# MD1260A
# 40/100G Ethernet Analyzer
# Remote Control
# Operation Manual

**Sixth 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 MD1260A 40/100G Ethernet Analyzer Operation Manual. Please also refer to this document before using the equipment.**
- **Keep this manual with the equipment.**

**ANRITSU CORPORATION**

Document No.: M-W3406AE-6.0

# 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. |

MD1260A
40/100G Ethernet Analyzer
Remote Control Operation Manual

14 July 2010 (First Edition)
5 November 2012 (Sixth 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.

# About This Manual

The manuals for the MD1260A 40/100G Ethernet Analyzer are configured in two parts.

**MD1260A 40/100G Ethernet Analyzer**
**Operation Manual Operation**

This describes the usage precautions, product overview, installation method, panel operation, maintenance, specifications, and each function.

**MD1260A 40/100G Ethernet Analyzer**
**Operation Manual Remote Control (this manual)**

This describes the remote command for controlling the MD1260A.

**MD1260A 40/100G Ethernet Analyzer**
**Add-on Function Operation Manual**

This describes how to use the Add-on function.

This manual explains the remote control commands. For the connection of the power source and peripheral devices, panel operation, and maintenance, refer to the following manual, refer to the MD1260A 40/100G Ethernet Analyzer Operation Manual (W3406AE).

This operation manual assumes the reader has the following information:

- The reader has read through the MD1260A 40/100G Ethernet Analyzer Operation Manual.
- The reader can create the C or Basic program.

# Table of Contents

# Table of Commands

# 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 MD1260A 40/100G Ethernet Analyzer supports the Ethernet interface. (When the option is installed, the GPIB interface can be used.) When using either interface, set the number to distinguish the MD1260A 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 MD1260A are called command. The command is composed by the ASCII character strings. For example, the following command sets when the counter measurement starts.

```
:CALCulate:COUNter:STARt
```

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 clock frequency value set at the instrument.

```
:CALCulate:DATA? RX_FREQ
```

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

```
103125000000
```

The receiving clock frequency is 103,125,000,000 Hz.

When the MD1260A is measured via remote control, the operation screen is locked. Only the power switch and the key [Local/Panel Unlock] on the system menu are valid in this situation. This situation is called the remote control status. To unlock the remote control status, 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.

## 1.3 Abbreviation

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 |
| SCPI | Standard Commands for Programmable Interfaces |
| SRER | Service Request Enable Register |
| SRQ | Service Request |
| STB | Status Byte Register |
| TR | Transition Filter |

# 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 option installed)
- GPIB cable (when option installed)
- Program development tools

### Ethernet Interface

Prepare an interface meeting the following specifications:

10BASE-T

100BASE-TX

Furthermore, use a cable matching both specifications.

### GPIB Interface

Use GPIB interfaces that conform to IEEE 488.2.

### 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

Get a PC supporting the operating environment for the GPIB interface and program development tools.

## 2.2 Connecting Equipment

### 2.2.1 Connecting Ethernet

Connect the Ethernet connector (upper side) on the side-panel of the MD1260A to the external devices using LAN cables.

Use a LAN crossover cable to connect the MD1260A and the PC controller directly. Use a LAN straight-through cable via a network hub to connect multiple external devices.

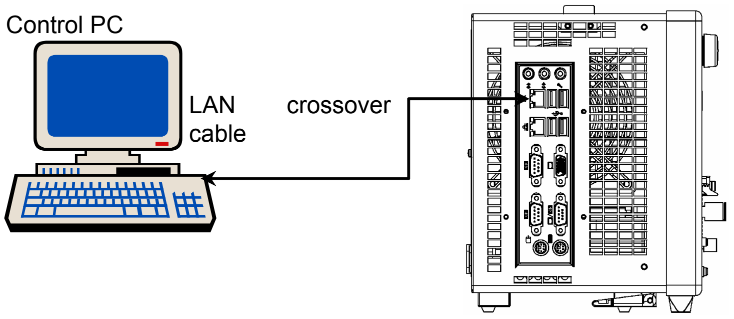


**Figure 2.2.1-1 Direct Connection between MD1260A and Controller PC**

***Note:***

When connecting the MD1260A via LAN, confirm the network settings before measurement.

- The IP addresses of the MD1260A and the other devices must not overlap.
- The IP address of the PC controller must be in the address range set at the subnet mask.

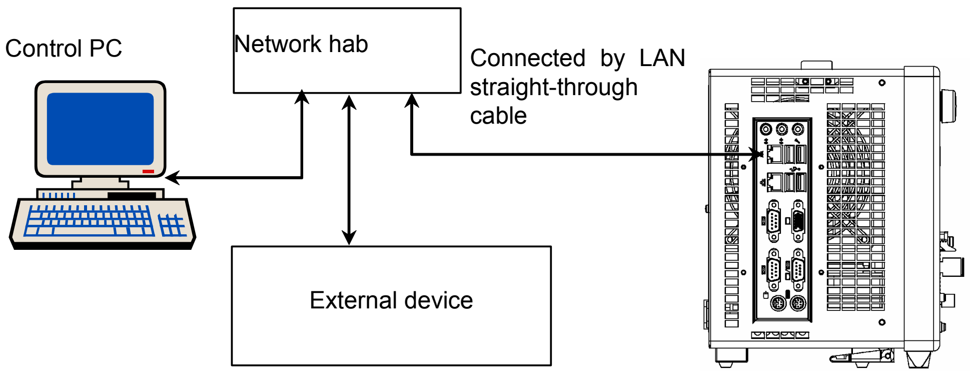


**Figure 2.2.1-2 Typical Connection with Multiple External Devices**

***Note:***

The PC controller may sometimes have difficulty communicating with the MD1260A, depending on the communications status. Direct connection is recommended to ensure stable communications.

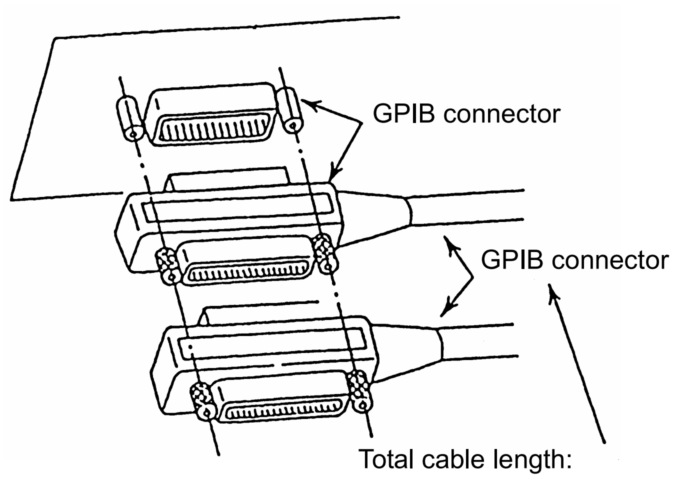
### 2.2.2 Connecting GPIB

Connect the GPIB connector on the rear panel of the MD1260A to the external device using a GPIB cable.

**CAUTION** ⚠

**Always connect the GPIB cable BEFORE turning on the power to the MD1260A. Connecting it while the power is on may damage internal circuits.**

Up to 15 devices, including the PC controller, can be connected to one MD1260A unit. Always follow the conditions shown below when connecting devices.



| | |
|---|---|
| Total cable length: | Up to 20 m |
| Cable length between devices: | Up to 4 m |
| Number of devices that can be connected: | Up to 15 |

**Figure 2.2.2-1 GPIB Cable Connection 1**

Connect cables without forming ground loops.

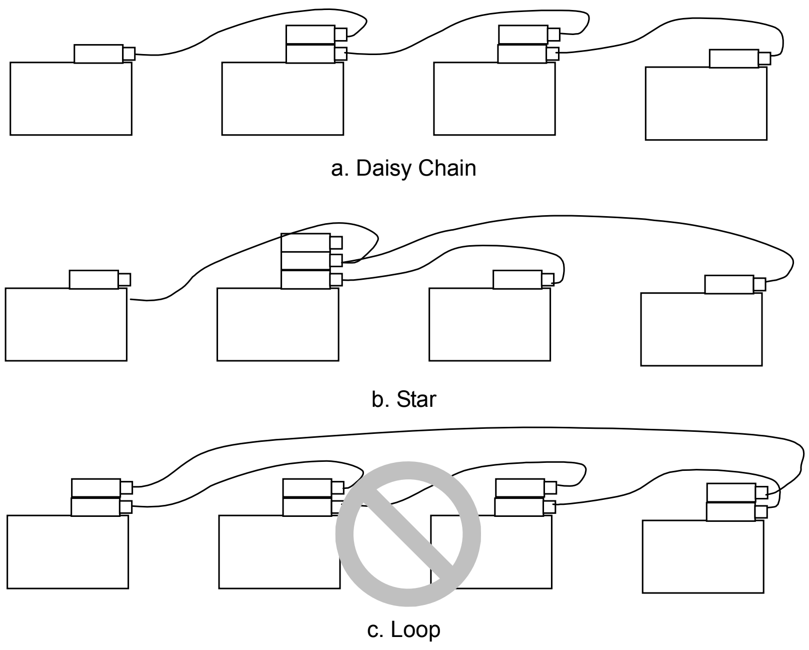


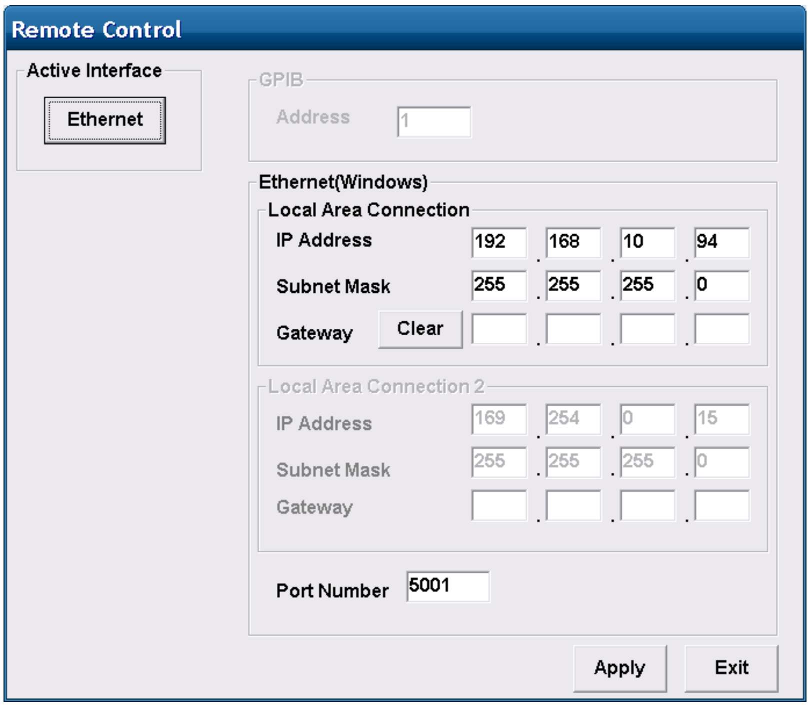
**Figure 2.2.2-2　GPIB Cable Connection 2**

## 2.3 Setting Interface

### 2.3.1 Setting Ethernet

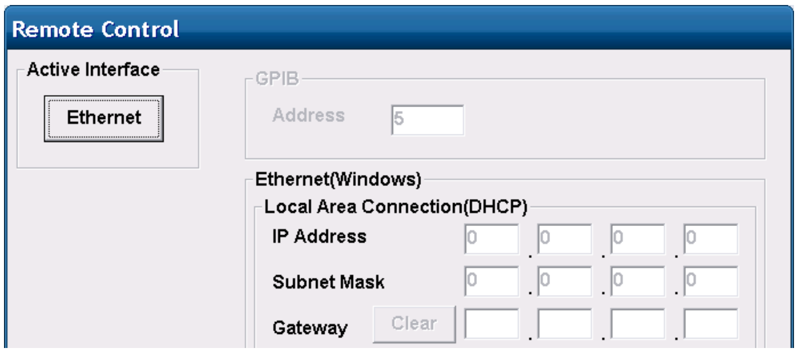
Set the remote control interface to Ethernet as follows and input the IP address.

1. Turn on the MD1260A.
2. Touch the [Utility] tab on the Selector screen.
3. Touch [Setup Utility].
4. Touch [Remote Control].
5. Touch the Active Interface button to set the button display to [Ethernet].

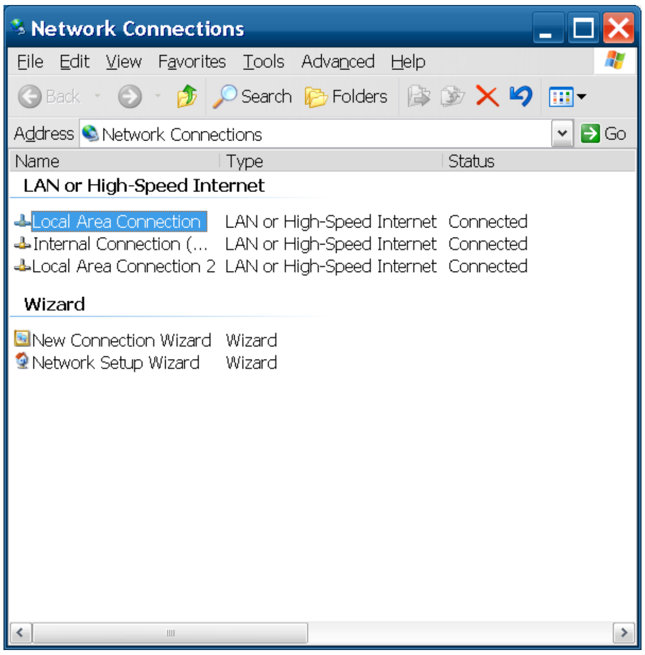


6. Set the IP address, subnet mask, gateway and port number.
   The gateway address may be omitted.
   To display the numeric input panel, touch the text box.
   You cannot enter numeric values directly into the text box using the attached keyboard.
   The port number can be set from 1024 to 5001.
7. Touch [Apply] to complete the settings.
   Touch [Exit] to delete the settings.

If Local Area Connection (DHCP) is displayed at step 5, cancel automatic acquisition of the network address as follows:

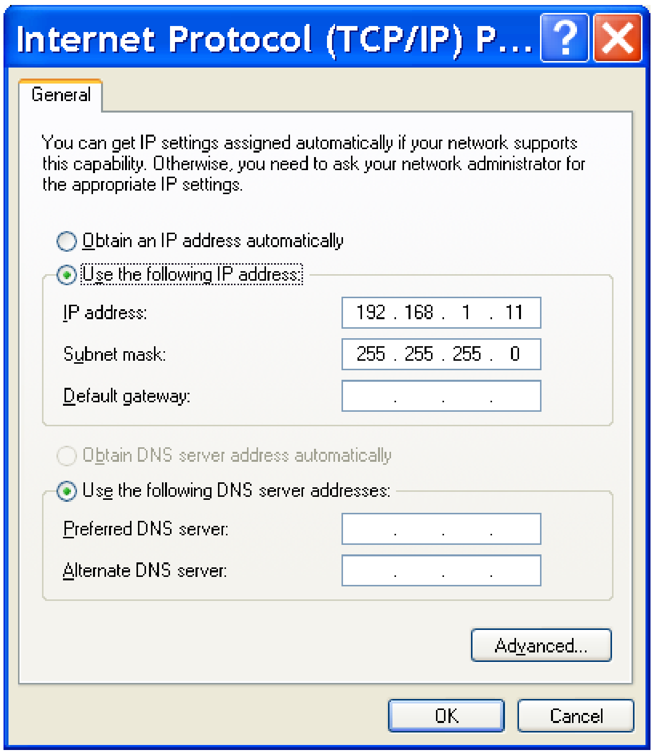


1. Connect the keyboard and mouse to the MD1260A.
2. Press the Windows key on the connected keyboard.
   If there is no Windows key on the keyboard, display the desktop and click [Start].
   For how to display the desktop, refer to Section 2.6.1 Displaying windows desktop in the MD1260A 40/100G Ethernet Analyzer Operation Manual.
3. Click [Control Panel].
4. Double-click [Network Connections].


Network Connections
File Edit View Favorites Tools Advanced Help
Back Search Folders
Address Network Connections
Go
Name Type Status
LAN or High-Speed Internet
Local Area Connection LAN or High-Speed Internet Connected
Internal Connection (... LAN or High-Speed Internet Connected
Local Area Connection 2 LAN or High-Speed Internet Connected
Wizard
New Connection Wizard Wizard
Network Setup Wizard Wizard

5. Right-click [Local Area Connection], and click [Properties].

6. The Local Area Connection Properties window opens.
   Click [Internet Protocol (TCP/IP)] in the list box, and click the [Properties] button.



Internet Protocol (TCP/IP) P...

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically
● Use the following IP address:

IP address: 192 . 168 . 1 . 11
Subnet mask: 255 . 255 . 255 . 0
Default gateway: . . .

○ Obtain DNS server address automatically
● Use the following DNS server addresses:

Preferred DNS server: . . .
Alternate DNS server: . . .

7. Put a checkmark in [Use the following IP address].

   The settings at the factory shipment are as follows.
   IP address 192.168.100.101, Subnet mask 255.255.255.0
8. Click the [OK] button.
9. Click the [OK] button in [Local Area Connection Properties].

## CAUTION ⚠

**Do not change the setting of [Internal Connection (Don't Change)]. If it is changed, the MD1260A will not function properly.**

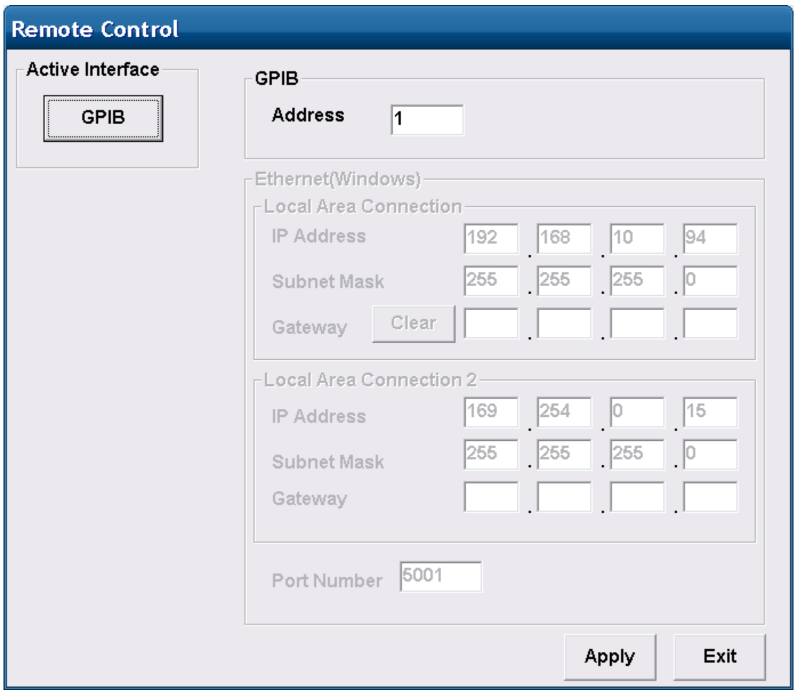
**Do not change the setting of [Local Area Connection2]. If it is changed, the multiport function will be disabled.**

## 2.3.2 Setting GPIB

Set the remote control interface to GPIB as follows and input the GPIB address.

1. Turn on the MD1260A.
2. Touch the [Utility] tab on the Selector screen.
3. Touch [Setup Utility].
4. Touch [Remote Control].
5. Touch the Active Interface button to set the button display to [GPIB].

   If Opt-030/130 is not installed, the Active Interface button is disabled.



Remote Control

Active Interface: GPIB

GPIB
Address: 1

Ethernet(Windows)

Local Area Connection
IP Address: 192 . 168 . 10 . 94
Subnet Mask: 255 . 255 . 255 . 0
Gateway: Clear . . .

Local Area Connection 2
IP Address: 169 . 254 . 0 . 15
Subnet Mask: 255 . 255 . 255 . 0
Gateway: . . .

Port Number: 5001

Apply Exit

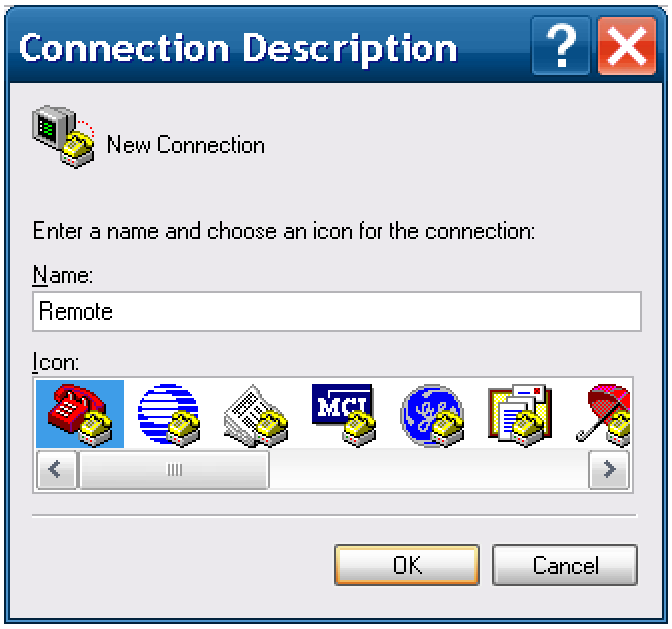
6. Touch the GPIB address on the test box.
7. Set the GPIB address using the numeric value input panel.
   You cannot enter numeric values directly into the text box using the attached keyboard.
8. Touch [Apply] to complete the settings.
   Touch [Exit] to delete the settings.

## 2.4 Checking Connection

Check that the link between the PC and MD1260A has been established via Ethernet.

### 2.4.1 When using Ethernet (Windows XP)

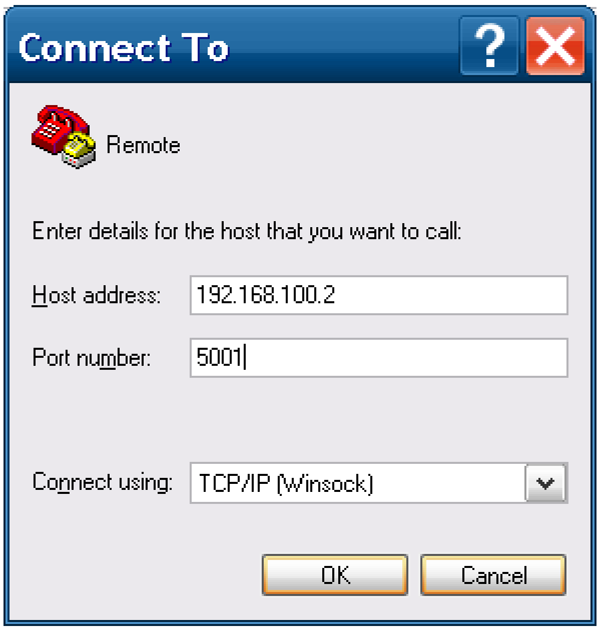
1. Click Programs at the Windows Start menu.
2. Click [Accessories].
3. Click [HyperTerminal] from the Communication submenu.
4. When the following screen opens, input a name and click [OK].



Enter a name and choose an icon for the connection:

Name:
Remote

5. Set the connection method to TCP/IP and input the host address and port number.
   The following figure shows how to set the IP address to 192.168.100.2 and the port number to 5001 as described in Section 2.3.1 Setting Ethernet.



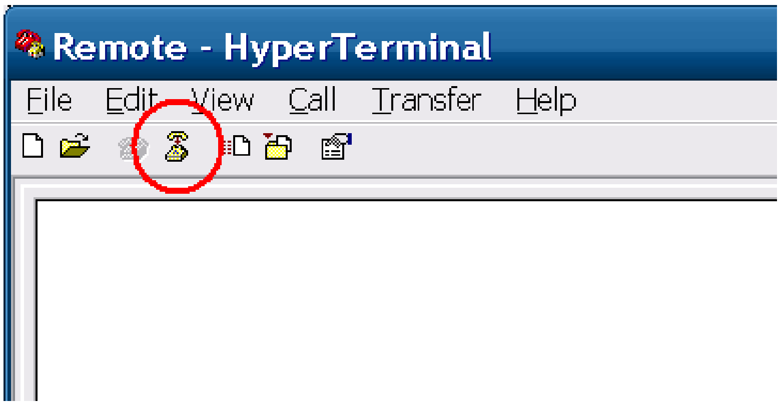
Enter details for the host that you want to call:

Host address: 192.168.100.2

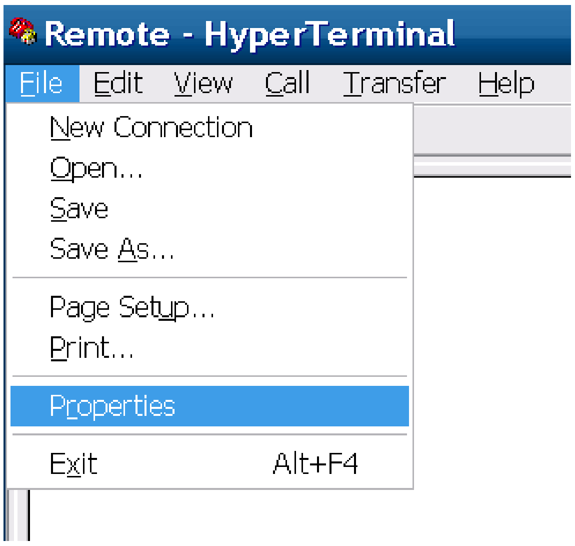
Port number: 5001

Connect using: TCP/IP (Winsock)

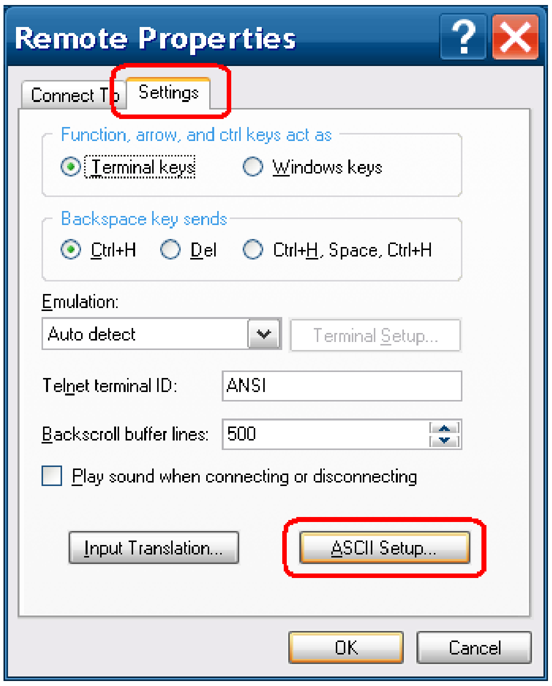
6. If the message "192.168.100.2 cannot be connected to the port 5001." is displayed, the PC cannot recognize the MD1260A. Check that:
   - The application is started.
     (Refer to Section 3.1.1 Starting Application in the MD1260A 40/100G Ethernet Analyzer Operation Manual.)
   - The MD1260A IP address and port numbers are correct.
   - The Ethernet cable type (straight through/crossover) is correct.
   - The Ethernet cable is connected to the correct Ethernet connector on the left side of the MD1260A.
   - The cables and connectors are not damaged.
7. When the display changes to "Connected", the PC controller and the MD1260A can be connected via the Ethernet.



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



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



Function, arrow, and ctrl keys act as: [✓ Terminal keys] Windows keys

Backspace key sends: [✓ Ctrl+H] Del Ctrl+H, Space, Ctrl+H

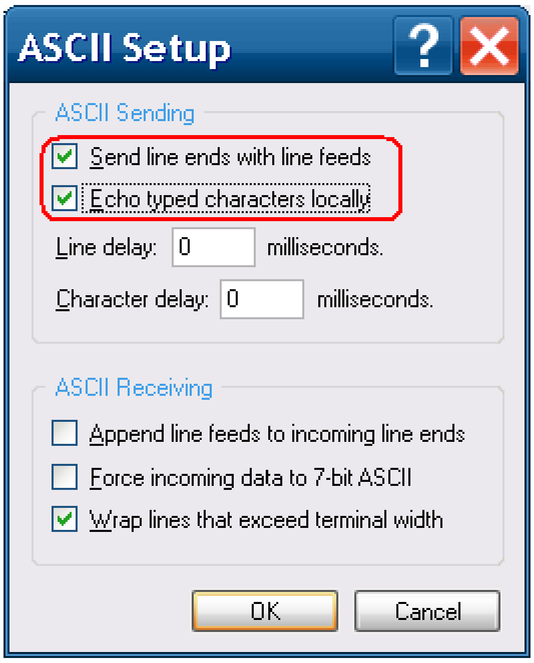
Emulation: Auto detect

Telnet terminal ID: ANSI

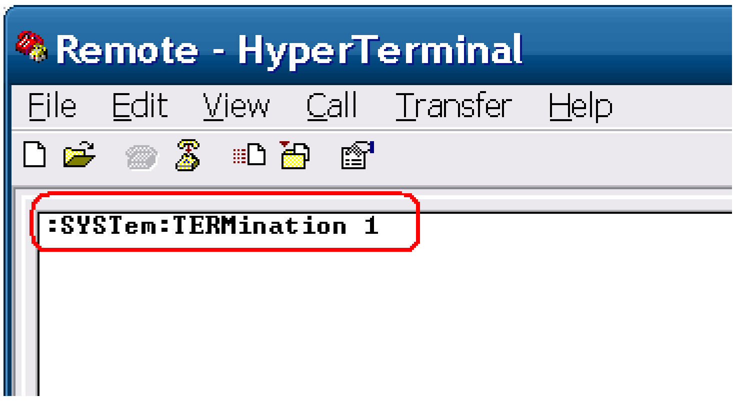
Backscroll buffer lines: 500

☐ Play sound when connecting or disconnecting

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

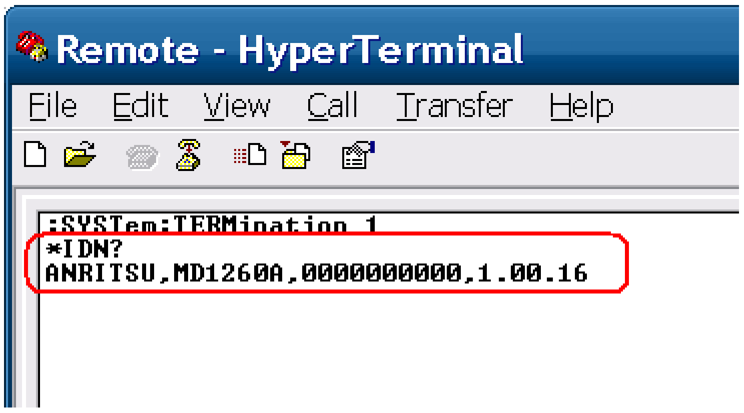

ASCII Setup
ASCII Sending
Send line ends with line feeds
Echo typed characters locally
Line delay: 0 milliseconds.
Character delay: 0 milliseconds.
ASCII Receiving
Append line feeds to incoming line ends
Force incoming data to 7-bit ASCII
Wrap lines that exceed terminal width
OK
Cancel

11. 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).



```
:SYSTem:TERMination 1
```

12. 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,MD1260A,0000000000,1.00.16 is displayed.



```
:SYSTem:TERMination 1
*IDN?
ANRITSU,MD1260A,0000000000,1.00.16
```

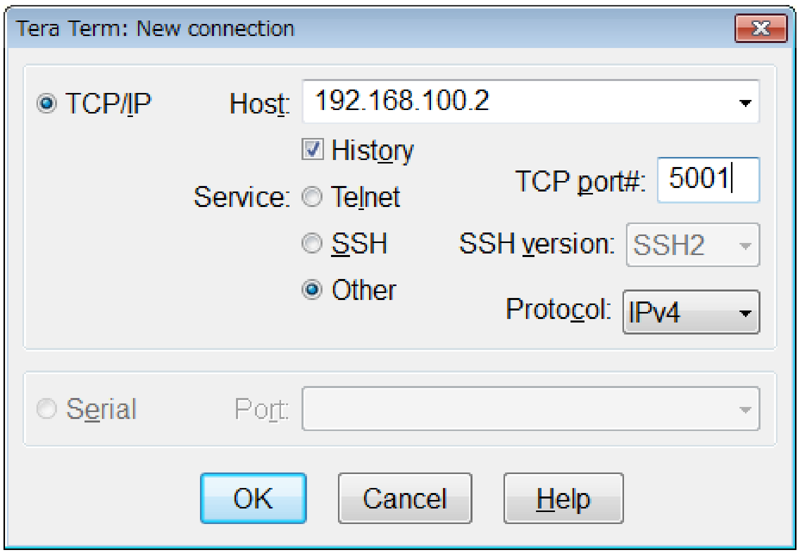
The above procedures confirm that control is possible using remote commands. Similar commands can be input to confirm the operation of remote commands.

## 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].

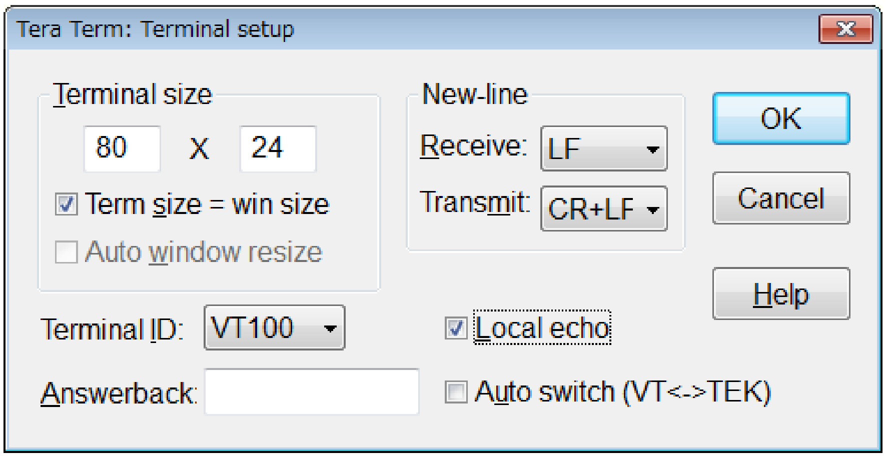
   If the IP address is set to 192.168.100.2 at Section 2.3.1 Setting Ethernet, and the port number is set to 5001, set as follows.



[✓ TCP/IP] Host: 192.168.100.2
[✓ History]
Service: Telnet / SSH / [✓ Other]
TCP port#: 5001
SSH version: SSH2
Protocol: IPv4
Serial Port:

2. When the MD1260A 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].


Tera Term: Terminal setup

Terminal size: 80 X 24

☑ Term size = win size

☐ Auto window resize

New-line

Receive: LF

Transmit: CR+LF

Terminal ID: VT100

Answerback:

☑ Local echo

☐ Auto switch (VT<->TEK)

6. Send *IDN?.
   Confirm that the response is displayed from the MD1260A.

***Note:***

When the panel lock is released by the panel operation, the communication with the MD1260A 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 character strings indicating the 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 controller to instrument

There are two types of the program messages:

- Command
  This can be used for setting measurement conditions and starting measurement.
- 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:

- 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: `:CALCulate:COUNter:STARt`

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

Example: `*ESE?`

### 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 `TestResult` at the title is shown below.

Example:
```
:MMEMory:STORe "TestResult",RESULT
:MMEMory:STORe 'TestResult',RESULT
```

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’.”
```

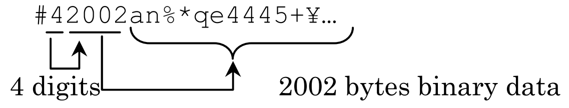
The numeric values can be described by using numeric data. Input numeric values either as decimal, binary, octal, or hexadecimal numbers. When using binary, octal, or hexadecimal numbers, prefix the data with #B,#O, or #H, respectively.

Example:
```
        10   #B1010         #O12          #HA
        1550 #B11000001110  #O3016        #H60E
```
When using decimal numbers, use integer numbers, fixed point, and floating point. The following examples indicate the same values.

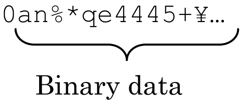
Example:
```
        -10         -10.00         -1E1
        1250        1250.000       1.25E3
        0.0023                     2.3E-4
```

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+¥…`



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

Example: `:CALCulate:DATA? RX_FREQ,RX_FREQ_D`

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

Example:
```
        :MMEMory:STORe "TestResult",RESULT;:SYSTem:ERRor?
```

## 2.5.3 Common commands

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

The common commands are divided into mandatory and option commands. The MD1260A supports the common commands listed in Table 2.5.3-1.

**Table 2.5.3-1 Common Commands**

| Command | Explanation |
|---|---|
| *CLS | Clears standard 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 |
| *RST | Initializes MD1260A setting conditions |
| *SRE | Sets and queries SRER |
| *STB | Queries status byte register |
| *TRG | Starts measurement |
| *WAI | Waits for previous sent message completion |

## 2.5.4 Device dependent commands

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

## 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 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.

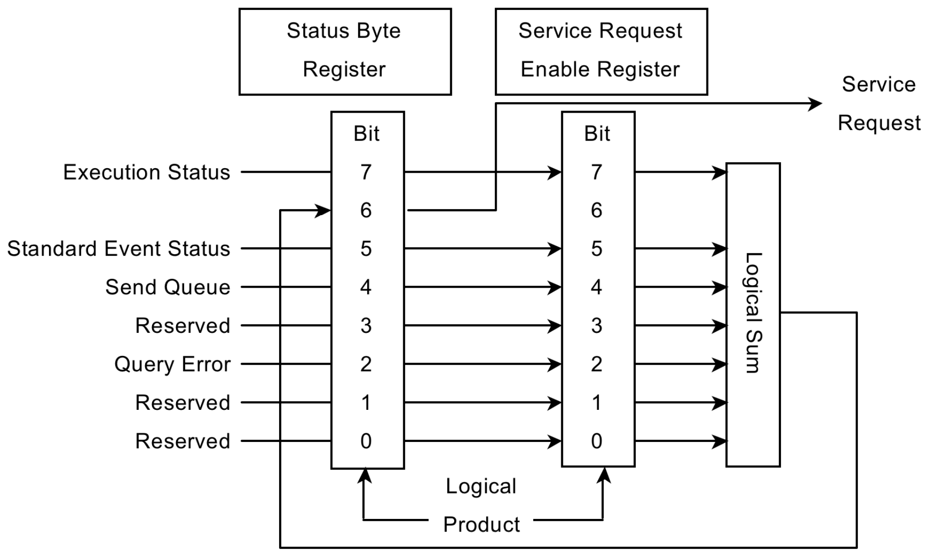


**Figure 2.6.1-1 Configuration of Status Byte Register and Service Request Enable Register**

***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 2.6.1-1 Meaning of Status Byte Register**

| Bit | Explanation |
|---|---|
| 7 | Not used; always 0 |
| 6 | MSS (Master Summary Register)<br>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)<br>This is always 1 when there is a response message in the output queue of this instrument |
| 3 | Not used; always 0 |
| 2 | Not used; always 0 |
| 1 | Not used; always 0 |
| 0 | Not used; always 0 |

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 2.6.2 Standard Event Status Register.

Bits 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`.

## 2.6.2 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.

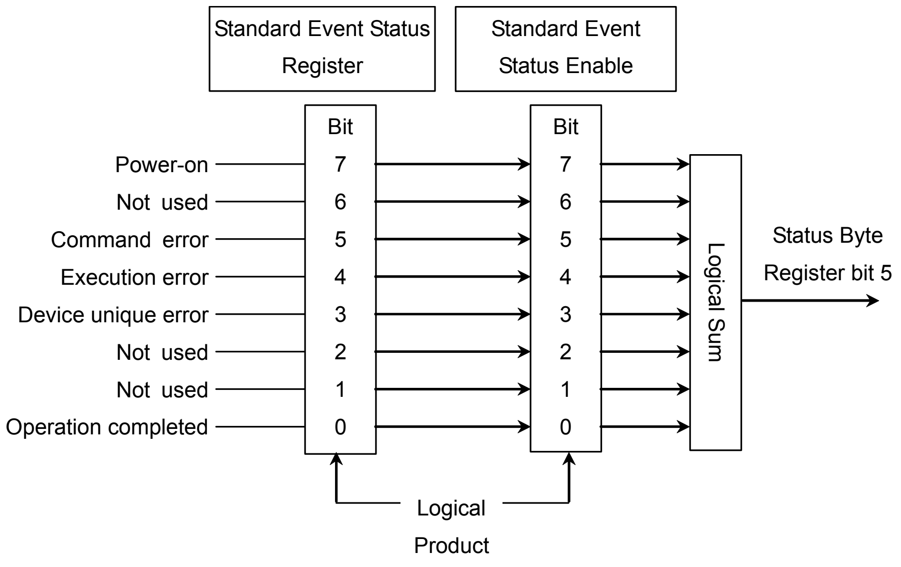


**Figure** 2.6.2-1 **Configuration of Standard Event Status Register and Standard Event Status Enable Register**

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

**Table 2.6.2-1 Meaning of Standard Event Status Register**

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

A bit 5 command error occurs under the following circumstances:

- When instruments received message not described in Section 3.
  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

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.

## 2.7 Confirming Message Execution Status

When using Ethernet interface:

A message can be sent even while the MD1260A is executing a program message. However, the next message is not processed until processing of the previous message is complete.

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

Example:

```
:MMEMory:STORe "TestResult",RESULT
                                      Saves measurement results
:System:Error?                        Queries error
> 0,"No Error"                        No errors when 0 read
```

In these examples, there may be a delay ranging up to a few seconds until the response message is returned after sending `:System:Error?` Execution of the `:MMEMory:STORe` and `:System:Error?` messages is confirmed by receiving the response message.

When using GPIB interface:

The message cannot be sent to the MD1260A while executing the previously sent message. When using the GPIB interface, set the PC controller GPIB interface timeout to ensure that communications do not time-out while this instrument is executing messages.

## 2.8 Remote Control When Operating Multiport Function

When controlling the MD1260A using the multiport function via remote operation, connect the PC controller to the master MD1260A.

For the multiport function, refer to Chapter 8 Multiport Function in the MD1260A 40/100G Ethernet Analyzer Operation Manual.

The master and slave displayed on the top menu of the application screen can be controlled via remote operation.

The Unit ID list of the MD1260A can be queried using the command :UENTry:LIST?.

To control a specific MD1260A unit, specify the Unit ID of the MD1260A using :UENTry:ID.

The [:AUNit] command in the header is executed by all MD1260A units by adding :AUNit.

# Chapter 3 Message Details

This chapter describes the message details of remote control commands for the MD1260A.

## 3.1 Description of Message Explanations

The following table shows the rules for describing messages.

**Table 3.1-1 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.<br>When A \| B({C \| D}) can be chosen, select one of them. |
| <CHAR> | Displays a choice for mnemonic<br>Equivalent to IEEE488.2 <CHARACTER PROGRAM DATA>, <CHARACTER RESPONSE DATA> |
| <BOOLEAN> | Displays enable/disable setting<br>To enable the setting, set ON or 1. To disable the setting, set OFF or 0.<br>1 is returned for enabled queries and 0 is returned for disabled. |
| <NR1> | Numeric values<br>Example: 123, –500 |
| <NR2> | Displays binary fixed point format.<br>Example: 12.345, –500.0 |
| <NR3> | Displays numeric value of floating point format.<br>Example: 0.00E-11, 3.05E–11 |
| <HEX> | Displays hexadecimal values.<br>Specifies hexadecimal values after #H<br>Example: #H0011EEFF |
| <BINARY> | Displays binary numbers.<br>Specifies binary numbers after #B<br>Example: #B0011010 |
| <STRING> | Character string data<br>For details, refer to Section 2.5.2 Message Configuration. |

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

Example :SYSTem:ERRor?
This header can be written as follows:

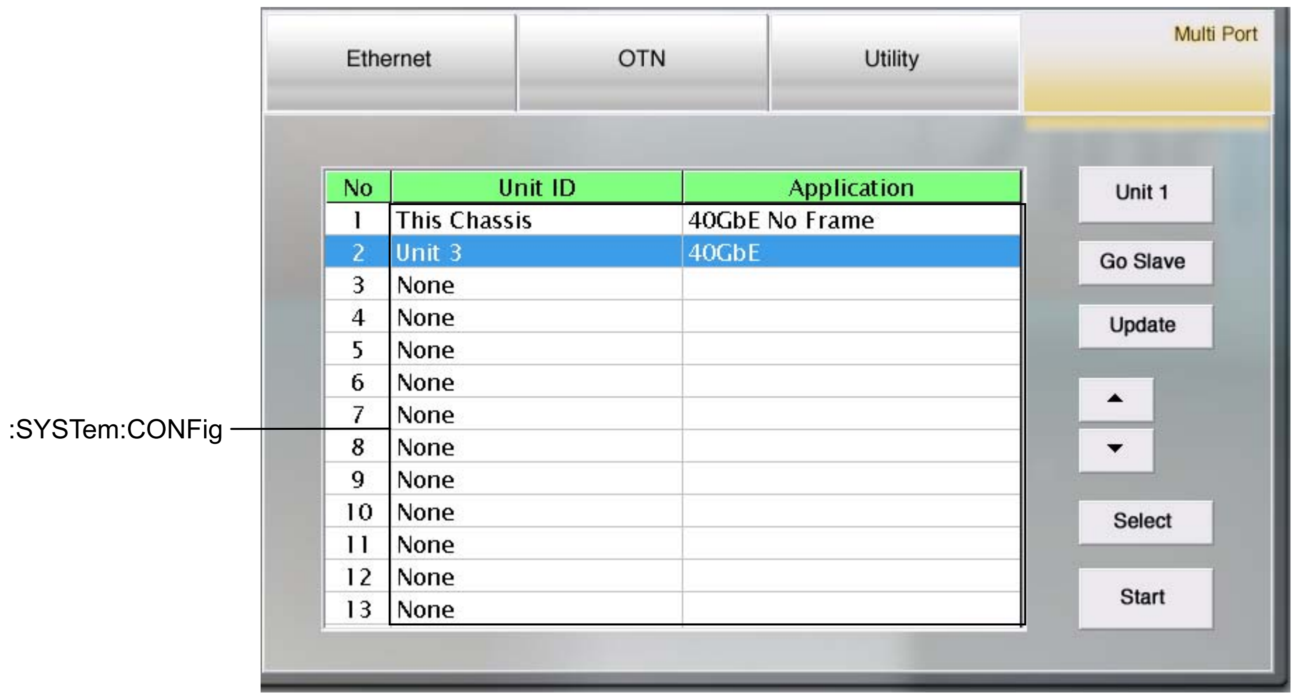
```
:SYSTem:ERRor?
:SYST:ERR?
:SYSTEM:ERROR?
:SYSTem:ERR?
```

These messages are interpreted as the same meanings in the MD1260A.

## 3.2 Correspondence between Panel Operation and Messages

This section explains correspondence between panel operations and messages.

### 3.2.1 Messages Corresponding to Common Operations



| No | Unit ID | Application |
|---|---|---|
| 1 | This Chassis | 40GbE No Frame |
| 2 | Unit 3 | 40GbE |
| 3 | None | |
| 4 | None | |
| 5 | None | |
| 6 | None | |
| 7 | None | |
| 8 | None | |
| 9 | None | |
| 10 | None | |
| 11 | None | |
| 12 | None | |
| 13 | None | |

**Figure 3.2.1-1 Messages Corresponding to Multi-port Settings**

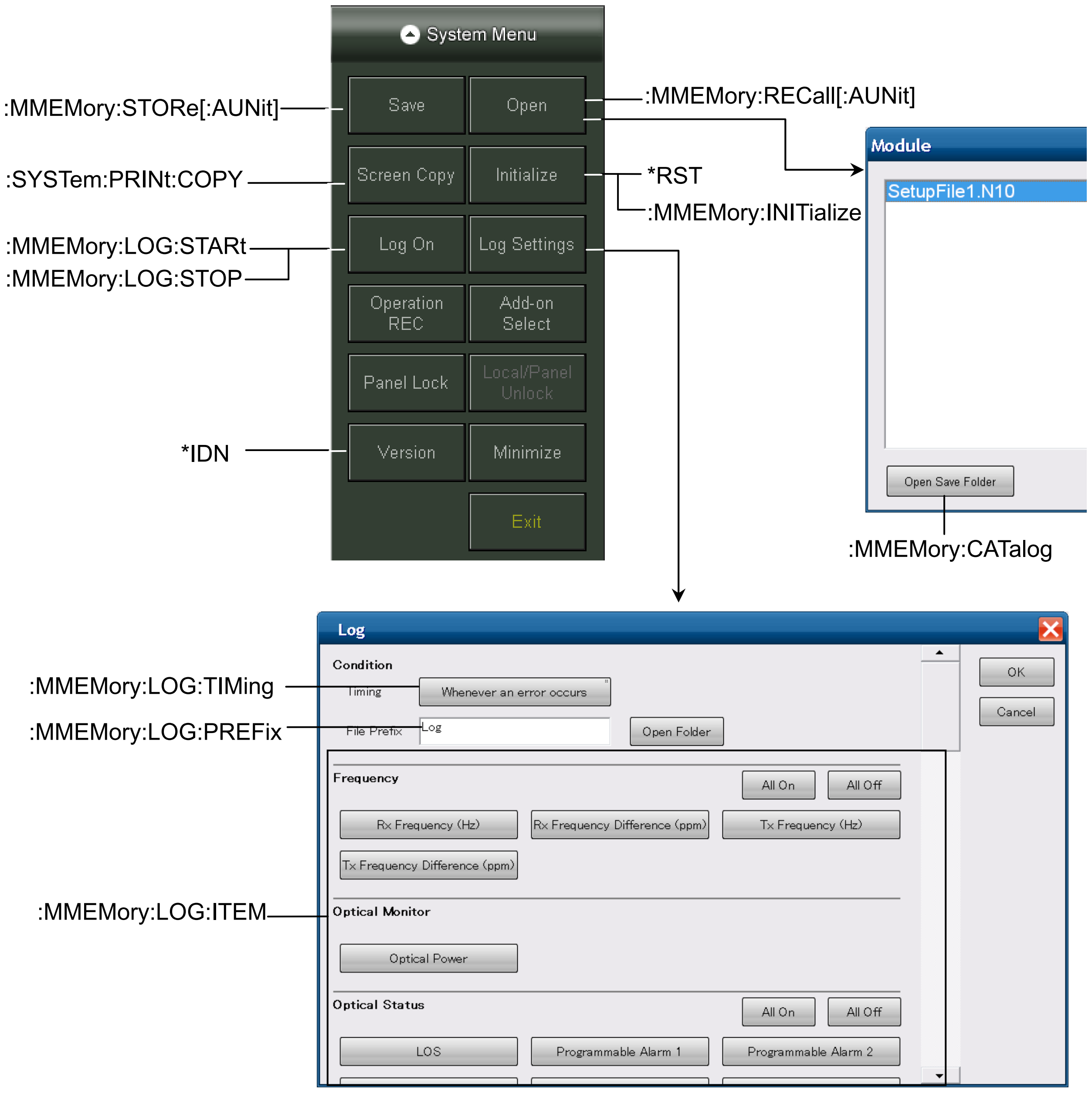


**Figure 3.2.1-2 Messages Corresponding to System Menu**

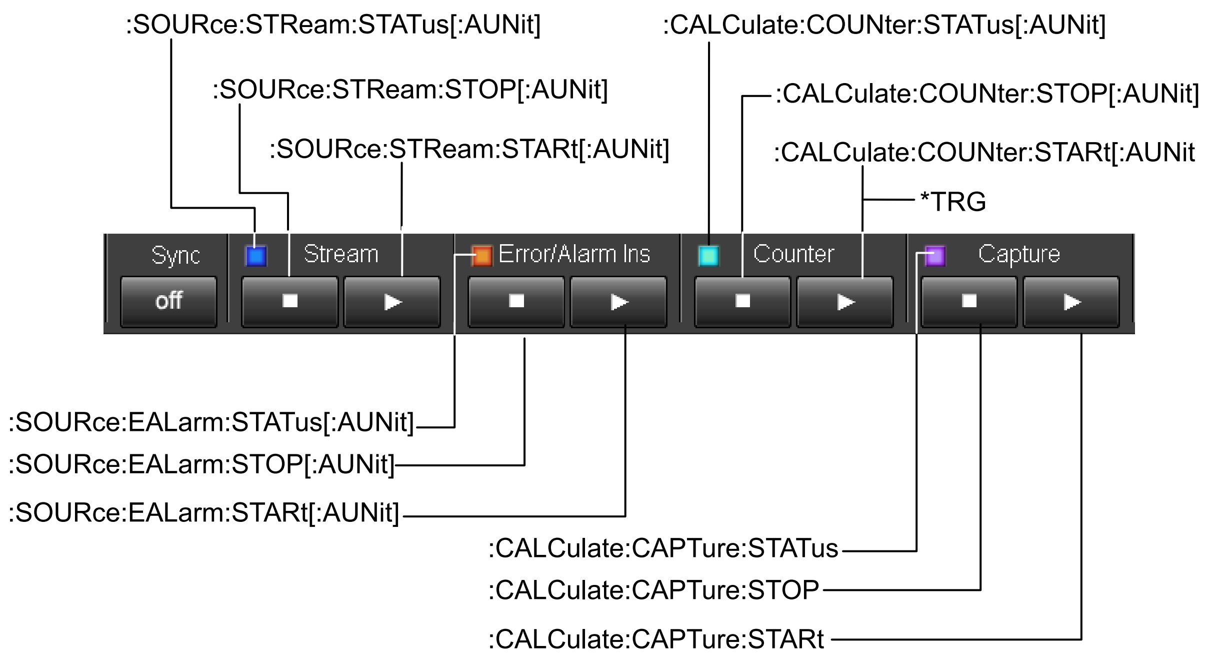

:SOURce:STReam:STATus[:AUNit]
:SOURce:STReam:STOP[:AUNit]
:SOURce:STReam:STARt[:AUNit]
:CALCulate:COUNter:STATus[:AUNit]
:CALCulate:COUNter:STOP[:AUNit]
:CALCulate:COUNter:STARt[:AUNit
*TRG
Sync
off
Stream
Error/Alarm Ins
Counter
Capture
:SOURce:EALarm:STATus[:AUNit]
:SOURce:EALarm:STOP[:AUNit]
:SOURce:EALarm:STARt[:AUNit]
:CALCulate:CAPTure:STATus
:CALCulate:CAPTure:STOP
:CALCulate:CAPTure:STARt

**Figure 3.2.1-3 Messages Corresponding to Operation Area**

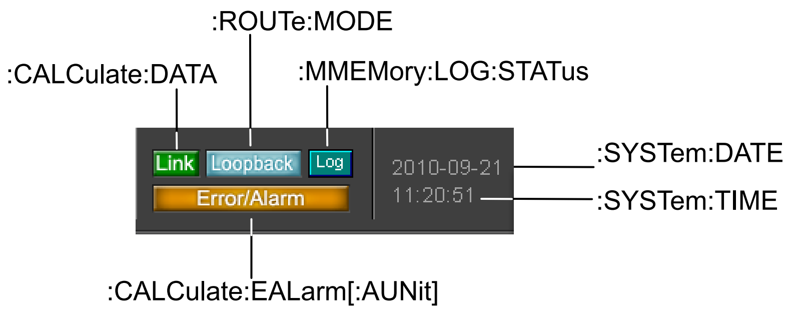


**Figure 3.2.1-4 Messages Corresponding to Summary Message/Time Display Area**

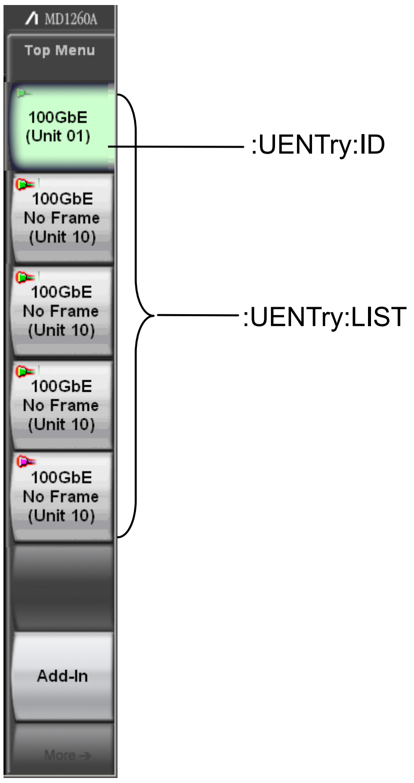
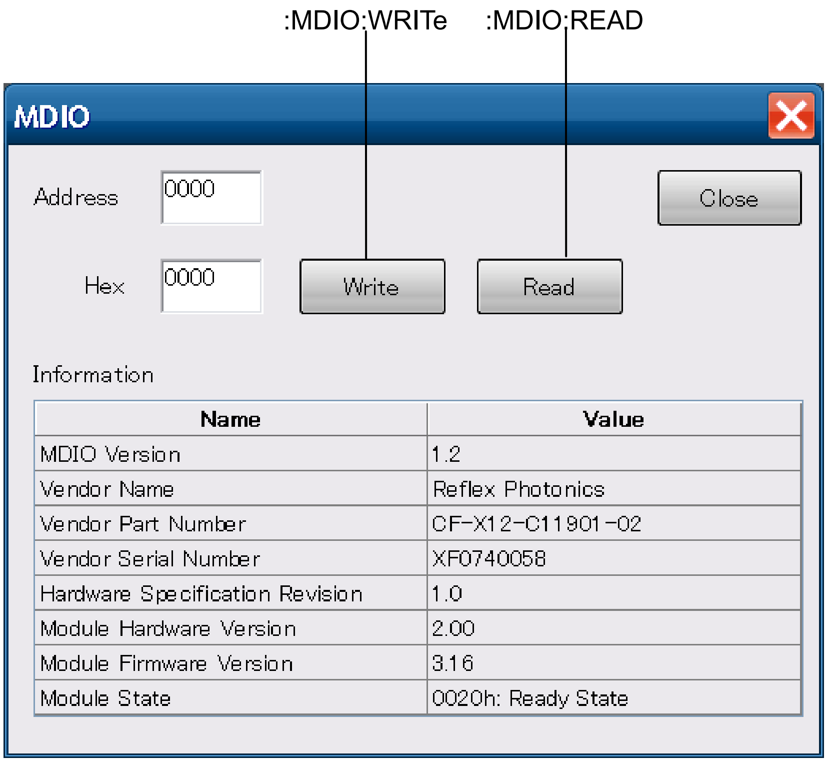

MD1260A
Top Menu
100GbE (Unit 01)
100GbE No Frame (Unit 10)
100GbE No Frame (Unit 10)
100GbE No Frame (Unit 10)
100GbE No Frame (Unit 10)
Add-In
More ->
:UENTry:ID
:UENTry:LIST

**Figure 3.2.1-5 Messages Corresponding to Top Menu**



| Name | Value |
|---|---|
| MDIO Version | 1.2 |
| Vendor Name | Reflex Photonics |
| Vendor Part Number | CF-X12-C11901-02 |
| Vendor Serial Number | XF0740058 |
| Hardware Specification Revision | 1.0 |
| Module Hardware Version | 2.00 |
| Module Firmware Version | 3.16 |
| Module State | 0020h: Ready State |

**Figure 3.2.1-6 Messages Corresponding to Setting Area [MDIO]**

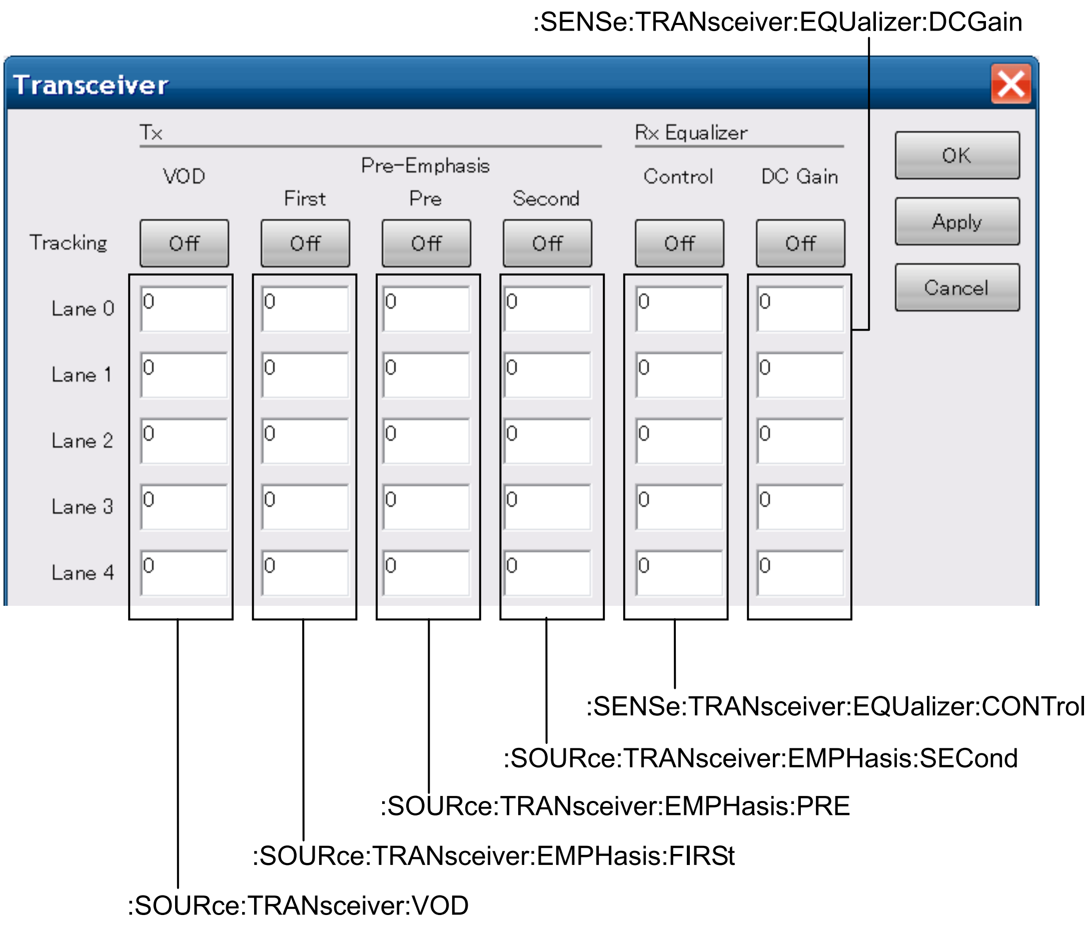
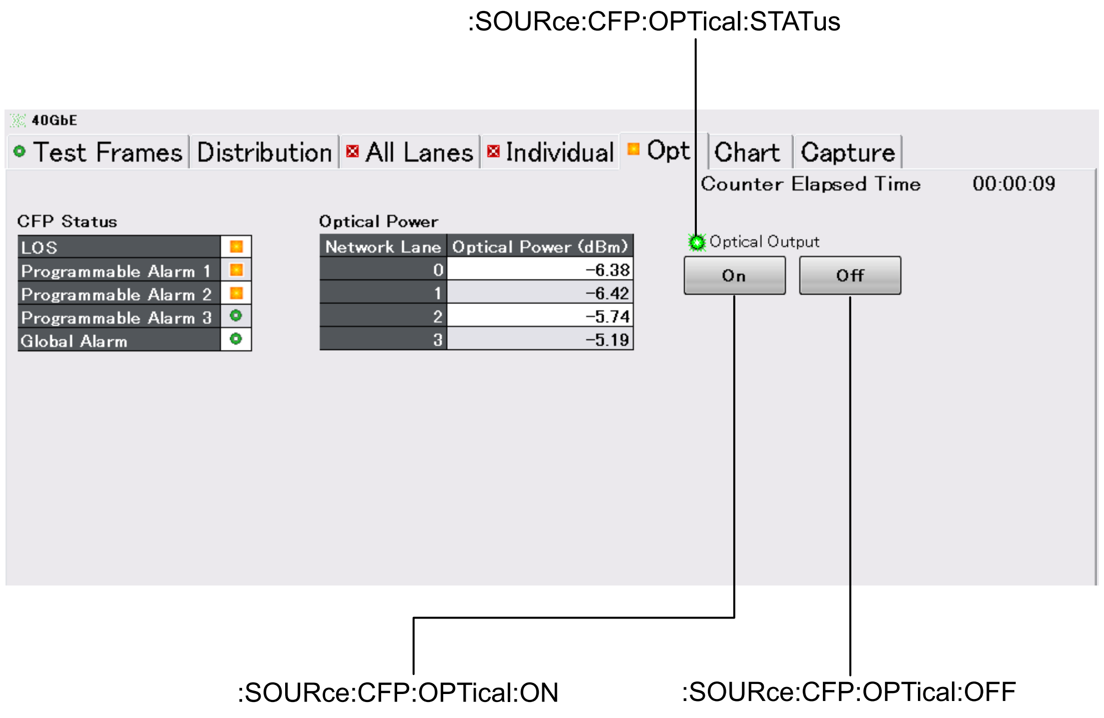

:SENSe:TRANsceiver:EQUalizer:DCGain
Transceiver
Tx
Rx Equalizer
VOD
Pre-Emphasis
First
Pre
Second
Control
DC Gain
Tracking
Off
Lane 0
Lane 1
Lane 2
Lane 3
Lane 4
OK
Apply
Cancel
:SENSe:TRANsceiver:EQUalizer:CONTrol
:SOURce:TRANsceiver:EMPHasis:SECond
:SOURce:TRANsceiver:EMPHasis:PRE
:SOURce:TRANsceiver:EMPHasis:FIRSt
:SOURce:TRANsceiver:VOD

**Figure 3.2.1-7 Messages Corresponding to Setting Area [Transceiver]**



Counter Elapsed Time 00:00:09

CFP Status

| | |
|---|---|
| LOS | 🟧 |
| Programmable Alarm 1 | 🟧 |
| Programmable Alarm 2 | 🟧 |
| Programmable Alarm 3 | ● |
| Global Alarm | ● |

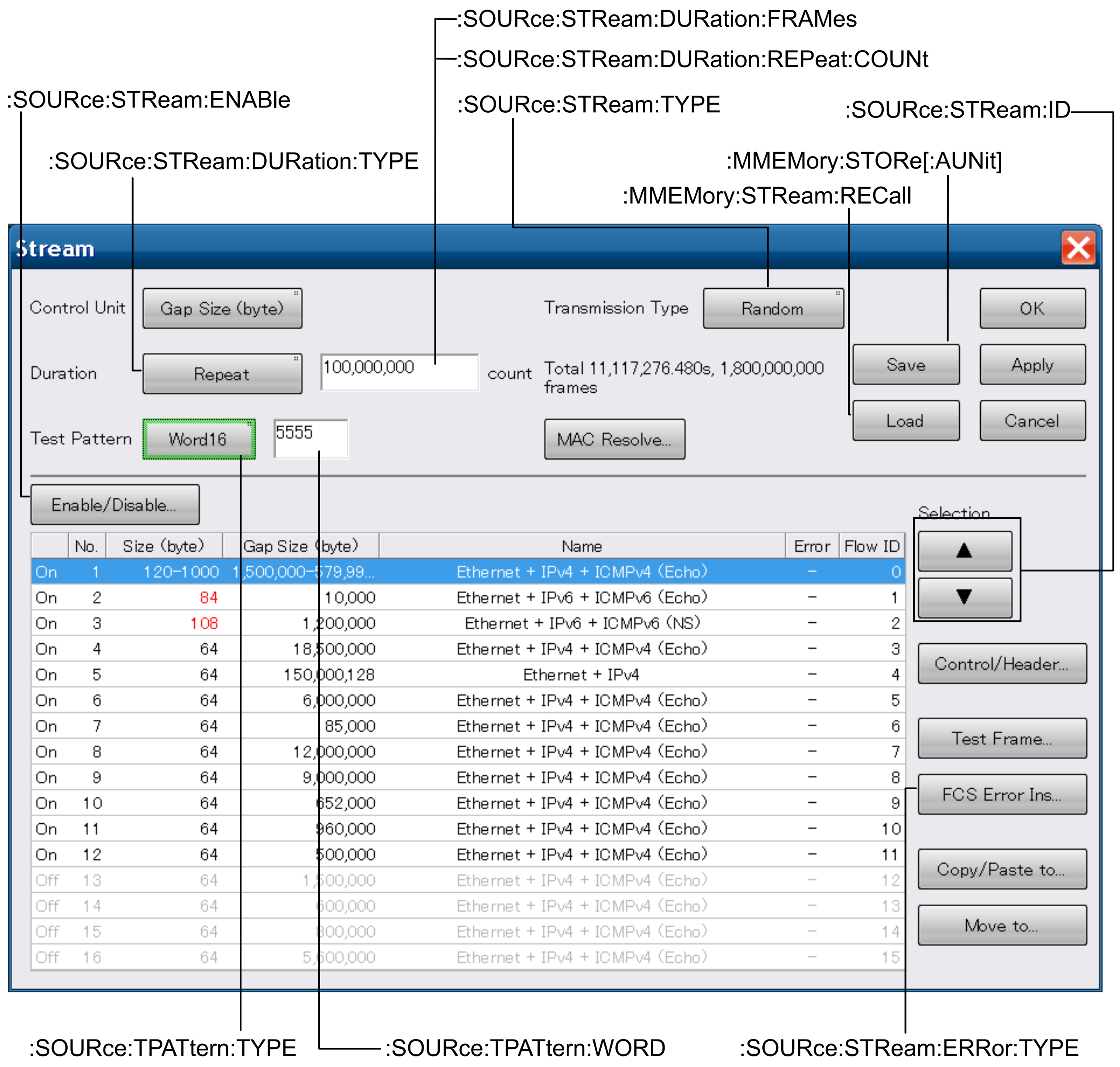
Optical Power

| Network Lane | Optical Power (dBm) |
|---|---|
| 0 | −6.38 |
| 1 | −6.42 |
| 2 | −5.74 |
| 3 | −5.19 |

Figure 3.2.1-8 Message Corresponding to Measurement Result Display Area (Opt)

### 3.2.2 Messages Corresponding to 40GbE/100GbE

For the Messages Corresponding to the setting area of [MDIO] and [Transceiver], refer to Figure 3.2.1-6 Messages Corresponding to Setting Area [MDIO] and Figure 3.2.1-7 Messages Corresponding to Setting Area [Transceiver].



Control Unit: Gap Size (byte)

Transmission Type: Random

Duration: Repeat 100,000,000 count

Total 11,117,276.480s, 1,800,000,000 frames

Test Pattern: Word16 5555

| | No. | Size (byte) | Gap Size (byte) | Name | Error | Flow ID |
|---|---|---|---|---|---|---|
| On | 1 | 120-1000 | 1,500,000-579,99... | Ethernet + IPv4 + ICMPv4 (Echo) | - | 0 |
| On | 2 | 84 | 10,000 | Ethernet + IPv6 + ICMPv6 (Echo) | - | 1 |
| On | 3 | 108 | 1,200,000 | Ethernet + IPv6 + ICMPv6 (NS) | - | 2 |
| On | 4 | 64 | 18,500,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 3 |
| On | 5 | 64 | 150,000,128 | Ethernet + IPv4 | - | 4 |
| On | 6 | 64 | 6,000,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 5 |
| On | 7 | 64 | 85,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 6 |
| On | 8 | 64 | 12,000,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 7 |
| On | 9 | 64 | 9,000,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 8 |
| On | 10 | 64 | 652,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 9 |
| On | 11 | 64 | 960,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 10 |
| On | 12 | 64 | 500,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 11 |
| Off | 13 | 64 | 1,500,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 12 |
| Off | 14 | 64 | 600,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 13 |
| Off | 15 | 64 | 800,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 14 |
| Off | 16 | 64 | 5,600,000 | Ethernet + IPv4 + ICMPv4 (Echo) | - | 15 |

**Figure3.2.2-1 Messages Corresponding to Setting Area [Stream]**

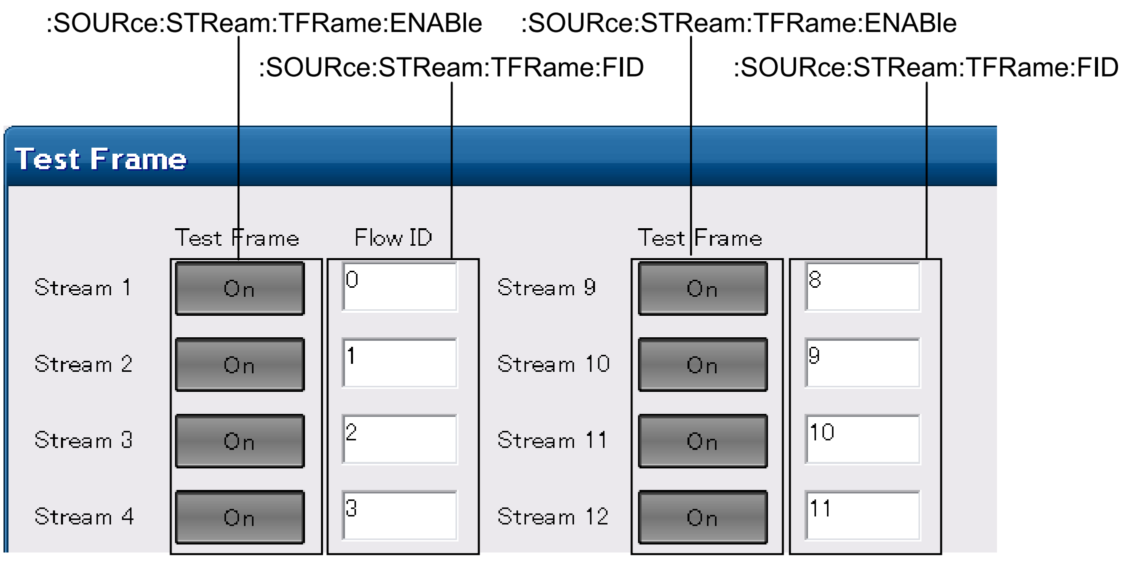


**Figure 3.2.2-2 Messages Corresponding to Stream-TestFrame Screen**

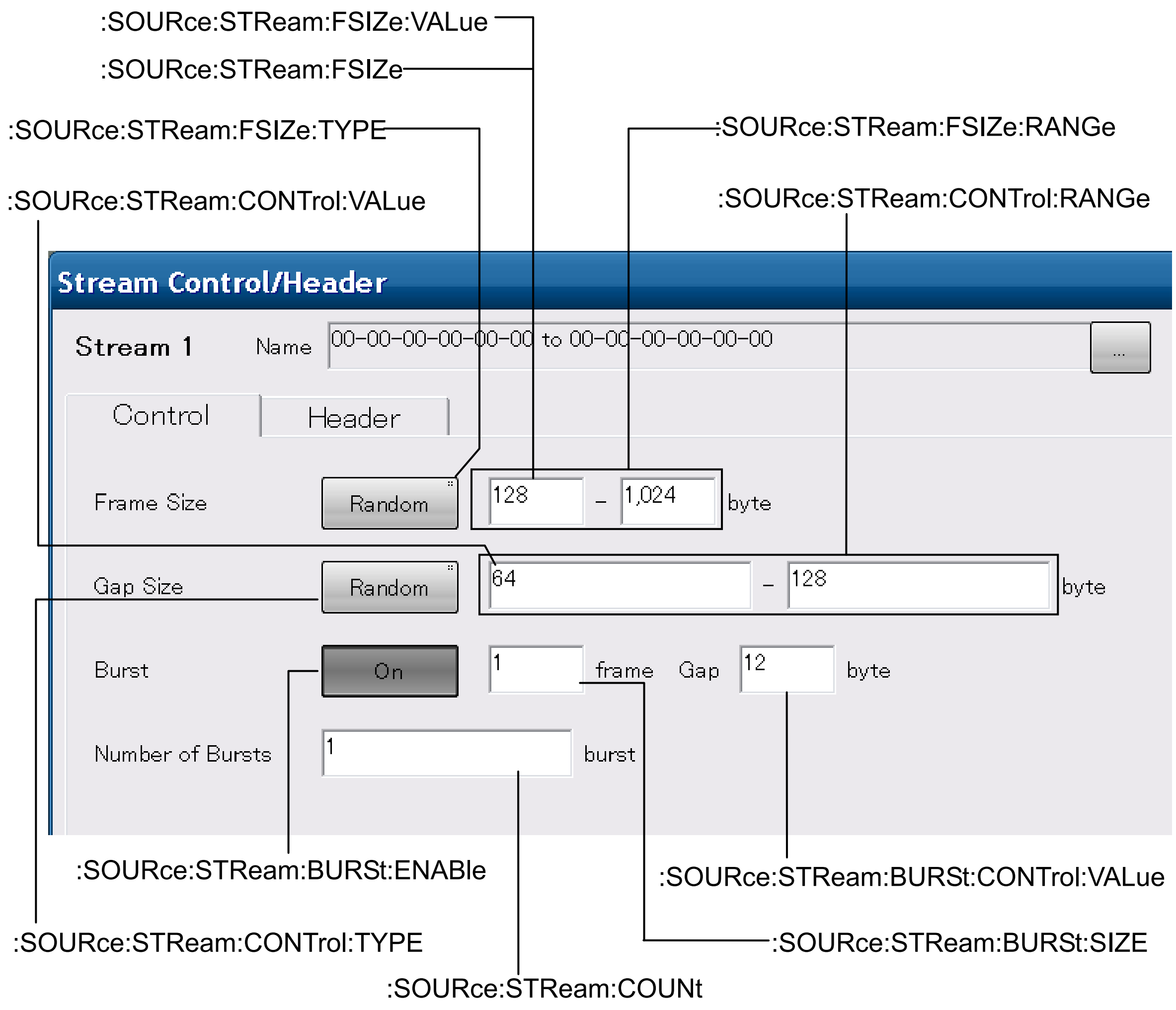
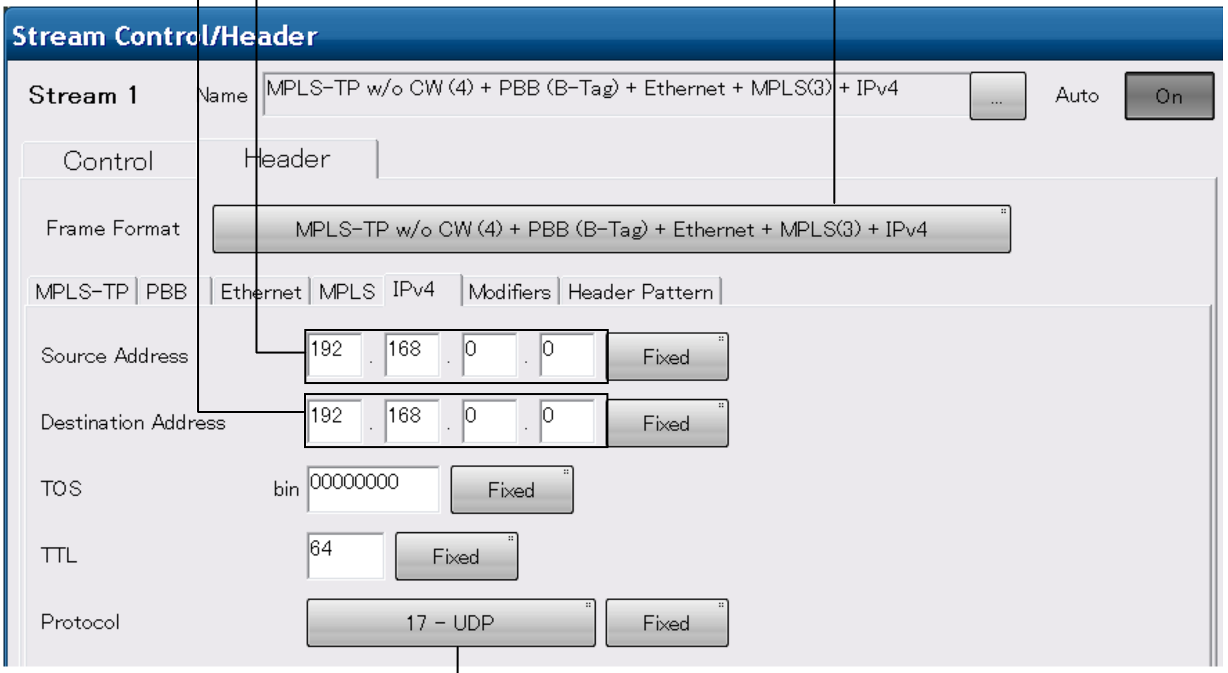

:SOURce:STReam:FSIZe:VALue
:SOURce:STReam:FSIZe
:SOURce:STReam:FSIZe:TYPE
:SOURce:STReam:FSIZe:RANGe
:SOURce:STReam:CONTrol:VALue
:SOURce:STReam:CONTrol:RANGe
Stream Control/Header
Stream 1
Name
00-00-00-00-00-00 to 00-00-00-00-00-00
Control
Header
Frame Size
Random
128
1,024
byte
Gap Size
Random
64
128
byte
Burst
On
1
frame
Gap
12
byte
Number of Bursts
1
burst
:SOURce:STReam:BURSt:ENABle
:SOURce:STReam:BURSt:CONTrol:VALue
:SOURce:STReam:CONTrol:TYPE
:SOURce:STReam:BURSt:SIZE
:SOURce:STReam:COUNt

**Figure 3.2.2-3 Messages Corresponding to Stream Screen–Control Tab**

:SOURce:STReam:HEADer:ETHer:DA

:SOURce:STReam:HEADer:ETHer:SA

:SOURce:STReam:HEADer


Stream Control/Header

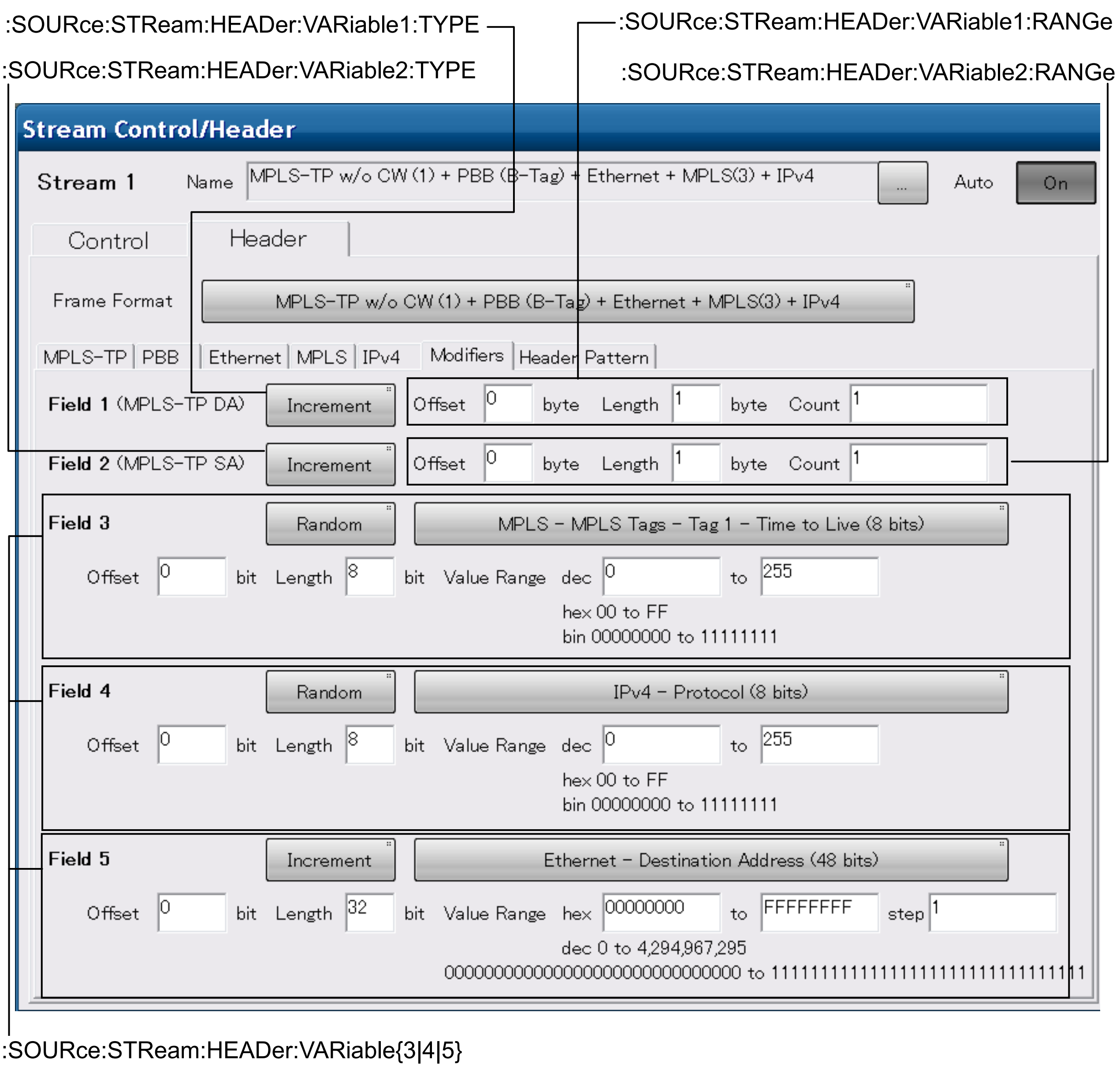
Stream 1 Name MPLS-TP w/o CW (4) + PBB (B-Tag) + Ethernet + MPLS(3) + IPv4 Auto On

Frame Format: MPLS-TP w/o CW (4) + PBB (B-Tag) + Ethernet + MPLS(3) + IPv4

| | | |
|---|---|---|
| Source Address | 192 . 168 . 0 . 0 | Fixed |
| Destination Address | 192 . 168 . 0 . 0 | Fixed |
| TOS | bin 00000000 | Fixed |
| TTL | 64 | Fixed |
| Protocol | 17 - UDP | Fixed |

:SOURce:STReam:HEADer:ETHer:TYPE

**Figure 3.2.2-4 Messages Corresponding to Stream Screen-Header Tab 1**



:SOURce:STReam:HEADer:VARiable1:TYPE

:SOURce:STReam:HEADer:VARiable2:TYPE

:SOURce:STReam:HEADer:VARiable1:RANGe

:SOURce:STReam:HEADer:VARiable2:RANGe

:SOURce:STReam:HEADer:VARiable{3|4|5}

Stream Control/Header

Stream 1 Name MPLS-TP w/o CW (1) + PBB (B-Tag) + Ethernet + MPLS(3) + IPv4 Auto On

Control | Header

Frame Format: MPLS-TP w/o CW (1) + PBB (B-Tag) + Ethernet + MPLS(3) + IPv4

MPLS-TP | PBB | Ethernet | MPLS | IPv4 | Modifiers | Header Pattern

Field 1 (MPLS-TP DA) Increment — Offset 0 byte Length 1 byte Count 1

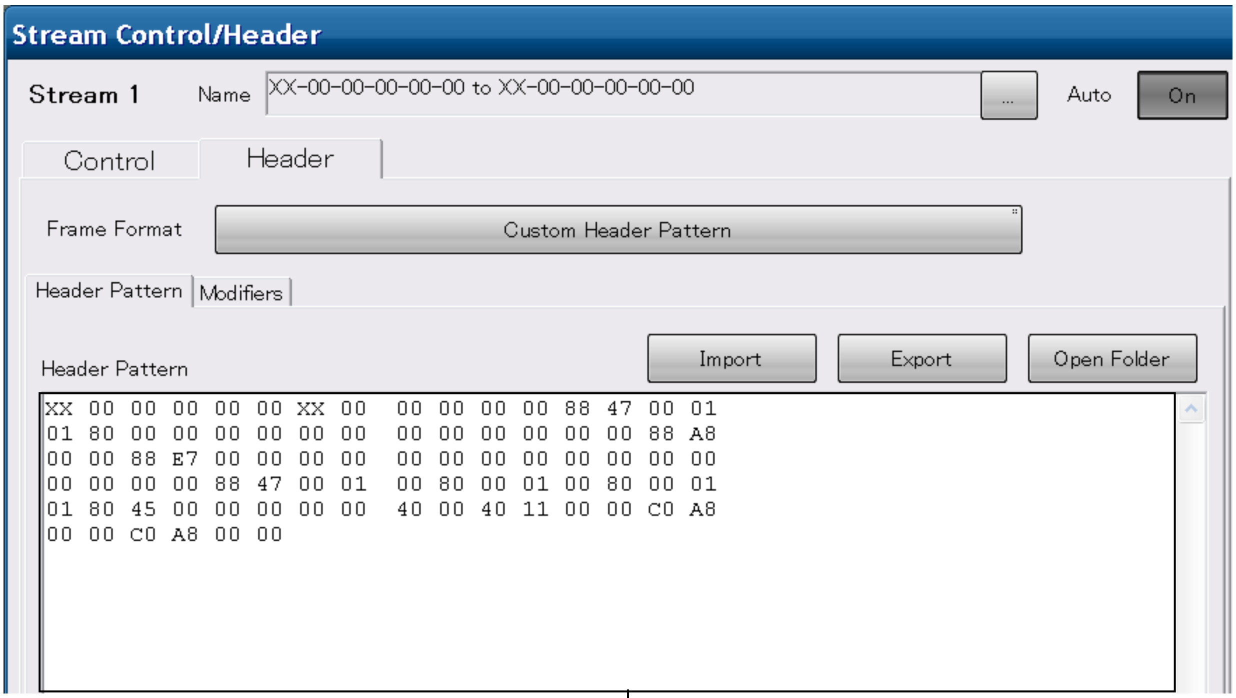
Field 2 (MPLS-TP SA) Increment — Offset 0 byte Length 1 byte Count 1

Field 3 Random — MPLS – MPLS Tags – Tag 1 – Time to Live (8 bits)
Offset 0 bit Length 8 bit Value Range dec 0 to 255
hex 00 to FF
bin 00000000 to 11111111

Field 4 Random — IPv4 – Protocol (8 bits)
Offset 0 bit Length 8 bit Value Range dec 0 to 255
hex 00 to FF
bin 00000000 to 11111111

Field 5 Increment — Ethernet – Destination Address (48 bits)
Offset 0 bit Length 32 bit Value Range hex 00000000 to FFFFFFFF step 1
dec 0 to 4,294,967,295
00000000000000000000000000000000 to 11111111111111111111111111111111

**Figure 3.2.2-5 Messages Corresponding to Stream Screen-Header Tab 2**



Name: XX-00-00-00-00-00 to XX-00-00-00-00-00

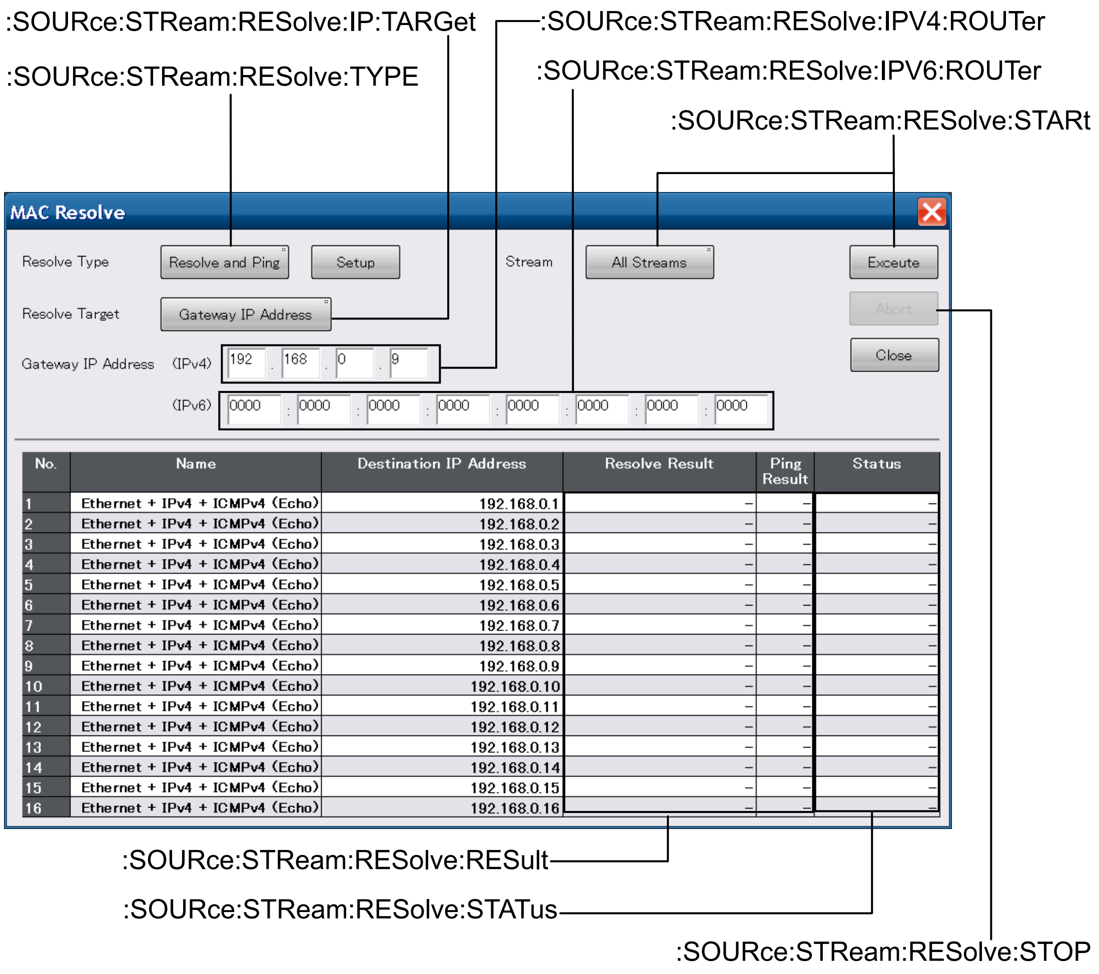
Frame Format: Custom Header Pattern

Header Pattern

