User's Manual

AQ7277B Remote OTDR Communication Interface

YOKOGAWA
Yokogawa Test & Measurement Corporation

IM AQ7277B-17EN
2nd Edition

Foreward

Thank you for purchasing YOKOGAWA's AQ7277B.
This Communication Interface User's Manual describes the functions and commands of USB and Ethernet interface. To ensure proper use of the USB/Ethernet interfaces, please read this manual thoroughly.Keep the manual in a safe place for quick reference whenever a question arises.

Notes

- The contents of this manual are subject to change without prior notice as a result of improvements in instrument's performance and functions.
- Every effort has been made in the preparation of this manual to ensure the accuracy of its contents. However, should you have any questions or find any errors, please contact your nearest YOKOGAWA representative.
- Copying or reproduction of all or any part of the contents of this manual without YOKOGAWA's permission is strictly prohibited.

Trademarks

- Microsoft, Internet Explorer, Windows, Windows 7, Windows 8, and Windows 8.1 are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
- Adobe, Acrobat, and PostScript are either trademarks or registered trademarks of Adobe Systems incorporated.
- For purposes of this manual, the ® and TM symbols do not accompany their respective registered trademark names or trademark names.
- Other product names are registered trademarks or trademarks of their respective holders.

Revisions

1st Edition: September 2016
2nd Edition: October 2017

2nd Edition :October 2017 (YMI)

How to Use this Manual

Structure of the Manual

This User's Manual consists of the following sections:

Chapter 1 Remote Control Function Overview

Gives an overview of the various communication interfaces.

Chapter 2 Connecting the PC

Describes the functions and specifications of the USB interface and the Ethernet Interface used to control the AQ7277B from a PC.

Chapter 3 Before Programming

Describes the syntax used to transmit commands.

Chapter 4 Remote Commands

Describes each command that is available.

Conventions Used in This Manual

Notations Used in the Procedural Explanations

On pages that describe the operating procedures in each chapter, the following notations are used to distinguish the procedure from their explanations.

Procedure This subsection contains the operating procedure used to carry out the function described in the current section. The procedures are written with inexperienced users in mind; experienced users may not need to carry out all the steps.

Explanation This subsection describes the setup parameters and the limitations on the procedures.

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

Units

Symbol	Description	Example
k	1000	400km
K	1024	459KB (file size)

Symbols Used in Syntax Descriptions

Symbols which are used in the syntax descriptions in Chapter 4 are shown below. These symbols are referred to as BNF notation (Backus-Naur Form).
For detailed information, see section 3.4, “Data.”

Symbol	Description	Example	Example of Input
<>	Defined value	`SET:M<x>  <x> = 1,2,3`	`-> SET:M2`
{}	One of the options in {} is selected.	`LMTechnique {LSA|TPA}`	`-> LMTechnique TPA`
|	Exclusive OR		

Index

1.1 Remote Interface

The AQ7277B is equipped with the following remote interfaces.

USB 2.0 Interface (Type B. See Chapter 2)

This interface is used by a controller such as a PC to remotely control the AQ7277B.
A controller is connected to this interface.
Remote commands are used to control the AQ7277B.
Dedicated commands for the AQ7277B complying with SCPI (Standard Commands for Programmable Instruments) are available for the remote commands (see chapter 4).

Ethernet Interface (See Chapter 2)

This interface is used by a controller such as a PC to remotely control the AQ7277B via the network.

Downloading the Library and Driver

The items below are needed on the PC to use the communication functions via the USB interface.

- Communication Library(TMCTL)
- USB connection device driver between the PC and AQ7277B(YKUSB)

The items below are needed on the PC to use the communication functions via the Ethernet interface.

- Communication Library(TMCTL)

The library and driver above can be downloaded from the following Web page.
http://www.yokogawa.com/tm/dl_driver.htm

1.2 Switching between Remote and Local Modes

When Switching from Local to Remote Mode

The AQ7277B is in local mode when it is being controlled with the remote controller for the AQ7277B Remote OTDR. It is in remote mode when it is being controlled through the use of communication commands.
In remote mode, the soft keys on the remote controller for the AQ7277B Remote OTDR are disabled.

- All keys except the "Local" key are disabled.
- Settings entered in local mode are passed on even when the AQ7277B switches to remote mode.

When Switching from Remote to Local Mode

In remote mode, if you press Local on the remote controller for the AQ7277B Remote OTDR, the AQ7277B switches to local mode.

- You will be able to use the remote controller for the AQ7277B Remote OTDR.
- Settings entered in remote mode are retained even when the AQ7277B switches to local mode.

2.1 Connecting via the USB

You can control the AQ7277B from a PC using the USB interface.

USB Interface Functions and Specifications

Reception Function

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

Transmission Function

Outputs measured and computed data.
Outputs setup data and the status byte.
Outputs error codes that have occurred.

USB Interface Specifications

Electrical and mechanical specifications:	Conforms to USB Rev.2.0
Connector:	Type B connector (receptacle)
Number of ports:	1
Power supply:	Self-powered
Compatible PC systems:	PCs with standard USB ports running Windows 7, Windows 8,or Windows 8.1. (A separate device driver is required to connect to a PC.)

Connection Procedure

Connecting a USB Cable to Remotely Control the AQ7277B

Connect a USB cable to the Type B connector.

Precautions to Be Taken When Connecting the Cable

- Connect the USB cable by inserting the connector firmly into the USB connector.
- Do not connect or disconnect the USB cable after the power is turned ON until the AQ7277B is ready for operation (approximately 20 s).

2.2 Connecting via the Ethernet Interface

You can control the AQ7277B from a PC by connecting the AQ7277B to a LAN using the Ethernet interface.

Ethernet Interface Functions and Specifications

Reception Function

You can specify the same settings as those specified by AQ7277B Remote Controller.
Receives output requests for measured and computed data, setup data, and error codes.

Transmission Function

Outputs measured and computed data.
Outputs setup data and the status byte.
Outputs error codes that have occurred.

Ethernet Interface Specifications

Number of communication ports:	1
Electrical and mechanical specifications:	Conforms to IEEE802.3
Transmission system:	Ethernet (10BASE-T/100BASE-TX)
Data rate:	10 Mbps/100 Mbps
Communication protocol:	TCP/IP
Connector type:	RJ45 connector
Port number used:	10001/tcp

Note

When using the Ethernet interface, a user name and password are required when connecting to the network. For the procedure to enter the user name and password, see the User's Manual of AQ7277B Remote Controller

Connection Procedure

Connect a UTP (Unshielded Twisted-Pair) cable or an STP (Shielded Twisted-Pair) cable that is connected to a hub, for example, to the 100BASE-TX port.

Precautions to Be Taken When Connecting the Cable

- Be sure to use a straight cable via a hub for the connection between the AQ7277B and the PC.
- When using a UTP cable (straight cable), use a cable of category 5.

Note

Set the TCP/IP address using the remote controller for the AQ7277B Remote OTDR. For the procedure, see the user's manual for the remote controller for the AQ7277B Remote OTDR.

3.1 Messages

Blocks of message data are transferred between the controller and this instrument during communications.Messages sent from the controller to this instrument are called program messages, and messages sent back from this instrument to the controller are called response messages.
If a program message contains a query command, i.e.a command which requests a response, this instrument returns a response message. A single response message is always returned in reply to a program message.

Program Messages

Program Message Unit

A program message consists of one or more programmessage units; each unit corresponds to one command. This instrument executes commands one by one according to the order in which they are received.
Program message units are delimited by a ";.
"For a description of the format of the program message unit, refer to the explanation given furtherbelow.

Example `:ACQuire:AVERage:MODE HIREFLECTION;TIME AUTO<PMT>`

<PMT>

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

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

<Program Header>

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

<Program Data>

If certain conditions are required for the execution of a command, program data must be added. Program data must be separated from the header by a space (ASCII code "20H"). If multiple items of program data are included, they must be separated by a "," (comma). For details, refer to page 3-5.

Example `:ACQuire:AVERage:MODE HIREFLECTION<PMT>`

Response Messages

<Response Message Units>

A response message consists of one or more response message units: each response message unit corresponds to one response.Response message units are delimited by a ";" (semicolon).

Example `:ACQuire:AVERage:MODE HIREFLECTION;TIME AUTO<PMT>`

<RMT>

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

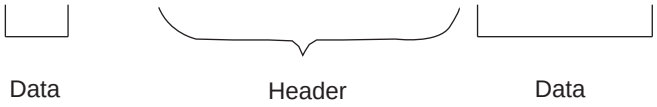
<Response Header>

A response header sometimes precedes the response data. Response data must be separated from the header by a space. For details, refer to page 3-5.

<Response Data>

Response data is used to define a response. If multiple items of response data are used, they must be separated by a "," (comma). For details, refer to page 3-5.

Example `1200<RMT>  :ACQuire:AVERage:MODE HIREFLECTION<PMT>`



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

Points to Note Concerning Message Transmission

- It is always possible to send a program message if the previous message which was sent did not contain any queries.
- If the previous message contained a query, it is not possible to send another program message until a response message has been received. An error will occur if a program message is sent before a response message has been received in its entirety. A response message which has not been receivedwill be discarded.
- If an attempt is made by the controller to receive a response message, even if there it no response message, an error will occur. An error will also occur if the controller makes an attempt to receive a response message before transmission of a program message has been completed.
- If a program message of more than one unit is sent and some of the units are incomplete, this instrument receives program message units which the instrument thinks complete and attempts to execute them. However, these attempts may not always be successful and a response may not always be returned, even if the program message contains queries.

3.2 Commands

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

Common command header

Common Command Header Commands defined in IEEE 488.2-1987 are called common commands.
An asterisk (*) must always be attached to the beginning of a command.
An example of a common command
`*CLS`

Compound header

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

When Concatenating Commands

Command Group

A command group is a group of commands which havethe same compound header. A command group maycontain sub-groups.

Example Commands relating to acquisition settings

```
:ACQuire:AVERage:MODE
:ACQuire:ATTenuation
:ACQuire:AVERage:TYPE
:ACQuire:DRANge
:ACQuire:PWIDth
:ACQuire:REALtime:STARt
```

When Concatenating Commands of the Same Group

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

Example

```
:ACQuire:AVERage:MODE HIREFLECTION;TIME AUTO<PMT>
```

When Concatenating Commands of DifferentGroups

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

Example

```
:ACQuire:AVERage:MODE HIREFLECTION;:DISPlay:CURSor
:SECond ON<PMT>
```

When Concatenating Common Commands

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

Example

```
:ACQuire:AVERage:MODE HIREFLECTION;*CLS;TIME AUTO<PMT>
```

When Separating Commands with <PMT>

If a terminator is used to separate two commands,each command is a separate message. Therefore, thecommon header must be typed in for each commandeven when commands of the same command groupare being concatenated.

Example

```
:ACQuire:AVERage:MODE HIREFLECTION<PMT>:ACQuire:AVERage
:TIME AUTO<PMT>
```

Upper-level Query

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

Example

```
:ANALysis:THReshold?<PMT>
->:ANAL:THR:EOF 3.0;FERL 40.0;FESL 1.00;RLOS 70.0
;SLOS 0.03
```

Note

- In reply to a query, a response can be returned as a program message to this instrument. Transmitting a response can restore the settings made when the query was executed.
- Not all a group's information will necessarily be sent out as a response. Some upper-levelqueries will not return setup data which is not currently in use.

Header Interpretation Rules

This instrument interprets the header receivedaccording to the following rules.

- Mnemonics are not case sensitive.
  Example
  "DRANge" can also be written as"drange" or "Drange."

- The lowercase part of a header can be omitted.
  Example
  "DRANge" can also be written as"DRANG" or "DRAN."

- If the header ends with a question mark, the command is a query. It is not possible to omit the question mark.
  Example
  "DRANge?" cannot be abbreviated to anything shorter than "DRAN?."

- If the "x" at the end of a mnemonic is omitted, it is assumed to be "1."
  ExampleIf "M<x>" is written as "M,"this represents "M1."

Note

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

3.3 Response

Form

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

Response Consisting of a Header and Data

If the query can be used as a program messagewithout any change, a command header is attachedto the query, which is then returned.

Example :ACQuire:AVERage:MODE?<PMT>
->:ACQuire:AVERage:MODE HIREFLECTION<RMT>

Response Consisting of Data Only

If the query cannot be used as a program messageunless changes are made to it (i.e. it is a query-onlycommand), no header is attached and only the datais returned. Some query-only commands can bereturned after a header is attached to them.

Example :MEASure:CHANnel1:PTOPeak:VALue?<PMT> -> 10.0E+00<RMT>

When Returning a Response without a Header

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

Abbreviated Form

Normally, the lowercase part is removed from are sponse header before the response is returned to the controller. Naturally, the full form of the header canalso be used. For this, the
"COMMunicate:VERBose" command is used. The part enclosed by [ ] is also omitted in the abbreviatedform.

3.4 Data

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

Data	Description
<Decimal>	Value expressed as a decimal number (Example: Average time -> :ACQuire:AVERage:TIME 60)
<Distance><Time> <Wavelength><Loss>	Physical value (Example: Distance range -> :ACQuire:DRANge 500)
<Register>	Register value expressed as either binary,octal, decimalor hexadecimal (Example: Extended event register value -> :STATus:EESE #HFE)
<Character data>	Specified character string (mnemonic). Can be selected from { } (Example: Setup mode -> :ACQuire:AVERage:MODE {HIREFLECTION|HISPEED})
<Boolean>	Indicates ON/OFF. Set to ON or OFF (Example: Fault event display -> :ANALysis:FEDetection ON)
<Character string data>	Arbitrary character string (Example: Comment to be saved -> :FILE:SAVe:COMMent "ABCDEF")
<Filename>	Gives the name of a file. (Example: Name of file to be saved -> :FILE:SAVE:WAVeform:NAME "CASE1")
<Block data>	Arbitrary 8-bit data (Example: Response to acquired waveform data -> #800000010ABCDEFGHIJ)

<Decimal>

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

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

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

<Distance>, <Time>, <Wavelength>, and <Loss>

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

Form	Example
<NRf><Multiplier><Unit>	0.85UM
<NRf><Unit>	500m
<NRf><Multiplier>	5M
<NRf>	5E -3

<Multiplier>

Multipliers which can be used are shown below.

