms to the split window.

Syntax
```
:DISPlay:SMAPping:{CHANnel<x>|MATH<x>}
{<NRf>}
:DISPlay:SMAPping:{CHANnel<x>|MATH<x>}?
CHANnel<x>=1 to 4
<x> of MATH<x>=1, 2
<NRf>=0 to 5
```
Example
```
(Below is an example of CH1.)
:DISPLAY:SMAPPING:CHANNEL1
0:DISPLAY:SMAPPING:CHANNEL1?→:DISPLAY:
SMAPPING:CHANNEL1 0
```

## :DISPlay:SVALue (Scale VALUE)

Function Selects/queries whether scaling values are displayed.

Syntax
```
:DISPlay:SVALue {<Boolean>}
:DISPlay:SVALue?
```
Example
```
:DISPLAY:SVALUE OFF
:DISPLAY:SVALUE?→:DISPLAY:SVALUE 0
```

## :DISPlay:TLABel (Trace LABEL)

Function Selects/queries whether waveform labels are displayed.

Syntax
```
:DISPlay:TLABel {<Boolean>}
:DISPlay:TLABel?
```
Example
```
:DISPLAY:TLABEL ON
:DISPLAY:TLABEL?→:DISPLAY:TLABEL 1
```
Description You can set a user defined waveform label using the ":CHANnel<x>:LABel" command.

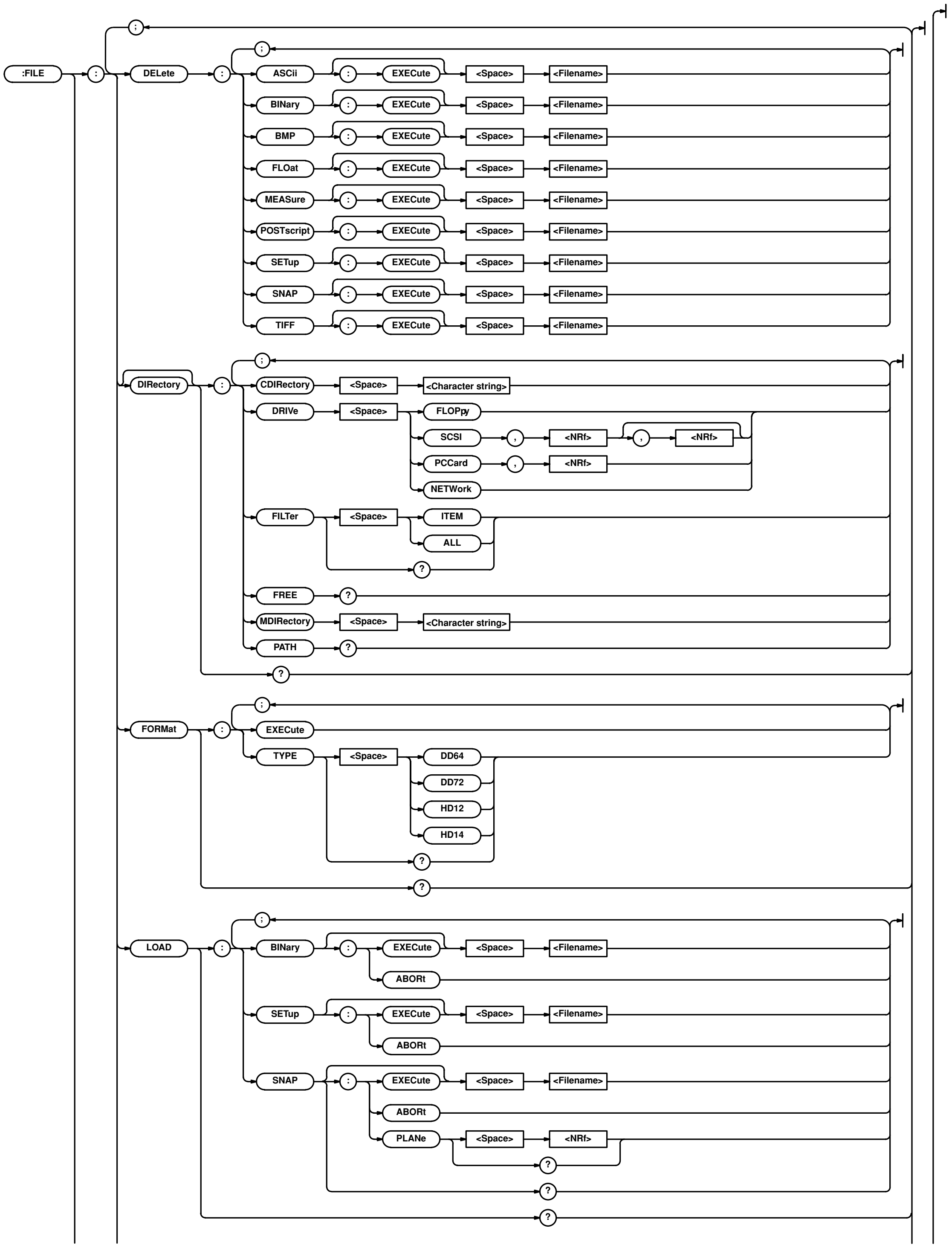
## :DISPlay:TRANslucent

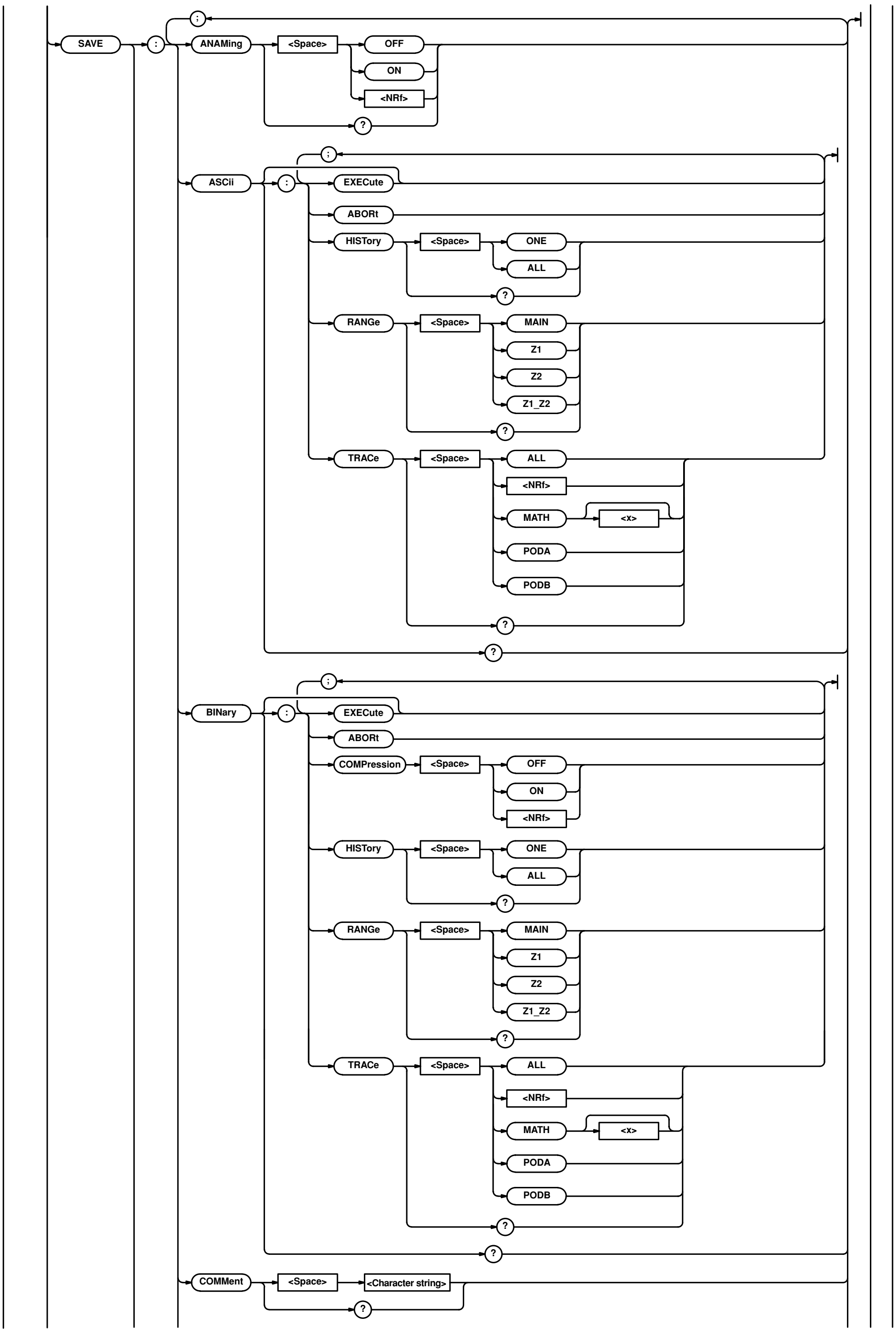
Function Sets/queries the ON/OFF condition of the translucent mode of the pop-up menu.

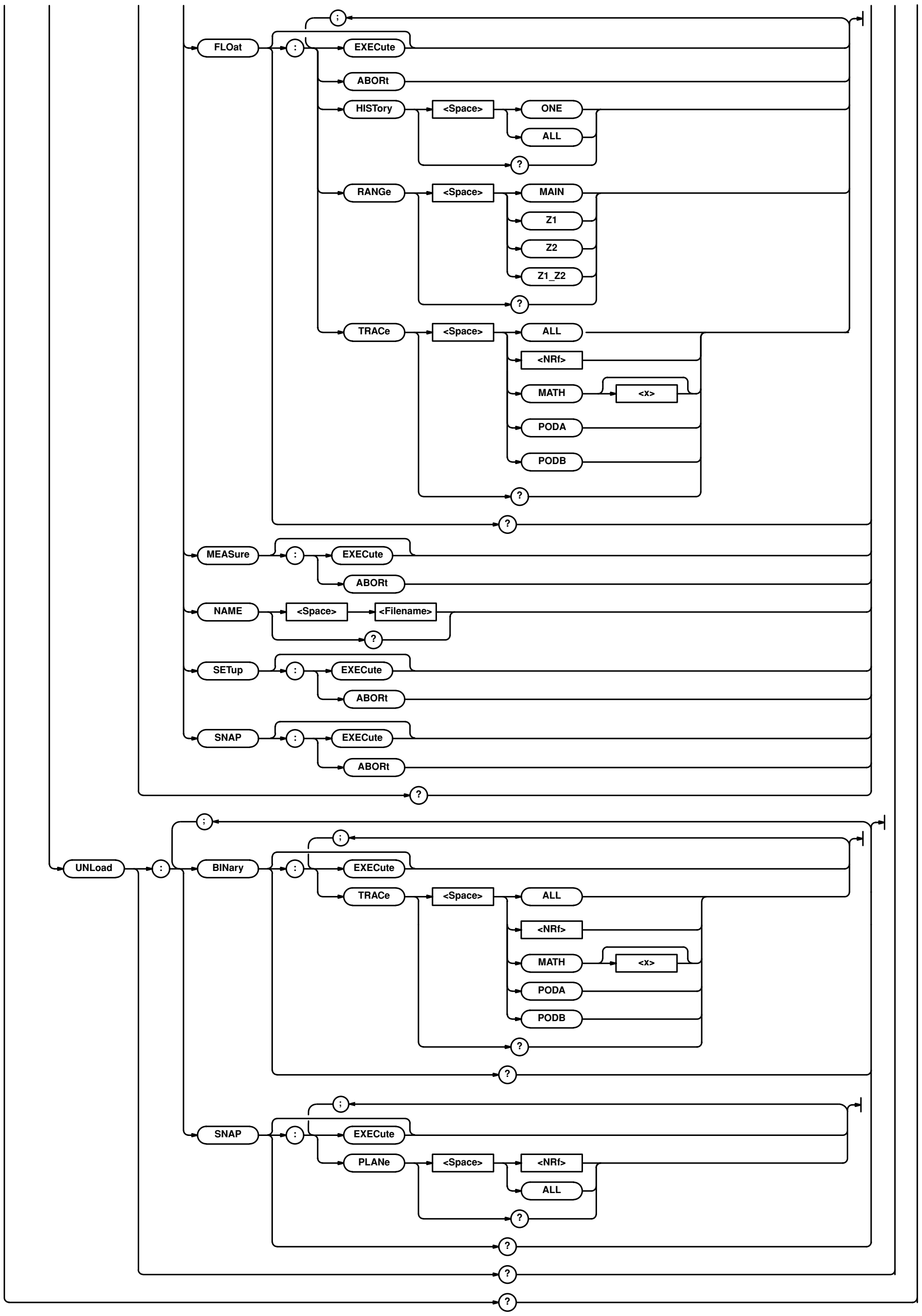
Syntax
```
:DISPlay:TRANslucent {<Boolean>}
:DISPlay:TRANslucent?
```
Example
```
:DISPLAY:TRANSLUCEENT ON
:DISPLAY:TRANSLUCEENT?→:DISPLAY:
TRANSLUCEENT 1
```

## 5.10 FILE Group

These commands control operation of floppy disks, MO disks, and external SCSI devices. This allows you to make the same settings and queries that can be made using the FILE key.




SAVE
:
;
ANAMing
<Space>
OFF
ON
<NRf>
?
ASCii
:
;
EXECute
ABORt
HISTory
<Space>
ONE
ALL
?
RANGe
<Space>
MAIN
Z1
Z2
Z1_Z2
?
TRACe
<Space>
ALL
<NRf>
MATH
<x>
PODA
PODB
?
?
BINary
:
;
EXECute
ABORt
COMPression
<Space>
OFF
ON
<NRf>
HISTory
<Space>
ONE
ALL
?
RANGe
<Space>
MAIN
Z1
Z2
Z1_Z2
?
TRACe
<Space>
ALL
<NRf>
MATH
<x>
PODA
PODB
?
?
COMMent
<Space>
<Character string>
?


FLOat
:
;
EXECute
ABORt
HIStory
<Space>
ONE
ALL
?
RANGe
<Space>
MAIN
Z1
Z2
Z1_Z2
?
TRACe
<Space>
ALL
<NRf>
MATH
<x>
PODA
PODB
?
?
MEASure
:
EXECute
ABORt
NAME
<Space>
<Filename>
?
SETup
:
EXECute
ABORt
SNAP
:
EXECute
ABORt
?
UNLoad
:
;
BINary
:
;
EXECute
TRACe
<Space>
ALL
<NRf>
MATH
<x>
PODA
PODB
?
?
SNAP
:
;
EXECute
PLANe
<Space>
<NRf>
ALL
?
?
?
?

## :FILE?

Function Queries the settings for the specified medium.

Syntax `:FILE?`

Example `:FILE?→:FILE:DIRECTORY:FILTER ITEM;:FILE:SAVE:BINARY:TRACE 1;RANGE MAIN;HISTORY ONE;COMPRESSION 0;:FILE:ASCII:TRACE 1;RANGE MAIN;HISTORY ONE;:FILE:FLOAT:TRACE 1:RANGE MAIN;HISTORY ONE;:SAVE:ANAMING 0;:NAME "SAVED1";:COMMENT "TEST1";:FILE:LOAD:SNAP:PLANE 1;:FILE:UNLOAD:BINARY:TRACE 1;FILE:UNLOAD:SNAP:PLANE 1;:FILE:COPY:DISTINATION:DRIVE FLOPPY;:FILE:FORMAT:TYPE HD14`

## :FILE:DELete:{ASCii|BINary|BMP|FLOat|MEASure|POSTscript|SETup|SNAP|TIFF}:[EXECute]

Function Deletes various types of data files. This is an overlap command.

Syntax `:FILE:DELete:{ASCii|BINary|BMP|FLOat|MEASure|POSTscript|SETup|SNAP|TIFF}:[EXECute] <Filename>`

Example (The following example is for the setup data.)
`:FILE:DELete:SETUP "COPYED1"`

Description Select the medium using the "`:FILE:DIRectory:DRIVe`" command.

## :FILE:DIRectory?

Function Queries all settings related to the current medium.

Syntax `:FILE:DIRectory?`

Example `:FILE:DIRECTORY?→:FILE:DIRECTORY:FILTER ITEM`

## :FILE[:DIRectory]:CDIRectory (Change Directory)

Function Changes the current directory.

Syntax `:FILE[:DIRectory]:CDIRectory <Character string>`

Example `:FILE[:DIRECTORY]:CDIRECTORY "NO_1"`

## :FILE[:DIRectory]:DRIVe

Function Selects the current medium.

Syntax `:FILE[:DIRectory]:DRIVe {FLOPpy|SCSI,<NRf>[,<NRf>]|PCCard,<NRf>|NETWork}`
The first <NRf>=SCSI address (0 to 7)
The second <NRf>=Partitions (0 to 4)
PCCard's <NRf>=Partitions (0 to 4)

Example `:FILE[:DIRECTORY]:DRIVE SCSI,0,2`

Description If the drive does not contain partitions, omit the second <NRf>. "`PC Card`" can only be set for models with the PC Card option installed. "`Network`" can be set on a DL7200 or on a DL7100 with firmware version 2.01 and later.

## :FILE[:DIRectory]:FILTer

Function Sets a filter on the current file or queries the current setting.

Syntax `:FILE[:DIRectory]:FILTer {ITEM|ALL}`
`:FILE[:DIRectory]:FILTer?`

Example `:FILE[:DIRECTORY]:FILTER ITEM`
`:FILE[:DIRECTORY]:FILTER?→:FILE[:DIRECTORY]:FILTER ITEM`

## :FILE[:DIRectory]:FREE?

Function Queries the free space (number of bytes) on the current medium.

Syntax `:FILE[:DIRectory]:FREE?`

Example `:FILE:DIRECTORY:FREE?→:FILE:DIRECTORY:FREE 65536`

## :FILE[:DIRectory]:MDIRectory (Make Directory)

Function Creates a directory in the current directory. This is an overlap command.

Syntax `:FILE[:DIRectory]:MDIRectory <Character string>`

Example `:FILE:DIRECTORY:MDIRECTORY "NO_1"`

## :FILE[:DIRectory]:PATH?

Function Queries the current path.

Syntax `:FILE[:DIRectory]:PATH?`

Example `:FILE[:DIRECTORY]:PATH?→:FILE[:DIRECTORY]:PATH "PATH = FD0"`

## :FILE:FORMat?

Function Queries all settings related to the formatting of the floppy disk.

Syntax `:FILE:FORMat?`

Example `:FILE:FORMAT?→:FILE:FORMAT:TYPE HD14`

### :FILE:FORMat:EXECute

Function Formats the floppy disk. This is an overlap command.

Syntax `:FILE:FORMat:EXECute`

Example `:FILE:FORMAT:EXECUTE`

Description Only the medium that is selected with the "`FILE:FORMat:DRIVe`" command is formatted.

### :FILE:FORMat:TYPE

Function Sets the format type for the floppy disk or queries the current setting.

Syntax
```
:FILE:FORMat:TYPE {DD64|DD72|HD12|HD14}
:FILE:FORMat:TYPE?
```

Example
```
:FILE:FORMAT:TYPE HD12
:FILE:FORMAT:TYPE?→:FILE:FORMAT:
TYPE HD12
```

### :FILE:LOAD?

Function Queries all settings related to file loading.

Syntax `:FILE:LOAD?`

Example `:FILE:LOAD?→:FILE:LOAD:SNAP:PLANE 1`

### :FILE:LOAD:{BINary|SETup|SNAP}:ABORt

Function Aborts loading the data.

Syntax `:FILE:LOAD:{BINary|SETup|SNAP}:ABORt`

Example
```
(The following example is for the setup
data.)
:FILE:LOAD:SETUP:ABORT
```

### :FILE:LOAD:{BINary|SETup|SNAP}[:EXECute]

Function Loads the data. This is an overlap command.

Syntax
```
:FILE:LOAD:{BINary|SETup|SNAP}[:EXECute]
<Filename>
```

Example
```
(The following example is for the setup
data.)
:FILE:LOAD:SETUP:EXECUTE "CASE1"
```

### :FILE:LOAD:PLANe

Function Sets the plane number of the snapshot waveform data or queries the current setting.

Syntax
```
:FILE:LOAD:SNAP:PLANe {<NRf>}
:FILE:LOAD:SNAP:PLANe?
<NRf>=1 to 4
```

Example
```
:FILE:LOAD:SNAP:PLANE 1
:FILE:LOAD:SNAP:PLANE?→:FILE:LOAD:SNAP:
PLANE 1
```

Description This query is the same as the "`:FILE:LOAD?`" and "`:FILE:LOAD:SNAP?`" queries.

### :FILE:LOAD:SNAP?

Function Queries all settings related to the loading of the snapshot waveform data.

Syntax `:FILE:LOAD:SNAP?`

Example
```
:FILE:LOAD:SNAP?→:FILE:LOAD:SNAP:
PLANE 1
```

### :FILE:SAVE?

Function Queries all settings related to file saving.

Syntax `:FILE:SAVE?`

Example
```
:FILE:SAVE?→:FILE:SAVE:BINARY:TRACE 1;
RANGE MAIN;HISTORY ONE;COMPRESSION ON;:
FILE:ASCII:TRACE 1;RANGE MAIN;
HISTORY ONE;:FILE:FLOAT:TRACE 1:
RANGE MAIN;HISTORY ONE;:SAVE:
ANAMING OFF;:NAME "SAVED1";:
COMMENT "TEST1"
```

### :FILE:SAVE:ANAMing

Function Turns the auto naming function of files ON/OFF or queries the current setting.

Syntax
```
:FILE:SAVE:ANAMing  {<Boolean>}
:FILE:SAVE:ANAMing?
```

Example
```
:FILE:SAVE:ANAMING ON
:FILE:SAVE:ANAMING?→:FILE:SAVE:
ANAMING 1
```

### :FILE:SAVE:{ASCii|BINary|FLOat}?

Function Queries all settings related to various types of data.

Syntax `:FILE:SAVE:{ASCii|BINary|FLOat}?`

Example
```
:FILE:SAVE:BINARY?→:FILE:SAVE:BINARY:
TRACE 1;RANGE MAIN;HISTORY ONE;
COMPRESSION ON
```

### :FILE:SAVE:{ASCii|BINary|FLOat|MEASure|SETup|SNAP}:ABORt

Function Aborts saving the data.

Syntax
```
:FILE:SAVE:{ASCii|BINary|FLOat|MEASure|
SETup|SNAP}:ABORt
```

Example
```
(The following example is for the
waveform data.)
:FILE:SAVE:BINARY:ABORT
```

### :FILE:SAVE:{ASCii|BINary|FLOat|MEASure|SETup|SNAP}[:EXECute]

Function Saves the data. This is an overlap command.

Syntax
```
:FILE:SAVE:{ASCii|BINary|FLOat|MEASure|
SETup|SNAP}[:EXECute]
```

Example
```
(The following example is for the
waveform data.)
:FILE:SAVE:BINARY:EXECUTE
```

### :FILE:SAVE:{ASCii|BINary|FLOat}:HISTory

Function Sets the number of histories to save or queries the current setting.

Syntax
```
:FILE:SAVE:{ASCii|BINary|FLOat}:HISTory
{ONE|ALL}
:FILE:SAVE:{ASCii|BINary|FLOat}:HISTory?
```

Example
```
:FILE:SAVE:BINARY:HISTORY ALL
:FILE:SAVE:BINARY:HISTORY?→:FILE:SAVE:
BINARY:HISTORY ALL
```

### :FILE:SAVE:{ASCii|BINary|FLOat}:RANGe

Function Sets the range of data to be saved or queries the current setting.

Syntax
```
:FILE:SAVE:{ASCii|BINary|FLOat}:RANGe
{MAIN|Z1|Z2|Z1_Z2}
:FILE:SAVE:{ASCii|BINary|FLOat}:RANGe?
```

Example
```
:FILE:SAVE:BINARY:RANGE MAIN
:FILE:SAVE:BINARY:RANGE?→:FILE:SAVE:
BINARY:RANGE MAIN
```

### :FILE:SAVE:{ASCii|BINary|FLOat}:TRACe

Function Sets the waveform to be saved or queries the current setting.

Syntax
```
:FILE:SAVE:{ASCii|BINary|FLOat}:TRACe
{ALL|<NRf>|MATH<x>}
:FILE:SAVE:{ASCii|BINary|FLOat}:TRACe?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:FILE:SAVE:ASCII:TRACE 1
:FILE:SAVE:ASCII:TRACE?→:FILE:SAVE:
ASCII:TRACE 1
```

### :FILE:SAVE:BINary:COMPression

Function Sets data compression on the waveform to be saved or queries the current setting.

Syntax
```
:FILE:SAVE:BINary:COMPression
{<Boolean>}
:FILE:SAVE:BINary:COMPression?
```

Example
```
:FILE:SAVE:BINARY:COMPRESSION ON
:FILE:SAVE:BINARY:COMPRESSION?→:FILE:
SAVE:BINARY:COMPRESSION 1
```

Description
- The range of data that is saved is the Main screen.
- Only binary data can be compressed.

### :FILE:SAVE:COMMent

Function Sets comments or queries the current setting.

Syntax
```
:FILE:SAVE:COMMent <Character string>
:FILE:SAVE:COMMent?
<Character string>=Up to 25 characters
```

Example
```
:FILE:SAVE:
COMMENT "THIS IS TEST.       "
:FILE:SAVE:COMMENT?→:FILE:SAVE:
COMMENT "THIS IS TEST.       "
```

Description Only the characters and symbols on the keyboard that appear on the instrument's screen can be used.
The "Ω" and "μ" symbols are represented by ASCII codes "1EH" and "1FH," respectively.

### :FILE:SAVE:NAME

Function Sets the name of the file to be saved or queries the current setting.

Syntax
```
:FILE:SAVE:NAME <Filename>
:FILE:SAVE:NAME?
```

Example
```
:FILE:SAVE:NAME "CASE1"
:FILE:SAVE:NAME?→:FILE:SAVE:
NAME "CASE1"
```

### :FILE:UNLoad?

Function Queries all settings related to clearing the loaded data.

Syntax
```
:FILE:UNLoad?
```

Example
```
:FILE:UNLOAD?→:FILE:UNLOAD:BINARY:
TRACE 1;SNAP:PLANE 1
```

### :FILE:UNLoad:{BINary|SNAP}?

Function Queries all settings related to clearing of loaded data.

Syntax
```
:FILE:UNLoad:{BINary|SNAP}?
```

Example
```
:FILE:UNLOAD:BINARY?→:FILE:UNLOAD:
BINARY:TRACE 1
```

### :FILE:UNLoad:{BINary|SNAP}[:EXECute]

Function Clears the loaded data. This is an overlap command.

Syntax
```
:FILE:UNLoad:{BINary|SNAP}[:EXECute]
```

Example
```
:FILE:UNLOAD:BINARY:EXECUTE
```

### :FILE:UNLoad:BINary:TRACe

Function Sets the waveform to be cleared from the loaded binary data or queries the current setting.

Syntax
```
:FILE:UNLoad:BINary:TRACe {ALL|<NRf>|
MATH<x>|PODA|PODB}
:FILE:UNLoad:BINary:TRACe?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:FILE:UNLOAD:BINARY:TRACE 1
:FILE:UNLOAD:BINARY:TRACE?→:FILE:
UNLOAD:BINARY:TRACE 1
```

### :FILE:UNLoad:SNAP:PLANe

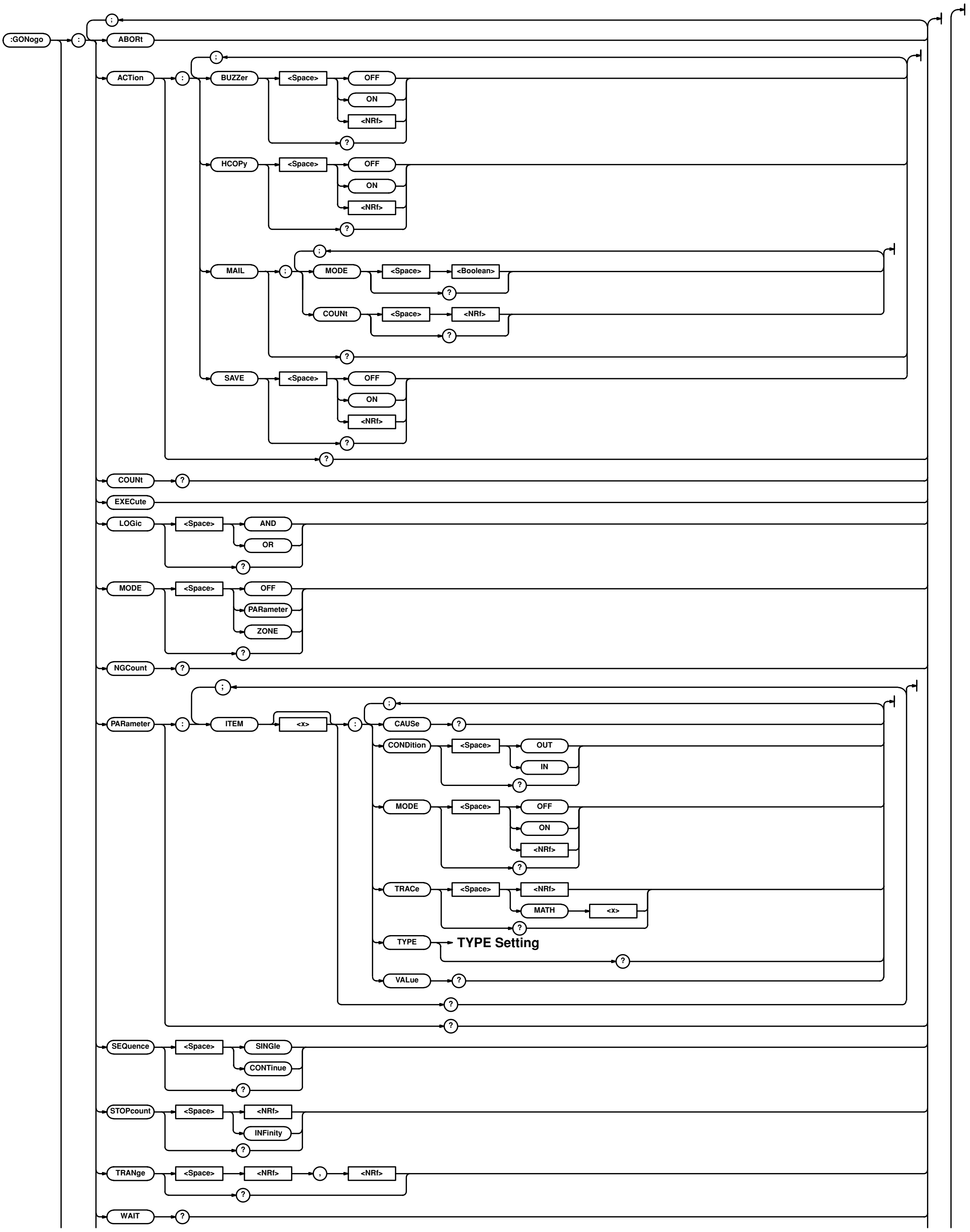
Function Sets the plane number to be cleared from the loaded snapshot data or queries the current setting.

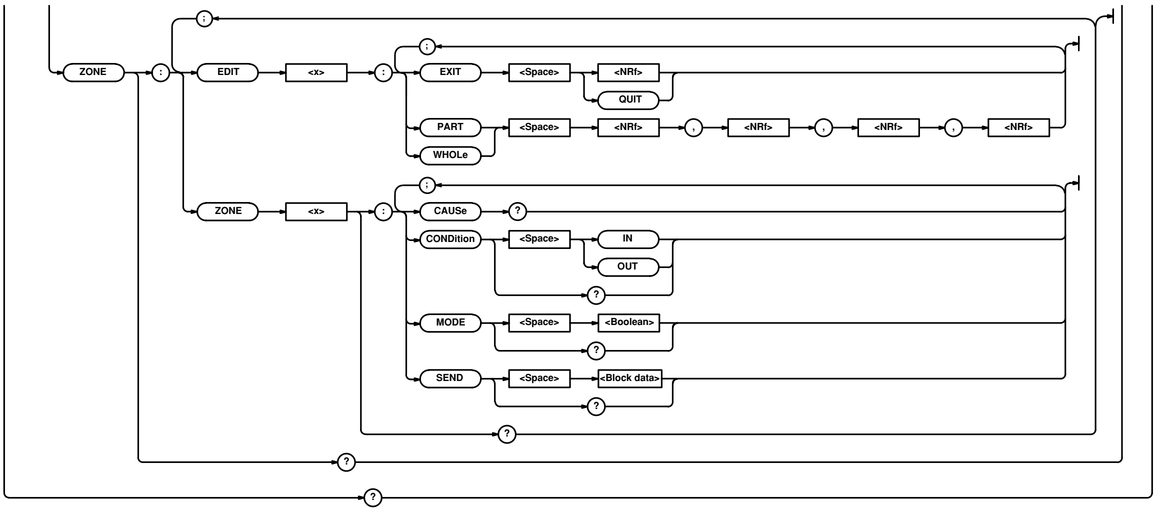
Syntax
```
:FILE:UNLoad:SNAP:PLANe {ALL|<NRf>}
:FILE:UNLoad:SNAP:PLANe?
<NRf>=1 to 4
```

Example
```
:FILE:UNLOAD:SNAP:PLANE ALL
:FILE:UNLOAD:SNAP:PLANE?→:FILE:UNLOAD:
SNAP:PLANE ALL
```

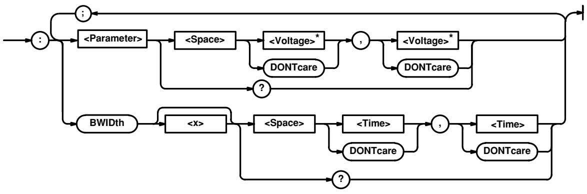
## 5.11 GONogo Group

The commands in the GONogo group deal with the GO/NO-GO determination. These commands can be used to make the same settings and inquiries as when the GO/NOGO key on the front panel is pressed.





TYPE Setting



* The data value is (1) <voltage> when <parameter> is AVERage, HIGH, LOW, MAXimum, MINimum, PTOPeak, RMS, or SDEViation (however, the value is <NRf> when the parameter is being linearly scaled), (2) <time> when <parameter> is AVGPeriod, BWIDth1, BWIDth2, DELay, FALL, NWIDth, PERiod, PWIDth, or RISE, (3) <frequency> when <parameter> is AVGFreq or FREQuency, and (4) <NRf> when <parameter> is DUTYcycle, NOVershoot, PNUMber, POVershoot, TY1Integ, TY2Integ, XY1Integ, or XY2Integ.

## :GONogo?

Function Queries all settings related to GO/NO-GO determination.

Syntax `:GONogo?`

Example
```
:GONOGO?→:GONOGO:MODE OFF;PARAMETER:
ITEM1:MODE 0;TRACE 1;TYPE:
MAXIMUM DONTCARE,DONTCARE;:GONOGO:
PARAMETER:ITEM1:CONDITION OUT;:GONOGO:
PARAMETER:ITEM2:MODE 0;TRACE 1;TYPE:
MINIMUM DONTCARE,DONTCARE;:GONOGO:
PARAMETER:ITEM2:CONDITION OUT;:GONOGO:
PARAMETER:ITEM3:MODE 0;TRACE 1;TYPE:
RMS DONTCARE,DONTCARE;:GONOGO:PARAMETER:
ITEM3:CONDITION OUT;:GONOGO:PARAMETER:
ITEM4:MODE 0;TRACE 1;TYPE:
AVERAGE DONTCARE,DONTCARE;HIGH DONTCARE,
DONTCARE;:GONOGO:PARAMETER:ITEM4:
CONDITION OUT;:GONOGO:ZONE:ZONE1:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE2:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE3:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE4:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE5:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE6:
MODE 0;CONDITION OUT;:GONOGO:LOGIC AND;
SEQUENCE SINGLE;STOPCOUNT INFINITE;
ACTION:SAVE 0;HCOPY 0;BUZZER 0;:GONOGO:
TRANGE -5.0000000,5.0000000
```

## :GONogo:ABORt

Function Aborts GO/NO-GO determination.

Syntax `:GONogo:ABORt`

Example `:GONOGO:ABORT`

## :GONogo:ACTion?

Function Queries all settings related to the action to be performed when the result of the determination is NO-GO and also the determination criteria.

Syntax `:GONogo:ACTion?`

Example
```
:GONOGO:ACTION?→:GONOGO:ACTION:
BUZZER 0;HCOPY 0;SAVE 0
```

## :GONogo:ACTion:BUZZer

Function Sets whether or not to sound an alarm when the determination is NO-GO or queries the current setting.

Syntax
```
:GONogo:ACTion:BUZZer {<Boolean>}
:GONogo:ACTion:BUZZer?
```

Example
```
:GONOGO:ACTION:BUZZER OFF
:GONOGO:ACTION:BUZZER?→:GONOGO:ACTION:
BUZZER 0
```

### :GONogo:ACTion:HCOPy (Hard COPY)

Function Sets whether or not to make a hard copy to the built-in printer (option) when the determination results in NO-GO or queries the current setting.

```
Syntax      :GONogo:ACTion:HCOPy {<Boolean>}
            :GONogo:ACTion:HCOPy?
Example     :GONOGO:ACTION:HCOPY OFF
            :GONOGO:ACTION:HCOPY?→:GONOGO:ACTION:
            HCOPY 0
```

### :GONogo:ACTion:MAIL?

Function Queries the all settings  for  an e-mail to be sent when the determination is NO-GO.

```
Syntax      :GONogo:ACTion:MAIL?
Example     :GONOGO:ACTION:MAIL?→:GONOGO:
            ACTION:MAIL:MODE 1;COUNT 100
```

Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

### :GONogo:ACTion:MAIL:COUNt

Function Sets/queries the maximum number of e-mail to be sent when the determination is NO-GO.

```
Syntax      :GONogo:ACTion:MAIL:COUNt {<NRf>}
            :GONogo:ACTion:MAIL:COUNt?
            <NRf>=1 to 1000
Example     :GONOGO:ACTION:MAIL:COUNT 100
            :GONOGO:ACTION:MAIL:COUNT?→:GONOGO:
            ACTION:MAIL:COUNT 100
```

Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

### :GONogo:ACTion:MAIL:MODE

Function Sets/queries whether an e-mail report is sent when the determination is NO-GO.

```
Syntax      :GONogo:ACTion:MAIL:MODE {<Boolean>}
            :GONogo:ACTion:MAIL:MODE?
Example     :GONOGO:ACTION:MAIL:MODE OFF
            :GONOGO:ACTION:MAIL:MODE?→:GONOGO:
            ACTION:MAIL:MODE 0
```

Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

### :GONogo:ACTion:SAVE

Function Sets whether or not to save the NO-GO result to the medium or queries the current setting.

```
Syntax      :GONogo:ACTion:SAVE {<Boolean>}
            :GONogo:ACTion:SAVE?
Example     :GONOGO:ACTION:SAVE OFF
            :GONOGO:ACTION:SAVE?→:GONOGO:ACTION:
            SAVE 0
```

Description Use the ":FILE:DIRECTORY:DRIVE" command to set or query the medium type.

### :GONogo:COUNt?

Function Queries the actual number of times the GO/NO-GO determination has been performed.

```
Syntax      :GONogo:COUNt?
Example     :GONOGO:COUNT?→:GONOGO:COUNT 100
```

### :GONogo:EXECute

Function Executes GO/NO-GO determination.  This is an overlap command.

```
Syntax      :GONogo:EXECute
Example     :GONOGO:EXECUTE
```

### :GONogo:LOGic

Function Sets the GO/NO-GO determination logic or queries the current setting.

```
Syntax      :GONogo:LOGic {AND|OR}
            :GONogo:LOGic?
Example     :GONOGO:LOGIC AND
            :GONOGO:LOGIC?→:GONOGO:LOGIC AND
```

### :GONogo:MODE

Function Sets the GO/NO-GO determination type or queries the current setting.

```
Syntax      :GONogo:MODE {OFF|PARameter|ZONE}
            :GONogo:MODE?
Example     :GONOGO:MODE PARAMETER
            :GONOGO:MODE?→:GONOGO:MODE PARAMETER
```

### :GONogo:NGCount?

Function Queries the number of NO-GO results.

```
Syntax      :GONogo:NGCount?
Example     :GONOGO:NGCOUNT?→:GONOGO:NGCOUNT 5
```

### :GONogo:PARameter?

Function Queries all settings related to the parameter determination.

```
Syntax      :GONogo:PARameter?
Example     :GONOGO:PARAMETER?→:GONOGO:PARAMETER:
            ITEM1:MODE 0;TRACE 1;TYPE:
            MAXIMUM 1.0000E+00,-1.0000E+00;:GONOGO:
            PARAMETER:ITEM1:CONDITION OUT;:GONOGO:
            PARAMETER:ITEM2:MODE 0;TRACE 1;TYPE:
            MINIMUM 1.0000E+00,-1.0000E+00;:GONOGO:
            PARAMETER:ITEM2:CONDITION OUT;:GONOGO:
            PARAMETER:ITEM3:MODE 0;TRACE 1;TYPE:RMS
            1.0000E+00,-1.0000E+00;:GONOGO:
            PARAMETER:ITEM3:CONDITION OUT;:GONOGO:
            PARAMETER:ITEM4:MODE 0;TRACE 1;TYPE:
            AVERAGE 1.0000E+00,-1.0000E+00;:GONOGO:
            PARAMETER:ITEM4:CONDITION OUT
```

## :GONogo:PARameter:ITEM<x>?

Function Queries all settings related to waveform parameters.

Syntax
```
:GONogo:PARameter:ITEM<x>?
<x>=1 to 4
```

Example
```
:GONOGO:PARAMETER:ITEM1?→:GONOGO:
PARAMETER:ITEM1:MODE 0;TRACE 1;TYPE:
MAXIMUM 1.0000E+00,-1.0000E+00;:GONOGO:
PARAMETER:ITEM1:CONDITION OUT
```

## :GONogo:PARameter:ITEM<x>:CAUSe?

Function Queries whether or not the waveform parameter is the cause of the NO-GO result.

Syntax
```
GONogo:PARameter:ITEM<x>:CAUSe?
<x>=1 to 4
```

Example
```
:GONOGO:PARAMETER:ITEM1:CAUSE?→:GONOGO:
PARAMETER:ITEM1:CAUSE 1
```

Description The query returns a "1" if the parameter is the cause of the NO-GO result or a "0" if it is not.

## :GONogo:PARameter:ITEM<x>:CONDition

Function Sets the criteria for each waveform parameter or queries the current setting.

Syntax
```
:GONogo:PARameter:ITEM<x>:CONDition {IN|
OUT}
:GONogo:PARameter:ITEM<x>:CONDition?
<x>=1 to 4
```

Example
```
:GONOGO:PARAMETER:ITEM1:CONDITION OUT
:GONOGO:PARAMETER:ITEM1:CONDITION?→
:GONOGO:PARAMETER:ITEM1:CONDITION OUT
```

## :GONogo:PARameter:ITEM<x>:MODE

Function Turns the waveform parameter ON/OFF or queries the current setting.

Syntax
```
:GONogo:PARameter:ITEM<x>:MODE
{<Boolean>}
:GONogo:PARameter:ITEM<x>:MODE?
```

Example
```
:GONOGO:PARAMETER:ITEM<x>:MODE OFF
:GONOGO:PARAMETER:ITEM<x>:MODE?→
:GONOGO:PARAMETER:ITEM<x>:MODE 0
```

## :GONogo:PARameter:ITEM<x>:TRACe

Function Sets the waveform on which to use the waveform parameter or queries the current setting.

Syntax
```
:GONogo:PARameter:ITEM<x>:TRACe {<NRf>|
MATH<x>}
:GONogo:PARameter:ITEM<x>:TRACe?
The <x> in ITEM<x>=1 to 4
<NRf>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:GONOGO:PARAMETER:ITEM1:TRACE 1
:GONOGO:PARAMETER:ITEM1:TRACE?→
:GONOGO:PARAMETER:ITEM1:TRACE 1
```

## :GONogo:PARameter:ITEM<x>:TYPE?

Function Queries the waveform parameter and the upper and lower limits.

Syntax
```
:GONogo:PARameter:ITEM<x>:TYPE?
<x>=1 to 4
```

Example
```
:GONOGO:PARAMETER:ITEM1:TYPE?→:GONOGO:
PARAMETER:ITEM1:TYPE:
PTOPEAK 2.000E+00,1.000E+00
```

## :GONogo:PARameter:ITEM<x>:TYPE:<parameter>

Function Sets the upper and lower limits of each waveform parameter or queries the current setting.

Syntax
```
:GONogo:PARameter:ITEM<x>:TYPE:
<parameter> {<{voltage|DONTcare}>,
<{voltage|DONTcare}>|<{time|DONTcare}>,
<{time|DONTcare}>|<{frequency|DONTcare}>,
<frequency|DONTcare}>|<{{<NRf>}|
DONTcare}>,<{{<NRf>}|DONTcare}>}
:GONogo:PARameter:ITEM<x>:TYPE:
<parameter>?
<parameter>={AVERage|AVGFreq|AVGPeriod|
             BWIDth1|BWIDth2|DELay|
             DUTYcycle|FALL|FREQuency|
             HIGH|LOW|MAXimum|MINimum|
             NOVershoot|NWIDth|PERiod|
             PNUMber|POVershoot|PTOPeak|
             PWIDth|RISE|RMS|SDEViation|
             TY1Integ|TY2Integ|XY1Integ|
             XY2Integ}
<voltage>,<time>,<frequency>,<NRf>=
      See the IM701410-01E User's Manual.
```

Example
```
:GONOGO:PARAMETER:ITEM1:TYPE:AVERAGE
100mV,-100mV
:GONOGO:PARAMETER:ITEM1:TYPE:AVERAGE?→
:GONOGO:PARAMETER:ITEM1:TYPE:
AVERAGE 100.0E-03,-100.0E-03
```

Description Only the parameters that are set using this command can be queried.

## :GONogo:PARameter:ITEM<x>:VALue?

Function Queries the measured value of each waveform parameter.

Syntax
```
:GONogo:PARameter:ITEM<x>:VALue?
<x>=1 to 4
```

Example
```
:GONOGO:PARAMETER:ITEM1:VALUE?→
:GONOGO:PARAMETER:ITEM1:VALUE 1.98E-03
```

Description If the measurement is not possible (the mode is OFF, for example), "NAN(Not A Number)" is returned.

## :GONogo:SEQuence

Function: Selects single or continuous action or queries the current setting.

Syntax
```
:GONogo:SEQuence {CONTinuous|SINGle}
:GONogo:SEQuence?
```

Example
```
:GONOGO:SEQUENCE SINGLE
:GONOGO:SEQUENCE?→:GONOGO:
SEQUENCE SINGLE
```

## :GONogo:STOPcount

Function: Sets the number of waveform acquisitions on which to perform GO/NO-GO determination or queries the current setting.

Syntax
```
:GONogo:STOPcount {<NRf>|INFinity}
:GONogo:STOPcount?
<NRf>=1 to 65536
```

Example
```
:GONOGO:PARAMETER:STOPCOUNT 10
:GONOGO:PARAMETER:STOPCOUNT?→:GONOGO:
STOPCOUNT 10
```

## :GONogo:TRANge

Function: Sets the range over which to perform GO/NO-GO determination or queries the current setting.

Syntax
```
:GONogo:TRANge {<NRf>},{<NRf>}
:GONogo:TRANge?
<NRf>=-5.00 to 5.00 (10 div/displayed
      record length steps)
```

Example
```
:GONOGO:TRANGE -5.00,5.00
:GONOGO:TRANGE?→:GONOGO:
TRANGE -5.00,5.00
```

## :GONogo:WAIT?

Function: Waits for the GO/NO-GO determination to complete with a timeout.

Syntax
```
:GONogo:WAIT? {<NRf>}
<NRf>=0 to 8640000 (in units of 10 ms)
```

Example
```
(Set the timeout to 5 s.)
:GONOGO:WAIT? 500→:GONOGO:WAIT 0
```

Description: Returns "1" if it times out, "0" if it completes before timing out.

## :GONogo:ZONE?

Function: Queries all settings related to zone determination.

Syntax
```
:GONogo:ZONE?
```

Example
```
:GONOGO:ZONE?→:GONOGO:ZONE:ZONE1:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE2:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE3:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE4:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE5:
MODE 0;CONDITION OUT;:GONOGO:ZONE:ZONE6:
MODE 0;CONDITION OUT;:GONOGO:LOGIC AND
```

## :GONogo:ZONE:EDIT<x>:EXIT

Function: Exits from the edit menu of the zone for zone determination.

Syntax
```
:GONogo:ZONE:EDIT<x>:EXIT {<NRf>|QUIT}
<X>= 1 to 12 (1 to 4 are traces; 5 and 6
    is MATH1 and 2; 7 to 12 are
    determination zones 1 to 6.)
<NRf>= 1 to 6 (destination zone No. for
      saving the zone being edited)
```

Example
```
:GONOGO:ZONE:EDIT1:EXIT 1
```

Description: This command must be issued to keep the zones specified using "GONogo:ZONE:EDIT<x>:{PART|WHOLe}."
An error occurs if this command is issued when the zone is not being edited.

## :GONogo:ZONE:EDIT<x>:PART

Function: Edits a portion of the zone for zone determination.

Syntax
```
:GONogo:ZONE:EDIT<x>:PART {<NRf>,<NRf>,
<NRf>,<NRf>}
<X>= 1 to 12 (1 to 4 are traces; 5 and 6
    is MATH1 and 2; 7 to 12 are
    determination zones 1 to 6.)
<NRf>=-5.00 to 5.00 (div, 0.01 steps: T1
      and T2 cursor)
      -8.00 to 8.00 (div, 0.05 steps: up
      and down)
      (in the order T1, T2, ↑, and ↓)
```

Example
```
:GONOGO:ZONE:EDIT1:PART -2.50,2.50,1.00,
1.00
```

Description: When you finish editing, you must exit the edit mode using the "GONogo:ZONE:EDIT<x>:EXIT" command to keep the new zone.

