Thank you for purchasing the SL1000 API Control. This Communication Interface User’s Manual describes the functions and commands of the following communication interfaces.

- USB Interface
- Ethernet Interface (Optional)

To ensure correct use, please read this manual thoroughly before beginning operation. After reading the manual, keep it in a convenient location for quick reference whenever a question arises during operation.

The following manuals are provided for the SL1000. Please read all of them.

<table>
<thead>
<tr>
<th>Manual Title</th>
<th>Manual No.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SL1000 High-Speed Data Acquisition Unit User’s Manual</td>
<td>IM 720120-01E</td>
<td>Explains how to install the SL1000 and its input modules, and explains features related to the hardware, such as the display, and how to operate them.</td>
</tr>
<tr>
<td>SL1000 Acquisition Software User’s Manual</td>
<td>IM 720120-61E</td>
<td>Explains all functions and procedures of the Acquisition Software used to configure and control the SL1000.</td>
</tr>
<tr>
<td>SL1000 Input Module User’s Manual</td>
<td>IM 720120-51E</td>
<td>Explains the specifications of the input modules that can be installed in the SL1000.</td>
</tr>
<tr>
<td>701992/701994 Xviewer User’s Manual</td>
<td>IM 701992-01E</td>
<td>Explains all functions and procedures of the Xviewer software used to display the measured data as waveforms on a PC. This manual is not included with the /XV0 option.</td>
</tr>
<tr>
<td>SL1000 Control API User’s Manual</td>
<td>IM 720320-01E</td>
<td>It explains the functions for controlling the SL1000 (the SL1000 control API).</td>
</tr>
<tr>
<td>SL1000 High-Speed Data Acquisition Unit Communication Interface User’s Manual</td>
<td>IM 720320-17E</td>
<td>This manual. Explains the communication interface functions of the SL1000</td>
</tr>
</tbody>
</table>

Notes

- This manual, IM 720320-17E, applies to SL1000 High-Speed Data Acquisition Unit with firmware version 2.03 or later.
  If the most recent firmware version is not running on your SL1000 not all of the features described in this manual can be used.
  You can check the firmware version of your SL1000 on the overview screen. For instructions on how to open the overview screen, see section 9.5 in the User’s Manual IM 720120-61E. For instructions on how to update the firmware and for information about firmware versions, see the following Webpage.
  http://www.yokogawa.com/tm/
- The contents of this manual are subject to change without prior notice as a result of continuing improvements to the instrument’s performance and functions.
- Every effort has been made in the preparation of this manual to ensure the accuracy of its contents. However, should you have any questions or find any errors, please contact your nearest YOKOGAWA dealer.
- Copying or reproducing all or any part of the contents of this manual without the permission of Yokogawa Electric Corporation is strictly prohibited.
Trademarks

• Microsoft, MS-DOS, Visual C++, Windows, Windows XP, and Windows NT are registered trademarks of Microsoft Corporation in the United States and/or other countries.
• Adobe and Acrobat are trademarks of Adobe Systems Incorporated.
• For purposes of this manual, the TM and ® symbols do not accompany their respective trademark names or registered trademark names.
• Other company and product names are trademarks or registered trademarks of their respective holders.

Revisions

1st Edition: December 2008
2nd Edition: December 2009
3rd Edition: September 2013
4th Edition: July 2017
How to Use This Manual

Structure of This Manual
This user’s manual consists of the following sections.

Chapter 1 Connecting to the PC
Describes the procedure for connecting to the PC using the USB and Ethernet interfaces.

Chapter 2 Before Programming
Describes the syntax used to transmit commands.

Chapter 3 Command
Describes all the commands one by one.

Chapter 4 Status Reports
Describes the status byte, various registers, and queues.

Appendix
Describes reference material such as an ASCII character code table.
Symbols and Notations Used in This Manual

Safety Markings
The following markings are used in this manual.

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

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

Notation Used in the Procedural Explanations
On pages that describe the operating procedures in chapters 1 through 3, the following notations are used to distinguish the procedures from their explanations.

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

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

Notation of User Controls

**Operation/Soft Key Names and Menu Items Set in Boldface**
Boldface type indicates the names of user-controlled operation keys on the instrument panel, and soft key items and menu items displayed on screen.

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

**Unit**
- k: Denotes “1000.” Example: 100 kS/s (sample rate)
- K: Denotes “1024.” Example: 720 KB (file data size)
Symbols Used in the Syntax

The following table indicates symbols that are used in the syntax mainly in chapters 2 and 3. These symbols are referred to as BNF (Backus-Naur Form) symbols.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Example</th>
<th>Example of Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&gt;</td>
<td>Defined value</td>
<td>CHANNEL1&lt;x&gt; &lt;x&gt; = 1 to 4</td>
<td>CHANNEL2</td>
</tr>
<tr>
<td>{}</td>
<td>Select from values given in {}</td>
<td>COUPling {AC</td>
<td>DC</td>
</tr>
<tr>
<td></td>
<td>Exclusive OR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[]</td>
<td>Can be omitted</td>
<td>TRIGger [:SIMPle]:SLOPe</td>
<td>TRIGger:SLOPe</td>
</tr>
</tbody>
</table>
# Contents

How to Use This Manual ......................................................................................................................... iii

## Chapter 1 Connecting to a PC
1.1 Connecting via USB ......................................................................................................................... 1-1
1.2 Connecting via Ethernet (Optional) ................................................................................................. 1-2

## Chapter 2 Before Programming
2.1 Messages ........................................................................................................................................ 2-1
2.2 Commands ..................................................................................................................................... 2-3
2.3 Response ....................................................................................................................................... 2-5
2.4 Data ............................................................................................................................................... 2-6
2.5 Synchronization with the Controller ............................................................................................... 2-8

## Chapter 3 Commands
3.1 Notes on Use of Communication Commands ............................................................................... 3-1
3.2 A List of Commands ....................................................................................................................... 3-2
3.3 ACQuire Group ............................................................................................................................... 3-12
3.4 ALARm Group ............................................................................................................................... 3-14
3.5 ASETup Group ............................................................................................................................... 3-17
3.6 CALibrate Group ............................................................................................................................. 3-18
3.7 CHANnel Group ............................................................................................................................. 3-19
3.8 COMMunicate Group ...................................................................................................................... 3-33
3.9 CONTrol Group .............................................................................................................................. 3-35
3.10 DATa Group ................................................................................................................................ 3-36
3.11 ETHernet Group ............................................................................................................................ 3-37
3.12 FILE Group .................................................................................................................................. 3-38
3.13 GONogo Group ............................................................................................................................. 3-40
3.14 HISTory Group ............................................................................................................................. 3-42
3.15 INITialize Group ............................................................................................................................ 3-43
3.16 MEASure Group ............................................................................................................................ 3-44
3.17 MONitor Group .............................................................................................................................. 3-47
3.18 MRECord Group ............................................................................................................................ 3-49
3.19 MTRigger Group ............................................................................................................................. 3-51
3.20 SELFtest Group ............................................................................................................................. 3-52
3.21 SSTart Group ................................................................................................................................ 3-53
3.22 START Group ................................................................................................................................ 3-54
3.23 STATus Group ............................................................................................................................... 3-55
3.24 STOP Group .................................................................................................................................. 3-56
3.25 SYSTem Group ............................................................................................................................. 3-57
3.26 TIMebase Group ............................................................................................................................ 3-59
3.27 TRIGger Group ............................................................................................................................... 3-60
3.28 WAVeform Group ........................................................................................................................... 3-64
3.29 Common Command Group ............................................................................................................. 3-67
Chapter 4  Status Reports
  4.1  Overview of the Status Report .......................................................... 4-1
  4.2  Status Byte ....................................................................................... 4-3
  4.3  Standard Event Register .................................................................... 4-4
  4.4  Extended Event Register .................................................................... 4-5
  4.5  Output Queue and Error Queue .......................................................... 4-6

Appendix
  Appendix 1  ASCII Character Codes.......................................................... App-1
  Appendix 2  Error Messages...................................................................... App-2
Chapter 1    Connecting to a PC

1.1 Connecting via USB

Procedure

Use the following procedure to connect the SL1000 High-Speed Data Acquisition Unit (hereinafter, the SL1000 unit) to the PC.

Installing the Acquisition Software
Install the acquisition software that came with the SL1000 unit on the PC. For instructions, see section 2.2, “Installing or Uninstalling the Acquisition Software” in the SL1000 Acquisition Software User’s Manual (IM720120-61E).

Connecting via USB Cable
Connect the SL1000 unit to the PC by following the procedure in section 4.1, “Connecting to a PC” in the SL1000 High-Speed Data Acquisition Unit User’s Manual (IM720120-01E).

Installing the USB Driver
Install the USB driver on the PC by following the procedure in section 2.3, “Installing the USB Driver” in the SL1000 Acquisition Software Unit User’s Manual (IM720120-61E). This is only necessary the first time the SL1000 unit is connected to the PC.

Note
The USB and Ethernet interfaces cannot be used at the same.
1.2 Connecting via Ethernet (Optional)

Procedure

Use the following procedure to connect the SL1000 unit to the PC.

**Note**

Because communications settings are also entered on the SL1000 unit when connecting via Ethernet, at first it is necessary to connect via USB.

**Installing the Acquisition Software (First Time Only)**

Install the acquisition software that came with the SL1000 unit on the PC. For instructions, see section 2.2, “Installing or Uninstalling the Acquisition Software” in the SL1000 Acquisition Software User’s Manual (IM720120-61E).

**Connecting via USB Cable**

Connect the SL1000 unit to the PC by following the procedure in section 4.1, “Connecting to a PC” in the SL1000 High-Speed Data Acquisition Unit User’s Manual (IM720120-01E).

**Installing the USB Driver (First Time Only)**

Install the USB driver on the PC by following the procedure in section 2.3, “Installing the USB Driver” in the SL1000 Acquisition Software Unit User’s Manual (IM720120-61E). This is only necessary the first time the SL1000 unit is connected to the PC.

**Entering Communication Settings (TCP/IP Settings)**

Start the acquisition software, then enter communication settings for the SL1000 unit. For instructions, see section 3.2, “Specifying Communication Settings (When Using the Optional Ethernet Interface)” in the SL1000 Acquisition Software User’s Manual (IM720120-61E).

**Connecting via Ethernet Cable**

Close the acquisition software, then turn OFF the power to the SL1000 unit. Connect the SL1000 unit to the PC via Ethernet by following the procedure in section 4.1, “Connecting to a PC” in the SL1000 High-Speed Data Acquisition Unit User’s Manual (IM720120-01E).
Messages
Messages are used to exchange information between the controller and the instrument. Messages that are sent from the controller to the instrument are called program messages and messages that are sent back from the instrument to the controller are called response messages.
If a program message contains a message unit that requests a response (a query), the instrument returns a response message upon receiving the program message. A single response message is always returned in response to a single program message.

Program Messages
The program message format is shown below

<PMT>
PMT is a program message terminator. The following three types are available.
NL (New Line): Same as LF (Line Feed). ASCII code "0AH"
^EOM: The END message as defined by USBTMC (The data byte that is sent simultaneously with the END message is the last data of the program message.)
NL^EOM: NL with an END message added (NL is not included in the program message.)

Program Message Unit Format
The program message unit format is shown below

<Program header>
Space
<Program data>

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

<Program Data>
If certain conditions are required in executing a command, program data is added. A space (ASCII code "20H") separates the program data from the header. If there are multiple sets of program data, they are separated by commas (,).
For details, see page 4-5.

Example
:ACQuire:MODE NORMal;MEASure:MODE ON<PMT>

Response Messages
The response message format is shown belo.

<Response message unit>  <RMT>

<Response Message Unit>
A response message consists of one or more response message units; each response message unit corresponds to one response.
Response message units are separated by a semicolon (;).
For details regarding the format of the response message unit, see the next section.

Example
:ACQuire:MODE NORMal;MEASure:MODE ON<PMT>

<RMT>
A response message terminator. It is NL^EOM.

2.1 Messages

Response Message Unit Format
The response message unit format is shown below.

<Response header> Space <Response data>

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

<Response Data>
Response data contains the content of the response. If there are multiple sets of response data, they are separated by commas (,). For details, see page 4-5.

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

If there are multiple queries in a program message, responses are made in the same order as the queries. In most cases, a single query returns a single response message unit, but there are a few queries that return multiple units. The first response message unit always corresponds to the first query, but the nth response unit may not necessarily correspond to the nth query. Therefore, if you want to make sure that every response is retrieved, divide the program messages into individual messages.

Precautions to Be Taken when Transferring Messages
• If a program message containing multiple message units is sent, and the message contains incomplete units, the instrument attempts to execute the ones that are believed to be complete. However, these attempts may not always be successful. In addition, if the message contains queries, the responses may not be returned.

Deadlock
The instrument can store in its buffer program and response messages of length 1024 bytes or more (The number of available bytes varies depending on the operating conditions). When both the transmit and receive buffers become full at the same time, the instrument can no longer continue to operate. This state is called a deadlock. In this case, operation can be resumed by discarding the program message. Deadlock will not occur if the program message (including the <PMT>) is kept below 1024 bytes. Furthermore, deadlock never occurs if a program message does not contain a query.
2.2 Commands

Commands
There are three types of commands (program headers) that are sent from the controller to the instrument. They differ in their program header formats.

Common Command Header
Commands that are defined in the USBTMC-USB488 are called common commands. The header format of a common command is shown below. An asterisk (*) is always placed in the beginning of a command.

- **Common command example:** *CLS

Compound Header
Dedicated commands used by the instrument are classified and arranged in a hierarchy according to their functions. The format of a compound header is shown below. A colon (:) must be used to specify a lower hierarchy.

- **Compound header example:** :ACQuire:MODE

Simple Header
These commands are functionally independent and do not have a hierarchy. The format of a simple header is shown below.

- **Simple header example:** :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 that have common compound headers arranged in a hierarchy. A command group may contain sub-groups.

- **Example**
  Group of commands related to acquisition
  - :ACQuire:ECLock?
  - :ACQuire:ECLock:PCOunt
  - :ACQuire:ECLock:COUNt
  - :ACQuire:MMOde
  - :ACQuire:MODE
  - :ACQuire:TIME

  - **When Concatenating Commands of the Same Group**
The instrument stores the hierarchical level of the command that is currently being executed, and performs analysis on the assumption that the next command sent will also belong to the same level. Therefore, common header sections can be omitted for commands belonging to the same group.
  - **Example**
    ACQuire:MODE NORMal;
    TIME 0,0,0,0,500,0<PMT>

  - **When Concatenating Commands of Different Groups**
If the following command does not belong to the same group, a colon (:) is placed in front of the header (cannot be omitted).
  - **Example**
    :ACQuire:MODE
    NORMal;:CHANNEL1:ACCL:COUPLING GND<PMT>

  - **When Concatenating Simple Headers**
If a simple header follows another command, a colon (:) is placed in front of the simple header (cannot be omitted).
  - **Example**
    :ACQuire:MODE
    NORMal;:STARt<PMT>

  - **When Concatenating Common Commands**
Common commands that are defined in the USBTMC-USB488 are independent of hierarchy. Colons (:) are not needed before a common command.
  - **Example**
    :ACQuire:MODE NORMal;*CLS;
    TIME 0,0,0,0,500,0<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 specified for each command even when commands belonging to the same command group are being concatenated.
  - **Example**
    :ACQuire:MODE
    NORMal<PMT>;:ACQuire:TIME
    0,0,0,0,500,0<PMT>
2.2 Commands

Upper-Level Query
An upper-level query is a query in which a question mark (?) is appended to the highest level command of a group. Execution of an upper-level query allows all settings that can be specified in the group to be received at once. Some query groups which are comprised of more than three hierarchical levels can output all the lower level settings.

The response to an upper-level query can be transmitted as a program message back to the instrument. In this way, the settings that existed when the upper-level query was made can be restored. However, some upper-level queries do not return setup information that is not currently in use. It is important to remember that not all the group’s information is necessarily returned as part of a response.

Header Interpretation Rules
The instrument interprets the header that is received according to the rules below.

- Mnemonics are not case sensitive.
  Example “CONTrol” can also be written as “controL” or “ConTroL.”
- The lower-case section of the header can be omitted.
  Example “CONTrol” can also be written as “CONTR” or “CONt.”
- The question mark (?) at the end of a header indicates that it is a query. The question mark (?) cannot be omitted.
  Example The shortest abbreviation for CONTrol? is CONT?.
- If the <x> (value) at the end of a mnemonic is omitted, it is interpreted as a 1.
  Example If “CHANnel<x>” is written as “CHAN,” it means “CHANnel1.”
- The section enclosed by braces ([ ]) can be omitted.
  Example CHANnel{<VOLTage>:COUPling} can also be written as CHAN1:COUP.
  However, the last section enclosed by braces ([ ]) cannot be omitted in an upper-level query.
  Example “CHANnel1?” and “CHANnel1:COUPling?” are different queries.
2.3 Response

Response
When the controller sends a message unit that has a question mark (?) in its program header (query), the instrument returns a response message to the query. A response message is returned in one of the following two forms.

- **Response Consisting of a Header and Data**
  If the response can be used as a program message without any change, it is returned with a command header attached.
  
  **Example**
  
  :ACQUire:MODE?<PMT>
  -> :ACQUire:MODE NORMAL<RMT>

- **Response Consisting of Data Only**
  If the response cannot be used as a program message unless changes are made to it (query-only command), only the data section is returned. However, there are query-only commands that return responses with the header attached.

When You Wish to Return a Response without a Header
Responses that return both header and data can be set so that only the data section is returned. The "COMMunicate:HEADer" command is used to do this.

Abbreviated Form
Normally, the lower-case section 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 sections enclosed by braces ([ ]) are also omitted in the abbreviated form.
## 2.4 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.

<table>
<thead>
<tr>
<th>Data</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Decimal&gt;</td>
<td>A value expressed as a decimal number</td>
</tr>
<tr>
<td>&lt;Voltage&gt;</td>
<td>A physical value</td>
</tr>
<tr>
<td>&lt;Frequency&gt;</td>
<td>Time axis range</td>
</tr>
<tr>
<td>&lt;Current&gt;</td>
<td>Register value expressed as binary, octal, decimal or hexadecimal.</td>
</tr>
<tr>
<td>&lt;Register&gt;</td>
<td>Predefined character string (mnemonic). Can be selected from {}.</td>
</tr>
<tr>
<td>&lt;Character Data&gt;</td>
<td>An arbitrary character string (Example: Comment to a screen data output)</td>
</tr>
<tr>
<td>&lt;Boolean&gt;</td>
<td>Indicates ON and OFF. Set using ON, OFF or a value</td>
</tr>
<tr>
<td>&lt;String data&gt;</td>
<td>An arbitrary character string (Example: Select the input coupling of CH1)</td>
</tr>
<tr>
<td>&lt;Filename&gt;</td>
<td>Indicates a file name.</td>
</tr>
<tr>
<td>&lt;Block data&gt;</td>
<td>Arbitrary 8-bit data</td>
</tr>
<tr>
<td>&lt;Decimal&gt;</td>
<td>Indicates a value expressed as a decimal number as shown in the table below.</td>
</tr>
</tbody>
</table>

### <Decimal>

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

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NR1&gt;</td>
<td>Integer</td>
<td>125-1</td>
</tr>
<tr>
<td>&lt;NR2&gt;</td>
<td>Fixed-point number</td>
<td>125.0 ± 0.9 0.001</td>
</tr>
<tr>
<td>&lt;NR3&gt;</td>
<td>Floating-point number</td>
<td>125.0±E0 +9E-1 +1E4</td>
</tr>
<tr>
<td>&lt;NR&gt;</td>
<td>Any of the forms &lt;NR1&gt; to &lt;NR3&gt; is allowed.</td>
<td></td>
</tr>
</tbody>
</table>

- The instrument can receive decimal values that are sent from the controller in any of the forms, <NR1> to <NR3>. This is represented by <NR>.  
- For response messages that the instrument returns to the controller, the form (<NR1> to <NR3> to be used) is determined by the query. The same form is used regardless of the size of the value.  
- For the <NR3> format, the “+” sign after the “E” can be omitted. However, the “-” sign cannot be omitted.  
- If a value outside the setting range is entered, the value is normalized so that it is just inside the range.  
- If a value has more significant digits than the available resolution, the value is rounded.

### <Voltage>, <Time>, <Frequency>, and <Current>

<Voltage>, <Time>, <Frequency>, and <Current> indicate decimal values that have physical significance. <Multiplier> or <Unit> can be attached to the <NRf> form that was described earlier. It is expressed in one of the following forms.

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NRf&gt;&lt;Multiplier&gt;&lt;Unit&gt;</td>
<td>5MV</td>
</tr>
<tr>
<td>&lt;NRf&gt;&lt;Unit&gt;</td>
<td>5E-3V</td>
</tr>
<tr>
<td>&lt;NRf&gt;&lt;Multiplier&gt;</td>
<td>5M</td>
</tr>
<tr>
<td>&lt;NRf&gt;</td>
<td>5E-3</td>
</tr>
</tbody>
</table>

### <Multiplier>

<Multipliers> which can be used are indicated below.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Word</th>
<th>Multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>EX</td>
<td>Exa</td>
<td>10^18</td>
</tr>
<tr>
<td>PE</td>
<td>Peta</td>
<td>10^15</td>
</tr>
<tr>
<td>T</td>
<td>Tera</td>
<td>10^12</td>
</tr>
<tr>
<td>MA</td>
<td>Mega</td>
<td>10^9</td>
</tr>
<tr>
<td>K</td>
<td>Kilo</td>
<td>10^6</td>
</tr>
<tr>
<td>M</td>
<td>Milli</td>
<td>10^3</td>
</tr>
<tr>
<td>U</td>
<td>Micro</td>
<td>10^-6</td>
</tr>
<tr>
<td>N</td>
<td>Nano</td>
<td>10^-9</td>
</tr>
<tr>
<td>P</td>
<td>Pico</td>
<td>10^-12</td>
</tr>
<tr>
<td>F</td>
<td>Femto</td>
<td>10^-15</td>
</tr>
<tr>
<td>A</td>
<td>Ampo</td>
<td>10^-18</td>
</tr>
</tbody>
</table>

### <Unit>

<Units> that can be used are indicated below.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Word</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>Volt</td>
<td>Voltage</td>
</tr>
<tr>
<td>S</td>
<td>Second</td>
<td>Time</td>
</tr>
<tr>
<td>HZ</td>
<td>Hertz</td>
<td>Frequency</td>
</tr>
<tr>
<td>MHZ</td>
<td>Megahertz</td>
<td>Frequency</td>
</tr>
<tr>
<td>A</td>
<td>Ampere</td>
<td>Current</td>
</tr>
</tbody>
</table>

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

---

IM 720320-17E
<Register>
<Register> indicates an integer, and can be expressed in hexadecimal, octal, or binary as well as a decimal number. It is used when each bit of the value has a particular meaning. It is expressed in one of the following forms.

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NRf&gt;</td>
<td>1</td>
</tr>
<tr>
<td>#H&lt;Hexadecimal value made up of the digits 0 to 9 and A to F&gt;</td>
<td>#0F</td>
</tr>
<tr>
<td>#Q&lt;Octal value made up of the digits 0 to 7&gt;</td>
<td>#777</td>
</tr>
<tr>
<td>#B&lt;Binary value made up of the digits 0 and 1&gt;</td>
<td>#001100</td>
</tr>
</tbody>
</table>

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

<Character Data>
.gradle: set the character 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.

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[AC</td>
<td>DC</td>
</tr>
</tbody>
</table>

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

<Boolean>
<Boolean> is data that indicates ON or OFF. It is expressed in one of the following forms.

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ON</td>
<td>OFF]&lt;NRf&gt;</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;String data&gt;</td>
<td>'ABC' &quot;IEEE488.2-1987&quot;</td>
</tr>
</tbody>
</table>

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

<Filename>
<Filename> is data that indicates a file name. It is expressed in one of the following forms.

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;NRf&gt;:&lt;Character data&gt;</td>
<td>1 CASE &quot;CASE&quot;</td>
</tr>
</tbody>
</table>

- <NRf> is rounded to an 8-digit integer and converted to ASCII code. The result is the file name (example: 1 becomes "00000001"). Negative values are not allowed.
- Response messages are always returned in the <String data> form.
- For <Character data>, the first 12 characters become the file name.
- For <String data>, the first 259 characters become the file name.
- For a description of the number of characters of the <String data> file name, see the DL9500/DL9700 User's Manual.

<Block data>
<Block data> is arbitrary 8-bit data. It is only used in response messages on the DL9500/DL9700. Below is the syntax.

<table>
<thead>
<tr>
<th>Form</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>#N&lt;N-digit decimal number&gt;</td>
<td>#00000010ABCDEFGHIJ</td>
</tr>
<tr>
<td>&lt;Data byte sequence&gt;</td>
<td></td>
</tr>
</tbody>
</table>

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

Overlap Commands and Sequential Commands
There are two types of commands, overlap commands and sequential commands. In the case of overlap commands, the execution of the next command may start before the execution of the previous command is completed.

For example, if the next program message is transmitted when specifying the V/div value and querying the result, the response always returns the most recent setting (5 V in this case).

:CHANnel1:VDIV 5V;VDIV?<PMT>
This is because the next command is forced to wait until the processing of "CHANnel1:VDIV" itself is completed. This type of command is called a sequential command.

On the contrary, let us assume that you send the next program message when you wish to load a file and query the V/div value of the result.

:FILE:LOAD:SETup:EXECute "CASE1";:
CHANnel1:VDIV?
In this case, "CHANnel1:VDIV?" is executed before the loading of the file is completed, and the V/div value that is returned is the value before the file is loaded. The act of executing the next command before the processing of itself is completed such as with "FILE:LOAD:SETup:EXECute "CASE1"" is called an overlap operation. A command that operates in this way is called an overlap command.

In such case, the overlap operation can be prevented by using the methods below.

Synchronizing with Overlap Commands
• Using the *WAI Command
The *WAI command holds the subsequent commands until the overlap command is completed.

Example :COMMunicate:OPSE #H0040;:
FILE:LOAD:SETup:
EXECute "CASE1";*WAI;:
CHANnel1:VDIV?<PMT>
"COMMunicate:OPSE" is a command used to select the "*WAI" target. Here, media access is specified. Because "*WAI" is executed immediately before "CHANnel1:VDIV?," "CHANnel1:VDIV?" is not executed until the file loading is complete.

• Using the COMMunicate:OVERIap command
The COMMunicate:OVERIap command enables (or disables) overlap operation.

Example :COMMunicate:OVERIap #HFFBF;:
FILE:LOAD:SETup:
EXECute "CASE1";:CHANnel1:VDIV?<PMT>
"COMMunicate:OVERIap #HFFBF" enables overlap operation on commands other than media access. Because the overlap operation of file loading is disabled, "FILE:LOAD:SETup:EXECute "CASE1"" operates in the same way as a sequential command. Therefore, "CHANnel1:VDIV?" is not executed until the file loading is complete.

• Using the *OPC Command
The *OPC command sets the OPC bit, bit 0 of the standard event register (see page 6-4), to 1 when the overlap operation is completed.

Example :COMMunicate:OPSE #H0040;:
*ESE 1;
*ESR?;*SRE 32;:FILE:LOAD:SETup:
EXECute "CASE1";*OPC<PMT>
(Read the response to *ESR?)
(Wait for a service request)
:CHANnel1:VDIV?<PMT>
"COMMunicate:OPSE" is a command used to select the "*OPC" target. Here, media access is specified. **ESE 1" and "*SRE 32" indicate that a service request is generated only when the OPC bit is 1. *ESR? clears the standard event register.

In the example above, "CHANnel1:VDIV?" is not executed until a service request is generated.
• Using the *OPC? Query
The *OPC? query generates a response when an overlap operation is completed.

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

(Read the response to *OPC?)

"COMMunicate:OPSE" is a command used to select the "*OPC?" target. Here, media access is specified.

Because "*OPC?" does not generate a response until the overlap operation is completed, the loading of the file will have been completed by the time the response to "*OPC?" is read.

Note:
Most commands are sequential commands. Overlap commands are indicated as overlap commands in chapter 5. All other commands are sequential commands.

Achieving Synchronization without Using Overlap Commands
Even for sequential commands, synchronization is sometimes required for non-communication-related reasons such as a trigger occurrence.

For example, if the next program message is transmitted to make an inquiry about the waveform data which has been acquired with the trigger mode set to single, the WAVeform:SEND? command may be executed regardless of whether the acquisition has been completed or not and may result in command execution error.

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

In this case, the following method must be used to synchronize with the end of the acquisition.