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

<Unit>

Units which can be used are shown below.

Symbol	Word	Description
M	Meter	Distance
S	Second	Time
dB	Decibel	Level
UM	Micro meter	Wavelength

- <Multiplier> and <Unit> are not case sensitive.
- "U" is used to indicate "µ."
- "MA" is used for Mega (M) to distinguish it from Milli.
- If both <Multiplier> and <Unit> are omitted, thedefault unit will be used.

<Register>

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

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

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

<Character Data>

<Character data> is a specified string of character data(a mnemonic). It is mainly used to indicate options,and is chosen from the character strings given in { }.
For interpretation rules, refer to “Header Interpretation Rules” on page 3-5.

Form	Example
{HIREFLECTION|HISPEED}	HISPEED

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

<Boolean>

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

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

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

<Character String Data>

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

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

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

<Filename>

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

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

- If you input an <NRf> value, the system converts the value (after rounding to the nearest integer) to the corresponding 8-character ASCII string. (If you set the value to 1, the name becomes "00000001".) Note that negative values are not allowed.
- The first 12 characters of <Character Data> are assigned as the file name.
- The first 14 characters of <Character String Data> are assigned as the file name.
- Response messages always return filenames as <character string> arguments.

<Block Data>

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

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

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

3.5 Synchronization with the Controller

Achieving Synchronization

If you send the following program message when receiving averaging measurement data, the data may be retrieved before the completion of the measurement.

```
ACQuire: AVERage: STARt; : WAVedata: DISPlay: SEND: ASCii?
```

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

Using the STATus:CONDition? Query

The "`STATus:CONDition?`" command is used to query the contents of the condition register.

Whether averaging measurement waveforms are being retrieved can be determined by reading bit 1 of the condition register.

If bit 1 of the condition register is "`1`", waveforms are being retrieved. Otherwise, it is stopped.(Bit 1 is "`0`".)

Note

For details on the condition register, see section 4.4, "Condition Register."

4.1 List of Commands

Common Commands

Command	Function	Page
*CLS(Clear Status)	Clears all event status registers, the summary of which is reflected in the status byte register.	4-5
*ESE(Standard Event Status Enable)	Sets/queries the standard event enable register.	4-5
*ESR?(Standard Event Status Register)	Queries the standard event status register and simultaneously clears it.	4-5
*IDN? (Identification)	Queries the instrument type and firmware version.	4-5
*RST (Reset)	Executes a device reset to return the instrument to the known (default) status.	4-5
*SRE(Service Request Enable)	Sets/queries the service request enable register.	4-5
*STB?(Read Status Byte)	Queries the current value of the status byte register.	4-6
*TST?(Self Test)	Performs the instrument's self-test and queries the status.	4-6

Device-Specific Commands

ACQuire Group

:ACQuire:ADSave	Sets or queries auto data saving to files after executing averaged measurement.	4-7
:ACQuire:AESearch	Sets or queries auto event detection after executing averaged measurement.	4-7
:ACQuire:ATTenuation	Sets or queries the attenuation.	4-7
:ACQuire:AUTO:ATTenuation?	Sets or queries the attenuation for auto mode.	4-7
:ACQuire:AUTO:DRANge?	Queries the distance range for auto mode.	4-7
:ACQuire:AUTO:PWIDth?	Queries the pulse width for auto mode.	4-7
:ACQuire:AVERage:COUNt	Queries the current average count.	4-7
:ACQuire:AVERage:INDex	Sets or queries the average times.	4-7
:ACQuire:AVERage:MODE	Sets or queries the average method.	4-7
:ACQuire:AVERage:STARt	Executes averaged measurement.	4-7
:ACQuire:AVERage:STOP	Stops averaged measurement.	4-8
:ACQuire:AVERage:TIME	Sets or queries the average duration.	4-8
:ACQuire:AVERage:TYPE	Sets or queries the average unit.	4-8
:ACQuire:DRANge	Sets or queries the distance range.	4-8
:ACQuire:MAXdata	Sets or queries the maximum number of data points.	4-8
:ACQuire:OFFSet	Sets or queries the measurement start position on the horizontal axis.	4-8
:ACQuire:PLUGcheck	Sets or queries the optical plug connection check.	4-9
:ACQuire:PON:STATe	Sets or queries PON (high SNR mode).	4-9
:ACQuire:PWIDth	Sets or queries the pulse width.	4-9
:ACQuire:REALtime:STARt	Executes realtime measurement.	4-9
:ACQuire:REALtime:STOP	Stops realtime measurement.	4-9
:ACQuire:SELection:ATTenuation?	Queries for a list of attenuators.	4-9
:ACQuire:SELection:DRANge?	Queries for a list of distance ranges.	4-9
:ACQuire:SELection:PWIDth?	Queries for a list of pulse widths.	4-9
:ACQuire:SELection:SMPinterval?	Queries for a list of sampling intervals.	4-9
:ACQuire:SELection:WAVelength?	Queries for a list of wavelengths.	4-9
:ACQuire:SMPinterval:DATA	Sets or queries the sampling interval.	4-10
:ACQuire:SMPinterval:VALue?	Queries the sampling interval.	4-10
:ACQuire:WAVelength	Sets or queries the measurement wavelength.	4-10

ANALysis Group

:ANALysis:ASEarch:EXECute	Executes auto detection.	4-10
:ANALysis:ASEarch:NUMBer?	Queries the number of automatically detected events.	4-10
:ANALysis:BCOefficient	Sets or queries a backscattering light level of the current wavelength.	4-10
:ANALysis:CURSor:DECibel?	Queries the cursor dB (decibels).	4-10
:ANALysis:CURSor:DELete	Delete the cursor.	4-10
:ANALysis:CURSor:DISTance	Sets or queries the cursor position.	4-10
:ANALysis:CURSor:LINK	Sets or queries the cursor link.	4-11
:ANALysis:DUNit	Sets or queries the distance unit.	4-11
:ANALysis:EMARker:LMTechnique	Sets or queries the approximation method (event).	4-11
:ANALysis:EMARker:SET:M1	Sets or queries marker M1 of the current event.	4-11

4.1 A List of Commands

Command	Function	Page
:ANALysis:EMARker:SET:M2	Sets or queries marker M2 of the current event.	4-11
:ANALysis:EMARker:SET:M3	Sets or queries marker M3 of the current event.	4-11
:ANALysis:EMARker:SET:Y2	Sets or queries marker Y2 of the current event.	4-11
:ANALysis:EVENt:CURRent:CUMLoss?	Queries the cumulate loss of the current event.	4-11
:ANALysis:EVENt:CURRent:DISTance?	Queries the distance of the current event.	4-11
:ANALysis:EVENt:CURRent:INDex	Moves the current event.	4-11
:ANALysis:EVENt:CURRent:IOR?	Queries the section group index of the current event.	4-11
:ANALysis:EVENt:CURRent:LOSS?	Queries the splice loss of the current event.	4-11
:ANALysis:EVENt:CURRent:NOTE	Sets or queries the event note.	4-12
:ANALysis:EVENt:CURRent:PEAK?	Queries the peak value of the current event.	4-12
:ANALysis:EVENt:CURRent:RETurnloss?	Queries the return loss of the current event.	4-12
:ANALysis:EVENt:CURRent:TYPE?	Queries the event type of the current event.	4-12
:ANALysis:EVENt:CURRent:UNITloss?	Queries the dB/km of the current event.	4-12
:ANALysis:EVENt:DELete	Deletes the current event.	4-12
:ANALysis:EVENt:FIX:MODE	Sets or queries the event fix mode.	4-12
:ANALysis:EVENt:FIX:STATe	Sets or queries the event fix state.	4-12
:ANALysis:EVENt:INSert	Inserts an event at the cursor position.	4-12
:ANALysis:EVENt:IOR	Sets or queries the section group index of the current event.	4-12
:ANALysis:FEDetection	Sets or queries the fault event display.	4-12
:ANALysis:FMARker:DELete	Deletes markers.	4-12
:ANALysis:FMARker:LEFT:DISTance?	Queries the distance between markers 1 and 2.	4-12
:ANALysis:FMARker:LEFT:LOSS?	Queries the loss between markers 1 and 2.	4-12
:ANALysis:FMARker:LEFT:UNITloss?	Queries the slope between markers 1 and 2.	4-13
:ANALysis:FMARker:LMTechnique	Sets or queries the marker approximation method.	4-13
:ANALysis:FMARker:LOSS?	Queries the splice loss.	4-13
:ANALysis:FMARker:REFLection:SATurated?	Queries the saturated state of the reflection level.	4-13
:ANALysis:FMARker:REFLection:VALue?	Queries the reflection level.	4-13
:ANALysis:FMARker:RETurnloss:SATurated?	Queries the saturated state of the return loss.	4-13
:ANALysis:FMARker:RETurnloss:VALue?	Queries the return loss.	4-13
:ANALysis:FMARker:RIGHt:DISTance?	Queries the distance between markers 2 and 3.	4-13
:ANALysis:FMARker:RIGHt:LOSS?	Queries the loss between markers 2 and 3.	4-13
:ANALysis:FMARker:RIGHt:UNITloss?	Queries the slope between markers 2 and 3.	4-13
:ANALysis:FMARker:SET:M<x>	Sets or queries the specified marker.	4-13
:ANALysis:FMARker:SET:Y<x>	Sets or queries the specified auxiliary marker.	4-13
:ANALysis:IOR	Sets or queries the index of refraction of the current wavelength.	4-13
:ANALysis:REFerence:DELete	Delete the distance reference.	4-14
:ANALysis:REFerence:DISTance	Sets the distance reference.	4-14
:ANALysis:SECTion:BASelevel?	Queries the dB value of the reference point of the section data.	4-14
:ANALysis:SECTion:DELete	Clears the interval analysis data.	4-14
:ANALysis:SECTion:DISTance?	Queries the distance of the section data.	4-14
:ANALysis:SECTion:END	Sets or queries the end position of the section data.	4-14
:ANALysis:SECTion:LMTechnique	Sets or queries the section analysis approximation method.	4-14
:ANALysis:SECTion:LOSS?	Queries the loss of the section data.	4-14
:ANALysis:SECTion:REFerence	Sets the reference point of the section data.	4-14
:ANALysis:SECTion:RETurnloss:SATurated?	Queries the saturated state of the return loss of the section data.	4-14
:ANALysis:SECTion:RETurnloss:VALue?	Queries the return loss of the section data.	4-14
:ANALysis:SECTion:STARt	Sets or queries the start position of the section data.	4-14
:ANALysis:THReshold:EOFiber	Sets or queries the end-of-fiber threshold.	4-14

Command	Function	Page
:ANALysis:THReshold:FECLoss	Sets or queries the connector loss threshold of the fault event.	4-15
:ANALysis:THReshold:FERLoss	Sets or queries the return loss threshold of the fault event.	4-15
:ANALysis:THReshold:FESLoss	Sets or queries the splice loss threshold of the fault event.	4-15
:ANALysis:THReshold:FESPloss	Sets or queries the splitter loss threshold of the fault event.	4-15
:ANALysis:THReshold:FETLoss	Sets or queries the total loss threshold of the fault event.	4-15
:ANALysis:THReshold:FEULoss	Sets or queries the dB/km threshold of the fault event.	4-15
:ANALysis:THReshold:RLOSs	Sets or queries the return loss threshold.	4-15
:ANALysis:THReshold:SLOSs	Sets or queries the splice loss threshold.	4-15
:ANALysis:THReshold:SPLOss	Sets or queries the splitter loss threshold.	4-16
COMMunicate Group		
:COMMunicate:HEADer	Sets or queries whether a header is added to the response to a query (example with header: "ACQuire:DRANge 5000"; example without header: "5000").	4-16
:COMMunicate:VERBose	Sets or queries whether the response to a query is returned fully spelled out (e.g., ACQuire:DRANge 5000) or in its abbreviated form (e.g., ACQ:DRAN 5000).	4-16
DISPlay Group		
:DISPlay:DECibel:UPPer	Sets or queries the display start level.	4-17
:DISPlay:DISTance:LEFT	Sets or queries the display start distance.	4-17
:DISPlay:DIVide:DECibel	Sets or queries the magnification of the vertical axis.	4-17
:DISPlay:DIVide:DISTance	Sets or queries the magnification of the horizontal axis.	4-17
FILE Group		
:FILE:CSV:EVENt:GET?	Queries the event list in CSV format.	4-17
:FILE:CSV:EVENt:VALid?	Queries whether the event list can be output in CSV format.	4-17
:FILE:CSV:TRACe:GET?	Queries the waveform data in CSV format.	4-17
:FILE:CSV:TRACe:VALid?	Queries whether the waveform data can be output in CSV format.	4-17
:FILE:DRIVe:FREE?	Queries the free space on the current drive.	4-17
:FILE:DRIVe:SET	Sets or queries the current drive setting.	4-17
:FILE:FILE:GET?	Queries the specified file.	4-18
:FILE:FILE:EXISt?	Checks whether the specified file exists.	4-18
:FILE:FILE:NAME	Specifies the name of the file that will be handled with file retrieve, file size retrieve, and file send commands.	4-18
:FILE:FILE:SEND	Sends the file.	4-18
:FILE:FILE:SIZE?	Queries the size of the specified file.	4-18
:FILE:FOLDer:LIST?	Queries for a list of current folders.	4-18
:FILE:FOLDer:MAKE	Creates a folder.	4-18
:FILE:FOLDer:PATH	Sets or queries the current folder name.	4-18
:FILE:LOAD:EXECute	Plays the file.	4-18
:FILE:SAVE:COMMent<x>	Sets or queries the save comment.	4-18
:FILE:SAVE:EXECute	Saves the file.	4-18
:FILE:SAVE:ID	Sets or queries the ID number when saving the file.	4-19
:FILE:SAVE:ITEM<x>	Sets or queries the file save item.	4-19
:FILE:SAVE:SEParator	Sets or queries the file save separator.	4-19
:FILE:SAVE:SUB	Sets or queries the sub number when saving the file.	4-19
:FILE:SAVE:TYPE	Sets or queries the file name type when saving the file.	4-19
:FILE:SOR:GET?	Queries the SOR file image.	4-19
:FILE:SOR:VALid?	Queries whether the SOR file image is enabled.	4-19
:FILE:SUBFolder:LIST?	Queries for a list of sub folders in the current folder.	4-20
:FILE:TYPE	Sets or queries the file type when saving the file.	4-20
LABel Group		
:LABel:CABLe:CODE	Sets or queries the cable code.	4-20
:LABel:CABLe:ID	Sets or queries the cable ID.	4-20
:LABel:COMPany	Sets or queries the company name.	4-20
:LABel:DFLag:CURRent	Sets or queries the current data flag.	4-20
:LABel:FIBer:ID	Sets or queries the fiber ID.	4-20
:LABel:FIBer:TYPE	Sets or queries the fiber type.	4-21
:LABel:LABel	Sets or queries the label.	4-21
:LABel:LOCation:ORIGinating	Sets or queries the start position label.	4-21
:LABel:LOCation:TERMinating	Sets or queries the end position label.	4-21
:LABel:OPERator	Sets or queries the operator name.	4-21