## :GONogo:ZONE:EDIT<x>:WHOLe

Function: Edits the zone for zone determination.

Syntax
```
:GONogo:ZONE:EDIT<x>:WHOLe {<NRf>,<NRf>,
<NRf>,<NRf>}
<X>= 1 to 12 (1 to 4 are traces; 5 and 6
    is MATH1 and 2; 7 to 12 are
    determination zones 1 to 6.)
<NRf>=0 to 5.00 (div, 0.01 steps: left
      and right)
      0 to 8.00 (div, 0.05 steps: up and
      down)
      (in the order ←,→,↑, and ↓)
```

Example
```
:GONOGO:ZONE:EDIT1:WHOLE -2.50,2.50,
1.00,1.00
```

Description: When you finish editing, you must exit the edit mode using the "GONogo:ZONE:EDIT<x>:EXIT" command to keep the new zone.

## :GONogo:ZONE:ZONE<x>?

Function Queries all settings related to zone determination of the specified zone.

Syntax `:GONogo:ZONE:ZONE<x>?`
`<x>=1 to 6`

Example `:GONOGO:ZONE:ZONE1?→:GONOGO:ZONE:ZONE1:MODE 0;CONDITION OUT`

## :GONogo:ZONE:ZONE<x>:CAUSe?

Function Queries whether or not the specified zone for zone determination is the cause of the NO-GO result.

Syntax `:GONogo:ZONE:ZONE<x>:CAUSe?`
`<x>=1 to 6`

Example `:GONOGO:ZONE:ZONE1:CAUSE?→:GONOGO:ZONE:ZONE1:CAUSE 1`

Description The query returns a "1" if the zone is the cause of the NO-GO result or a "0" if it is not.

## :GONogo:ZONE:ZONE<x>:CONDition

Function Sets the criteria for the zone for zone determination or queries the current setting.

Syntax `:GONogo:ZONE:ZONE<x>:CONDition {IN|OUT}`
`:GONogo:ZONE:ZONE<x>:CONDition?`
`<x>=1 to 6`

Example `:GONOGO:ZONE:ZONE1:CONDITION IN`
`:GONOGO:ZONE:ZONE1:CONDITION?→:GONOGO:ZONE:ZONE1:CONDITION IN`

## :GONogo:ZONE:ZONE<x>:MODE

Function Turns ON/OFF the determination of the zone or queries the current setting.

Syntax `:GONogo:ZONE:ZONE<x>:MODE {<Boolean>}`
`:GONogo:ZONE:ZONE<x>:MODE?`
`<x>=1 to 6`

Example `:GONOGO:ZONE:ZONE1:MODE ON`
`:GONOGO:ZONE:ZONE1:MODE?→:GONOGO:ZONE:ZONE1:MODE 1`

## :GONogo:ZONE:ZONE<x>:SEND

Function Sets the zone data specified for the determination zone or queries the current setting.

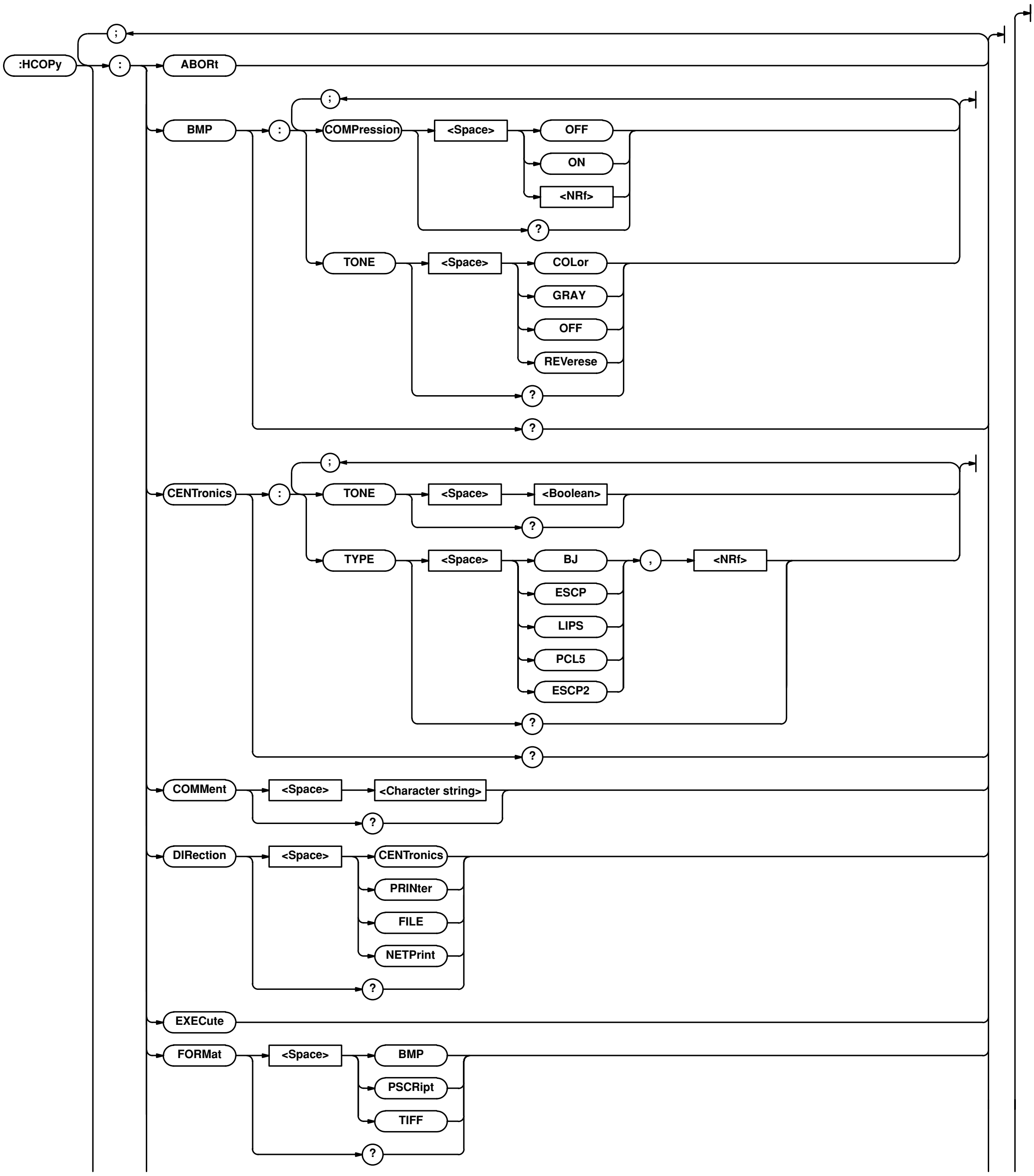
Syntax `:GONogo:ZONE:ZONE<x>:SEND #6004028 (data string)`
`:GONnogo:ZONE:ZONE<x>:SEND?`
`<x>=1 to 6`

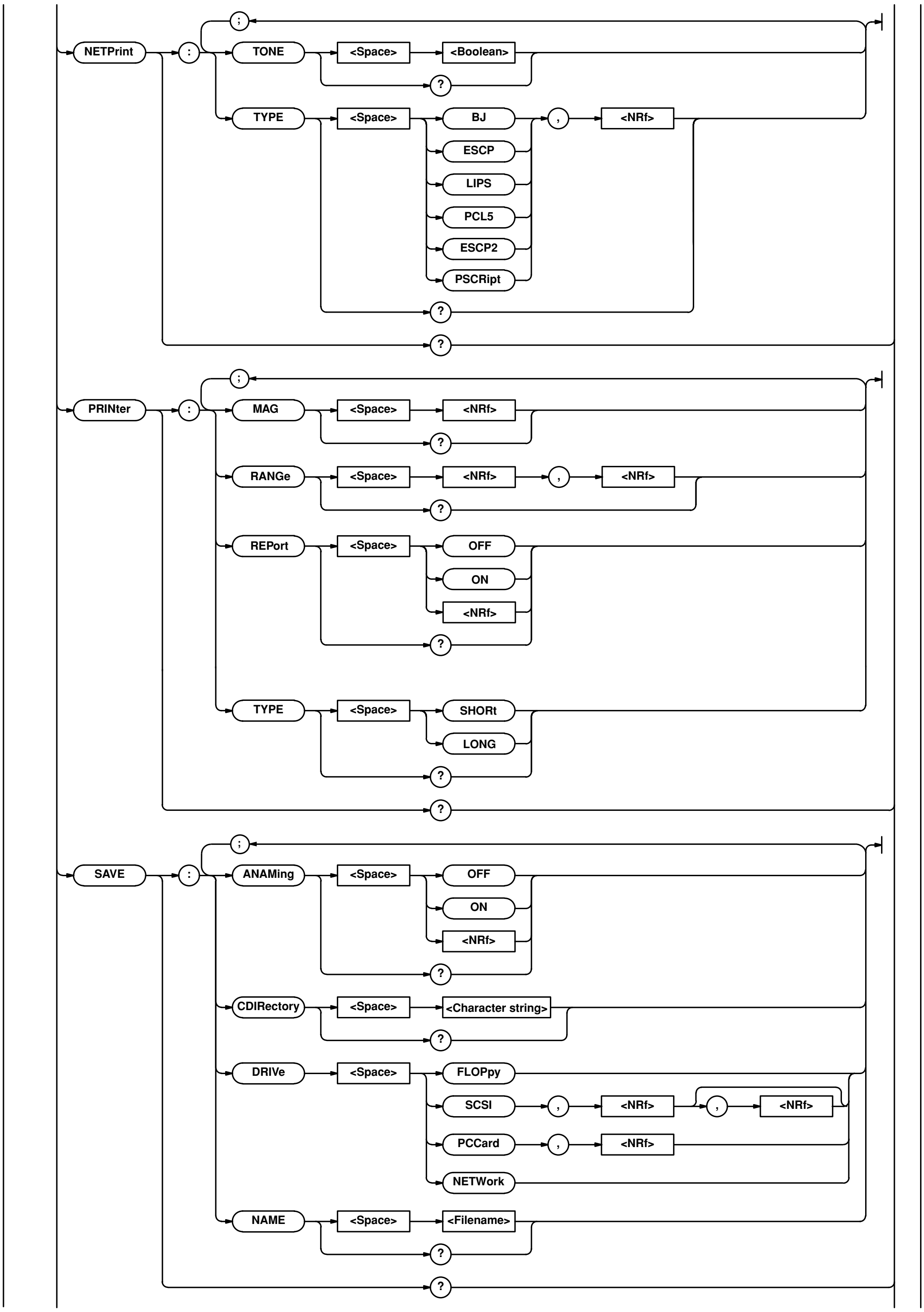
Example `:GONOGO:ZONE:ZONE1:SEND #6004028 (data string)`
`:GONOGO:ZONE:ZONE1:SEND?→:GONOGO:ZONE:ZONE1:SEND #6004028(data string)`

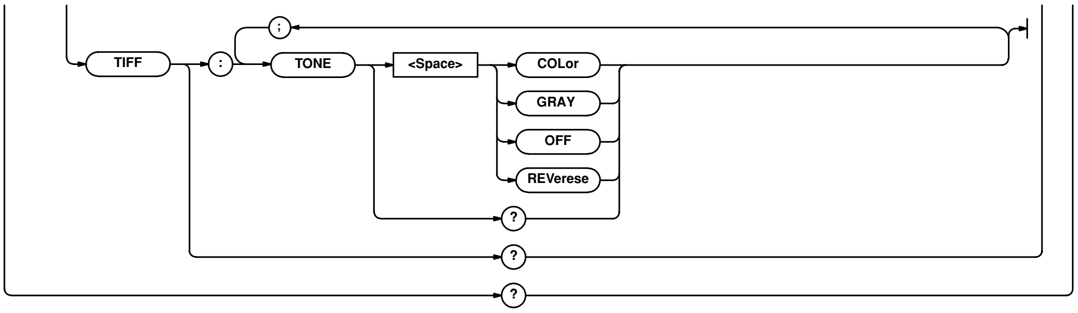
Description
- The (data string) consists of zone data of length 4028 bytes.
- Zone data consist of 8-bit data (0 to 255). This data string may contain ASCII codes "0AH" or "00H" that stand for "NL." Hence, care must be taken when programming the controller.
- This command is used to save the zone data to the PC and set the data as zone data on the instrument.
- When this command is used to set the zone data to the instrument to perform the determination, issue the "GONogo:ZONE:ZONE<x>:MODE ON" command afterwards.

## 5.12 HCOPy Group

The commands in the HCOPy group are used to make settings and queries about screen image data output, for example, output to the optional built-in printer. You can make the same settings that you can make by pressing the SHIFT key, then pressing the COPY key.




NETPrint
:
;
TONE
<Space>
<Boolean>
?
TYPE
<Space>
BJ
ESCP
LIPS
PCL5
ESCP2
PSCRipt
,
<NRf>
?
?
PRINter
:
;
MAG
<Space>
<NRf>
?
RANGe
<Space>
<NRf>
,
<NRf>
?
REPort
<Space>
OFF
ON
<NRf>
?
TYPE
<Space>
SHORt
LONG
?
?
SAVE
:
;
ANAMing
<Space>
OFF
ON
<NRf>
?
CDIRectory
<Space>
<Character string>
?
DRIVe
<Space>
FLOPpy
SCSI
,
<NRf>
,
<NRf>
PCCard
,
<NRf>
NETWork
NAME
<Space>
<Filename>
?
?



## :HCOPy? (Hard COPY)

Function Queries all settings relating to output of screen image data.

Syntax `:HCOPy?`

Example
```
:HCOPY?→:HCOPY:DIRECTION FILE;:HCOPY:
PRINTER:TYPE SHORT;REPORT 0;
RANGE -5.00,5.00;MAG 2;:HCOPY:
CENTRONICS:TYPE BJ,360;TONE COLOR;:
HCOPY:SAVE:ANAMING 0;NAME "DISP_1";:
HCOPY:FORMAT BMP;:HCOPY:BMP:TONE OFF;
COMPRESSION 0;:HCOPY:TIFF:TONE OFF;
COMPRESSION 0
```

## :HCOPy:ABORt

Function Aborts data output and paper feed. This is an overlap command.

Syntax `:HCOPy:ABORt`

Example `:HCOPY:ABORT`

## :HCOPy:{BMP|TIFF}?

Function Queries all the settings relating to BMP (TIFF) format.

Syntax `:HCOPy:{BMP|TIFF}?`

Example
```
(An example for BMP is given below.)
:HCOPY:BMP?→:HCOPY:BMP:TONE OFF;
COMPRESSION 0
```

## :HCOPy:BMP:COMPression

Function Sets/queries use of screen image data compression relating to BMP format.

Syntax
```
:HCOPy:BMP:COMPression {<Boolean>}
:HCOPy:BMP:COMPression?
```

Example
```
(An example for BMP is given below.)
:HCOPy:BMP:COMPression ON
:HCOPy:BMP:COMPression?→:HCOPY:BMP:
COMPRESSION 1
```

Description If "`HCOPy:FORMat`" is set to BMP, compression can be specified only when "`HCOPy:BMP:TONE`" is set to COLor.

## :HCOPy:{BMP|TIFF}:TONE

Function Sets/queries the color tone for BMP (TIFF) format.

Syntax
```
:HCOPy:{BMP|TIFF}:TONE{COLor|OFF|
REVerese|GRAY}
:HCOPy:{BMP|TIFF}:TONE?
```

Example
```
(An example for BMP is given below.)
:HCOPY:BMP:TONE COLOR
:HCOPY:BMP:TONE?→:HCOPY:BMP:TONE COLOR
```

## :HCOPy:CENTronics?

Function Queries all setting values relating to the output to the external printer.

Syntax `:HCOPy:CENTronics?`

Example
```
:HCOPY:CENTRONICS?→:HCOPY:CENTRONICS:
TYPE BJ,360;TONE 1
```

## :HCOPy:CENTronics:TONE

Function Sets/queries the half tone setting for the external printer.

Syntax
```
:HCOPy:CENTronics:TONE {<Boolean>}
:HCOPy:CENTronics:TONE?
```

Example
```
:HCOPY:CENTRONICS ON
:HCOPY:CENTRONICS?→:HCOPY:
CENTRONICS TONE 1
```

## :HCOPy:CENTronics:TYPE

Function Sets/queries the command type to output to the external printer.

Syntax
```
:HCOPy:CENTronics:TYPE {BJ|ESCP|LIPS|
PCL5|ESCP2},{<NRf>}
:HCOPy:CENTronics:TYPE?
<NRf>=180, 300 or 360 (dpi)
```

Example
```
:HCOPY:CENTRONICS:TYPE BJ,360
:HCOPY:CENTRONICS:TYPE?→:HCOPY:
CENTRONICS:TYPE BJ,360
```

Description <NRf> is specified only when the type is set to "`BJ`."

## :HCOPy:COMMent

Function Sets/queries the comment to be displayed at the screen.

Syntax :HCOPy:COMMent <Character string>
:HCOPy:COMMent?
<Character string>=Within 20 characters

Example :HCOPY:COMMENT "THIS IS TEST."
:HCOPY:COMMENT?→:HCOPY:
COMMENT "THIS IS TEST."

## :HCOPy:DIRection

Function Sets/queries the data output destination.

Syntax :HCOPy:DIRection {CENTronics|PRINter|
FILE|NETPrint}
:HCOPy:DIRection?

Example :HCOPY:DIRECTION FILE
:HCOPY:DIRECTION?→:HCOPY:DIRECTION FILE

Description "NETPrint" can be set on a DL7200 or on a DL7100 with firmware version 2.01 and later.

## :HCOPy:EXECute

Function Executes data output. This is an overlap command.

Syntax :HCOPy:EXECute

Example :HCOPY:EXECute

## :HCOPy:FORMat

Function Sets the format of the image that is to be saved to the file or queries the current setting.

Syntax :HCOPy:FORMat {BMP|PSCRipt|TIFF}
:HCOPy:FORMat?

Example :HCOPY:FORMAT BMP
:HCOPY:FORMAT?→:HCOPY:FORMAT BMP

Description This setting is void if "FILE" is not selected with the "HCOPy:DIRection" command.

## :HCOPy:NETprint?

Function Queries all settings values involving output to network printers.

Syntax :HCOPy:NETPrint?

Example :HCOPY:NETPRINT?→:HCOPY:NETPRINT:
TYPE POSTSCRIPT;TONE 1

Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

## :HCOPy:NETprint:TONE

Function Sets/queries halftones for output to network printers.

Syntax :HCOPy:NETPrint:TONE {<Boolean>}
:HCOPy:NETPrint:TONE?

Example :HCOPY:NETPRINT:TONE ON
:HCOPY:NETPRINT:TONE?→:HCOPY:NETPRINT:
TONE 1

Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

## :HCOPy:NETprint:TYPE

Function Sets/queries command types for output to network printers.

Syntax :HCOPy:NETPrint:TYPE {BJ|ESCP|LIPS|PCL5|
ESCP2|PSCRipt,<NRf>}
:HCOPy:NETPrint:TYPE?

Example :HCOPY:NETPRINT:TYPE PSCRIPT
:HCOPY:NETPRINT:TYPE?→:HCOPY:NETPRINT:
TYPE PSCRIPT

Description <NRf> is specified only when the type is set to "BJ." This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

## :HCOPy:PRINter?

Function Queries all settings relating to the built-in printer.

Syntax :HCOPy:PRINter?

Example :HCOPY:PRINTER?→:HCOPY:PRINTER:
TYPE SHORT;REPORT 0;RANGE -5.00,5.00;
MAG 2

## :HCOPy:PRINter:MAG

Function Sets the zoom rate to use when using the long copy function to print to the built-in printer or queries the current setting.

Syntax :HCOPy:PRINter:MAG {<NRf>}
:HCOPy:PRINter:MAG?
<NRf>=2 to 250000 (See the IM701410-01E
User's Manual.)

Example :HCOPY:PRINTER:MAG 1
:HCOPY:PRINTER:MAG?→:HCOPY:PRINTER:
MAG 1

## :HCOPy:PRINter:RANGe

Function Sets the output range when using the long copy function to print to the built-in printer or queries the current setting.

Syntax :HCOPy:PRINter:RANGe {<NRf>},{<NRf>}
:HCOPy:PRINter:RANGe?
<NRf>=–5.00 to 5.00 (10 div/displayed
record length steps)

Example :HCOPY:PRINTER:RANGE -2.50,2.50
:HCOPY:PRINTER:RANGE?→:HCOPY:PRINTER:
RANGE -2.50,2.50

## :HCOPy:PRINter:REPort

Function Sets whether or not to output additional information to the built-in printer or queries the current setting.

Syntax :HCOPy:PRINter:REPort {<Boolean>}
:HCOPy:PRINter:REPort?

Example :HCOPY:PRINTER:REPORT ON
:HCOPY:PRINTER:REPORT?→:HCOPY:PRINTER:
REPORT 1

### :HCOPy:PRINter:TYPE

Function Selects short copy or long copy when printing to the built-in printer or queries the current setting.

```
Syntax      :HCOPy:PRINter:TYPE {LONG|SHORt}
            :HCOPy:PRINter:TYPE?
Example     :HCOPY:PRINTER:TYPE SHORT
            :HCOPY:PRINTER:TYPE?→:HCOPY:PRINTER:
            TYPE SHORT
```

### :HCOPy:SAVE?

Function Queries all file saving settings.

```
Syntax      :HCOPy:SAVE?
Example     :HCOPY:SAVE?→:HCOPY:SAVE:DRIVE FLOPPY;
            DIRECTORY "NO_1";ANAMING 0;NAME "DISP_1"
```

### :HCOPy:SAVE:ANAMing

Function Enables/disables/queries automatic naming of output files.

```
Syntax      :HCOPy:ANAMing {<Boolean>}
            :HCOPy:SAVE:ANAMing?
Example     :HCOPY:ANAMING ON
            :HCOPY:SAVE:ANAMING?→:HCOPY:SAVE:
            ANAMING 1
```

### :HCOPy:SAVE:CDIRectory

Function Changes the directory.

```
Syntax      :HCOPy:SAVE:DIRectory <Character string>
            <Character string>=Within 8 characters.
Example     :HCOPY:SAVE:DIRECTORY "NO_1"
```

### :HCOPy:SAVE:DRIVe

Function Selects the medium on which to create the file.

```
Syntax      :HCOPy:SAVE:DRIVe {FLOPpy|SCSI,<NRf>
            [,<NRf>]|PCCard[,<NRf>]}
            :HCOPy:SAVE:DRIVe?
            The first <NRf> of SCSI= SCSI address (0
                                     to 7)
            The second <NRf> of SCSI=Partitions (0
                                     to 4)
            <NRf> of PCCard=Partitions (0 to 4)
Example     :HCOPY:SAVE:DRIVE FLOPPY
            :HCOPY:SAVE:DRIVE?→:HCOPY:SAVE:
            DRIVE FLOPPY
```

Description If the drive does not contain partitions, omit the <NRf> corresponding to partitions.

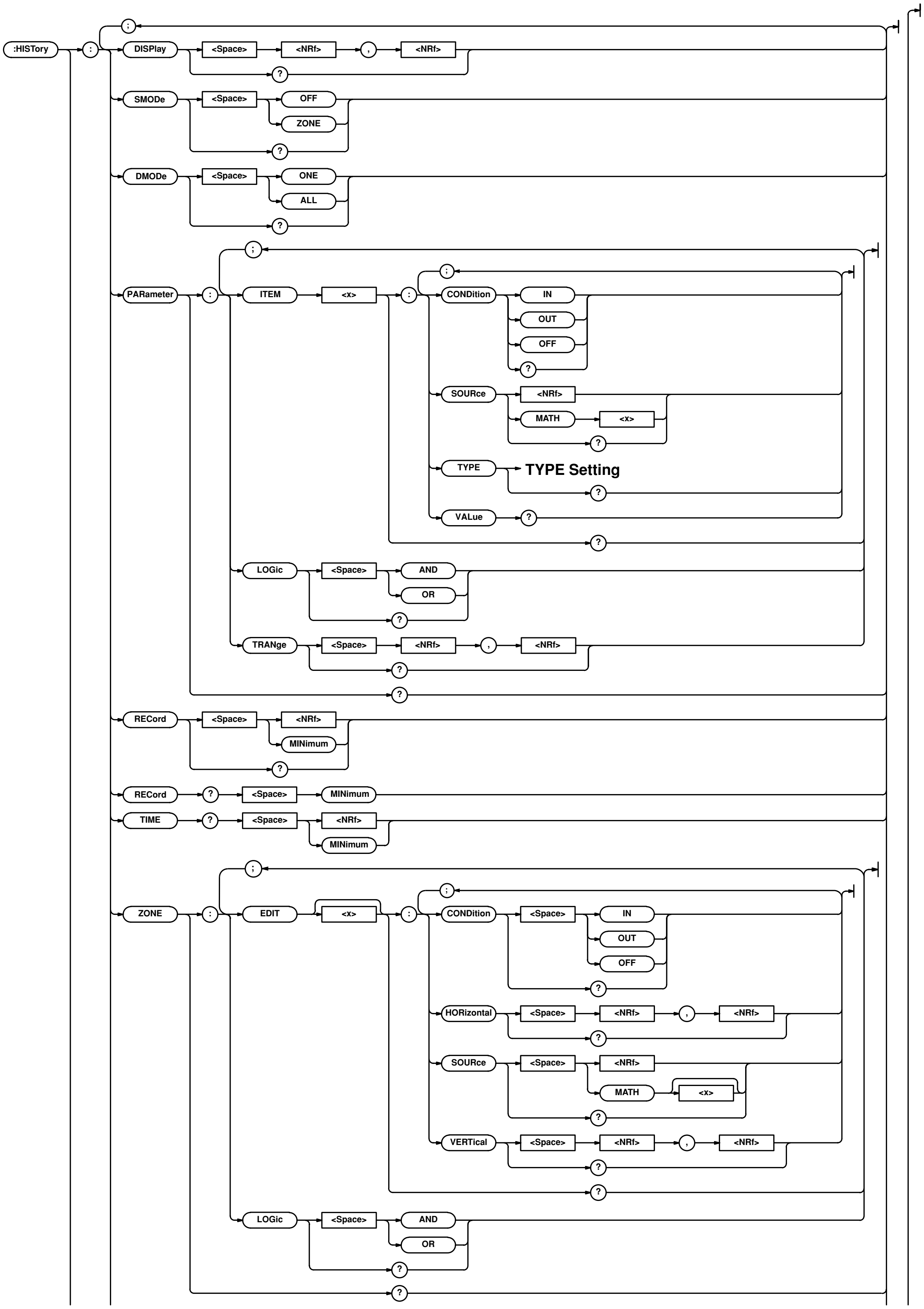
### :HCOPy:SAVE:NAME

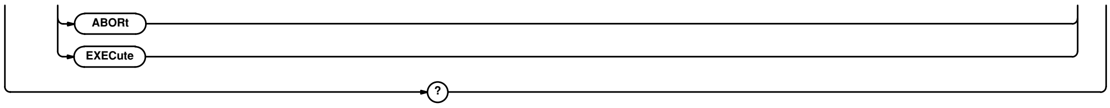
Function Sets/queries the name of the file to be created.

```
Syntax      :HCOPy:SAVE:NAME <Filename>
            :HCOPy:SAVE:NAME?
Example     :HCOPY:SAVE:NAME "DISP_1"
            :HCOPY:SAVE:NAME?→:HCOPY:SAVE:
            NAME "DISP_1"
```

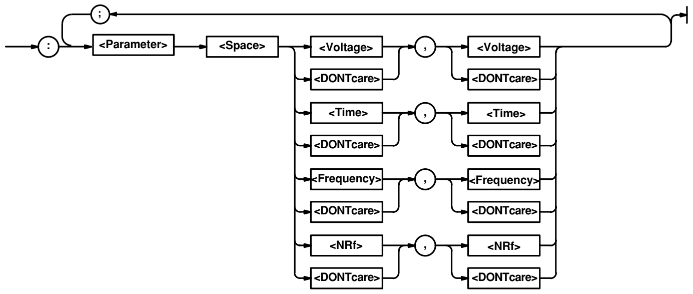
# 5.13 HISTory Group

The commands in the HISTory group are used to make settings and queries about recalling of data from the history memory. You can make the same settings that you can make using the HISTORY key on the front panel.





TYPE Setting



## :HISTory?

Function Queries all settings relating to the history memory function.

Syntax :HISTory?

Example :HISTORY?→:HISTORY:RECORD 0;DMODE ONE;
DISPLAY 0,-127;SMODE OFF;ZONE:EDIT1:
CONDITION OFF;SOURCE 1;HORIZONTAL 0.50,
-0.50;VERTICAL -3.0000000,-2.5000000;:
HISTORY:ZONE:EDIT2:CONDITION OFF;
SOURCE 1;HORIZONTAL 0.50,-0.50;
VERTICAL -1.0000000,-0.5000000;:HISTORY:
ZONE:EDIT3:CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL 1.0000000,1.5000000;:HISTORY:
ZONE:EDIT4:CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL 3.0000000,3.5000000;:HISTORY:
ZONE:LOGIC AND;:HISTORY:PARAMETER:ITEM1:
CONDITION OFF;SOURCE 1;TYPE:
PTOPEAK DONTCARE,DONTCARE;:HISTORY:
PARAMETER:ITEM2:CONDITION OFF;SOURCE 1;
TYPE:MAXIMUM DONTCARE,DONTCARE;:HISTORY:
PARAMETER:ITEM3:CONDITION OFF;SOURCE 1;
TYPE:MINIMUM DONTCARE,DONTCARE;:HISTORY:
PARAMETER:ITEM4:CONDITION OFF;SOURCE 1;
TYPE:RMS DONTCARE,DONTCARE;:HISTORY:
PARAMETER:LOGIC AND;TRANGE -5.0000000,
5.0000000

## :HISTory:ABORt

Function Aborts the search.

Syntax :HISTory:ABORt

Example :HISTORY:ABORT

## :HISTory:DISPlay

Function Sets the start/end number of the display record or queries the current setting.

Syntax :HISTory:DISPlay {<NRf>,<NRf>}
:HISTory:DISPlay?
<NRf>=0 to –2047 (DL7100)
0 to –4095 (DL7200)

Example :HISTORY:DISPLAY 0,-100
:HISTORY:DISPLAY?→:HISTORY:DISPLAY 0,
-100

Description The number of records that can be specified varies depending on the acquisition setting and the model. For details, see the IM701410-01E User's Manual.

## :HISTory:EXECute

Function Executes the search. This is an overlap command.

Syntax :HISTory:EXECute

Example :HISTORY:EXECUTE

## :HISTory:SMODe (Search Mode)

Function Sets the history memory's search mode or queries the current setting.

Syntax :HISTory:SMODe {OFF|ZONE}
:HISTory:SMODe?

Example :HISTORY:SMODE ZONE
:HISTORY:SMODE?→:HISTORY:SMODE ZONE

## :HISTory:DMODe (Display Mode)

Function Sets whether to display one record or all records or queries the current setting.

Syntax :HISTory:DMODe {ONE|ALL}
:HISTory:DMODe?

Example :HISTORY:DMODE ALL
:HISTORY:DMODE?→:HISTORY:DMODE ALL

Description If "ALL" is specified, the records that are specified with the ":HISTory:DISPlay" command are accumulated on the screen.

### :HISTory:PARameter?

Function Queries all settings related to the parameter search.

Syntax
```
:HISTory:PARameter?
```

Example
```
:HISTORY:PARAMETER?→:HISTORY:PARAMETER:
ITEM1:CONDITION OFF;SOURCE 1;TYPE:
PTOPEAK DONTCARE,DONTCARE;:HISTORY:
PARAMETER:ITEM2:CONDITION OFF;SOURCE 1;
TYPE:MAXIMUM DONTCARE,DONTCARE;:HISTORY:
PARAMETER:ITEM3:CONDITION OFF;SOURCE 1;
TYPE:MINIMUM DONTCARE,DONTCARE;:HISTORY:
PARAMETER:ITEM4:CONDITION OFF;SOURCE 1;
TYPE:RMS DONTCARE,DONTCARE;:HISTORY:
PARAMETER:LOGIC AND;TRANGE -5.0000000,
5.0000000
```

### :HISTory:PARameter:ITEM<x>?

Function Queries all settings related to the specified parameter of the parameter search.

Syntax
```
:HISTory:PARameter:ITEM<x>?
<x>=1 to 4
```

Example
```
:HISTORY:PARAMETER:ITEM1?→:HISTORY:
PARAMETER:ITEM1:CONDITION OFF;SOURCE 1;
TYPE:PTOPEAK DONTCARE,DONTCARE
```

### :HISTory:PARameter:ITEM<x>:CONDition

Function Sets the criteria for the specified parameter of the parameter search or queries the current setting.

Syntax
```
:HISTory:PARameter:ITEM<x>:
CONDition {OFF|IN|OUT}
:HISTory:PARameter:ITEM<x>:CONDition?
<x>=1 to 4
```

Example
```
:HISTORY:PARAMETER:ITEM1:CONDITION IN
:HISTORY:PARAMETER:ITEM1:CONDITION?→:
HISTORY:PARAMETER:ITEM1:CONDITION IN
```

### :HISTory:PARameter:ITEM<x>:SOURce

Function Sets the trace on which to perform the parameter search of the specified parameter or queries the current setting.

Syntax
```
:HISTory:PARameter:ITEM<x>:
SOURce {<NRf>|MATH<x>}
:HISTory:PARameter:ITEM<x>:SOURce?
The <x> in ITEM<x>=1 to 4
<NRf>=1 to 4
The <x> in MATH<x>=1 or 2.
```

Example
```
:HISTORY:PARAMETER:ITEM1:SOURCE 1
:HISTORY:PARAMETER:ITEM1:SOURCE?→:
HISTORY:PARAMETER:ITEM1:SOURCE 1
```

### :HISTory:PARameter:ITEM<x>:TYPE:<parameter>

Function Sets the upper and lower limits for the specified parameter of the parameter search or queries the current setting.

Syntax
```
:HISTory:PARameter:ITEM<x>:TYPE:
<parameter> {<{voltage|DONTcare}>,
<{voltage|DONTcare}>|<{time|DONTcare}>,
<{time|DONTcare}>|<{frequency|
DONTcare}>,<frequency|DONTcare}>|
<{{<NRf>}|DONTcare}>,<{{<NRf>}|
DONTcare}>}
:HISTory:PARameter:ITEM<x>:TYPE?
<x>=1 to 4
<parameter>={AVERage|AVGFreq|AVGPeriod|
            BWIDth1|BWIDth2|DELay|
            DUTYcycle|FALL|FREQuency|
            HIGH|LOW|MAXimum|MINimum|
            NOVershoot|NWIDth|PERiod|
            PNUMber|POVershoot|PTOPeak|
            PWIDth|RISE|RMS|SDEViation|
            TY1Integ|TY2Integ|XY1Integ|
            XY2Integ}
<voltage>,<time>,<frequency>,<NRf>=See
the IM701410-01E User's Manual.
```

Example
```
:HISTORY:PARAMETER:ITEM1:TYPE:
AVERAGE 1.00,-1.00
```

### :HISTory:PARameter:ITEM<x>:VALue?

Function Queries the measured value of the specified parameter of the parameter search or queries the current setting.

Syntax
```
:HISTory:PARameter:ITEM<x>:VALue?
<x>=1 to 4
```

Example
```
:HISTORY:PARAMETER:ITEM1:VALUE?→:
HISTORY:PARAMETER:ITEM1:VALUE 1.00E-03
```

### :HISTory:PARameter:LOGic

Function Sets the determination logic of the parameter search or queries the current setting.

Syntax
```
:HISTory:PARameter:LOGic {AND|OR}
:HISTory:PARameter:LOGic?
```

Example
```
:HISTORY:PARAMETER:LOGIC AND
:HISTORY:PARAMETER:LOGIC?→:HISTORY:
PARAMETER:LOGIC AND
```

### :HISTory:PARameter:TRANge

Function Sets the range over which to perform the parameter search or queries the current setting.

Syntax
```
:HISTory:PARameter:TRANge {<NRf>,<NRf>}
:HISTory:PARameter:TRANge?
<NRf>=-5.00 to 5.00 (10 div/displayed
      record length steps)
```

Example
```
:HISTORY:PARAMETER:TRANGE -5.00,5.00
:HISTORY:PARAMETER:TRANGE?→:HISTORY:
PARAMETER:TRANGE -5.00,5.00
```

## :HISTory:RECord

Function Sets the target record or queries the current setting.

Syntax
```
:HISTory:RECord {<NRf>|MINimum}
:HISTory:RECord?
<NRf>=0 to –2047 (DL7100)
      0 to –4095 (DL7200)
```

Example
```
:HISTORY:RECORD 0
:HISTORY:RECORD?→:HISTORY:RECORD 0
```

Description
- If "MINimum" is specified, the minimum record number is selected.
- The selectable target record varies depending on the model, record length, and interleave mode settings. For details, see the IM701410-01E User's Manual.

## :HISTory:RECord? MINimum

Function Queries the minimum record number.

Syntax
```
:HISTory:RECord? MINimum
```

Example
```
:HISTORY:RECORD? MINIMUM→-999
```

Description The minimum record number varies depending on the model, record length, and interleave mode settings. For details, see the IM701410-01E User's Manual.

## :HISTory:TIME?

Function Queries the time of the specified record number.

Syntax
```
:HISTory:TIME? {<NRf>|MINimum}
```

Example
```
:HISTORY:TIME? -100→:HISTORY:
TIME "-0100 10:20:30.40"
```

Description If "MINimum" is specified, the minimum record number is selected.

## :HISTory:ZONE?

Function Queries all settings related to zone search.

Syntax
```
:HISTory:ZONE?
```

Example
```
:HISTORY:ZONE?→:HISTORY:ZONE:EDIT1:
CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL -3.0000000,-1.0000000;:HISTORY:
ZONE:EDIT2:CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL -1.0000000,1.0000000;:HISTORY:
ZONE:EDIT3:CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL 1.0000000,3.0000000;:HISTORY:
ZONE:EDIT4:CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL 3.0000000,-2.5000000;:
HISTORY:ZONE:LOGIC AND
```

## :HISTory:ZONE:EDIT<x>

Function Queries all settings related to all zone cursors.

Syntax
```
:HISTory:ZONE:EDIT<x>?
<x>=1 to 4
```

Example
```
:HISTORY:ZONE:EDIT1?→:HISTORY:ZONE:
EDIT1:CONDITION OFF;SOURCE 1;
HORIZONTAL 0.50,-0.50;
VERTICAL -3.0000000,-1.0000000
```

## :HISTory:ZONE:EDIT<x>:CONDition

Function Sets the search condition of the zone cursor or queries the current setting.

Syntax
```
:HISTory:ZONE:EDIT<x>:CONDition {OFF|IN|
OUT}
:HISTory:ZONE:EDIT<x>:CONDition?
<x>=1 to 4
```

Example
```
:HISTORY:ZONE:EDIT1:CONDITION IN
:HISTORY:ZONE:EDIT1:CONDITION?→
:HISTORY:ZONE:EDIT1:CONDITION IN
```

Description Use EDIT<x> to specify the zone cursor.

## :HISTory:ZONE:EDIT<x>:HORizontal

Function Sets the vertical position (upper and lower values) of the specified zone cursor or queries the current setting.

Syntax
```
:HISTory:ZONE:EDIT<x>:HORizontal {<NRf>,
<NRf>}
:HISTory:ZONE:EDIT<x>:HORizontal?
<x>=1 to 4
<NRf>=–4 to 4 div (0.01 steps, in the
      order Upper and Lower)
```

Example
```
:HISTORY:ZONE:EDIT1:
HORIZONTAL 4.00,-4.00
:HISTORY:ZONE:EDIT1:HORIZONTAL?→
:HISTORY:ZONE:EDIT1:
HORIZONTAL 4.00,-4.00
```

Description Use EDIT<x> to specify the zone cursor.

## :HISTory:ZONE:EDIT<x>:SOURce

Function Sets the waveform on which to use the specified zone cursor or queries the current setting.

Syntax
```
:HISTory:ZONE:EDIT<x>:SOURce {<NRf>|
MATH<x>}
:HISTory:ZONE:EDIT<x>:SOURce?
<x>=1 to 4
<NRf>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:HISTORY:ZONE:EDIT1:SOURCE 1
:HISTORY:ZONE:EDIT1:SOURCE?→:HISTORY:
ZONE:EDIT1:SOURCE 1
```

Description Use EDIT<x> to specify the zone cursor.

### :HISTory:ZONE:EDIT<x>:VERTical

Function Sets the horizontal position (left and right values) of the specified zone cursor or queries the current setting.

Syntax
```
:HISTory:ZONE:EDIT<x>:VERTical {<NRf>,
<NRf>}
:HISTory:ZONE:EDIT<x>:VERTical?
<x>=1 to 4
<NRf>=–5 to 5 div (10 div/displayed
      record length steps, in the order
      left and right)
```

Example
```
:HISTORY:ZONE:EDIT1:VARTICAL -2,2
:HISTORY:ZONE:EDIT1:VARTICAL?→:HISTORY:
ZONE:EDIT1:VARTICAL -2.0000000,2.0000000
```

Description Use EDIT<x> to specify the zone cursor.

### :HISTory:ZONE:LOGic

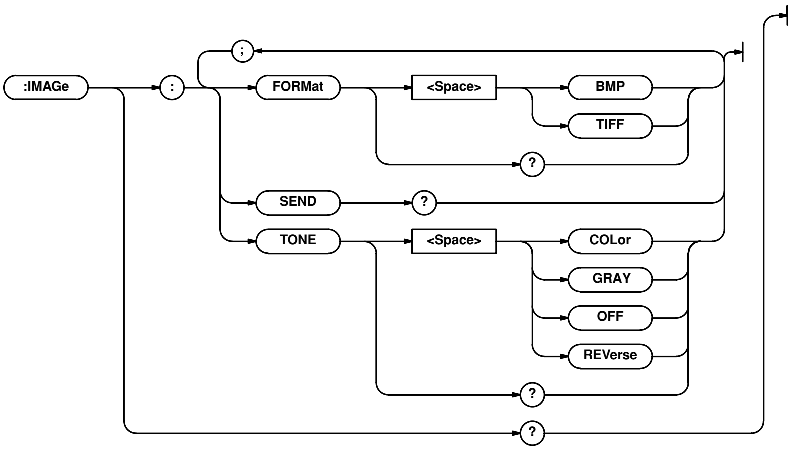
Function Sets the search logic between zone cursors or queries the current setting.

Syntax
```
:HISTory:ZONE:LOGic {AND|OR}
:HISTory:ZONE:LOGic?
```

Example
```
:HISTORY:ZONE:LOGIC AND
:HISTORY:ZONE:LOGIC?→:HISTORY:ZONE:
LOGIC AND
```

## 5.14 IMAGe Group

Use this group to set or query the screen image data output settings. There are no corresponding front-panel keys for these operations.



### :IMAGe?

Function Queries all screen image data output settings.

Syntax `IMAGe?`

Example `:IMAGE?→:IMAGE:TONE OFF;FORMAT TIFF`

### :IMAGe:FORMat

Function Sets/queries the screen image data output format.

Syntax `:IMAGe:FORMat {TIFF|BMP}`
`:IMAGe:FORMat?`

Example `:IMAGE:FORMAT TIFF`
`:IMAGE:FORMAT?→:IMAGE:FORMAT TIFF`

### :IMAGe:SEND?

Function Queries screen image data.

Syntax `:IMAGe:SEND?`

Example `:IMAGE:SEND?→#6 (number of bytes[6-digit value]) (Data byte string) (Block data)`

Description Number of bytes in <block data> is {(2+6+Number of data+1(delimiter)}. For information about block data, refer to page 4-6.

### :IMAGe:TONE

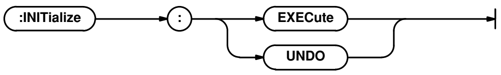
Function Sets/queries the color tone of the BMP (TIFF) format of the screen image data to output.

Syntax `:IMAGe:TONE {COLor|OFF|REVerse|GRAY}`
`:IMAGe:TONE?`

Example `:IMAGE:TONE COLOR`
`:IMAGE:TONE?→:IMAGE:TONE COLOR`

## 5.15 INITialize Group

The commands in the INITialize group are related to initializing.  You can make the same settings using the Initialize menu of the SETUP key on the front panel.



### :INITialize:EXECute

Function Executes initializing.

Syntax `:INITialize:EXECute`

Example `:INITIALIZE:EXECUTE`

### :INITialize:UNDO

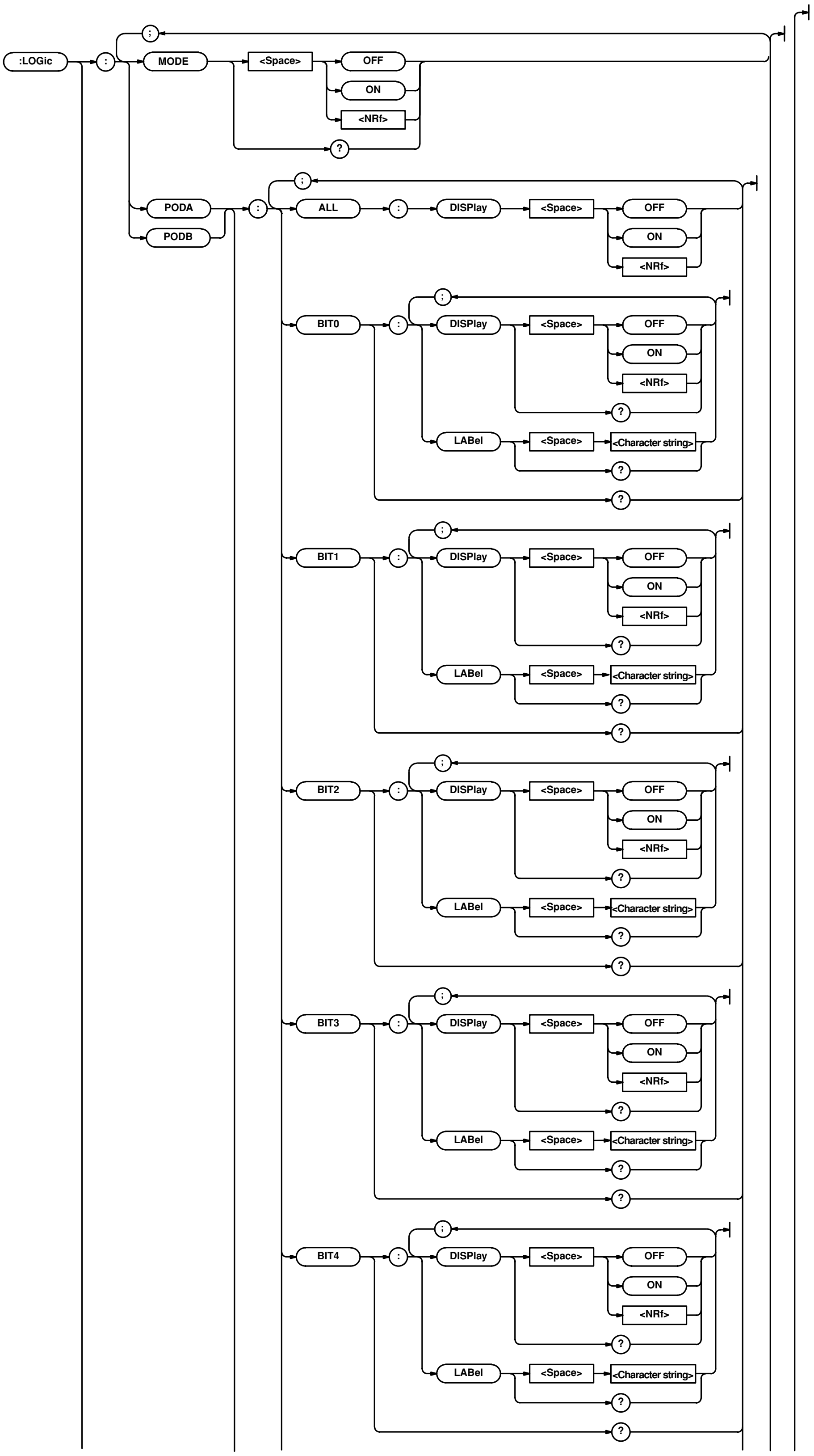
Function Nullifies initializing.

