MP2100A BERTWave /
MP2101A BERTWave PE/
MP2102A BERTWave SS
Remote Control
Operation Manual
(SCPI)

Ninth Edition

- **For safety and warning information, please read this manual before attempting to use the equipment.**
- **Additional safety and warning information is provided within the MP2100A BERTWave Operation Manual. Please also refer to this document before using the equipment.**
- **Keep this manual with the equipment.**

ANRITSU CORPORATION

Document No.: M-W3350AE-9.0

Safety Symbols

To prevent the risk of personal injury or loss related to equipment malfunction, Anritsu Corporation uses the following safety symbols to indicate safety-related information. Ensure that you clearly understand the meanings of the symbols BEFORE using the equipment. Some or all of the following symbols may be used on all Anritsu equipment. In addition, there may be other labels attached to products that are not shown in the diagrams in this manual.

Symbols used in manual

⚠ DANGER This indicates a very dangerous procedure that could result in serious injury or death if not performed properly.

⚠ WARNING This indicates a hazardous procedure that could result in serious injury or death if not performed properly.

⚠ CAUTION This indicates a hazardous procedure or danger that could result in light-to-severe injury, or loss related to equipment malfunction, if proper precautions are not taken.

Safety Symbols Used on Equipment and in Manual

The following safety symbols are used inside or on the equipment near operation locations to provide information about safety items and operation precautions. Ensure that you clearly understand the meanings of the symbols and take the necessary precautions BEFORE using the equipment.

This indicates a prohibited operation. The prohibited operation is indicated symbolically in or near the barred circle.

This indicates an obligatory safety precaution. The obligatory operation is indicated symbolically in or near the circle.

This indicates a warning or caution. The contents are indicated symbolically in or near the triangle.

This indicates a note. The contents are described in the box.

These indicate that the marked part should be recycled.

MP2100A/MP2101A/MP2102A
BERTWave (SCPI)
Remote Control Operation Manual

15 February 2010 (First Edition)
28 January 2013 (Ninth Edition)

Notes On Export Management

This product and its manuals may require an Export License/Approval by the Government of the product's country of origin for re-export from your country.

Before re-exporting the product or manuals, please contact us to confirm whether they are export-controlled items or not.

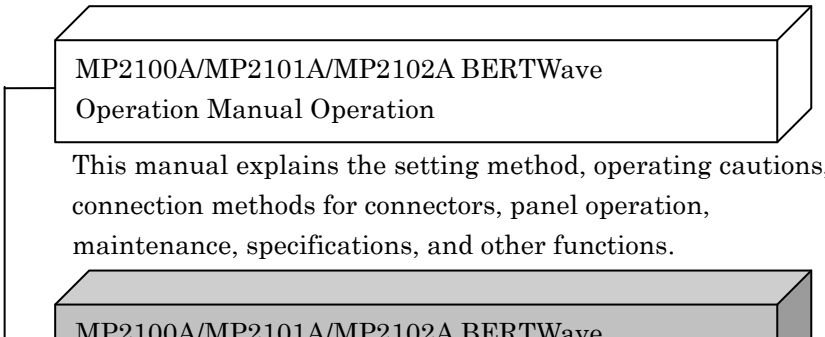
When you dispose of export-controlled items, the products/manuals need to be broken/shredded so as not to be unlawfully used for military purpose.

Trademark and Registered Trademark

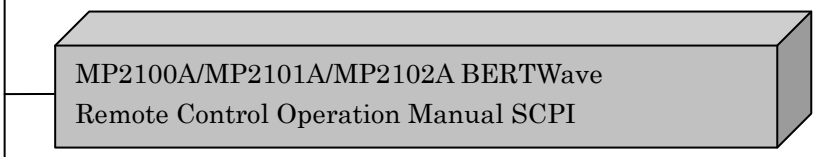
Windows® is a registered trademark of Microsoft Corporation in the United States and/or other countries.

About This Manual

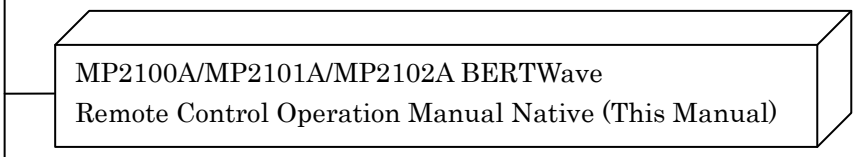
The manuals for the MP2100A/MP2101A/MP2102A BERTWave are configured in three parts.



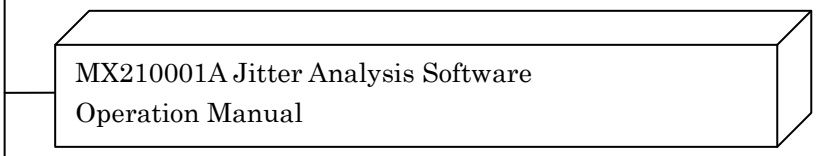
This manual explains the setting method, operating cautions, connection methods for connectors, panel operation, maintenance, specifications, and other functions.



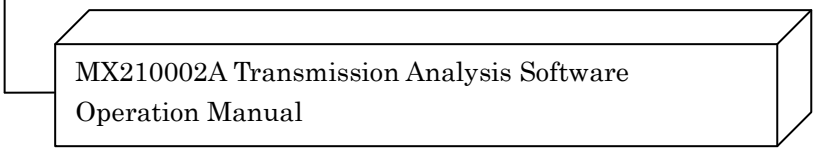
This manual explains the commands to control the MP2100A/MP2101A/MP2102A, status register configuration, and sample programs.



This manual explains the commands which allow the use of remote interface software for previous Anritsu pulse pattern generator and error detector models with the MP2100A/MP2101A/MP2102A BERTWave.



This manual explains the operation method and remote control commands for the MX210001A Jitter Analysis Software.



This manual explains the operation method and remote control commands for the MX210002A Transmission Analysis Software.

This manual explains the remote control commands. The remote control commands explained in this manual are conformed to the specifications of the SCPI（Standard Commands for Programmable Interfaces）.
This operation manual assumes the reader has the following information:

- The reder has read through the MP2100A/MP2101A/MP2102A BERTWave Operation Manual Operation.
- The reder can create the C or Basic program.

For the connection of the power source and peripheral devices, panel operation, and maintenance, refer to the following manual:
MP2100A/MP2101A/MP2102A BERTWave Operation Manual Operation（M-W3349AE）

Table of Contents

Table of Command

Chapter 1 Outline

This chapter explains the outline of the remote control, main uses, and glossary.

1.1 About Remote Control

The remote control function sends commands via the communications interface from the remote control PC to set the measuring instrument and read the measurement results and measuring instrument conditions.

The MP2100A BERTWave, MP2101A BERTWave PE, and MP2102A BERTWave SS (MP2100A/MP2101A/MP2102A, hereafter) supports the Ethernet interface. (When Opt-030/130 is installed, the GPIB interface can be used.)
When using either interface, set the number to distinguish the MP2100A/MP2101A/MP2102A from other equipment. When using the Ethernet interface, the IP address is set, and when using the GPIB, the GPIB address is set.

The character strings for controlling the MP2100A/MP2101A/MP2102A are called command. The command is composed by the ASCII character strings. For example, the following command sets when the signal of the pulse pattern generator (PPG) is output to the connector.

```
:OUTPUT:DATA:OUTPUT ON
```

A command for reading data from this instrument is called a query message. A query command has the question symbol (?) appended to the string. For example, sending the following command queries the PPG bit rate set at the instrument.

```
:OUTPUT:CMU:FREQ?
```

The controller PC receives the following response against the query message from the instrument.

```
1250000
```

The bit rate is 1250000 kbit/s.

When the MP2100A/MP2101A/MP2102A is measured via remote control, the Remote lamp on the screen is lit. Only the power switch and the key [Local/Panel Unlock] on the system menu are valid in this situation. This situation is called panel lock. To unlock the panel, touch [Local/Panel Unlock] on the system menu.

1.2 Main Uses for Remote Control

The main uses for remote control are listed below.

Automating Measurements

Instead of touch-panel or rotary knob operations, measurement can be automated by controlling the instrument by executing programs. Writing the measurement control procedures using the program makes the measurement automatically.

Remote Control of Instruments

Measuring instruments at remote locations can be controlled over communications lines to collect measurement data.

Controlling Multiple Instruments

The characteristics of DUTs can be measured simultaneously by remote control of multiple instruments.

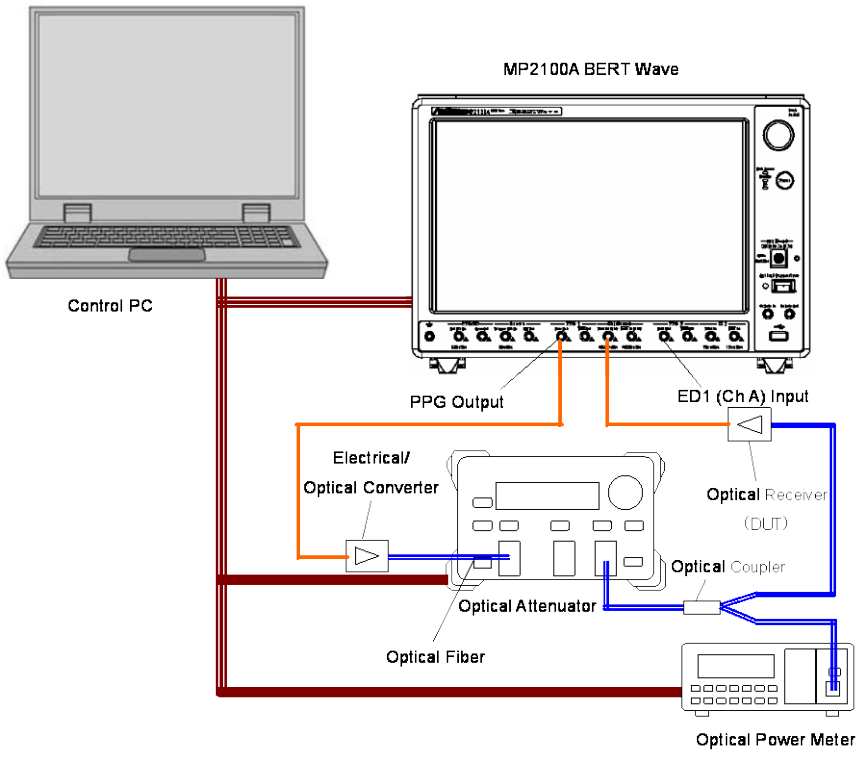


Figure 1.2-1 Example of Controlling Multiple Instruments

Figure 1.2-1 shows an example of controlling multiple instruments. In this example, the bit error rates are measured with changes in the optical input level of the optical receiver. The attenuation of the optical attenuator is controlled remotely from the PC and the optical level is read by the optical power meter. Table 1.2-1 shows the measurement result.

Table 1.2-1 Bit Error Rate of Optical Receiver

Optical Power (dBm)	Bit Error Rate
–25.034	0.011442
–24.523	0.0048758
–24.031	0.001631
–23.536	0.00044241
–23.030	0.000078419
–22.523	0.0000088616
–22.031	0.000000616
–21.524	0.000000016
–21.037	0.00000000028235

1.3 Technical Terms

Table 1.3-1 indicates what abbreviations are used in this operation manual.

Table 1.3-1 Abbreviation

Abbreviation	Formal name
ASCII	American Standard Code for Information Interchange
CR	Carriage Return
EOI	End or Identity
ESE	Event Status Enable Register
ESR	Event Status Register
GPIB	General Purpose Interface Bus
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
LAN	Local Area Network
LF	Line Feed
MAV	Message Available summery
MSS	Master Summery Status
OSER	Operation Status Enable Register
OSR	Operation Status Register
PC	Personal Computer
SCPI	Standard Commands for Programmable Interfaces
SRER	Service Request Enable Register
SRQ	Service Request
STB	Status Byte Register
TR	Transition Filter
VISA	Virtual Instrument Software Architecture

Chapter 2 Before Use

This chapter explains the preparations for using remote control.

2.1 Preparing Equipment

The following equipment/parts are required to perform remote control.

- PC
- Ethernet interface
- Ethernet cable
- GPIB interface (when Opt-030/130 installed)
- GPIB cable (when Opt-030/130 installed)
- Program development tools

Ethernet Interface

Prepare an interface meeting the following specifications:

10BASE-T

100BASE-TX

Furthermore, use a cable matching each specification.

GPIB Interface

Use GPIB interfaces that conform to IEEE 488.2.
When using the sample program described in Chapter 3, VISA is required.
This instrument has only been tested with the National Instruments' VISA product, which we recommend.

Program Development Tools

Prepare some tools for developing and running programs for performing remote control. Refer to the VISA and Interface manuals for the specifications required by the program development tools.

PC

Procure the PC conforming to the operation Environment for the GPIB interface, VISA, and program development tools.

2.2 Connecting Equipment

2.2.1 Connecting Ethernet

Connect the Ethernet connector on the side-panel of the MP2100A/MP2101A/MP2102A and external devices using LAN cables.

Use a LAN crossover cable to connect the MP2100A/MP2101A/MP2102A and a control PC directly. Use a LAN straight cable via a network hub when connecting to multiple external devices.

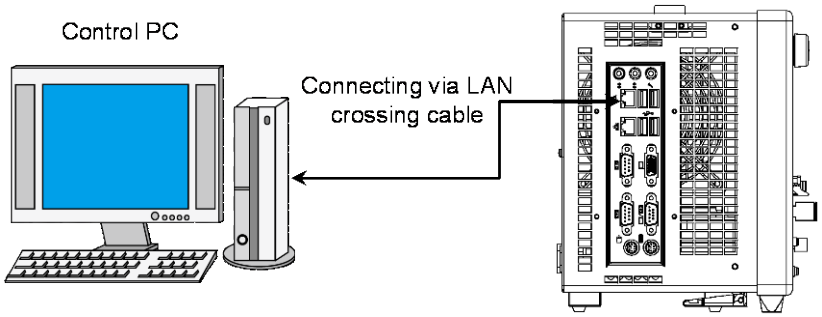


Figure 2.2.1-1 Direct Connection between MP2100A/MP2101A/MP2102A and Control PC

Note:

When connecting the MP2100A/MP2101A/MP2102A via LAN, confirm the network settings before measurement.

- The IP address of the MP2100A/MP2101A/MP2102A and that of the other devices are not overlapped.
- The IP address of the control PC is included in the address range set ar the subnet mask.

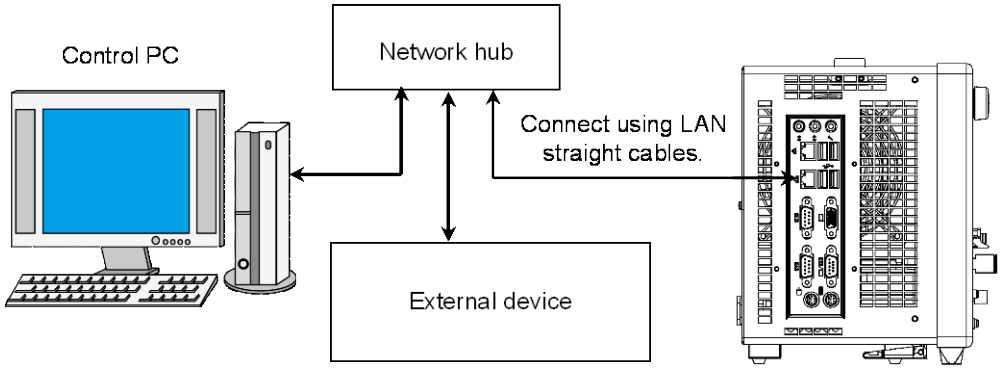


Figure 2.2.1-2 Sample Connection with Multiple External Devices

Note :

The control PC may have difficulty in communicating with the MP2100A/MP2101A/MP2102A, depending on the status of communications between them. The direct connection is recommended to ensure communication stability.

2.2.2 Connecting GPIB

Connect the GPIB connector on the rear panel of the MP2100A/MP2101A/MP2102A and an external device using a GPIB cable.

CAUTION

Always connect the GPIB cable BEFORE turning on the power to the MP2100A/MP2101A/MP2102A. Connecting it while the power is on may damage internal circuits.

Up to 15 devices, including the external PC controller, can be connected to one MP2100A/MP2101A/MP2102A unit. Always follow the conditions shown below when connecting devices.

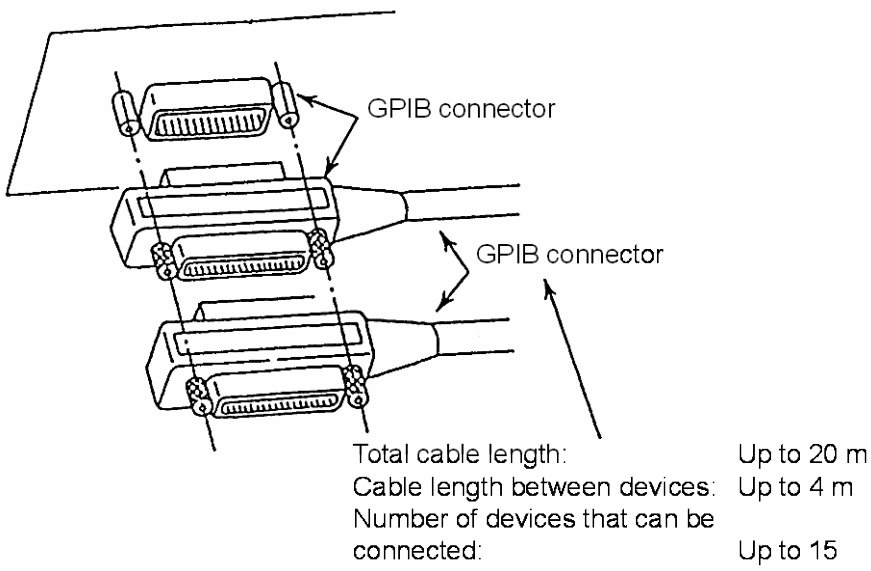


Figure 2.2.2-1 GPIB Cable Connection 1

Connect cables without forming loops.

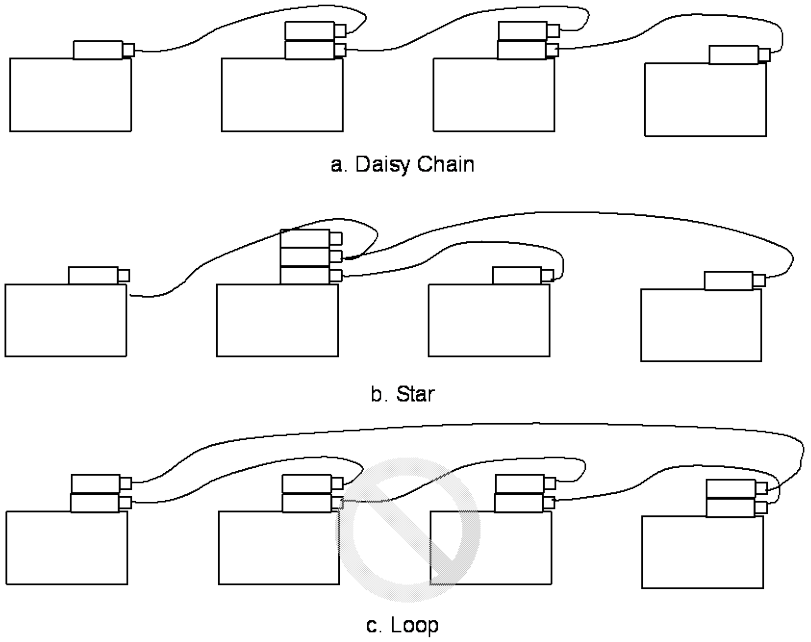


Figure 2.2.2-2 GPIB Cable Connection 2

2.3 Setting Interface

2.3.1 Setting Interface

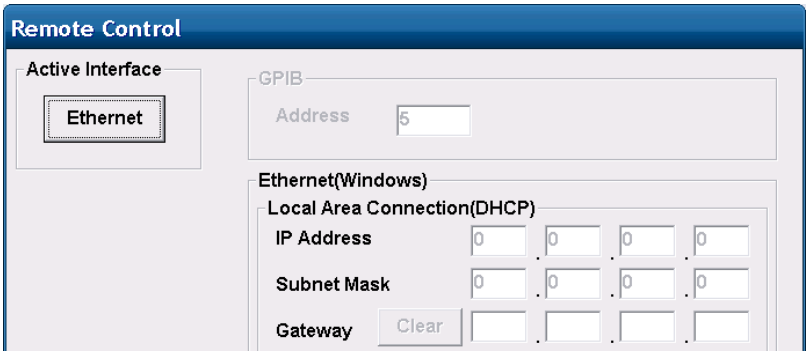
Set the remote control interface to the Ethernet using the following method, and enter the IP address.

1. Switch on the power to the MP2100A/MP2101A/ MP2102A.
2. Touch [Setup Utility] at the Selector screen.
3. Touch [Remote Control].
4. Touch the Active Interface button to set the button display to [Ethernet].
   When Opt-030/130 is not installed, the Active Interface button is disabled.
5. Set the IP address, subnet mask, gateway and port number.
   The gateway address can be omitted.
   To display the numeric entry panel, touch the text box.
   You cannot enter the numeric value in the text box directly, using the attached key board.
   The IP address for Local Area Connection is the IP address for the connector on the upper left side.
   The IP address for Local Area Connection 2 is the IP address for the connector on the lower left side.
   The port number can be set from 1024 to 5001.
6. Touch [Apply], and then the settings are completed.
   Touch [Exit], and then the set value is deleted.

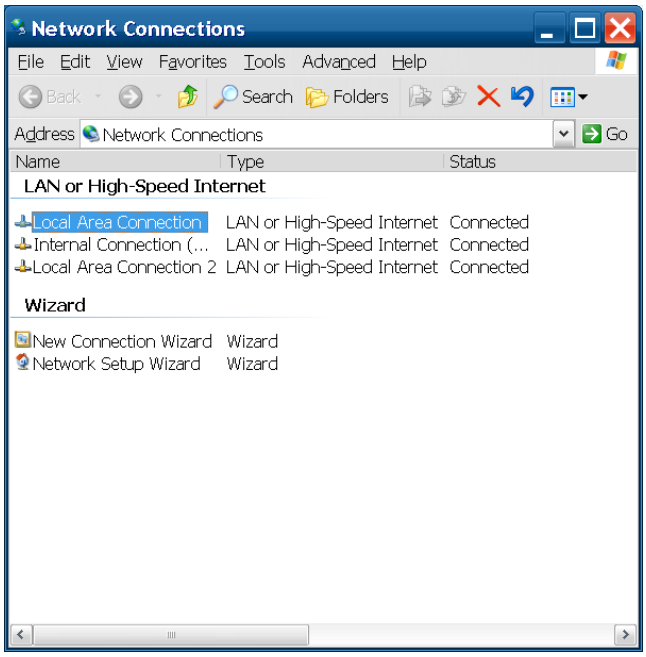
Note:

Do not set the following IP address.
192.168.1.0 ~192.168.1.255

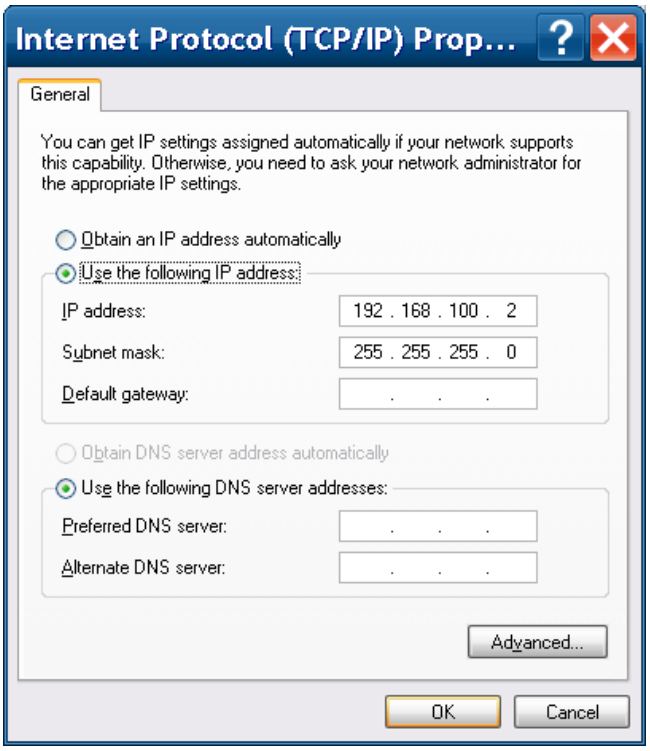
If the message “Local Area Connection (DHCP)”appears in Step 5, cancel the automatic acquisition of the network address using the following procedures.



1. Connect the keyboard and mouse to the MP2100A/MP2101A/MP2102A.
2. Press the Windows key on the connected keyboard.
   If there is no Windows key on the keyboard, display the desktop and click [Start].
   For how to display the desktop, refer to section 2.10 Control Panel Settings in the MP2100A/MP2101A/MP2102A BERTWave Operation Manual Operation (W3349AE).
3. Click [Control Panel].
4. Double-click [Network Connections].



5. Right-click [Local Area Connection] or [Local Area Connection2], and click [Property].
6. The Local Area Connection Properties window opens.
   Click [Internet Protocol (TCP/IP)] in the list box, and press the [Property] button.



7. Check [Use the following IP Address].
8. Press the [OK] button.
9. Press the [OK] button in [Local Area Connection Properties].

2.3.2 Setting GPIB

Set the remote control interface to the GPIB using the following method, and enter the GPIB address.

1. Switch on the power to the MP2100A/MP2101A/ MP2102A.
2. Touch [Setup Utility] at the Selector screen.
3. Touch [Remote Control].
4. Touch the Active Interface button to set the button display to [GPIB].

   If neither Opt-030/130 is installed, the Active Interface button is invalid.
5. Touch the GPIB address on the test box.
6. Set the GPIB address using the numeric value input panel.
   You cannot enter the numeric value in the text box directly using the attached key board.
7. Put a checkmark in [Remote High Speed Response] to shorten the time from when the remote command is sent to the MP2100A/MP2101A/ MP2102A until the measurement result is received.
   However, the bit error measurement results and measurement progress display are not updated during remote control.
8. Touch [Apply] to complete the setting.
   Touch [Exit], and then the values set at step 4 to 7 are deleted.

2.4 Checking Connection

Check that the link between the PC and MP2100A/MP2101A/MP2102A has been established via the Ethernet.

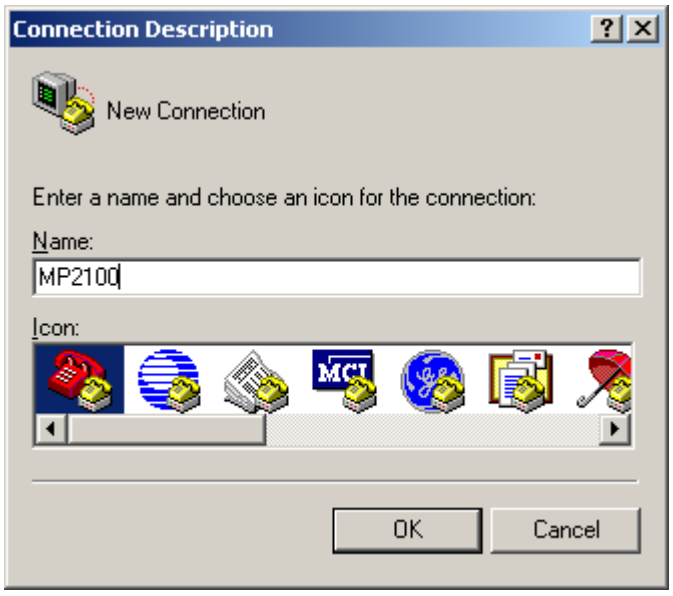
Caution

Remote control is not supported after a system alarm.

Switch off the power and resolve the cause of the alarm.

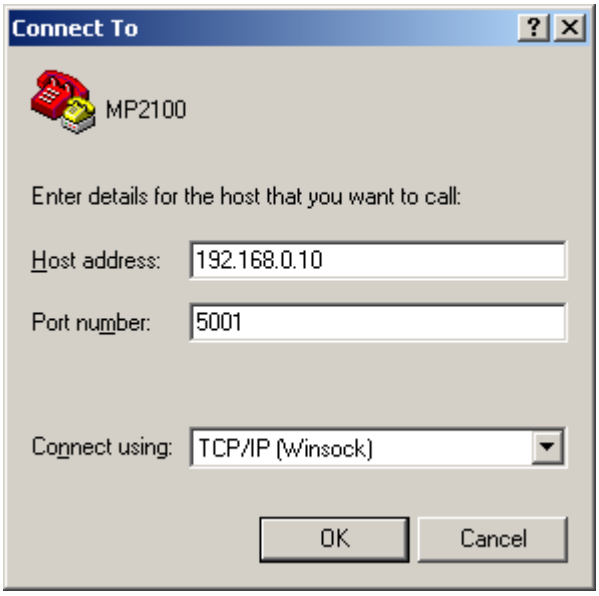
2.4.1 When using Ethernet (Windows XP)

1. Click Programs at the Windows Start menu.
2. Click [Accessories].
3. Click [HyperTerminal] from the Communication submenu.
4. When the following screen opens, enter the name and click [OK].

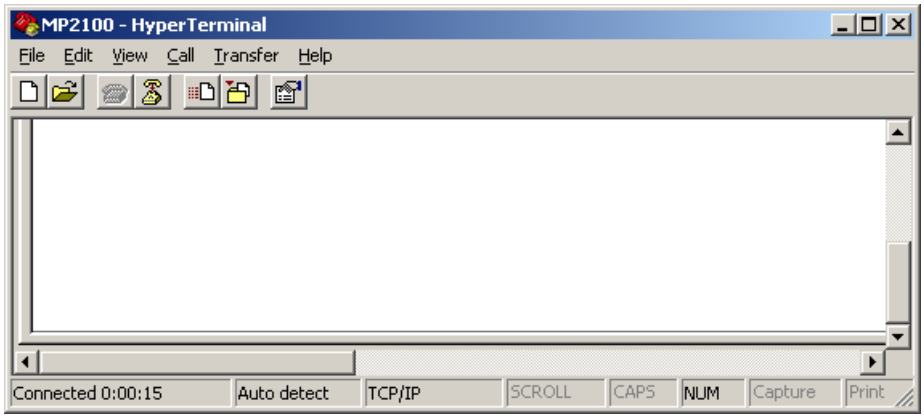


5. Set the connection method to TCP/IP.

6. Enter the host address and port number.
The settings for the MP2100A/ MP2101A/MP2102A set at Section 2.3.1 Setting Ethernet are as follows: the IP address is 192.168.0.10, and the port number is 5001.



7. When the message “192.168.0.10 cannot be connected to the port 5001.” is displayed, the PC cannot recognize the MP2100A/ MP2101A/MP2102A. Check that:
   - The MP2100A/ MP2101A/MP2102A IP address and port numbers are correct
   - The Ethernet cable type (straight/crossing) is correct.
   - The Ethernet cable is connected to the correct Ethernet connector on the left side of the MP2100A/ MP2101A/MP2102A.
   - The cable and connectors are not damaged.
8. When the display changes to “Connected”, the PC controller and the MP2100A/ MP2101A/MP2102A can be connected via the Ethernet.

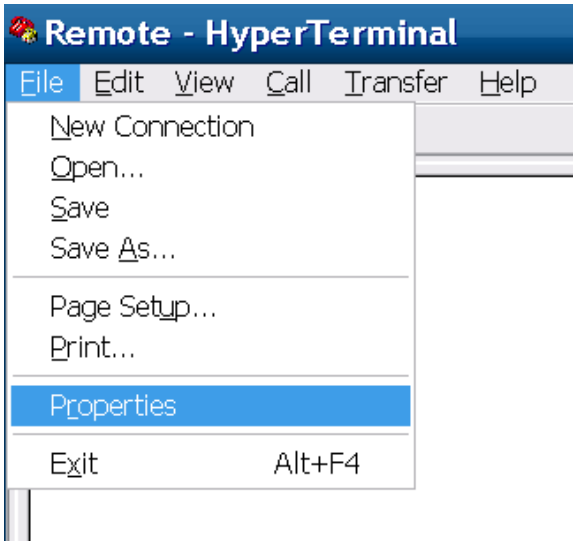


Note:

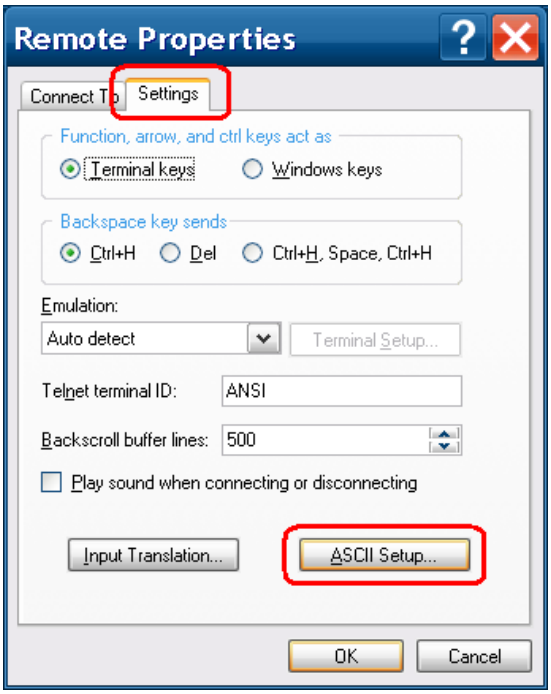
To remote-control the MP2100A/ MP2101A/MP2102A using HyperTerminal, change the terminator to CR+LF using another method.

To change the terminator, use the :SYEStem:TERMination command.

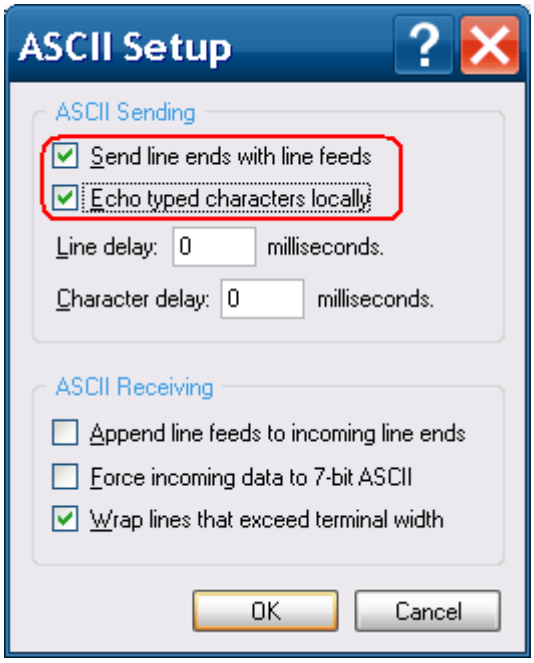
9. Change the HyperTerminal setting to control this equipment using remote commands by selecting [File] – [Properties].



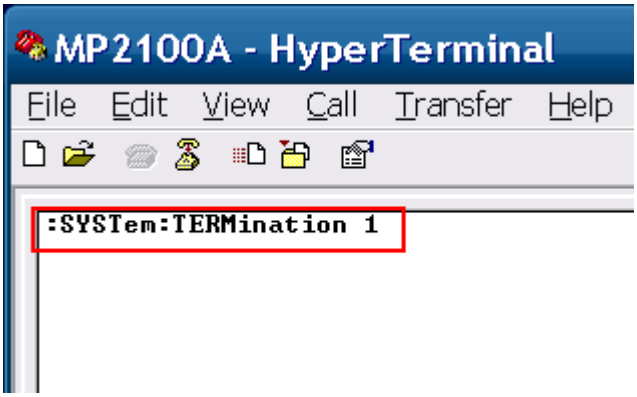
10. Change the HyperTerminal setting to control this equipment using remote commands by clicking [Settings] and clicking the [ASCII Setup…] button.



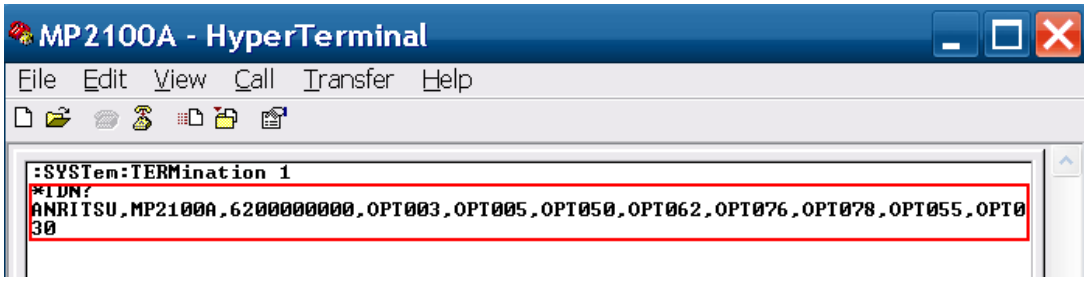
11. Put checkmarks in [Send line ends with line feeds] and [Echo typed characters locally] and click the [OK] button.



12. To display the response from the equipment correctly with HyperTerminal, responses from the equipment must be terminated with CR+LF. Input :SYSTem:TERMination 1 and press the Enter key. When the command is received by this equipment, the operation screen becomes locked (Remote status).



13. To confirm whether the response from this equipment can be received, input *IDN? and press the Enter key. When the response from the equipment is received, the character string ANRITSU,MP2100A,6200000000,OPT003 is displayed.



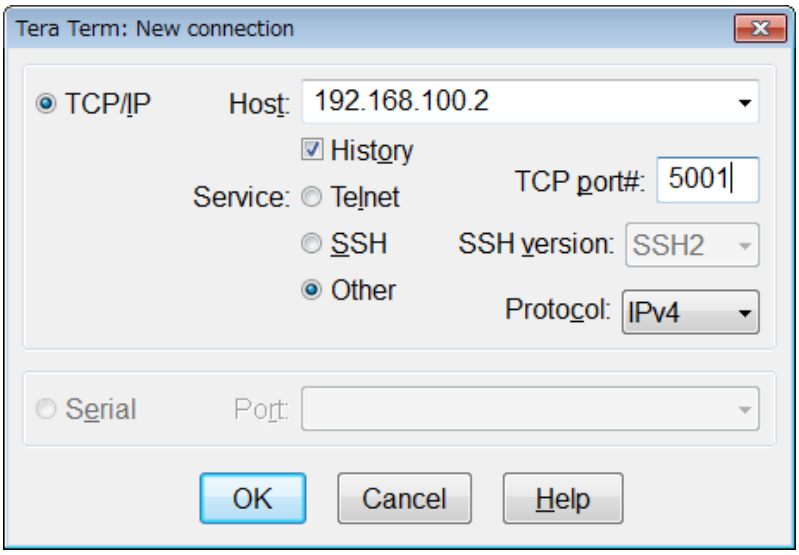
The above confirms that control using remote commands is possible. Input of similar commands can be used to confirm the operation of remote commands.

2.4.2 When using Ethernet (Windows 7/Vista)

This section explains how to use the free software, Tera Term Version 4.69.

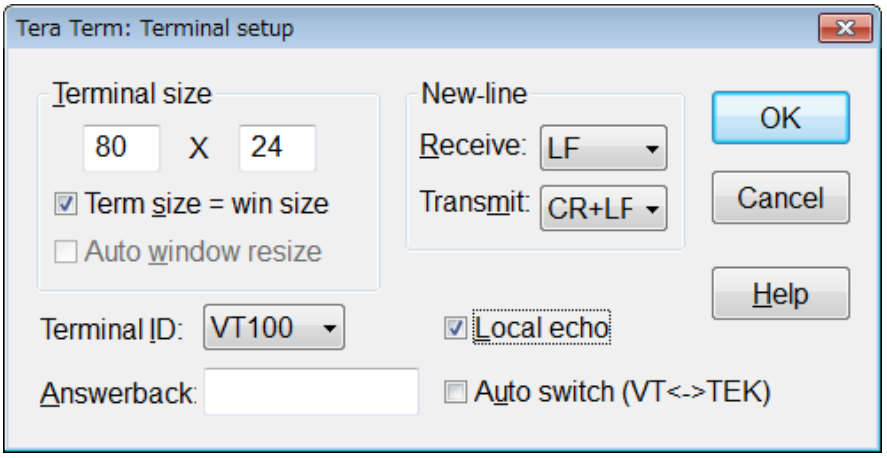
1. When starting Tera Term, the [New connection] window is opened.
   Enter the IP address and TCP port number in the [Host].
   Set the service to [Others] and protocol to [IPv4].
   Click [OK].

   If the IP address is set to 192.168.100.2, and the port number is set to 5001, set as follows.



2. When the MP2100A/MP2101A/MP2102A BERTWave is recognized, the communication window is displayed.
3. Click [Settings (S)] - [Terminal (T)..] on the menu.

4. Set the return cord reception to [LF] and those of counterpart to [CR+LF]. Check the local echo and click [OK].



6. Send *IDN?.
   Confirm that the response is displayed from the MP2100A/MP2101A/MP2102A BERTWave.

Note:

When the panel lock is released by the panel operation, the communication with the MP2100A/MP2101A/MP2102A BERTWave is terminated and Tera Term will be closed.

2.4.3 When using GPIB

1. Install the software drivers for the GPIB interface.
2. Run the software.
   For the operation method, refer to the GPIB interface operation manual.
3. Check the displayed instrument address.

2.5 Message Format

2.5.1 Message Types

Messages are composed of the character strings indicating message and message end. The character string indicating the message end is LF (Line Feed) or CR (Carriage Return)+LF.

Messages are composed of the following types depending on the transmission direction:

Program Messages

Messages sent from PC to instrument

There are two types of the program messages:

- Command
  This can be used for measurement condition settings and measurement start.
- Query
  This queries the status and settings of the measuring instrument.
  When transmitting the query, the instrument creates a response message to the query.

Response Messages

Messages sent from instrument to PC controller

2.5.2 Message Configuration

The messages are composed of header and data parts separated by more than a half width space.

Program messages always have a header but sometimes have no data.
Response messages always have data but sometimes have no header.

Header

The command header has the following types:

- Simple header
  The header is composed of alphanumeric characters and underbars, and the initial character is an alphabetic character.
  Example: `STA`
- Common command header
  The header is composed of alphanumeric characters and underbars, and the initial character is an asterisk (*).
  Example: `*CLS`
- Multiple headers
  Single headers are linked by colons. Colons can be used at the header. Multiple headers can be used to configure layered processing.
  Example: `:SENSE:MEASURE:START`

Queries have a question mark (?) appended to the header.

Example:
```
*ESE?
:CONFIGURE?
```

Data

The data format is character string data, numeric data, and binary data.

String data is ASCII code enclosed in quotation marks.
An example of the program message when inputting Model ANR-005 at the title is shown below.

Example:
```
:SYSYEM:MEMORY:STORE 'Model ANR-005',0,ALL
:SYESEM:MEMORY:STORE "Model ANR-005",0,ALL
```

When quotation marks are included in the character string, paired marks are used.

Example:
```
He said "Good product". → "He said ""Good Product""."
He said 'Good product'. → 'He said ''Good Product''.'
```

In addition, paired quotation marks can be used inside other paired quotation marks.

Example:

```
He said "Good product". → 'He said "Good Product".'
He said 'Good product'. → "He said 'Good Product'."
```

The numeric values can be described by using numeric data, input numeric values either as decimal, binary, octal, or hexadecimal numbers. When using the binary, octal, or hexadecimal numbers, put #B,#O, or #H before the data.

Example:

```
10   #B1010         #O12          #HA
1550 #B11000001110  #O3016        #H60E
```

When using decimal numbers, use integer number, fixed point, and floating point. The following examples indicate the same values.

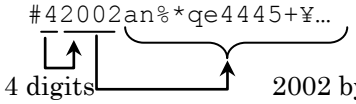
Example:

```
-10          -10.00        -1E1
1250         1250.000      1.25E3
0.0023                     2.3E-4
```

For the binary data, the head string starts with a sign (#) and continues with data after a numeric value indicating the data length.

The data length is displayed when the next character of the sign (#) is other than 0.
The binary data follows the number indicating the data length.

Example: `#42002an%*qe4445+¥…`



When the character after the sharp symbol (#) is 0, binary data continues after 0.

Example: `#0an%*qe4445+¥…`

Binary data

Messages with multiple data use commas (,) to separate data parts.

Example:
```
:INPUT:DATA:ATTFACTOR 1,6
:SENSE:MEASURE:EALARM:PERIOD 0,0,1,0
```

When linking multiple program messages, separate the message using semicolons (;).

Example:
```
*CLS; :DISPLAY:MODE:EYE ; :SAMPLING:STATUS RUN
```

2.5.3 Common Commands

The GPIB specifications (IEEE 488.2) define equipment commands. In this manual, these defined commands are called common commands.

The common commands are divided into mandatory and option commands. The MP2100A/MP2101A/MP2102A supports the common commands listed in Table 2.5.3-1.

Table 2.5.3-1 Common Commands

Command	Explanation
*CLS	Clears stand event register and output queue
*ESE	Sets and queries standard event enable register
*ESR	Queries standard event register
*IDN	Queries product information
*OPC	Sets/queries bit setting and bit 0 for status byte indicating message processing completion
*OPT	Queries option information
*RST	Initializes MP2100A/MP2101A/MP2102A setting conditions
*SRE	Sets and queries SRER
*STB	Queries status byte register
*TRG	Starts measurement
*WAI	Waits previous sent message completion

2.5.4 Device Dependant Commands

In this manual, commands that differ according to the functions of the measuring instrument are called Device Dependant Commands.

This instrument has two types of Device Dependant Commands.

- SCPI
  Commands meeting SCPI standard
- Native
  Commands consisting of at least three ASCII characters
  For details, refer to the Native version of the MP2100A/MP2101A/MP2102A BERTWave Remote Control Operation Manual.

2.6 Checking Instrument Status

This instrument has registers indicating status, such as errors and command execution status. This section explains these registers.

2.6.1 Register Structure

Figure 2.6.1-1 shows the structure of the registers indicating the instrument status.