Command	Function	Page
MENU Group		
:MENU:ERRor:CLEar	Deletes the error dialog.	4-21
:MENU:FUNCtion	Sets or queries the function mode.	4-21
:MENU:MARKer	Sets or queries the marker mode.	4-21
MISC Group		
:MISC:DATE:DAY	Sets or queries the day.	4-22
:MISC:DATE:GET?	Queries the date and time.	4-22
:MISC:DATE:HOUR	Sets or queries the time.	4-22
:MISC:DATE:MINute	Sets or queries the minute.	4-22
:MISC:DATE:MODE	Sets or queries the date display type.	4-22
:MISC:DATE:MONTh	Sets or queries the month.	4-22
:MISC:DATE:SECond	Sets or queries the second.	4-22
:MISC:DATE:SET	Executes the changing of the date and time.	4-22
:MISC:DATE:YEAR	Sets or queries the year.	4-22
:MISC:LANGuage	Sets or queries the language.	4-23
:MISC:RLOSsmode	Sets or queries the return loss display.	4-23
NETWork Group		
:NETWork:CONTrol:PASSword	Sets or queries the password.	4-23
:NETWork:CONTrol:TIMeout	Sets or queries the timeout value.	4-23
:NETWork:CONTrol:USERname	Sets or queries the user name.	4-23
:NETWork:DHCP	Turns DHCP on and off.	4-23
:NETWork:GATeway	Sets or queries the gateway.	4-24
:NETWork:IPADdress	Sets or queries the IP address.	4-24
:NETWork:NETMask	Sets or queries the net mask.	4-24
:NETWork:STATe	Turns the network on or off.	4-24
SETup Group		
:SETup:ALLinit	Initializes all settings to their factory default values and formats the internal memory.	4-24
:SETup:INITialize	Initializes all settings to their factory default values.	4-24
STATus Group		
:STATus:CONDition?	Queries the contents of the condition register.	4-25
:STATus:ERRor?	Queries the error code and message of the last error that has occurred (top of the error queue).	4-25
:STATus:QENable	Sets or queries whether messages other than errors will be stored to the error queue.	4-25
:STATus:QMESsage	Sets or queries whether message information will be attached to the response to the STATus:ERRor? query.	4-25
SYSTem Group		
:SYSTem:REBoot	Restarts the system.	4-25
WAVedata Group		
:WAVedata:DISPlay:SEND:ASCii?	Queries the displayed waveform data in ASCII format.	4-26
:WAVedata:DISPlay:SEND:BINary?	Queries the displayed waveform data in binary format.	4-26
:WAVedata:LENGth?	Queries the number of waveform data points.	4-26
:WAVedata:OLDType:DISPlay:SEND?	Queries the displayed waveform data in Dot4 format.	4-26
:WAVedata:OLDType:SEND?	Queries the waveform data in Dot4 format.	4-26
:WAVedata:SEND:ASCii?	Queries the waveform data in ASCII format.	4-26
:WAVedata:SEND:BINary?	Queries the waveform data in binary format.	4-26
:WAVedata:SEND:SIZE	Sets or queries the number of retrieved waveform data points.	4-26
:WAVedata:SEND:STARt	Sets or queries the start distance of the waveform data.	4-27

4.2 Common Commands

*CLS(Clear Status)

Function Clears all event status registers, the summary of which is reflected in the status byte register.

Syntax `*CLS`

Example `*CLS`

Description
- Clears all queues, with the exception of the output queue, and all event registers, with the exception of the MAV summary message.
- After executing this command, OCIS (Operation Complete Command Idle State) and OQIS (Operation Complete Query Idle State) are brought about.

*ESE(Standard Event Status Enable)

Function Sets/queries the standard event enable register.

Syntax `*ESE<wsp><integer>`
`*ESE?`
`<integer> = 0 to 255`

Example `*ESE 251`
`*ESE? -> 251`

Description
- An item having had its bit set becomes enabled.
- Resets to the default value in the following cases:
  When power is ON
  When “0” is set
- The set value remains the same in the following cases:
  *RST
  *CLS
  Device clear (DCL, SDC)
- The default is 0.

*ESR?(Standard Event Status Register)

Function Queries the standard event status register and simultaneously clears it.

Syntax `*ESR?`

Example `*ESR? -> 251`

Description The return value of this query is not affected by ESE (Event Status Enable Register).

*IDN? (Identification)

Function Queries the instrument type and firmware version.

Syntax `*IDN?`

Example `*IDN? -> YOKOGAWA,AQ7277B-B02,`
`SN123456789,F1.00`
`SN123456789: Serial number`
`(9 digit string)`
`F1.00:Firmware version`

Description Outputs 4 field data delimited by a comma.
Field 1: Manufacturer “YOKOGAWA”
Field 2: Model “AQ7277B-B02”
Field 3: Instrument serial number "SN123456789"
Field 4: Software version "F1.00"

*RST (Reset)

Function Executes a device reset to return the instrument to the known (default) status.

Syntax `*RST`

Example `*RST`

Description
- Stops operation being processed and returns the instrument to the known set value (default value) immediately.
- This unit’s parameters are cleared.
- The following items will remain the same.Output queue
  SRE
  ESE
  Calibration data affecting the instrument’s specifications

*SRE(Service Request Enable)

Function Sets/queries the service request enable register.

Syntax `*SRE <wsp><integer>`
`*SRE?`
`<integer> = 0 to 255`

Example `*SRE 250`
`*SRE? -> 250`

Description
- An item having had its bit set becomes enabled.
- Resets to the default value in the following cases:
  When power is ON
  When “0” is set
- The set value remains the same in the following cases:
  *RST
  *CLS
  Device clear (DCL, SDC)
- The default is 0.

*STB?(Read Status Byte)

Function Queries the current value of the status byte register.

Syntax `*STB?`

Example `*STB? -> 251`

Description STB will not be cleared even when the contents of the register are read.

*TST?(Self Test)

Function Performs the instrument's self-test and queries the status.

Syntax `*TST?`

Example `*TST? -> 0`

Description Executes the operations below among the initialization sequence at startup and outputs the results. The screen retains the waveform screen while the initialization is in progress.
Internal battery check
Internal memory read check
"0" is returned if both checks are successful, "1" if they are not.

4.3 Device-Specific Commands

ACQuire Group

The commands in this group deal with the waveform acquisition. Change the measurement conditions through remote control while the measurement is stopped. You cannot change the measurement conditions while realtime or averaging measurement is in progress. Only :REALtime:STOP or AVERage:STOP is valid while the measurement is in progress.

:ACQuire:ADSave

Function Sets or queries auto data saving to files after executing averaged measurement.

Syntax
```
:ACQuire:ADSave {<Boolean>}
:ACQuire:ADSave?
```

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

Description If automatic saving is set to ON, the auto save mode is set to “user defined.”

:ACQuire:AESearch

Function Sets or queries auto event detection after executing averaged measurement.

Syntax
```
:ACQuire:AESearch {<Boolean>}
:ACQuire:AESearch?
```

Example
```
:ACQUIRE:AESEARCH OFF
:ACQUIRE:AESEARCH?
-> :ACQUIRE:AESEARCH 0
```

:ACQuire:ATTenuation

Function Sets or queries the attenuation.

Syntax
```
:ACQuire:ATTenuation {<NRf>|AUTO}
:ACQuire:ATTenuation?
```
<NRf> = 0.00 to 35.00 (in 2.5 dB steps)

Example
```
:ACQUIRE:ATTENUATION 2.5
:ACQUIRE:ATTENUATION AUTO
:ACQUIRE:ATTENUATION?
-> :ACQUIRE:ATTENUATION 25.00
```

Description
- The range that can be specified varies depending on the wavelength, distance range, and pulse width.
- You cannot set the attenuation when the average method is Hi-Reflection.
- You cannot set the attenuation when the distance range is set to auto.

:ACQuire:AUTO:ATTenuation?

Function Sets or queries the attenuation for auto mode.

Syntax
```
:ACQuire:AUTo:ATTenuation?
```

Example
```
:ACQUIRE:AUTO:ATTENUATION?
-> :ACQ:AUTO:ATTENUATION 0.00
```

:ACQuire:AUTO:DRANge?

Function Queries the distance range for auto mode.

Syntax
```
:ACQuire:AUTo:DRANge?
```

Example
```
:ACQUIRE:AUTO:DRANGE?
-> :ACQ:AUTO:DRANGE 20000
```

Description The unit is m (meters).

:ACQuire:AUTO:PWIDth?

Function Queries the pulse width for auto mode.

Syntax
```
:ACQuire:AUTo:PWIDth?
```

Example
```
:ACQUIRE:AUTO:PWIDTH?
-> :ACQ:AUTO:PWIDTH 100E-09
```

Description The unit is s (seconds).

:ACQuire:AVERage:COUNt

Function Queries the current average count.

Syntax
```
:ACQuire:AVERage:COUNt?
```

Example
```
:ACQUIRE:AVERAGE:COUNT?
-> ACQUIRE:AVERAGE:COUNT 0
```

:ACQuire:AVERage:INDex

Function Sets or queries the average times.

Syntax
```
:ACQuire:AVERage:INDex{AUTO|E2_10
|E2_11…….E2_20}
:ACQuire:AVERage:INDex?
AUTO  :  Auto
E2_10  :  2E+10
E2_11  :  2E+11
-------------
E2_20  :  2E+20
```

Example
```
:ACQUIRE:AVERAGE:INDEX AUTO
:ACQUIRE:AVERAGE:INDEX E2_12
:ACQUIRE:AVERAGE:INDEX?
-> :ACQ:AVER:IND E2_15
```

:ACQuire:AVERage:MODE

Function Sets or queries the average method.

Syntax
```
:ACQuire:AVERage:MODE
{HIREFLECTION|HISPEED}
:ACQuire:AVERage:MODE?
HIREFLECTION :  Hi-Reflection
HISPEED      :  Hi-Speed
```

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

:ACQuire:AVERage:STARt

Function Executes averaged measurement.

Syntax
```
:ACQuire:AVERage:STARt
```

Example
```
:ACQUIRE:AVERAGE:START
```

Description This command can be used only when measurement is stopped.

:ACQuire:AVERage:STOP

Function Stops averaged measurement.
Syntax :ACQuire:AVERage:STOP
Example :ACQUIRE:AVERAGE:STOP
Description This command can be used only when averaged measurement is in progress.

:ACQuire:AVERage:TIME

Function Sets or queries the average duration.
Syntax :ACQuire:AVERage:TIME {<NRf>|AUTO}
:ACQuire:AVERage:TIME?
<NRf> = 5, 10, 20, 30, 60, 180, 300, 600, 1200, 1800 (s)
AUTO : Auto
5 : 5 s
10 : 10 s
20 : 20 s
30 : 30 s
60 : 1 min
180 : 3 min
300 : 5 min
600 : 10 min
1200 : 20 min
1800 : 30 min
Example :ACQUIRE:AVERAGE:TIME AUTO
:ACQUIRE:AVERAGE:TIME 1200
:ACQUIRE:AVERAGE:TIME? ->
:ACQUIRE:AVERAVE:TIME 60

:ACQuire:AVERage:TYPE

Function Sets or queries the average unit.
Syntax :ACQuire:AVERage:TYPE {TIMES|DURATION}
ACQuire:AVERage:TYPE?
TIMES : Times
DURATION : Time
Example :ACQUIRE:AVERAGE:TYPE TIMES
:ACQUIRE:AVERAGE:TYPE DURATION
:ACQUIRE:AVERAGE:TYPE?
-> :ACQUIRE:AVERAGE:TYPE DURATION

:ACQuire:DRANge

Function Sets or queries the distance range.
Syntax :ACQuire:DRANge {<NRf>|AUTO}
:ACQuire:DRANge?
<NRf> = 500m to 512000m(0.5km to 512km)
AUTO : Auto:
500 : 500 m
1000 : 1 km
2000 : 2 km
5000 : 5 km
10000 : 10 km
20000 : 20 km
30000 : 30 km
50000 : 50 km
100000 : 100 km
200000 : 200 km
300000 : 300 km
400000 : 400 km
512000 : 512 km
Example :ACQUIRE:DRANGE AUTO
:ACQUIRE:DRANGE 500
:ACQUIRE:DRANGE 500M
:ACQUIRE:DRANGE 5000
:ACQUIRE:DRANGE 100KM
:ACQUIRE:DRANGE?
-> :ACQUIRE:DRANGE 1000
Description The values that you can specify vary depending on the wavelength setting.