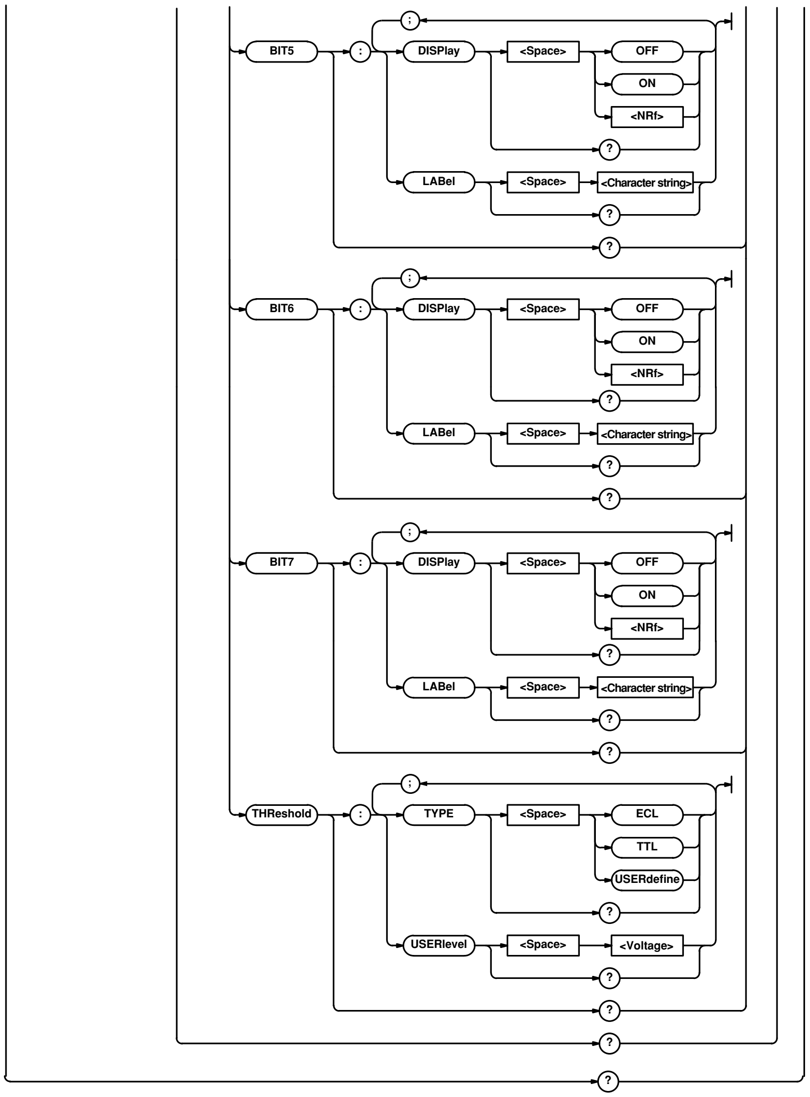
Syntax `:INITialize:UNDO`

Example `:INITIALIZE:UNDO`

## 5.16 LOGic Group (Option)

The commands in the LOGic group deal with the logic probe. These commands can be used to make the same settings and inquiries as when the LOGIC key on the front panel is pressed. However, if the logic probe is not installed, these settings and inquiries cannot be made.




BIT5
:
;
DISPlay
<Space>
OFF
ON
<NRf>
?
LABel
<Space>
<Character string>
?
?
BIT6
:
;
DISPlay
<Space>
OFF
ON
<NRf>
?
LABel
<Space>
<Character string>
?
?
BIT7
:
;
DISPlay
<Space>
OFF
ON
<NRf>
?
LABel
<Space>
<Character string>
?
?
THReshold
:
;
TYPE
<Space>
ECL
TTL
USERdefine
?
USERlevel
<Space>
<Voltage>
?
?
?
?

## :LOGic?

Function Queries all settings related to the logic probe.

Syntax
```
:LOGic?
```

Example
```
:LOGIC?→:LOGIC:MODE
1;:PODA:BIT0:DISPLAY 0;LABEL "A-0";:
LOGIC:PODA:BIT1:DISPLAY 0;LABEL "A-1";:
LOGIC:PODA:BIT2:DISPLAY 0;LABEL "A-2";:
LOGIC:PODA:BIT3:DISPLAY 0;LABEL "A-3";:
LOGIC:PODA:BIT4:DISPLAY 0;LABEL "A-4";:
LOGIC:PODA:BIT5:DISPLAY 0;LABEL "A-5";:
LOGIC:PODA:BIT6:DISPLAY 0;LABEL "A-6";:
LOGIC:PODA:BIT7:DISPLAY 0;LABEL "A-7";:
LOGIC:PODA:THRESHOLD:TYPE TTL;
USERLEVEL 1.4E+00;:LOGIC:PODB:BIT0:
DISPLAY 0;LABEL "B-0";:LOGIC:PODB:BIT1:
DISPLAY 0;LABEL "B-1";:LOGIC:PODB:BIT2:
DISPLAY 0;LABEL "B-2";:LOGIC:PODB:BIT3:
DISPLAY 0;LABEL "B-3";:LOGIC:PODB:BIT4:
DISPLAY 0;LABEL "B-4";:LOGIC:PODB:BIT5:
DISPLAY 0;LABEL "B-5";:LOGIC:PODB:BIT6:
DISPLAY 0;LABEL "B-6";:LOGIC:PODB:BIT7:
DISPLAY 0;LABEL "B-7";:LOGIC:PODB:
THRESHOLD:TYPE TTL;USERLEVEL 1.4E+00
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:MODE

Function Turns the logic mode ON/OFF or queries the current setting.

Syntax
```
:LOGic:MODE {<Boolean>}
:LOGic:MODE?
```

Example
```
:LOGIC:MODE ON
:LOGIC:MODE?→:LOGIC:MODE 1
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}?

Function Queries all settings related to the specified logic probe.

Syntax
```
:LOGic:{PODA|PODB}?
```

Example
```
:LOGIC:PODA?→:LOGIC:PODA:BIT0:
DISPLAY 0;LABEL "A-0";:LOGIC:PODA:BIT1:
DISPLAY 0;LABEL "A-1";:LOGIC:PODA:BIT2:
DISPLAY 0;LABEL "A-2";:LOGIC:PODA:BIT3:
DISPLAY 0;LABEL "A-3";:LOGIC:PODA:BIT4:
DISPLAY 0;LABEL "A-4";:LOGIC:PODA:BIT5:
DISPLAY 0;LABEL "A-5";:LOGIC:PODA:BIT6:
DISPLAY 0;LABEL "A-6";:LOGIC:PODA:BIT7:
DISPLAY 0;LABEL "A-7";:LOGIC:PODA:
THRESHOLD:TYPE TTL;USERLEVEL 1.4E+00
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}:ALL:DISPlay

Function Turns the display of all the bits of the specified logic probe ON/OFF.

Syntax
```
:LOGic:ALL:DISPlay {<Boolean>}
```

Example
```
:LOGIC:ALL:DISPLAY ON
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}:{BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BIT6|BIT7}:DISPlay

Function Displays the bits of the specified logic probe or queries the current setting.

Syntax
```
:LOGic:{PODA|PODB}:{BIT0|BIT1|BIT2|BIT3|
BIT4|BIT5|BIT6|BIT7}:DISPlay {<Boolean>}
:LOGic:{PODA|PODB}:{BIT0|BIT1|BIT2|BIT3|
BIT4|BIT5|BIT6|BIT7}:DISPlay?
```

Example
```
:LOGIC:PODA:BIT1:DISPLAY ON
:LOGIC:PODA:BIT1:DISPLAY?→:LOGIC:PODA:
BIT1:DISPLAY 1
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}:{BIT0|BIT1|BIT2|BIT3|BIT4|BIT5|BIT6|BIT7}:LABel

Function Sets a label to the bits of the specified logic probe or queries the current setting.

Syntax
```
:LOGic:{PODA|PODB}:{BIT0|BIT1|BIT2|BIT3|
BIT4|BIT5|BIT6|BIT7}:LABel <Character
string>
:LOGic:{PODA|PODB}:{BIT0|BIT1|BIT2|BIT3|
BIT4|BIT5|BIT6|BIT7}:LABel?
<Character string>=Within 8 characters.
```

Example
```
:LOGIC:PODA:BIT1:LABEL "NO_1"
:LOGIC:PODA:BIT1:LABEL?→:LOGIC:PODA:
BIT1:LABEL 1
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}:THReshold?

Function Queries all settings related to the threshold level.

Syntax
```
:LOGic:{PODA|PODB}:THReshold?
```

Example
```
:LOGIC:PODA:THRESHOLD?→:LOGIC:PODA:
THRESHOLD:TYPE TTL;USERLEVEL 1.400E+00
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}:THReshold:TYPE

Function Executes the threshold level preset or queries the current setting.

Syntax
```
:LOGic:{PODA|PODB}:THReshold:TYPE {ECL|
TTL|USERdefine}
:LOGic:{PODA|PODB}:THReshold:TYPE?
```

Example
```
:LOGIC:PODA:THRESHOLD:TYPE ECL
:LOGIC:PODA:THRESHOLD:TYPE?→:LOGIC:
PODA:THRESHOLD:TYPE ECL
```

Description An error results if the model does not support the logic probe (option).

## :LOGic:{PODA|PODB}:THReshold:USERlevel

Function Sets the user-defined threshold level or queries the current setting.

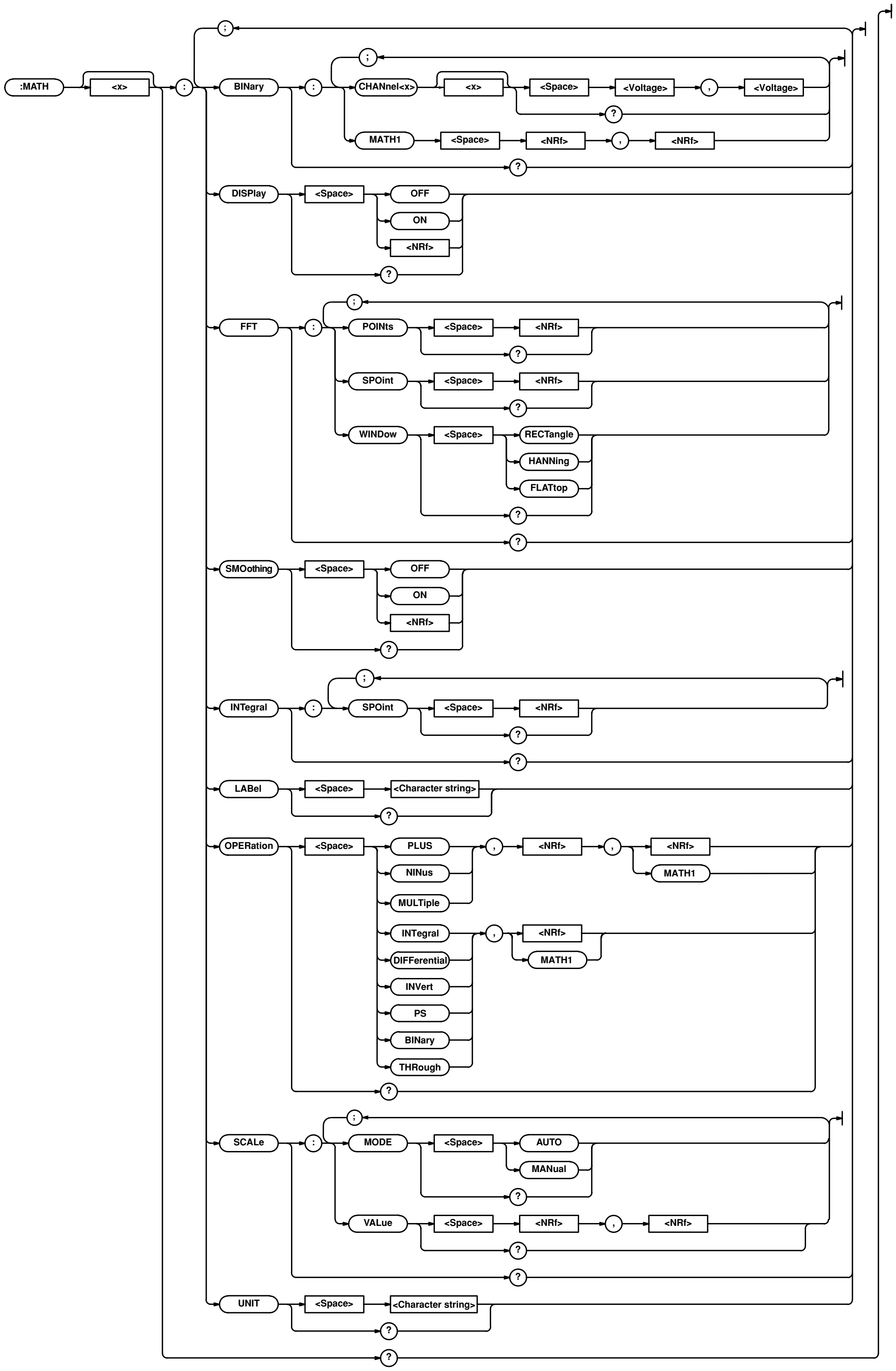
Syntax
```
:LOGic:{PODA|PODB}:THReshold:USERlevel
{<voltage>}
:LOGic:{PODA|PODB}:THReshold:USERlevel?
<voltage>=−10 to 10 V (0.1 V steps)
```

Example
```
:LOGIC:PODA:THRESHOLD:USERLEVEL 1V
:LOGIC:PODA:THRESHOLD:USERLEVEL?→
:LOGIC:PODA:THRESHOLD:USERLEVEL 1.0E+00
```

Description An error results if the model does not support the logic probe (option).

## 5.17 MATH Group

The commands in the MATH group are used to make settings and queries about computation. You can make the same settings that you can make using the MATH key on the front panel.



### :MATH<x>?

Function Queries all settings relating to computation.

Syntax
```
:MATH<x>?
<x>=1, 2
```

Example
```
:MATH1?→:MATH1:DISPLAY 0;SCALE:
MODE AUTO;
VALUE 400.000E+00,-400.000E+00;:
MATH1:OPERATION PLUS,1,2;FFT:
POINTS 1000;WINDOW HANNING;
SPOINT -5.0000000;:MATH1:BINARY:
CHANNEL1 0.0E+00,0.0E+00;
CHANNEL2 0.0E+00,0.0E+00;
CHANNEL3 0.0E+00,0.0E+00;
CHANNEL4 0.0E+00,0.0E+00;
MATH1 0.00,0.00;:MATH1:INTEGRAL:
SPOINT -5.0000000;:MATH1:SMOOTHING 0;
UNIT "EU";LABEL "Math1"
```

### :MATH<x>:BINary?

Function Queries all settings relating to binary computation.

Syntax
```
:MATH<x>:BINary?
<x>=1 or 2
```

Example
```
:MATH1:BINARY?→:MATH1:BINARY:
CHANNEL1 0.0E+00,0.0E+00;
CHANNEL2 0.0E+00,0.0E+00;
CHANNEL3 0.0E+00,0.0E+00;
CHANNEL4 0.0E+00,0.0E+00;MATH1 0.00,0.00
```

### :MATH<x>:BINary:{CHANnel<x>|MATH1}

Function Sets/queries the threshold level for binary computation of the channel.

Syntax
```
:MATH<x>:BINary:{CHANnel<x>|MATH1}
{<Voltage>,<Voltage>|<NRf>,<NRf>}
:MATH<x>:BAINary:{CHANnel<x>|MATH1}?
The <x> in MATH <x>=1 or 2
The <x> in CHANnel <x>=1 to 4
<Voltage>= 8 div on the screen (0.01 div
           steps)
<NRf>=–4 to 4 div (0.01 div steps)
```

Example
```
:MATH1:BINARY:CHANNEL1 0V,0V
:MATH1:BINARY:CHANNEL1?→:MATH1:
BINARY:CHANNEL1 0.000E+00,0.000E+00
```

Description The ":MATH<x>:BINary:MATH1" command can be used when <x>=2. If <x>=1 an error occurs.

### :MATH<x>:DISPlay

Function Sets whether or not to display computation waveforms or queries the current setting.

Syntax
```
:MATH<x>:DISPlay {<Boolean>}
:MATH<x>:DISPlay?
<x>=1 or 2
```

Example
```
:MATH1:DISPLAY ON
:MATH1:DIPSLAY?→:MATH1:DISPLAY 1
```

### :MATH<x>:FFT?

Function Queries all FFT computation settings.

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

Example
```
:MATH1:FFT?→:MATH1:FFT:POINTS 1000;
WINDOW RECTANGLE;SPOINT -5.0000000
```

### :MATH<x>:FFT:POINt

Function Sets/queries the number of points for FFT computation.

Syntax
```
:MATH<x>:FFT:POINt {<NRf>}
:MATH<x>:FFT:POINt?
<x>=1 or 2
<NRf>=1000, 10000
```

Example
```
:MATH1:FFT:POINT 1000
:MATH1:FFT:POINT?→:MATH1:FFT:POINT 1000
```

### :MATH<x>:FFT:SPOint (Start Point)

Function Sets the start point of the FFT or queries the current setting.

Syntax
```
:MATH<x>:FFT:SPOint {<NRf>}
:MATH<x>:FFT:POINt?
<x>=1 or 2
<NRf>=–5 to 5 div (10 div/displayed
      record length steps)
```

Example
```
:MATH1:FFT:SPOINT 1.5
:MATH1:FFT:SPOINT?→:MATH1:FFT:
SPOINT 1.5000000
```

### :MATH<x>:FFT:WINDow

Function Sets/queries FFT window function.

Syntax
```
:MATH<x>:FFT:WINDow {FLATtop|HANNing|
RECTangle}
:MATH<x>:FFT:WINDow?
<x>=1 or 2
```

Example
```
:MATH1:FFT:WINDOW HANNING
:MATH1:FFT:WINDOW?→:MATH1:FFT:
WINDOW HANNING
```

### :MATH<x>:INTegral?

Function Queries all settings related to integration.

Syntax :MATH<x>:INTegral?
```
<x>=1 or 2
```

Example
```
:MATH1:INTEGRAL?→:MATH1:INTEGRAL:
SPOINT -5.0000000
```

### :MATH<x>:INTegral:SPOint (Start Point)

Function Sets the start point of the integration or queries the current setting.

Syntax
```
:MATH<x>:INTegral:SPOint {<NRf>}
:MATH<x>:INTegral:POINt?
<x>=1 or 2
<NRf>=–5 to 5 div (10 div/displayed
      record length steps)
```

Example
```
:MATH1:INTEGRAL:SPOINT 1.5
:MATH1:INTEGARL:SPOINT?→:MATH1:
INTEGARL:SPOINT 1.5000000
```

### :MATH<x>:LABel

Function Sets/queries the math's

Syntax
```
:MATH<x>:LABel {<Character string>}
:MATH<x>:LABel?
<x>=1 or 2
<Character string>=Within 8 characters
```

Example
```
:MATH1:LABEL "MATH1"
:MATH1:LABEL?→:MATH1:LABEL "MATH1"
```

### :MATH<x>:OPERation

Function Sets the type of computation or queries the current setting.

Syntax
```
:MATH<x>:OPERation {PLUS|MINus|MULTiple|
BINary|INVert|DIFFerential|INTegral|PS|
THRough},{<NRf>}[,{<NRf>|MATH1}]
:MATH<x>:OPERation?
<x>=1 or 2
<NRf>=1 to 4
```

Example
```
:MATH1:OPERATION PLUS,1,2
:MATH1:OPERATION?→:MATH1:
OPERATION PLUS,1,2
```

Description
- For unary operators (BINary|INVert|DIFFerential|INTegral|PS|THRough), select the waveform in the first <NRf>.
- For binomial operators (PLUS|MINus|MULTiple), select the first waveform in the first <NRf> and the second waveform in the second <NRf>.

### :MATH<x>:SCALe?

Function Queries all scaling setting values.

Syntax
```
:MATH<x>:SCALe?
<x>=1 to 8
```

Example
```
:MATH1:SCALE?→:MATH1:SCALE:MODE AUTO;
VALUE 200.000E+00,0.0E+00
```

### :MATH<x>:SCALe:MODE

Function Sets/queries the scaling method.

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

Example
```
:MATH1:SCALE:MODE AUTO
:MATH1:SCALE:MODE?→:MATH1:SCALE:
MODE AUTO
```

### :MATH<x>:SCALe:VALue

Function Sets/queries the upper and lower limits for the manual scaling.

Syntax
```
:MATH<x>:SCALe:VALue {<NRf>,<NRf>}
:MATH<x>:SCALe:VALue?
<x>=1 or 2
<NRf>=–1.0000E+30 to 1.0000E+30
```

Example
```
:MATH1:SCALE:VALUE 1,-1
:MATH1:SCALE:VALUE?→:MATH1:SCALE:
VALUE 1.0000E+00,-1.0000E+00
```

Description The larger of the two values automatically becomes the upper limit.

### :MATH<x>:SMOothing

Function Turns smoothing ON/OFF or queries the current setting.

Syntax
```
:MATH<x>:SMOothing {<Boolean>}
:MATH<x>:SMOothing?
<x>=1 or 2
```

Example
```
:MATH1:SMOOTHING ON
:MATH1:SMOOTHING?→:MATH1:SMOOTHING 1
```

Description This command is void, if "PS" is selected with the "MATH<x>:OPERation" command.

### :MATH<x>:UNIT

Function Sets/queries the dimensional unit appended to computation result.

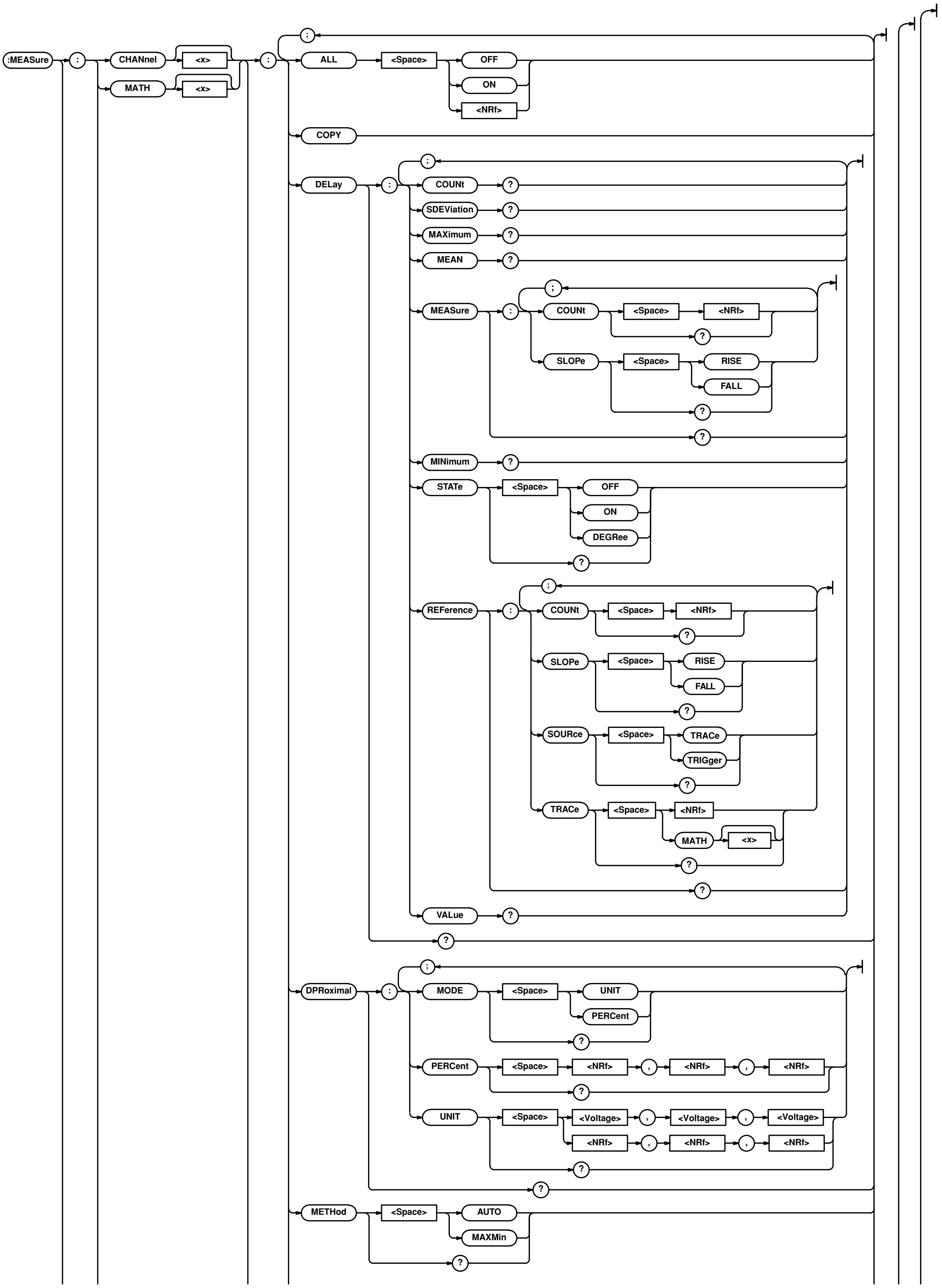
Syntax
```
:MATH<x>:UNIT {<Character string>}
:MATH<x>:UNIT?
<x>=1 or 2
<Character string>=Within 4 characters
```

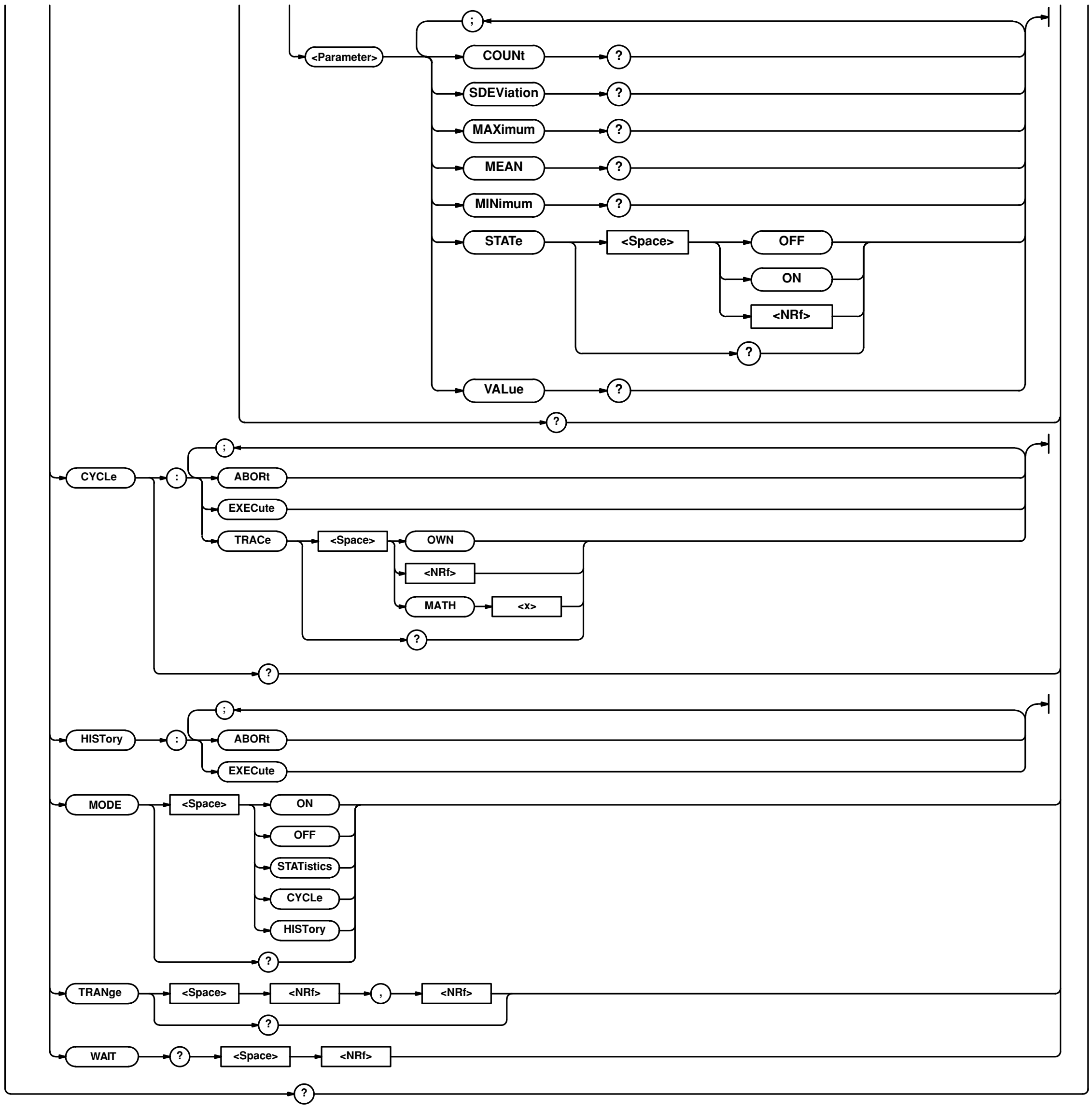
Example
```
:MATH1:UNIT "EU"
:MATH1:UNIT?→:MATH1:UNIT "EU"
```

Description The unit is reflected in the scale value. The unit setting has no effect on the calculation result.

## 5.18 MEASure Group

The commands in the MEASure group are used to make settings and queries about automatic measurement of waveform parameters. You can make the same settings which you can make using the MEASURE key on the front panel.





<Parameter>
;
COUNt
?
SDEViation
?
MAXimum
?
MEAN
?
MINimum
?
STATe
<Space>
OFF
ON
<NRf>
?
VALue
?
?
CYCLe
:
;
ABORt
EXECute
TRACe
<Space>
OWN
<NRf>
MATH
<x>
?
?
HISTory
:
;
ABORt
EXECute
MODE
<Space>
ON
OFF
STATistics
CYCLe
HISTory
?
TRANge
<Space>
<NRf>
,
<NRf>
?
WAIT
?
<Space>
<NRf>
?

## :MEASure?

Function Queries all settings relating to automatic measurement of waveform parameters.

Syntax :MEASure?

Example

```
:MEASURE?→:MEASURE:MODE OFF;CHANNEL1:
AVERAGE:STATE 0;:MEASURE:CHANNEL1:
AVGFREQ:STATE 0;:MEASURE:CHANNEL1:
AVGPERIOD:STATE 0;:MEASURE:CHANNEL1:
BWIDTH1:STATE 0;:MEASURE:CHANNEL1:
BWIDTH2:STATE 0;:MEASURE:CHANNEL1:
DUTYCYCLE:STATE 0;:MEASURE:CHANNEL1:
FALL:STATE 0;:MEASURE:CHANNEL1:
FREQUENCY:STATE 0;:MEASURE:CHANNEL1:
HIGH:STATE 0;:MEASURE:CHANNEL1:LOW:
STATE 0;:MEASURE:CHANNEL1:MAXIMUM:
STATE 0;:MEASURE:CHANNEL1:MINIMUM:
STATE 0;:MEASURE:CHANNEL1:NOVERSHOOT:
STATE 0;:MEASURE:CHANNEL1:NWIDTH:
STATE 0;:MEASURE:CHANNEL1:PERIOD:
STATE 0;:MEASURE:CHANNEL1:PNUMBER:
STATE 0;:MEASURE:CHANNEL1:POVERSHOOT:
STATE 0;:MEASURE:CHANNEL1:PTOPEAK:
STATE 0;:MEASURE:CHANNEL1:PWIDTH:
STATE 0;:MEASURE:CHANNEL1:RISE:STATE 0;:
MEASURE:CHANNEL1:RMS:STATE 0;:MEASURE:
CHANNEL1:SDEVIATION:STATE 0;:MEASURE:
CHANNEL1:TY1INTEG:STATE 0;:MEASURE:
CHANNEL1:TY2INTEG:STATE 0;:MEASURE:
CHANNEL1:XY1INTEG:STATE 0;:MEASURE:
CHANNEL1:XY2INTEG:STATE 0;:MEASURE:
CHANNEL1:DELAY:STATE 0;MEASURE:
SLOPE RISE;COUNT 1;:MEASURE:CHANNEL1:
DELAY:REFERENCE:TRACE 1;SLOPE RISE;
COUNT 1;:MEASURE:CHANNEL1:DPROXIMAL:
MODE PERCENT;PERCENT 10.0,50.0,90.0;
UNIT -150.000E+00,0.0E+00,150.000E+00;:
MEASURE:CHANNEL1:METHOD AUTO;:MEASURE:
CHANNEL2:AVERAGE:STATE 0;:MEASURE:
CHANNEL2:AVGFREQ:STATE 0;:MEASURE:
CHANNEL2:AVGPERIOD:STATE 0;:MEASURE:
CHANNEL2:BWIDTH1:STATE 0;:MEASURE:
CHANNEL2:BWIDTH2:STATE 0;:MEASURE:
CHANNEL2:DUTYCYCLE:STATE 0;:MEASURE:
CHANNEL2:FALL:STATE 0;:MEASURE:
CHANNEL2:FREQUENCY:STATE 0;:MEASURE:
CHANNEL2:HIGH:STATE 0;:MEASURE:
CHANNEL2:LOW:STATE 0;:MEASURE:
CHANNEL2:MAXIMUM:STATE 0;:MEASURE:
CHANNEL2:MINIMUM:STATE 0;:MEASURE:
CHANNEL2:NOVERSHOOT:STATE 0;:MEASURE:
CHANNEL2:NWIDTH:STATE 0;:MEASURE:
CHANNEL2:PERIOD:STATE 0;:MEASURE:
CHANNEL2:PNUMBER:STATE 0;:MEASURE:
CHANNEL2:POVERSHOOT:STATE 0;:MEASURE:
CHANNEL2:PTOPEAK:STATE 0;:MEASURE:
CHANNEL2:PWIDTH:STATE 0;:MEASURE:
CHANNEL2:RISE:STATE 0;:MEASURE:
CHANNEL2:RMS:STATE 0;:MEASURE:
CHANNEL2:SDEVIATION:STATE 0;:MEASURE:
CHANNEL2:TY1INTEG:STATE 0;:MEASURE:
CHANNEL2:TY2INTEG:STATE 0;:MEASURE:
CHANNEL2:XY1INTEG:STATE 0;:MEASURE:
CHANNEL2:XY2INTEG:STATE 0;:MEASURE:
CHANNEL2:DELAY:STATE 0;MEASURE:
SLOPE RISE;COUNT 1;:MEASURE:CHANNEL2:
DELAY:REFERENCE:TRACE 2;SLOPE RISE;
COUNT 1;:MEASURE:CHANNEL2:DPROXIMAL:
MODE PERCENT;PERCENT 10.0,50.0,90.0;
UNIT -150.000E+00,0.0E+00,150.000E+00;:
MEASURE:CHANNEL2:METHOD AUTO;:MEASURE:
CHANNEL3:AVERAGE:STATE 0;:MEASURE:
CHANNEL3:AVGFREQ:STATE 0;:MEASURE:
CHANNEL3:AVGPERIOD:STATE 0;:MEASURE:
CHANNEL3:BWIDTH1:STATE 0;:MEASURE:
CHANNEL3:BWIDTH2:STATE 0;:MEASURE:
CHANNEL3:DUTYCYCLE:STATE 0;:MEASURE:
CHANNEL3:FALL:STATE 0;:MEASURE:
CHANNEL3:FREQUENCY:STATE 0;:MEASURE:
CHANNEL3:HIGH:STATE 0;:MEASURE:
CHANNEL3:LOW:STATE 0;:MEASURE:
CHANNEL3:MAXIMUM:STATE 0;:MEASURE:
CHANNEL3:MINIMUM:STATE 0;:MEASURE:
CHANNEL3:NOVERSHOOT:STATE 0;:MEASURE:
CHANNEL3:NWIDTH:STATE 0;:MEASURE:
CHANNEL3:PERIOD:STATE 0;:MEASURE:
CHANNEL3:PNUMBER:STATE 0;:MEASURE:
CHANNEL3:POVERSHOOT:STATE 0;:MEASURE:
CHANNEL3:PTOPEAK:STATE 0;:MEASURE:
CHANNEL3:PWIDTH:STATE 0;:MEASURE:
CHANNEL3:RISE:STATE 0;:MEASURE:
CHANNEL3:RMS:STATE 0;:MEASURE:CHANNEL3:
SDEVIATION:STATE 0;:MEASURE:CHANNEL3:
TY1INTEG:STATE 0;:MEASURE:CHANNEL3:
TY2INTEG:STATE 0;:MEASURE:CHANNEL3:
XY1INTEG:STATE 0;:MEASURE:CHANNEL3:
XY2INTEG:STATE 0;:MEASURE:CHANNEL3:
DELAY:STATE 0;MEASURE:SLOPE RISE;
COUNT 1;:MEASURE:CHANNEL3:DELAY:
REFERENCE:TRACE 3;SLOPE RISE;COUNT 1;:
MEASURE:CHANNEL3:DPROXIMAL:MODE PERCENT;
PERCENT 10.0,50.0,90.0;
UNIT -150.000E+00,0.0E+00,150.000E+00;:
MEASURE:CHANNEL3:METHOD AUTO;:MEASURE:
CHANNEL4:AVERAGE:STATE 0;:MEASURE:
CHANNEL4:AVGFREQ:STATE 0;:MEASURE:
CHANNEL4:AVGPERIOD:STATE 0;:MEASURE:
CHANNEL4:BWIDTH1:STATE 0;:MEASURE:
CHANNEL4:BWIDTH2:STATE 0;:MEASURE:
CHANNEL4:DUTYCYCLE:STATE 0;:MEASURE:
CHANNEL4:FALL:STATE 0;:MEASURE:
CHANNEL4:FREQUENCY:STATE 0;:MEASURE:
CHANNEL4:HIGH:STATE 0;:MEASURE:CHANNEL4:
LOW:STATE 0;:MEASURE:CHANNEL4:MAXIMUM:
STATE 0;:MEASURE:CHANNEL4:MINIMUM:
```

```
STATE 0;:MEASURE:CHANNEL4:NOVERSHOOT:
STATE 0;:MEASURE:CHANNEL4:NWIDTH:
STATE 0;:MEASURE:CHANNEL4:PERIOD:
STATE 0;:MEASURE:CHANNEL4:PNUMBER:
STATE 0;:MEASURE:CHANNEL4:POVERSHOOT:
STATE 0;:MEASURE:CHANNEL4:PTOPEAK:
STATE 0;:MEASURE:CHANNEL4:PWIDTH:
STATE 0;:MEASURE:CHANNEL4:RISE:STATE 0;:
MEASURE:CHANNEL4:RMS:STATE 0;:MEASURE:
CHANNEL4:SDEVIATION:STATE 0;:MEASURE:
CHANNEL4:TY1INTEG:STATE 0;:MEASURE:
CHANNEL4:TY2INTEG:STATE 0;:MEASURE:
CHANNEL4:XY1INTEG:STATE 0;:MEASURE:
CHANNEL4:XY2INTEG:STATE 0;:MEASURE:
CHANNEL4:DELAY:STATE 0;MEASURE:
SLOPE RISE;COUNT 1;:MEASURE:CHANNEL4:
DELAY:REFERENCE:TRACE 4;SLOPE RISE;
COUNT 1;:MEASURE:CHANNEL4:DPROXIMAL:
MODE PERCENT;PERCENT 10.0,50.0,90.0;
UNIT -150.000E+00,0.0E+00,150.000E+00;:
MEASURE:CHANNEL4:METHOD AUTO;:MEASURE:
MATH1:AVERAGE:STATE 0;:MEASURE:MATH1:
AVGFREQ:STATE 0;:MEASURE:MATH1:
AVGPERIOD:STATE 0;:MEASURE:MATH1:
BWIDTH1:STATE 0;:MEASURE:MATH1:BWIDTH2:
STATE 0;:MEASURE:MATH1:DUTYCYCLE:
STATE 0;:MEASURE:MATH1:FALL:STATE 0;:
MEASURE:MATH1:FREQUENCY:STATE 0;:
MEASURE:MATH1:HIGH:STATE 0;:MEASURE:
MATH1:LOW:STATE 0;:MEASURE:MATH1:
MAXIMUM:STATE 0;:MEASURE:MATH1:MINIMUM:
STATE 0;:MEASURE:MATH1:NOVERSHOOT:
STATE 0;:MEASURE:MATH1:NWIDTH:STATE 0;:
MEASURE:MATH1:PERIOD:STATE 0;:MEASURE:
MATH1:PNUMBER:STATE 0;:MEASURE:MATH1:
POVERSHOOT:STATE 0;:MEASURE:MATH1:
PTOPEAK:STATE 0;:MEASURE:MATH1:PWIDTH:
STATE 0;:MEASURE:MATH1:RISE:STATE 0;:
MEASURE:MATH1:RMS:STATE 0;:MEASURE:
MATH1:SDEVIATION:STATE 0;:MEASURE:MATH1:
TY1INTEG:STATE 0;:MEASURE:MATH1:
TY2INTEG:STATE 0;:MEASURE:MATH1:
XY1INTEG:STATE 0;:MEASURE:MATH1:
XY2INTEG:STATE 0;:MEASURE:MATH1:DELAY:
STATE 0;MEASURE:SLOPE RISE;COUNT 1;:
MEASURE:MATH1:DELAY:REFERENCE:
TRACE MATH1;SLOPE RISE;COUNT 1;:MEASURE:
MATH1:DPROXIMAL:MODE PERCENT;
PERCENT 10.0,50.0,90.0;
UNIT -3.00,0.00,3.00;:MEASURE:MATH1:
METHOD AUTO;:MEASURE:MATH2:AVERAGE:
STATE 0;:MEASURE:MATH2:AVGFREQ:STATE 0;:
MEASURE:MATH2:AVGPERIOD:STATE 0;:
MEASURE:MATH2:BWIDTH1:STATE 0;:MEASURE:
MATH2:BWIDTH2:STATE 0;:MEASURE:MATH2:
DUTYCYCLE:STATE 0;:MEASURE:MATH2:FALL:
STATE 0;:MEASURE:MATH2:FREQUENCY:
STATE 0;:MEASURE:MATH2:HIGH:STATE 0;:
MEASURE:MATH2:LOW:STATE 0;:MEASURE:
MATH2:MAXIMUM:STATE 0;:MEASURE:MATH2:
MINIMUM:STATE 0;:MEASURE:MATH2:
NOVERSHOOT:STATE 0;:MEASURE:MATH2:
NWIDTH:STATE 0;:MEASURE:MATH2:PERIOD:
STATE 0;:MEASURE:MATH2:PNUMBER:STATE 0;:
MEASURE:MATH2:POVERSHOOT:STATE 0;:
MEASURE:MATH2:PTOPEAK:STATE 0;:MEASURE:
MATH2:PWIDTH:STATE 0;:MEASURE:MATH2:
RISE:STATE 0;:MEASURE:MATH2:RMS:
STATE 0;:MEASURE:MATH2:SDEVIATION:
STATE 0;:MEASURE:MATH2:TY1INTEG:
STATE 0;:MEASURE:MATH2:TY2INTEG:
STATE 0;:MEASURE:MATH2:XY1INTEG:
STATE 0;:MEASURE:MATH2:XY2INTEG:
STATE 0;:MEASURE:MATH2:DELAY:STATE 0;
MEASURE:SLOPE RISE;COUNT 1;:MEASURE:
MATH2:DELAY:REFERENCE:TRACE MATH2;
SLOPE RISE;COUNT 1;:MEASURE:MATH2:
DPROXIMAL:MODE PERCENT;
PERCENT 10.0,50.0,90.0;
UNIT -3.00,0.00,3.00;:MEASURE:MATH2:
METHOD AUTO;:MEASURE:CYCLE:TRACE 1;:
MEASURE:TRANGE -4.0000000,4.0000000
```

## :MEASure:{CHANnel<x>|MATH<x>}?

Function Queries all specified parameter ON/OFF settings.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}?
<x> (CHANnel)=1 to 4
<x> (MATH)=1 or 2
```

Example (An example for CH1 is given below.)
```
:MEASURE:CHANNEL1?→:MEASURE:CHANNEL1:
AVERAGE:STATE 0;:MEASURE:CHANNEL1:
AVGFREQ:STATE 0;:MEASURE:CHANNEL1:
AVGPERIOD:STATE 0;:MEASURE:CHANNEL1:
BWIDTH1:STATE 0;:MEASURE:CHANNEL1:
BWIDTH2:STATE 0;:MEASURE:CHANNEL1:
DUTYCYCLE:STATE 0;:MEASURE:CHANNEL1:
FALL:STATE 0;:MEASURE:CHANNEL1:
FREQUENCY:STATE 0;:MEASURE:CHANNEL1:
HIGH:STATE 0;:MEASURE:CHANNEL1:LOW:
STATE 0;:MEASURE:CHANNEL1:MAXIMUM:
STATE 0;:MEASURE:CHANNEL1:MINIMUM:
STATE 0;:MEASURE:CHANNEL1:NOVERSHOOT:
STATE 0;:MEASURE:CHANNEL1:NWIDTH:
STATE 0;:MEASURE:CHANNEL1:PERIOD:
STATE 0;:MEASURE:CHANNEL1:PNUMBER:
STATE 0;:MEASURE:CHANNEL1:POVERSHOOT:
STATE 0;:MEASURE:CHANNEL1:PTOPEAK:
STATE 0;:MEASURE:CHANNEL1:PWIDTH:
STATE 0;:MEASURE:CHANNEL1:RISE:STATE 0;:
MEASURE:CHANNEL1:RMS:STATE 0;:MEASURE:
CHANNEL1:SDEVIATION:STATE 0;:MEASURE:
CHANNEL1:TY1INTEG:STATE 0;:MEASURE:
CHANNEL1:TY2INTEG:STATE 0;:MEASURE:
CHANNEL1:XY1INTEG:STATE 0;:MEASURE:
CHANNEL1:XY2INTEG:STATE 0;:MEASURE:
CHANNEL1:DELAY:STATE 0;MEASURE:
SLOPE RISE;COUNT 1;:MEASURE:CHANNEL1:
DELAY:REFERENCE:TRACE 1;SLOPE RISE;
COUNT 1;:MEASURE:CHANNEL1:DPROXIMAL:
MODE PERCENT;PERCENT 10.0,50.0,90.0;
UNIT -150.000E+00,0.0E+00,150.000E+00;:
MEASURE:CHANNEL1:METHOD AUTO 0
```

## :MEASure:{CHANnel<x>|MATH<x>}:ALL

Function Turns all the measurement parameters of the waveform ON/OFF at once.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:ALL
{<Boolean>}
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example (The following example is for CH1.)
```
:MEASURE:CHANNEL1:ALL OFF
```

## :MEASure:{CHANnel<x>|MATH<x>}:COPY

Function Copies the ON/OFF state of all measurement parameters of one waveform to all other waveforms.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:COPY
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:COPY
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay?

Function Queries all settings related to the delay between waveform channels.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY?→:MEASURE:
CHANNEL1:DELAY:STATE 0;MEASURE:
SLOPE RISE;COUNT 1;:MEASURE:CHANNEL1:
DELAY:REFERENCE:TRACE 1;SLOPE RISE;
COUNT 1
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay: MEASure?

Function Queries all settings related to the measured waveform when measuring the delay between channels.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
MEASure?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:MEASURE?→
:MEASURE:CHANNEL1:MEASURE:SLOPE RISE;
COUNT 1
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay: MEASure:COUNt

Function Sets the edge count of the measured waveform when measuring the delay between channels or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
MEASure:COUNt {<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
MEASure:COUNt?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<NRf>=1 to 9
```

Example
```
:MEASURE:CHANNEL1:DELAY:MEASURE:COUNT 2
:MEASURE:CHANNEL1:DELAY:MEASURE:
COUNT?→:MEASURE:CHANNEL1:DELAY:MEASURE:
COUNT 2
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:MEASure:SLOPe

Function Sets the slope of the measured waveform when measuring the delay between channels or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
MEASure:SLOPe {RISE|FALL}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
MEASure:SLOPe?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:MEASURE:
SLOPE RISE
:MEASURE:CHANNEL1:DELAY:MEASURE:
SLOPE?→:MEASURE:CHANNEL1:DELAY:MEASURE:
SLOPE RISE
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence?

Function Queries all settings related to the reference waveform when measuring the delay between channels.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE?→
:MEASURE:CHANNEL1:DELAY:REFERENCE:
TRACE 1;SLOPE RISE;COUNT 1
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:COUNt

Function Sets the edge count of the reference waveform when measuring the delay between channels or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:COUNt {<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:COUNt?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<NRf>=1 to 9
```

Example
```
:MEASURE:TRACE1:DELAY:REFERENCE:COUNT 2
:MEASURE:TRACE1:DELAY:REFERENCE:COUNT?→
:MEASURE:TRACE1:DELAY:REFERENCE:COUNT 2
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SLOPe

Function Sets the slope of the reference waveform when measuring the delay between channels or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:SLOPe {RISE|FALL}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:SLOPe?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:
SLOPE RISE
:MEASURE:CHANNEL1:DELAY:REFERENCE:
SLOPE?→:MEASURE:CHANNEL1:DELAY:
REFERENCE:SLOPE RISE
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:SOURce

Function Sets whether or not to set the reference to the trigger point or the waveform when measuring the delay between channels or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:SOURce {TRACe|TRIGger}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:SOURce?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:
SOURCE TRACE
:MEASURE:CHANNEL1:DELAY:REFERENCE:
SOURCE?→:MEASURE:CHANNEL1:DELAY:
REFERENCE:SOURCE TRACE
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:REFerence:TRACe

Function Sets the reference trace when measuring the delay between channels or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:TRACe {<NRf>|MATH<x>}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
REFerence:TRACe?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:REFERENCE:
TRACE 2
:MEASURE:CHANNEL1:DELAY:REFERENCE:
TRACE?→:MEASURE:CHANNEL1:DELAY:
REFERENCE:TRACE 2
```

## :MEASure:{CHANnel<x>|MATH<x>}:DELay:STATe

Function Turns the delay parameter ON/OFF and sets the display format or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
STATe {OFF|ON|DEGRee}
:MEASure:{CHANnel<x>|MATH<x>}:DELay:
STATe?
The <X> in CHANnel<x>=1 to 4
The <X> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DELAY:STATE ON
:MEASURE:CHANNEL1:DEALY:STATE?→:
MEASURE:CHANNEL1:DELAY:STATE ON
```

## :MEASure:{CHANnel<x>|MATH<x>}:DPRoximal?

Function Queries all settings related to distal, mesial, and proximal values.