• Using the STATus:CONDition? Query
The "STATus:CONDition?" query is used to query the contents of the condition register (page 6-5).

Whether waveforms are being retrieved can be determined by reading bit 0 of the condition register. If bit 0 of the condition register is "1," waveforms are being retrieved. Otherwise, it is stopped.

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

(Read the response. If bit 0 is 1, repeat this command until it becomes 1.)

`:WAVeform:SEND?<PMT>`

The WAVeform:SEND? command will not be executed until bit 0 of the condition register is set to "0."

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

Example: `:STATus:FILTer1 FALL;:STATus:EESE 1;:STATus:EESR?`

(Read the response to STATus:EESR?)

A service request will be generated on the generation of the "*SRE 8" command.

The "STATus:EESE 1" command is used to clear the extended event register.

The "*SRE 8" command is used to generate a service request solely on the cause of the extended event register.

The WAVeform:SEND? command is not 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:EESE 1;:TRIGger:MODE SINGle;:STATus:CONDition?<PMT>`

(Read the response to STATus:EESR?)

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

For a description of "STATus:FILTer1 FALL" and "STATus:EESE?" see the previous section regarding the extended event register.

The "COMMunicate:WAIT 1" command indicates that the program will wait for bit 0 of the extended event register to be set to "1."

The WAVeform:SEND? command will not be executed until bit 0 of the extended event register is set to "1."
3.1 Notes on Use of Communication Commands

When using communication commands, please note the following.

- The following functions of the SL1000 High-Speed Data Acquisition Unit are not available when using communication commands.
  - Synchronized operation of multiple units (the API can be used for synchronized operation)
  - Automatic recording to PC
  - Reading of data per WAVeform group (only possible while measurement is stopped).
  - At least one module must be set to measurement group 1 (using the: TIMebase: MODUle<x>:GROUp command).
  - The initial value of the: CHANnel<x>:DISPlay command that turns measurement ON/OFF on individual channels is OFF. Turn this setting ON when using the command for the first time.
  - When using the MRECord group commands for auto recording, the initial value for the recording destination is “PC.” When auto-recording, you must change the recording destination to HDD.
## 3.2 A List of Commands

### ACQuire Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ACQuire?</td>
<td>Queries all settings related to waveform acquisition.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:CLOCK</td>
<td>Sets or queries the time base (internal/external clock).</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:COUNt</td>
<td>Sets or queries the waveform acquisition count for normal mode.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:EClock? (Ext CLock)</td>
<td>Sets or queries all settings related to the external sample clock.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:EClock:PCount (Ext CLock Pretrigger CCount)</td>
<td>Sets or queries the pre-trigger count when using the external sample clock.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:EClock:COUNT (Ext CLock COUNT)</td>
<td>Sets or queries the sample count when using the external sample clock.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:MMOde (Motion MODe)</td>
<td>Sets or queries the waveform acquisition operation mode.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:MODE</td>
<td>Sets or queries the waveform acquisition mode.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:TIME</td>
<td>Sets or queries the measurement time.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:PROtate</td>
<td>Sets or queries the Pulse/Rotate setting when inputting the external clock.</td>
<td>3-12</td>
</tr>
<tr>
<td>:ACQuire:NNINum?</td>
<td>Queries the maximum history number for trigger measurement mode.</td>
<td>3-13</td>
</tr>
</tbody>
</table>

### ALARm Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ALARm?</td>
<td>Queries all settings related to alarms.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:ACK:EXECute</td>
<td>Clears alarm output.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:ACount?</td>
<td>Queries the number of acquisitions, which is counted from the start of measurement, when an alarm occurs.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:COMBination</td>
<td>Sets or queries the AND/OR state of the alarms of each channel.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CONDition?</td>
<td>Queries the alarm output terminal condition.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CHANnel&lt;x?&gt;</td>
<td>Queries all settings related to channel alarms.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CHANnel&lt;x&gt;:=CONDITION?</td>
<td>Queries the alarm condition of the specified channel.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CHANnel&lt;x1&gt;:HYSTeresis&lt;x2&gt;</td>
<td>Sets or queries the alarm hysteresis of a channel.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CHANnel&lt;x1&gt;:LEVEL&lt;x2&gt;</td>
<td>Sets or queries the alarm level of a channel (when the input of the specified channel is not logic).</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CHANnel&lt;x1&gt;:TYPE</td>
<td>Sets or queries the alarm type of a channel.</td>
<td>3-14</td>
</tr>
<tr>
<td>:ALARm:CHANnel&lt;x1&gt;:AVALUE? (Alarm VALue)</td>
<td>Queries the measured value at the alarm occurrence on the specified channels as an ASCII string.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:CTIMe?</td>
<td>Queries the time of the most recent channel alarm condition change.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:HDLOD</td>
<td>Sets or queries the alarm hold.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:MODE</td>
<td>Sets or queries the alarm operation mode.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:CMODE</td>
<td>Sets or queries the channel alarm operation mode.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:SMODE</td>
<td>Sets or queries the system alarm operation mode.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:TERminal</td>
<td>Sets or queries the alarm output terminal on/off state.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:SOURce</td>
<td>Sets or queries the alarm detection source.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:STATus?</td>
<td>Queries the channel alarm status.</td>
<td>3-15</td>
</tr>
<tr>
<td>:ALARm:SSTatus?</td>
<td>Queries the system alarm status value.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:SYSTem?</td>
<td>Queries all settings related to the system alarm.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:SYSTem:SOURce?</td>
<td>Queries all settings related to system alarm detection.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:SYSTem:SOURce:BOVerrun</td>
<td>Sets or queries system alarm buffer overrun detection.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:SYSTem:SOURce:FSTop</td>
<td>Sets or queries system alarm fan stop detection.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:SYSTem:SOURce:DFULL</td>
<td>Sets or queries system alarm HDD full detection.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:SYSTem:SOURce:MRTime</td>
<td>Sets or queries the maximum recording time detection for free-run automatic recording mode.</td>
<td>3-16</td>
</tr>
<tr>
<td>:ALARm:STIMe?</td>
<td>Queries the time of the most recent system alarm condition change.</td>
<td>3-16</td>
</tr>
</tbody>
</table>

### ASETup Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ASETup?</td>
<td>Queries all settings related to auto setup.</td>
<td>3-17</td>
</tr>
<tr>
<td>:ASETup:EXECute</td>
<td>Execute auto setup.</td>
<td>3-17</td>
</tr>
<tr>
<td>:ASETup:TARGet</td>
<td>Sets or queries the target channel for auto setup.</td>
<td>3-17</td>
</tr>
<tr>
<td>:ASETup:UNDO</td>
<td>Cancels auto setup.</td>
<td>3-17</td>
</tr>
</tbody>
</table>
### 3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CALibrate Group</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>:CALibrate?</td>
<td>Queries all settings related to calibration.</td>
<td>3-18</td>
</tr>
<tr>
<td>:CALibrate[:EXECute]</td>
<td>Executes calibration.</td>
<td>3-18</td>
</tr>
<tr>
<td>:CALibrate:MODE</td>
<td>Sets or queries the ON/OFF state of auto calibration.</td>
<td>3-18</td>
</tr>
<tr>
<td><strong>CHANnel Group</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:ACCL]?</td>
<td>Queries all settings on the channel with the acceleration/voltage module installed.</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL?</td>
<td>Queries all settings related to the vertical axis of the channel.</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL:BIAS</td>
<td>Sets or queries the on/off status of the acceleration sensor’s bias current when an Acceleration/Voltage Module is installed in the specified channel (slot).</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL:BWIDth</td>
<td>Sets or queries the filter when an Acceleration Voltage Module is installed in the specified channel (slot).</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL:COUPling</td>
<td>Sets or queries input coupling when an Acceleration/Voltage Module is installed in the specified channel (slot).</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL:GAIN</td>
<td>Sets or queries the gain when an Acceleration/Voltage Module is installed in the specified channel (slot).</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL:SENSitivity</td>
<td>Sets or queries the sensitivity when an Acceleration/Voltage Module is installed in the specified channel (slot).</td>
<td>3-19</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ACCL:UNIT</td>
<td>Sets or queries the unit of measurement of the upper and lower limit values when an Acceleration/Voltage Module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:ASET?</td>
<td>Queries whether the specified channel is able to set an auto setup or not.</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:DISPLAY</td>
<td>Sets or queries the channel acquisition ON/OFF state.</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ?</td>
<td>Queries all settings when a frequency module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut?</td>
<td>Queries all settings related to the input when a frequency module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:BWIDth</td>
<td>Sets or queries the bandwidth limit when a frequency module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:CELimation</td>
<td>Sets or queries chattering elimination when a frequency module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:COUPling</td>
<td>Sets or queries input coupling when a frequency module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:HYSTeresis</td>
<td>Sets or queries hysteresis when a frequency module is installed in the specified channel (slot).</td>
<td>3-20</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:PRESet</td>
<td>Sets or queries the preset when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:PROBe</td>
<td>Sets or queries the probe attenuation when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:PULLup</td>
<td>Sets or queries the pull-up on/off state when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:SLOPe</td>
<td>Sets or queries the slope when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:THReshold</td>
<td>Sets or queries the threshold level when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:INPut:VRANGe</td>
<td>Sets or queries the voltage range when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale?</td>
<td>Queries all settings related to linear scaling when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale:AVALue</td>
<td>Sets or queries linear scaling coefficient A when a frequency module is installed in the specified channel (slot).</td>
<td>3-21</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale:BVALue</td>
<td>Sets or queries linear scaling coefficient B when a frequency module is installed in the specified channel (slot).</td>
<td>3-22</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale:GETMeasure</td>
<td>Measures the X values of P1 and P2 for linear scaling when a frequency module is installed in the specified channel (slot).</td>
<td>3-22</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale:MODE</td>
<td>Sets or queries linear scaling when a frequency module is installed in the specified channel (slot).</td>
<td>3-22</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale:(P1X</td>
<td>P1Y</td>
<td>P2X</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:LSCale:UNIT</td>
<td>Sets or queries the unit of measurement to attach to the result of linear scaling when a frequency module is installed in the specified channel (slot).</td>
<td>3-22</td>
</tr>
</tbody>
</table>
### 3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:OFFSet</td>
<td>Sets or queries the offset value when a frequency module is installed in the specified channel (slot).</td>
<td>3-22</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup?</td>
<td>Queries all settings related to FV setup when a frequency module is installed in the specified channel (slot).</td>
<td>3-22</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:CFrequency</td>
<td>Sets or queries the center frequency when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:DECEleration</td>
<td>Sets or queries the on/off state of deceleration prediction when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:DPULse</td>
<td>Sets or queries the distance per pulse when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER?</td>
<td>Queries all settings related to the filter when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER:SMoothing?</td>
<td>Queries all settings related to smoothing when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER:SMoothing:MODE</td>
<td>Sets or queries the on/off state of smoothing when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER:SMoothing:VALue</td>
<td>Sets or queries the moving average order of smoothing when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER:PAVerage?</td>
<td>Queries all settings related to pulse average when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER:PAVerage:MODE</td>
<td>Sets or queries the on/off state of pulse average mode when a frequency module is installed in the specified channel (slot).</td>
<td>3-23</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FILTER:PAVerage:VALue</td>
<td>Sets or queries the pulse average count when a frequency module is installed in the specified channel (slot).</td>
<td>3-24</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:FUNCTION</td>
<td>Sets or queries the measuring mode when a frequency module is installed in the specified channel (slot).</td>
<td>3-24</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:LRSet</td>
<td>Sets or queries the over limit reset when a frequency module is installed in the specified channel (slot).</td>
<td>3-24</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:MPULse</td>
<td>Sets or queries whether the measurement pulse is positive or negative when a frequency module is installed in the specified channel (slot).</td>
<td>3-24</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:PROTate</td>
<td>Sets or queries the number of pulses per rotation when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:RESet</td>
<td>Resets the pulse count when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:STOPpredict</td>
<td>Sets or queries the on/off state of stop prediction when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:TUNit</td>
<td>Sets or queries the time unit when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:UNIT</td>
<td>Sets or queries the pulse integration unit when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:UPULse</td>
<td>Sets or queries the unit/pulse when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:SETup:VUNIT</td>
<td>Sets or queries the unit of velocity when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:FREQ:VDIV</td>
<td>Sets or queries the Value/Div when a frequency module is installed in the specified channel (slot).</td>
<td>3-25</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:LABe1</td>
<td>Sets or queries the waveform label of the specified channel.</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:MODule?</td>
<td>Queries the module installed in the channel (slot).</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:RECord</td>
<td>Sets or queries whether to record the specified channel or not.</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:BALance?</td>
<td>Queries the balance setting when a strain module is installed in the specified channel (slot).</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:BALance:EXECute</td>
<td>Sets or queries the channel on which balancing is to be executed when a strain module is installed in the specified channel (slot).</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:BALance:BWIDTH</td>
<td>Balances strain when a strain module is installed in the specified channel (slot).</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:BALance:INVert</td>
<td>Sets or queries the filter when a strain module is installed in the specified channel (slot).</td>
<td>3-26</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:EXCitation</td>
<td>Sets or queries the bridge voltage when a strain module is installed in the specified channel (slot).</td>
<td>3-27</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:GFActor</td>
<td>Sets or queries the gauge factor when a strain module is installed in the specified channel (slot).</td>
<td>3-27</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:INVert</td>
<td>Sets or queries whether or not the display is inverted when a strain module is installed in the specified channel (slot).</td>
<td>3-27</td>
</tr>
<tr>
<td>Command</td>
<td>Function</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>------</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale?</td>
<td>Queries all settings related to linear scaling when a strain module is installed in the specified channel (slot).</td>
<td>3-27</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:AValue</td>
<td>Sets or queries linear scaling coefficient A when a strain module is installed in the specified channel (slot).</td>
<td>3-27</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:BValue</td>
<td>Sets or queries offset value B when a strain module is installed in the specified channel (slot).</td>
<td>3-27</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:DISPLAYtype:MODE</td>
<td>Sets or queries the display format for linear scaling.</td>
<td>3-27</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:DISPLAYtype:DECimalnum</td>
<td>Sets or queries the decimal place when the display format for linear scaling is set to Floating.</td>
<td>3-28</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:GETMeasure</td>
<td>Measures the X values of P1 and P2 for linear scaling when a strain module is installed in the specified channel (slot).</td>
<td>3-28</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:DISPLAYtype:MODE</td>
<td>Sets or queries the linear scaling method when a strain module is installed in the specified channel (slot). (The method can only be set to SHUNt with a strain module with DSUB, Shunt-Cal.)</td>
<td>3-28</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:SHUNt</td>
<td>Executes shunt calibration when a strain module is installed in the specified channel (slot). (This command only works with a strain module with DSUB, Shunt-Cal.)</td>
<td>3-28</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:LSCale:UNIT</td>
<td>Sets or queries the unit of measurement to attach to the result of linear scaling when a strain module is installed in the specified channel (slot).</td>
<td>3-28</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:RANGE</td>
<td>Sets or queries the measuring range when a strain module is installed in the specified channel (slot).</td>
<td>3-28</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:STRain:UNIT</td>
<td>Sets or queries the unit of measurement when a strain module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature?</td>
<td>Queries all settings when a temperature, high precision voltage module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature:BURNout</td>
<td>Sets or queries whether or not burnout is detected when a temperature, high precision voltage isolation module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature:BWIDth</td>
<td>Sets or queries the bandwidth limit when a temperature, high precision voltage module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature:COUPling</td>
<td>Sets or queries input coupling when a temperature, high precision voltage module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature:RJC</td>
<td>Sets or queries the RJC when a temperature, high precision voltage module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature:TYPE</td>
<td>Sets or queries the thermocouple type when a temperature, high precision voltage module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:TEMPerature:UNIT</td>
<td>Sets or queries the unit of measurement values when a temperature, high precision voltage module is installed in the specified channel (slot).</td>
<td>3-29</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:UNIT?</td>
<td>Queries the unit added to the channel.</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:VOLTage?</td>
<td>Queries all settings when a voltage module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:BWIDth</td>
<td>Sets or queries the bandwidth limit when a temperature, high precision voltage isolation module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:COUPling</td>
<td>Sets or queries input coupling when a voltage module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:INVert</td>
<td>Sets or queries whether or not the display is inverted when a voltage module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:LSCale?</td>
<td>Queries all settings related to linear scaling when a voltage module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:LSCale:AValue</td>
<td>Sets or queries scaling coefficient A of linear scaling when a voltage module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:LSCale:BValue</td>
<td>Sets or queries linear scaling offset value B when a voltage module is installed in the specified channel (slot).</td>
<td>3-30</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:DISPLAYtype:MODE</td>
<td>Sets or queries the display format for linear scaling.</td>
<td>3-31</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;[:VOLTage]:DISPLAYtype:DECimalnum</td>
<td>Sets or queries the decimal place when the display format for linear scaling is set to Floating.</td>
<td>3-31</td>
</tr>
</tbody>
</table>
### 3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CHANnel&lt;x&gt;:VOLTage:LSCale:DISPLAYtype:SUBunit</td>
<td>Sets or queries the sub unit when the display format for linear scaling is set to Floating.</td>
<td>3-31</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:VOLTage:LSCale:GETMeasure</td>
<td>Measures the X values of P1 and P2 for linear scaling when a voltage module is installed in the specified channel (slot).</td>
<td>3-31</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:VOLTage:LSCale:MODE</td>
<td>Sets or queries linear scaling when a voltage module is installed in the specified channel (slot).</td>
<td>3-31</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:VOLTage:LSCale:{P1X</td>
<td>P1Y</td>
<td>P2X</td>
</tr>
<tr>
<td>:CHANnel&lt;x&gt;:VOLTage:LSCale:UNIT</td>
<td>Sets or queries the unit of measurement to attach to the result of linear scaling when a voltage module is installed in the specified channel (slot).</td>
<td>3-32</td>
</tr>
</tbody>
</table>

#### COMMunicate Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:COMMunicate?</td>
<td>Queries all settings related to communications.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:HEAder</td>
<td>Sets or queries whether or not to add a header to responses to queries.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:LOCKout</td>
<td>Sets or clears local lockout.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:OPSE</td>
<td>Sets or queries the overlap command that is used by the *OPC, *OPC? and *WAI commands.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:OPSR?</td>
<td>Queries the value of the operation pending status register.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:OVERlap</td>
<td>Sets or queries the commands to operate as overlap commands.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:REMote</td>
<td>Sets remote or local. ON is remote mode</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:STATus?</td>
<td>Queries the line-specific status.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:VERBose</td>
<td>Sets or queries whether to return the response to a query using full spelling or abbreviations.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:WAIT</td>
<td>Waits for one of the specified extended events to occur.</td>
<td>3-33</td>
</tr>
<tr>
<td>:COMMunicate:WAIT?</td>
<td>Creates the response that is returned when the specified event occurs.</td>
<td>3-34</td>
</tr>
</tbody>
</table>

#### CONTrol Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:CONTrol?</td>
<td>Queries all settings related to the station.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:FREE?</td>
<td>Queries all settings related to free-run measurement.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:FREE:LATCH</td>
<td>Executes the latch for the measurement data access for free-run measurement. (Unit firmware holds ‘Acq memory writing address’ of measurement group 1 data.)</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:FREE:LCount?</td>
<td>Queries the sample counts, which are counted from the start of measurement, at the latch point for free-run measurement.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:FREE:LENGth&lt;x&gt;?</td>
<td>Queries the measured efficient data counts at the latch point for free-run measurement.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:FREE:STIMe?</td>
<td>Queries the start time of measurement for free-run measurement.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:FREE:GDELAY&lt;x&gt;?</td>
<td>Queries the delay between the measurement start points of each measurement group.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:HDCapacity?</td>
<td>Queries the total capacity of internal HDD.</td>
<td>3-35</td>
</tr>
<tr>
<td>:CONTrol:HRDFree?</td>
<td>Queries the amount of free space in the internal HDD.</td>
<td>3-35</td>
</tr>
</tbody>
</table>

#### DATa Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:DATA?</td>
<td>Queries all settings related to waveform data.</td>
<td>3-36</td>
</tr>
<tr>
<td>:DATA:BYTeorder</td>
<td>Sets or queries the transmission order when using word format of two bytes or more.</td>
<td>3-36</td>
</tr>
<tr>
<td>:DATA:FWRAW?</td>
<td>Queries the specified waveform data during free-run measurement. The acquisition start point and the acquisition count are specified by the count of the measurement groups to which a specified trace belongs.</td>
<td>3-36</td>
</tr>
</tbody>
</table>

#### ETHernet Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:ETHernett?</td>
<td>Queries all settings related to the network.</td>
<td>3-37</td>
</tr>
<tr>
<td>:ETHernett:TCP?</td>
<td>Queries all setting related to TCP.</td>
<td>3-37</td>
</tr>
<tr>
<td>:ETHernett:TCPip:DHCp</td>
<td>Sets or queries DHCP.</td>
<td>3-37</td>
</tr>
<tr>
<td>:ETHernett:TCPip:GATEway</td>
<td>Sets or queries default gateway.</td>
<td>3-37</td>
</tr>
<tr>
<td>:ETHernett:TCPip:IPAddress</td>
<td>Sets or queries the IP address.</td>
<td>3-37</td>
</tr>
<tr>
<td>:ETHernett:TCPip:NETMask</td>
<td>Sets or queries the subnet mask.</td>
<td>3-37</td>
</tr>
</tbody>
</table>

#### FILE Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:FILE?</td>
<td>Queries all settings related to the internal hard disk.</td>
<td>3-38</td>
</tr>
</tbody>
</table>
## 3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:FILE:DELe</td>
<td>Deletes files.</td>
<td>3-38</td>
</tr>
<tr>
<td>:FILE:DIRectory</td>
<td>Queries all settings related to the directory of the storage media.</td>
<td>3-38</td>
</tr>
<tr>
<td>:FILE[:DIRectory]:CDIRectory</td>
<td>Changes the current directory of the storage media.</td>
<td>3-38</td>
</tr>
<tr>
<td>:FILE[:DIRectory]:DRive</td>
<td>Sets the storage media to be controlled.</td>
<td>3-38</td>
</tr>
<tr>
<td>:FILE[:DIRectory]:FREE?</td>
<td>Queries the free disk space (bytes) on the target media.</td>
<td>3-38</td>
</tr>
<tr>
<td>:FILE[:DIRectory]:RMDirectory</td>
<td>Deletes the specified directory in the count directory. This is an overlap command.</td>
<td>3-38</td>
</tr>
</tbody>
</table>

### GONogo Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:GONogo?</td>
<td>Queries all settings related to GO/NO-GO judgment.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:ACONDITION</td>
<td>Sets or queries the GO/NO-GO judgment action condition.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:ACTion?</td>
<td>Queries all settings related to the action taken when the execution condition is met.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:ACTion:BUZZer</td>
<td>Sets or queries whether or not a beep is sounded when the condition is met.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:ACTion:SAVE[:MODE]</td>
<td>Sets or queries whether or not waveform data is saved to the storage media when the condition is met.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:AREA</td>
<td>Sets or queries the waveform area that is judged.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:COUNT?</td>
<td>Queries the number of performed GO/NO-GO judgments.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:RStatus?</td>
<td>Queries the most recent GO/NO-GO judgment.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:LOGic</td>
<td>Sets or queries the GO/NO-GO logical condition.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:MODE</td>
<td>Sets or queries the GO/NO-GO judgment mode.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:NOCount?</td>
<td>Queries the GO/NO-GO judgment NO-GO count.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:PARameter?</td>
<td>Queries all settings related to parameter judgment.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:PARameter:ITEM&lt;x&gt;?</td>
<td>Queries all settings related to waveform parameters of the parameter judgment.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:PARameter:ITEM&lt;x&gt;:CAUSE?</td>
<td>Queries whether or not the specified waveform parameter is the cause of a NO-GO judgment.</td>
<td>3-40</td>
</tr>
<tr>
<td>:GONogo:PARameter:ITEM&lt;x&gt;:MODE</td>
<td>Sets or queries the judgment criteria of the specified waveform parameter of the parameter judgment.</td>
<td>3-41</td>
</tr>
<tr>
<td>:GONogo:PARameter:ITEM&lt;x&gt;:TRACE</td>
<td>Sets or queries the target waveform of the measurement of the specified waveform parameters of the parameter judgment.</td>
<td>3-41</td>
</tr>
<tr>
<td>:GONogo:PARameter:ITEM&lt;x&gt;:TYPE?</td>
<td>Queries the item and the upper and lower limits of the measurement of the specified waveform parameter of the parameter judgment.</td>
<td>3-41</td>
</tr>
</tbody>
</table>

### HISTory Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:HISTory?</td>
<td>Queries all settings related to the history memory function.</td>
<td>3-42</td>
</tr>
<tr>
<td>:HISTory:CLear</td>
<td>Clears all history memory data (all data in memory).</td>
<td>3-42</td>
</tr>
<tr>
<td>:HISTory:REcord? MINimum</td>
<td>Queries the minimum record number.</td>
<td>3-42</td>
</tr>
</tbody>
</table>
3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:HISTORY:DATE?</td>
<td>Queries the trigger date of the target record number.</td>
<td>3-42</td>
</tr>
<tr>
<td>:HISTORY:TIME?</td>
<td>Queries the trigger time of the target record number.</td>
<td>3-42</td>
</tr>
<tr>
<td>;INITialize:EXECute</td>
<td>Initializes settings.</td>
<td>3-43</td>
</tr>
<tr>
<td>;INITialize:UNDO</td>
<td>Undoes the initialization of settings.</td>
<td>3-43</td>
</tr>
<tr>
<td>;MEASURE?</td>
<td>Queries all settings related to the automated measurement of waveform parameters.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;??</td>
<td>Queries the On/Off state of all of the waveform parameters of the specified channel.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:AREA</td>
<td>Sets or queries the automatically measured waveform area for the waveform parameters.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:MODE (Count Range)</td>
<td>Sets or queries the auto measurement mode of the waveform parameter.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:METHod</td>
<td>Sets or queries the high/low point setting method.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:DPProximal?</td>
<td>Queries all settings related to distal, mesial, and proximal.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:DPProximal:MODE</td>
<td>Sets or queries the distal, mesial, and proximal point mode setting.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:DPProximal:PERCent</td>
<td>Sets or queries the distal, mesial, and proximal points as percentages.</td>
<td>3-44</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:DPProximal:UNIT</td>
<td>Sets or queries the distal, mesial, and proximal points.</td>
<td>3-45</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:&lt;Parameter&gt;??</td>
<td>Queries all settings related to the specified waveform parameter (measurement item).</td>
<td>3-45</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:&lt;Parameter&gt;:STATE</td>
<td>Sets or queries the on/off state of the measurement of the specified waveform parameter.</td>
<td>3-45</td>
</tr>
<tr>
<td>;MEASURE:CHANnel&lt;x&gt;:&lt;Parameter&gt;:VALue?</td>
<td>Queries the measured value of the specified waveform parameter.</td>
<td>3-45</td>
</tr>
<tr>
<td>;MEASURE:FILE?</td>
<td>Queries all settings related to the file format output data of automatic measurement results.</td>
<td>3-45</td>
</tr>
<tr>
<td>;MEASURE:FILE:TINFomation (Time Information)</td>
<td>Sets or queries the ON/OFF state of the addition of the trigger time information in the file format output data of automated measurement results.</td>
<td>3-45</td>
</tr>
<tr>
<td>;MEASURE:FILE:SEND?</td>
<td>Executes the file format output of the automatic measurement results.</td>
<td>3-46</td>
</tr>
<tr>
<td>;MONitor:ASENd?</td>
<td>Outputs the numeric monitor data (ASCII format) of all channels.</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:ASENd:CHANnel&lt;x&gt;?</td>
<td>Outputs the numeric monitor data (ASCII format) of the specified channel.</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:BITs:CHANnel&lt;x&gt;?</td>
<td>Queries the valid bit length of the specified channel.</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:BYTeorder</td>
<td>Sets or queries the transmission order when using word format of two bytes or more.</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:FORMAT:CHANnel&lt;x&gt;?</td>
<td>Sets or queries the format of the data to be transmitted .</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:OFFSET:CHANnel&lt;x&gt;?</td>
<td>Queries the offset value used to convert the numeric monitor data of the specified channel into physical values.</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:GAIN:CHANnel&lt;x&gt;?</td>
<td>Queries the gain value used to convert the numeric monitor data of the specified channel into physical values.</td>
<td>3-47</td>
</tr>
<tr>
<td>;MONitor:RANGE:CHANnel&lt;x&gt;?</td>
<td>Queries the range value used to convert the numeric monitor data of the specified channel into physical values.</td>
<td>3-48</td>
</tr>
<tr>
<td>;MONitor:SEND:{ALL</td>
<td>CHANnel&lt;x&gt;}?</td>
<td>Outputs the numeric monitor data in binary format.</td>
</tr>
<tr>
<td>;MONitor:VERBose</td>
<td>Sets or queries whether or not to add 'label' and 'unit' to the response format of MONitor:ASENd?.</td>
<td>3-48</td>
</tr>
<tr>
<td>;MONitor:LATCH:ASENd?</td>
<td>Outputs the numeric monitor data (ASCII format) of all channels at the latch.</td>
<td>3-48</td>
</tr>
<tr>
<td>;MONitor:LATCH:ASENd:CHANnel&lt;x&gt;?</td>
<td>Outputs the numeric monitor data (ASCII format) of the specified channels at the latch.</td>
<td>3-48</td>
</tr>
<tr>
<td>;MONitor:LATCH:EXECute</td>
<td>Latches the monitor data and the alarm data.</td>
<td>3-48</td>
</tr>
<tr>
<td>;MONitor:LATCH:SEND:{ALL</td>
<td>CHANnel&lt;x&gt;}?</td>
<td>Outputs the numeric monitor data at the latch.</td>
</tr>
<tr>
<td>;MONitor:LATCH:ALARm:{ALL</td>
<td>CHANnel&lt;x&gt;}?</td>
<td>Outputs the channel alarm data at the latch.</td>
</tr>
</tbody>
</table>
### 3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:MRECord?</td>
<td>Queries all settings related to automatic data recording.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:STARt</td>
<td>Starts automatic data recording.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:STOP</td>
<td>Aborts automatic data recording.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:DESTination</td>
<td>Sets or queries the destination of the record of the automatic data recording to the internal media.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:SCONdition (Start Condition)</td>
<td>Sets or queries the start condition of automatic data recording.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:STIME</td>
<td>Sets or queries the start time if the start condition is the clock time.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:ECOndition (End Condition)</td>
<td>Sets or queries the end condition of automatic data recording.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:ETIME</td>
<td>Sets or queries the end time if the end condition is the clock time.</td>
<td>3-49</td>
</tr>
<tr>
<td>:MRECord:INTerval?</td>
<td>Queries all settings related to the recording interval.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:INTerval:TIME</td>
<td>Sets or queries the recording interval if the recording interval mode is the TIME.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:INTerval:MODE</td>
<td>Sets or queries the recording interval mode of automatic data recording.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:COUNT</td>
<td>Sets or queries the number of recordings of automatic data recording.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:AREA?</td>
<td>Queries all settings related to the method of the recording on the disk.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:AREA:MODE</td>
<td>Sets or queries the method of recording (recording area) on the disk.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:AREA:NUMBER</td>
<td>Sets or queries the method of recording (circular file number) on the disk.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:ECLock?</td>
<td>Queries all settings related to recording when using the external sample clock.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:ECLock:COUNT</td>
<td>Sets or queries the recording counts when using the external sample clock.</td>
<td>3-50</td>
</tr>
<tr>
<td>:MRECord:ECLock:INTerval</td>
<td>Sets or queries the recording interval counts when using the external sample clock.</td>
<td>3-50</td>
</tr>
</tbody>
</table>