```
XX 00 00 00 00 00 XX 00  00 00 00 00 88 47 00 01
01 80 00 00 00 00 00 00  00 00 00 00 00 00 88 A8
00 00 88 E7 00 00 00 00  00 00 00 00 00 00 00 00
00 00 00 00 88 47 00 01  00 80 00 01 00 80 00 01
01 80 45 00 00 00 00 00  40 00 40 11 00 00 C0 A8
00 00 C0 A8 00 00
```

:SOURce:STReam:HEADer

**Figure 3.2.2-6 Messages Corresponding to Stream Screen-Header Tab 3**



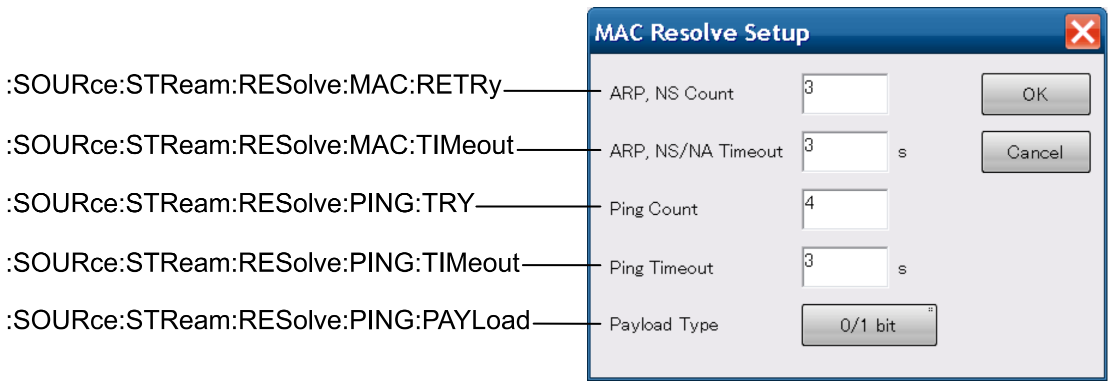
:SOURce:STReam:RESolve:IP:TARGet
:SOURce:STReam:RESolve:TYPE
:SOURce:STReam:RESolve:IPV4:ROUTer
:SOURce:STReam:RESolve:IPV6:ROUTer
:SOURce:STReam:RESolve:STARt
:SOURce:STReam:RESolve:RESult
:SOURce:STReam:RESolve:STATus
:SOURce:STReam:RESolve:STOP

MAC Resolve

Resolve Type: Resolve and Ping — Setup
Stream: All Streams
Resolve Target: Gateway IP Address
Gateway IP Address (IPv4) 192 . 168 . 0 . 9
(IPv6) 0000 : 0000 : 0000 : 0000 : 0000 : 0000 : 0000 : 0000

| No. | Name | Destination IP Address | Resolve Result | Ping Result | Status |
|---|---|---|---|---|---|
| 1 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.1 | – | – | – |
| 2 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.2 | – | – | – |
| 3 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.3 | – | – | – |
| 4 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.4 | – | – | – |
| 5 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.5 | – | – | – |
| 6 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.6 | – | – | – |
| 7 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.7 | – | – | – |
| 8 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.8 | – | – | – |
| 9 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.9 | – | – | – |
| 10 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.10 | – | – | – |
| 11 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.11 | – | – | – |
| 12 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.12 | – | – | – |
| 13 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.13 | – | – | – |
| 14 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.14 | – | – | – |
| 15 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.15 | – | – | – |
| 16 | Ethernet + IPv4 + ICMPv4 (Echo) | 192.168.0.16 | – | – | – |

**Figure 3.2.2-7 Messages Corresponding to Stream Screen-MAC Resolve**



:SOURce:STReam:RESolve:MAC:RETRy — ARP, NS Count: 3
:SOURce:STReam:RESolve:MAC:TIMeout — ARP, NS/NA Timeout: 3 s
:SOURce:STReam:RESolve:PING:TRY — Ping Count: 4
:SOURce:STReam:RESolve:PING:TIMeout — Ping Timeout: 3 s
:SOURce:STReam:RESolve:PING:PAYLoad — Payload Type: 0/1 bit

**Figure 3.2.2-8 Messages Corresponding to Stream Screen-MAC Setup**

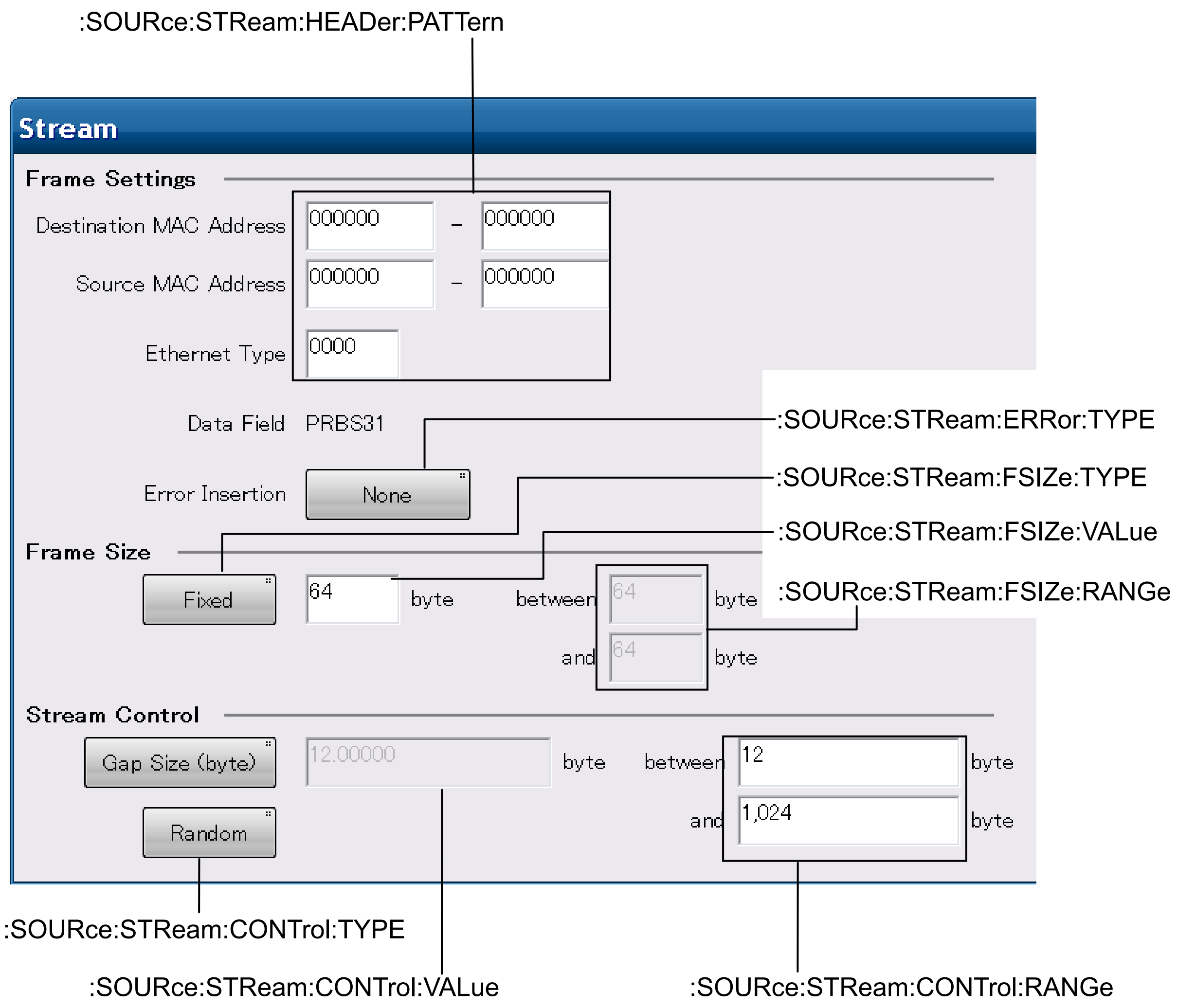


**Figure 3.2.2-9 Messages Corresponding to Setting Area [Stream] (Frame BER)**

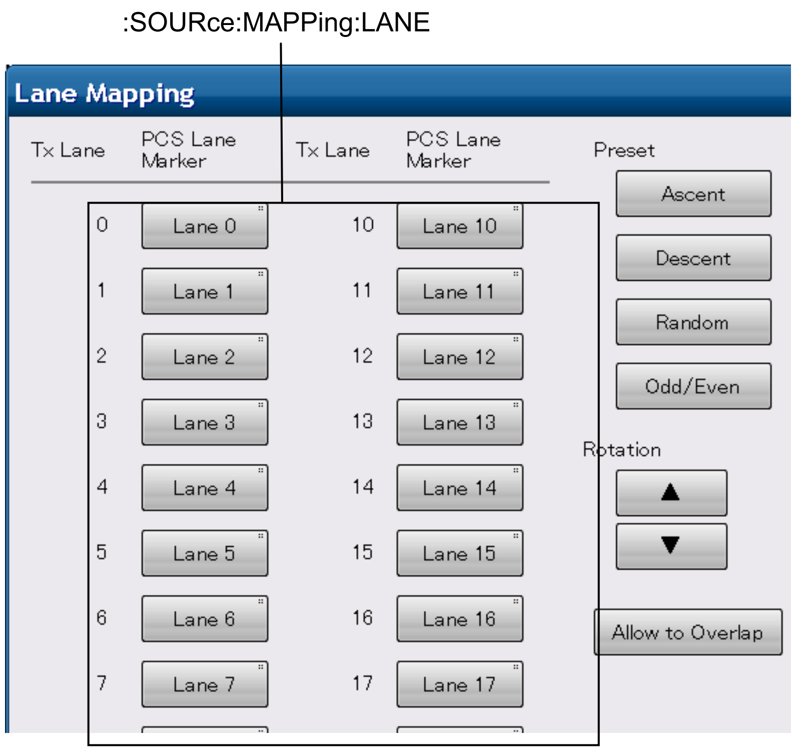


Figure 3.2.2-10 Messages Corresponding to Setting Area [Lane Mapping]

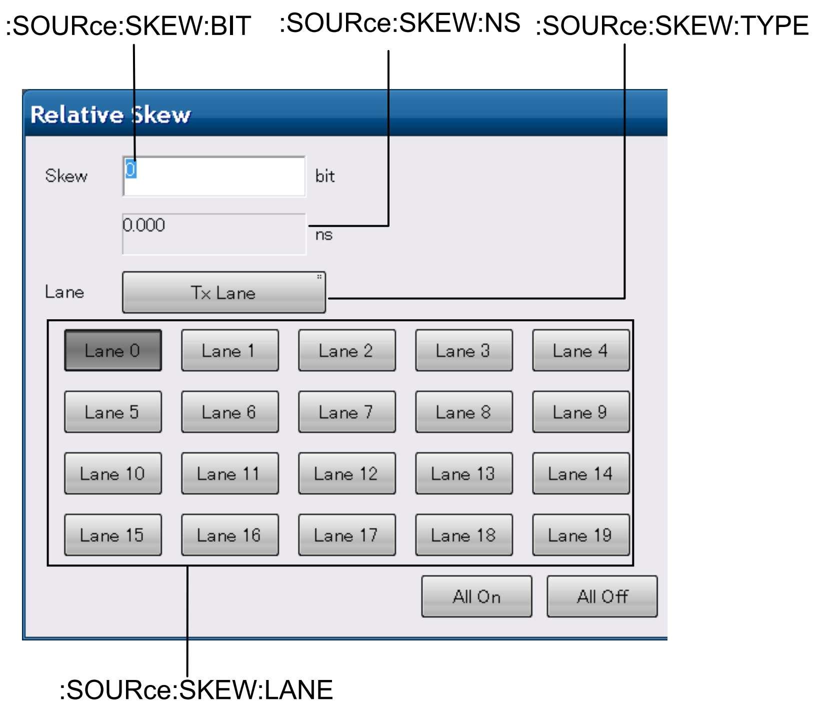


Figure 3.2.2-11 Messages Corresponding to Setting Area [Relative Skew]

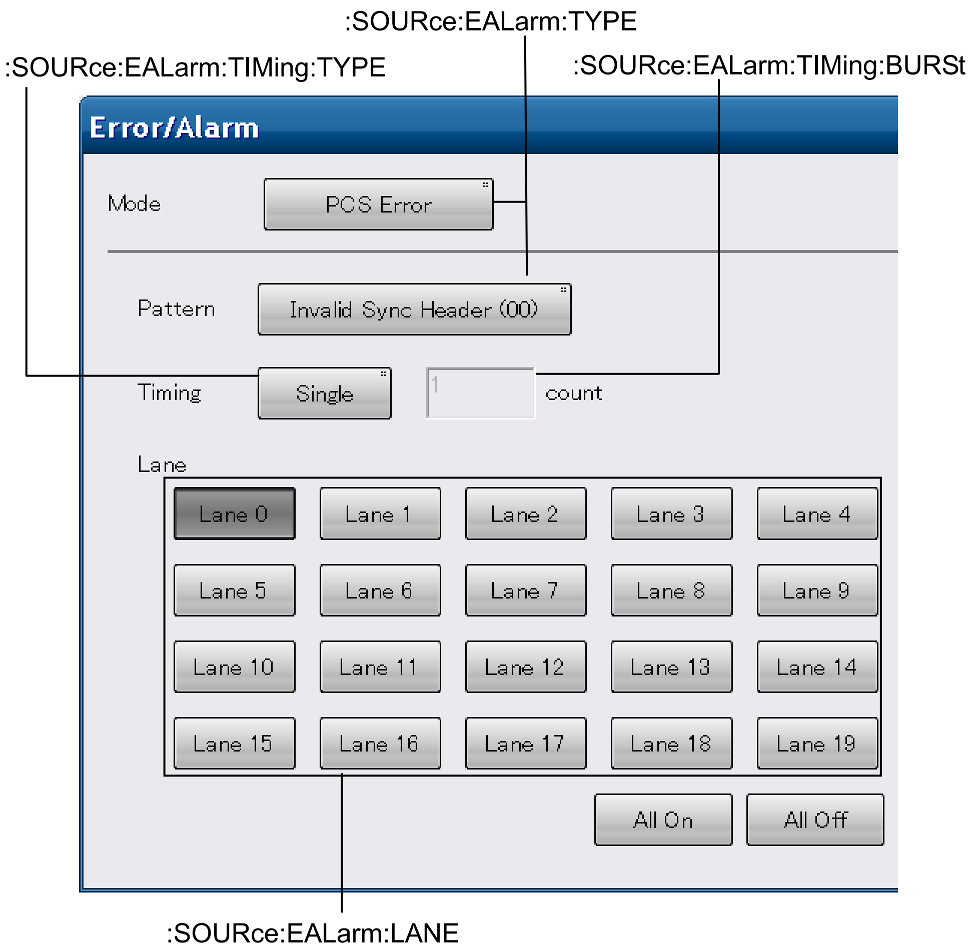


**Figure 3.2.2-12　Messages Corresponding to Setting Area [Error/Alarm]**

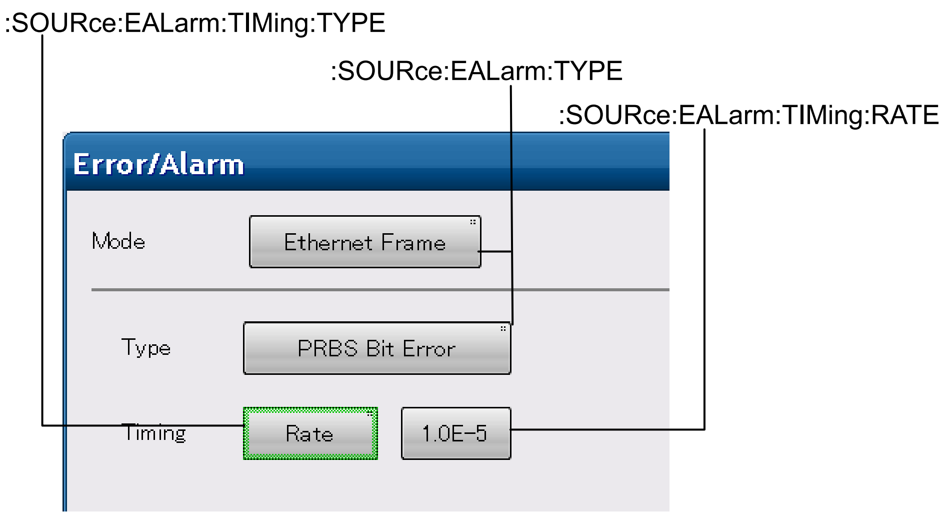
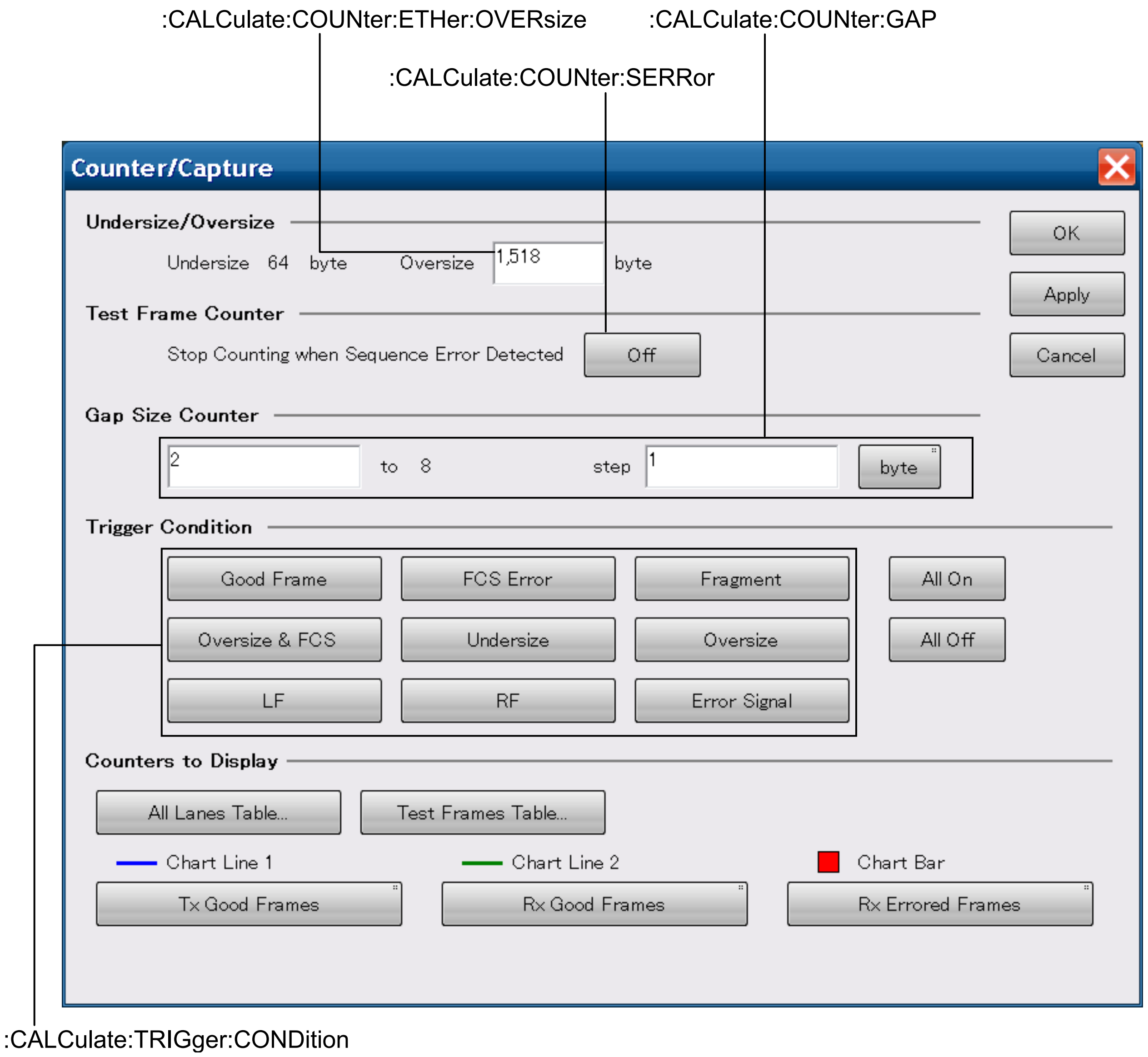


**Figure 3.2.2-13　Messages Corresponding to Setting Area [Error/Alarm] (Rate)**



Undersize/Oversize
Undersize 64 byte Oversize 1,518 byte

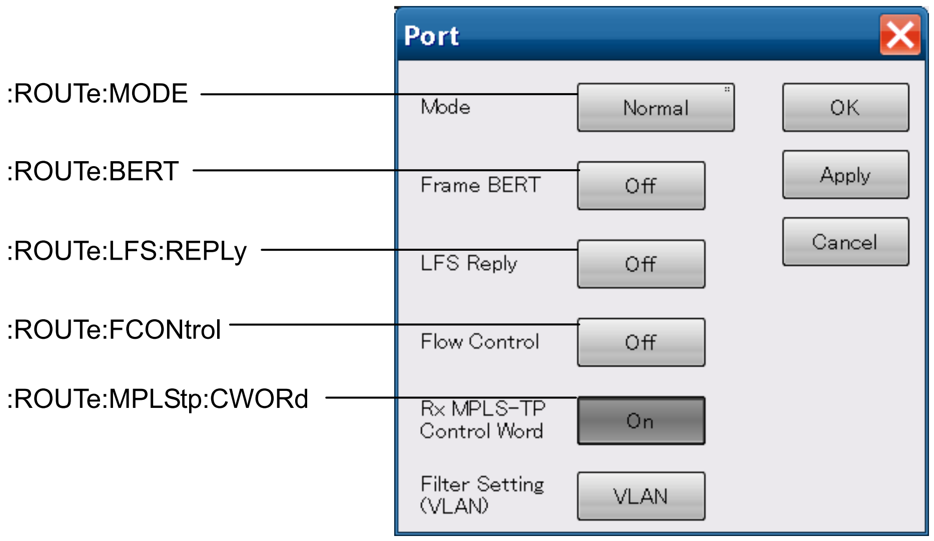
Test Frame Counter
Stop Counting when Sequence Error Detected: Off

Gap Size Counter
2 to 8 step 1 byte

Trigger Condition

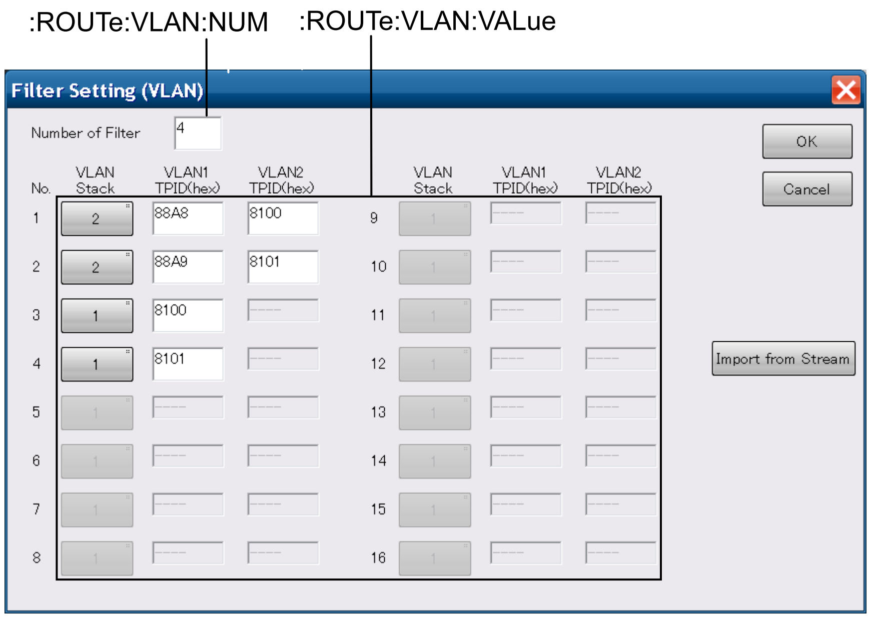
Counters to Display
Chart Line 1: Tx Good Frames
Chart Line 2: Rx Good Frames
Chart Bar: Rx Errored Frames

**Figure 3.2.2-14 Messages Corresponding to Setting Area [Capture/Counter]**



Mode: Normal
Frame BERT: Off
LFS Reply: Off
Flow Control: Off
Rx MPLS-TP Control Word: On
Filter Setting (VLAN): VLAN

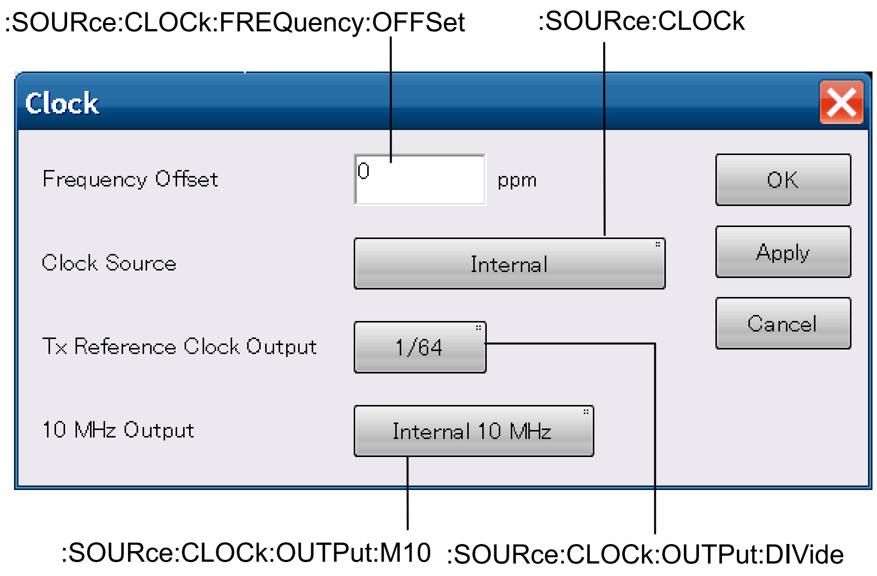
**Figure 3.2.2-15 Messages Corresponding to Setting Area [Port]**



Number of Filter: 4

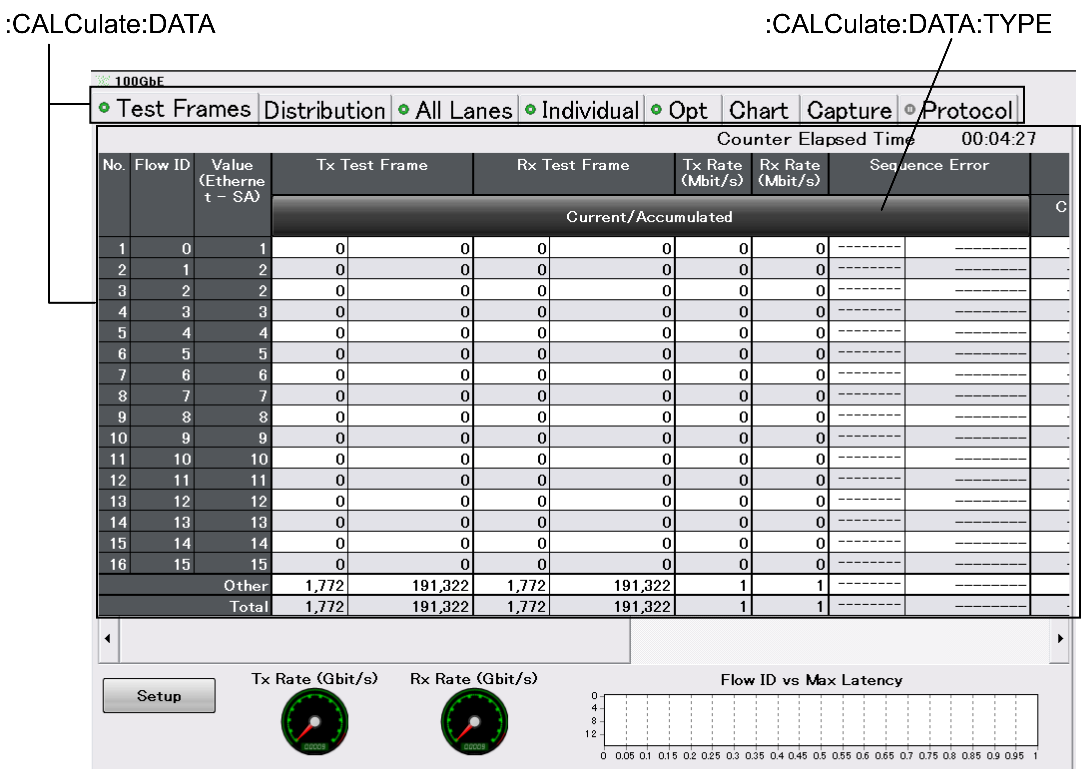
| No. | VLAN Stack | VLAN1 TPID(hex) | VLAN2 TPID(hex) | No. | VLAN Stack | VLAN1 TPID(hex) | VLAN2 TPID(hex) |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 88A8 | 8100 | 9 | 1 | ---- | ---- |
| 2 | 2 | 88A9 | 8101 | 10 | 1 | ---- | ---- |
| 3 | 1 | 8100 | ---- | 11 | 1 | ---- | ---- |
| 4 | 1 | 8101 | ---- | 12 | 1 | ---- | ---- |
| 5 | 1 | ---- | ---- | 13 | 1 | ---- | ---- |
| 6 | 1 | ---- | ---- | 14 | 1 | ---- | ---- |
| 7 | 1 | ---- | ---- | 15 | 1 | ---- | ---- |
| 8 | 1 | ---- | ---- | 16 | 1 | ---- | ---- |

**Figure 3.2.2-16 Messages Corresponding to Setting Area [Port]-Filter Setting (VLAN)**



| | | |
|---|---|---|
| Frequency Offset | 0 | ppm |
| Clock Source | Internal | |
| Tx Reference Clock Output | 1/64 | |
| 10 MHz Output | Internal 10 MHz | |

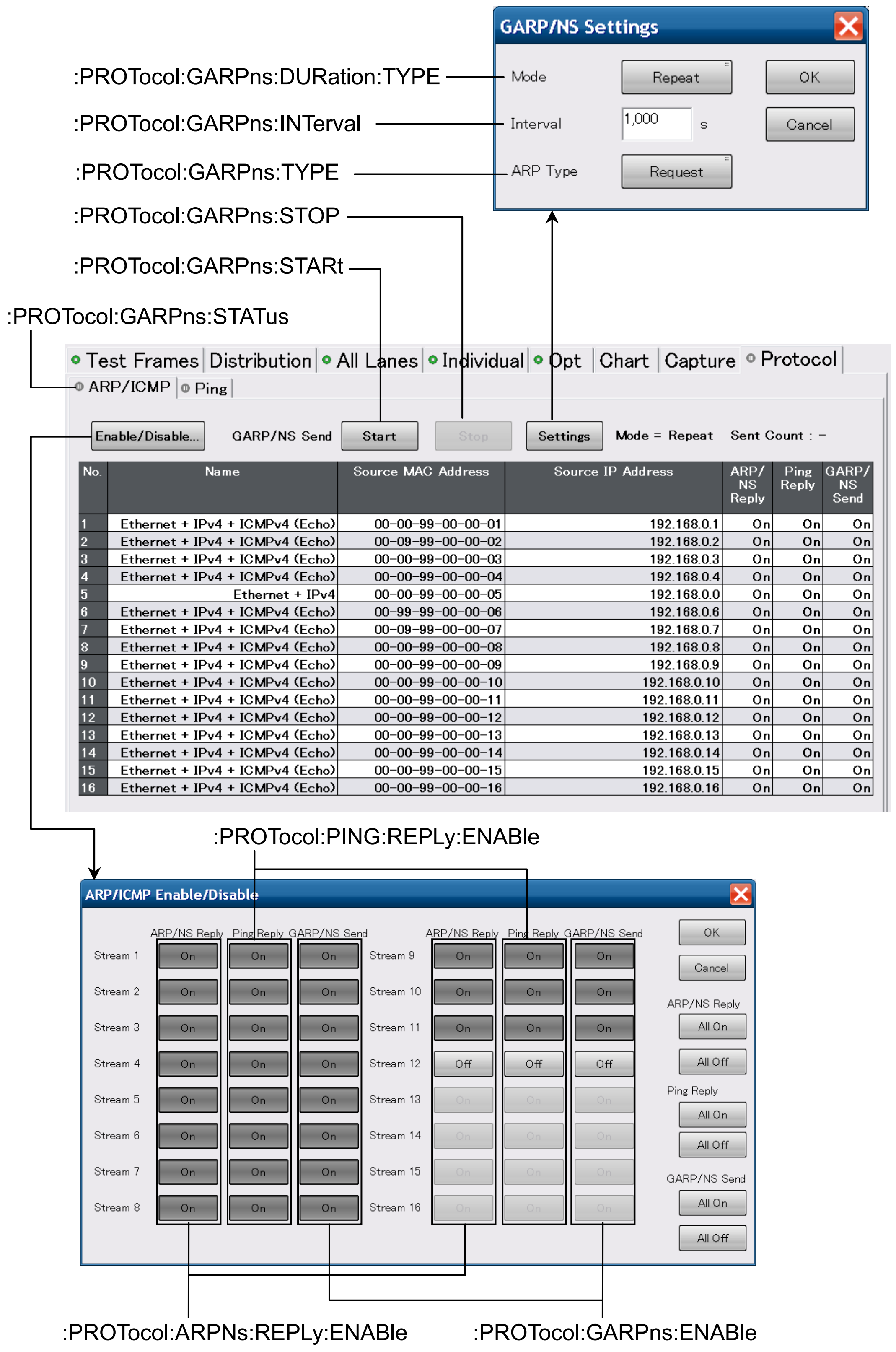
Figure 3.2.2-17 Messages Corresponding to Setting Area



Counter Elapsed Time 00:04:27

| No. | Flow ID | Value (Ethernet - SA) | Tx Test Frame | | Rx Test Frame | | Tx Rate (Mbit/s) | Rx Rate (Mbit/s) | Sequence Error | | C |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Current/Accumulated | | | | | | | | |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 2 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 3 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 4 | 3 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 5 | 4 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 6 | 5 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 7 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 8 | 7 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 9 | 8 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 10 | 9 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 11 | 10 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 12 | 11 | 11 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 13 | 12 | 12 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 14 | 13 | 13 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 15 | 14 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| 16 | 15 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | -------- | -------- | |
| | | Other | 1,772 | 191,322 | 1,772 | 191,322 | 1 | 1 | -------- | -------- | |
| | | Total | 1,772 | 191,322 | 1,772 | 191,322 | 1 | 1 | -------- | -------- | |

Figure 3.2.2-18 Messages Corresponding to Measurement Results Area



| | |
|---|---|
| Mode | Repeat |
| Interval | 1,000 s |
| ARP Type | Request |

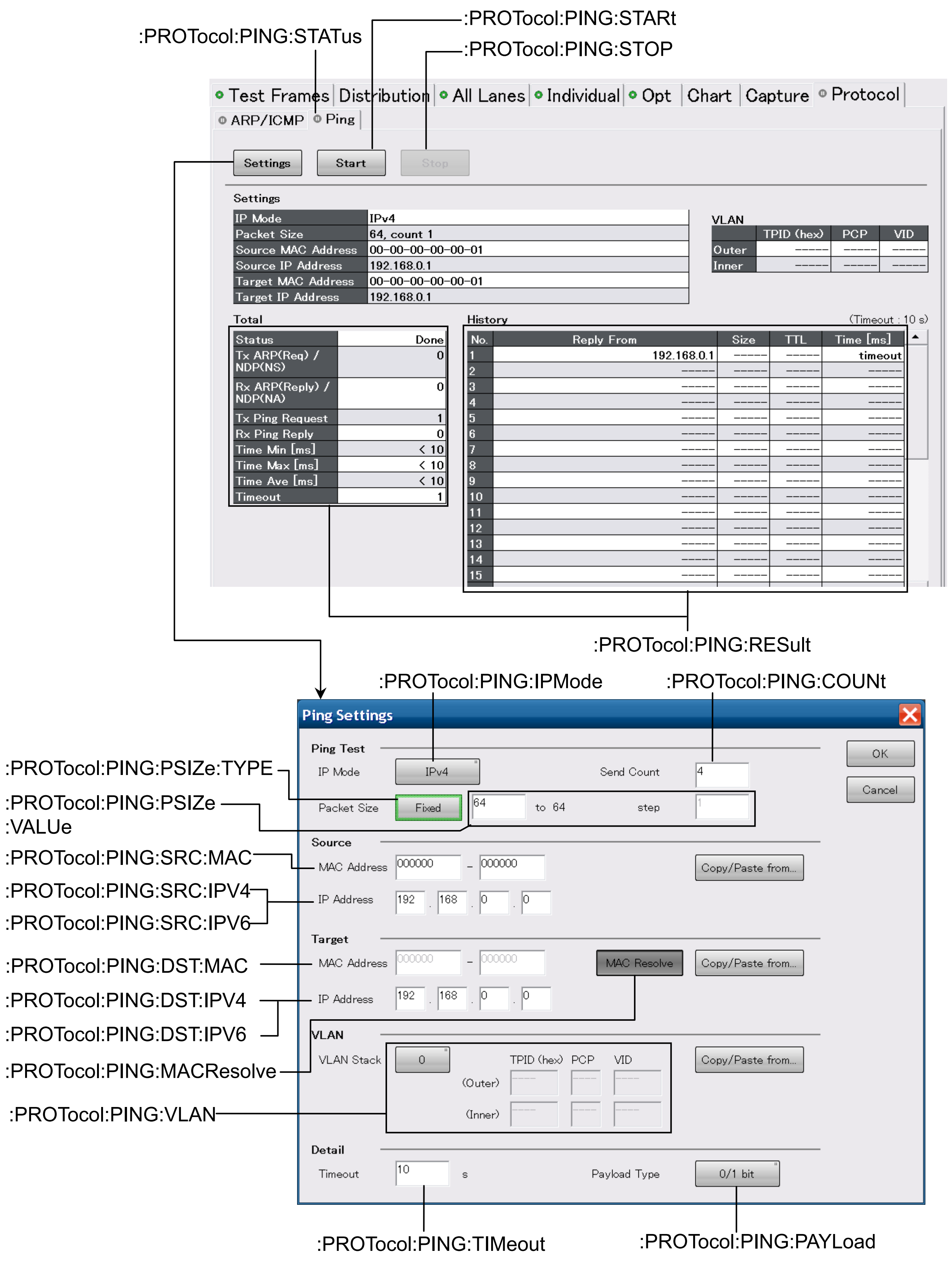
GARP/NS Send Mode = Repeat Sent Count : –

| No. | Name | Source MAC Address | Source IP Address | ARP/ NS Reply | Ping Reply | GARP/ NS Send |
|---|---|---|---|---|---|---|
| 1 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-01 | 192.168.0.1 | On | On | On |
| 2 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-09-99-00-00-02 | 192.168.0.2 | On | On | On |
| 3 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-03 | 192.168.0.3 | On | On | On |
| 4 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-04 | 192.168.0.4 | On | On | On |
| 5 | Ethernet + IPv4 | 00-00-99-00-00-05 | 192.168.0.0 | On | On | On |
| 6 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-99-99-00-00-06 | 192.168.0.6 | On | On | On |
| 7 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-09-99-00-00-07 | 192.168.0.7 | On | On | On |
| 8 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-08 | 192.168.0.8 | On | On | On |
| 9 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-09 | 192.168.0.9 | On | On | On |
| 10 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-10 | 192.168.0.10 | On | On | On |
| 11 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-11 | 192.168.0.11 | On | On | On |
| 12 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-12 | 192.168.0.12 | On | On | On |
| 13 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-13 | 192.168.0.13 | On | On | On |
| 14 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-14 | 192.168.0.14 | On | On | On |
| 15 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-15 | 192.168.0.15 | On | On | On |
| 16 | Ethernet + IPv4 + ICMPv4 (Echo) | 00-00-99-00-00-16 | 192.168.0.16 | On | On | On |

| | ARP/NS Reply | Ping Reply | GARP/NS Send | | ARP/NS Reply | Ping Reply | GARP/NS Send |
|---|---|---|---|---|---|---|---|
| Stream 1 | On | On | On | Stream 9 | On | On | On |
| Stream 2 | On | On | On | Stream 10 | On | On | On |
| Stream 3 | On | On | On | Stream 11 | On | On | On |
| Stream 4 | On | On | On | Stream 12 | Off | Off | Off |
| Stream 5 | On | On | On | Stream 13 | On | On | On |
| Stream 6 | On | On | On | Stream 14 | On | On | On |
| Stream 7 | On | On | On | Stream 15 | On | On | On |
| Stream 8 | On | On | On | Stream 16 | On | On | On |

ARP/NS Reply: All On / All Off
Ping Reply: All On / All Off
GARP/NS Send: All On / All Off

**Figure 3.2.2-19 Messages Corresponding to Measurement Results Area (Protocol-ARP/ICMP)**



Settings

| | |
|---|---|
| IP Mode | IPv4 |
| Packet Size | 64, count 1 |
| Source MAC Address | 00-00-00-00-00-01 |
| Source IP Address | 192.168.0.1 |
| Target MAC Address | 00-00-00-00-00-01 |
| Target IP Address | 192.168.0.1 |

VLAN

| | TPID (hex) | PCP | VID |
|---|---|---|---|
| Outer | ----- | ----- | ----- |
| Inner | ----- | ----- | ----- |

Total

| | |
|---|---|
| Status | Done |
| Tx ARP(Req) / NDP(NS) | 0 |
| Rx ARP(Reply) / NDP(NA) | 0 |
| Tx Ping Request | 1 |
| Rx Ping Reply | 0 |
| Time Min [ms] | < 10 |
| Time Max [ms] | < 10 |
| Time Ave [ms] | < 10 |
| Timeout | 1 |

History (Timeout : 10 s)

| No. | Reply From | Size | TTL | Time [ms] |
|---|---|---|---|---|
| 1 | 192.168.0.1 | ----- | ----- | timeout |
| 2 | ----- | ----- | ----- | ----- |
| 3 | ----- | ----- | ----- | ----- |
| 4 | ----- | ----- | ----- | ----- |
| 5 | ----- | ----- | ----- | ----- |
| 6 | ----- | ----- | ----- | ----- |
| 7 | ----- | ----- | ----- | ----- |
| 8 | ----- | ----- | ----- | ----- |
| 9 | ----- | ----- | ----- | ----- |
| 10 | ----- | ----- | ----- | ----- |
| 11 | ----- | ----- | ----- | ----- |
| 12 | ----- | ----- | ----- | ----- |
| 13 | ----- | ----- | ----- | ----- |
| 14 | ----- | ----- | ----- | ----- |
| 15 | ----- | ----- | ----- | ----- |

Ping Test
IP Mode: IPv4 Send Count: 4
Packet Size: Fixed 64 to 64 step 1

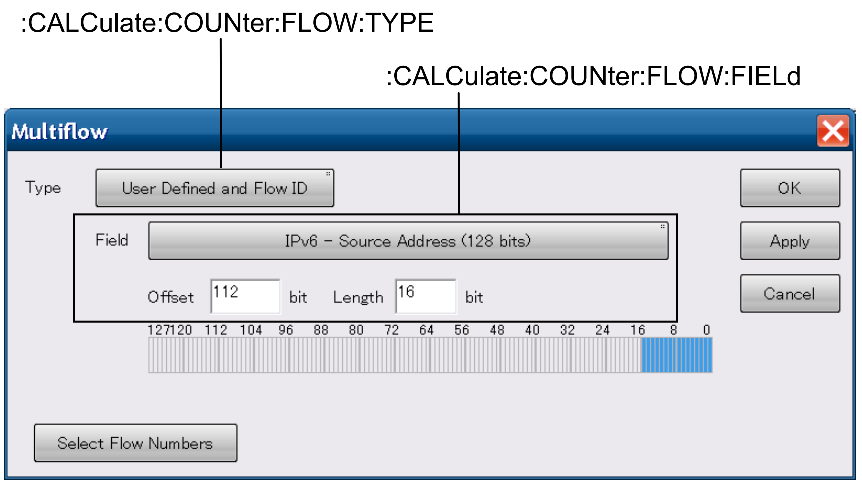
Source
MAC Address: 000000 - 000000
IP Address: 192 . 168 . 0 . 0

Target
MAC Address: 000000 - 000000
IP Address: 192 . 168 . 0 . 0

VLAN
VLAN Stack: 0 TPID (hex) PCP VID
(Outer) ---- ---- ----
(Inner) ---- ---- ----

Detail
Timeout: 10 s Payload Type: 0/1 bit

**Figure 3.2.2-20 Messages Corresponding to Measurement Results Area (Protocol-Ping)**



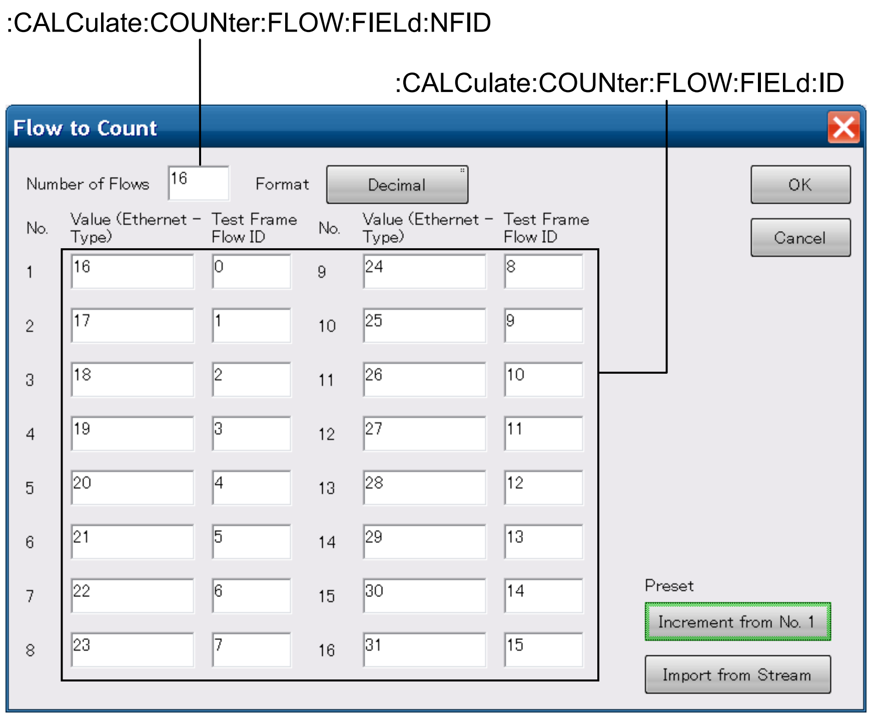
Type: User Defined and Flow ID

Field: IPv6 − Source Address (128 bits)

Offset 112 bit Length 16 bit