Syntax
```
MEASure:{CHANnel<x>|MATH<x>}:DPRoximal?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DPROXIMAL?→:MEASURE:
CHANNEL1:DPROXIMAL:MODE PERCENT;
PERCENT 10.0,50.0,90.0;
UNIT -2.000E+00,0.0E+00,2.000E+00
```

## :MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:MODE

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

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:
MODE {PERCent|UNIT}
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:
MODE?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:MODE PERCENT
:MEASURE:CHANNEL1:DPROXIMAL:MODE?→
:MEASURE:CHANNEL1:DPROXIMAL:MODE PERCENT
```

## :MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:PERCent

Function Sets the distal, mesial, and proximal points in terms of a percentage or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:
PERCent {<NRf>,<NRf>,<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:
PERCent?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<NRf>=0 to 100 (%, 1% steps)
```

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:
PERCENT 80,60,40
:MEASURE:CHANNEL1:DPROXIMAL:PERCENT?→
:MEASURE:CHANNEL1:DPROXIMAL:
PERCENT 80,60,40
```

## :MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:UNIT

Function Sets the distal, mesial, and proximal points in terms of a voltage or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:
UNIT {<voltage>,<voltage>,<voltage>|
<NRf>,<NRf>,<NRf>}
:MEASure:{CHANnel<x>|MATH<x>}:DPRoximal:
UNIT?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<voltage>= 8 div on the screen (0.01 div
           steps)
<NRf>=-4 to 4 div (0.01 div steps)
```

Example
```
:MEASURE:CHANNEL1:DPROXIMAL:
UNIT 50V,0V,-50V
:MEASURE:CHANNEL1:DPROXIMAL:UNIT?→
:MEASURE:CHANNEL1:DPROXIMAL:
UNIT 50.0E+00,0.0E+00,-50.0E+00
```

## :MEASure:{CHANnel<x>|MATH<x>}:METHod

Function Sets the High and Low points or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:METHod
{AUTO|MAXMin}
:MEASure:{CHANnel<x>|MATH<x>}:METHod?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
```

Example
```
:MEASURE:CHANNEL1:METHOD AUTO
:MEASURE:CHANNEL1:METHOD?→:MEASURE:
CHANNEL1:METHOD AUTO
```

### :MEASure:{CHANnel<x>|MATH<x>}:<parameter>?

Function Queries all settings related to the parameters of each waveform.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:
<parameter>?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<parameter>= {AVERage|AVGFreq|AVGPeriod|
             BWIDth1|BWIDth2|DUTYcycle|
             FALL|FREQuency|HIGH|LOW|
             MAXimum|MINimum|NOVershoot|
             NWIDth|PERiod|PNUMber|
             POVershoot|PTOPeak|PWIDth|
             RISE|RMS|SDEViation|
             TY1Integ|TY2Integ|XY1Integ|
             XY2Integ}
```

Example (The following example is for the average value of CH1.)
```
:MEASURE:CHANNEL1:AVERAGE?→:MEASURE:
CHANNEL1:AVERAGE:STATE 1
```

### :MEASure:{CHANnel<x>|MATH<x>}:<parameter>:COUNt?

Function Queries the number of statistical processing of each parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:
<parameter>:COUNt?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<parameter>= {AVERage|AVGFreq|AVGPeriod|
             BWIDth1|BWIDth2|DELay|
             DUTYcycle|FALL|FREQuency|
             HIGH|LOW|MAXimum|MINimum|
             NOVershoot|NWIDth|PERiod|
             PNUMber|POVershoot|PTOPeak|
             PWIDth|RISE|RMS|SDEViation|
             TY1Integ|TY2Integ|XY1Integ|
             XY2Integ}
```

Example (The following example is for the average value of CH1.)
```
:MEASURE:CHANNEL1:AVERAGE:COUNT?→
:MEASURE:CHANNEL1:AVERAGE:COUNT 100
```

### :MEASure:{CHANnel<x>|MATH<x>}:<parameter>:{MAXimum|MEAN|MINimum|SDEViation}?

Function Queries the statistics of each parameter.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:
<parameter>:{MAXimum|MEAN|MINimum|
SDEViation}?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<parameter>= {AVERage|AVGFreq|AVGPeriod|
             BWIDth1|BWIDth2|DELay|
             DUTYcycle|FALL|FREQuency|
             HIGH|LOW|MAXimum|MINimum|
             NOVershoot|NWIDth|PERiod|
             PNUMber|POVershoot|PTOPeak|
             PWIDth|RISE|RMS|SDEViation|
             TY1Integ|TY2Integ|XY1Integ|
             XY2Integ}
```

Example (The following example is for the maximum value.)
```
:MEASURE:CHANNEL1:AVERAGE:MAXIMUM?→
:MEASURE:CHANNEL1:AVERAGE:
MAXIMUM -40.0000E-06
```

Description If the statistics cannot be found, "NAN(Not A Number)" is returned.

### :MEASure:{CHANnel<x>|MATH<x>}:<parameter>:STATe

Function Turns the parameter ON/OFF one by one or queries the current setting.

Syntax
```
:MEASure:{CHANnel<x>|MATH<x>}:
<parameter>:STATe {<Boolean>}
:MEASure:{CHANnel<x>|MATH<x>}:
<parameter>:STATe?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2
<parameter>= {AVERage|AVGFreq|AVGPeriod|
             BWIDth1|BWIDth2|DUTYcycle|
             FALL|FREQuency|HIGH|LOW|
             MAXimum|MINimum|NOVershoot|
             NWIDth|PERiod|PNUMber|
             POVershoot|PTOPeak|PWIDth|
             RISE|RMS|SDEViation|
             TY1Integ|TY2Integ|XY1Integ|
             XY2Integ}
```

Example (The following example is for the average value of CH1.)
```
:MEASURE:CHANNEL1:AVERAGE:STATE ON
:MEASURE:CHANNEL1:AVERAGE:STATE?→
:MEASURE:CHANNEL1:AVERAGE:STATE 1
```

### :MEASure:{CHANnel<x>|MATH<x>}:<parameter>:VALue?

Function Queries the value of the automated measurement of each parameter.

```
Syntax      :MEASure:{CHANnel<x>|MATH<x>}:
            <parameter>:VALue?
            The <x> in CHANnel<x>=1 to 4
            The <x> in MATH<x>=1 or 2
            <parameter>= {AVERage|AVGFreq|AVGPeriod|
                          BWIDth1|BWIDth2|DELay|
                          DUTYcycle|FALL|FREQuency|
                          HIGH|LOW|MAXimum|MINimum|
                          NOVershoot|NWIDth|PERiod|
                          PNUMber|POVershoot|PTOPeak|
                          PWIDth|RISE|RMS|SDEViation|
                          TY1Integ|TY2Integ|XY1Integ|
                          XY2Integ}
Example     (The following example is for the
            average value of CH1.)
            :MEASURE:CHANNEL1:AVERAGE:VALUE?→
            :MEASURE:CHANNEL1:AVERAGE:
            VALUE 4.950E+00
```

Description If the measurement is not possible, "NAN(Not A Number)" is returned.

### :MEASure:CYCLe?

Function Queries all settings related to the periodic statistical processing of the automated measurement of waveform parameters.

```
Syntax      :MEASure:CYCLe?
Example     :MEASURE:CYCLE?→:MEASURE:CYCLE:TRACE 1
```

### :MEASure:CYCLe:ABORt

Function Aborts the periodic statistical processing of the automated measurement of waveform parameters.

```
Syntax      :MEASure:CYCLe:ABORt
Example     :MEASURE:CYCLE:ABORT
```

### :MEASure:CYCLe:EXECute

Function Executes the periodic statistical processing of the automated measurement of waveform parameters.

```
Syntax      :MEASure:CYCLe:EXECute
Example     :MEASURE:CYCLE:EXECUTE
```

### :MEASure:CYCLe:TRACe

Function Sets the waveform used to determine the period for the periodic statistical processing of automated measurement of waveform parameters or queries the current setting.

```
Syntax      :MEASure:CYCLe:TRACe {OWN|<NRf>|MATH<x>}
            :MEASure:CYCLe:TRACe?
            <NRf>=1 to 4
            <x>=1 or 2
Example     :MEASURE:CYCLE:TRACE 1
            :MEASURE:CYCLE:TRACE?→:MEASURE:CYCLE:
            TRACE 1
```

### :MEASure:HISTory:ABORt

Function Aborts the statistical processing within the historical data of the automated measurement of waveform parameters.

```
Syntax      :MEASure:HISTory:ABORt
Example     :MEASURE:HISTORY:ABORT
```

### :MEASure:HISTory:EXECute

Function Executes the statistical processing within the historical data of the automated measurement of waveform parameters.

```
Syntax      :MEASure:HISTory:EXECute
Example     :MEASURE:HISTORY:EXECUTE
```

### :MEASure:MODE

Function Turns the automated measurement of waveform parameters ON/OFF and sets the statistical processing or queries the current setting.

```
Syntax      :MEASure:MODE {OFF|ON|STATistics|CYCLE|
            HISTory}
            :MEASure:MODE?
Example     :MEASURE:MODE ON
            :MEASURE:MODE?→:MEASURE:MODE ON
```

### :MEASure:TRANge (Time Range)

Function Sets the measurement range or queries the current setting.

```
Syntax      :MEASure:TRANge {<NRf>,<NRf>}
            :MEASure:TRANge?
            <NRf>=-5 to 5 div (10 div/displayed
                  record length steps)
Example     :MEASURE:TRANGE -4,4
            :MEASURE:TRANGE?→:MEASURE:
            TRANGE -4.00E+00,4.00E+00
```

### :MEASure:WAIT?

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

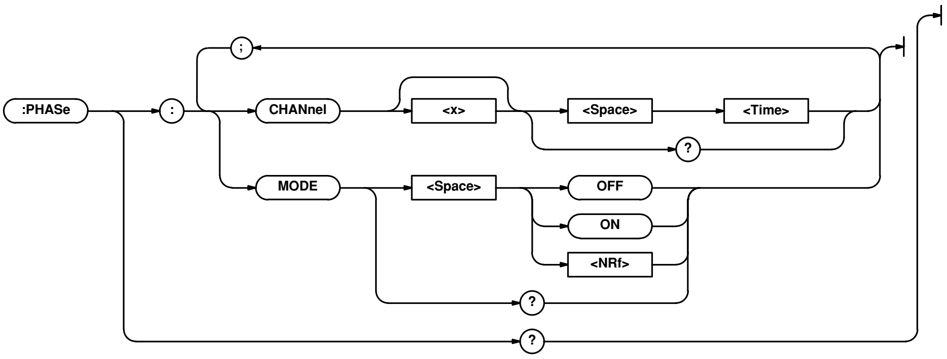
Syntax `:MEASure:WAIT? {<NRf>}`
`<NRf>=1 to 360000 (timeout period, 10 ms steps)`

Example `:MEASURE:WAIT? 100→:MEASURE:WAIT 1`

Description
- Returns "0" if the automated measurement completes within the timeout period or "1" if it is not complete or automated measurement is not in progress.
- Even if the timeout period is sufficiently long, "0" is returned when the automated measurement completes.

## 5.19 PHASe Group

The commands in the PHASe group deal with the phase shift function. These commands can be used to make the same settings and inquiries as when the SHIFT key + MATH key on the front panel is pressed.



### :PHASe?

Function Queries all settings related to the phase shift function.

Syntax `:PHASe?`

Example `:PHASe?→:PHASE:MODE 0;CHANNEL1 0.0E+00;CHANNEL2 0.0E+00;CHANNEL3 0.0E+00;CHANNEL4 0.0E+00`

### :PHASe:CHANnel<x>

Function Sets the amount of phase shift of the waveform or queries the current setting.

Syntax `:PHASe:CHANnel<x> {<time>}`
`:PHASe:CHANnel<x>?`
`<x>=1 to 4`
`<time>=time between -T/div*5 and T/div*5`

Example `:PHASE:CHANNEL2 1us`
`:PHASE:CHANNEL2?→:PHASE:CHANNEL2 1.000000E-06`

Description
- Phase shift may not be possible on some channels depending on the interleave mode setting.
- When the time base is set to external clock, the setting and query is made in terms of the number of points.

### :PHASe:MODE

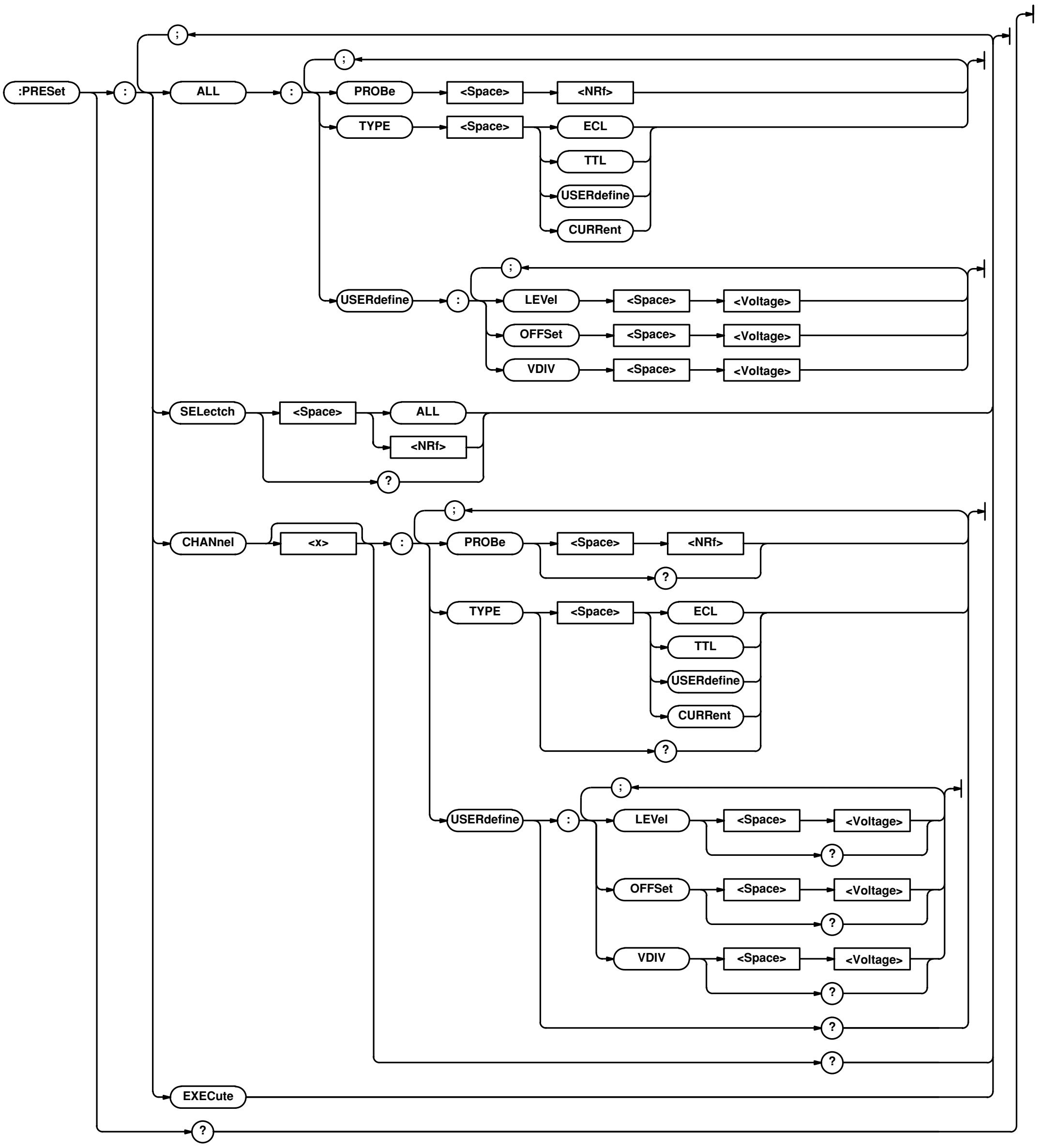
Function Turns the phase shift function ON/OFF or queries the current setting.

Syntax `:PHASe:MODE {<Boolean>}`
`:PHASe:MODE?`

Example `:PHASE:MODE ON`
`:PHASE:MODE?→:PHASE:MODE 1`

## 5.20 PRESet Group

The commands in the PRESet group deal with the preset function.  These commands can be used to make the same settings and inquiries as when the PRESET key on the front panel is pressed.



## :PRESet?

Function Queries all settings related to the preset function.

Syntax :PRESet?

Example :PRESET?→:PRESET:SELECTCH 1;CHANNEL1:TYPE TTL;USERDEFINE:VDIV 50.000E+00;LEVEL 0.0E+00;OFFSET 0.0E+00;:PRESET:CHANNEL1:PROBE 10;:PRESET:CHANNEL2:TYPE TTL;USERDEFINE:VDIV 50.000E+00;LEVEL 0.0E+00;OFFSET 0.0E+00;:PRESET:CHANNEL2:PROBE 10;:PRESET:CHANNEL3:TYPE TTL;USERDEFINE:VDIV 50.000E+00;LEVEL 0.0E+00;OFFSET 0.0E+00;:PRESET:CHANNEL3:PROBE 10;:PRESET:CHANNEL4:TYPE TTL;USERDEFINE:VDIV 50.000E+00;LEVEL 0.0E+00;OFFSET 0.0E+00;:PRESET:CHANNEL4:PROBE 10

## :PRESet:ALL:PROBe

Function Sets the preset values of the probe attenuation of all channels.

Syntax :PRESet:ALL:PROBe {<NRf>}
<NRf>=1,10,100,1000

Example :PRESET:ALL:PROBE 10

## :PRESet:ALL:TYPE

Function Sets preset values to all channels.

Syntax :PRESet:ALL:TYPE {ECL|TTL|USERdefine|CURRent}

Example :PRESET:ALL:TYPE USERDEFINE

## :PRESet:ALL:USERdefine:LEVel

Function Sets the preset trigger level of all channels.

Syntax :PRESet:ALL:USERdefine:LEVel {<voltage>}
<voltage>= 8 div on the screen (0.01 div steps)

Example :PRESET:ALL:USERDEFINE:LEVEL 0V

## :PRESet:ALL:USERdefine:OFFSet

Function Sets the preset offset voltage of all channels.

Syntax :PRESet:ALL:USERdefine:OFFSet {<voltage>}
<voltage>= See the IM701410-01E User's Manual.

Example :PRESET:ALL:USERDEFINE:OFFSET 0V

## :PRESet:ALL:USERdefine:VDIV

Function Sets the preset V/div value of all channels.

Syntax :PRESet:ALL:USERdefine:VDIV {<voltage>}
<voltage>= 2 mV to 10 V (when the probe attenuation is 1:1, See the IM701410-01E User's Manual.)

Example :PRESET:ALL:USERDEFINE:VDIV 2V

## :PRESet:CHANnel<x>?

Function Queries the preset information of an arbitrary channel.

Syntax :PRESet:CHANnel<x>?
<x>=1 to 4

Example :PRESET:CHANNEL1?→:PRESET:CHANNEL1:TYPE TTL;USERDEFINE:VDIV 50.000E+00;LEVEL 0.0E+00;OFFSET 0.0E+00;:PRESET:CHANNEL1:PROBE 10

## :PRESet:CHANnel<x>:PROBe

Function Sets the preset probe attenuation of an arbitrary channel or queries the current setting.

Syntax :PRESet:CHANnel<x>:PROBe {<NRf>}
:PRESet:CHANnel<x>:PROBe?
<x>=1 to 4
<NRf>=1,10,100,1000

Example :PRESET:CHANNEL1:PROBE 1
:PRESET:CHANNEL1:PROBE?→:PRESET:CHANNEL1:PROBE 1

## :PRESet:CHANnel<x>:TYPE

Function Sets the preset type to the channel or queries the current setting.

Syntax :PRESet:CHANnel<x>:TYPE {ECL|TTL|USERdefine|CURRent}
:PRESet:CHANnel<x>:TYPE?
<x>=1 to 4

Example :PRESET:CHANNEL1:TYPE USERDEFINE
:PRESET:CHANNEL1:TYPE?→:PRESET:CHANNEL1:TYPE USERDEFINE

## :PRESet:CHANnel<x>:USERdefine?

Function Queries all preset values when using User preset.

Syntax :PRESet:CHANnel<x>:USERdefine?
<x>=1 to 4

Example :PRESET:CHANNEL1:USERDEFINE?→:PRESET:CHANNEL1:USERDEFINE:VDIV 50.000E+00;LEVEL 0.0E+00;OFFSET 0.0E+00

## :PRESet:CHANnel<x>:USERdefine:LEVel

Function Sets the preset trigger level of the channel or queries the current setting.

Syntax :PRESet:CHANNEL<x>:USERdefine:LEVel {<voltage>}
:PRESet:CHANNEL<x>:USERdefine:LEVel?
<x>=1 to 4
<voltage>= 8 div on the screen (0.01 div steps)

Example :PRESET:CHANNEL1:USERDEFINE:LEVEL 0V
:PRESET:CHANNEL1:USERDEFINE:LEVEL?→
:PRESET:CHANNEL1:USERDEFINE:LEVEL 0.000E+00

## :PRESet:CHANnel<x>:USERdefine:OFFSet

Function Sets the preset offset voltage of the channel or queries the current setting.

```
Syntax      :PRESet:CHANNEL<x>:USERdefine:OFFSet
            {<voltage>}
            :PRESet:CHANNEL<x>:USERdefine:OFFSet?
            <x>=1 to 4
            <voltage>= See the IM701410-01E User's
                       Manual.
Example     :PRESET:CHANNEL1:USERDEFINE:OFFSET 0V
            :PRESET:CHANNEL1:USERDEFINE:OFFSET?→
            :PRESET:CHANNEL1:USERDEFINE:
            OFFSET 0.000E+00
```

## :PRESet:CHANnel<x>:USERdefine:VDIV

Function Sets the preset V/div value of the channel or queries the current setting.

```
Syntax      :PRESet:CHANNEL<x>:USERdefine:VDIV
            {<voltage>}
            :PRESet:CHANNEL<x>:USERdefine:VDIV?
            <x>=1 to 4
            <voltage>= 2 mV to 10 V(when the probe
                       attenuation is 1:1, See the
                       IM701410-01E User's Manual.)
Example     :PRESET:CHANNEL1:USERDEFINE:VDIV 2V
            :PRESET:CHANNEL1:USERDEFINE:VDIV?→
            :PRESET:CHANNEL1:USERDEFINE:
            VDIV 2.000E+00
```

## :PRESet:EXECute

Function Executes preset.

```
Syntax      :PRESet:EXECute
Example     :PRESET:EXECUTE
```

Description Preset the channel which is selected by the "PRESet:SELectch" command.

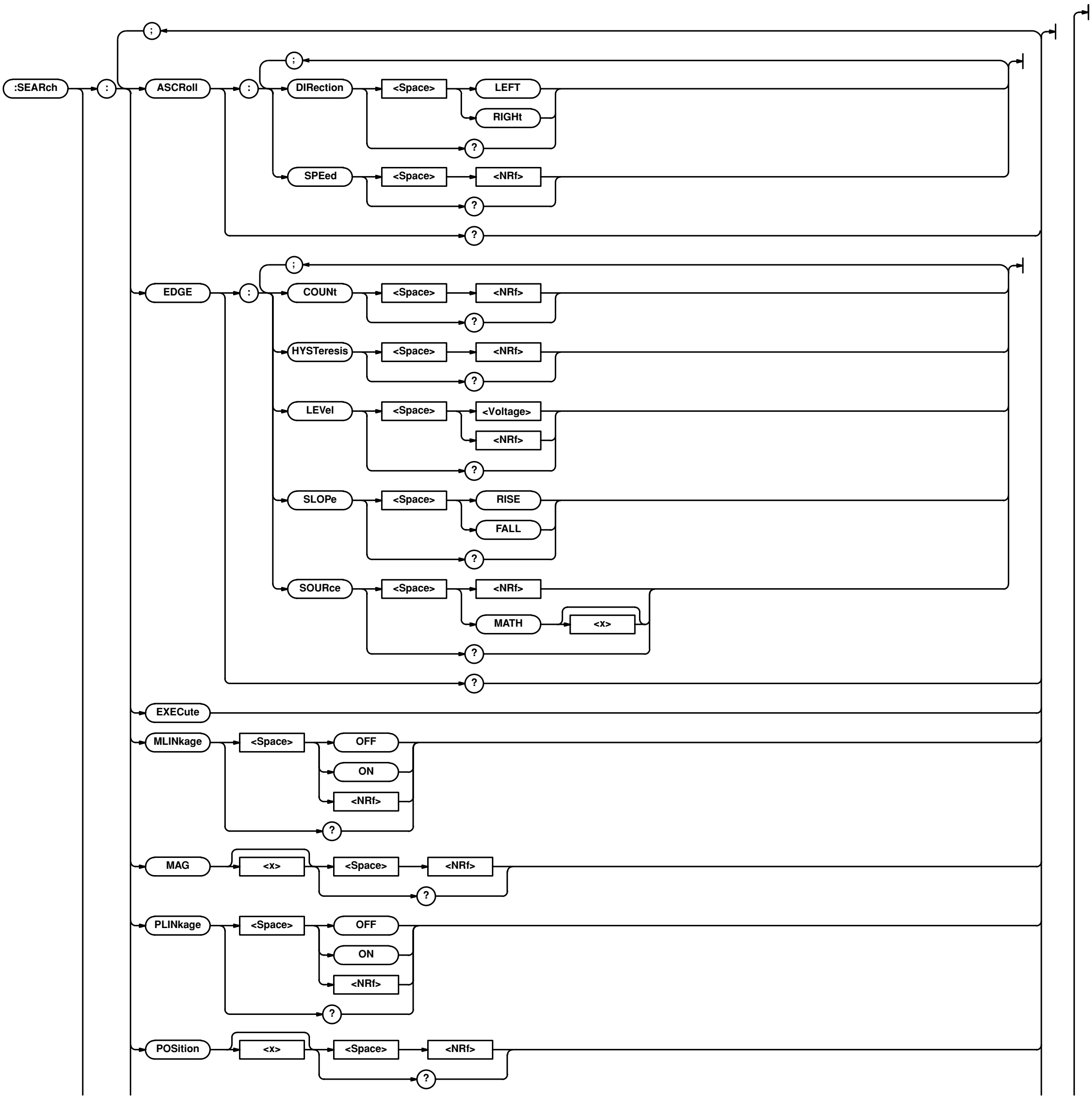
## :PRESet:SELectch

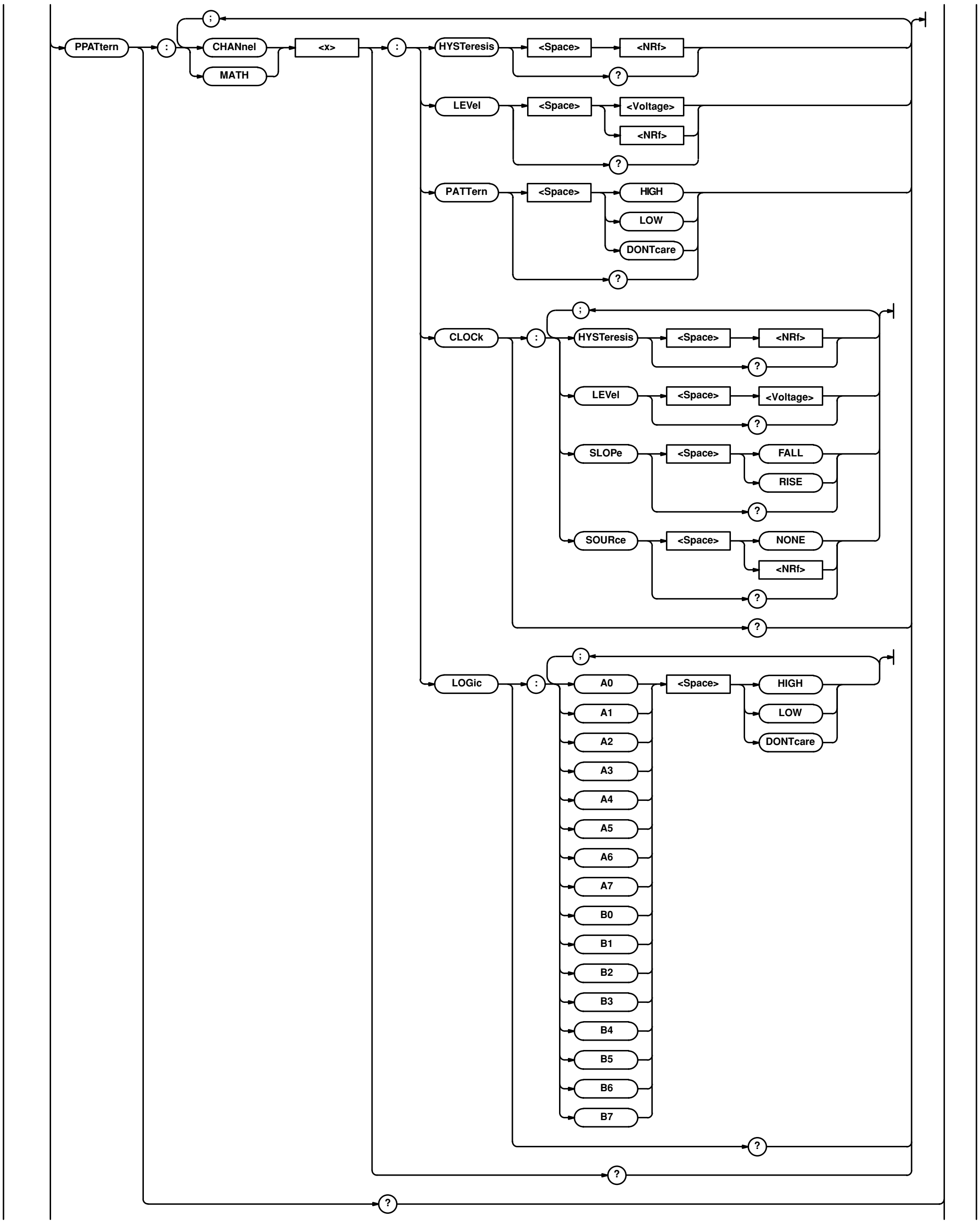
Function Sets the channel to be preset or queries the current setting.

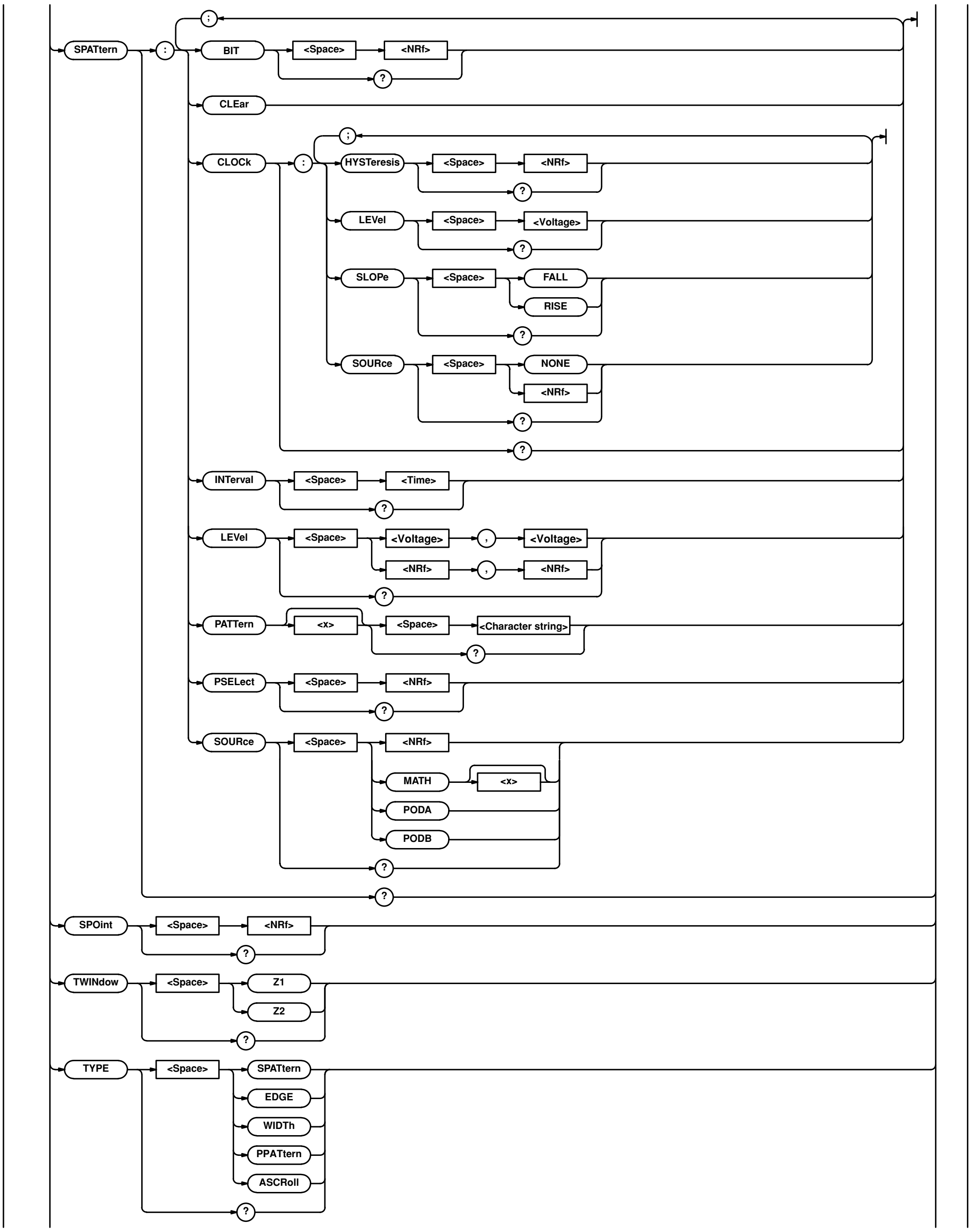
```
Syntax      :PRESet:SELectch {ALL|<NRf>}
            :PRESet:SELectch?
            <NRf>=1 to 4
Example     :PRESET:SELECTCH 1
            :PRESET:SELECTCH?→:PRESET:SELECTCH 1
```

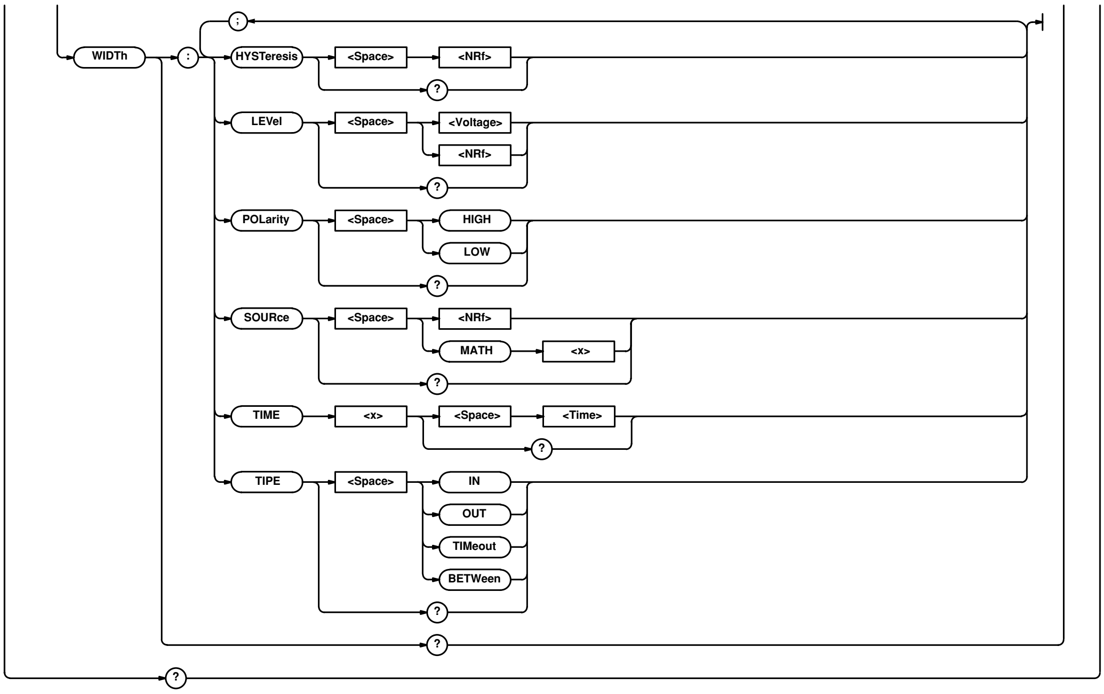
# 5.21 SEARch Group

The commands in the SEARch group are used to search an arbitrary pattern in the acquired waveform. These commands can be used to carry out the same operations or to make the same settings and inquiries as when the SHIFT key + ZOOM key on the front panel is pressed.




PPATtern
:
;
CHANnel
MATH
<x>
:
HYSTeresis
<Space>
<NRf>
?
LEVel
<Space>
<Voltage>
<NRf>
?
PATTern
<Space>
HIGH
LOW
DONTcare
?
CLOCk
:
;
HYSTeresis
<Space>
<NRf>
?
LEVel
<Space>
<Voltage>
?
SLOPe
<Space>
FALL
RISE
?
SOURce
<Space>
NONE
<NRf>
?
?
LOGic
:
;
A0
A1
A2
A3
A4
A5
A6
A7
B0
B1
B2
B3
B4
B5
B6
B7
<Space>
HIGH
LOW
DONTcare
?
?
?


SPATtern
:
;
BIT
<Space>
<NRf>
?
CLEar
CLOCk
:
;
HYSTeresis
<Space>
<NRf>
?
LEVel
<Space>
<Voltage>
?
SLOPe
<Space>
FALL
RISE
?
SOURce
<Space>
NONE
<NRf>
?
?
INTerval
<Space>
<Time>
?
LEVel
<Space>
<Voltage>
,
<Voltage>
<NRf>
,
<NRf>
?
PATTern
<x>
<Space>
<Character string>
?
PSELect
<Space>
<NRf>
?
SOURce
<Space>
<NRf>
MATH
<x>
PODA
PODB
?
?
SPOint
<Space>
<NRf>
?
TWINdow
<Space>
Z1
Z2
?
TYPE
<Space>
SPATtern
EDGE
WIDTh
PPATtern
ASCRoll
?


WIDTh
:
;
HYSTeresis
<Space>
<NRf>
?
LEVel
<Space>
<Voltage>
<NRf>
?
POLarity
<Space>
HIGH
LOW
?
SOURce
<Space>
<NRf>
MATH
<x>
?
TIME
<x>
<Space>
<Time>
?
TIPE
<Space>
IN
OUT
TIMeout
BETWeen
?
?
?

### :SEARch?

Function Queries all settings related to the search function.

Syntax
```
:SEARch?
```

Example
```
:SEARCH?→:SEARCH:TYPE EDGE;EDGE:
SOURCE 1;LEVEL 0.0E+00;SLOPE RISE;
HYSTERESIS 0.30;COUNT 1;:SEARCH:
SPATTERN:CLOCK:SOURCE NONE;:SEARCH:
SPATTERN:INTERVAL 10.000E-09;SOURCE 1;
LEVEL 0.0E+00,0.0E+00;PSELECT 1;
PATTERN1 "HXXXXXXX";PATTERN2 "XXXXXXXX";
PATTERN3 "XXXXXXXX";PATTERN4 "XXXXXXXX";
PATTERN5 "XXXXXXXX";PATTERN6 "XXXXXXXX";
PATTERN7 "XXXXXXXX";PATTERN8 "XXXXXXXX";
BIT 0;:SEARCH:WIDTH:TYPE IN;SOURCE 1;
LEVEL 0.0E+00;POLARITY HIGH;
HYSTERESIS 0.30;TIME1 1.000000E-06;
TIME2 1.000000E-06;:SEARCH:PPATTERN:
CLOCK:SOURCE NONE;:SEARCH:PPATTERN:
CHANNEL1:LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN HIGH;:SEARCH:PPATTERN:CHANNEL2:
LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
CHANNEL3:LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
CHANNEL4:LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:MATH1:
LEVEL 0.00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
MATH2:LEVEL 0.00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:LOGIC:
A0 DONTCARE;A1 DONTCARE;A2 DONTCARE;
A3 DONTCARE;A4 DONTCARE;A5 DONTCARE;
A6 DONTCARE;A7 DONTCARE;B0 DONTCARE;
B1 DONTCARE;B2 DONTCARE;B3 DONTCARE;
B4 DONTCARE;B5 DONTCARE;B6 DONTCARE;
B7 DONTCARE;:SEARCH:ASCROLL:
DIRECTION RIGHT;SPEED 0;:SEARCH:
TWINDOW Z1;MLINKAGE 1;MAG1 2.0;MAG2 2.0;
PLINKAGE 0;POSITION1 0.0000000;
POSITION2 0.0000000;SPOINT -5.0000000;
SELECT NAN
```

### :SEARch:ASCRoll?

Function Queries all settings related to the auto scroll function.

Syntax
```
:SEARch:ASCRoll?
```

Example
```
:SEARCH:ASCROLL?→:SEARCH:ASCROLL:
DIRECTION RIGHT;SPEED 0
```

### :SEARch:ASCRoll:DIRection

Function Sets the direction for moving the zoom box of the auto scroll or queries the current setting.

Syntax
```
:SEARch:ASCRoll:DIRection {LEFT|RIGHt}
:SEARch:ASCRoll:DIRection?
```

Example
```
:SEARCH:ASCROLL:DIRECTION LEFT
:SEARCH:ASCROLL:DIRECTION?→:SEARCH:
ASCROLL:DIRECTION LEFT
```

### :SEARch:ASCRoll:SPEed

Function Sets the speed for moving the zoom box of the auto scroll or queries the current setting.

Syntax
```
:SEARch:ASCRoll:SPEed {<NRf>}
:SEARch:ASCRoll:SPEed?
<NRf>=1 to 7
```

Example
```
:SEARCH:ASCROLL:SPEED 1
:SEARCH:ASCROLL:SPEED?→:SEARCH:ASCROLL:
SPEED 1
```

### :SEARch:EDGE?

Function Queries all settings related to the edge search.

Syntax
```
:SEARch:EDGE?
```

Example
```
:SEARCH:EDGE?→:SEARCH:EDGE:SOURCE 1;
LEVEL 0.0E+00;SLOPE RISE;HYSTERESIS LOW;
COUNT 1
```

### :SEARch:EDGE:COUNt

Function Sets the edge search count or queries the current setting.

Syntax
```
:SEARch:EDGE:COUNt {<NRf>}
:SEARch:EDGE:COUNt?
<NRf>=1 to 1000000
```

Example
```
:SEARCH:EDGE:COUNT 100
:SEARCH:EDGE:COUNT?→:SEARCH:EDGE:
COUNT 100
```

### :SEARch:EDGE:HYSTeresis

Function Sets the hysteresis on the edge search level or queries the current setting.

Syntax
```
:SEARch:EDGE:HYSTeresis {<NRf>}
:SEARch:EDGE:HYSTeresis?
<NRf>=0.3 to 4 div (0.1 steps)
```

Example
```
:SEARCH:EDGE:HYSTERESIS 1
:SEARCH:EDGE:HYSTERESIS?→:SEARCH:EDGE:
HYSTERESIS 1.0
```

## :SEARch:EDGE:LEVEl

Function Sets the edge search level or queries the current setting.

Syntax

```
:SEARch:EDGE:LEVEl {<voltage>|<NRf>}
:SEARch:EDGE:LEVEl?
<voltage>= 8 div on the screen (0.01 div
           steps)
<NRf>=-4 to 4 div (0.01 div steps)
```

Example

```
:SEARCH:EDGE:LEVEL -10mV
:SEARCH:EDGE:LEVEL?→:SEARCH:EDGE:
LEVEL -10.00E-03
```

Description Specify a voltage when the ":SEARch:EDGE:SOURce" is set to channel, <NRf> when it is set to MATH.

## :SEARch:EDGE:SLOPe

Function Sets the slope of the edge search or queries the current setting.

Syntax

```
:SEARch:EDGE:SLOPe {FALL|RISE}
:SEARch:EDGE:SLOPe?
```

Example

```
:SEARCH:EDGE:SLOPE RISE
:SEARCH:EDGE:SLOPE?→:SEARCH:EDGE:
SLOPE RISE
```

## :SEARch:EDGE:SOURce

Function Sets the waveform on which to perform the edge search or queries the current setting.

Syntax

```
:SEARch:EDGE:SOURce {<NRf>|MATH<x>}
:SEARch:EDGE:SOURce?
<NRf>=1 to 4
<x>=1 or 2
```

Example

```
:SEARCH:EDGE:SOURCE MATH1
:SEARCH:EDGE:SOURCE?→:SEARCH:EDGE:
SOURCE MATH1
```

## :SEARch:EXECute

Function Executes the search.  This is an overlap command.

Syntax

```
:SEARch:EXECute
```

Example

```
:SEARCH:EXECUTE
```

## :SEARch:MLINkage (Mag Linkage)

Function Turns the link ON/OFF when changing the zoom rate or queries the current setting.

Syntax

```
:SEARch:MLINkage {<Boolean>}
:SEARch:MLINkage?
```

Example

```
:SEARCH:MLINKAGE ON
:SEARCH:MLINKAGE?→:SEARCH:MLINKAGE 1
```

## :SEARch:MAG<x>

Function Sets the zoom rate of the zoom window or queries the current setting.

Syntax

```
:SEARch:MAG<x> {<NRf>}
:SEARch:MAG<x>?
<x>=1 or 2
<NRf>=Up to the point in which 40 points
      are displayed, See the IM701410-
      01E User's Manual.
```

Example

```
:SEARCH:MAG1 100
:SEARCH:MAG1?→:SEARCH:MAG1 100
```

## :SEARch:PLINkage (Position Linkage)

Function Turns the link ON/OFF when moving the zoom box or queries the current setting.

Syntax

```
:SEARch:PLINkage {<Boolean>}
:SEARch:PLINkage?
```

Example

```
:SEARCH:PLINKAGE OFF
:SEARCH:PLINKAGE?→:SEARCH:PLINKAGE 0
```

## :SEARch:POSition<x>

Function Sets the zoom box position or queries the current setting.

Syntax

```
:SEARch:POSition<x> {<NRf>}
:SEARch:POSition<x>?
<x>=1 or 2
<NRf>=-5 to 5 div (10 div/displayed
      record length steps)
```

Example

```
:SEARCH:POSITION1 2.5
:SEARCH:POSITION1?→:SEARCH:
POSITION1 2.5000000
```

## :SEARch:PPATtern?

Function Queries all settings related to the parallel pattern search.

Syntax
```
:SEARch:PPATtern?
```

Example
```
:SEARCH:PPATTERN?→:SEARCH:PPATTERN:
CLOCK:SOURCE NONE;:SEARCH:PPATTERN:
CHANNEL1:LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN HIGH;:SEARCH:PPATTERN:CHANNEL2:
LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
CHANNEL3:LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
CHANNEL4:LEVEL 0.0E+00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
MATH1:LEVEL 0.00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:
MATH2:LEVEL 0.00;HYSTERESIS 0.30;
PATTERN DONTCARE;:SEARCH:PPATTERN:LOGIC:
A0 DONTCARE;A1 DONTCARE;A2 DONTCARE;
A3 DONTCARE;A4 DONTCARE;A5 DONTCARE;
A6 DONTCARE;A7 DONTCARE;B0 DONTCARE;
B1 DONTCARE;B2 DONTCARE;B3 DONTCARE;
B4 DONTCARE;B5 DONTCARE;B6 DONTCARE;
B7 DONTCARE
```

## :SEARch:PPATtern:{CHANnel<x>|MATH<x>}?

Function Queries all settings related to the waveform of the parallel pattern search.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2.
```

Example
```
:SEARCH:PPATTERN:CHANNEL1?→:SEARCH:
PPATTERN:CHANNEL1:LEVEL 0.0E+00;
HYSTERESIS 0.30;PATTERN HIGH
```

## :SEARch:PPATtern:{CHANnel<x>|MATH<x>}:HYSTeresis

Function Sets the hysteresis on the waveform on which to perform the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:
HYSTeresis {<NRf>}
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:
HYSTeresis?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2.
<NRf>=0.3 to 4 div (0.1 steps)
```

Example
```
:SEARCH:PPATTERN:CHANNEL1:HYSTERESIS 0.3
:SEARCH:PPATTERN:CHANNEL1:HYSTERESIS?→:
SEARCH:PPATTERN:CHANNEL1:HYSTERESIS 0.30
```

## :SEARch:PPATtern:{CHANnel<x>|MATH<x>}:LEVel

Function Sets the search level on the waveform on which to perform the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:
LEVel {<voltage>|<NRf>}
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:
LEVel?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2.
<voltage>=8 div on the screen (when the
          waveform is CH1 to CH4)
<NRf>=-4 to 4 div (0.01 steps when the
      waveform is MATH1 or MATH2)
```

Example
```
:SEARCH:PPATTERN:CHANNEL1:LEVEL 1v
:SEARCH:PPATTERN:CHANNEL1:LEVEL?→:
SEARCH:PPATTERN:CHANNEL1:LEVEL 1.000E+00
```

## :SEARch:PPATtern:{CHANnel<x>|MATH<x>}:PATTern

Function Sets the search pattern on the waveform on which to perform the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:
PATTern {HIGH|LOW|DONTcare}
:SEARch:PPATtern:{CHANnel<x>|MATH<x>}:
PATTern?
The <x> in CHANnel<x>=1 to 4
The <x> in MATH<x>=1 or 2.
```

Example
```
:SEARCH:PPATTERN:CHANNEL1:PATTERN HIGH
:SEARCH:PPATTERN:CHANNEL1:PATTERN?→:
SEARCH:PPATTERN:CHANNEL1:PATTERN HIGH
```

## :SEARch:PPATtern:CLOCk?

Function Queries all settings related to the clock channel of the parallel pattern search.