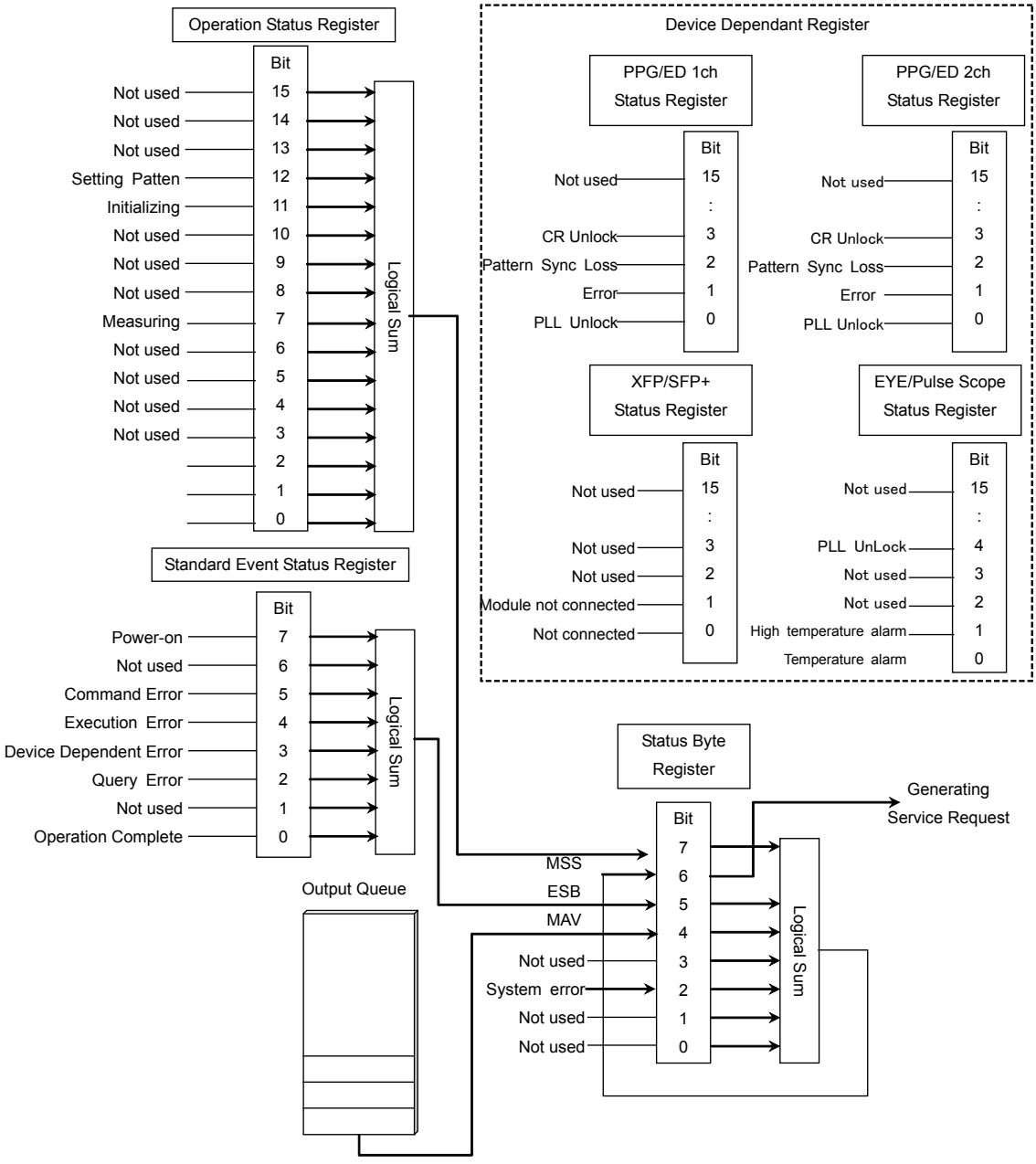


Figure 2.6.1-1 Register Structure

Each register uses 8-bit or 16-bit data. The register output values are the decimal totals for each bit shown in Table 2.6.1-1.

Table 2.6.1-1 Register Bit Decimal Conversion Values

Bit	Decimal value	Bit	Decimal value
0	1	8	256
1	2	9	512
2	4	10	1024
3	8	11	2048
4	16	12	4096
5	32	13	8192
6	64	14	16382
7	128	15	32764

The service request enable register (SRER) has a corresponding status byte register.

2.6.2 Status Byte Register

The status byte register (STB) displays the status of equipment defined by the GPIB standards. When the equipment status changes, the value in the STB changes too. It can be used to generate interrupts to the PC controller. These interrupts are called service requests.

There is a service request enable register (SRER) for the STB. The SRER can select the status byte bit generating the service request.

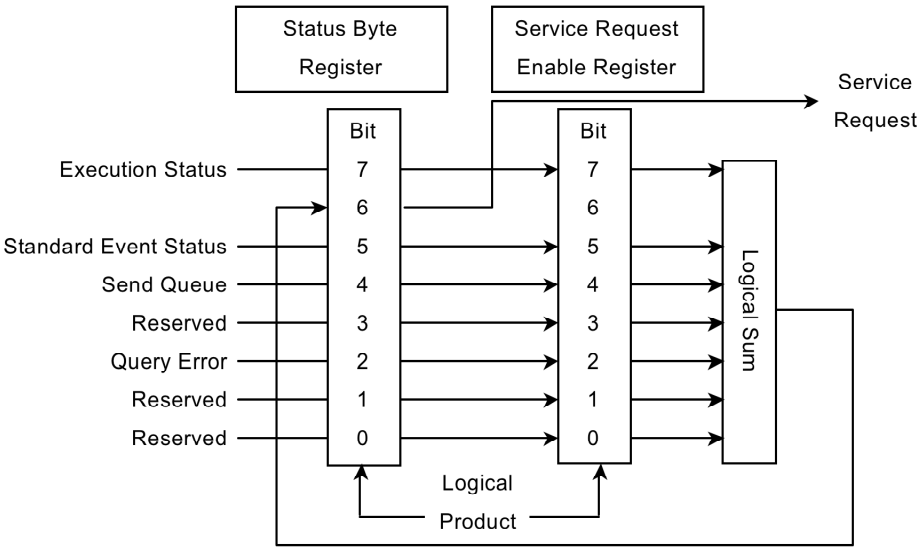


Figure 2.6.2-1 Configuration of Status Byte Register and Service Request Enable Register

Note:

When using the GPIB interface, the service request is enabled.

The following methods are used to read the status byte register.

- Using common *STB? command
- Using GPIB serial poll (when Opt-030 installed)

Read the GPIB interface manual for the serial poll method.
When using serial polling, even if bit 6 is 1, it becomes 0 after reading once.

The `*SRE` and `*SRE?` common commands can be used for setting and reading the SRER for setting reading of the status byte register. To output the STB data, set the bit corresponding to the SRER to 1.

The meaning of each bit of the STB is shown in the following table.

Table 2.6.2-1 Meaning of Status Byte Register

Bit	Explanation
7	This is the logical sum of each bit of the logical product of the OSR and its event enable register.
6	MSS (Master Summary Register) It is the logical sum of the bit 5 to 0, bit 7 logical product of the STB and the SRER.
5	This is the logical sum of each bit of the logical product of the standard event status register and standard event enable register.
4	MAV (Message Available summary) This is always 1 when there is a response message in the output queue of this instrument
3	Not used; always 0
2	Becomes 1 at System Error
1	Not used; always 0
0	Not used; always 0

Bit 7 of the STB indicates information about the OSR.
For details about the information, refer to section 2.6.4 Operation Status Register.

Bit 6 of the STB is called the master summary status (MSS) bit. When it is 1, there is a notification from this instrument to the PC controller. When it changes to 1 from 0, a service request is generated.

Bit 5 of the STB indicates information about the standard status register. For details about the information, refer to section 2.6.3 Standard Event Status Register.

The device dependant register data is not indicated in the STB.

Bits 7 and 5 of the STB can be set to 0 using the `*CLS` common command. When `*CLS` is sent after a command or when a query is sent after `*CLS`, the send queue is cleared and bit 4 is set to 0.

The SRER cannot be set to 0 by `*CLS`, so use `*SRE`.

2.6.3 Standard Event Status Register

There is a standard event status enable register (ESE) for the standard event status register (ESR). The logical product of these two registers and the logical sum of each bit of this result is output to bit 5 of the STB.

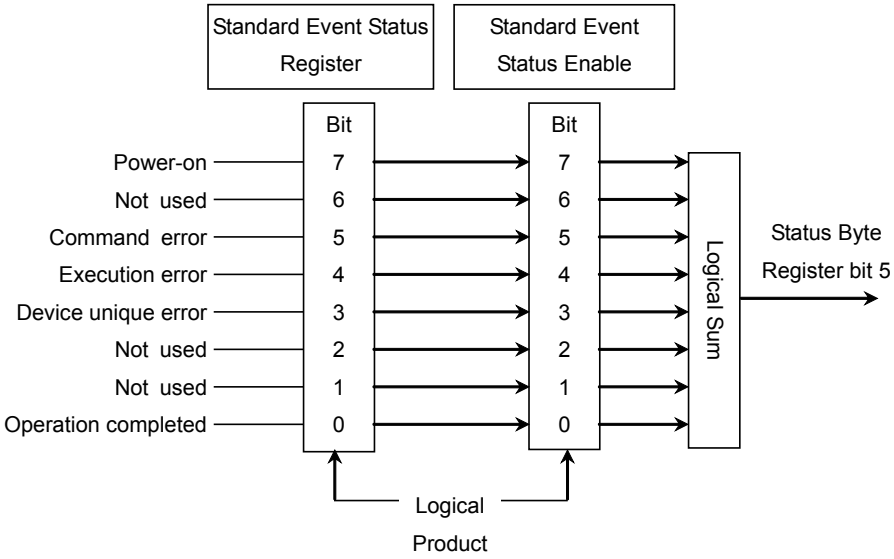


Figure 2.6.3-1 Configuration of Standard Event Status Register and Standard Event Status Enable Register

The meaning of each bit of the ESR is listed in the table below.

Table 2.6.3-1 Meaning of Standard Event Status Register

Bit	Explanation
7	Power-on Becomes 1 at power-on and returns 0 when read
6	Not used; always 0
5	Command Error Becomes 1 when received undefined program message, message that cannot executed according to syntax, or message with spelling error
4	Execution Error Becomes 1 when received program message that cannot be executed
3	Device Dependent Error Becomes 1 at errors other than command, execution and query errors
2	Not used; always 0
1	Not used; always 0
0	Operation Complete Becomes 1 when entire command operation completed after *OPC command operation

For the details of bits 5, 4, 3, refer to Appendix B Message Code.

A bit 5 command error occurs under the following circumstances:

- When instruments received message not described in Section 4.
  Examples:
  Typographical error in message
  2 byte code character in message
  No space separator between message and parameter
  No semicolon separators between multiple messages
  Omitted characters in message that cannot be omitted
- Incorrect parameter count
  Examples:
  Sent two parameters when only 1 parameter defined in message
  No comma separators between parameters
- When parameter format incorrect
  Example:
  Sent string to message where numerical value defined as parameters
  Character string not enclosed by quotation marks

A bit 4 execution error occurs under the following circumstances:

- When the parameter setting is out of range
  Examples:
  Bit rate offset set to 150 ppm (Setting range: –100~100)
  External attenuation factor set to 100 dB (Setting range:0~30)
- Command correct but cannot be executed in current equipment status
  Examples:
  PPG/ED2 bit rate set when no PPG/ED Ch2 option installed
  CRU Loop Band set when CRU option not installed

A bit 3 device error occurs under the following circumstances:

- When a system alarm is generated
  Examples:
  System alarm (Temperature) generated when equipment internal temperature rises
  System error (PLL Unlock) generated when clock signal not detected while using external clock

Bit 7 to bit 0 of the ESR can be read by the `*ESR?` command.
The standard event register returns to 0 when read.

The ESE can be set and read using the `*ESE` and `*ESE?` commands. To output standard event register data, set the bit corresponding to the enable register to 1.
The bit 0 can be read using the `*OPC` command.
The standard register can be set to 0 using the `*CLS` command.

2.6.4 Operation Status Register

The operation status register (OSR) is composed of the following registers:

- Operation status condition register
- Transition filter
- Operation status event register
- Operation status enable register (OSER)

The operation status condition register indicates changes in the status. When the status changes, the value of this register also changes.

The OSER records changes in the value of the execution status condition register. There is a transition filter that defines the write condition before the OSER. The transition filter sets the OSER to 1 under any of the following conditions:

- When bit changes from 0 to 1
- When bit changes from 1 to 0
- When bit changes from 0 to 1 or bit changes from 1 to 0

The OSER sets the OSER output at each bit. The logical product these two registers is obtained and the logical sum of each bit of the result is output at bit 7 of the STB.

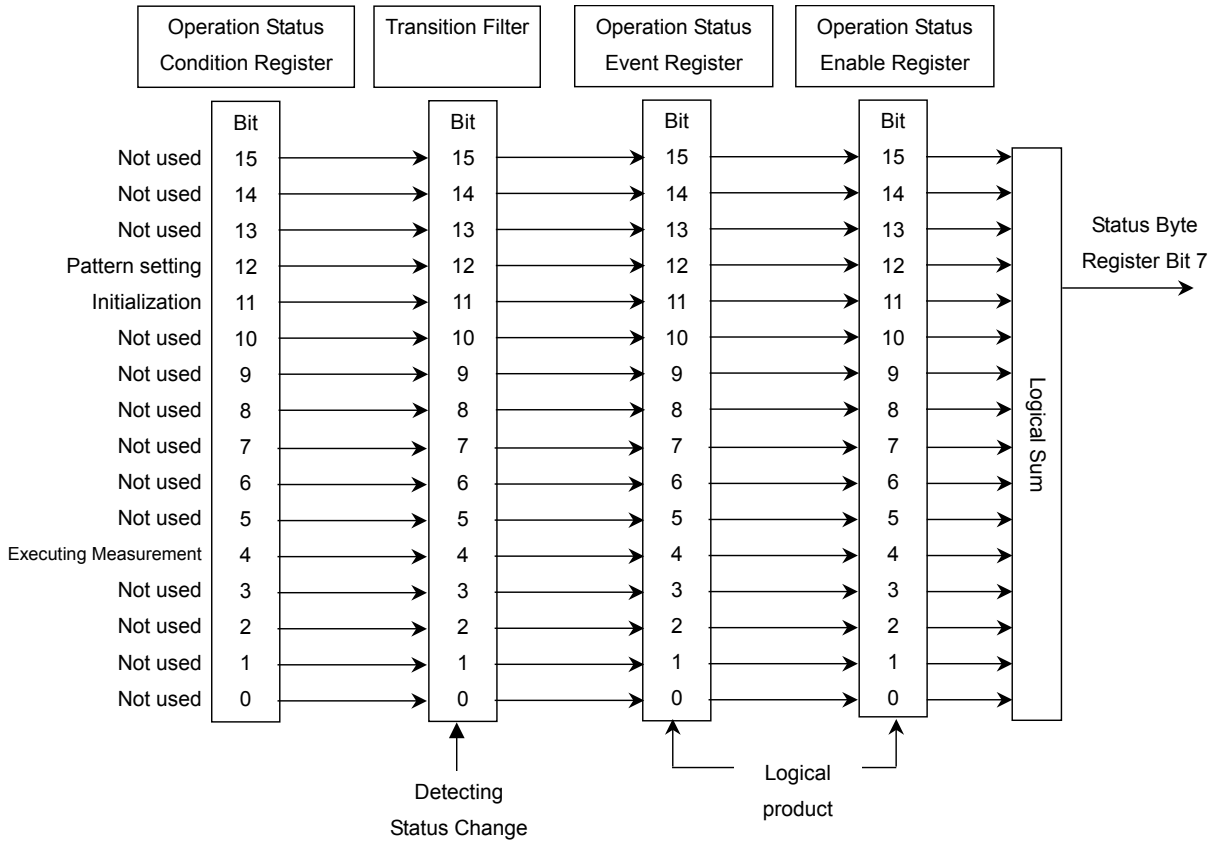


Figure 2.6.4-1 Configuration of Operation Status Condition Register, Operation Status Event Register, Operation Status Enable Register, and Transition Filter

Each bit definition of the execution status is as follows.

Table 2.6.4-1 Bit Definition of Operation Status Register

Bit	Explanation
16–13	Not used; always 0
12	Sets to 1 at start or end time for initializing processing
11	Sets to 1 at start or end time for pattern setting processing
10–5	Not used; always 0
4	Sets to 1 at start or end time for next measurement • Error measurement using error detector • Performing sampling using sampling scope
3–0	Not used; always 0

The commands for confirming the execution start or end time at the OSR are shown in the following table.

Table 2.6.4-2 Commands for Confirming Execution of operation at Operation Status Register

Operation Status Register Bit	Command
12	:SYSTem:MEMory:INITialize
11	:SENSe:MMEMory:PATTern:RECall, :SENSe:PATTern:TYPE, :SOURce:MMEMory:PATTern:RECall, :SOURce:PATTern:TYPE
4	:SENSe:MEASure:ASTP :SENSe:MEASure:ASTRt, :SENSe:MEASure:STARt, :SENSe:MEASure:STOP, [:SENSe]:SAMPling:STATus

To detect the execution start, the transition filter bit response to STATus:OPERation:PTRansition is set to 1.
To detect the execution end, the transition filter bit response to :STATus:OPERation:NTRansition.
The OSER can be read using :STATus:OPERation:[EVENt]?. When the register is read, the OSR returns to 0.

The operation status condition register can be read using:STATus:OPERation:CONDition?.

To set the OSER, use:STATus:OPERation:ENBle. To read the OSER, use STATus:OPERation:ENBle?. To output the OSR data, set the bit for the status setting enable register to 1.

When sending :STATus:OPERation:RESet, the operation status event register is set to 0.
Even when sending :STATus:OPERation:RESet, the OSER is not changed.

2.6.5 Device Dependant Register

The following register is called the device dependant register.

- PPG/ED Ch1 Status Register
- PPG/ED Ch2 Status Register
- XFP/SFP+ Status Register
- EYE/Pulse Scope Status Register

The device dependant status register has the same type of condition register, transition filter, and event register. However there is no enable register for switching the output at each bit on/off.

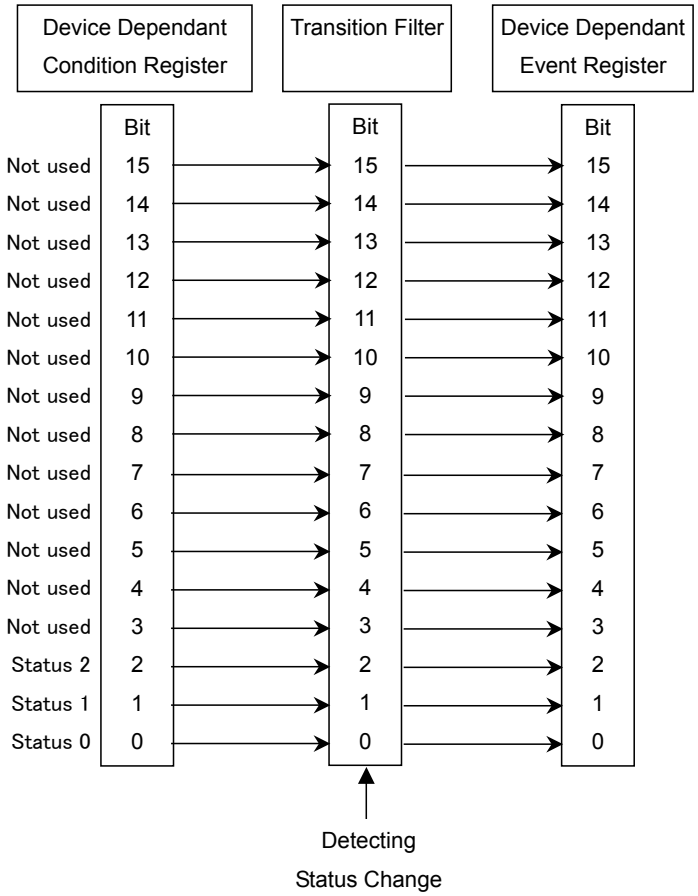


Figure 2.6.5-1 Configuration of Device Dependant Register

When the value of the device dependant status register changes, there is no effect on the STB. As a result, a service request is not generated to the PC controller.

Each bit definition of the device dependant register is as follows.

Table 2.6.5-1 Meaning of PPG/ED Ch1 and PPG/ED Ch2 Status Register

Bit	Explanation
16~4	Not used; always 0
3	Indicates CR Unlock occurs
2	Indicates Pattern Sync Loss occurs
1	Indicates Error occurs
0	Indicates PLL Unlock occurs

Table 2.6.5-2 Meaning of XFP/SFP+ Status Bit

Bit	Explanation
16~2	Not used; always 0
1	Indicates LOS occurs
0	Indicates Ready status

Table 2.6.5-3 Bit Meaning of EYE/Pulse Scope Status Register

Bit	Explanation
16~2	Not used; always 0
1	Indicates temperature alarm occurs (Fatal Temperature)
0	Indicates temperature alarm occurs (Temperature)

To detect the occurrence of these phenomena, set the transition filter bit to 1 using the following commands:

```
:INSTrument:PE1:PTRansition
:INSTrument:PE2:PTRansition
:INSTrument:XSFP:PTRansition
:INSTrument:WAV:PTRansition
```

To detect the end of these phenomena, set the transition filter bit to 1using the following commands:

```
:INSTrument:PE1:NTRansition
:INSTrument:PE2:NTRansition
:INSTrument:XSFP:NTRansition
:INSTrument:WAV:NTRansition
```

The device dependant event register can be read using the following queries:

```
:INSTrument:PE1:[EVENt]?
:INSTrument:PE2:[EVENt]?
:INSTrument:XSFP:[EVENt]?
:INSTrument:WAV:[EVENt]?
```

The device dependant condition register can be read using the following queries:

```
:INSTrument:PE1:CONDition?
:INSTrument:PE2:CONDition?
:INSTrument:XSFP:CONDition?
:INSTrument:WAV:CONDition?
```

The device dependant event register can be initialized using the following queries:

```
:INSTrument:PE1:RESet
:INSTrument:PE2:RESet
:INSTrument:XSFP:RESet
:INSTrument:WAV:RESet
```

2.7 Confirming Message Execution Status

When using the GPIB interface, the message cannot be sent to the MP2100A/MP2101A/MP2102A while executing the message sent previously.

CAUTION

When using the GPIB interface, set the PC controller GPIB interface timeout to at least 30 seconds to ensure that communications do not time-out while this instrument is executing messages.

When using the Ethernet interface, the message can be sent even while MP2100A/MP2101A/MP2102A is executing the program message. However, the following message is not processed until the previous message processing is completed.

When using the Ethernet interface, confirm the completion of the message sent before sending the next message.
To confirm the message execution end, sent the query.
After the previous message processing is completed, the query processed then PC receives the response.

Example of Use:

`:Calibrate:Amplitude`	Starts Level calibration for EYE/Pulse Scope
`:System:Error?`	Queries error
`> 0,"No Error"`	No errors when 0 read

In these examples, there may be a delay ranging from a few seconds to 20 or 30 seconds until the response message is returned after sending `:System:Error?`
Execution of the `:Calibrate:Amplitude` and `:System:Error?` messages is confirmed by receiving the response message.

Chapter 3 Sample Program

This chapter explains examples of sample programs and how to execute them.

3.1 Executing Sample Programs

3.1.1 Setting Sample Program Operating Environment

The sample program operating environment is as follows.

PC

OS: Windows XP Professional Service Pack 2

VISA: NI-VISA Version 4.6

Program tool: Microsoft Visual Visual C# 2008

MP2100A BERTWave

GPIB Address: 1

IP Address: 198.168.12.10

Subnet Mask: 255.255.255.0

Port Number: 5001

Installing NI-VISA

To use VISA at Visual C# 2008, add the following function at installation.

- Development Support .NET Framework 3.5 Language Support
- NI Measurement & Automation Explore —.NET Framework 3.5 Language Support

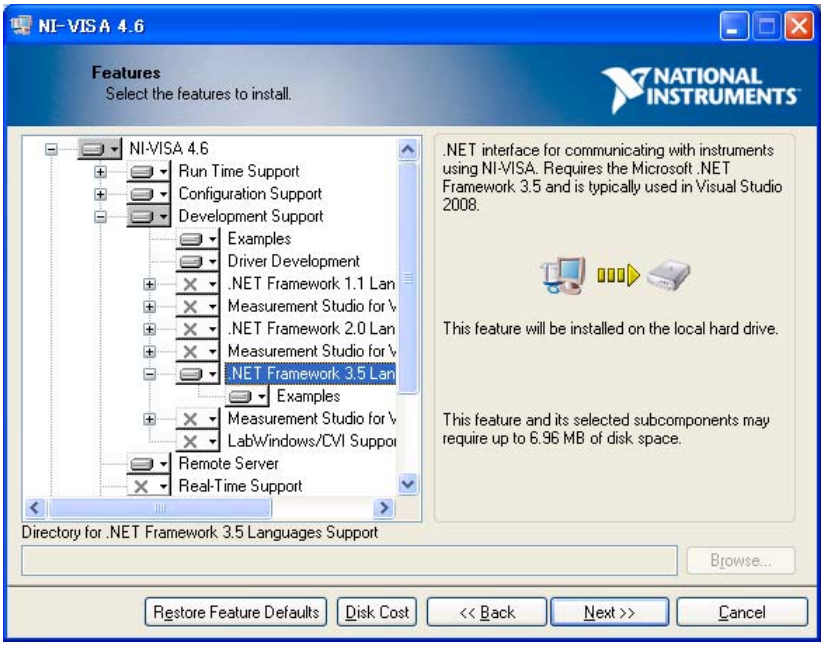


Figure 3.1.1-1 Function Selection Screen at VISA Install

Setting Visual C# 2008

To use VISA at Visual C# 2008, operate as follows.

1. Click [Add Reference] at the Project menu.
2. Click the .NET tab in the Add Reference dialog box.
3. Select National Instruments Common and National Instruments VisaNS, and click OK.

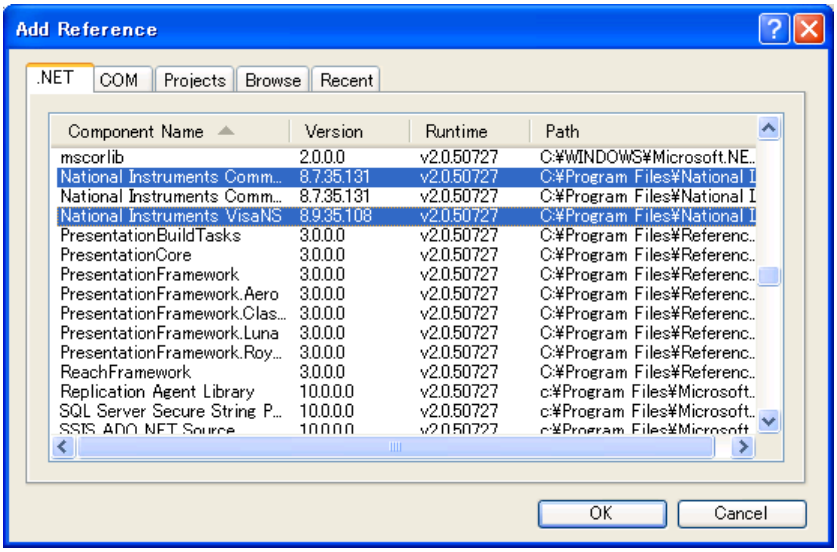
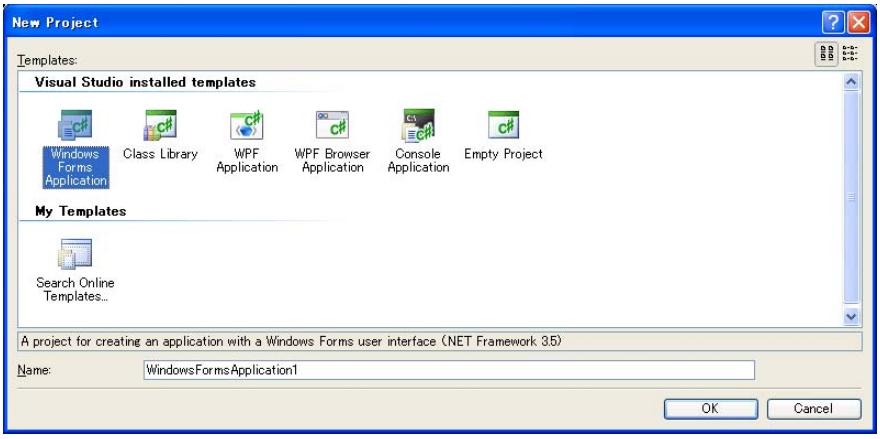


Figure 3.1.1-2 Add Reference Dialog Box

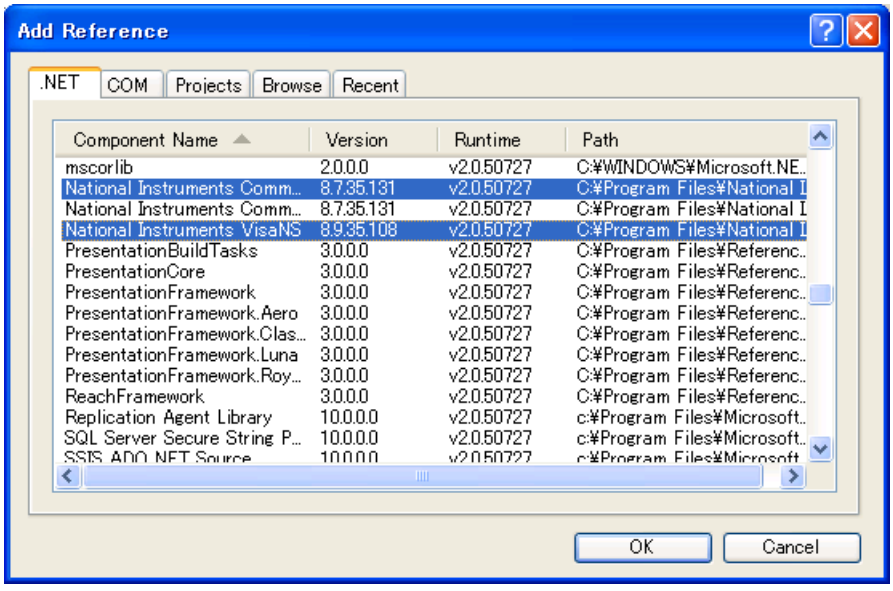
3.1.2 Executing Sample Program

The executing procedure for the sample program is as follows.

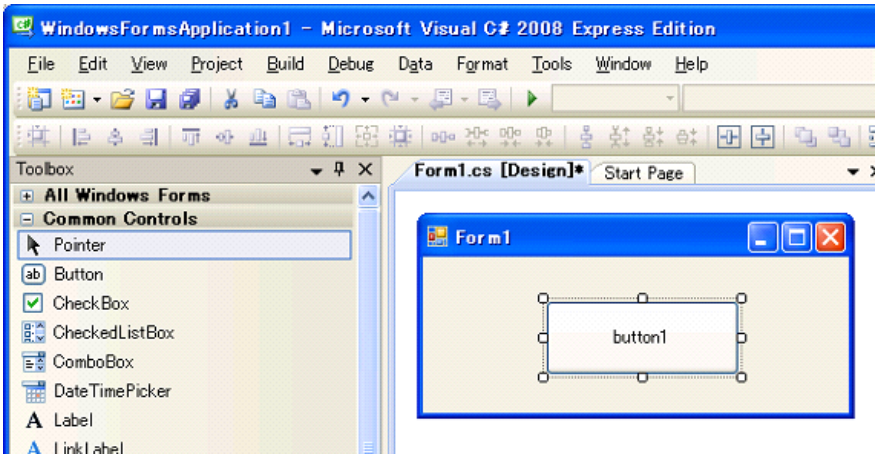
1. Start the Visual C# 2008.
2. Click [New Project] from the File menu.
3. Select the Visual C# Windows Forms Application and click [OK].



4. Start the screen editor and click [Add Reference] at the Project menu.
5. Click the .NET tab in the Add Reference dialog box.
6. Select National Instruments Common and National Instruments VisaNS and click [OK].



7. Referring to the sample program screen design figure, arrange control of the buttons in Form1.cs [Design].



8. Double-click the arranged button to open the screen for inputting the source code.

```
private void button1_Click(object sender, System.EventArgs
  e)
{
}
```

9. Copy the sample program in this document and paste it into the Form1.cs screen.

```
private void button1_Click(object sender, System.EventArgs
    e)
{
            //Paste it into this part.
    }
```

10. Change the IP address and GPIB address.

    For a LAN connection, the part "192.168.12.10" in the program must be changed to the IP address set at the MP2100A/MP2101A/MP2102A.

    For a GPIB connection, number of the part "GPIB::1::INSTR" in the program must be changed to GPIB address of MP2100A/MP2101A/MP2102A.

11. Click [Open Debug] from the Debug menu.

3.2 Example 1: Controlling Pulse Pattern Generator

This sample program controls the instrument via the Ethernet interface.

Add the following declaration before starting the program.

```
using System.Net;
using System.Net.Sockets;
```

Processing Flow

1. Define the TCP/IP client class for the IP address 192.168.12.10 and port number 5001.
2. Send :MODULE:ID 1 to set the control target to PPG/CH_Ch1.
3. Set the reference clock to the internal clock.
4. Set the bit rate specifications to 1GbE.
5. Set the pattern to PRBS2^23–1.
6. Set the amplitude to 0.5 V.
7. Set the error insertion to Off.
8. Output the signal of the PPG/CH_Ch1.
9. Set all module output to On.
10. Query errors.

```
//  set IP address of BERTWave
string IPadr="192.168.12.10";
//  set  port number of BERTWave
Int32 port = 5001;
TcpClient client = new TcpClient(IPadr, port);
NetworkStream stream = client.GetStream();

//  send messages
string message = ":MOD:ID 1¥n";
Byte[] data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":OUTPUT:CMU:REFCLOCK INTERNAL¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":OUTPUT:BITRATE:STANDARD '1GBE'¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":SOURCE:PATTERN:TYPE PRBS23¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":OUTPUT:DATA:AMPLITUDE DATA,0.5¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":SOURCE:PATTERN:EADDITION:SET OFF¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":OUTPUT:DATA:OUTPUT ON¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":SYSTEM:ERROR?¥n";
```

```
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

//  read response
data = new Byte[256];
String responseData = String.Empty;
Int32 bytes = stream.Read(data, 0, data.Length);
responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes);
Console.WriteLine("Received: {0}", responseData);

// Close everything.
stream.Close();
client.Close();
```

3.3 Example 2: Controlling Error Detector

This sample program controls the instrument via the GPIB interface.

Processing Flow

1. Define the class of the GPIB address 1.
2. Send :MODULE:ID 1 to set the control target to PPG/CH_Ch1.
3. Set the bit rate specifications to 10G FC.
4. Set the pattern to PRBS2^23–1.
5. Set the input connector to Data only.
6. Set the threshold value to 0 V.
7. Set the auto-pattern sync to On.
8. Set the threshold value of the auto-pattern sync threshold value to 1E−5.
9. Set the error measurement method to Single.
10. Set the measurement time to 20 seconds.
11. Start the measurement.
12. Query errors.

```
// Open session
NationalInstruments.VisaNS.MessageBasedSession mbs =
(NationalInstruments.VisaNS.MessageBasedSession)
NationalInstruments.VisaNS.ResourceManager.GetLocalManager().
Open("GPIB::1::INSTR");

mbs.Timeout = 30000; // Timeout 30sec

//  Initialize BERTWave.
mbs.Write("*RST¥n");
//  Select module as PPG/ED_Ch1.
mbs.Write(":MODULE:ID 1¥n");
//  Set bitrate standard of ED as 1 Giga bit Ethernet.
mbs.Write( ":INPUT:BITRATE:STANDARD '10G_FC'¥n");
//   Set test pattern of ED as PRBS2^23-1.
mbs.Write(":SENSE:PATTERN:TYPE PRBS23¥n");
//   Set input connector as Data only.
mbs.Write( ":INPUT:DATA:INTERFACE DATA¥n");
//   Set threshold voltage as 0V.
mbs.Write( ":INPUT:DATA:THRESHOLD 0¥n");
//   Set automatic pattern synchronization as On.
mbs.Write(":SENSE:PATTERN:SYNC:ASYNC ON¥n");
//   Set threshold level of automatic synchronization as 10^-5.
mbs.Write(":SENSE:PATTERN:SYNC:THRESHOLD E_5¥n");
//   Set measure mode as Single.
mbs.Write(":SENSE:MEASURE:EALARM:MODE SINGLE¥n");
//   Set measuring time as 20 seconds.
mbs.Write( ":SENSE:MEASURE:EALARM:PERIOD 0,0,0,20¥n");
//   Start measurement.
mbs.Write(":SENSE:MEASURE:START¥n");
string ret = mbs.Query(":SYSTEM:ERROR?¥n");
Console.WriteLine(ret);
```

3.4 Example 3: Controlling Optical Transceiver

This sample program controls the instrument via the Ethernet interface.

Add the following declaration before starting the program.

```
using System.Net;
using System.Net.Sockets;
```


Processing Flow

1. Define the TCP/IP client class for the IP address 192.168.12.10 and port number 5001.
2. Send :MODULE:ID 3 to set the control target to XFP/SFP+.
3. Query whether the optical transceiver is installed or not.
4. Query the wavelength of the optical transceiver.
5. Set the optical transceiver output to ON.
6. Query errors.

```
// set IP address of BERTWave
string IPadr = "192.168.12.10";
// set port number of BERTWave
Int32 port = 5001;
TcpClient client = new TcpClient(IPadr, port);
NetworkStream stream = client.GetStream();

// send messages
string message = ":MOD:ID 3¥n";
Byte[] data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":CALCULATE:OPTICAL:STATUS? 'READY'¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

// read response
data = new Byte[256];
String responseData = String.Empty;
Int32 bytes = stream.Read(data, 0, data.Length);
responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes);
Console.WriteLine("Received: {0}", responseData);

if (responseData == "N")
{
// Result is "None"
Console.WriteLine("Optical Transceiver does not exist.");
return;
}

message = ":SOURCE:OPTICAL:SIGNAL:WLENGTH?¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

// read response
data = new Byte[256];
responseData = String.Empty;
bytes = stream.Read(data, 0, data.Length);
responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes);
Console.WriteLine("Received: {0}", responseData);
```

```
message = ":SOURCE:OPTICAL:SIGANL:OUTPUT 1¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

message = ":SYSTEM:ERROR?¥n";
data = System.Text.Encoding.ASCII.GetBytes(message);
stream.Write(data, 0, data.Length);
Console.WriteLine("Sent: {0}", message);

//  read response
data = new Byte[256];
responseData = String.Empty;
bytes = stream.Read(data, 0, data.Length);
responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes);
Console.WriteLine("Received: {0}", responseData);

// Close everything.
stream.Close();
client.Close();
```

3.5 Example 4: Controlling EYE/Pulse Scope

This sample program controls the instrument via the GPIB interface.

Processing Flow

1. Define the class of the GPIB address 1.
2. Send :MODULE:ID 5 to set the control target to EYE/Pulse Scope.
3. Set the display mode to the pulse mode.
4. Start the measurement.
5. Query the measurement status.
6. Query errors.

```
// Open session
 NationalInstruments.VisaNS.MessageBasedSession mbs =
(NationalInstruments.VisaNS.MessageBasedSession)
NationalInstruments.VisaNS.ResourceManager.GetLocalManager().
Open("GPIB::1::INSTR");

 mbs.Timeout = 30000; // Timeout 30sec

 mbs.Write(":MODULE:ID 5¥n");
 mbs.Write(":DISPLAY:WINDOW:CLEAR¥n");
 //   Set measuring mode as EYE mode.
 mbs.Write(":SENSE:DISPLAY:MODE PULSE¥n");
 //   Start measurement.
 mbs.Write(":SENSE:SAMPLING:STATUS RUN¥n");
 string ret = mbs.Query(":SENSE:SAMPLING:STATUS?¥n");
 Console.WriteLine(ret);
 ret = mbs.Query(":SYSTEM:ERROR?¥n");
 Console.WriteLine(ret);
```

Chapter 4 Message Details

This chapter describes the message details of remote control commands for MP2100A/MP2101A/MP2102A.

4.1 Description of Message Explanations

The following table shows the rules for describing messages.

Table 4.1 -1 Rules for Describing Messages

Symbols	Usage
<>	Parameters in angled bracket are input by the programmer.
[]	Messages or parameters in square brackets can be omitted.
|	One of several choices can be chosen. For example, if A|B|C|D are choices, select one of them.
{}	Group the choices. When A|B({C|D}) can be chosen, select one of them.
<binary>	This string is in binary data format.
<character>	Alphabet or numeric characters
<file_name>	The character strings indicate file name and path. The double quotation marks or single quotation marks are needed at the beginning and end of the data. ¥,/,:,*,?,",<,>, | are not used in the file name. Example: "PATTERN005"
<integer>	Decimal integer value Example: –100, 12500000
<numeric>	Decimal numeric value Example: 0, 1.2E–6, 2.35
<string>	Character string data The double quotation marks or single quotation marks are needed at the beginning and end of the data.
<switch>	Message original selection Example: 0,1,OFF,ON
<version>	Numeric value indicating version Multiple decimal points may be included.

Some parts of the header character strings can be omitted.
The small letter can be omitted, but the capital letter cannot be omitted.

Example: `:STATus:OPERation:EVENt?`

This header can be written as follows:

```
:STAT:OPER:EVEN?
:STAT:OPERATION:EVEN?
:STATUS:OPERAT:EVENT?
:STATUS:OPERATION:EVEN?
:STATUS:OPERATION:EVENT?
```

These messages are interpreted as the same meanings in the MP2100A/MP2101A/MP2102A.

4.2 Correspondence between Panel Operation and Message

This section explains correspondence between panel operation and message.

4.2.1 Message Corresponding to Common Operation

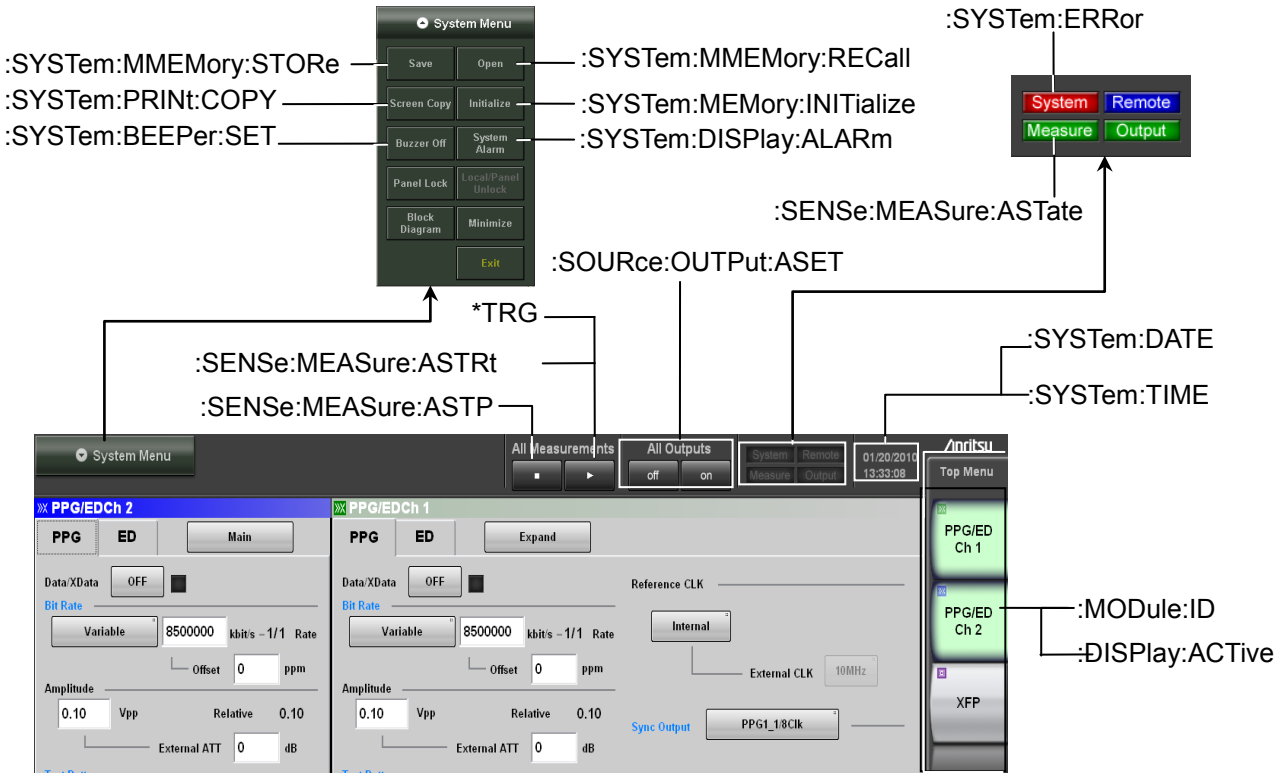


Figure 4.2.1-1 Message Corresponding to Common Operation

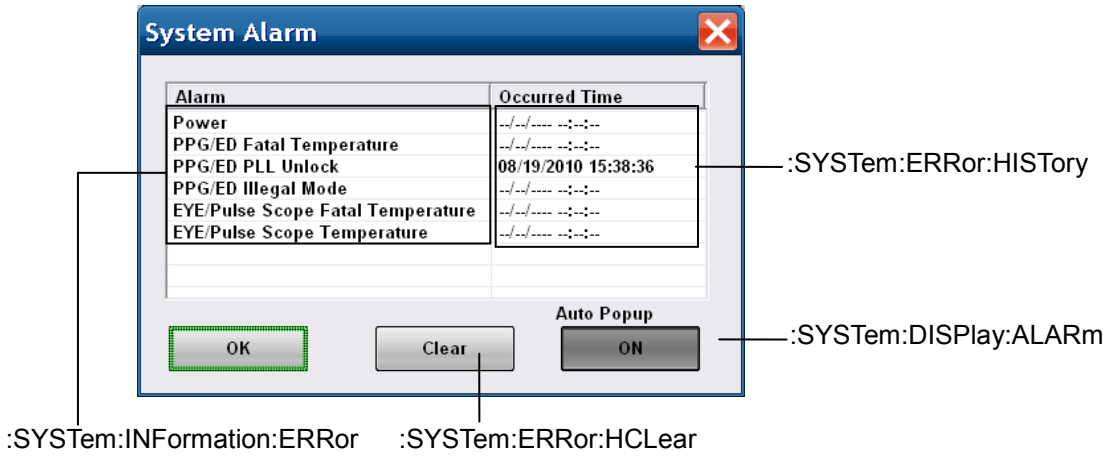
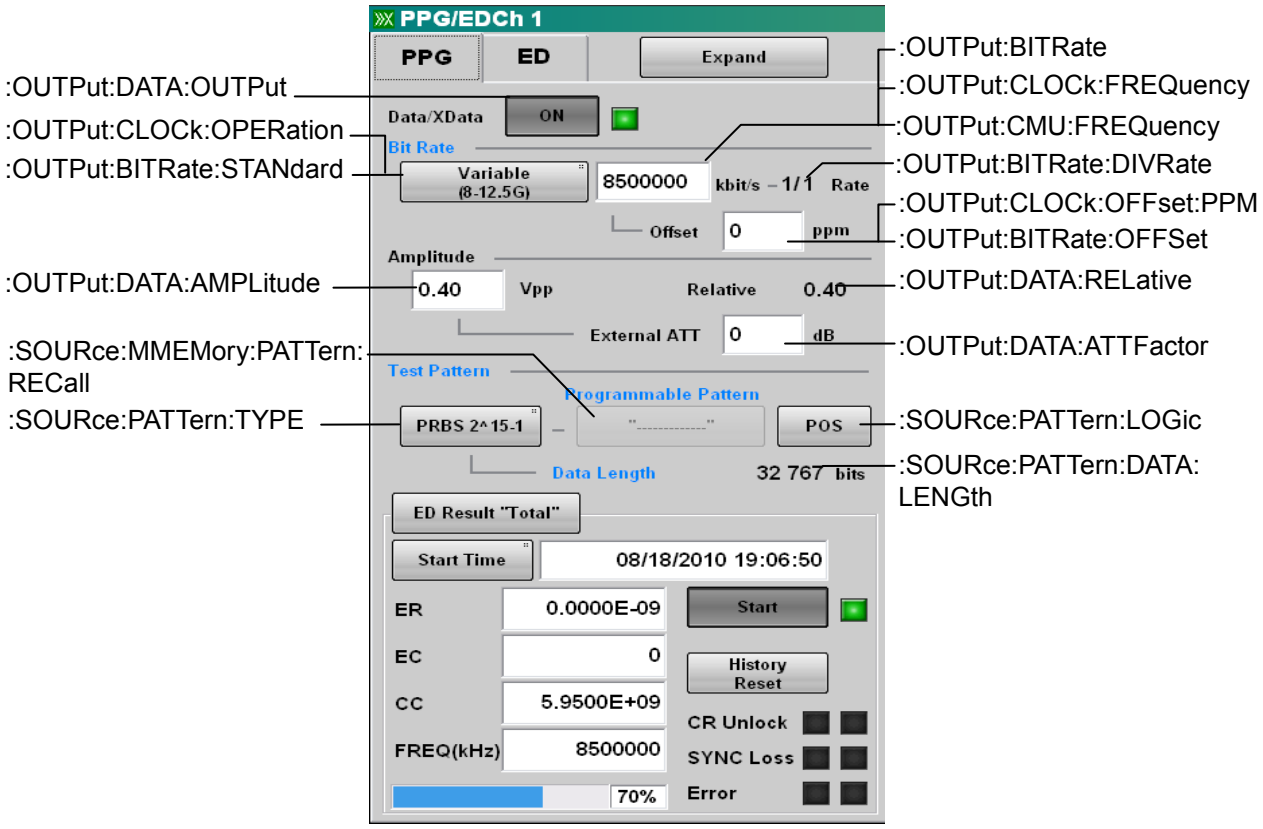


Figure 4.2.1-2 Message Corresponding to System Alarm Dialog

4.2.2 Message Corresponding to PPG/ED

When controlling PPG/ED Ch1, send :MODule:ID 1 at first.
When controlling PPG/ED Ch2. send :MODule:ID 2 at first.