```
127120  112  104  96  88  80  72  64  56  48  40  32  24  16  8  0
```

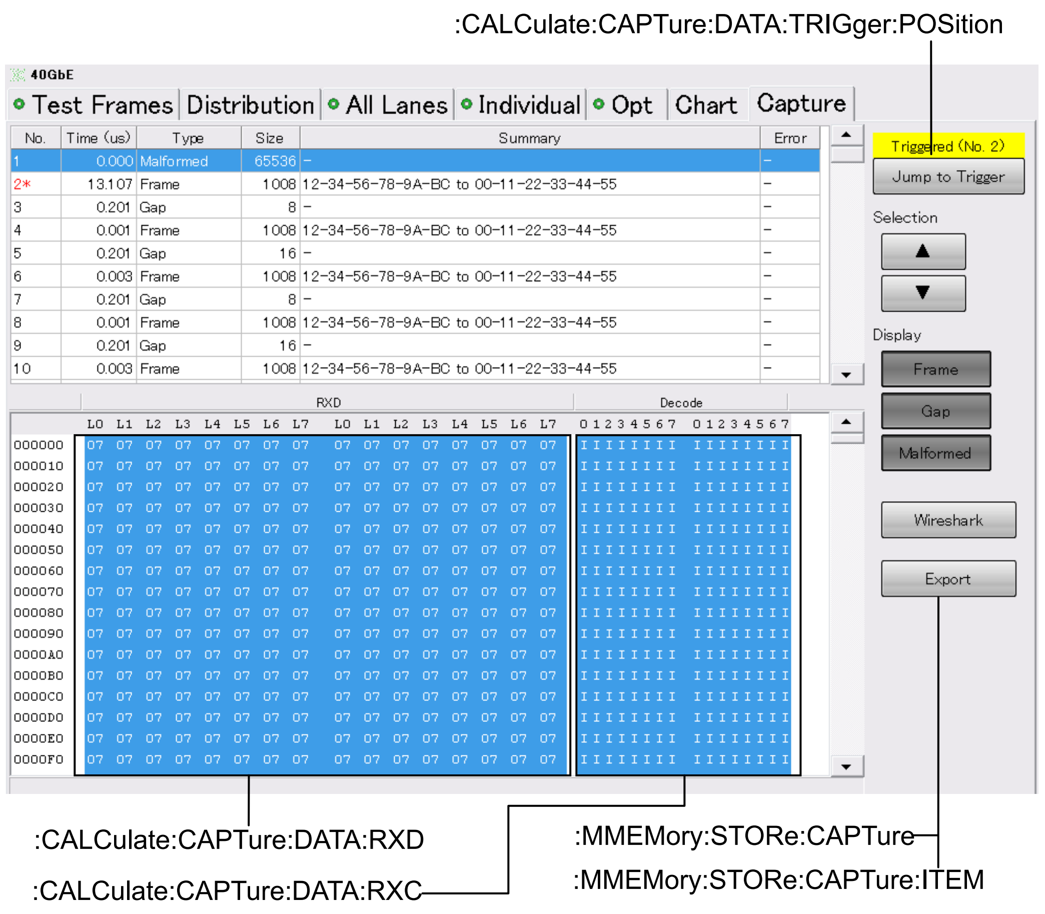
**Figure 3.2.2-21 Message Corresponding to Measurement Result Display Area (Test Frames) MultiFlow Screen**


:CALCulate:COUNter:FLOW:FIELd:NFID
:CALCulate:COUNter:FLOW:FIELd:ID
Flow to Count
OK
Cancel
Preset
Increment from No. 1
Import from Stream

Number of Flows 16 Format Decimal

| No. | Value (Ethernet − Type) | Test Frame Flow ID | No. | Value (Ethernet − Type) | Test Frame Flow ID |
|---|---|---|---|---|---|
| 1 | 16 | 0 | 9 | 24 | 8 |
| 2 | 17 | 1 | 10 | 25 | 9 |
| 3 | 18 | 2 | 11 | 26 | 10 |
| 4 | 19 | 3 | 12 | 27 | 11 |
| 5 | 20 | 4 | 13 | 28 | 12 |
| 6 | 21 | 5 | 14 | 29 | 13 |
| 7 | 22 | 6 | 15 | 30 | 14 |
| 8 | 23 | 7 | 16 | 31 | 15 |

**Figure 3.2.2-22 Message Corresponding to Measurement Result Display Area (Test Frames) Flow to Count Screen**



:CALCulate:CAPTure:DATA:TRIGger:POSition

| No. | Time (us) | Type | Size | Summary | Error |
|---|---|---|---|---|---|
| 1 | 0.000 | Malformed | 65536 | – | – |
| 2* | 13.107 | Frame | 1008 | 12-34-56-78-9A-BC to 00-11-22-33-44-55 | – |
| 3 | 0.201 | Gap | 8 | – | – |
| 4 | 0.001 | Frame | 1008 | 12-34-56-78-9A-BC to 00-11-22-33-44-55 | – |
| 5 | 0.201 | Gap | 16 | – | – |
| 6 | 0.003 | Frame | 1008 | 12-34-56-78-9A-BC to 00-11-22-33-44-55 | – |
| 7 | 0.201 | Gap | 8 | – | – |
| 8 | 0.001 | Frame | 1008 | 12-34-56-78-9A-BC to 00-11-22-33-44-55 | – |
| 9 | 0.201 | Gap | 16 | – | – |
| 10 | 0.003 | Frame | 1008 | 12-34-56-78-9A-BC to 00-11-22-33-44-55 | – |