Syntax
```
:SEARch:PPATtern:CLOCk?
```

Example
```
:SEARCH:PPATTERN:CLOCK?→:SEARCH:
PPATTERN:CLOCK:SOURCE 1;LEVEL 1.000E+00;
HYSTERESIS 0.30;SLOPE RISE
```

### :SEARch:PPATtern:CLOCk:HYSYeresis

Function Sets the hysteresis on the clock channel of the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:CLOCk:
HYSYeresis {<NRf>}
:SEARch:PPATtern:CLOCk:HYSYeresis?
<NRf>=0.3 to 4 div (0.1 steps)
```

Example
```
:SEARCH:PPATTERN:CLOCK:HYSTERESIS 4.0
:SEARCH:PPATTERN:CLOCK:HYSTERESIS?→:
SEARCH:PPATTERN:CLOCK:HYSTERESIS 4.00
```

Description An error occurs if "SEARch:PPATtern:CLOCk:SOURce" is set to "NONE."

### :SEARch:PPATtern:CLOCk:LEVel

Function Sets the search level of the clock channel of the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:CLOCk:LEVel {<voltage>}
:SEARch:PPATtern:CLOCk:LEVel?
8 divisions within the screen (0.01
division steps).
```

Example
```
:SEARCH:PPATTERN:CLOCK:LEVEL 1v
:SEARCH:PPATTERN:CLCOK:LEVEL?→:SEARCH:
PPATTERN:CLOCK:LEVEL 1.000E+00
```

Description An error occurs if "SEARch:PPATtern:CLOCk:SOURce" is set to "NONE."

### :SEARch:PPATtern:CLOCk:SLOPe

Function Sets the slope of the clock channel of the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:CLOCk:SLOPe {FALL|RISE}
:SEARch:PPATtern:CLOCk:SLOPe?
```

Example
```
:SEARCH:PPATTERN:CLOCK:SLOPE FALL
:SEARCH:PPATTERN:CLOCK:SLOPE?→:SEARCH:
PPATTERN:CLOCK:SLOPE FALL
```

Description An error occurs if "SEARch:PPATtern:CLOCk:SOURce" is set to "NONE."

### :SEARch:PPATtern:CLOCk:SOURce

Function Sets the waveform used as the clock channel of the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:CLOCk:SOURce {NONE|
<NRf>}
:SEARch:PPATtern:CLOCk:SOURce?
```

Example
```
:SEARCH:PPATTERN:CLOCK:SOURce 1
:SEARCH:PPATTERN:CLOCK:SOURce?→:SEARCH:
PPATTERN:CLOCK:SOURCE 1
```

### :SEARch:PPATtern:LOGic?

Function Queries all settings related to the logic channel of the parallel pattern search.

Syntax
```
:SEARch:PPATtern:LOGic?
```

Example
```
:SEARCH:PPATTERN:LOGIC?→:SEARCH:
PPATTERN:LOGIC:A0 DONTCARE;A1 DONTCARE;
A2 DONTCARE;A3 DONTCARE;A4 DONTCARE;
A5 DONTCARE;A6 DONTCARE;A7 DONTCARE;
B0 DONTCARE;B1 DONTCARE;B2 DONTCARE;
B3 DONTCARE;B4 DONTCARE;B5 DONTCARE;
B6 DONTCARE;B7 DONTCARE
```

### :SEARch:PPATtern:LOGic:{A0|A1|A2|A3|A4|A5|A6|A7|B0|B1|B2|B3|B4|B5|B6|B7}

Function Sets the bit condition of the clock channel of the parallel pattern search or queries the current setting.

Syntax
```
:SEARch:PPATtern:LOGic:{A0|A1|A2|A3|A4|
A5|A6|A7|B0|B1|B2|B3|B4|B5|B6|B7} {HIGH|
LOW|DONTcare}
```

Example
```
:SEARCH:PPATTERN:LOGIC:A0 DONTcare
:SEARCH:PPATTERN:LOGIC:A0?→:SEARCH:
PPATTERN:LOGIC:A0 DONTCARE
```

### :SEARch:SELect

Function Queries the storage number of the found position and the position that is stored in the specified storage number.

Syntax
```
:SEARch:SELect {<NRf>|MAXimum}
:SEARch:SELect?
<NRf>=1 to 1000
```

Example
```
:SEARCH:SELECT 1
:SEARCH:SELECT?→:SEARCH:
SELECT 2.5000000
```

Description If no positions are found, "NAN" is returned.

### :SEARch:SELect? MAXimum

Function Queries the maximum storage number of the found positions.

Syntax
```
:SEARch:SELect? {MAXimum}
```

Example
```
:SEARCH:SELECT? MAXIMUM→:SEARCH:
SELECT 100
```

Description If no positions are found, "NAN" is returned.

## :SEARch:SPATtern? (Serial Pattern)

Function Queries all settings related to the serial pattern search.

Syntax :SEARch:SPATtern?

Example :SEARCH:SPATTERN?→:SEARCH:SPATTERN:
CLOCK:SOURCE NONE;:SEARCH:SPATTERN:
INTERVAL 1.0E-06;SOURCE 1;LEVEL 0.0E+00;
PSELECT 1;PATTERN1 "HXXXXXXX";
PATTERN2 "XXXXXXXX";PATTERN3 "XXXXXXXX";
PATTERN4 "XXXXXXXX";PATTERN5 "XXXXXXXX";
PATTERN6 "XXXXXXXX";PATTERN7 "XXXXXXXX";
PATTERN8 "XXXXXXXX"

## :SEARch:SPATtern:BIT

Function Sets the bit of the logic waveform to be searched or queries the current setting.

Syntax :SEARch:SPATtern:BIT {<NRf>}
:SEARch:SPATtern:BIT?
<NRf>=Bit 0 to 7

Example :SEARCH:SPATTERN:BIT 0
:SEARCH:SPATTERN:BIT?→:SEARCH:SPATTERN:
BIT 0

## :SEARch:SPATtern:CLEar

Function Clears all search patterns (Don't care).

Syntax :SEARch:SPATtern:CLEar

Example :SEARCH:SPATTERN:CLEAR

## :SEARch:SPATtern:CLOCk?

Function Queries all settings related to the clock channel.

Syntax :SEARch:SPATtern:CLOCk?

Example :SEARCH:SPATTERN:CLOCK?→:SEARCH:
SPATTERN:CLOCK:SOURCE 1;LEVEL 0.0E+00;
SLOPE RISE;HYSTERESIS 0.3

## :SEARch:SPATtern:CLOCk:HYSTeresis

Function Sets the hysteresis of the clock channel or queries the current setting.

Syntax :SEARch:SPATtern:CLOCk:
HYSTeresis {<NRf>}
:SEARch:SPATtern:CLOCk:HYSTeresis?
<NRf>=0.3 to 4 div (0.1 steps)

Example :SEARCH:SPATTERN:CLOCK:HYSTERESIS 1
:SEARCH:SPATTERN:CLOCK:HYSTERESIS?→
:SEARCH:SPATTERN:CLOCK:HYSTERESIS 1.0

Description An error occurs if "SEAEch:SPATtern:CLOCk:SOURce" is set to "NONE."

## :SEARch:SPATtern:CLOCk:LEVEl

Function Sets the level of the clock channel or queries the current setting.

Syntax :SEARch:SPATtern:CLOCk:LEVEl {<voltage>}
:SEARch:SPATtern:CLOCk:LEVEl?
<voltage>= 8 div on the screen (0.01 div steps)

Example :SEARCH:SPATTERN:CLOCK:LEVEL -10mV
:SEARCH:SPATTERN:CLOCK:LEVEL?→:SEARCH:
SPATTERN:CLOCK:LEVEL -10.00E-03

Description An error occurs if "SEAEch:SPATtern:CLOCk:SOURce" is set to "NONE."

## :SEARch:SPATtern:CLOCk:SLOPe

Function Sets the slope of the clock channel or queries the current setting.

Syntax :SEARch:SPATtern:CLOCk:SLOPe {FALL|RISE}
:SEARch:SPATtern:CLOCk:SLOPe?

Example :SEARCH:SPATTERN:CLOCK:SLOPE RISE
:SEARCH:SPATTERN:CLOCK:SLOPE?→:SEARCH:
SPATTERN:CLOCK:SLOPE RISE

Description An error occurs if "SEAEch:SPATtern:CLOCk:SOURce" is set to "NONE."

## :SEARch:SPATtern:CLOCk:SOURce

Function Sets the clock channel source waveform or queries the current setting.

Syntax :SEARch:SPATtern:CLOCk:SOURce {<NRf>|
NONE}
:SEARch:SPATtern:CLOCk:SOURce?
<NRf>=1 to 4

Example :SEARCH:SPATTERN:CLOCK:SOURCE 1
:SEARCH:SPATTERN:CLOCK:SOURCE?→:SEARCH:
SPATTERN:SOURCE 1

## :SEARch:SPATtern:INTerval

Function Sets the search interval of the waveform to be searched or queries the current setting.

Syntax :SEARch:SPATtern:INTerval {<time>}
:SEARch:SPATtern:INTerval?
<time>=10 ps to 500 s

Example :SEARCH:SPATTERN:INTERVAL 100us
:SEARCH:SPATTERN:INTERVAL?→:SEARCH:
SPATTERN:INTERVAL 100.00E-06

## :SEARch:SPATtern:LEVel

Function Sets the upper and lower limits of the search level of the waveform to be searched or queries the current setting.

Syntax
```
:SEARch:SPATtern:LEVel {<voltage>,
<voltage>|<NRf>,NRf>}
:SEARch:SPATtern:LEVel?
<voltage>= 8 div on the screen (when the
           waveform is CH1 to CH4)
<NRf>=–4 to 4 div (0.01 steps when the
      waveform is  MATH1 or MATH2)
```

Example
```
:SEARCH:SPATTERN:LEVEL -1V,1V
:SEARCH:SPATTERN:LEVEL?→:SEARCH:
SPATTERN:LEVEL -1.000E+00,1.000E+00
```

## :SEARch:SPATtern:PATTern<x>

Function Sets the search pattern or queries the current setting.

Syntax
```
:SEARch:SPATtern:PATTern<x> <Character
string>
:SEARch:SPATtern:PATTern<x>?
<x>=1 to 8
<Character string>=8 characters of the
                   combination of
                   'H','L', and 'X' (The
                   left most character
                   represents the
                   smallest bit.  Total
                   of 8 bits.)
```

Example
```
:SEARCH:SPATTERN:PATTERN1 "HLXHLXHL"
:SEARCH:SPATTERN:PATTERN1?→:SEARCH:
SPATTERN:PATTERN1 "HLXHLXHL"
```

Description
- <x> represents the 8 bit block number. Thus, when <x>=1, the left most character represents bit 0 and the right most character represents bit 7. The relationship between <x> and the character string is as follows.
  <x>=1: <character string>=bits 0 to 7,
  <x>=2: <character string>=bits 8 to 15,
  <x>=3: <character string>=bits 16 to 23,
  <x>=4: <character string>=bits 24 to 31,
  <x>=5: <character string>=bits 32 to 39,
  <x>=6: <character string>=bits 40 to 47,
  <x>=7: <character string>=bits 48 to 55,
  <x>=8: <character string>=bits 56 to 63
- The pattern is assigned the pattern number that is selected with the ":SEARch:SPATtern:PSELect" command.

## :SEARch:SPATtern:PSELect (Pattern Select)

Function Selects the pattern number to which the search pattern is to be set or queries the current setting.

Syntax
```
:SEARch:SPATtern:PSELect {<NRf>}
:SEARch:SPATtern:PSELect?
<x>=1 to 4
```

Example
```
:SEARCH:SPATTERN:PSELECT 1
```

Description The values 1 to 4 of <x> correspond to pattern A to D.

## :SEARch:SPATtern:SOURce

Function Sets the waveform on which to perform the serial pattern search or queries the current setting.

Syntax
```
:SEARch:SPATtern:SOURce {<NRf>|MATH<x>|
PODA|PODB}
:SEARch:SPATtern:SOURce?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:SEARCH:SPATTERN:SOURCE 1
:SEARCH:SPATTERN:SOURCE?→:SEARCH:
SPATTERN:SOURCE 1
```

## :SEARch:SPOint (Start Point)

Function Sets the start position of the search or queries the current setting.

Syntax
```
:SEARch:SPOint {<NRf>}
:SEARch:SPOint?
<NRf>=–5 to 5 div (10 div/displayed
      record length steps)
```

Example
```
:SEARCH:SPOINT 3
:SEARCH:SPOINT?→:SEARCH:
SPOINT 3.0000000
```

## :SEARch:TWINdow (Target WINdow)

Function Sets the zoom window in which to display the sections that are found or queries the current setting.

Syntax
```
:SEARch:TWINdow {Z1|Z2}
:SEARch:TWINdow?
```

Example
```
:SEARCH:TWINDOW
Z1:SEARCH:TWINDOW?→:SEARCH:TWINDOW Z1
```

## :SEARch:TYPE

Function Sets the search type or queries the current setting.

Syntax
```
:SEARch:TYPE {SPATtern|EDGE|WIDTh|
PPATtern|ASCRoll|CANBus|SPIBus}
:SEARch:TYPE?
```

Example
```
:SEARCH:TYPE SPATTERN
:SEARCH:TYPE?→:SEARCH:TYPE SPATTERN
```

Description ":SEARch:TYPE CANBus" cannot be specified unless the CAN Bus  signal analysis function (option) is connected.

## :SEARch:WIDTh?

Function Queries all settings related to the

Syntax
```
:SEARch:WIDTh?
```

Example
```
:SEARCH:WIDTH?→:SEARCH:WIDTH:TYPE IN;
SOURCE 1;LEVEL 0.0E+00;POLARITY HIGH;
HYSTERESIS 0.30;TIME1 1.000000E-06;
TIME2 1.000000E-06
```

## :SEARch:WIDTh:HYSTeresis

Function Sets the hysteresis on the pulse width search level or queries the current setting.

Syntax
```
:SEARch:WIDTh:HYSTeresis {<NRf>}
:SEARch:WIDTh:HYSTeresis?
<NRf>=0.3 to 4.0div (0.1 steps)
```

Example
```
:SEARCH:WIDTH:HISTERESIS 0.3
:SEARCH:WIDTH:HISTERESIS?→:SEARCH:
WIDTH:HYSTERESIS 0.30
```

## :SEARch:WIDTh:LEVel

Function Sets the pulse width search level or queries the current setting.

Syntax
```
:SEARch:WIDTh:LEVel {<voltage>|<NRf>}
:SEARch:WIDTh:LEVel?
<voltage>=8 div on the screen (0.01div
          steps)
<NRf>=-4 to 4div(0.01div steps)
```

Example
```
:SEARCH:WIDTH:LEVEL 1V
:SEARCH:WIDTH:LEVEL?→:SEARCH:WIDTH:
LEVEL 1.000E+00
```

## :SEARch:WIDTh:POLarity

Function Sets the pulse width search level's polarity or queries the current setting.

Syntax
```
:SEARch:WIDTh:POLarity {HIGH|LOW}
:SEARch:WIDTh:POLarity?
```

Example
```
:SEARCH:WIDTH:POLARUTY HIGH
:SEARCH:WIDTH:POLARITY?→:SEARCH:WIDTH:
POLARITY HIGH
```

## :SEARch:WIDTh:SOURce

Function Sets the search source for the pulse width search or queries the current setting.

Syntax
```
:SEARch:WIDTh:SOURce {<NRf>|MATH<x>}
:SEARch:WIDTh:SOURce?
<NRf>=1 to 4
<x>=1 or 2
```

Example
```
:SEARCH:WIDTH:SOURCE 1
:SEARCH:WIDTH:SOURCE?→:SEARCH:WIDTH:
SOURCE 1
```

## :SEARch:WIDTh:TIME<x>

Function Sets the pulse width when using the pulse width search or queries the current setting.

Syntax
```
:SEARch:WIDTh:TIME<x> {<time>}
:SEARch:WIDTh:TIME<x>?
<x>=1 or 2
<time>= 10ps to 500s(when <x>=1 and
        "T1<Pulse<T2" is not true))
        10ps to 499.95s(when <x>=1 and
        "T1<Pulse<T2" )
        20ps to 500s(when <x>=2 and
        "T1<Pulse<T2" )
```

Example
```
:SEARCH:WIDTH:TIME1 20ps
:SEARCH:WIDTH:TIME1?→:SEARCH:WIDTH:
TIME1 20.0E-12
```

## :SEARch:WIDTh:TYPE

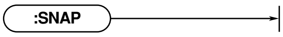
Function Sets the search type when using the pulse width search or queries the current setting.

Syntax
```
:SEARch:WIDTh:TYPE {IN|OUT|TIMeout|
BETWeen}
:SEARch:WIDTh:TYPE?
```

Example
```
:SEARCH:WIDTH:TYPE IN
:SEARCH:WIDTH:TYPE?→:SEARCH:WIDTH:
TYPE IN
```

## 5.22 SNAP Group

The SNAP command is used to execute a snapshot. The same function can be performed using the SNAPSHOT key on the front panel.

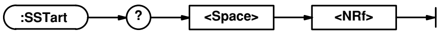


### :SNAP

Function Executes a snapshot.
Syntax `:SNAP`
Example `:SNAP`

## 5.23 SSTart Group

The commands in the SSTart group are used to execute the single start operation. It operates in the same fashion as the single trigger.



### :SSTart?

Function Starts the data acquisition with the trigger mode set to single. If the operation is stopped within the specified time period, 0 is returned. Otherwise, 1 is returned.

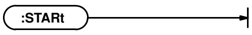
Syntax `:SSTart? {<NRf>}`
`<NRf>=1 to 360000 (10 ms resolution: wait period, START and wait)`
`0 (START only. No wait.)`
`-360000 to -1 (10 ms resolution: wait period, START and wait)`

Example `:SSTART? 100→:SSTART 0`

Description
- If the specified time period is positive, data acquisition is started in the SINGLE TRIGGER mode and waits for the operation to stop.
- If the specified time period is 0, data acquisition is started and 0 is returned without waiting for the operation to stop.
- If the specified time period is negative, the instrument simply waits for the operation to stop. Data acquisition is not started.

## 5.24 STARt Group

The STARt command is used to start acquisition. The same function can be performed by pressing the START/ STOP key on the front panel.

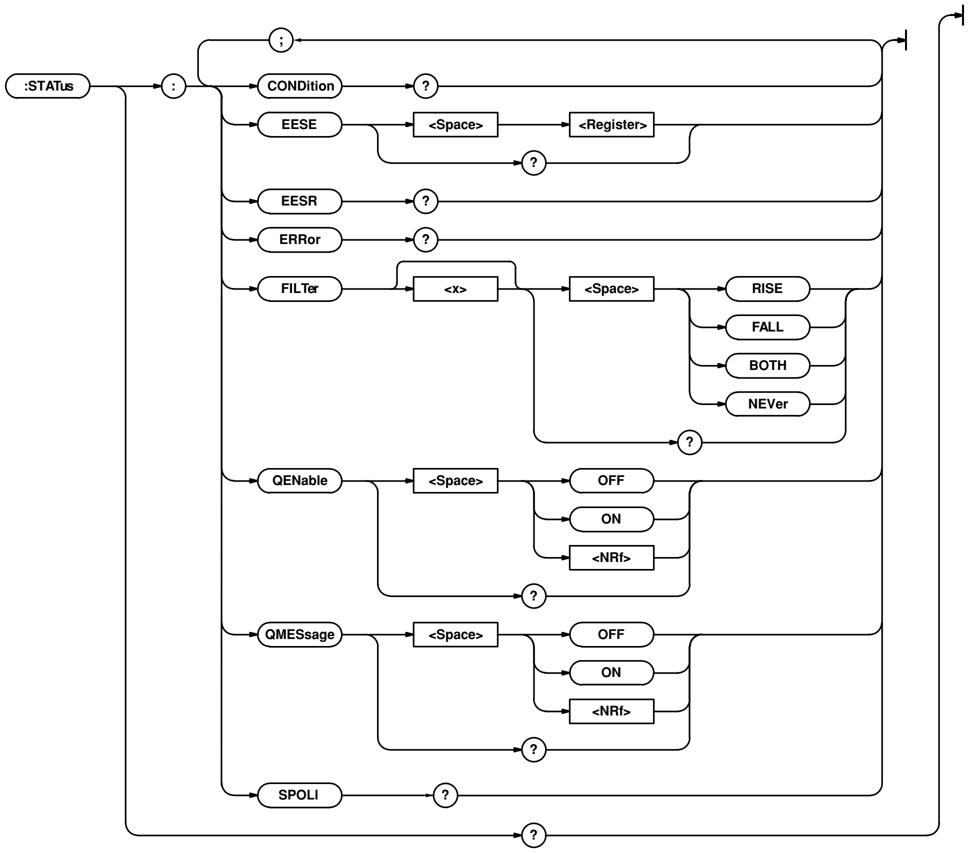


### :STARt

Function Starts acquisition.
Syntax `:STARt`
Example `:START`
Description To stop acquisition, use "`STOP.`"

# 5.25 STATus Group

The commands in the STATus group are used to make settings and queries about the status report. There is no front panel key for this function. For details of the status report, refer to Chapter 6.



## :STATus?

Function Inquires about all the settings relating to the communications status function.

Syntax `:STATus?`

Example `:STATUS?→:STATUS:EESE 0;FILTER1 NEVER;FILTER2 NEVER;FILTER3 NEVER;FILTER4 NEVER;FILTER5 NEVER;FILTER6 NEVER;FILTER7 NEVER;FILTER8 NEVER;FILTER9 NEVER;FILTER10 NEVER;FILTER11 NEVER;FILTER12 NEVER;FILTER13 NEVER;FILTER14 NEVER;FILTER15 NEVER;FILTER16 NEVER;QENABLE 1;QMESSAGE 1`

## :STATus:CONDition?

Function Queries the contents of the condition register, and clears the register.

Syntax `:STATus:CONDition?`

Example `:STATUS:CONDITION?→16`

Description For a description of the event register, refer to chapter 6.

## :STATus:EESE

Function Sets/queries the extended event enable register.

Syntax `:STATus:EESE <Register>`
`:STATus:EESE?`
`<Register>=0 to 65535`

Example `:STATUS:EESE #B00000000`
`:STATUS:EESE?→:STATUS:EESE 0`

Description For a description of the event register, refer to chapter 6.

## :STATus:EESR?

Function Queries the content of the extended event register, and clears the register.

Syntax `:STATus:EESR?`

Example `:STATUS:EESR?→1`

Description For a description of the event register, refer to chapter 6.

### :STATus:ERRor?

Function Queries the code and message (at the beginning of the error queue) of the error which has occurred.
Syntax `:STATus:ERRor?`
Example `:STATUS:ERROR?→113",Undefined header"`

### :STATus:FILTer<x>

Function Sets/queries the transit filter.
Syntax `:STATus:FILTer<x> {RISE|FALL|BOTH|NEVer}`
`:STATus:FILTer<x>?`
`<x>=1 to 16`
Example `:STATUS:FILTER2 RISE`
`:STATUS:FILTER2?→:STATUS:FILTER2 RISE`
Description For a description of the event register, refer to chapter 6.

### :STATus:QENable

Function Selects/queries whether messages other than errors are stored in the error queue.
Syntax `:STATus:QENable {<Boolean>}`
`:STATus:QENable?`
Example `:STATUS:QENABLE ON`
`:STATUS:QENABLE?→:STATUS:QENABLE 1`

### :STATus:QMESsage

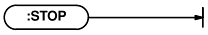
Function Selects/queries whether message content is appended to responses to "`STATus:ERRor?`".
Syntax `:STATus:QMESsage {<Boolean>}`
`:STATus:QMESsage?`
Example `:STATUS:QMESSAGE ON`
`:STATUS:QMESSAGE?→:STATUS:QMESSAGE 1`

### :STATus:SPOLl? (Serial Poll)

Function Executes the serial polling.
Syntax `:STATus:SPOLl?`
Example `:STATUS:SPOLL?→:STATUS:SPOLL 0`
Description This is an exclusive command for the serial (RS-232) interface.

## 5.26 STOP Group

The STOP command is used to stop acquisition. The same function can be performed using the START/STOP key on the front panel.

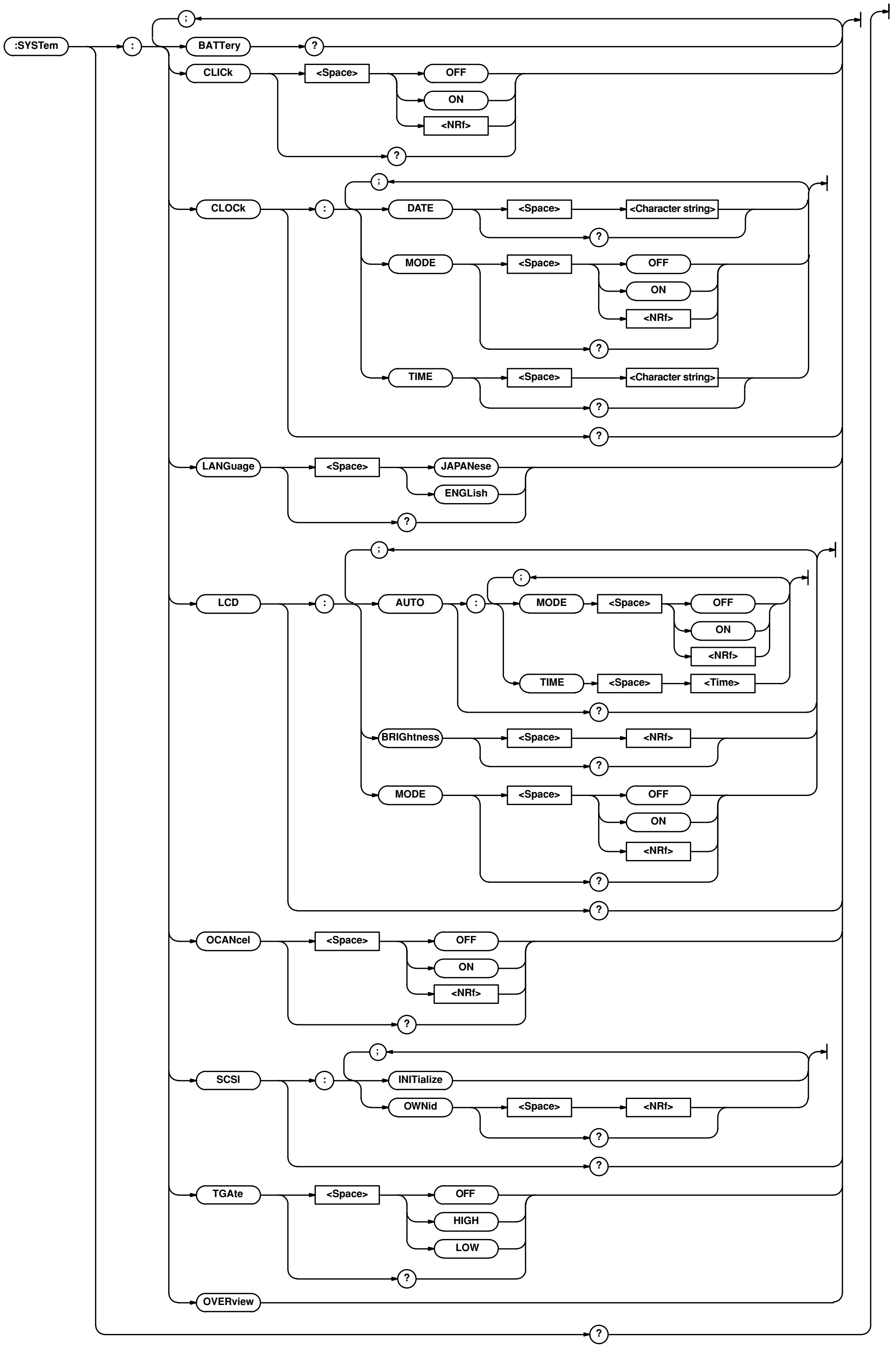


### :STOP

Function Stops acquisition.
Syntax `:STOP`
Example `:STOP`
Description To start acquisition, use "`STARt`."

# 5.27 SYSTem Group

The commands in the SYSTem group are used to make settings and queries about the system. The same settings can be made using the System Config menu obtained when the MISC key on the front panel is pressed.



### :SYSTem?

Function Queries all system settings.

```
Syntax      :SYSTem?
Example     :SYSTEM?→:SYSTEM:SCSI:OWNID 0;:SYSTEM:
            CLOCK:MODE 1;DATE "1999/06/19";
            TIME "17:47:17";:SYSTEM:
            LANGUAGE JAPANESE;CLICK 1;TGATE OFF;LCD:
            MODE 1;BRIGHTNESS 7;AUTO:MODE 0;TIME 1;:
            SYSTEM:OCANCEL 0
```

### :SYSTem:BATTery?

Function Queries the condition of the internal lithium battery.

```
Syntax      :SYSTem:BATTery?
Example     :SYSTEM:BATTERY?→:SYSTEM:BATTERY 1
```

Description "1" is returned if the battery is functioning, and "0" is returned if the battery has run out.

### :SYSTem:CLICk

Function Sets the click sound ON or OFF, or queries the current setting.

```
Syntax      :SYSTem:CLICk {<Boolean>}
            :SYSTem:CLICk?
Example     :SYSTEM:CLICK ON
            :SYSTEM:CLICK?→:SYSTEM:CLICK 1
```

### :SYSTem:CLOCk?

Function Queries all settings related to date and time.

```
Syntax      :SYSTem:CLOCk?
Example     :SYSTEM:CLOCK?→:SYSTEM:CLOCK:MODE 1;
            DATE "1999/06/19";TIME "17:47:17"
```

### :SYSTem:CLOCk:DATE

Function Sets/queries the date.

```
Syntax      :SYSTem:CLOCK:DATE <Character string>
            :SYSTem:CLOCK:DATE?
            <Character string>=YY/MM/DD,refer to
                               User's Manual
                               IM701410-01E
Example     :SYSTEM:CLOCK:DATE "1999/09/30"
            :SYSTEM:CLOCK:DATE?→:SYSTEM:CLOCK:
            DATE "1999/09/30"
```

### :SYSTem:CLOCk:MODE

Function Turns the date and time display ON/OFF or queries the current setting.

```
Syntax      :SYSTem:CLOCk:MODE {<Boolean>}
            :SYSTem:CLOCk:MODE?
Example     :SYSTEM:CLOCK:MODE ON
            :SYSTEM:CLOCK:MODE?→:SYSTEM:CLOCK:
            MODE 1
```

### :SYSTem:CLOCk:TIME

Function Sets the time or queries the current setting.

```
Syntax      :SYSTem:CLOCk:TIME <Character string>
            :SYSTem:CLOCk:TIME?
            <Character string>=HH:MM:SS, See the
                               IM701410-01E User's
                               Manual.
Example     :SYSTEM:CLOCK:TIME "14:30:00"
            :SYSTEM:CLOCK:TIME?→:SYSTEM:CLOCK:
            TIME "14:30:00"
```

### :SYSTem:LANGuage

Function Sets/queries the message language.

```
Syntax      :SYSTem:LANGuage {ENGLish|JAPANese}
            :SYSTem:LANGuage?
Example     :SYSTEM:LANGUAGE JAPANESE
            :SYSTEM:LANGUAGE?→:SYSTEM:
            LANGUAGE JAPANESE
```

### :SYSTem:LCD?

Function Queries all settings related to the LCD monitor.

```
Syntax      :SYSTem:LCD?
Example     :SYSTEM:LCD?→:SYSTEM:LCD:MODE 1;
            BRIGHTNESS 1;AUTO:MODE 0;TIME 1
```

### :SYSTem:LCD:AUTO?

Function Queries all settings related to the LCD backlight auto-off function.

```
Syntax      :SYSTem:LCD:AUTO?
Example     :SYSTEM:LCD:AUTO?→:SYSTEM:LCD:AUTO:
            MODE 0;TIME 1
```

### :SYSTem:LCD:AUTO:MODE

Function Turns the LCD backlight ON/OFF or queries the current setting.

```
Syntax      :SYSTem:LCD:AUTO:MODE {<Boolean>}
            :SYSTem:LCD:AUTO:MODE?
Example     :SYSTEM:LCD:AUTO:MODE ON
            :SYSTEM:LCD:AUTO:MODE?→:SYSTEM:LCD:
            AUTO:MODE 1
```

### :SYSTem:LCD:AUTO:TIME

Function Sets the time for the LCD backlight to automatically turn OFF or queries the current setting.

```
Syntax      :SYSTem:LCD:AUTO:TIME {<NRf>}
            :SYSTem:LCD:AUTO:TIME?
            <NRf>=1 to 60 (s)
Example     :SYSTEM:LCD:AUTO:TIME 1
            :SYSTEM:LCD:AUTO:TIME?→:SYSTEM:LCD:
            AUTO:TIME 1
```

## :SYSTem:LCD:BRIGhtness

Function Sets/queries the brightness of the screen.

Syntax
```
:SYSTem:LBRightness{<NRf>}
:SYSTEM:LBRightness?
<NRf>=0 to 7
```

Example
```
:SYSTEM:LCD:BRIGTHNESS 2
:SYSTEM:LCD:BRIGTHNESS?→:SYSTEM:
LCD:BRIGTHNESS 2
```

## :SYSTem:LCD:MODE

Function Turns the LCD backlight ON/OFF or queries the current setting.

Syntax
```
:SYSTem:LCD:MODE {<Boolean>}
:SYSTem:LCD:MODE?
```

Example
```
:SYSTEM:LCD:MODE OFF
:SYSTEM:LCD:MODE?→:SYSTEM:LCD:MODE 0
```

## :SYSTem:OCANcel (Offset Cancel)

Function Sets whether or not to reflect the offset voltage to the measured results or the computed results (ON/OFF) or queries the current setting.

Syntax
```
:SYSTem:OCANcel {<Boolean>}
:SYSTem:OCANcel?
```

Example
```
:SYSTEM:OCANCEL ON
:SYSTEM:OCANCEL?→:SYSTEM:OCANCEL 0
```

## :SYSTem:OVERview

Function Displays the system information.

Syntax `:SYSTem:OVERview`

Example `:SYSTEM:OVERVIEW`

## :SYSTem:SCSI?

Function Queries all SCSI-ID settings.

Syntax `:SYSTem:SCSI?`

Example `:SYSTEM:SCSI?→:SYSTEM:SCSI:OWNID 0`

## :SYSTem:SCSI:INITialize

Function Initializes the SCSI.

Syntax `:SYSTem:SCSI:INITialize`

Example `:SYSTEM:SCSI:INITIALIZE`

Description This command should always be issued following change of the oscilloscope's SCSI-ID (by the "`:SYSTem:SCSI:OWNid`" command).

## :SYSTem:SCSI:OWNid

Function Sets/queries own SCSI-ID.

Syntax
```
:SYSTem:SCSI:OWNid {<NRf>}
:SYSTem:SCSI:OWNid?
<NRf>=0 to 7
```

Example
```
:SYSTEM:SCSI:OWNID 7
:SYSTEM:SCSI:OWNID?→:SYSTEM:SCSI:
OWNID 7
```

## :SYSTem:TGATe (Trigger Gate)

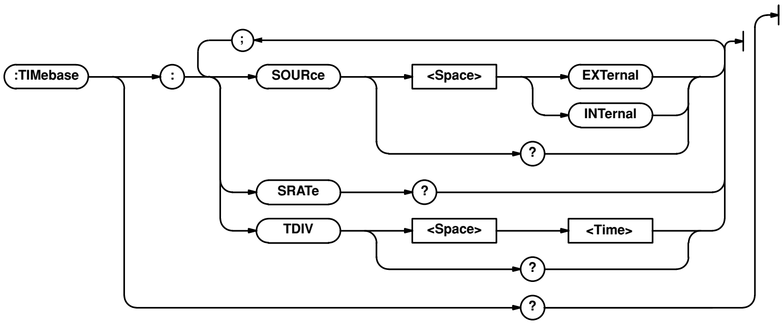
Function Sets the external trigger gate or queries the current setting.

Syntax
```
:SYSTem:TGAte {<OFF|HIGH|LOW>}
:SYSTem:TGAte?
```

Example
```
:SYSTEM:TGATE OFF
:SYSTEM:TGATE?→:SYSTEM:TGATE 0
```

## 5.28 TIMebase Group

The commands in the TIMebase group are used to make settings and queries about the time base. These settings and inquiries can also be made using the TIME/DIV knob on the front panel.



### :TIMebase?

Function Queries all the time base settings.

Syntax :TIMebase?

Example :TIMEBASE?→:TIMEBASE:SOURCE INTERNAL; TDIV 500.0E-06

### :TIMebase:SOURce

Function Sets the time base or queries the current setting.

Syntax :TIMebase:SOURce {EXTernal|INTernal}
:TIMebase:SOURce?

Example :TIMEBASE:SOURCE INTERNAL
:TIMEBASE:SOURCE?→:TIMEBASE: SOURCE INTERNAL

### :TIMebase:SRATe? (Sample RATE)

Function Queries the sampling rate.

Syntax :TIMebase:SRATe?

Example :TIMEBASE:SRATE?→:TIMEBASE: SRATE 1.0E+09

### :TIMebase:TDIV

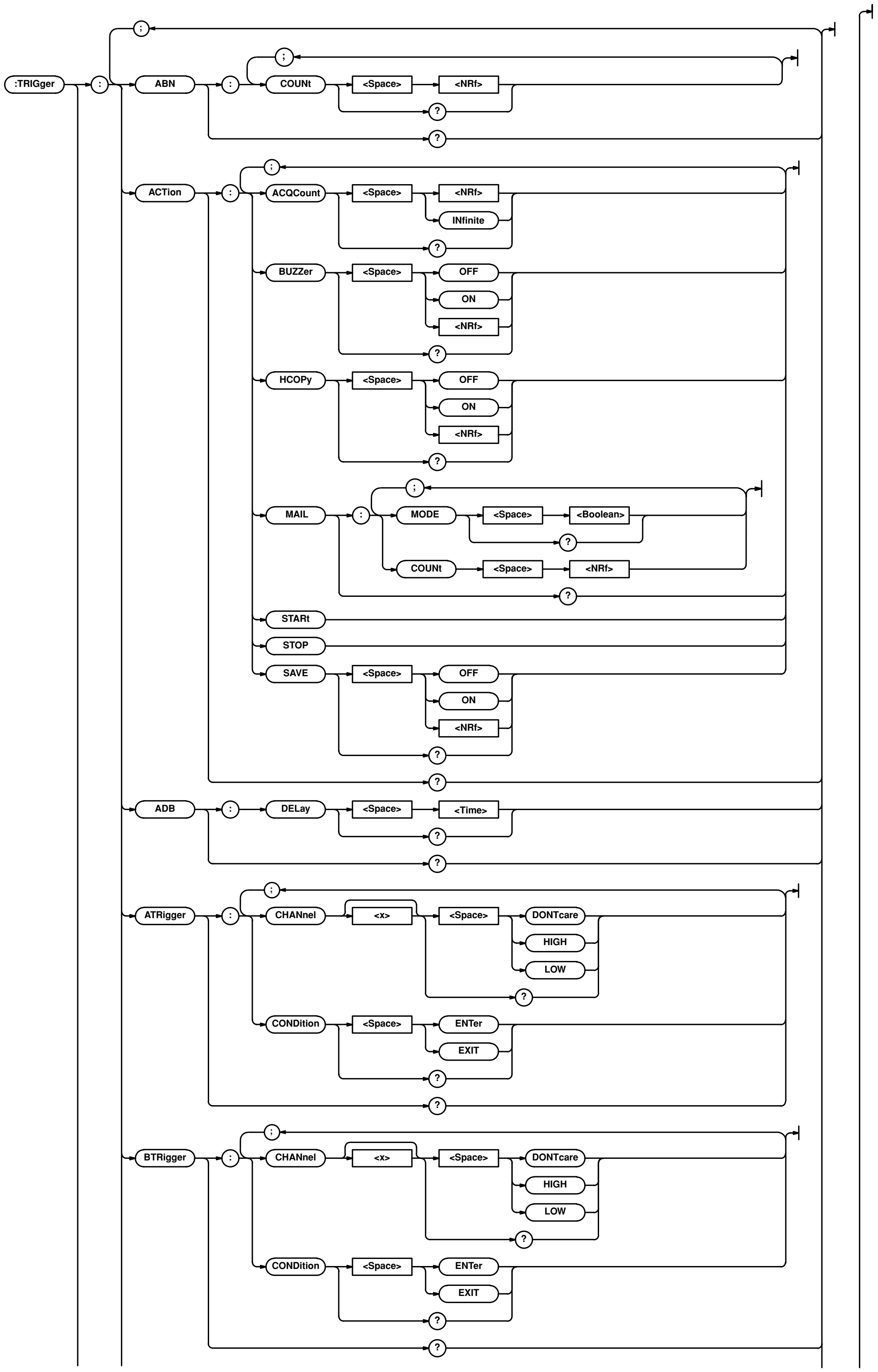
Function Sets/queries the T/div setting.

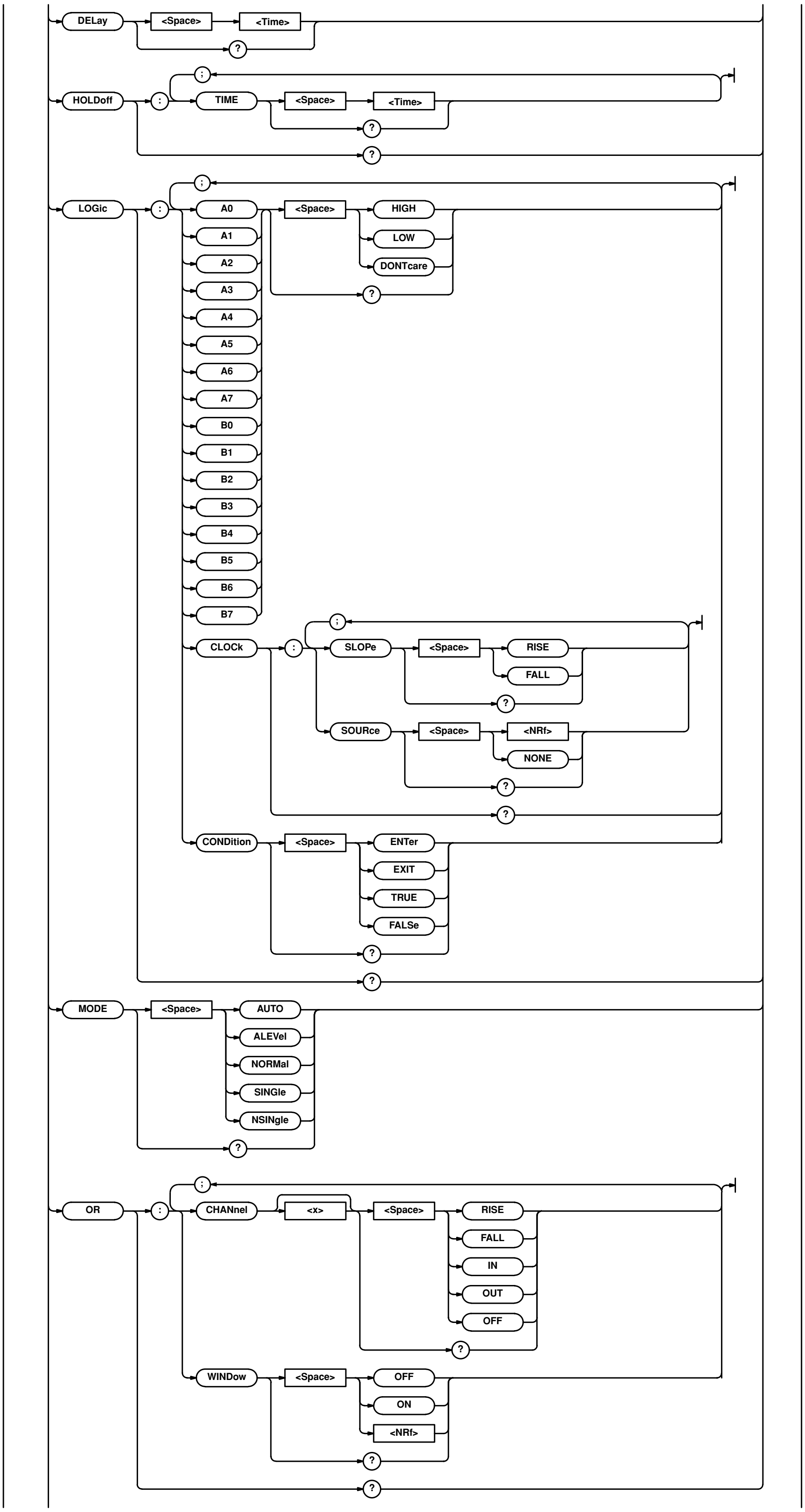
Syntax :TIMebase:TDIV {<Time>}
:TIMebase:TDIV?
<Time>=1 ns to 50 s

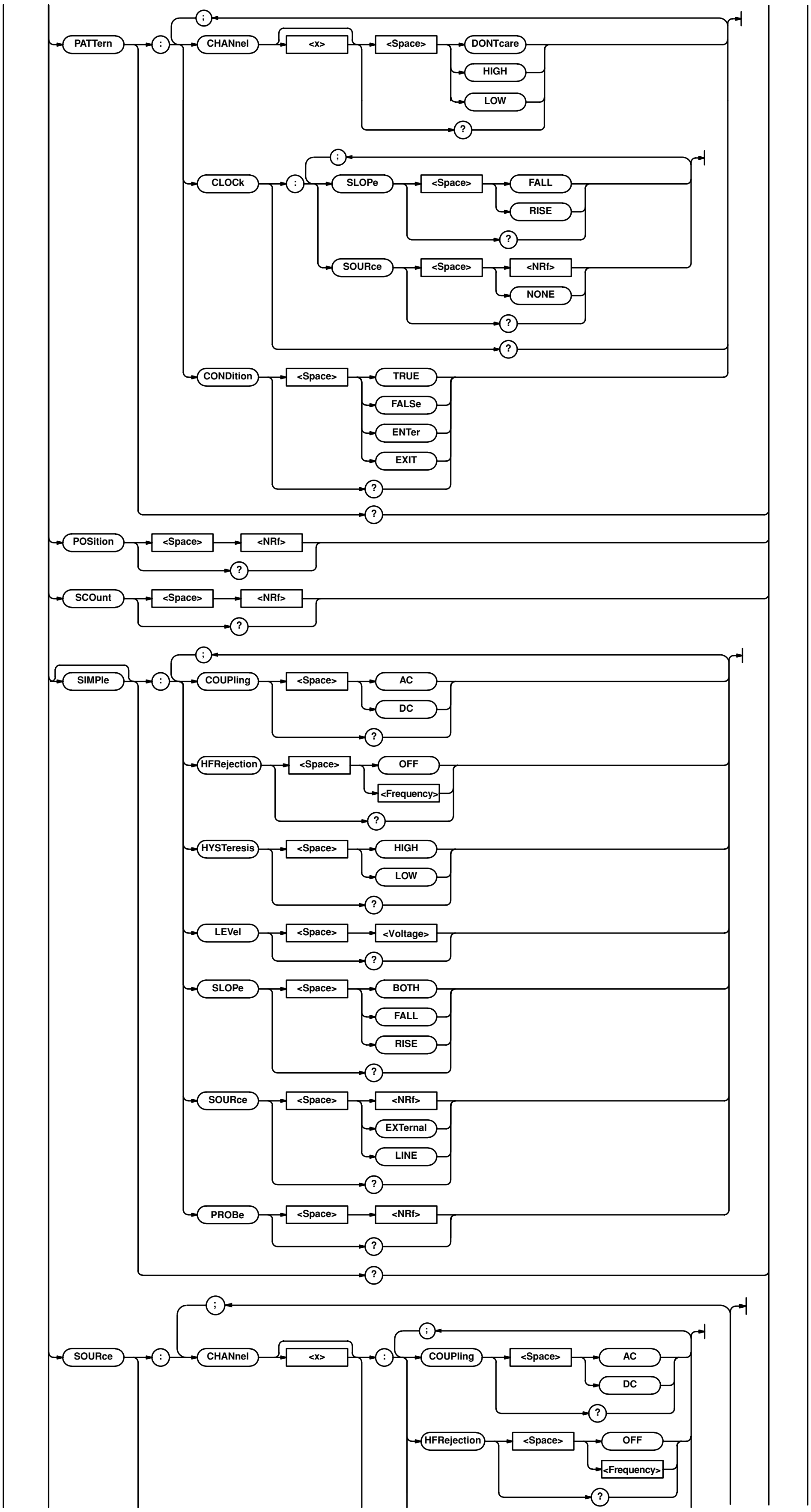
Example :TIMEBASE:TDIV 1MS
:TIMEBASE:TDIV?→:TIMEBASE: TDIV 1.000E-03

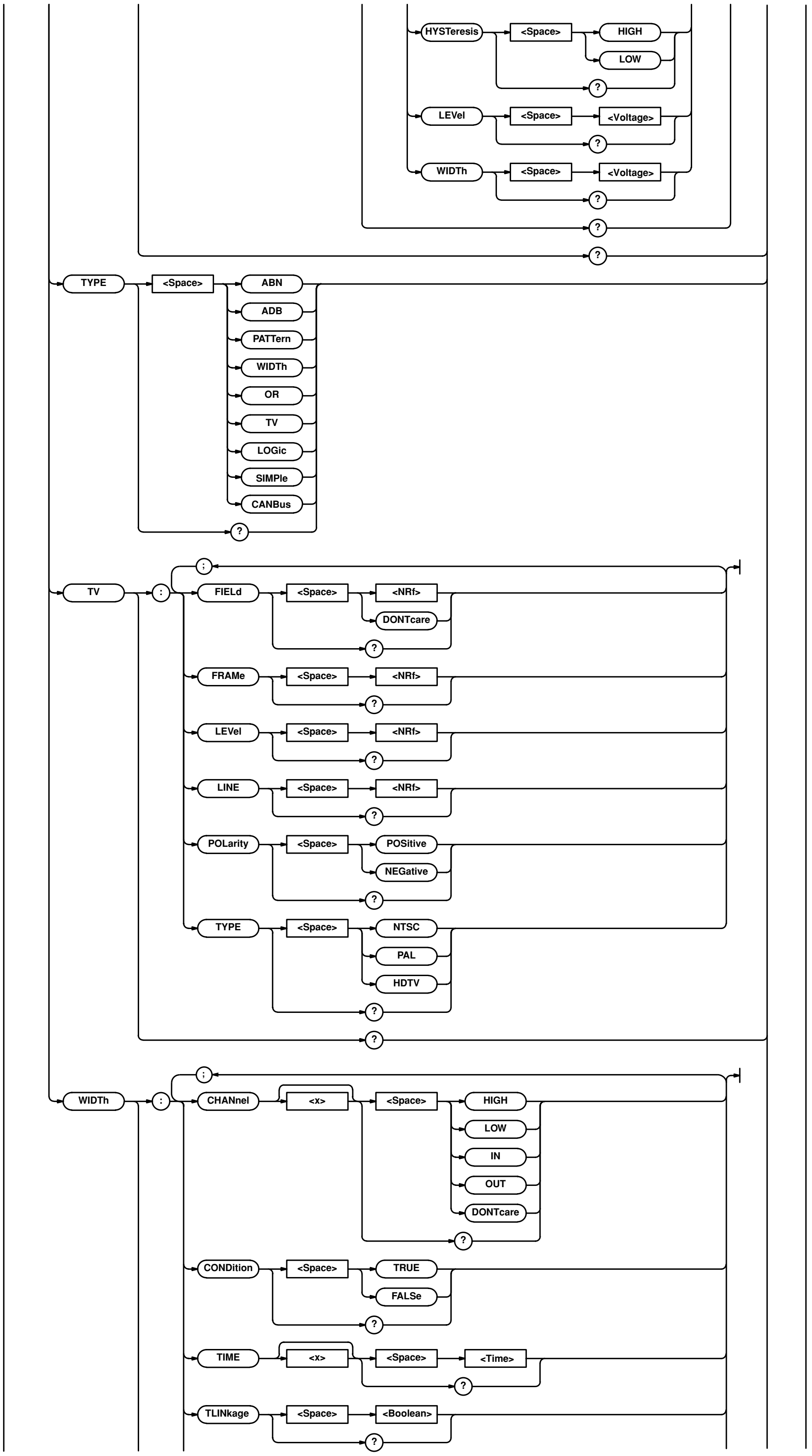
## 5.29 TRIGger Group

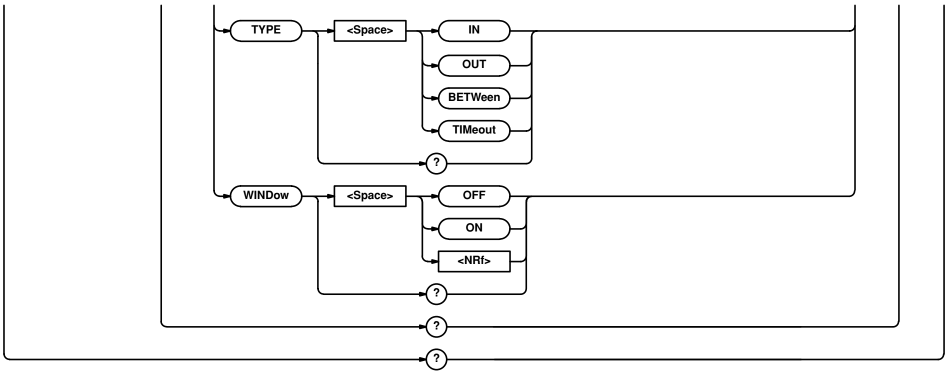
The commands in the TRIGger group are used to make settings and queries about the trigger. The same settings can be made using the TRIGGER group keys (the SIMPLE and ENHANCED keys, and LEVEL nob, and POSITION/DELAY keys).