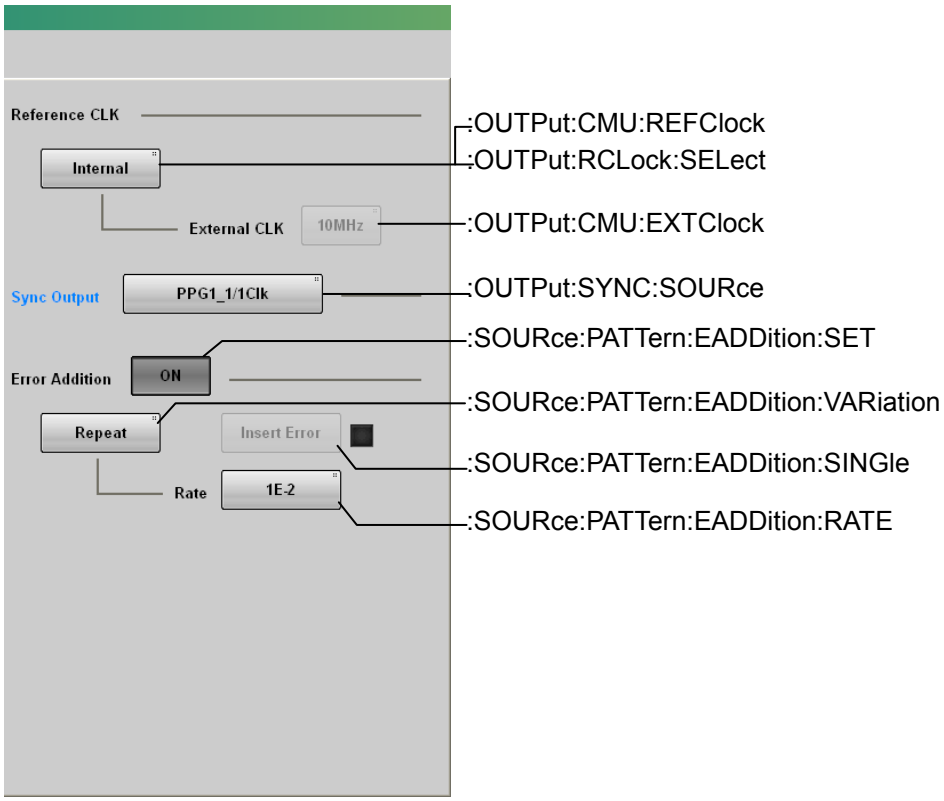


Figure 4.2.2-1 Message Corresponding to PPG Panel

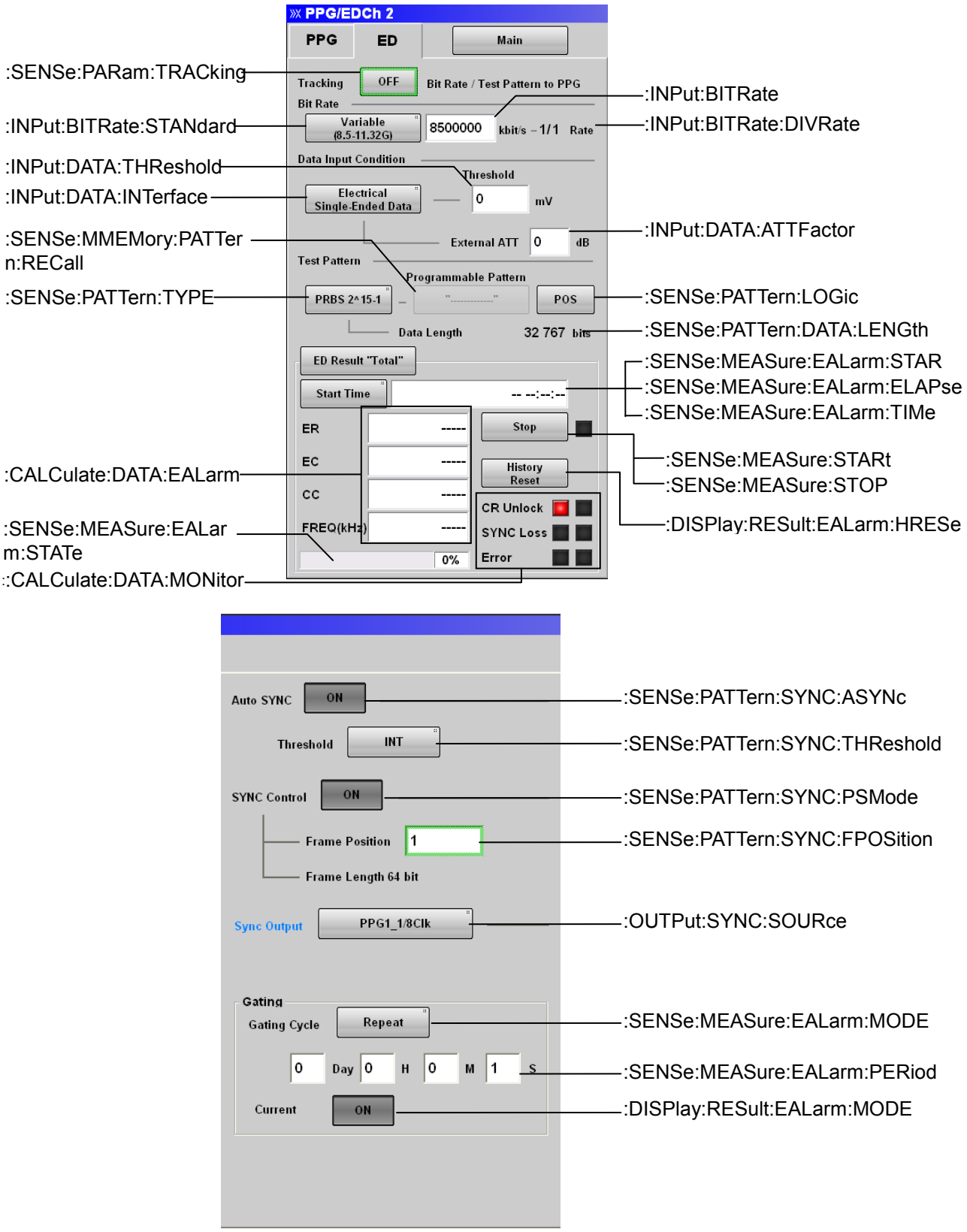


Figure 4.2.2-2 Message Corresponding to ED Panel

4.2.3 Message Corresponding to XFP/SFP+

When controlling XFP/SFP+, send :MODule:ID 3 at first.

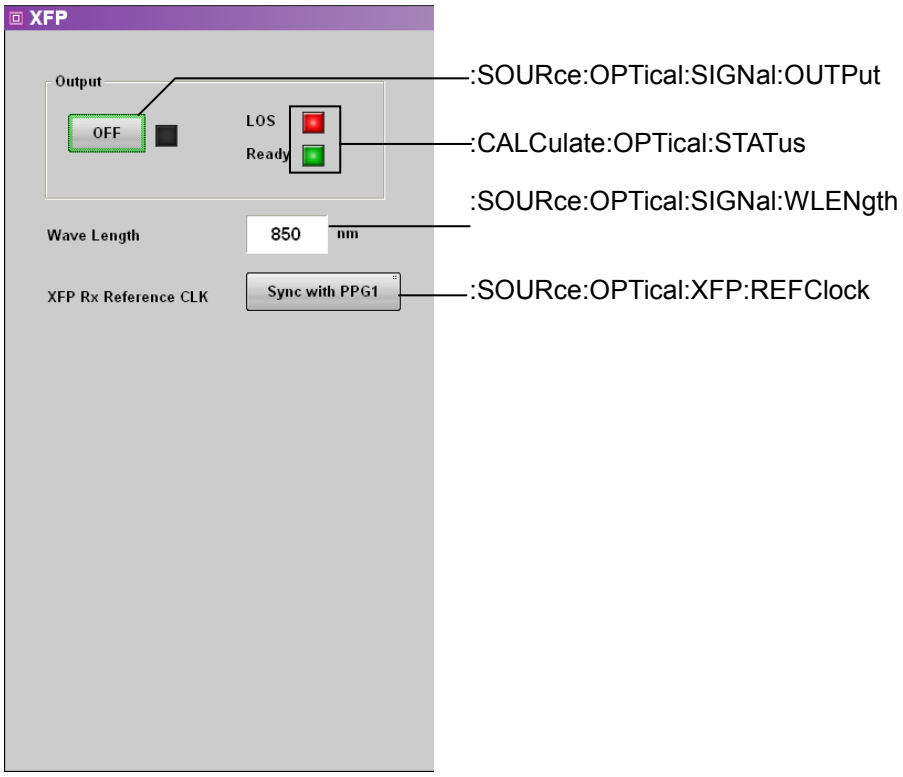


Figure 4.2.3-1 Message Corresponding to XFP/SFP+ Panel

4.2.4 Message Corresponding to O/E

When controlling O/E, send :MODule:ID 4 at first.

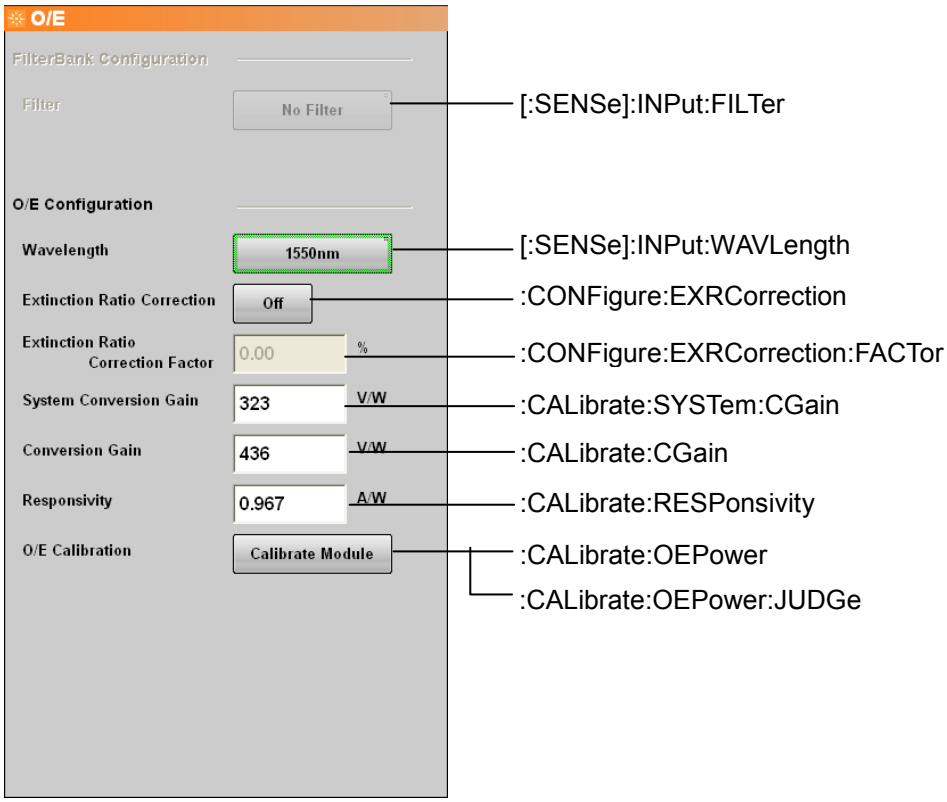


Figure 4.2.4-1 Message Corresponding to O/E Panel

4.2.5 Message Corresponding to EYE/Pulse Scope

When controlling EYE/Pulse Scope, send :MODule:ID 5 at first.

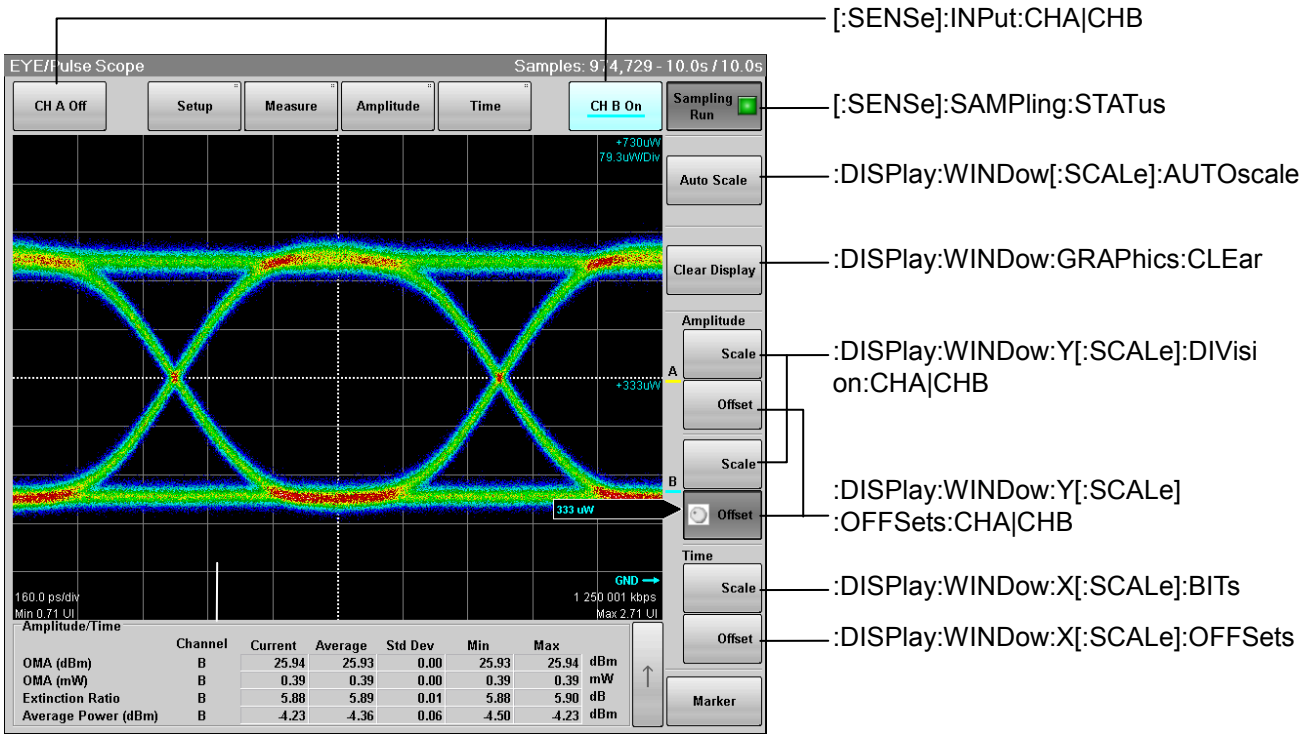


Figure 4.2.5-1 Message Corresponding to EYE/Pulse Scope Panel 1

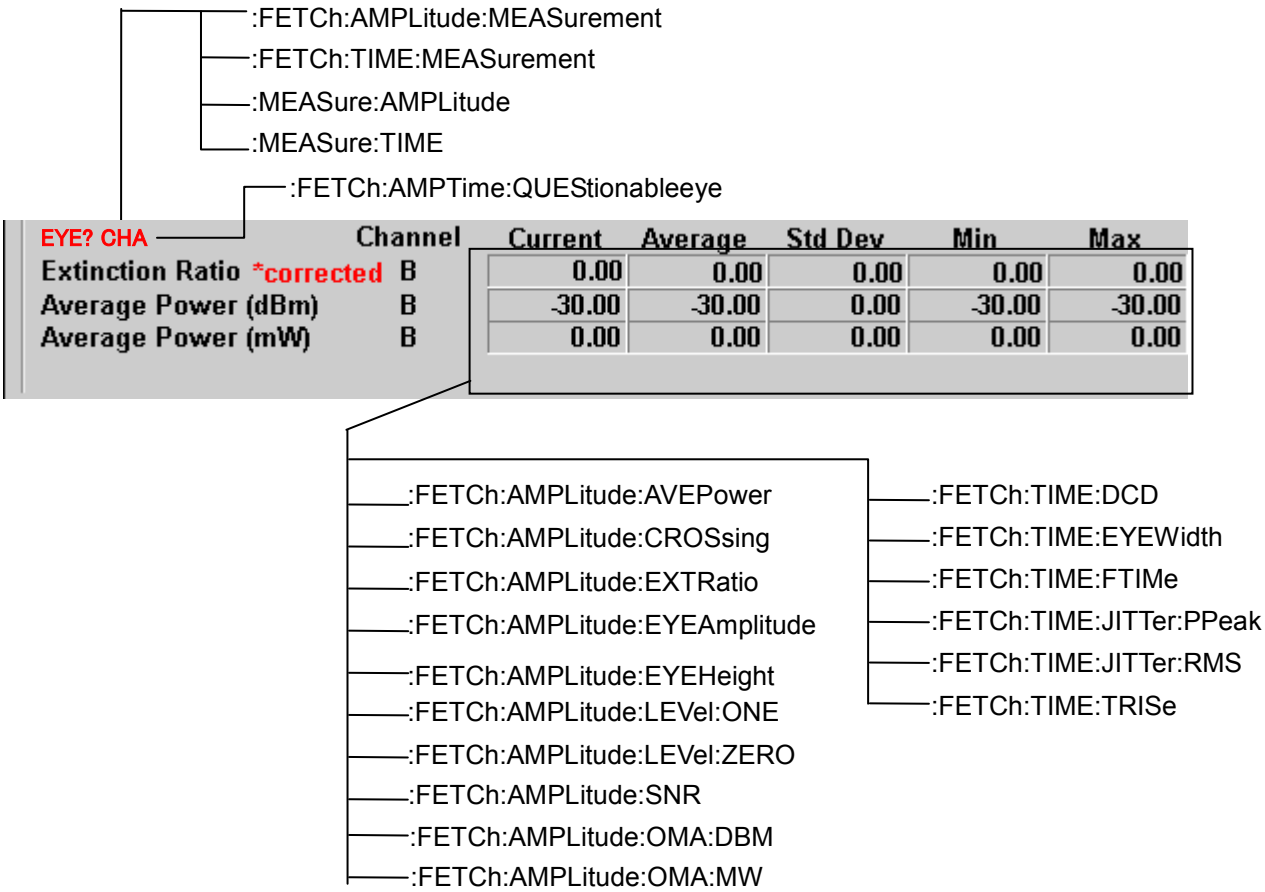


Figure 4.2.5-2 Message Corresponding to Amplitude/Time Measurement Result

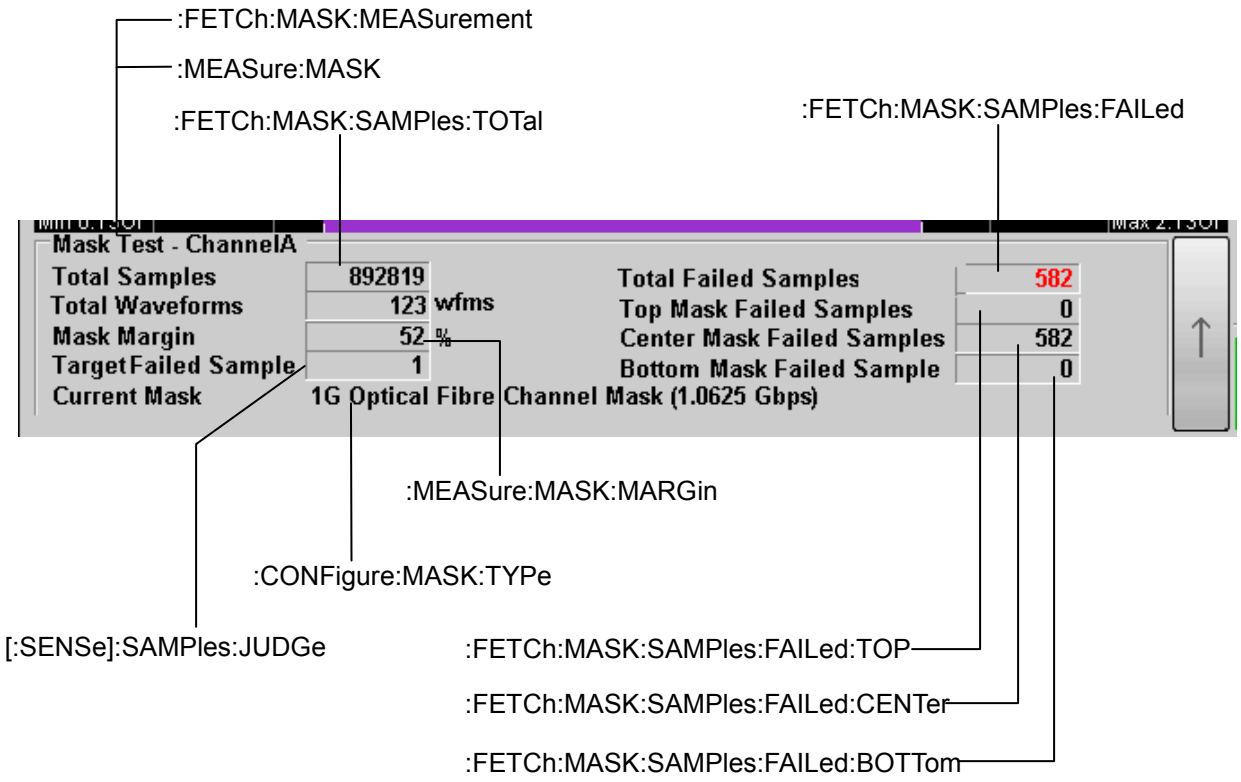


Figure 4.2.5-3 Message Corresponding to Mask Test Measurement Result

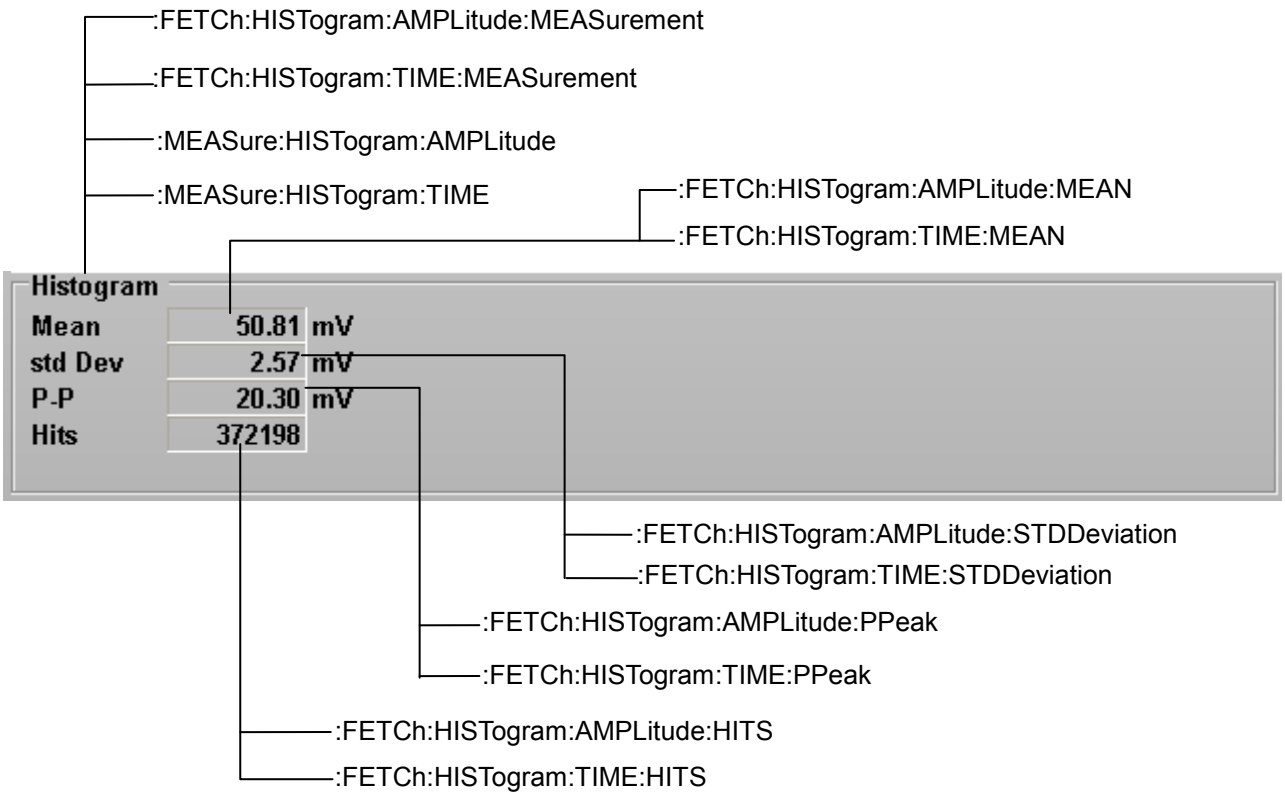


Figure 4.2.5-4 Message Corresponding to Histogram Measurement Result

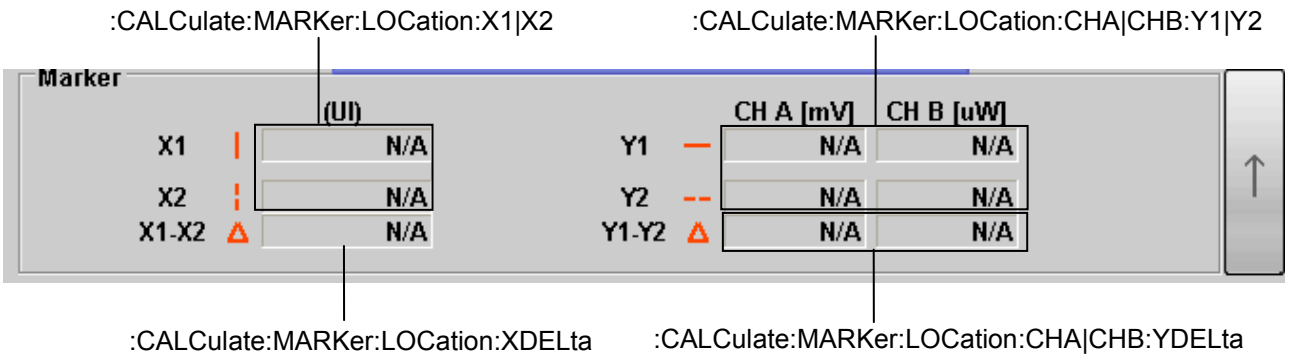


Figure 4.2.5-5 Message Corresponding to Marker DisplayMarker

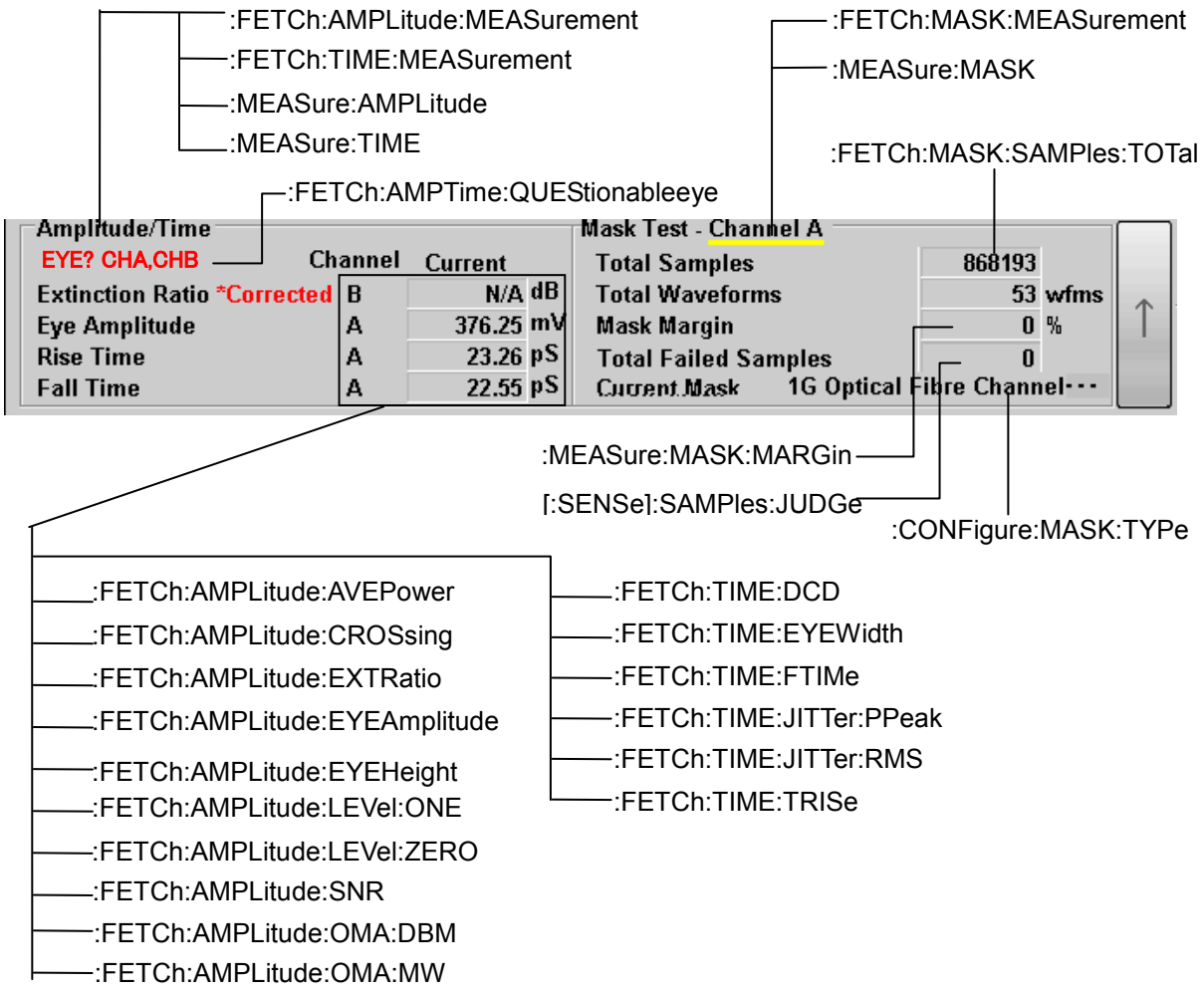


Figure 4.2.5-6 Message Corresponding to Amplitude/Time&Mask Measurement Result

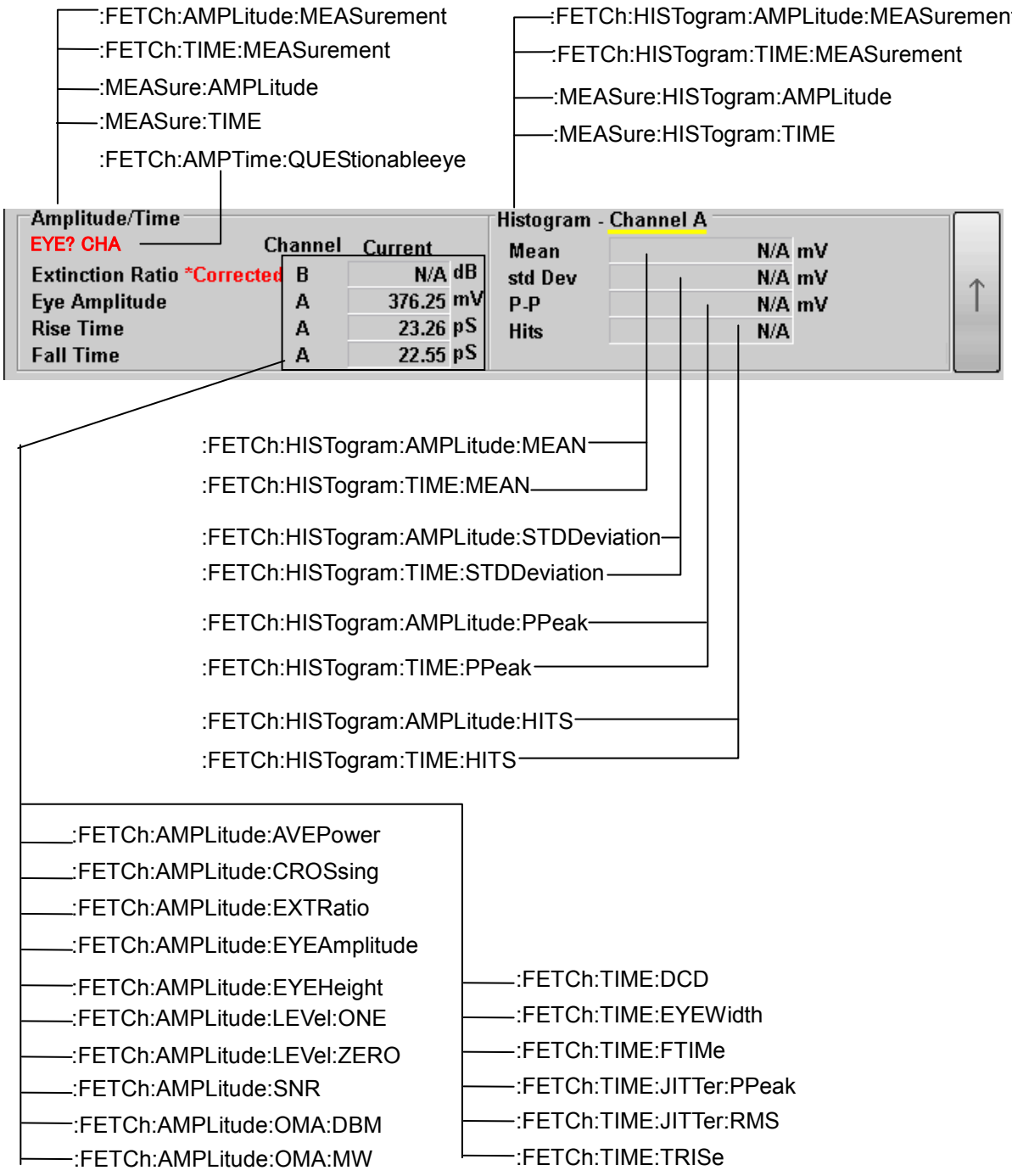


Figure 4.2.5-7 Message Corresponding to Amplitude/Time&Histogram Measurement Result

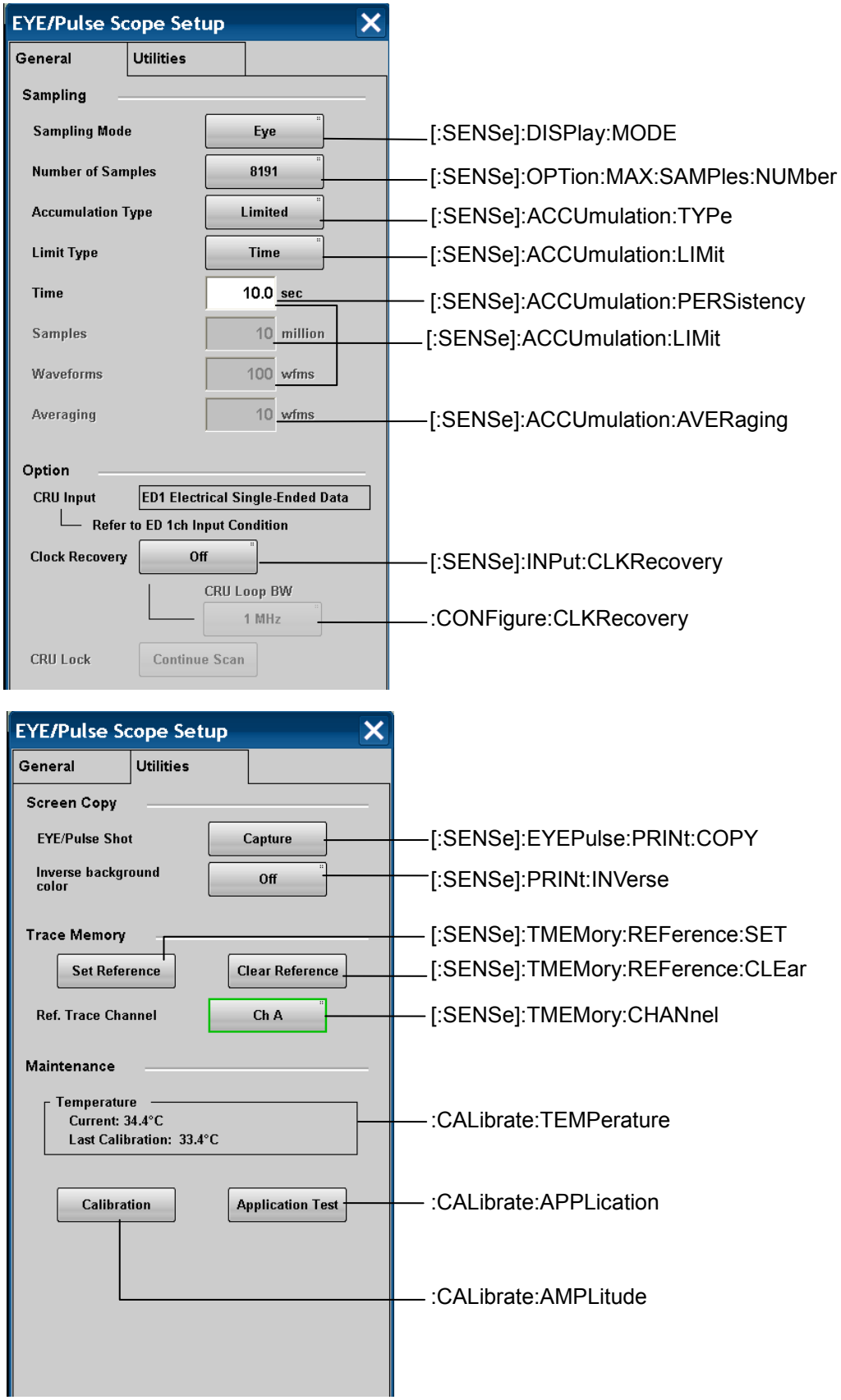


Figure 4.2.5-8 Message Corresponding to Setup DialogSetup

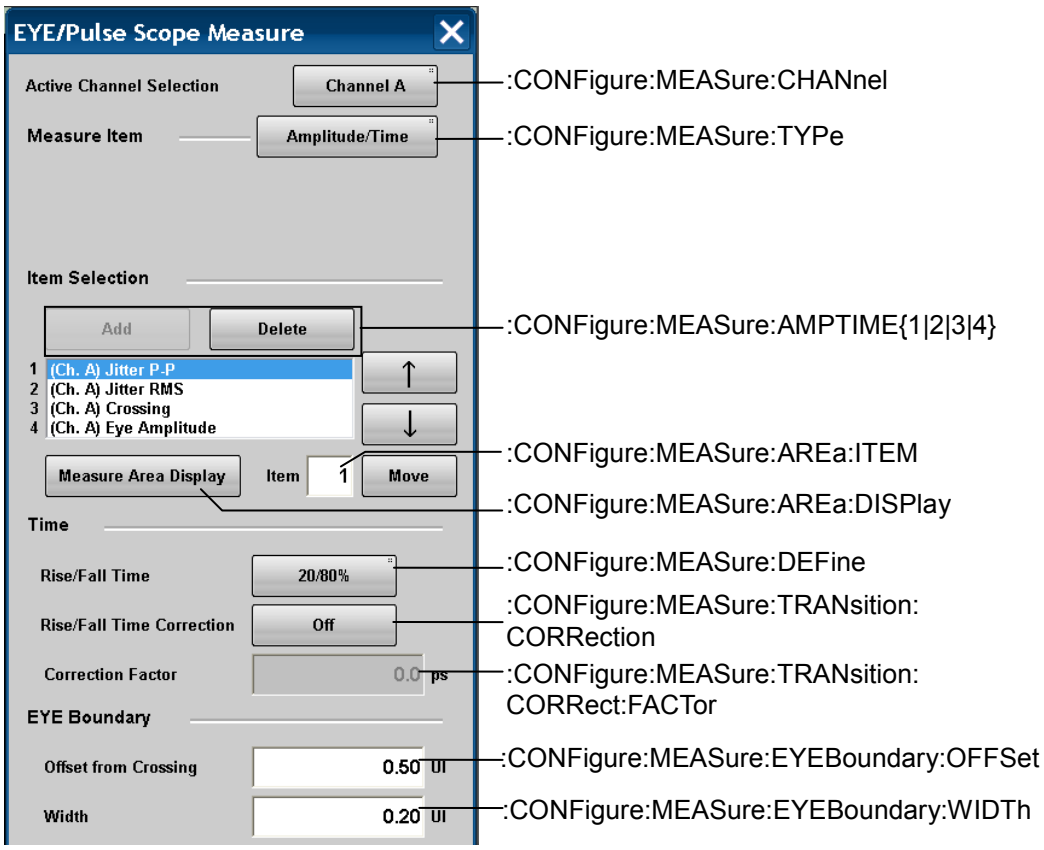


Figure 4.2.5-9 Message Corresponding to Measure Dialog

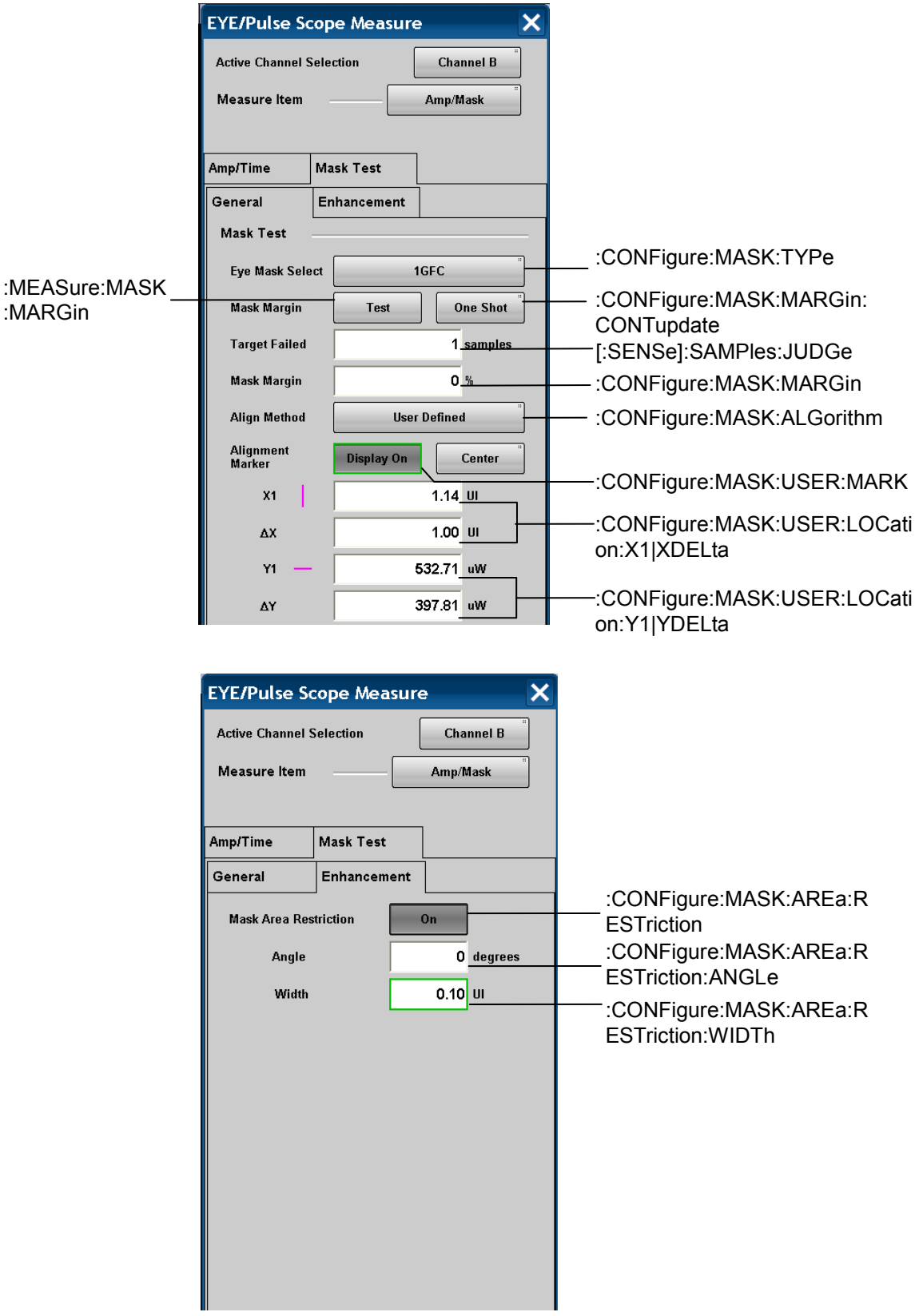


Figure 4.2.5-10 Message Corresponding to Measure Dialog Measure (Mask Test,Amplitude/Time&Mask)