**MTRigger Group**

:MTTrigger | Activates manual triggering. | 3-51

**SELFtest Group**

:SELFtest:HDD:EXECute? | Executes the self-test of the internal HDD and outputs the results. | 3-52 |
| :SELFtest:HDDFormat | Formats the internal HDD. | 3-52 |

**SSTart Group**

:SSTart | Executes single start. | 3-53 |
| :SSTart? {<NRf> } | Executes single start and waits for the completion. | 3-53 |

**START Group**

:START | The START group is used to start waveform acquisition. | 3-54

**STATus Group**

:STATus? | Queries all settings related to the communication status function. | 3-55 |
| :STATus:CONDition? | Queries the contents of the condition register. | 3-55 |
| :STATus:EESE | Sets or queries the contents of the extended event enable register. | 3-55 |
| :STATus:EESR? | Queries the contents of the extended event register and clears the register. | 3-55 |
| :STATus:ERRor? | Queries the error code and message information. | 3-55 |
| :STATus:FILTER<x> | Sets or queries the transition filter. | 3-55 |
| :STATus:QENable | Sets or queries whether or not to store messages other than errors to the Error queue (ON/OFF). | 3-55 |
| :STATus:QMESSAGE | Sets or queries whether or not to attach message information to the response to the “STATus:ERRor?” query (ON/OFF). | 3-55 |

**STOP Group**

:STOP | Stops waveform acquisition. | 3-56

**SYSTem Group**

:SYSTem? | Queries all settings related to the system. | 3-57 |
| :SYSTem:CLK? | Queries all settings related to the date and time. | 3-57 |
3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:SYSTem:CLOCk:DATE</td>
<td>Sets or queries the date.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:CLOCk:TIME</td>
<td>Sets or queries the time.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:KEYLock</td>
<td>Sets or queries the ON/OFF state of the keylock of the main unit.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:LCD:BRIGHTness</td>
<td>Sets or queries the brightness of the LCD.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:LCD:CONTrast</td>
<td>Sets or queries the contrast of the LCD.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:LCD:MODE</td>
<td>Sets or queries the ON/OFF state of the LCD backlight.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:LCD:DTOut</td>
<td>Sets or queries the ON/OFF state of the function that brings the contents of the LCD back to the specified screen in 30 seconds.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:LCMD:MODe</td>
<td>Sets or queries the contents of LCD.</td>
<td>3-57</td>
</tr>
<tr>
<td>:SYSTem:RM:MODe</td>
<td>Sets or queries the measurement stop mode when the remote terminal is controlling START/STOP.</td>
<td>3-58</td>
</tr>
<tr>
<td>:SYSTem:STAtion:GNUMBER?</td>
<td>Queries the area group number of the unit.</td>
<td>3-58</td>
</tr>
<tr>
<td>:SYSTem:STAtion:M:MODe</td>
<td>x?&gt;</td>
<td>Returns the attributes of the specified slot.</td>
</tr>
<tr>
<td>:SYSTem:STAtion:N</td>
<td>Ume?</td>
<td>Queries the number of the unit.</td>
</tr>
<tr>
<td>:SYSTem:STAtion:NAME</td>
<td>Sets or queries the name of the unit.</td>
<td>3-58</td>
</tr>
<tr>
<td>:SYSTem:STAtion:GNAMe</td>
<td>Sets or queries the name of the group.</td>
<td>3-58</td>
</tr>
</tbody>
</table>

**TIMebase Group**

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:TIMebase?</td>
<td>Queries all settings related to the time base.</td>
<td>3-59</td>
</tr>
<tr>
<td>:TIMebase:M:GROUp</td>
<td>x&gt;?</td>
<td>Sets or queries the measurement group to which the specified module belongs.</td>
</tr>
<tr>
<td>:TIMebase:SOURCE</td>
<td>Sets or queries the time base.</td>
<td>3-59</td>
</tr>
<tr>
<td>:TIMebase:S:RATE</td>
<td>Sets or queries the sample rate of measurement group 1.</td>
<td>3-59</td>
</tr>
<tr>
<td>:TIMebase:GROUP&lt;x&gt;:S:RATE</td>
<td>Sets or queries the sample rate of measurement groups 2 to 4.</td>
<td>3-59</td>
</tr>
</tbody>
</table>

**TRIGger Group**

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:TRIGger?</td>
<td>Queries all settings related to triggers.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination?</td>
<td>Queries all settings related to the combination trigger class.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination:CHANnel&lt;x&gt;?</td>
<td>Queries the settings of the specified channel in the combination trigger class.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination:CHANnel&lt;x&gt;:HYSTeresis&lt;x2&gt;</td>
<td>Sets or queries the trigger hysteresis of the specified channel in the combination trigger class.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination:CHANnel1&lt;x1&gt;:LEVEL1&lt;x2&gt;</td>
<td>Sets or queries the trigger level of the specified channel in the combination trigger class.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination:CHANnel&lt;x&gt;:TYPE</td>
<td>Sets or queries the trigger type of the specified channel in the combination trigger class.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination:EXternal:TYPE</td>
<td>Sets or queries the external trigger type of the specified channel in the combination trigger class.</td>
<td>3-60</td>
</tr>
<tr>
<td>:TRIGger:COMBination:MODE</td>
<td>Sets or queries the combination mode of the combination trigger class.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:DElay</td>
<td>Sets or queries the delay (time from the trigger point to the trigger position).</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:R:OFF</td>
<td>Sets or queries the hold off time.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:M:TIME</td>
<td>Sets or queries the waveform acquisition mode for trigger measurement.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:POSition</td>
<td>Sets or queries the trigger position.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:[SIMple]:HYSTeresis</td>
<td>Sets or queries the hysteresis of the simple trigger.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:[SIMple]:LEVEL</td>
<td>Sets or queries the trigger level of the simple trigger of the specified channel.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:[SIMple]:SLOPe</td>
<td>Sets or queries the simple trigger type of the channel specified.</td>
<td>3-61</td>
</tr>
<tr>
<td>:TRIGger:S:SOURCE</td>
<td>Sets or queries the trigger source of the simple trigger.</td>
<td>3-62</td>
</tr>
<tr>
<td>:TRIGger:TYPE</td>
<td>Sets or queries all settings related to the timer trigger.</td>
<td>3-62</td>
</tr>
<tr>
<td>:TRIGger:DATE</td>
<td>Sets or queries the date of the timer trigger.</td>
<td>3-62</td>
</tr>
<tr>
<td>:TRIGger:IT:INTERval</td>
<td>Sets or queries the trigger time interval of the timer trigger.</td>
<td>3-62</td>
</tr>
<tr>
<td>:TRIGger:TIME</td>
<td>Sets or queries the time of the timer trigger.</td>
<td>3-62</td>
</tr>
</tbody>
</table>

**WAVEform Group**

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:WAVEform?</td>
<td>Queries all settings related to waveform data.</td>
<td>3-64</td>
</tr>
<tr>
<td>:WAVEform:A:COUNT</td>
<td>Queries the acquisition count.</td>
<td>3-64</td>
</tr>
</tbody>
</table>
### 3.2 A List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>:WAVEform:BITS?</td>
<td>Queries the bit length of the waveform data specified by 'WAVEform:TRACe'.</td>
<td>3-64</td>
</tr>
<tr>
<td>:WAVEform:BYTeorder</td>
<td>Sets or queries the byte order when using word format of two bytes or more.</td>
<td>3-64</td>
</tr>
<tr>
<td>:WAVEform:DIVision?</td>
<td>Queries the Division value used when converting the waveform data specified by 'WAVEform:TRACe' to physical values.</td>
<td>3-64</td>
</tr>
<tr>
<td>:WAVEform:END</td>
<td>Sets or queries the final data point of the waveform (main waveform) specified by 'WAVEform:TRACe'.</td>
<td>3-64</td>
</tr>
<tr>
<td>:WAVEform:FORMAT</td>
<td>Sets or queries the format of the data to be transmitted.</td>
<td>3-64</td>
</tr>
<tr>
<td>:WAVEform:GAIN?</td>
<td>Queries the gain value used when converting the waveform data specified by 'WAVEform:TRACe' to physical values.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:HMAX? (History MAX)</td>
<td>Queries the maximum number of history that can be acquired by the currently specified unit.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:LENGth?</td>
<td>Queries all waveform data points (main side) specified by 'WAVEform:TRACe'.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:MODule?</td>
<td>Queries the module corresponding to the waveform specified by</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:OFFSet?</td>
<td>Queries the offset value used when converting the waveform data specified by 'WAVEform:TRACe' to physical values.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:RANGe?</td>
<td>Queries the range value when converting the waveform data specified by 'WAVEform:TRACe' to physical values.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:RECORD</td>
<td>Sets or queries the target record number of the main waveform for the commands in the WAVEform group.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:RECORD? MINimum</td>
<td>Queries the minimum record number of the history (main waveform).</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:SEND?</td>
<td>Queries the waveform data (raw data) specified by 'WAVEform:TRACe'.</td>
<td>3-65</td>
</tr>
<tr>
<td>:WAVEform:SIGN?</td>
<td>Queries the existence of a sign when querying the waveform data specified by :WAVEform:TRACe using block data.</td>
<td>3-66</td>
</tr>
<tr>
<td>:WAVEform:SRATe?</td>
<td>Queries the sample rate of the record specified by :WAVEform:RECORD.</td>
<td>3-66</td>
</tr>
<tr>
<td>:WAVEform:START</td>
<td>Sets or queries the first data point of the waveform (main waveform) specified by 'WAVEform:TRACe'.</td>
<td>3-66</td>
</tr>
<tr>
<td>:WAVEform:TRACe</td>
<td>Sets or queries the target waveform.</td>
<td>3-66</td>
</tr>
<tr>
<td>:WAVEform:TRIGger?</td>
<td>Queries the trigger position of the record specified by :WAVEform:RECORD.</td>
<td>3-66</td>
</tr>
<tr>
<td>:WAVEform:TYPE?</td>
<td>Queries the acquisition mode of the waveform specified by :WAVEform:TRACe.</td>
<td>3-66</td>
</tr>
<tr>
<td>:WAVEform:GDELay?</td>
<td>The delay of ‘the measurement start point’ or ‘the trigger point’ of the waveform specified by 'WAVEform:TRACe' is returned.</td>
<td>3-66</td>
</tr>
</tbody>
</table>

#### Common Command Group

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>*CAL?</td>
<td>Performs calibration and queries the result.</td>
<td>3-67</td>
</tr>
<tr>
<td>*CLS</td>
<td>Clears the standard event register, extended event register, and error queue.</td>
<td>3-67</td>
</tr>
<tr>
<td>*ESE</td>
<td>Sets the standard event enable register or queries the current setting.</td>
<td>3-67</td>
</tr>
<tr>
<td>*ESR?</td>
<td>Queries the standard event register and clears the register.</td>
<td>3-67</td>
</tr>
<tr>
<td>*IDN?</td>
<td>Queries the instrument model.</td>
<td>3-67</td>
</tr>
<tr>
<td>*OPC</td>
<td>Sets a “1” to bit 0 (OPC bit) of the standard event register bit upon the completion of the specified overlap command.</td>
<td>3-67</td>
</tr>
<tr>
<td>*OPT?</td>
<td>The specified overlap command is completed, ASCII code ‘1’ is returned.</td>
<td>3-67</td>
</tr>
<tr>
<td>*RST?</td>
<td>Initializes the current settings.</td>
<td>3-67</td>
</tr>
<tr>
<td>*SRE</td>
<td>Sets or queries the service request enable register value.</td>
<td>3-68</td>
</tr>
<tr>
<td>*STB?</td>
<td>Queries the status byte register.</td>
<td>3-68</td>
</tr>
<tr>
<td>*TST?</td>
<td>Performs a self-test and queries the result.</td>
<td>3-68</td>
</tr>
<tr>
<td>*WAI</td>
<td>Holds the subsequent command until the completion of the specified overlap operation.</td>
<td>3-68</td>
</tr>
</tbody>
</table>
3.3 ACQuire Group

The ACQuire group deals with waveform acquisition.

**:ACQuire?**
- **Function**: Queries all settings related to waveform acquisition.
- **Syntax**: :ACQuire?

**:ACQuire:CLOCK**
- **Function**: Sets or queries the time base (internal/external clock).
- **Syntax**: :ACQuire:CLOCK {INTernal|EXTernal}
- **Example**: :ACQuire:CLOCK INTERNAL

**:ACQuire:COUNT**
- **Function**: Sets or queries the waveform acquisition count for normal mode.
- **Syntax**: :ACQuire:COUNT {<NRf>|INFinity}
- **Example**: :ACQuire:COUNT 2

**:ACQuire:ECLock? (Ext CLock)**
- **Function**: Sets or queries all settings related to the external sample clock.
- **Syntax**: :ACQuire:ECLock?

**:ACQuire:ECLock:PCOunt (Ext CLock Pretrigger COunt)**
- **Function**: Sets or queries the pre-trigger count when using the external sample clock.
- **Syntax**: :ACQuire:ECLock:PCOunt {<NRf>}
- **Example**: :ACQuire:ECLock:PCOunt 1000

**:ACQuire:ECLock:COUNt (Ext CLock COUNT)**
- **Function**: Sets or queries the sample count when using the external sample clock.
- **Syntax**: :ACQuire:ECLock:COUNt {<NRf>|MAX}
- **Example**: :ACQuire:ECLock:COUNt 1000

**:ACQuire:MMode (Motion MODe)**
- **Function**: Sets or queries the waveform acquisition operation mode.
- **Syntax**: :ACQuire:MMode {FREErun|TRIGGER}
- **Example**: :ACQuire:MMode TRIGGER

**:ACQuire:MODE**
- **Function**: Sets or queries the waveform acquisition mode.
- **Syntax**: :ACQuire:MODE {BAVerage|ENVelope|NORMal}
- **Example**: :ACQuire:MODE ENVELOPE

**:ACQuire:TIME**
- **Function**: Sets or queries the measurement time.
- **Syntax**: :ACQuire:TIME {<NRf>,<NRf>,<NRf>,<NRf>,<NRf>,<NRf>}
- **Example**: :ACQuire:TIME 0,0,0,0,500,0

**Description**
For the SL1000, the Single N count can also be set using this command

**:ACQuire:EClOck? (Ext CLock)**
- **Function**: Sets or queries all settings related to the external sample clock.
- **Syntax**: :ACQuire:EClOck?

**:ACQuire:EClOck:PCOunt (Ext CLock Pretrigger COunt)**
- **Function**: Sets or queries the pre-trigger count when using the external sample clock.
- **Syntax**: :ACQuire:EClOck:PCOunt {<NRf>}
- **Example**: :ACQuire:EClOck:PCOunt 1000

**:ACQuire:EClOck:COUNt (Ext CLock COUNT)**
- **Function**: Sets or queries the sample count when using the external sample clock.
- **Syntax**: :ACQuire:EClOck:COUNt {<NRf>|MAX}
- **Example**: :ACQuire:EClOck:COUNt 1000

**Example**: :ACQuire:EClOck:PCOunt 1000

**Example**: :ACQuire:EClOck:COUNt 1000
3.3 ACQuire Group

:ACQuire:MHNum?
Function Queries the maximum history number for trigger measurement mode.
Syntax :ACQuire:MHNum? {<NRf>}
<NRf> = measurement points
Example :ACQUIRE:MHNUM? 10000 -> ACQUIRE: MHNUM 5000
Description The function returns the maximum history number that is calculated from the measurement points and number of channels in use.
### 3.4 ALARM Group

The ALARM group deals with alarms (warnings).

- **:ALARm?**
  - **Function**: Queries all settings related to alarms.
  - **Syntax**: :ALARm?

- **:ALARm:ACK:EXECute**
  - **Function**: Clears alarm output.
  - **Syntax**: :ALARm:ACK:EXECute
  - **Example**: :ALARm:ACK:EXECute

- **:ALARm:ACCOUNT?**
  - **Function**: Queries the number of acquisitions, which is counted from the start of measurement, when an alarm occurs.
  - **Syntax**: :ALARm:ACCOUNT?
  - **Example**: :ALARm:ACCOUNT? -> :ALARm:ACCOUNT 46

- **:ALARm:COMBination**
  - **Function**: Sets or queries the AND/OR state of the alarms of each channel.
  - **Syntax**: :ALARm:COMBination {AND|OR}
  - **Example**: :ALARm:COMBination AND

- **:ALARm:CONDITION?**
  - **Function**: Queries the alarm output terminal condition.
  - **Syntax**: :ALARm:CONDITION?
  - **Example**: :ALARm:CONDITION? -> :ALARm:CONDITION 0
  - **Description**: If output is on, the command returns 1. If output is off, the command returns 0.

- **:ALARm:CHANnel<x>?**
  - **Function**: Queries all settings related to channel alarms.
  - **Syntax**: :ALARm:CHANnel<x>?
  - **Example**: :ALARm:CHANnel1:CONDITION? -> :ALARm:CHANnel1:CONDITION 0

- **:ALARm:CHANnel<x>:HYSTeresis<x2>**
  - **Function**: Sets or queries the alarm hysteresis of a channel.
  - **Syntax**: :ALARm:CHANnel<x>:HYSTeresis<x2> {HIGH|LOW|MIDDle}
    - Example**: :ALARm:CHANnel1:HYSTeresis1 HIGH

- **:ALARm:CHANnel<x>:LEVEL<x2>**
  - **Function**: Sets or queries the alarm level of a channel (when the input of the specified channel is not logic).
  - **Syntax**: :ALARm:CHANnel<x>:LEVEL<x2> {<Voltage>|<NRf>|<Current>}
    - Example**: :ALARm:CHANnel1:LEVEL1 10
  - **Description**: The Au7Fe temperature measuring range is 0 to 280 K (-273 to 7°C).

- **:ALARm:CHANnel<x>:TYPE**
  - **Function**: Sets or queries the alarm type of a channel.
  - **Syntax**: :ALARm:CHANnel<x>:TYPE {HIGH|LOW|OFF|WLIn|WLOut}
    - Example**: :ALARm:CHANnel1:TYPE HIGH
### 3.4 ALARm Group

**:ALARm:CHANnel<x>:AVALue?**

**Function**
Queries the measured value at the alarm occurrence on the specified channels as an ASCII string.

**Syntax**
:ALARm:CHANnel<x>:AVALue?

**Example**
$:ALARm:CHANnel1:AVALue? 
$ \rightarrow :ALARm:CHANnel1:AVALue "CH1 1.022V"

**Description**
Output format is the same as the ASCII output format for the 'Monitor group'.

---

**:ALARm:SMODe**

**Function**
Sets or queries the system alarm operation mode.

**Syntax**
:ALARm:SMODe {<Boolean>}

**Example**
:ALARm:SMODe ON
:ALARm:SMODe? \rightarrow :ALARm:SMODe 1

**Description**
Sets whether to detect or not detect the system alarm.

---

**:ALARm:OTERminal**

**Function**
Sets or queries the alarm output terminal on/off state.

**Syntax**
:ALARm:OTERminal {<Boolean>}

**Example**
:ALARm:OTERminal ON
:ALARm:OTERminal? \rightarrow :ALARm:OTERminal 1

**Description**
If the alarm output terminal is switched off, it will remain off regardless of the alarm settings or conditions.

---

**:ALARm:SOURce**

**Function**
Sets or queries the alarm detection source.

**Syntax**
:ALARm:SOURce {CHANnel|SYSTem}

**Example**
:ALARm:SOURce CHANNEL
:ALARm:SOURce? \rightarrow :ALARm:SOURce CHANNEL

---

**:ALARm:STATus?**

**Function**
Queries the channel alarm status.

**Syntax**
:ALARm:STATus?

**Example**
When the alarm condition is ON at CH1 and CH3.
$:ALARm:STATus? \rightarrow :ALARm:STATus 40960$

**Description**
Since the :ALARm:CHANnel<x>:CONDition? command must be used repeatedly to acquire the alarm statuses of all channels, this command returns a bit pattern. The MSB of the returned value (bit 15) represents channel 1. The LSB (bit 0) represents channel 16.

---

**:ALARm:CHANnel<x>:CONDition?**

**Function**
Queries the channel alarm condition.

**Syntax**
:ALARm:CHANnel<x>:CONDition?

**Example**
$:ALARm:CHANnel1:CONDition? \rightarrow :ALARm:CHANnel1:CONDition 0$

**Description**
Sets whether to detect or not detect the channel alarm.
3.4 ALARM Group

:ALARM:SSTATUS?
Function Queries the system alarm status value.
Syntax :ALARM:SSTATUS?
Description The system status bit assignments are shown in the table below.

<table>
<thead>
<tr>
<th>bit</th>
<th>symbol</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HDD_FULL</td>
<td>HDD disk Full</td>
</tr>
<tr>
<td>2</td>
<td>FUN_STOP</td>
<td>Fun stop</td>
</tr>
<tr>
<td>3</td>
<td>BUF_OVERRUN_PP</td>
<td>Real time record index data full</td>
</tr>
<tr>
<td>4</td>
<td>BUF_OVERRUN_HOST</td>
<td>Host (PC) buffer overrun</td>
</tr>
<tr>
<td>5</td>
<td>BUF_OVERRUN_UNIT</td>
<td>SL1000 buffer overrun</td>
</tr>
<tr>
<td>6</td>
<td>(system reserved)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>(reserved)</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>(reserved)</td>
<td></td>
</tr>
</tbody>
</table>

:ALARM:SYSTem?
Function Queries all settings related to the system alarm.
Syntax :ALARM:SYSTem?

:ALARM:SYSTem:SOURce?
Function Queries all settings related to system alarm detection.
Syntax :ALARM:SYSTem:SOURce?

:ALARM:SYSTem:SOURce:BOVerrun
Function Sets or queries system alarm buffer overrun detection.
Syntax :ALARM:SYSTem:SOURce:BOVerrun {<Boolean>}

:ALARM:SYSTem:SOURce:FSTop
Function Sets or queries system alarm fan stop detection.
Syntax :ALARM:SYSTem:SOURce:FSTop {<Boolean>}

:ALARM:SYSTem:SOURce:DFULL
Function Sets or queries system alarm HDD full detection.
Syntax :ALARM:SYSTem:SOURce:DFULL {<Boolean>}

:ALARM:SYSTem:SOURce:MRTime
Function Sets or queries the maximum recording time detection for free-run automatic recording mode.
Syntax :ALARM:SYSTem:SOURce:MRTime {<Boolean>}

:ALARM:STIME?
Function Queries the time of the most recent system alarm condition change.
Syntax :ALARM:STIME?
Example :ALARM:STIME? -> :ALARM:STIME 2008,7,10,10,14,8,300
Description The function returns the time of the system alarm condition change in order of year/month/day/hour/minute/second/microsecond. If no status changes occur after measurement starts, the function returns the time when measurement started. If measurement has not started, the function returns an undefined value.
3.5 ASETup Group

The ASETup group deals with auto setup.

:ASETup?
Function Queries all settings related to auto setup.
Syntax :ASETup?

:ASETup:EXECute
Function Execute auto setup.
Syntax :ASETup:EXECute
Example :ASETUP:EXECUTE
Description When the measurement mode is free-run, only
the settings related to the vertical axes are set up
automatically.
Attention Do not send other commands while executing auto
setup.

:ASETup:TARGet
Function Sets or queries the target channel for auto setup.
Syntax :ASETup:TARGet \{ALL|<NRf>\}
 Example :ASETUP:TARGET 1
 Description An error occurs if you specify a channel in which a
module is not installed.

:ASETup:UNDO
Function Cancels auto setup.
Syntax :ASETup:UNDO
Example :ASETUP:UNDO
Attention Do not send other commands while canceling auto
setup. The processing status can be inferred from Bit
9 of the status register.
3.6 CALibrate Group

The CALibrate group deals with calibration.

::CALibrate?
Function: Queries all settings related to calibration.
Syntax: ::CALibrate?

::CALibrate[:EXECute]
Function: Executes calibration.
Syntax: ::CALibrate:EXECute
Example: ::CALIBRATE:EXECUTE
Description: Unlike the common command "CAL2?, the function will not return the result at the end of calibration.

::CALibrate:MODE
Function: Sets or queries the ON/OFF state of auto calibration.
Syntax: ::CALibrate:MODE {AUTO|OFF}
Example: ::CALIBRATE:MODE AUTO
         ::CALIBRATE:MODE? -> ::CALIBRATE: MODE AUTO
3.7 CHANnel Group

The CHANnel group deals with the vertical axis of each channel.

:CHANnel<x>?
Function Queries all settings related to the vertical axis of the channel.
Syntax :CHANnel<x>?
\(<x>\)=1 to 16
Description An error occurs if a module is not installed in the channel (slot).

:CHANnel<x>:ACCL?
Function Queries all settings on the channel with the acceleration/voltage module installed.
Syntax :CHANnel<x>:ACCL?
\(<x>\)=1 to 16
Description An error occurs if the acceleration/voltage module is not installed.

:CHANnel<x>:ACCL:BIAS
Function Sets or queries the on/off status of the acceleration sensor’s bias current when an Acceleration/Voltage Module is installed in the specified channel (slot).
Syntax :CHANnel<x>:ACCL:BIAS {<Boolean>}
:CHANnel<x>:ACCL:BIAS?
\(<x>\)=1 to 16
Example :CHANNEL1:ACCL:BIAS ON
Description An error occurs if an Acceleration/Voltage Module is not installed.