```
                                  RXD                                       Decode
        L0 L1 L2 L3 L4 L5 L6 L7   L0 L1 L2 L3 L4 L5 L6 L7   0 1 2 3 4 5 6 7  0 1 2 3 4 5 6 7
000000  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000010  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000020  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000030  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000040  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000050  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000060  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000070  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000080  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
000090  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
0000A0  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
0000B0  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
0000C0  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
0000D0  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
0000E0  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
0000F0  07 07 07 07 07 07 07 07   07 07 07 07 07 07 07 07   I I I I I I I I  I I I I I I I I
```

:CALCulate:CAPTure:DATA:RXD

:CALCulate:CAPTure:DATA:RXC

:MMEMory:STORe:CAPTure

:MMEMory:STORe:CAPTure:ITEM

**Figure 3.2.2-23 Messages Corresponding to Measurement Results Area (Capture)**

### 3.2.3　Messages Corresponding to OTU3/OTU4

For the Messages Corresponding to the setting area of [MDIO] and [Transceiver], refer to Figure 3.2.1-7 Messages Corresponding to Setting Area [Transceiver], and Figure 3.2.2-12 Messages Corresponding to Setting Area [Clock].

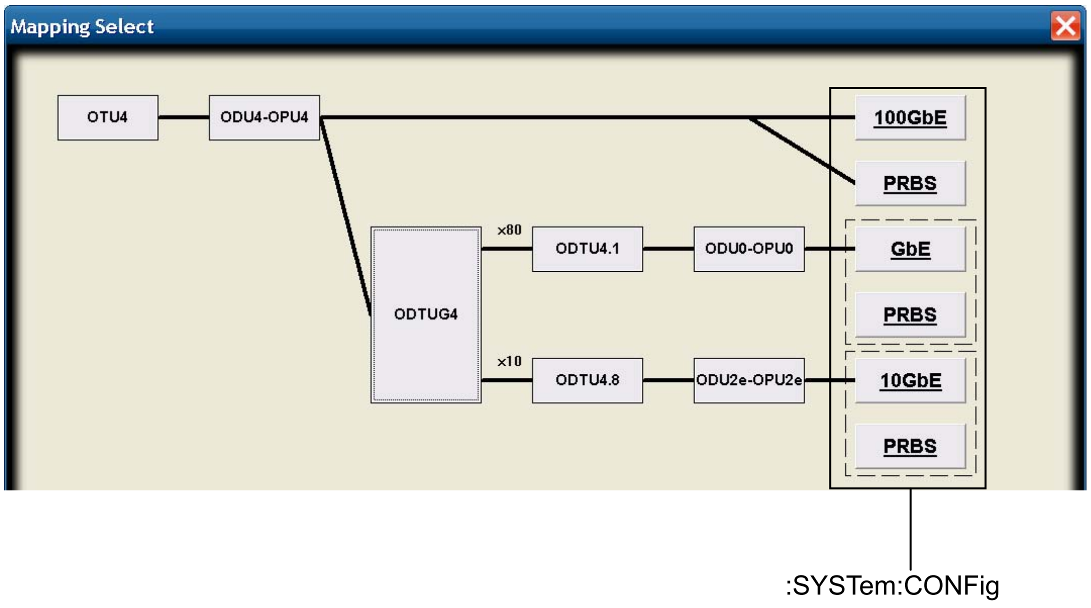


**Figure 3.2.3-1　Messages Corresponding to [Mapping Select] (OTU4)**

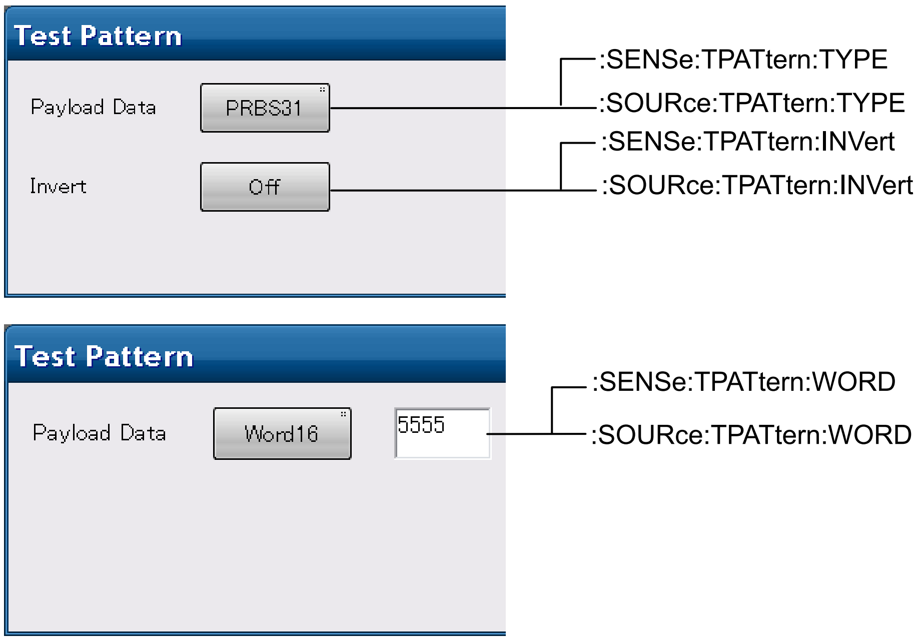


**Figure 3.2.3-2　Messages Corresponding to [Test Pattern]**

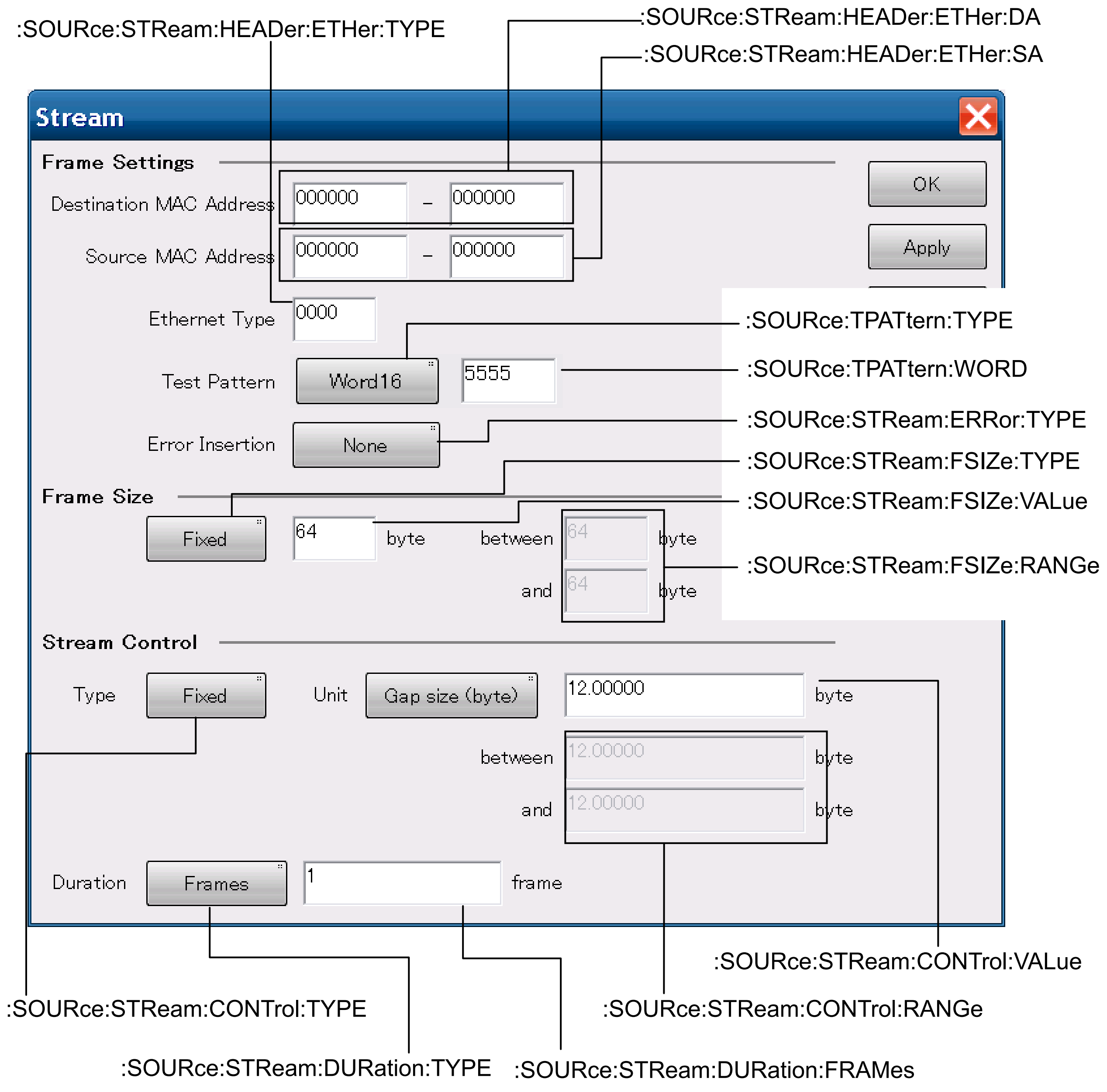
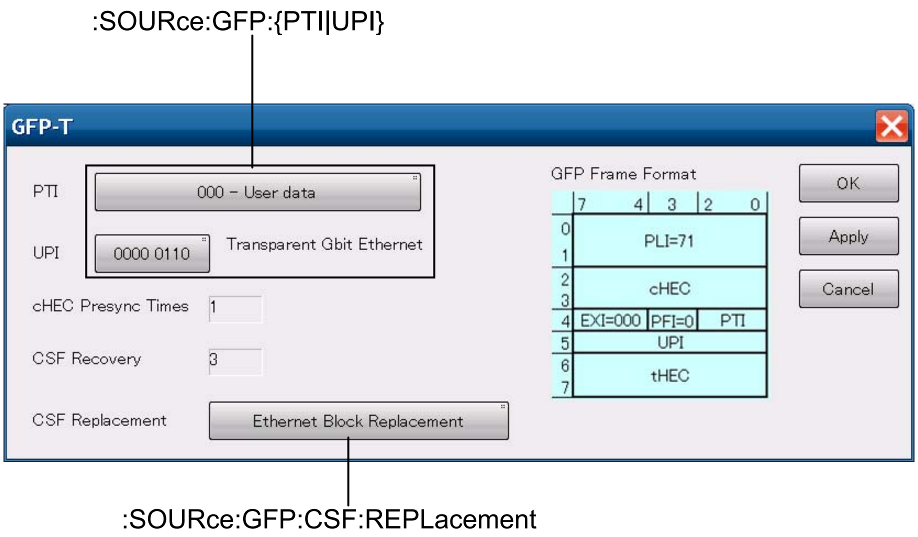
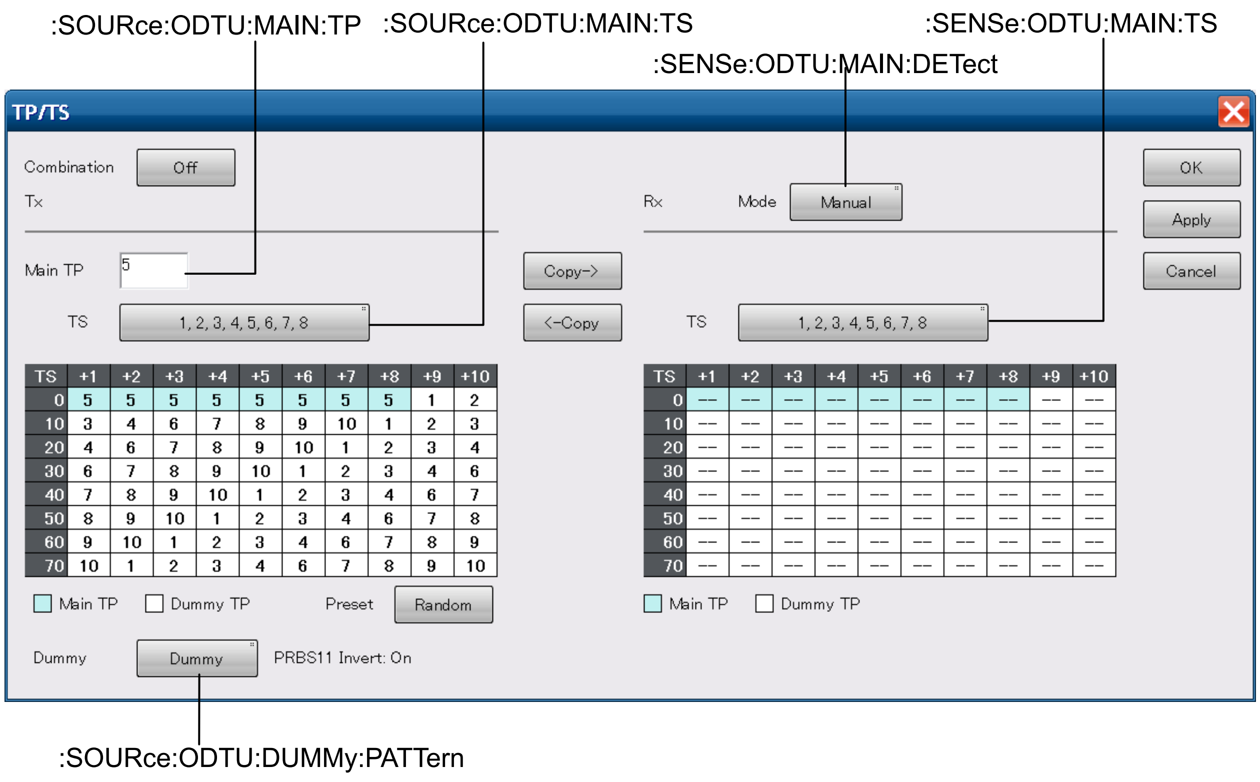


**Figure 3.2.3-3 Messages Corresponding to Operation Area [Stream]**



| | |
|---|---|
| PTI | 000 – User data |
| UPI | 0000 0110 Transparent Gbit Ethernet |
| cHEC Presync Times | 1 |
| CSF Recovery | 3 |
| CSF Replacement | Ethernet Block Replacement |

**Figure 3.2.3-4 Messages Corresponding to Setting Area [GFP-T]**



Combination: Off

Tx

Main TP: 5

TS: 1, 2, 3, 4, 5, 6, 7, 8

| TS | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 1 | 2 |
| 10 | 3 | 4 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 |
| 20 | 4 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 |
| 30 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 6 |
| 40 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 6 | 7 |
| 50 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 6 | 7 | 8 |
| 60 | 9 | 10 | 1 | 2 | 3 | 4 | 6 | 7 | 8 | 9 |
| 70 | 10 | 1 | 2 | 3 | 4 | 6 | 7 | 8 | 9 | 10 |

Preset: Random

Dummy: Dummy PRBS11 Invert: On

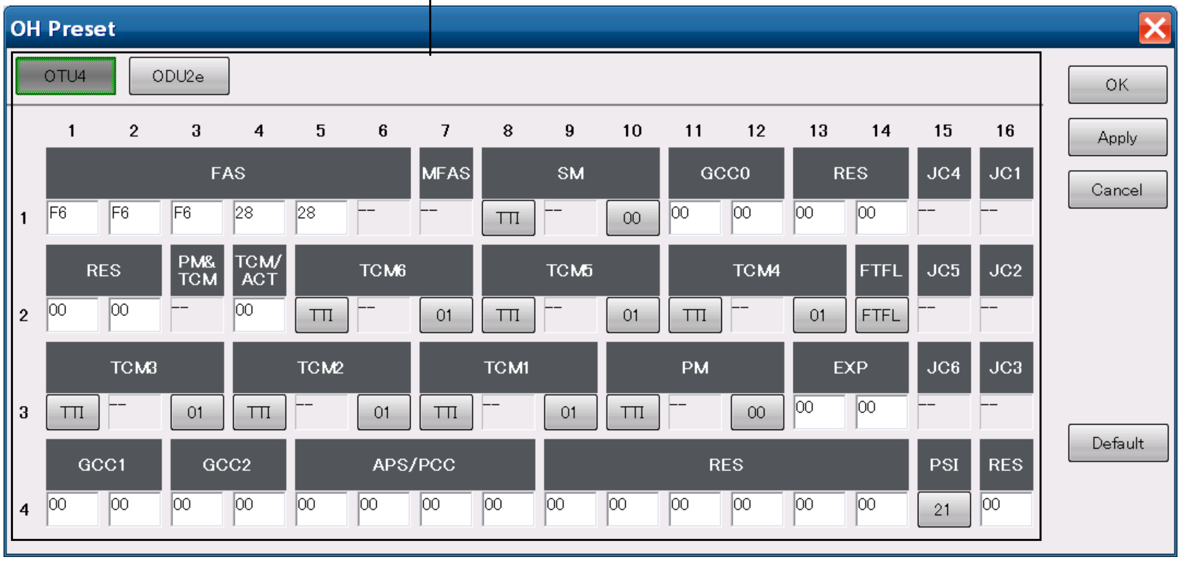
Rx Mode: Manual

TS: 1, 2, 3, 4, 5, 6, 7, 8

| TS | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 10 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 20 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 30 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 40 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 50 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 60 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |
| 70 | –– | –– | –– | –– | –– | –– | –– | –– | –– | –– |

**Figure 3.2.3-5 Messages Corresponding to Setting Area [TP/TS]**

:SOURce:OTN:OH[:L]


OH Preset

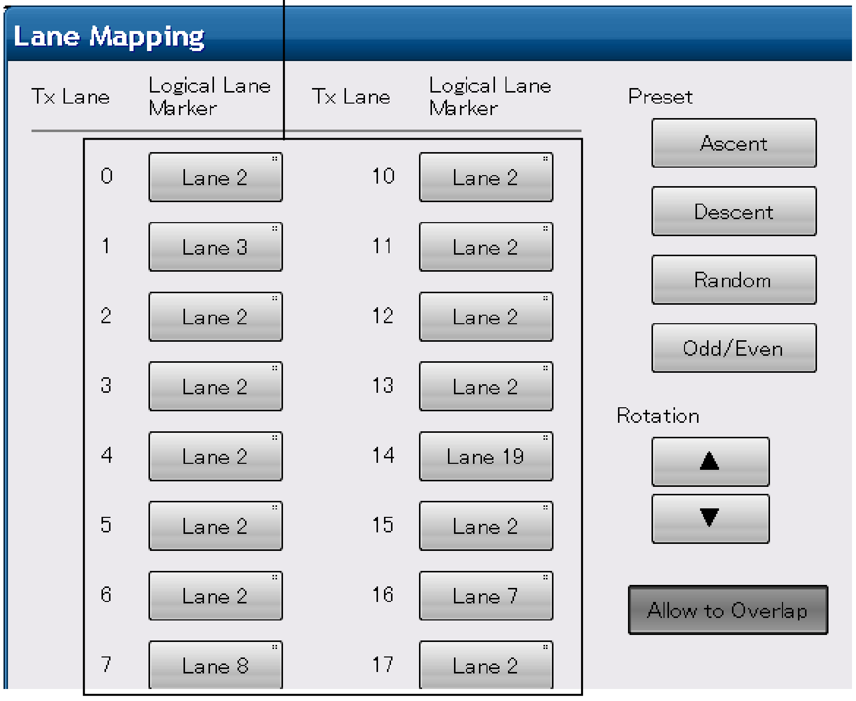
OTU4 ODU2e

|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | FAS |  |  |  |  |  | MFAS | SM |  |  | GCC0 |  | RES |  | JC4 | JC1 |
| 1 | F6 | F6 | F6 | 28 | 28 | -- | -- | TTI | -- | 00 | 00 | 00 | 00 | 00 | -- | -- |
|  | RES |  | PM& TCM | TCM/ ACT | TCM6 |  |  | TCM5 |  |  | TCM4 |  |  | FTFL | JC5 | JC2 |
| 2 | 00 | 00 | -- | 00 | TTI | -- | 01 | TTI | -- | 01 | TTI | -- | 01 | FTFL | -- | -- |
|  | TCM3 |  |  | TCM2 |  |  | TCM1 |  |  | PM |  |  | EXP |  | JC6 | JC3 |
| 3 | TTI | -- | 01 | TTI | -- | 01 | TTI | -- | 01 | TTI | -- | 00 | 00 | 00 | -- | -- |
|  | GCC1 |  | GCC2 |  | APS/PCC |  |  |  | RES |  |  |  |  |  | PSI | RES |
| 4 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 21 | 00 |

OK Apply Cancel Default

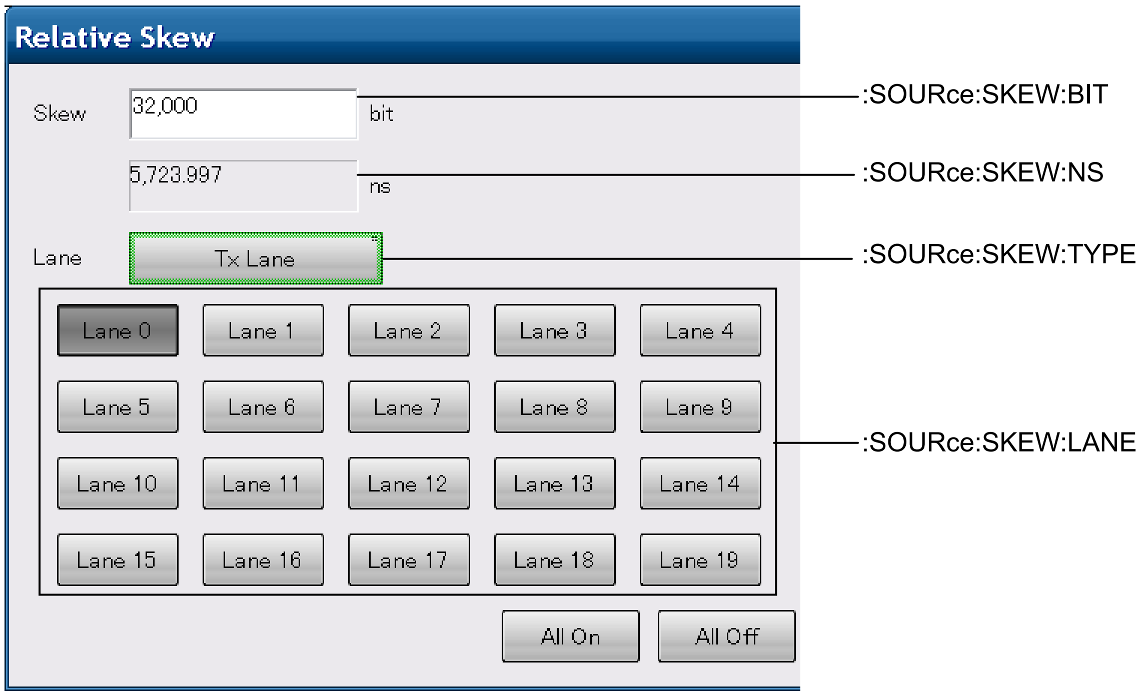
Figure 3.2.3-6 Messages Corresponding to Operation Area [OH Preset]

:SOURce:MAPPing:LANE


Lane Mapping

| Tx Lane | Logical Lane Marker | Tx Lane | Logical Lane Marker |
|---|---|---|---|
| 0 | Lane 2 | 10 | Lane 2 |
| 1 | Lane 3 | 11 | Lane 2 |
| 2 | Lane 2 | 12 | Lane 2 |
| 3 | Lane 2 | 13 | Lane 2 |
| 4 | Lane 2 | 14 | Lane 19 |
| 5 | Lane 2 | 15 | Lane 2 |
| 6 | Lane 2 | 16 | Lane 7 |
| 7 | Lane 8 | 17 | Lane 2 |

Figure 3.2.3-7 Messages Corresponding to Operation Area [Lane Mapping]



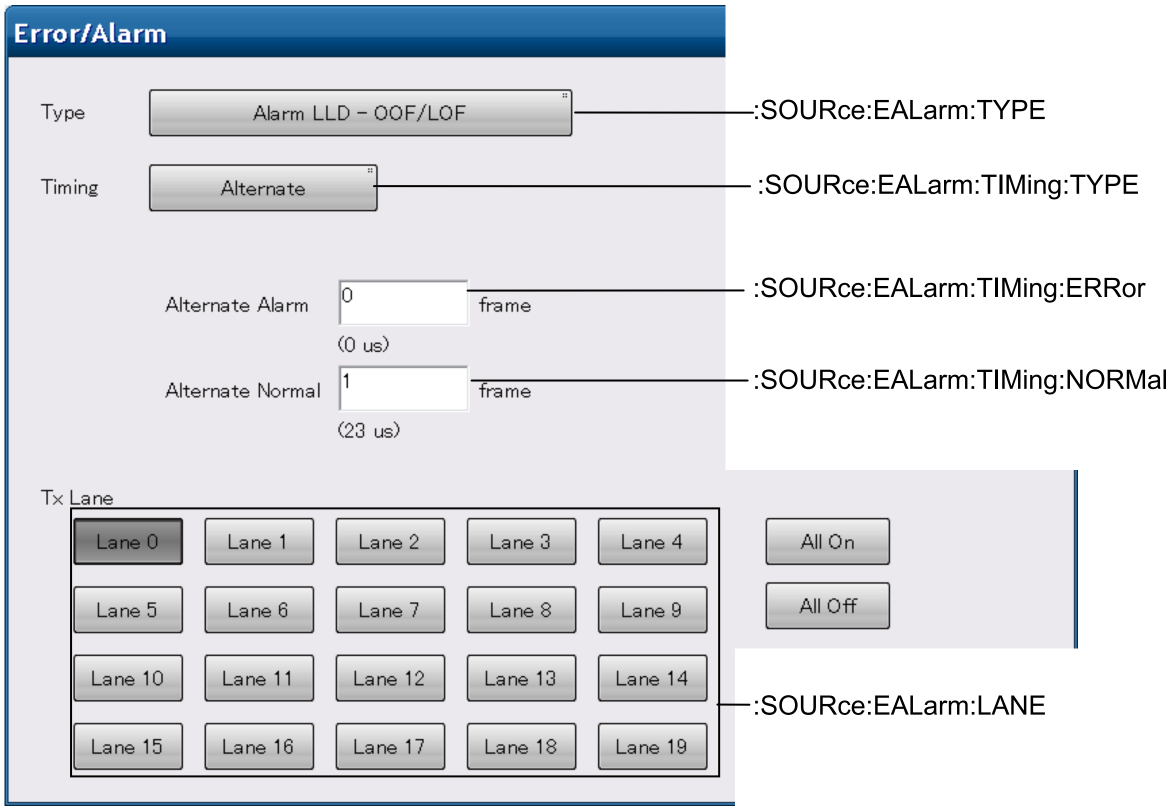
:SOURce:SKEW:BIT

:SOURce:SKEW:NS

:SOURce:SKEW:TYPE

:SOURce:SKEW:LANE

**Figure 3.2.3-8 Messages Corresponding to Setting Area [Relative Skew]**



:SOURce:EALarm:TYPE

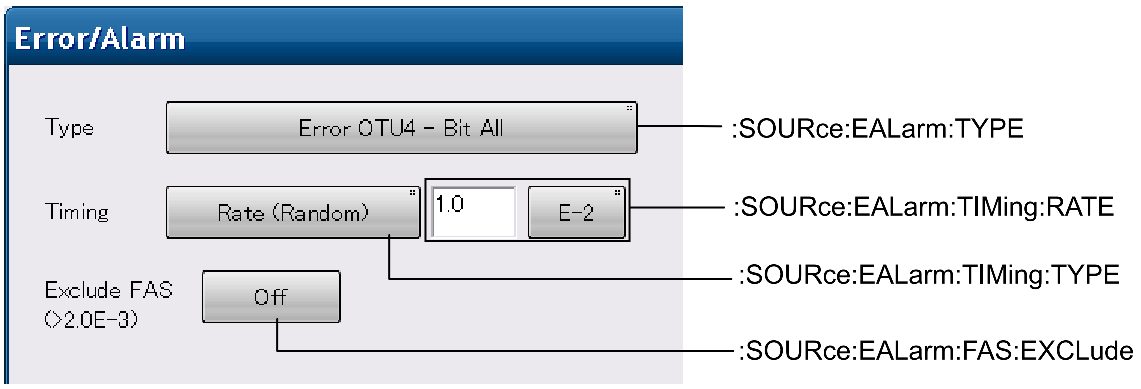
:SOURce:EALarm:TIMing:TYPE

:SOURce:EALarm:TIMing:ERRor

:SOURce:EALarm:TIMing:NORMal

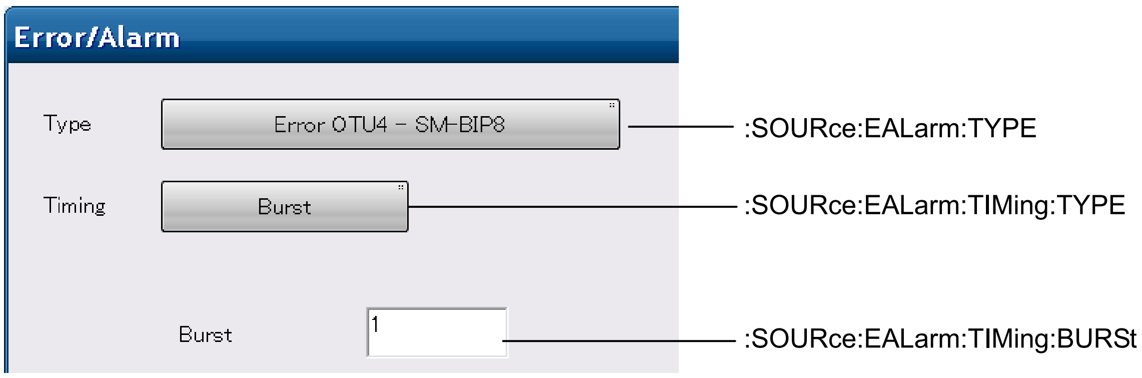
:SOURce:EALarm:LANE

**Figure 3.2.3-9 Messages Corresponding to Setting Area [Error/Alarm] (Alternate)**



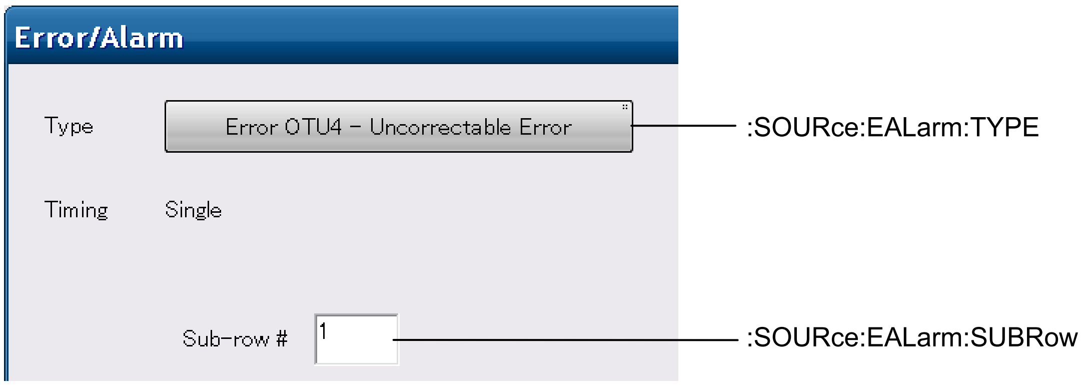
:SOURce:EALarm:TYPE
:SOURce:EALarm:TIMing:RATE
:SOURce:EALarm:TIMing:TYPE
:SOURce:EALarm:FAS:EXCLude

**Figure 3.2.3-10 Messages Corresponding to Setting Area [Error/Alarm] (Rate)**



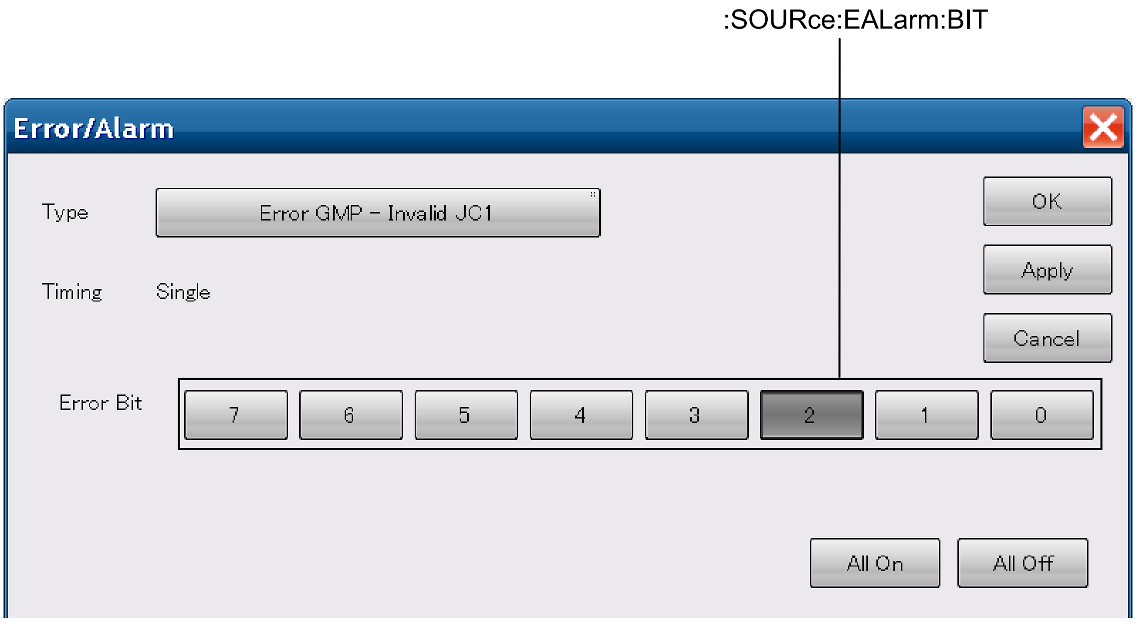
:SOURce:EALarm:TYPE
:SOURce:EALarm:TIMing:TYPE
:SOURce:EALarm:TIMing:BURSt

**Figure 3.2.3-11 Messages Corresponding to Setting Area [Error/Alarm] (Burst)**



:SOURce:EALarm:TYPE
:SOURce:EALarm:SUBRow

**Figure 3.2.3-12 Messages Corresponding to Setting Area [Error/Alarm] (Subrow)**

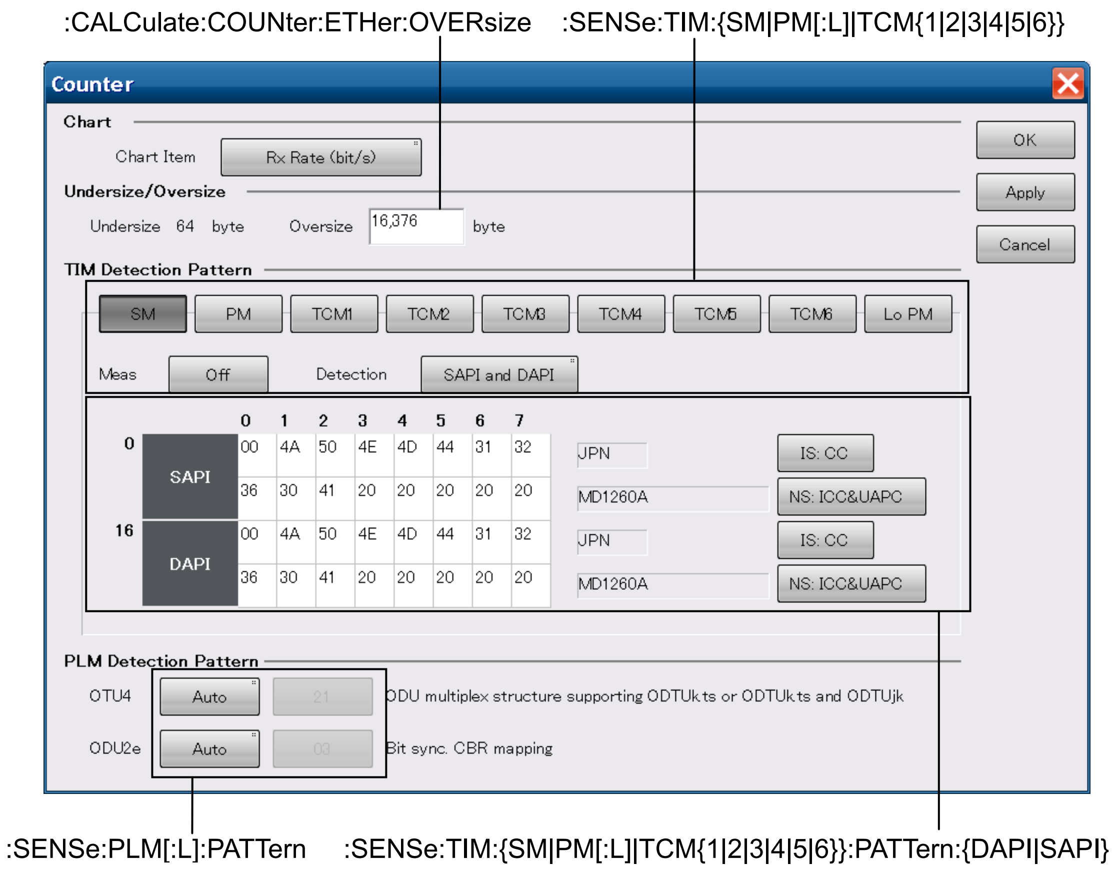


Type: Error GMP – Invalid JC1

Timing: Single

Error Bit: 7 6 5 4 3 [✓ 2] 1 0

Figure 3.2.3-13 Messages Corresponding to Setting Area [Error/Alarm] (Bit)


:CALCulate:COUNter:ETHer:OVERsize
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}
:SENSe:PLM[:L]:PATTern
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}:PATTern:{DAPI|SAPI}
Counter
OK
Apply
Cancel

Chart

Chart Item: Rx Rate (bit/s)

Undersize/Oversize

Undersize 64 byte Oversize 16,376 byte

TIM Detection Pattern

[✓ SM] PM TCM1 TCM2 TCM3 TCM4 TCM5 TCM6 Lo PM

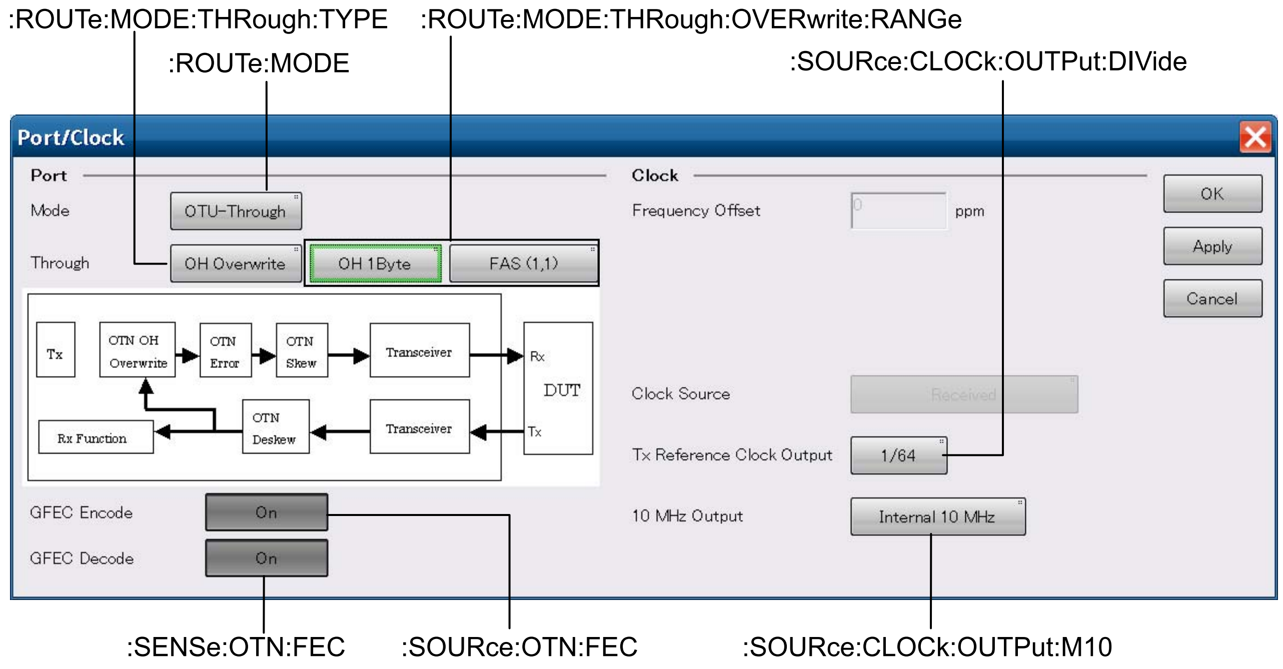
Meas: Off Detection: SAPI and DAPI

| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | SAPI | 00 | 4A | 50 | 4E | 4D | 44 | 31 | 32 | JPN | IS: CC |
| | | 36 | 30 | 41 | 20 | 20 | 20 | 20 | 20 | MD1260A | NS: ICC&UAPC |
| 16 | DAPI | 00 | 4A | 50 | 4E | 4D | 44 | 31 | 32 | JPN | IS: CC |
| | | 36 | 30 | 41 | 20 | 20 | 20 | 20 | 20 | MD1260A | NS: ICC&UAPC |

PLM Detection Pattern

| | | | |
|---|---|---|---|
| OTU4 | Auto | 21 | ODU multiplex structure supporting ODTUk.ts or ODTUk.ts and ODTUjk |
| ODU2e | Auto | 03 | Bit sync. CBR mapping |

Figure 3.2.3-14 Messages Corresponding to Setting Area [Counter]



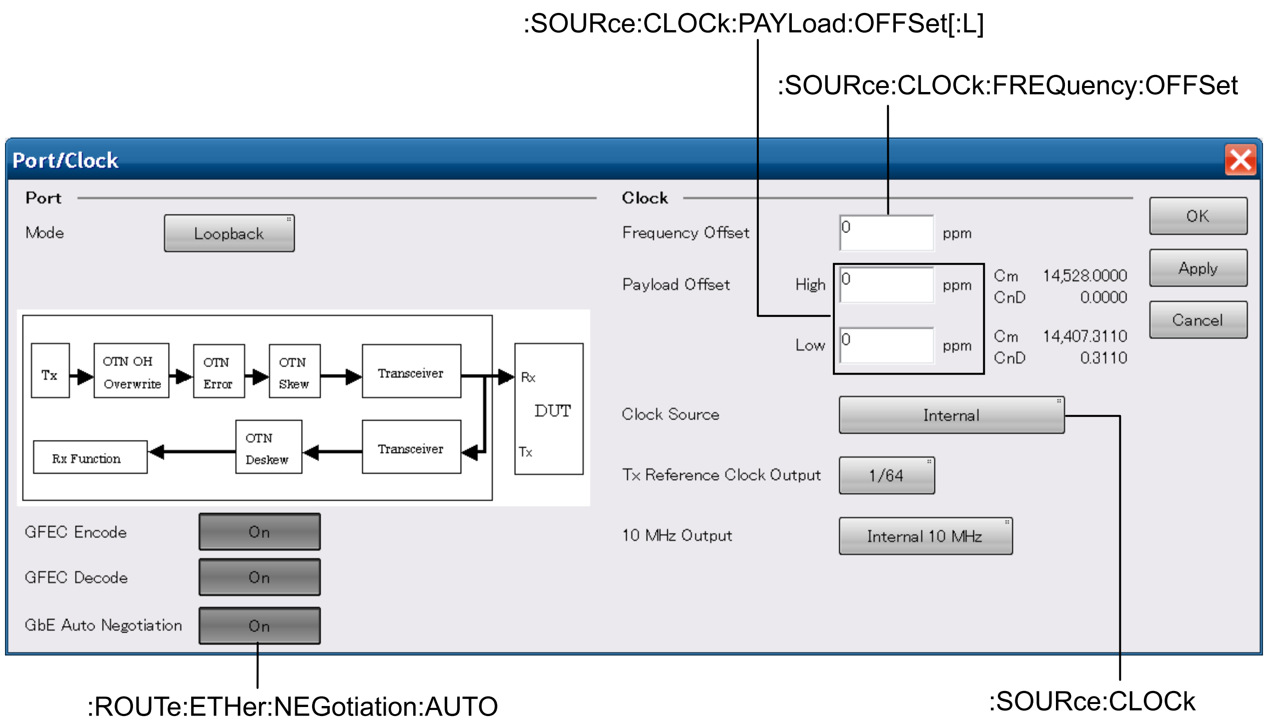
:ROUTe:MODE:THRough:TYPE
:ROUTe:MODE
:ROUTe:MODE:THRough:OVERwrite:RANGe
:SOURce:CLOCk:OUTPut:DIVide
:SENSe:OTN:FEC
:SOURce:OTN:FEC
:SOURce:CLOCk:OUTPut:M10

Port/Clock

Port
Mode: OTU-Through
Through: OH Overwrite, OH 1Byte, FAS (1,1)
GFEC Encode: On
GFEC Decode: On

Clock
Frequency Offset: 0 ppm
Clock Source: Received
Tx Reference Clock Output: 1/64
10 MHz Output: Internal 10 MHz

**Figure 3.2.3-15 Messages Corresponding to Setting Area [Port/Clock] 1**



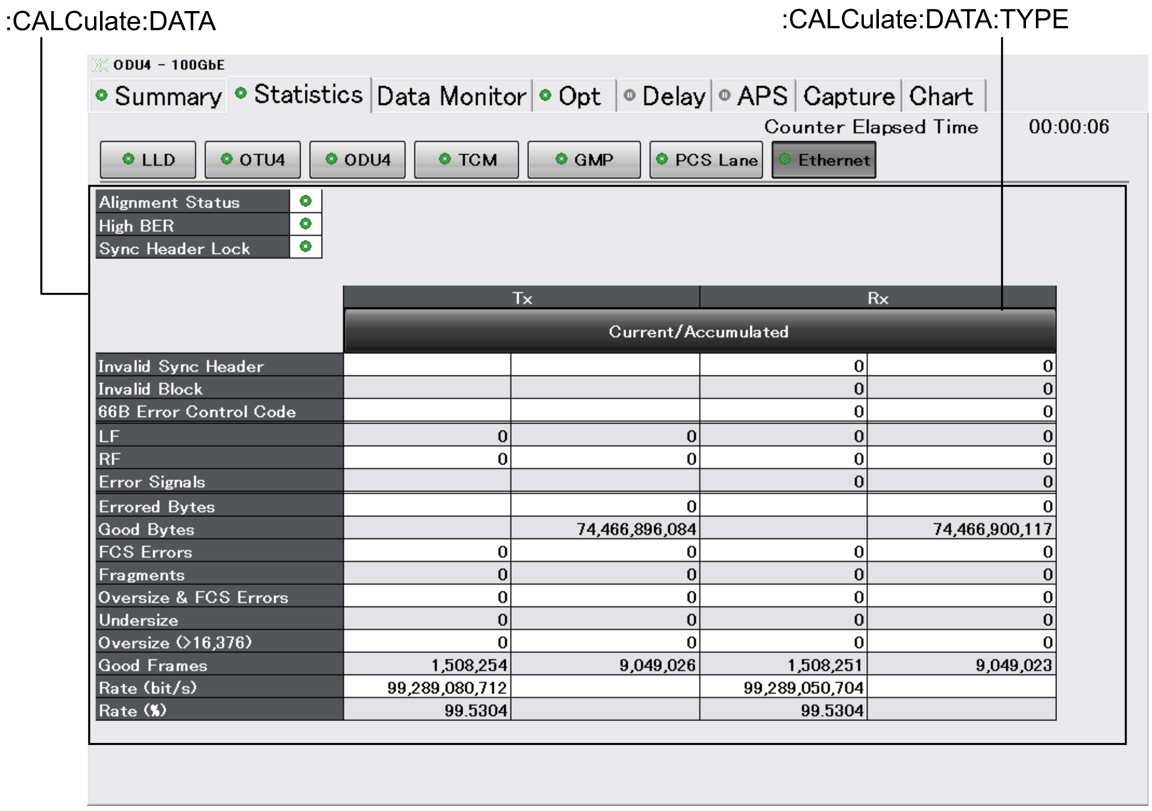
:SOURce:CLOCk:PAYLoad:OFFSet[:L]
:SOURce:CLOCk:FREQuency:OFFSet
:ROUTe:ETHer:NEGotiation:AUTO
:SOURce:CLOCk

Port/Clock

Port
Mode: Loopback
GFEC Encode: On
GFEC Decode: On
GbE Auto Negotiation: On

Clock
Frequency Offset: 0 ppm
Payload Offset High: 0 ppm  Cm 14,528.0000  CnD 0.0000
Payload Offset Low: 0 ppm  Cm 14,407.3110  CnD 0.3110
Clock Source: Internal
Tx Reference Clock Output: 1/64
10 MHz Output: Internal 10 MHz

**Figure 3.2.3-16 Messages Corresponding to Setting Area [Port/Clock] 2**

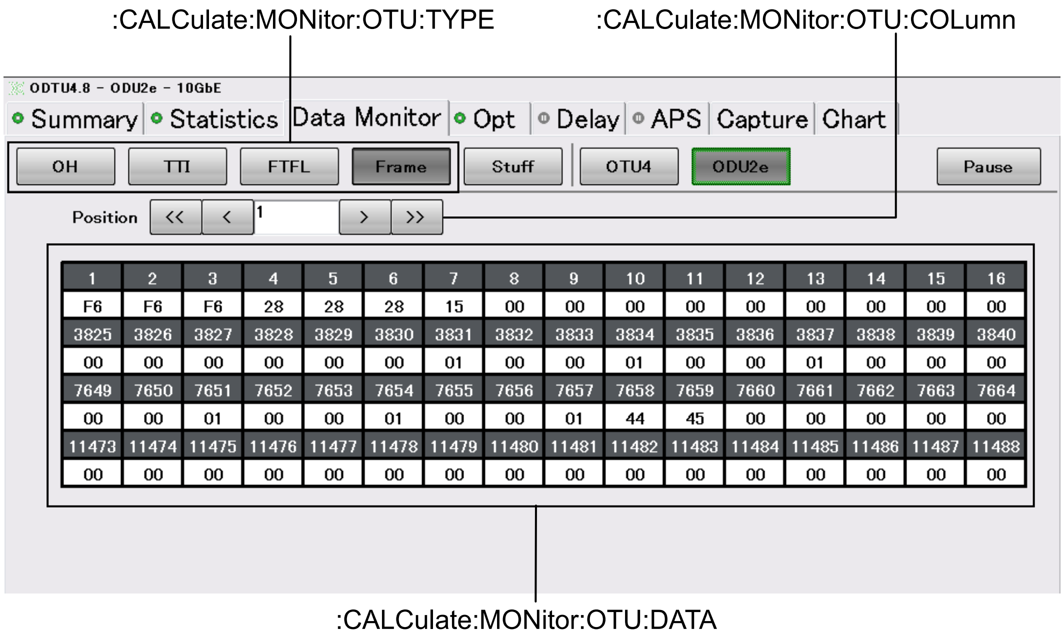


Counter Elapsed Time 00:00:06

| | |
|---|---|
| Alignment Status | ● |
| High BER | ● |
| Sync Header Lock | ● |

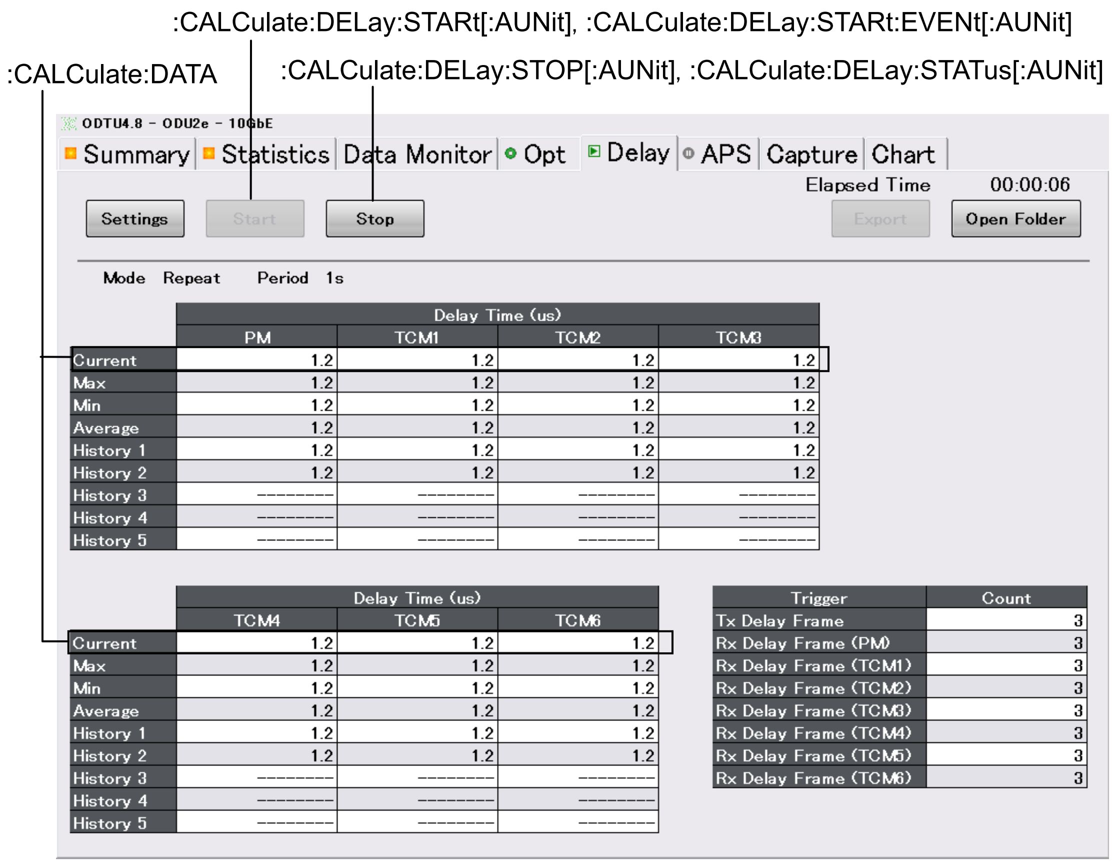
| | Tx | | Rx | |
|---|---|---|---|---|
| | Current/Accumulated | | | |
| Invalid Sync Header | | | 0 | 0 |
| Invalid Block | | | 0 | 0 |
| 66B Error Control Code | | | 0 | 0 |
| LF | 0 | 0 | 0 | 0 |
| RF | 0 | 0 | 0 | 0 |
| Error Signals | | | 0 | 0 |
| Errored Bytes | | 0 | | 0 |
| Good Bytes | | 74,466,896,084 | | 74,466,900,117 |
| FCS Errors | 0 | 0 | 0 | 0 |
| Fragments | 0 | 0 | 0 | 0 |
| Oversize & FCS Errors | 0 | 0 | 0 | 0 |
| Undersize | 0 | 0 | 0 | 0 |
| Oversize (>16,376) | 0 | 0 | 0 | 0 |
| Good Frames | 1,508,254 | 9,049,026 | 1,508,251 | 9,049,023 |
| Rate (bit/s) | 99,289,080,712 | | 99,289,050,704 | |
| Rate (%) | 99.5304 | | 99.5304 | |

**Figure 3.2.3-17 Messages Corresponding to Measurement Result Display Area (Statistics)**



| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F6 | F6 | F6 | 28 | 28 | 28 | 15 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 3825 | 3826 | 3827 | 3828 | 3829 | 3830 | 3831 | 3832 | 3833 | 3834 | 3835 | 3836 | 3837 | 3838 | 3839 | 3840 |
| 00 | 00 | 00 | 00 | 00 | 00 | 01 | 00 | 00 | 01 | 00 | 00 | 01 | 00 | 00 | 00 |
| 7649 | 7650 | 7651 | 7652 | 7653 | 7654 | 7655 | 7656 | 7657 | 7658 | 7659 | 7660 | 7661 | 7662 | 7663 | 7664 |
| 00 | 00 | 01 | 00 | 00 | 01 | 00 | 00 | 01 | 44 | 45 | 00 | 00 | 00 | 00 | 00 |
| 11473 | 11474 | 11475 | 11476 | 11477 | 11478 | 11479 | 11480 | 11481 | 11482 | 11483 | 11484 | 11485 | 11486 | 11487 | 11488 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

**Figure 3.2.3-18 Messages Corresponding to Measurement Result Display Area (Data Monitor)**


:CALCulate:DELay:STARt[:AUNit], :CALCulate:DELay:STARt:EVENt[:AUNit]
:CALCulate:DATA
:CALCulate:DELay:STOP[:AUNit], :CALCulate:DELay:STATus[:AUNit]
ODTU4.8 - ODU2e - 10GbE
Summary
Statistics
Data Monitor
Opt
Delay
APS
Capture
Chart
Settings
Start
Stop
Export
Open Folder

Elapsed Time 00:00:06

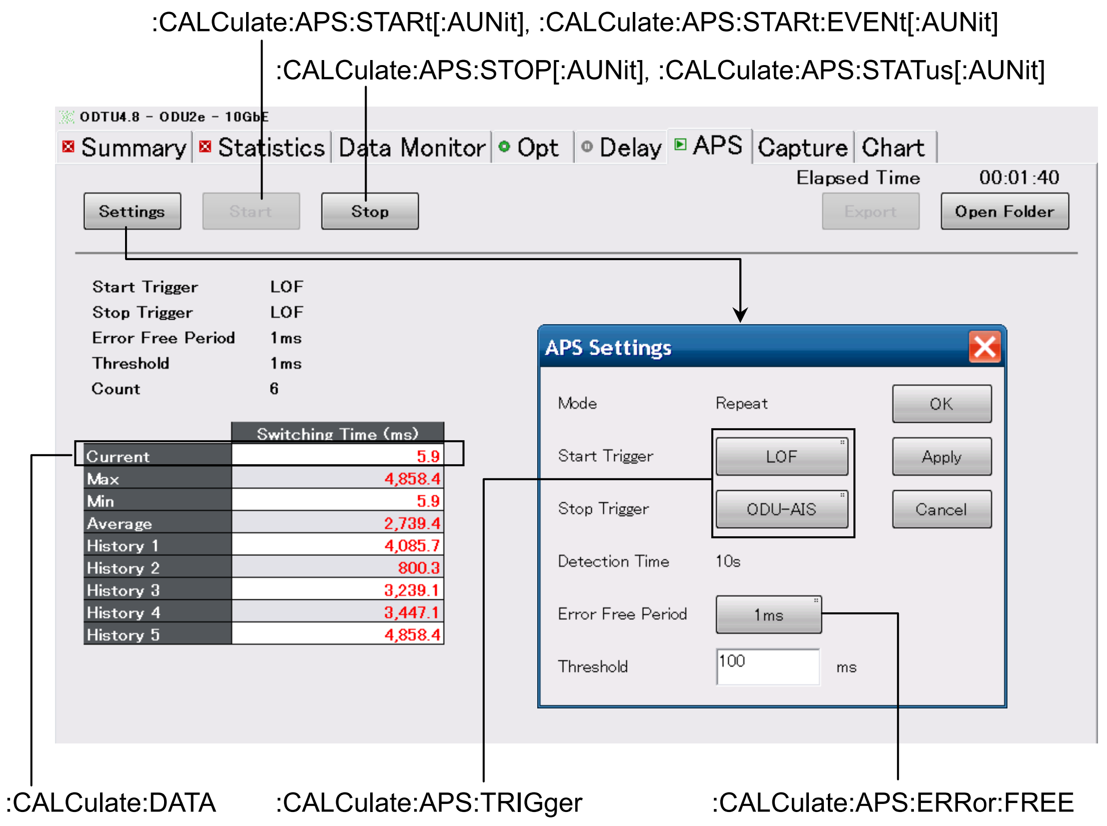
Mode Repeat Period 1s

| | Delay Time (us) | | | |
|---|---|---|---|---|
| | PM | TCM1 | TCM2 | TCM3 |
| Current | 1.2 | 1.2 | 1.2 | 1.2 |
| Max | 1.2 | 1.2 | 1.2 | 1.2 |
| Min | 1.2 | 1.2 | 1.2 | 1.2 |
| Average | 1.2 | 1.2 | 1.2 | 1.2 |
| History 1 | 1.2 | 1.2 | 1.2 | 1.2 |
| History 2 | 1.2 | 1.2 | 1.2 | 1.2 |
| History 3 | -------- | -------- | -------- | -------- |
| History 4 | -------- | -------- | -------- | -------- |
| History 5 | -------- | -------- | -------- | -------- |

| | Delay Time (us) | | |
|---|---|---|---|
| | TCM4 | TCM5 | TCM6 |
| Current | 1.2 | 1.2 | 1.2 |
| Max | 1.2 | 1.2 | 1.2 |
| Min | 1.2 | 1.2 | 1.2 |
| Average | 1.2 | 1.2 | 1.2 |
| History 1 | 1.2 | 1.2 | 1.2 |
| History 2 | 1.2 | 1.2 | 1.2 |
| History 3 | -------- | -------- | -------- |
| History 4 | -------- | -------- | -------- |
| History 5 | -------- | -------- | -------- |

| Trigger | Count |
|---|---|
| Tx Delay Frame | 3 |
| Rx Delay Frame (PM) | 3 |
| Rx Delay Frame (TCM1) | 3 |
| Rx Delay Frame (TCM2) | 3 |
| Rx Delay Frame (TCM3) | 3 |
| Rx Delay Frame (TCM4) | 3 |
| Rx Delay Frame (TCM5) | 3 |
| Rx Delay Frame (TCM6) | 3 |

**Figure 3.2.3-19 Messages Corresponding to Measurement Result Display Area (Delay)**



:CALCulate:APS:STARt[:AUNit], :CALCulate:APS:STARt:EVENt[:AUNit]

:CALCulate:APS:STOP[:AUNit], :CALCulate:APS:STATus[:AUNit]

Elapsed Time 00:01:40

|  |  |
|---|---|
| Start Trigger | LOF |
| Stop Trigger | LOF |
| Error Free Period | 1ms |
| Threshold | 1ms |
| Count | 6 |

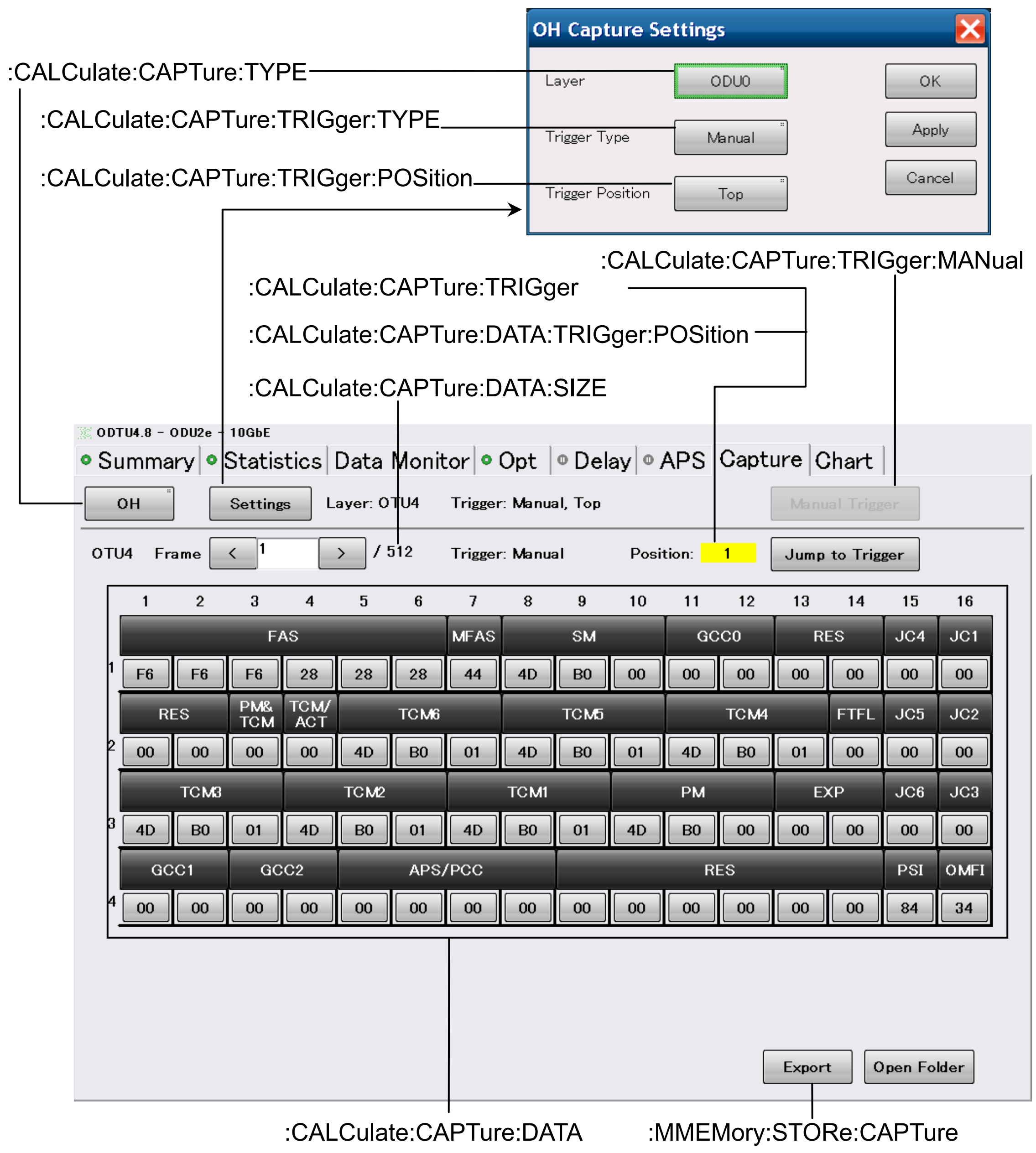
|  | Switching Time (ms) |
|---|---|
| Current | 5.9 |
| Max | 4,858.4 |
| Min | 5.9 |
| Average | 2,739.4 |
| History 1 | 4,085.7 |
| History 2 | 800.3 |
| History 3 | 3,239.1 |
| History 4 | 3,447.1 |
| History 5 | 4,858.4 |

APS Settings

|  |  |
|---|---|
| Mode | Repeat |
| Start Trigger | LOF |
| Stop Trigger | ODU-AIS |
| Detection Time | 10s |
| Error Free Period | 1ms |
| Threshold | 100 ms |

:CALCulate:DATA :CALCulate:APS:TRIGger :CALCulate:APS:ERRor:FREE

**Figure 3.2.3-20 Messages Corresponding to Measurement Result Display Area (APS)**


:CALCulate:CAPTure:TYPE

:CALCulate:CAPTure:TRIGger:TYPE

:CALCulate:CAPTure:TRIGger:POSition

:CALCulate:CAPTure:TRIGger:MANual

:CALCulate:CAPTure:TRIGger

:CALCulate:CAPTure:DATA:TRIGger:POSition

:CALCulate:CAPTure:DATA:SIZE

:CALCulate:CAPTure:DATA

:MMEMory:STORe:CAPTure

OH Capture Settings

Layer: ODU0

Trigger Type: Manual

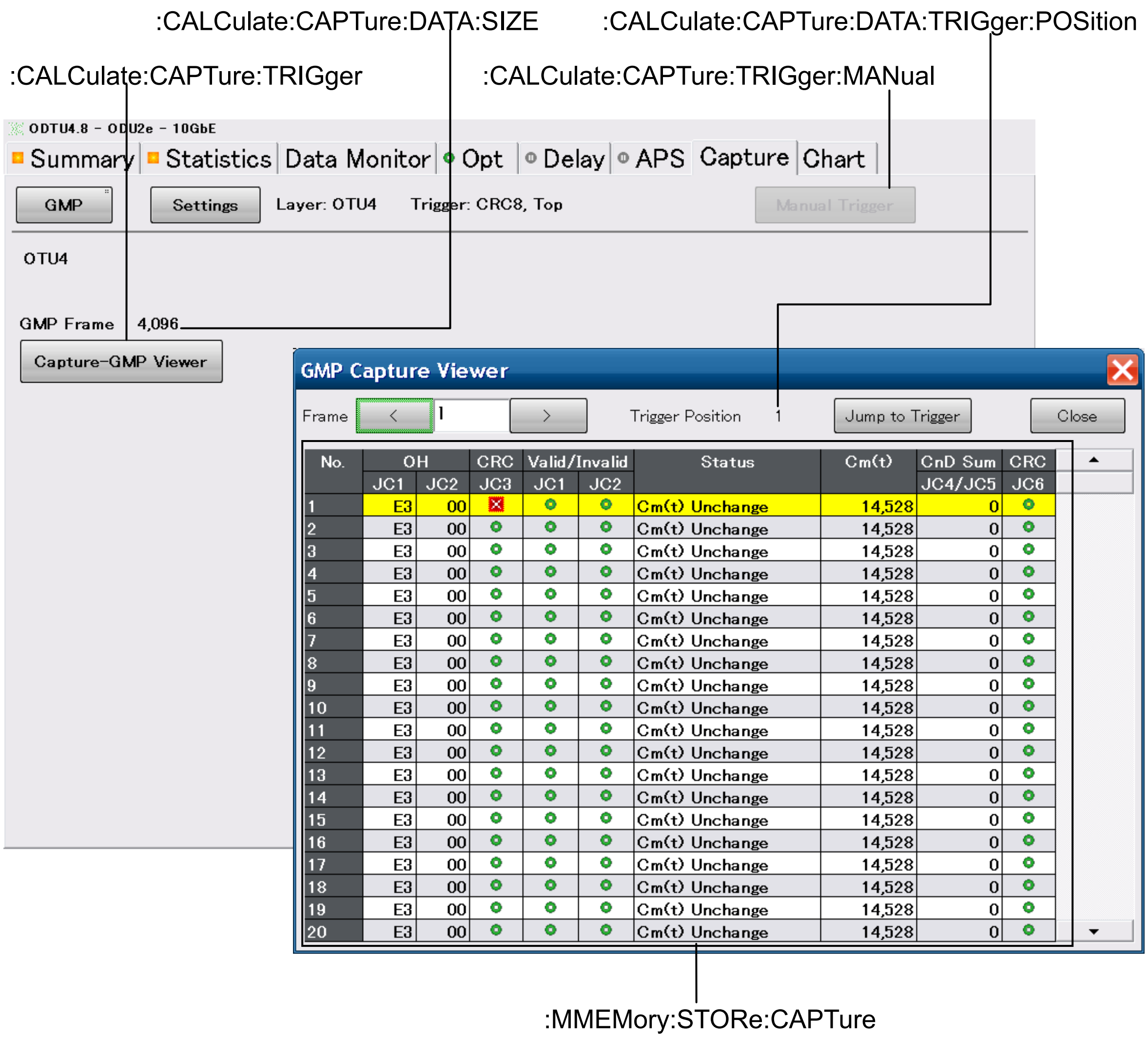
Trigger Position: Top

Layer: OTU4 Trigger: Manual, Top

OTU4 Frame 1 / 512 Trigger: Manual Position: 1

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FAS | | | | | | MFAS | SM | | | GCC0 | | RES | | JC4 | JC1 |
| 1 | F6 | F6 | F6 | 28 | 28 | 28 | 44 | 4D | B0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| | RES | | PM& TCM | TCM/ ACT | TCM6 | | | TCM5 | | | TCM4 | | | FTFL | JC5 | JC2 |
| 2 | 00 | 00 | 00 | 00 | 4D | B0 | 01 | 4D | B0 | 01 | 4D | B0 | 01 | 00 | 00 | 00 |
| | TCM3 | | | TCM2 | | | TCM1 | | | PM | | | EXP | | JC6 | JC3 |
| 3 | 4D | B0 | 01 | 4D | B0 | 01 | 4D | B0 | 01 | 4D | B0 | 00 | 00 | 00 | 00 | 00 |
| | GCC1 | | GCC2 | | APS/PCC | | | | RES | | | | | | PSI | OMFI |
| 4 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 84 | 34 |

**Figure 3.2.3-21 Messages Corresponding to Measurement Result Display Area (Capture) 1**



Layer: OTU4 Trigger: CRC8, Top

OTU4

GMP Frame 4,096

Frame 1 Trigger Position 1

| No. | OH | | CRC | Valid/Invalid | | Status | Cm(t) | CnD Sum | CRC |
|---|---|---|---|---|---|---|---|---|---|
| | JC1 | JC2 | JC3 | JC1 | JC2 | | | JC4/JC5 | JC6 |
| 1 | E3 | 00 | ✗ | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 2 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 3 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 4 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 5 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 6 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 7 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 8 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 9 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 10 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 11 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 12 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 13 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 14 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 15 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 16 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 17 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 18 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 19 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |
| 20 | E3 | 00 | ● | ● | ● | Cm(t) Unchange | 14,528 | 0 | ● |

**Figure 3.2.3-22 Messages Corresponding to Measurement Result Display Area (Capture) 2**

### 3.2.4 Messages Corresponding to No Frame

For the messages corresponding to the setting area of [MDIO], [Clock], and [Transceiver], refer to Figure 3.2.1-6 Messages Corresponding to Setting Area [MDIO], Figure 3.2.1-7 Messages Corresponding to Setting Area [Clock], and Figure 3.2.2-12 Messages Corresponding to Setting Area [Transceiver].

There are no messages corresponding to the setting area of [Counter].

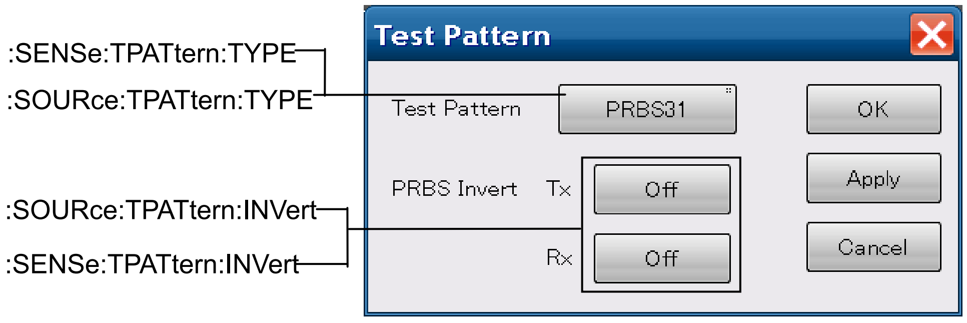


**Figure 3.2.4-1 Messages Corresponding to Setting Area [Test Pattern]**

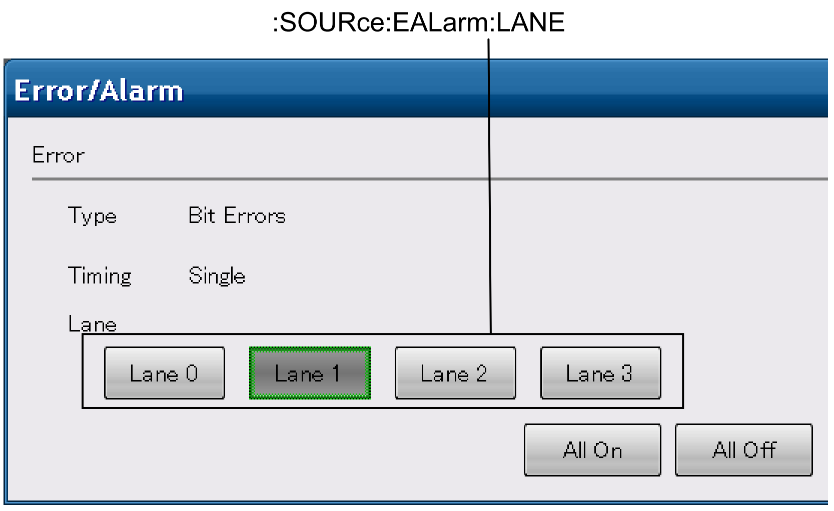
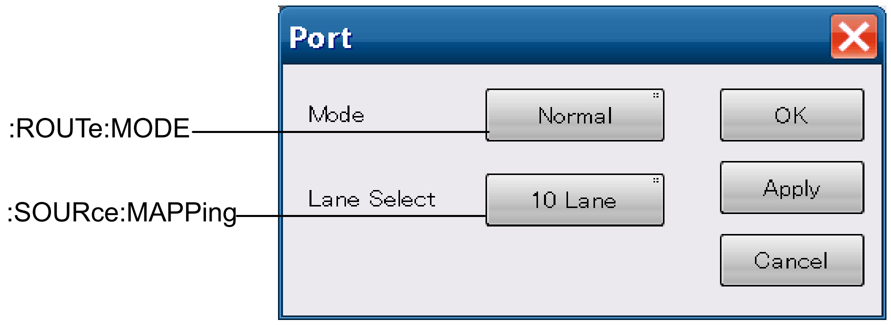


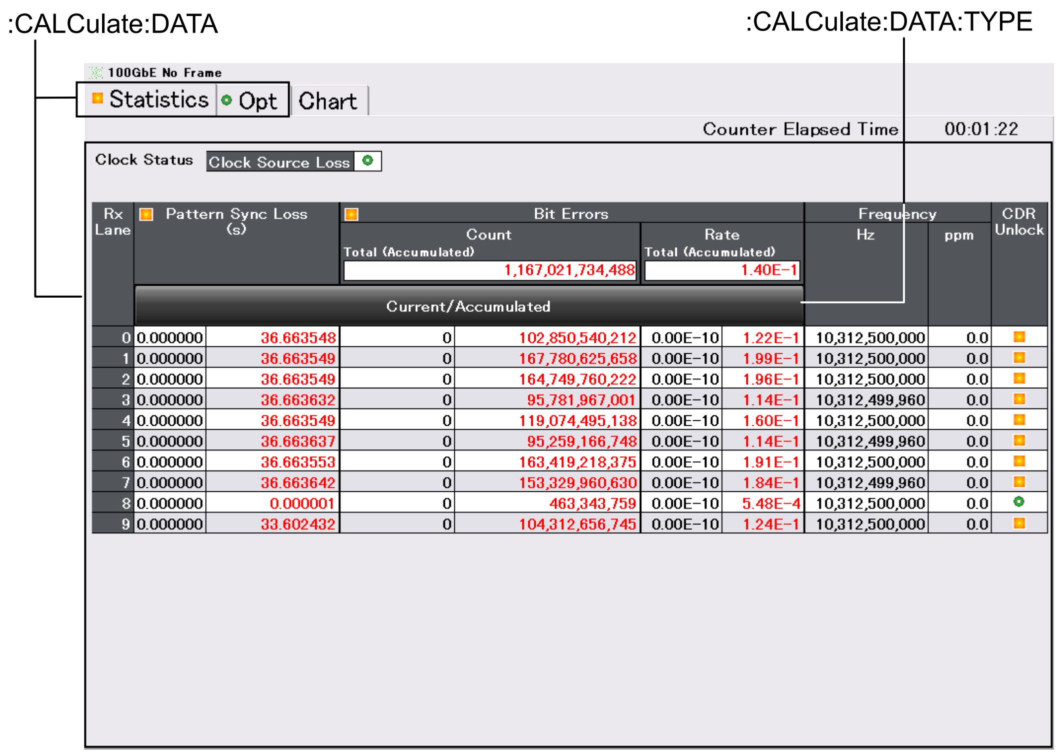
**Figure 3.2.4-2 Messages Corresponding to Setting Area [Error/Alarm]**



Mode: Normal

Lane Select: 10 Lane

**Figure 3.2.4-3   Messages Corresponding to Setting Area [Port] (100GbE No Frame/OTU4 No Frame)**



Counter Elapsed Time 00:01:22

Clock Status: Clock Source Loss ●

Bit Errors Count Total (Accumulated): 1,167,021,734,488

Bit Errors Rate Total (Accumulated): 1.40E−1

Current/Accumulated

| Rx Lane | Pattern Sync Loss (s) | | Bit Errors Count | | Bit Errors Rate | | Frequency Hz | Frequency ppm | CDR Unlock |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.000000 | 36.663548 | 0 | 102,850,540,212 | 0.00E−10 | 1.22E−1 | 10,312,500,000 | 0.0 | ■ |
| 1 | 0.000000 | 36.663549 | 0 | 167,780,625,658 | 0.00E−10 | 1.99E−1 | 10,312,500,000 | 0.0 | ■ |
| 2 | 0.000000 | 36.663549 | 0 | 164,749,760,222 | 0.00E−10 | 1.96E−1 | 10,312,500,000 | 0.0 | ■ |
| 3 | 0.000000 | 36.663632 | 0 | 95,781,967,001 | 0.00E−10 | 1.14E−1 | 10,312,499,960 | 0.0 | ■ |
| 4 | 0.000000 | 36.663549 | 0 | 119,074,495,138 | 0.00E−10 | 1.60E−1 | 10,312,500,000 | 0.0 | ■ |
| 5 | 0.000000 | 36.663637 | 0 | 95,259,166,748 | 0.00E−10 | 1.14E−1 | 10,312,499,960 | 0.0 | ■ |
| 6 | 0.000000 | 36.663553 | 0 | 163,419,218,375 | 0.00E−10 | 1.91E−1 | 10,312,500,000 | 0.0 | ■ |
| 7 | 0.000000 | 36.663642 | 0 | 153,329,960,630 | 0.00E−10 | 1.84E−1 | 10,312,499,960 | 0.0 | ■ |
| 8 | 0.000000 | 0.000001 | 0 | 463,343,759 | 0.00E−10 | 5.48E−4 | 10,312,500,000 | 0.0 | ● |
| 9 | 0.000000 | 33.602432 | 0 | 104,312,656,745 | 0.00E−10 | 1.24E−1 | 10,312,500,000 | 0.0 | ■ |

**Figure 3.2.4-4   Messages Corresponding to Measurement Result Display Area**

## 3.2.5 Add-on Function

There are no messages corresponding to the following Add-on functions.

- RFC2544
- CFP MDIO Analysis
- Lambda Measure
- Service Disruption

The remote control cannot be performed while executing the Add-on function.

## 3.2.6 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 3.2.6-1 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 MD1260A setting conditions |
| *SRE | Sets/queries service request enable register |
| *STB | Queries status byte register |
| *WAI | Waits for previous sent message completion |

**Table 3.2.6-2 Messages with No Corresponding Panel Operation**

| Command | Details |
|---|---|
| :CALCulate:APS:STARt:EVENt[:AUNit] | Queries APS measurement start event |
| :CALCulate:CAPTure:DATA:SIZE | Queries captured data size |
| :CALCulate:CAPTure:STARt:EVENt | Queries capture start event |
| :CALCulate:COUNter:STARt:EVENt[:AUNit] | Queries counter start event |
| :CALCulate:DELay:STARt:EVENt[:AUNit] | Queries delay time measurement start event |
| :MMEMory:LOG:FNAMe | Queries latest log file name |
| :PROTocol:PING:STARt:EVENt | Queries Ping measurement start event |
| :SOURce:EALarm:STARt:EVENt[:AUNit] | Queries error/alarm start event |
| :SOURce:STReam:STARt:EVENt[:AUNit] | Queries delay time measurement start event |
| :SYSTem:ERRor | Queries command error occurrence |
| :SYSTem:STATus | Queries application startup status |
| :SYSTem:TERMination | Queries setting of terminal |
| :SYSTem:VERSion | Queries SCPI version |

### 3.2.7 Panel operation not controlled remotely

Button operations and data readings without remote commands displayed on the screens in Section 3.2.1 "Messages Corresponding to Common Operations" to Section 3.2.5 "Messages corresponding to No Frame" cannot be controlled remotely.
In addition, the following panel operations cannot be operated remotely.

- Any operation of Utility tab in the Selector screen
- Go Slave button of Multi Port tab in the Selector screen
- Shut Down button in the Selector screen
- Switching of the application screen display with tab operation
- Setting screen display for the system menu and setting area
- Waveform data acquisition with Chart tab in the application screen
- Any operation for the Add-on function
- GMP screen operation and data acquisition with OTU4 application Data Monitor tab
- Acquisition of GMP Capture Viewer screen with OTU4 application Capture tab (GMP)
  However, the saving measurement result to files can be controlled remotely.

## 3.3 Device Message Details

### 3.3.1 Common Messages

#### *CLS [Clear Status]

**Function**

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

   Therefore, bit 5 of the status byte register becomes 0.
   The setting value of each enable register does not change 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.

**Syntax**

```
*CLS
```

## *ESE [Event Status Enable]

### Function

This command sets the standard event status enable register.
The standard event status mask bit is set to 0.
*ESE? queries the standard event status enable register value.

### Syntax

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

<integer> = bit0 + bit1 + bit2 + bit3 + bit4 + bit5 + bit6 + bit7
The correspondence between the register bit and hexadecimal value is shown below.

| | |
|---|---|
| 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 : $2^3 = 8$ | Device dependent 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>
*ESE?: Total bits for standard event status enable register are 0 to 255.

### Example of Use

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

```
*ESE 56
*ESE?
>56
```

## *ESR [Standard Event Status Register]

**Function**

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

**Syntax**

```
*ESE?
```

**Response Data**

<integer>:
*ESR?: Total bits for standard event status register is 0 to 255.

**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
```

Details of error occurrences can be investigated using the:SYSTem:ERRor? command.