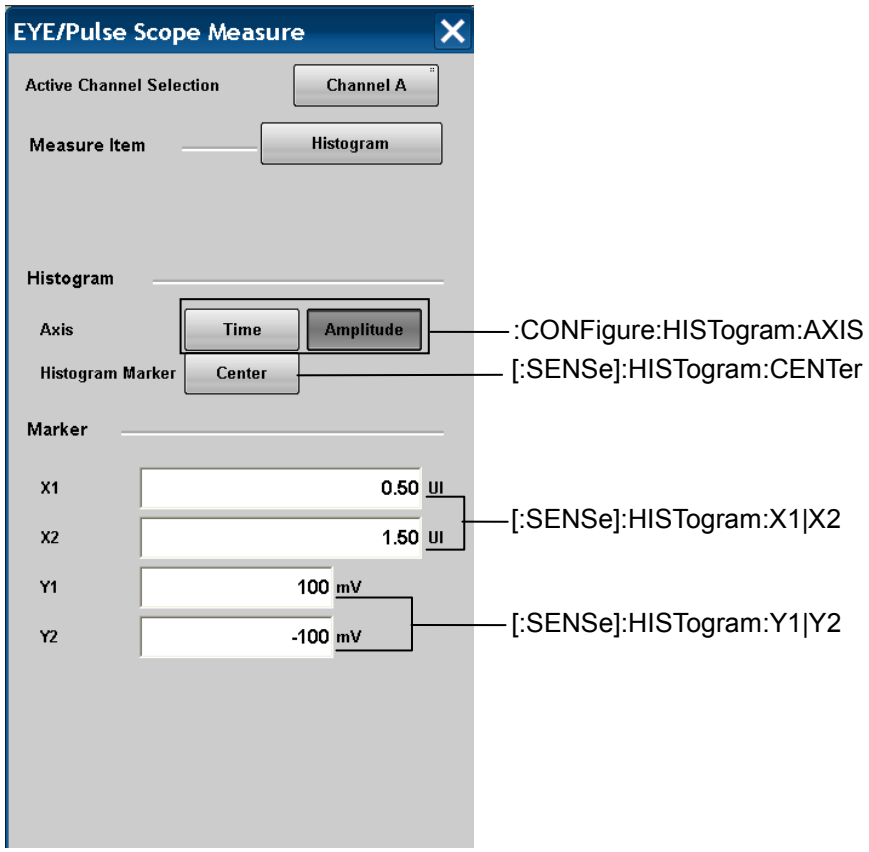


Figure 4.2.5-11 Message Corresponding to Measure Dialog

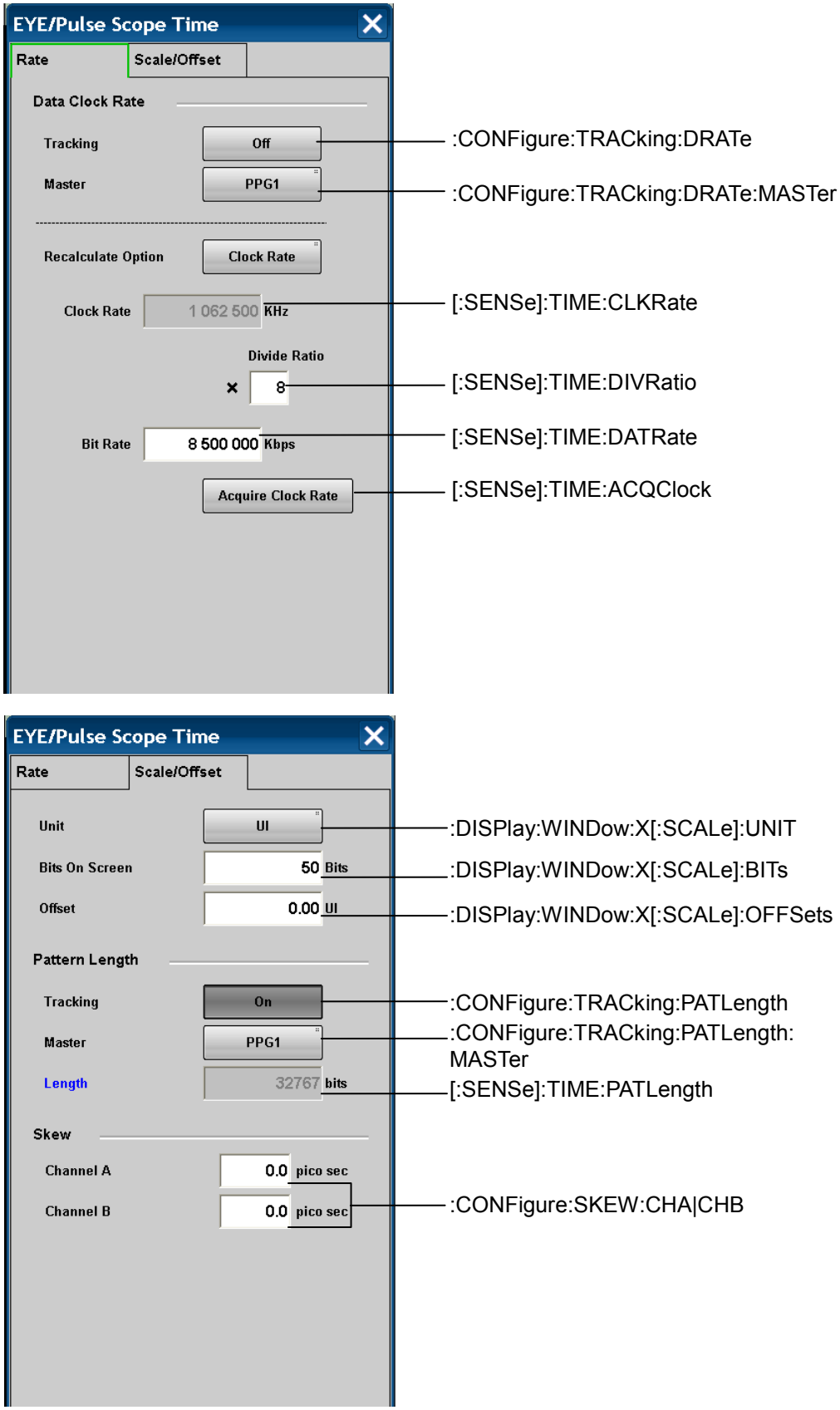


Figure 4.2.5-12 Message Corresponding to Time DialogTime

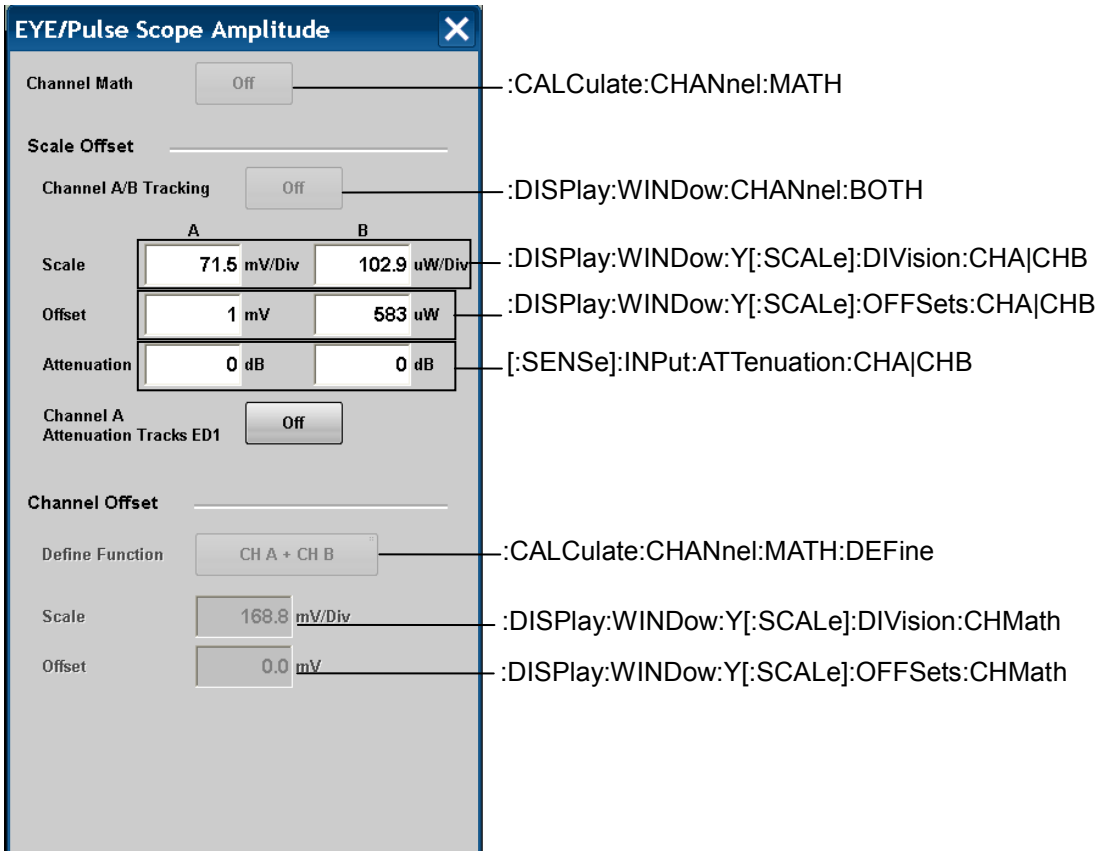


Figure 4.2.5-13 Message Corresponding to Amplitude Dialog

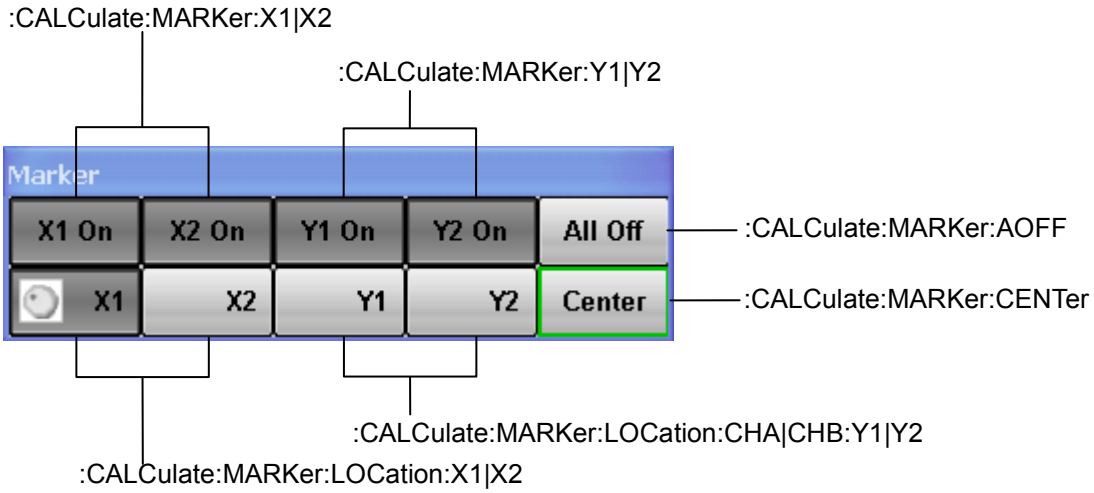


Figure 4.2.5-14 Message Corresponding to Marker Dialog Maker

Figure 4.2.5-15 Message Corresponding to Measure Dialog 2 (Mask Test,Amplitude/Time&Mask)

4.2.6 Panel Keys for Set-up Utility

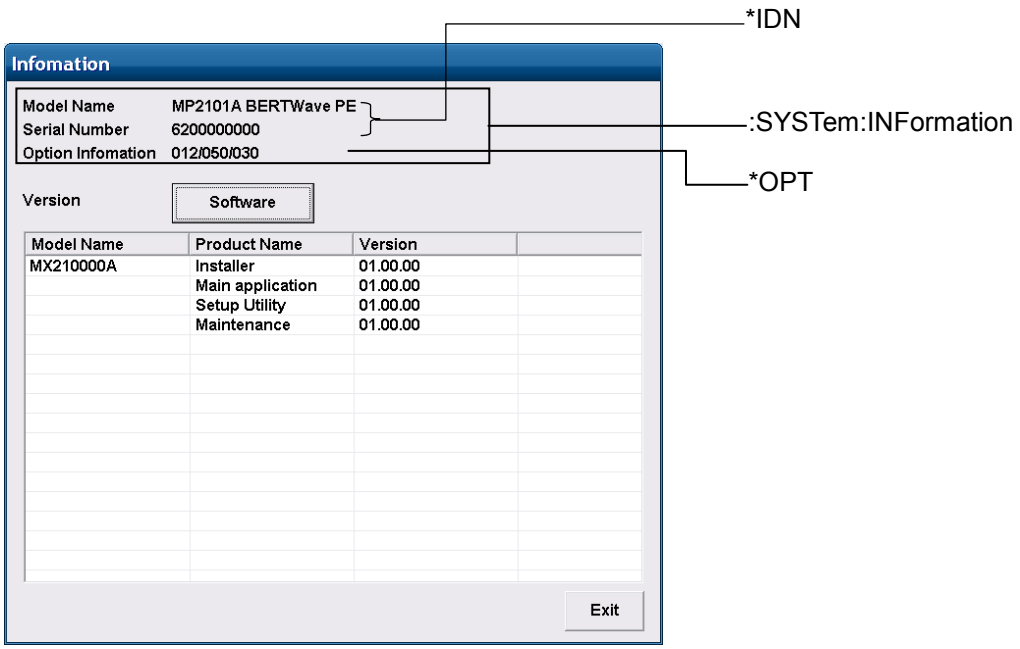


Figure 4.2.6-1 Message Corresponding to Information Dialog

4.2.7 Messages with No Corresponding Panel Operation

Command messages with no corresponding panel operation are listed below.

Query-only messages are described by omitting the question mark.

Messages that are both command and query are described as command-only.

Table 4.2.7-1 Messages with No Corresponding Panel Operation (Common Commands)

Command	Details
*CLS	Clears standard event register and output queue
*ESE	Sets/queries standard event enable register
*ESR	Queries standard event register
*OPC	Sets/queries bit setting and bit 0 for status byte indicating message processing completion
*RST	Initializes MP2100A/MP2101A/MP2102A setting conditions
*SRE	Sets/queries service request enable register
*STB	Queries status byte register
*WAI	Waits previous sent message completion

Table 4.2.7-2 Messages with No Corresponding Panel Operation (SCPI)

Command	Details
:INSTrument:PE1:CONDition	Queries the device dependant condition register for the PPG/ED Ch1
:INSTrument:PE1[:EVENt]	Queries the device dependant event register for PPG/ED Ch1
:INSTrument:PE1:NTRansition	Sets and queries the transition filter (phenomena end) settings for the PPG/ED Ch1.
:INSTrument:PE1:PTRansition	Sets and queries the transition filter (phenomena occurrence) settings for the PPG/ED Ch1.
:INSTrument:PE1:RESet	Initializes the device dependant condition register for the PPG/ED Ch1.
:INSTrument:PE2:CONDition	Queries the device dependant condition register for the PPG/ED Ch2.
:INSTrument:PE1[:EVENt]	Queries the device dependant condition register for the PPG/ED Ch2.
:INSTrument:PE2:NTRansition	Sets and queries the transition filter (phenomena end) settings for the PPG/ED Ch2.
:INSTrument:PE2:PTRansition	Sets and queries the transition filter (phenomena occurrence) settings for the PPG/ED Ch2.
:INSTrument:PE2:RESet	Initializes the device dependant condition register for the PPG/ED Ch2.
:INSTrument:WAV:CONDition	Queries the device dependant condition register on EYE/Pulse Scope.
:INSTrument:WAV[:EVENt]	Queries the device dependant event register on EYE/Pulse Scope.
:INSTrument:WAV:NTRansition	Sets and queries the transition filter (phenomena end) settings on EYE/Pulse Scope.

Table 4.2.7-2 Messages with No Corresponding Panel Operation (SCPI) (Cont'd)

Command	Details
:INSTrument:WAV:PTRansition	Sets and queries the transition filter (phenomena occurrence) settings on EYE/Pulse Scope.
:INSTrument:WAV:RESet	Initializes the device dependant condition register on EYE/Pulse Scope
:INSTrument:XSFP:CONDition	Queries the device dependant condition register for the XFP/SFP+
:INSTrument:XSFP[:EVENt]	Queries the device dependant event register for the XFP/SFP+
:INSTrument:XSFP:NTRansition	Sets and queries the transition filter (phenomena end) settings for the XFP/SFP+
:INSTrument:XSFP:PTRansition	Sets and queries the transition filter (phenomena occurrence) settings for the XFP/SFP+
:INSTrument:XSFP:RESet	Initializes the device dependant condition register for the XFP/SFP+
:OUTPut:CMU:RESolution	Sets and queries bit rate unit
:SENSe:MEASure:EALarm:STOP	Queries the measurement end time of ED.
:SENSe:PARam:AEXECute	Sets and queries the error measurement for the 10 ms cycle
:STATus:OPERation:CONDition	Queries the condition register of the operation status condition register.
:STATus:OPERation:ENABle	Sets and queries the enable register of the operation status register
:STATus:OPERation[:EVENt]	Sets the enable register of the operation status register
:STATus:OPERation:NTRansition	Sets and queries the transition filter (phenomena end) settings of the operation status register.
:STATus:OPERation:PTRansition	Sets and queries the transition filter (phenomena occurrence) settings of the operation status register.
:STATus:PRESet	Initializes the operation status register and device dependant register
:SYSTem:DISPlay:DATA	Queries the screen file.
:SYSTem:DISPlay:RESult	Sets and queries faster access via remote control by not updating the screen display.
:SYSTem:TERMination	Sets and queries the terminator.
:SYSTem:VERSion	Queries SCPI version
:TRACe[:DATA]:CHANnelA|CHANnelB|CHANnels	Queries the waveform data of EYE/Pulse Scope This command is enabled only the query mode for the waveform data.
:TRACe[:DATA]:END	Ends the query mode of waveform data for EYE/Pulse Scope
:TRACe[:DATA]:PREPare	Starts the query mode of waveform data for EYE/Pulse Scope

4.3 Command Tree

These messages have a colon-separated tree format.
The level at the highest part of the tree is called the root. Commands are branched into functional groups. The first group branch is called a subsystem.

The command tree is described by the following rules.

- Query-only messages are described by omitting the question mark.
- Messages that are both command and query are described as command-only.

As an example, the command for displaying the EYE/Pulse Scope marker at the center is as follows:
`:CALCulate:MARKer:CHA:CENTer`

The first colon-separated character string is CALCulate so the message is a CALCulate subsystem message.
Ignoring the subsystem character string, the command tree is described as follows:

```
:MARKer
      :CHA
          :CENTer
```


Common Command

*CLS
*ESE
*ESR
*IDN
*OPC
*OPT
*RST
*SRE
*STB
*TRG
*WAI

CALCulate Subsystem

:CHANnel
:MATH
:DEFine
:DATA
:EALarm
:MONitor
:MARKer
:AOFF
:CENTer
:LOCation
:CHA|CHB
:Y1|Y2
:YDELta
:X1|X2
:XDELta
:X1|X2
:Y1|Y2
:OPTical
:STATus <string>

CALibrate Subsystem

:AMPLitude
:APPLication
:CGain
:OEPower
:JUDGe
:PESPonsivity
:SYSTem
:CGain
:TEMPerature

CONFigure Subsystem

:AREa
:DISPlay
:ITEM
:CLKRecovery
:EXRCorection
:FACTor
:HISTogram
:AXIS

```
:MASK
        :ALGolithm
        :AREa
            :RESTriction
                    :ANGLe
                    :WIDTh
        :MARGin <numeric>
                    :CONTupdate
        :TYPe <integer>
        :USER
            :LOCation
                    :X1|XDELta
                    :Y1|YDELta
            :MARKer

:MEASure
        AMPTIME{1|2|3|4}
        :CHANnel A|B
        :DEFine
        :EYEBoundary
            :OFFSet
            :WIDTh
        :TYPe AMPTIME|HISTogram|MASK|OFF
:SKEW
        :CHA|CHB
:TRACking
        :DRATe
            :MASTer
            :PATLength
         :MASTer
:TRANsition
         :CORRect
            :FACTor
:CORRection
```

DISPlay Subsystem

```
:ACTive 1|2|3|4|5|6|7
:RESult
        :EALarm
            HRESet
            MODe <integer>
:WINDow
```

:CHANnel
:BOTH
:GRAPhics
CLEar
[:SCALe]
:AUTOscale[BOTH | HORIzontal | VERTical]
:X
:[SCALe]
:BITS <integer>
:OFFSets <integer>
:UNIT
:Y
:[SCALe]
: DIVision
:CHA | CHB
:CHMath
:OFFSets
:CHA | CHB
:CHMath

FETCh Subsystem

:AMPLitude
:AVEPower
:CROSsing
:EXTRatio
:EYEAmplitude
:EYEHeight
:LEVel
:ONE
:ZERO
:MEASurement
:OMA
:DBM
:MW
:SNR
:AMPTime
:QUEStionableeye
:HISTogram
:AMPLitude
:HITS
:MEAN
:MEASurement
:PPeak

```
            :STDDeviation
        :TIME
            :HITS
            :MEAN
            :MEASurement
            :PPeak
            :STDDeviation
:MASK
        :MEASurement
        :SAMPles
            :FAILed
                :BOTTom
                :CENTer
                :TOP
            :TOTal
:TIME
        :DCD
        :EYEWidth
        :FTIMe
        :JITTer
            :PPeak
            :RMS
        :MEASurement
        :TRISe
```

INPut Subsystem

```
:BITRate
        :DIVRate
        :STANdard
:DATA
        :ATTFactor
        :INTerface
        :THRehshold
```

INSTrument Subsystem

```
:PE1
        :CONDition
        :[EVENt]
        :NTRansition <numeric>
        :PTRansition <numeric>
        :RESet
:PE2
```

:CONDition
:[EVENt]
:NTRansition <numeric>
:PTRansition <numeric>
:RESet
:WAV
:CONDition
:[EVENt]
:NTRansition <numeric>
:PTRansition <numeric>
:RESet
:XSFP
:CONDition
:[EVENt]
:NTRansition <numeric>
:PTRansition <numeric>
:RESet

MEASure Subsystem

:AMPLitude
:HISTogram
:AMPLitude
:TIME

:MASK
:MARGin
:TIME

MODule Subsystem

:ID 1|2|3|4|5|6|7

OUTPut Subsystem

:BITRate
    :DIVRate <character>
    :OFFSet <numeric>
    :STANdard <string>
:CLOCk
    :FREQuency <numeric>
    :OFFset
        :PPM <numeric>
    :OPERation
:CMU
    :EXTClock
    :FREQuency
    :REFClock
    :RESolution
:DATA
    :AMPLitude
    :ATTFactor
    :OUTPut
    :RELative
:RCLock
    :SELect
:SYNC
    :SOURce

SENSe Subsystem

:ACCUmulation
    :LIMit
    :PERSistency
    :TYPe
:AVERaging
    :WAVforms
:DISPlay
    :MODE EYE|PULSe
:EYEPulse
    :PRINt
        :COPY
:HISTogram
    :CENTer
    :X1|X2
    :Y1|Y2

:INPut
    :ATTenuation
        :CHA|CHB
    :CLKRecovery
    :FILter <integer>
    :WAVLength 1310|1550|850
:MEASure
    :ASTate
    :ASTP
    :ASTRt
    :ELAarm
        :ELAPsed
        :MODE
        :PERiod <integer>,<integer>,<integer>,<integer>
        :STArt
        :STATe
        :STOP
        :TIMed
    :STARt
    :STOP
:MMEMory
    :PATTern
        :RECall <file_name>,<file_type>
:OPTion
    :MAX
        :SAMPles
            :NUMber
:PARam
    :AEXECute
    :TRACking 0|1|OFF|ON
:PATTern
    :DATA
        :LENGth
    :LOGic
    :SYNC
        :ASYNc OFF|ON
        :FPOSition <integer>
        :PSMode OFF|ON
        :THReshold
    :TYPE
:PRINt
    :INVerse
:SAMPles
    :JUDGe

:STATus
:TIME
:ACQClock
:CLKRate
:DATRate
:DIVRatio
:PATLength
:TMEMory
:CHANnel
:REFerence
:CLEar
:SET

SOURce Subsystem

:MMEMory
:PATTern
:RECall <file_name>,<file_type>
:OPTical
:SIGNal
:OUTPut OFF|ON
WLENgth <numeric>
:XFP
REFClock <numeric>
:OUTPut
:ASET
:PATTern
:DATA
:LENGth
:EADDition
:RATE
:SET
:SINGle
:VARiation
:LOGic
:TYPe

STATus Subsystem

:OPERation
:CONDition
:ENABle <numeric>
:[EVENt]

:NTRansition <numeric>
:PTRansition <numeric>
:PREset

SYSTem Subsystem

:BEEPer
:SET OFF|ON
:DATE
:DISPlay
:ALARm
:DATA
:RESult OFF|ON
:ERRor
:HCLear
:HISTory
:INFormation
:ERRor
:MEMory
:INITialize
:MMEMory
:RECall <file_name>,<module>,<data_type>
:STORe <file_name>,<module>,<data_type>
:PRINt
:COPY
:TERMination
:TIME
:VERSion

TRACe Subsystem

:[DATA]
:CHANnelA
:CHANnelB
:CHANnels
:END
:PREPare CHA|CHB|BOTH

4.4 Device Message Details

4.4.1 IEEE488.2 Common Message

*CLS [Clear Status]

Function

1. *CLS common command clears the following registers.
   - Standard event status register
   - Output queue

   Therefore, bits 5 of status byte register became 0.
   The setting value of each enable register does not vary depending on *CLS.
   - Standard event status enable register
   - Service request enable register
   - Operation status register
   - Device dependant status register

2. The *CLS common command clears the status byte register when sending *CLS command before the query after the program message terminator.
   All unread messages in the output queue are cleared at this time.
   The relevant message example indicates below.

```
SENS:BIT 8500000
*CLS ; SENS:BIT?
```

   When receiving SENS:BIT? after *CLS, the status byte register is cleared.

Syntax

*CLS

*ESE [Event Status Enable]

Function

This command sets the standard event status enable register.
The setting of 0 to 255 is equivalent to 8-bit binary.
The standard event status mask bit is set to 0.
The standard event status enable register value is returned for the query.

Syntax

```
*ESE <integer>
*ESE?
```

<integer>= bit0 + bit1 + bit2 + bit3 + bit4 + bit5 + bit6 + bit7

bit7 : 2^7 = 128	Power-on
bit6 : 2^6 = 64	Not used
bit5 : 2^5 = 32	Command error
bit4 : 2^4 = 16	Operation error
bit3 : 2^3 = 8	Dependant device error
bit2 : 2^2 = 4	Not used
bit1 : 2^1 = 2	Not used
bit0 : 2^0 = 1	Completion of operation

Range 0 to 255

Response Data

<integer>:Total bits for standard event enable register is 0 to 255
For the correspondence between register bits and decimal number, refer to Table 2.6.1-1.

Example of Use

The following example shows how to mask bits 7 to 4 and permit bits 3 to 0.

```
*ESE 15
*ESE?
>15
```

*ESR [Standard Event Status Register]

Function

This command returns the standard event status register value.
The standard event status register value is cleared after readout.
This value is the logical product of the 8 bits set by *ESE.

Syntax

```
*ESR?
```

Response Data

Total bits for standard event enable register is 0 to 255
For the correspondence between register bits and decimal number, refer to Table 2.6.1-1.

Example of Use

The following example queries the value of the standard event status register. The data is the value when an execution error or command error occurs.

```
*ESR?
>48
```

*IDN [Identification]

Function

This command queries product supplier name, model name, serial number, and installer version.

Syntax

```
*IDN?
```

Response Data

<character>,<character>, <character>, <character>

Example of Use

```
*IDN?
>Anritsu,MP2100A,6200123456,3.01.00
```

*OPC [Operation Complete]

Function

If a *OPC command is received, the operation completion bit (bit 0) is set to 1 once all active processes are complete.
If a *OPC? query is received, 1 is returned once all active processes are complete.
The wait for operation completion set by *OPC/*OPC? is disabled after the following events:

- Power ON
- Reception of DCL or SCL on the IEEE488.1 interface
- Reception of the `*CLS` command
- Reception of the `*RST` command
- Completion of all active processing

Note:

This instrument cannot process other messages while processing a message. As a result, processing of the *OPC? query is suspended during execution of a previously sent message. Since *OPC? is processed after processing of the previously sent message is completed, the response data is always 1.

Syntax

```
*OPC
*OPC?
```

Response Data

0|1

Example of Use

```
*OPC?
>1
```

*OPT [Option Identification Query]

Function

This command queries what options are installed.

Syntax

```
*OPT?
```

Response Data

<character>[,<character>][,<character>][,<character>]…

<character> is the following character strings indicating option. Specify the number of the option and separate the options with commas (,).

Table 4.4.1-1 Character Strings and Option Name

Character Strings	Option Name
OPT001	Dual Electrical Receiver
OPT003	Optical/Electrical Receiver
OPT005	Extended PPG/ED Channel
OPT007	1ch Electrical BERT and Optical/Single-ended Electrical Scope
OPT011	1CH BERT
OPT012	2CH BERT
OPT021	Dual Electrical Receiver
OPT023	Optical/Electrical Receiver
OPT030	GPIB
OPT050	XFP Slot
OPT051	SFP+ Slot
OPT055	CRU for Scope
OPT056	Low Pass Filter bank (8.5G/10G/10.7G)
OPT057	Low Pass Filter bank (2G/4G/8.5G/10G)
OPT061	1 High Bit Rate Filter
OPT062	2 High Bit Rate Filter Bank
OPT063	3 to 4 High Bit Rate Filter Bank
OPT064	1 to 2 Low Bit Rate Filter Bank
OPT065	3 to 4 Low Bit Rate Filter Bank
OPT066	1 High Bit Rate/1 to 2 Low Bit Rate Filter Bank
OPT067	1 to 2 High Bit Rate/3 to 4 Low Bit Rate Filter Bank
OPT068	2 to 3 High Bit Rate/1 to 2 Low Bit Rate Filter Bank
OPT069	3 High Bit Rate/3 Low Bit Rate Filter Bank

Table 4.4.1-1 Character Strings and Option Name (Cont'd)

Character Strings	Option Name
OPT070	LPF for 156M (L)
OPT071	LPF for 622M (L)
OPT072	LPF for 1.0G (L)
OPT073	LPF for 1.2G (L)
OPT074	LPF for 2.1G (H)
OPT075	LPF for 2.5G (H)
OPT076	LPF for 2.6G (H)
OPT077	LPF for 3.1G (H)
OPT078	LPF for 4.2G (H)
OPT079	LPF for 5.0G (H)
OPT080	LPF for 6.2G (H)
OPT081	LPF for 8.5G (H)
OPT082	LPF for 9.9G to 10.3G (H)
OPT083	LPF for 10.5G to 11.3G (H)
OPT084	LPF for 156M (L)
OPT085	LPF for 622M (L)
OPT086	LPF for Multi 10G (H)
OPT090	Bit rate Extension for PPG/ED
OPT130	GPIB

The option number of the optical connector is not output.

Example of Use

```
*OPT?
>OPT001,OPT030,OPT050
```

*RST [Reset]

Function

This command resets the following items to the factory-default settings.

Item	Settings
System Alarm	The system alarm lamp is turned off.
Output	The output of the pulse pattern generator is set to OFF.
Measure	The error measurement using the ED and the EYE/Pulse Scope data collection are stopped. The data during measurement is deleted.
Optical Output	The XFP/SFP+ optical output is set to OFF.

Syntax

```
*RST?
```

*SRE [Service Request Enable]

Function

This command sets the service request enable register.
The setting values, 0 to 255, are equivalent to 8-bit binary.
The bit masking the status byte register is set to 0.
The service request enable register value is returned for the query.

Syntax

```
*SRE <integer>
*SRE?
```

<integer>= bit0 + bit1 + bit2 + bit3 + bit4 + bit5 + bit6 + bit7

bit7 : 2^7 = 128	Operation status register
bit6 : 2^6 = 64	Always 0
bit5 : 2^5 = 32	Standard event status register
bit4 : 2^4 = 16	MAV
bit3 : 2^3 = 8	Not used
bit2 : 2^2 = 4	System error
bit1 : 2^1 = 2	Not used
bit0 : 2^0 = 1	Not used

Range 0 to 255

Example of Use

The following example shows how to mask bits 7,6,3,1 and 0 and permit bits 5,4, and 2.

```
*SRE 52
*SRE?
>52
```

*STB [Status Byte]

Function

This command reads the status byte register.

Syntax

```
*STB?
```

Response Data

<integer>= bit0 + bit1 + bit2 + bit3 + bit4 + bit5 + bit6 + bit7

Range 0 to 255

For each bit displayed information, refer to the explanation of *SRE.

*TRG [Trigger]

Function

This command starts the measurement of all modules (ED Channel 1 and 2, EYE/Pulse Scope Channel A and B).
This is the same processing as sending:SENSe:MEASure:ASTRt.

Syntax

```
*TRG
```

*WAI [Wait to Continue]

Function

This command holds execution of the next message until processing of the message sent before *WAI is completed.

Note:

This instrument cannot process other messages while processing a message. Since the message is processed after processing of the previously sent message is completed, *WAI is not required to use.

Syntax

```
*WAI
```

4.4.2 Device Dependant Command

:CALCulate:CHANnel:MATH

Function

This command sets and queries the calculation between channels of EYE/Pulse Scope.

Syntax

```
:CALCulate:CHANnel:MATH 0|1
:CALCulate:CHANnel:MATH?
```

0: Does not calculate the waveform of channel A and channel B
1: Calculates the waveform of channel A and channel B.

Response Data

1|0

Example of Use

```
:CALC:CHAN:MATH 1
```

:CALCulate:CHANnel:MATH:DEFine

Function

This sets and queries the calculation formula between channels of EYE/Pulse Scope.

Syntax

:CALCulate:CHANnel:MATH:DEFine 0|1|2
:CALCulate:CHANnel:MATH:DEFine?

0: Channel A + Channel B
1: Channel A –Channel B
2: Channel B–Channel A

Response Data

0|1|2

Example of Use

```
:CALC:CHAN:MATH:DEF?
>1
```

:CALCulate:DATA:EALarm

Function

This command queries the measurement results of the error detector (ED).

Syntax

```
CALCulate:DATA:EALarm? <string>

<string>="{CURRent|LAST}:{AINTerval:{CRUNlock|PSLoss}|
          CC:TOTal|{EC|ER}:{INSertion|OMISsion|TOTal}|
          FREQuency}"
```

The details specified by the character strings used in the parameter are as follows.

CURRent	Values displayed on the current screen
LAST	Setting value in the time set at Gating Time , when Gating Type is Single or Repeat
AINTerval:CRUNlock	Clock detection
AINTerval:PSLoss	Pattern sync
CC:TOTal	Clock count
EC:INSertion	Error count (insertion)
EC:OMISsion	Error count (omission)
EC:TOTal	Error count (total sum of insertion and omission)
ER:INSertion	Bit error rate (insertion)
ER:OMISsion	Bit error rate (omission)
ER:TOTal	Bit error rate (total sum of insertion and omission)
FREQuency	Frequency (Hz)

Response Data

<string>

Used character for query	Response type
AINTerval:CRUNlock, AINTerval:PSLoss	Form1 integer type
CC:TOTal	Form1 integer type
EC:INSertion,EC:OMISsion,EC:TOTal	Form1 integer type
ER:INSertion,ER:OMISsion,ER:TOTal	Form2 decimal point type
FREQuency	Form3 frequency type

For the response format, refer to :CALCulate:DATA:EALarm in Table 4.4.2-1.

Example of Use

To query the bit error rate displayed on the screen:

```
:CALC:DATA:EAL? "CURR:ER:TOT"
> "0.0000E−16"
```

Table 4.4.2-1 Response Format

Form	Format	Explanation
Form1 integer type	"XXXXXXX"	For 0 to 9999999
	"X.XXXXEXX"	For 1.0000E07 to 9.9999E17
	"-------"	When no data corresponds to the query
Form2 decimal point type	"X.XXXXE−XX"	For 0.0001E−18 to 1.0000E00
	"-------"	When no data corresponds to the query
Form3 frequency type	"XXXXXXXXX"	For 0 to MAX(Hz)
	"-------"	When no data corresponds to the query

:CALCulate:DATA:MONitor

Function

This command queries the following alarm status found from the measurement results of the ED.

- Error: Bit error occurrence alarm
- CR Unlock: Signal detection alarm
- SYNC Loss: Pattern sync loss alarm

Syntax

```
:CALCulate:DATA:MONitor? "BIT:TOTal"|"CRUNlock"|"PSLoss"
```

"BIT:TOTal"	Error: Bit error occurrence
"CRUNlock"	CR Unlock: Signal detection fail alarm
"PSLoss"	SYNC Loss: Pattern sync loss

Response Data

"------"|"Not Occur"|"Occur"

"------"	No data corresponding to query
"Not Occur"	No alarm occurs
"Occur"	Alarm occurs (Alarm indicates red)

Example of Use

To query the Error alarm status:

```
:CALC:DATA:MON? "BIT:TOT"
>"Occur"
```

:CALCulate:MARKer:AOFF

Function

This command deletes the marker displayed in the EYE/Pulse Scope screen.

Syntax

```
:CALCulate:MARKer:AOFF
```

Example of Use

```
:CALC:MARK:AOFF
```

:CALCulate:MARKer:CENTer

Function

This command displays all markers of EYE/Pulse Scope at the center of the screen.

Syntax

```
:CALCulate:MARKer:CENTer
```

Example of Use

```
:CALC:MARK:CENT
```

:CALCulate:MARKer:LOCation:CHA|CHB:Y1|Y2

Function

This command sets the marker value of Y1 or Y2 of EYE/Pulse Scope. If the marker is not displayed, the marker is displayed.
This command queries the present value of marker Y1 or Y2. The response data when the marker is not displayed is "N/A. "

Syntax

```
:CALCulate:MARKer:LOCation:CHA|CHB:Y1|Y2 <numeric>
:CALCulate:MARKer:LOCation:CHA|CHB:Y1|Y2?
```

<numeric>: Amplitude displaying marker
When inputting electricity:mV, When inputting optical light:μW
The settable range is as follows.

Unit	Range
mV	–1000~1000
μW	Divided values: divides –1000~1000 by O/E conversion gain.

Response Data

<numeric>|"Marker Off"

Example of Use

To query the value of the marker Y2 for the Channel A waveform.

```
:CALC:MARK:LOC:CHA:Y2?
>55.35
```

:CALCulate:MARKer:LOCation:CHA|CHB:YDELta

Function

This command queries the marker difference between Y1 and Y2. The response data when both or either of markers are not displayed is "N/A. "

Syntax

```
:CALCulate:MARKer:LOCation:CHA|CHB:YDELta?
```

Response Data

<numeric>|"Marker Off"
<numeric>: Amplitude displaying marker
When inputting electricity:mV, When inputting optical light:□W

Example of Use

To query the marker difference between Y1 and Y2 for Channel B waveform.

:CALC:MARK:LOC:CHB:YDEL?

```
>15.3
```

:CALCulate:MARKer:LOCation:X1|X2

Function

This command sets the X1 or X2 marker value of EYE/Pulse Scope. If the marker is not displayed, the marker is displayed.
This command queries the X1 or X2 current marker value. The response data when the marker is not displayed is "N/A. "

Syntax

```
:CALCulate:MARKer:LOCation:X1|X2 <numeric>
:CALCulate:MARKer:LOCation:X1|X2?
```

<numeric>:Time displaying marker (Unit: UI or ps)
The setting range is as follows.

Unit	Range
ps	0~4294967295
UI	0~4294967

Response Data

<numeric>|"Marker Off"

Example of Use

To display the marker X2 to the 1.5 UI position

```
:CALC:MARK:LOC:X2 1.5
```

:CALCulate:MARKer:LOCation:XDELta

Function

This command queries the time marker difference between X1 and X2. The response data when both or either of markers are not displayed is "N/A. "

Syntax

```
:CALCulate:MARKer:LOCation:XDELta?
```

Response Data

<numeric>|"Marker Off"
<numeric>: Time displaying marker (Unit: UI or ps)

Example of Use

```
:CALC:MARK:LOC:XDEL?
>152.33
```

:CALCulate:MARKer:X1|X2

Function

This command sets and queries the display of time marker X1 or X2 of EYE/Pulse Scope.

Syntax

```
:CALCulate:MARKer:X1|X2 ON|OFF|1|0
:CALCulate:MARKer:X1|X2?
```

ON|1: Displays the marker
OFF|0: Deletes the marker display

Response Data

1|0

Example of Use

To query whether the marker X1 is displayed

```
:CALC:MARK:X1?
>1
```

:CALCulate:MARKer:Y1|Y2

Function

This command sets and queries the display of amplitude marker Y1 or Y2 of EYE/Pulse Scope.

Syntax

```
:CALCulate:MARKer:Y1|Y2 ON|OFF|1|0
:CALCulate:MARKer:Y1|Y2?
```

Parameter

ON|1: Displays the marker
OFF|0: Deletes the marker display

Response Data

1|0

Example of Use

To query whether the marker Y2 is displayed

```
:CALC:MARK:Y2?
>1
```

:CALCulate:OPTical:STATus

Function

This command queries the status of the XFP/SFP+ slot optical transceiver.

Syntax

```
:CALCulate:OPTical:STATus? "LOS"|"READy"
```

READy: Optical transceiver installed
LOS: Optical input detected

Response Data

"----"|"None"| "Occur"

"----": Unknown
"None": No LOS occurs
Or, the optical transceiver not installed
"Occur":LOS occurs
Or, the optical transceiver installed

Example of Use

```
:CALC:OPT:STAT? LOS
>NONE
```

:CALibrate:AMPLitude

Function

This command initiates an amplitude calibration for EYE/Pulse Scope Channel A and B. The calibration could take more than 30 seconds. When using the Ethernet, the calibration result is returned after the calibration. When using the GPIB, the calibration result is stored in the output queue after the calibration.

Syntax

```
:CALibrate:AMPLitude
```

:CALibrate:APPLication

Function

This command starts the self-diagnosis of the EYE/Pulse Scope.
Also, this command queries the self-diagnosis result of the EYE/Pulse Scope.

Syntax

```
:CALibrate:APPLication
:CALibrate:APPLication?
```

Response Data

```
"Self Test Passed!"
```

" Self Test Passed!": The self-diagnosis ended normally.
When the self-diagnosis error occurs, the response message is not returned.

Example of Use

```
:CAL:APPL?
>"Self Test Passed!"
```

:CALibrate:CGain

Function

This command sets and queries the O/E converter gain (V/W).

Syntax

```
:CALibrate:CGain <numeric>
:CALibrate:CGain?
```

< numeric >: Conversion Gain Range1~9999 (V/W)

Response Data

< integer >:1 to 9999

Example of Use

```
:CAL:CG?
>320
```

:CALibrate:OEPower

Function

This command calibrates the O/E converter.

Syntax

```
:CALibrate:OEPower
```

:CALibrate:OEPower:JUDGe

Function

This command judges if the O/E converter can be calibrated.
It queries the judgment result about the O/E converter calibration.

Syntax

```
:CALibrate:OEPower:JUDGe
:CALibrate:OEPower:JUDGe?
```

Response Data

Pass | Fail

Pass: The O/E converter can be calibrated. (Optical input level ≤ –30 dBm)
Fail: The O/E converter cannot be calibrated. (Optical input level > –30 dBm)

Example of Use

```
:CAL:OEP:JUDG
:CAL:OEP:JUDG?
>PASS
```

:CALibrate:RESPonsivity

Function

This command sets the responsivity (A/W) for photodiode used by the O/E converter. The value is used for average optical power measurement for the amplitude measurement.
The query returns present value for responsivity (A/W).

Syntax

```
:CALibrate:RESPonsivity <numeric>
:CALibrate:RESPonsivity?
```

<numeric>:Responsivity Range 0.001 to 9999

Response Data

<numeric>:0.001 to 9999

Example of Use

```
:CAL:RESP?
>0.853
```

:CALibrate:SYSTem:CGain

Function

This command sets and queries the system conversion gain (V/W) for the O/E converter.