:CHANnel<x>:ACCL:BWIDTH
Function Sets or queries the filter when an Acceleration Voltage Module is installed in the specified channel (slot).
Syntax :CHANnel<x>:ACCL:BWIDTH {FULL|AUTO|<Frequency>}
:CHANnel<x>:ACCL:BWIDTH?
\(<x>\)=1 to 16
\(<Frequency>\)= 4 kHz, 400 Hz, or 40 Hz
Example :CHANNEL1:ACCL:BWIDTH AUTO
Description An error occurs if an Acceleration/Voltage Module is not installed.

:CHANnel<x>:ACCL:COUPling
Function Sets or queries input coupling when an Acceleration/ Voltage Module is installed in the specified channel (slot).
Syntax :CHANnel<x>:ACCL:COUPling {AC|DC|ACCL|GND}
:CHANnel<x>:ACCL:COUPling?
\(<x>\)=1 to 16
Example :CHANNEL1:ACCL:COUPling GND
Description An error occurs if an Acceleration/Voltage Module is not installed.

:CHANnel<x>:ACCL:GAIN
Function Sets or queries the gain when an Acceleration/ Voltage Module is installed in the specified channel (slot).
Syntax :CHANnel<x>:ACCL:GAIN {<NRf>}
:CHANnel<x>:ACCL:GAIN?
\(<x>\)=1 to 16
\(<NRf>\)=0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100
Example :CHANNEL1:ACCL:GAIN 100
:CHANNEL1:ACCL:GAIN? -> :CHANNEL1:ACCL:GAIN 100.0
Description An error occurs if an Acceleration/Voltage Module is not installed.

:CHANnel<x>:ACCL:SENSitivity
Function Sets or queries the sensitivity when an Acceleration/ Voltage Module is installed in the specified channel (slot).
Syntax :CHANnel<x>:ACCL:SENSitivity {<NRf>}
:CHANnel<x>:ACCL:SENSitivity?
\(<x>\)=1 to 16
\(<NRf>\)=0.1 to 2000
Example :CHANNEL1:ACCL:SENSitivity 10
Description An error occurs if an Acceleration/Voltage Module is not installed.
3.7 CHANnel Group

:CHANnel<x>:ACCL:UNIT
Function Sets or queries the unit of measurement of the upper and lower limit values when an Acceleration/Voltage Module is installed in the specified channel(slot).
Syntax :CHANnel<x>:ACCL:UNIT {< String >}
:CHANnel<x>:ACCL:UNIT?
<x>=1 to 16
Example :CHANNEL1:ACCL:UNIT "ACCL"
Description An error occurs if an Acceleration/Voltage Module is not installed.

:CHANnel<x>:ASET?
Function Queries whether the specified channel is able to set an auto setup or not.
Syntax :CHANnel<x>:ASET?
<x>=1 to 16
Example :CHANNEL1:ASET? -> :CHANNEL1:ASET 1
Description If auto setup is able to set, the function returns 1. If auto setup is impossible to set, the function returns 0. An error occurs if the module is not installed in the specified channel (slot).

:CHANnel<x>:DISPLAY
Function Sets or queries the channel acquisition ON/OFF state.
Syntax :CHANnel<x>:DISPLAY {<Boolean>}
:CHANnel<x>:DISPLAY?
<x>=1 to 16
Example :CHANNEL1:DISPLAY ON
:CHANNEL1:DISPLAY? -> :CHANNEL1:DISPLAY 1
Description The default value of this command is OFF. Please set to ON first to use the channel. An error occurs if the module is not installed in the specified channel (slot).

:CHANnel<x>:FREQ?
Function Queries all settings when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ?
<x>=1 to 16
Description An error occurs if the frequency module is not installed.

:CHANnel<x>:FREQ:INPut?
Function Queries all settings related to the input when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:INPut?
<x>=1 to 16
Description An error occurs if the frequency module is not installed.

:CHANnel<x>:FREQ:INPut:BWIDth
Function Sets or queries the bandwidth limit when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:INPut:BWIDth {FULL|< Frequency >}
:CHANnel<x>:FREQ:INPut:BWIDth?
<x>=1 to 16
< Frequency >= 100 Hz, 1 kHz, 10 kHz, 100 kHz
Example :CHANNEL1:FREQ:INPut:BWIDTH FULL
:CHANNEL1:FREQ:INPut:BWIDTH?
-> :CHANNEL1:FREQ:INPut:BWIDTH FULL
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:CELimination
Function Sets or queries chattering elimination when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:INPut:CELimination {< Time >}
:CHANnel<x>:FREQ:INPut:CELimination?
<x>=1 to 16
< Time >=0 to 1000 ms
Example :CHANNEL1:FREQ:INPut:CELIMINATION 100ms
:CHANNEL1:FREQ:INPut:CELIMINATION?
-> :CHANNEL1:FREQ:INPut:CELIMINATION 0.100
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:COUPling
Function Sets or queries input coupling when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:INPut:COUPling {AC|DC}
:CHANnel<x>:FREQ:INPut:COUPling?
<x>=1 to 16
Example :CHANNEL1:FREQ:INPut:COUPLING DC
:CHANNEL1:FREQ:INPut:COUPLing?
-> :CHANNEL1:FREQ:INPut:COUPLING DC
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:HYSTeresis
Function Sets or queries hysteresis when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:INPut:HYSTeresis {HIGH|LOW|MIDDle}
:CHANnel<x>:FREQ:INPut:HYSTeresis?
<x>=1 to 16
Example :CHANNEL1:FREQ:INPut:HYSTERESIS LOW
:CHANNEL1:FREQ:INPut:HYSTERESIS?
-> :CHANNEL1:FREQ:INPut:HYSTERESIS LOW
Description An error occurs if a frequency module is not installed.
3.7 CHANnel Group

:CHANnel<x>:FREQ:INPut:PRESet
Function Sets or queries the preset when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:INPut:PRESet {AC100v|AC200v|EMPlchup|LOG12v|LOG24v|LOG3v|LOG5v|PULLup|USER|ZERO}
:CHANnel<x>:FREQ:INPut:PRESet?
Example  :CHANNEL1:FREQ:INPut:PRESET USER
       :CHANNEL1:FREQ:INPut:PRESET?
-> :CHANNEL1:FREQ:INPut:PRESET USER
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:PROBe
Function Sets or queries the probe attenuation when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:INPut:PROBe {<NRf>}
:CHANnel<x>:FREQ:INPut:PROBe?
Example  :CHANNEL1:FREQ:INPut:PROBE 10
       :CHANNEL1:FREQ:INPut:PROBE?
-> :CHANNEL1:FREQ:INPut:PROBE 10
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:PULLup
Function Sets or queries the pull-up on/off state when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:INPut:PULLup {<Boolean>}
:CHANnel<x>:FREQ:INPut:PULLup?
Example  :CHANNEL1:FREQ:INPut:PULLUP ON
       :CHANNEL1:FREQ:INPut:PULLUP?
-> :CHANNEL1:FREQ:INPut:PULLUP 1
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:SLOPe
Function Sets or queries the slope when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:INPut:SLOPe {FALL|RISE}
:CHANnel<x>:FREQ:INPut:SLOPe?
Example  :CHANNEL1:FREQ:INPut:SLOPe FALL
       :CHANNEL1:FREQ:INPut:SLOPe?
-> :CHANNEL1:FREQ:INPut:SLOPe FALL
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:THReshold
Function Sets or queries the threshold level when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:INPut:THReshold {< Voltage >}
:CHANnel<x>:FREQ:INPut:THReshold?
Example  :CHANNEL1:FREQ:INPut:THRESHOLD 10
       :CHANNEL1:FREQ:INPut:THRESHOLD?
-> :CHANNEL1:FREQ:INPut:THRESHOLD 10.000E+00
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:INPut:VRANGe
Function Sets or queries the voltage range when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:INPut:VRANGe {< Voltage >}
:CHANnel<x>:FREQ:INPut:VRANGe?
Example  :CHANNEL1:FREQ:INPut:VRANGE 10
       :CHANNEL1:FREQ:INPut:VRANGE?
-> :CHANNEL1:FREQ:INPut:VRANGE 10
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:LSCale?
Function Queries all settings related to linear scaling when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:LSCale?
Example  :CHANNEL1:FREQ:LSCALE?
       :CHANNEL1:FREQ:LSCALE?
-> :CHANNEL1:FREQ:LSCALE 10.000E+00
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:LSCale:AVALue
Function Sets or queries linear scaling coefficient A when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:LSCale:AVALue {<NRf>}
:CHANnel<x>:FREQ:LSCale:AVALue?
Example  :CHANNEL1:FREQ:LSCALE:AVALUE 10
       :CHANNEL1:FREQ:LSCALE:AVALUE?
-> :CHANNEL1:FREQ:LSCALE:AVALUE 10.000E+00
Description An error occurs if a frequency module is not installed.
3.7 CHANnel Group

:CHANnel<x>:FREQ:LSCale:BVALue
Function Sets or queries linear scaling coefficient B when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:LSCale:BVALue {<NRf>}
:CHANnel<x>:FREQ:LSCale:BVALue?
<x>=1 to 16
<NRf>=-9.9999E+30 to 9.9999E+30
Example :CHANNEL1:FREQ:LSCale:BVALUE 10
:CHANNEL1:FREQ:LSCale:BVALUE?
-> :CHANNEL1:FREQ:LSCale:BVALUE 10.0000E+00
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:LSCale:GETMeasure
Function Measures the X values of P1 and P2 for linear scaling when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:LSCale:GETMeasure {P1X|P2X}
<x>=1 to 16
Example :CHANNEL1:FREQ:LSCale:GETMEASURE P1X
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:LSCale:MODE
Function Sets or queries linear scaling when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:LSCale:MODE {AXB|OFF|P12}
:CHANnel<x>:FREQ:LSCale:MODE?
<x>=1 to 16
Example :CHANNEL1:FREQ:LSCale:MODE OFF
:CHANNEL1:FREQ:LSCale:MODE?
-> :CHANNEL1:FREQ:LSCale:MODE OFF
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:LSCale:{P1X|P1Y|P2X|P2Y}
Function Sets or queries the X or Y value of P1 or P2 for linear scaling when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:LSCale:{P1X|P1Y|P2X|P2Y} {<NRf>}
:CHANnel<x>:FREQ:LSCale:{P1X|P1Y|P2X|P2Y}?
<x>=1 to 16
For P1X and P2X, <NRf> = ?9.9999E+30 to 9.9999E+30
For P1Y and P2Y, <NRf> = -9.9999E+25 to 9.9999E+25
Example :CHANNEL1:FREQ:LSCale:P1X 10
:CHANNEL1:FREQ:LSCale:P1X?
-> :CHANNEL1:FREQ:LSCale:P1X 10.0000E+00
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:LSCale:UNIT
Function Sets or queries the unit of measurement to attach to the result of linear scaling when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:LSCale:UNIT {< String >}
:CHANnel<x>:FREQ:LSCale:UNIT?
<x>=1 to 16
<String>= Up to 4 characters
Example :CHANNEL1:FREQ:LSCale:UNIT "AAA"
:CHANNEL1:FREQ:LSCale:UNIT?
-> :CHANNEL1:FREQ:LSCale:UNIT "AAA"
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:OFFSET
Function Sets or queries the offset value when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:OFFSET {<NRf>|< Frequency >|< Time >}
:CHANnel<x>:FREQ:OFFSET?
<x>=1 to 16
Example :CHANNEL1:FREQ:OFFSET 1
:CHANNEL1:FREQ:OFFSET?
-> :CHANNEL1:FREQ:OFFSET 1.000000E+00
Description An error occurs if a frequency module is not installed.

Offset Range
Function Max Resolution

:CHANnel<x>:FREQ:SETup?
Function Queries all settings related to FV setup when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:SETup?
<x>=1 to 16
Description An error occurs if a frequency module is not installed.
3.7 CHANnel Group

:CHANnel<x>:FREQ:SETup:CFRequency
Function  Sets or queries the center frequency when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:CFRequency (< Frequency >)
:CHANnel<x>:FREQ:SETup:CFRequency?
<x>=1 to 16
Example  :CHANNEL1:FREQ:SETUP:CFREQUENCY 50
:CHANNEL1:FREQ:SETUP:CFREQUENCY?
-> :CHANNEL1:FREQ:SETUP:CFREQUENCY 50
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:DECeleration
Function  Sets or queries the on/off state of deceleration prediction when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:DECeleration (<Boolean>)
:CHANnel<x>:FREQ:SETup:DECeleration?
<x>=1 to 16
Example  :CHANNEL1:FREQ:SETUP:DECELERATION ON
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:DPULse
Function  Sets or queries the distance per pulse when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:DPULse (<NRf>)
:CHANnel<x>:FREQ:SETup:DPULse?
<x>=1 to 16
<NRf>= -9.9999E+30 to 9.9999E+30
Example  :CHANNEL1:FREQ:SETUP:DPULSE 1e15
:CHANNEL1:FREQ:SETUP:DPULSE?
-> :CHANNEL1:FREQ:SETUP:DPULSE 1.00000E+15
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FILTer?
Function  Queries all settings related to the filter when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:FILTer?
<x>=1 to 16
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FILTer:SMOothin
Function  Queries all settings related to smoothing when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:FILTer:SMOothing?
<x>=1 to 16
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FILTer:SMOothin:MODE
Function  Sets or queries the on/off state of smoothing when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:FILTer:SMOothing
:MODE {<Boolean>}
:CHANnel<x>:FREQ:SETup:FILTer:SMOothing
:MODE?
<x>=1 to 16
Example  :CHANNEL1:FREQ:SETUP:FILTER:SMOOTHING:MODE ON
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FILTer:SMOothin:VALue
Function  Sets or queries the moving average order of smoothing when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:FILTer:SMOothing
:VALue (< Time >)
:CHANnel<x>:FREQ:SETup:FILTer:SMOothing
:VALue?
<x>=1 to 16
< Time >= 0 to 1000
Example  :CHANNEL1:FREQ:SETUP:FILTER:SMOOTHING:VALUE 10ms
:CHANNEL1:FREQ:SETUP:FILTER:SMOOTHING:VALUE?
-> :CHANNEL1:FREQ:SETUP:FILTER:SMOOTHING:VALUE 0.0100
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FILTer:PAVera
Function  Queries all settings related to pulse average when a frequency module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:FREQ:SETup:FILTer:PAVera?
<x>=1 to 16
Description An error occurs if a frequency module is not installed.
3.7 CHANnel Group

:CHANnel<x>:FREQ:SETup:FILTER:PAverage:MODE
Function Sets or queries the on/off state of pulse average mode when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:SETup:FILTER:PAverage:MODE {<Boolean>}
:CHANnel<x>:FREQ:SETup:FILTER:PAverage:MODE?
<x>=1 to 16
Example :CHANNEL1:FREQ:SETUP:FILTER:PAVERAGE:MODE ON
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FILTER:PAverage:VALUE
Function Sets or queries the pulse average count when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:SETup:FILTER:PAverage:VALUE {<NRf>}
:CHANnel<x>:FREQ:SETup:FILTER:PAverage:VALUE?
<x>=1 to 16
<NRf>=1 to 4096
Example :CHANNEL1:FREQ:SETUP:FILTER:PAVERAGE:VALUE 10
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:FUNCTION
Function Sets or queries the measuring mode when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:SETup:FUNCTION {DUTY|FREQuency|PERiod|PFReq|PINTeg|PWIDth|RPM|RPS|VELocity}
:CHANnel<x>:FREQ:SETup:FUNCTION?
<x>=1 to 16
Example :CHANNEL1:FREQ:SETUP:FUNCTION FREQUENCY
:CHANNEL1:FREQ:SETUP:FUNCTION?
-> :CHANNEL1:FREQ:SETUP:FUNCTION FREQUENCY
Description An error occurs if a frequency module is not installed.

Setting Command (Parameter)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>FREQuency</td>
</tr>
<tr>
<td>Revolution (rpm)</td>
<td>RPM</td>
</tr>
<tr>
<td>Revolution (rps)</td>
<td>RPS</td>
</tr>
<tr>
<td>Period</td>
<td>PERiod</td>
</tr>
<tr>
<td>Duty</td>
<td>DUTY</td>
</tr>
<tr>
<td>Power Freq.</td>
<td>PFReq</td>
</tr>
<tr>
<td>Pulse Width</td>
<td>PWIDth</td>
</tr>
<tr>
<td>Pulse Integ</td>
<td>PINTeg</td>
</tr>
<tr>
<td>Velocity</td>
<td>VELocity</td>
</tr>
</tbody>
</table>

:CHANnel<x>:FREQ:SETup:LRESET
Function Sets or queries the over limit reset when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:SETup:LRESET {<Boolean>}
:CHANnel<x>:FREQ:SETup:LRESET?
<x>=1 to 16
Example :CHANNEL1:FREQ:SETUP:LRESET ON
:CHANNEL1:FREQ:SETUP:LRESET?
-> :CHANNEL1:FREQ:SETUP:LRESET 1
Description An error occurs if a frequency module is not installed.

:CHANnel<x>:FREQ:SETup:MPULSE
Function Sets or queries whether the measurement pulse is positive or negative when a frequency module is installed in the specified channel (slot).
Syntax :CHANnel<x>:FREQ:SETup:MPULSE {POSitive|NEGative}
:CHANnel<x>:FREQ:SETup:MPULSE?
<x>=1 to 16
Example :CHANNEL1:FREQ:SETUP:MPULSE POSITIVE
:CHANNEL1:FREQ:SETUP:MPULSE?
-> :CHANNEL1:FREQ:SETUP:MPULSE POSITIVE
Description An error occurs if a frequency module is not installed.
### 3.7 CHANnel Group

:**CHANnel<x>:FREQ:SETup:PROTate**

**Function**
Sets or queries the number of pulses per rotation when a frequency module is installed in the specified channel (slot).

**Syntax**
:CHANnel<x>:FREQ:SETup:PROTate {<NRf>}
:CHANnel<x>:FREQ:SETup:PROTate?

`<x>` = 1 to 16
`<NRf>` = 1 to 99999

**Example**
:CHANNeL1:FREQ:SETUP:PROTATE 10
:CHANNeL1:FREQ:SETUP:PROTATE?

**Description**
An error occurs if a frequency module is not installed.

:**CHANnel<x>:FREQ:SETup:RESet**

**Function**
 Resets the pulse count when a frequency module is installed in the specified channel (slot)

**Syntax**
:CHANnel<x>:FREQ:SETup:RESet

`<x>` = 1 to 16

**Example**
:CHANNeL1:FREQ:SETUP:RESET

**Description**
An error occurs if a frequency module is not installed.

:**CHANnel<x>:FREQ:SETup:STOPpredict**

**Function**
 Sets or queries the on/off state of stop prediction when a frequency module is installed in the specified channel (slot).

**Syntax**
:CHANnel<x>:FREQ:SETup:STOPpredict {<NRf>|OFF}
:CHANnel<x>:FREQ:SETup:STOPpredict?

`<x>` = 1 to 16
`<NRf>` = 1.5, 2, 3, 4, 5, 6, 7, 8, 9, 10

**Example**
:CHANNeL1:FREQ:SETUP:STOPPREDICT OFF
:CHANNeL1:FREQ:SETUP:STOPPREDICT?

**Description**
An error occurs if a frequency module is not installed.

:**CHANnel<x>:FREQ:SETup:TUNit**

**Function**
 Sets or queries the time unit when a frequency module is installed in the specified channel (slot).

**Syntax**
:CHANnel<x>:FREQ:SETup:TUNit {HOUR|MIN|SEC}
:CHANnel<x>:FREQ:SETup:TUNit?

`<x>` = 1 to 16

**Example**
:CHANNeL1:FREQ:SETUP:TUNIT SEC
:CHANNeL1:FREQ:SETUP:TUNIT?

**Description**
An error occurs if a frequency module is not installed.

:**CHANnel<x>:FREQ:SETup:UPULse**

**Function**
 Sets or queries the unit/pulse when a frequency module is installed in the specified channel (slot).

**Syntax**
:CHANnel<x>:FREQ:SETup:UPULse {<NRf>}
:CHANnel<x>:FREQ:SETup:UPULse?

`<x>` = 1 to 16
`<NRf>` = -9.9999E+30 to 9.9999E+30

**Example**
:CHANNeL1:FREQ:SETUP:UPULSE 1e15
:CHANNeL1:FREQ:SETUP:UPULSE?

**Description**
An error occurs if a frequency module is not installed.

:**CHANnel<x>:FREQ:SETup:VUNit**

**Function**
 Sets or queries the unit of velocity when a frequency module is installed in the specified channel (slot).

**Syntax**
:CHANnel<x>:FREQ:SETup:VUNit {< string >}
:CHANnel<x>:FREQ:SETup:VUNit?

`<x>` = 1 to 16
`< string >` = Up to 4 characters

**Example**
:CHANNeL1:FREQ:SETUP:VUNIT "BBB"
:CHANNeL1:FREQ:SETUP:VUNIT?

**Description**
An error occurs if a frequency module is not installed.

:**CHANnel<x>:FREQ:VDIV**

**Function**
 Sets or queries the Value/Div when a frequency module is installed in the specified channel (slot).

**Syntax**
:CHANnel<x>:FREQ:VDIV {<NRf>|< Frequency >|< Time >}
:CHANnel<x>:FREQ:VDIV?

`<x>` = 1 to 16
`{<NRf>|< Frequency >|< Time >}` = See the SL1000 High Speed Data Acquisition Unit User's Manual for details

**Example**
:CHANNeL1:FREQ:VDIV 10
:CHANNeL1:FREQ:VDIV? ->
:CHANNeL1:FREQ:VDIV 10.0000E+00

**Description**
An error occurs if a frequency module is not installed.
3.7 CHANnel Group

:CHANnel<x>:LABel
Function  Sets or queries the waveform label of the specified channel.
Syntax  :CHANnel<x>:LABel {< String >}
        :CHANnel<x>:LABel?
        <x>=1 to 16
        < String >= Up to 8 characters
Example  :CHANNEL1:LABEL "TRACE1"
Description An error occurs if a module is not installed in the channel (slot).

:CHANnel<x>:MODule?
Function  Queries the module installed in the channel (slot).
Syntax  :CHANnel<x>:MODule?
        <x>=1 to 16
Example  :CHANNEL1:MODULE? -> :CHANNEL1:MODULE M701250
Description The following values are returned from the module.
<table>
<thead>
<tr>
<th>String</th>
<th>Model No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMODULE</td>
<td>No module</td>
</tr>
<tr>
<td>M720210</td>
<td>720210(HS100M12)</td>
</tr>
<tr>
<td>M720211</td>
<td>720211(HS100M12)</td>
</tr>
<tr>
<td>M701250</td>
<td>701250(HS10M12)</td>
</tr>
<tr>
<td>M720250</td>
<td>720250(HS10M12)</td>
</tr>
<tr>
<td>M701251</td>
<td>701251(HS1M16)</td>
</tr>
<tr>
<td>M701255</td>
<td>701255(NONISO_10M12)</td>
</tr>
<tr>
<td>M701260</td>
<td>701260(1000V)</td>
</tr>
<tr>
<td>M701268</td>
<td>701268(HV(with RMS))</td>
</tr>
<tr>
<td>M701261</td>
<td>701261(100V)</td>
</tr>
<tr>
<td>M701262</td>
<td>701262(UNIV_AAF)</td>
</tr>
<tr>
<td>M701265</td>
<td>701265(TEMP/HPV)</td>
</tr>
<tr>
<td>M702256</td>
<td>702256(TEMP/HPV)</td>
</tr>
<tr>
<td>M701270</td>
<td>701270(TEMP/HPV)</td>
</tr>
<tr>
<td>M701271</td>
<td>701271(TEMP/DSUB)</td>
</tr>
<tr>
<td>M701275</td>
<td>701275(STRAIN_NDIS)</td>
</tr>
<tr>
<td>M701280</td>
<td>701280(FREQ)</td>
</tr>
<tr>
<td>M701281</td>
<td>701281(FREQ)</td>
</tr>
<tr>
<td>M702281</td>
<td>702281(FREQ)</td>
</tr>
</tbody>
</table>

*M701260* is returned also for the 701267 module.

:CHANnel<x>:RECode
Function  Sets or queries whether to record the specified channel or not.
Syntax  :CHANnel<x>:RECode {<Boolean>}
        :CHANnel<x>:RECode?
        <x>=1 to 16
Example  :CHANNEL1:RECORD ON
Description Specify automatic recording, and the target channel for file saving.

:CHANnel<x>:STRain?
Function  Queries all settings related to the Strain Module when a strain module is installed in the specified channel (slot).
Syntax  :CHANnel<x>:STRain?
        <x>=1 to 16
Example  :CHANNEL1:STRAIN?
Description An error occurs if a strain module is not installed.
3.7 CHANnel Group

:CHANnel<x>:STRAIN:EXCitation
Function Sets or queries the bridge voltage when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRAIN:EXCitation {< Voltage >}
<Voltage> = 2 V, 5 V, 10 V
Example :CHANNEL1:STRAIN:EXCITATION 2V
> :CHANNEL1:STRAIN:EXCITATION 2.000000E+00
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:GFACtor
Function Sets or queries the gauge factor when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRAIN:GFACtor {<NRf>}
<NRF> = 1.90 to 2.20
Example :CHANNEL1:STRAIN:GFACtor 2.00
> :CHANNEL1:STRAIN:GFACtor 2.00
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:INVert
Function Sets or queries whether or not the display is inverted when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRAIN:INVert {<Boolean>}
<Boolean> = ON, OFF
Example :CHANNEL1:STRAIN:INVERT ON
> :CHANNEL1:STRAIN:INVERT 1
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:LSCale?
Function Queries all settings related to linear scaling when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRAIN:LSCale?
Example :CHANNEL1:STRAIN:LSCALE:
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:LSCale:AVALue
Function Sets or queries linear scaling coefficient A when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRAIN:LSCale:AVALue {< NRf >}
<NRF> = 1 to 16
Example :CHANNEL1:STRAIN:LSCALE:AVALUE 10
> :CHANNEL1:STRAIN:LSCALE:
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:LSCale:BVALue
Function Sets or queries offset value B when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRAIN:LSCale:BVALue {<NRf>}
<NRF> = -9.9999E+30 to 9.9999E+30
Example :CHANNEL1:STRAIN:LSCALE:BVALUE 5
> :CHANNEL1:STRAIN:LSCALE:
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:LSCale:DISPLAYtype?
Function Queries all settings related to the display type when using linear scaling.
Syntax :CHANnel<x>:STRAIN:LSCale:DISPLAYtype?
Example :CHANNEL1:STRAIN:LSCALE:DISPLAYTYPE?
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRAIN:LSCale:DISPLAYtype:MODE
Function Sets or queries the display format for linear scaling.
Syntax :CHANnel<x>:STRAIN:LSCale:DISPLAYtype:MODE {EXPONENT|FLOATing}
Example :CHANNEL1:STRAIN:LSCALE:DISPLAYTYPE:MODE EXPONENT
> :CHANNEL1:STRAIN:LSCALE:DISPLAYTYPE:MODE EXPONENT
3.7 CHANnel Group

:CHANnel<x>:STRain:LScale:DISPlaytype:DECimalnum
Function Sets or queries the decimal place when the display format for linear scaling is set to Floating.
Syntax :CHANnel<x>:STRain:LScale:DISPlaytype:D ECimalnum {<NRf>|AUTO}
:CHANnel<x>:STRain:LScale:DISPlaytype:D ECimalnum?
<x>=1 to 16
<NRf>=0 to 3
Example :CHANNEL1:STRAIN:LSCALE:DISPLAYTYPE:DECIMALNUM AUTO

:CHANnel<x>:STRain:LScale:GETMeasure
Function Measures the X values of P1 and P2 for linear scaling when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRain:LScale:GETMeasure {P1X|P2X|P1Y|P2Y}
<x>=1 to 16
Example :CHANNEL1:STRAIN:LSCALE:GETMeasure P1X
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRain:LScale:MODE
Function Sets or queries the linear scaling method when a strain module is installed in the specified channel (slot). (The method can only be set to SHUNT with a strain module with DSUB, Shunt-Cal.)
Syntax :CHANnel<x>:STRain:LScale:MODE {AXB|OFF|P12|SHUNT}
<x>=1 to 16
Example :CHANNEL1:STRAIN:LSCALE:MODE AXB
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRain:LScale:P1X
Function Sets or queries the X value of P1 for linear scaling when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRain:LScale:P1X {<NRf>}
<x>=1 to 16
For P1X, <NRf> = -9.9999E+30 to 9.9999E+30
For P2X, <NRf> = -9.9999E+25 to 9.9999E+25
Example :CHANNEL1:STRAIN:LSCALE:P1X 10
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRain:LScale:UNIT
Function Sets or queries the unit of measurement to attach to the result of linear scaling when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRain:LScale:UNIT {< String >}
<x>=1 to 16
<String>= Up to 4 characters
Example :CHANNEL1:STRAIN:LSCALE:UNIT "X"
Description An error occurs if a strain module is not installed.