:ACQuire:MAXdata

Function Sets or queries the maximum number of data points.
Syntax :ACQuire:MAXdata {DATA50K|DATA256K}
:ACQuire:MAXdata?
DATA50K : 50000
DATA256K : 256000
Example :ACQUIRE:MAXdata DATA50K
Description Set the maximum number of data points that can be measured in a single measurement.

:ACQuire:OFFSet

Function Sets or queries the measurement start position on the horizontal axis.
Syntax :ACQuire:OFFSet {<NRf>}
:ACQuire:OFFSet?
Example :ACQUIRE:OFFSET 1000
:ACQUIRE:OFFSET?
-> :ACQUIRE:OFFSET 999.99265E+00
Description The unit is m (meters).

:ACQuire:PLUGcheck

Function Sets or queries the optical plug connection check.
Syntax :ACQuire:PLUGcheck {<Boolean>}
:ACQuire:PLUGcheck?
Example :ACQUIRE:PLUGCHECK ON
:ACQUIRE:PLUGCHECK?
-> :ACQUIRE:PLUGCHECK 0

:ACQuire:PON:STATe

Function Sets or queries PON (high SNR mode).
Syntax :ACQuire:PON:STATe {<Boolean>}
:ACQuire:PON:STATe?
Example :ACQUIRE:PON:STATE ON
:ACQUIRE:PON:STATE?
-> :ACQUIRE:PON:STATE 0

:ACQuire:PWIDth

Function Sets or queries the pulse width.
Syntax :ACQuire:PWIDth {<NRf>|AUTO}
:ACQuire:PWIDth?
<NRf> = 3ns to 10us(3E-9 to 10E-6)
AUTO : Auto
3E-09 : 3 ns
10E-09 : 10 ns
20E-09 : 20 ns
30E-09 : 30 ns
50E-09 : 50 ns
100E-09 : 100 ns
200E-09 : 200 ns
300E-09 : 300 ns
500E-09 : 500 ns
1E-06 : 1 s
2E-06 : 2 s
5E-06 : 5 s
10E-06 : 10 s
20E-06 : 20 s
Example :ACQUIRE:PWIDTH AUTO
:ACQUIRE:PWIDTH 3E-9
:ACQUIRE:PWIDTH 3NS
:ACQUIRE:PWIDTH 10US
:ACQUIRE:PWIDTH 10E-6
:ACQUIRE:PWIDTH?
-> :ACQUIRE:PWIDTH 10.0E-06
Description
- The value that can be specified varies depending on the wavelength and distance range.
- You cannot set the attenuation when the distance range is set to auto.

:ACQuire:REALtime:STARt

Function Executes realtime measurement.
Syntax :ACQuire:REALtime:STARt
Example :ACQUIRE:REALTIME:START
Description This command can be used only when measurement is stopped.

:ACQuire:REALtime:STOP

Function Stops realtime measurement.
Syntax :ACQuire:REALtime:STOP
Example :ACQUIRE:REALTIME:STOP
Description This command can be used only when realtime measurement is in progress.

:ACQuire:SELection:ATTenuation?

Function Queries for a list of attenuators.
Syntax :ACQuire:SELection:ATTenuation?
Example :ACQUIRE:SELection:ATTenuation?
Description :ACQUIRE:SELECTION:ATTENUATION?
-> :ACQUIRE:SELECTION:ATTENUATION
0.00,2.50,5.00,…, 27.50

:ACQuire:SELection:DRANge?

Function Queries for a list of distance ranges.
Syntax :ACQuire:SELection:DRANge?
Example :ACQUIRE:SELection:DRANge?
Description :ACQUIRE:SELECTION:DRANGE?
-> :ACQUIRE:SELECTION:DRANGE
500,1000,…,4000000

:ACQuire:SELection:PWIDth?

Function Queries for a list of pulse widths.
Syntax :ACQuire:SELection:PWIDth?
Example :ACQUIRE:SELection:PWIDth?
Description :ACQUIRE:SELECTION:PWIDTH?
-> :ACQUIRE:SELECTION:PWIDTH
3E-9.10E-9, …, 20E-6

:ACQuire:SELection:SMPinterval?

Function Queries for a list of sampling intervals.
Syntax :ACQuire:SELection:SMPinterval?
Example :ACQUIRE:SELection:SMPInterval?
Description :ACQUIRE:SELECTION:SMPINTERVAL?
-> :ACQUIRE:SELECTION:SMPINTERVAL
0.05,0.10,0.20,…4.00

:ACQuire:SELection:WAVelength?

Function Queries for a list of wavelengths.
Syntax :ACQuire:SELection:WAVelength?
Example :ACQUIRE:SELection:WAVelength?
Description :ACQUIRE:SELECTION:WAVELENGTH?
-> :ACQUIRE:SELECTION:WAVELENGTH
1.310E-6.1.550E-6

:ACQuire:SMPinterval:DATA

Function Sets or queries the sampling interval.
Syntax :ACQuire:SMPinterval:DATA{<NRf>|NORMAL|HI}
<NRf>: 0.2 to 32 (m)
:ACQuire:SMPinterval:DATA?
Example :ACQUIRE:SMPINTERVAL:DATA 2.0
:ACQUIRE:SMPINTERVAL:DATA NORMAL
:ACQUIRE:SMPINTERVAL:DATA? ->
:ACQ:SMP:DATA HI
Description You can only set the attenuation to NORMAL or HI when the distance range is set to auto.

Normal	:	NORMAL
High resolution	:	HI
2 cm	:	0.02
5 cm	:	0.05
10 cm	:	0.10
20 cm	:	0.20
50 cm	:	0.50
1m	:	1.00
2m	:	2.00
4m	:	4.00
8m	:	8.00
16m	:	16.00
32m	:	32.00

:ACQuire:SMPinterval:VALue?

Function Queries the sampling interval.
Syntax :ACQuire:SMPinterval:VALue?
Example :ACQUIRE:SMPINTERVAL:VALUE?
-> :ACQUIRE:SMPINTERVAL:VALUE 8.0

:ACQuire:WAVelength

Function Sets or queries the measurement wavelength.
Syntax :ACQuire:WAVelength {<NRf>}
:ACQuire:WAVelength?
<NRf> =0.850 m to 1.650 m
(850e-9 to 1650E-9)
Example :ACQUIRE:WAVELENGTH 0.85UM
:ACQUIRE:WAVELENGTH 1650E-9
:ACQUIRE:WAVELENGTH 1.650E-6
:ACQUIRE:WAVELENGTH?
-> :ACQUIRE:WAVELENGTH 1550E-09
Description The wavelengths that you can specify vary depending on the model.
Measure wavelength settings
1310 nm : 1310E-09
1550 nm : 1550E-09

ANALysis Group

The commands in this group deal with waveform analysis.

:ANALysis:ASEarch:EXECute

Function Executes auto detection.
Syntax :ANALysis:ASEarch:EXECute
Example :ANALYSIS:ASEARCH:EXECUTE
Description After execution, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:ASEarch:NUMBer?

Function Queries the number of automatically detected events.
Syntax :ANALysis:ASEarch:NUMber?
Example :ANALYSIS:ASEARCH:NUMBER?
-> :ANALYSIS:ASEARCH:NUMBER 2

:ANALysis:BCOefficient

Function Sets or queries a backscattering light level of the current wavelength.
Syntax :ANALysis:BCOefficient {<NRf>}
:ANALysis:BCOefficient?
<NRf> = -64.99 to -10.00 (in 0.01 steps)
Example :ANALYSIS:BCOEFFICIENT -25.00
:ANALYSIS:BCOEFFICIENT?
-> :ANALYSIS:BCOEFFICIENT -50.00

:ANALysis:CURSor:DECibel?

Function Queries the cursor dB (decibels).
Syntax :ANALysis:CURSor:DECibel?
Example :ANALYSIS:CURSOR:DECIBEL?
-> :ANALYSIS:CURSOR:DECIBEL 32.878
Description Least squares method : LSA
2-point markers : TPA

:ANALysis:CURSor:DELete

Function Delete the cursor.
Syntax :ANALysis:CURSor:DELete
Example :ANALYSIS:CURSOR:DELETE

:ANALysis:CURSor:DISTance

Function Sets or queries the cursor position.
Syntax :ANALysis:CURSor:DISTance {<NRf>}
:ANALysis:CURSor:DISTance?
<NRf> = 0 to distance range (m)
Example :ANALYSIS:CURSOR:DISTANCE 10.00
:ANALYSIS:CURSOR:DISTANCE?
-> :ANALYSIS:CURS:DISTANCE
11.529900E+00
Description This depends on the distance range, distance reference, and index of refraction.

:ANALysis:CURSor:LINK
Function Sets or queries the cursor link.
Syntax :ANALysis:CURSor:LINK {<Boolean>}
:ANALysis:CURSor:LINK?
Example :ANALYSIS:CURSOR:LINK ON
:ANALYSIS:CURSOR:LINK?
-> :ANALYSIS:CURSOR:LINK 1

:ANALysis:DUNit
Function Sets or queries the distance unit.
Syntax :ANALysis:DUNit {KM|MILE|KF}
:ANALysis:DUNit?
KM : km (kilometers)
MILE : mile (miles)
KF : kf (kilofeet)
Example :ANALYSIS:DUNIT KM
:ANALYSIS:DUNIT?
-> :ANALYSIS:DUNIT MILE

:ANALysis:EMARker:LMTechnique
Function Sets or queries the approximation method (event).
Syntax :ANALysis:EMARker:LMTechnique {LSA|TPA}
:ANALysis:EMARker:LMTechnique?
LSA : Least squares method
TPA : 2-point marker
Example :ANALYSIS:EMARKER:LMTECHNIQUE LSA
:ANALYSIS:EMARKER:LMTECHNIQUE?
-> :ANALYSIS:EMARKER:LMTECHINIQUE TPA

:ANALysis:EMARker:SET:M1
Function Sets or queries marker M1 of the current event.
Syntax :ANALysis:EMARker:SET:M1
:ANALysis:EMARker:SET:M1?
Example :ANALYSIS:EMARKER:SET:M1
:ANALYSIS:EMARKER:SET:M1?
-> :ANALYSIS:EMARKER:SET:M1
5.0471900E+03
Description The marker is set to the cursor position.

:ANALysis:EMARker:SET:M2
Function Sets or queries marker M2 of the current event.
Syntax :ANALysis:EMARker:SET:M2
:ANALysis:EMARker:SET:M2?
Example :ANALYSIS:EMARKER:SET:M2
:ANALYSIS:EMARKER:SET:M2?
-> :ANALYSIS:EMARKER:SET:M2
7.2463500E+03
Description The marker is set to the cursor position.

:ANALysis:EMARker:SET:M3
Function Sets or queries marker M3 of the current event.
Syntax :ANALysis:EMARker:SET:M3
:ANALysis:EMARker:SET:M3?
Example :ANALYSIS:EMARKER:SET:M3
:ANALYSIS:EMARKER:SET:M3?
-> :ANALYSIS:EMAR:SET:M3
9.0471600E+03
Description The marker is set to the cursor position.

:ANALysis:EMARker:SET:Y2
Function Sets or queries marker Y2 of the current event.
Syntax :ANALysis:EMARker:SET:Y2
:ANALysis:EMARker:SET:Y2?
Example :ANALYSIS:EMARKER:SET:Y2
:ANALYSIS:EMARKER:SET:Y2?
-> :ANALYSIS:EMARKER:SET:Y2
7.3777700E+03
Description The marker is set to the cursor position.

:ANALysis:EVENt:CURRent:CUMLoss?
Function Queries the cumulate loss of the current event.
Syntax :ANALysis:EVENt:CURRent:CUMLoss?
Example :ANALYSIS:EVENT:CURRENT:CUMLOSS?
-> 1.810
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:DISTance?
Function Queries the distance of the current event.
Syntax :ANALysis:EVENt:CURRent:DISTance?
Example :ANALYSIS:EVENT:CURRENT:DISTANCE? -> 987.000
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:INDex
Function Moves the current event.
Syntax :ANALysis:EVENt:CURRent:INDex {<NRf>}
<NRf> = event number (-1 to 100)
1 to 100 : Event number
-1 : Point S, point R
0 : Point E
Example :ANALYSIS:EVENT:CURRENT:INDEX 5
Description The event of the specified number becomes the current event.

:ANALysis:EVENt:CURRent:IOR?
Function Queries the section group index of the current event.
Syntax :ANALysis:EVENt:CURRent:IOR?
Example :ANALYSIS:EVENT:CURRENT:IOR?
-> 1.46000
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:LOSS?
Function Queries the splice loss of the current event.
Syntax :ANALysis:EVENt:CURRent:LOSS?
Example :ANALYSIS:EVENT:CURRENT:LOSS?
-> 2.000
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:NOTE
Function Sets or queries the event note.
Syntax :ANALysis:EVENt:CURRent:NOTE {<String>}
:ANALysis:EVENt:CURRent:NOTE?
<String> = Up to 36 characters
Example :ANALYSIS:EVENt:CURRent:NOTE "aaa"
:ANALysis:EVENt:CURRent:NOTE?
-> :ANAL:EVEN:CURR:NOTE "123"
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.
A comment is entered for the current event (up to 36 characters).