Syntax

```
:CALibrate:SYSTem:CGain <numeric>
:CALibrate:SYSTem:CGain?
```

<numeric>: System Conversion Gain Range 1~9999 (V/W)

Response Data

<integer>:1 to 9999

Example of Use

```
:CAL:SYST:CG?
>160
```

:CALibrate:TEMPerature

Function

This command queries the current temperature and temperature during the calibration on the EYE/Pulse Scope module.

Syntax

```
:CALibrate:TEMPerature?
```

Response Data

<numeric>,<numeric>

The first parameter: Current temperature (°C)
The second parameter: Temperature during calibration (°C)

Example of Use

```
:CAL:TEMP?
>39.6,24.4
```

:CONFigure:CLKRecovery

Function

This command sets the bandwidth of the clock recovery unit for EYE/Pulse Scope.
The query responses the bandwidth of the clock recovery unit for EYE/Pulse Scope.

Syntax

```
:CONFigure:CLKRecovery 1MHz|2MHz|4MHz|8MHz
:CONFigure:CLKRecovery?
```

1MHz	Sets the bandwidth to 1 MHz
2MHz	Sets the bandwidth to 2 MHz
4MHz	Sets the bandwidth to 4 MHz
8MHz	Sets the bandwidth to 8 MHz

Response Data

1MHz|2MHz|4MHz|8MHz

Example of Use

To set the bandwidth of clock recovery unit to 4 MHz

```
:CONF:CLKR 4MHz
```

To query the bandwidth of clock recovery unit

```
:CONF:CLKR?
>4MHz
```

:CONFigure:EXRCorrection

Function

This command sets and queries the execution of the extinction ratio correction for the O/E converter.

Syntax

```
:CONFigure:EXRCorrection 0|1
:CONFigure:EXRCorrection?
```

0 Set extinction ratio correction to OFF.
1 Set extinction ratio correction to ON.

Response Data

0|1

Example of Use

To set extinction ratio correction to ON:

```
:CONF:EXRC 1
```

To query execution settings for extinction ratio correction:

```
:CONF:EXRC?
>1
```

:CONFigure:EXRCorrection:FACTor

Function

This command sets and queries the factor of the extinction ratio correction for the O/E converter.

Syntax

```
:CONFigure:EXRCorrection:FACTor <numeric>
:CONFigure:EXRCorrection:FACTor?
```

<numeric>: Extinction Ratio Correction Factor
Range –9.99~9.99 (%), 0.01 step

Response Data

<numeric>

Example of Use

To set extinction ratio correction factor to 1.2%

```
:CONF:EXRC:FACT 1.20
```

To query execution settings for extinction ratio correction factor:

```
:CONF:EXRC:FACT?
>1.20
```

:CONFigure:HISTogram:AXIS

Function

This command sets the histogram axis to the time or amplitude width. The query responses the currently selecting histogram axis.

Syntax

```
:CONFigure:HISTogram:AXIS TIME|AMPLitude
:CONFigure:HISTogram:AXIS?
```

TIME: Sets time
AMPLitude: Sets amplitude

Response Data

TIME|AMPLitude

:CONFigure:MASK:ALGorithm

Function

This command sets and queries the mask position adjusting method of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:ALGorithm 0|2
:CONFigure:MASK:ALGorithm?
```

0 Adjusts the mask position by detecting the intersection position at 0 level and 1 level
2 Adjusts mask position by the user's operation

Response Data

0|2

Example of Use

```
:CONF:MASK:ALG 2

:CONF:MASK:ALG?
>2
```

:CONFigure:MASK:AREa:RESTriction

Function

This command sets and queries the mask area limitation of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:AREa:RESTriction 0|1
:CONFigure:MASK:AREa:RESTriction?
```

0 Releases the mask area limitation.
1 Enables the mask area limitation.

Response Data

0|1

Example of Use

```
:CONF:MASK:ARE:REST 1

:CONF:MASK:ARE:REST?
>1
```

:CONFigure:MASK:AREa:RESTriction:ANGLe

Function

This command sets and queries the angle limiting the mask area of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:AREa:RESTriction:ANGLe <integer>
:CONFigure:MASK:AREa:RESTriction:ANGLe?
```

<integer>:–90~90 Angle limiting mask area

Response Data

<integer>

Example of Use

```
:CONF:MASK:ARE:REST:ANGL -30

:CONF:MASK:ARE:REST:ANGL?
>-30
```

:CONFigure:MASK:AREa:RESTriction:WIDTh

Function

This command sets and queries the width restricting the mask area of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:AREa:RESTriction:WIDTh <numeric>
:CONFigure:MASK:AREa:RESTriction:WIDTh?
```

<numeric>:0.01~1.00 Width restricting mask area (UI)

Response Data

<numeric>

Example of Use

```
:CONF:MASK:ARE:REST:WIDT 0.15

:CONF:MASK:ARE:REST:WIDT?
>0.15
```

:CONFigure:MASK:MARGin

Function

This command sets and queries the mask margin for the test mask of the EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:MARGin <integer>
:CONFigure:MASK:MARGin?
```

<integer>: Mask Margin Range –100 to 100 (%)

Response Data

<integer>

Example of Use

To set mask margin to 20 %:

```
:CONF:MASK:MARG 20
```

To query mask margin:

```
:CONF:MASK:MARG?
>20
```

:CONFigure:MASK:MARGin:CONTupdate

Function

This command sets and queries the mask margin updating method for the mask test of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:MARGin:CONTupdate 0|1
:CONFigure:MASK:MARGin:CONTupdate?
```

0 Updates the mask margin only once.
1 Updates the mask margin whenever measuring

Response Data

0|1

Example of Use

```
:CONF:MASK:MARG:CONT?
>0
```

:CONFigure:MASK:TYPe

Function

This command selects the mask for using the mask test of the EYE/Pulse Scope.
The query returns the index of the currently selecting mask.

Syntax

```
:CONFigure:MASK:TYPe <integer>[,<file_name>]
:CONFigure:MASK:TYPe?
```

<integer>	Mask
–1	User Defined
0	1GFC
1	2GFC
2	4GFC
3	8GFC
4	8GFC_Elect_Rx
5	8GFC_Elect_Tx
6	10GFC
7	10GbE FEC
8	1GbE
9	2GbE
10	10GbE WAN
11	10GbE LAN/PHY
12	10GFC FEC
13	OC48/STM16
14	OTU-1
15	OC192/STM64
16	OC192/STM64 FEC(G.975)
17	OTU-2 1310 nm
18	OTU-2 1550 nm
19	OTU-2 1550 nm Expand
20	OTU-2 Amplified

If –1 is set to <integer>, the file name is set to <file_name>.
If 0 to 20 is set to <integer>, <file_name> can be omitted. If <file_name> is not omitted, setting "" is required.

Response Data

<integer>,<file_name>

Example of Use

To select Mask other than User Mask

```
:CONF:MASK:TYP?
> 8,""
```

To select “Test.txt” mask at User Mask

```
:CONF:MASK:TYP?
> -1,"Test.txt"
```

To set “test.txt“ file as the user mask file

```
:CONFigure:MASK:TYPe -1, "test.txt"
```

:CONFigure:MASK:UPDate

Function

This command updates the position of the mask of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:UPDate
```

:CONFigure:MASK:USER:LOCation:X1|XDELta

Function

This command sets and queries the position for the horizontal direction of the mask of EYE/Pulse Scope in UI unit.

Syntax

```
:CONFigure:MASK:USER:LOCation:X1|XDELta <numeric>
:CONFigure:MASK:USER:LOCation:X1|XDELta?
```

X1	Position of user adjustment marker X1
XDELta	Interval of user adjustment marker X1 and X2
<numeric>	Position in horizontal direction of mask (UI) When XDELta is selected, the positive value is set.

Response Data

<numeric>

Example of Use

To set the user adjustment marker X1 to 0.25UI and the user adjustment marker X2 to 1 .25UI, respectively

```
:CONF:MASK:USER:LOC:X1 0.25
:CONF:MASK:USER:LOC:XDEL 1
```

:CONFigure:MASK:USER:LOCation:Y1|YDELta

Function

This command sets and queries the position for the vertical direction of the mask of EYE/Pulse Scope in mV unit.

Syntax

```
:CONFigure:MASK:USER:LOCation:Y1|YDELta <numeric>
:CONFigure:MASK:USER:LOCation:Y1|YDELta?
```

Y1	Position of user adjustment marker Y1
YDELta	Interval of user adjustment marker Y1 and Y2
<numeric>	Position in vertical direction of mask (mV) When YDELta is selected, the positive value is set.

Response Data

<numeric>

Example of Use

To set the user adjustment marker Y1 to 10 mV and the user adjustment marker Y2 to –10 mV, respectively

```
:CONF:MASK:USER:LOC:Y1 10
:CONF:MASK:USER:LOC:YDEL 20
```

:CONFigure:MASK:USER:MARKer

Function

This command sets and queries the user adjustment marker display of the mask of EYE/Pulse Scope.

Syntax

```
:CONFigure:MASK:USER:MARKer 0|1
:CONFigure:MASK:USER:MARKer?
```

0	Deletes user adjustment marker
1	Displays user adjustment marker

Response Data

0|1

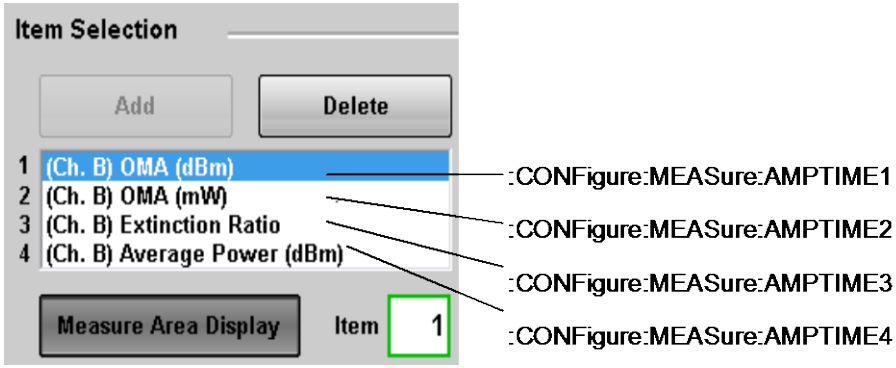
Example of Use

```
:CONF:MASK:USER:MARK 1
```

:CONFigure:MEASure:AMPTIME{1|2|3|4}

Function

This command sets and queries the setting measurement items related to the amplitude and time displayed on the EYE/Pulse Scope screen.



Syntax

```
:CONFigure:MEASure:AMPTIME{1|2|3|4} {CHA|CHB},<integer>
:CONFigure:MEASure:AMPTIME{1|2|3|4}?
```

CHA: Channel A
CHB: Channel B

<integer>	Measurement Item
0	One Level
1	Zero Level
2	Eye Amplitude
3	Eye Height
4	Crossing
5	SNR
6	Average Power (dBm)
7	Average Power (mW)
8	Extinction Ratio
9	Jitter p-p
10	Jitter RMS
11	Rise Time
12	Fall Time
13	Eye Width
14	DCD
15	OMA (mW)
16	OMA (dBm)

Response Data

{ CHA | CHB },<integer>

Example of Use

The following measurement result is displayed on the screen:

- Channel A jitter (p-p)
- Channel A jitter (RMS)
- Channel A Crossing
- Channel A eye amplitude

```
:CONF:MEAS:AMPTIME1 CHA,9; :CONF:MEAS:AMPTIME2 CHA,10
:CONF:MEAS:AMPTIME3 CHA,4; :CONF:MEAS:AMPTIME4 CHA,2
```

To query the measurement result displayed on the screen:

```
:CONF:MEAS:AMPTIME1?
>CHA,9
:CONF:MEAS:AMPTIME2?
>CHA,10
:CONF:MEAS:AMPTIME3?
>CHA,4
:CONF:MEAS:AMPTIME4?
>CHA,2
```

:CONFigure:MEASure:AREa:DISPlay

Function

This command sets and queries the Amplitude/Time measurement area display of EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:AREa:DISPlay 0|1
:CONFigure:MEASure:AREa:DISPlay?
```

0 Off
1 On

Response Data

0|1

Example of Use

```
:CONF:MEAS:ARE:DISP 1
:CONF:MEAS:ARE:DISP?
>1
```

:CONFigure:MEASure:AREa:ITEM

Function

This command sets and queries the measurement item number displayed in the Amplitude/Time measurement area of EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:AREa:ITEM 1|2|3|4
:CONFigure:MEASure:AREa:ITEM?
```

1 measurement item 1
2 measurement item 2
3 measurement item 3
4 measurement item 4

Response Data

1|2|3|4

Example of Use

```
:CONF:MEAS:ARE:ITEM 4
:CONF:MEAS:ARE:ITEM?
>4
```

:CONFigure:MEASure:CHANnel

Function

This command sets the active channel of the waveform measurement on EYE/Pulse Scope.
The query returns the current active channel.

Syntax

```
:CONFigure:MEASure:CHANnel A|B
:CONFigure:MEASure:CHANnel?
```

A: Channel A
B: Channel B

Response Data

A|B

Example of Use

```
:CONF:MEAS:CHAN A
:CONF:MEAS:CHAN?
>A
```

:CONFigure:MEASure:DEFine

Function

This command sets and queries the level measuring the Rise/Fall time on EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:DEFine 0|1
:CONFigure:MEASure:DEFine?
```

0 20/80
1 10/90

Response Data

0|1

Example of Use

```
:CONF:MEAS:DEF 1

:CONF:MEAS:DEF?
>1
```

:CONFigure:MEASure:EYEBoundary:OFFSet

Function

This command sets the horizontal position measuring 1 level and 0 level on EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:EYEBoundary:OFFSet <numeric>
:CONFigure:MEASure:EYEBoundary:OFFSet?
```

<numeric>:0.00~1.00 Position measuring level (UI)

Response Data

<numeric>

Example of Use

```
:CONF:MEAS:EYEB:OFFS 0.3

:CONF:MEAS:EYEB:OFFS?
>0.3
```

:CONFigure:MEASure:EYEBoundary:WIDTh

Function

This command sets and queries the horizontal width measuring 1 level and 0 level on EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:EYEBoundary:WIDTh <numeric>
:CONFigure:MEASure:EYEBoundary:WIDTh?
```

<numeric>:0.00~1.00 Width measuring level (UI)

Response Data

<numeric>

Example of Use

```
:CONF:MEAS:EYEB:WIDT 0.4

:CONF:MEAS:EYEB:WIDT?
>0.4
```

:CONFigure:MEASure:TRANsition:CORRect:FACTor

Function

The bandwidth of the sampling scope is corrected with the measurement value of the Rise/Fall time on EYE/Pulse Scope. The query returns the Rise/Fall time correction factor on EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:TRANsiton:CORRect:FACTor <numeric>
:CONFigure:MEASure:TRANsiton:CORRect:FACTor?
```

<numeric>:Correction factor 0.0~9999.9

Response Data

<numeric>

Example of Use

```
:CONF:MEAS:TRAN:CORR:FACT 1.06

:CONF:MEAS:TRAN:CORR:FACT?
>1.06
```

:CONFigure:MEASure:TRANsition:CORRection

Function

This command sets and queries whether to use the Rise/Fall time correction factor on EYE/Pulse Scope.

Syntax

```
:CONFigure:MEASure:TRANsiton:CORRection 0|1
:CONFigure:MEASure:TRANsiton:CORRection?
```

0 Does not use correction factor
1 Uses correction factor

Response Data

0|1

Example of Use

```
:CONF:MEAS:TRAN:CORR 1

:CONF:MEAS:TRAN:CORR?
>1
```

:CONFigure:MEASure:TYPe

Function

This command sets the measurement items on EYE/Pulse Scope and measures the set item.
The query returns the current measurement item.

As for the following subsystem query message, set the measurement items using the :CONFigure:MEASure:TYPe before the measurement.

:FETCH:AMPLitude
:FETCH:HISTogram
:FETCH:MASK
:FETCH:TIME

As for the query message acquiring the following measurement result, do not set the measurement items using the :CONFigure:MEASure:TYPe before the measurement.

:MEASure:AMPLitude
:MEASure:HISTogram:AMPLitude
:MEASure:HISTogram:TIME
:MEASure:TIME
:MEASure:MASK

Syntax

```
:CONFigure:MEASure:TYPe
AMPHistogram|AMPMAsk|AMPTIME|HISTogram|MASK|OFF
:CONFigure:MEASure:TYPe?
```

AMPHistogram	Measures amplitude and histogram.
AMPMask	Measures amplitude and performs mask test.
AMPTIME	Displays Eye Mode, and measure the items related to the amplitude and time. Average Power(dBm), Average Power(mW), Crossing, DCD, Extinction Ratio,Eye Amplitude, Eye Height, Eye Width, Fall Time, Jitter p-p, Jitter RMS, OMA(dBm), OMA(mW), One Level, SNR, Rise Time, and Zero Level
HISTogram	Measures histogram. Hits of Time/Amplitude, Mean value, Peak-to-Peak, and Standard Deviation
MASK	Performs mask test.
OFF	Stop measurement.

Response Data

AMPHistogram | AMPMAsk | AMPTIME | HISTogram | MASK | OFF

Example of Use

To set the measurement items on EYE/Pulse Scope to the mask test:

```
:CONF:MEAS:TYP MASK
```

To query the measurement item settings on EYE/Pulse Scope:

```
:CONF:MEAS:TYP?
>MASK
```

:CONFigure:SKEW:CHA|CHB

Function

This command sets and queries the skew on EYE/Pulse Scope.

Syntax

```
:CONFigure:SKEW:CHA|CHB <numeric>
:CONFigure:SKEW:CHA|CHB?
```

<numeric>: Skew (Unit:ps)

The setting range is as follows.

Unit	Range
ps	–999.9~999.9

Response Data

<numeric>

Example of Use

```
:CONF:SKEW:CHA 6.4

:CONF:SKEW:CHA?
>6.4
```

:CONFigure:TRACking:DRATe

Function

This command sets and queries the bit and clock rate tracking on EYE/Pulse Scope.

Syntax

```
:CONFigure:TRACking:DRATe 0|1
:CONFigure:TRACking:DRATe?
```

0 Releases tracking
1 Performs tracking

Response Data

0|1

Example of Use

```
:CONF:TRAC:DRAT 0

:CONF:TRAC:DRAT?
>0
```

:CONFigure:TRACking:DRATe:MASTer

Function

This command sets the tracking factor to be synchronized on EYE/Pulse Scope.
The query returns the tracking factor to be synchronized on EYE/Pulse Scope.

Syntax

```
:CONFigure:TRACking:DRATe 0|1|2|3
:CONFigure:TRACking:DRATe?
```

0 PPG1
1 ED1
2 PPG2
3 ED2

Response Data

0|1|2|3

Example of Use

```
:CONF:TRAC:DRAT:MAST 2

:CONF:TRAC:DRAT:MAST?
>2
```

:CONFigure:TRACking:PATLength

Function

This command sets and queries the tracking of the pattern length on EYE/Pulse Scope.

Syntax

```
:CONFigure:TRACking:PATLength 0|1
:CONFigure:TRACking:PATLength?
```

0 Releases tracking
1 Performs tracking

Response Data

0|1

Example of Use

```
:CONF:TRAC:PATL 0
```

```
:CONF:TRAC:PATL?
>0
```

:CONFigure:TRACking:PATLength:MASTer

Function

This command sets the factor of the pattern length to be synchronized on EYE/Pulse Scope.
The query returns the factor of the pattern length to be synchronized on EYE/Pulse Scope.

Syntax

```
:CONFigure:TRACking:PATLength:MASTer 0|1|2|3
:CONFigure:TRACking:PATLength:MASTer?
```

0 PPG1
1 ED1
2 PPG2
3 ED2

Response Data

0|1|2|3

Example of Use

```
:CONF:TRAC:PATL:MAST 2

:CONF:TRAC:PATL:MAST?
>2
```

:DISPlay:ACTive

Function

This command sets the screen to be displayed on the display.
This command switches the screen to be displayed. To change the target module for the remote control, use :MODule:ID.

Syntax

```
:DISPlay:ACTive 1|2|3|4|5|6|7
```

1: PPG/ED 1ch
2: PPG/ED 2ch
3: XFP/SFP+
4: O/E
5: EYE/Pulse Scope
6: Jitter Analysis
7: Transmission Analysis

Example of Use

To display the PPG/ED 1ch panel

```
:DISPlay:ACTive 1
```

:DISPlay:RESult:EALarm:HRESet

Function

This command resets the histories on the ED measurement result screen.

Syntax

```
:DISPlay:RESult:EALarm:HRESet
```

Example of Use

```
:DISP:RES:EAL:HRES
```

:DISPlay:RESult:EALarm:MODE

Function

This command sets and queries whether to update the measurement results immediately on the ED measurement result screen.

Syntax

```
:DISPlay:RESult:EALarm:MODE 0|1|OFF|ON
:DISPlay:RESult:EALarm:MODE?
```

0|OFF Measurement results are not updated immediately at OFF
1|ON Measurement results are updated immediately at ON

Response Data

0|1

Example of Use

To set the display of the measurement result data to On immediately

```
:DISP:RES:EAL:MODE 1
```

To query the updated settings of the measurement result data

```
:DISP:RES:EAL:MODE?
>1
```

:DISPlay:WINDow:CHANnel:BOTH

Function

This commnad sets the scales of Channel A and channel B of EYE/Pulse Scope to the same scales.
The query returns whether the scales of channel A and channel B of EYE/Pulse Scope are set to the same scales.

Syntax

```
:DISPlay:WINDow:CHANnel:BOTH 0|1
:DISPlay:WINDow:CHANnel:BOTH?
```

0 Sets the scale of Channel A and B separately.
1 Sets the scale of Channel A and B to the same one.

Response Data

0|1

Example of Use

```
:DISP:WIND:CHAN:BOTH 1
```

:DISPlay:WINDow:GRAPhics:CLEar

Function

This command erases the trace on the EYE/Pulse Scope screen.

Syntax

```
:DISPlay:WINDow:GRAPhics:CLEar
```

Example of Use

```
:DISP:WIND:GRAP:CLE
```

:DISPlay:WINDow[:SCALe]:AUTOscale

Function

This command rescales the eye pattern display to the proper amplitude and time scale so that the waveform of the EYE/Pulse Scope is centered on the screen.

With the EYE mode, the parameter can be specified. In this case, the waveform frequency is not measured. The offset on the horizontal axis and the scale on the vertical axis are adjusted.

Syntax

```
:DISPlay:WINDow[:SCALe]:AUTOscale
[BOTH|HORIzontal|VERTical]
```

BOTH:	Scale on the vertical axis and offset on the horizontal axis auto-adjusted
HORizontal:	Offset on the horizontal axis auto-adjusted
VERTical:	Scale on the vertical axis auto-adjusted

Example of Use

```
:DISP:WIND:AUTO
```

:DISPlay:WINDow:X[:SCALe]:BITs

Function

This sets the horizontal scale of the EYE/Pulse Scope by the number of data rate bits.
The query returns the current horizontal scale by the number of data rate bits.

Syntax

```
:DISPlay:WINDow:X[:SCALe]:BITs <integer>
:DISPlay:WINDow:X[:SCALe]:BITs?
```

<integer>: 1 to 65535, 1 bit step

Response Data

<integer>

Example of Use

```
:DISP:WIND:X:BIT 1000
:DISP:WIND:X:BIT?
> 1000
```

:DISPlay:WINDow:X[:SCALe]:OFFSets

Function

This command sets and queries the offset value of the horizontal scale of the EYE/Pulse Scope. The unit is UI or pico seconds. The unit can be set using the :DISPlay:WINDow:X[:SCALe]:UNIT.

Syntax

```
:DISPlay:WINDow:X[:SCALe]:OFFSets <numeric>
:DISPlay:WINDow:X[:SCALe]:OFFSets?
```

<numeric>
When unit is UI: 0 to 16777215
When unit is ps: Offset (UI) / Date-Rate (Tbps or 1000 Gbps)

Response Data

<numeric>

Example of Use

```
:DISP:WIND:X:OFFS 150
```

:DISPlay:WINDow:X[:SCALe]:UNIT

Function

This command sets the horizontal scale unit of EYE/Pulse Scope. Also, this command queries the current horizontal scale unit of EYE/Pulse Scope.

Syntax

```
:DISPlay:WINDow:X[:SCALe]:UNIT UI|PS
:DISPlay:WINDow:X[:SCALe]:UNIT?
```

UI: Sets the unit to Unit Interval .
PS: Sets the unit to pico second (10^{-12} seconds)

Response Data

PS|UI

Example of Use

```
:DISP:WIND:X:UNIT UI

:DISP:WIND:X:UNIT?
>UI
```

:DISPlay:WINDow:Y[:SCALe]:DIVision:CHA|CHB

Function

This command sets the vertical scale of the EYE/Pulse Scope.
The query returns the current vertical scale.
The vertical scale can set either of electrical interface or optical interface using this message. The units, mV and µW, are used for the electrical and optical interface, respectively.
The maximum value of the settable scale varies with the attenuation set at each channel.The attenuation can be set using [:SENSe]:INPut:ATTenuation:CHA|CHB. For the optical interface, the settable maximum values vary with the O/E conversion gain. The O/E conversion gain can be set using :CALibrate:Cgain.

Syntax

```
:DISPlay:WINDow:Y[:SCALe]:CHA|CHB <numeric>
:DISPlay:WINDow:Y[:SCALe]:CHA|CHB?
```

: 1.0 mV to 250.0 × [10 (attenuation/20)] mV
Optical: Divided by Gain (V/W) (Conversion gain)
The setting range is the same as electrical.

Response Data

<numeric>

Example of Use

```
:DISP:WIND:Y:OFFS:CHB?
>100
```

:DISPlay:WINDow:Y[:SCALe]:DIVision:CHMath

Function

This command sets the vertical scale of the EYE/Pulse Scope when the Channel Math is On.
The unit is mV.
The query returns the current vertical scale.
The maximum value of the settable scale varies with the attenuation set at each channel. The attenuation can be set using [:SENSe]:INPut:ATTenuation:CHA|CHB.

Syntax

```
:DISPlay:WINDow:Y[:SCALe]:CHMath <numeric>
:DISPlay:WINDow:Y[:SCALe]:CHMath?
```

<numeric>
1 to 200 mV, 0.1 mV step

Response Data

<numeric>

Example of Use

```
:DISP:WIND:Y:OFFS:CHM?
>100
```

:DISPlay:WINDow:Y[:SCALe]:OFFSets:CHA|CHB

Function

This command sets the offset value of the vertical scale of the EYE/Pulse Scope.
The query returns the offset value of the current vertical scale.
The vertical scale can set either of electrical or optical interfaces using this message. The units, mV and µW, are used for the electrical and optical interfaces, respectively.
The maximum value of the settable scale varies with the attenuation set at each channel. The attenuation can be set using [:SENSe]:INPut:ATTenuation:CHA|CHB. For the optical interface, the settable maximum values vary with the O/E conversion gain. The O/E conversion gain can be set using: CALibrate:Cgain.

Syntax

```
:DISPlay:WINDow:Y[:SCALe]:OFFSets:CHA|CHB <numeric>
:DISPlay:WINDow:Y[:SCALe]:OFFSets:CHA|CHB?
```

<numeric>
Electrical interface:–500.0 mV ~ +500.0 mV
Optical interface: Divided value (nm) (divides –500.0 mV ~ +500.0 mV by Gain (V/W) (conversion gain))

Response Data

<numeric>

Example of Use

```
:DISP:WIND:OFFS:Y:CHB?
>-50
```

:DISPlay:WINDow:Y[:SCALe]:OFFSets:CHMath

Function

This command sets the offset value of the vertical scale of the EYE/Pulse Scope when the Channel Math is On. The unit is mV.
The query returns the offset value of the current vertical scale.
The maximum value of the settable scale varies with the attenuation set at each channel. The attenuation can be set using [:SENSe]:INPut:ATTenuation:CHA|CHB.

Syntax

```
:DISPlay:WINDow:Y[:SCALe]:OFFSets:CHMath <numeric>
:DISPlay:WINDow:Y[:SCALe]:OFFSets:CHMath?
```

<numeric>
–1000.0 to +1000.0 mV, 0.1 mV step

Response Data

<numeric>

Example of Use

```
:DISP:WIND:OFFS:Y:CHM?
>-50
```

:FETCh:AMPLitude:AVEPower

Function

This command queries the optical average power input to the O/E converter.
The valid value is acquired when the active channel is the optical interface.

Syntax

```
:FETCh:AMPLitude:AVEPower?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric> | N/A
This indicates the power light, mean value, its standard deviation, and minimum/maximum values in μW and dBm.
The response data is output in the following order.

Order	Item	Unit	Content
1	Average Power	μW	Average power
2	Average Power	dBm	Average power
3	Average	μW	Mean value of average power
4	Average	dBm	Mean value of average power
5	Std.Dev.	μW	Standard deviation of average power
6	Std.Dev.	dB	Standard deviation of average power
7	Min	μW	Minimum value of average power
8	Min	dBm	Minimum value of average power
9	Max	μW	Maximum value of average power
10	Max	δBμ	Maximum value of average power

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.
- The O/E converter doesn't exist.
- An active channel of the waveform measurement is set to A.

Example of Use

```
:FETC:AMPL:AVEP?
>25.00,-16.02,25.50,-15.93,0.02,0.05,24.86,-16.05,26.12,
-15.83
```

:FETCh:AMPLitude:CROSsing

Function

This command queries the eye cross ratio for the amplitude measurement function of the EYE/Pulse Scope.

Syntax

```
:FETCh:AMPLitude:CROSsing?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the eye crossing rate, its mean value, deviation value, and minimum/maximum values in % within the range of 0.00 to 100.00
The response data is output in the following order.

Order	Item	Unit	Content
1	Crossing	%	Eye cross rate
2	Average	%	Mean value of eye cross rate
3	Std.Dev.	%	Standard deviation of eye cross rate
4	Min	%	Minimum value of eye cross rate
5	Max	%	Maximum value of eye cross rate

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:AMPL:CROS?
>46.01,45.80,0.19,45.27,46.41
```

:FETCh:AMPLitude:EXTRatio

Function

This command queries the extinction ratio for the amplitude measurement function of the EYE/Pulse Scope.
The valid value is acquired when the active channel is the optical interface.

Syntax

```
:FETCh:AMPLitude:EXTRatio?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the extinct ratio, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order. There is no unit.

Order	Item	Unit	Content
1	Extinction Ratio		Extinct rate
2	Average		Mean value of extinct rate
3	Std.Dev.		Standard deviation of extinct rate
4	Min		Minimum value of extinct rate
5	Max		Maximum value of extinct rate

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.
- The O/E converter doesn't exist.
- An active channel of the waveform measurement is set to A.

Example of Use

```
:FETC:AMPL:EXTR?
>6.82,6.77,0.13,6.38,7.16
```

:FETCh:AMPLitude:EYEAmplitude

Function

This command queries the eye amplitude for the amplitude measurement function of the EYE/Pulse Scope.
The electrical channel is set in mV, and the optical channel is set in the μW unit.

Syntax

```
:FETCh:AMPLitude:EYEAmplitude?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the eye amplitude, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

Order	Item	Unit	Content
1	Eye amplitude	mV/μW	Eye amplitude
2	Average	mV/μW	Mean value of eye amplitude
3	Std.Dev.	mV/μW	Standard deviation of eye amplitude
4	Min	mV/μW	Minimum value of eye amplitude
5	Max	mV/μW	Maximum value of eye amplitude

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:AMPL:EYEA?
>55.22,54.89,0.12,54.53,55.25
```

:FETCh:AMPLitude:EYEHeight

Function

This command queries the eye height for the amplitude measurement function of the EYE/Pulse Scope.
The electrical channel is set in mV, and the optical channel is set in the μW unit.

Syntax

```
:FETCh:AMPLitude:EYEHeight?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the eye height, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

Order	Item	Unit	Content
1	Eye height	mV/μW	Eye height
2	Average	mV/μW	Mean value of eye height
3	Std.Dev.	mV/μW	Standard deviation of eye height
4	Min	mV/μW	Minimum value of eye height
5	Max	mV/μW	Maximum value of eye height

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:AMPL:EYEH?
>45.81,45.77,0.08,45.53,46.01
```

:FETCh:AMPLitude:LEVel:ONE

Function

This command queries 1 level for the amplitude measurement function of the EYE/Pulse Scope.
The electrical channel is set in mV, and the optical channel is set in the μW unit.

Syntax

```
:FETCh:AMPLitude:LEVel:ONE?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the 1 level, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

Order	Item	Unit	Content
1	One Level	mV/μW	1 Level
2	Average	mV/μW	Mean value of 1 Level
3	Std.Dev.	mV/μW	Standard deviation of 1 Level
4	Min	mV/μW	Minimum value of 1 Level
5	Max	mV/μW	Maximum value of 1 Level

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:AMPL:LEV:ONE?
>35.16,35.11,0.11,34.78,35.44
```

:FETCh:AMPLitude:LEVel:ZERO

Function

This command queries 0 level for the amplitude measurement of the EYE/Pulse Scope.
The electrical channel is set in mV, and the optical channel is set in the μW unit.

Syntax

```
:FETCh:AMPLitude:LEVel:ZERO?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the 0 level, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

Order	Item	Unit	Content
1	Zero Level	mV/μW	0 Level
2	Average	mV/μW	Mean value of 0 Level
3	Std.Dev.	mV/μW	Standard deviation of 0 Level
4	Min	mV/μW	Minimum value of 0 Level
5	Max	mV/μW	Maximum value of 0 Level

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:AMPL:LEV:ZERO?
>-15.12,-15.20,0.05,-15.35,-15.05
```

:FETCh:AMPLitude:MEASurement

Function

This command queries all items for the amplitude measurement function of the EYE/Pulse Scope.
The measurement items are 11 items; 1 level, 0 level, eye amplitude, eye height, crossing ratio, signal to noise ratio, average power (μW), average power (dBm), extinct ratio, OMA (mW), and OMA (dBm).
The present value, mean value, standard deviation, and minimum/maximum values for this 11 measurement items can be read out. The output number of data is 55.

Syntax

```
:FETCh:AMPLitude:MEASurement?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,{<numeric>|N/A},{<numeric>|N/A},{<numeric>|N/A},{<numeric>|N/A},{<numeric>|N/A}……|N/A

The output order is as follows:

Table 4.4.2-2 Data Order

Item	Unit	Present value	Mean value	Standard deviation	Minimum value	Maximum value
One Level	mV/ μW	1	12	23	34	45
Zero Level	mV/ μW	2	13	24	35	46
Eye amplitude	mV/ μW	3	14	25	36	47
Eye height	mV/ μW	4	15	26	37	48
Crossing	%	5	16	27	38	49
SNR		6	17	28	39	50
Average Power	μW	7	18	29	40	51
Average Power	dBm	8	19	30	41	52
Extinction Ratio		9	20	31	42	53
OMA (mW)	mW	10	21	32	43	54
OMA (dBm)	dBm	11	22	33	44	55

When an optical channel is not set to an active channel, the following data becomes N/A.

- Average power (μW)
- Average power (dBm)
- Extinct rate

- OMA (mW)
- OMA (dBm)

The response data is N/A when the measurement item is set to the following:

- Histogram
- Mask
- OFF

Example of Use

```
:FETC:AMPL:MEAS?
:FETC:AMPL:MEAS?
>35.12,-15.01,50.53,46.29,46,10.5,N/A,N/A,N/A,N/A,N/A,
36.98,-14.85,52.66,40.44,43.09,10.22,N/A,N/A,N/A,N/A,N/A,
0.23,0.15,1.46,1.25,0.69,0.38,N/A,N/A,N/A,N/A,N/A,
36.29,-15.3,48.28,36.39,41.02,9.08,N/A,N/A,N/A,N/A,N/A,
37.67,-14.4,57.04,44.19,45.16,11.36,N/A,N/A,N/A,N/A,N/A
```


:FETCh:AMPLitude:OMA:DBM

Function

This command queries an optical amplitude of the EYE/Pulse Scope amplitude measurement function in dBm.

Syntax

```
:FETCh:AMPLitude:OMA:DBM?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the signal to noise ratio, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order. There is no unit.

No.	Item	Unit	Content
1	OMA	dBm	Optical amplitude
2	Average	dBm	Mean value of optical amplitude
3	Std.Dev.	dBm	Standard deviation of optical amplitude
4	Min	dBm	Minimum value of optical amplitude
5	Max	dBm	Maximum value of optical amplitude

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.
- The O/E converter doesn't exist.
- An active channel of the waveform measurement is set to A.

Example of Use

```
:FETC:AMPL:OMA:DBM?
>-8.22,-8.24,0.21,-8.85,-7.59
```

:FETCh:AMPLitude:OMA:MW

Function

This command queries an optical amplitude of the EYE/Pulse Scope amplitude measurement function in the mW unit.

Syntax

```
:FETCh:AMPLitude:OMA:MW?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the signal to noise ratio, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order. There is no unit.

No.	Item	Unit	Content
1	OMA	mW	Optical amplitude
2	Average	mW	Mean value of optical amplitude
3	Std.Dev.	mW	Standard deviation of optical amplitude
4	Min	mW	Minimum value of optical amplitude
5	Max	mW	Maximum value of optical amplitude

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.
- The O/E converter doesn't exist.
- An active channel of the waveform measurement is set to A.

Example of Use

```
:FETC:AMPL:OMA:MW?
>0.15,0.16,0.03,0.06,0.25
```

:FETCh:AMPLitude:SNR

Function

This command queries the signal-to-noise ratio for the amplitude measurement function of the EYE/Pulse Scope.

Syntax

```
:FETCh:AMPLitude:SNR?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the signal to noise ratio, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order. There is no unit.

No.	Item	Content
1	SNR	SNR (signal to noise ratio)
2	Average	Mean value of SNR
3	Std.Dev.	Standard deviation of SNR
4	Min	Minimum value of SNR
5	Max	Maximum value of SNR

The response data in the following cases is N/A.

- The measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:AMPL:SNR?
>10.08,10.11,0.19,9.55,10.70
```

:FETCh:AMPTime:QUEStionableeye

Function

This command queries whether to display "* EYE" on the EYE/Pulse Scope amplitude measurement screen.

Syntax

```
:FETCh:AMPTime:QUEStionableeye?
```

Response Data

0 | 1

0: Not display
1: Display

Example of Use

```
:FETC:AMPT:QUES?
>1
```

:FETCh:HISTogram:AMPLitude:HITS

Function

This command queries the number of samples (this number of samples is called the number of hits) in the area of the histogram measurement on the amplitude axis in the function of the measurement of the histogram of EYE/Pulse Scope.
This message queries data that the measurement ends. To execute the histogram measurement of the amplitude axis and to update the result, use the following command: MEASure:HISTogram:AMPLitude?.

Syntax

```
:FETCh:HISTogram:AMPLitude:HITS?
```

Response Data

When the measurement item is not set to the following or histogram on the time axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

<integer> | N/A

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:AMPL:HITS?
>89632
```

:FETCh:HISTogram:AMPLitude:MEAN

Function

This command queries the mean amplitude of the histogram measurement on the amplitude axis in the EYE/Pulse Scope histogram measurement function.
mV and μ W is used as the unit for electric channel and optical channel, respectively.
This message queries data that the measurement ends. To execute the histogram measurement of the amplitude axis and to update the result, use the following command; MEASure:HISTogram:AMPLitude?.

Syntax

```
:FETCh:HISTogram:AMPLitude:MEAN?
```

Response Data

When the measurement item is not set to the following or histogram at the time axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:AMPL:MEAN?
>32.1
```

:FETCh:HISTogram:AMPLitude:MEASurement

Function

This command queries four results of the histogram measurement of the amplitude axis in the EYE/Pulse Scope histogram measurement function.
This message queries data that the measurement ends. To execute the histogram measurement of the amplitude axis and to update the result, use the following command; MEASure:HISTogram:AMPLitude?.

Syntax

```
:FETCh:HISTogram:AMPLitude:MEASurement?
```

Response Data

<numeric>,<numeric>,<numeric>,<integer> | N/A
The response data is output in the following order.

Item	Unit	Content
Mean	mV/ μW	Mean value
StdDiv	mV/ μW	Standard deviation
Peak to Peak	mV/ μW	Amplitude
Hits		Number of samples in area

When the measurement item is not set to the following or histogram at the time axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

Example of Use

```
:FETC:HIST:AMPL:MEAS?
>32.1,4.53,28.1,89632
```

:FETCh:HISTogram:AMPLitude:PPeak

Function

This command queries the amplitude (Peak to Peak) of the histogram measurement on the amplitude axis in the EYE/Pulse Scope histogram measurement function.
mV and μW is used as the units for electric channel and optical channel, respectively.
This message queries data that the measurement ends. To execute the histogram measurement of the amplitude axis and to update the result, use the following command; :MEASure:HISTogram:AMPLitude?.

Syntax

```
:FETCh:HISTogram:AMPLitude:PPeak?
```

Response Data

When the measurement item is not set to the following or histogram at the time axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:AMPL:PP?
>28.1
```

:FETCh:HISTogram:AMPLitude:STDDeviation

Function

This command queries the standard deviation of the histogram measurement result on the amplitude axis in the EYE/Pulse Scope histogram measurement function.
mV and μW is used as the units for electric channel and optical channel, respectively.
This message queries data that the measurement ends. To execute the histogram measurement of the amplitude axis and to update the result, use the following command; :MEASure:HISTogram:AMPLitude?.

Syntax

```
:FETCh:HISTogram:AMPLitude:STDDeviation?
```

Response Data

When the measurement item is not set to the following or histogram at the time axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:AMPL:STDD?
>4.53
```

:FETCh:HISTogram:TIME:HITS

Function

This command queries the sample count of the histogram measurement on the time axis in the EYE/Pulse Scope histogram measurement function.
This message queries data that the measurement ends. To execute the histogram measurement of the time axis and to update the result, use the following command; :MEASure:HISTogram:TIME?.

Syntax

```
:FETCh:HISTogram:TIME:HITS?
```

Response Data

When the measurement item is not set to the following or histogram at the amplitude axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:TIME:MEAS?
>6831
```

:FETCh:HISTogram:TIME:MEAN

Function

This command queries the mean value of the histogram measurement on the time axis in the EYE/Pulse Scope histogram measurement function. The unit is ps.
This message queries data that the measurement ends. To execute the histogram measurement of the time axis and to update the result, use the following command; :MEASure:HISTogram:TIME?.

Syntax

```
:FETCh:HISTogram:TIME:MEAN?
```

Response Data

When the measurement item is not set to the following or histogram at the amplitude axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:TIME:HITS?
>1.53
```

:FETCh:HISTogram:TIME:MEASurement

Function

This command queries four histogram measurement results the time axis in the EYE/Pulse Scope histogram measurement function.
This message queries data that the measurement ends. To execute the histogram measurement of the time axis and to update the result, use the following command; :MEASure:HISTogram:TIME?.

Syntax

```
:FETCh:HISTogram:TIME:MEASurement?
```

Response Data

When the measurement item is not set to the following or histogram at the amplitude axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

The response data is output in the following order.

Item	Unit	Content
Mean	ps	Mean value
StdDiv	ps	Standard deviation
Peak to Peak	ps	Amplitude
Hits		Number of samples in area

The response data is N/A when the measurement item is not Histogram.

Example of Use

```
:FETC:HIST:TIME:MEAS?
>1.53,0.022,0.081,6831
```

:FETCh:HISTogram:TIME:PPeak

Function

This command queries the time difference (Peak to Peak) of the histogram measurement on the time axis in the EYE/Pulse Scope histogram measurement function. The unit is ps.
This message queries data that the measurement ends. To execute the histogram measurement of the time axis and to update the result, use the following command; :MEASure:HISTogram:TIME?.

Syntax

```
:FETCh:HISTogram:TIME:PPeak?
```

Response Data

<numeric>|N/A
When the measurement item is not set to the following or histogram at the amplitude axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFFF

Example of Use

```
:FETC:HIST:TIME:PP?
>0.081
```

:FETCh:HISTogram:TIME:STDDeviation

Function

This command queries the standard deviation of the histogram measurement results on the time axis in the EYE/Pulse Scope histogram measurement function. The unit is ps or UI.
This message queries data that the measurement ends. To execute the histogram measurement of the time axis and to update the result, use the following command; MEASure:HISTogram:TIME?.

Syntax

```
:FETCh:HISTogram:TIME:STDDeviation?
```

Response Data

<numeric>|N/A

When the measurement item is not set to the following or histogram at the amplitude axis, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Mask
- Mask
- OFF

Example of Use

```
:FETC:HIST:TIME:STDD?
>0.022
```

:FETCh:MASK:MEASurement

Function

This command queries the most recent results for the mask measurement of the EYE/Pulse Scope.

Data are returned as 2 comma-separated values: total samples, failed samples.

To update the result after executing the mask test, use `:MEASure:MASK?`.

Syntax

```
:FETCh:MASK:MEASurement?
```

Response Data

<integer>,<integer>|N/A

The output order is as follows:

Item	Unit	Details
Total Samples	–	Total sample count
Failed	–	Failed sample count

When the measurement item is not set to the following, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Histogram
- Histogram
- OFF

Example of Use

```
:FETC:MASK:MEAS?
>16831,0
```

:FETCh:MASK:SAMPles:FAILed

Function

For the mask measurement of the EYE/Pulse Scope, this command queries the most recent failed samples in the Top, Center, and Bottom mask area. The unit is not available.
To update the result after executing the mask test, use `:MEASure:MASK?`.

Syntax

```
:FETCh:MASK:SAMPles:FAILed?
```

Response Data

<integer> | N/A
When the measurement item is not set to the following, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Histogram
- Histogram
- OFF

Example of Use

```
:FETC:MASK:SAMP:FAIL?
>0
```

:FETCh:MASK:SAMPles:FAILed:BOTTom

Function

For the mask measurement of the EYE/Pulse Scope, this command queries the sample count in the bottom mask area. The unit is not available.
To update the result after executing the mask test, use `:MEASure:MASK?`.

Syntax

```
:FETCh:MASK:SAMPles:FAILed:BOTTom?
```

Response Data

<integer> | N/A

When the measurement item is not set to the following, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Histogram
- Histogram
- OFF

Example of Use

```
:FETC:MASK:SAMP:FAIL:BOTT?
>0
```

:FETCh:MASK:SAMPles:FAILed:CENTer

Function

For the mask measurement of the EYE/Pulse Scope, this command queries the sample count in the center mask area. The unit is not available.