## *IDN [Identification]

**Function**

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

**Syntax**

```
*IDN?
```

**Response Data**

```
ANRITSU,MD1260A,<serial_number>,<version>
```

<serial_number>
This command returns the instrument serial number (10 digits).

<version>
This command returns the version of the software installed in the instrument in the format X.XX.XX.

**Example of Use**

```
*IDN?
>ANRITSU,MD1260A,6200123456,1.00.16
```

## *OPC [Operation Complete]

### Function

If a *OPC command is received, the operation completion bit (bit 0) is set to 1 when all active processes are complete.

A*OPC? Queries bit 0 (OPC bit) of the standard event status register. O is returned while the processing is performed, and 1 is returned if the processing is completed.

The wait for operation completion set by *OPC/*OPC? is disabled after the following events:

- Power-on
- Reception of DCL or SCL on IEEE488.1 interface
- Reception of `*CLS` command
- Reception of `*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 the previously sent message. Since *OPC? is processed after processing of the previously sent message is completed, the response data is always 1. For how to confirm the end of the message execution, refer to Section 2.6 Confirming Message Execution Status.

### Syntax

```
*OPC
*OPC?
```

### Response Data

1

### Example of Use

```
*OPC?
>1
```

## *RST [Reset]

**Function**

This command resets the following items to the factory-default settings. However, the following items cannot be set back to the factory-default settings.

- Windows settings such as GPIB address, IP address, etc.
- Transceiver settings `(:SOURce:TRANsceiver:*)`

  :SOURce:TRANsceiver:EMPHasis:FIRSt

  :SOURce:TRANsceiver:EMPHasis:PRE

  :SOURce:TRANsceiver:EMPHasis:SECond

  :SOURce:TRANsceiver:VOD

  :SENSe:TRANsceiver:EQUalizer:CONTrol

  :SENSe:TRANsceiver:EQUalizer:DCGain
- Response terminator settings (:SYSTem:TERMination)

***Note:***

When multiple MD1260A are connected, all MD1260A units are initialized by sending *RST.

To initialize individual MD1260A units, send :MMEMory:INITialize.

**Syntax**

```
*RST
```

## *SRE [Service Request Enable]

**Function**

*SRE sets the service request enable register.

The 0 to 255 setting values are equivalent to 8-bit binary.

The bit masking the status byte register is set to 0.

**Syntax**

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

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

| | |
|---|---|
| bit7 : $2^7 = 128$ | Not used |
| bit6 : $2^6 = 64$ | Always 0 for Service Request Enable Register |
| bit5 : $2^5 = 32$ | Event status register summary |
| bit4 : $2^4 = 16$ | Indicates output queue is empty |
| bit3 : $2^3 = 8$ | Not used |
| bit2 : $2^2 = 4$ | Indicates error and event queue is empty |
| bit1 : $2^1 = 2$ | Not used |
| bit0 : $2^0 = 1$ | Not used |

Range 0 to 255

## *STB [Status Byte]

**Function**

*SRE? queries the service request enable register value.

*STB? reads the status byte register.

**Syntax**

```
*STB?
```

**Response Data**

<integer>

*STB?: Total bits for status byte register is 0 to 255

**Example of Use**

To mask bits 2 and 5:

```
*SRE 36
*SRE?
>36
```

## *TRG [Trigger]

**Function**

This is the same operation as sending:CALCulate:COUNter:STARt[:AUNit].

This command starts counting all modules when connecting multiple units.

**Syntax**

```
*TRG
```

## *WAI [Wait to Continue]

**Function**

This command holds execution of the next message until processing of the preceding message 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 complete, using*WAI is not required.

**Syntax**

```
*WAI
```

## 3.3.2 Device Dependent Commands

### :CALCulate:APS:ERRor:FREE

**Function**

This command sets and queries the APS measurement error free evaluation condition for the OTU3 or OTU4 application.

**Syntax**

```
:CALCulate:APS:ERRor:FREE <NR1>
:CALCulate:APS:ERRor:FREE?
```

| | |
|---|---|
| <NR1>: | Error Free Period (ms) |
| | 1 \| 10 \| 100 \| 200 \| 300 \| 400 \| 500 \| 600 \| 700 \| 800 \| 900 \| 1000 |

**Response Data**

<NR1>

**Example of Use**

```
:CALCulate:APS:ERRor:FREE 200
:CALCulate:APS:ERRor:FREE?
> 200
```

### :CALCulate:APS:STARt[:AUNit]

**Function**

This command starts the APS measurement for the OTU3 or OTU4 application.
If APS is already being measured, the measurement is restarted.
If :AUNit is added, APS measurements for all units connected are started simultaneously.

**Syntax**

```
:CALCulate:APS:STARt[:AUNit]
```

**Example of Use**

To wait for the error/alarm insertion start:

1. Send :CALCulate:APS:STARt:EVENt?.
   The start event is cleared.
2. Send :CALCulate:APS:STARt.
   The APS measurement is started.
3. Send :CALCulate:APS:STARt:EVENt?.
   Repeat the transmission until the response data becomes 1 (start).

## :CALCulate:APS:STARt:EVENt[:AUNit]

**Function**

This command queries the APS measurement start event for the OTU3 or OTU4 application.

**Syntax**

```
:CALCulate:APS:STARt:EVENt[:AUNit]?
```

**Response Data**

0|1

0: Before the start of APS measurement
After this command is read out, the response data becomes 0.

1: The APS measurement has been started.
If :AUNit is added to the header, the response data becomes 1 when APS measurements for all units connected are started.

The event value is cleared to 0 when the response data 1 is read.

**Example of Use**

Refer to the example for :SOURce:EALarm:STARt[:AUNit].

## :CALCulate:APS:STATus[:AUNit]

**Function**

This command queries the status of APS measurement for the OTU3 or OTU4 application.

**Syntax**

```
:CALCulate:APS:STATus[:AUNit]?
```

**Response Data**

0|1

0: APS measurement stopped

1: APS measurement in progress
If :AUNit is added to the header, the response data becomes 1 when APS measurements for any one of or multiple connected units are started, and it becomes 0 when APS measurements for all connected units are stopped.

**Example of Use**

Refer to the example for :SOURce:EALarm:STOP[:AUNit].

## :CALCulate:APS:STOP[:AUNit]

**Function**

This command stops the APS measurement for the OTU3 or OTU4 application.
Adding :AUNit stops the APS measurements for all units connected.

**Syntax**

```
:CALCulate:APS:STOP[:AUNit]
```

**Example of Use**

To confirm that the APS measurement is stopped as follows:
Send :CALCulate:APS:STOP to stop the APS measurement.

Send :CALCulate:COUNter:STATus? repeatedly until the response data is 0 (stop).

## :CALCulate:APS:TRIGger

**Function**

This command queries and sets the trigger of the APS measurement.

**Syntax**

```
:CALCulate:APS:TRIGger <start>,<stop>
```

<start>,<stop>=<CHAR>

| Trigger | <CHAR> |
|---|---|
| LOF | LOF |
| OOF | OOF |
| ODU-AIS | AIS_ODU |
| ODU-OCI | OCI |
| ODU-LCK | LCK |
| SM-BIP8 | SM_BIP8 |
| PM-BIP8 | PM_BIP8 |
| Any Error (OR for all items) | ANY_ERROR |

**Example of Use**

To confirm that the APS measurement is stopped as follows:

1. Send :CALCulate:APS:STOP to stop the APS measurement.
2. Send :CALCulate:COUNter:STATus?: repeatedly until the response data is 0 (stop).

## :CALCulate:CAPTure:DATA

**Function**

This command queries the capture data of the OTU3 or OTU4 application.
If the capture target is GMP, the data cannot be queried with this command.
Use :MMEMory:STORe:CAPTure.

**Syntax**

```
:CALCulate:CAPTure:DATA? {<row>,<column>}|
{<frame_num>[,<offset>,<length>]}
```

<row>,<column>,<frame_num>,<offset>,<length> = <NR1>

When the capture target is OH,

| | |
|---|---|
| <row>: | Row number |
| <column>: | Column number |

When the capture target is Frame,

| | |
|---|---|
| <frame_num>: | Frame number |
| <offset>: | Offset (bytes) |
| <length>: | Length (bytes) |

**Response Data**

<STRING>

Comma separated hexadecimal character string. Byte with invalid data is displayed as --.
The maximum buffer size for response data captured at one time is 65536 bytes.
When the capture target is OH, it is the multiframe sequence data of the specified byte position. The data length is the value specified with :CALCulate:CAPTure:DATA:SIZE.

## :CALCulate:CAPTure:DATA:RXC

**Function**

This command queries the capture data of the control signal (RXC) for the 40 GbE or 100 GbE application.
It also specifies the data position querying the offset and length as the number of bits.

**Syntax**

```
:CALCulate:CAPTure:DATA:RXC? <offset>,<length>
```

<offset>,<length> = <NR1>

| | |
|---|---|
| <offset>: | Offset (bytes) |
| <length>: | Length (bytes) |

**Response Data**

<BINARY>

In the following cases, –220,"Parameter Error" is returned for the :SYSTem:ERRor command.

- <length> that exceeds the buffer size* has been specified.
- The range that exceeds :CALCulate:CAPTure:DATA:SIZE has been specified with <offset> and <length>.

*: The buffer size (maximum number of characters that can be returned as response data) is 65536 bytes. Therefore, the maximum number of RXC data that can be specified with <length> is 65536-2 = 65534 bits.

For details about RXC, refer to IEEE802.3ba Clause 81.

## :CALCulate:CAPTure:DATA:RXD

**Function**

This command queries the transmission capture data (RXD) for the 40 GbE or 100 GbE application. It also specifies the data position querying the offset and length as the number of bits.

**Syntax**

```
:CALCulate:CAPTure:DATA:RXD? <offset>,<length>
```

<offset>,<length> = <NR1>

| | |
|---|---|
| <offset>: | Offset (bytes) |
| <length>: | Length (bytes) |

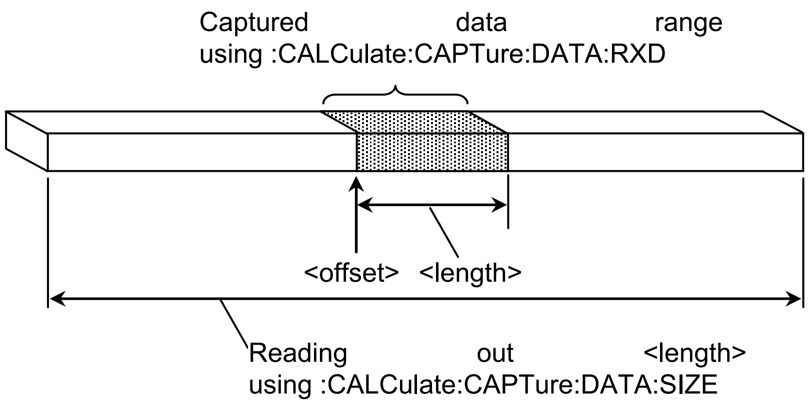
**Response Data**

<HEX>

In the following cases, –220,"Parameter Error" is returned for the :SYSTem:ERRor command.

- <length> that exceeds the buffer size* has been specified.
- The range that exceeds :CALCulate:CAPTure:DATA:SIZE has been specified with <offset> and <length>.

*: The buffer size (maximum number of characters that can be returned as response data) is 65536 bytes. Therefore, the maximum number of RXD data that can be specified with <length> is (65536-2)/2 = 32767 bits.


Captured data range using :CALCulate:CAPTure:DATA:RXD
<offset>
<length>
Reading out <length> using :CALCulate:CAPTure:DATA:SIZE

For details about RXD, refer to IEEE802.3ba Clause 81.

**Example of Use**

To read bytes1000 to 1015:

```
:CALCulate:CAPTure:DATA:RXD? 1000,16
> #H002523FE0A0B5523000000000000111A
```

To set an offset exceeding the capture data:

```
:CALCulate:CAPTure:DATA:RXD? 100000,16
> OUT_OF_RANGE
```

## :CALCulate:CAPTure:DATA:SIZE

**Function**

This command queries the transmission data capture size (RXD) using the number of RXD bytes for the 40 GbE or 100 GbE application.
It queries the capture data size using the number of frames for the OTU3 or OTU4 application.

**Syntax**

```
:CALCulate:CAPTure:DATA:SIZE?
```

**Response Data**

| | |
|---|---|
| <NR1>: | Capture data size (0~131072) |
| | Displays 0 while capturing |

## :CALCulate:CAPTure:DATA:TRIGger:POSition

**Function**

This command queries the trigger position in the capture data using the offset (bytes) from the header of the capture data.
When the trigger condition is the frame sequence, the trigger position is the last byte of the frame sequence.

**Syntax**

```
:CALCulate:CAPTure:DATA:TRIGger:POSition?
```

**Response Data**

| | |
|---|---|
| <NR1>: | Trigger position |

## :CALCulate:CAPTure:STARt

**Function**

This command starts capture.

**Syntax**

```
:CALCulate:CAPTure:STARt
```

***Note:***

Capture cannot be started simultaneously at multiple MD1260A units.

**Example of Use**

To start capture as follows:

1. Send :CALCulate:CAPTure:STARt:EVENt? to clear the start event.
2. Send :CALCulate:CAPTure:STAR to start capture.
3. Send repeatedly until :CALCulate:CAPTure:STARt:EVENt? response is 1 (start).

## :CALCulate:CAPTure:STARt:EVENt

**Function**

This command queries whether to start capture or not.

**Syntax**

```
:CALCulate:CAPTure:STARt:EVENt?
```

**Response Data**

0|1

| | |
|---|---|
| 1: | Becomes 1 when capture starts and cleared to 0 at read |
| 0: | Before capture starts |

***Note:***

Capture starting conditions cannot be queried at multiple MD1260A units.

**Example of Use**

Refer to the Example of Use for :CALCulate:CAPTure:STARt.

## :CALCulate:CAPTure:STATus

### Function

This command queries whether to read capture data.

### Syntax

```
:CALCulate:CAPTure:STATus?
```

### Response Data

0|1

| | |
|---|---|
| 0: | Can read capture data |
| 1: | Capturing |

Capturing is the state when data processing after starting capture is not complete such as when waiting for the trigger or capturing data after capture stops.
When reading captured data at this time, there may be an attempt to read previously captured data and the expected data may not be captured.

***Note:***

It is not possible to query the existence of capture data for multiple MD1260A units.

### Example of Use

Refer to the example for :CALCulate:CAPTure:STOP.

## :CALCulate:CAPTure:STOP

**Function**

This command is sent to forcibly stop capture when in the trigger wait state.

Capture is stopped automatically after starting after the trigger occurs. Consequently, it is not necessary to send this command after capture starts.

**Syntax**

```
:CALCulate:CAPTure:STOP
```

***Note:***

Capture at multiple MD1260A units cannot be stopped simultaneously.

**Example of Use**

To stop capture:

1. Stop capture by sending :CALCulate:CAPTure:STOP.
2. Send repeatedly until the :CALCulate:CAPTure:STATus? response becomes 0 (stopped).
3. Send repeatedly until the :CALCulate:CAPTure:DATA:SIZE? response becomes 1 or more.

## :CALCulate:CAPTure:TRIGger

**Function**

This command queries whether a trigger has occurred after capture start.

**Syntax**

```
:CALCulate:CAPTure:TRIGger?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Trigger not generated |
| 1: | Trigger generated |

Starting capture clears the setting to 0.

***Note:***

Trigger occurrence query for multiple MD1260A is unavailable.

## :CALCulate:CAPTure:TRIGger:MANual

**Function**

This command generates the manual trigger when the OTU3 or OTU4 applications capture trigger is Manual.

**Syntax**

```
:CALCulate:CAPTure:TRIGger:MANual
```

## :CALCulate:CAPTure:TRIGger:POSition

**Function**

This command sets and queries the capture trigger position of the OTU3 or OTU4 application.

**Syntax**

```
:CALCulate:CAPTure:TRIGger:POSition TOP|MIDDLE
:CALCulate:CAPTure:TRIGger:POSition?
```

| | |
|---|---|
| TOP: | The top of the memory is the trigger position |
| MIDDLE: | The middle of the memory is the trigger position |

**Response Data**

TOP|MIDDLE

## :CALCulate:CAPTure:TRIGger:TYPE

### Function

This command sets and queries the capture trigger type of the OTU3 or OTU4 application.

### Syntax

```
:CALCulate:CAPTure:TRIGger:TYPE <CHAR>
:CALCulate:CAPTure:TRIGger:TYPE?
```

| Trigger | <CHAR> | Capture target | | | | |
|---|---|---|---|---|---|---|
| | | FRAME | GMP | GMP (Low order) | OH | OH (Low order) |
| Manual | MANUAL | ✓ | ✓ | ✓ | ✓ | ✓ |
| MFAS = 0 | MFAS0 | – | – | – | ✓ | ✓ |
| MFAS | MFAS<NR1> *1 | ✓ | – | – | – | – |
| FAS | FAS | – | – | – | ✓ | ✓ |
| OMFI | OMFI<NR1> *2 | ✓ | – | – | – | – |
| SM-BIP8 | SM_BIP8 | ✓ | – | – | ✓ | – |
| PM-BIP8 | PM_BIP8 | ✓ | – | – | ✓ | ✓ |
| OOF | OOF | – | – | – | ✓ | ✓ |
| OOM | OOM | ✓ | – | – | ✓ | ✓ |
| ODU-AIS | AIS_ODU | ✓ | – | – | ✓ | ✓ |
| ODU-OCI | OCI | ✓ | – | – | ✓ | ✓ |
| ODU-LCK | LCK | ✓ | – | – | ✓ | ✓ |
| MSIM | MSIM *3 | ✓ | – | – | ✓ | – |
| Lock->Unlock | LOCK_UNLOCK | – | ✓ | ✓ | – | – |
| Unlock->Lock | UNLOCK_LOCK | – | ✓ | ✓ | – | – |
| CRC8 Error | GMP_CRC8 | – | ✓ | ✓ | – | – |
| CRC5 Error | GMP_CRC5 | – | ✓ | – | – | – |

*1: <NR1>=0~255

*2: For OTU3, <NR1> = 0 to 31. For OTU4, <NR1> = 0 to 79.

*3: Can be set with the mapping below.

ODTU4.8-ODU2e-PRBS, ODTU4.8-ODU2e-10GbE, ODTU4.1-ODU0-PRBS, ODTU4.1-ODU0-GbE

### Response Data

<CHAR>

## :CALCulate:CAPTure:TYPE

### Function

This command sets and queries the capture target of the OTU3 or OTU4 application.

### Syntax

```
:CALCulate:CAPTure:TYPE FRAME|GMP|GMP_L|OH|OH_L
:CALCulate:CAPTure:TYPE?
```

| | |
|---|---|
| FRAME: | Frame |
| GMP: | (High order) Cm(t), decode |
| GMP_L: | Low order Cm(t), decode |
| OH: | (High order) overhead |
| OH_L: | Low order overhead |

### Response Data

FRAME|GMP|GMP_L|OH|OH_L

### Example of Use

Set the capture condition as follows:

| | |
|---|---|
| Capture target | FRAME |
| Trigger position | TOP |
| Trigger type | MFAS=16 |

```
:CALCulate:CAPTure:TYPE FRAME
:CALCulate:CAPTure:POSition TOP
:CALCulate:CAPTure:TYPE MFAS16
```

## :CALCulate:COUNter:ETHer:OVERsize

**Function**

This command sets the counter oversize.

**Syntax**

```
:CALCulate:COUNter:ETHer:OVERsize <NR1>
:CALCulate:COUNter:ETHer:OVERsize?
```

<NR1>
This sets the Oversize evaluation threshold value in byte units.

**Response Data**

<NR1>

**Example of Use**

```
:CALCulate:COUNter:ETHer:OVERsize 1522
:CALCulate:COUNter:ETHer:OVERsize?
> 1522
```

## :CALCulate:COUNter:FLOW:FIELd

**Function**

This command sets the field and bit range when the Test Frame counter filter method is User Defined or User Defined and Flow ID.

**Syntax**

```
:CALCulate:COUNter:FLOW:FIELd
<format>,<field>,<offset>,<length>
:CALCulate:COUNter:FLOW:FIELd?
```

<format>=<STRING>
<field>=<CHAR>
<offset>,<length>=<NR1>

For details on the <format> character string, refer to :SOURce:STReam:HEADer.
Specify the string in the following table to <field>.Fields not in this table cannot be set.

Table 3.3.2-3 String to Specify Field

| <field> | Field Name |
|---|---|
| MPLS_TP_DA | MPLS-TP Destination Address |
| MPLS_TP_SA | MPLS-TP Source Address |
| MPLS_TP_LABEL{1\|2\|3\|4\|5} | MPLS-TP Label |
| MPLS_TP_EXP{1\|2\|3\|4\|5} | MPLS-TP Experimental Use |
| MPLS_TP_BTM{1\|2\|3\|4\|5} | MPLS-TP Bottom of Stack |
| MPLS_TP_TTL{1\|2\|3\|4\|5} | MPLS-TP Time to Live |
| CW_NIB | PWMCW First nibble |
| CW_FLAG | PWMCW Flag |
| CW_FRG | PWMCW FRG |
| CW_LENGTH | PWMCW Length |
| CW_SEQ | PWMCW Sequence Number |
| PBB_DA | PBB Destination Address |
| PBB_SA | PBB Source Address |
| PBB_{B\|I}_TPID | PBB TPID |
| PBB_{B\|I}_PCP | PBB PCP |
| PBB_{B\|I}_DEI | PBB DEI |
| PBB_B_VID | PBB VID |
| PBB_I_SID | PBB SID |
| PBB_I_RSV | PBB reserved |
| ETH_DA | Ethernet Destination Address |
| ETH_SA | Ethernet Source Address |
| ETH_TYPE | Ethernet Type |
| VLAN_{1\|2}_TPID | Ethernet VLAN TPID |
| VLAN_{1\|2}_PCP | Ethernet VLAN PCP |
| VLAN_{1\|2}_CFI | Ethernet VLAN CFI |
| VLAN_{1\|2}_VID | Ethernet VLAN DEI |
| MPLS_LABEL{1\|2\|3} | MPLS Label |
| MPLS_EXP{1\|2\|3} | MPLS Experimental Use |
| MPLS_TTL{1\|2\|3} | MPLS Time to Live |

Table 3.3.2-3 String to Specify Field (Continued)

| <field> | Field Name |
|---|---|
| IPV4_VER | IPv4 Version |
| IPV4_HLEN | IPv4 Header Length |
| IPV4_TOS | IPv4 ToS |
| IPV4_PLEN | IPv4 Packet Length |
| IPV4_ID | IPv4 Identification |
| IPV4_FLAG | IPv4 Flags |
| IPV4_OFFSET | IPv4 Fragment Offset |
| IPV4_TTL | IPv4 TTL |
| IPV4_PROT | IPv4 Protocol |
| IPV4_CHKSUM | IPv4 Header Checksum |
| IPV4_SA | IPv4 Source Address |
| IPV4_DA | IPv4 Destination Address |
| IPV6_VER | IPv6 Version |
| IPV6_TCLASS | IPv6 Traffic Class |
| IPV6_FLABEL | IPv6 Flow Label |
| IPV6_PLEN | IPv6 Packet Length |
| IPV6_NEXT | IPv6 Next Header |
| IPV6_HOP | IPv6 Hop Limit |
| IPV6_SA | IPv6 Source Address |
| IPV6_DA | IPv6 Destination Address |
| ARP_SMAC | ARP Source MAC Address |
| ARP_SIP | ARP Source IP Address |
| ARP_TMAC | ARP Target MAC Address |
| ARP_TIP | ARP Target IP Address |
| ARP_OPE | ARP Operation |
| ICMPV4_CODE | ICMPv4 Code |
| ICMPV4_EC_ID | ICMPv4 Identifier |
| ICMPV4_EC_SEQ | ICMPv4 Sequence No. |
| ICMPV6_CODE | ICMPv6 Code |
| ICMPV6_EC_ID | ICMPv6 Identifier |
| ICMPV6_EC_SEQ | ICMPv6 Sequence No. |
| ICMPV6_NSNA_TADDR | ICMPv6 Target Address |
| ICMPV6_NSNA_SADDR | ICMPv6 Source Link Layer Address |

**Response Data**

<format>,<field>,<offset>,<length>

**Example of Use**

```
:CALCulate:COUNter:FLOW:FIELd
USER_DEFINED,”ETHENET”,SOURCE_ADDRESS,8,8
:CALCulate:COUNter:FLOW:FIELd?
> USER_DEFINED,”ETHENET”,SOURCE_ADDRESS,8,8
```

## :CALCulate:COUNter:FLOW:FIELd:ID

**Function**

This command sets the filter value of the Test Frame counter.
The number of parameters depends on the filter method and test frame count.

**Syntax**

```
:CALCulate:COUNter:FLOW:FIELd:ID
MONITOR,<filter>,<num>,<id>,…,<id>
:CALCulate:COUNter:FLOW:FIELd:ID?
```

<filter>=TEST_FRAME | TF_AND_UD | USER_DEFINED

| | |
|---|---|
| TEST_FRAME: | Test Frame Flow ID |
| TF_AND_UD: | User Defined and Flow ID |
| USER_DEFINED: | User Defined |

<num>=<NR1>
When <filter>=TEST_FRAME: <id>=<flow_id>
When <filter>=USER_DEFINED: <id>=<user_id>
When <filter>=TF_AND_UD: <id>=<flow_id>,<user_id>
<flow_id>,<user_id>=<NR1> | <BINARY> | <HEX>
<flow_id> and <user_id> can be set also in a hexadecimal format and binary format.

**Response Data**

MONITOR,<filter>,<num><id>,…,<id>
<id> is in a hexadecimal format.

**Example of Use**

```
:CALCulate:COUNter:FLOW:FIELd:ID MONITOR,USER_DEFIND,
8,#H10,#H20,#H30,#H40,#H50,#H60,#H70,#H80
:CALCulate:COUNter:FLOW:FIELd:ID?
> MONITOR,USER_DEFIND,8,#H10,#H20,#H30,#H40,#H50,#H60,
#H70,#H80
```

## :CALCulate:COUNter:FLOW:FIELd:NFID

**Function**

This command queries the test frame count of the Test Frame counter.

**Syntax**

```
:CALCulate:COUNter:FLOW:FIELd:NFID?
```

**Response Data**

<NR1>

**Example of Use**

```
:CALCulate:COUNter:FLOW:FIELd:NFID 15
:CALCulate:COUNter:FLOW:FIELd:NFID?
> 15
```

## :CALCulate:COUNter:FLOW:TYPE

**Function**

This command sets the filter method of the Test Frame counter.

**Syntax**

```
:CALCulate:COUNter:FLOW:TYPE <filter>
:CALCulate:COUNter:FLOW:TYPE?
```

<filter>=TEST_FRAME | TF_AND_UD | USER_DEFINED

| | |
|---|---|
| TEST_FRAME: | Test Frame Flow ID |
| TF_AND_UD: | User Defined and Flow ID |
| USER_DEFINED: | User Defined |

**Response Data**

TEST_FRAME | TF_AND_UD | USER_DEFINED

**Example of Use**

```
:CALCulate:COUNter:FLOW:TYPE TEST_FRAME
:CALCulate:COUNter:FLOW:TYPE?
> TEST_FRAME
```

## :CALCulate:COUNter:GAP

**Function**

This command sets the measurement range of the Gap Size counter.

**Syntax**

```
:CALCulate:COUNter:GAP <min>,<step>
```

<min>,<step>=<NR1>

## :CALCulate:COUNter:SERRor

**Function**

This command sets whether to stop the test frame counter at a sequence error.

**Syntax**

```
:CALCulate:COUNter:SERRor 0|1|ON|OFF
:CALCulate:COUNter:SERRor?
```

| | |
|---|---|
| ON\|1: | Stops test frame counter at sequence error |
| OFF\|0: | Does not stop test frame counter at sequence error |

**Response Data**

0|1

## :CALCulate:COUNter:STARt[:AUNit]

**Function**

This command starts counting.
If the counter is already running, it is restarted.
Adding :AUNit starts the counters of all units connected at the same time.

**Syntax**

```
:CALCulate:COUNter:STARt[:AUNit]
```

**Example of Use**

To start counting:

1. Clear the start event by sending :CALCulate:COUNter:STARt:EVENt?.
2. Start counter by sending :CALCulate:COUNter:STARt.
3. Send repeatedly until the :CALCulate:COUNter:STARt:EVENt? response becomes 1 (started).

## :CALCulate:COUNter:STARt:EVENt[:AUNit]

**Function**

This command queries the counter start event.

**Syntax**

```
:CALCulate:COUNter:STARt:EVENt[:AUNit]?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | 0 before counter started |
| 1: | Cleared to 0 after reading with this command<br>Adding :AUNit to the header sets the counter of all connected units to 1 when started. |

**Example of Use**

Refer to the example for:CALCulate:COUNter:STARt[:AUNit].

## :CALCulate:COUNter:STATus[:AUNit]

**Function**

This command confirms the counter operating status.

**Syntax**

```
:CALCulate:COUNter:STATus[:AUNit]?
```

**Response Data**

0|1

0: Stops operation

1: Running
Adding :AUNit to the header returns 1 when any one of the connected units is started, and 0 when all counters are stopped.

**Example of Use**

Refer to the example for:CALCulate:COUNter:STOP[:AUNit] .

## :CALCulate:COUNter:STOP[:AUNit]

**Function**

This command stops the counter.
Adding :AUNit stops the counters of all connected units.

**Syntax**

```
:CALCulate:COUNter:STOP[:AUNit]
```

*Relationship with Port ID commands
The value of this event is the logical product of the event for each port. Immediately after 1 is read by this command, the event value for each port is cleared to 0.

**Example of Use**

To stop counter:

1. Stop counter by sending :CALCulate:COUNter:STOP.
2. Send repeatedly until the :CALCulate:COUNter:STATus? response becomes 0 (stopped).

## :CALCulate:DATA

### Function

This command queries the measurement results.

### Syntax

```
:CALCulate:DATA? <item>[,<item>,…]
```

<item>=<CHAR>
Sets the ID of the measurement items to be acquired.
For the ID to be set, refer to Appendix A Measurement Item List.

Multiple IDs can be set using comma (,) separators.
However, if the response data string overflows the buffer size (65536 characters), an error is returned (–310,"System error").

### Response Data

<value>[,<value>,...] | --------
Measurement data of set ID
The data format varies according to the set items.
It becomes ------------ when the measurement data is disabled.
The measurement items per lane are queried, separated by commas (,).

When capturing the counter value, select the type of data to be captured (Current or Accumulated) using :CALCulate:DATA:TYPE.

### Example of Use

To query the Rx clock frequency (Hz and ppm):

```
:CALCulate:DATA? RX_FREQ,RX_FREQ_D
> 103125000000,0.0
```

## :CALCulate:DATA:TYPE

**Function**

This command sets and queries the counter data types.

**Syntax**

```
:CALCulate:DATA:TYPE ACCUM|CURRENT
:CALCulate:DATA:TYPE?
```

CURRENT : Current
ACCUM : Accumulated

**Response Data**

ACCUM|CURRENT

**Example of Use**

To query Counter elapsed time, bit error count (Accumulated) and Rx clock frequency:

```
:CALCulate:DATA:TYPE ACCUM
:CALCulate:DATA? ELAPSED,BER_CNT,RX_FREQ
> 20,1,103135312400
```

## :CALCulate:DELay:STARt[:AUNit]

**Function**

This command starts the delay time measurement for the OTU3 or OTU4 application.
[Delay Settings] Mode is [Single].
Adding :AUNit starts the delay time measurements for all units connected at the same time.

**Syntax**

```
:CALCulate:DELay:STARt
```

**Example of Use**

To wait for the start of the delay time measurement:

1. Send :CALCulate:DELay STARt to start the capture.
2. Send :CALCulate:DELay:STARt:EVENt? repeatedly until the response data becomes 1 (started).
3. Send :CALCulate:DELay:STARt:STATus? repeatedly until the response data becomes 0 (measurement result readable).
4. Read the measurement result with :CALCulate:DATA? PM_DELAY.

## :CALCulate:DELay:STARt:EVENt[:AUNit]

**Function**

This command queries whether the delay time measurement for the OTU3 or OTU4 application has been started.

**Syntax**

```
:CALCulate:DELay:STARt:EVENt?
```

**Response Data**

0|1

1: Becomes 1 when the delay time measurement is started.
After reading, it is cleared to 0.

0: The delay time measurement has not been started.

**Example of Use**

Refer to the example for :CALCulate:DELay:STARt[:AUNit].

## :CALCulate:DELay:STATus[:AUNit]

**Function**

This command queries whether the delay time measurement result for the OTU3 or OTU4 application can be read.

**Syntax**

```
:CALCulate:DELay:STATus?
```

**Response Data**

0|1

0: The delay time measurement result can be read.

1: The delay time measurement in progress

"The delay time measurement in progress" means the status where the measurement result is unavailable because the measurement time has not passed.
When delay time measurement result is read at this time, expected measurement results may not be obtained because the delay time measured last time may be read in some cases.

**Example of Use**

Refer to the example for :CALCulate:DELay:STOP[:AUNit].

## :CALCulate:DELay:STOP[:AUNit]

**Function**

This command is sent when the delay time measurement for the OTU3 or OTU4 application is to be stopped.
The delay time measurement stops automatically when the measurement time passes. Therefore, this command does not have to be sent after the measurement time has passed.
Adding :AUNit stops the delay time measurements for all units connected.

**Syntax**

```
:CALCulate:DELay:STOP
```

## :CALCulate:EALarm[:AUNit]

**Function**

This command queries the Error/Alarm LED status in the summary status/time display area.

**Syntax**

```
:CALCulate:EALarm[:AUNit]?
```

**Response Data**

0|1|2
Omitting AUNit displays the information only for units set by :UENTry:ID.

| | |
|---|---|
| 0: | Normal (green) |
| 1: | Abnormal (red) |
| 2: | History (orange) |

Adding AUNit to the header displays the following status.

| | |
|---|---|
| 0: | All connected units normal (green) |
| 1: | One or more connected units abnormal status (red) |
| 2: | One or more connected units has history log (orange)<br>No connected units have had abnormal status (red) |

**Example of Use**

When error/alarm does not occur:

```
:CALCulate:EALarm?
> 0
```

## :CALCulate:MONitor:OTU:COLumn

**Function**

This command sets and queries the read header column when the OTU data monitor is in Frame display.

**Syntax**

```
:CALCulate:MONitor:OTU:COLumn <NR1>
:CALCulate:MONitor:OTU:COLumn?
```

| | |
|---|---|
| <NR1>: | Column number (1 to 4065) |

**Response Data**

<NR1>

**Example of Use**

Refer to the example for :CALCulate:MONitor:OTU:DATA.

## :CALCulate:MONitor:OTU:DATA

**Function**

This command queries the monitor value of the OTU frame.
The header column of the OTU frame monitor data is set by the command, :CALCulate:MONitor:OTU:COLumn.

**Syntax**

```
:CALCulate:MONitor:OTU:DATA? [<offset>,<length>]
```

| | |
|---|---|
| <offset>=<NR1>: | Column number |
| <length>=<NR1>: | Data length |

Omitting <offset> and <length> makes the response data the maximum number of bytes data.
The maximum value that can be specified to the data length depends on the setting of :CALCulate:MONitor:OTU:TYPE.
FRAME, FRAME_L: 64, FTFL, FTFL_L: 255, MSI: 80, TTI: 64

**Response Data**

The response data depends on the setting of :CALCulate:MONitor:OTU:TYPE.

| Type of Monitor Data | Response Data | Format |
|---|---|---|
| FRAME, FRAME_L | <STRING> | Up to 64 bytes hexadecimal by 1 byte with "," separator. Set the top column of the OTU frame monitor data to be obtained with the :CALCulate:MONitor:OTU:COLumn command. |
| FTFL | <STRING> | Up to 256 bytes hexadecimal by 1 byte with "," separator |
| MSI | <STRING> | Up to 80 bytes hexadecimal by 1 byte with "," separator |
| TTI_PM, TTI_PM_L, TTI_SM, TTI_TCM {1\|2\|3\|4\|5\|6} | <STRING> | Up to 64 bytes hexadecimal by 1 byte with "," separator |

***Note:***

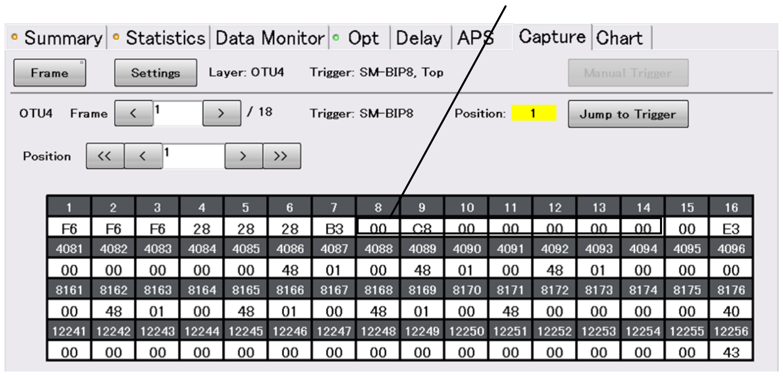
Wait at least 1 second before capturing column data in case of the following cases.

The column of the frame to be captured has been changed with :CALCulate:MONitor:OTU:COLumn.

**Example of Use**

To capture the data of column 8 to 14 in the first column of the OUT frame:

```
:CALCulate:MONitor:OTU:TYPE FRAME
:CALCulate:MONitor:OTU:COLumn 1
:CALCulate:MONitor:OTU:DATA? 7,7
>"00,C8,00,00,00,00,00"
```



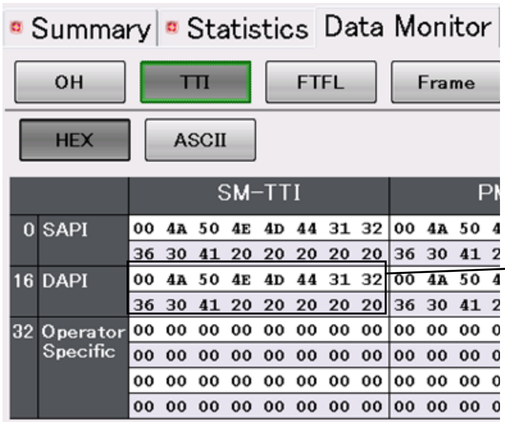
Layer: OTU4 Trigger: SM-BIP8, Top

OTU4 Frame 1 / 18 Trigger: SM-BIP8 Position: 1

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F6 | F6 | F6 | 28 | 28 | 28 | B3 | 00 | C8 | 00 | 00 | 00 | 00 | 00 | 00 | E3 |
| 4081 | 4082 | 4083 | 4084 | 4085 | 4086 | 4087 | 4088 | 4089 | 4090 | 4091 | 4092 | 4093 | 4094 | 4095 | 4096 |
| 00 | 00 | 00 | 00 | 00 | 48 | 01 | 00 | 48 | 01 | 00 | 48 | 01 | 00 | 00 | 00 |
| 8161 | 8162 | 8163 | 8164 | 8165 | 8166 | 8167 | 8168 | 8169 | 8170 | 8171 | 8172 | 8173 | 8174 | 8175 | 8176 |
| 00 | 48 | 01 | 00 | 48 | 01 | 00 | 48 | 01 | 00 | 48 | 00 | 00 | 00 | 00 | 40 |
| 12241 | 12242 | 12243 | 12244 | 12245 | 12246 | 12247 | 12248 | 12249 | 12250 | 12251 | 12252 | 12253 | 12254 | 12255 | 12256 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 43 |

To capture 16 bytes of SM-TTI data:

```
:CALCulate:MONitor:OTU:TYPE TTI_SM
:CALCulate:MONitor:OTU:DATA? 16,16
>"00,4A,50,4E,4D,44,31,32,36,30,41,20,20,20,20,20"
```



| | | SM-TTI | PM |
|---|---|---|---|
| 0 | SAPI | 00 4A 50 4E 4D 44 31 32 | 00 4A 50 4 |
| | | 36 30 41 20 20 20 20 20 | 36 30 41 2 |
| 16 | DAPI | 00 4A 50 4E 4D 44 31 32 | 00 4A 50 4 |
| | | 36 30 41 20 20 20 20 20 | 36 30 41 2 |
| 32 | Operator Specific | 00 00 00 00 00 00 00 00 | 00 00 00 0 |
| | | 00 00 00 00 00 00 00 00 | 00 00 00 0 |
| | | 00 00 00 00 00 00 00 00 | 00 00 00 0 |
| | | 00 00 00 00 00 00 00 00 | 00 00 00 0 |

## :CALCulate:MONitor:OTU:TYPE

### Function

This command sets and queries the monitor data type for the OUT frame.

### Syntax

```
:CALCulate:MONitor:OTU:TYPE <CHAR>
:CALCulate:MONitor:OTU:TYPE?
```

<CHAR>

The header bit becomes bit0.

| <CHAR> | Type of Monitor Data |
|---|---|
| FRAME | () OTU Frame *1 |
| FRAME_L | Low order OTU frame |
| FTFL | FTFL *1 |
| FTFL_L | Low order FTFL |
| MSI | MSI |
| TTI_PM | PM-TTI *1 |
| TTI_PM_L | Low order PM-TTI |
| TTI_SM | SM-TTI |
| TTI_TCM{1\|2\|3\|4\|5\|6} | TCM{1\|2\|3\|4\|5\|6}-TTI |

*1: For the mapping of ODTU4.1 or ODTU4.8, the high order data is specified.

*2: Stuff monitor data cannot be captured with the remote command.

### Response Data

<CHAR>

### Example of Use

```
:CALCulate:MONitor:OTU:TYPE FTFL
:CALCulate:MONitor:OTU:TYPE?
> FTFL

:CALCulate:MONitor:OTU:TYPE TTI_TCM3
:CALCulate:MONitor:OTU:TYPE?
> TTI_TCM3
```

## :CALCulate:TRIGger:CONDition

**Function**

This command sets and queries the capture trigger condition for the 100 GbE or 40 GbE application.

**Syntax**

```
:CALCulate:TRIGger:CONDition <BINARY>
:CALCulate:TRIGger:CONDition?
```

At <BINARY>, the bit for enabling the trigger conditions is set to 1.
The header bit becomes 0.

| bit | Trigger Condition |
|---|---|
| 0 | Good Frame |
| 1 | LF(Local Fault) |
| 2 | RF(Remote Fault) |
| 3 | Error Signal |
| 4 | FCS Error |
| 5 | Undersize |
| 6 | Fragment |
| 7 | Oversize |
| 8 | Oversize & FCS |
| 9-31 | Reservation |

**Response Data**

<<BINARY>>
The bit length is 32 bits.

**Example of Use**

To set Any Frame to the trigger condition:

```
:CALCulate:TRIGger:CONDition #B100011111
:CALCulate:TRIGger:CONDition?
> #B10001111100000000000000000000000
```

To set Any Frame Error to trigger condition:

```
:CALCulate:TRIGger:CONDition #B000011111
```

To set Any MII Error to trigger condition:

```
:CALCulate:TRIGger:CONDition #B0111
```

To set Any Error to the trigger condition:

```
:CALCulate:TRIGger:CONDition #B011111111
```

## :MDIO:READ

### Function

This command queries the value of the CFP MDIO register.

### Syntax

```
:MDIO:READ? <address>
```

<address>=<HEX>
Register address (#H0000 to #HFFFF)

### Response Data

<value>=<HEX>
This is the register value to be read. (#H0000 to #HFFFF)

### Example of Use

To read 0xA016 details of MDIO register:

```
:MDIO:READ? #HA016
> #H0020
```

## :MDIO:WRITe

### Function

This command sets the value of the CFP MDIO register.

### Syntax

```
:MDIO:WRITe <address>,<value>
```

<address>=<HEX>
Register address (#H0000 to #HFFFF)
<value>=<HEX>
Register value (#H0000 to #HFFFF)

## :MMEMory:CATalog

**Function**

This command queries the setting file information to be saved in the following places.
C:\Documents and Settings\Administrator
\My Documents\Anritsu\MD1260A\UserData\Setting

**Syntax**

```
:MMEMory:CATalog?
```

**Response Data**

<NR1> [,<file_entry>,...]

| | |
|---|---|
| <NR1>: | Number of the setting file to be saved<br>The file extension is the number of the MD1260A setting file. |

<file_entry>
The following is output in comma-separated format for each setting file.

| | |
|---|---|
| <file_name>: | <STRING> Format file name<br>Including extension of file name |
| <file type>: | Setting file types |

| | |
|---|---|
| E40G: | 40GbE |
| E40G_N: | 40GbE No Frame |
| E100G: | 100GbE |
| E100G_N: | 100GbE No Frame |
| OTU3: | OTU3 |
| OTU3_N: | OTU3 No Frame |
| OTU4: | OTU4 |
| OTU4_E: | ODU4-100GbE |
| OTU4_ODU2E: | ODTU4.8-ODU2e-PRBS |
| OTU4_ODU2E_E: | ODTU4.8-ODU2e-10GbE |
| OTU4_ODU0: | ODTU4.1-ODU0-PRBS |
| OTU4_ODU0_E: | ODTU4.1-ODU0-GbE |
| OTU4_N : | OTU4 No Frame |

Example: Response data when the following files are in folder
1. 100 GbE setting file called "Setup100GbE"
2. OTU4 setting file called "SetupOTU4"

```
> 2,"Setup100GbE",E100G,"SetupOTU4",OTU4,"AllSamp",ALL
```

Example: Response data when no files in folder

```
> 0
```

**Example of Use**

1. To check currently loading application:

```
:SOURce:MAPPing?
> E100G
```

2. To check file to be saved:

```
:MMEMory:CATalog?
> 2,"Setup100GbE",E100G,"SetupOTU4",OTU4
```

3. To read setting file of currently loading application:

```
:MMEMory:RECall "Setup100GbE"
```

4. To save current measurement result:

```
:MMEMory:STORe "Result100GbE",RESULT
```

## :MMEMory:INITialize

**Function**

This command returns the settings for the MD1260A with the Unit ID specified by to the factory defaults.
However, the following contents are not returned to the factory defaults.