:ANALysis:EVENt:CURRent:PEAK?
Function Queries the peak value of the current event.
Syntax :ANALysis:EVENt:CURRent:PEAK?
Example :ANALYSIS:EVENT:CURRENT:PEAK?
-> 2.000
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:RETurnloss?
Function Queries the return loss of the current event.
Syntax :ANALysis:EVENt:CURRent:RETurnloss?
Example :ANALYSIS:EVENT:CURRENT:RETURNLOSS?
-> 2.000
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:TYPE?
Function Queries the event type of the current event.
Syntax :ANALysis:EVENt:CURRent:TYPE?
REFLectance : Reflection
SPLus : Positive loss
SMINus : Negative loss
Example :ANALYSIS:EVENT:CURRENT:TYPE?
-> REFLECTANCE
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:CURRent:UNITloss?
Function Queries the dB/km of the current event.
Syntax :ANALysis:EVENt:CURRent:UNITloss?
Example :ANALYSIS:EVENT:CURRENT:UNITLOSS?
-> 2.000
Description Use the :ANALysis:EVENt:CURRent:INDex command to set the index number first, and then use this command.

:ANALysis:EVENt:DELete
Function Deletes the current event.
Syntax :ANALysis:EVENt:DELete
Example :ANALYSIS:EVENT:DELETE

:ANALysis:EVENt:FIX:MODE
Function Sets or queries the event fix mode.
Syntax :ANALysis:EVENt:FIX:MODE
Example :ANALYSIS:EVENT:FIX:MODE
:ANALYSIS:EVENT:FIX:MODE?
-> :ANALYSIS:EVENT:FIX:MODE

:ANALysis:EVENt:FIX:STATe
Function Sets or queries the event fix state.
Syntax :ANALysis:EVENt:FIX:STATe
Example :ANALYSIS:EVENT:FIX:STATE
:ANALYSIS:EVENT:FIX:STATE?
-> :ANALYSIS:EVENT:FIX:STATE

:ANALysis:EVENt:INSert
Function Inserts an event at the cursor position.
Syntax :ANALysis:EVENt:INSert
Example :ANALYSIS:EVENT:INSERT

:ANALysis:EVENt:IOR
Function Sets or queries the section group index of the current event.
Syntax :ANALysis:EVENt:IOR {<NRf>}
:ANALysis:EVENt:IOR?
<NRf> = 1.30000 to 1.79999 (in 0.00001 steps)
Example :ANALYSIS:EVENT:IOR 1.47
:ANALYSIS:EVENT:IOR?
-> :ANALYSIS:EVENT:IOR 1.46000

:ANALysis:FEDetection
Function Sets or queries the fault event display.
Syntax :ANALysis:FEDetection {<Boolean>}
:ANALysis:FEDetection?
Example :ANALYSIS:FEDETECTION ON
:ANALYSIS:FEDETECTION?
-> :ANALYSIS:FEDETECTION 1

:ANALysis:FMARker:DELete
Function Deletes markers.
Syntax :ANALysis:FMARker:DELete
Example :ANALYSIS:FMARKER:DELETE

:ANALysis:FMARker:LEFT:DISTance?
Function Queries the distance between markers 1 and 2.
Syntax :ANALysis:FMARker:LEFT:DISTance?
Example :ANALYSIS:FMARKER:LEFT:DISTANCE?
-> :ANALYSIS:FMARKER:LEFT:DISTANCE
490.20000E+00

:ANALysis:FMARker:LEFT:LOSS?
Function Queries the loss between markers 1 and 2.
Syntax :ANALysis:FMARker:LEFT:LOSS?
Example :ANALYSIS:FMARKER:LEFT:LOSS?
-> :ANALYSIS:FMARKER:LEFT:LOSS
137.00000E-03

:ANALysis:FMARker:LEFT:UNITloss?

Function Queries the slope between markers 1 and 2.
Syntax :ANALysis:FMARker:LEFT:UNITloss?
Example :ANALYSIS:FMARKER:LEFT:UNITLOSS?
-> :ANALYSIS:FMARKER:LEFT:UNITLOSS 279.00000E-03

:ANALysis:FMARker:LMTechnique

Function Sets or queries the marker approximation method.
Syntax :ANALysis:FMARker:LMTechnique {LSA|TPA}
:ANALysis:FMARker:LMTechnique?
LSA : Least squares method
TPA : 2-point marker
Example :ANALYSIS:FMARKER:LMTECHNIQUE LSA
:ANALYSIS:FMARKER:LMTECHNIQUE?
-> :ANALYSIS:FMARKER:LMTEC TPA

:ANALysis:FMARker:LOSS?

Function Queries the splice loss.
Syntax :ANALysis:FMARker:LOSS?
Example :ANALYSIS:FMARKER:LOSS?
-> :ANALYSIS:FMARKER:LOSS 162.00000E-03

:ANALysis:FMARker:REFLection:SATurated?

Function Queries the saturated state of the reflection level.
Syntax :ANALysis:FMARker:REFLection:SATurated?
0 : Not saturated
1 : Saturated
Example :ANALYSIS:FMARKER:REFLECTION:SATURATED?
-> :ANALYSIS:FMARKER:REFLECTION:SATURATED 0

:ANALysis:FMARker:REFLection:VALue?

Function Queries the reflection level.
Syntax :ANALysis:FMARker:REFLection:VALue?
Example :ANALYSIS:FMARKER:REFLECTION:VALUE?
-> :ANALYSIS:FMARKER:REFLECTION:VALUE 1.0640000E+00

:ANALysis:FMARker:RETurnloss:SATurated?

Function Queries the saturated state of the return loss.
Syntax :ANALysis:FMARker:RETurnloss:SATurated?
0 : Not saturated
1 : Saturated
Example :ANALYSIS:FMARKER:RETURNLOSS:SATURATED?
->:ANALYSIS:FMARKER:RETURN:SATURATED 1

:ANALysis:FMARker:RETurnloss:VALue?

Function Queries the return loss.
Syntax :ANALysis:FMARker:RETurnloss:VALue?
Example :ANALYSIS:FMARKER:RETURNLOSS:VALUE?
->:ANALYSIS:FMARKER:RETURNLOSS:VALUE 47.003000E+00

:ANALysis:FMARker:RIGHt:DISTance?

Function Queries the distance between markers 2 and 3.
Syntax :ANALysis:FMARker:RIGHt:DISTance?
Example :ANALYSIS:FMARKER:RIGHT:DISTANCE?
-> ANALYSIS:FMARKER:RIGHT:DISTANCE 232.95000E+00

:ANALysis:FMARker:RIGHt:LOSS?

Function Queries the loss between markers 2 and 3.
Syntax :ANALysis:FMARker:RIGHt:LOSS?
Example :ANALYSIS:FMARKER:RIGHT:LOSS?
-> :ANALYSIS:FMARKER:RIGHT:LOSS -159.00000E-03

:ANALysis:FMARker:RIGHt:UNITloss?

Function Queries the slope between markers 2 and 3.
Syntax :ANALysis:FMARker:RIGHt:UNITloss?
Example :ANALYSIS:FMARKER:RIGHT:UNITLOSS?
-> :ANALYSIS:FMARKER:RIGHT:UNITLOSS -683.00000E-03

:ANALysis:FMARker:SET:M<x>

Function Sets or queries the specified marker.
Syntax :ANALysis:FMARker:SET:M<x>
:ANALysis:FMARker:SET:M<x>?
<x> = 1, 2, 3
Example :ANALYSIS:FMARKER:SET:M1
:ANALYSIS:FMARKER:SET:M1?
-> ANALYSIS:FMARKER:SET:M1 7.2648300E+03

:ANALysis:FMARker:SET:Y<x>

Function Sets or queries the specified auxiliary marker.
Syntax :ANALysis:FMARker:SET:Y<x>
:ANALysis:FMARker:SET:Y<x>?
<x> = 1, 2, 3
Example :ANALYSIS:FMARKER:SET:Y2
:ANALYSIS:FMARKER:SET:Y2?
-> ANALYSIS:FMARKER:SET:Y2 6.1786000E+03

:ANALysis:IOR

Function Sets or queries the index of refraction of the current wavelength.
Syntax :ANALysis:IOR {<NRf>}
:ANALysis:IOR?
<NRf> = 1.30000 to 1.79999 (in 0.00001 steps)
Example :ANALYSIS:IOR 1.48000
:ANALYSIS:IOR?
-> :ANALYSIS:IOR 1.48000

:ANALysis:REFerence:DELete
Function Delete the distance reference.
Syntax :ANALysis:REFerence:DELete
Example :ANALYSIS:REFERENCE:DELETE

:ANALysis:REFerence:DISTance
Function Sets the distance reference.
Syntax :ANALysis:REFerence:DISTance
:ANALysis:REFerence:DISTance?
Example :ANALYSIS:REFERENCE:DISTANCE
:ANALYSIS:REFERENCE:DISTANCE?
-> :ANAL:REF:DIST 39.999710E+03

:ANALysis:SECTion:BASelevel?
Function Queries the dB value of the reference point of the section data.
Syntax :ANALysis:SECTion:BASelevel?
Example :ANALYSIS:SECTION:BASELEVEL?
-> :ANALYSIS:SECTION:BASELEVEL
34.268000E+00

:ANALysis:SECTion:DELete
Function Clears the interval analysis data.
Syntax :ANALysis:SECTion:DELete
Example :ANALYSIS:SECTION:DELETE

:ANALysis:SECTion:DISTance?
Function Queries the distance of the section data.
Syntax :ANALysis:SECTion:DISTance?
Example :ANALYSIS:SECTION:DISTANCE?
-> :ANALYSIS:SECTION:DISTANCE
80.262240E+03

:ANALysis:SECTion:END
Function Sets or queries the end position of the section data.
Syntax :ANALysis:SECTion:END {<NRf>}
:ANALysis:SECTion:END?
<NRf>: Start position
Example :ANALYSIS:SECTION:END
:ANALYSIS:SECTION:END?
-> :ANALYSYS:SECTION:END
119.99912E+03

:ANALysis:SECTion:LMTechnique
Function Sets or queries the section analysis approximation method.
Syntax :ANALysis:SECTion:LMTechnique
<LSA|TPA>
:ANALysis:SECTion:LMTechnique?
LSA : Least squares method
TPA : 2-point marker
Example :ANALYSIS:SECTION:LMTECHNIQUE
:ANALYSIS:SECTION:LMTECHNIQUE?
-> :ANALYSIS:SECTION:LMTECHNIQUE LSA

:ANALysis:SECTion:LOSS?
Function Queries the loss of the section data.
Syntax :ANALysis:SECTion:LOSS?
Example :ANALYSIS:SECTION:LOSS?
-> :ANALYSIS:SECTION:LOSS
7.9300000E+00

:ANALysis:SECTion:REFerence
Function Sets the reference point of the section data.
Syntax :ANALysis:SECTion:REFerence
Example :ANALYSIS:SECTION:REFERENCE

:ANALysis:SECTion:RETurnloss:SATurated?
Function Queries the saturated state of the return loss of the section data.
Syntax :ANALysis:SECTion:RETurnloss:
SATurated?
Example :ANALYSIS:SECTION:RETURNLOSS:
SATURATED?
-> :ANALYSIS:SECTION:RETURN:
SATURATED 1

:ANALysis:SECTion:RETurnloss:VALue?
Function Queries the return loss of the section data.
Syntax :ANALysis:SECTion:RETurnloss:VALue?
Example :ANALYSIS:SECTION:RETURNLOSS:VALUE?
-> :ANALYSIS:SECTION:RETURNLOSS:VALUE
48.250000E+00

:ANALysis:SECTion:STARt
Function Sets or queries the start position of the section data.
Syntax :ANALysis:SECTion:STARt {<NRf>}
:ANALysis:SECTion:STARt?
<NRf>: End position
Example :ANALYSIS:SECTION:START
:ANALYSIS:SECTION:START?
-> :ANALYSIS:SECTION:START
39.736870E+03

:ANALysis:THReshold:EOFiber
Function Sets or queries the end-of-fiber threshold.
Syntax :ANALysis:THReshold:EOFiber {<NRf>}
:ANALysis:THReshold:EOFiber?
<NRf> = 3 to 10
Example :ANALYSIS:THRESHOLD:EOFIBER 5
:ANALYSIS:THRESHOLD:EOFIBER?
-> :ANALYSIS:THRESHOLD:EOFIBER 10
Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:FECLoss

Function Sets or queries the connector loss threshold of the fault event.

Syntax
```
:ANALysis:THReshold:FECLoss {<NRf>}
:ANALysis:THReshold:FECLoss?
```
<NRf> = 0.01 to 9.99

Example
```
:ANALYSIS:THRESHOLD:FECLOSS 0.01
:ANALYSIS:THRESHOLD:FECLOSS?
-> :ANALYSIS:THRESHOLD:FECLOSS 1.23
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:FERLoss

Function Sets or queries the return loss threshold of the fault event.

Syntax
```
:ANALysis:THReshold:FERLoss {<NRf>}
:ANALysis:THReshold:FERLoss?
```
<NRf> = 20 to 70

Example
```
:ANALYSIS:THRESHOLD:FERLOSS 50
:ANALYSIS:THRESHOLD:FERLOSS?
-> :ANALYSIS:THRESHOLD:FERLOSS 30
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:FESLoss

Function Sets or queries the splice loss threshold of the fault event.

Syntax
```
:ANALysis:THReshold:FESLoss {<NRf>}
:ANALysis:THReshold:FESLoss?
```
<NRf> 0.01 to 9.99

Example
```
:ANALYSIS:THRESHOLD:FESLOSS 0.01
:ANALYSIS:THRESHOLD:FESLOSS?
-> :ANALYSIS:THRESHOLD:FESLOSS 1.23
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:FESPloss

Function Sets or queries the splitter loss threshold of the fault event.

Syntax
```
:ANALysis:THReshold:FESPloss {<NRf>}
:ANALysis:THReshold:FESPloss?
```
<NRf> 1 to 20

Example
```
:ANALYSIS:THRESHOLD:FESPLOSS 1
:ANALYSIS:THRESHOLD:FESPLOSS?
-> :ANALYSIS:THRESHOLD:FESPLOSS 20
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:FETLoss

Function Sets or queries the total loss threshold of the fault event.