To update the result after executing the mask test, use `:MEASure:MASK?`.

Syntax

```
:FETCh:MASK:SAMPles:FAILed:CENTer?
```

Response Data

<integer> | N/A

When the measurement item is not set to the following, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Histogram
- Histogram
- OFF

Example of Use

```
:FETC:MASK:SAMP:FAIL:CENT?
>5693
```

:FETCh:MASK:SAMPles:FAILed:TOP

Function

For the mask measurement of the EYE/Pulse Scope, this command queries the sample count in the top mask area. The unit is not available.
To update the result after executing the mask test, use :MEASure:MASK?.

Syntax

```
:FETCh:MASK:SAMPles:FAILed:TOP?
```

Response Data

<integer> | N/A
When the measurement item is not set to the following, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Histogram
- Histogram
- OFF

Example of Use

```
:FETC:MASK:SAMP:FAIL:TOP?
>12
```

:FETCh:MASK:SAMPles:TOTal

Function

For the mask measurement of the EYE/Pulse Scope, this command queries total sample count.
The unit is not available.
To update the result after executing the mask test,
use :MEASure:MASK?.

Syntax

```
:FETCh:MASK:SAMPles:TOTal?
```

Response Data

<integer> | N/A
When the measurement item is not set to the following, the response data is N/A.

- Amplitude/Time
- Amplitude/Time&Histogram
- Histogram
- OFF

Example of Use

```
:FETC:MASK:SAMP:TOT?
>16831
```

:FETCh:TIME:DCD

Function

This command queries the most recent Duty Cycle Distortion result for the time measurement function of the EYE/Pulse Scope. The % unit can be used.

Syntax

```
:FETCh:TIME:DCD?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A
This indicates the duty cycle distortion, its mean value, standard deviation, and minimum/maximum values a within the range of 0 ~100. The response data is output in the following order. There is no unit.

No	Item	Unit	Content
1	Duty Cycle Distortion	%	Duty Cycle Distortion
2	Average	%	Mean value of Duty Cycle Distortion
3	Std.Dev.	%	Standard deviation of Duty Cycle Distortion
4	Min	%	Minimum value of Duty Cycle Distortion
5	Max	%	Maximum value of Duty Cycle Distortion

The response data is N/A when the measurement item is set to Histogram, Mask, or Off. .

Example of Use

```
:FETC:TIME:DCD?
>47.2,45.22,1.22,41.56,48.88
```

:FETCh:TIME:EYEWidth

Function

This command queries the eye width for the time measurement function of the EYE/Pulse Scope. The ps or UI unit can be used.

Syntax

```
:FETCh:TIME:EYEWidth?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the eye width, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

No	Item	Unit	Content
1	Eye Width	ps/UI	Eye width
2	Average	ps/UI	Mean value of Eye width
3	Std.Dev.	ps/UI	Standard deviation of Eye width
4	Min	ps/UI	Minimum value of Eye width
5	Max	ps/UI	Maximum value of Eye width

The response data is N/A when the measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:TIME:EYEW?
>208.60,206.15,3.32,216.11,196.19
```

:FETCh:TIME:FTIMe

Function

This command queries the rise-time for the time measurement function of the EYE/Pulse Scope.
The ps or UI unit can be used.

Syntax

```
:FETCh:TIME:FTIMe?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the Fall time, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

No	Item	Unit	Content
1	Fall Time	ps/UI	Fall time
2	Average	ps/UI	Mean value of Fall time
3	Std.Dev.	ps/UI	Standard deviation of Fall time
4	Min	ps/UI	Minimum value of Fall time
5	Max	ps/UI	Maximum value of Fall time

The response data is N/A when the measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:TIME:FTIM?
>133.66,129.96,2.59,122.19,137.75
```

:FETCh:TIME:JITTer:PPeak

Function

This command queries the jitter (Peak to Peak) for the time measurement function of the EYE/Pulse Scope.
The ps or UI unit ca be used.

Syntax

```
:FETCh:TIME:JITTer:PPeak?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the jitter (Peak to Peak), its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

No	Item	Unit	Content
1	Jitter (Peak to Peak)	ps/UI	Jitter (Peak to Peak)
2	Average	ps/UI	Mean value of Jitter (Peak to Peak)
3	Std.Dev.	ps/UI	Standard deviation of Jitter (Peak to Peak)
4	Min	ps/UI	Minimum value of Jitter (Peak to Peak)
5	Max	ps/UI	Maximum value of Jitter (Peak to Peak)

The response data is N/A when the measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:TIME:JITT:PP?
>66.25,65.89,0.98,63.95,68.83
```

:FETCh:TIME:JITTer:RMS

Function

This command queries the jitter (RMS) for the time measurement function of the EYE/Pulse Scope.
The ps or UI unit can be used.

Syntax

```
:FETCh:TIME:JITTer:RMS?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the jitter (Peak to Peak), its mean value, standard deviation, and minimum/maximum values within the range of 0 ~100.
The response data is output in the following order.

No	Item	Unit	Content
1	Jitter (RMS)	ps/UI	Jitter (RMS)
2	Average	ps/UI	Mean value of Jitter (RMS)
3	Std.Dev.	ps/UI	Standard deviation of Jitter (RMS)
4	Min	ps/UI	Minimum value of Jitter (RMS)
5	Max	ps/UI	Maximum value of Jitter (RMS)

The response data is N/A when the measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:TIME:JITT:RMS?
>15.31,15.52,0.26,14.74,16.30
```

:FETCh:TIME:MEASurement

Function

This command queries each item for the time measurement of the EYE/Pulse Scope.
Data are returned as 6 items: jitter p-p, jitter RMS, rise time, fall time, eye width, and DCD.
This command can read 30 values in total from the following six items: present value, mean value, standard deviation, and minimum/maximum values.

Syntax

```
:FETCh:TIME:MEASurement?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>……
|N/A
The output order is as follows:

Item	Unit	Details
Jitter p-p	ps/ UI	Jitter (Peak to peak)
Jitter RMS	ps/ UI	Jitter(RMS)
Rise Time	ps/ UI	Rise-time
Fall Time	ps/ UI	Fall-time
Eye Width	ps/ UI	Eye width
DCD	%	Duty Cycle Distortion

Table 4.4.2-3 Data Order

Item	Unit	Present value	Mean value	Standard deviation	Minimum value	maximum value
Jitter p-p	ps/ UI	1	7	13	19	25
Jitter RMS	ps/ UI	2	8	14	20	26
Rise Time	ps/ UI	3	9	15	21	27
Fall Time	ps/ UI	4	10	16	22	28
Eye Width	ps/ UI	5	11	17	23	29
DCD	%	6	12	18	24	30

The response data is N/A when the measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:TIME:MEAS?
>66.29,15.03,128.26,133.69,208.61,47.22,68.03,15.33,125.
99,134.01,203.98,47.01,1.99,0.31,2.21,1.86,3.19,0.55,62.
06,14.40,119.36,128.43,194.41,45.36,74.00,16.26,132.62,1
39.59,213.55,48.66
```

:FETCh:TIME:TRISe

Function

This command queries the rise-time for the time measurement function of the EYE/Pulse Scope.
The ps or UI unit can be used.

Syntax

```
:FETCh:TIME:TRISe?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>|N/A

This indicates the Rise time, its mean value, standard deviation, and minimum/maximum values.
The response data is output in the following order.

No	Item	Unit	Content
1	Rise Time	ps/UI	Rise time
2	Average	ps/UI	Mean value of Rise time
3	Std.Dev.	ps/UI	Standard deviation of Rise time
4	Min	ps/UI	Minimum value of Rise time
5	Max	ps/UI	Maximum value of Rise time

The response data is N/A when the measurement item is set to Histogram, Mask, or Off.

Example of Use

```
:FETC:TIME:TRIS?
>128.22,130.11,1.52,125.55,134.67
```

:INPut:BITRate

Function

This command sets the bit rate when the bit rate standard for the ED is Variable, Variable-1/2, Variable-1/4, Variable-1/8, Variable-1/16, Variable-1/32, or Variable-1/64.
The query returns the bit rate for the ED.

Syntax

```
:INPut:BITRate <numeric>
:INPut:BITRate?
```

<numeric> The setting range is as shown in the below table.

Bit Rate Specifications	With Option 090	Without Option 090
Variable	8000000~12500000	8500000~11320000
Variable-1/2	4000000~6250000	4250000~5660000
Variable-1/4	2000000~3125000	2125000~2830000
Variable-1/8	1000000~1562500	1062500~415000
Variable-1/16	500000~781250	531250~707500
Variable-1/32	250000~390625	265625~353750
Variable-1/64	125000~195312	132813~176875

Can be set by 1 kbit/s step

Response Data

<integer>

Example of Use

To set bit rate for ED to 8.5 Gbit/s:

```
:INP:BITR 8500000
```

To query bit rate for ED:

```
:INP:BITR?
>8500000
```

:INPut:BITRate:DIVRate

Function

This command queries the clock divide ratio of the error detector.

Syntax

```
:INPut:BITRate:DIVRate?
```

Response Data

<character>:

<character>	Setting value	<character>	Setting value
1_1	Rate 1/1	1_16	Rate 1/16
1_2	Rate 1/2	1_32	Rate 1/32
1_4	Rate 1/4	1_64	Rate 1/64
1_8	Rate 1/8		

Example of Use

```
:INP:BITR:DIVR?
>1_2
```

:INPut:BITRate:STANdard

Function

This command sets and queries the bit rate standard for the ED.

Syntax

```
:INPut:BITRate:STANdard <string>
:INPut:BITRate:STANdard?
```

The following table shows the character strings used in <string>.

Table 4.4.2-4 Character Strings for Setting Bit Rate Standard

<string>	Specifications
"10G_FC"	10GFC (10.518G)
"10G_FC_FEC"	10GFC FEC (11.3168G)
"10G_LAN"	10GbE LAN/PHY (10.3125G)
"10G_OTU1E"	10GbE OTU1e (11.049G)
"10G_OTU2E"	10 GbE OTU2e (11.095G)
"10G_WAN"	10GbE WAN (9.95328G)
"1G_FC"*	1GFC (1.0625G)
"1GBE"*	Giga bit Ethernet
"2G_FC"*	2GFC (2.125G)
"2GBE"*	2 Giga bit Ethernet
"4G_FC"	4GFC (4.25G)
"8G_FC"	8GFC (8.5G)
"CPRI"*	CPRI (614.4M)
"CPRI-2"*	CPRI×2 (1.2288G)
"CPRI-4"*	CPRI×4 (2.4576G)
"CPRI-5"*	CPRI×5 (3.072G)
"CPRI-10"*	CPRI×10 (6.144G)
"INF"*	Infiniband (2.5G)
"INF5G"	Infiniband×2 (5G)
"INF10G"	Infiniband×4 (10G)
"OBSAIRP3"*	OBSAI (768M)
"OBSAIRP3-2"*	OBSAI×2 (1.536G)
"OBSAIRP3-4"*	OBSAI×4 (3.072G)
"OBSAIRP3-8"*	OBSAI×8 (6.144G)
"OC-12"*	OC-12/STM-4 (622.08M)
"OC-192"	OC-192/STM-64 (9.95328G)
"OC-192FEC"	G.975 FEC (10.664G)
"OC-24"*	OC-24 (1.244G)
"OC-3"*	OC-3/STM-1 (155.22M)
"OC-48"*	OC-48/STM16 (2.488G)
"OTU-1"*	OTU-1 (2.666057G)
"OTU-2"	OTU-2 (10.709G)
"VARIABLE"	Set value at Bit Rate
"VARIABLE-1/2"	Set value at Bit Rate
"VARIABLE-1/4"*	Set value at Bit Rate
"VARIABLE-1/8"*	Set value at Bit Rate
"VARIABLE-1/16"*	Set value at Bit Rate
"VARIABLE-1/32"*	Set value at Bit Rate
"VARIABLE-1/64"*	Set value at Bit Rate

Response Data

<string>:Character strings in Table 4.4.2-4

Example of Use

To set bit rate standard for ED to 10G_FC:

```
:INP:BITR:STAN "10G_FC"
```

To query bit rate standard for ED:

```
:INP:BITR:STAN?
>"10G_FC"
```

Note*

The following bit rate is enabled only when Option 090 is installed.

```
"1G_FC"
"1GBE"
"2G_FC"
"2GBE"
"CPRI"
"CPRI-2"
"CPRI-4"
"CPRI-5"
"CPRI-10"
"OBSAIRP3"
"OBSAIRP3-2"
"OBSAIRP3-4"
"OBSAIRP3-8"
"OC-12"
"OC-24"
"OC-3"
"OC-48"
"VARIABLE-1/4”
"VARIABLE-1/8"
```


:INPut:DATA:ATTFactor

Function

This command sets and queries the external attenuation factor for the ED in dB unit.

Syntax

```
:INPut:DATA:ATTFactor DATA,<numeric >
:INPut:DATA:ATTFactor? DATA
```

< numeric>: 0 to 30 dB/1 dB Step

Response Data

<integer>: 0 to 30 Unit: dB

Example of Use

To set external attenuation factor for ED to 30 dB:

```
:INP:DATA:ATTF DATA,30
```

To query external attenuation for ED:

```
:INP:DATA:ATTF? DATA
>30
```

:INPut:DATA:INTerface

Function

This command sets and queries the connector inputting signal in the ED.

Syntax

```
:INPut:DATA:INTerface DATA|DIFF|OPT|XDATa
:INPut:DATA:INTerface?
```

Parameter	Input Connector
DATA	Data In
XDATa	$\overline{\text{Data}}$ In
DIFF	Data In/ $\overline{\text{Data}}$ In
OPT	O/E Data In (Optical Connector)

Response Data

DATA|DIFF|OPT|XDAT

Example of Use

To set input connector in ED to Data In:

```
:INP:DATA:INT DATA
```

To query input connector in ED:

```
:INP:DATA:INT?
>DATA
```

:INPut:DATA:THReshold

Function

This command sets the input threshold value for the ED in the mV unit. The query returns the input threshold value for the ED.

Syntax

```
:INPut:DATA:THReshold <numeric>
:INPut:DATA:THReshold?
```

<numeric>: Threshold voltage
When ATT is set to the external attenuation, the setting range is as follows:
Maximum value: –85× 10^(ATT/20) Figures under decimals truncated
Minimum value: 85 × 10^(ATT/20) Figures under decimals truncated
Setting resolution: 10^(ATT/20) Figures under decimals rounded

Response Data

<integer>

Example of Use

To set the threshold value to –300 mV:

```
:INP:DATA:THR—300
```

To query the data input threshold value for the port:

```
:INP:DATA:THR?
> —300
```

:INSTrument:PE1:CONDition

Function

This command queries the details of the PPG/ED 1ch condition register.

Syntax

```
:INSTrument:PE1:CONDition?
```

Response Data

<integer>:Total value of condition register bits: 0 to 63 (decimal number)
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	
1 (bit 0)	PLL Unlock
2 (bit 1)	Total Error
4 (bit 2)	Pattern Sync Loss
8 (bit 3)	CR Unlock
16 (bit 4)	Occurs Insertion Error
32 (bit 5)	Occurs Omission Error

Example of Use

```
:INST:PE1:COND?
> 1
```

:INSTrument:PE1[:EVENt]

Function

This command queries the details of the PPG/ED 1ch event register.

Syntax

```
:INSTrument:PE1[:EVENt]?
```

Response Data

<integer>:Total value of event register 0 to 63 (decimal number)
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
1 (bit 0)	PLL Unlock
2 (bit 1)	Total Error
4 (bit 2)	Pattern Sync Loss
8 (bit 3)	CR Unlock
16 (bit 4)	Occurs Insertion Error
32 (bit 5)	Occurs Omission Error

Example of Use

```
:INSTrument:PE1?
>2
```

:INSTrument:PE1:NTRansition

Function

This command sets and queries the transition filter (negative transition) of the PPG/ED 1ch status.

Syntax

```
:INSTrument:PE1:NTRansition <integer>
:INSTrument:PE1:NTRansition?
```

<integer>:Total value of the filter bits 0 to 63 (decimal number)
If the event register is set to 1 when the condition register is changed from 1 to 0, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits (decimal number)

Example of Use

```
:INST:PE1:PTR?
>15
```

:INSTrument:PE1:PTRansition

Function

This command sets and queries the transition filter (positive transition) of the PPG/ED 1ch status.

Syntax

```
:INSTrument:PE1:PTRansition <integer>
:INSTrument:PE1:PTRansition?
```

<integer>:Total value of the filter bits 0 to 63 (decimal number)
If the event register is set to 1 when the condition register is changed from 0 to 1, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits

Example of Use

```
:INST:PE1:PTR?
>3
```

:INSTrument:PE1:RESet

Function

This command initializes the PPG/ED 1ch status event register.

Syntax

```
:INSTrument:PE1:RESet
```

:INSTrument:PE2:CONDition

Function

This command queries the details of the PPG/ED 2ch status condition register.

Syntax

```
:INSTrument:PE2:CONDition?
```

Response Data

<integer>:Total value of the condition register bit 0 to 63 (decimal number)
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit

1 (bit0)	PLL Unlock
2 (bit1)	Total Error
4 (bit2)	Pattern Sync Loss
8 (bit3)	CR Unlock
16 (bit 4)	Occurs Insertion Error
32 (bit 5)	Occurs Omission Error

Example of Use

```
:INST:PE2:COND?
>1
```

:INSTrument:PE2[:EVENt]

Function

This command queries the details of the PPG/ED 2ch event register.

Syntax

```
:INSTrument:PE2[:EVENt]?
```

Response Data

<integer>: Total value of the condition register bit 0 to 63 (decimal number)

For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
1 (bit0)	PLL Unlock
2 (bit1)	Total Error
4 (bit2)	Pattern Sync Loss
8 (bit3)	CR Unlock
16 (bit 4)	Occurs Insertion Error
32 (bit 5)	Occurs Omission Error

Example of Use

```
:INST:PE2?
>1
```

:INSTrument:PE2:NTRansition

Function

This command sets and queries the transition filter (negative transition) of the PPG/ED 2ch status.

Syntax

```
:INSTrument:PE2:NTRansition <integer>
:INSTrument:PE2:NTRansition?
```

<integer>:Total value of the filter bits 0 to 63 (decimal number)

If the event register is set to 1 when the condition register is changed from 1 to 0, the bit is set to 1.

Response Data

<integer>:Total value of the filter bit

Example of Use

```
:INST:PE2:PTR?
>15
```

:INSTrument:PE2:PTRansition

Function

This command sets and queries the transition filter (positive transition) of the PPG/ED 2ch status.

Syntax

```
:INSTrument:PE2:PTRansition <integer>
:INSTrument:PE2:PTRansition?
```

<integer>: Total value of the filter bits 0 to 63 (decimal number)
If the event register is set to 1 when the condition register is changed from 0 to 1, the bit is set to 1.

Response Data

<integer>:Total value of the filter bit

Example of Use

```
:INST:PE2:PTR?
>3
```

:INSTrument:PE2:RESet

Function

This command initializes the PPG/ED 2ch status event register.

Syntax

```
:INSTrument:PE2:RESet
```

:INSTrument:WAV:CONDition

Function

This command queries the details of the EYE/Pulse Scope status condition register.

Syntax

```
:INSTrument:WAV:CONDition?
```

Response Data

<integer>:Total value of the condition register bits
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
1 (bit 0)	CAL alarm (±5°C)
2 (bit 1)	CAL alarm (±2.5°C)
4 (bit 2)	Amplitude alarm
8 (bit 3)	Not used
16 (bit 4)	PLL Unlock

Example of Use

```
:INST:WAV:COND?
>1
```

:INSTrument:WAV[:EVENt]

Function

This command queries the details of the EYE/Pulse Scope status event register.

Syntax

```
:INST:WAV[:EVENt]?
```

Response Data

<integer>:Total value of the event register bits: 0 to 3, 16 to 19
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
1 (bit0)	Temperature alarm
2 (bit1)	Fatal Temperature alarm
4 (bit2)	Amplitude alarm
8 (bit3)	Not used
16 (bit4)	PLL Unlock

Example of Use

```
:INST:WAV?
>1
```

:INSTrument:WAV:NTRansition

Function

This command sets and queries the transition filter (negative transition) of the EYE/Pulse Scope status register.

Syntax

```
:INSTrument:WAV:NTRansition <integer>
:INSTrument:WAV:NTRansition?
```

<integer>: Total value of the filter bits: (decimal number)
If the event register is set to 1 when the condition register is changed from 1 to 0, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits

Example of Use

```
:INST:WAV:NTR?
>1
```

:INSTrument:WAV:PTRansition

Function

This command sets and queries the transition filter (positive transition) of the EYE/Pulse Scope status register.

Syntax

```
:INSTrument:WAV:PTRansition <integer>
:INSTrument:WAV:PTRansition?
```

<integer>: Total value of the filter bits: (decimal number)
If the event register is set to 1 when the condition register is changed from 0 to 1, the bit is set to 1.

Response Data

<integer>: Total value of the filter bits

Example of Use

```
:INST:WAV:PTR?
>3
```

:INSTrument:WAV:RESet

Function

This command initializes the EYE/Pulse Scope status event register.

Syntax

```
:INSTrument:WAV:RESet
```

:INSTrument:XSFP:CONDition

Function

This command queries the condition register details on the XFP/SFP+ status register.

Syntax

```
:INSTrument:XSFP:CONDition?
```

Response Data

<integer>: Total value of the condition register bits 0 to 3: (decimal number)
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
1 (bit 0)	Ready
2 (bit 1)	LOS

Example of Use

```
:INSTrument:XSFP:CONDition?
>0
```

:INSTrument:XSFP[:EVENt]

Function

This command queries the details on the XFP/SFP+ status event register.

Syntax

```
:INSTrument:XSFP[:EVENt]?
```

Response Data

<integer>: Total value of the condition register bits 0 to 3: (decimal number)
For the correspondence table between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
1 (bit 0)	Ready
2 (bit 1)	LOS

Example of Use

```
:INST:XSFP?
>0
```

:INSTrument:XSFP:NTRansition

Function

This command sets and queries the transition filter (negative transition) of the XFP/SFP+ status.

Syntax

```
:INSTrument:XSFP:NTRansition <integer>
:INSTrument:XSFP:NTRansition?
```

<integer>:Total value of the filter bits 0 to 3: (decimal number)
If the event register is set to 1 when the condition register is changed from 1 to 0, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits

Example of Use

```
:INST:XSFP:NTR?
>3
```

:INSTrument:XSFP:PTRansition

Function

This command sets and queries the transition filter (positive transition) of the XFP/SFP+ status.

Syntax

```
:INSTrument:XSFP:PTRansition <integer>
:INSTrument:XSFP:PTRansition?
```

<integer>:Total value of the filter bits 0 to 3: (decimal number)
If the condition register is set to 1 when the event register is changed from 0 to 1, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits

Example of Use

```
:INST:XSFP:PTR?
>3
```

:INSTrument:XSFP:RESet

Function

This command initializes the XFP/SFP+ status event register.

Syntax

```
:INSTrument:XSFP:RESet
```

:MEASure:AMPLitude

Function

This commad sets the Sampling mode of EYE/Pulse Scope to [Eye] and the measurement function to amplitude and starts the measurement. Before using this command, sets the Accumulation Type of the Setup dialog to [Limited] and the measurement time to [Limit Type].
After passing the set measurement time period, this command queries the measurement results.
The measurement results divided in 11 items are returned: One Level, Zero Level, Eye amplitude, Eye height, Crossing, SNR, Average power (μW), Average power (dBm), Extinction ratio,OMA(mW), and OMA(dBm).
The following items are available for the optical channel. When the optical channel is not set to active channel, “N/A" is returned.

- Average power(μW)
- Average power(dBm)
- Extinction ratio
- OMA(mW)
- OMA(dBm)

The current measurement can be queried using `:CONFigure:MEASure:TYPe?`.

Syntax

```
:MEASure:AMPLitude?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,{<numeric>|"N/A"},{<numeric>|"N/A"},{<numeric>|"N/A"},{<numeric>|"N/A"},{<numeric>|"N/A"}

The output order is as follows:

Item	Unit	Details
One Level	mV/ μW	1 level
Zero Level	mV/ μW	0 level
Eye amplitude	mV/ μW	Eye width
Eye height	mV/ μW	Eye height
Crossing	%	Cross ratio
SNR		Signal-to-noise ratio
Average Power	μW	Averaging power
Average Power	dBm	Averaging power
Extinction Ratio	dB	Extinction ratio
OMA (mW)	mW	Optical amplitude
OMA (dBm)	dBm	Optical amplitude

Example of Use

```
:MEAS:AMPL?
>35.2,-15.1,50.3,46.2,46,10.5,"N/A","N/A","N/A","N/A","N
/A"
```

Note:

The response does not return during the set measurement time.

When using GPIB, read the data after measurement is finished. When reading data after sending a query command, set a longer GPIB timeout time than the measurement time so that a timeout error does not occur.

When using Ethernet, even when the next command is sent during measurement, it is not processed until measurement is finished.

:MEASure:HISTogram:AMPLitude

Function

This commad sets the Sampling mode of EYE/Pulse Scope to [Eye] and the measurement function to the histogram measurement function at the amplitude axis and starts the measurement.
Before using this command, sets the Accumulation Type of the Setup dialog to [Limited] and the measurement time to [Limit Type].
After passing the set measurement time period, this command queries the measurement results.

To read the histogram measurement results of the amplitude individually, use the following command; : FETCh:HISTogram:AMPLitude:xxxx?.
To confirm whether to set the histogram measurement axis to the amplitude, use the following command; : CONFigure:HISTogram:AXIS?.

Syntax

```
:MEASure:HISTogram:AMPLitude?
```

Response d Data

<numeric>,<numeric>,<numeric>,<integer>
The response data is output in the following order.

Item	Unit	Content
Mean	mV/ μW	Mean value
StdDiv	mV/ μW	Standard deviation
Peak to Peak	mV/ μW	Amplitude
Hits		Number of sample in area

Example of Use

```
:MEAS:HIST:AMPL?
>32.1,4.53,28.1,89632
```

Note

The response does not return during the set measurement time.

When using GPIB, read the data after measurement is finished. When reading data after sending a query command, set a longer GPIB timeout time than the measurement time so that a timeout error does not occur.

When using Ethernet, even when the next command is sent during measurement, it is not processed until measurement is finished..

:MEASure:HISTogram:TIME

Function

This command sets the Sampling mode of EYE/Pulse Scope to [Eye] and the measurement function to the histogram measurement function at the time axis and starts the measurement.
Before using this command, sets the Accumulation Type of the Setup dialog to [Limited] and the measurement time to [Limit Type].
After passing the set measurement time period, this command queries the measurement results.
To read the histogram measurement results of the time individually, use the following command; :FETCh:HISTogram:TIME:xxxx?.
To confirm whether to set the histogram measurement axis to the amplitude, use the following command; :CONFigure:HISTogram:AXIS?.
As for the present measurement, use :CONFigure:MEASurement:TYPe?.

Syntax

```
:MEASure:HISTogram:TIME?
```

Response d Data

<numeric>,<numeric>,<numeric>,<integer>
The response data is output in the following order.

Item	Unit	Content
Mean	ps	Mean value
StdDiv	ps	Standard deviation
Peak to Peak	ps	Amplitude
Hits		Number of sample in area

Example of Use

```
:MEAS:HIST:TIME?
>1.53,0.022,0.081,6831
```

Note

The response does not return during the set measurement time.

When using GPIB, read the data after measurement is finished. When reading data after sending a query command, set a longer GPIB timeout time than the measurement time so that a timeout error does not occur.

When using Ethernet, even when the next command is sent during measurement, it is not processed until measurement is finished.

:MEASure:MASK

This command sets the Sampling mode of EYE/Pulse Scope to [Eye] and the measurement function to the mask test and starts the measurement. Before using this command, sets the Accumulation Type of the Setup dialog to [Limited] and the measurement time to [Limit Type].
After passing the set measurement time period, this command queries the measurement results.
The measurement result can be queried
using :FETCh:MASK:MEASurement ?.
To query specific data, use :FETCh:MASK:SAMPles:xxxx?.

Syntax

```
:MEASure:MASK?
```

Response Data

<integer>,<integer>
The output order is as follows:

Item	Unit	Details
Total Samples	—	Total sample count
Failed	—	Sample count in the mask area

Example of Use

```
:MEAS:MASK?
>16831,0
```

Note

The response does not return during the set measurement time.

When using GPIB, read the data after measurement is finished. When reading data after sending a query command, set a longer GPIB timeout time than the measurement time so that a timeout error does not occur.

When using Ethernet, even when the next command is sent during measurement, it is not processed until measurement is finished.

:MEASure:MASK:MARGin

When the measurement function of EYE/Pulse Scope is set to mask test, this command measures the mask margin. Also, this command queries the measurement result.

Syntax

```
:MEASure:MASK:MARGin?
```

Response Data

<integer>:–100~100 Unit %

Example of Use

```
:MEAS:MASK:MARG?
>12
```

:MEASure:TIME

Function

This command sets the Sampling mode of EYE/Pulse Scope to [Eye] and the measurement function to the time and starts the measurement.
Before using this command, sets the Accumulation Type of the Setup dialog to [Limited] and the measurement time to [Limit Type].
After passing the set measurement time period, this command queries the measurement results.
The measurement result can be queried using :FETCh:TIME:MEASurement?. To :OUTPut:CMU:REFClock query the specific result, use :FETCh:TIME:xxxx?.

Syntax

```
:MEASure:TIME?
```

Response Data

<numeric>,<numeric>,<numeric>,<numeric>,<numeric>,<numeric>
The output order is as follows:

Item	Unit	Details
Jitter p-p	ps	Jitter (Peak to peak)
Jitter RMS	ps	Jitter (RMS)
Rise Time	ps	Rise-time
Fall Time	ps	Fall-time
Eye Width	ps	Eye width
DCD	%	Duty Cycle Distortion

Example of Use

```
:MEAS:TIME?
>66.2,15.3,128.2,133.6,208.6,47.2
```

Note

The response does not return during the set measurement time.

When using GPIB, read the data after measurement is finished. When reading data after sending a query command, set a longer GPIB timeout time than the measurement time so that a timeout error does not occur.

When using Ethernet, even when the next command is sent during measurement, it is not processed until measurement is finished.

:MODule:ID

Function

This command sets and queries the target module for remote control. This command cannot be switched to the screen to be displayed on the display. To switch to the screen to be displayed on the display, use, :DISPlay:ACTive.

Syntax

```
:MODule:ID 1|2|3|4|5|6|7
:MODule:ID?
```

1: PPG/ED 1ch
2: PPG/ED 2ch
3: XFP/SFP+
4: O/E
5: EYE/Pulse Scope
6: Jitter Analysis
7: Transmission Analysis

Response Data

1|2|3|4|5|6|7

Example of Use

To set the EYE/Pulse Scope to the target module for remote control:

```
:MOD:ID 5
```

To query the currently controlled module via the remote operation:

```
:MOD:ID?
>5
```

:OUTPut:BITRate

Function

The bit rate can be set when the PPG settings are satisfied with the following two conditions:

- When bit rate standard is Variable, Variable-1/2, Variable-1/4, Variable-1/8, Variable-1/16, Variable-1/32, or Variable-1/64.
- When Reference CLK is Internal or External-10 MHz

Furthermore, this command queries the bit rate of the PPG.

Syntax

```
:OUTPut:BITRate <numeric>
:OUTPut:BITRate?
```

<numeric> The setting range is as shown in the below table.

Table 4.4.2-5 Parameter Setting Range (With Option 090)

Bit Rate Specifications	MHz	kHz
Variable	8000~12500	8500000~12500000
Variable-1/2	4000~6250	4250000~6250000
Variable-1/4	2000~3125	2125000~3125000
Variable-1/8	1000~1565.25	1062500~1562500
Variable-1/16	500~781.25	531250~781250
Variable-1/32	250~390.625	265625~390625
Variable-1/64	125~195.312	132813~195312

Table 4.4.2-6 Parameter Setting Range (Without Option 090)

Bit Rate Specifications	MHz	kHz
Variable	8500~11320	8500000~11320000
Variable-1/2	4250~5660	4250000~5660000
Variable-1/4	2125~2830	2125000~2830000
Variable-1/8	1062.5~1415	1062500~415000
Variable-1/16	531.25~707.5	531250~707500
Variable-1/32	265.625~353.75	265625~353750
Variable-1/64	132.813~176.875	132813~176875

The unit can be set by :OUTPut:CMU:RESolution.

Response Data

< numeric >

< numeric > For the setting range, refer to Table 4.4.2-5 and Table 4.4.2-6.

Example of Use

```
:OUTP:BITR:8500000
```

:OUTPut:BITRate:DIVRate

Function

This command sets the clock divide ration of PPG when a standard clock of PPG is set to the Ext. Clk In connector of the front panel.
Also, this command queries the clock divide rate of PPG.

Syntax

```
:OUTPut:BITRate:DIVRate <character>
:OUTPut:BITRate:DIVRate?
```

<character>:

<character>	Setting value	<character>	Setting value
1_1	Rate 1/1	1_16	Rate 1/16
1_2	Rate 1/2	1_32	Rate 1/32
1_4	Rate 1/4	1_64	Rate 1/64
1_8	Rate 1/8		

Response Data

<character>

Example of Use

```
:OUTPut:BITRate:DIVRate 1_2
:OUTPut:BITRate:DIVRate?
>1_2
```

:OUTPut:BITRate:OFFSet

Function

This command sets and queries the bit rate offset for the PPG.

Syntax

```
:OUTPut:BITRate:OFFSet <integer>
:OUTPut:BITRate:OFFSet?
```

<integer>: –100~100 Unit ppm
This can be set using 1 ppm step.

Response Data

<integer>: –100~100

Example of Use

```
:OUTPut:BITRate:OFFSet 100
:OUTPut:BITRate:OFFSet?
>100
```

:OUTPut:BITRate:STANdard

Function

This command sets and queries the bit rate standard for the PPG.

Syntax

```
:OUTPut:BITRate:STANdard <string>
:OUTPut:BITRate:STANdard?
```

The character strings used in <string> are listed below.

Table 4.4.2-7 Character Strings for Setting Bit Rate Standard

<string>	Specifications
"10G_FC"	10GFC (10.518G)
"10G_FC_FEC"	10GFC FEC (11.3168G)
"10G_LAN"	10GbE LAN/PHY (10.3125G)
"10G_OTU1E"	10GbE OTU1e (11.049G)
"10G_OTU2E"	10 GbE OTU2e (11.095G)
"10G_WAN"	10GbE WAN (9.95328G)
"1G_FC"	1GFC (1.0625G)
"1GBE"	Giga bit Ethernet
"2G_FC"	2GFC (2.125G)
"2GBE"	2 Giga bit Ethernet
"4G_FC"	4GFC (4.25G)
"8G_FC"	8GFC (8.5G)
"CPRI"	CPRI (614.4M)
"CPRI-2"	CPRI×2 (1.2288G)
"CPRI-4"	CPRI×4 (2.4576G)
"CPRI-5"*	CPRI×5 (3.072G)
"CPRI-10"*	CPRI×10 (6.144G)
"INF"	Infiniband (2.5G)
"INF5G"	Infiniband×2 (5G)
"INF10G"	Infiniband×4 (10G)
"OBSAIRP3"*	OBSAI (768M)
"OBSAIRP3-2"*	OBSAI×2 (1.536G)
"OBSAIRP3-4"*	OBSAI×4 (3.072G)
"OBSAIRP3-8"*	OBSAI×8 (6.144G)
"OC-12"	OC-12/STM-4 (622.08M)
"OC-192"	OC-192/STM-64 (9.95328G)
"OC-192FEC	G.975 FEC (10.664G)
"OC-24"	OC-24 (1.244G)
"OC-3"	OC-3/STM-1 (155.22M)
"OC-48"	OC-48/STM16 (2.488G)
"OTU-1"	OTU-1 (2.666057G)
"OTU-2"	OTU-2 (10.709G)
"VARIABLE"	Set value at Bit Rate
"VARIABLE-1/2"	Set value at Bit Rate
"VARIABLE-1/4"	Set value at Bit Rate
"VARIABLE-1/8"	Set value at Bit Rate
"VARIABLE-1/16"	Set value at Bit Rate
"VARIABLE-1/32"	Set value at Bit Rate
"VARIABLE-1/64"	Set value at Bit Rate

Response Data

<string>:Character strings in Table 4.4.2-7

Example of Use

To set the bit rate standard for the PPG to 1G_FC:

```
:OUTP:BITR:STAN "1G_FC"
```

Note*

The following bit rate is enabled only when Option 090 is installed.

"CPRI-5"
"CPRI-10"
"OBSAIRP3"
"OBSAIRP3-2"
"OBSAIRP3-4"
"OBSAIRP3-8"

:OUTPut:CLOCk:FREQuency

Function

The bit rate can be set when the PPG settings are satisfied with the following two conditions:

- When bit rate standard is variable, Variable-1/2, Variable-1/4, Variable-1/8, Variable-1/16, Variable-1/32, or Variable-1/64.
- When Reference CLK is Internal or External-10 MHz

Furthermore, this command queries the PPG bit rate.

Syntax

```
:OUTPut:CLOCk:FREQuency <numeric>
:OUTPut:CLOCk:FREQuency?
```

For the setting range of < numeric >, refer to Table 4.4.2-5 and Table 4.4.2-6 for :OUTPut:BITRate.

This can be set by :1 kbit/s step.
The unit can be set by :OUTPut:CMU:RESolution.

Response Data

<integer>
For the setting range of <integer>, refer to the item of the kHz unit in Table 4.4.2-5 and Table 4.4.2-6.

Example of Use

To set the PPG bit rate to 8500000 kbit/s:

```
:OUTP:CLOC:FREQ 8500000
```

To query the PPG bit rate:

```
:OUTP:CLOC:FREQ?
>8500000
```

:OUTPut:CLOCk:OFFset:PPM

Function

This command sets and queries the bit rate offset for the PPG.

Syntax

```
:OUTPut:CLOCk:OFFset:PPM <numeric>
:OUTPut:CLOCk:OFFset:PPM?
```

<numeric>: –100 to 100 Unit ppm
This can be set by 1 ppm step.

Response Data

<integer>: –100~100

Example of Use

To set the bit rate offset of the PPG to 0 ppm:

```
:OUTP:CLOC:OFF:PPM 0
```

To query the bit rate offset of the PPG:

```
:OUTP:CLOC:OFF:PPM?
>0
```

:OUTPut:CLOCk:OPERation

Function

This command sets and queries the bit rate standard for the PPG.

Syntax

```
:OUTPut:CLOCk:OPERation <string>
:OUTPut:CLOCk:OPERation?
```

For the character strings using in <string>, refer to: :OUTPut:BITRate:STANdard in Table 4.4.2-7.

Response Data

<string>:Character strings in Table 4.4.2-7

Example of Use

To set the synthesizer operation to Variable:

```
:OUTP:CLOC:OPER VARIABLE

:OUTP:CLOC:OPER?
>VARIABLE
```

:OUTPut:CMU:EXTClock

Function

This command sets the input connector of the reference clock when the reference clock of the PPG/ED is the external clock.
Furthermore, the query returns the input connector of the external reference clock for the PPG/ED.

Syntax

```
:OUTPut:CMU:EXTClock 10M|1_16
```

:OUTPut:CMU:EXTClock?

10M External 10 MHz Input connector on the rear-panel
1_16 Ext. Clk In connector on the front-panel

Response Data

10M|1_16

Example of Use

To set Ext. Clk In connector on the front panel to the external clock input:

```
:OUTP:CMU:EXTC 1_16
```

To query external clock input connector:

```
:OUTP:CMU:EXTC?
>1_16
```

:OUTPut:CMU:FREQuency

Function

This command sets and queries the PPG frequency.
This frequency is the same as the bit rate (kbit/s) displayed on the PPG screen.

Syntax

```
:OUTPut:CMU:FREQuency <numeric>
:OUTPut:CMU:FREQuency?
```

For the setting range of < numeric >, refer to Table 4.4.2-5 and Table 4.4.2-6 for :OUTPut:BITRate.

Response Data

< numeric >
For the setting range of < numeric >, refer to Table 4.4.2-5 and Table 4.4.2-6.

Example of Use

To set the bit rate of the PPG to 10 Gbit/s:
(Before sending this message, set the unit to kHz using :OUTPut:CMU:RESolution.)