:CHANnel<x>:STRain:RANGe
Function Sets or queries the measuring range when a strain module is installed in the specified channel (slot).
Syntax :CHANnel<x>:STRain:RANGe {<NRf>}
<x>=1 to 16
<NRf>= 0.25, 0.5, 1, 2.5, 5, 10 (mV/V)
500, 1000, 2000, 5000, 10000, 20000 (uSTR)
Example :CHANNEL1:STRAIN:RANGE 5000
Description An error occurs if a strain module is not installed.
3.7 CHANnel Group

:CHANnel<x>:STRain:UNIT
Function  
Sets or queries the unit of measurement when a strain module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:STRain:UNIT {MV|USTR}  
<x>=1 to 16

Example  
:CHANNEL1:STRAIN:UNIT USTR  
:CHANNEL1:STRAIN:UNIT?  
-> :CHANNEL1:STRAIN:UNIT USTR

Description  
An error occurs if a strain module is not installed.

:CHANnel<x>:TEMPerature?
Function  
Queries all settings when a temperature, high precision voltage module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature?  
<x>=1 to 16

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.

:CHANnel<x>:TEMPerature:BURNOut
Function  
Sets or queries whether or not burnout is detected when a temperature, high precision voltage isolation module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature:BURNOut  
{<Boolean>}  
:CHANnel<x>:TEMPerature:BURNOut?  
<x>=1 to 16

Example  
:CHANNEL1:TEMPERATURE:BURNOUT ON  
:CHANNEL1:TEMPERATURE:BURNOUT?  
-> :CHANNEL1:TEMPERATURE:BURNOUT 1

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.

:CHANnel<x>:TEMPerature:BWIDTH
Function  
Sets or queries the bandwidth limit when a temperature, high precision voltage module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature:BWIDTH {FULL|< Frequency >}  
:CHANnel<x>:TEMPerature:BWIDTH?  
<x>=1 to 16

Example  
:CHANNEL1:TEMPERATURE:BWIDTH 2.0HZ  
:CHANNEL1:TEMPERATURE:BWIDTH?  
-> :CHANNEL1:TEMPERATURE:BWIDTH 2.0E+00

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.

:CHANnel<x>:TEMPerature:COUPling
Function  
Sets or queries input coupling when a temperature, high precision voltage module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature:COUPling {TC|DC|GND}  
:CHANnel<x>:TEMPerature:COUPling?  
<x>=1 to 16

Example  
:CHANNEL1:TEMPERATURE:COUPLING DC  
:CHANNEL1:TEMPERATURE:COUPLING?  
-> :CHANNEL1:TEMPERATURE:COUPLING DC

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.

:CHANnel<x>:TEMPerature:RJC
Function  
Sets or queries the RJC when a temperature, high precision voltage module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature:RJC {<Boolean>}  
:CHANnel<x>:TEMPerature:RJC?  
<x>=1 to 16

Example  
:CHANNEL1:TEMPERATURE:RJC ON  
:CHANNEL1:TEMPERATURE:RJC?  
-> :CHANNEL1:TEMPERATURE:RJC 1

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.

:CHANnel<x>:TEMPerature:TYPE
Function  
Sets or queries the thermocouple type when a temperature, high precision voltage module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature:TYPE {K|E|J|T|L|U|S|B|Au7fe}  
:CHANnel<x>:TEMPerature:TYPE?  
<x>=1 to 16

Example  
:CHANNEL1:TEMPERATURE:TYPE K  
:CHANNEL1:TEMPERATURE:TYPE?  
-> :CHANNEL1:TEMPERATURE:TYPE K

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.

:CHANnel<x>:TEMPerature:UNIT
Function  
Sets or queries the unit of measurement values when a temperature, high precision voltage module is installed in the specified channel (slot).

Syntax  
:CHANnel<x>:TEMPerature:UNIT {C|K}  
:CHANnel<x>:TEMPerature:UNIT?  
<x>=1 to 16

Example  
:CHANNEL1:TEMPERATURE:UNIT C  
:CHANNEL1:TEMPERATURE:UNIT?  
-> :CHANNEL1:TEMPERATURE:UNIT? C

Description  
An error occurs if a temperature, high precision voltage, universal module is not installed.
3.7 CHANnel Group

:CHANnel<x>:UNIT?
Function Queries the unit added to the channel.
Syntax :CHANnel<x>:UNIT?
Example (When a voltage module is installed in channel 7 (slot 3) and the coupling is DC)
:CHANNEL7:UNIT? -> :CHANNEL7:UNIT "V"

:CHANnel<x>:VOLTage?
Function Queries all settings when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>:VOLTage?
Example (When a voltage module is installed in channel 7 (slot 3) and the coupling is DC)
:CHANNEL7:VOLTage? -> :CHANNEL7:VOLTage

:CHANnel<x>:VOLTage:BWIDTH
Function Sets or queries the bandwidth limit when a temperature, high precision voltage isolation module is installed in the specified channel (slot).
Syntax :CHANnel<x>:VOLTage:BWIDTH {FULL|<Frequency>}
Example :CHANNEL1:VOLTAGE:BWIDTH FULL

:CHANnel<x>:VOLTage:COUPling
Function Sets or queries input coupling when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>:VOLTage:COUPling {AC|DC|GND|ACRMS|DCRMS|TC|ACCL}
Example :CHANNEL1:VOLTAGE:COUPLING DC

:CHANnel<x>:VOLTage:LSCALE?
Function Queries all settings related to linear scaling when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>:VOLTage:LSCALE?
Example :CHANNEL1:VOLTAGE:LSCALE

:CHANnel<x>:VOLTage:LSCALE:AVALue
Function Sets or queries scaling coefficient A of linear scaling when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>:VOLTage:LSCALE:AVALue {<NRf>}
Example :CHANNEL1:VOLTAGE:LSCALE:AVALUE 10

Description
An error occurs if a voltage module is not installed.
3.7 CHANnel Group

:CHANnel<x>[:VOLTage]:LSCale:BVALue
Function Sets or queries linear scaling offset value B when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>[:VOLTage]:LSCale:BVALue [<NRf>]
Example :CHANnel1[:VOLTage]:LSCale:BVALue 10
Description An error occurs if a voltage module is not installed.

:CHANnel<x>[:VOLTage]:LSCale:DISPlaytype?
Function Queries all settings related to the display type when using linear scaling.
Syntax :CHANnel<x>[:VOLTage]:LSCale:DISPlaytype?
Example :CHANnel1[:VOLTage]:LSCale:DISPlaytype?

:CHANnel<x>[:VOLTage]:LSCale:DISPlaytype:MODE
Function Sets or queries the display format for linear scaling.
Syntax :CHANnel<x>[:VOLTage]:LSCale:DISPlaytype:MODE {EXPonent|FLOATing}
Example :CHANnel1[:VOLTage]:LSCale:DISPlaytype:MODE EXPONENT

:CHANnel<x>[:VOLTage]:LSCale:DISPlaytype:DECimalnum
Function Sets or queries the decimal place when the display format for linear scaling is set to Floating.
Syntax :CHANnel<x>[:VOLTage]:LSCale:DISPlaytype:DECimalnum [<NRf>]
Example :CHANnel1[:VOLTage]:LSCale:DISPlaytype:DECimalnum AUTO

:CHANnel<x>[:VOLTage]:LSCale:DISPlaytype:SUBunit
Function Sets or queries the sub unit when the display format for linear scaling is set to Floating.
Syntax :CHANnel<x>[:VOLTage]:LSCale:DISPlaytype:SUBunit {AUTO|NONE|PICO|NANO|MICro|KILO|MEGA|GIGA|TERA}
Example :CHANnel1[:VOLTage]:LSCale:DISPlaytype:SUBunit AUTO

:CHANnel<x>[:VOLTage]:LSCale:GETMeasure
Function Measures the X values of P1 and P2 for linear scaling when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>[:VOLTage]:LSCale:GETMeasure {P1X|P2X}
Example :CHANnel1[:VOLTage]:LSCale:GETMeasure P1X

:CHANnel<x>[:VOLTage]:LSCale:MODE
Function Sets or queries linear scaling when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>[:VOLTage]:LSCale:MODE {AXB|OFF|P12}
Example :CHANnel1[:VOLTage]:LSCale:MODE AXB

:CHANnel<x>[:VOLTage]:LSCale:{P1X|P1Y|P2X|P2Y}
Function Sets or queries the X or Y value of P1 or P2 for linear scaling when a voltage module is installed in the specified channel (slot).
Syntax :CHANnel<x>[:VOLTage]:LSCale:{P1X|P1Y|P2X|P2Y} [<NRf>]
Example :CHANnel1[:VOLTage]:LSCale:P1X 10

Description An error occurs if a voltage module is not installed.
3.7 COMMunicate Group

:CHANnel<x>[:VOLTage]:LSCALE:UNIT

Function: Sets or queries the unit of measurement to attach to the result of linear scaling when a voltage module is installed in the specified channel (slot).

Syntax:
:CHANnel<x>[:VOLTage]:LSCALE:UNIT {<String>}
:CHANnel<x>[:VOLTage]:LSCALE:UNIT?

<x>=1 to 16

<String>= Up to 4 characters

Example:
:CHANNEL1:VOLTAGE:LSCALE:UNIT "RPM"
:CHANNEL1:VOLTAGE:LSCALE:UNIT?

-> :CHANNEL1:VOLTAGE:LSCALE:UNIT "RPM"

Description:
An error occurs if a voltage module is not installed.

:CHANnel<x>[:VOLTage]:VDIV

Function: Sets or queries the V/div value when a voltage module is installed in the specified channel (slot).

Syntax:
:CHANnel<x>[:VOLTage]:VDIV {<Voltage>|<Current>}
:CHANnel<x>[:VOLTage]:VDIV?

<x>=1 to 16

<Voltage>= Depends on module. See the figure below.

Example:
:CHANNEL1:VOLTAGE:VDIV 5V
:CHANNEL1:VOLTAGE:VDIV? ->
:CHANNEL1:VOLTAGE:VDIV 5.000E+00

Description:
An error occurs if a voltage module is not installed.

Voltage Module and VDIV Range (probe 1:1)

<table>
<thead>
<tr>
<th>Type</th>
<th>VDIV Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>M701250/701255/720250</td>
<td>5 mV to 20 V</td>
</tr>
<tr>
<td>M701251</td>
<td>1 mV to 20 V</td>
</tr>
<tr>
<td>M701260/701267/720268</td>
<td>20 mV to 200 V</td>
</tr>
<tr>
<td>M701261/701262</td>
<td>5 mV to 20 V</td>
</tr>
<tr>
<td>M701265</td>
<td>0.1 mV to 10 V</td>
</tr>
<tr>
<td>M720266</td>
<td>0.1 mV to 20 V</td>
</tr>
<tr>
<td>M701275</td>
<td>5 mV to 10 V</td>
</tr>
<tr>
<td>M720210/720211</td>
<td>10 mV to 20 V</td>
</tr>
</tbody>
</table>
3.8 COMMunicate Group

The COMMunicate group deals with communications.

:COMMunicate?
Function
Queries all settings related to communications.
Syntax
:COMMunicate?

:COMMunicate:HEADER
Function
Sets or queries whether or not to add a header to responses to queries.
Syntax
:COMMunicate:HEADER {<Boolean>}
:COMMunicate:HEADER?
Example
:COMMUNICATE:HEADER ON
:COMMUNICATE:HEADER? -> :COMMUNICATE:HEADER 1

:COMMunicate:LOCKout
Function
Sets or clears local lockout.
Syntax
:COMMunicate:LOCKout {<Boolean>}
:COMMunicate:LOCKout?
Example
:COMMUNICATE:LOCKOUT ON
:COMMUNICATE:LOCKOUT? -> :COMMUNICATE:LOCKOUT 1

:COMMunicate:OPSE
Function
Sets or queries the overlap command that is used by the *OPC, *OPC? and *WAI commands.
Syntax
:COMMunicate:OPSE {<Register>}
:COMMunicate:OPSE?
Example
:COMMUNICATE:OPSE 65535
:COMMUNICATE:OPSE? -> :COMMUNICATE:OPSE 584
Description
Bits fixed to 0 are not set to 1.

:COMMunicate:OPSR?
Function
Queries the value of the operation pending status register.
Syntax
:COMMunicate:OPSR?
Example
:COMMUNICATE:OPSR? -> 0
Description
For details on the operation pending status register, see the figure for the :COMMunicate:WAIT? command.

:COMMunicate:OVERlap
Function
Sets or queries the commands to operate as overlap commands.
Syntax
:COMMunicate:OVERlap {<Register>}
:COMMunicate:OVERlap?
Example
:COMMUNICATE:OVERLAP 65535
:COMMUNICATE:OVERLAP? ->
:COMMUNICATE:OVERLAP 584
Description
Bits fixed to 0 are not set to 1.

:COMMunicate:REMOTE
Function
Sets remote or local. ON is remote mode
Syntax
:COMMunicate:REMOTE {<Boolean>}
:COMMunicate:REMOTE?
Example
:COMMUNICATE:REMOTE ON
:COMMUNICATE:REMOTE? ->
:COMMUNICATE:REMOTE 1

:COMMunicate:STATUS?
Function
Queries the line-specific status.
Syntax
:COMMunicate:STATUS?
Example
:COMMUNICATE:STATUS? ->
:COMMUNICATE:STATUS 0
Description
For the SL1000, 0 is always returned.

:COMMunicate:VERBOSE
Function
Sets or queries whether to return the response to a query using full spelling or abbreviations.
Syntax
:COMMunicate:VERBOSE {<Boolean>}
:COMMunicate:VERBOSE?
Example
:COMMUNICATE:VERBOSE ON
:COMMUNICATE:VERBOSE? ->
:COMMUNICATE:VERBOSE 1

:COMMunicate:WAIT
Function
Waits for one of the specified extended events to occur.
Syntax
:COMMunicate:WAIT {<Boolean>}
<Register>=0 to 65535
Example
:COMMUNICATE:WAIT 65535
3.8 CONTrol Group

:COMMunicate:WAIT?

Function: Creates the response that is returned when the specified event occurs.

Syntax: :COMMunicate:WAIT? {<Register>}

<Register>=0 to 65535 (extended event register)

Example: :COMMUNICATE:WAIT? 65535 -> 1

Description: Operation pending status register/overlap enable register

<table>
<thead>
<tr>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ACS</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

When bit 6 (ACS) = 1: Access to the medium not complete.
3.9 CONTROL Group

The CONTROL group deals with operational control for the station.

:CONTROL?
Function Queries all settings related to the station.
Syntax :CONTROL?

:CONTROL:FREE?
Function Queries all settings related to free-run measurement.
Syntax :CONTROL:FREE?

:CONTROL:FREE:LATCH
Function Executes the latch for the measurement data access for free-run measurement. (Unit firmware holds 'Acq memory writing address' of measurement group 1 data.)
Syntax :CONTROL:FREE:LATCH
Example :CONTROL:FREE:LATCH

:CONTROL:FREE:LCOUNT?
Function Queries the sample counts, which are counted from the start of measurement, at the latch point for free-run measurement.
Syntax :CONTROL:FREE:LCOUNT?
Description The function returns the acquisition counts of measurement group 1 using a 64 bit integer value.

:CONTROL:FREE:LENGTH<x>?
Function Queries the measured efficient data counts at the latch point for free-run measurement.
Syntax :CONTROL:FREE:LENGTH<x>?
Description The function returns 'the possible maximum acquisition counts' when the acquisition data count exceeds the possible maximum count per a channel for free-run measurement.

:CONTROL:FREE:STIME?
Function Queries the start time of measurement for free-run measurement.
Syntax :CONTROL:FREE:STIME?
Description The function returns the measurement start time in order of year/month/day/hour/minute/second/microsecond.

:CONTROL:FREE:GDELAY<x>?
Function Queries the delay between the measurement start points of each measurement group.
Syntax :CONTROL:FREE:GDELAY<x>?
Description The function returns the delay in measurement groups 2-4 from measurement group 1, expressed as measurement points of measurement group 1.

:CONTROL:HDCapacity?
Function Queries the total capacity of internal HDD.
Syntax :CONTROL:HDCapacity?
Example :CONTROL:HDCAPACITY 39053983

:CONTROL:HDFree?
Function Queries the amount of free space in the internal HDD.
Syntax :CONTROL:HDFree?
Example :CONTROL:HDFREE? -> :CONTROL:HDFREE 39044384
Description The function returns the value in units of Kbytes.
## 3.10 DATa Group

The DATa group deals with waveform data (internal data).

### :DATa

**Function**
Queries all settings related to waveform data.

**Syntax**
:DATa?

### :DATa:BYTeorder

**Function**
Sets or queries the transmission order when using word format of two bytes or more.

**Syntax**
:DATa:BYTeorder {LSBFirst|MSBFirst}
:DATa:BYTeorder?

**Example**
:DATA:BYTEORDER LSBFIRST
:DATA:BYTEORDER? -> :DATA:BYTEORDER LSBFIRST

**Description**
This command is effective only for DATA group commands.

### :DATa:FRAW?

**Function**
Queries the specified waveform data during free-run measurement. The acquisition start point and the acquisition count are specified by the count of the measurement groups to which a specified trace belongs.

**Syntax**
:DATA:FRAW? {<NRf>,<NRf>,<NRf>}
First <NRf> = Specify trace (1 to 16)
Second <NRf> = Specify the acquisition start point specified by the count of the measurement groups to which a specified trace belongs.
The specified range is acquired in advance as the sampled counts of Measurement Group 1 with the ‘CONTrol:FREE:COUNT?’ command.
Specify the data using a 64 bit integer value.
Third <NRf> = Data points specified by the count of the measurement groups to which a specified trace belongs.

**Example**
When acquiring the 1000th data point from the beginning of measurement on CH1
:DATA:FRAW? 1, 1000, 1 -> #12(sequence of data byte)

**Description**
The function returns “#0” when parameter trouble (trace number, data points etc) and overrun occurs.
3.11 ETHernet Group

The ETHernet group deals with the network.

:ETHernet?
Function Queries all settings related to the network.
Syntax :ETHernet?

:ETHernet:TCP?
Function Queries all settings related to TCP.
Syntax :ETHernet:TCP?

:ETHernet:TCPip:DHCP
Function Sets or queries DHCP.
Syntax :ETHernet:TCPip:DHCP {<Boolean>}
:ETHernet:TCPip:DHCP? <NRf>=0 to 255
Example :ETHERNET:TCPIP:DHCP ON
:ETHERNET:TCPIP:DHCP? ->
:ETHERNET:TCPIP:DHCP 0

:ETHernet:TCPip:GATEway
Function Sets or queries default gateway.
Syntax :ETHernet:TCPip:GATEway {<NRf>,<NRf>,<NRf>,<NRf>}
:ETHernet:TCPip:GATEway? <NRf>=0 to 255
Example :ETHERNET:TCPIP:GATEWAY 192,168,0,1
:ETHERNET:TCPIP:GATEWAY? ->
:ETHERNET:TCPIP:GATEWAY 192,168,0,1
Description The function returns the default gateway acquired from the DHCP server when DHCP is ON.

:ETHernet:TCPip:IPADDRESS
Function Sets or queries the IP address.
Syntax :ETHernet:TCPip:IPADDRESS {<NRf>,<NRf>,<NRf>,<NRf>}
:ETHernet:TCPip:IPADDRESS? <NRf>=0 to 255
Example :ETHERNET:TCPIP:IPADDRESS 192,168,0,2
:ETHERNET:TCPIP:IPADDRESS? ->
:ETHERNET:TCPIP:IPADDRESS 192,168,0,2
Description The function returns the IP address acquired from the DHCP server when DHCP is ON.
### 3.12 FILE Group

The FILE group deals with the internal hard disk.

#### :FILE?
**Function** Queries all settings related to the internal hard disk.
**Syntax** :FILE?

#### :FILE:DELETE
**Function** Deletes files.
**Syntax** :FILE:DELETE [<Filename>]

**Example** :FILE:DELETE "CASE1.WDF"

**Description** The target media to be deleted is selected using :FILE[:DIRECTORY]:DRIVE. The target directory to be deleted is selected using :FILE[:DIRECTORY]:CDIRECTORY. Specify the file name with the extension.

#### :FILE[:DIRECTORY]?
**Function** Queries all settings related to the directory of the storage media.
**Syntax** :FILE[:DIRECTORY]?

#### :FILE[:DIRECTORY]::CDIRECTORY
**Function** Changes the current directory of the storage media.
**Syntax** :FILE[:DIRECTORY]:CDIRECTORY [<String>]

**Example** :FILE:DIRECTORY:CDIRECTORY "NO_1"

#### :FILE[:DIRECTORY]::DRIVE
**Function** Sets the storage media to be controlled.
**Syntax** :FILE[:DIRECTORY]:DRIVE (ATA, <NRf>, <NRf>)

- The first <NRf> = Drive number (1 to 9)
- The second <NRf> = Partition number (0 to 9)

**Example** :FILE:DIRECTORY:DRIVE ATA,1,0

#### :FILE[:DIRECTORY]::FREE?
**Function** Queries the free disk space (bytes) on the target media.
**Syntax** :FILE[:DIRECTORY]::FREE?

**Example** :FILE:DIRECTORY:FREE 39.981449E+09

#### :FILE[:DIRECTORY]::MDIRECTORY
**Function** Creates a new directory in the current directory.
**Syntax** :FILE[:DIRECTORY]:MDIRECTORY (String)

**Example** :FILE:DIRECTORY:MDIRECTORY "NO_1"

**Description** The string is specified with a relative path.

#### :FILE[:DIRECTORY]::PATH?
**Function** Queries the current directory.
**Syntax** :FILE[:DIRECTORY]:PATH?

**Example** :FILE:DIRECTORY:PATH "Path = ATA,1,0"

#### :FILE:LOAD:SETup:ABORt
**Function** Aborts loading of data.
**Syntax** :FILE:LOAD:SETup:ABORt

#### :FILE:LOAD:SETup[:EXECute]
**Function** Loads data. This is an overlap command.
**Syntax** :FILE:LOAD:SETup[:EXECute] [<Filename>]

**Example** :FILE:LOAD:SETUP:EXECUTE "CASE1"

**Description** Describe <Filename> with 'No extension'.

#### :FILE:SAVE?
**Function** Queries all settings related to the saving of files.
**Syntax** :FILE:SAVE?

#### :FILE:SAVE:ANAMing
**Function** Sets or queries the auto naming function for saved files.
**Syntax** :FILE:SAVE:ANAMing {DATE|NUMBering|OFF}

**Example** :FILE:SAVE:ANAMING NUMBERING

**Description** For DATE, returns the file name in the date and time format.
For NUMbering, returns the auto-numbered name.
For OFF, returns file name of 'FILE:SAVE:NAME'.

#### :FILE:SAVE:AREA:MODE
**Function** Sets or queries the save area function.
**Syntax** :FILE:SAVE:AREA:MODE {<Boolean>}

**Example** :FILE:SAVE:AREA:MODE

**Description** The setting value of the 'FILE:SAVE:AREA:COUNt' command becomes valid when MODE is ON.
3.12 FILE Group

:FILE:SAVE:AREA:COUNt
Function: Sets or queries the save area.
Syntax: :FILE:SAVE:AREA:COUNt {<NRf>,<NRf>}
Example: :FILE:SAVE:AREA:COUNt 0,10000

:FILE:SAVE:AREA:COUNt?
Function: Queries the save area.
Example: :FILE:SAVE:AREA:COUNt?

:FILE:SAVE:BINary?
Function: Queries all settings related to the saving of waveform data.
Syntax: :FILE:SAVE:BINary?
Example: :FILE:SAVE:BINary?

:FILE:SAVE:{BINary|MEASure|SETup}:ABORT
Function: Aborts the saving of data.
Syntax: :FILE:SAVE:{BINary|MEASure|SETup}:ABORT
Example: :FILE:SAVE:BINARY:ABORT

:FILE:SAVE:BINARY:ACOUNT
Function: Sets or queries the target acquisition count when saving files.
Syntax: :FILE:SAVE:BINARY:ACOUNT {<NRf>}
Example: :FILE:SAVE:BINARY:ACOUNT 5

:FILE:SAVE:BINARY:ACOUNT?
Function: Queries the target acquisition count when saving files.
Example: :FILE:SAVE:BINARY:ACOUNT?

:FILE:SAVE:BINARY:EXECUTE
Function: Executes the saving of data to a file. This is an overlap command.
Syntax: :FILE:SAVE:{BINary|MEASure|SETup}[:EXECute]
Example: :FILE:SAVE:BINARY:EXECUTE

:FILE:SAVE:BINARY:HISTORY
Function: Sets or queries the save target of the history memory of the data.
Syntax: :FILE:SAVE:BINARY:HISTORY {ONE,ALL}
Example: :FILE:SAVE:BINARY:HISTORY ALL

:FILE:SAVE:BINARY:HISTORY?
Function: Queries the save target of the history memory of the data.
Example: :FILE:SAVE:BINARY:HISTORY?

:FILE:SAVE:BINARY:TALL (Trace All)
Function: Sets or queries the selection method of the trace when saving files.
Syntax: :FILE:SAVE:BINARY:TALL {<Boolean>}
Example: :FILE:SAVE:BINARY:TALL ON

:FILE:SAVE:COMMENT
Function: Sets or queries the comment of data to be saved.
Syntax: :FILE:SAVE:COMMENT {<String>}
Example: :FILE:SAVE:COMMENT "comment"

:FILE:SAVE:NAME
Function: Sets or queries the name of the file to be saved.
Syntax: :FILE:SAVE:NAME {<Filename>}
Example: :FILE:SAVE:NAME "CASE1"

Description: '/", ", ?, *, :|", "", <, >" cannot be used in the text.

Description: "aux", "con", "prn", "nul", "clock", "com1 to com9", "lpt1 to lpt9" cannot be used as the names of files.
### 3.13 GONogo Group

The GONogo group deals with GO/NO-GO judgment.

#### :GONogo?

- **Function**: Queries all settings related to GO/NO-GO judgment.
- **Syntax**: :GONogo?

#### :GONogo:ACONdition

- **Function**: Sets or queries the GO/NO-GO judgment action condition.
- **Syntax**: :GONogo:ACONdition
  - `{ALWAYS|FAILURE|SUCCESS}`
- **Example**: :GONOGO:ACONDITION FAILURE

#### :GONogo:ACTion?

- **Function**: Queries all settings related to the action taken when the execution condition is met.
- **Syntax**: :GONogo:ACTion?

#### :GONogo:ACTion:BUZZer

- **Function**: Sets or queries whether or not a beep is sounded when the condition is met.
- **Syntax**: :GONogo:ACTion:BUZZer {<Boolean>}
- **Example**: :GONOGO:ACTION:BUZZER OFF

#### :GONogo:ACTion:SAVE[:MODE]

- **Function**: Sets or queries whether or not waveform data is saved to the storage media when the condition is met.
- **Syntax**: :GONogo:ACTion:SAVE[:MODE] {<Boolean>}
- **Example**: :GONOGO:ACTION:SAVE:MODE OFF

#### :GONogo:AREA

- **Function**: Sets or queries the waveform area that is judged.
- **Syntax**: :GONogo:AREA {CURSOR|FULL}
- **Example**: :GONOGO:AREA CURSOR

#### :GONogo:COUNt?

- **Function**: Queries the number of performed GO/NO-GO judgments.
- **Syntax**: :GONogo:COUNt?
- **Example**: :GONOGO:COUNt? -> :GONOGO:COUNt 0

#### :GONogo:RSTatus?

- **Function**: Queries the most recent GO/NO-GO judgment.
- **Syntax**: :GONogo:RSTatus?
- **Example**: :GONOGO:RSTATUS? -> :GONOGO:RSTATUS 0
- **Description**: The command returns 0 when the judgment is GO and returns 1 when the judgment is NO-GO.