- GPIB address and Windows settings such as IP address
- Transceiver settings
  :SOURce:TRANsceiver:EMPHasis:FIRSt
  :SOURce:TRANsceiver:EMPHasis:PRE
  :SOURce:TRANsceiver:EMPHasis:SECond
  :SOURce:TRANsceiver:VOD
  :SENSe:TRANsceiver:EQUalizer:CONTrol
  :SENSe:TRANsceiver:EQUalizer:DCGain
- Response terminator setting (:SYSTem:TERMination)

**Syntax**

```
:MMEMory:INITialize
```

**Example of Use**

To initialize unit with Unit ID2:

```
:UENTry:ID 2
:MMEMory:INITialize
```

## :MMEMory:LOG:FNAMe

**Function**

This command queries the latest log file name.
When multiple units are connected, the log file of the unit set by :UENTry:ID is queried.

**Syntax**

```
:MMEMory:LOG:FNAMe?
```

**Response Data**

<STRING>: Following file name or ""

- During Log operation: Name of currently output Log file
- Not during Log operation: Name of Log file output immediately previously
- When not outputting Log file again after running application: Empty string ""

**Example of Use**

```
:MMEMory:LOG:FNAMe?
> "Test20100301T123456_U010000_0.csv"
```

## :MMEMory:LOG:ITEM

**Function**

This sets the items out to the Log.
When multiple units are connected, the item output to the log for the unit set by :UENTry:ID is set.

**Syntax**

```
:MMEMory:LOG:ITEM <id1>[,<id2>,...]
:MMEMory:LOG:ITEM?
```

<id>
Specifies ID of items output to Log
To set the ID, refer to Appendix A Measurement Item List.
ID can be specified as a comma-separated order.

**Response Data**

<id1>[,<id2>,...] | OFF
When there are no output items, it becomes OFF.

**Example of Use**

```
:MMEMory:LOG:ITEM?
> OFF
```

## :MMEMory:LOG:PREFix

**Function**

This command sets the header of the character strings for the log file name.
When multiple units are connected, the same character string is set at all units.

**Syntax**

```
:MMEMory:LOG:PREFix <STRING>
:MMEMory:LOG:PREFix?
```

<STRING>
To set the header of the character strings for the log file name to Test
Example: "Test"

**Response Data**

<STRING>

**Example of Use**

```
:MMEMory:LOG:PREFix "Test"
:MMEMory:LOG:PREFix?
"Test"
```

## :MMEMory:LOG:STARt

**Function**

This command starts Log output. If the Log is already being output, it creates a new Log file and starts Log output.
When multiple units are connected, the log output is started at all units.

**Syntax**

```
:MMEMory:LOG:STARt
```

**Example of Use**

Refer to the example for :MMEMory:LOG:STATus.

## :MMEMory:LOG:STATus

**Function**

This command queries the Log output operation status.

**Syntax**

```
:MMEMory:LOG:STATus?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Log output stopped |
| 1: | Outputting Log |

Displays following when multiple units connected

| | |
|---|---|
| 0: | Log output stopped for all units |
| 1: | Outputting Log for one or more units |

**Example of Use**

```
:MMEMory:LOG:STARt
:MMEMory:LOG:STATus?
> 1
```

## :MMEMory:LOG:STOP

**Function**

This command stops Log output.
When multiple units are connected, the log output is stopped at all units.

**Syntax**

```
:MMEMory:LOG:STOP
```

**Example of Use**

```
:MMEMory:LOG:STOP
:MMEMory:LOG:STATus?
> 0
```

## :MMEMory:LOG:TIMing

**Function**

This command sets the timing of the log output.
When multiple units are connected, the same log output timing is set at all units.
It also queries the timing setting of the log output.

**Syntax**

```
:MMEMory:LOG:TIMing <CHAR>
:MMEMory:LOG:TIMing?
```

<CHAR>
Select as follows:
E1S : Every 1 s
E10S : Every 10 s
E1MIN : Every 1 minute
ERROR : When error occurs

**Response Data**

<CHAR>

**Example of Use**

```
:MMEMory:LOG:TIMing E1S
:MMEMory:LOG:TIMing?
> E1S
```

## :MMEMory:RECall[:AUNit]

**Function**

This command reads the application settings from a file.
Adding :AUNit reads the applications settings files for all connected units.

**Syntax**

```
:MMEMory:RECall[:AUNit] <filename>[,<select>]
```

<filename>=<STRING>
Set the name of the file to be read in <STRING> format.
The file name does not include the directory or extension.

This command reads the settings file that matches the currently running application.
<select>=<CHAR>
Set the range in the <CHAR> format when only specific information is selected among information included in the file to be loaded.
STREAM : Only Stream setting information in the file
If this parameter is omitted, all information items in the file will be loaded.
For :MMEMory:RECall:AUNit, this parameter is omitted and all information items in the file are loaded.

If a difference is detected between the current configuration and unit configuration saved in the file, the –310 (System) error is returned.

**Example of Use**

To read settings file for file named "Setup100GbE" in unit specified by :UENTry:ID:

```
:MMEMory:RECall "Setup100GbE"
```

To read the settings file for file named "SetupAll" in all units.

```
:MMEMory:RECall:AUNit "SetupAll"
```

## :MMEMory:STORe[:AUNit]

**Function**

This command saves the application settings.
Adding :AUNit saves the applications settings files for all connected units
The measurement results report is output.
Adding :AUNit outputs the a measurement results report for all connected units.

**Syntax**

```
:MMEMory:STORe[:AUNit] <filename>,RESULT|SETUP
```

<filename>
Set the name of the file to be saved in <STRING> format.
The file name does not include the directory or extension.

Select the saved contents from the following.

| | |
|---|---|
| SETUP: | Sets application |
| RESULT: | Outputs the report of the measurement result |

**Example of Use**

Refer to the example for :MMEMory:CATalog.
Example: To save the settings file for the unit specified by :UENTry:ID as the file named "Setup100GbE".

```
:MMEMory:STORe "Setup100GbE",SETUP
```

Example : To save the settings file for the unit specified by :UENTry:ID as the file named " Result100GbE".

```
:MMEMory:STORe "Result100GbE",RESULT
```

Example: To save the settings file for all units as the file named " SetupAll ".

```
:MMEMory:STORe:AUNit "SetupAll",SETUP
```

Example: To save the settings file for all unit results as the file named " ResultAll ".

```
:MMEMory:STORe:AUNit "ResultAll",RESULT
```

## :MMEMory:STORe:CAPTure

**Function**

This command saves the currently displayed capture data to the file.

**Syntax**

```
:MMEMory:STORe:CAPTure <filename>
```

<filename>

The file name to be saved (not including the directory name and extinction) is specified in <STRING> format.

For the 40GbE or100GbE application, the contents to be output to the file is specified with :MMEMory:STORe:CAPTure:ITEM.
When the captured data is not displayed, the –310 (System) error is displayed.
Captured data files cannot be saved for all connected multiple units.

## :MMEMory:STORe:CAPTure:ITEM

**Function**

This command sets the file output item of the capture data.

**Syntax**

```
:MMEMory:STORe:CAPTure:ITEM <id>
```

<id>

To set any of the following:
LIBPCAP : Creates nsec libpcap format (*.pcap) file
TEXT_TABLE: Outputs contents of summary area as text file
TEXT_BYTES: Outputs all data contents of detailed areas as text file

Capture file output items cannot be specified for all connected multiple units.

## :MMEMory:STReam:RECall

**Function**

This command reads out only the stream information from the application setting file.
It can be used with 40GbE/100GbE application.

**Syntax**

```
:MMEMory:STReam:RECall <filename>[,<file_type>]
```

<filename>=<STRING>
Set the file name to be loaded.The file name does not include a directory name and extension.

<file_type>=E100G | E40G

| | |
|---|---|
| E100G: | 100GbE measurement condition file |
| E40G: | 40GbE measurement condition file |

If <file_type> is omitted, the measurement condition file of the currently running application will be set.

-310 (System error) is returned when the specified file has been saved with Frame BERT [On].

**Example of Use**

To load the setting file with the name of "MPLS_IPv4_100GbE" to the unit set with :UENTry:ID.

```
:MMEMory:STReam:RECall " MPLS_IPv4_100GbE",E100G
```

## :PROTocol:ARPNs:REPLy:ENABle

**Function**

This command sets whether to send ARP/NA responses to Streams 1 to 16 and queries whether it is set to send ARP/NA responses to Streams 1 to 16.

**Syntax**

```
:PROTocol:ARPNs:REPLy:ENABle <NR1>,<NR1>,…,<NR1>
:PROTocol:ARPNs:REPLy:ENABle?
```

<NR1>=0|1

| | |
|---|---|
| 0: | Not transmitting ARP reply |
| 1: | Transmitting ARP reply |

**Response Data**

<NR1>,<NR1>,…,<NR1>

**Example of Use**

To send ARP/NA responses to Streams 1 to 8 and not to send ARP/NA responses to Streams 9 to 16.

```
:PROTocol:ARPNs:REPLy:ENABle
1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0
:PROTocol:ARPNs:REPLy:ENABle?
> 1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0
```

## :PROTocol:GARPns:DURation:TYPE

### Function

This command sets the method of sending GARP or NS.

### Syntax

```
:PROTocol:GARPns:DURation:TYPE <char>
:PROTocol:GARPns:DURation:TYPE?
```

<char>=REPEAT|SINGLE

| | |
|---|---|
| REPEAT: | Repeat |
| SINGLE: | Single |

### Response Data

REPEAT|SINGLE

### Example of Use

```
:PROTocol:GARPns:DURation:TYPE SINGLE
:PROTocol:GARPns:DURation:TYPE?
> SINGLE
```

## :PROTocol:GARPns:ENABle

**Function**

This command sets whether to send GARP or NS to Streams 1 to 16 and queries whether it is set to send GARP or NS to Streams 1 to 16.

**Syntax**

```
:PROTocol:GARPns:ENABle <NR1>,<NR1>,…,<NR1>
:PROTocol:GARPns:ENABle?
```

<NR1>=0|1

| | |
|---|---|
| 0: | Not transmitting GARP/NS reply |
| 1: | Transmitting GARP/NS |

**Response Data**

<NR1>,<NR1>,…,<NR1>

**Example of Use**

To send GARP or NS to Streams 1 to 8 and not to send GARP or NS to Streams 9 to 16.

```
:PROTocol:GARPns:ENABle 1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0
:PROTocol:GARPns:ENABle?
> 1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0
```

## :PROTocol:GARPns:INTerval

**Function**

This command sets and queries the interval time of sending ARP or NS packets.

**Syntax**

```
:PROTocol:GARPns:INTerval <NR1>
:PROTocol:GARPns:INTerval?
```

<NR1>=1 to 100

**Response Data**

<NR1>

**Example of Use**

To set the interval time of sending ARP or NS packets to 10:

```
:PROTocol:GARPns:INTerval 10
:PROTocol:GARPns:INTerval?
> 10
```

## :PROTocol:GARPns:STARt

**Function**

This command starts to send GARP or NS.

**Syntax**

```
:PROTocol:GARPns:STARt
```

## :PROTocol:GARPns:STATus

**Function**

This command queries the status of sending GARP or NS.

**Syntax**

```
:PROTocol:GARPns:STATus?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Sending GARP/NS completed. |
| 1: | Sending GARP/NS in progress. |

**Example of Use**

```
:PROTocol:GARPns:STARt
:PROTocol:GARPns:STATus?
> 1
```

## :PROTocol:GARPns:STOP

**Function**

This command stops to send GARP or NS.

**Syntax**

```
:PROTocol:GARPns:STOP
```

**Example of Use**

```
:PROTocol:GARPns:STOP
:PROTocol:GARPns:STATus?
> 0
```

## :PROTocol:GARPns:TYPE

**Function**

This command sets and queries the type of GARP.

**Syntax**

```
:PROTocol:GARPns:TYPE <CHAR>
:PROTocol:GARPns:TYPE?
```

<CHAR>=REPLY|REQUEST

| | |
|---|---|
| REPLY: | Reply |
| REQUEST: | Request |

**Response Data**

REPLY|REQUEST

**Example of Use**

```
:PROTocol:GARPns:TYPE REQUEST
:PROTocol:GARPns:TYPE?
> REQUEST
```

## :PROTocol:PING:COUNt

**Function**

This command sets and queries the number of times the Ping test will run.

**Syntax**

```
:PROTocol:PING:COUNt <NR1>
:PROTocol:PING:COUNt?
```

<NR1>=1 to 100

**Response Data**

<NR1>

**Example of Use**

```
:PROTocol:PING:COUNt 20
:PROTocol:PING:COUNt?
>20
```

## :PROTocol:PING:DST:IPV4

**Function**

This command sets and queries the target IP address (IPv4) for the Ping test.

**Syntax**

```
:PROTocol:PING:DST:IPV4 <HEX>
:PROTocol:PING:DST:IPV4?
```

<HEX>: Target IP address for the Ping test (4-byte hexadecimal number)

**Example of Use**

To set target IPv4 address for Ping test to 192.168.1.2 (0xC0A80102):

```
:PROTocol:PING:DST:IPV4 #HC0A80102
:PROTocol:PING:DST:IPV4?
>#HC0A80102
```

## :PROTocol:PING:DST:IPV6

**Function**

This command sets and queries the target IP address (IPv6) for the Ping test.

**Syntax**

```
:PROTocol:PING:DST:IPV6 <HEX>
:PROTocol:PING:DST:IPV6?
```

<HEX>: Target IP address for the Ping test (16-byte hexadecimal number)

**Example of Use**

To set target IPv6 address for Ping test to 2001::10:

```
:PROTocol:PING:DST:IPV6
#H20010000000000000000000000000010
:PROTocol:PING:DST:IPV6?
>#H20010000000000000000000000000010
```

## :PROTocol:PING:DST:MAC

**Function**

This command sets and queries the target MAC address for the Ping test.

**Syntax**

```
:PROTocol:PING:DST:MAC <HEX>
:PROTocol:PING:DST:MAC?
```

<HEX>: Target MAC address for the Ping test (6-byte hexadecimal number)

**Example of Use**

To set target MAC address for Ping test to 0x0A0000123456:

```
:PROTocol:PING:DST:MAC #H0A0000123456
:PROTocol:PING:DST:MAC?
>#H0A0000123456
```

## :PROTocol:PING:IPMode

**Function**

This command sets and queries the IP version of the Ping test.

**Syntax**

```
:PROTocol:PING:IPMode IPV4|IPV6
:PROTocol:PING:IPMode?
```

| | |
|---|---|
| IPV4: | IPv4 |
| IPV6: | IPv6 |

**Response Data**

IPV4|IPV6

**Example of Use**

```
:PROTocol:PING:IPMode IPV6
:PROTocol:PING:IPMode?
> IPV6
```

## :PROTocol:PING:MACResolve

**Function**

This command sets to turn on and off the MAC address resolution for the Ping test or queries the on/off state.
If the MAC address resolution is turned on, the target IP address automatically resolves the target MAC address.

**Syntax**

```
:PROTocol:PING:MACResolve 0|1|ON|OFF
:PROTocol:PING:MACResolve?
```

| | |
|---|---|
| ON\|1: | MAC Resolve On. |
| OFF\|0: | MAC Resolve Off. |

**Response Data**

0|1

**Example of Use**

```
:PROTocol:PING:MACResolve 1
:PROTocol:PING:MACResolve?
> 1
```

## :PROTocol:PING:PAYLoad

**Function**

This command sets and queries the pattern of packet payload for the Ping test.

**Syntax**

```
:PROTocol:PING:PAYLoad ALL0|ALL1|ALTERNATE
:PROTocol:PING:PAYLoad?
```

| | |
|---|---|
| ALL0: | All 0 |
| ALL1: | All 1 |
| ALTERNATE: | 0/1 bit |

**Response Data**

ALL0|ALL1|ALTERNATE

**Example of Use**

```
:PROTocol:PING:PAYLoad ALL0
:PROTocol:PING:PAYLoad?
> ALL0
```

## :PROTocol:PING:PSIZe:TYPE

**Function**

This command sets and queries the mode of packet size for the Ping test.

**Syntax**

```
:PROTocol:PING:PSIZe:TYPE FIX|INCREMENT
:PROTocol:PING:PSIZe:TYPE?
```

| | |
|---|---|
| FIX: | Fixed |
| INCREMENT: | Increment |

**Response Data**

FIX|INCREMENT

**Example of Use**

```
:PROTocol:PING:PSIZe:TYPE FIX
:PROTocol:PING:PSIZe:TYPE?
> FIX
```

## :PROTocol:PING:PSIZe:VALUe

**Function**

This command sets and queries the packet size for the Ping test.

**Syntax**

```
:PROTocol:PING:PSIZe:VALUe <start>[,<step>]
:PROTocol:PING:PSIZe:VALUe?
```

<start>,<step>=<NR1>

| | |
|---|---|
| <start>: | Lower limit for packet size. |
| <step>: | Change in packet size. |

If the packet size increment is Off, then the value in <step> will be ignored.

**Response Data**

<NR1>,<NR1>

**Example of Use**

```
:PROTocol:PING:PSIZe:VALUe 64,8
:PROTocol:PING:PSIZe:VALUe?
> 64,8
```

## :PROTocol:PING:REPLy:ENABle

**Function**

This command sets whether to send Ping responses to Streams 1 to 16 and queries whether it is set to send Ping responses to Streams 1 to 16.

**Syntax**

```
:PROTocol:PING:REPLy:ENABle <NR1>,<NR1>,…,<NR1>
:PROTocol:PING:REPLy:ENABle?
```

<NR1>=0|1

| | |
|---|---|
| 0: | Not transmitting Ping reply |
| 1: | Transmitting Ping reply |

**Response Data**

<NR1>,<NR1>,…,<NR1>

**Example of Use**

To send Ping responses to Streams 1 to 4 and not to send Ping responses to Streams 5 to 16.

```
:PROTocol:PING:REPLy:ENABle
1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
:PROTocol:PING:REPLy:ENABle?
> 1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0
```

## :PROTocol:PING:RESult

### Function

This command queries the result of the Ping test.

### Syntax

```
:PROTocol:PING:RESult? <CHAR>
```

<CHAR>=HISTORY1 | HISTORY2 | … | HISTORY32 | TOTAL

| | |
|---|---|
| HISTORY1 | First result in the History table. |
| HISTORY2 | Second result in the History table. |
| : | |
| HISTORY32 | 32nd result in the History table. |
| TOTAL: | Result in the Total table. |

### Response Data

If HISTORY1 to HISTORY32 is specified,

| | |
|---|---|
| <from>=<HEX>: | IP address |
| <size>=<NR1>: | Packet size |
| <ttl>=<NR1>: | TTL |
| <time>=<NR1>: | Round trip time [ms] |

Specifying the History number without measurement result causes a parameter error.

If TOTAL is specified,

| | |
|---|---|
| <tx_arpreq>=<NR1>: | Tx ARP Request |
| <rx_arprep>=<NR1>: | Number of times to receive Rx ARP Reply. |
| <tx_pingreq>=<NR1>: | Number of times to send Tx Ping Request. |
| <rx_pingrep>=<NR1>: | Rx Ping Reply |
| <time_min>=<NR1>: | Time Min. |
| <time_max>=<NR1>: | Time Max. |
| <time_ave>=<NR1>: | Time Ave. |
| <timeout>=<NR1>: | Timeout |

### Example of Use

```
:PROTocol:PING:RESult? HISTORY2
> #HC0A80101,128,128,10
```

## :PROTocol:PING:SRC:IPV4

**Function**

This command sets and queries the source IP address (IPv4) for the Ping test.

**Syntax**

```
:PROTocol:PING:SRC:IPV4 <HEX>
:PROTocol:PING:SRC:IPV4?
```

<HEX>: Source IP address for Ping test (4-byte hexadecimal number)

**Example of Use**

To set source IPv4 address for Ping test to 192.168.1.4 (0xC0A80104):

```
:PROTocol:PING:SRC:IPV4 #HC0A80104
:PROTocol:PING:SRC:IPV4?
>#HC0A80104
```

## :PROTocol:PING:SRC:IPV6

**Function**

This command sets and queries the source IP (IPv6) address for the Ping test.

**Syntax**

```
:PROTocol:PING:SRC:IPV6 <HEX>
:PROTocol:PING:SRC:IPV6?
```

<HEX>: Source IP address for Ping test (16-byte hexadecimal number)

**Example of Use**

To set source IPv6 address for Ping test to 2001::12:

```
:PROTocol:PING:SRC:IPV6
#H20010000000000000000000000000012
:PROTocol:PING:SRC:IPV6?
>#H20010000000000000000000000000012
```

## :PROTocol:PING:SRC:MAC

**Function**

This command sets and queries the source MAC address for the Ping test.

**Syntax**

```
:PROTocol:PING:SRC:MAC <HEX>
:PROTocol:PING:SRC:MAC?
```

<HEX>: Source MAC address for Ping test (6-byte hexadecimal number)

**Example of Use**

To set source MAC address for Ping test to 0x0A0000123478:

```
:PROTocol:PING:DEST:MAC #H0A0000123478
:PROTocol:PING:DEST:MAC?
>#H0A0000123478
```

## :PROTocol:PING:STARt

**Function**

This command starts the Ping test.

**Syntax**

```
:PROTocol:PING:STARt
```

## :PROTocol:PING:STARt:EVENt

**Function**

This command queries the start event for the Ping test.
If the start event is read, then the response data is 0.

**Syntax**

```
:PROTocol:PING:STARt:EVENt?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Ping test not executed.<br>Event has been read after the start of Ping test. |
| 1: | Ping test started. |

**Example of Use**

```
:PROTocol:PING:STARt
```

```
:PROTocol:PING:STARt:EVENt?
> 1
:PROTocol:PING:STARt:EVENt?
> 0
```

## :PROTocol:PING:STATus

**Function**

This command queries the status of the Ping test.

**Syntax**

```
:PROTocol:PING:STATus?
```

**Response Data**

0|1|2

| | |
|---|---|
| 0: | Ping test completed. |
| 1: | Ping test in progress. |
| 2: | Execution error. |

**Example of Use**

```
:PROTocol:PING:STARt
:PROTocol:PING:STATus?
> 1
```

## :PROTocol:PING:STOP

**Function**

This command stops the Ping test.

**Syntax**

```
:PROTocol:PING:STOP
```

**Example of Use**

```
:PROTocol:PING:STOP
:PROTocol:PING:STATus?
> 0
```

## :PROTocol:PING:TIMeout

**Function**

This command sets and queries the timeout for the Ping test.

**Syntax**

```
:PROTocol:PING:TIMeout <NR1>
:PROTocol:PING:TIMeout?
```

| | |
|---|---|
| <NR1>: | Ping timeout 1 to 10 seconds |

**Response Data**

<NR1>

**Example of Use**

```
:PROTocol:PING:TIMeout 10
:PROTocol:PING:TIMeout?
> 10
```

## :PROTocol:PING:VLAN

**Function**

This command sets and queries VLAN for Ping test transmission frames.

**Syntax**

```
:PROTocol:PING:VLAN
<stack>[,<tpid1>,<pcp1>,<vid1>[,<tpid2>,<pcp2>,<vid2>]]
:PROTocol:PING:VLAN?
```

<stack>=0|1|2
<tpid1>, <tpid2>=<HEX>
<pcp1>,<vid1>, <pcp1>,<vid1>=<NR1>
If 0 is specified to <stack>, then <tpid1> and later can be omitted.
If 1 is specified to <stack>, then <tpid2> and later can be omitted.

**Response Data**

<stack>[,<tpid1>,<pcp1>,<vid1>[,<tpid2>,<pcp2>,<vid2>]]

**Example of Use**

```
:PROTocol:PING:VLAN 2,#H88A8,0,0,#H8100,0,0
:PROTocol:PING:VLAN?
> 2,#H88A8,0,0,#H8100,0,0
```

## :ROUTe:BERT

**Function**

This command sets the Port setting Frame BERT setting On/Off.
It can also be used to query the Port setting Frame BERT setting.

**Syntax**

```
:ROUTe:BERT 0|1|OFF|ON
:ROUTe:BERT?
```

| | |
|---|---|
| 0\|OFF: | Frame BERT setting Off |
| 1\|ON: | Frame BERT setting On |

**Response Data**

0|1

## :ROUTe:ETHer:NEGotiation:AUTO

**Function**

This command sets and queries the auto negotiation when the mapping is GbE for the OTU3 or OTU4 application.

**Syntax**

```
:ROUTe:ETHer:NEGotiation:AUTO 0|1|OFF|ON
:ROUTe:ETHer:NEGotiation:AUTO?
```

| | |
|---|---|
| 0\|OFF: | Auto negotiation Off |
| 1\|ON: | Auto negotiation On |

**Response Data**

0|1

**Example of Use**

```
:ROUTe:ETHer:NEGotiation:AUTO ON
:ROUTe:ETHer:NEGotiation:AUTO?
> 1
```

## :ROUTe:FCONtrol

**Function**

This command sets the Port setting Flow Control setting On/Off.
It can also be used to query the Port setting Flow Control setting.

**Syntax**

```
:ROUTe:FCONtrol 0|1|OFF|ON
:ROUTe:FCONtrol?
```

| | |
|---|---|
| 0 \| OFF: | Flow Control setting Off |
| 1 \| ON: | Flow Control setting On |

**Response Data**

**0|1**

## :ROUTe:LFS:REPLy

**Function**

This command sets and queries the On/Off setting of LFS Reply in Port setting for the 40GbE or 100GbE application.
It cannot be used when the mapping is 40GbE or10GbE for the OTU3 application and when the mapping is 100GbE or 10GbE for the OTU4 application.

**Syntax**

```
:ROUTe:LFS:REPLy 0|1|OFF|ON
:ROUTe:LFS:REPLy?
```

| | |
|---|---|
| 0 \| OFF: | LFS Reply setting Off |
| 1 \| ON: | LFS Reply setting On |

**Response Data**

0 | 1

## :ROUTe:MODE

**Function**

This command sets the Port mode.

**Syntax**

```
:ROUTe:MODE LOOPBACK|NORMAL|THROUGH
:ROUTe:MODE?
```

Set any of the following:
NORMAL: Release normal mode
LOOPBACK: Set loopback mode
THROUGH: Through mode
Can be set for the OTN3 or OTN4 application.

**Response Data**

LOOPBACK|NORMAL|THROUGH

**Example of Use**

To set to Loopback:

```
:ROUTe:MODE LOOPBACK
:ROUTe:MODE?
> LOOPBACK
```

## :ROUTe:MODE:THRough:OVERwrite:RANGe

### Function

This command sets and queries the range of data to be overwritten when the through mode is OH Overwrite.
The data range can be set for the OTU3 or OTU4 application.

### Syntax

```
:ROUTe:MODE:THRough:OVERwrite:RANGe
<type>[,<row>,<column>]
:ROUTe:MODE:THRough:OVERwrite:RANGe
```

<type>=ALL|BYTE|ODU|OPU|OTU|OTU_ODU
<row>=<NR1>:For BYTE of <type>, the data row number is 1 to 4.
<column>=<NR1>: For BYTE of <type>, the data column number is 1 to 16.

### Response Data

<type>[,<row>,<column>]

### Example of Use

To set only 1 byte of row 4 and column 13 to overwrite:

```
:ROUTe:MODE:THRough:OVERwrite:RANGe BYTE,4,13
:ROUTe:MODE:THRough:OVERwrite:RANGe?
> BYTE,4,13
```

### :ROUTe:MODE:THRough:TYPE

**Function**

This command sets and queries the type of the through mode.
It can be set for the OTU3 or OTU4 application.

**Syntax**

```
:ROUTe:MODE:THRough:TYPE ANALYZED|OVERWRITE|TRANSPARENT
:ROUTe:MODE:THRough:TYPE?
```

Set any of the following:
ANALYZED
OVERWRITE
TRANSPARENT

**Response Data**

ANALYZED|OVERWRITE|TRANSPARENT

**Example of Use**

To set the type of the through mode to OH Overwrite:

```
:ROUTe:MODE:THRough:TYPE OVERWRITE
:ROUTe:MODE:THRough:TYPE?
> OVERWRITE
```

### :ROUTe:MPLStp:CWORd

**Function**

This command sets On/Off of the reception signal MPLS-TP Control Word.
This can be set for the 40 GbE/100 GbE applications.

**Syntax**

```
:ROUTe:MPLStp:CWORd 0|1
:ROUTe:MPLStp:CWORd?
```

| | |
|---|---|
| 0: | Off (Control Word Off) |
| 1: | On (Control Word On) |

**Response Data**

0|1

## :ROUTe:VLAN:NUM

**Function**

For 40GbE/100GbE applications, this command sets and queries the number of VLAN filters for Port setting.

**Syntax**

```
:ROUTe:VLAN:NUM <NR1>
:ROUTe:VLAN:NUM?
```

<NR1>=0 to 16

**Response Data**

<NR1>

**Example of Use**

To set the number of filters to 4.

```
:ROUTe:VLAN:NUM 4
:ROUTe:VLAN:NUM?
> 4
```

## :ROUTe:VLAN:VALue

**Function**

For 40GbE/100GbE applications, this command sets and queries the VLAN stack and TPID for Port setting.

**Syntax**

```
:ROUTe:VLAN:VALue <no>,<stack>,<tpid1>,<tpid2>
:ROUTe:VLAN:VALue? <no>
```

| | |
|---|---|
| <no>=<NR1>: | Filter number 1 to 16 |
| <stack>=1\|2: | Stack |
| <tpid1>=<HEX>: | VLAN#1 TPID 0x0000 to 0xFFFF |
| <tpid2>=<HEX>: | VLAN#2 TPID 0x0000 to 0xFFFF |

**Response Data**

<no>,<stack>,<tpid1>,<tpid2>

**Example of Use**

To set the filter number 1 to VLAN stack 2, TPID1 0x88A2, and TPID2 0x8102.

```
:ROUTe:VLAN:VALue 1,2,#H88A2,#H8102
:ROUTe:VLAN:VALue? 1
```

```
> 2,#H88A2,#H8102
```

## :SENSe:MAPPing

**Function**

This command operates in the same way as :SOURce:MAPPing.

## :SENSe:ODTU:MAIN:DETect

**Function**

This command sets and queries the Main TS (Tributary Slot) detection method on the receive side.
Setting Main TS detection method sets the Combination of the TP/TS setting to Off.

**Syntax**

```
:SENSe:ODTU:MAIN:DETect <CHAR>
:SENSe:ODTU:MAIN:DETect?
```

Specify either of the following for <CHAR>.

| <CHAR> | Description |
|---|---|
| MANUAL | Specifies TS with :SENSe:ODTU:MAIN:TS. |
| TP<NR1> | Detects the TP number automatically. |
| TS<NR1> | Automatically detects the TP number stored in the specified TS number. |

<NR1>: For ODTU4.1: 1 to 80. For ODTU4.8: 1 to 10

**Response Data**

<CHAR>

**Example of Use**

For ODTU4.1:

```
:SENSe:ODTU:MAIN:DETect TP80
:SENSe:ODTU:MAIN:DETect?
> TP80

:SENSe:ODTU:MAIN:DETect MANUAL
:SENSe:ODTU:MAIN:DETect?
> MANUAL
```

## :SENSe:ODTU:MAIN:TS

**Function**

This command sets and queries the Main TS (Tributary Slot) on the receive side.
One TS for ODTU4.1 and 8 TSs for ODTU4.8 are set. Setting TS sets the Combination of the TP/TS setting to Off.

**Syntax**

```
:SENSe:ODTU:MAIN:TS
<NR1>|<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>
:SENSe:ODTU:MAIN:TS?
```

| | |
|---|---|
| <NR1>: | For ODTU4.1: 1~80 |
| | For ODTU4.8: 1~10 |

**Response Data**

<NR1>|{<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>}

**Example of Use**

For ODTU4.1:

```
:SENSe:ODTU:MAIN:TS 80
:SENSe:ODTU:MAIN:TS?
> 80
```

For ODTU4.8:

```
:SENSe:ODTU:MAIN:TS 1,2,3,4,5,6,7,8
:SENSe:ODTU:MAIN:TS?
> 1,2,3,4,5,6,7,8
```

## :SENSe:OTN:FEC

**Function**

This command sets and queries GFEC Decode for Port setting.

**Syntax**

```
:SENSe:OTN:FEC 0|1|OFF|ON
:SENSe:OTN:FEC?
```

| | |
|---|---|
| 0\|OFF: | GFEC Decode setting Off |
| 1\|ON: | GFEC Decode setting On |

**Response Data**

0|1

**Example of Use**

To set GFEC Decode to On.

```
:SENSe:OTN:FEC ON
:SENSe:OTN:FEC?
> 1
```

## :SENSe:PLM[:L]:PATTern

**Function**

This command sets and queries the payload type to detect PLM.
To set and query the low order payload type, describe :L.

**Syntax**

```
:SENSe:PLM[:L]:PATTern <HEX>
:SENSe:PLM[:L]:PATTern?
```

<HEX>: 1-byte value to indicate the payload type

**Response Data**

<HEX>

**Example of Use**

To set 0x12 to the high order payload type:

```
:SENSe:PLM:PATTern #H21
```

To set 0x07 to the low order payload type:

```
:SENSe:PLM:L:PATTern #H07
```

## :SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}

### Function

This command sets and queries the detection method for `TIM`.
To set and query the low order PM-TIM detection method, describe :L.

### Syntax

```
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}} <CHAR>
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}?
```

<CHAR>
Select the `TIM` detection method from the following:

| | |
|---|---|
| OFF: | TIM is not detected. |
| DAPI: | TIM is detected only for DAPI. |
| SAPI: | TIM is detected only for SAPI. |
| SAPI_DAPI: | TIM is detected for SAPI and DAPI. |

### Response Data

DAPI | DAPI_SAPI | OFF | SAPI

### Example of Use

To set SM-TIM detection method to Off:
```
:SENSe:TIM:SM OFF
```
To set PM-TIM detection method to DAPI:
```
:SENSe:TIM:PM DAPI
```
To set TCM2-TIM detection method to SAPI and DAPI:
```
:SENSe:TIM:TCM2 SAPI_DAPI
```
To query the high order PM-TIM detection method:
```
:SENSe:TIM:PM?
> DAPI_SAPI
```
To query the low order PM-TIM detection method:
```
:SENSe:TIM:PM:L?
> OFF
```

## :SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}:PATTern:{DAPI|SAPI}

**Function**

This command sets and queries the TIM detection pattern.
To set and query the low order PM-TIM detection pattern, describe :L.

**Syntax**

```
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}:PATTern:{DAPI|SA
PI} <HEX>
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}:PATTern:{DAPI|SA
PI}?
```

<HEX>: Value of 16 bytes or less to indicate the pattern data

**Response Data**

<HEX>

**Example of Use**

To set "JPNMD1260A " to DAPI for SM-TTI:

```
:SENSe:TIM:SM:PATTern:DAPI
#H004A504E4D44313236304120202020202020
```

To set "USAMD1260A " to DAPI for SM-TTI:

```
:SENSe:TIM:TCM6:PATTern:SAPI
#H005553404D44313236304120202020202020
```

## :SENSe:TPATtern:INVert

### Function

In case of application is not No Frame, this command inverts the PRBS test pattern settings (Rx). Use :SOURce:TPATtern:INVert to invert the Tx pattern.

### Syntax

```
:SENSe:TPATtern:INVert 0|1|OFF|ON
:SENSe:TPATtern:INVert?
```

| | |
|---|---|
| 0\|OFF: | Invert pattern Off |
| 1\|ON: | Invert pattern On |

### Response Data

0|1

### Example of Use

To invert test pattern for both Tx and Rx:

:SENSe:TPATtern:INVert ON

```
:SOURce:TPATtern:INVert ON
:SENSe:TPATtern:INVert?
> 1
:SOURce:TPATtern:INVert?
> 1
```

## :SENSe:TPATtern:TYPE

**Function**

This command selects the test pattern type of receive side (Rx).
It queries the test pattern type of receive side (Rx).
The setting value of this command is linked with :SOURce:TPATtern:TYPE. Sending either one of these commands sets both the transmit side and the receive side.

**Syntax**

```
:SENSe:TPATtern:TYPE <CHAR>
:SENSe:TPATtern:TYPE?
```

<CHAR>
Select the test pattern type from the following:
PRBS7
PRBS9
PRBS11
PRBS15
PRBS23
PRBS31
ALL0
ALL1
WORD16
SQUARE : Square Wave

**Response Data**

ALL0|ALL1|PRBS7|PRBS9|PRBS11|PRBS15|PRBS23|PRBS31|SQUARE|WORD16

**Example of Use**

```
:SENSe:TPATtern:TYPE WORD16
:SENSe:TPATtern:TYPE?
>WORD16
```

## :SENSe:TPATtern:WORD

**Function**

This command sets the pattern when Word16 is selected for the test pattern of receive side (Rx).
It queries the Word16 pattern of receive side (Rx).
The setting value of this command is linked with :SOURce:TPATtern:WORD. Sending either one of these commands sets both the transmit side and the receive side.

**Syntax**

```
:SENSe:TPATtern:WORD <HEX>
:SENSe:TPATtern:WORD?
```

|  |  |
|---|---|
| <HEX>: | Value of Data Field (2 bytes) |

**Response Data**

<HEX>

**Example of Use**

```
:SENSe:TPATtern:TYPE WORD16
:SENSe:TPATtern:WORD #HFF02
:SENSe:TPATtern:WORD?
>#HFF02
```

## :SENSe:TRANsceiver:EQUalizer:CONTrol

**Function**

This command sets the Equalizer (Control) of the transceiver.
It also queries the settings of the Equalizer (Control) of the transceiver.

**Syntax**

```
:SENSe:TRANsceiver:EQUalizer:CONTrol <lane>,<value >
:SENSe:TRANsceiver:EQUalizer:CONTrol? <lane>
```

<lane>,<value>=<NR1>

| | |
|---|---|
| <lane>=<NR1>: | Lane number |
| <value>=<NR1>: | Setting value of Equalizer (Control) |

**Response Data**

<value>

For the setting of Pre-Emphasis and VOD, refer to the following commands.
:SOURce:TRANsceiver:EMPHasis:FIRSt
:SOURce:TRANsceiver:EMPHasis:PRE
:SOURce:TRANsceiver:EMPHasis:SECond
:SOURce:TRANsceiver:VOD

## :SENSe:TRANsceiver:EQUalizer:DCGain

**Function**

This command sets the Equalizer (DC Gain) of the transceiver.
It also queries the settings of the Equalizer (DC Gain) of the transceiver.

**Syntax**

```
:SENSe:TRANsceiver:EQUalizer:DCGain <lane>,<value>
:SENSe:TRANsceiver:EQUalizer:DCGain? <lane>
```

<lane>,<value>=<NR1>

| | |
|---|---|
| <lane>=<NR1>: | Lane number |
| <value>=<NR1>: | Setting value of Equalizer (DC Gain) |

**Response Data**

<value>

## :SOURce:CFP:OPTical:OFF

**Function**

This command turns Off the CFP optical output.

**Syntax**

```
:SOURce:CFP:OPTical:OFF
```

It takes time until CFP optical output is turned off after the command is transmitted.
* Confirm the end of the processing with the *OPC command.

## :SOURce:CFP:OPTical:ON

**Function**

This command turns On the CFP optical output.

**Syntax**

```
:SOURce:CFP:OPTical:ON
```

It takes time until CFP optical output is turned on after the command is transmitted.
* Confirm the end of the processing with the *OPC command.

## :SOURce:CFP:OPTical:STATus

**Function**

This command queries the CFP optical output status.

**Syntax**

```
:SOURce:CFP:OPTical:STATus?
```

**Response Data**

| | |
|---|---|
| 0: | Optical output Off |
| 1: | Optical output On |

**Example of Use**

```
:SOURce:CLOCk INTERNAL
:SOURce:CLOCk?
> INTERNAL
```

## :SOURce:CLOCk

### Function

This command sets the clock source (Tx clock). It also queries the setting of the clock source (Tx clock).

### Syntax

```
:SOURce:CLOCk <CHAR>
:SOURce:CLOCk?
```

For <CHAR>, select any of the following.

| <CHAR> | Clock Source for Tx Clock |
|---|---|
| INTERNAL | Internal oscillator clock |
| INPUT_10MHZ | Clock input to 10 MHz Input terminal |
| INPUT_REFCLK | Clock input to x Ref Clock Input terminal |
| RECEIVE | Clock synchronized with Rx clock |
| INPUT_SYNC | Clock input to Sync Input terminal |

### Response Data

<char>

### Example of Use

```
:SOURce:CLOCk INTERNAL
:SOURce:CLOCk?
> INTERNAL
```

## :SOURce:CLOCk:FREQuency:OFFSet

**Function**

This command sets the offset of the Tx clock frequency. It also queries the setting of the offset of the Tx clock frequency.

**Syntax**

```
:SOURce:CLOCk:FREQuency:OFFSet <NR1>
:SOURce:CLOCk:FREQuency:OFFSet?
```

<NR1>:
Specify the offset (–120 to +120 ppm) from the reference clock frequency.

**Response Data**

< NR1>

**Example of Use**

To set the offset of the Tx clock frequency to 1 ppm:

```
:SOURce:CLOCk:FREQuency:OFFSet 1
:SOURce:CLOCk:FREQuency:OFFSet?
> 1
```

## :SOURce:CLOCk:OUTPut:DIVide

**Function**

This command sets and queries the clock division rate output to the Tx Ref Clock Output terminal.

**Syntax**

```
:SOURce:CLOCk:OUTPut:DIVide DIV16|DIV64
:SOURce:CLOCk:OUTPut:DIVide?
```

Set any of the following.
DIV16: 1/16
DIV64: 1/64

**Response Data**

DIV16|DIV64

**Example of Use**

```
:SOURce:CLOCk:OUTPut:DIVide DIV64
:SOURce:CLOCk:OUTPut:DIVide?
> DIV64
```

## :SOURce:CLOCk:OUTPut:M10

### Function

This command sets the 10 MHz signal output to the 10 MHz Output terminal. It also queries the setting of the 10 MHz signal output to the 10 MHz Output terminal.

### Syntax

```
:SOURce:CLOCk:OUTPut:M10 INTERNAL|LOCKED
:SOURce:CLOCk:OUTPut:M10?
```

Set any of the following.

| | |
|---|---|
| INTERNAL: | 10 MHz of internal oscillator |
| LOCKED: | 10 MHz synchronized with Rx clock |

### Response Data

INTERNAL|LOCKED

### Example of Use

```
:SOURce:CLOCk:OUTPut:M10 INTERNAL
:SOURce:CLOCk:OUTPut:M10?
> INTERNAL
```

## :SOURce:CLOCk:PAYLoad:OFFSet[:L]

**Function**

This command sets and queries the payload offset for the OTU3 or OTU4 application.
To set and query the low order payload offset, describe :L.

**Syntax**

```
:SOURce:CLOCk:PAYLoad:OFFSet[:L] <NR1>
:SOURce:CLOCk:PAYLoad:OFFSet[:L]?
```

<NR1>: Payload offset
When the low order does not exist: (–120 to +120ppm)
When the high order/low order exists:
High order: (–40 to +40ppm), low order: (–120 to +120ppm)

**Response Data**

< NR1>

**Example of Use**

To set the payload offset to 10ppm:
For the ODTU4.1 or ODTU4.8, set the high order payload offset.