Syntax
```
:ANALysis:THReshold:FETLoss {<NRf>}
:ANALysis:THReshold:FETLoss?
```
<NRf> 1 to 65

Example
```
:ANALYSIS:THRESHOLD:FETLOSS 1
:ANALYSIS:THRESHOLD:FETLOSS?
-> :ANALYSIS:THRESHOLD:FETLOSS 65
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:FEULoss

Function Sets or queries the dB/km threshold of the fault event.

Syntax
```
:ANALysis:THReshold:FEULoss {<NRf>}
:ANALysis:THReshold:FEULoss?
```
<NRf> 0.01 to 9.99

Example
```
:ANALYSIS:THRESHOLD:FEULOSS 0.01
:ANALYSIS:THRESHOLD:FEULOSS?
-> :ANALYSIS:THRESHOLD:FEULOSS 1.23
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:RLOSs

Function Sets or queries the return loss threshold.

Syntax
```
:ANALysis:THReshold:RLOSs {<NRf>}
:ANALysis:THReshold:RLOSs?
```
<NRf> = 20 to 70

Example
```
:ANALYSIS:THRESHOLD:RLOSS 50
:ANALYSIS:THRESHOLD:RLOSS?
-> :ANALYSIS:THRESHOLD:RLOSS 30
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:SLOSs

Function Sets or queries the splice loss threshold.

Syntax
```
:ANALysis:THReshold:SLOSs {<NRf>}
:ANALysis:THReshold:SLOSs?
```
<NRf> = 0.01 to 9.99

Example
```
:ANALYSIS:THRESHOLD:SLOSS 0.01
:ANALYSIS:THRESHOLD:SLOSS?
-> :ANALYSIS:THRESHOLD:SLOSS 1.23
```

Description After setting the value, wait for the AS flag in the value obtained with STATus:CONDition? to become zero, and then obtain the event results.

:ANALysis:THReshold:SPLOss

Function Sets or queries the splitter loss threshold.

Syntax
```
:ANALysis:THReshold:SPLOSs {<NRf>}
:ANALysis:THReshold:SPLOSs?
```
<NRf> = 1 to 20

Example
```
:ANALYSIS:THRESHOLD:SPLOSS 1
:ANALYSIS:THRESHOLD:SPLOSS?
-> :ANALYSIS:THRESHOLD:SPLOSS 20
```

Description After setting the value, wait for the AS flag in the value obtained with `STATus:CONDition?` to become zero, and then obtain the event results.

COMMunicate Group

The commands in this group deal with communications.

:COMMunicate:HEADer

Function Sets or queries whether a header is added to the response to a query (example with header: "ACQuire:DRANge 5000"; example without header: "5000").

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

Example
```
:COMMUNICATE:HEADER ON
:COMMUNICATE:HEADER? ->:COMM:HEAD 1
```

:COMMunicate:VERBose

Function Sets or queries whether the response to a query is returned fully spelled out (e.g., ACQuire:DRANge 5000) or in its abbreviated form (e.g., ACQ:DRAN 5000).

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

Example
```
:COMMUNICATE:VERBOSE ON
:COMMUNICATE:VERBOSE? ->:COMM:VERB 0
```

DISPlay Group

The commands in this group deal with the screen display.

:DISPlay:DECibel:UPPer

Function Sets or queries the display start level.
Syntax `:DISPlay:DECibel:UPPer {<NRf>}`
`:DISPlay:DECibel:UPPer?`
<NRf> = 1.6 to 70(dB)
Example `:DISPLAY:DECIBEL:UPPER 60`
`:DISPLAY:DECIBEL:UPPER?`
`-> :DISPLAY:DECIBEL:UPPER 50.0`
Description The range that can be specified varies depending on magnification of the vertical axis.

:DISPlay:DISTance:LEFT

Function Sets or queries the display start distance.
Syntax `:DISPlay:DISTance:LEFT {<NRf>}`
`DISPlay:DISTance:LEFT?`
Example `:DISPLAY:DISTANCE:LEFT 1000`
`DISPLAY:DISTANCE:LEFT?`
`-> :DISPLAY:DISTANCE:LEFT`
`498.97000E+00`
Description The range that can be specified varies depending on magnification of the horizontal axis.

:DISPlay:DIVide:DECibel

Function Sets or queries the magnification of the vertical axis.
Syntax `:DISPlay:DIVide:DECibel {<NRf>}`
`:DISPlay:DIVide:DECibel?`
<NRf> = 0.2 to 7.5 (0.2, 0.5, 1.0, 2.0, 5.0, 7.5) (dB)
Example `:DISPLAY:DIVIDE:DECIBEL 1.0`
`:DISPLAY:DIVIDE:DECIBEL?`
`-> :DISPLAY:DIVIDED:DECIBEL 1.0`
Description The unit is dB (decibels).

:DISPlay:DIVide:DISTance

Function Sets or queries the magnification of the horizontal axis.
Syntax `:DISPlay:DIVide:DISTance {<NRf>}`
`:DISPlay:DIVide:DISTance?`
<NRf> = 1m to 40km
Example `:DISPLAY:DIVIDE:DISTANCE 2000`
`:DISPLAY:DIVIDE:DISTANCE?`
`-> :DISPLAY:DIVIDE:DISTNCE 10.0E+03`
Description The range that can be specified varies depending on the distance range.
100k range : 50 m to 10 km
50k range : 20 m to 5 km
20k range : 10 m to 2 km
10k range : 5 m to 1 km
5k range : 5 m to 500 m
2k range : 2 m to 200 m
1k range : 1 m to 100 m
500m range : 1 m to 50 m

FILE Group

The commands in this group can output event lists and waveform data in CSV and other file formats.

:FILE:CSV:EVENt:GET?

Function Queries the event list in CSV format.
Syntax `:FILE:CSV:EVENt:GET?`
Example `:FILE:CSV:EVENT:GET?`
`-> #812345678....`
Description Block data format.

:FILE:CSV:EVENt:VALid?

Function Queries whether the event list can be output in CSV format.
Syntax `:FILE:CSV:EVENt:VALid?`
Example `:FILE:CSV:EVENT:VALID?`
`-> ON`

:FILE:CSV:TRACe:GET?

Function Queries the waveform data in CSV format.
Syntax `:FILE:CSV:TRACe:GET?`
Example `:FILE:CSV:TRACE:GET?`
`-> #812345678....`
Description Block data format.

:FILE:CSV:TRACe:VALid?

Function Queries whether the waveform data can be output in CSV format.
Syntax `:FILE:CSV:TRACe:VALid?`
Example `:FILE:CSV:TRACE:VALID`
`-> ON`

:FILE:DRIVe:FREE?

Function Queries the free space on the current drive.
Syntax `:FILE:DRIVe:FREE?`
Example `:FILE:DRIVE:FREE?`
`->:FILE:DRIV:FREE 190.70976E+06`
Description The value is the number of bytes.

:FILE:DRIVe:SET

Function Sets or queries the current drive setting.
Syntax `:FILE:DRIVe:SET`
`{INTERNAL|SD|USB1|USB2}`
`:FILE:DRIVe:SET?`
Example `:FILE:DRIVE:SET USB1`
`:FILE:DRIVE:SET?`
`-> :FILE:DRIV:SET INTERNAL`
Description Driver settings
Internal memory: INTERNAL
SD : External SD memory card
USB1 : USB memory
USB2 : USB memory
The report output destination will also be the drive specified by this command.

:FILE:FILE:GET?
Function Queries the specified file.
Syntax :FILE:FILE:GET?
Example :FILE:FILE:GET?
-> #220ABCDEFGHIJ1234567890
Description Block data header is inserted in front of the loaded data.

:FILE:FILE:EXISt?
Function Checks whether the specified file exists.
Syntax :FILE:FILE:EXISt?
Example :FILE:FILE:EXIST?
-> :FILE:FILE:EXIST 1
Description 1 if the file exists.
0 if the file does not exist.

:FILE:FILE:NAME
Function Specifies the name of the file that will be handled with file retrieve, file size retrieve, and file send commands.
Syntax :FILE:FILE:NAME {<String>}
:FILE:FILE:NAME?
Example :FILE:FILE:NAME "0.SOR"
:FILE:FILE:NAME?
-> :FILE:FILE:NAME "0.SOR"
Description The following characters cannot be used in folder names.
" * / : < > ? \ |
The following character strings cannot be used in folder names.
"AUX", "CON", "PRN", "NUL", "CLOCK",
"LPT1", "LPT2", "LPT3", "LPT4", "LPT5",
"LPT6", "LPT7", "LPT8", "LPT9",
"COM1", "COM2", "COM3", "COM4", "COM5",
"COM6", "COM7", "COM8", "COM9"

:FILE:FILE:SEND
Function Sends the file.
Syntax :FILE:FILE:SEND?
Example :FILE:FILE:SEND <BINARY>
Description !!!DELETE!!!

:FILE:FILE:SIZE?
Function Queries the size of the specified file.
Syntax :FILE:FILE:SIZE?
Example :FILE:FILE:SIZE?
-> :FILE:FILE:SIZE 230781

:FILE:FOLDer:LIST?
Function Queries for a list of current folders.
Syntax :FILE:FOLDer:LIST?
Example :FILE:FOLDER:LIST?
-> :FILE:FOLDER:LIST "3,ABC.SOR,DEF.SOR,MACRO/"
Description The information is returned as follows: number of items, file (or folder), . . . , file (or folder). A slash is appended to folder names.

:FILE:FOLDer:MAKE
Function Creates a folder.
Syntax :FILE:FOLDer:MAKE {<String>}
<String> = Folder name
Example :FILE:FOLDER:MAKE "DATA"

:FILE:FOLDer:PATH
Function Sets or queries the current folder name.
Syntax :FILE:FOLDer:PATH {<String>}
:FILE:FOLDer:PATH?
<String> = Folder name
Example :FILE:FOLDER:PATH "AQ7277B"
:FILE:FOLDER:PATH?
-> :FILE:FOLD:PATH "AQ7277B"
Description Moves one level at a time from the current position (".." returns to one level higher).
The following characters cannot be used in folder names.
" * / : < > ? \ |
The following character strings cannot be used in folder names.
"AUX", "CON", "PRN", "NUL", "CLOCK",
"LPT1", "LPT2", "LPT3", "LPT4", "LPT5",
"LPT6", "LPT7", "LPT8", "LPT9",
"COM1", "COM2", "COM3", "COM4", "COM5",
"COM6", "COM7", "COM8", "COM9"
The report output destination will also be the path specified by this command.

:FILE:LOAD:EXECute
Function Plays the file.
Syntax :FILE:LOAD:EXECute {<String>}
<String> = File name
Example :FILE:LOAD:EXECUTE "0001.SOR"

:FILE:SAVE:COMMent<x>
Function Sets or queries the save comment.
Syntax :FILE:SAVE:COMMent<x> {<String>}
<x>=1 to 10
:FILE:SAVE:COMMent<x>?
Example :FILE:SAVE:COMMENT<x> "AQ7277B"
:FILE:SAVE:COMMENT<x>?
->"AQ7277B"
Description Up to 30 characters can be specified.

:FILE:SAVE:EXECute
Function Saves the file.
Syntax :FILE:SAVE:EXECute
Example :FILE:SAVE:EXECUTE
Description To check for errors after saving, use STATus:CONDition? to verify that file access is finished, and then use STATus:ERRor.

:FILE:SAVE:ID

Function Sets or queries the ID number when saving the file.

Syntax `:FILE:SAVE:ID {<NRf>}`
`:FILE:SAVE:ID?`
<NRf> = 0 to 9999

Example `:FILE:SAVE:ID 100`
`:FILE:SAVE:ID?`
`-> :FILE:SAVE:ID 100`

:FILE:SAVE:ITEM<x>

Function Sets or queries the file save item.

Syntax `:FILE:SAVE:ITEM<x> {NONE|WAVELENGTH|ID|COMMENT1|COMMENT2|COMMENT3|COMMENT4|COMMENT5|COMMENT6|COMMENT7|COMMENT8|COMMENT9|COMMENT10|COMPANY|OPERATOR|CABLEID|FIBERID|CABLECODE|ORIGINATING|TERMINATING}`
<x>=1 to 10
`:FILE:SAVE:ITEM<x>?`

NONE	:	None
WAVELENGTH	:	Wavelength
ID	:	ID No.
COMMENT1 to 10	:	Comment 1 to 10
COMPANY	:	Company name
OPERATOR	:	Name
CABLEID	:	Cable ID
FIBERID	:	Fiber ID
FIBERTYPE	:	Fiber type
CABLECODE	:	Cable code
ORIGINATING	:	Start position
TERMINATING	:	End position

Example `:FILE:SAVE:ITEM1 WAVELENGTH`
`:FILE:SAVE:ITEM1?`
`-> :FILE:SAVE:ITEM WAVELENGTH`

Description
- The specified items are added to the file name in order from <x>=1.
- If <x>=1, NONE cannot be specified.
- You can use `:FILE:SAVE:TYPE` to set the file name in a simple manner.

:FILE:SAVE:SEParator

Function Sets or queries the file save separator.

Syntax `:FILE:SAVE:SEParator {NONE|UNDERBAR|TILDE|HAT}`
`:FILE:SAVE:SEParator?`

NONE	:	None
UNDERBAR	:	Underscore (_)
TILDE	:	Tilde (~)
HAT	:	Hat (^)

Example `:FILE:SAVE:SEPARATORUNDERBAR`
`:FILE:SAVE:SEPARATOR?`
`-> :FILE:SAVE:SEPARATOR UNDERBAR`

:FILE:SAVE:SUB

Function Sets or queries the sub number when saving the file.

Syntax `:FILE:SAVE:SUB {OFF|AB|AC|AD|AE|AF|AG|AH}`
`:FILE:SAVE:SUB?`

OFF	:	OFF
AB	:	a-b
AC	:	a-c
AD	:	a-d
AE	:	a-e
AF	:	a-f
AG	:	a-g
AH	:	a-h

Example `:FILE:SAVE:SUB AH`
`:FILE:SAVE:SUB?`
`-> :FILE:SAVE:SUB AH`

:FILE:SAVE:TYPE

Function Sets or queries the file name type when saving the file.

Syntax `:FILE:SAVE:TYPE {NO|COMMENT|CMNO|NOCM|CMWLNO|NOCMWL|WLCMNO}:FILE:SAVE:TYPE?`

NO	:	Number
COMMENT	:	Comment
CMNO	:	Comment + number
NOCM	:	Number + comment
CMWLNO	:	Comment + wavelength + number
NOCMWL	:	Number + comment + wavelength
WLCMNO	:	Wavelength + comment + number

Example `:FILE:SAVE:TYPE COMMENT`
`:FILE:SAVE:TYPE?`
`-> :FILE:SAVE:TYPE COMMENT`

Description
- This is a simplified version of `:FILE:SAVE:ITEM<x>`.

:FILE:SOR:GET?

Function Queries the SOR file image.

Syntax `:FILE:SOR:GET?`

Example `:FILE:SOR:GET?`
`-> #6123456ABCDEFGHJ`

Description This query can be used only when `:FILE:SOR:VALID?` indicates that the data is valid.