#### :GONogo:LOGic

- **Function**: Sets or queries the GO/NO-GO logical condition.
- **Syntax**: :GONogo:LOGic {AND|OR}
- **Example**: :GONOGO:LOGIC AND
  - :GONOGO:LOGIC? -> GONOGO:LOGIC AND

#### :GONogo:MODE

- **Function**: Sets or queries the GO/NO-GO judgment mode.
- **Syntax**: :GONogo:MODE {OFF|PARAMeter}
- **Example**: :GONOGO:MODE PARAMETER

#### :GONogo:NGCount?

- **Function**: Queries the GO/NO-GO judgment NO-GO count.
- **Syntax**: :GONogo:NGCount?
- **Example**: :GONOGO:NGCOUNT? -> :GONOGO:NGCOUNT 10

#### :GONogo:PARameter?

- **Function**: Queries all settings related to parameter judgment.
- **Syntax**: :GONogo:PARameter?

#### :GONogo:PARameter:ITEM<x>?

- **Function**: Queries all settings related to waveform parameters of the parameter judgment.
- **Syntax**: :GONogo:PARameter:ITEM<x>?
- **Example**: :GONOGO:PARaMeter:ITEM1:CAUSE?

#### :GONogo:PARameter:ITEM<x>:CAUSE?

- **Function**: Queries whether or not the specified waveform parameter is the cause of a NO-GO judgment.
- **Syntax**: :GONogo:PARameter:ITEM<x>:CAUSE?
- **Example**: :GONOGO:PARaMeter:ITEMx:CAUSE?
  - :GONOGO:PARaMeter:ITEMx:CAUSE? -> :GONOGO:PARaMeter:ITEMx:CAUSE 0
- **Description**: When the waveform parameter is the cause of a NO-GO judgment, the command returns 1. Otherwise, the command returns 0.
### 3.13 GONogo Group

**:GONogo:PARameter:ITEM<x>:MODE**

**Function:** Sets or queries the judgment criteria of the specified waveform parameter of the parameter judgment.

**Syntax:**

:GONogo:PARameter:ITEM<x>:MODE {OFF|IN|OUT}

Example:

:GONOGO:PARAMETER:ITEM1:MODE IN

-> :GONOGO:PARAMETER:ITEM1:MODE IN

**:GONogo:PARameter:ITEM<x>:TRACe**

**Function:** Sets or queries the target waveform of the measurement of the specified waveform parameters of the parameter judgment.

**Syntax:**

:GONogo:PARameter:ITEM<x>:TRACe <NRf> <x>=1 to 16

Example:

:GONOGO:PARAMETER:ITEM1:TRACe 1

-> :GONOGO:PARAMETER:ITEM1:TRACe 1

**:GONogo:PARameter:ITEM<x>:TYPE**

**Function:** Queries the item and the upper and lower limits of the measurement of the specified waveform parameter of the parameter judgment.

**Syntax:**

:GONogo:PARameter:ITEM<x>:TYPE?

Example:

:GONOGO:PARAMETER:ITEM1:TYPE?

-> :GONOGO:PARAMETER:ITEM1:TYPE:

MAXIMUM 1.00000E+00,-1.00000E+00

**:GONogo:PARameter:ITEM<x>:TYPE:<Parameter>**

**Function:** Sets or queries the upper and lower limits of the judgment area for the specified judgment condition.

**Syntax:**

: GONogo:PARameter:ITEM<x>:TYPE:<Parameter>{<Voltage|DONTcare>,<Current|DONTcare>,<Time|DONTcare>,<Frequency|DONTcare>,<<NRf>|DONTcare>}{<AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth1|BWIDth2|DUTYcycle|FALL|FREQuency|HIGH|LOW|MAXimum|MIDDle|MINimum|NOVershoot|NWIDth1|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ} <x>=1 to 16

Example:

:GONOGO:PARAMETER:ITEM1:TYPE:MAXIMUM 1V,-1V

-> :GONOGO:PARAMETER:ITEM1:TYPE:

MAXIMUM 1.00000E+00,-1.00000E+00

**:GONogo:PARameter:ITEM<x>:VALUE?**

**Function:** Queries the automated measurement value of the specified GO/NO-GO judgment parameter.

**Syntax:**

:GONogo:PARameter:ITEM<x>:VALUE?

Example:

:GONOGO:PARAMETER:ITEM1:VALUE?

-> :GONOGO:PARAMETER:ITEM1:

VALUE 500.00000E-03

**:GONogo:PARameter:ITEM<x>:PARam?**

**Function:** Queries the measurement item of the specified waveform parameter.

**Syntax:**

:GONogo:PARameter:ITEM<x>:PARam?

Example:

:GONOGO:PARAMETER:ITEM1:PARam?

-> :GONOGO:PARAMETER:ITEM1:

PARAM MAXIMUM

**:GONogo:PARameter:ITEM<x>:PARAM**

**Function:** Queries the measurement item of the specified waveform parameter.

**Syntax:**

:GONogo:PARameter:ITEM<x>:PARam?

Example:

:GONOGO:PARAMETER:ITEM1:PARam?

-> :GONOGO:PARAMETER:ITEM1:

PARAM MAXIMUM
3.14 HISTory Group

The HISTory group deals with the history memory.

```
:HISTory?
Function Queries all settings related to the history memory function.
Syntax :
Example :
```

```
:HISTory:CLEar
Function Clears all history memory data (all data in memory).
Syntax :
Example :
```

```
:HISTory:REcord? MINimum
Function Queries the minimum record number.
Syntax :
Example :
```
```
Description The function returns the oldest record number currently in the history.
The value is fixed to 0 during measurement.
```

```
:HISTory:DATE?
Function Queries the trigger date of the target record number.
Syntax :
Example :
```
```
Description The function returns"--------" when a record number that is smaller than minimum is specified.
```

```
:HISTory:TIME?
Function Queries the trigger time of the target record number.
Syntax :
Example :
```
```
Description The function returns"--------" when a record number that is smaller than minimum is specified.
```
3.15 INITialize Group

The INITialize group deals with the initialization of settings.

:INITialize:EXECute
Function: Initializes settings.
Syntax: :INITialize:EXECute
Example: :INITIALIZE:EXECUTE
Description: The contents not initialized by this command are as follows.
   - ETHenet group settings.
     Station name is set using SYSTem:STATion:NAME
     Group name is set using SYSTem:STATion:GNAME
   - Executing this command will clear the history memory.
   - Attention: Do not send other commands during the initialization. The processing status can be inferred from Bit 9 of the status register.

:INITialize:UNDO
Function: Undoes the initialization of settings.
Syntax: :INITialize:UNDO
Example: :INITIALIZE:UNDO
Description: The cleared history data will not be recovered.
   - Attention: Do not send other commands during the initialization. The processing status can be inferred from Bit 9 of the status register.
### 3.16 MEASure Group

The MEASure group deals with the automated measurement of waveform parameters.

**:MEASure?**
- **Function**: Queries all settings related to the automated measurement of waveform parameters.
- **Syntax**: :MEASure?

**:MEASure:CHANnel<x>?**
- **Function**: Queries the On/Off state of all of the waveform parameters of the specified channel.
- **Syntax**: :MEASure:CHANnel<x>?
  - <x>=1 to 16

**:MEASure:AREA**
- **Function**: Sets or queries the automatically measured waveform area for the waveform parameters.
- **Syntax**: :MEASure:AREA {CURSor|FULL}
- **Example**: :MEASURE:AREA CURSOR
- **Description**: When FULL is specified, the entire history waveform memory is subject to computation. When CURSor is specified, only the range set with MEASURE:CRANge is subject to computation.

**:MEASure:MODE**
- **Function**: Sets or queries the auto measurement mode of the waveform parameter.
- **Syntax**: :MEASure:MODE {OFF|ON}
- **Example**: :MEASURE:MODE ON

**:MEASure:CRANge (Count Range)**
- **Function**: Sets or queries the waveform parameter measurement range.
- **Syntax**: :MEASure:CRANge {<NRf>,<NRf>}
  - <NRf>=0 to 134217728
- **Example**: :MEASURE:CRANGE 2500,7500

**:MEASure:CHANnel<x>:DPRoximal?**
- **Function**: Queries all settings related to distal, mesial, and proximal.
- **Syntax**: :MEASure:CHANnel<x>:DPRoximal?
  - <x>=1 to 16

**:MEASure:CHANnel<x>:DPRoximal:MODE**
- **Function**: Sets or queries the distal, mesial, and proximal point mode setting.
- **Syntax**: :MEASure:CHANnel<x>:DPRoximal:MODE {PERCent|UNIT}
- **Example**: :MEASURE:CHANNEL1:DPROXIMAL:MODE PERCENT

**:MEASure:CHANnel<x>:DPRoximal:PERCent**
- **Function**: Sets or queries the distal, mesial, and proximal points as percentages.
- **Syntax**: :MEASure:CHANnel<x>:DPRoximal:PERCent {<NRf>,<NRf>,<NRf>}
  - <NRf>=0 to 100 (% in 0.1% steps)
- **Example**: :MEASURE:CHANNEL1:DPROXIMAL:PERCENT 40.0,60.0,80.0

**:MEASure:CHANnel<x>:METHod**
- **Function**: Sets or queries the high/low point setting method.
- **Syntax**: :MEASure:CHANnel<x>:METHod {AUTO|MAXMin}
  - <x>=1 to 16
- **Example**: :MEASURE:CHANNEL1:METHod AUTO
3.16 MEASure Group

:MEASURE:CHANnel<x>:DPROximal:UNIT
Function  Sets or queries the distal, mesial, and proximal points.
Syntax   :MEASURE:CHANnel<x>:DPROximal:UNIT {<Voltage>,<Voltage>,<Voltage>|<Current>,<Current>,<Current>|<NRf>,<NRf>,<NRf>}
<Voltage>=1 to 16
The settable ranges of <Voltage>, <Current>, and <NRf> vary depending on the range and offset settings.
The values specify the proximal, mesial, and distal points in that order.
Example  :MEASURE:CHANNEL1:DPROXIMAL:UNIT
-2V,0V,2V
:MEASURE:CHANnel<x>:DPROximal:UNIT?
-> :MEASURE:CHANnel1:DPROXIMAL:UNIT
-2.000E+00,0.0E+00,2.000E+00

:MEASURE:CHANnel<x>:<Parameter>?
Function  Queries all settings related to the specified waveform parameter (measurement item).
Syntax   :MEASURE:CHANnel<x>:< parameter >?
<x>=1 to 16
< Parameter >={AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth1|BWIDth2|DUTYcycle|FALL|FREQuency|HIGH|LOW|MAXimum|MIDDle|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ}
Example  :MEASURE:CHANNEL1:AMPLITUDE?
-> :MEASURE:CHANNEL1:AMPLITUDE:STATE 0

:MEASURE:CHANnel<x>:<Parameter>:STATE
Function  Sets or queries the on/off state of the measurement of the specified waveform parameter.
Syntax   :MEASURE:CHANnel<x>:< Parameter >:STATE {<Boolean>}
<Parameter >={AMPLitude|AVERage|AVGFreq|AVGPeriod|BWIDth1|BWIDth2|DUTYcycle|FALL|FREQuency|HIGH|LOW|MAXimum|MIDDle|MINimum|NOVershoot|NWIDth|PERiod|PNUMber|POVershoot|PTOPeak|PWIDth|RISE|RMS|SDEViation|TY1Integ|TY2Integ}
Example  :MEASURE:CHANNEL1:AMPLITUDE:STATE ON
-> :MEASURE:CHANNEL1:AMPLITUDE:STATE 1

:MEASURE:FILE?
Function  Queries all settings related to the file format output data of automatic measurement results.
Syntax   :MEASURE:FILE?

:MEASURE:FILE:TINFomation (Time Information)
Function  Sets or queries the ON/OFF state of the addition of the trigger time information in the file format output data of automated measurement results.
Syntax   :MEASURE:FILE:TINFomation {<Boolean>}
Example  :MEASURE:FILE:TINFOMATION ON
-> :MEASURE:FILE:TINFOMATION 1

Description The reply is the same as the reply of MEASURE:CHANnel<x>:<Parameter>:STATE?

Example  :MEASURE:CHANNel1:AMPLITUDE:STATE 0
-> :MEASURE:CHANNel1:AMPLITUDE:STATE 0

Example  :MEASURE:CHANNel1:AMPLITUDE:STATE ON
-> :MEASURE:CHANNel1:AMPLITUDE:STATE 1

Description NAN (not a number) is returned if measurement is not possible. Measurement can be impossible when the automatic measurement mode is not ON, or if the specified range of automatic measurement is not calculated as being 10M or more. <NRf> can be omitted. If omitted, the latest history parameter is queried. When <NRf> is used, queries from the latest waveform backward to the <NRf>'th waveform parameter value. If the specified history waveform does not exist, NAN is returned.
3.16 MEASure Group

:MEASURE:FILE:SEND?

Function: Executes the file format output of the automatic measurement results.

Syntax: :MEASURE:SEND?

Example: :MEASURE:SEND?

Description: Output the following CSV data in "block data format".

<table>
<thead>
<tr>
<th>Date</th>
<th>Time*</th>
<th>CH5 P-P</th>
<th>CH5 Min</th>
<th>CH6 P-P</th>
<th>CH6 Min</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008/04/01</td>
<td>10:00:00:000000</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
</tr>
<tr>
<td>2008/04/01</td>
<td>10:00:00:000000</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
</tr>
<tr>
<td>2008/04/01</td>
<td>10:00:00:000000</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
</tr>
<tr>
<td>2008/04/01</td>
<td>10:00:00:000000</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
</tr>
<tr>
<td>2008/04/01</td>
<td>10:00:00:000000</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
</tr>
<tr>
<td>2008/04/01</td>
<td>10:00:00:000000</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
<td>0.0033+00</td>
</tr>
</tbody>
</table>

* These two items will be appended when "with the trigger time information" is selected.
3.17 MONitor Group

The MONitor group deals with numeric monitor output.

**:MONitor:ASEND?**

**Function**: Outputs the numeric monitor data (ASCII format) of all channels.

**Syntax**: :MONitor:ASEND?

**Example**: :MONITOR:ASEND? -> Refer Format below

**Description**: Measured values of valid channels are output with each measured value delimited by a semicolon (0x3b). Valid channels means analog channels that are inserted. For channels whose display is OFF, the measured value is output as off. Outputs the data with 'label' and 'unit' or only the measured value depending on the VERBose setting.

**Format**
```
| Label 8 characters | Value 11 characters | Unit 4 characters |
```

**Attention**: The number of unit characters will exceed four if supplementary units exist.

**:MONitor:ASEND:CHANnel<x>??**

**Function**: Outputs the numeric monitor data (ASCII format) of the specified channel.

**Syntax**: :MONitor:ASEND:CHANnel<x>?<x>=1 to 16

**Example**: :MONITOR:ASEND:CHANNEL1?

```
-> "CH1 -1550.0mV"
```

**Description**: Outputs the data with 'label' and 'unit' or only the measured value depending on the VERBose setting. When display is OFF, the measured value is output as off.

**Attention**: The number of unit characters will exceed four if supplementary units exist.

**:MONitor:OFFSet:CHANnel<x>??**

**Function**: Queries the offset value used to convert the numeric monitor data of the specified channel into physical values.

**Syntax**: :MONitor:OFFSet:CHANnel<x>?<x>=1 to 16

**Example**: :MONITOR:OFFSET:CHANNEL1?

```
-> 0.0000000E+00
```

**Description**: The function returns the value including scaling data when using linear scaling.

**:MONitor:GAIN:CHANnel<x>??**

**Function**: Queries the gain value used to convert the numeric monitor data of the specified channel into physical values.

**Syntax**: :MONitor:GAIN:CHANnel<x>?<x>=1 to 16

**Example**: :MONITOR:GAIN:CHANNEL1?

```
-> 208.33333E-06
```

**Description**: The function returns the value including scaling data when using linear scaling.
3.17 MONitor Group

:MONitor:RANGE:CHANnel<x>? Function Queries the range value used to convert the numeric monitor data of the specified channel into physical values.
Syntax :MONitor:RANGE:CHANnel<x>?
<x>=1 to 16
Description The function returns the value including scaling data when using linear scaling.

:MONitor:SEND:{ALL|CHANnel<x>}? Function Outputs the numeric monitor data in binary format.
Syntax :MONitor:SEND:{ALL|CHANnel<x>}?
<x>=1 to 16
Example :MON:SEND:CHAN1? -> #?(? digit byte) (data byte sequence)
Description The number of the output byte of the specified channel follows the setting of MONitor:FORMat:CHANnel<x>. For "ALL", the data displays in ascending order. Channels whose displays are "OFF" will not be output.

:MONitor:VERBose Function Sets or queries whether or not to add 'label' and 'unit' to the response format of MONitor:ASENd?.
Syntax :MONitor:VERBose {<Boolean>}
Example :MONITOR:VERBOSE ON
:MONITOR:VERBOSE? -> :MONITOR:VERBOSE 1

:MONitor:LATCH:ASENd? Function Outputs the numeric monitor data (ASCII format) of all channels at the latch.
Syntax :MONitor:LATCH:ASENd?
Example :MONITOR:LATCH:ASEND?
Description The function returns "0" for the measurement value if the latch is not executed.

:MONitor:LATCH:ASENd:CHANnel<x>? Function Outputs the numeric monitor data (ASCII format) of the specified channels at the latch.
Syntax :MONitor:LATCH:ASENd:CHANnel<x>?
<x>=1 to 16
Example :MONITOR:LATCH:ASEND:CHANNEL1?
"CH1 -2396.7mV"
Description The function returns "0" for the measurement value if the latch is not executed.

:MONitor:LATCH:EXECute Function Latches the monitor data and the alarm data.
Syntax :MONitor:LATCH:EXECute
Example :MONITOR:LATCH:EXEC
Description Execute this command before using latch series output commands.

:MONitor:LATCH:SEND:{ALL|CHANnel<x>}? Function Outputs the numeric monitor data at the latch.
Syntax :MONitor:LATCH:SEND:{ALL|CHANnel<x>}?
<x>=1 to 16
Example :MONITOR:LATCH:SEND:CHANNEL1? -> #?(? digit byte) (data byte sequence)

:MONitor:LATCH:ALARm:{ALL|CHANnel<x>}? Function Outputs the channel alarm data at the latch.
Syntax :MONitor:LATCH:ALARm:{ALL|CHANnel<x>}?
<x>=1 to 16
Example :MONITOR:LATCH:ALARM:CHANNEL1? -> #?(? digit byte) (data byte sequence)
Description When "ALL" is specified, the response value returned is the value assigned by bit in channel number order from the leading bit. Transmissions are made in units of bytes. For example, the function returns one byte when the number of mounted channels is 1 to 8, and will return two bytes when the number of mounted channels is 9 to 16.
3.18 MRECord Group

The MRECord group deals with automatic data recording to the internal media.

**:MRECord?**
- **Function**: Queries all settings related to automatic data recording.
- **Syntax**: :MRECord?

**:MRECord:START**
- **Function**: Starts automatic data recording.
- **Syntax**: :MRECord:START
- **Example**: :MRECord:START
- **Description**: When this command is issued, measurement starts at the same time if the measurement is discontinued. Please set ":MRECord:DEST ination HDD" before starting recording.

**:MRECord:STOP**
- **Function**: Aborts automatic data recording.
- **Syntax**: :MRECord:STOP
- **Example**: :MRECord:STOP
- **Description**: Measurement does not stop even if this command is issued.

**:MRECord:DESTination**
- **Function**: Sets or queries the destination of the record of the automatic data recording to the internal media.
- **Syntax**: :MRECord:DESTination {HDD}
- **Example**: :MRECord:DESTINATION HDD
- **Description**: Specify the destination of the record of the automatic data recording using this command when executing automatic data recording.

**:MRECord:ECONdition (End CONdition)**
- **Function**: Sets or queries the end condition of automatic data recording.
- **Syntax**: :MRECord:ECONdition {CONTinue|TIME|ALARm|ETRise|ETFall|ACQStop|TUP}
- **Example**: :MRECord:ECONDITION ACQSTOP
- **Description**: Sets or queries the end condition of automatic data recording.

**:MRECord:ETIMe**
- **Function**: Sets or queries the start time if the start condition is the clock time.
- **Syntax**: :MRECord:ETIMe {<NRf>,<NRf>,<NRf>,<NRf>,<NRf>,<NRf>}
- **Example**: :MRECord:ETIME 2008,7,9,19,00,00
- **Description**: Sets or queries the end time if the end condition is the clock time.

**:MRECord:STIMe**
- **Function**: Sets or queries the start time if the start condition is the clock time.
- **Syntax**: :MRECord:STIMe {<NRf>,<NRf>,<NRf>,<NRf>,<NRf>,<NRf>}
- **Example**: :MRECord:STIME 2008,7,9,18,30,00

**:MRECord:SCONdition (Start CONdition)**
- **Function**: Sets or queries the start condition of automatic data recording.
- **Syntax**: :MRECord:SCONdition {TIME|ALARm|ETRise|ETFall|ACQStart}
- **Example**: :MRECord:SCONDITION ACQSTART
- **Description**: Sets or queries the start condition of automatic data recording.
3.18 MRECORD Group

: MRECORD: RTIME
Function Sets or queries the recording time if the end condition
is the recording time.
Syntax :MRECORD:RTIME {<NRf>,<NRf>,<NRf>,
<NRf>,<NRf>},
1st <NRf> = day (0 to 30)
2nd <NRf> = hour (0 to 23)
3rd <NRf> = minute (0 to 59)
4th <NRf> = second (0 to 59)
5th <NRf> = millisecond (0 to 999)
Setting range: 0 days 0 hours 0 minutes 0 seconds
001 milliseconds to 30 days 0 hours 0 minutes 0
seconds 000 milliseconds
Example :MRECORD:RTIME 0,1,0,0,0
:MRECORD:RTIME? -> :MRECORD:
RTIME 0,1,0,0,0

: MRECORD: INTERVAL?
Function Queries all settings related to the recording interval.
Syntax :MRECORD:INTERVAL?

: MRECORD: INTERVAL: TIME
Function Sets or queries the recording interval if the recording
interval mode is the TIME.
Syntax :MRECORD:INTERVAL: TIME {<NRf>,<NRf>,<NRf>},
1st <NRf> = hour (0 to 23)
2nd <NRf> = minute (0 to 59)
3rd <NRf> = second (0 to 59)
Setting range: 0 hours 0 minutes 0 seconds to 24
hours 0 minutes 0 seconds
Example :MRECORD:INTERVAL:TIME 0,30,0
:MRECORD:INTERVAL:TIME? -> :MRECORD:
INTERVAL:TIME 0,30,0

: MRECORD: INTERVAL: MODE
Function Sets or queries the recording interval mode of
automatic data recording.
Syntax :MRECORD:INTERVAL: MODE {OFF|TIME}
: MRECORD:INTERVAL:MODE?
Example :MRECORD:INTERVAL:MODE TIME
:MRECORD:INTERVAL:MODE? -> :MRECORD:
INTERVAL:MODE TIME

: MRECORD: COUNT
Function Sets or queries the number of recordings of automatic
data recording.
Syntax :MRECORD:COUNT {<NRf>|INFINITY}
Example :MRECORD:COUNT 10

: MRECORD: AREA?
Function Queries all settings related to the method of the
recording on the disk.
Syntax :MRECORD:AREA?

: MRECORD: AREA: MODE
Function Sets or queries the method of recording (recording
area) on the disk.
Syntax :MRECORD:AREA:MODE {CYCLic|SEQUential}
Example :MRECORD:AREA:MODE CYCLic
:MRECORD:AREA:MODE? -> :MRECORD:
AREA:MODE CYCLic

: MRECORD: AREA: FNUMBER
Function Sets or queries the method of recording (circular file
number) on the disk.
Syntax :MRECORD:AREA:NUMBER {<NRf>}
Example :MRECORD:AREA:NUMBER 10
:MRECORD:AREA:NUMBER? -> :MRECORD:
AREA:NUMBER 10

: MRECORD: ECLock?
Function Queries all settings related to recording when using
the external sample clock.
Syntax :MRECORD:ECLock?

: MRECORD: ECLock: COUNT
Function Sets or queries the recording counts when using the
external sample clock.
Syntax :MRECORD:ECLock:COUNT {<NRf>|MAX}
Example :MRECORD:ECLock:COUNT 100000
:MRECORD:ECLock:COUNT? -> :MRECORD:
ECLock:COUNT 100000

: MRECORD: ECLock: INTERVAL
Function Sets or queries the recording interval counts when
using the external sample clock.
Syntax :MRECORD:ECLock:INTERVAL {<NRf>}
Example :MRECORD:ECLock:INTERVAL 150000
:MRECORD:ECLock:INTERVAL? -> :MRECORD:
ECLock:INTERVAL 150000
3.19 MTRigger Group

The MTRigger group deals with manual triggering.

:MTRigger
Function   Activates manual triggering.
Syntax     :MTRigger
Example    :MTRIGGER
3.20 SELFtest Group

The SELFtest group deals with the self-test.

:SELFtest:HDD:EXECute?
Function: Executes the self-test of the internal HDD and outputs the results.
Syntax: :SELFtest:HDD:EXECute?
Example: :SELFTEST:HDD:EXECUTE? ->
:SELFTEST:HDD:EXECUTE 0
Description: The function returns '0' when the self-test is terminated normally, '1' when terminated abnormally, and '2' when an HDD is not mounted.

:SELFtest:HDFormat
Function: Formats the internal HDD.
Syntax: :SELFtest:HDFormat
Example: :SELFTEST:HDFORMAT
3.21 SSTAtert Group

The SSTAtert group deals with the execution of single start. Sets the ACQ operation mode to trigger, the trigger mode to single, and starts waveform acquisition.

:SSTAtert
Function  Executes single start.
Syntax  :SSTAtert
Example  :SSTAtert

:SSTaert? {<NRf>}
Function  Executes single start and waits for the completion.
Syntax  :SSTaert? {<NRf>}
<NRf>=1 to 36000 (100 ms unit: wait time, START and wait)
0 (START only. No wait.)
-36000 to -1 (100 ms resolution: wait period, START and wait)
Example  :SSTAert? 100 -> :SSTAert 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.
3.22 START Group

The START group is used to start waveform acquisition

:START

Function: The START group is used to start waveform acquisition.
Syntax: :START
Example: :START
Description: Use 'STOP' to stop waveform acquisition.
3.23 STATUS Group

The STATUS group deals with the settings and the inquiries related to the status report.

:STATus?
Function Queries all settings related to the communication status function.
Syntax :STATus?

:STATus:CONDition?
Function Queries the contents of the condition register.
Syntax :STATus:CONDition?
Example :STATus:CONDition? -> 4101

:STATus:EESE
Function Sets or queries the contents of the extended event enable register.
Syntax :STATus:EESE <Register>
Example :STATus:EESE *B00000000

:STATus:EESR?
Function Queries the contents of the extended event register and clears the register.
Syntax :STATus:EESR?
Example :STATus:EESR? -> 0

:STATus:ERRor?
Function Queries the error code and message information.
Syntax :STATus:ERRor?
Example :STATus:ERRor? -> 109, "Missing parameter:CHAN1:VDIV"
Description When there is no error, 0 (No error) is returned. You can specify whether or not to add the message using the "STATus:QMEssage" command.

:STATus:FILTer<x>
Function Sets or queries the transition filter.
Syntax :STATus:FILTer<x> {RISE|FALL|BOTH|NEVer}
Example :STATus:FILTer2 RISE

:STATus:QENable
Function Sets or queries whether or not to store messages other than errors to the Error queue (ON/OFF).
Syntax :STATus:QENable {<Boolean>}
Example :STATus:QENable ON

:STATus:QMEssage
Function Sets or queries whether or not to attach message information to the response to the "STATus:ERRor?" query (ON/OFF).
Syntax :STATus:QMEssage {<Boolean>}
Example :STATus:QMEssage ON
3.24 STOP Group

The STOP group deals with waveform acquisition stop.

:STOP
Function: Stops waveform acquisition.
Syntax: :STOP
Example: :STOP
Description: Recording will stop during the recording when the waveform acquisition is stopped.
3.25 SYSTem Group

The SYSTem group deals with the system.

:SYSTem?
Function Queries all settings related to the system.
Syntax :SYSTem?

:SYSTem: CLOCk?
Function Queries all settings related to the date and time.
Syntax :SYSTem:CLOCk?

:SYSTem: CLOCk:DATE
Function Sets or queries the date.
Syntax :SYSTem:CLOCk:DATE {< String >}
< String >=YYYY/MM/DD
YYYY: 2000 to 2099, MM: 1 to 12, DD: 1 to 31
Example :SYSTem:CLOCk:DATE "2008/07/10"

:SYSTem: CLOCk:TIME
Function Sets or queries the time.
Syntax :SYSTem:CLOCk:TIME {<String>}
< String >=HH:MM:SS
HH: 0 to 23, MM: 0 to 59, SS: 0 to 59
Example :SYSTem:CLOCk:TIME "13:57:00"

:SYSTem: KEYLock
Function Sets or queries the ON/OFF state of the keylock of
the main unit.
Syntax :SYSTem:KEYLock {<Boolean>}
Example :SYSTem:KEYLock ON
:SYSTem:KEYLock? -> :SYSTem:KEYLock 1
Description START/STOP key is locked. The status of the keylock
on the main unit is saved even if turning the power to
the main unit OFF and ON.