DELay
<Space>
<Time>
?
HOLDoff
:
;
TIME
<Space>
<Time>
?
?
LOGic
:
;
A0
A1
A2
A3
A4
A5
A6
A7
B0
B1
B2
B3
B4
B5
B6
B7
<Space>
HIGH
LOW
DONTcare
?
CLOCk
:
;
SLOPe
<Space>
RISE
FALL
?
SOURce
<Space>
<NRf>
NONE
?
?
CONDition
<Space>
ENTer
EXIT
TRUE
FALSe
?
?
MODE
<Space>
AUTO
ALEVel
NORMal
SINGle
NSINgle
?
OR
:
;
CHANnel
<x>
<Space>
RISE
FALL
IN
OUT
OFF
?
WINDow
<Space>
OFF
ON
<NRf>
?
?


PATTern
:
;
CHANnel
<x>
<Space>
DONTcare
HIGH
LOW
?
CLOCk
:
;
SLOPe
<Space>
FALL
RISE
?
SOURce
<Space>
<NRf>
NONE
?
?
CONDition
<Space>
TRUE
FALSe
ENTer
EXIT
?
?
POSition
<Space>
<NRf>
?
SCOunt
<Space>
<NRf>
?
SIMPle
:
;
COUPling
<Space>
AC
DC
?
HFRejection
<Space>
OFF
<Frequency>
?
HYSTeresis
<Space>
HIGH
LOW
?
LEVel
<Space>
<Voltage>
?
SLOPe
<Space>
BOTH
FALL
RISE
?
SOURce
<Space>
<NRf>
EXTernal
LINE
?
PROBe
<Space>
<NRf>
?
?
SOURce
:
;
CHANnel
<x>
:
;
COUPling
<Space>
AC
DC
?
HFRejection
<Space>
OFF
<Frequency>
?


HYSTeresis
<Space>
HIGH
LOW
?
LEVel
<Space>
<Voltage>
?
WIDTh
<Space>
<Voltage>
?
?
?
TYPE
<Space>
ABN
ADB
PATTern
WIDTh
OR
TV
LOGic
SIMPle
CANBus
?
TV
:
;
FIELd
<Space>
<NRf>
DONTcare
?
FRAMe
<Space>
<NRf>
?
LEVel
<Space>
<NRf>
?
LINE
<Space>
<NRf>
?
POLarity
<Space>
POSitive
NEGative
?
TYPE
<Space>
NTSC
PAL
HDTV
?
?
WIDTh
:
;
CHANnel
<x>
<Space>
HIGH
LOW
IN
OUT
DONTcare
?
CONDition
<Space>
TRUE
FALSe
?
TIME
<x>
<Space>
<Time>
?
TLINkage
<Space>
<Boolean>
?



## :TRIGger?

Function Queries all trigger settings.

Syntax :TRIGger?

Example :TRIGGER?→:TRIGGER:ACTION:BUZZER 0;SAVE 0;HCOPY 0;ACQCOUNT INFINITE;:TRIGGER:SIMPLE:SOURCE 1;LEVEL 0.0E+00;SLOPE RISE;COUPLING DC;HFREJECTION OFF;HYSTERESIS LOW;PROBE 1;:TRIGGER:SOURCE:CHANNEL1:LEVEL 0.0E+00;COUPLING DC;HFREJECTION OFF;HYSTERESIS LOW;WIDTH 500.000E-03;:TRIGGER:SOURCE:CHANNEL2:LEVEL 0.0E+00;COUPLING DC;HFREJECTION OFF;HYSTERESIS LOW;WIDTH 50.000E+00;:TRIGGER:SOURCE:CHANNEL3:LEVEL 0.0E+00;COUPLING DC;HFREJECTION OFF;HYSTERESIS LOW;WIDTH 1.000E+00;:TRIGGER:SOURCE:CHANNEL4:LEVEL 0.0E+00;COUPLING DC;HFREJECTION OFF;HYSTERESIS LOW;WIDTH 50.000E+00;:TRIGGER:ABN:COUNT 1;:TRIGGER:HOLDOFF:TIME 80.000E-09;:TRIGGER:ADB:DELAY 1.000E-09;:TRIGGER:DELAY 0.0E+00;PATTERN:CLOCK:SOURCE 1;SLOPE RISE;:TRIGGER:PATTERN:CHANNEL1 HIGH;CHANNEL2 DONTCARE;CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;CONDITION TRUE;:TRIGGER:POSITION 50.000;MODE AUTO;WIDTH:TYPE IN;CHANNEL1 HIGH;CHANNEL2 HIGH;CHANNEL3 HIGH;CHANNEL4 HIGH;CONDITION TRUE;TLINKAGE 0 TIME1 1.000E-09;TIME2 1.000E-09;WINDOW 0;:TRIGGER:OR:WINDOW 0;CHANNEL1 RISE;CHANNEL2 OFF;CHANNEL3 OFF;CHANNEL4 OFF;:TRIGGER:SCOUNT 2;TV:TYPE NTSC;POLARITY NEGATIVE;LEVEL 5000000.0;FIELD 1;LINE 5;FRAME 1;:TRIGGER:TYPE SIMPLE;LOGIC:CLOCK:SOURCE 1;SLOPE RISE;:TRIGGER:LOGIC:CONDITION TRUE;A0 HIGH;A1 DONTCARE;A2 DONTCARE;A3 DONTCARE;A4 DONTCARE;A5 DONTCARE;A6 DONTCARE;A7 DONTCARE;B0 DONTCARE;B1 DONTCARE;B2 DONTCARE;B3 DONTCARE;B4 DONTCARE;B5 DONTCARE;B6 DONTCARE;B7 DONTCARE;:TRIGGER:ATRIGGER:CHANNEL1 HIGH;CHANNEL2 DONTCARE;CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;CONDITION ENTER;:TRIGGER:BTRIGGER:CHANNEL1 DONTCARE;CHANNEL2 HIGH;CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;CONDITION ENTER

## :TRIGger:ABN? (A→B (n))

Function Queries all A→B (n) trigger settings.

Syntax :TRIGger:ABN?

Example :TRIGGER:ABN?→:TRIGGER:ABN:COUNT 1

## :TRIGger:ABN:COUNt

Function Sets/queries the number of times the pattern for the A→B (n) trigger should become true.

Syntax :TRIGger:ABN:COUNt {<NRf>}
:TRIGger:ABN:COUNt?
<NRf>=1 to 100000000

Example :TRIGGER:ABN:COUNT 1
:TRIGGER:ABN:COUNT?→:TRIGGER:ABN:COUNT 1

## :TRIGger:ACTion?

Function Queries all settings relating to the action on trigger.

Syntax :TRIGger:ACTion?

Example :TRIGGER:ACTION?→:TRIGGER:ACTION:BUZZER 0;SAVE 0;HCOPY 0:ACQCOUNT INFINITE

## :TRIGger:ACTion:ACQCount

Function Sets the number of times to perform the action for the action-on-trigger or queries the current setting.

Syntax :TRIGger:ACTion:ACQCount {<NRf|INFinite>}
:TRIGger:ACTion:ACQCount?
<NRf>=1 to 65536

Example :TRIGGER:ACTION:ACQCOUNT 10
:TRIGGER:ACTION:ACQCOUNT?→:TRIGGER:ACTION:ACQCOUNT 10

### :TRIGger:ACTion:BUZZer

Function Sets/queries the ON/OFF condition of the buzzer at the time of the trigger action.

Syntax
```
:TRIGger:ACTion:BUZZer {<Boolean>}
:TRIGger:ACTion:BUZZer?
```
Example
```
:TRIGGER:ACTION:BUZZER ON
:TRIGGER:ACTION:BUZZER?→:TRIGGER:
ACTION:BUZZER 1
```

### :TRIGger:ACTion:HCOPy

Function Sets/queries whether the screen image data is output at the time of the trigger action.

Syntax
```
:TRIGger:ACTion:HCOPy {<Boolean>}
:TRIGger:ACTion:HCOPy?
```
Example
```
:TRIGGER:ACTION:HCOPY ON
:TRIGGER:ACTION:HCOPY?→:TRIGGER:ACTION:
HCOPY 1
```

### :TRIGger:ACTion:MAIL?

Function Queries all the setting values for mail reports sent as an ACTion.

Syntax
```
:TRIGger:ACTion:MAIL?
```
Example
```
:TRIGGER:ACTION:MAIL:MODE?→:TRIGGER:
ACTION:MAIL:MODE 0;COUNT 100
```
Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

### :TRIGger:ACTion:MAIL:COUNt

Function Sets/queries the limit for the number of reports that are sent as an ACTion.

Syntax
```
:TRIGger:ACTion:MAIL:COUNt {<NRf>}
:TRIGger:ACTion:MAIL:COUNt?
 <NRf>=1 to 1000
```
Example
```
:TRIGGER:ACTION:MAIL:COUNT 100
:TRIGGER:ACTION:MAIL:COUNT?→:TRIGGER:
ACTION:MAIL:COUNT 100
```
Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

### :TRIGger:ACTion:MAIL:MODE

Function Sets/queries whether mail reports are sent as an ACTion.

Syntax
```
:TRIGger:ACTion:MAIL:MODE {<Boolean>}
```
Example
```
:TRIGGER:ACTION:MAIL:MODE OFF
:TRIGGER:ACTION:MAIL:MODE?→:TRIGGER:
ACTION:MAIL:MODE 0
```
Description This function is supported by the DL7200 and the DL7100 firmware version 2.01 or later.

### :TRIGger:ACTion:SAVE

Function Sets/queries whether the waveform data is saved to the medium at the time of the trigger action.

Syntax
```
:TRIGger:ACTion:SAVE {<Boolean>}
:TRIGger:ACTion:SAVE?
```
Example
```
:TRIGGER:ACTION:SAVE ON
:TRIGGER:ACTION:SAVE?→:TRIGGER:ACTION:
SAVE 1
```

### :TRIGger:ACTion:STARt

Function Starts action-on-trigger.

Syntax
```
:TRIGger:ACTion:STARt
```
Example
```
:TRIGGER:ACTION:START
```
Description Use `TRIGger:ACTION:STOP` to abort the trigger action.

### :TRIGger:ACTion:STOP

Function Aborts action-on-trigger.

Syntax
```
:TRIGger:ACTion:STOP
```
Example
```
:TRIGGER:ACTION:STOP
```
Description This command only aborts the trigger action. It does not cancel the settings.
The `TRIGger:ACTION:STARt` command can be used to resume the trigger action.

### :TRIGger:ADB? (A Delay B)

Function Queries all A Delay B trigger settings.

Syntax
```
:TRIGger:ADB?
```
Example
```
:TRIGGER:ADB?→:TRIGGER:ADB:
DELAY 0.000E+00
```

### :TRIGger:ADB:DELay

Function Sets/queries the delay time for pattern B for an A Delay B trigger.

Syntax
```
:TRIGger:ADB:DELay {<Time>}
:TRIGger:ADB:DELay?
<Time>=3 ns to 5 s
```
Example
```
:TRIGGER:ADB:DELAY 1NS
:TRIGGER:ADB:DELAY?→:TRIGGER:ADB:
DELAY 1.0E-09
```

### :TRIGger:ATRigger?

Function Queries all the pattern A settings.

Syntax
```
:TRIGger:ATRigger?
```
Example
```
:TRIGGER:ATRIGGER?→:TRIGGER:ATRIGGER:
CHANNEL1 HIGH;CHANNEL2 DONTCARE;
CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;
CONDITION ENTER
```

## :TRIGger:ATRigger:CHANnel<x>

Function Sets/queries the channel condition for pattern A.

Syntax
```
:TRIGger:ATRigger:CHANnel<x> {DONTcare|
HIGH|LOW}
:TRIGger:ATRigger:CHANnel<x>?
<x>=1 to 4
```

Example
```
(An example for CH1 is given below.)
:TRIGGER:ATRIGGER:CHANNEL1 HIGH
:TRIGGER:ATRIGGER:CHANNEL1?→:TRIGGER:
ATRIGGER:CHANNEL1 HIGH
```

## :TRIGger:ATRigger:CONDition

Function Sets/queries the conditions under which pattern A should become true.

Syntax
```
:TRIGger:ATRigger:CONDition {ENTer|EXIT}
:TRIGger:ATRigger:CONDition?
```

Example
```
:TRIGGER:ATRIGGER:CONDITION ENTER
:TRIGGER:ATRIGGER:CONDITION?→:TRIGGER:
ATRIGGER:CONDITION ENTER
```

Description This command is valid when the trigger type is "ABN" or "ADB."

## :TRIGger:BTRigger?

Function Queries all pattern B settings.

Syntax
```
:TRIGger:BTRigger?
```

Example
```
:TRIGGER:BTRIGGER?→:TRIGGER:BTRIGGER:
CHANNEL1 DONTCARE;CHANNEL2 HIGH;
CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;
CONDITION ENTER
```

## :TRIGger:BTRigger:CHANnel<x>

Function Sets/queries the channel condition for pattern B.

Syntax
```
:TRIGger:BTRigger:CHANnel<x> {DONTcare|
HIGH|LOW}
:TRIGger:BTRigger:CHANnel<x>?
<x>=1 to 4
```

Example
```
(An example for CH1 is given below.)
:TRIGGER:BTRIGGER:CHANNEL1 HIGH
:TRIGGER:BTRIGGER:CHANNEL1?→:TRIGGER:
BTRIGGER:CHANNEL1 HIGH
```

## :TRIGger:BTRigger:CONDition

Function Sets/queries the conditions under which pattern B becomes true.

Syntax
```
:TRIGger:BTRigger:CONDition {ENTer|EXIT}
:TRIGger:BTRigger:CONDition?
```

Example
```
:TRIGGER:BTRIGGER:CONDITION ENTER
:TRIGGER:BTRIGGER:CONDITION?→:TRIGGER:
BTRIGGER:CONDITION ENTER
```

Description This command is valid when the trigger type is "ABN" or "ADB."

## :TRIGger:DELay

Function Sets/queries the trigger delay (time between trigger point and trigger position).

Syntax
```
:TRIGger:DELay {<Time>}
:TRIGger:DELay?
<Time>=0 to 4 s (1/sampling rate steps)
```

Example
```
:TRIGGER:DELLAY 2US
:TRIGGER:DELAY?→:TRIGGER:DELAY 2.00E-06
```

Description If timebase is external clock, the value is fixed at 0.
On this oscilloscope, the delay is the time difference between the trigger point and the trigger position. You can set/query the trigger position using the ":TRIGger:POSition" command.

## :TRIGger:HOLDoff?

Function Queries all settings related to the hold off.

Syntax
```
:TRIGger:HOLDoff?
```

Example
```
:TRIGGER:HOLDOFF?→:TRIGGER:HOLDOFF:
TIME 80.0E-09
```

## :TRIGger:HOLDoff:TIME

Function Sets the hold off time or queries the current setting.

Syntax
```
:TRIGger:HOLDoff:TIME {<time>}
:TRIGger:HOLDoff:TIME?
<time>=80 ns to 10 s
```

Example
```
:TRIGGER:HOLDOFF:TIME 80NS
:TRIGGER:HOLDOFF:TIME?→:TRIGGER:
HOLDOFF:TIME 80.0E-09
```

## :TRIGger:LOGic?

Function Queries all settings related to logic trigger edge.

Syntax
```
:TRIGger:LOGic?
```

Example
```
:TRIGGER:LOGIC?→:TRIGGER:LOGIC:CLOCK:
SOURCE NONE;:TRIGGER:LOGIC:
CONDITION TRUE;:TRIGGER:LOGIC:A0 HIGH;
A1 DONTCARE;A2 DONTCARE;A3 DONTCARE;
A4 DONTCARE;A5 DONTCARE;A6 DONTCARE;
A7 DONTCARE;B0 DONTCARE;B1 DONTCARE;
B2 DONTCARE;B3 DONTCARE;B4 DONTCARE;
B5 DONTCARE;B6 DONTCARE;B7 DONTCARE
```

Description Query is not possible if the logic probe is not connected.

## :TRIGger:LOGic:{A0|A1|A2|A3|A4|A5|A6|A7|B0|B1|B2|B3|B4|B5|B6|B7}

Function Sets the condition of each bit or queries the current setting.

Syntax
```
:TRIGger:LOGic:{A0|A1|A2|A3|A4|A5|A6|A7|
B0|B1|B2|B3|B4|B5|B6|B7}
{DONTcare|HIGH|LOW}
:TRIGger:LOGic:{A0|A1|A2|A3|A4|A5|A6|A7|
B0|B1|B2|B3|B4|B5|B6|B7}?
```

Example
```
:TRIGGER:LOGIC:A0 HIGH
:TRIGGER:LOGIC:A0?→:TRIGGER:LOGIC:
A0 HIGH
```

Description Query/Setting is not possible if the logic probe is not connected.

## :TRIGger:LOGic:CLOCk?

Function Queries all settings related to the clock channel when using the logic trigger.

Syntax
```
:TRIGger:LOGic;CLOCk?
```

Example
```
:TRIGGER:LOGIC:CLOCK?→:TRIGGER:LOGIC:
CLOCK:SOURCE 1;SLOPE RISE
```

Description Query is not possible if the logic probe is not connected.

## :TRIGger:LOGic:CLOCk:SLOPe

Function Sets the slope of the clock channel when using the logic trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic;CLOCk:SLOPe {RISE|FALL}
:TRIGger:LOGic;CLOCk:SLOPe?
```

Example
```
:TRIGGER:LOGIC:CLOCK:SLOPE RISE
:TRIGGER:LOGIC:CLOCK:SLOPE?→:TRIGGER:
LOGIC:CLOCK:SLOPE RISE
```

Description
- Query/Setting is not possible if the logic probe is not connected.
- An error results if "TRIGger:CLOCk:SOURce NONE" is specified.

## :TRIGger:LOGic:CLOCk:SOURce

Function Sets the clock channel when using the logic trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic;CLOCk:SOURce {<NRf>|NONE}
:TRIGger:LOGic;CLOCk:SOURce?
```

Example
```
:TRIGGER:LOGIC:CLOCK:SOURCE 1
:TRIGGER:LOGIC:CLOCK:SOURCE?→:TRIGGER:
LOGIC:CLOCK:SOURCE 1
```

Description Query/Setting is not possible if the logic probe is not connected.

## :TRIGger:LOGic:CONDition

Function Sets the trigger condition when using the logic trigger or queries the current setting.

Syntax
```
:TRIGger:LOGic:CONDition {ENTer|EXIT|
FALSe|TRUE}
:TRIGger:LOGic:PATTern:CONDition?
```

Example
```
:TRIGGER:LOGIC:CONDITION ENTER
:TRIGGER:LOGIC:CONDITION?→:TRIGGER:
LOGIC:CONDITION ENTER
```

Description
- Query/Setting is not possible if the logic probe is not connected.
- Select the condition from {ENTer|EXIT} when "TRIGger:LOGic:CLOCk:SOURce NONE" is specified and from {FALSe|TRUE}, otherwise.

## :TRIGger:MODE

Function Sets the trigger mode or queries the current setting.

Syntax
```
:TRIGger:MODE {AUTO|ALEVel|NORMal|
SINGle|NSINgle}
:TRIGger:MODE?
```

Example
```
:TRIGGER:MODE AUTO
:TRIGGER:MODE?→:TRIGGER:MODE AUTO
```

Description Use the ":TRIGger:SCOunt" command to query the count when "NSINgle" is selected.

## :TRIGger:OR?

Function Queries all settings related to the OR trigger.

Syntax
```
:TRIGger:OR?
```

Example
```
:TRIGGER:OR?→:TRIGGER:OR:WINDOW 0;
CHANNEL1 OFF;CHANNEL2 OFF;CHANNEL3 OFF;
CHANNEL4 OFF
```

## :TRIGger:OR:CHANnel<x>

Function Sets the edge of the OR trigger of each channel or queries the current setting.

Syntax
```
:TRIGger:OR:CHANnel<x> {OFF|RISE|FALL|
IN|OUT}
:TRIGger:OR:CHANnel<x>?
<x>=1 to 4
```

Example
```
(The following example is for CH1.)
:TRIGGER:OR:CHANNEL1 RISE
:TRIGGER:OR:CHANNEL1?→:TRIGGER:OR:
CHANNEL1 RISE
```

Description Select from {IN|OUT} when "TRIGger:OR:WINDow ON" is specified and from {FALL|RISE}, otherwise.

## :TRIGger:OR:WINDow

Function Turns the OR trigger window ON/OFF or queries the current setting.

Syntax
```
:TRIGger:OR:WINDow {<Boolean>}
:TRIGger:OR:WINDow?
```

Example
```
:TRIGGER:OR:WINDOW ON
:TRIGGER:OR:WINDOW?→:TRIGGER:OR:
WINDOW 1
```

## :TRIGger:PATTern?

Function Queries all settings related to the pattern trigger.

Syntax
```
:TRIGger:PATTern?
```

Example
```
:TRIGGER:PATTERN?→:TRIGGER:PATTERN:
CLOCK:SOURCE NONE;:TRIGGER:PATTERN:
CHANNEL1 HIGH;CHANNEL2 DONTCARE;
CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;
CONDITION ENTER
```

## :TRIGger:PATTern:CHANnel<x>

Function Sets the state of each channel when using the pattern trigger or queries the current setting.

Syntax
```
:TRIGger:PATTern:CHANnel<x> {DONTcare|
HIGH|LOW}
:TRIGger:PATTern:CHANnel<x>?
<x>=1 to 4
```

Example
```
:TRIGGER:PATTERN:CHANNEL1 DONTCARE
:TRIGGER:PATTERN:CHANNEL1?→:TRIGGER:
PATTERN:CHANNEL1 DONTCARE
```

## :TRIGger:PATTern:CLOCk?

Function Queries all settings related to the clock channel when using the pattern trigger.

Syntax
```
:TRIGger:PATTern:CLOCk?
```

Example
```
:TRIGGER:PATTERN:CLOCK?→:TRIGGER:
PATTERN:CLOCK:SOURCE 1;SLOPE RISE
```

## :TRIGger:PATTern:CLOCk:SLOPe

Function Sets the slope of the clock channel when using the pattern trigger or queries the current setting.

Syntax
```
:TRIGger:PATTern:CLOCk:SLOPe{FALL|RISE}
:TRIGger:PATTern:CLOCk:SLOPe?
```

Example
```
:TRIGGER:PATTERN:CLOCK:SLOPE RISE
:TRIGGER:PATTERN:CLOCK:SLOPE?→:TRIGGER:
SIMPLE:PATTERN:CLOCK:SLOPE RISE
```

Description An error results if "TRIGger:PATTern:CLOCk:SOURce NONE" is specified.

## :TRIGger:PATTern:CLOCk:SOURce

Function Sets the clock channel when using the pattern trigger or queries the current setting.

Syntax
```
:TRIGger:PATTern:CLOCk:SOURce {<NRf>|
NONE}
:TRIGger:PATTern:CLOCk:SOURce?
<NRf>=1 to 4
```

Example
```
:TRIGGER:PATTERN:CLOCK:SOURCE 1
:TRIGGER:PATTERN:CLOCK:SOURCE?→
:TRIGGER:PATTERN:CLOCK:SOURCE 1
```

## :TRIGger:PATTern:CONDition

Function Sets the trigger condition when using the pattern trigger or queries the current setting.

Syntax
```
:TRIGger:PATTern:CONDition {ENTer|EXIT|
FALSe|TRUE}
:TRIGger:PATTern:CONDition?
```

Example
```
:TRIGGER:PATTERN:CONDITION ENTER
:TRIGGER:PATTERN:CONDITION?→:TRIGGER:
PATTERN:CONDITION ENTER
```

Description Select the condition from {ENTer|EXIT} when "TRIGger:PATTERN:CLOCk:SOURce NONE" is specified and from {FALSe|TRUE}, otherwise.

## :TRIGger:POSition

Function Sets the trigger position or queries the current setting.

Syntax
```
:TRIGger:POSition {<NRf>}
:TRIGger:POSition?
<NRf>=0 to 100 (%, 0.1 steps)
```

Example
```
:TRIGGER:POSITION 50
:TRIGGER:POSITION?→:TRIGGER:
POSITION 50.000
```

## :TRIGger:SCOunt (Single (N) Count)

Function Sets the number of times the trigger is activated when the trigger mode is set to Single (N) or queries the current setting.

Syntax
```
:TRIGger:SCOunt {<NRf>}
:TRIGger:SCOunt?
<NRf>=1 to 2048 (DL7100)
      1 to 4096 (DL7200)
```

Example
```
:TRIGGER:SCOUNT 100
:TRIGGER:SCOUNT?→:TRIGGER:SCOUNT 100
```

## :TRIGger[:SIMPle]?

Function Queries all settings related to the simple trigger.

Syntax
```
:TRIGger:SIMPle?
```

Example
```
:TRIGGER:SIMPLE?→:TRIGGER:SIMPLE:
SOURCE 1;LEVEL 0.0E+00;SLOPE RISE;
COUPLING DC;HFREJECTION OFF;
HYSTERESIS LOW
```

## :TRIGger[:SIMPle]:COUPling

Function Sets the trigger coupling for the simple trigger or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:COUPling {AC|DC}
:TRIGger[:SIMPle]:COUPling?
```

Example
```
:TRIGGER:SIMPLE:COUPLING DC
:TRIGGER:SIMPLE:COUPLING?→:TRIGGER:
SIMPLE:COUPLING DC
```

Description Setting/Querying is not possible when the trigger source is set to "EXTernal" or "LINE."

## :TRIGger[:SIMPle]:HFRejection (HighFrequencyREJECTION)

Function Turns the low-pass filter (HF rejection) for the trigger signal ON/OFF or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:HFRejection {OFF|
<frequency>}
:TRIGger[:SIMPle]:HFRejection?
{<frequency>}=20MHz,15KHz
```

Example
```
:TRIGGER:SIMPLE:HFREJECTION 20MHZ
:TRIGGER:SIMPLE:HFREJECTION?→:TRIGGER:
SIMPLE:HFREJECTION 20.0E+06
```

Description Setting/Querying is not possible when the trigger source is set to "EXTernal" or "LINE."

## :TRIGger[:SIMPle]:HYSTeresis

Function Sets the hysteresis on the search level of the simple trigger or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:HYSTeresis {HIGH|LOW}
:TRIGger[:SIMPle]:HYSTeresis?
```

Example
```
:TRIGGER:SIMPlE:HYSTERESIS HIGH
:TRIGGER:SIMPlE:HYSTERESIS?→:TRIGGER:
SIMPlE:HYSTERESIS HIGH
```

Description Setting/Querying is not possible when the trigger source is set to "EXTernal" or "LINE."

## :TRIGger[:SIMPle]:LEVel

Function Sets the trigger level of the channel specified with the "TRIGger[:SIMPle]:SOURce" command or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:LEVel {<voltage>}
:TRIGger[:SIMPle]:LEVel?
<voltage>= See the IM701410-01E User's
           Manual.
```

Example
```
:TRIGGER:SIMPLE:LEVEL 0V
:TRIGGER:SIMPLE:LEVEL?→:TRIGGER:SIMPLE:
LEVEL 0.000E+00
```

Description Setting/Querying is not possible when the trigger source is set to "LINE."

## :TRIGger[:SIMPle]:PROBe

Function Sets the probe when the trigger source is set to "EXTernal" or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:PROBe {<NRf>}
:TRIGger[:SIMPle]:PROBe?
<NRf>=1,10
```

Example
```
:TRIGGER:SIMPLE:PROBE 1
:TRIGGER:SIMPLE:PROBE→:TRIGGER:SIMPLE:
PROBE 1
```

Description Setting/Querying is not possible unless the trigger source is set to "EXTernal."

## :TRIGger[:SIMPle]:SLOPe

Function Sets the trigger slope of the channel specified with the "TRIGger[:SIMPle]:SOURce" command or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:SLOPe {BOTH|FALL|RISE}
:TRIGger[:SIMPle]:SLOPe?
```

Example
```
:TRIGGER:SIMPLE:SLOPE RISE
:TRIGGER:SIMPLE:SLOPE?→:TRIGGER:SIMPLE:
SLOPE RISE
```

Description Setting/Querying is not possible when the trigger source is set to "LINE."

## :TRIGger[:SIMPle]:SOURce

Function Sets the trigger source for the simple trigger or queries the current setting.

Syntax
```
:TRIGger[:SIMPle]:SOURce {<NRf>|
EXTernal|LINE}
:TRIGger[:SIMPle]:SOURce?
<NRf>=1 to 4
```

Example
```
:TRIGGER:SIMPLE:SOURCE 1
:TRIGGER:SIMPLE:SOURCE?→:TRIGGER:
SIMPLE:SOURCE 1
```

Description Within the :TRIGger:SIMPle series of commands, "SIMPle" may be omitted except in the case of the :TRIGger:SIMPle:SOURce command.

### :TRIGger:SOURce?

Function Enhanced Queries all settings related to the trigger source for the enhanced trigger.

Syntax
```
:TRIGger:SOURce?
```

Example
```
:TRIGGER:SOURCE?→:TRIGGER:SOURCE:
CHANNEL1:LEVEL 0.0E+00;COUPLING DC;
HFREJECTION OFF;HYSTERESIS HIGH;
CENTER 0.0E+00;WIDTH 2.000E+00;:TRIGGER:
SOURCE:CHANNEL2:LEVEL 0.0E+00;
COUPLING DC;HFREJECTION OFF;
HYSTERESIS HIGH;CENTER 0.0E+00;
WIDTH 20.000E+00;:TRIGGER:SOURCE:
CHANNEL3:LEVEL 0.0E+00;COUPLING DC;
HFREJECTION OFF;HYSTERESIS HIGH;
CENTER 0.0E+00;WIDTH 2.000E+00;:TRIGGER:
SOURCE:CHANNEL4:LEVEL 0.0E+00;
COUPLING DC;HFREJECTION OFF;
HYSTERESIS HIGH;CENTER 0.0E+00;
WIDTH 50.000E+00
```

### :TRIGger:SOURce:CHANnel<x>?

Function Queries all settings related to the trigger source of the enhanced trigger of the specified channel.

Syntax
```
:TRIGger:SOURce:CHANnel<x>?
<x>=1 to 4
```

Example
```
:TRIGGER:SOURCE:CHANNEL1?→:TRIGGER:
SOURCE:CHANNEL1:LEVEL 0.0E+00;
COUPLING DC;HFREJECTION OFF;
HYSTERESIS LOW;CENTER 0.0E+00;
WIDTH 2.000E+00
```

### :TRIGger:SOURce:CHANnel<x>:COUPling

Function Sets the trigger coupling of each channel when using the enhanced trigger or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:COUPling {AC|
DC}
:TRIGger:SOURce:CHANnel<x>:COUPling?
<x>=1 to 4
```

Example
```
:TRIGGER:SOURCE:CHANNEL1:COUPLING DC
:TRIGGER:SOURCE:CHANNEL1:COUPLING?→
:TRIGGER:SOURCE:CHANNEL1:COUPLING DC
```

### :TRIGger:SOURce:CHANnel<x>:HFRejection (HighFrequencyREJECTION)

Function Sets the low-pass filter when using the enhanced trigger or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:HFRejection
{OFF|<frequency>}
:TRIGger:SOURce:CHANnel<x>:HFRejection?
<x>=1 to 4
<frequency>=20 MHz, 15 KHz
```

Example
```
:TRIGGER:SOURCE:CHANNEL1:
HFREJECTION 15KHZ
:TRIGGER:SOURCE:CHANNEL1:HFREJECTION?→
:TRIGGER:SOURCE:CHANNEL1:
HFREJECTION 15.0E+03
```

### :TRIGger:SOURce:CHANnel<x>:HYSTeresis

Function Sets the hysteresis on the search level when using the enhanced trigger or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:HYSTeresis
{HIGH|LOW}
:TRIGger:SOURce:CHANnel<x>:HYSTeresis?
<x>=1 to 4
```

Example
```
:TRIGGER:SOURCE:CHANNEL1:HYSTERESIS HIGH
:TRIGGER:SOURCE:CHANNEL1:HYSTERESIS?→
:TRIGGER:SOURCE:CHANNEL1:HYSTERESIS HIGH
```

### :TRIGger:SOURce:CHANnel<x>:LEVel

Function Sets the trigger level of each channel when using the enhanced trigger or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:LEVel
{<voltage>}
:TRIGger:SOURce:CHANnel<x>:LEVel?
<x>=1 to 4
<voltage>= See the IM701410-01E User's
           Manual.
```

Example
```
:TRIGGER:SOURCE:CHANNEL1:LEVEL 0V
:TRIGGER:SOURCE:CHANNEL1:LEVEL?→
:TRIGGER:SOURCE:CHANNEL1:LEVEL 0.000E+00
```

### :TRIGger:SOURce:CHANnel<x>:WIDTh

Function Sets the width of the window trigger or queries the current setting.

Syntax
```
:TRIGger:SOURce:CHANnel<x>:WIDTh
{<voltage>}
:TRIGger:SOURce:CHANnel<x>:WIDTh?
<x>=1 to 4
<voltage>= one to eight times V/div (1/
           100 of V/div steps)
```

Example
```
:TRIGGER:SOURCE:CHANNEL1:WIDTH 5V
:TRIGGER:SOURCE:CHANNEL1:WIDTH?→
:TRIGGER:SOURCE:CHANNEL1:WIDTH 5.0E+00
```

## :TRIGger:TV?

Function: Queries all settings related to the TV trigger.

Syntax
```
:TRIGger:TV?
```

Example
```
:TRIGGER:TV?→:TRIGGER:TV:TYPE NTSC;
POLARITY NEGATIVE;LEVEL 0.5;FIELD 1;
LINE 5;FRAME 1
```

Description: The trigger source for the TV trigger is fixed to CH1.

## :TRIGger:TV:FIELd

Function: Sets the field on which to activate the TV trigger or queries the current setting.

Syntax
```
:TRIGger:TV:FIELd {DONTcare|<NRf>}
:TRIGger:TV:FIELd?
<NRf>=1 or 2
```

Example
```
:TRIGGER:TV:FIELD DONTCARE
:TRIGGER:TV:FIELD?→:TRIGGER:TV:
FIELD DONTCARE
```

## :TRIGger:TV:FRAMe

Function: Sets the frame skip function of the TV trigger or queries the current setting.

Syntax
```
:TRIGger:TV:FRAMe {<NRf>}
:TRIGger:TV:FRAMe?
<NRf>=1, 2, 4, or 8
```

Example
```
:TRIGGER:TV:FRAME 2
:TRIGGER:TV:FRAME?→:TRIGGER:TV:FRAME 2
```

## :TRIGger:TV:LEVel

Function: Sets the trigger level of the TV trigger or queries the current setting.

Syntax
```
:TRIGger:TV:LEVel {<NRf>}
:TRIGger:TV:LEVel?
<NRf>=0.1 to 2 div (0.1 div steps)
```

Example
```
:TRIGGER:TV:LEVEL 0.1
:TRIGGER:TV:LEVEL?→:TRIGGER:TV:
LEVEL 0.1
```

## :TRIGger:TV:LINE

Function: Sets the line on which to activate the TV trigger or queries the current setting.

Syntax
```
:TRIGger:TV:LINE {<NRf>}
:TRIGger:TV:LINE?
<NRf>=5 to 1054 (for NTSC)
      2 to 1251 (for PAL)
      2 to 2251 (for HDTV)
```

Example
```
:TRIGGER:TV:LINE 5
:TRIGGER:TV:LINE?→:TRIGGER:TV:LINE 5
```

## :TRIGger:TV:POLarity

Function: Sets the TV trigger's input polarity or queries the current setting.

Syntax
```
:TRIGger:TV:POLarity {POSitive|NEGative}
:TRIGger:TV:POLarity?
```

Example
```
:TRIGGER:TV:POLARITY POSITIVE
:TRIGGER:TV:POLARITY?→:TRIGGER:TV:
POLARITY POSITIVE
```

## :TRIGger:TV:TYPE

Function: Sets the TV trigger's input type or queries the current setting.

Syntax
```
:TRIGger:TV:TYPE {HDTV|NTSC|PAL}
:TRIGger:TV:TYPE?
```

Example
```
:TRIGGER:TV:TYPE NTSC
:TRIGGER:TV:TYPE?→:TRIGGER:TV:TYPE NTSC
```

Description: The trigger source for the TV trigger is fixed to CH1.

## :TRIGger:TYPE

Function: Sets the trigger type or queries the current setting.

Syntax
```
:TRIGger:TYPE {ABN|ADB|PATTern|WIDTh|OR|
TV|LOGic|SIMPle|CANBus}
:TRIGger:TYPE?
```

Example
```
:TRIGGER:TYPE
SIMPLE:TRIGGER:TYPE?→:TRIGGER:
TYPE SIMPLE
```

Description: "`:TRIGger:TYPE LOGic`" cannot be specified unless the logic probe (option) is connected.
"`:TRIGger:TYPE CANBus`" cannot be specified unless the CAN Bus signal analysis function (option) is connected.

## :TRIGger:WIDTh?

Function: Queries all settings related to the pulse width trigger.

Syntax
```
:TRIGger:WIDTh?
```

Example
```
:TRIGGER:WIDTH?→:TRIGGER:WIDTH:TYPE IN;
CHANNEL1 HIGH;CHANNEL2 HIGH;
CHANNEL3 HIGH;CHANNEL4 HIGH;
CONDITION TRUE;TLINKAGE 0;TIME1 1.0E-09;
TIME2 2.0E-09;WINDOW 0
```

### :TRIGger:WIDTh:CHANnel<x>

Function Sets the trigger condition of each channel when using the pulse width trigger  or queries the current setting.

Syntax
```
:TRIGger:WIDTh:CHANnel<x> {HIGH|LOW|IN|
OUT|DONTcare}
:TRIGger:WIDTh:CHANnel<x>?
<x>=1 to 4
```

Example
```
:TRIGGER:WIDTH:CHANNEL1 IN
:TRIGGER:WIDTH:CHANNEL1?→:TRIGGER:
WIDTH:CHANNEL1 IN
```

Description Select from {IN|OUT} when "TRIGger:OR:WINDow ON" is specified and from {FALL|RISE}, other wise.

### :TRIGger:WIDTh:CONDition

Function Sets the trigger condition when using the pulse width trigger or queries the current setting.

Syntax
```
:TRIGger:WIDTh:CONDition{TRUE|FALSe}
:TRIGger:WIDTh:CONDition?
```

Example
```
:TRIGGER:WIDTH:CONDITION TRUE
:TRIGGER:WIDTH:CONDITION?→:TRIGGER:
WIDTH:CONDITION TRUE
```

### :TRIGger:WIDTh:TIME<x>

Function Sets the pulse width when using the pulse width trigger or queries the current setting.

Syntax
```
:TRIGger:WIDTh:TIME<x> <time>
:TRIGger:WIDTh:TIME<x>?
<x>=1 or 2
<time>=1 ns to 1 s (when <x>=1 and
       "T1<Pulse<T2" is not true)
       1 ns to 0.999999999 s (when <x>=1
       and "T1<Pulse<T2")
       2 ns to 1 s (when <x>=2 and
       "T1<Pulse<T2")
```

Example
```
:TRIGGER:WIDTH:TIME1 10NS
:TRIGGER:WIDTH:TIME1?→:TRIGGER:WIDTH:
TIME1 10.0E-09
```

### :TRIGger:WIDTh:TLINkage

Function Turns the link ON/OFF when changing the pulse width for the pulse width trigger or queries the current setting.

Syntax
```
:TRIGger:WIDTh:TLINkage {<Boolean>}
:TRIGger:WIDTh:TLINkage?
```

Example
```
:TRIGGER:WIDTH:TLINKAGE ON
:TRIGGER:WIDTH:TLINKAGE?→:TRIGGER:
WIDTH:TLINKAGE 1
```

### :TRIGger:WIDTh:TYPE

Function Sets the trigger type when using the pulse width trigger or queries the current setting.

Syntax
```
:TRIGger:WIDTh:TYPE {IN|OUT|BETWeen|
TIMeout}
:TRIGger:WIDTh:TYPE?
```

Example
```
:TRIGGER:WIDTH:TYPE BETWEEN
:TRIGGER:WIDTH:TYPE?→:TRIGGER:WIDTH:
TYPE BETWEEN
```

### :TRIGger:WIDTh:WINDow

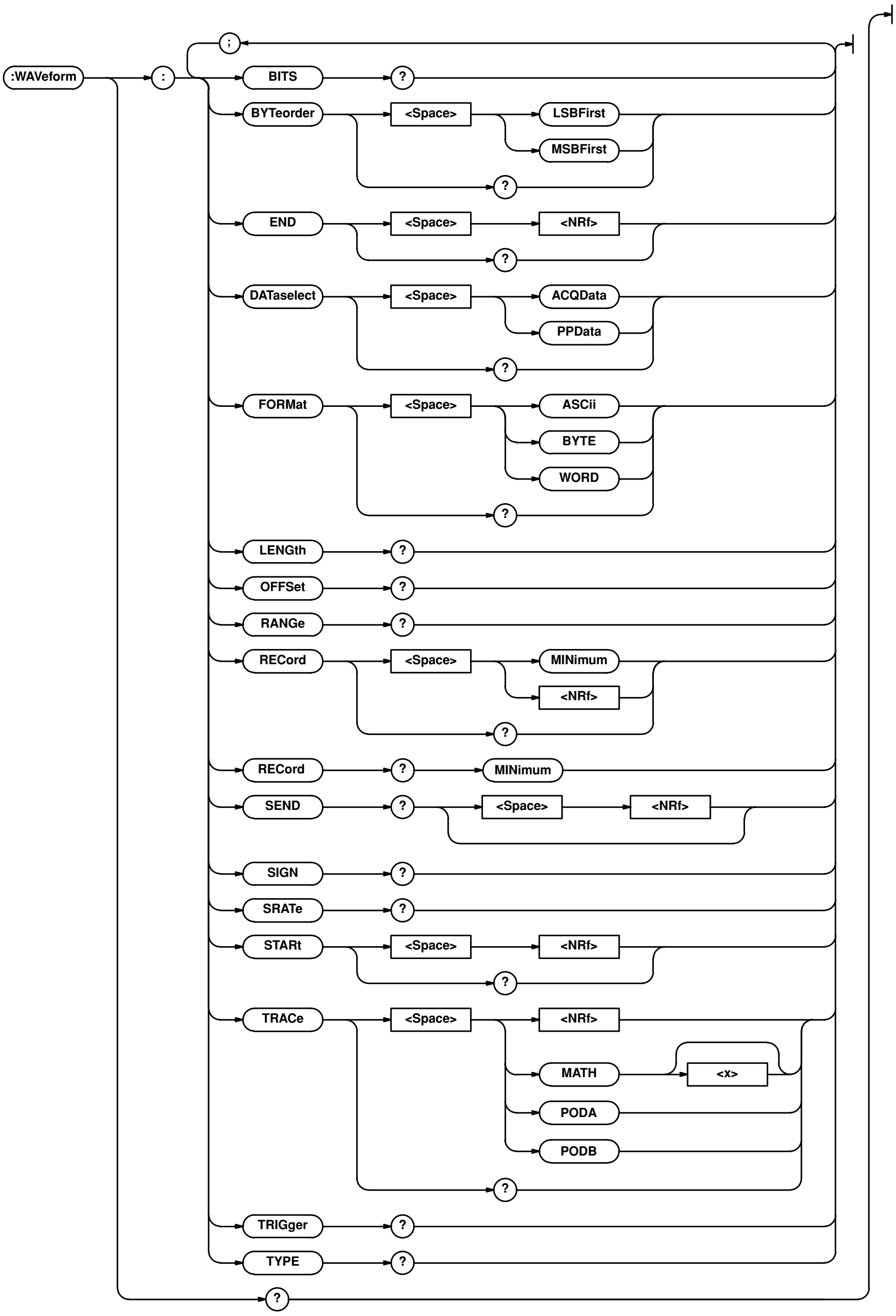
Function Turns the window ON/OFF when using the pulse width trigger or queries the current setting.

Syntax
```
:TRIGger:WIDTh:WINDow {<Boolean>}
:TRIGger:WIDTh:WINDow?
```

Example
```
:TRIGGER:WIDTH:WINDOW ON
:TRIGGER:WIDTH:WINDOW?→:TRIGGER:WIDTH:
WINDOW 1
```

## 5.30 WAVeform Group

The commands in the WAVeform group are used to make settings and queries about acquired waveform data. There is no front panel key for this function.



### :WAVeform?

Function Queries all information relating to the waveform data.

Syntax :WAVeform?

Example :WAVEFORM?→:WAVEFORM:TRACE 1;RECORD 0; START 0;END 0;DATASELECT ACQDATA; FORMAT WORD;BYTEORDER LSBFIRST

### :WAVeform:BITS?

Function Queries the bit length of the waveform data designated by "WAVeform:TRACe."

Syntax :WAVeform:BITS?

Example :WAVEFORM:BITS?→:WAVEFORM:BITS 16

### :WAVeform:BYTeorder

Function Sets/queries the order used to transmit words consisting of multiple bytes.

Syntax :WAVeform:BYTeorder {LSBFirst|MSBFirst}
:WAVeform:BYTeorder?

Example :WAVEFORM:BYTEORDER LSBFIRST
:WAVEFORM:BYTEORDER?→:WAVEFORM:
BYTEORDER LSBFIRST

### :WAVeform:DATaselect

Function Sets whether to use the ACQ data or P-P data when querying the waveform that is specified with the "WAVeform:TRACe" command or queries the current setting.

Syntax :WAVeform:DATaselect {ACQData|PPData}
:WAVeform:DATaselect?

Example :WAVEFORM:DATASELECT ACQDATA
:WAVEFORM:DATASELECT?→:WAVEFORM:
DATASELECT ACQDATA

### :WAVeform:END

Function Sets/queries the point at which the last item of data of the waveform designated by "WAVeform:TRACe" is to be located.

Syntax :WAVeform:END {<NRf>}
:WAVeform:END?
<NRf>=0 to 16031999

Example· :WAVEFORM:END 1001
:WAVEFORM:END?→:WAVEFORM:END 1001

Description You can query total data length (points) with the ":WAVeform:LENgth" command.