The query retrieves the file image in binary format.

:FILE:SOR:VALid?

Function Queries whether the SOR file image is enabled.

Syntax `:FILE:SOR:VALID?`

Example `:FILE:SOR:VALID?`
`-> :FILE:SOR:VALID 1`

Description Measured data

0	:	Valid
1	:	Invalid

:FILE:SUBFolder:LIST?

Function Queries for a list of sub folders in the current folder.
Syntax :FILE:SUBFolder:LIST?
Example :FILE:SUBFOLDER:LIST?
-> :FILE:SUBFOLDER:LIST
"3,ABC/,DEF/,MACRO/"
Description The information is returned as follows: number of items, file (or folder), . . . , file (or folder). A slash is appended to folder names.

:FILE:TYPE

Function Sets or queries the file type when saving the file.
Syntax :FILE:TYPE {SET|SOR|CSV_WAVE|CSV_EVENT|BMP|PNG|JPG|CFG}
:FILE:TYPE?
SET : Setup file
SOR : SOR format
CSV_WAVE : CSV (waveform)
CSV_EVENT : CSV (event)
BMP : Bitmap format
JPG : JPEG format
CFG : CFG format
Example :FILE:TYPE SOR
:FILE:TYPE?
-> :FILE:TYPE SOR

LABel Group

The commands in this group can be used to set labels and query for settings.

:LABel:CABLe:CODE

Function Sets or queries the cable code.
Syntax :LABel:CABLe:CODE {<String>}
:LABel:CABLe:CODE?
<String> = Up to 36 characters
Example :LABEL:CABLE:CODE "123"
:LABEL:CABLE:CODE
->:LAB:CABLE:CODE "123"

:LABel:CABLe:ID

Function Sets or queries the cable ID.
Syntax :LABel:CABLe:ID {<String>}
:LABel:CABLe:ID?
<String> = Up to 36 characters
Example :LABEL:CABLE:ID "123"
:LABEL:CABLE:ID?
->:LABEL:CABLE:ID "123"

:LABel:COMPany

Function Sets or queries the company name.
Syntax :LABel:COMPany {<String>}
:LABel:COMPany?
<String> = Up to 36 characters
Example :LABEL:COMPANY "123"
:LABEL:COMPANY?
->:LABEL:COMPANY "123"

:LABel:DFLag:CURRent

Function Sets or queries the current data flag.
Syntax :LABel:DFLag:CURRent {BC|RC|OT|CC}
:LABel:DFLag:CURRent?
BC : as-Built Condition
RC : as-Repaired Condition
OT : Other
CC : Current Condition
Example :LABEL:DFLAG:CURRENT BC
:LABEL:DFLAG:CURRENT?
-> :LABEL:DFLAG:CURRENT BC

:LABel:FIBer:ID

Function Sets or queries the fiber ID.
Syntax :LABel:FIBer:ID {<String>}
:LABel:FIBer:ID?
<String> = Up to 36 characters
Example :LABEL:FIBER:ID "123"
:LABEL:FIBER:ID?
-> :LABEL:FIBER:ID "123"

:LABel:FIBer:TYPE

Function Sets or queries the fiber type.
Syntax :LABel:FIBer:TYPE {SMF|DSF|NZ_DSF|MMF}
:LABel:FIBer:TYPE?
SMF : Single-mode optical fiber
DSF : Dispersion-shifted, single-mode fiber
NZ_DSF : Non-zero, dispersion-shifted, single-mode fiber
MMF : Multi-mode optical fiber
Example :LABEL:FIBER:TYPE SMF
:LABEL:FIBER:TYPE?
-> :LABEL:FIBER:TYPE SMF

:LABel:LABel

Function Sets or queries the label.
Syntax :LABel:LABel {<String>}
:LABel:LABel?
<String> = Up to 36 characters
Example :LABEL:LABEL "123"
:LABEL:LABEL?
-> :LABEL:LABEL "123"

:LABel:LOCation:ORIGinating

Function Sets or queries the start position label.
Syntax :LABel:LOCation:ORIGinating {<String>}
:LABel:LOCation:ORIGinating?
<String> = Up to 36 characters
Example :LABEL:LOCATION:ORIGINATING "123"
:LABEL:LOCATION:ORIGINATING?
-> :LABEL:LOC:ORIGINATING "123"

:LABel:LOCation:TERMinating

Function Sets or queries the end position label.
Syntax :LABel:LOCation:TERMinating {<String>}
:LABel:LOCation:TERMinating?
<String> = Up to 36 characters
Example :LABEL:LOCATION:TERMINATING "123"
:LABEL:LOCATION:TERMINATING?
-> :LABEL:LOCATION:TERMINATING"123"

:LABel:OPERator

Function Sets or queries the operator name.
Syntax :LABel:OPERator {<String>}
:LABel:OPERator?
<String> = Up to 36 characters
Example :LABEL:OPERATOR "123"
:LABEL:OPERATOR?
-> :LABEL:OPERATOR "123"

MENU Group

The commands in this group can be used to set functions or marker mode and query for settings.

:MENU:ERRor:CLEar

Function Deletes the error dialog.
Syntax :MENU:ERRor:CLEar
Example :MENU:ERROR:CLEAR

:MENU:FUNCtion

Function Sets or queries the function mode.
Syntax :MENU:FUNCtion {TOP|OTDR}
TOP : Top menu
OTDR : OTDR
Example :MENU:FUNCTION TOP
:MENU:FUNCTION?
-> :MENU:FUNCTION TOP

:MENU:MARKer

Function Sets or queries the marker mode.
Syntax :MENU:MARKer {MARKER|LINE}
:MENU:MARKer?
MARKER : Marker
LINE : Line
Example :MENU:MARKER MARKER
:MENU:MARKER?
-> :MENU:MARKER MARKER

MISC Group

The commands in this group deal with dates, language, and power supply management.

:MISC:DATE:DAY

Function Sets or queries the day.
Syntax `:MISC:DATE:DAY {<NRf>}`
<NRf>:1 to 31 (1 Step)
Example `:MISC:DATE:DAY 1`
`:MISC:DATE:DAY?`
`-> :MISC:DATE:DAY 1`
Description You must execute `:MISC:DATE:SET` for this to take effect.

:MISC:DATE:GET?

Function Queries the date and time.
Syntax `:MISC:DATE:GET?`
Example `:MISC:DATE:GET?`
`-> :MISC:DATE:GET 2009/01/31 23:59:59`

:MISC:DATE:HOUR

Function Sets or queries the time.
Syntax `:MISC:DATE:HOUR {<NRf>}`
<NRf>:0 to 23 (1 Step)
Example `:MISC:DATE:HOUR 0`
`:MISC:DATE:HOUR?`
`-> :MISC:DATE:HOUR 0`
Description You must execute `:MISC:DATE:SET` for this to take effect.

:MISC:DATE:MINute

Function Sets or queries the minute.
Syntax `:MISC:DATE:MINute {<NRf>}`
<NRf>:0 to 59 (1 Step)
Example `:MISC:DATE:MINUTE 0`
`:MISC:DATE:MINUTE?`
`-> :MISC:DATE:MINUTE 0`
Description You must execute `:MISC:DATE:SET` for this to take effect.

:MISC:DATE:MODE

Function Sets or queries the date display type.
Syntax `:MISC:DATE:MODE {TYPE1|TYPE2|TYPE3}`
`:MISC:DATE:MODE?`
TYPE1 : 2009/08/29 12:16
TYPE2 : 08/29/2009 12:16
TYPE3 : 2009/AUG/29 12:16
Example `:MISC:DATE:MODE TYPE1`
`:MISC:DATE:MODE?`
`-> :MISC:DATE:MODE TYPE1`

:MISC:DATE:MONTh

Function Sets or queries the month.
Syntax `:MISC:DATE:MONTh {<NRf>}`
<NRf>: 1 to 12 (1 Step)
Example `:MISC:DATE:MONTH 1`
`:MISC:DATE:MONTH?`
`-> :MISC:DATE:MONTH 1`
Description You must execute `:MISC:DATE:SET` for this to take effect.

:MISC:DATE:SECond

Function Sets or queries the second.
Syntax `:MISC:DATE:SECond {<NRf>}`
<NRf>: 0 to 59 (1 Step)
Example `:MISC:DATE:SECOND 0`
`:MISC:DATE:SECOND?`
`-> :MISC:DATE:SECOND 0`
Description You must execute `:MISC:DATE:SET` for this to take effect.

:MISC:DATE:SET

Function Executes the changing of the date and time.
Syntax `:MISC:DATE:SET`
Example `:MISC:DATE:SET`

:MISC:DATE:YEAR

Function Sets or queries the year.
Syntax `:MISC:DATE:YEAR {<NRf>}`
<NRf>: 2009 to 2035 (1 Step)
Example `:MISC:DATE:YEAR 2009`
`:MISC:DATE:YEAR?`
`-> :MISC:DATE:YEAR 2009`
Description You must execute `:MISC:DATE:SET` for this to take effect.

:MISC:LANGuage

Function Sets or queries the language.

Syntax
```
:MISC:LANGuage {ENGLISH|FRENCH|
GERMANY|ITALIAN|SPANISH|TCHINESE|
DUTCH|FINNISH|NORWEGIAN|POLISH|
CZECH|SWEDISH|TURKEY|PORTUGUESE|
ROMANIAN|BRAZILIAN}
:MISC:LANGuage?
```

Example
```
:MISC:LANGUAGE ENGLISH
:MISC:LANGUAGE?
-> :MISC:LANGUAGE ENGLISH
```

Description Language settings

ENGLISH	:	English
FRENCH	:	French
GERMANY	:	German
ITALIAN	:	Italian
SPANISH	:	Spanish
TCHINESE	:	Traditional Chinese
DUTCH	:	Dutch
FINNISH	:	Finnish
NORWEGIAN	:	Norwegian
POLISH	:	Polish
CZECH	:	Czech
SWEDISH	:	Swedish
TURKEY	:	Turkish
PORTUGUESE	:	Portuguese
ROMANIAN	:	Romanian
BRAZILIAN	:	Brazilian (Portuguese)

:MISC:RLOSsmode

Function Sets or queries the return loss display.

Syntax
```
:MISC:RLOSsmode {NORMAL|NTT}
:MISC:MARKer?
```
NORMAL : Return loss
NTT : Return

Example
```
:MISC:RLOSSMODENORMAL
:MISC:RLOSSMODE ?
-> :MISC:RLOSSMODE NORMAL
```

NETWork Group

The commands in this group can be used to set and query network-related settings, such as IP address and user name.

:NETWork:CONTrol:PASSword

Function Sets or queries the password.

Syntax
```
:NETWork:CONTrol:PASSword <String>
```

Example
```
:NETWORK:CONTROLl:PASSWORD "ABC"
:NETWORK:CONTROLl:PASSWORD?
->:NETWORK:CONTROLl:PASSWORD "ABC"
```

Description You cannot set the password if `:NETWork:STATe` is OFF.
You do not need to set the password when the user name is set to “anonymous.”
After setting the password, you must execute `:SYSTEM:REBOOT`.

:NETWork:CONTrol:TIMeout

Function Sets or queries the timeout value.

Syntax
```
:NETWork:CONTrol:TIMeout
{<NRf>|INFinite}
:NETWork:CONTrol:TIMeout?
```
<NRf>:1 to 7200 (s)

Example
```
:NETWORK:CONTROL:TIMEOUT 30
->:NETWORK:CONTROL:TIMEOUT 30
```

Description The unit is s (seconds). You cannot set the password if `:NETWork:STATe` is OFF.
The setting takes effect on the next connection.

:NETWork:CONTrol:USERname

Function Sets or queries the user name.

Syntax
```
:NETWork:CONTrol:USERname <String>
```

Example
```
:NETWORK:CONTROL:USERNAME "anonymous"
:NETWORK:CONTROL:USERNAME?
-> :NETWORK:CONTROL:USERNAME
"anonyumous"
```

Description You cannot set the password if `:NETWork:STATe` is OFF.
After setting the password, you must execute `:SYSTEM:REBOOT`.

:NETWork:DHCP

Function Turns DHCP on and off.

Syntax
```
:NETWork:DHCP <Boolean>
:NETWork:DHCP?
```

Example
```
:NETWORK:DHCP ON
:NETWORK:DHCP?
->:NETWORK:DHCP 1
```

Disabled: OFF or 0
Enabled : ON or 1

Description You cannot set the password if `:NETWork:STATe` is OFF.
After setting the password, you must execute `:SYSTEM:REBOOT`.

:NETWork:GATeway

Function Sets or queries the gateway.
Syntax :NETWork:GATeway <String>
:NETWork:GATeway?
Example :NETWORK:GATEWAY "255.255.255.0"
:NETWORK:GATEWAY?
->:NETWORK:GATEWAY "255.255.255.0"
Description You cannot set the password if :NETWork:STATe is OFF.
You cannot set the gateway if :NETWork:DHCP is OFF.
After setting the password, you must execute :SYSTEM:REBOOT.

:NETWork:IPADdress

Function Sets or queries the IP address.
Syntax :NETWork:IPADdress <String>
:NETWork:IPADdress?
Example :NETWORK:IPADDRESS "192.168.0.1"
:NETWORK:IPADDRESS?
->:NETWORK:IPADDRESS "192.168.0.1"
Description You cannot set the password if :NETWork:STATe is OFF.
You cannot set the gateway if :NETWork:DHCP is OFF.
After setting the password, you must execute :SYSTEM:REBOOT.

:NETWork:NETMask

Function Sets or queries the net mask.
Syntax :NETWork:NETMask <String>
:NETWork:NETMask?
Example :NETWORK:NETMASK "255.255.255.0"
:NETWORK:NETMASK?
->:NETWORK:NETMASK "255.255.255.0"
Description You cannot set the password if :NETWork:STATe is OFF.
You cannot set the gateway if :NETWork:DHCP is OFF.
After setting the password, you must execute :SYSTEM:REBOOT.

:NETWork:STATe

Function Turns the network on or off.
Syntax :NETWork:STATe <Boolean>
:NETWork:STATe?
Example :NETWORK:STATE ON
:NETWORK:STATE?
->:NETWORK:STATE 1
OFF or 0 : Invalid
ON or 1 : Valid
Description After setting the password, you must execute :SYSTEM:REBOOT.

SETup Group

The commands in this group deal with initializing the settings. The settings can be initialized to factory default values.

:SETup:ALLinit

Function Initializes all settings to their factory default values and formats the internal memory.
Syntax :SETup:ALLInit
Example :SETup:ALLINIT

:SETup:INITialize

Function Initializes all settings to their factory default values.
Syntax :SETup:INITialize
Example :SETUP:INITIALIZE

STATus Group

The commands in this group are used to make settings and queries related to the communication status feature.
There are no front panel keys that correspond to the commands in this group.

:STATus:CONDition?

Function Queries the contents of the condition register.
Syntax `:STATus:CONDition?`
Example `:STATUS:CONDITION? -> 16`
Description For information about the condition register, see section 4.4, “Condition Register.”