:SYSTem: LCD?
Function Queries all settings related to the LCD.
Syntax :SYSTem:LCD?

:SYSTem: LCD: BRIGHTness
Function Sets or queries the brightness of the LCD.
Syntax :SYSTem:LCD:BRIGHTness {<NRf>}
<NRf>=1 to 9
Example :SYSTem:LCD:BRIGHTness 3

:SYSTem: LCD: CONTRast
Function Sets or queries the contrast of the LCD.
Syntax :SYSTem:LCD:CONTrast {<NRf>}
<NRf>=1 to 10
Example :SYSTem:LCD:CONTrast 4

:SYSTem: LCD: MODE
Function Sets or queries the ON/OFF state of the LCD
backlight.
Syntax :SYSTem:LCD:MODE {<Boolean>}
Example :SYSTem:LCD:MODE ON

:SYSTem: LCD: DTOut (Display Time Out)
Function Sets or queries the ON/OFF state of the function that
brings the contents of the LCD back to the specified
screen in 30 seconds.
Syntax :SYSTem:LCD:DTOut {<Boolean>}
Example :SYSTem:LCD:DTOut ON

:SYSTem: LCD: DMODe (Display MODe)
Function Sets or queries the contents of LCD.
Syntax :SYSTem:LCD:DMODe {MODule|ERRor|CPARam}
Example :SYSTem:LCD:DMODe MODULE
3.25 SYSTem Group

:SYSTem:RCMode (Remote Control Mode)
Function Sets or queries the measurement stop mode when the remote terminal is controlling START/STOP.
Syntax :SYSTem:RCMode {<Boolean>}
Example :SYSTem:RCMode ON
:SYSTem:RCMode? -> :SYSTem:RCMode 1
Description Set "ON" or "1" to stop measurement using Low->High. Set "OFF" or "0" to not stop measurement with the remote terminal. Note that the function that does not stop measurement with the remote terminal is only valid in Trigger mode. Initial value is "1".

:SYSTem:STATion?
Function Queries all settings related to the unit.
Syntax :SYSTem:STATion?

:SYSTem:STATion:GNUMber?
Function Queries the area group number of the unit.
Syntax :SYSTem:STATion:GNUMber?
Example :SYSTEM:STATION:GNUMBER? -> :SYSTEM:STATION:GNUMBER 0
Description The function returns the rotary switch 1 value (setting of the area group number).
Attention The function retains and returns the state when the unit is powered ON.

:SYSTem:STATion:MODULE<x>?
Function Returns the attributes of the specified slot.
Syntax :SYSTem:STATion:MODULE<x>?
Example :SYSTEM:STATION:MODULE1? -> :SYSTEM:STATION:MODULE1 "M720210, HS100M12,2,1,0x21"
Description The function returns the module's kind, abbreviation, number of channels, channel start number, and module FPGA version text. The function returns "NOMODULE, NOMODULE, 0,1,0x11" if no module is installed in the specified slot.

<table>
<thead>
<tr>
<th>String</th>
<th>Abbreviation</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMODULE</td>
<td>NOMODULE</td>
<td>No module</td>
</tr>
<tr>
<td>M701250</td>
<td>HS10M12</td>
<td>701250</td>
</tr>
<tr>
<td>M701251</td>
<td>HS1M16</td>
<td>701251</td>
</tr>
<tr>
<td>M701255</td>
<td>NONISO 10M12</td>
<td>701255</td>
</tr>
<tr>
<td>M701260</td>
<td>HV (withRMS)</td>
<td>701260/67*</td>
</tr>
<tr>
<td>M701261</td>
<td>UNIVERSAL</td>
<td>701261</td>
</tr>
<tr>
<td>M701262</td>
<td>UNIVERSAL(AAF)</td>
<td>701262</td>
</tr>
<tr>
<td>M701265</td>
<td>TEMP/HPV</td>
<td>701265</td>
</tr>
<tr>
<td>M701270</td>
<td>STRAIN NDIS</td>
<td>701270</td>
</tr>
<tr>
<td>M701271</td>
<td>STRAIN DSUB</td>
<td>701271</td>
</tr>
<tr>
<td>M701275</td>
<td>ACCL/VOLT</td>
<td>701275</td>
</tr>
<tr>
<td>M701280</td>
<td>FREQ</td>
<td>701280</td>
</tr>
<tr>
<td>M701281</td>
<td>FREQ</td>
<td>701281</td>
</tr>
<tr>
<td>M720210</td>
<td>HS100M12</td>
<td>720210</td>
</tr>
<tr>
<td>M720211</td>
<td>HS100M12</td>
<td>720211</td>
</tr>
<tr>
<td>M720250</td>
<td>HS10M12</td>
<td>720250</td>
</tr>
<tr>
<td>M720266</td>
<td>TEMP/HPV</td>
<td>720266</td>
</tr>
<tr>
<td>M720268</td>
<td>HV (withRMS, AAF)</td>
<td>720268</td>
</tr>
<tr>
<td>M720281</td>
<td>FREQ</td>
<td>720281</td>
</tr>
</tbody>
</table>

"M701260" is returned also for the 701267 module.

:SYSTem:STATion:NUMBer?
Function Queries the number of the unit.
Syntax :SYSTem:STATion:NUMBer?

:SYSTem:STATion:NAME
Function Sets or queries the name of the unit.
Syntax :SYSTem:STATion:NAME {<strings>}
Example :SYSTEM:STATION:NAME "SL1000_1"

:SYSTem:STATion:GNAME
Function Sets or queries the name of the group.
Syntax :SYSTem:STATion:GNAME {<strings>}
Example :SYSTEM:STATION:GNAME "Group1"
### 3.26 TIMebase Group

The TIMebase group deals with the time base.

**:TIMebase?**

**Function:** Queries all settings related to the time base.

**Syntax:** :TIMebase?

**:TIMebase:MODULE<x>:GROUP**

**Function:** Sets or queries the measurement group to which the specified module belongs.

**Syntax:**

:TIMebase:MODULE<x>:GROUP {<NRf>}

:TIMebase:MODULE<x>:GROUP?

<x>=1 to 8

<NRf>=1 to 4

**Example:**

:TIMEBASE:MODULE1:GROUP 1

:TIMEBASE:MODULE1:GROUP? -> :TIMEBASE:MODULE1:GROUP 1

**Description:** The function returns "0" if the queries references a nonexistent module. One or more modules should be set in the measurement group 1.

**:TIMebase:SOURce**

**Function:** Sets or queries the time base.

**Syntax:**

:TIMebase:SOURce {EXTERNAL|INTERNAL}

:TIMebase:SOURce?

**Example:**

:TIMEBASE:SOURce INTERNAL

:TIMEBASE:SOURce? -> :TIMEBASE:SOURce INTERNAL

**:TIMebase:SRATe**

**Function:** Sets or queries the sample rate of measurement group 1.

**Syntax:**

:TIMebase:SRATe {< Frequency >}

:TIMebase:SRATe?

<Frequency> = 5 Hz, 10 Hz, 20 Hz, 50 Hz, 100 Hz, 200 Hz, 500 Hz, 1 kHz, 2 kHz, 5 kHz, 10 kHz, 20 kHz, 50 kHz, 100 kHz, 200 kHz, 500 kHz, 1 MHz, 2 MHz, 5 MHz, 10 MHz, 20 MHz, 50 MHz, 100 MHz

**Example:**

:TIMEBASE:SRAte 100MHz

:TIMEBASE:SRAte? -> :TIMEBASE:SRAte 100.00000E+06

**Description:** The sample rate which is set for measurement groups 2 to 4 can not be set to a faster value than measurement group 1. When the sample rate of measurement group 1 is set to 5 in steps of 1,2,5, the sample rate immediately thereafter (of measurement groups 2 to 4) can not be set to 2 in steps of 1,2,5. For example, you can not set 2 MHz for the sample rate of measurement groups 2 to 4 when the sample rate of measurement group 1 is 5 MHz.

**:TIMebase:GROUP<x>:SRATe**

**Function:** Sets or queries the sample rate of measurement groups 2 to 4.

**Syntax:**

:TIMebase:GROUP<x>:SRATe {< Frequency >}

:TIMebase:GROUP<x>:SRATe?

<x>=2 to 4

<Frequency>= 5 Hz, 10 Hz, 20 Hz, 50 Hz, 100 Hz, 200 Hz, 500 Hz, 1 kHz, 2 kHz, 5 kHz, 10 kHz, 20 kHz, 50 kHz, 100 kHz, 200 kHz, 500 kHz, 1 MHz, 2 MHz, 5 MHz, 10 MHz, 20 MHz, 50 MHz, 100 MHz

**Example:**

:TIMEBASE:GROUP2:SRAte 100MHz


**Description:** The sample rate which is set for measurement groups 2 to 4 can not be set to a faster value than measurement group 1. When the sample rate of measurement group 1 is set to 5 in steps of 1,2,5, the sample rate immediately thereafter (of measurement groups 2 to 4) can not be set to 2 in steps of 1,2,5. For example, you can not set 2 MHz for the sample rate of measurement groups 2 to 4 when the sample rate of measurement group 1 is 5 MHz.
3.27 TRIGger Group

The TRIGger group deals with triggers.

### :TRIGger?

**Function** Queries all settings related to triggers.

**Syntax**

```
:TRIGger?
```

### :TRIGger:COMBination?

**Function** Queries all settings related to the combination trigger class.

**Syntax**

```
:TRIGger:COMBination?
```

### :TRIGger:COMBination:CHANnel<x>?

**Function** Queries the settings of the specified channel in the combination trigger class.

**Syntax**

```
:TRIGger:COMBination:CHANnel<x>?
```

#### Example

```
:TRIGGER:COMBINATION:CHANNEL1:HYSTERESIS1 HIGH
```

### :TRIGger:COMBination:CHANnel<x1>:LEVEL<x2>

**Function** Sets or queries the trigger level of the specified channel in the combination trigger class.

**Syntax**

```
:TRIGger:COMBination:CHANnel<x1>:LEVEL<x2> {< Voltage >|<NRf>|< Current >} <x1>=1 to 16 <x2>=1,2
```

#### Example

```
:TRIGGER:COMBINATION:CHANNEL1:LEVEL1 10
```

#### Description

The setting range is the range that can be measured by the current range setting. The setting range is not reflected in the linear scale information.

### :TRIGger:COMBination:CHANnel<x1>:TYPE

**Function** Sets or queries the trigger type of the specified channel in the combination trigger class.

**Syntax**

```
:TRIGger:COMBination:CHANnel<x1>:TYPE {OFF|RISE|FALL|HIGH|LOW|BISlope|WLIn|WLOut|WINIn|WINOut} <x1>=1 to 16
```

#### Example

```
:TRIGGER:COMBINATION:CHANNEL1:TYPE RISE
```

### :TRIGger:COMBination:EXTernal:TYPE

**Function** Sets or queries the external trigger type of the specified channel in the combination trigger class.

**Syntax**

```
:TRIGger:COMBination:EXTernal:TYPE {OFF|RISE|FALL} <x>=1 to 16
```

#### Example

```
:TRIGGER:COMBINATION:EXTERNAL:TYPE RISE
```
### :TRIGger:COMBination:MODE

**Function**
Sets or queries the combination mode of the combination trigger class.

**Syntax**
`:TRIGGER:COMBINATION:MODE {AND|OR}
`:TRIGGER:COMBINATION:MODE?

**Example**
``:TRIGGER:COMBINATION:MODE AND

### :TRIGger:DELay

**Function**
Sets or queries the delay (time from the trigger point to the trigger position).

**Syntax**
`:TRIGGER:DELAY (< Time >)
`:TRIGGER:DELAY?

**Resolution**
0 to 10 s

**Example**
`:TRIGGER:DELAY 0.1
`:TRIGGER:DELAY? -> :TRIGGER:DELAY 100.00000E-03

**Description**
Fixed to 0 when the time base clock is set to external clock.

### :TRIGger:HOLDoff?

**Function**
Queries all settings related to the hold off.

**Syntax**
`:TRIGGER:HOLDoff?

### :TRIGger:HOLDoff:TIME

**Function**
Sets or queries the hold off time.

**Syntax**
`:TRIGGER:HOLDoff:TIME (< Time >)
`:TRIGGER:HOLDoff:TIME?

**Example**
`:TRIGGER:HOLDoff:TIME 0.1

### :TRIGger:MMODe

**Function**
Sets or queries the waveform acquisition mode for trigger measurement.

**Syntax**
`:TRIGGER:MMODE {NORMal|NSINgle|SINGle}
`:TRIGGER:MMODE?

**Example**
`:TRIGGER:MMODE NSINGLE

### :TRIGger:POSition

**Function**
Sets or queries the trigger position.

**Syntax**
`:TRIGGER:POSITION (<NRf>)
`:TRIGGER:POSITION?

**Example**
`:TRIGGER:POSITION 50
`:TRIGGER:POSITION? -> :TRIGGER:POSITION 50.00000E+00

### :TRIGger[:SIMple]?

**Function**
Queries all settings related to the simple trigger.

**Syntax**
`:TRIGGER[:SIMple]?

### :TRIGger[:SIMple]:HYSTeresis

**Function**
Sets or queries the hysteresis of the simple trigger.

**Syntax**
`:TRIGGER[:SIMple]:HYSTERESIS {HIGH|LOW|MIDDLE}
`:TRIGGER[:SIMple]:HYSTERESIS?

**Example**
`:TRIGGER[:SIMple]:HYSTERESIS HIGH
`:TRIGGER[:SIMple]:HYSTERESIS? -> :TRIGGER[:SIMple]:HYSTERESIS HIGH

**Description**
The hysteresis cannot be set or queried when the trigger source is set to EXTERNAL, LINE, or TIME.

### :TRIGger[:SIMple]:LEVel

**Function**
Sets or queries the trigger level of the simple trigger of the specified channel.

**Syntax**
`:TRIGGER[:SIMple]:LEVEL (< Voltage >|<NRf>|< Current >)
`:TRIGGER[:SIMple]:LEVEL?

**Example**
`:TRIGGER[:SIMple]:LEVEL 1v
`:TRIGGER[:SIMple]:LEVEL? -> :TRIGGER[:SIMple]:LEVEL 1.000E+00

**Description**
The hysteresis cannot be set or queried when the trigger source is set to EXTERNAL, LINE, or TIME. The setting range is the range that can be measured by the current range setting. The setting range is not reflected in the linear scale information.

### :TRIGger[:SIMple]:SLOPe

**Function**
Sets or queries the simple trigger type of the channel specified.

**Syntax**
`:TRIGGER[:SIMple]:SLOPE {RISE|FALL|BISLope}
`:TRIGGER[:SIMple]:SLOPE?

**Example**
`:TRIGGER[:SIMple]:SLOPE RISE
`:TRIGGER[:SIMple]:SLOPE? -> :TRIGGER[:SIMple]:SLOPE RISE

**Description**
The simple trigger type cannot set or queried when the trigger source is set to EXTERNAL, LINE, or TIME.
### 3.27 TRIGger Group

#### :TRIGger:SIMPle:SOURce

**Function**
Sets or queries the trigger source of the simple trigger.

**Syntax**
```
:TRIGger:SIMPle:SOURce {<NRf>|EXTernal|LINE|TIME}
```

**Example**
```
:TRIGGER:SIMPLE:SOURCE 1
:TRIGGER:SIMPLE:SOURCE? ->
:TRIGGER:SIMPLE:SOURCE 1
```

**Description**
An error occurs if a channel is specified for which no modules are installed.

#### :TRIGger:TIMer?

**Function**
Queries all settings related to the timer trigger.

**Syntax**
```
:TRIGger:TIMer?
```

#### :TRIGger:TIMer:DATE

**Function**
Sets or queries the date of the timer trigger.

**Syntax**
```
:TRIGger:TIMer:DATE {< String >}
```

**Example**
```
:TRIGGER:TIMER:DATE "2008/07/01"
:TRIGGER:TIMER:DATE? ->
:TRIGGER:TIMER:DATE "2008/07/01"
```

#### :TRIGger:TIMer:INTERval

**Function**
Sets or queries the trigger time interval of the timer trigger.

**Syntax**
```
:TRIGger:TIMer:INTERval {MIN1|MIN2|MIN3|MIN4|MIN5|MIN6|MIN7|MIN8|MIN9|MIN10|MIN15|MIN20|MIN25|MIN30|MIN40|MIN45|MIN50|HOUR1|HOUR2|HOUR3|HOUR4|HOUR5|HOUR6|HOUR7|HOUR8|HOUR9|HOUR10|HOUR11|HOUR12|HOUR18|HOUR24}
```

**Example**
```
:TRIGGER:TIMER:INTERVAL HOUR1
:TRIGGER:TIMER:INTERVAL? ->
:TRIGGER:TIMER:INTERVAL HOUR1
```

#### :TRIGger:TIMer:TIME

**Function**
Sets or queries the time of the timer trigger.

**Syntax**
```
:TRIGger:TIMer:TIME {< String >}
```

**Example**
```
:TRIGGER:TIMER:TIME "12:34:56"
:TRIGGER:TIMER:TIME? ->
:TRIGGER:TIMER:TIME "12:34:56"
```
### Trigger Level Range and Resolution

Excluding Frequency Module (M701280/701281/720281)

<table>
<thead>
<tr>
<th>Input</th>
<th>Set Range</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>± (V/div) × 10 V/div</td>
<td>1 / 100 V/div</td>
</tr>
<tr>
<td></td>
<td>Ex. 1 V/div</td>
<td>0.01 V/div</td>
</tr>
<tr>
<td></td>
<td>500 mV/div</td>
<td>0.005 V/div</td>
</tr>
<tr>
<td></td>
<td>200 mV/div</td>
<td>0.002 V/div</td>
</tr>
<tr>
<td>Temperature</td>
<td>Every type of measuring range</td>
<td>0.1 (C, K, F All)</td>
</tr>
<tr>
<td>Strain</td>
<td>± (Measuring range)</td>
<td>1 μSTR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.0005 m/V/V</td>
</tr>
<tr>
<td>Acceleration</td>
<td>×0.1</td>
<td>±100000 unit</td>
</tr>
<tr>
<td></td>
<td>×0.2</td>
<td>± 50000 unit</td>
</tr>
<tr>
<td></td>
<td>×0.5</td>
<td>± 20000 unit</td>
</tr>
<tr>
<td></td>
<td>×1</td>
<td>± 10000 unit</td>
</tr>
<tr>
<td></td>
<td>×2</td>
<td>± 5000 unit</td>
</tr>
<tr>
<td></td>
<td>×5</td>
<td>± 2000 unit</td>
</tr>
<tr>
<td></td>
<td>×10</td>
<td>± 1000 unit</td>
</tr>
<tr>
<td></td>
<td>×20</td>
<td>± 500 unit</td>
</tr>
<tr>
<td></td>
<td>×50</td>
<td>± 200 unit</td>
</tr>
<tr>
<td></td>
<td>×100</td>
<td>±100 unit</td>
</tr>
</tbody>
</table>

Frequency Module (M701280/701281/720281)

<table>
<thead>
<tr>
<th>Input</th>
<th>Set Range</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>± (V/div) × 10</td>
<td>0.0005 div or 0.001 div</td>
</tr>
<tr>
<td></td>
<td>min 0.00 1Hz</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ex 1 Hz/div</td>
<td>0.001 Hz</td>
</tr>
<tr>
<td></td>
<td>2 Hz/div</td>
<td>0.002 Hz</td>
</tr>
<tr>
<td></td>
<td>5 Hz/div</td>
<td>0.005 Hz</td>
</tr>
<tr>
<td></td>
<td>10 Hz/div</td>
<td>0.005 Hz</td>
</tr>
<tr>
<td>RPM</td>
<td>± (V/div) × 10</td>
<td>0.0005 div or 0.001 div</td>
</tr>
<tr>
<td></td>
<td>Ex 1000 rpm/div</td>
<td>0.5 rpm</td>
</tr>
<tr>
<td></td>
<td>2000 rpm/div</td>
<td>2 rpm</td>
</tr>
<tr>
<td></td>
<td>5000 rpm/div</td>
<td>5 rpm</td>
</tr>
<tr>
<td>RPS (701280)</td>
<td>± (V/div) × 10</td>
<td>0.0005 div or 0.001 div</td>
</tr>
<tr>
<td></td>
<td>Ex 1 rps/div</td>
<td>0.5 mrps</td>
</tr>
<tr>
<td></td>
<td>2 rps/div</td>
<td>0.002 rps</td>
</tr>
<tr>
<td></td>
<td>5 rps/div</td>
<td>0.005 rps</td>
</tr>
<tr>
<td>Period</td>
<td>± (V/div) × 10</td>
<td>0.0005 div or 0.001 div</td>
</tr>
<tr>
<td></td>
<td>Ex 1 msec/div</td>
<td>0.5 usec</td>
</tr>
<tr>
<td></td>
<td>2 msec/div</td>
<td>2 usec</td>
</tr>
<tr>
<td></td>
<td>5 msec/div</td>
<td>5 usec</td>
</tr>
<tr>
<td>Duty</td>
<td>± (V/div) × 10</td>
<td>0.001 div</td>
</tr>
<tr>
<td></td>
<td>1%/div</td>
<td>0.001%</td>
</tr>
<tr>
<td></td>
<td>2%/div</td>
<td>0.002%</td>
</tr>
<tr>
<td></td>
<td>5%/div</td>
<td>0.005%</td>
</tr>
<tr>
<td>Power Freq</td>
<td>± (V/div) × 10</td>
<td>2 Hz/div</td>
</tr>
<tr>
<td></td>
<td>Ex 1 Hz/div</td>
<td>0.002 Hz</td>
</tr>
<tr>
<td></td>
<td>0.5 Hz/div</td>
<td>0.001 Hz</td>
</tr>
<tr>
<td></td>
<td>0.2 Hz/div</td>
<td>0.001 Hz</td>
</tr>
<tr>
<td></td>
<td>0.1 Hz/div</td>
<td>0.001 Hz</td>
</tr>
<tr>
<td>Pulse Width</td>
<td>± (V/div) × 10</td>
<td>0.0005 div or 0.001 div</td>
</tr>
<tr>
<td></td>
<td>Ex 1 msec/div</td>
<td>0.5 usec</td>
</tr>
<tr>
<td></td>
<td>2 msec/div</td>
<td>2 usec</td>
</tr>
<tr>
<td></td>
<td>5 msec/div</td>
<td>5 usec</td>
</tr>
<tr>
<td>Integ</td>
<td>± (V/div) × 10</td>
<td>Float Setting</td>
</tr>
<tr>
<td>Velocity</td>
<td>± (V/div) × 10</td>
<td>Float Setting</td>
</tr>
</tbody>
</table>
The WAVeform group deals with acquired waveform data.

**:WAVeform?**
Function Queries all settings related to waveform data.
Syntax :WAVeform?

**:WAVeform:ACOunt?**
Function Queries the acquisition count.
Syntax :WAVeform:ACOunt?
Example :WAVEFORM:ACCOUNT? -> :WAVEFORM:ACCOUNT 10
Description The function returns the acquired acquisition count. The function returns "0" when data has not been acquired at least once.

**:WAVeform:BITS?**
Function Queries the bit length of the waveform data specified by ":WAVeform:TRACe".
Syntax :WAVeform:BITS?
Example :WAVEFORM:BITS? -> :WAVEFORM:BITS 16
Description The bit length which is output with this command is not the same as 'the valid bit length of A/D'. For the SL1000, all analog Ch will be '16'.

**:WAVeform:BYTeorder**
Function Sets or queries the byte order when using word format of two bytes or more.
Syntax :WAVeform:BYTeorder {LSBFirst|MSBFirst}
Example :WAVEFORM:BYTEORDER LSBFIRST
**:WAVeform:BYTeorder?**
Syntax :WAVeform:BYTeorder?
Description This command is valid for only the commands of the WAVeform group.

**:WAVeform:DIVision?**
Function Queries the Division value used when converting the waveform data specified by 'WAVEform:TRACe' to physical values.
Syntax :WAVeform:DIVision?
Description The data is output per the current 'WAVEform:FROMat' setting of the specified trace. The Division value varies according to the kind of input module as follows.

<table>
<thead>
<tr>
<th>Input</th>
<th>BYTE</th>
<th>WORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Temperature</td>
<td>25.6</td>
<td>0.1</td>
</tr>
<tr>
<td>Strain</td>
<td>187.5</td>
<td>48000</td>
</tr>
<tr>
<td>Acceleration</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Frequency</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>RPM</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Period</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Duty</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Power Freq</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Pulse Width</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Pulse Integ</td>
<td>93.75</td>
<td>24000</td>
</tr>
<tr>
<td>Velocity</td>
<td>93.75</td>
<td>24000</td>
</tr>
</tbody>
</table>

**:WAVeform:END**
Function Sets or queries the final data point of the waveform (main waveform) specified by 'WAVEform:TRACe'.
Syntax :WAVEform:END {<NRf>}
Example :WAVEFORM:END 100000
**:WAVEform:END?**
Syntax :WAVEform:END?
Example :WAVEFORM:END? -> :WAVEFORM:END 100000
Description The total number of data points can be queried using :WAVEform:CAPTure:LENGth?

**:WAVEform:FORMAT**
Function Sets or queries the format of the data to be transmitted.
Syntax :WAVEform:FORMAT {ASCii|BYTE|WORD}
Example :WAVEFORM:FORMAT WORD
**:WAVEform:FORMAT?**
Syntax :WAVEform:FORMAT?
Example :WAVEFORM:FORMAT? -> :WAVEFORM:FORMAT WORD
### 3.28 WAVeform Group

#### :WAVeform:GAIN?

**Function**
Queries the gain value used when converting the waveform data specified by :WAVeform:TRACe to physical values.

**Syntax**
`:WAVeform:GAIN?`

**Example**
`:WAVEFORM:GAIN?` → `:WAVEFORM:GAIN 2.08333333333333E-03`

**Description**
The function returns the value including the scaling data when using linear scale. The function returns the value including the scaling data when using linear scale.

#### :WAVeform:HMAX? (History MAX)

**Function**
Queries the maximum number of history that can be acquired by the currently specified unit.

**Syntax**
`:WAVeform:HMAX?`

**Example**
`:WAVEFORM:HMAX?` → `:WAVEFORM:HMAX 64`

#### :WAVeform:LENGTH?

**Function**
Queries all waveform data points (main side) specified by :WAVeform:TRACe.

**Syntax**
`:WAVeform:LENGTH?`

**Example**
`:WAVEFORM:LENGTH?` → `:WAVEFORM:LENGTH 500500`

**Description**
The function returns the actual number of acquired points when measurement stopped (the points actually acquired from the measurement group to which the specified trace belonged.)

#### :WAVeform:MODule?

**Function**
Queries the module corresponding to the waveform specified by :WAVeform:TRACe.

**Syntax**
`:WAVeform:MODule?`

**Example**
`:WAVEFORM:MODULE?` → `:WAVEFORM:MODULE M720210`

**Description**
The following values are returned from the module.

<table>
<thead>
<tr>
<th>String</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOMODULE</td>
<td>No Module</td>
</tr>
<tr>
<td>M701250</td>
<td>701250</td>
</tr>
<tr>
<td>M701251</td>
<td>701251</td>
</tr>
<tr>
<td>M701255</td>
<td>701255</td>
</tr>
<tr>
<td>M701260</td>
<td>701260/67*</td>
</tr>
<tr>
<td>M701261</td>
<td>701261</td>
</tr>
<tr>
<td>M701262</td>
<td>701262</td>
</tr>
<tr>
<td>M701265</td>
<td>701265</td>
</tr>
<tr>
<td>M701270</td>
<td>701270</td>
</tr>
<tr>
<td>M701271</td>
<td>701271</td>
</tr>
<tr>
<td>M701275</td>
<td>701275</td>
</tr>
<tr>
<td>M701280</td>
<td>701280</td>
</tr>
<tr>
<td>M701281</td>
<td>701281</td>
</tr>
<tr>
<td>M720210</td>
<td>720210</td>
</tr>
<tr>
<td>M720211</td>
<td>720211</td>
</tr>
<tr>
<td>M720250</td>
<td>720250</td>
</tr>
<tr>
<td>M720266</td>
<td>720266</td>
</tr>
<tr>
<td>M720268</td>
<td>720268</td>
</tr>
<tr>
<td>M720281</td>
<td>720281</td>
</tr>
</tbody>
</table>

“M701260” is returned also for the 701267 module.