### :WAVeform:FORMat

Function Sets/queries the format in which the data is to be transmitted.

Syntax :WAVeform:FORMat {ASCii|BYTE|WORD}
:WAVeform:FORMat?

Example :WAVEFORM:FORMAT ASCII
:WAVEFORM:FORMAT?→:WAVEFORM:
FORMAT ASCII

Description For information on the different formats selected with this command, refer to the Description section of "WAVeform:SEND?."

### :WAVeform:LENGth?

Function Queries the number of records (total number of data points) in the waveform designated by "WAVeform:TRACe."

Syntax :WAVeform:LENGth?

Example :WAVEFORM:LENGTH?→:WAVEFORM:LENGTH 1002

### :WAVeform:OFFSet?

Function Queries the offset value that is used when converting the waveform specified by the "WAVeform:TRACe" command to physical values.

Syntax :WAVeform:OFFSet?

Example :WAVEFORM:OFFSET?→0.000E+00

Description This offset value is used when converting the <block data> that are output by the "WAVeform:SEND?" command to physical values.

### :WAVeform:RANGe?

Function Queries the range that is used when converting the waveform specified by the ":WAVeform:TRACe" command to physical values.

Syntax :WAVeform:RANGe?

Example :WAVeform:RANGe?→5.000E+00

Description This value is used when converting the <block data> that is output with the "WAVeform:SEND?" command to physical values.

### :WAVeform:RECord

Function Sets/queries the target record No. for the WAVeform group.

Syntax :WAVeform:RECord {MINimum|<NRf>}
:WAVeform:RECord?
<NRf>=0 to -2047 (DL7100)
      0 to -4095 (DL7200)

Example :WAVEFORM:RECORD 0
:WAVEFORM:RECORD?→:WAVEFORM:RECORD 0

Description If "MINimum" is specified, the record number is set to the minimum value. The record numbers that can be selected vary depending on the extension memory and the acquisition settings. For details, see the IM701410-01E User's Manual.

### :WAVeform:RECord? MINimum

Function Queries the smallest record number of the history of the channel.

Syntax :WAVeform:RECord? MINimum

Example :WAVEFORM:RECORD?
MINimum→:WAVEFORM:RECORD -1023

## :WAVeform:SEND?

Function Queries the waveform data specified with the "WAVeform:TRACe" command.

Syntax
```
:WAVeform:SEND? [{<NRf>}]
<NRf>=0 to 2048 (DL7100)
      0 to 4096 (DL7200)
(depends on the record length setting)
```

Example
```
:WAVEFORM:SEND?→#8 (number of bytes, 8
digits)(data string) or, <NRf>,<NRf>,
....
```

Description The output format of "WAVeform:SEND?" varies depending on the "WAVeform:FORMat" setting.

(1)When set to "ASCii"

- When the "WAVeform:TRACe" is 1 to 4 and the linear scaling is OFF, the values are returned in the following format: <voltage>, <voltage>, ...<voltage>
- When the "WAVeform:TRACe" is a logic probe, the values are returned in the following format: <NR1>, <NR1>, ..., <NR1>
  <NR1> is an 8-bit pattern (bit 7 to 0) that has been made into a decimal.
- For all other cases, the values are returned in the form: <NRf>, <NRf>, ..., <NRf>.

(2)When set to "BYTE" or "WORD"

The values are returned in the <block data> format.

- When the "WAVeform:TRACe" is not set to a logic probe, the following equation can be used to convert the values.
  Voltage (computed value) =
  (Range × data ÷ division*) + offset
  * "BYTE":Division = 12
  "WORD":Division = 3072
  Query is not possible if "WAVeform:TRACe" set to MATH1 or MATH2 and the computation is turned OFF.
- When the "WAVeform:TRACe" is a logic probe
  An 8-bit pattern (bits 7 to 0) is placed in the MSB byte when "WORD" is selected.
  An 8-bit pattern (bits 7 to 0) is placed in the byte.
- <NRf> can be omitted. If <NRf> is added, waveform data are queried <NRf> times from the record number specified by the ":WAVeform:RECord" command - <NRf> in order.

## :WAVeform:SIGN?

Function Queries the presence of the sign, when querying the waveform specified with "WAVeform:TRACe" in binary data form.

Syntax
```
:WAVeform:SIGN?
```

Example
```
:WAVEFORM:SIGN?→:WAVEFORM:SIGN 1
```

## :WAVeform:SRATe? (Sample RATE)

Function Queries the sampling rate for the record designated by "WAVeform:RECord".

Syntax
```
:WAVeform:SRATe?
```

Example
```
:WAVEFORM:SRATE?→:WAVEFORM:
SRATE 1.0E+09
```

## :WAVeform:STARt

Function Sets/queries the point at which the first item of data of the waveform designated by "WAVeform:TRACe" is to be located.

Syntax
```
:WAVeform:STARt {<NRf>}
:WAVeform:STARt?
<NRf>=0 to 16031999
```

Example
```
:WAVEFORM:START 0
:WAVEFORM:START?→:WAVEFORM:START 0
```

## :WAVeform:TRACe

Function Sets/queries the target waveform for the WAVeform group.

Syntax
```
:WAVeform:TRACe {<NRf>|MATH<x>|PODA|
PODB}
:WAVeform:TRACe?
<x>=1 or 2
<NRf>=1 to 4
```

Example
```
:WAVEFORM:TRACE 1
:WAVEFORM:TRACE?→:WAVEFORM:TRACE 1
```

## :WAVeform:TRIGger?

Function Queries the trigger point for the record designated by "WAVeform:RECord".

Syntax
```
:WAVeform:TRIGger?
```

Example
```
:WAVEFORM:TRIGGER?→:WAVEFORM:
TRIGGER 2000
```

Description Returns the number of points between the record start and the trigger position.

## :WAVeform:TYPE?

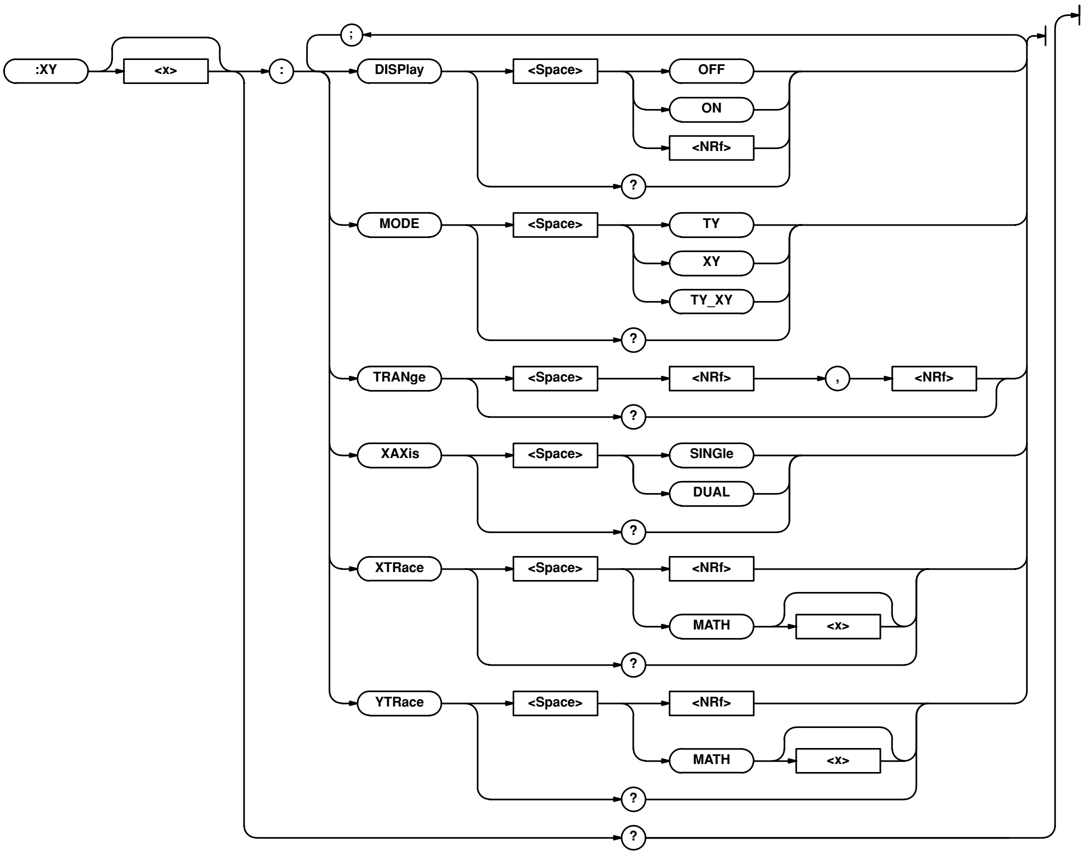
Function Queries the acquisition mode for the waveform specified by the "WAVeform:TRACe" command.

Syntax
```
:WAVeform:TYPE?
```

Example
```
:WAVEFORM:TYPE?→:WAVEFORM:TYPE NORMAL
```

## 5.31 XY Group

The commands in the XY group are used to make settings and queries about the XY display. You can make the same settings that can make using the X-Y key (SHIFT+DISPLAY) on the front panel.



### :XY<x>?

Function Queries all settings related to the X-Y display.

Syntax `:XY<x>?`
`<x>=1 or 2`

Example `:XY1?→:XY1:MODE XY;XAXIS SINGLE;`
`DISPLAY 1;TRANGE -5.00,5.00;XTRACE 1;`
`YTRACE 2`

### :XY<x>:DISPlay

Function Sets whether or not to display the X-Y waveform or queries the current setting.

Syntax `:XY<x>:DISPlay {<Boolean>}`
`:XY<x>:DISPlay?`
`<x>=1 or 2`

Example `:XY1:DISPlay ON`
`:XY1:DISPlay?→:XY1:DISPlay 1`

### :XY<x>:MODE

Function Sets the display mode or queries the current setting.

Syntax `:XY<x>:MODE {T-Y|TY_XY|XY}`
`:XY<x>:MODE?`
`<x>=1 or 2`

Example `:XY1:MODE XY`
`:XY1:MODE?→:XY1:MODE XY`

Description The setting or query is the same whether <x> is set to 1 or 2.

### :XY:TRANge (Time Range)

Function Sets the range of the T-Y waveform to be used in the X-Y display or queries the current setting.

Syntax `:XY<x>:TRANge {<NRf>,<NRf>}`
`:XY<x>:TRANge?`
`<x>=1 or 2`
`<NRf>=-5 to 5 div (10 div/displayed`
`record length steps)`

Example `:XY1:TRANGE -4,4`
`:XY1:TRANGE?→:XY1:TRANGE -4.00,4.00`

### :XY<x>:XAXis

Function Sets whether or not use the same channel for the X-axis for XY1 and XY2 in the X-Y display.

Syntax
```
:XY<x>:XAXis {SINGle|DUAL}
:XY<x>:XAXis?
<x>=1 or 2
```

Example
```
:XY1:XAXIS SINGLE
:XY1:XAXIS?→:XY1:XAXIS SINGLE
```

Description The setting or query is the same whether <x> is set to 1 or 2.

### :XY<x>:XTRace

Function Sets the channel that is to be assigned to the X-axis in the X-Y display or queries the current setting.

Syntax
```
:XY<x>:XTRace {<NRf>|MATH<x>}
:XY<x>:XTRace?
The <x> in XY<x>=1 or 2
The <x> in MATH<x>=1 or 2
<NRf>=1 to 4
```

Example
```
:XY1:XTRACE 1
:XY1:XTRACE?→:XY1:XTRACE 1
```

Description
- When "XY<x>:XAXis" is set to SINGle
  <NRf>=1 to 4
  The <x> in MATH<x>=1 or 2
- When "XY<x>:XAXis" is set to DUAL
  If XY<x>=XY1:
    <NRf>=1 or 2
    The <x> in MATH<x>=1
  If XY<x>=XY2:
    <NRf>=3 or 4
    The <x> in MATH<x>=2

### :XY<x>:YTRace

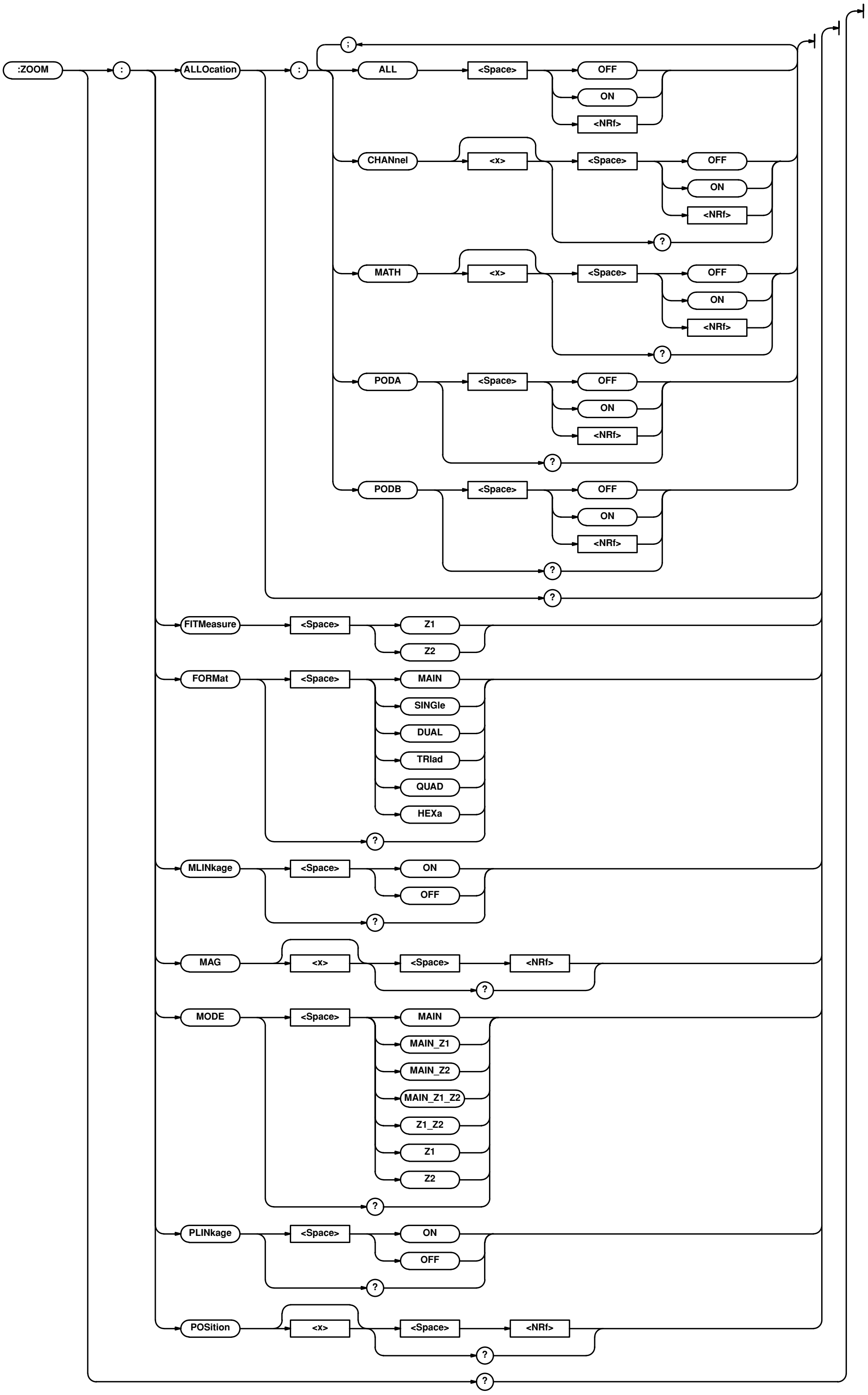
Function Sets the channel that is to be assigned to the Y-axis in the X-Y display or queries the current setting.

Syntax
```
:XY<x>:YTRace {<NRf>|MATH<x>}
:XY<x>:YTRace?
The <x> in XY<x>=1 or 2
When the <x> in XY<x> is 1:
  <NRf>=1 or 2
  The <x> in MATH<x>=1
When the <x> in XY<x> is 2:
  <NRf>=3, 4
  The <x> in MATH<x>=2
```

Example
```
:XY1:YTRACE 1
:XY1:YTRACE?→:XY1:YTRACE 1
```

## 5.32 ZOOM Group

The commands in the ZOOM group are used to make settings and queries about the zoom function. These settings can also be made using the ZOOM key on the front panel.



## :ZOOM?

Function Queries all waveform zoom settings.

Syntax
```
:ZOOM?
```

Example
```
:ZOOM?→:ZOOM:MODE MAIN;FORMAT MAIN;
ALLOCATION:CHANNEL1 1;CHANNEL2 0;
CHANNEL3 0;CHANNEL4 0;MATH1 0;MATH2 0;
PODA 0;PODB 0;:ZOOM:MLINKAGE ON;
MAG1 2.0E+00;MAG2 2.0E+00;PLINKAGE OFF;
POSITION1 0.0000000;POSITION2 0.0000000
```

## :ZOOM:ALLOcation?

Function Queries settings for the zoomed waveform.

Syntax
```
:ZOOM:ALLOcation?
```

Example
```
:ZOOM:ALLOCATION?→:ZOOM:ALLOCATION:
CHANNEL1 1;CHANNEL2 0;CHANNEL3 0;
CHANNEL4 0;MATH1 0;MATH2 0;PODA 0;PODB 0
```

## :ZOOM:ALLOcation:ALL

Function Selects all waveforms to be zoomed.

Syntax
```
:ZOOM:ALLOcation:ALL {<Boolean>}
```

Example
```
:ZOOM:ALLOCATION:ALL ON
```

## :ZOOM:ALLOcation:{CHANnel<x>|MATH<x>}

Function Selects/queries the zoomed waveform.

Syntax
```
:ZOOM:ALLOcation:{CHANnel<x>|MATH<x>}
{<Boolean>}
:ZOOM:ALLOcation:{CHANnel<x>|MATH<x>}?
<x>(CHANnel)=1 to 4
<x>(MATH)=1 or 2
```

Example
```
:ZOOM:ALLOCATION:CHANNEL1 ON
:ZOOM:ALLOCATION:CHANNEL1?→:ZOOM:
ALLOCATION:CHANNEL1 1
```

## :ZOOM:ALLOcation:{PODA|PODB}

Function Sets the pod to be zoomed or queries the current setting.

Syntax
```
:ZOOM:ALLOcation:{PODA|PODB} {<Boolean>}
:ZOOM:ALLOcation:{PODA|PODB}?
```

Example
```
:ZOOM:ALLOCATION:PODA ON
:ZOOM:ALLOCATION:PODA?→:ZOOM:
ALLOCATION:PODA 1
```

Description Setting/Querying is not possible if the logic probe is not connected.

## :ZOOM:FITMeasure

Function Changes the range of the automated measurement of waveform parameters to the ends of the zoom box.

Syntax
```
:ZOOM:FITMeasure {Z1|Z2}
:ZOOM:FITMeasure?
```

Example
```
:ZOOM:FITMEAURE Z1
:ZOOM:FITMEAURE?→:ZOOM:FITMEAURE Z1
```

## :ZOOM:FORMat

Function Sets/queries the zoom display format.

Syntax
```
:ZOOM:FORMat {MAIN|SINGle|DUAL|TRIad|
QUAD|HEXa}
:ZOOM:FORMat?
```

Example
```
:ZOOM:FORMAT SINGLE
:ZOOM:FORMAT?→:ZOOM:FORMAT SINGLE
```

## :ZOOM:MLINkage (Mag Linkage)

Function Enables/disables/queries zoom ratio linkage.

Syntax
```
:ZOOM:MLINKage {<Boolean>}
:ZOOM:MLINKage?
```

Example
```
:ZOOM:MLINKAGE ON
:ZOOM:MLINKAGE?→:ZOOM:MLINKAGE 1
```

## :ZOOM:MAG<x>

Function Sets/queries the zoom ratio.

Syntax
```
:ZOOM:MAG<x> {<NRf>}
:ZOOM:MAG<x>?
<x>=1, 2
<NRf>=magnification up to 40 points.
      See the IM701410-01E User's
      Manual.
```

Example
```
:ZOOM:MAG1 2.5
:ZOOM:MAG1?→:ZOOM:MAG1 2.5
```

## :ZOOM:MODE

Function Sets/queries the zoom display mode.

Syntax
```
:ZOOM:MODE {MAIN|MAIN_Z1|MAIN_Z2|
MAIN_Z1_Z2|Z1_Z2|Z1|Z2}
:ZOOM:MODE?
```

Example
```
:ZOOM:MODE MAIN_Z1_Z2
:ZOOM:MODE?→:ZOOM:MODE MAIN_Z1_Z2
```

## :ZOOM:PLINkage (Position Linkage)

Function Turns the link ON/OFF when moving the zoom box or queries the current setting.

Syntax
```
:ZOOM:PLINkage {<Boolean>}
:ZOOM:PLINkage?
```

Example
```
:ZOOM:PLINKAGE ON
:ZOOM:PLINKAGE?→:ZOOM:PLINKAGE 1
```

## :ZOOM:POSition<x>

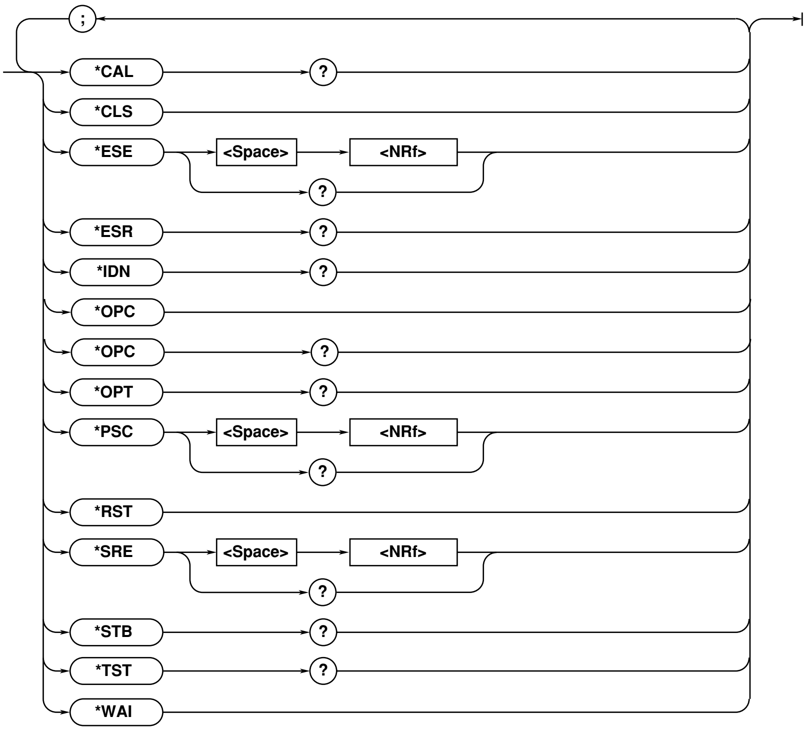
Function Sets/queries the zoom box position.

Syntax
```
:ZOOM:POSition<x> {<NRf>}
:ZOOM:POSition<x>?
<x>=1, 2
<NRf>=-5 to 5 div(in steps of 10 div/
      display-record-length)
```

Example
```
:ZOOM:POSITION1 2
:ZOOM:POSITION1?→:ZOOM:
POSITION1 2.00E+00
```

# 5.33 Common Command Group

The commands in the common command group are independent of the instrument’s functions, and are specified in IEEE 488.2-1987. There is no front panel key that corresponds to this group.



## *CAL? (CALibrate)

Function Performs calibration and queries about the result.

Syntax `*CAL?`

Example `*CAL?→0`

Description "0" will be returned if calibration is completed properly, and "1" will be returned if an abnormality has been detected during calibration.

## *CLS (CLear Status)

Function Clears the standard event register, extended event register and error queue.

Syntax `*CLS`

Example `*CLS`

Description
- The output queue will also be cleared if a *CLS command is appended after the program message terminator.
- For details of the registers and queue, refer to Chapter 6.

## *ESE (standard Event Status Enable register)

Function Sets the value for the standard event enable register/queries about the current setting.

Syntax
```
*ESE {<NRf>}
*ESE?
<NRf>=0 to 255
```

Example
```
*ESE 251
*ESE?→251
```

Description
- <NRf> is the sum of the bits expressed as a decimal number.
- For example, if "*ESE 251" is set, the standard event enable register will be set to "11111011." This means that bit 2 of the standard event register is disabled so that bit 5 (ESB) of the status byte register will not be set to "1," even if an query error occurs.
- Default is "*ESE 0," i.e. all bits are disabled.
- The standard event enable register will not be cleared, even if an query is made using "*ESE?."
- For details of the standard event enable register, refer to page 6-3.

## *ESR? (standard Event Status Register)

Function Queries about the value of the standard event register and clears it at the same time.

Syntax `*ESR?`

Example `*ESR?→32`

Description
- The sum of the bits is returned as a decimal value.
- It is possible to ascertain the type of event which has occurred, while SRQ is occurring.
- For example, if "32" is returned, this means that the standard event register is "00100000," i.e. the SRQ has occurred due to a command syntax error.
- If an query is made using "*ESR?," the standard event register will be cleared.
- For details of the standard event register, refer to page 6-3.

## *IDN? (IDeNtify)

Function Queries about the instrument model.

Syntax `*IDN?`

Example `*IDN?→YOKOGAWA,701410,0,F1.10`

Description A reply consists of the following sequence: <Manufacturer>, <Model>, <Serial No.> and <Firmware version>. "0" is always returned as the <Serial No.>.

<Model>: DL7100 2 M model: 701410
8 M model: 701420
DL7200 4 M model: 701430
16 M model: 701440

## *LRN? (LeaRN)

Function Queries about all the current settings for the following command groups.
`ACQuire, CHANnel<x>, TIMebase, TRIGger`

Syntax `*LRN?`

Example

```
*LRN?→:ACQUIRE:RLENGTH 10000;
MODE NORMAL;AVERAGE:COUNT
INFINITY;EWEIGHT 16;:ACQUIRE:
COUNT INFINITY;INTERLEAVE 0;
REPETITIVE 0;CLOCK INTERNAL;:CHANNEL1:
DISPLAY 1;VDIV 50.000E+00;POSITION 0.00;
COUPLING DC;PROBE 10;OFFSET 0.0E+00;
BWIDTH FULL;VARIABLE 50.000E+00;
LSCALE:MODE 0;AVALUE 1.00000E+00;
BVALUE 0.00000E+00;UNIT "";:CHANNEL1:
LABEL "CH1";:CHANNEL2:DISPLAY 1;V
DIV 50.000E+00;POSITION 0.00;
COUPLING DC;PROBE 10;OFFSET 0.0E+00;
BWIDTH FULL;VARIABLE 50.000E+00;LSCALE:
MODE 0;AVALUE 1.00000E+00;
BVALUE 0.00000E+00;UNIT
"";:CHANNEL2:LABEL "CH2";:CHANNEL3:
DISPLAY 1;VDIV 50.000E+00;POSITION 0.00;
COUPLING DC;PROBE 10;OFFSET 0.0E+00;
BWIDTH FULL;VARIABLE50.000E+00;LSCALE:
MODE 0;AVALUE 1.00000E+00;
BVALUE 0.00000E+00;UNIT "";:CHANNEL3:
LABEL "CH3";:CHANNEL4:DISPLAY 1;
VDIV 50.000E+00;POSITION 0.00;
COUPLING DC;PROBE 10;OFFSET 0.0E+00;
BWIDTH FULL;VARIABLE 50.000E+00;LSCALE:
MODE 0;AVALUE 1.00000E+00;
BVALUE 0.00000E+00;UNIT
"";:CHANNEL4:LABEL "CH4";:TIMEBASE:
SOURCE INTERNAL;TDIV 1.0E-03;:TRIGGER:
ACTION:BUZZER 0;SAVE 0;HCOPY 0;
ACQCOUNT INFINITE;:TRIGGER:SIMPLE:
SOURCE 1;LEVEL 0.0E+00;SLOPE RISE;
COUPLING DC;HFREJECTION OFF;
HYSTERESIS LOW;PROBE 1;:TRIGGER:SOURCE:
CHANNEL1:LEVEL 0.0E+00;COUPLING DC;
HFREJECTION OFF;HYSTERESIS LOW;
WIDTH 50.000E+00;:TRIGGER:SOURCE:
CHANNEL2:LEVEL 0.0E+00;COUPLING DC;
HFREJECTION OFF;HYSTERESIS LOW;
WIDTH 50.000E+00;:TRIGGER:SOURCE:
CHANNEL3:LEVEL 0.0E+00;COUPLING DC;
HFREJECTION OFF;HYSTERESIS LOW;
WIDTH 50.000E+00;:TRIGGER:SOURCE:
CHANNEL4:LEVEL 0.0E+00;COUPLING DC;
HFREJECTION OFF;HYSTERESIS LOW;
WIDTH 50.000E+00;:TRIGGER:ABN:COUNT 1;:
TRIGGER:HOLDOFF:TIME 80.000E-09;:
TRIGGER:ADB:DELAY 1.000E-09;:TRIGGER:
DELAY 0.0E+00;PATTERN:CLOCK:SOURCE 1;
SLOPE RISE;:TRIGGER:PATTERN:
CHANNEL1 HIGH;CHANNEL2 DONTCARE;
CHANNEL3 DONTCARE;CHANNEL4 DONTCARE;
CONDITION TRUE;:TRIGGER:POSITION 50.000;
MODE AUTO;WIDTH:TYPE IN;CHANNEL1 HIGH;
CHANNEL2 HIGH;CHANNEL3 HIGH;
CHANNEL4 HIGH;CONDITION TRUE;
TIME1 1.000E-09;TIME2 1.000E-09;
WINDOW 0;:TRIGGER:OR:WINDOW 0;
CHANNEL1 RISE;CHANNEL2 OFF;CHANNEL3 OFF;
CHANNEL4 OFF;:TRIGGER:SCOUNT 2;TV:
TYPE NTSC;POLARITY NEGATIVE;
LEVEL 5000000.0;FIELD 1;LINE 5;FRAME 1;:
TRIGGER:TYPE SIMPLE;LOGIC:CLOCK:
SOURCE 1;SLOPE RISE;:TRIGGER:LOGIC:
CONDITION TRUE;A0 HIGH;A1 DONTCARE;
A2 DONTCARE;A3 DONTCARE;A4 DONTCARE;
A5 DONTCARE;A6 DONTCARE;A7 DONTCARE;
B0 DONTCARE;B1 DONTCARE;B2 DONTCARE;
B3 DONTCARE;B4 DONTCARE;B5 DONTCARE;
B6 DONTCARE;B7 DONTCARE;:TRIGGER:
ATRIGGER:CHANNEL1 HIGH;
CHANNEL2 DONTCARE;CHANNEL3 DONTCARE;
CHANNEL4 DONTCARE;CONDITION ENTER;:
TRIGGER:BTRIGGER:CHANNEL1 DONTCARE;
CHANNEL2 HIGH;CHANNEL3 DONTCARE;
CHANNEL4 DONTCARE;CONDITION
```

## *OPC (OPeration Complete)

Function: Sets bit 0 (OPC bit) of the standard event register to "1" when execution of an overlap command is completed.

Syntax: `*OPC`

Example: `*OPC`

Description:
- For a description of the synchronization method using "`*OPC`," refer to page 4-8.
- Designation of an overlap command is performed using "`COMMunicate:OPSE`."
- Operation is not guaranteed if "`*OPC`" is not appended to the end of the message.

## *OPC? (OPeration Complete)

Function: After "`*OPC?`" is sent, "1" (ASCII) will be returned if execution of the designated overlap command has been completed.

Syntax: `*OPC?`

Example: `*OPC?→1`

Description:
- For a description of the synchronization method using "`*OPC?`," refer to page 4-8.
- Designation of an overlap command is performed using "`COMMunicate:OPSE`."
- Operation is not guaranteed if "`*OPC`" is not appended to the end of the message.

## *OPT? (OPTion)

Function: Queries the installed options.

Syntax: `*OPT?`

Example: `*OPT?→CH8MW,PRINTER,LOGIC`

Description:
- Returns the existence of the <memory model>, <printer>, <logic option>, and <ethernet>.
- The "`*OPT?`" command must be the last query in a program message. Otherwise, an error results.

## *PSC (Power-on Status Clear)

Function: Selects/queries whether following registers are cleared when power is turned ON. However, they cannot be cleared if the parameter is "0" when rounded.
- Standard event enable register
- Extended event enable register
- Transit filter

Syntax:
```
*PSC {<NRf>}
*PSC?
<NRf>=0(does not clear the registers), a
      value other than 0 (clears the
      registers)
```

Example:
```
*PSC 1
*PSC?→1
```

Description: For details of each register, refer to Chapter 6.

## *RST (ReSeT)

Function: Resets the current settings of the following groups at once.
`ACCumulate, ACQuire, CHANnel<x>, TIMebase, TRIGger`

Syntax: `*RST`

Example: `*RST`

Description: Previously sent `*OPC` and `*OPC?` commands are also cleared.

## *SRE (Service Request Enable register)

Function: Sets the value of the service request enable register/inquires about the current setting.

Syntax:
```
*SRE <NRf>
*SRE?
<NRf>=0 to 255
```

Example:
```
*SRE 239
*SRE?→239
```

Description:
- <NRf> is the sum of the bits expressed as a decimal number. For example, if "`*ESE 239`" is set, the service request enable register will be set to "11101111." This means that bit 4 of the service request enable register is disabled, so that bit 5 (ESB) of the status byte register will not be set to "1," even if the output queue is not empty. However, bit 6 (MSS) of the status byte register is the MSS bit, so it will be ignored.
- Default is "`*SRE 0`," i.e. all bits are disabled.
- The service request enable register will not be cleared, even if a query is made using "`*SRE?`."
- For details of the service request enable register, refer to page 6-1.

## *STB? (STatus Byte)

Function: Queries about the value of the status byte register.

Syntax: `*STB?`

Example: `*STB?→4`

Description:
- The sum of the bits expressed as a decimal value is returned.
- Bit 6 is MSS not RQS, since the register is read without serial polling. For example, if "4" is returned, the status byte register is set to "00000100," i.e. the error queue is not empty (an error has occurred).
- The status byte register will be cleared, even if a query is made using "`*STB?`."
- For details of the status byte register, refer to page 6-2.

### *TST?

Function Executes self test and queries the result. The self test involves internal memory tests.

Syntax `*TST?`

Example `*TST?→0`

Description "`0`" is returned if the self test is successful, "`1`" if it is not.

### *WAI (WAIt)

Function Waits for the command following "`*WAI`" until execution of the designated overlap command is completed.

Syntax `*WAI`

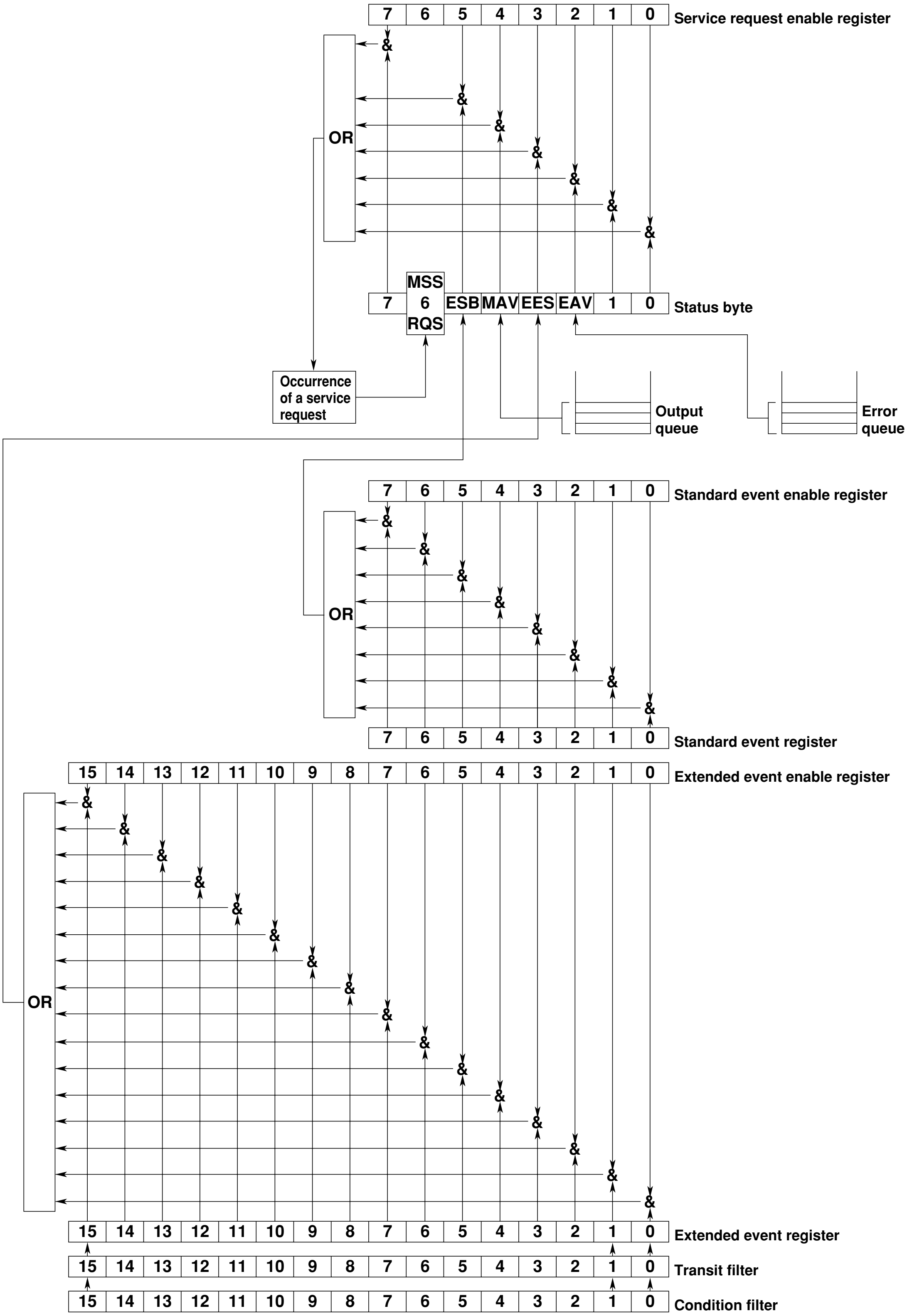
Example `*WAI`

Description
- For a description of the synchronization method using "`*WAI`," refer to page 4-8.
- Designation of overlap commands can be performed using "`COMMunicate:OPSE`."

# Chapter 6 Status Report

## 6.1 Overview of the Status Report

The figure below shows the status report which is read by a serial poll. This is an extended version of the one specified in IEEE 488.2-1992.



### Overview of Registers and Queues

| Name | Function | Writing | Reading |
|---|---|---|---|
| Status byte | | — | Serial poll (RQS), *STB?(MSS) |
| Service request enable register | Masks status byte. | *SRE | *SRE? |
| Standard event register | Change in device status | — | *ESR? |
| Standard event enable register | Masks standard event register | *ESE | *ESE? |
| Extended event register | Change in device status | — | STATus:EESR? |
| Extended event enable register | Masks standard event register | STATus:EESE | STATus:EESE? |
| Condition register | Current instrument status | — | STATus:CONDition? |
| Transit filter | Extended event occurrence conditions | STATus:FILTer<x> | STATus:FILTer<x>? |
| Output queue | Stores response message to a query. | All executable queues | |
| Error queue | Stores error Nos. and messages. | — | STATus:ERRor? |

### Registers and Queues which Affect the Status Byte

Registers which affect each bit of the status byte are shown below.

| | |
|---|---|
| Standard event register | : Sets bit 5 (ESB) of status byte to "1" or "0." |
| Output queue | : Sets bit 4 (MAV) of status byte to "1" or "0." |
| Extended event register | : Sets bit 3 (EES) of status byte to "1" or "0." |
| Error queue | : Sets bit 2 (EAV) of status byte to "1" or "0." |

### Enable Registers

Registers which mask a bit so that the bit does not affect the status byte, even if the bit is set to "1," are shown below.

| | |
|---|---|
| Status byte | : Masks bits using the service request enable register. |
| Standard event register | : Masks bits using the standard event enable register. |
| Extended event register | : Masks bits using the extended event enable register. |

### Writing/Reading from Registers

The *ESE command is used to set bits in the standard event enable register to "1" or "0," and the *ESE query is used to check whether bits in that register are set to "1" or "0." For details of these commands, refer to Chapter 5.

## 6.2 Status Byte

### Overview of Status Byte

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | RQS<br>6<br>MSS | ESB | MAV | EES | EAV | 1 | 0 |

**Bits 0, 1 and 7**

Not used (always "0")

**Bit 2 EAV (Error Available)**

Set to "1" when the error queue is not empty, i.e. when an error occurs. For details, refer to page 6-5.

**Bit 3 EES (Extended Event Summary Bit)**

Sets to "1" when the logical "AND" of an Extended Event Register bit and the corresponding Enable Register bit is equal to "1."—that is, when an event takes place in the instrument. Refer to page 6-4.

**Bit 4 MAV (Message Available)**

Set to "1" when the output queue is not empty, i.e. when there is data which is to be output when an query is made. Refer to page 6-5.

**Bit 5 ESB (Event Summary Bit)**

Set to "1" when the logical AND of the standard event register and the corresponding enable register is "1," i.e. when an event takes place in the instrument. Refer to page 6-3.

**Bit 6 RQS (Request Status)/MSS (Master Summary Status)**

Sets to "1" when the logical "AND" of any one of the Status Byte bits (other than bit 6) and the corresponding Service Request Enable Register bit becomes "1"—that is, when the instrument is requesting service from the controller.

RQS is set to "1" when MSS changes from "0" to "1," and is cleared when a serial poll is performed or when MSS changes to "0."

### Bit Masking

To mask a bit in the status byte so that it does not cause an SRQ, set the corresponding bit of the service request enable register to "0."

For example, to mask bit 2 (EAV) so that no service will be requested, even if an error occurs, set bit 2 of the service request enable register to "0." This can be done using the *SRE command. To query whether each bit of the service request enable register is "1" or "0," use *SRE?. For details of the *SRE command, refer to Chapter 5.

### Operation of the Status Byte

A service request is issued when bit 6 of the status byte becomes "1." Bit 6 becomes "1" when any of the other bits becomes "1" (or when the corresponding bit in the service request enable register becomes "1").

For example, if an event takes place and the logical OR of each bit of the standard event register and the corresponding bit in the enable register is "1," bit 5 (ESB) will be set to "1." In this case, if bit 5 of the service request enable register is "1," bit 6 (MSS) will be set to "1," thus requesting service from the controller.

It is also possible to check what type of event has occurred by reading the contents of the status byte.

### Reading from the Status Byte

The following two methods are provided for reading the status byte.

- **Inquiry using the** *STB? **query**
  Making an query using the *STB? query sets bit 6 to MSS. This causes the MSS to be read. After completion of the read-out, none of the bits in the status byte will be cleared.
- **Serial poll**
  Execution of a serial poll changes bit 6 to RQS. This causes RQS to be read. After completion of the read-out, only RQS is cleared. Using a serial poll, it is not possible to read MSS.

### Clearing the Status Byte

No method is provided for forcibly clearing all the bits in the status byte. Bits which are cleared are shown below.

- **When an query is made using the** *STB? **query**
  No bit is cleared.
- **When a serial poll is performed**
  Only the RQS bit is cleared.
- **When the** *CLS **command is received**
  When the *CLS command is received, the status byte itself is not cleared, but the contents of the standard event register (which affects the bits in the status byte) are cleared. As a result, the corresponding bits in the status byte are cleared, except bit 4 (MAV), since the output queue cannot be emptied by the *CLS command. However, the output queue will also be cleared if the *CLS command is received just after a program message terminator.

## 6.3 Standard Event Register

### Overview of the Standard Event Register

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PON | URQ | CME | EXE | DDE | QYE | RQC | OPC |

**Bit 7 PON (Power ON)**

Bit 7 PON (Power ON) Set to "1" when power is turned ON

**Bit 6 URQ (User Request)**

Not used (always "0")

**Bit 5 CME (Command Error)**

Set to "1" when the command syntax is incorrect.

Examples: Incorrectly spelled command name; "9" used in octal data.

**Bit 4 EXE (Execution Error)**

Set to "1" when the command syntax is correct but the command cannot be executed in the current state.

Examples: Parameters are outside the setting range: an attempt is made to make a hard copy during acquisition.

**Bit 3 DDE (Device Dependent Error)**

Set to "1" when execution of the command is not possible due to an internal problem in the instrument that is not a command error or an execution error.

Example: The circuit breaker is reset.

**Bit 2 QYE (Query Error)**

Set to "1" if the output queue is empty or if the data is missing even after a query has been sent.

Examples: No response data; data is lost due to an overflow in the output queue.

**Bit 1 RQC (Request Control)**

Not used (always "0")

**Bit 0 OPC (Operation Complete)**

Set to "1" when the operation designated by the *OPC command has been completed. Refer to Chapter 5.

### Bit Masking

To mask a bit in the standard event register so that it does not cause bit 5 (ESB) of the status byte to change, set the corresponding bit in the standard event enable register to "0."

For example, to mask bit 2 (QYE) so that ESB will not be set to "1," even if a query error occurs, set bit 2 of the standard event enable register to "0." This can be done using the *ESE command. To inquire whether each bit of the standard event enable register is "1" or "0," use the *ESE?. For details of the *ESE command, refer to Chapter 5.

### Operation of the Standard Event Register

The standard event register is provided for eight different kinds of event which can occur inside the instrument. Bit 5 (ESB) of the status byte is set to "1" when any of the bits in this register becomes "1" (or when the corresponding bit of the standard event enable register becomes "1").

Examples

1. A query error occurs.
2. Bit 2 (QYE) is set to "1."
3. Bit 5 (ESB) of the status byte is set to "1" if bit 2 of the standard event enable register is "1."

It is also possible to check what type of event has occurred inside the instrument by reading the contents of the standard event register.

### Reading from the Standard Event Register

The contents of the standard event register can be read by the *ESR command. After completion of the read-out, the register will be cleared.

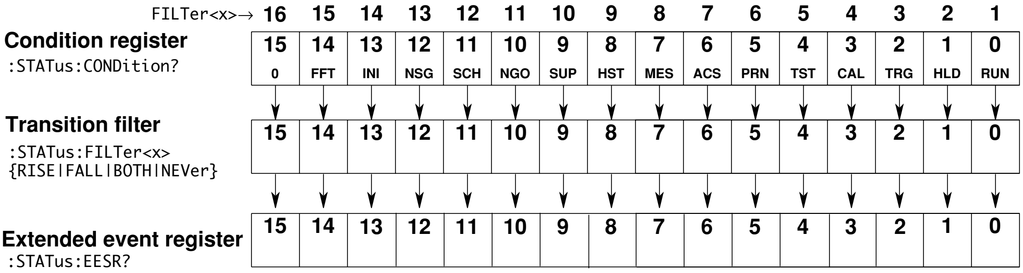
### Clearing the Standard Event Register

The standard event register is cleared in the following three cases.

- When the contents of the standard event register are read using *ESR?
- When the *CLS command is received
- When power is turned ON again

## 6.4 Extended Event Register

**Reading the extended event register tells you whether changes in the condition register (reflecting internal conditions) have occurred. A filter can be applied which allows you to decide which events are reported to the extended event register.**



| FILTer<x>→ | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Condition register<br>:STATus:CONDition? | 15<br>0 | 14<br>FFT | 13<br>INI | 12<br>NSG | 11<br>SCH | 10<br>NGO | 9<br>SUP | 8<br>HST | 7<br>MES | 6<br>ACS | 5<br>PRN | 4<br>TST | 3<br>CAL | 2<br>TRG | 1<br>HLD | 0<br>RUN |
| Transition filter<br>:STATus:FILTer<x><br>{RISE\|FALL\|BOTH\|NEVer} | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Extended event register<br>:STATus:EESR? | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

**The meaning of each bit of the condition register is as follows.**

| | |
|---|---|
| Bit 0 RUN (Running) | Set to "1" during acquisition. |
| Bit 1 HLD (Hold) | Set to "1" for waveforms loaded from a storage medium. |
| Bit 2 TRG (Awaiting trigger) | Set to "1" when waiting for a trigger. |
| Bit 3 CAL (Calibrating) | Set to "1" during calibration. |
| Bit 4 TST (Testing) | Set to "1" during self-test. |
| Bit 5 PRN (Printing) | Set to "1" while the built-in printer is in operation. |
| Bit 6 ACS (Accessing) | Sets to "1" while floppy drive, or external SCSI device is being accessed. |
| Bit 7 MES (Measuring) | Set to "1" when automated measurement of waveform parameters or cursor measurement is in progress. |
| Bit 8 HST (History Search) | Set to "1" while executing history search. |
| Bit 9 SUP (Set-up) | Set to "1" during auto set-up. |
| Bit 10 NGO (Go/No-go) | Set to "1" while performing GO/NO-GO determination. |
| Bit 11 SCH (Search) | Set to "1" while executing edge/pattern search. |
| Bit 12 NSG (N-Single) | Set to "1" during continuous acquisition when the trigger mode is set to single (N). |
| Bit 13 INI (Initializing) | Sets to "1" during initialization. |
| Bit 14 FFT | Set to "1" while performing the FFT. |

**The filter is applied to each bit of the condition register seperately, and can be selected from the following. Note that the numbering of the bits used in the filter setting differs from the actual bit number (1 to 16 vs. 0 to 15).**

| | |
|---|---|
| Rise | The bit of the extended event register becomes "1" when the bit of the condition register changes from "0" to "1." |
| Fall | The bit of the extended event register becomes "1" when the bit of the condition register changes from "1" to "0." |
| Both | The bit of the extended event register becomes "1" when the bit of the condition register changes from "0" to "1," or from "1" to "0." |
| Never | The bit of the extended event register is disabled and always "0." |

## 6.5 Output Queue and Error Queue

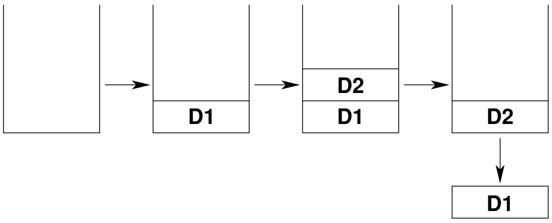
### Overview of the Output Queue

The output queue is provided to store response messages to queries. For example, when the WAVeform:SEND? query is sent to request output of the acquired waveform, the response data will be stored in the output queue until it is read out.