:STATus:ERRor?

Function Queries the error code and message of the last error that has occurred (top of the error queue).
Syntax `:STATus:ERRor?`
Example `:STATUS:ERROR? ->113,"Undefined header"`

:STATus:QENable

Function Sets or queries whether messages other than errors will be stored to the error queue.
Syntax `:STATus:QENable {<Boolean>}:STATus:QENable?`
Example `:STATUS:QENABLE ON`
`:STATUS:QENABLE? -> :STAT:QEN 1`

:STATus:QMESsage

Function Sets or queries whether message information will be attached to the response to the STATus:ERRor? query.
Syntax `:STATus:QMESsage {<Boolean>}`
`:STATus:QMESsage?`
Example `:STATUS:QMESSAGE OFF`
`:STATUS:QMESSAGE? -> :STAT:QMES 1`

SYSTem Group

The commands in this group deal with system control.

:SYSTem:REBoot

Function Restarts the system.
Syntax `:SYSTem:REBoot`
Example `:SYSTEM:REBOOT`
Description
- The communication will be disconnected, so execute disconnection (TmcFinish).
- After setting network parameters, execute this command.

WAVedata Group

The commands in this group deal with the waveform data.

:WAVedata:DISPlay:SEND:ASCii?

Function Queries the displayed waveform data in ASCII format.

Syntax `WAVedata:DISPlay:SEND:ASCii?`

Example `WAVEDATA:DISPLAY:SEND:ASCII?`
`-> 7.526, •••`

Description Requests for waveform data shown on the screen in ASCII format.
Return format : XXXXX, XXXXX, . . .
Return value Example : 45000 for 45.000 dB

:WAVedata:DISPlay:SEND:BINary?

Function Queries the displayed waveform data in binary format.

Syntax `WAVedata:DISPlay:SEND:BINary?`

Example `WAVEDATA:DISPLAY:SEND:BINARY?`
`-> #6123456ABCDEFGHJ1234567890`

Description Requests for waveform data shown on the screen in binary format.
The return value can be obtained as an integer with the value in 0.001 dB multiplied 1000 times. One data value is 4 bytes. The upper byte is output first and then the lower byte. (in units of 0.001dB). For example, 1000 represents 1.000 dB.

:WAVedata:LENGth?

Function Queries the number of waveform data points.

Syntax `WAVedata:LENGth?`

Example `WAVEDATA:LENGTH?`
`-> :WAVE:LENGTH 25000`

:WAVedata:OLDType:DISPlay:SEND?

Function Queries the displayed waveform data in Dot4 format.

Syntax `WAVedata:OLDType:DISPlay:SEND?`

Example `WAVEDATA:OLDTYPE:DISPLAY:SEND?`
`-> #42000•••`

Description Requests for waveform data shown on the screen in binary format.
The return value consists of the integer part of data×250. The upper and lower bytes are output alternately (in units of 0.001 dB).

:WAVedata:OLDType:SEND?

Function Queries the waveform data in Dot4 format.

Syntax `WAVedata:OLDType:SEND?`

Example `WAVEDATA:OLDTYPE:SEND?`
`-> #6123456ABCDEFGHJ1234567890`

Description Requests for the entire waveform data in binary format.
The return value consists of the integer part of data×250.
The upper and lower bytes are output alternately (in units of 0.001 dB).

:WAVedata:SEND:ASCii?

Function Queries the waveform data in ASCII format.

Syntax `WAVedata:SEND:ASCii?`

Example `WAVEDATA:SEND:ASCII?`
`-> 7.526, •••`

Description Requests for the entire waveform data in ASCII format.
Return format : XX.XXX , XX.XXX , •••

:WAVedata:SEND:BINary?

Function Queries the waveform data in binary format.

Syntax `WAVedata:SEND:BINary?`

Example `WAVEDATA:SEND:BINARY?`
`-> #6123456ABCDEFGHJ1234567890`

Description Requests for the entire waveform data in binary format.
The return value can be obtained as an integer with the value in 0.001 dB multiplied 1000 times. One data value is 4 bytes. The upper byte is output first and then the lower byte. (in units of 0.001dB). For example, 1000 represents 1.000 dB.

:WAVedata:SEND:SIZE

Function Sets or queries the number of retrieved waveform data points.

Syntax `WAVedata:SEND:SIZE {<NRf>}`
`:WAVedata:SEND:SIZE?`
<NRf> = 1 to number of measurement points

Example `WAVEDATA:SEND:SIZE {<NRF>}`
`:WAVEDATA:SEND:SIZE?`
`-> :WAVEDATA:SEND:SIZE 20000`
<NRF> = 1 to number of measurement points

Description Values that exceed the selectable range are rounded.

:WAVedata:SEND:STARt

Function Sets or queries the start distance of the waveform data.

Syntax
```
WAVedata:SEND:STARt {<NRf>}
:WAVedata:SEND:STARt?
```
<NRf> = 0 to measurement distance [m]

Example
```
WAVEDATA:SEND:START 20000
:WAVEDATA:SEND:START?
-> :WAVEDATA:SEND:START 20000
```

Description Values that exceed the selectable range are rounded.

4.4 Condition Register

The condition register indicates the internal condition of the instrument.

Condition Register
`:STATus:CONDition?`

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	PRT	LS	PME	PMZ	PMM	0	AR	PC	FIA	AS		AVE	REAL

The condition register bits are described below.

Bit 0	REAL	Set to 1 during real-time measurement.
Bit 1	AVE	Set to 1 during averaged measurement.
Bit 2	FILE	Set to 1 during file access.
Bit 3	AS	Set to 1 during auto search.
Bit 4	FIA	Set to 1 during fiber in use check.
Bit 5	PC	Set to 1 during plug check.
Bit 6	AR	Set to 1 during auto range measurement.
Bit 8	PMM	Set to 1 during power meter measurement.
Bit 9	PMZ	Set to 1 during power meter zero-set.
Bit 10	PME	Set to 1 during power meter measurement error.
Bit 11	LS	Set to 1 while the light source is emitting.
Bit 12	PRT	Set to 1 during printing.

4.5 Output Queue and Error Queue

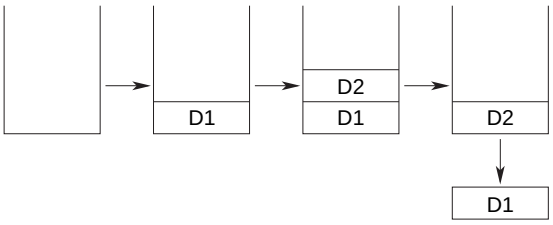
Overview of the Output Queue

The output queue is provided to store responsemessages to queries. For example, when the :WAVeform:SEND? query is sent to request output of the acquired waveform, the response data will be stored in the output queue until it is read out.The example below shows that data is stored recordby record in the output queue, and is read out oldestitem first, newest item last.

The output queue is emptied in the following cases (in addition to whenread-out is performed).

- When a new message is received from thecontroller
- When dead lock occurs (page 4-2)
- When a device clear command (DCL or SDC) isreceived
- When power is turned ON again

The output queue cannot be emptied using the *CLS command. To see whether the output queue is emptyor not, check bit 4 (MAV) of the status byte.



Overview of the Error Queue

The error queue stores the error No. and message when an error occurs. For example, if the controller sends an incorrect program message, the number,“113, “Undefined header”,” and the error message are stored in the error queue, when the error is displayed.The contents of the error queue can be read using the:STATus:ERRor? query. As with the output queue,messages are read oldest first, newest last (refer to theprevious page).If the error queue becomes full, the final message willbe replaced by message “350, “Queue overflow”.”

The error queue is emptied in the following cases (inaddition to when read-out is performed).

- When the *CLS command is received
- When power is turned ON again

To see whether the error queue is empty or not, checkbit 2 (EAV) of the status byte.

4.6 Error Messages

Error in Execution

Code	Message
50	*OPC/? exists in message.
102	Syntax error.
103	Invalid separator.
104	Data type error.
108	Parameter not allowed.
109	Missing parameter.
111	Header separator error.
112	Program mnemonic too long.
113	Undefined header.
114	Header suffix out of range.
120	Numeric data error.
123	Exponent too large.
124	Too many digits.
128	Numeric data not allowed.
131	Invalid suffix.
134	Suffix too long.
138	Suffix not allowed.
141	Invalid character data.
144	Character data too long.
148	Character data not allowed.
150	String data error.
151	Invalid string data.
158	String data not allowed.
161	Invalid block data.
168	Block data not allowed.
171	Invalid expression.
178	Expression data not allowed.
181	Invalid outside macro definition.
210	Communication not allowed during trace analysis
211	Communication not allowed during measurement.
221	Setting conflict.
222	Data out of range.
223	Data invalid
224	Illegal parameter value.
241	Hardware missing.
260	Expression error.
270	Macro error.
272	Macro execution error.
273	Improper macro label.
275	Macro definition too long.
276	Macro recursion error.
277	Macro redefinition not allowed.
278	Macro header not found.
350	Queue overflow.
410	Query INTERRUPTED.
420	Query UNTERMINATED.
430	Query DEADLOCKED.
440	Query UNTERMINATED after indefinite response.

Code	Message
500	The measurement condition is failed. Please reset ****** in SETTING menu.
501	Not executable during measurement. Please stop the measurement and execute again.
502	The measurement cannot be started during printing. Please either wait until the end of the printing or interrupt the printing.
503	The measurement cannot be started during the file operation. Please either interrupt the file operation or wait until the end of the file operation.
504	The measurement could not be completed within the specified averaging duration. Please modify the averaging duration.
505	Hardware failed, and needs to be repaired. Please contact Yokogawa's representatives.
506	Hardware failed, and needs to be repaired. Please contact Yokogawa's representatives.
507	Hardware failed, and needs to be repaired. Please contact Yokogawa's representatives.
508	Hardware failed, and needs to be repaired. Please contact Yokogawa's representatives.
509	Hardware failed, and needs to be repaired. Please contact Yokogawa's representatives.
560	When the event fix is ON, the distance reference setup cannot be performed.
800	This function is not supported.
814	Failed network initialize. Please confirm network connection and setting.
816	Network setting is updated. Rebooting is required to reflect the changes.
817	Test Error occurred.
818	Test succeeded.
900	Backup data is damaged. this instrument starts up with the factory setting.
901	Optical connector is not correctly connected. Please clean and reconnect the connector again.
909	The temperature inside the instrument is too high. The instrument may be damaged if it is kept using in this condition. The instrument will be powered off in 10 sec. Please do not power it on till the battery is cooled down.
910	The temperature inside the instrument is too low. The instrument may be damaged if it is kept using in this condition. The instrument will be powered off in 10 sec. Please do not power it on till the battery is cooled down.
911	The Voltage of AC power supply is too low. The instrument may be damaged if it is kept using in this condition. The instrument will be powered off in 10 sec. Please make sure of using the dedicated AC adapter.
912	The Voltage of AC power supply is too high. The instrument may be damaged if it is kept using in this condition. The instrument will be powered off in 10 sec. Please make sure of using the dedicated AC adapter.
917	The temperature inside the instrument is increasing. Please stop measuring, and wait for a while.
918	The temperature inside the instrument is too high. The measurement was aborted. Please power it off.
921	There is incoming light. Or, optical module may be damaged.
922	Incorrect date and time setting. Set the correct date and time.
923	The ambient temperature is out of range. The light source cannot be turned on.
925	Please use AC adapter.

Index