#### :WAVeform:OFFSet?

**Function**
Queries the offset value used when converting the waveform data specified by :WAVeform:TRACe to physical values.

**Syntax**
`:WAVeform:OFFSet?`

**Example**
`:WAVEFORM:OFFSET?` → `:WAVEFORM:OFFSET 0.0000000E+00`

**Description**
The function returns the value including linear scaling data when using linear scale.

#### :WAVeform:RANGe?

**Function**
Queries the range value when converting the waveform data specified by :WAVeform:TRACe to physical values.

**Syntax**
`:WAVeform:RANGe?`

**Example**
`:WAVEFORM:RANGE?` → `:WAVEFORM:RANGE 5.0000000E+00`

**Description**
The function returns the value including linear scaling data when using linear scale.

#### :WAVeform:RECORD

**Function**
Sets or queries the target record number of the main waveform for the commands in the WAVeform group.

**Syntax**
`:WAVeform:RECORD {MINimum|<NRf>} [MINimum]`

**Example**
`:WAVEFORM:RECORD 0` → `:WAVEFORM:RECORD 0`

**Description**
Specifying ‘MINimum’ sets the record to the minimum record number (the record number of the oldest data). The parameter ‘0’ means the latest record. Please set a minus number to get a historical record.

#### :WAVeform:RECORD? MINimum

**Function**
Queries the minimum record number of the history (main waveform).

**Syntax**
`:WAVeform:RECORD? MINimum`

**Example**
`:WAVEFORM:RECORD? MINIMUM` → `:WAVEFORM:RECORD -9`

#### :WAVeform:SEND?

**Function**
Queries the waveform data (raw data) specified by :WAVeform:TRACe.

**Syntax**
`:WAVeform:SEND? {<NRf>} `<NRf>={0 to -4999}`

**Example**
`:WAVEFORM:SEND?` → `#?(number of bytes, ? digits)(series of data bytes)`

**Description**
<NRf> can be omitted. If <NRf> is included, waveform data is queried <NRf> times in order from the record number specified by :WAVeform:RECORD `<NRf>`.
3.29 WAVeform Group

**:WAVeform:SIGN?**

**Function**
Queries the existence of a sign when querying the waveform data specified by :WAVeform:TRACe using block data.

**Syntax**
**:WAVeform:SIGN?**

**Example**
**:WAVEFORM:SIGN? -> :WAVEFORM:SIGN 1

**Description**
For the SL1000, '1' is always returned.

**:WAVeform:SRATe?**

**Function**
Queries the sample rate of the record specified by :WAVeform:RECord.

**Syntax**
**:WAVeform:SRATe?**

**Example**
**:WAVEFORM:SRATE? -> :WAVEFORM:SRATE 5.0000000E+06

**Description**
The function returns the sample rate of the measurement group to which the trace specified by 'WAVeform:SRATe' belongs.

**:WAVeform:STARt**

**Function**
Sets or queries the first data point of the waveform (main waveform) specified by 'WAVEform:TRACe'.

**Syntax**
**:WAVeform:STARt {<NRf>}
**:WAVeform:STARt?**

**Example**
**:WAVEFORM:START 0
**:WAVEFORM:START? -> :WAVEFORM:START 0

**Description**
The total number of data points can be queried using :WAVeform:LENGth?

**:WAVeform:TRACe**

**Function**
Sets or queries the target waveform.

**Syntax**
**:WAVeform:TRACe {<NRf>}

**Example**
**:WAVEFORM:TRACE 1
**:WAVEFORM:TRACE? -> :WAVEFORM:TRACE 1

**Description**
An error occurs if a module is not installed in the channel.

**:WAVeform:TRIGger?**

**Function**
Queries the trigger position of the record specified by :WAVeform:RECord.

**Syntax**
**:WAVeform:TRIGger?**

**Example**
**:WAVEFORM:TRIGGER? -> :WAVEFORM:TRIGGER 250000

**Description**
Queries the number of points from the first point of the record to the trigger position.

**:WAVeform:TYPE?**

**Function**
Queries the acquisition mode of the waveform specified by :WAVeform:TRACe.

**Syntax**
**:WAVeform:TYPE?**

**Example**
**:WAVEFORM:TYPE? -> :WAVEFORM:TYPE NORMAL

**Description**
AVERage, ENvelope, BAverage, or NORMal is returned.

**:WAVeform:GDELay?**

**Function**
The delay of 'the measurement start point' or 'the trigger point' of the waveform specified by 'WAVeform:TRACe' is returned.

**Syntax**
**:WAVeform:GDELay?**

**Example**
**:WAVEFORM:GDELAY? -> :WAVEFORM:GDELAY 3,3

**Description**
The function converts and returns the delay of the measurement group to which the specified waveform data belongs relative to measurement group 1. The target record number references the setting value of 'WAVeform:RECord'.
# 3.29 Common Command Group

The commands in the common command group are defined in IEEE488.2-1987 and are independent of the instrument’s functions.

### *CAL?*
- **Function**: Performs calibration and queries the result.
- **Syntax**: `*CAL?`
- **Example**: `*CAL? -> 0`
- **Description**: If the calibration terminates normally, "0" is returned. If an error is detected, "1" is returned.

### *CLS*
- **Function**: Clears the standard event register, extended event register, and error queue.
- **Syntax**: `*CLS`
- **Example**: `
  *CLS`
- **Description**: If the *CLS command is located immediately after the program message terminator, the output queue is also cleared.

### *ESE*
- **Function**: Sets the standard event enable register or queries the current setting.
- **Syntax**: `*ESE {<NRf>}
  *ESE?
  <NRf>=0 to 255`
- **Example**: `
  *ESE 251
  *ESE? -> 251`
- **Description**:• Specify the value as a sum of decimal values of each bit.
  • For example, specifying "ESE 251" will cause the standard enable register to be set to "11110011." In this case, bit 2 of the standard event register is disabled which means that bit 5 (ESB) of the status byte register is not set to "1," even if a "query error" occurs.
  • The default value is "ESE 0" (all bits disabled). This result is not saved by the power-off.
  • A query using *ESE? will not clear the contents of the standard event enable register.

### *ESR?*
- **Function**: Queries the standard event register and clears the register.
- **Syntax**: `*ESR?`
- **Example**: `
  *ESR? 188`
- **Description**: A sum of decimal values of each bit is returned.
  You can check what type of events occurred when an SRQ is generated.
  For example, if a value of "32" is returned, this indicates that the standard event register is set to "00100000." In this case, you can see that the SRQ occurred due to a "command syntax error." A query using *ESR? will clear the contents of the standard event register.

### *IDN?*
- **Function**: Queries the instrument model.
- **Syntax**: `*IDN?`
- **Example**: `
  *IDN? -> YOKOGAWA, 720120, 0, F1.10`
- **Description**: The information is returned in the following form:
  `<Manufacturer>, <Model>, <Serial No.>, <Firmware version>.

### *OPC*
- **Function**: Sets a "1" to bit 0 (OPC bit) of the standard event register bit upon the completion of the specified overlap command.
- **Syntax**: `*OPC`
- **Example**: `
  *OPC`
- **Description**: The COMMunicate:OPSE command is used to specify the overlap command. If *OPC is not the last command of the message, the operation is not guaranteed.

### *OPC?*
- **Function**: The specified overlap command is completed, ASCII code '1' is returned.
- **Syntax**: `*OPC?`
- **Example**: `
  *OPC? -> 1`
- **Description**: The COMMunicate:OPSE command is used to specify the overlap command. If *OPC? is not the last command of the message, the operation is not guaranteed.

### *OPT?*
- **Function**: Queries the installed options.
- **Syntax**: `*OPT?`
- **Example**: `
  *OPT? -> 128MW, HD, PROBEPOWER`
- **Description**: The information of installed memory length:
  Total capacity of the waveform data memory '128MW'.
  installing the internal HD: 'HD'
  loading the probe power option: 'PROBEPOWER'
  loading the ether net option: 'ETHER'

### *RST*
- **Function**: Initializes the current settings.
- **Syntax**: `*RST`
- **Example**: `
  *RST`
- **Description**: Also clears the *OPC and *OPC? commands that were sent earlier.
### 3.29 Common Command Group

**`*SRE`**

**Function:**
Sets or queries the service request enable register value.

**Syntax:**
```
*SRE {<NRf>}
*SRE?
```

**Example:**
```
*SRE 239
*SRE? -> 175
```

**Description:**
- Specify the value as a sum of decimal values of each bit.
  - For example, specifying "*SRE 239" will cause the service request enable register to be set to "11101111." In this case, bit 4 of the service request enable register is disabled which means that bit 4 (MAV) of the status byte register is not set to "1," even if "the output queue is not empty."
  - Bit 6 (MSS) of the status byte register is the MSS bit itself, and therefore, is ignored.
  - The default value is "*SRE 0" (all bits disabled). This result is not saved by the power-off.
  - A query using *SRE? will not clear the contents of the service request enable register.

**`*STB?`**

**Function:**
Queries the status byte register.

**Syntax:**
```
*STB?
```

**Example:**
```
*STB? -> 4
```

**Description:**
- The sum of the bits is returned as a decimal value.
  - Since the register is read without executing serial polling, bit 6 is a MSS bit not RQS.
  - For example, if a value of "4" is returned, this indicates that the status byte register is set to "00000100." In this case, you can see that "the error queue is not empty" (an error occurred).
  - * A query using *STB? will not clear the contents of the status byte register.

**`*TST?`**

**Function:**
Performs a self-test and queries the result.

**Syntax:**
```
*TST?
```

**Example:**
```
*TST? -> 0
```

**Description:**
- The test is of each internal memory, and is executed in the order: system memory test, register test, and Acq memory test.
  - If the test passes, the command returns 0. If the test fails, the command returns 1.

**`*WAI`**

**Function:**
Holds the subsequent command until the completion of the specified overlap operation.

**Syntax:**
```
*WAI
```

**Example:**
```
*WAI
```

**Description:**
The COMMunicate:OPSE command is used to specify the overlap command.
4.1 Overview of the Status Report

Status Reports
The figure below shows the status report that is read by serial polling. This status report is an extended version of the status report defined in IEEE 488.2-1992.

![Status Report Diagram](image-url)
4.1 Overview of the Status Report

Overview of the Registers and Queues

<table>
<thead>
<tr>
<th>Name</th>
<th>Functions</th>
<th>Writing</th>
<th>Reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status byte</td>
<td></td>
<td></td>
<td>Serial polling (RQS)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>*STB? (MSS)</td>
</tr>
<tr>
<td>Service request enable register</td>
<td>Masks status byte</td>
<td>*SRE</td>
<td>*SRE?</td>
</tr>
<tr>
<td>Standard event register</td>
<td>Changes in device status</td>
<td></td>
<td>*ESR?</td>
</tr>
<tr>
<td>Standard event enable register</td>
<td>Masks standard event register</td>
<td>*ESE</td>
<td>*ESE?</td>
</tr>
<tr>
<td>Extended event register</td>
<td>Changes in device status</td>
<td></td>
<td>STATus:EEESR?</td>
</tr>
<tr>
<td>Extended event enable register</td>
<td>Masks extended event register</td>
<td>STATus:EESE</td>
<td>STATus:EESE?</td>
</tr>
<tr>
<td>Condition register</td>
<td>Current instrument status</td>
<td></td>
<td>STATus:CONDITION?</td>
</tr>
<tr>
<td>Transition filter</td>
<td>Conditions that change the</td>
<td>STATus:FILTer&lt;x&gt;</td>
<td>STATus:FILTer&lt;x&gt;?</td>
</tr>
<tr>
<td>Output queue</td>
<td>Stores a response message to a query</td>
<td></td>
<td>All query commands</td>
</tr>
<tr>
<td>Error queue</td>
<td>Stores the error number and message</td>
<td></td>
<td>STATus:ERRORor?</td>
</tr>
</tbody>
</table>

Registers and Queues That Affect the Status Byte

Registers that affect the bits of the status byte are shown below.

<table>
<thead>
<tr>
<th>Register Type</th>
<th>Affect Status Byte Bit(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard event register</td>
<td>Sets bit 5 (ESB)</td>
</tr>
<tr>
<td>Output queue</td>
<td>Sets bit 4 (MAV)</td>
</tr>
<tr>
<td>Extended event register</td>
<td>Sets bit 3 (EES)</td>
</tr>
<tr>
<td>Error queue</td>
<td>Sets bit 2 (EAV)</td>
</tr>
</tbody>
</table>

Enable Registers

Registers that are used to mask a bit so that the bit will not affect the status byte even when it is set to 1, are shown below.

<table>
<thead>
<tr>
<th>Register Type</th>
<th>Mask Bit(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status byte</td>
<td>Mask the bits using the service request enable register.</td>
</tr>
<tr>
<td>Standard event register</td>
<td>Mask the bits using the standard event enable register.</td>
</tr>
<tr>
<td>Extended event register</td>
<td>Mask the bits using the extended event enable register.</td>
</tr>
</tbody>
</table>

Writing/Reading from Registers

The "ESE" command is used to set the bits in the standard event enable register to 1’s or 0’s. The "ESE?" command is used to query whether the bits in the standard event enable register are 1’s or 0’s. For details regarding these commands, see chapter 5.
## 4.2 Status Byte

### Status Byte

<table>
<thead>
<tr>
<th>Bits 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQS MESS MAV EAV</td>
</tr>
</tbody>
</table>

- **Bits 0 and 7**
  Not used (always 0)

- **Bits 1**
  Not used (always 0)

- **Bit 2 EAV (Error Available)**
  Set to 1 when the error queue is not empty. In other words, this bit is set to 1 when an error occurs. See the page 6-6.

- **Bit 3 EES (Extend Event Summary Bit)**
  Set to 0 when the logical product of the extended event register and the corresponding enable register is 1. In other words, this bit is set to 1 when an event takes place inside the instrument. See the page 4-5.

- **Bit 4 MAV (Message Available)**
  Set to “1” when the output queue is not empty. In other words, this bit is set to 1 when there are data to be transmitted. See the page 4-6.

- **Bit 5 ESB (Event Summary Bit)**
  Set to 0 when the logical product of the standard event register and the corresponding enable register is 1. In other words, this bit is set to 1 when an event takes place inside the instrument. See the page 4-4.

- **Bit 6 RQS(Master Status Summary)**
  Set to 1 when the logical AND of the status byte excluding Bit 6 and the service request enable register is not 0. In other words, this bit is set to 1 when the instrument is requesting service from the controller.
  RQS is set to 1 when the MSS bit changes from 0 to 1, and cleared when serial polling is carried out or when the MSS bit changes to 0.

### Operation of the Status Byte

A service request is issued when bit 6 of the status byte becomes 1. Bit 6 is set to 1 when any of the other bits becomes a 1 (when the corresponding bit of the service request enable register is also set to 1).
For example, if an event occurs and the logical AND of the standard event register and the corresponding enable register becomes a 1, then bit 5 (ESB) is 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.
In addition, you can also check what type of event 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 inquiry 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 polling**
  Execution of a serial polling changes bit 6 to RQS. This causes RQS to be read. After completion of the read-out, only RQS is cleared. It is not possible to read MSS using serial polling.

### Clearing the Status Byte

No method is provided for forcibly clearing all the bits in the status byte. The bits that are cleared for each operation are shown below.

- **When a query is made using the *STB? command**
  No bits are cleared.

- **When serial polling is executed**
  Only the RQS bit is cleared.

- **When a *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.
4.3 Standard Event Register

Standard Event Register

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>PON</td>
<td>URQ</td>
<td>CME</td>
<td>EXE</td>
<td>DDE</td>
<td>QYE</td>
<td>RQC</td>
<td>OPC</td>
</tr>
</tbody>
</table>

- **Bit 7 PON (Power ON)**
  Set to 1 when the 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.
  Example: 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.
  Example: Received a command with a parameter outside the range or attempted to output a hard copy while waveform acquisition is in progress.

- **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.

- **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.
  Example: 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 (see chapter 3) has been completed.

**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).

Example
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 the register is read, it is 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 the *ESR command.
- When a *CLS command is received.
- When the instrument is power cycled.

**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. Refer to Chapter 2.

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 on the *ESE command, see chapter 3.
4.4 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.

### Condition register

```
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CAP</td>
<td>REC</td>
<td>TRG</td>
<td>CAL</td>
<td>TST</td>
</tr>
</tbody>
</table>
```

- **Bit 0** CAP (Capture): Set to 1 while waveform acquisition is in progress (including trigger wait and during pretrigger).
- **Bit 1** REC: Set to 1 while recording is in progress.
- **Bit 2** TRG (Awaiting trigger): Set to 1 when waiting for a trigger.
- **Bit 3** CAL (Calibration): Set to 1 while calibration is in progress.
- **Bit 4** TST (Testing): Set to 1 while self-test is in progress.
- **Bit 5** Reserved
- **Bit 6** ACS (Accessing): Set to 1 while a storage drive is being accessed.
- **Bit 7** ALA (Alarming): Set to 1 during system alarms.
- **Bit 8** CNT (CONnecting): Set to 1 while connected from a host.
- **Bit 9** SUP (Setup): Set to 1 while auto setup or initialization is in progress.
- **Bit 10** MES (Measuring): Set to 1 when automated measurement of waveform parameters or GO/NOGO is in progress.
- **Bit 11** Reserved
- **Bit 12** RUN (Running): Set to 1 during measurement (including during operations linked with the lighting of the start key, for example during sequential and other computations).
- **Bit 13** KLK (Key lock): Set to 1 while the key lock is engaged.
- **Bit 14** Reserved
- **Bit 15** Reserved

### Transition filter

```
:STATus:FILTer<x>
(RISE|FALL|BOTH|NEVer)
```

- **RISE** The specified bit of the extended event register is set to 1 when the bit of the condition register changes from 1 to 0.
- **FALL** The specified bit of the extended event register is set to 1 when the bit of the condition register changes from 0 to 1.
- **BOTH** The bit of the extended event register is set to 1 when the bit of the condition register changes from 0 to 1 or from 1 to 0.
- **NEVer** Always 0.

### Extended event register

```
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

The meaning of each bit of the condition register is as follows:

- **Bit 0** CAP: Set to 1 while waveform acquisition is in progress (including trigger wait and during pretrigger).
- **Bit 1** REC: Set to 1 while recording is in progress.
- **Bit 2** TRG: Set to 1 when waiting for a trigger.
- **Bit 3** CAL: Set to 1 while calibration is in progress.
- **Bit 4** TST: Set to 1 while self-test is in progress.
- **Bit 5** Reserved
- **Bit 6** ACS: Set to 1 while a storage drive is being accessed.
- **Bit 7** ALA: Set to 1 during system alarms.
- **Bit 8** CNT: Set to 1 while connected from a host.
- **Bit 9** SUP: Set to 1 while auto setup or initialization is in progress.
- **Bit 10** MES: Set to 1 when automated measurement of waveform parameters or GO/NOGO is in progress.
- **Bit 11** Reserved
- **Bit 12** RUN: Set to 1 during measurement (including during operations linked with the lighting of the start key, for example during sequential and other computations).
- **Bit 13** KLK: Set to 1 while the key lock is engaged.
- **Bit 14** Reserved
- **Bit 15** Reserved
4.5 Output Queue and Error Queue

Output Queue
The output queue is provided to store response messages to queries. For example, if you send the WAVeform:SEND? command, which requests the output of acquired data, the data is stored in the output queue until it is read. As shown below, data are stored in order and read from the oldest ones first. 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 a deadlock occurs.
- When a device clear command (DCL or SDC) is received.
- When the instrument is power cycled.

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.

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 error number and message “113, "Undefined header"” are stored in the error queue when the error is displayed. The STATus:ERRor? query can be used to read the contents of the error queue. As with the output queue, the messages are read from the oldest ones first. When the error queue overflows, the last message is replaced by the message “350, "Queue overflow".”

The error queue is also cleared for the following cases:

- When a *CLS command is received.
- When the instrument is power cycled.

To see whether the error queue is empty or not, check bit 2 (EAV) of the status byte.
# Appendix 1 ASCII Character Codes

The following table shows the ASCII character codes.

<table>
<thead>
<tr>
<th>Address (hex)</th>
<th>Universal commands</th>
<th>Listener address</th>
<th>Talker address</th>
<th>Secondary commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>02</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>03</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>04</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>05</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>06</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>07</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>08</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>09</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0A</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0B</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1A</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1B</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Example

<table>
<thead>
<tr>
<th>Octal</th>
<th>P PU</th>
<th>ASCII character code</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>NAK</td>
<td>GP-IB code</td>
</tr>
<tr>
<td>15</td>
<td>21</td>
<td>DEL (RUBOUT)</td>
</tr>
</tbody>
</table>

### Hexadecimal

<table>
<thead>
<tr>
<th>Hexadecimal</th>
<th>P PU</th>
<th>ASCII character code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x15</td>
<td>21</td>
<td>DEL (RUBOUT)</td>
</tr>
</tbody>
</table>
Appendix 2  Error Messages

This section describes the error messages related to communications.

- The messages can be displayed in English or Japanese on the SL1000. However, when the messages are read from a PC or other similar computers, the messages are displayed in English.
- If servicing is required, contact your nearest YOKOGAWA dealer for repairs.
- Only error messages related to communications are listed here. For other error messages, see User’s Manual IM 720120-61E.
  - Communication syntax error 100~199
  - Communication execution error 200~299
  - Model specific (other) 300~398
  - Communication query error 400~499
  - System error (communications) 399

Details given below.

Error in Communication Command (100-199)

<table>
<thead>
<tr>
<th>Code</th>
<th>Messages</th>
<th>Corrective Action</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>Syntax error.</td>
<td>Invalid syntax.</td>
<td>Chapter 2, 3</td>
</tr>
<tr>
<td>103</td>
<td>Invalid separator.</td>
<td>Use a comma to separate the data.</td>
<td>2-1</td>
</tr>
<tr>
<td>104</td>
<td>Data type error.</td>
<td>Write using the correct data form.</td>
<td>2-6 to 2-7</td>
</tr>
<tr>
<td>105</td>
<td>GET not allowed.</td>
<td>GET is not supported for responses to interface messages.</td>
<td>–</td>
</tr>
<tr>
<td>108</td>
<td>Parameter not allowed.</td>
<td>Check the number of data points.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>109</td>
<td>Missing parameter.</td>
<td>Enter the required data.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>111</td>
<td>Header separator error.</td>
<td>Use a space to separate the header and data.</td>
<td>2-1</td>
</tr>
<tr>
<td>112</td>
<td>Program mnemonic too long.</td>
<td>Check the mnemonic (alphanumeric character string).</td>
<td>Chapter 3</td>
</tr>
<tr>
<td>113</td>
<td>Undefined header.</td>
<td>Check the header.</td>
<td>2-4, Chapter 3</td>
</tr>
<tr>
<td>120</td>
<td>Numeric data error.</td>
<td>A number is required in the &lt;NRf&gt; form.</td>
<td>2-6</td>
</tr>
<tr>
<td>123</td>
<td>Exponent too large.</td>
<td>Use a smaller exponent for &lt;NR3&gt; format.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>124</td>
<td>Too many digits.</td>
<td>The value must be less than equal to 255 digits.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>128</td>
<td>Numeric data not allowed.</td>
<td>Enter in a format other than &lt;NRf&gt; format.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>131</td>
<td>Invalid suffix.</td>
<td>Check the unit of the &lt;Voltage&gt;, &lt;Time&gt;, &lt;Frequency&gt;, and &lt;Current&gt;.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>138</td>
<td>Suffix too long.</td>
<td>Check the spelling of the character strings in {...</td>
<td>...</td>
</tr>
<tr>
<td>141</td>
<td>Invalid character data.</td>
<td>Select character data from the selections available in {...</td>
<td>...</td>
</tr>
<tr>
<td>144</td>
<td>Character data too long.</td>
<td>Check the spelling of the character strings in {...</td>
<td>...</td>
</tr>
<tr>
<td>148</td>
<td>Character data not allowed.</td>
<td>Write in a data form other than {...</td>
<td>...</td>
</tr>
<tr>
<td>150</td>
<td>String data error.</td>
<td>Enclose &lt;String&gt; in double quotation or single quotation marks.</td>
<td>2-7</td>
</tr>
<tr>
<td>151</td>
<td>Invalid string data.</td>
<td>&lt;String&gt; is too long or contains characters which cannot be used.</td>
<td>2-7, Chapter 3</td>
</tr>
<tr>
<td>158</td>
<td>String data not allowed.</td>
<td>Enter in a data format other than &lt;Character string&gt;.</td>
<td>2-6, Chapter 3</td>
</tr>
<tr>
<td>161</td>
<td>Invalid block data.</td>
<td>&lt;Block data&gt; is not allowed.</td>
<td>2-7, Chapter 3</td>
</tr>
<tr>
<td>168</td>
<td>Block data not allowed.</td>
<td>&lt;Block data&gt; is not allowed.</td>
<td>2-7, Chapter 3</td>
</tr>
<tr>
<td>171</td>
<td>Invalid expression.</td>
<td>Equations cannot be used.</td>
<td>Chapter 3</td>
</tr>
<tr>
<td>178</td>
<td>Expression data not allowed.</td>
<td>Equations cannot be used.</td>
<td>Chapter 3</td>
</tr>
<tr>
<td></td>
<td>Invalid outside macro definition.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
</tbody>
</table>

100~199: Details given below.
### Error in Communication Execution (200 to 299)

<table>
<thead>
<tr>
<th>Code</th>
<th>Messages</th>
<th>Corrective Action</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>221</td>
<td>Setting conflict.</td>
<td>Check the relevant settings.</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>222</td>
<td>Data out of range.</td>
<td>Check the range.</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>223</td>
<td>Too much data.</td>
<td>Check the length of the data.</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>224</td>
<td>Illegal parameter value.</td>
<td>Check the range.</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>241</td>
<td>Hardware missing.</td>
<td>Check the installed options.</td>
<td>–</td>
</tr>
<tr>
<td>260</td>
<td>Expression error.</td>
<td>Equations cannot be used.</td>
<td>–</td>
</tr>
<tr>
<td>270</td>
<td>Macro error.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
<tr>
<td>272</td>
<td>Macro execution error.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
<tr>
<td>273</td>
<td>Illegal macro label.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
<tr>
<td>275</td>
<td>Macro definition too long.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
<tr>
<td>276</td>
<td>Macro recursion error.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
<tr>
<td>277</td>
<td>Macro redefinition not allowed.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
<tr>
<td>278</td>
<td>Macro header not found.</td>
<td>The SL1000 does not support the IEEE488.2 macro specifications.</td>
<td>–</td>
</tr>
</tbody>
</table>

### Error in Communication Query (400 to 499)

<table>
<thead>
<tr>
<th>Code</th>
<th>Messages</th>
<th>Corrective Action</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>410</td>
<td>Query INTERRUPTED.</td>
<td>Check transmission/reception order.</td>
<td>4-2</td>
</tr>
<tr>
<td>420</td>
<td>Query UNTERMINATED.</td>
<td>Check transmission/reception order.</td>
<td>4-2</td>
</tr>
<tr>
<td>430</td>
<td>Query DEADLOCKED.</td>
<td>Limit the length of the program message including &lt;PMT&gt; to 1024 bytes or less.</td>
<td>4-2</td>
</tr>
<tr>
<td>440</td>
<td>Query UNTERMINATED after indefinite response.</td>
<td>Do not specify a query after the *IDN? or *OPT? command.</td>
<td>–</td>
</tr>
</tbody>
</table>

### Error in System Operation (399)

<table>
<thead>
<tr>
<th>Code</th>
<th>Messages</th>
<th>Corrective Action</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>399</td>
<td>Fatal error in the communication driver.</td>
<td>Maintenance service is required.</td>
<td>–</td>
</tr>
</tbody>
</table>

### Warning (50)

<table>
<thead>
<tr>
<th>Code</th>
<th>Messages</th>
<th>Corrective Action</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>*OPC/*OPC? exists in message.</td>
<td>Place the *OPC or *OPC? command at the end of the program message.</td>
<td>–</td>
</tr>
</tbody>
</table>

### Other Errors (350)

<table>
<thead>
<tr>
<th>Code</th>
<th>Messages</th>
<th>Corrective Action</th>
<th>Reference Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>350</td>
<td>Queue overflow.</td>
<td>Read the error queue.</td>
<td>6-6</td>
</tr>
</tbody>
</table>

**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.