```
:OUTP:CMU:FREQ 10000000
```

To query the bit rate of the PPG:

```
:OUTP:CMU:FREQ?
>10000000
```

:OUTPut:CMU:REFClock

Function

This command sets the reference clock of the PPG/ED to the internal or external clock.
Also, the query responses whether to set the reference clock of the PPG/ED to the internal or external clock.

Syntax

```
:OUTPut:CMU:REFClock
INTernal|EXTernal|CH1External|CH2External|SYNChronize
:OUTPut:CMU:REFClock?
```

INTernal Uses internal clock
EXTernal Uses external clock CH1External CH1: External clock, CH2: Internal clock
CH2External CH1: Internal clock, CH2: External clock,
SYNChronize CH1: Internal clock, CH2: External clock,

CH1External and CH2External can be set if the PPG2 already exists.
SYNChronize can be set if the PPG2 already exists and the Option 052 is added.

Response Data

INT|EXT|CHIE|CH2E| SYNC

Example of Use

To set the reference clock to the internal clock:

```
:OUTP:CMU:REFC INT
```

To query the reference clock settings:

```
:OUTP:CMU:REFC?
>INT
```

:OUTPut:CMU:RESolution

Function

This command sets the frequency unit for the following message parameter.

:OUTPut:CLOCk:FREQuency
:OUTPut:CMU:FREQuency
:OUTPut:BITRate

Furthermore, this command queries the frequency unit for the message parameter.

The unit of the bit rate for the PPG/ED panel does not vary depending on this message.

Syntax

```
:OUTPut:CMU:RESolution KHZ|MHZ
:OUTPut:CMU:RESolution?
```

KHZ Sets the unit to kHz or kbit/s
MHZ Sets the unit to MHz or Mbit/s

Response Data

KHZ|MHZ

Example of Use

To set the bit rate setting unit to kbit/s using the remote control command:

```
:OUTP:CMU:RES KHZ
```

To query the bit rate setting unit using the remote control command:

```
:OUTP:CMU:RES?
>KHZ
```

:OUTPut:DATA:AMPLitude

Function

This command sets the signal amplitude output to the Data Out and $\overline{\text{Data}}$ Out of the PPG in the V unit.
Also, the query returns the output amplitude of the PPG.

Syntax

```
:OUTPut:DATA:AMPLitude DATA,<numeric>
:OUTPut:DATA:AMPLitude? DATA
```

<numeric> Range 0.10 to 0.80 Unit V, 0.01V Step

Response Data

<numeric> Range 0.10 to 0.80

Example of Use

To set the output amplitude of the PPG to 0.5 V:

```
:OUTP:DATA:AMPL DATA,0.5
```

To query the output amplitude of the PPG:

```
:OUTP:DATA:AMPL? DATA
>0.5
```

:OUTPut:DATA:ATTFactor

Function

This command sets the external attenuator factor of the PPG in the dB unit.

The query returns the external attenuation factor of the PPG.

Syntax

```
:OUTPut:DATA:ATTFactor DATA,<numeric>
:OUTPut:DATA:ATTFactor? DATA
```

<numeric>: 0 to 30 dB, 1 dB Step

Response Data

<integer>: 0 to 30 Unit: dB

Example of Use

To set the external attenuation factor of the PPG to 20dB:

```
:OUTP:DATA:ATTF DATA,20
```

To query the external attenuation factor of the PPG:

```
:OUTP:DATA:ATTF? DATA
>20
```

:OUTPut:DATA:OUTPut

Function

This command sets whether to output the signal to the connector of the PPG.

Furthermore, this command sets the $\overline{\text{Data}}$ Out and Data Out connector to output ON/OFF.

The query returns the data output settings of the PPG.

Syntax

```
:OUTPut:DATA:OUTPut 0|1|OFF|ON
:OUTPut:DATA:OUTPut?
```

0|OFF	Signal output OFF
1|ON	Signal output ON

Response Data

0|1

Example of Use

To output signal to the $\overline{\text{Data}}$ Out and Data Out connectors:

```
:OUTP:DATA:OUTP: ON
```

To query signal output settings of $\overline{\text{Data}}$ Out and Data Out connectors:

```
:OUTP:DATA:OUTP?
>1
```

:OUTPut:DATA:RELative

Function

This command queries the output amplitude correcting the external attenuation factor of the PPG.

Syntax

```
:OUTPut:DATA:RELative? DATA
```

Response Data

<numeric>:0.00 to 0.80Unit V

Example of Use

```
:OUTP:DATA:REL? DATA
>0.4
```

:OUTPut:RCLock:SELect

Function

This command sets the reference clock of the PPG/ED to either the internal or external clock.
Also, the query responses whether to set the reference clock of the PPG/ED to either the internal or external clock.

Syntax

```
:OUTPut:RCLock:SELect
INTernal|EXTernal|CH1External|CH2External|SYNChronize
:OUTPut:RCLock:SELect?
```

INTernal	Internal clock
EXTernal	External clock
CH1External	CH1: External clock, CH2: Internal clock
CH2External	CH1:Internal clock, CH2: External clock
SYNChronize	CH1: Internal clock, CH2:External clock

CH1External and CH2External can be set if the PPG2 already exists.
SYNChronize can be set if the PPG2 already exists and the Option 052 is added.

Response Data

INT|EXT|CHIE|CH2E| SYNC

Example of Use

To set the external clock to the reference clock of the PPG:

```
:OUTP:RCL:SEL EXT
```

To query the reference clock settings:

```
:OUTP:RCL:SEL?
>EXT
```

:OUTPut:SYNC:SOURce

Function

This command sets the signal source output to the Sync Output connector and divide ratio.
The query returns the settings of the signal output to the Sync Output connector.

Syntax

```
:OUTPut:SYNC:SOURce <character>
:OUTPut:SYNC:SOURce?
```

The character strings used in the <character>, clock signal source and divide ratio are as follows:

<character>	Clock Source	Divide Ratio
PPG1CLOC1	PPG Channel 1	1
PPG1CLOC2		2
PPG1CLOC4		4
PPG1CLOC8		8
PPG1CLOC16		16
PPG1CLOC64		64
PPG1PATT		Value of Data Length
PPG2CLOC1	PPG Channel 2	1
PPG2CLOC2		2
PPG2CLOC4		4
PPG2CLOC8		8
PPG2CLOC16		16
PPG2CLOC64		64
PPG2PATT		Value of Data Length
ED1CLOC4	ED Channel 1	4
ED1CLOC8		8
ED1CLOC16		16
ED2CLOC4	ED Channel 2	4
ED2CLOC8		8
ED2CLOC16		16

Response Data

PPG1CLOC1 | PPG1CLOC2 | PPG1CLOC4 | PPG1CLOC8 | PPG1CLOC16 | PPG1CLOC64 | PPG1PATT | PPG2CLOC1 | PPG2CLOC2 | PPG1CLOC4 | PPG2CLOC8 | PPG2CLOC16 | PPG2CLOC64 | PPG2PATT | ED1CLOC4 | ED1CLOC8 | ED1CLOC16 | ED2CLOC4 | ED2CLOC8 | ED2CLOC16

Example of Use

To set the output signal to the Sync Output connector and the PPG Channel 1 to 1/16 divide clock:

```
:OUTP:SYNC:SOUR PPG1CLOC16
```

To query the output signal settings to the Sync Output connector:

```
:OUTP:SYNC:SOUR?
> PPG1CLOC16
```

[:SENSe]:ACCUmulation:AVERaging

Function

This command sets and queries the averaging process count of EYE/Pulse Scope.

Syntax

```
[:SENSe]:ACCUmulation:AVERaging <integer>
[:SENSe]:ACCUmulation:AVERaging?
```

<integer>:1~9999

Response Data

<integer>:1~9999

Example of Use

```
:ACCU:AVER?
>1000
```

[:SENSe]:ACCUmulation:LIMit

Function

This command sets the limitation method and the number of limitations of data collection processes when the data collection process of EYE/Pulse Scope is Limited

Also, this command queries the limitation method and the number of limitations of data collection processes of EYE/Pulse Scope.

Syntax

```
[:SENSe]:ACCUmulation:LIMit TIME|SAMPle|WAVeform,
<numeric>
[:SENSe]:ACCUmulation:LIMit?
```

TIME	The data collection is limited at time.
SAMPle	The data collection is limited by the number of samples.
WAVeform	The data collection is limited by the number of waveforms.
<numeric>	When the data collection is limited at time, the time unit is used per second. When the data collection is limited by the number of samples, a number of samples of one million units is used. The range is 1~99999. When the data collection is limited by the number of waveforms, the number of waveforms is used. The range is 1~999999.

The unit (second or one million) need not be specified by the command.
The time restriction is processed to the sample limitation by one million units every second.
When this message is transmitted while correcting the data (when Sampling of the screen is [RUN]), the displayed waveform is deleted and the data correction is done over again.

Response Data

TIME | SAMPle | WAVeform, <integer>

Example of Use

```
:ACCU:LIM?
>TIME,60
```

[:SENSe]:ACCUmulation:PERSistency

Function

This command sets and queries the data display time when the data collection process of EYE/Pulse Scope is Persistency.

Syntax

```
[:SENSe]:ACCUmulation:PERSistency <numeric>
[:SENSe]:ACCUmulation:PERSistency?
```

<numeric> Time to display collected data (seconds)

Response Data

<integer>

Example of Use

```
:ACCU:PERS?
>5
```

[:SENSe]:ACCUmulation:TYPe

Function

This command sets and queries the data collection process of EYE/Pulse Scope.

Syntax

```
[:SENSe]:ACCUmulation:TYPe <character>
[:SENSe]:ACCUmulation:TYPe?
```

Either of following is set in <character>.

NONe	The data collection is not overwritten. When the fresh data is collected, the displayed data is deleted.
INFinite	The data collection is overwritten. The acquired data does not go out of the screen.
LIMited	The data collection is limited by the number of samples and time. When it reaches the limited conditions, the data collection is ended. The acquired data does not go out of the screen.
PERSistency	The data collection is overwritten. After the fixed time, the acquired data goes out of the screen.
AVERaging	The mean value of the collected data is displayed. Only when Sampling Mode is set to Pulse, this can be used.

Response Data

NONe | INFinite | LIMited | PERSistency | AVERage

Example of Use

```
:ACCU:TYP?
>LIMited
```

[:SENSe]:DISPlay:MODE

Function

This command sets the EYE/Pulse Scope display mode to the eye mode or pulse mode.
The query returns the EYE/Pulse Scope display mode.

Syntax

```
[:SENSe]:DISPlay:MODE COHErenteye|EYE|PULSE
[:SENSe]:DISPlay:MODE?
```

COHErenteye	Sets the display to the coherent eye mode.
EYE	Displays eye mode
PULSE	Displays pulse mode

Response Data

COHErenteye | EYE | PULSE

Example of Use

```
:DISP:MODE EYE
```

[:SENSe]:EYEPulse:PRINt:COPY

Function

The file of the EYE/Pulse Scope waveform screen is saved.
<When setting the first argument >
The screen of EYE/Pulse Scope is saved to the screen file by set folder/file names. If the set folder does not exist, create a folder and save the image files.
<When omitting the first argument >
The saving destination is the following folder when omitting the folder/file names.
C:\Program Files\Anritsu\MP2100A\ MX210000A\UserData\Screen Copy

The file format is saved by the second argument. The PNG files are saved when the second argument is omitted,

Syntax

```
[:SENSe]:EYEPulse:PRINt:COPY[<string>,<string>][<string>
,<string>][,{JPEG|PNG}]
```

The first character strings can be set as the file name.
The second character strings can be set as the folder name.
One of the file name or folder name can be omitted.

JPEG:	JPEG Files
PNG:	PNG Files

Example of Use

```
:MOD:ID 5
:EYEP:PRIN:COPY "CPRI_Eye-12","D:¥User¥CPRI",JPEG
```

[:SENSe]:HISTogram:CENTer

Function

This command moves the marker position of the histogram measurement of EYE/Pulse Scope to the center of the screen.

Syntax

```
[:SENSe]:HISTogram:CENTer
```

[:SENSe]:HISTogram:X1|X2

Function

This command sets and queries the position of the marker X1 or X2 for setting the histogram measurement area of EYE/Pulse Scope.

Syntax

```
[:SENSe]:HISTogram:X1|X2 <numeric>
[:SENSe]:HISTogram:X1|X2?
```

<numeric> indicates the time in which marker is displayed. The unit is UI or ps.
The setting range is as follows.

Unit	Range
ps	0~4294967295
UI	0~4294967

Response Data

<numeric>

Example of Use

```
:HIST:X1?
>10050
```

[:SENSe]:HISTogram:Y1|Y2

Function

This command sets and queries the position of the marker Y1 or Y2 for setting the histogram measurement area of EYE/Pulse Scope.

Syntax

```
[:SENSe]:HISTogram:Y1|Y2 <numeric>
[:SENSe]:HISTogram:Y1|Y2?
```

<numeric> indicates the amplitude in which marker is displayed.
For electrical input:mV, For optical input: μW
The setting range is as follows.

Unit	Range
mV	Minimum/maximum values of screen display
μW	Minimum/maximum values of screen display

Minimum value:Offset–Scale*5,
Maximum value:Offset+Scale*5

Response Data

<numeric>

Example of Use

```
:HIST:Y2?
>-60.6
```

[:SENSe]:INPut:ATTenuation:CHA|CHB

Function

This command sets and queries the amount of attenuation for adjusting the EYE/Pulse Scope amplitude scale in the 0.01 dB unit.

Syntax

```
[:SENSe]:INPut:ATTenuation:CHA|CHB <numeric>
[:SENSe]:INPut:ATTenuation:CHA|CHB?
```

<numeric> indicates the attenuation for setting Channel A or Channel B. The setting range is from 0.00 to 30.00.

Response Data

< numeric >

Example of Use

```
:INP:ATT:CHA 20.00
```

[:SENSe]:INPut:CHA|CHB

Function

This command sets and queries the Channel A or B of EYE/Pulse Scope.

Syntax

```
[:SENSe]:INPut:CHA|CHB ON|OFF
[:SENSe]:INPut:CHA|CHB?
```

ON Displays waveform on screen
OFF Does not display waveform on screen

Response Data

ON|OFF

Example of Use

```
:INP:CHA ON
:INP:CHA?
>ON
```

[:SENSe]:INPut:CLKRecovery

Function

This command sets the usage of the EYE/Pulse Scope clock recovery and frequency bandwidth. Also, this command queries the setting information of the EYE/Pulse Scope clock recovery.

Syntax

```
[:SENSe]:INPut:CLKRecovery OFF|LESS27|85
[:SENSe]:INPut:CLKRecovery?
```

OFF	Sets the clock recovery output to Off
LESS27	Sets the clock recovery output to On and frequency to 0.1~2.7 GHz
85	Sets the clock recovery output to On and frequency to 8.5~12.5 GHz

Response Data

OFF|LESS27|85

Example of Use

```
:INP:CLKR 98
```

[:SENSe]:INPut:FILTer

Function

This command sets the filter of the filter bank option.
The query responses the filter of the filter bank option.

Syntax

```
[:SENSe]:INPut:FILTer <integer>
[:SENSe]:INPut:FILTer?
```

Table 4.4.2-8 Number of Filter Specifications

<integer>	Specifications	<integer>	Specifications
0	None	16	Infiniband Optical x2
1	2GFC	17	Infiniband Optical x4
2	4GFC	18	OC-3/STM-1
3	8GFC	19	OC-12/STM-4
4	10GFC	20	OC-24
5	10GbE_WAN	21	OC-48/STM-16
6	10GbE_LAN/PHY	22	OTU-1
7	OC192/STM64	23	CPRI
8	G975 FEC	24	CPRI×2
9	OTU-2	25	CPRI×4
10	1GFC	26	CPRI×5
11	10GFC FEC	27	CPRI×8
12	1GbE	28	CPRI×10
13	2GbE	29	10GBASE-LX4
14	10GbE FEC	30	10GFC-LX4
15	Infiniband Optical	31	XAUI Optical×2

Response Data

<integer>:0～31

Example of Use

To set the filter to OC192/STM64

```
:INP:FILT 7
```

To query the filter settings

```
:INP:FILT?
>7
```

[:SENSe]:INPut:WAVLength

Function

This command sets and queries the wavelength of the O/E converter.

Syntax

```
[:SENSe]:INPut:WAVLength 850|1310|1550
[:SENSe]:INPut:WAVLength?
```

850 Sets wavelength to 850 nm
1310 Sets wavelength to 1310 nm
1550 Sets wavelength to 1550 nm

Response Data

850|1310|1550

Example of Use

To set the O/E converter wavelength to 1550 nm

```
:INP:WAVL 1550
```

To query the O/E converter wavelength

```
:INP:WAVL?
> 1550
```

:SENSe:MEASure:ASTate

Function

This command queries the measurement status for all modules (ED_1Ch, ED_2Ch, and EYE/Pulse Scope).

Syntax

```
:SENSe:MEASure:ASTate?
```

Response Data

0|1

0 Measurement stops for all modules.
1 At least one module, measurement in progress

Example of Use

```
:SENS:MEAS:AST?
>0
```

:SENSe:MEASure:ASTP

Function

This command stops the measurement for all modules (ED_1Ch, ED_2Ch, and EYE/Pulse Scope).

Syntax

```
:SENSe:MEASure:ASTP
```

:SENSe:MEASure:ASTRt

Function

This command starts the measurement for all modules (ED_1Ch, ED_2Ch, and EYE/Pulse Scope).

Syntax

```
:SENSe:MEASure:ASTRt
```

:SENSe:MEASure:EALarm:ELAPsed

Function

This command queries the measurement elapsed time during bit error measurement.

Syntax

```
:SENSe:MEASure:EALarm:ELAPsed?
```

Response Data

<integer>,<integer>,<integer>,<integer>

The elapsed time is output in the following order: day, hour, min, and second.

The range of the <integer> is as follows:

Item	Range
day	0 to 9
hour	0 to 23
min	0 to 59
second	0 to 59

Example of Use

To query the measurement elapsed time of the ED:

```
SENS:MEAS:EAL:ELAP?
>0,0,2,10
```

:SENSe:MEASure:EALarm:MODE

Function

This command sets and queries the measurement method of the ED.

Syntax

```
:SENSe:MEASure:EALarm:MODE REPeat|SINGle|UNTimed
:SENSe:MEASure:EALarm:MODE?
```

REPeat	Performs measurement repeatedly in the set interval at Gating Time
SINGle	Performs single measurement in the set time at Gating Time
UNTimed	Finishes measurement using panel operation or continues measurement until the :SENSe:MEASure:STOP is set.

Response Data

REP | SING | UNT

Example of Use

To set the measurement method of the ED to the Repeat:

```
:SENSe:MEAS:EAL:MODE REP
```

To query the measurement method of the ED:

```
:SENS:MEAS:EAL:MODE?
> REP
```

:SENSe:MEASure:EALarm:PERiod

Function

This command sets and queries the measurement period of the ED.

Syntax

```
:SENSe:MEASure:EALarm:PERiod
<integer>,<integer>,<integer>,<integer>
:SENSe:MEASure:EALarm:PERiod?
```

The measurement period parameter is set in the following order: day, hour, min, and second.
The setting range of the <integer> is as follows:

Item	Range
day	0 to 9
hour	0 to 23
min	0 to 59
second	0 to 59

Response Data

<integer>,<integer>,<integer>,<integer>
The measurement period is output in the following order: day, hour, min, and second.

Example of Use

To set the measurement period of the ED to 0 day 0 hour 1 min 0 second:

```
:SENSe:MEASure:EALarm:PERiod 0,0,1,0
```

To query the measurement period of the ED:

```
:SENSe:MEASure:EALarm:PERiod?
>0,0,1,0
```

:SENSe:MEASure:EALarm:STARt

Function

This command queries the measurement start time of the ED.

Syntax

```
:SENSe:MEASure:EALarm:STARt?
```

Response Data

<integer>,<integer>,<integer>,<integer>,<integer>,<integer>

The measurement start time is set in the following order: year, month, day, hour, min, and second.

The setting range of the <integer> is as follows:

Item	Range
year	2009 to 2039
month	1 to 12
day	1 to 31
hour	0 to 23
min	0 to 59
second	0 to 59

Example of Use

To query the measurement start time of the ED:

```
SENS:MEAS:EAL:STAR?
>2009,10,5,16,25,40
```

:SENSe:MEASure:EALarm:STATe

Function

This command queries measurement status of the ED.

Syntax

```
:SENSe:MEASure:EALarm:STATe?
```

Response Data

0|1

0 Measurement stops

1 During measurement

Example of Use

To query the measurement status of the ED:

```
SENS:MEAS:EAL:STAT?
>0
```

:SENSe:MEASure:EALarm:STOP

Function

This command queries the measurement end time of the ED.

Syntax

```
:SENSe:MEASure:EALarm:STOP?
```

Response Data

<integer>,<integer>,<integer>,<integer>,<integer>,<integer>

The measurement end time is set in the following order: year, month, day, hour, min, and second.

When Gated Cycle is Untimed, all measurement items are set to 0.

The setting range of the <integer> is as follows:

Item	Range
year	0|2009~2039
month	0~12
day	0~31
hour	0~23
min	0~59
second	0~59

Example of Use

To query the measurement end time when Gated Cycle is Single or Repeat:

```
SENS:MEAS:EAL:STOP?
>2009,10,5,16,25,40
```

To query the measurement end time when Gated Cycle is Untimed:

```
SENS:MEAS:EAL:STOP?
>0,0,0,0,0,0
```

:SENSe:MEASure:EALarm:TIMed

Function

This command queries the measurement remaining time until the ED completes the measurement.

Syntax

```
:SENSe:MEASure:EALarm:TIMed?
```

Response Data

<integer>,<integer>,<integer>,<integer>

The measurement remaining time is output in the following order: day, hour, min, and second.

The setting range of the <integer> is as follows:

Item	Range
day	0 to 9
hour	0 to 23
min	0 to 59
second	0 to 59

Example of Use

To query the remaining time until the ED completes the measurement:

```
SENS:MEAS:EAL:TIM?
>0,0,0,50
```

:SENSe:MEASure:STARt

Function

This command starts the ED measurement.
When the ED is measured, clear the data during measurement and restart the measurement.

Syntax

```
:SENSe:MEASure:STARt
```

:SENSe:MEASure:STOP

Function

This command stops the ED measurement.

Syntax

```
:SENSe:MEASure:STOP
```

:SENSe:MMEMory:PATTern:RECall

Function

This command reads the pattern data used in the test pattern of the ED from the file.

Syntax

```
:SENSe:MMEMory:PATTern:RECall <file_name>,{BIN|TXT}
```

<file_name> File name including extension
When specifying the drive and file, the file path is included.
BIN Binary file
TXT Text file

Example of Use

After setting the test pattern to Programmable Pattern using :SENSe:PATTern:TYPE USER, specify the file name and format, and read the pattern file.

```
:SENS:MMEM:PATT:REC "CJPAT.dat",BIN
```

[:SENSe]:OPTion:MAX:SAMPles:NUMber

Function

This command sets and queries the number of data acquired in the data collection at one time per channel of EYE/Pulse Scope.

Syntax

When Sampling mode is set to [EYE]:
[:SENSe]:OPTion:MAX:SAMPles:NUMber
509|1021|2039|4093|8191|16381
When Sampling mode is set to [Coherent Eye] or [Pulse]:
[:SENSe]:OPTion:MAX:SAMPles:NUMber
512|1024|2048|4096|8192|16384

```
[:SENSe]:OPTion:MAX:SAMPles:NUMber?
```

Response Data

When Sampling mode is set to [EYE]:
509|1021|2039|4093|8191|16381
When Sampling mode is [Coherent Eye] or [Pulse]:
512|1024|2048|4096|8192|16384

Example of Use

```
:SENS:OPT:MAX:SAMP:NUM?
>8191
```

:SENSe:PARam:AEXECute

Function

The ED can be measured when setting Gating Time to 10 ms using the remote control.
This command sets multiple parameters at the same time.
When setting Gating Time to 10 ms at the current setting condition executing the measurement results, the query returns the measurement results.

Syntax

```
:SENSe:PARam:AEXECute 0|1,0|1,0|1,0|1, <string>,
<integer>,<integer>,<character>,<numeric>
:SENSe:PARam:AEXECute?
```

This message sets nine parameters.

No.	Parameter	Explanation
1	0|1	Set whether to enable the parameter settings of PPG-1ch. 0 OFF, 1 ON
2	0|1	Set whether to enable the parameter settings of PPG-2ch. 0 OFF, 1 ON
3	0|1	Set whether to enable the parameter settings of ED-1ch 0 OFF, 1 ON
4	0|1	Set whether to enable the parameter settings of ED-2ch. 0 OFF, 1 ON
5	<string>	Set bit rate specifications using "Table 4.4.2-4 Character Strings for Setting Bit Rate Standard".
6	<integer>	When setting Variable, Variable1/2, Variable-1/4, Variable-1/8, Variable-1/16, Variable-1/32, Variable-1/64 to the fifth parameter, the bit rate can be set in kbit/s unit. For details, refer to the parameter setting range in Table 4.4.2-9 and Table 4.4.2-10.
7	<integer>	As well as the sixth parameter, when setting Variable, the bit rate offset is set in ppm unit. Range –100 to +100
8	<character>	The test parameter is set using following characters. PRBS7|PRBS9|PRBS15|PRBS23|PRBS31|USER
9	<numeric>	The signal amplitude is set in V unit. Range 0.1 to 0.8

Setting example (Parameter 1 to 4)

1) When setting PPG/ED 1ch:
:SENSe:PARam:AEXECute 1,0,1,0, • • • • •

2) When setting PPG/ED 2ch:
:SENSe:PARam:AEXECute 0,1,0,1, • • • • •

3) When setting PPG/ED 1ch and PPG/ED 2ch:
:SENSe:PARam:AEXECute 1,1,1,1, • • • • •

Response Data

Response Data has the following 8 parameters.
For the parameter format, refer to the :CALCulate:DATA:EALarm message in Table 4.4.2-1.

No.	Parameter	Format	Explanation
1	<string>	Form2 decimal point type	ED1 error ratio (ER)
2	<string>	Form1 integer type	ED1 error count (EC)
3	<string>	Form1 integer type	ED1 pattern sync alarm "Not Occur": not alarm occurs "Occur": alarm occurs
4	<string>	Form1 integer type	ED1 pattern sync alarm Equal to the third parameter
5	<string>	Form2 decimal point type	ED2 error ratio (ER)
6	< string >	Form1 integer type	ED2 error count (EC)
7	<string>	Form1 integer type	ED2 signal detector alarm Equal to the third parameter
8	<string>	Form1 integer type	ED2 pattern sync alarm Equal to the third parameter

Example of Use

The measurement parameters of the PPG Channel 1 and 2 and ED Channel 1 and 2 are set.

Set the tracking of ED Channel 1 to OFF.

```
:MOD:ID 1
:SENS:PAR:TRAC OFF
```

Set the tracking of ED Channel 2 to OFF.

```
:MOD:ID 2
:SENS:PAR:TRAC OFF
```

To set the bit rate to 8500000 bit/s, the bit rate offset to 10 ppm, the test pattern to PRBS 2^15−1, and the amplitude to 0.8 V:

```
:SENS:PAR:AEXEC 1,1,1,1,"VARIABLE ",8500000,10,PRBS15,0.80
```

When measuring at Gating Time 10 ms, the re-drawing screen process is stopped.

:SYST:DISP:RES OFF

To perform the Gating Time:10 ms measurement and return the measurement result:

```
:SENS:PAR:AEXEC?
>"0.0000E-07","0","0","1.6000E-07","2","0"
```

If the measurement of Gating Time:10 ms is executed under the following conditions, an execution error occurs.

The re-drawing screen process is turned on.
During the measurement
While transmittin the user-pattern

:SENSe:PARam:TRACking

Function

This command sets and queries the tracking of the ED.

Syntax

```
:SENSe:PARam:TRACking 0|1|OFF|ON
:SENSe:PARam:TRACking?
```

0 OFF
1 ON

Response Data

0|1

Example of Use

To set the ED tracking to On

```
:SENS:PAR:TRAC 1
```

To query the ED tracking settings

```
:SENS:PAR:TRAC?
>1
```

:SENSe:PATTern:DATA:LENGth

Function

This command queries the pattern length when the test pattern of the ED is Programmable Pattern.

Syntax

```
:SENSe:PATTern:DATA:LENGth?
```

Response Data

<integer>:2 to 1305600

Example of Use

```
:SENS:PATT:DATA:LENG?
>16384
```

:SENSe:PATTern:LOGic

Function

This command sets and queries pattern logic (negative/positive logic) of the ED.

Syntax

```
:SENSe:PATTern:LOGic POSitive| NEGative
:SENSe:PATTern:LOGic?
```

POSitive	Positive logic
NEGative	Negative logic

Response Data

POS|NEG

Example of Use

To set the pattern logic of the ED to the positive logic:

```
:SENS:PATT:LOG POS
```

To query the pattern logic of the ED:

```
:SENS:PATT:LOG?
> POS
```

:SENSe:PATTern:SYNC:ASYNc

Function

This command sets whether to automatically perform resynchronization (Auto Sync) for the pattern of the ED.
The query responses the settings of the resynchronization (Auto Sync) procedures for the pattern of the ED.

Syntax

```
:SENSe:PATTern:SYNC:ASYNc 0|1|OFF|ON
:SENSe:PATTern:SYNC:ASYNc?
```

0|OFF	Auto SYNC OFF: Not perform resynchronization (Auto Sync)
1|ON	Auto SYNC ON: Perform resynchronization (Auto Sync)

Response Data

0|1

Example of Use

To set resynchronization (Auto Sync) for the pattern of the ED:

```
:SENS:PATT:SYNC:ASYN ON
```

To query the settings of the resynchronization (Auto Sync) procedures for the pattern of the ED:

```
:SENS:PATT:SYNC:ASYN?
>1
```

:SENSe:PATTern:SYNC:FPOSition

Function

This command sets and queries the frame synchronization start position when the test pattern of the ED is Programmable Pattern.

Syntax

```
:SENSe:PATTern:SYNC:FPOSition <numeric>
:SENSe:PATTern:SYNC:FPOSition?
```

<numeric>: 1~Data Length-64 bits/1 bit Step

Response Data

<integer>: 1~Data Length-64 bits/1 bit Step

Example of Use

To set the frame synchronization start position to bit 65 when the synchronization mode is set to FRAMe (frame detection ON):

```
:SENSe:PATT:SYNC:FPOS 65
```

To query the header position of the bit string

```
:SENS:PATT:SYNC:FPOS?
> 65
```

:SENSe:PATTern:SYNC:PSMode

Function

This command sets and queries the synchronization mode for the test pattern when the test pattern of the ED is Programmable Pattern.

Syntax

```
:SENSe:PATTern:SYNC:PSMode |FRAMe|NORMal
:SENSe:PATTern:SYNC:PSMode?
```

FRAMe	SYNC Control ON: Sets pattern sync control
NORMal	SYNC Control OFF: Does not set pattern sync control

Response Data

FRAM|NORM

Example of Use

To set the pattern sync of the ED to ON:

```
:SENS:PATT:SYNC:PSM FRAM
```

To query the settings of the pattern sync of the ED:

```
:SENS:PATT:SYNC:PSM?
>FRAM
```

:SENSe:PATTern:SYNC:THReshold

Function

This command sets and queries the synchronization detection threshold for resynchronization.

Syntax

```
:SENSe:PATTern:SYNC:THReshold <character>
:SENSe:PATTern:SYNC:THReshold?
```

The character strings set in <character> and the corresponding screen display to the Sync Threshold are as follows:

<character>	Screen display
INT	INT
E_2	1E−2
E_3	1E−3
E_4	1E−4
E_5	1E−5
E_6	1E−6
E_7	1E−7
E_8	1E−8

Response Data

<character>:INT | E_2 | E_3 | E_4 | E_5 | E_6 | E_7 | E_8

Example of Use

To set the synchronization detection threshold of the ED to INT:

```
:SENS:PATT:SYNC:THR INT
```

To query the synchronization detection threshold of the ED:

```
:SENS:PATT:SYNC:THR?
>INT
```

:SENSe:PATTern:TYPE

Function

This command sets and queries the test pattern of the ED.

Syntax

```
:SENSe:PATTern:TYPE <character>
:SENSe:PATTern:TYPE?
```

The character strings setting in <character> and the corresponding screens are as follows:

<character>	Screen Display
PRBS7	PRBS 2^7−1
PRBS9	PRBS 2^9−1
PRBS15	PRBS 2^15−1
PRBS23	PRBS 2^23−1
PRBS31	PRBS 2^31−1
USER	ProgrammablePattern

Response Data

<character>:PRBS7 | PRBS9 | PRBS15 | PRBS23 | PRBS31 | USER

Example of Use

To set the test pattern of the ED to PRBS2^23-1:

```
:SENSe:PATTern:TYPE PRBS23
```

To query the test pattern of the ED:

```
:SENSe:PATT:TYPE?
>PRBS23
```

To set the test pattern of the ED to ProgrammablePattern

```
:SENSe:PATT:TYPE USER
```

To set the test pattern of the ED to CJPAT.

```
:SENSe:MMEM:PATT:REC "CJPAT.dat",TXT
```

[:SENSe]:PRINt:INVerse

Function

This command sets whether to reverse the screen color of EYE/Pulse Scope and to save it to the screen file.
Also, this command queries the color setting when the screen of EYE/Pulse Scope is saved to the screen file.

Syntax

```
[:SENSe]:PRINt:INVerse 0|1
[:SENSe]:PRINt:INVerse?
```

0 Saves to the screen file by the same color as the screen.
1 Reverses the screen color and saves to the screen file.

Response Data

0|1

Example of Use

```
:PRIN:INV 1
```

[:SENSe]:SAMPles:JUDGe

Function

This command sets the sample point count in the mask area to test the Mask margin of the EYE/Pulse Scope.
The query responses the sample point counts to be set for testing the mask margin.

Syntax

```
[:SENSe]:SAMPles:JUDGe <numeric>
[:SENSe]:SAMPles:JUDGe?
```

<numeric>: Sample point counts in the mask area

Response Data

<integer>

Example of Use

To set the sample point counts in the mask area for the mask margin test to 10:

```
:SENS:SAMP:JUDG 10
```

To query the sample point setting values in the mask area for the mask margin test:

```
:SENS:SAMP:JUDG?
>10
```

[:SENSe]:SAMPling:STATus

Function

This command sets the data collection start and stop of the EYE/Pulse. The query responses the data collection status of the EYE/Pulse Scope.

Syntax

```
[:SENSe]:SAMPling:STATus RUN|HOLD
[:SENSe]:SAMPling:STATus?
```

RUN	Collects the data of the EYE/Pulse Scope and updates the screen display.
HOLD	Collects the data of the EYE/Pulse Scope and holds the screen display.

Response Data

RUN|HOLD

Example of Use

To start the EYE/Pulse Scope data collection

```
:SAMP:STAT RUN
```

[:SENSe]:TIME:ACQClock

Function

This command acquires the clock frequency input to the Trigger Clk In connector of the EYE/Pulse Scope and sets the measurement result to the Clock Rate parameter. Furthermore, the query responses the acquired frequency result.

Syntax

```
[:SENSe]:TIME:ACQClock?
```

Response Data

<integer>:Unit Hz

[:SENSe]:TIME:CLKRate

Function

This command sets the clock rate of the EYE/Pulse Scope. When changing the clock rate, the bit rate is changed to the value multiplexed clock rate by divide ratio.
The query responses the clock rate set in the EYE/Pulse Scope.

Syntax

```
[:SENSe]:TIME:CLKRate <numeric> [GHZ|MHZ|KHZ]
[:SENSe]:TIME:CLKRate?
```

<numeric>:Clock rate
The following unit can be used.
GHZ: GHz
KHZ: kHz
MHz: MHz
When omitting the unit, the unit is fixed to MHz.

Response Data

<numeric> MHz

Example of Use

To set the clock rate of the EYE/Pulse Scope to 10312.5 MHz

```
:TIME:CLKR 10312.5
```

To query the clock rate of the EYE/Pulse Scope

```
:TIME:CLKR?
>10312.50 MHz
```

[:SENSe]:TIME:DATRate

Function

This command sets the bit rate of the EYE/Pulse Scope. When changing the bit rate, the clock rate is changed to the value multiplexed bit rate by divide Ratio.
The query responses the bit rate set to the EYE/Pulse Scope.

Syntax

```
[:SENSe]:TIME:DATRate <numeric> [ Gbps|kbps|Mbps]
[:SENSe]:TIME:DATRate?
```

<numeric>: Bit rate
The following unit can be used.
Gbps: Gbit/s
kbps: kbit/s
Mbps: Mbit/s
When omitting the unit, the unit is fixed to Mbps.

Response Data

<numeric>:Bit rate Unit Mbit/s

Example of Use

To set the bit rate of the EYE/Pulse Scope to 155220 kbit/s

```
:TIME:DATR 155220 kbps
```

To query the bit rate of the EYE/Pulse Scope

```
:TIME:DATR?
>155.220 Mbps
```

[:SENSe]:TIME:DIVRatio

Function

This command sets the clock divide ratio of the EYE/Pulse Scope. When changing the clock divide ratio, either of the bit rate or clock frequency is changed.
The query responses the clock divide ratio currently set in the EYE/Pulse Scope.

Syntax

```
[:SENSe]:TIME:DIVRatio <integer>,{CLKR|DATA}
[:SENSe]:TIME:DIVRatio?
```

<integer> Divide ratio 1 to 64
DATR The bit rate is re-calculated from the divide ratio and clock frequency.
CLKR The clock frequency is re-calculated from the divide ratio and bit rate.

Response Data

<integer> Divide ratio 1~64

Example of Use

To set the 1/16 value of the clock frequency to the bit rate:

```
:TIME:DIVR 16,CLKR
```

[:SENSe]:TIME:PATLength

Function

This command sets and queries the data pattern length using the pulse pattern mode of the EYE/Pulse Scope.

Syntax

```
[:SENSe]:TIME:PATLength <numeric>
[:SENSe]:TIME:PATLength?
```

<numeric>: 1~16777216

Response Data

<integer>: 1~16777216

Example of Use

```
:TIME:PATL 8388607
```

[:SENSe]:TMEMory:CHANnel

Function

This command sets and queries the channel saved as a reference trace of EYE/Pulse Scope.
When saving the reference trace, use [:SENSe]:TMEMory:REFerence:SET.

Syntax

```
[:SENSe]:TMEMory:CHANnel BOTH|CHA|CHB
[:SENSe]:TMEMory:CHANnel?
```

BOTH : Channel A and Channel B
CHA : Channel A
CHB : Channel B

Response Data

BOTH | CHA | CHB

Example of Use

```
:TIME:CHAN CHA
:TIME:CHAN?
>CHA
```

[:SENSe]:TMEMory:REFerence:CLEar

Function

This command deletes the reference trace of EYE/Pulse Scope.

Syntax

```
[:SENSe]:TMEMory:REFErence:CLEar
```

Example of Use

```
:TMEM:REF:CLE
```

[:SENSe]:TMEMory:REFerence:SET

Function

This command saves the trace displayed onEYE/Pulse Scope as the reference trace.

Syntax

```
[:SENSe]:TMEMory:REFErence:SET
```

Example of Use

```
:TMEM:REF:SET
```

:SOURce:MMEMory:PATTern:RECall

Function

This command reads the pattern data used in the test pattern of the PPG from the file.

Syntax

```
:SOURce:MMEMory:PATTern:RECall <file_name>,{BIN|TXT}
```

<file_name> File name including extension
When specifying the drive and file, the file path is included.
BIN Binary file
TXT Text file

Example of Use

After setting the test pattern to USER using : :SOUR:PATT:TYPE USER, specify the file name and format and read the pattern file.

```
:SOUR:MMEM:PATT:REC "CJPAT.dat",BIN
```

:SOURce:OPTical:SIGNal:OUTPut

Function

This command sets and queries the optical output of the optical transceiver (XFP/SFP+).

Syntax

```
:SOURce:OPTical:SIGNal:OUTPut 0|1|OFF|ON
:SOURce:OPTical:SIGNal:OUTPut?
```

0|OFF	Optical output OFF
1|ON	Optical output ON

Response Data

0|1|----

0	Optical output OFF
1	Optical output ON
----	Does not install the optical transceiver.

Example of Use

To set the optical output to ON:

```
:SOUR:OPT:SIGN:OUTP ON
```

To query the optical output settings:

```
:SOUR:OPT:SIGN:OUTP?
>1
```

:SOURce:OPTical:SIGNal:WLENgth

Function

This command queries the wavelength of the optical transceiver (XFP/SFP+).

Syntax

```
:SOURce:OPTical:SIGNal:WLENgth?
```

Response Data

<string>

"xxxx"	xxxx ,character strings, indicating wavelength (Up to four digits) Four characters, right-aligned when digits are 3 or less
"----"	Optical transceiver not installed

Example of Use

```
:SOUR:OPT:SIGN:WLEN?
>" 850"
```

:SOURce:OPTical:XFP:REFClock

Function

This command sets and queries the reference clock of the optical transceiver (XFP).

Syntax

```
:SOURce:OPTical:XFP:REFClock <character>
:SOURce:OPTical:XFP:REFClock?
```

The character strings set in the <character> and the corresponding screen to Reference CLK display are as follows:

<character>	Screen Display
ED1Sync	Sync with ED1
ED2Sync	Sync with ED2
PPG1Sync	Sync with PPG1
PPG2Sync	Sync with PPG2

Response Data

< character >: ED1Sync| ED2Sync| PPG1Sync| PPG2Sync

Example of Use

To set the sync clock of the PPG Channel 1 to the reference clock of the optical transceiver (XFP):

```
:SOUR:OPT:XFP:REF PPG1SYNC
```

To query the reference clock settings of the optical transceiver (XFP):

```
:SOUR:OPT:XFP:REF?
> PPG1Sync
```

:SOURce:OUTPut:ASET

Function

This command sets the signal output of the PPG Channel 1 and 2 and the optical output of the optical transceiver (XFP/SFP+) simultaneously. The query response whether all output settings of the PPG and optical transceiver are set to ON.

Syntax

```
:SOURce:OUTPut:ASET 0|1|OFF|ON
:SOURce:OUTPut:ASET?
```

0|OFF	PPG output and optical output OFF
1|ON	PPG output and optical output ON

Response Data

0 | 1

0 All PPG output and optical output OFF
1 At least, one of PPG output or optical output ON

Example of Use

To set the signal output of the PPG and the optical output of the optical transceiver to ON:

```
:SOURce:OUTPut:ASET ON
```

To query whether to set one or more output settings of the PPG and optical transceiver to ON:

```
:SOURce:OUTPut:ASET?
>1
```

:SOURce:PATTern:DATA:LENGth

Function

This command queries the pattern length when the test pattern of the PPG is Programmable Pattern.

Syntax

```
:SOURce:PATTern:DATA:LENGth?
```

Response Data

<integer>:2~1305600

Example of Use

```
:SOUR:PATT:DATA:LENG?
>16384
```

:SOURce:PATTern:EADDition:RATE

Function

This command sets and queries the error generating rate when the error generating method of the PPG is REPEAT.

Syntax

```
:SOURce:PATTern:EADDition:RATE <character>[,<numeric>]
:SOURce:PATTern:EADDition:RATE?
```

<character>:this is the numeric part of the error insertion rate. Either of the following is set.

<character>	Rate
E_2	10^{-2}
E_3	10^{-3}
E_4	10^{-4}
E_5	10^{-5}
E_6	10^{-6}
E_7	10^{-7}
E_8	10^{-8}
E_9	10^{-9}
E_10	10^{-10}
E_11	10^{-11}
E_12	10^{-12}

<numeric>:This is the mantissa of the error generating rate. 1 is set. When the mantissa is omitted, the omitted value is regarded as 1.

Response Data

<character>,1

Example of Use

To set the error generating rate to 1×10^{-9}:

```
:SOUR:PATT:EADD:RATE E_9,1
```

To query the error generating rate:

```
:SOUR:PATT:EADD:RATE?
E_9,1
```

:SOURce:PATTern:EADDition:SET

Function

This command sets whether to generate a bit error to the test pattern for the PPG.

The query responses the error generating settings for the PPG.

Syntax

```
:SOURce:PATTern:EADDition:SET 0|1|OFF|ON
:SOURce:PATTern:EADDition:SET?
```

0|OFF	Not error occurs
1|ON	Error occurs

Response Data

0|1

Example of Use

To generate the error to the test pattern:

```
:SOUR:PATT:EADD:SET ON
```

To query the error generating settings:

```
:SOUR:PATT:EADD:SET?
>1
```

:SOURce:PATTern:EADDition:SINGle

Function

This command generates single error to the test pattern when the error generating method of the PPG is SINGLE.

Syntax

```
:SOURce:PATTern:EADDition:SINGle
```

Example of Use

To generate single error for the test pattern:

```
:SOUR:PATT:EADD:SING
```

:SOURce:PATTern:EADDition:VARiation

Function

This command sets and queries the error generating method of the PPG.

Syntax

```
:SOURce:PATTern:EADDition:VARiation REPeat|SINGle
:SOURce:PATTern:EADDition:VARiation?
```

REPeat:	Generating error continuously
SINGle:	Generating single error when touching the [Insert Error] button or sending the :SOURce:PATTern:EADDition:SINGle

Response Data

REP|SING

Example of Use

To set the error generating method of the PPG to Repeat:

```
:SOUR:PATT:EADD:VAR REP
```

To query the error generating method of the PPG:

```
:SOUR:PATT:EADD:VAR?
>REP
```

:SOURce:PATTern:LOGic

Function

This command sets and queries the test pattern logic (positive/negative logic) of the PPG.

Syntax

```
:SOURce:PATTern:LOGic NEGative|POSitive
:SOURce:PATTern:LOGic?
```

POSitive	Positive logic
NEGative	Negative logic

Response Data

NEG|POS

Example of Use

To set the test pattern logic of the PPG to the negative logic:

```
:SOUR:PATT:LOG NEG
```

To query the test pattern logic of the PPG:

```
:SOUR:PATT:LOG?
>NEG
```

:SOURce:PATTern:TYPE

Function

This command sets and queries the test pattern of the PPG.

Syntax

```
:SOURce:PATTern:TYPE <character>
:SOURce:PATTern:TYPE?
```

The character strings set in the <character> and the correspondence screen are as follows:

<character>	Screen Display
PRBS7	PRBS 2^7−1
PRBS9	PRBS 2^9−1
PRBS15	PRBS 2^15−1
PRBS23	PRBS 2^23−1
PRBS31	PRBS 2^31−1
USER	Programmable Pattern

Response Data

<character>:PRBS7 | PRBS9 | PRBS15 | PRBS23 | PRBS31 | USER

Example of Use

To set the test pattern of the PPG to PRBS2^31−1:

```
:SOUR:PATT:TYPE PRBS31
```

To query the test pattern of the PPG:

```
:SOUR:PATT:TYPE?
PRBS31
```

:STATus:OPERation:CONDition

Function

This command queries the details of the operation status condition register.

Syntax

```
:STATus:OPERation:CONDition?
```

Response Data

<integer>:Bit total of condition register 0 to 6160
For the correspondence between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Status
16 (bit 4)	During measurement
2048 (bit 11)	During pattern settings
4096 (bit 12)	During initialization

Example of Use

```
:STAT:OPER:COND?
>16
```

:STATus:OPERation:ENABle

Function

This command sets and queries the operation status enable register.

Syntax

```
:STATus:OPERation:ENABle <integer>
:STATus:OPERation:ENABle?
```

<integer>: Bit sum of event enable register (decimal number) 0~6160

Valid bit	Status
16 (bit 4)	During measurement
2048 (bit 11)	During pattern settings
4096 (bit 12)	During initialization

When setting the parameter to 0, all bits are masked.

Response Data

<integer>:0 to 6160 Bit sum of operation status enable register (decimal number)

Example of Use

To read only bit 4 of operation status event register:
(At this time, 2^4=16 is set in the operation status enable register.)

```
:STAT:OPER:ENAB 16
```

To query the value of the operation status enable register:

```
:STAT:OPER:ENAB?
>16
```

:STATus:OPERation[:EVENt]

Function

This command queries the operation status event register.

Syntax

```
:STATus:OPERation[:EVENt]?
```

Response Data

<integer>:Total value of the filter bits0~6160
For the correspondence between register bit and decimal number, refer to Table 2.6.1-1.

Valid bit	Details
16 (bit 4)	During measurement
2048 (bit 11)	During pattern settings
4096 (bit 12)	During initialization

Example of Use

```
:STAT:OPER?
>16
```

:STATus:OPERation:NTRansition

Function

This command sets and queries the transition filter (negative transition) of the operation status register.

Syntax

```
:STATus:OPERation:NTRansition <integer>
:STATus:OPERation:NTRansition?
```

<integer>:Total value of filter bits
If the event register is set to 1 when the condition register is changed from 1 to 0, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits

Example of Use

To set bit 4 of operation status event register to bit 1 when bit 4 of operation status condition register changed from 1 to 0:
(At this time, 2^4=16 is set to the transition filter (negative transition).)

```
:STAT:OPER:NTR 16
```

To query transition filter (negative transition) of operation status event register:

```
:STAT:OPER:NTR?
>16
```

:STATus:OPERation:PTRansition

Function

This command sets and queries the transition filter (positive transition) of the operation status register.

Syntax

```
:STATus:OPERation:PTRansition <integer>
:STATus:OPERation:PTRansition?
```

<integer>:Total value of the filter bits
If the condition register is set to 1 when the event register is changed from 0 to 1, the bit is set to 1.

Response Data

<integer>:Total value of the filter bits

Example of Use

To set bit 11 of operation status event register to bit 1 when bit 11 of operation status condition register changes from 0 to 1:
(At this time, 2^{11}=2048 is set in the transition filter (positive transition).)