The example below shows that data is stored record by record in the output queue, and is read out oldest item first, newest item last. The output queue is emptied in the following cases (in addition to when read-out is performed).

- When a new message is received from the controller
- When dead lock occurs (page 4-2)
- When a device clear command (DCL or SDC) is received
- When power is turned ON again

The output queue cannot be emptied using the *CLS command. To see whether the output queue is empty or not, check bit 4 (MAV) of the status byte.



### Overview of the Error Queue

The error queue stores the error No. and message when an error occurs. For example, if the controller sends an incorrect program message, the number, "113, "Undefined header"," and the error message are stored in the error queue, when the error is displayed.

The contents of the error queue can be read using the STATus:ERRor? query. As with the output queue, messages are read oldest first, newest last (refer to the previous page).

If the error queue becomes full, the final message will be replaced by message "350, "Queue overflow"."

The error queue is emptied in the following cases (in addition to when read-out is performed).

- When the *CLS command is received
- When power is turned ON again

To see whether the error queue is empty or not, check bit 2 (EAV) of the status byte.

# Chapter 7  Sample Program

## 7.1 Before Programming

### Environment

| | |
|---|---|
| Model: | MS-DOS/V Computer |
| Language: | Visual BASIC Ver5.0 Professional Edition or more |
| GP-IB board: | AT-GPIB/TNT IEEE-488.2 board from National Instruments. |

### Setting up the Visual Basic

| | |
|---|---|
| Compornent: | MSComm |
| Standardmodule: | Niglobal.bas |
| | Vbib-32.bas |

### Setting up the DL7100/DL7200

- **GPIB**

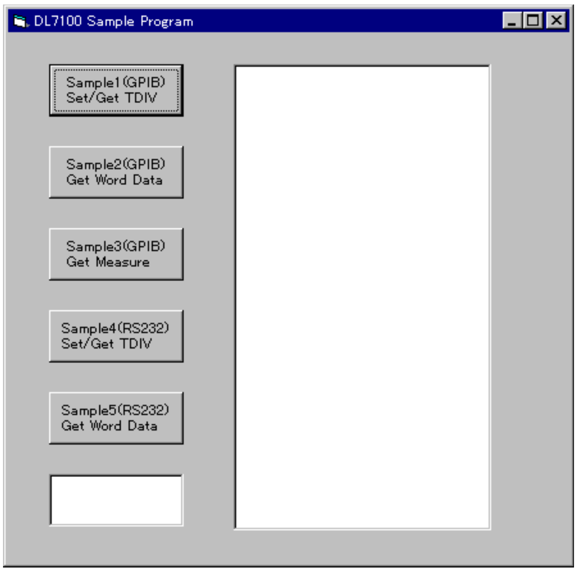
  All the sample programs given in this chapter use address 1 for the DL7100/DL7200, so be sure to assign the instrument to address 1 as described on page 1-6.

- **Serial (RS232)**

  All the sample programs in this chapter assume the below settings, so be sure to set parameters as described on page 2-8.

| | |
|---|---|
| Baud Rate | 19200 |
| Handshake | CTS-RTS |
| Parity Check | NO |
| Stop Bit | 1 |
| Character Length | 8 |
| Terminator | LF |

# 7.2 Image of Sample Program



# 7.3 Initialize/Error/Execute

```
Option Explicit
Dim StartFlag As Integer                           'Start Flag
Dim TimerCount As Integer                          'Timeout(RS232)
Dim Addr As Integer                                'GPIB Address
Dim Timeout As Integer                             'Timeout
Dim Dev As Integer                                 'Device ID(GPIB)
Dim CtsFlag As Integer                             'CTS Flag
Dim Term As String                                 'Terminator
Dim Query(3) As String                             'Query String
Dim WaveBuffer(20070) As Integer                   'WaveData Buffer(GPIB)
Dim Dummy As Integer
```

```
Private Function InitGpib() As Integer
    Dim eos As Integer                             'EOS
    Dim eot As Integer                             'EOI
    Dim brd As Integer                             'GPIB Board ID
    Dim sts As Integer

    eos = &HC0A                                    'Terminator = LF
    eot = 1                                        'EOI = Enable
    Term = Chr(10)
    Timeout = T10s                                 'Timeout = 10s

    brd = ilfind("GPIB0")
    If (brd < 0) Then
        Call DisplayGPIBError(brd, "ilfind")
        InitGpib = 1
        Exit Function
    End If
    Dev = ildev(0, Addr, 0, Timeout, eot, eos)
    If (Dev < 0) Then
        Call DisplayGPIBError(Dev, "ildev")
        InitGpib = 1
        Exit Function
    End If
    sts = ilsic(brd)                               'Set IFC
    If (sts < 0) Then
        Call DisplayGPIBError(sts, "ilsic")
        InitGpib = 1
        Exit Function
    End If
    InitGpib = 0
End Function
```

```
Private Function InitSerial() As Integer
    Dim rat As String

    MSComm1.CommPort = 1                           'Port = COM1

    rat = "19200,N,8,1"                            'Rate = 19200, NoParity, 8Bit,
1Stopbit

    MSComm1.Settings = rat

    MSComm1.Handshaking = comRTS                   'Handshake = CTS-RTS
    MSComm1.RTSEnable = True                       'RTS = TRUE
    CtsFlag = 1
    Term = Chr(10)                                 'Terminator = LF
    Timeout = 10                                   'Timeout = 10s
    InitSerial = 0
End Function
```

```
Private Sub DisplayGPIBError(ByVal sts As Integer, ByVal msg As String)
    Dim wrn As String
    Dim ers As String
    Dim ern As Integer

    If (sts And TIMO) Then
        wrn = "Time out" + Chr(13)
    Else
        wrn = ""
    End If
    If (sts And EERR) Then
        ern = iberr
        If (ern = EDVR) Then
            ers = "EDVR:System error"
        ElseIf (ern = ECIC) Then
            ers = "ECIC:Function requires GPIB board to be CIC"
        ElseIf (ern = ENOL) Then
            ers = "ENOL:No Listeners on the GPIB"
        ElseIf (ern = EADR) Then
            ers = "EADR:GPIB board not addressed correctly"
        ElseIf (ern = EARG) Then
            ers = "EARG:Invalid argument to function call"
        ElseIf (ern = ESAC) Then
            ers = "ESAC:GPIB board not System Controller as required"
        ElseIf (ern = EABO) Then
            ers = "EABO:I/O operation aborted(timeout)"
        ElseIf (ern = ENEB) Then
            ers = "ENEB:Nonexistent GPIB board"
        ElseIf (ern = EDMA) Then
            ers = "EDMA:DMA error"
        ElseIf (ern = EOIP) Then
            ers = "EOIP:I/O operation started before previous operation completed"
        ElseIf (ern = ECAP) Then
            ers = "ECAP:No capability for intended operation"
        ElseIf (ern = EFSO) Then
            ers = "EFSO:File system operation error"
        ElseIf (ern = EBUS) Then
            ers = "EBUS:GPIB bus error"
        ElseIf (ern = ESTB) Then
            ers = "ESTB:Serial poll status byte queue overflow"
        ElseIf (ern = ESRQ) Then
            ers = "ESRQ:SRQ remains asserted"
        ElseIf (ern = ETAB) Then
            ers = "ETAB:The return buffer is full"
        ElseIf (ern = ELCK) Then
            ers = "ELCK:Address or board is locked"
        Else
            ers = ""
        End If
    Else
        ers = ""
    End If

    MsgBox ("Status No. " + Str(sts) + Chr(13) + wrn + "Error No. " + Str(ern) + Chr(13)
+ ers + Chr(13) + msg), vbExclamation, "Error!"
    Call ibonl(Dev, 0)
    Dev = -1
End Sub
```

```
Private Sub DIsplayRS232Error(ByVal erm As String, Optional ByVal msg As String = "")
    MsgBox (erm + Chr(13) + msg), vbExclamation, "Error!"
End Sub
```

```
Private Sub Command1_Click()
    Dim sts As Integer

    If (StartFlag = 1) Then
        Exit Sub
    End If
    StartFlag = 1
    Text1.Text = "START"
    List1.Clear
    Dummy = DoEvents()
    sts = GpibTdiv                                 'Run Sample1(GPIB) Set/Get TDIV
    If (sts = 0) Then
        List1.AddItem Query(0)
    End If
    Text1.Text = "END"
    StartFlag = 0
End Sub
```

```
Private Sub Command2_Click()
    Dim sts As Integer

    If (StartFlag = 1) Then
        Exit Sub
    End If
    StartFlag = 1
    Text1.Text = "START"
    List1.Clear
    Dummy = DoEvents()
    sts = GpibGetWord                              'Run Sample2(GPIB) Get Word Data
    If (sts = 0) Then
        List1.AddItem "END"
    End If
    Text1.Text = "END"
    StartFlag = 0
End Sub
```

```
Private Sub Command3_Click()
    Dim sts As Integer

    If (StartFlag = 1) Then
        Exit Sub
    End If
    StartFlag = 1
    Text1.Text = "START"
    List1.Clear
    Dummy = DoEvents()
    sts = GpibGetMeasure                           'Run Sample3(GPIB) Get Measure
    If (sts = 0) Then
        List1.AddItem Query(0)
        List1.AddItem Query(1)
        List1.AddItem Query(2)
    End If
    Text1.Text = "END"
    StartFlag = 0
End Sub
```

```
Private Sub Command4_Click()
    Dim sts As Integer

    If (StartFlag = 1) Then
        Exit Sub
    End If
    StartFlag = 1
    Text1.Text = "START"
    List1.Clear
    sts = RS232Tdiv                                'Run Sample4(RS232) Set/Get TDIV
    If (sts = 0) Then
        List1.AddItem Query(0)
    End If
    Text1.Text = "END"
    StartFlag = 0
End Sub
```

```
Private Sub Command5_Click()
    Dim sts As Integer

    If (StartFlag = 1) Then
        Exit Sub
    End If
    StartFlag = 1
    Text1.Text = "START"
    List1.Clear
    sts = RS232GetWord                             'Run Sample5(RS232) Get Word Data
    If (sts = 0) Then
        List1.AddItem "END"
    End If
    Text1.Text = "END"
    StartFlag = 0
End Sub
```

```
Private Sub Form_Load()

    StartFlag = 0                                  'Clear Start Flag
    Dev = -1                                       'Clear device id
    Addr = 1                                       'GPIB Address = 1
    Timer1.Interval = 0
    Command1.Caption = "Sample1(GPIB)" + Chr(13) + "Set/Get TDIV"
    Command2.Caption = "Sample2(GPIB)" + Chr(13) + "Get Word Data"
    Command3.Caption = "Sample3(GPIB)" + Chr(13) + "Get Measure"
    Command4.Caption = "Sample4(RS232)" + Chr(13) + "Set/Get TDIV"
    Command5.Caption = "Sample5(RS232)" + Chr(13) + "Get Word Data"
    Text1.Text = ""

End Sub
```

```
Private Sub MSComm1_OnComm()
    Dim evt As Integer

    evt = MSComm1.CommEvent
    Select Case evt
    'Error
        Case comBreak
            Call DIsplayRS232Error("comBreak:Break received")
        Case comCDTO
            Call DIsplayRS232Error("comCDTO CD(RLSD) timeout")
        Case comCTSTO
            Call DIsplayRS232Error("comCTSTO:CTS timeout")
        Case comDSRTO
            Call DIsplayRS232Error("commDSRTO:DSR timeout")
        Case comFrame
            Call DIsplayRS232Error("comFrame:Frame error")
        Case comOverrun
            Call DIsplayRS232Error("comOverrun:Overrun")
        Case comRxOver
            Call DIsplayRS232Error("comRxOver:Receive buffer overflow")
        Case comRxParity
            Call DIsplayRS232Error("commRxParity:Parity error")
        Case comTxFull
            Call DIsplayRS232Error("comTxFull:Send buffer overflow")
    'Event
        Case comEvReceive
        Case comEvCD
        Case comEvCTS
        Case comEvDSR
        Case comEvRing
        Case comEvSend
    End Select

End Sub
```

```
Private Sub Timer1_Timer()
    TimerCount = TimerCount + 1
End Sub
```

## 7.4 Sets/Queries the T/Div

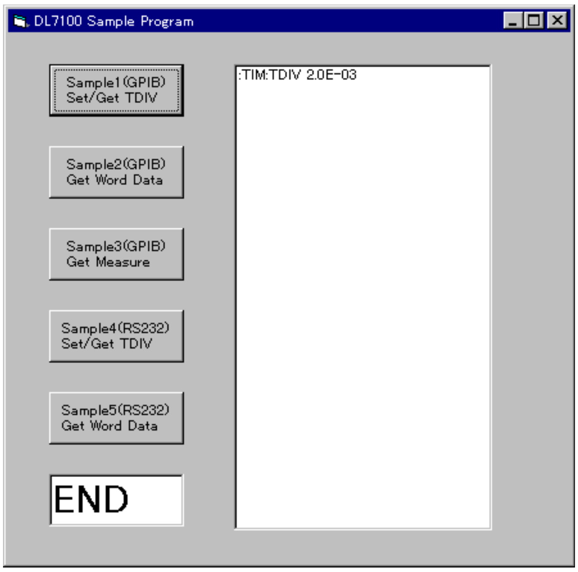
Sample1(GPIB) Set/Get TDIV

```
Private Function GpibTdiv() As Integer
    Dim msg As String                              'Command buffer
    Dim qry As String                              'Query biffer
    Dim sts As Integer

    msg = Space$(100)
    qry = Space$(100)

    sts = InitGpib                                 'Initialize GPIB
    If (sts <> 0) Then
        GpibTdiv = 1
        Exit Function
    End If

    msg = "TIMEBASE:TDIV 2ms" + Term               'Set T/div = 2ms
    sts = ilwrt(Dev, msg, Len(msg))                'Send Command
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibTdiv = 1
        Exit Function
    End If
    msg = "TIMEBASE:TDIV?" + Term                  'Get T/div value
    sts = ilwrt(Dev, msg, Len(msg))                'Send Command
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibTdiv = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, Len(qry))                 'Receive Query
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibTdiv = 1
        Exit Function
    End If
    Query(0) = Left(qry, ibcntl - 1)
    Call ibonl(Dev, 0)
    GpibTdiv = 0
End Function
```

Sample4(RS232) Set/Get TDIV

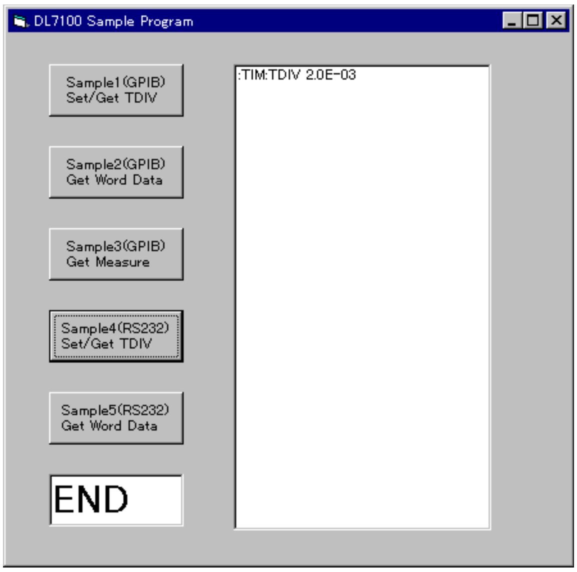
```
Private Function RS232Tdiv() As Integer
    Dim msg As String                              'Command buffer
    Dim qry As String                              'Query biffer
    Dim sts As Integer

    msg = Space$(100)
    qry = CStr(Empty)

    sts = InitSerial                               'Initialize RS232
    If (sts <> 0) Then
        Exit Function
    End If

    MSComm1.InputLen = 0                           'Receive All Data
    MSComm1.InputMode = comInputModeText           'Text Mode
    MSComm1.PortOpen = True                        'Port Open
    MSComm1.OutBufferCount = 0                     'Out Buffer Clear
    MSComm1.InBufferCount = 0                      'In Buffer Clear
    Timer1.Interval = 1000

    If CtsFlag = 1 Then                            'If CTS = FALSE
        TimerCount = 1                             'Wait until CTS = TRUE
        Do
            Dummy = DoEvents()
            If (TimerCount >= Timeout) Then
                Call DIsplayRS232Error("CTS Timeout")
                RS232Tdiv = 1
                GoTo finish
            End If
        Loop Until MSComm1.CTSHolding = True
    End If

    msg = "TIMEBASE:TDIV 2ms" + Term               'Set T/div = 2ms
    MSComm1.Output = msg                           'Send Command

    TimerCount = 1
    Do                                             'Wait until OutBufferCount = 0
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232Tdiv = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "TIMEBASE:TDIV?" + Term                  'Get T/div value
    MSComm1.Output = msg                           'Send Command

    TimerCount = 1
    Do                                             'Wait until OutBufferCount = 0
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232Tdiv = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    TimerCount = 1
    Do                                             'Receive Query
        qry = qry + MSComm1.Input                  'Wait until End Data = Terminator
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Receive Timeout", msg)
            RS232Tdiv = 1
            GoTo finish
        End If
    Loop Until Right$(qry, 1) = Term

    Query(0) = Left$(qry, Len(qry) - 1)
    RS232Tdiv = 0
finish:
    MSComm1.PortOpen = False                       'Port Close
    Timer1.Interval = 0

End Function
```

GP-IB


DL7100 Sample Program
Sample1(GPIB) Set/Get TDIV
Sample2(GPIB) Get Word Data
Sample3(GPIB) Get Measure
Sample4(RS232) Set/Get TDIV
Sample5(RS232) Get Word Data
END

:TIM:TDIV 2.0E-03

RS-232



DL7100 Sample Program
Sample1(GPIB) Set/Get TDIV
Sample2(GPIB) Get Word Data
Sample3(GPIB) Get Measure
Sample4(RS232) Set/Get TDIV
Sample5(RS232) Get Word Data
END

:TIM:TDIV 2.0E-03

# 7.5 Data Output in Word Format

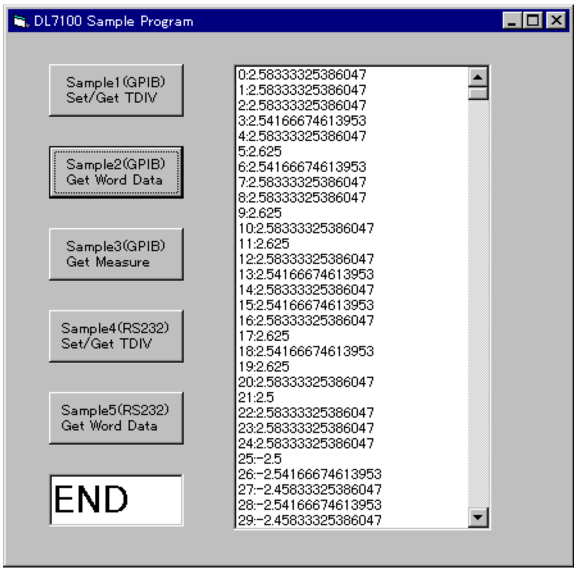
Sample2(GPIB) Get Word Data

```
Private Function GpibGetWord() As Integer
    Dim msg As String                               'Command buffer
    Dim qry As String                               'Query biffer
    Dim sts As Integer
    Dim vdv As Variant                              'Vdiv value
    Dim ofs As Variant                              'Offset value
    Dim eos As Integer                              'EOS
    Dim hlg As Integer                              'Block Header Length
    Dim dlg As Integer                              'Block Data Length
    Dim dat As Variant                              'Data
    Dim i   As Integer

    msg = Space$(100)
    qry = Space$(100)

    sts = InitGpib                                  'Initialize GPIB
    If (sts <> 0) Then
        GpibGetWord = 1
        Exit Function
    End If

    msg = "STOP" + Term                             'Stop Acquisition
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "COMMUNICATE:HEADER OFF" + Term           'Query Header Off(for Get V/div)
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:TRACE 1" + Term                 'Trace = 1
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:RECORD 0" + Term                'Record number = 0
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:FORMAT WORD" + Term             'Data Format = WORD
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:BYTEORDER LSBFIRST" + Term      'Data Byte order = LSB First(for
Little Endian)
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:START 0;END 1001" + Term        'START 0,END 1001(Length = 1002)
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:RANGE?" + Term                  'Get V/div value
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, Len(qry))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    vdv = Val(qry)
    msg = "WAVEFORM:OFFSET?" + Term                  'Get Offset value
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, Len(qry))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    ofs = Val(qry)

    eos = 0
    sts = ileos(Dev, eos)                         'Terminator = None(for Binary Data)
    If (sts < 0) Then
        Call DisplayGPIBError(sts, "ileos")
        GpibGetWord = 1
        Exit Function
    End If
    msg = "WAVEFORM:SEND?" + Term                 'Receive Waveform Data
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, 2)                       'Receive "#X"
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    hlg = Val(Mid$(qry, 2, 1))
    sts = ilrd(Dev, qry, hlg)                     'Receive Block Header
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    dlg = Val(Left$(qry, hlg))                    'dlg = Data Byte Length

    sts = ilrdi(Dev, WaveBuffer(), dlg + 1)       'Receive Waveform Data + LF
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If

    For i = 0 To (dlg / 2 - 1) Step 1
        dat = WaveBuffer(i) * vdv / 3072 + ofs
        List1.AddItem CStr(i) + ":" + CStr(dat)
    Next i

    eos = &HC0A
    sts = ileos(Dev, eos)                         'Terminator = LF
    If (sts < 0) Then
        Call DisplayGPIBError(sts, "ileos")
        GpibGetWord = 1
        Exit Function
    End If
    msg = "COMMUNICATE:HEADER ON" + Term          'Query Header On
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetWord = 1
        Exit Function
    End If
    Call ibonl(Dev, 0)
    GpibGetWord = 0
End Function
```

Sample5(RS232) Get Word Data

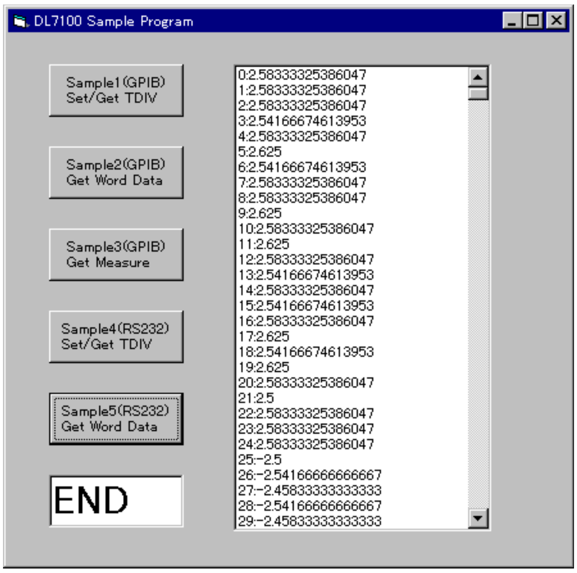
```
Private Function RS232GetWord() As Integer
    Dim msg As String                              'Command buffer
    Dim qry As String                              'Query biffer
    Dim sts As Integer
    Dim vdv As Variant                             'V/div value
    Dim ofs As Variant                             'Offset value
    Dim hlg As Integer                             'Block Header Length
    Dim dlg As Integer                             'Blocl Data Length
    Dim buf As Variant                             'temporary buffer
    Dim dat As Variant                             'data buffer
    Dim i   As Integer

    msg = Space$(100)
    qry = CStr(Empty)

    sts = InitSerial                               'Initialize RS232
    If (sts <> 0) Then
        Exit Function
    End If

    MSComm1.InputLen = 0                           'Receive All Data
    MSComm1.InputMode = comInputModeText           'Text Mode
    MSComm1.PortOpen = True                        'Port Open
    MSComm1.OutBufferCount = 0                     'Out Buffer Clear
    MSComm1.InBufferCount = 0                      'In Buffer Clear
    Timer1.Interval = 1000

    If CtsFlag = 1 Then                            'If CTS = FALSE
        TimerCount = 1                             'Wait unitl CTS = TRUE
        Do
            Dummy = DoEvents()
            If (TimerCount >= Timeout) Then
                Call DIsplayRS232Error("CTS Timeout")
                RS232GetWord = 1
                GoTo finish
            End If
        Loop Until MSComm1.CTSHolding = True
    End If

    msg = "STOP" + Term                            'Stop Acquisition
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "COMMUNICATE:HEADER OFF" + Term          'Query Header Off(for Get V/div)
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "WAVEFORM:TRACE 1" + Term                'Trace = 1
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "WAVEFORM:RECORD 0" + Term               'Record number = 0
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "WAVEFORM:FORMAT WORD" + Term            'Data Format = WORD
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "WAVEFORM:BYTEORDER LSBFIRST" + Term     'Data Byte order = LSB First(for
Little Endian)
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    msg = "WAVEFORM:START 0;END 1001" + Term       'START 0,END 1001(Length = 1002)
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    qry = CStr(Empty)
    msg = "WAVEFORM:RANGE?" + Term                  'Get V/div value
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0
    TimerCount = 1
    Do
        qry = qry + MSComm1.Input
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Receive Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until Right$(qry, 1) = Term
    vdv = Val(qry)

    qry = CStr(Empty)
    msg = "WAVEFORM:OFFSET?" + Term                  'Get Offset value
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0
    TimerCount = 1
    Do
        qry = qry + MSComm1.Input
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Receive Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until Right$(qry, 1) = Term
    ofs = Val(qry)

    msg = "WAVEFORM:SEND?" + Term                  'Receive Waveform Data
    MSComm1.Output = msg

    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    MSComm1.InputLen = 2                           'Receive "#X"
    TimerCount = 1
    Do Until MSComm1.InBufferCount >= 1
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Receive Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop
    qry = MSComm1.Input
    hlg = Val(Mid$(qry, 2, 1))

    MSComm1.InputLen = hlg                         'Receive Block Header
    TimerCount = 1
    Do Until MSComm1.InBufferCount >= hlg
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Receive Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop
    qry = MSComm1.Input
    dlg = Val(Left$(qry, hlg))                     'leng% = Data Byte Length

    MSComm1.InputMode = comInputModeBinary
    MSComm1.InputLen = 2                           'Receive Waveform Data(2 Byte)

    For i = 0 To (dlg / 2 - 1) Step 1              'Loop(dlg)
        TimerCount = 1
        Do Until MSComm1.InBufferCount >= 2
            Dummy = DoEvents()
            If (TimerCount >= Timeout) Then
                Call DIsplayRS232Error("Receive Timeout", msg)
                RS232GetWord = 1
                GoTo finish
            End If
        Loop
        buf = MSComm1.Input                        'Receive 1 Data(2 Byte)
        dat = buf(1) * 256 + buf(0)
        If (dat > 32767) Then
            dat = dat - 65536
        End If
        dat = dat * vdv / 3072 + ofs
        List1.AddItem CStr(i) + ":" + CStr(dat)
    Next i

    msg = "COMMUNICATE:HEADER ON" + Term           'Query Header On
    MSComm1.Output = msg
    TimerCount = 1
    Do
        Dummy = DoEvents()
        If (TimerCount >= Timeout) Then
            Call DIsplayRS232Error("Send Timeout", msg)
            RS232GetWord = 1
            GoTo finish
        End If
    Loop Until MSComm1.OutBufferCount = 0

    RS232GetWord = 0
finish:
    MSComm1.PortOpen = False                       'Port Close
    Timer1.Interval = 0

End Function
```

GP-IB


DL7100 Sample Program
Sample1(GPIB) Set/Get TDIV
Sample2(GPIB) Get Word Data
Sample3(GPIB) Get Measure
Sample4(RS232) Set/Get TDIV
Sample5(RS232) Get Word Data
END

```
0:2.58333325386047
1:2.58333325386047
2:2.58333325386047
3:2.54166674613953
4:2.58333325386047
5:2.625
6:2.54166674613953
7:2.58333325386047
8:2.58333325386047
9:2.625
10:2.58333325386047
11:2.625
12:2.58333325386047
13:2.54166674613953
14:2.58333325386047
15:2.54166674613953
16:2.58333325386047
17:2.625
18:2.54166674613953
19:2.625
20:2.58333325386047
21:2.5
22:2.58333325386047
23:2.58333325386047
24:2.58333325386047
25:-2.5
26:-2.54166674613953
27:-2.45833325386047
28:-2.54166674613953
29:-2.45833325386047
```

RS-232


DL7100 Sample Program
Sample1(GPIB) Set/Get TDIV
Sample2(GPIB) Get Word Data
Sample3(GPIB) Get Measure
Sample4(RS232) Set/Get TDIV
Sample5(RS232) Get Word Data
END

```
0:2.58333325386047
1:2.58333325386047
2:2.58333325386047
3:2.54166674613953
4:2.58333325386047
5:2.625
6:2.54166674613953
7:2.58333325386047
8:2.58333325386047
9:2.625
10:2.58333325386047
11:2.625
12:2.58333325386047
13:2.54166674613953
14:2.58333325386047
15:2.54166674613953
16:2.58333325386047
17:2.625
18:2.54166674613953
19:2.625
20:2.58333325386047
21:2.5
22:2.58333325386047
23:2.58333325386047
24:2.58333325386047
25:-2.5
26:-2.54166666666667
27:-2.45833333333333
28:-2.54166666666667
29:-2.45833333333333
```

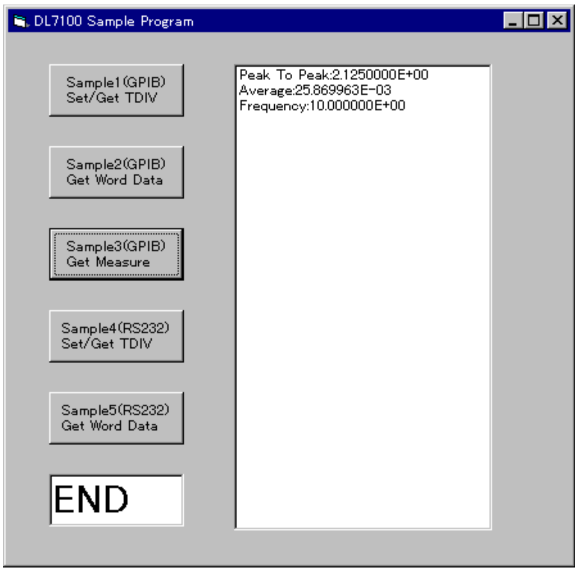
# 7.6 Sets/Queries Measure Value

Sample3(GPIB) Get Measure value

```
Private Function GpibGetMeasure() As Integer
    Dim msg As String                              'Command buffer
    Dim qry As String                              'Query biffer
    Dim sts As Integer

    msg = Space$(100)
    qry = Space$(100)

    sts = InitGpib                                 'Initialize GPIB
    If (sts <> 0) Then
        GpibGetMeasure = 1
        Exit Function
    End If

    msg = "STOP" + Term                            'Acquisition = Stop
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "COMMUNICATE:HEADER OFF" + Term          'Query Header Off(for Get V/div)
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "MEASURE:MODE OFF" + Term                'Measure Off
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "TRIGGER:MODE SINGLE;:STOP" + Term       'Trigger Mode = Single(S),
Acquisition = Stop
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "STATUS:FILTER1 FALL" + Term             'Filter1 Fall(Start->Stop)
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "CHANNEL:DISPLAY ON" + Term              'CH1 On
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "CHANNEL:PROBE 10" + Term                'CH1 Probe = 10:1
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "CHANNEL:VDIV 500mV" + Term              'CH1 V/div = 500mV
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "ACQUIRE:MODE NORMAL;RLENGTH 1000" + Term
                                                   'Acquisition mode = NORMAL, length = 1000
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    msg = "TIMEBASE:TDIV 100ms" + Term             'T/div = 100ms
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
```

```
msg = "TRIGGER:SIMPLE:LEVEL 500mV" + Term      'Trigger level = 500mV
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "MEASURE:CHANNEL1:PTOPEAK:STATE ON" + Term
                                               'Measure P-P On
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "MEASURE:CHANNEL1:AVERAGE:STATE ON" + Term
                                               'Measure Average On
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "MEASURE:CHANNEL1:FREQUENCY:STATE ON" + Term
                                               'Measure Frequency On
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "MEASURE:TRANGE -5,5" + Term             'Measure Time Range -5,5
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "STATUS:EESR?" + Term                    'Clear Extended Event Register
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
sts = ilrd(Dev, qry, Len(qry))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If

msg = "START" + Term                           'Start Acquisition(Single)
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "COMMUNICATE:WAIT? 1" + Term             'Wait until stop Acquisition
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
sts = ilrd(Dev, qry, Len(qry))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If

msg = "MEASURE:MODE ON" + Term                 'Start Measure
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
msg = "MEASURE:WAIT? 100" + Term               'Wait until stop Measure
sts = ilwrt(Dev, msg, Len(msg))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
    Exit Function
End If
sts = ilrd(Dev, qry, Len(qry))
If (sts < 0) Then
    Call DisplayGPIBError(sts, msg)
    GpibGetMeasure = 1
```

```
        Exit Function
    End If

    msg = "MEASURE:CHANNEL1:PTOPEAK:VALUE?" + Term 'Get P-P value
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, Len(qry))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    Query(0) = "Peak To Peak:" + Left$(qry, ibcntl - 1)

    msg = "MEASURE:CHANNEL1:AVERAGE:VALUE?" + Term 'Get Average value
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, Len(qry))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    Query(1) = "Average:" + Left$(qry, ibcntl - 1)

    msg = "MEASURE:CHANNEL1:FREQUENCY:VALUE?" + Term 'Get Freq value
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    sts = ilrd(Dev, qry, Len(qry))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    Query(2) = "Frequency:" + Left$(qry, ibcntl - 1)

    msg = "COMMUNICATE:HEADER ON" + Term           'Query Header On
    sts = ilwrt(Dev, msg, Len(msg))
    If (sts < 0) Then
        Call DisplayGPIBError(sts, msg)
        GpibGetMeasure = 1
        Exit Function
    End If
    Call ibonl(Dev, 0)
    GpibGetMeasure = 0
End Function
```


DL7100 Sample Program
Sample1(GPIB) Set/Get TDIV
Sample2(GPIB) Get Word Data
Sample3(GPIB) Get Measure
Sample4(RS232) Set/Get TDIV
Sample5(RS232) Get Word Data
END

```
Peak To Peak:2.1250000E+00
Average:25.869963E-03
Frequency:10.000000E+00
```

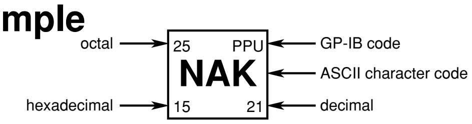
# Appendix

## Appendix 1 ASCII Character Code

ASCII character codes are given below.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0<br>NUL<br>0 0 | 20<br>DEL<br>10 16 | 40 0<br>SP<br>20 32 | 60 16<br>0<br>30 48 | 100 0<br>@<br>40 64 | 120 16<br>P<br>50 80 | 140 0<br>‘<br>60 96 | 160 16<br>p<br>70 112 |
| 1 | 1 GTL<br>SOH<br>1 1 | 21 LLO<br>DC1<br>11 17 | 41 1<br>!<br>21 33 | 61 17<br>1<br>31 49 | 101 1<br>A<br>41 65 | 121 17<br>Q<br>51 81 | 141 1<br>a<br>61 97 | 161 17<br>q<br>71 113 |
| 2 | 2<br>STX<br>2 2 | 22<br>DC2<br>12 18 | 42 2<br>”<br>22 34 | 62 18<br>2<br>32 50 | 102 2<br>B<br>42 66 | 122 18<br>R<br>52 82 | 142 2<br>b<br>62 98 | 162 18<br>r<br>72 114 |
| 3 | 3<br>ETX<br>3 3 | 23<br>DC3<br>13 19 | 43 3<br>#<br>23 35 | 63 19<br>3<br>33 51 | 103 3<br>C<br>43 67 | 123 19<br>S<br>53 83 | 143 3<br>c<br>63 99 | 163 19<br>s<br>73 115 |
| 4 | 4 SDC<br>EOT<br>4 4 | 24 DCL<br>DC4<br>14 20 | 44 4<br>$<br>24 36 | 64 20<br>4<br>34 52 | 104 4<br>D<br>44 68 | 124 20<br>T<br>54 84 | 144 4<br>d<br>64 100 | 164 20<br>t<br>74 116 |
| 5 | 5 PPC<br>ENQ<br>5 5 | 25 PPU<br>NAK<br>15 21 | 45 5<br>%<br>25 37 | 65 21<br>5<br>35 53 | 105 5<br>E<br>45 69 | 125 21<br>U<br>55 85 | 145 5<br>e<br>65 101 | 165 21<br>u<br>75 117 |
| 6 | 6<br>ACK<br>6 6 | 26<br>SYN<br>16 22 | 46 6<br>&<br>26 38 | 66 22<br>6<br>36 54 | 106 6<br>F<br>46 70 | 126 22<br>V<br>56 86 | 146 6<br>f<br>66 102 | 166 22<br>v<br>76 118 |
| 7 | 7<br>BEL<br>7 7 | 27<br>ETB<br>17 23 | 47 7<br>’<br>27 39 | 67 23<br>7<br>37 55 | 107 7<br>G<br>47 71 | 127 23<br>W<br>57 87 | 147 7<br>g<br>67 103 | 167 23<br>w<br>77 119 |
| 8 | 10 GET<br>BS<br>8 8 | 30 SPE<br>CAN<br>18 24 | 50 8<br>(<br>28 40 | 70 24<br>8<br>38 56 | 110 8<br>H<br>48 72 | 130 24<br>X<br>58 88 | 150 8<br>h<br>68 104 | 170 24<br>x<br>78 120 |
| 9 | 11 TCT<br>HT<br>9 9 | 31 SPD<br>EM<br>19 25 | 51 9<br>)<br>29 41 | 71 25<br>9<br>39 57 | 111 9<br>I<br>49 73 | 131 25<br>Y<br>59 89 | 151 9<br>i<br>69 105 | 171 25<br>y<br>79 121 |
| A | 12<br>LF<br>A 10 | 32<br>SUB<br>1A 26 | 52 10<br>*<br>2A 42 | 72 26<br>:<br>3A 58 | 112 10<br>J<br>4A 74 | 132 26<br>Z<br>5A 90 | 152 10<br>j<br>6A 106 | 172 26<br>z<br>7A 122 |
| B | 13<br>VT<br>B 11 | 33<br>ESC<br>1B 27 | 53 11<br>+<br>2B 43 | 73 27<br>;<br>3B 59 | 113 11<br>K<br>4B 75 | 133 27<br>[<br>5B 91 | 153 11<br>k<br>6B 107 | 173 27<br>{<br>7B 123 |
| C | 14<br>FF<br>C 12 | 34<br>FS<br>1C 28 | 54 12<br>,<br>2C 44 | 74 28<br><<br>3C 60 | 114 12<br>L<br>4C 76 | 134 28<br>\<br>5C 92 | 154 12<br>l<br>6C 108 | 174 28<br>\|<br>7C 124 |
| D | 15<br>CR<br>D 13 | 35<br>GS<br>1D 29 | 55 13<br>-<br>2D 45 | 75 29<br>=<br>3D 61 | 115 13<br>M<br>4D 77 | 135 29<br>]<br>5D 93 | 155 13<br>m<br>6D 109 | 175 29<br>}<br>7D 125 |
| E | 16<br>SO<br>E 14 | 36<br>RS<br>1E 30 | 56 14<br>.<br>2E 46 | 76 30<br>><br>3E 62 | 116 14<br>N<br>4E 78 | 136 30<br>^<br>5E 94 | 156 14<br>n<br>6E 110 | 176 30<br>~<br>7E 126 |
| F | 17<br>SI<br>F 15 | 37<br>US<br>1F 31 | 57 15<br>/<br>2F 47 | 77 UNL<br>?<br>3F 63 | 117 15<br>O<br>4F 79 | 137 UNT<br>_<br>5F 95 | 157 15<br>o<br>6F 111 | 177<br>DEL (RUBOUT)<br>7F 127 |
| | Address Command | Universal Command | Listener Address | | Talker Address | | Secondary Command | |

## Example



# Appendix 2 Error Messages

Error messages related to communications are given below.

- The instrument allows error messages to be displayed in either Japanese or English, however, they are shown only in English when they are displayed on a personal computer.
- When servicing is required, contact your nearest YOKOGAWA representative, given on the back cover of this manual.
- Only error messages relating to communications are given. For other error messages, refer to the User's Manual IM 701410-01E.

## Errors in communication command (100 to 199)

| Code | Message | Action | Reference Page |
|---|---|---|---|
| 102 | Syntax error | Incorrect syntax. | Chapter 4 , 5 |
| 103 | Invalid separator | Insert a comma between data items to separate them. | 4-1 |
| 104 | Data type error | Refer to pages 4-5 to 4-6 and enter using the correct data format. | 4-5 to 4-6 |
| 105 | GET not allowed | GET is not supported as response to an interface message. | 1-9 |
| 108 | Parameter not allowed | Check the number of parameters. | 4-5, Chapter 5 |
| 109 | Missing parameter | Enter required parameters. | 4-5, Chapter 5 |
| 111 | Header separator error | Insert a space between header and data to separate them. | 4-1 |
| 112 | Program mnemonic too long | Check the mnemonic (a character string consisting of letters and numbers). | Chapter 5 |
| 113 | Undefined header | Check the header. | Chapter 5 |
| 114 | Header suffix out of range | Check the header. | Chapter 5 |
| 120 | Numeric data error | Numeric value must be entered for <NRf> format. | 4-5 |
| 123 | Exponent too large | Use a smaller exponent for <NR3> format. | 4-5, Chapter 5 |
| 124 | Too many digits | Limit the number of digits to 255 or less. | 4-5, Chapter 5 |
| 128 | Numeric data not allowed | Enter in a format other than <NRf> format. | 4-5, Chapter 5 |
| 131 | Invalid suffix | Check the unit for <Voltage>, <Time> and <Frequency>. | 4-5 |
| 134 | Suffix too long | Check the units for <Voltage>, <Time> and <Frequency>. | 4-5 |
| 138 | Suffix not allowed | No units are allowed other than <Voltage>, <Time> and <Frequency>. | 4-5 |
| 141 | Invalid character data | Enter one of the character strings in {...\|...\|...}. | Chapter 5 |
| 144 | Character data too long | Check the character strings in {...\|...\|...}. | Chapter 5 |
| 148 | Character data not allowed | Enter in a format other than in {...\|...\|...}. | Chapter 5 |
| 150 | String data error | <Character string> must be enclosed by double quotation marks or single quotation marks. | 4-6 |
| 151 | Invalid string data | <Character string> is too long or contains characters which cannot be used. | Chapter 5 |
| 158 | String data not allowed | Enter in a data format other than <Character string>. | Chapter 5 |
| 161 | Invalid block data | <Block data> is not allowed. | 4-6, Chapter 5 |
| 168 | Block data not allowed | <Block data> is not allowed. | 4-6, Chapter 5 |
| 171 | Invalid expression | Equation is not allowed. | Chapter 5 |
| 178 | Expression data not allowed | Equation is not allowed. | Chapter 5 |
| 181 | Invalid outside macro definition | Does not conform to the macro function specified in IEEE488.2. | — |

## Error in communications execution (200 to 299)

| Code | Message | Action | Reference Page |
|---|---|---|---|
| 221 | Setting conflict | Check the relevant setting. | Chapter 5 |
| 222 | Data out of range | Check the setting range. | Chapter 5 |
| 223 | Too much data | Check the data byte length. | Chapter 5 |
| 224 | Illegal parameter value | Check the setting range. | Chapter 5 |
| 241 | Hardware missing | Check availability of options. | — |
| 260 | Expression error | Equation is not allowed. | — |
| 270 | Macro error | Does not conform to the macro function specified in IEEE488.2. | — |
| 272 | Macro execution error | Does not conform to the macro function specified in IEEE488.2. | — |
| 273 | Illegal macro label | Does not conform to the macro function specified in IEEE488.2. | — |
| 275 | Macro definition too long | Does not conform to the macro function specified in IEEE488.2. | — |
| 276 | Macro recursion error | Does not conform to the macro function specified in IEEE488.2. | — |
| 277 | Macro redefinition not allowed | Does not conform to the macro function specified in IEEE488.2. | — |
| 278 | Macro header not found | Does not conform to the macro function specified in IEEE488.2. | — |

## Error in communications Query (400 to 499)

| Code | Message | Action | Reference Page |
|---|---|---|---|
| 410 | Query INTERRUPTED | Check transmission/reception order. | 4-2 |
| 420 | Query UNTERMINATED | Check transmission/reception order. | 4-2 |
| 430 | Query DEADLOCKED | Limit the length of the program message including <PMT> to 1024 bytes or less. | 4-2 |
| 440 | Query UNTERMINATED after | Do not enter any query after *IDN? and *OPT?.<br>indefinite response | — |

## Error in System Operation (912 to 914)

| Code | Message | Action | Reference Page |
|---|---|---|---|
| 912 | Fatal error in Communication-driver | Servicing is required. | — |
| 914 | Time out occurs in Communication | Receive data within time-out time. The communications line may be faulty. | — |
| 915 | Can't detect listener | Check GP-IB connector<br>Place the plotter or AG in listen-only mode then connect it. | — |

## Warning

| Code | Message | Action | Reference Page |
|---|---|---|---|
| 5 | *OPC/? exists in message | Place the *OPC or *OPC? at the end of the program message. | — |

## Other errors (350 and 390)

| Code | Message | Action | Reference Page |
|---|---|---|---|
| 350 | Queue overflow | Read the error queue. Code 350 occurs when the error queue is full up. This message is output only for the STATus:ERRor? query and is not displayed on the screen. | 6-5 |
| 390 | Overrun error (only RS-232) | Execute with a lower baud rate. | — |

***Note***

Code 350 indicates overflow of error queue. This code is returned as a response to the "STATus:ERRor?" query; it does not appear on the screen.

# Appendix 3 Overview of IEEE 488.2-1992

The GP-IB interface provided with DL7100/DL7200 conforms to IEEE 488.2-1992. This standard requires the following 23 points be stated in this document. This Appendix describes these points.

1 **Subsets supported by IEEE 488.1 interface functions**
Refer to Section 1.4 "GP-IB Interface Specifications".

2 **Operation of device when the device is assigned to an address other than addresses 0 to 30.**
The DL7100 does not allow assignment to an address other than 0 to 30.

3 **Reaction when the user changes the address**
The current address is changed when a new address is set using the MISC key. The newly set address is valid until another new address is set.

4 **Device set-up at power ON. Commands which can be used at power ON**
Basically, the previous settings (i.e. the settings which were valid when power was turned OFF) are valid. All commands are available at power ON.

5 **Message transmission options**

a **Input buffer size**
1024 bytes

b **Queries which return multiple response messages**
Refer to Chapter 5, "Command List".

c **Queries which generate response data during analysis of the syntax**
Every query generates a response data when analysis of the syntax is completed.

d **Queries which generate response data during reception**
No query generates response data when the query is received by the controller.

e **Commands consisting of parameters which restrict one other**
Some commands, like the `CHANnel<x>:VOLTage:PROBe` and `VDIV`, have parameters which restrict unilaterally, but no commands have parameters which restrict bilaterally.

6 **Options included in command function elements and composite header elements**
Refer to Chapters 4 and 5.

7 **Buffer size which affects transmission of block data**
During transmission of block data, the output queue is extended according to the size of the data blocks.

8 **List of program data elements which can be used in equations, and nesting limit**
No equations can be used.

9 **Syntax of response to queries**
Refer to the description of the commands given in Chapter 5.

10 **Communications between devices which do not follow the response syntax**
Not supported.

**11 Size of data block of response data**
1 to 16032000 bytes

**12 List of supported common commands**
Refer to Section 5.33 "Common Command Group".

**13 Condition of device when calibration is successfully completed**
Same as the one under which measurements are performed

**14 Maximum length of block data which can be used for definition of** `*DDT` **trigger macro**
Not supported

**15 Maximum length of macro label used in definition of macro, maximum length of block data which can be used for definition of macro, processing when recursion is used in definition of macro**
Macro functions are not supported.

**16 Response to** `*IDN?`
Refer to Section 5.33 "Common Command Group".

**17 Size of storage area for protected user data for PUD and** `*PUD?`
`*PUD` and `*PUD?` are not supported.

**18 Length of** `*RDT` **and** `*RDT?` **resource name**
`*RDT` and `*RDT?` are not supported.

**19 Change in status due to** `*RST`**,** `*LRN?`**,** `*RCL` **and** `*SAV`
`*RST`, `*LRN?`
Refer to Section 5.33 "Common Command Group".
`*RCL`, `*SAV`
These commands are not supported.

**20 Execution range of self-test using the** `*TST?`
All the memory tests (for each internal memory) given in the Self Test menu displayed using the MISC key can be executed.

**21 Structure of extended return status**
Refer to Chapter 6.

**22 To find out whether each command is performed in parallel or sequentially**
Refer to Section 4.5 "Synchronization with the Controller" and to Chapter 5.

**23 Description of execution of each command**
Refer to Chapter 5 of this manual and to the User's Manual IM 701410-01E.

# Index