```
:STAT:OPER:PTR 2048
```

To query transition filter (positive transition) of operation status event register:

```
:STAT:OPER:PTR?
>2048
```

:STATus:PRESet

Function

This command initializes the following event register and transition filter. All bits of the event register and transition filter (negative transition) are set to 0. All bits of the transition filter (positive transition) are set to 1.

Operation status event register
Operation status transition filter (positive transition)
Operation status transition filter (negative transition)
PPG/ED Ch1 event register
PPG/ED Ch1 transition filter (positive transition)
PPG/ED Ch1 transition filter (negative transition)
PPG/ED Ch2 event register
PPG/ED Ch2 transition filter (positive transition)
PPG/ED Ch2 transition filter (negative transition)
XFP/SFP+ event register
XFP/SFP+ transition filter (positive transition)
XFP/SFP+ transition filter (negative transition)
EYE/Pulse Scope event register
EYE/Pulse Scope transition filter (positive transition)
EYE/Pulse Scope transition filter (negative transition)

Syntax

```
:STATus:PRESet
```

:SYSTem:BEEPer:SET

Function

This command sets and queries the buzzer ON/OFF.

Syntax

```
:SYSTem:BEEPer:SET 0|1|OFF|ON
:SYSTem:BEEPer:SET?
```

0|OFF	Buzzer OFF
1|ON	Buzzer ON

Response Data

0|1

Example of Use

To set buzzer ON:

```
:SYST:BEEP:SET ON
```

To query buzzer setting:

```
:SYST:BEEP:SET?
>1
```

:SYSTem:DATE

Function

This command queries the date of the MP2100A/MP2101A /MP2102A.

Syntax

```
:SYSTem:DATE?
```

Response Data

<integer>,<integer>,<integer>

The data is output in the following order: year, month, and day. The range of the <integer> is as follows:

Item	Range
year	2009 to 2039
month	1 to 12
date	1 to 31

Example of Use

```
:SYST:DATE?
>2009,10,24
```

:SYSTem:DISPlay:ALARm

Function

This command displays and queries the system alarm.

Syntax

```
:SYSTem:DISPlay:ALARm 0|1|ON|OFF
:SYSTem:DISPlay:ALARm?
```

0|OFF	Does not display system alarm
1|ON	Display system alarm

Response Data

0|1

Example of Use

```
:SYST:DISP:ALAR 1

:SYST:DISP:ALAR?
>1
```

:SYSTem:DISPlay:DATA

Function

This command queries the last saved screen file on the screen.

Syntax

```
:SYSTem:DISPlay:DATA?
```

Response Data

<binary_data>

Example of Use

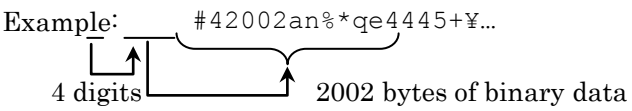
To save the screen display in the screen file

```
:SYST:PRIN:COPY
```

To query the screen file displaying on the screen

```
:SYST:DISP:DATA?
>#541056Avdl-*;E4"as…
```

For the binary data, the head string starts with a sign (#) and continues with data after a numeric value indicating the data length. The character after the sign (#) indicates the number of digits in the data length. The binary data follows the number indicating the data length.



:SYSTem:DISPlay:RESult

Function

This command executes or stops the drawing processing of the measurement result. Stopping the drawing processing makes the response speed of the remote control accelerated. When the drawing processing is stopped, the message “drawing process stopped” is displayed.
When sending the command to execute the drawing processing of the measurement result or touching [Local] on the system menu, the drawing processing of the measurement result is started again.
The query returns the execution and stop settings of the drawing processing.

Syntax

```
:SYSTem:DISPlay:RESult 0|1|OFF|ON
:SYSTem:DISPlay:RESult?
```

0|OFF Stops drawing processing
1|ON Executes drawing processing

Response Data

0|1

0 Stops drawing processing
1 Executes drawing processing

Example of Use

To stop drawing processing of measurement result:

```
:SYST:DISP:RES OFF
```

To query settings of drawing processing:

```
:SYST:DISP:RES?
>0
```

:SYSTem:ERRor

Function

This command queries the error code and error message.

Syntax

```
:SYSTem:ERRor?
```

Response Data

<integer>,<string>

<integer>:－32768 to 32767
0: indicates that no errors and events have occurred.
When error occurred, responses the error code and message in Appendix B.
<string>:This is the error message corresponding to the value of <integer>.
The maximum length of this character strings is 255.

Example of Use

```
:SYST:ERR?
> 0, "No error"
```

:SYSTem:ERRor:HCLear

Function

This command erases indication of the system alarm history.

Syntax

```
:SYSTem:ERRor:HCLear?
```

:SYSTem:ERRor:HISTory

Function

This command queries the system error history.

Syntax

```
:SYSTem:ERRor:HISTory?
```

Response Data

"Not Occurred"|"Occurred"

"Not Occurred": No system error history indicated
"Occurred" : System error history indicated

Example of Use

```
:SYST:ERR:HIST?
> "Not occurred"
```

:SYSTem:INFormation

Function

This command queries product supplier name, model name, serial number, and option.

Syntax

```
:SYSTem:INFormation?
```

Response Data

<character>,<character>, <character>, <character> [,<character>] [,<character>][,<character>]….

For options, refer to *OPT.

Example of Use

```
:SYST:INF?
>Anritsu,MP2100A,6200123456,OPT001,OPT050
```

:SYSTem:INFormation:ERRor

Function

This command queries the system alarm details.

Syntax

```
:SYSTem:INFormation:ERRor?
```

Response Data

{0|1|2|3|4|5|6}[, {2|3|4|5|6}][,{3|4|5|6}][,{4|5|6}][,{5|6}][,6]

0 None
1 PPG/ED Fatal Temperature
2 EYE/Pulse Scope Temperature
3 PPG/ED PLL Unlock
4 Power
5 EYE/Pulse Scope Fatal Temperature
6 PPG/ED Illegal Mode

The system alarm occurring error is displayed by comma-delimited format.

Example of Use

```
:SYST:INF:ERR?
>1,2,3
```

:SYSTem:MEMory:INITialize

Function

This command sets the measurement parameter to the factory default setting.

Syntax

```
:SYSTem:MEMory:INITialize
```

:SYSTem:MMEMory:RECall

Function

This command reads the module settings and measurement result data from the file and reflects the read information to the settings.

Syntax

```
:SYSTem:MMEMory:RECall
<file_name>,{0|1|2|3|4|5|6|7},<character>
```

<file_name>: File name including extension

The second parameter sets the module for setting the read data.
0: All
1: PPG/ED 1ch
2: PPG/ED 2ch
3: XFS/SFP+
4: O/E
5: EYE/Pulse Scope
6: Jitter Analysis
7: Transmission Analysis

<character>:This sets the data types using the following characters.

ALL	All data below
PE1	PPG/ED 1ch setting data
PE2	PPG/ED 2ch setting data
XFP	XFP setting data
SFP	SFP+ setting data
OES	O/E setting data
WFS	EYE/Pulse Scope setting data
JIT	Jitter Analysis Software setting data
TRA	Transmission Analysis Software setting data

Example of Use

To read the setting value of the PPG/ED 1ch from the file:

```
:SYST:MMEM:REC "abc1.PE1",1,PE1
```

:SYSTem:MMEMory:STORe

Function

This command sets the specified module and saves the measurement result data.

When the third parameter is ER1, ER2, WFR, JIT, TAR or WFE, the fourth parameter is valid. The fourth parameter is ignored when the third parameter other than theses is set.

Note:

Note that the setting information cannot be read when changing the saved file name.

Syntax

```
:SYSTem:MMEMory:STORe <file_name>,{0|1|2|3|4|5|6|7},
<character>,{CSV|S2P|TXT|WFE}
```

<file_name>: File name including extension

The second parameter sets the module for setting the read data.
0: All
1: PPG/ED 1ch
2: PPG/ED 2ch
3: XFS/SFP+
4: O/E
5: EYE/Pulse Scope
6: Jitter Analysis
7: Transmission Analysis

<character>: The third parameter sets the data types using the following characters.
ALL All data below
PE1 PPG/ED 1ch setting data
PE2 PPG/ED 2ch setting data
XFP XFP setting data
SFP SFP+ setting data
ER1 ED 1ch measurement result data
ER2 ED 2ch measurement result data
OES O/E setting data
WFS EYE/Pulse Scope data
WFR EYE/Pulse Scope measurement result data
JIT Jitter Analysis Software setting data
JIR Jitter Analysis Software measurement result data
TAS Transmission Analysis Software setting data
TAR Transmission Analysis Software measurement result data (Transmisssion Analysis)
WER Transmission Analysis Software measurement result data (Waveform Estimation)

The third parameter sets the file format.
CSV comma-delimited file
TXT test file
(When omitted) binary file (only when the third parameter is WFR)
S2P S parameter file (only when the third parameter is TAR)
WFE waveform file (only when the third parameter is WER)

The combination of second-to-fourth parameter is listed in the table below.

Second	Third	Fourth
1	ALL|PE1|ED1	CSV|TXT
2	ALL|PE2|ED2	CSV|TXT
3	ALL|SFP|XFP	CSV|TXT
4	ALL|OER	CSV|TXT
5	ALL|WFS	CSV|TXT
	WFR	
6	ALL|JIT|JIR	CSV|TXT
7	ALL|TAS	CSV|TXT
	TAR	S2P|TXT
	WER	WFE

Example of Use

To save the setting data of PPG/ED 1ch:
```
:SYSTem:MMEMory:STORe "ResultTest_PE1.PE1",1,PE1
```
To save the PPG/ED 2ch measurement result to the test file:
```
:SYST:MMEM:STOR "abc1.PE2",2,ER2,TXT
```
To save the EYE/Pulse Scope measurement result using the CSV format:
```
:SYST:MMEM:STOR "scope1.CSV",5,WFR,CSV
```
To save the setting data of Jitter Analysis Software:
```
:SYSTem:MMEMory:STORe "ResultTest_JIT.JIT",6,JIT
```
To save the measurement result of Transmission Analysis Software (Transmission Analysis) in S2P format:
```
:SYSTem:MMEMory:STORe "ResultTest_TAR.s2p",7,TAR,S2P
```
To save the measurement result of Transmission Analysis Software (Waveform Estimation):
```
:SYSTem:MMEMory:STORe "ResultTest_WER.WFE",7,WER,WFE
```

:SYSTem:PRINt:COPY

Function

The file of the EBEARTWave entire screen is saved.
<When setting the first argument>
This command saves the screen display to the screen file with png format by using the set folder/file names.
<When omitting the first argument>
The following folder is the saving destination when omitting the folder/file names.
C:\Program Files\Anritsu\MP2100A\MX210000A\UserData\Screen Copy

The file format is set by the second argument. The PNG files are set when the second argument is omitted.

Syntax

```
:SYSTem:PRINt:COPY [<string>,<string>][,{JPEG|PNG}]
```

The first character strings can be set as the file name.
The second character strings can be set as the folder name.
One of the file name or folder name can be omitted.
JPEG: JPEG Files
PNG: PNG Files

Example of Use

```
:SYST:PRIN:COPY "Sample-010","D:¥User",PNG
```

:SYSTem:TERMination

Function

This command sets the message terminator sent to the control PC from the MP2100A/MP2101A/MP2102A.
Also, this command queries the current terminator setting.

Syntax

```
:SYSTem:TERMination 0|1
:SYSTem:TERMination?
```

0 LF+EOI
1 CR+LF+EOI
LF: Line Field, ASCII code, up to 10 characters
CR: Carriage Return, ASCII code, up to 13 characters
EOI(End or Identify): The data end is detected depending on the hardware signal of the GPIB.

Response Data

0|1

Example of Use

To set the terminator type to LF+EOI:

```
:SYST:TERM 0
```

To query the terminator settings:

```
:SYST:TERM?
>0
```

:SYSTem:TIME

Function

This command queries the time of the MP2100A/MP2101A/ MP2102A.

Syntax

```
:SYSTem:TIME?
```

Response Data

<integer>,<integer>,<integer>

The time setting is output in the following order: hour, min, and second.

The range of the <integer> is as follows:

hour	0 to 23
Second	0 to 59

Example of Use

```
:SYST:TIME?
9,50,39
```

:SYSTem:VERSion

Function

This command queries the SCPI version conforming to the software of the MP2100A/MP2101A/MP2102A.

Syntax

```
:SYSTem:VERSion?
```

Response Data

<version> This displays the specifications issued year and version number.

Example of Use

```
:SYST:VERS?
>1999.0
```

:TRACe[:DATA]:CHANnelA|CHANnelB|CHANnels

Function

This command queries the trace data for the eye pattern mode for one or both sides of EYE/Pulse Scope.
This message is not used when EYE/Pulse Scope is pulse mode or Coherent EYE mode. Send `:TRACe[:DATA]:PREPare` before sending this message.
When `:TRACe[:DATA]:PREPare` is sent, updating of the EYE/Pulse Scope screen stops and the status for receiving the trace data query starts. To release this status, send :TRACe[:DATA]:END after completing reading of the trace data.

Note:

The following changes are made for the MX210000A of Version 3.00.00 or later.

- This message can be used when EYE/Pulse Scope is in pulse mode. However, this message cannot be used when in coherent eye.
- Sending of the command :TRACe[:DATA]:PREPare is not required before sending the data.
- Sending of the command :TRACe[:DATA]:END is not required after sending the data.

Syntax

```
:TRACe[:DATA]:{CHANnelA|CHANnelB|CHANnels}?
```

:TRACe[:DATA]:CHANnelA
This command requests sending of trace data for Channel A.
Before sending this command, always set Channel A to On and Channel B to Off.
:TRACe[:DATA]:CHANnelB
This command requests sending of trace data for Channel B.
Before sending this command, always set Channel B to On and Channel A to Off.

:TRACe[:DATA]:CHANnels
This command requests sending of trace data for Channel A and Channel B. Set both Channel A and Channel B to On.
When the display of the channel querying the trace data is set to Off, the data cannot be returned.

Response Data

- When channel to read not displayed: "Channel Off"
- When reading one channel:
  {CHA | CHB}-<integer>(<numeric>,<numeric>),(<numeric>,<numeric>),(<numeric>,<numeric>),....
- When reading one channel:
  CHA-<integer>(<numeric>,<numeric>),(<numeric>,<numeric>),(<numeric>,<numeric>),.... ,CHB-<integer> (<numeric>,<numeric>),(<numeric>,<numeric>),....

CHA: Channel A
CHB: Channel B
<integer>: Trace data score
(<numeric>,<numeric>): Each time and amplitude

Example of Use

To query the trace data for the Channel A:

```
:TRAC:CHANA?
>CHA-2039(86.0,39.97),(86.0,167.13),...
```

To query the trace data for the Channel A and B

```
:TRAC:CHAN?
CHA-2039(86.0,39.97),(86.0,167.13), ........ ,(285.9,-3.92),CHB-2039(86.0,152.10), ........
```

:TRACe[:DATA]:END

Function

This command terminates the status set by the :TRACe[:DATA]:PREPare command. When executing this command, the normal screen-update mode is returned.
Send this command after reading all the trace data after sending the trace data query.
This command is used in combination with the :TRACe[:DATA]:PREPare command.

Syntax

```
:TRACe:DATA::END
```

:TRACe[:DATA]:PREPare

Function

This command sets the instrument to the status in which the EYE/Pulse Scope trace data can be read via the remote interface.
When this command is sent, updating of the EYE/Pulse Scope waveform screen is stopped.
Send :TRACe[:DATA]:PREPare before
sending :TRACe[:DATA]:CHANnelA|CHANnelB|CHANnels? to query the trace data.
The status set by this command is released by :TRACe[:DATA]:END.

Syntax

```
:TRACe[:DATA]:PREPare CHA|CHB|BOTH
```

CHA: Sets status for reading only Channel A data
Before using this command, Channel A is set to On and Channel B is set to Off.
Send :TRACe[:DATA]:CHANnelA? after this parameter is set.
CHB: Sets status for reading only Channel B data
Before using this command, Channel B is set to On and Channel A is set to Off.
Send : TRACe[:DATA]:CHANnelB? after this parameter is set.
BOTH: Sets status for reading Channel A and B data
Before using this command, Both Channel A and B is set to On.
Send : TRACe[:DATA]:CHANnels? after this parameter is set.

Example of Use

To set the Channel A waveform display to On.
To activate the data reading status of Channel A.
To read out the data of Channel A.
To release the data reading status of Channel A

```
:INPut:CHA ON
:INPut:CHB OFF
:TRACe:PREPare CHA
:TRACe:CHANnelA?
:TRACe:END
```

Appendix A Message Compatibility

The following table shows the compatibility of messages with previous hardware.

✓: Compatible.

*: Partly compatible; occasional errors when sending commands for previous hardware to this instrument.

—: Incompatible; sending commands for previous hardware to this instrument always causes errors.

Table A-1 Message Compatibility

Message	MP1800A	MP1632C	MP1776A
:CALCulate:DATA:EALarm	*	—	—
:CALCulate:DATA:MONitor	✓	—	—
:CALCulate:OPTical:STATus	*	—	—
:CALibrate:AMPLitude	—	—	—
:CALibrate:APPLication	—	—	—
:CALibrate:Cgain	—	—	—
:CALibrate:OEPower	—	—	—
:CALibrate:RESPonsivity	—	—	—
:CALibrate:SYSTem:Cgain	—	—	—
:CONFigure:EXRCorrection	—	—	—
:CONFigure:MASK:MARGin	—	—	—
:CONFigure:MASK:TYPe	—	—	—
:CONFigure:MEASure:AMPTIME{1|2|3|4}	—	—	—
:CONFigure:MEASure:CHANnel	—	—	—
:CONFigure:MEASure:TYPe	—	—	—
:DISPlay:RESult:EALarm:HRESet	✓	✓	✓
:DISPlay:RESult:EALarm:MODE	✓	—	—
:DISPlay:WINDow:GRAPhics:CLEar	—	—	—
:DISPlay:WINDow[:SCALe]:AUTOscale	—	—	—
:DISPlay:WINDow:X[:SCALe]:BITs	—	—	—
:FETCh:AMPLitude:AVEPower	—	—	—
:FETCh:AMPLitude:CROSsing	—	—	—
:FETCh:AMPLitude:EXTRatio	—	—	—
:FETCh:AMPLitude:EYEAmplitude	—	—	—
:FETCh:AMPLitude:EYEHeight	—	—	—
:FETCh:AMPLitude:LEVel:ONE	—	—	—
:FETCh:AMPLitude:LEVel:ZERO	—	—	—
:FETCh:AMPLitude:MEASurement	—	—	—

Table A-1 Message Compatibility (Cont'd)

Message	MP1800A	MP1632C	MP1776A
:FETCh:AMPLitude:SNR	—	—	—
:FETCh:MASK:MEASurement	—	—	—
:FETCh:MASK:SAMPles:FAILed	—	—	—
:FETCh:MASK:SAMPles:FAILed:BOTTom	—	—	—
:FETCh:MASK:SAMPles:FAILed:CENTer	—	—	—
:FETCh:MASK:SAMPles:FAILed:TOP	—	—	—
:FETCh:MASK:SAMPles:TOTal	—	—	—
:FETCh:TIME:DCD	—	—	—
:FETCh:TIME:EYEWidth	—	—	—
:FETCh:TIME:FTIMe	—	—	—
:FETCh:TIME:JITTer:PPeak	—	—	—
:FETCh:TIME:JITTer:RMS	—	—	—
:FETCh:TIME:MEASurement	—	—	—
:FETCh:TIME:TRISe	—	—	—
:INPut:BITRate	—	—	—
:INPut:BITRate:DIVRate	—	—	—
:INPut:BITRate:STANdard	—	—	—
:INPut:DATA:ATTFactor	—	—	—
:INPut:DATA:INTerface	*	—	—
:INPut:DATA:THReshold	—	—	—
:INSTrument:OE:CONDition	—	—	—
:INSTrument:OE[:EVENt]	—	—	—
:INSTrument:OE:NTRansition	—	—	—
:INSTrument:OE:PTRansition	—	—	—
:INSTrument:OE:RESet	—	—	—
:INSTrument:PE1:CONDition	—	—	—
:INSTrument:PE1[:EVENt]	—	—	—
:INSTrument:PE1:NTRansition	—	—	—
:INSTrument:PE1:PTRansition	—	—	—
:INSTrument:PE1:RESet	—	—	—
:INSTrument:PE2:CONDition	—	—	—
:INSTrument:PE2[:EVENt]	—	—	—
:INSTrument:PE2:NTRansition	—	—	—
:INSTrument:PE2:PTRansition	—	—	—
:INSTrument:PE2:RESet	—	—	—
:INSTrument:WAV:CONDition	—	—	—

Table A-1 Message Compatibility (Cont'd)

Message	MP1800A	MP1632C	MP1776A
:INSTrument:WAV[:EVENt]	—	—	—
:INSTrument:WAV:NTRansition	—	—	—
:INSTrument:WAV:PTRansition	—	—	—
:INSTrument:WAV:RESet	—	—	—
:INSTrument:XSFP:CONDition	✓	—	—
:INSTrument:XSFP[:EVENt]	✓	—	—
:INSTrument:XSFP:NTRansition	✓	—	—
:INSTrument:XSFP:PTRansition	✓	—	—
:INSTrument:XSFP:RESet	✓	—	—
:MEASure:AMPLitude	—	—	—
:MEASure:MASK	—	—	—
:MEASure:TIME	—	—	—
:MODule:ID	✓	—	—
:OUTPut:BITRate	—	—	—
:OUTPut:BITRate:DIVRate	—	—	—
:OUTPut:BITRate:OFFSet	—	—	—
:OUTPut:BITRate:STANdard	—	—	—
:OUTPut:CLOCk:FREQuency	✓	✓	—
:OUTPut:CLOCk:OFFset:PPM	✓	—	—
:OUTPut:CLOCk:OPERation	✓	—	—
:OUTPut:CMU:EXTClock	✓	—	—
:OUTPut:CMU:FREQuency	✓	—	—
:OUTPut:CMU:REFClock	✓	—	—
:OUTPut:CMU:RESolution	✓	—	—
:OUTPut:DATA:AMPLitude	✓	✓	—
:OUTPut:DATA:ATTFactor	✓	—	—
:OUTPut:DATA:OUTPut	✓	—	—
:OUTPut:DATA:RELative	—	—	—
:OUTPut:RCLock:SELect	*	*	—
:OUTPut:SYNC:SOURce	—	—	—
[:SENSe]:DISPlay:MODE	—	—	—
[:SENSe]:INPut:FILter	—	—	—
[:SENSe]:INPut:WAVLength	—	—	—
:SENSe:MEASure:ASTate	✓	—	—
:SENSe:MEASure:ASTP	✓	—	—
:SENSe:MEASure:ASTRt	✓	—	—

Table A-1 Message Compatibility (Cont'd)

Message	MP1800A	MP1632C	MP1776A
:SENSe:MEASure:EALarm:ELAPsed	✓	✓	✓
:SENSe:MEASure:EALarm:MODE	✓	✓	✓
:SENSe:MEASure:EALarm:PERiod	✓	✓	✓
:SENSe:MEASure:EALarm:STARt	✓	✓	✓
:SENSe:MEASure:EALarm:STATe	✓	✓	✓
:SENSe:MEASure:EALarm:STOP	✓	✓	✓
:SENSe:MEASure:EALarm:TIMed	✓	✓	✓
:SENSe:MEASure:STARt	✓	✓	✓
:SENSe:MEASure:STOP	✓	✓	✓
:SENSe:MMEMory:PATTern:RECall	—	—	—
:SENSe:PATTern:DATA:LENGth	✓	—	—
:SENSe:PATTern:LOGic	✓	—	—
:SENSe:PATTern:SYNC:ASYNc	✓	✓	✓
:SENSe:PATTern:SYNC:FPOSition	✓	—	—
:SENSe:PATTern:SYNC:PSMode	*	*	*
:SENSe:PATTern:SYNC:THReshold	✓	—	✓
:SENSe:PATTern:TYPE	*	—	—
:SENSe:PARam:AEXECute	—	—	—
:SENSe:PARam:TRACking	—	—	—
:SENSe:SAMPles:JUDGs	—	—	—
[:SENSe]:SAMPling:STATus	—	—	—
[:SENSe]:TIME:ACQClock	—	—	—
[:SENSe]:TIME:CRKRate	—	—	—
[:SENSe]:TIME:DATRate	—	—	—
[:SENSe]:TIME:DIVRatio	—	—	—
[:SENSe]:TIME:PATLength	—	—	—
:SOURce:MMEMory:PATTern:RECall	—	—	—
:SOURce:OPTical:SIGNal:OUTPut	✓	—	—
:SOURce:OPTical:SIGNal:WLENgth	✓	—	—
:SOURce:OPTical:XFP:REFClock	—	—	—
:SOURce:OUTPut:ASET	✓	—	—
:SOURce:PATTern:DATA:LENGth	✓	—	—
:SOURce:PATTern:EADDition:RATE	✓	*	—
:SOURce:PATTern:EADDition:SET	✓	✓	—
:SOURce:PATTern:EADDition:SINGle	✓	✓	—

Table A-1 Message Compatibility (Cont'd)

Message	MP1800A	MP1632C	MP1776A
:SOURce:PATTern:EADDition:VARiation	✓	—	—
:SOURce:PATTern:LOGic	✓	—	—
:SOURce:PATTern:TYPE	*	—	—
:STATus:OPERation:CONDition	✓	✓	—
:STATus:OPERation:ENABle	✓	✓	—
:STATus:OPERation:NTRansition	✓	✓	—
:STATus:OPERation:PTRansition	✓	✓	—
:STATus:OPERation[:EVENt]	✓	✓	—
:STATus:PRESet	✓	✓	—
:SYSTem:BEEPer:SET	—	—	—
:SYSTem:CONDition	✓	—	—
:SYSTem:DATE	✓	—	—
:SYSTem:DISPlay:RESult	✓	—	—
:SYSTem:ERRor	✓	✓	✓
:SYSTem:ERRor:HCLear	—	—	—
:SYSTem:ERRor:HISTory	—	—	—
:SYSTem:INFormation:ERRor	✓	—	—
:SYSTem:MEMory:INITialize	✓	✓	✓
:SYSTem:MMEMory:RECall	—	—	—
:SYSTem:MMEMory:STORe	—	—	—
:SYSTem:ORGanization:HARDware	—	—	—
:SYSTem:PRINt:COPY	✓	✓	✓
:SYSTem:TERMination	✓	✓	—
:SYSTem:TIME	✓	—	—
:SYSTem:VERSion	✓	✓	✓
:TRACe[:DATA]:CHANnelA | CHANnelB | CHANnels	—	—	—
:TRACe[:DATA]:END	—	—	—
:TRACe[:DATA]:PREPare	—	—	—
:TRACe:PREamble	—	—	—

Appendix B Message Code

This appendix explains the code and message responses to the SYSTem:ERRor? query command.

- Command error
- Execution error
- Device unique error

When these errors occur, the standard event status register bit becomes 1. A service request can be generated when an error occurs depending on the setting of the standard event status enable register bit.

When an error occurs, the standard event status register bit that becomes 1 is listed in the table below.

Table B-1 Relationship between Error Number and Standard Event Register

Error Code	Message	Error Name	Standard Event Register Bit
–113	Undefined header	Command error	5
–220	Parameter error	Execution error	4
–310	System error	Device dependant error	3

Command error

Bit 5 of the standard event status register is set when the following errors occur. The errors are generated when the following events occur.

- When sending message not in conformance with syntax described in section 2.5 Message Format
  Example: At typographical error in header
  Header includes 2-byte character

- When sending message not in conformance with Common Commands or Device Unique Commands described in section 4.4 Explanation of Messages.

Execution error

Bit 4 of the standard event status register is set when the following errors occur. The errors are generated when the following events occur.

- When header continuation parameter value out of setting range
  Example: When 850000 set when bit rate setting range is 8500000 to 11320000

- When message cannot be executed in current equipment status
  Example: When sending message for setting EYE/Pulse Scope to instrument without EYE/Pulse Scope function

Device dependant error

The device dependant error is a system error generated by the instrument. Bit 3 of the standard event status register is set when the following errors occur.

- PPG/ED Fatal Temperature
- EYE/Pulse Scope Temperature
- PPG/ED PLL Unlock
- Power
- EYE/Pulse Scope Fatal Temperature
- PPG/ED Illegal Mode

Appendix C BASIC Sample Program

This appendix describes the sample program in Chapter 3 using the BASIC language.

C.1 Sample Program Operating Environment

C.1.1 Setting Sample Program Operating Environment

The sample program operating environment is as follows.

PC

OS: Windows XP Professional Service Pack 2

VISA: NI-VISA Version 4.6

Program tool: Microsoft Visual BASIC 2008 Express Edition

MS9740A Optical Spectrum Analyzer

GPIB Address: 1

IP Address: 198.168.12.10

Subnet Mask: 255.255.255.0

Installing NI-VISA

To use VISA at Visual BASIC 2008, add the following function at installation.

- Development Support .NET Framework 3.5 Language Support
- NI Measurement & Automation Explore —.NET Framework 3.5 Language Support

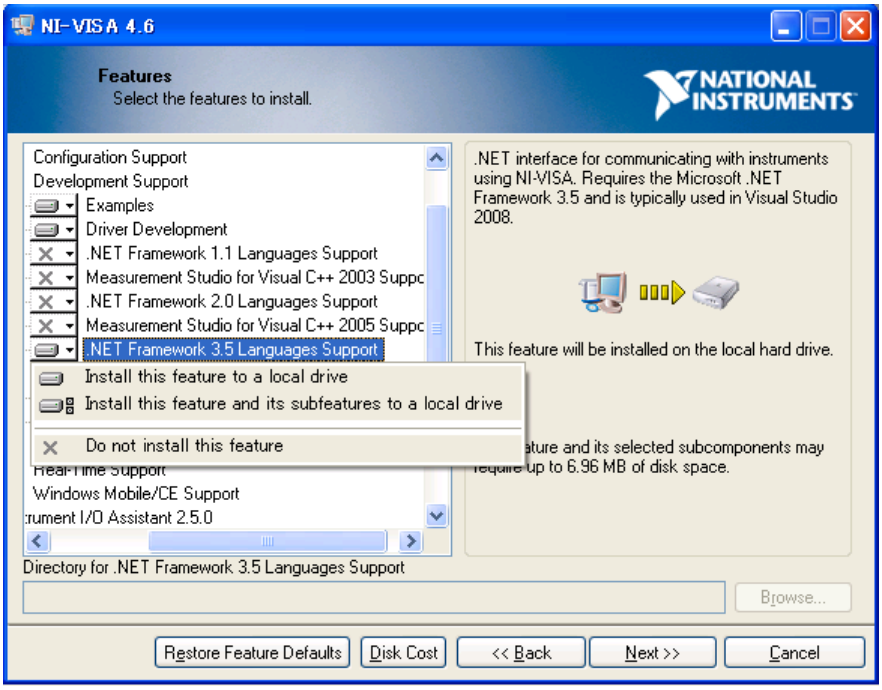
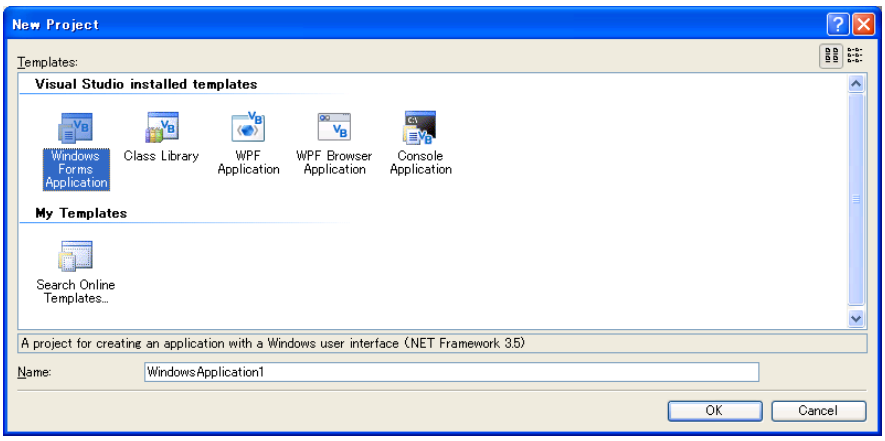


Figure C.1.1-1 Function Selection Screen at VISA Install

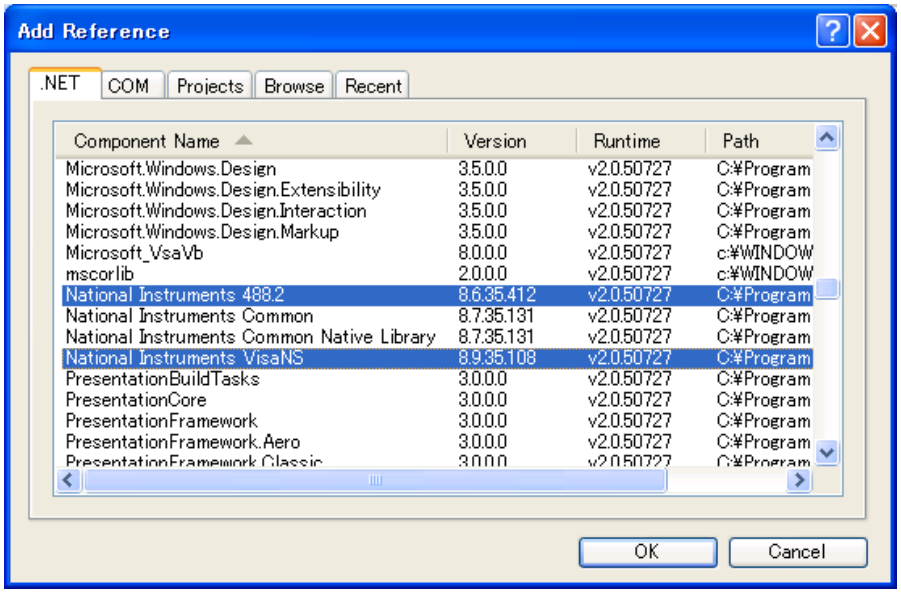
C.1.2 Executing Sample Program

To execute the sample program, follow the below procedures.

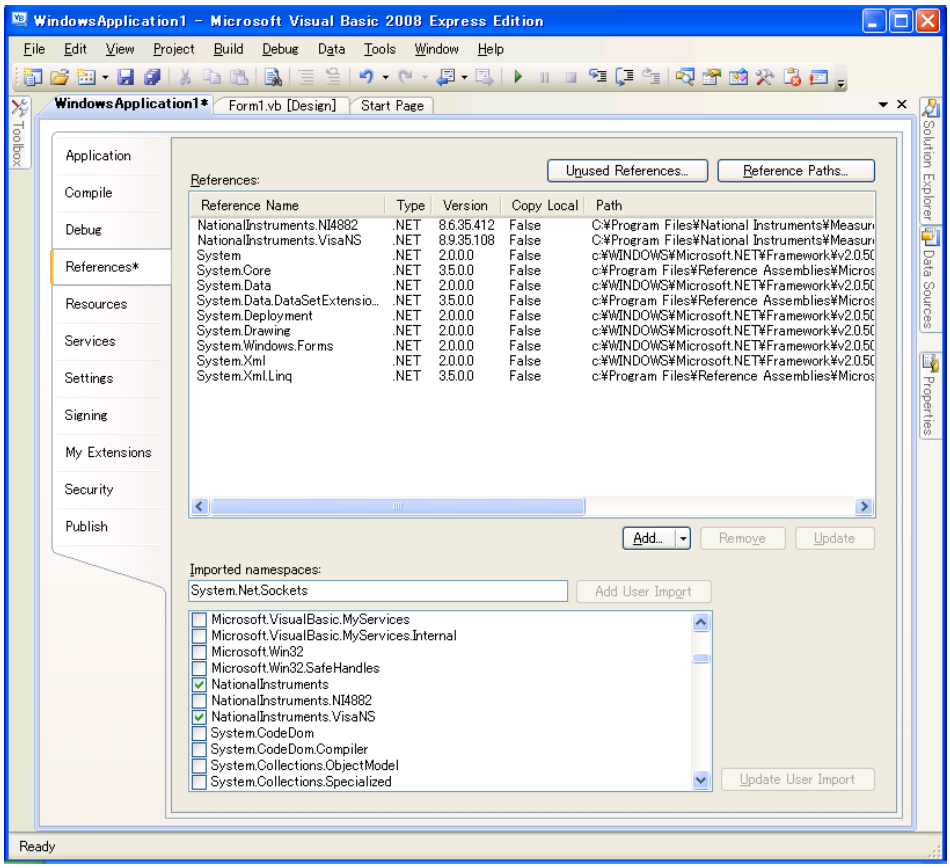
1. Start the Visual BASIC 2008.
2. Click [File]—[New Project].
3. Select the Visual Basic of the Windows application, and click [OK].



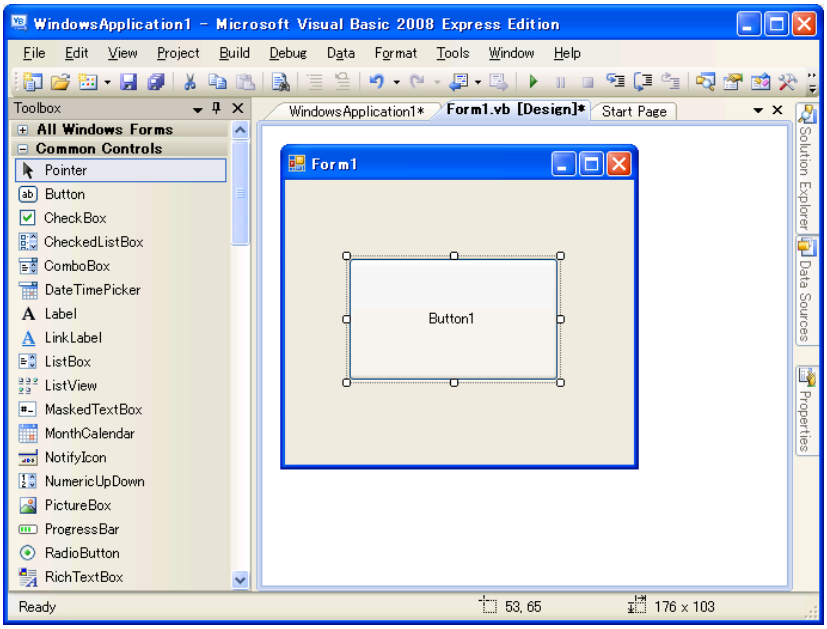
4. The editor to edit the screen is started. Click [Project] —[Add Reference] from the menu bar.
5. Click [.NET] at the Add Reference dialog.
6. Select [National Instruments Common] and [. National Instruments VisaNS] and click [OK].



7. Click [View] — [Solution Explore] from the menu bar.
8. Double-click [My Project] at the Solution Explore.
9. Check National Instruments Common and National Instruments VisaNS, System.Net.Socket in the imported name spaces.



10. Click [Add].
11. The Add Reference dialog is displayed. Click [OK].
12. Allocate the button control in the Form1.vb [Design].



13 Double-click the arranged button to open the screen for inputting the source code. The following code is added in the automatic-generated Form1.Desiner.cs.

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click

End Sub
```

14. Copy the sample program in this document and paste it into the Form1.Desiner.cs.

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click

    ' Paste it into this part.

End Sub
```

15. Change the IP address and GPIB address.

    For an Ethernet connection, the part "192.168.12.10" described above must be changed to the IP address set at the BERTWave.

    For a GPIB connection, the part " GPIB::1::INSTR " described above must be changed to the GPIB address of the BERTWave.

16. Set the Ethernet IP address using at the control PC.
    The IP address must be set the same as the BERTWave IP address.

17. Click [Open Debug] from the Debug menu.

C.2 Example 1: Controlling Pulse Pattern Generator

This sample program control the instrument via the Ethernet interface.

Processing Flow

1. Define the TCP/IP client class for the IP address 192.168.12.10 and port number 5001.
2. Send :MODULE:ID 1 to set the control target to PPG/ED_Ch1.
3. Set the reference clock to the internal clock.
4. Set the bit rate specifications to 1GbE.
5. Set the pattern to PRBS2^23-1.
6. Set the amplitude to 0.5 V.
7. Set the error insertion to Off.
8. Output the signal of the PPG Ch1.
9. Query errors.

```
' Create a TcpClient.
Dim LF As String = Chr(10)
Dim server As String = "192.168.12.10"
Dim Message As String
Dim port As Int32 = 5001
Dim client As New TcpClient(server, port)

' Get a client stream for reading and writing.
Dim stream As NetworkStream = client.GetStream()

 ' Translate the passed message into ASCII and store it as a Byte array.
 Message = ":MODULE:ID 1"
 Dim data As [Byte]() = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 ' Send the message to the connected TcpServer.
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":OUTPUT:CMU:REFCLOCK INTERNAL"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":OUTPUT:BITRATE:STANDARD '1GBE'"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":SYSTEM:ERROR?"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":SOURCE:PATTERN:TYPE PRBS23"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":SOURCE:PATTERN:EADDITION:SET OFF"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":OUTPUT:DATA:OUTPUT 1"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
```

```
        stream.Write(data, 0, data.Length)
        Console.WriteLine("Sent: " + Message)

        ' String to store the response ASCII representation.
        Dim responseData As [String] = [String].Empty

        ' Read the first batch of the TcpServer response bytes.
        Dim bytes As Int32 = stream.Read(data, 0, data.Length)
        responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes)
        Console.WriteLine("Received: " + responseData)

    ' Close everything.
    stream.Close()
    client.Close()
```

C.3 Example 2: Controlling Error Detector

This sample program controls the instrument via the GPIB interface.

Processing Flow

1. Define the class of the GPIB address 1.
2. Send :MODULE:ID 1 to set the control target to PPG/CH_Ch1.
3. Set the tracking of the PPG to Off.
4. Set the bit rate specifications to 10G FC.
5. Set the pattern to PRBS2^23-1.
6. Set the input connector to Single Ended Data.
7. Set the threshold value to 0 V.
8. Set the auto-pattern sync to On.
9. Set the threshold value of the auto-pattern sync to 1E-5.
10. Set the error measurement method to Single.
11. Set the measurement time to 20 seconds.
12. Start the measurement.
13. Query errors.

```
Dim gbs As GpibSession
Dim message, ret As String
gbs    =    CType(ResourceManager.GetLocalManager().Open("GPIB::1::INSTR"),
GpibSession)
gbs.Timeout = 30000

 '  Select module as PPG/ED_Ch1.
 message = ":MODULE:ID 1"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '  Set tracking to PPG off.
 message = ":SENSE:PARAM:TRACKING 0"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '  Set bitrate standard of ED as 10 Giga bit Fiber Channel.
 message = ":INPUT:BITRATE:STANDARD '10G_FC'"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Set test pattern of ED as PRBS2^23-1.
 message = ":SENSE:PATTERN:TYPE PRBS23"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)

 '   Set input connector as Data only.
 message = ":INPUT:DATA:INTERFACE DATA"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)

 '   Set threshold voltage as 0V.
 message = ":INPUT:DATA:THRESHOLD 0"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Set automatic pattern synchronization as On.
 message = ":SENSE:PATTERN:SYNC:ASYNC ON"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Set threshold level of automatic synchronization as 10^-5.
 message = ":SENSE:PATTERN:SYNC:THRESHOLD E_5"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Set gating mode as Single.
 message = ":SENSE:MEASURE:EALARM:MODE SINGLE"
 gbs.Write(message)
```

```
Console.WriteLine("Sent: " + message)
'   Set gating time as 20 seconds.
message = ":SENSE:MEASURE:EALARM:PERIOD 0,0,0,20"
gbs.Write(message)
Console.WriteLine("Sent: " + message)

'   Start measurement.
message = ":SENSE:MEASURE:START"
gbs.Write(message)
Console.WriteLine("Sent: " + message)

ret = gbs.Query(":SYSTEM:ERROR?")
Console.WriteLine("Received: " + ret)
```

C.4 Example 3: Controlling Optical Transceiver

This sample program controls the instrument via the Ethernet cable.

Processing Flow

1. Define the TCP/IP client class for the IP address 192.168.12.10 and port number 5001.
2. Send :MODULE:ID 3 to set the target control to XFP/SFP+.
3. Query whether the optical transceiver is installed or not.
   When the optical transceiver is installed, perform the processing at Step 4 and 5.
4. Query the wavelength of the optical transceiver.
5. Set the output of the optical transceiver to On.
6. Query errors.

```
' Create a TcpClient.
Dim LF As String = Chr(10)
Dim server As String = "192.168.12.10"
Dim Message As String
Dim port As Int32 = 5001
Dim client As New TcpClient(server, port)

' Get a client stream for reading and writing.
Dim stream As NetworkStream = client.GetStream()

 ' Translate the passed message into ASCII and store it as a Byte array.
 Message = ":MODULE:ID 3"
 Dim data As [Byte]() = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 ' Send the message to the connected TcpServer.
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 Message = ":CALCULATE:OPTICAL:STATUS? 'READY'"
 data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
 stream.Write(data, 0, data.Length)
 Console.WriteLine("Sent: " + Message)

 ' String to store the response ASCII representation.
 Dim responseData As [String] = [String].Empty

 ' Read the first batch of the TcpServer response bytes.
 Dim bytes As Int32 = stream.Read(data, 0, data.Length)
 responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes)
 Console.WriteLine("Received: " + responseData)

 If (responseData(1) = "N") Then

    '  Result is "None"
    Console.WriteLine("Optical Transceiver does not exist.")

 Else

    Message = ":SOURCE:OPTICAL:SIGNAL:WLENGTH?"
    data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
    stream.Write(data, 0, data.Length)
    Console.WriteLine("Sent: " + Message)

    responseData = [String].Empty
    bytes = stream.Read(data, 0, data.Length)
```

```
        responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes)
        Console.WriteLine("Received: " + responseData)

        Message = ":SOURCE:OPTICAL:SIGNAL:OUTPUT 1"
        data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
        stream.Write(data, 0, data.Length)
        Console.WriteLine("Sent: " + Message)

    End If

     Message = ":SYSTEM:ERROR?"
     data = System.Text.Encoding.ASCII.GetBytes(Message + LF)
     stream.Write(data, 0, data.Length)
     Console.WriteLine("Sent: " + Message)

     responseData = [String].Empty
     bytes = stream.Read(data, 0, data.Length)
     responseData = System.Text.Encoding.ASCII.GetString(data, 0, bytes)
     Console.WriteLine("Received: " + responseData)

    ' Close everything.
    stream.Close()
    client.Close()
```

C.5 Example 4: Controlling EYE/Pulse Scope

This sample program controls the instrument via the GPIB interface.

Processing Flow

1. Define the class of the GPIB address 1.
2. Send :MODULE:ID 5 to set the control target to EYE/Pulse Scope.
3. Clear the screen.
4. Set the display mode to the pulse mode.
5. Query the measurement execution status.
6. Start the measurement when Sampling HOLD is set.
7. Query errors.

```
Dim gbs As GpibSession
Dim message, ret As String
gbs     =     CType(ResourceManager.GetLocalManager().Open("GPIB::1::INSTR"),
GpibSession)
gbs.Timeout = 30000

 '   Select module as EYE/Pulse Scope.
 message = ":MODULE:ID 5"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Clear Display.
 message = ":DISPLAY:WINDOW:GRAPHICS:CLEAR"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Set measuring mode as Pulse mode.
 message = ":SENSE:DISPLAY:MODE PULSE"
 gbs.Write(message)
 Console.WriteLine("Sent: " + message)
 '   Query Status.
 ret = gbs.Query(":SENSE:SAMPLING:STATUS?")
 Console.WriteLine("Received: " + ret)

 If ret(0) = "H" Then
     '   Start measurement.
     message = ":SENSE:SAMPLING:STATUS RUN"
     gbs.Write(message)
     Console.WriteLine("Sent: " + message)
 End If

 ret = gbs.Query(":SYSTEM:ERROR?")
 Console.WriteLine("Received: " + ret)
```

Appendix D Bibliography

(1) IEEE488.1-1987 *IEEE Standard Digital Interface for Programmable Instrumentation -Description*

(2) IEEE488.2-1992 *IEEE Standard Codes, Formats, Protocols, and Common Commands for Use With IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation -Description*

(3) IVI Foundation *SCPI 1999*

(4) IEEE802.3-2005 *IEEE Standard for Information technology. Telecommunications and information exchange between systems. Local and metropolitan area networks. Specific requirements Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.*

(5) Microsoft Corporation Microsoft *Visual Studio 2008 Document*

(6) National Instruments Corporation *NI-VISA .NET Framework 3.5 Help*

(7) Anritsu Corporation *MX180000A Signal Quality Analyzer Control Software Remote Control Operation Manual*

(8) Anritsu Corporation *MP1026B Eye Pattern Analyzer Programming Manual*

Index

N

O

P

Q

R

S

T