```
:SOURce:CLOCk:PAYLoad:OFFSet 10
```

To set the low order payload offset to –50ppm:

```
:SOURce:CLOCk:PAYLoad:OFFSet:L -50
```

## :SOURce:EALarm:BIT

**Function**

This command sets and queries the bit adding errors/alarms.

**Syntax**

```
:SOURce:EALarm:BIT <BINARY>
:SOURce:EALarm:BIT?
```

<BINARY>
Specifies bit for adding errors/alarms in binary as 1 and for not adding errors/alarms as 0 (in order from header as bit 0).
For example, specify 8 bits for CRC8 Error, and 5 bits for CRC5 Error.
When a smaller number of bits than the number of lanes is specified, the remaining bits are padded with 0s. When more bits than bits are specified, the surplus bits are ignored.
However, for Invalid JC1&JC2, bits are set in order of Bit 0 to 7 of JC2 and then Bit 0 to 7 of JC1.

**Response Data**

<BINARY>
The number of digits depends on the number of bits that can be set.

**Example of Use**

To insert errors/alarms to Bit0 and Bit3 for CRC8 Error:

```
:SOURce:EALarm:BIT #B10010000
```

Or

```
:SOURce:EALarm:BIT #B1001
```

To insert errors to Bit0, 1, and 2 of JC2 and Bit0 of JC1 for Invalid JC1&JC2:

```
:SOURce:EALarm:BIT #B1110000010000000
```

## :SOURce:EALarm:FAS:EXCLude

### Function

This command sets and queries whether to exclude FAS of OTN frame from the error insertion range.

### Syntax

```
:SOURce:EALarm:FAS:EXCLude 0|1|OFF|ON
:SOURce:EALarm:FAS:EXCLude?
```

| | |
|---|---|
| 0\|OFF: | FAS Exclude Off |
| 1\|ON: | FAS Exclude On |

### Response Data

0|1

### Example of Use

```
:SOURce:EALarm:FAS:EXCLude ON
:SOURce:EALarm:FAS:EXCLude?
> 1
```

## :SOURce:EALarm:LANE

**Function**

This command sets and queries the lane adding errors/alarms.

**Syntax**

```
:SOURce:EALarm:LANE <BINARY>
:SOURce:EALarm:LANE?
```

<BINARY>

Specifies lane for adding errors/alarms in binary as 1 and for not adding errors/alarms as 0 (in order from header as Lane 0, Lane 1, ...)
For example, specify 20 bits (Lane 0~19) for 100GbE, and 4 bits (Lane 0~3) for 40GbE. When a smaller number of bits than the number of lanes is specified, the remaining bits are padded with 0s. When more bits than lanes are specified, the surplus bits are ignored.

**Response Data**

<BINARY>

**Example of Use**

To add errors/alarms to Lane 0, 3, 9 at 100GbE:

```
:SOURce:EALarm:LANE #B10010000010000000000
```

Or

```
:SOURce:EALarm:LANE #B1001000001
```

## :SOURce:EALarm:STARt[:AUNit]

**Function**

This command starts error/alarm addition.
When error/alarm addition is already running, it restarts.
Adding :AUNit starts simultaneous error/alarm addition for all connected units.

**Syntax**

```
:SOURce:EALarm:STARt[:AUNit]
```

**Example of Use**

To start error/alarm addition:

1. Clear the start event by sending :SOURce:EALarm:STARt:EVENt?.
2. Start the error/alarm addition by sending :SOURce:EALarm:STARt .
3. Send repeatedly until the :SOURce:EALarm:STARt:EVENt? response becomes 1 (started).

## :SOURce:EALarm:STARt:EVENt[:AUNit]

**Function**

This command queries start events added to errors/alarms.

**Syntax**

```
:SOURce:EALarm:STARt:EVENt[:AUNit]?
```

**Response Data**

**0|1**

| | |
|---|---|
| 0: | Before start of error/alarm insertion<br>The response data becomes 0 after this command is read. |
| 1: | After error/alarm insertion started<br>When :AUNit is added to the header, becomes 1 when error/alarm insertion starts at all units |

The event value is cleared to 0 when response data 1 is read.

**Example of Use**

Refer to the example for :SOURce:EALarm:STARt[:AUNit] .

## :SOURce:EALarm:STATus[:AUNit]

**Function**

This command queries operation state added to errors/alarms.

**Syntax**

```
:SOURce:EALarm:STATus[:AUNit]?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Error/alarm addition operation stopped |
| 1: | Error/alarm addition operation running<br>When :AUNit is added to the header, becomes 1 when error/alarm insertion starts at any of units and becomes 0 when all units stopped. |

**Example of Use**

Refer to the example for :SOURce:EALarm:STOP[:AUNit] .

## :SOURce:EALarm:STOP[:AUNit]

**Function**

This command stops error/alarm addition.
Adding :AUNit stops error/alarm addition for all connected units.

**Syntax**

```
:SOURce:EALarm:STOP[:AUNit]
```

**Example of Use**

To stop error/alarm addition

1. Send :SOURce:EALarm:STOP.
   Error/alarm addition stops.
2. Send :SOURce:EALarm:STATus?
   Sending is repeated until the response data is 0 (stopped).

## :SOURce:EALarm:SUBRow

**Function**

This command sets and queries SubRow added to errors/alarms.

**Syntax**

```
:SOURce:EALarm:SUBRow <NR1>
:SOURce:EALarm:SUBRow?
```

| | |
|---|---|
| <NR1>: | Number for SubRow |

**Response Data**

<NR1>

**Example of Use**

```
:SOURce:EALarm:SUBRow 1
:SOURce:EALarm:SUBRow?
> 1
```

## :SOURce:EALarm:TIMing:BURSt

**Function**

This command sets the count when the error/alarm addition timing is Burst. It also queries the count setting when the error/alarm addition timing is Burst.

**Syntax**

```
:SOURce:EALarm:TIMing:BURSt <NR1>
:SOURce:EALarm:TIMing:BURSt?
```

<NR1>
Specifies error/alarm addition count

**Response Data**

<NR1>

**Example of Use**

```
:SOURce:EALarm:TYPE INV_SH00
:SOURce:EALarm:TIMing:TYPE BURST
:SOURce:EALarm:TIMing:BURSt 1
:SOURce:EALarm:TIMing:BURSt?
> 1
```

## :SOURce:EALarm:TIMing:ERRor

**Function**

This command sets the error addition count when the error/alarm addition timing is Alternate. It also queries the settings of the error addition count when the error/alarm addition timing is Alternate.

**Syntax**

```
:SOURce:EALarm:TIMing:ERRor <NR1>
:SOURce:EALarm:TIMing:ERRor?
```

<NR1>:
Frame count of error/alarm addition (Error)

**Response Data**

<NR1>

**Example of Use**

Refer to the example for :SOURce:EALarm:TIMing:NORMal.

## :SOURce:EALarm:TIMing:NORMal

**Function**

This command sets the normal frame count when the error/alarm addition timing is Alternate. It also queries the settings of the normal frame count when the error/alarm addition timing is Alternate.

**Syntax**

```
:SOURce:EALarm:TIMing:NORMal <NR1>
:SOURce:EALarm:TIMing:NORMal?
```

<NR1>:
Frame count of no error/alarm (Normal)

**Response Data**

<NR1>

**Example of Use**

```
:SOURce:EALarm:TYPE FAS_MLD
:SOURce:EALarm:TIMing:TYPE ALTERNATE
:SOURce:EALarm:TIMing:ERRor 1
:SOURce:EALarm:TIMing:NORMal 2
:SOURce:EALarm:TIMing:ERRor?
> 1
:SOURce:EALarm:TIMing:NORMal?
> 2
```

## :SOURce:EALarm:TIMing:RATE

### Function

This command sets the error/alarm appended rate when selecting Rate using :SOURce:EALarm:TIMing:TYPE.
It also queries the appended error/alarm rate.

### Syntax

```
:SOURce:EALarm:TIMing:RATE <CHAR>|<NR3>
:SOURce:EALarm:TIMing:RATE?
```

<NR3>:Sets the appended rate in numeric values.
When $3.6\times10^{-6}$ is set, 3.6E–6 is set.

<CHAR>

Set the appended rate.

| <CHAR> | Appended Rate |
|---|---|
| R1E_2 | 1.0E–2 |
| R1E_3 | 1.0E–3 |
| R1E_4 | 1.0E–4 |
| R1E_5 | 1.0E–5 |
| R1E_6 | 1.0E–6 |
| R1E_7 | 1.0E–7 |
| R1E_8 | 1.0E–8 |
| R1E_9 | 1.0E–9 |

### Response Data

<NR3>

### Example of Use

```
:SOURce:EALarm:TYPE BIT_ERROR
:SOURce:EALarm:TIMing:TYPE CRATE
:SOURce:EALarm:TIMing:RATE R1E_6
:SOURce:EALarm:TIMing:RATE?
> 1.0E-6
```

## :SOURce:EALarm:TIMing:TYPE

**Function**

This command sets the error/alarm appended timing. It also queries the setting of the error/alarm appended timing.

**Syntax**

```
:SOURce:EALarm:TIMing:TYPE <CHAR>
:SOURce:EALarm:TIMing:TYPE?
```

<CHAR>

Select the timing from following.

| <CHAR> | Timing |
|---|---|
| SINGLE | Single |
| BURST | Burst |
| RRATE | Rate |
| ALTERNATE | Alternate |
| ALL | All |

RATE is specified at Bit Error of Ethernet.
When selecting BURST, specify the error/alarm appended volume using :SOURce:EALarm:TIMing:BURSt.
When selecting RRATE, specify the error/alarm appended rate using:SOURce:EALarm:TIMing:RATE.
When selecting ALTERNATE, specify the error/alarm appended volume using :SOURce:EALarm:TIMing:ERRor
and :SOURce:EALarm:TIMing:NORMal.

The selectable value varies with the selection of the application and error/alarm appended types.
When changing the error/alarm appended types, the values are rounded to the selectable value. For details, refer to the MD1260A 40/100G Ethernet Analyzer Operation Manual.

**Response Data**

<CHAR>

**Example of Use**

To set the error/alarm appended timing to Single:

```
:SOURce:EALarm:TIMing:TYPE SINGLE
:SOURce:EALarm:TIMing:TYPE?
> SINGLE
```

## :SOURce:EALarm:TYPE

**Function**

This command sets the error/alarm appended types.
It also queries the setting of the error/alarm appended types.

**Syntax**

```
:SOURce:EALarm:TYPE <CHAR>
:SOURce:EALarm:TYPE?
```

<CHAR>: Select the error alarm types from the following.
When changing the types, the timing settings are initialized.

40GbE, 100GbE:

| Error/Alarm Type | <CHAR> |
|---|---|
| Invalid Sync Header (00) | INV_SH00 |
| Invalid Sync Header (11) | INV_SH11 |
| Invalid Alignment Marker | INV_MARKER |
| BIP Error | BIP |
| Invalid Block Type (0x00) | INV_BT00 |
| Invalid Block Type (0x2d) | INV_BT2D |
| Invalid Block Type (0x33) | INV_BT33 |
| Invalid Block Type (0x66) | INV_BT66 |
| Hi-BER | HIBER |
| LF | LF |
| RF | RF |
| Bit Errors | BIT_ERROR |

OTN3, OTN4:

| Layer | Error Type | <CHAR> |
|---|---|---|
| LLD | FAS | FAS_LLD |
| OTU3/ OTU4 | FAS | FAS |
| | SM-BIP8 | SM_BIP8 |
| | SM-BEI | SM_BEI |
| | Uncorrectable Error | BIT_ALL_U |
| | Correctable Error | BIT_ALL_C |
| | Bit All | BIT_ALL |

OTN3, OTN4 (Cont'd):

| Layer | Error Type | <CHAR> |
|---|---|---|
| ODU3/ ODU4 | PM-BIP8 | PM_BIP8 |
| | PM-BEI | PM_BEI |
| | TCMi-BIP8 | TCM{1\|2\|3\|4\|5\|6}_BIP8 |
| | TCMi -BEI | TCM{1\|2\|3\|4\|5\|6}_BEI |
| ODU2e/ ODU0 | FAS | FAS_L |
| | PM-BIP8 | PM_BIP8_L |
| | PM-BEI | PM_BEI_L |
| GMP | CRC8 Error | GMP_CRC8 |
| | CRC5 Error | GMP_CRC5 |
| | Invalid JC1 | GMP_JC1 |
| | Invalid JC2 | GMP_JC2 |
| | Invalid JC1&JC2 | GMP_JC12 |
| GMP(Lo) | CRC8 Error | GMP_CRC8_L |
| | Invalid JC1 | GMP_JC1_L |
| | Invalid JC2 | GMP_JC2_L |
| | Invalid JC1&JC2 | GMP_JC12_L |
| GFP-T | Superblock CRC Error | GFP_SCRC |
| | cHEC Error | GFP_CHEC |
| | tHEC Error | GFP_THEC |
| Ethernet | Invalid Sync Header (00) | INV_SH00 |
| | Invalid Sync Header (11) | INV_SH11 |
| | Invalid Block Type (0x00) | INV_BT00 |
| | Invalid Block Type (0x2d) | INV_BT2D |
| | Invalid Block Type (0x33) | INV_BT33 |
| | Invalid Block Type (0x66) | INV_BT66 |
| | Invalid Alignment Marker | INV_MARKER |
| | BIP Error | BIP |
| | LF | LF |
| | RF | RF |
| | 66B Error | ERR_66B |
| | 10B Error | ERR_10B |
| Test Pattern | Bit Errors | BIT_ERROR |

OTN3, OTN4 (Cont'd):

| Layer | Alarm Type | <CHAR> |
|---|---|---|
| LLD | OOF/LOF | OOF_LLD |
| | OOR/LOR | OOR |
| OTU3/ OTU4 | OOF/LOF | OOF |
| | OOM/LOM | OOM |
| | SM-TIM | SM_TIM |
| | SM-BIAE | SM_BIAE |
| | SM-BDI | SM_BDI |
| | SM-IAE | SM_IAE |
| ODU3/ ODU4 | ODU-AIS | AIS_ODU |
| | ODU-OCI | OCI |
| | ODU-LCK | LCK |
| | PM-TIM | PM_TIM |
| | PM-BDI | PM_BDI |
| | TCMi-TIM | TCM{1\|2\|3\|4\|5\|6}_TIM |
| | TCMi-BIAE | TCM{1\|2\|3\|4\|5\|6}_BIAE |
| | TCMi IAE | TCM{1\|2\|3\|4\|5\|6}_IAE |
| | TCMi BDI | TCM{1\|2\|3\|4\|5\|6}_BDI |
| | TCMi-LTC | TCM{1\|2\|3\|4\|5\|6}_LTC |
| OPU3/ OPU4 | Client-AIS | AIS_C |
| | CSF | CSF |
| ODU2e/ ODU0 | OOF | OOF_L |
| | OOM | OOM_L |
| | ODU-AIS | AIS_ODU_L |
| | ODU-OCI | OCI_L |
| | ODU-LCK | LCK_L |
| | PM-TIM | PM_TIM_L |
| | PM-BDI | PM_BDI_L |
| Ethernet | High BER | HIBER |

40GbE No Frame: 100GbE No Frame,
OTN3 No Frame: OTN4 No Frame:

| Error/Alarm Types | <CHAR> |
|---|---|
| Bit Errors | BIT_ERROR |

Set errors/alarms for which addition timing can be selected as :SOURce:EALarm:TIMing:TYPE
In addition, when error/alarm insertion into lanes has been specified, specify the insertion lanes as :SOURce:EALarm:LANE.

For details of each error/alarm, refer to the MD1260A 40/100G Ethernet Analyzer Operation Manual.

**Response Data**

<CHAR>

**Example of Use**

```
:SOURce:EALarm:TYPE?
> BIT_ERROR
```

## :SOURce:GFP:CSF:REPLacement

**Function**

This command sets and queries the data to replace GFP-T errors.

**Syntax**

```
:SOURce:GFP:CSF:REPLacement BLOCK|CSF
:SOURce:GFP:CSF:REPLacement?
```

Set any of the following.
BLOCK: Ethernet Block
CSF: GFP-T CSF

**Response Data**

BLOCK|CSF

**Example of Use**

```
:SOURce:GFP:CSF:REPLacement BLOCK
:SOURce:GFP:CSF:REPLacement?
> BLOCK
```

## :SOURce:GFP:{PTI|UPI}

**Function**

This command sets and queries the payload of GFP-T.

**Syntax**

```
:SOURce:GFP:{PTI|UPI} <BINARY>
:SOURce:GFP:{PTI|UPI}?
```

| | |
|---|---|
| <BINARY>: | For PTI: 3 bits |
| | For UPI: 8 bits |

**Response Data**

<BINARY>

**Example of Use**

```
:SOURce:GFP:PTI #B000
:SOURce:GFP:PTI?
> #B001
:SOURce:GFP:UPI #B00000110
:SOURce:GFP:UPI?
> #B00000110
```

## :SOURce:MAPPing

**Function**

This command sets the number of physical lanes when the application is 100GbE No Frame or OTU4 No Frame.
It queries the type of the application running.
The application is set with :SYSTem:CONFig.

**Syntax**

```
:SOURce:MAPPing <CHAR>
:SOURce:MAPPing?
```

<CHAR>
Set any of the following.

| Application (Number of Lanes) | <CHAR> |
|---|---|
| 100GbE No Frame (10 Lanes) | E100G_N10 |
| 100GbE No Frame (20 Lanes) | E100G_N20 |
| OTU4 No Frame (10 Lanes) | OTU4_N10 |
| OTU4 No Frame (20 Lanes) | OTU4_N20 |

**Response Data**

<CHAR>

| Application/Mapping | <CHAR> |
|---|---|
| 40GbE | E40G |
| 100GbE | E100G |
| ODU3-PRBS | OTU3 |
| ODU4-PRBS | OTU4 |
| ODU4-100GbE | OTU4_E |
| ODTU4.8-ODU2e-PRBS | OTU4_ODU2E |
| ODTU4.8-ODU2e-10GbE | OTU4_ODU2E_E |
| ODTU4.1-ODU0-PRBS | OTU4_ODU0 |
| ODTU4.1-ODU0-GbE | OTU4_ODU0_E |
| 40GbE No Frame | E40G_N |
| 100GbE No Frame (10Lane) | E100G_N10 |
| 100GbE No Frame (20Lane) | E100G_N20 |
| OTU3 No Frame | OTU3_N |
| OTU4 No Frame (10Lane) | OTU4_N10 |
| OTU4 No Frame (20Lane) | OTU4_N20 |

**Example of Use**

When the loading application is 100GbE:

```
:SOURce:MAPPing?
> E100G
```

When the loading application is 100GbE No Frame and measurement of 10 Lanes is performed:

```
:SOURce:MAPPing E100G_N10
:SOURce:MAPPing?
> 100GE_N10
```

## :SOURce:MAPPing:LANE

### Function

This command sets the number of lanes for the Tx Lane lane marker in sequence from Tx Lane 0.
When this remote command is sent, the Allow to Overlap button on the screen is set to On.
It also queries the value of the lane marker for the Tx Lane.

### Syntax

```
:SOURce:MAPPing:LANE <NR1>,<NR1>,<NR1>,…
:SOURce:MAPPing:LANE?
```

<NR1>
Lane marker value for Tx Lane

### Response Data

<NR1>,<NR1>,<NR1>,..

### Example of Use

For 100GbE/OTU4

```
:SOURce:MAPPing:LANE
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19
```

For 40GbE/OTU3

```
:SOURce:MAPPing:LANE 0,1,2,3
```

## :SOURce:ODTU:DUMMy:PATTern

### Function

This command sets and queries the data to be sent to Dummy channel.

### Syntax

```
:SOURce:ODTU:DUMMy:PATTern COPY_MAIN|PRBS11
:SOURce:ODTU:DUMMy:PATTern?
```

Set any of the following.
COPY_MAIN: Same data as Main channel
PRBS11: Pseudorandom bit sequence

### Response Data

COPY_MAIN|PRBS11

### Example of Use

```
:SOURce:ODTU:DUMMy:PATTern COPY_MAIN
:SOURce:ODTU:DUMMy:PATTern?
> COPY_MAIN
```

## :SOURce:ODTU:MAIN:TP

### Function

This command sets and queries the Main TP (Tributary Port).

### Syntax

```
:SOURce:ODTU:MAIN:TP <NR1>
:SOURce:ODTU:MAIN:TP?
```

The setting range of <NR1> is as follows.

| Mapping | <NR1> |
|---|---|
| ODTU4.1-ODU0-GbE/PRBS | 1 to 80 |
| ODTU4.8-ODU2e-10GbE/PRBS | 1 to 10 |

### Response Data

<NR1>

### Example of Use

```
:SOURce:ODTU:MAIN:TP 16
:SOURce:ODTU:MAIN:TP?
> 16
```

## :SOURce:ODTU:MAIN:TS

**Function**

This command sets and queries the TS (Tributary Slot) that stores Main TP.

One TS for ODTU4.1 and 8 TSs for ODTU4.8 are set.

**Syntax**

```
:SOURce:ODTU:MAIN:TS
<NR1>|<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>
:SOURce:ODTU:MAIN:TS?
```

| | |
|---|---|
| <NR1>: | 1 to 80 |

**Response Data**

<NR1>|<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>

**Example of Use**

For ODTU4.1:

```
:SOURce:ODTU:MAIN:TS 2
:SOURce:ODTU:MAIN:TS?
> 2
```

For ODTU4.8:

```
:SOURce:ODTU:MAIN:TS 33,34,35,36,37,38,39,40
:SOURce:ODTU:MAIN:TS?
> 33,34,35,36,37,38,39,40
```

## :SOURce:OTN:FEC

**Function**

This command sets and queries the GFEC Encode of OTN frame.

**Syntax**

```
:SOURce:OTN:FEC 0|1|OFF|ON
:SOURce:OTN:FEC?
```

| | |
|---|---|
| 0\|OFF: | GFEC Encode Off |
| 1\|ON: | GFEC Encode On |

**Response Data**

0|1

**Example of Use**

```
:SOURce:OTN:FEC OFF
:SOURce:OTN:FEC?
> 0
```

## :SOURce:OTN:OH[:L]

**Function**

This command sets and queries the overhead value of OTN frame.
Omitting :L sets and queries the high order (OTU4) OH.
Adding :L sets and queries the low order (ODU2e or ODU0) OH.

**Syntax**

```
:SOURce:OTN:OH[:L] <row>,<column>,<data>
:SOURce:OTN:OH[:L]? <row>,<column>
```

| | |
|---|---|
| <row>=<NR1>: | Row number 1 to 4 |
| <column>=<NR1>: | Column number 1 to 16 |
| <data>=<HEX>: | Value #H00 to #HFF |

**Response Data**

<data>=<HEX>

**Example of Use**

```
:SOURce:OTN:OH:L 3,2,#H10
:SOURce:OTN:OH:L? 3,2
> #H10
```

## :SOURce:SKEW:BIT

**Function**

Ths command sets the skew amount. It also queries the skew amount.

**Syntax**

```
:SOURce:SKEW:BIT <NR1>
:SOURce:SKEW:BIT?
```

| | |
|---|---|
| <NR1>: | Skew setting in bit units |

**Response Data**

<NR1>

**Example of Use**

Refer to the example for :SOURce:SKEW:NS.

## :SOURce:SKEW:LANE

**Function**

This command sets the lane for inserting skew.
It also queries the setting lane for inserting skew.

**Syntax**

```
:SOURce:SKEW:LANE <BINARY>
:SOURce:SKEW:LANE?
```

<BINARY>
Specifies lane for adding skew in binary as 1 and for not adding skew as 0 (in order from header as Lane 0, Lane 1, ...)
For example, specify 20 bits (Lane 0~19) for 100GbE, and 4 bits (Lane 0~3) for 40GbE. When a smaller number of bits than the number of lanes is specified, the remaining bits are padded with 0s. When more bits than lanes are specified, the surplus bits are ignored.

**Response Data**

<BINARY>

**Example of Use**

To add skew to Tx Lane 0, 3, 9 at 100GbE/OTU4:

```
:SOURce:SKEW:LANE #B10010000010000000000
```

Or

```
:SOURce:SKEW:LANE #B1001000001
:SOURce:SKEW:LANE?
```

```
> #B10010000010000000000
```

Example:
To add skew Physical Lane 9 at 100GbE/OTU4

```
:SOURce:SKEW:LANE #B0000000001
```

***Note:***

When the :SOURce:SKEW:TYPE setting is changed, the setting of this command is initialized.

## :SOURce:SKEW:NS

**Function**

This command queries the skew time.

**Syntax**

```
:SOURce:SKEW:NS?
```

**Response Data**

| | |
|---|---|
| <NR2>: | Skew setting converted to ns units |

**Example of Use**

To generate skew of 66 bits (12.8 ns) at 100GbE:

```
:SOURce:SKEW:BIT 66
:SOURce:SKEW:NS?
>12.800
```

## :SOURce:SKEW:TYPE

**Function**

This command sets the skew insertion lane. It also queries the skew insertion lane.

**Syntax**

```
:SOURce:SKEW:TYPE PHY_LANE|TX_LANE
:SOURce:SKEW:TYPE?
```

Select any of the following.
TX_LANE: Tx Lane
PHY_LANE: Physical Lane

:SOURce:SKEW:LANE sets the skew generation lane and :SOURce:SKEW:BIT sets the bit count for skew generation. Query the time-converted skew generation using :SOURce:SKEW:NS?

**Response Data**

PHY_LANE|TX_LANE

**Example of Use**

To insert skew to Tx Lane:

```
:SOURce:SKEW:TYPE TX_LANE
:SOURce:SKEW:TYPE?
> TX_LANE
```

## :SOURce:STReam:BURSt:CONTrol:VALue

**Function**

This command sets the interburst gap. It also queries the interburst gap setting.

**Syntax**

```
:SOURce:STReam:BURSt:CONTrol:VALue <NR1>
:SOURce:STReam:BURSt:CONTrol:VALue?
```

| | |
|---|---|
| <NR1>: | Gap size (frames) |

**Response Data**

<NR1>

## :SOURce:STReam:BURSt:ENABle

**Function**

This command sets whether or not to send the burst. It also queries the burst send setting.

**Syntax**

```
:SOURce:STReam:BURSt:ENABle 0|1|OFF|ON
:SOURce:STReam:BURSt:ENABle?
```

| | |
|---|---|
| 0\|OFF: | Does not send burst |
| 1\|ON: | Sends burst |

**Response Data**

0|1

## :SOURce:STReam:BURSt:SIZE

**Function**

This command sets the burst size (frame count sent at one burst).
It also queries the burst size.

**Syntax**

```
:SOURce:STReam:BURSt:SIZE <NR1>
:SOURce:STReam:BURSt:SIZE?
```

| | |
|---|---|
| <NR1>: | Frame count |

**Response Data**

<NR1>

## :SOURce:STReam:CONTrol:RANGe

**Function**

This command sets the transmission gap for the Ethernet frame. It also queries the range of the transmission gap of the Ethernet frame.

**Syntax**

```
:SOURce:STReam:CONTrol:RANGe <value1>,<value2>
:SOURce:STReam:CONTrol:RANGe?
```

<value1>, <value2> = <NR1>
Sets Min. and Max values of gap size in bytes. The smaller of the two values is the Min. and the larger is the Max. T

**Response Data**

<value1>, <value2>= <NR1>

| | |
|---|---|
| <value1>: | Min. value of gap size |
| <value2>: | Max value of gap size |

**Example of Use**

To send the gap size in the range of 12 to 96 bytes:

```
:SOURce:STReam:CONTrol:TYPE RANDOM
:SOURce:STReam:CONTrol:RANGe 12,96
```

## :SOURce:STReam:CONTrol:TYPE

### Function

This command sets the Ethernet frame gap to a random or fixed value. It also queries the Ethernet frame gap.

### Syntax

```
:SOURce:STReam:CONTrol:TYPE FIXED|RANDOM
:SOURce:STReam:CONTrol:TYPE?
```

Select the setting type from following.

| | |
|---|---|
| FIXED | Fixed |
| RANDOM | Random |

When selecting FIXED, specify the gap size using :SOURce:STReam:CONTrol:VALue.
When selecting RANDOM, specify the range of the gap size using :SOURce:STReam:CONTrol:RANGe.

Specification of the sending rate as % or bits is set using a command. The gap size corresponding to the required send rate can be confirmed by switching the units of the Stream Control setting at the Test Pattern setting screen of the operation screen.

### Response Data

FIXED|RANDOM

### Example of Use

To send the gap size as 12 bytes (100%):

```
:SOURce:STReam:CONTrol:TYPE FIXED
:SOURce:STReam:CONTrol:VALue 12
```

## :SOURce:STReam:CONTrol:VALue

**Function**

This command sets the Ethernet frame transmission span (gap size). It also queries the setting of the transmission span.

**Syntax**

```
:SOURce:STReam:CONTrol:VALue <NR2>
:SOURce:STReam:CONTrol:VALue?
```

| | |
|---|---|
| <NR2>: | Gap size (bytes) |

**Response Data**

<NR2>

**Example of Use**

To send the gap size as 12 bytes (100%):

```
:SOURce:STReam:CONTrol:TYPE FIXED
:SOURce:STReam:CONTrol:VALue 12
```

## :SOURce:STReam:COUNt

**Function**

This command sets the frame or burst count during the stream.
It also queries the frame or burst count during the stream.

**Syntax**

```
:SOURce:STReam:COUNt <NR1>
:SOURce:STReam:COUNt?
```

| | |
|---|---|
| <NR1>: | Frame or burst count |

**Response Data**

<NR1>

## :SOURce:STReam:DURation:FRAMes

**Function**

This command sets and queries the number of frames of Test Pattern. It executes the same processing as :SOURce:STReam:DURation:REPeat:COUNt.

**Syntax**

```
:SOURce:STReam:DURation:FRAMes <NR1>
:SOURce:STReam:DURation:FRAMes?
```

| | |
|---|---|
| <NR1>: | Number of frames |

**Response Data**

<NR1>

**Example of Use**

Refer to the example for :SOURce:STReam:DURation:TYPE.

## :SOURce:STReam:DURation:REPeat:COUNt

**Function**

This command sets the repetition rate when the Test Pattern send cycle is Repeat. It also queries the repetition rate.
It executes the same processing as :SOURce:STReam:DURation:FRAMes.

**Syntax**

```
:SOURce:STReam:DURation:REPeat:COUNt <NR1>
:SOURce:STReam:DURation:REPeat:COUNt?
```

| | |
|---|---|
| <NR1>: | Send repeat count |

**Response Data**

<NR1>

**Example of Use**

Refer to the example for :SOURce:STReam:DURation:TYPE.

## :SOURce:STReam:DURation:TYPE

### Function

This command sets and queries the repetition method of the stream transmission.

### Syntax

```
:SOURce:STReam:DURation:TYPE CONT|FRAME|REPEAT
:SOURce:STReam:DURation:TYPE?
```

Select the setting type from the following.
Time cannot be specified by the remote command.

| | |
|---|---|
| CONT: | Continuous |
| FRAME: | Frame |
| REPEAT: | Repeat (Frames) |

Sets the stream cycle by :SOURce:STReam:DURation:REPeat:COUNt when setting REPEAT.
Sets the number of frames by :SOURce:STReam:DURation:FRAMes when setting REPEAT.

### Response Data

CONT|FRAME |REPEAT
When FRAME or REPEAT is set, the query response is as follows depending on the application.

100GbE, 40GbE: REPEAT
OTU4, OTU3: FRAME

When Time is set to Duration on the screen, the response data is REPEAT. In this case, the conversion factor for repeat count can be queried with :SOURce:STReam:DURation:REPeat:COUNt
or :SOURce:STReam:DURation:FRAMes.

### Example of Use

To send 1000 cycle stream:

```
:SOURce:STReam:DURation:TYPE REPEAT
:SOURce:STReam:DURation:REPeat:COUNt 1000
```

## :SOURce:STReam:ENABle

**Function**

This command sets sending for each stream. It also queries the sending setting for each stream.

**Syntax**

```
:SOURce:STReam:ENABle <NR1>[,<NR1>,...]
:SOURce:STReam:ENABle?
```

<NR1>[,<NR1>,...]
Sets number of enabled streams to 1 or more

**Response Data**

<NR1>[,<NR1>,...]

**Example of Use**

To send stream 5, 12, and 15:

```
:SOURce:STReam:ENABle 5,12,15
:SOURce:STReam:ENABle?
> 5,12,15
```

## :SOURce:STReam:ERRor:TYPE

**Function**

This command sets and queries the error types appended to the Ethernet frame.
The errors are added to the stream specified by:SOURce:STReam:ID.

**Syntax**

```
:SOURce:STReam:ERRor:TYPE FCS|OFF
:SOURce:STReam:ERRor:TYPE?
```

Selects error type from the following:

| | |
|---|---|
| OFF | None |
| FCS | FCS Error |

**Response Data**

FCS|OFF

**Example of Use**

To add the FCS error to all sent frames:

```
:SOURce:STReam:ERRor:TYPE FCS
:SOURce:STReam:ERRor:TYPE?
>FCS
```

## :SOURce:STReam:FSIZe

**Function**

This command queries the frame size of the sent Ethernet frame.

**Syntax**

```
:SOURce:STReam:FSIZe?
```

**Response Data**

<NR1>[,<NR1>]

<NR1>:

Ethernet frame size in byte units

When the frame size type is set to RANDOM, the Min. and Max. values of the transmitted frame size are returned .

## :SOURce:STReam:FSIZe:RANGe

**Function**

This command sets the sent Ethernet frame size range. It also queries the size range.

**Syntax**

```
:SOURce:STReam:FSIZe:RANGe <value1>,<value2>
:SOURce:STReam:FSIZe:RANGe?
```

<value1>, <value2> = <NR1>

Sets Min. and Max values of frame size in bytes, when selecting RANDOM. The smaller of the two values is the Min. and the larger is the Max.

**Response Data**

<value1>,<value2> = <NR1>

| | |
|---|---|
| <value1>: | Min. value of frame size |
| <value2>: | Max. value of frame size |

### Example of Use

To change the frame size range randomly from 64 to 1500 bytes:
:SOURce:STReam:FSIZe:TYPE RANDOM
```
:SOURce:STReam:FSIZe:RANGe 64,1500
```

## :SOURce:STReam:FSIZe:TYPE

### Function

This command sets whether or not to vary the Ethernet frame size length. It also queries whether the frame size length setting is fixed or variable.

### Syntax

```
:SOURce:STReam:FSIZe:TYPE FIXED|RANDOM
:SOURce:STReam:FSIZe:TYPE?
```

Select the setting type from the following.
FIXED
RANDOM

### Response Data

FIXED|RANDOM

When selecting FIXED, specify the frame size by :SOURce:STReam:FSIZe:VALue.

When selecting RANDOM, specify the range of the frame size sent by :SOURce:STReam:FSIZe:RANGe.

### Example of Use

Refer to the example for :SOURce:STReam:FSIZe:RANGe.

## :SOURce:STReam:FSIZe:VALue

**Function**

When the frame size is set to Fixed, the frame size of the Ethernet frame to be sent is set.
This command queries the frame size of the Ethernet frame.
When the pattern setting length*1 exceeds the Frame Size setting*2, frames are sent at the pattern setting length. The actually sent frame size (Min.) is returned at this time.

*1: Pattern setting length: Header Pattern length + Test Frame Min. length (18 at On) + FC length (4)

*2 The Frame Size setting is compared with the Min. setting at Random.

**Syntax**

```
:SOURce:STReam:FSIZe:VALue <NR1>
:SOURce:STReam:FSIZe:VALue?
```

| | |
|---|---|
| <NR1>: | Frame size when selecting FIXED (in byte unit) |

The actually sent frame size can be confirmed using :SOURce:STReam:FSIZe.

**Response Data**

<NR1>

**Example of Use**

To send a 512-byte frame:

```
:SOURce:STReam:FSIZe:TYPE FIXED
:SOURce:STReam:FSIZe:VALue 512
```

## :SOURce:STReam:HEADer

**Function**

This command sets the stream header format and value, and initializes the setting status of the command below.
:SOURce:STReam:HEADer:VARiable{3|4|5}
It queries the stream header format.

**Syntax**

```
:SOURce:STReam:HEADer <format>,<header>
:SOURce:STReam:HEADer?
```

| | |
|---|---|
| <format>=<STRING>: | Header format |
| <header>=<HEX>: | Header value |

Describe comma-delimited strings in <format> in the order of the following table.
The error (-220) will be generated if the order of group numbers is not in the ascending order or strings with the same order exist.

Set the header pattern (up to 75 bytes) in hexadecimal.
12 bytes of the MAC address (source address and destination address) and 63 bytes of the pattern including EtherType are included in 75 bytes.

**Table 3.3.2-4 String to Specify Header Format**

| Order | <format> | Header |
|---|---|---|
| 1 | NONE | Ethernet only [*1] |
| | CUSTOM | Custom Header [*1] |
| 2 | MPLS_TP{1\|2\|3\|4\|5} | With MPLS-TP Control Word [*2] |
| | MPLS_TP_NOCW{1\|2\|3\|4\|5} | Without MPLS-TP Control Word [*2] |
| 3 | B_TAG | PBB (B-TAG+I-TAG) |
| | I_TAG | PBB (I-TAG) |
| 4 | VLAN{1\|2} | VLAN [*3] |
| 5 | MPLS{1\|2\|3} | MPLS [*2] |
| 6 | IPV4 | IPv4[*4,*5] |
| | IPV6 | IPv6[*4,*6] |
| | ARP | ARP[*5,*6] |
| 7 | ICMPV4_ECHO | ICMPv4 Echo |
| | ICMPV6_ECHO | ICMPv6 Echo |
| | ICMPV6_NS | ICMPv6 NS |
| | ICMPV6_NA | ICMPv6 NA |

* 1: When NONE and CUSTOM are specified, strings of Order 2 and subsequent ones cannot be set. If they are set, the error (-220) will be generated.

* 2: Set the label count to the number.

* 3: Set the tag count to the number.

* 4: This cannot be set at the same time with ARP.

* 5: This cannot be set at the same time with IPv6.

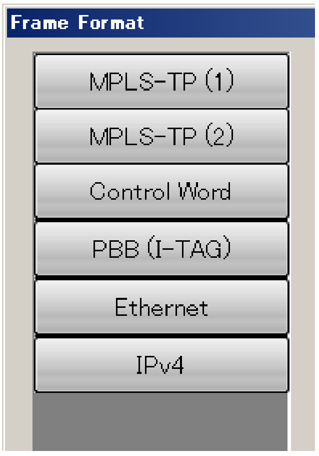
* 6: This cannot be set at the same time with IPv4.

**Response Data**

<STRING>,<HEX>

**Example of Use**

To set the header format shown in the figure below.



```
:SOURce:STReam:HEADer
"MPLS_TP2,I_TAG,IPV4",&H00FF001000F000FF011000E088470001
0080000101800000000000000500000300000A000008088E700000000
009230001F0000923000100000800
:SOURce:STReam:HEADer?
>
"MPLS_TP2,I_TAG,IPV4",&H00FF001000F000FF011000E088470001
0080000101800000000000000500000300000A000008088E700000000
009230001F00009230001000080045000002000040004011B99AC0A8
0000C0A80000
```

## :SOURce:STReam:HEADer:ETHer:DA

**Function**

This command sets and queries the value of Destination MAC Address field of the Ethernet frame to be sent.

**Syntax**

```
:SOURce:STReam:HEADer:ETHer:DA <HEX>
:SOURce:STReam:HEADer:ETHer:DA?
```

<HEX>: Destination MAC Address (6 octets hexadecimal)

**Response Data**

<HEX>

**Example of Use**

Refer to the example for :SOURce:STReam:HEADer:ETHer:SA.

## :SOURce:STReam:HEADer:ETHer:SA

**Function**

This command sets and queries the value of Source MAC Address field of the Ethernet frame to be sent.

**Syntax**

```
:SOURce:STReam:HEADer:ETHer:SA <HEX>
:SOURce:STReam:HEADer:ETHer:SA?
```

<HEX>: Source MAC Address (6 octets hexadecimal)

**Response Data**

<HEX>

**Example of Use**

To set the Destination MAC Address field to 00:00:00:00:00:01 and the Source MAC Address field to 00:00:00:00:00:02:

```
:SOURce:STReam:HEADer:ETHer:DA #H000000000001
:SOURce:STReam:HEADer:ETHer:SA #H000000000002
:SOURce:STReam:HEADer:ETHer:DA?
> #H000000000001
:SOURce:STReam:HEADer:ETHer:SA?
> #H000000000002
```

## :SOURce:STReam:HEADer:ETHer:TYPE

**Function**

This command sets the value of Type field of the Ethernet frame to be sent.

**Syntax**

```
:SOURce:STReam:HEADer:ETHer:TYPE <HEX>
:SOURce:STReam:HEADer:ETHer:TYPE?
```

<HEX>: Type field value (2 octets hexadecimal)

**Example of Use**

To set the value of Ethernet Type field to 0x0800 (IPv4):

```
:SOURce:STReam:HEADer:ETHer:TYPE #H0800
:SOURce:STReam:HEADer:ETHer:TYPE?
>#H0800
```

## :SOURce:STReam:HEADer:PATTern

### Function

This command sets and queries the header pattern of the stream.

### Syntax

```
:SOURce:STReam:HEADer:PATTern <HEX>
:SOURce:STReam:HEADer:PATTern?
```

<HEX>:

When the ROUTe:BERT command is set to On:

The header pattern (12 bytes of the MAC address (header of source address and destination of source address) and 2 bytes of Ethernet Type field) is set as a hexadecimal number.

When the ROUTe:BERT command is set to Off:

The header pattern (up to 257 bytes) is set as a hexadecimal number. The 257 bytes include 12 bytes of the MAC address (header of source address and destination of source address) and 2 bytes of the pattern including EtherType field.

***Note:***

When the ROUTe:BERT command is set to Off, setting the pattern with this command changes the format set with :SOURce:STReam:HEADer to CUSTOM.

When the ROUTe:BERT command is set to Off, set the pattern with :SOURce:STReam:HEADer.

### Response Data

<HEX>

### Example of Use

To set value of Ethernet Type field to 0x0800 (IPv4):

:SOURce:STReam:HEADer:PATTern
#H89ABCDABCDEF12345634567888A861008100E0100800450303CE00
0040004017A199C0A80A14C0A80A18

```
:SOURce:STReam:HEADer:PATTern?
>#H89ABCDABCDEF12345634567888A861008100E0100800450303CE0
00040004017A199C0A80A14C0A80A18
```

## :SOURce:STReam:HEADer:VARiable1:RANGe

**Function**

This command sets the destination MAC address variable range. Also, it sets the variable range of the Stream Control/Header dialog Modifiers tab Field 1 simultaneously.
It queries the destination MAC address variable range.

**Syntax**

```
:SOURce:STReam:HEADer:VARiable1:RANGe
<offset>,<length>,<count>
:SOURce:STReam:HEADer:VARiable1:RANGe?
```

<offset>,<length>,<count>=<NR1>

| | |
|---|---|
| <offset>: | Number of bytes from Ethernet frame header. (0 to 5) |
| <length>: | Variable byte count (1 to 6) |
| <count>: | Generated count (1 to n) |

Sets as <offset>+<length>< 7
The value of n varies with <length>.

| <length> | <count> |
|---|---|
| 1 | 1 to 256 |
| 2 | 1 to 65536 |
| 3 | 1 to 16777216 |
| 4 | 1 to 4294967296 |
| 5 | 1099511627776 |
| 6 | 281474976710656 |

When <length> is set to 5 or 6, the value is ignored even though setting the value other than the numeric values listed in Table to <count>.

**Response Data**

<offset>,<length>,<count>

## :SOURce:STReam:HEADer:VARiable1:TYPE

**Function**

This command sets the destination MAC address variable method (Off/Decrement/Increment/Random).Also, it sets the variable method of the Stream Control/Header dialog Modifiers tab Field 1 simultaneously. It queries the destination MAC address variable method.

**Syntax**

```
:SOURce:STReam:HEADer:VARiable1:TYPE
DECREMENT|INCREMENT|OFF|RANDOM
:SOURce:STReam:HEADer:VARiable1:TYPE?
```

Select the type from following.

| | |
|---|---|
| OFF | OFF indicates Fixed at Stream Control/Header screen |
| DECREMENT | |
| INCREMENT | |
| RANDOM | |

**Response Data**

DECREMENT|INCREMENT|OFF|RANDOM

## :SOURce:STReam:HEADer:VARiable2:RANGe

**Function**

This command sets the source MAC address variable range. Also, it sets the variable range of the Stream Control/Header dialog Modifiers tab Field 2 simultaneously.
It queries the source MAC address variable range.

**Syntax**

```
:SOURce:STReam:HEADer:VARiable2:RANGe
<offset>,<length>,<count>
:SOURce:STReam:HEADer:VARiable2:RANGe?
```

<offset>,<length>,<count>=<NR1>

| | |
|---|---|
| <offset>: | Number of bytes from Ethernet frame header. (6 to 11) |
| <length>: | Variable byte count (1 to 6) |
| <count>: | Generated count (1 to n) |

Sets as <offset>+<length><13
The value n varies with <length>.

| <length> | <count> |
|---|---|
| 1 | 1 to 256 |
| 2 | 1 to 65536 |
| 3 | 1 to 16777216 |
| 4 | 1 to 4294967296 |
| 5 | 1099511627776 |
| 6 | 281474976710656 |

When <length> is set to 5 or 6, the value is ignored even though setting the value other than the numeric values listed in Table to <count>.
The offset setting range is different at the screen and by remote command.
At the Stream Control/Header screen, the offset value is from the Source MAC address header but at remote command it is from the Ethernet frame header.

**Response Data**

<offset>,<length>,<count>

## :SOURce:STReam:HEADer:VARiable2:TYPE

**Function**

This command sets the source MAC address variable method (Off/Decrement/Increment/Random).Also, it sets the variable method of the Stream Control/Header dialog Modifiers tab Field 2 simultaneously. It queries the source MAC address variable method.

**Syntax**

```
:SOURce:STReam:HEADer:VARiable1:TYPE
DECREMENT|INCREMENT|OFF|RANDOM
:SOURce:STReam:HEADer:VARiable1:TYPE?
```

Select the type from following.

| | |
|---|---|
| OFF | OFF indicates Fixed at Stream Control/Header screen |
| DECREMENT | |
| INCREMENT | |
| RANDOM | |

**Response Data**

DECREMENT|INCREMENT|OFF|RANDOM

## :SOURce:STReam:HEADer:VARiable{3|4|5}

**Function**

This command sets parameters of the Stream Control/Header dialog Modifiers tab Field 3, 4, and 5.

It queries parameters of the Stream Control/Header dialog Modifiers tab Field 3, 4, and 5.

**Syntax**

```
:SOURce:STReam:HEADer:VARiable{3|4|5}
<type>[,<field>,<offset>,<length>,<min>,<max>,<step>]
:SOURce:STReam:HEADer:VARiable{3|4|5}?
```

<type>=<CHAR>

Specify the following strings to <type>.

OFF Parameters on the Stream Control/Header dialog are Fixed.

DECREMENT

INCREMENT

RANDOM

When OFF is set to <type>, the second parameter and subsequent ones can be omitted. In addition, even if the second parameter and subsequent ones are set, they are ignored.

<field>==CHAR>

Specify the string in the following table to <field>.

**Table 3.3.2-5 String to Specify Offset Location**

| <CHAR> | Header offset location |
|---|---|
| TOP | Top of Frame *1 |
| MPLS_TP_EXP{1\|2\|3\|4\|5} | MPLS-TP Tags Label |
| MPLS_TP_LABEL{1\|2\|3\|4\|5} | MPLS-TP Tags Exp |
| MPLS_TP_TTL{1\|2\|3\|4\|5} | MPLS-TP Tags TTL |
| CW_NIB | MPLS Control Word first nibble |
| CW_FLAG | MPLS Control Word flags |
| CW_FRG | MPLS Control Word FRG (Fragmentation) |
| CW_LENGTH | MPLS Control Word Length |
| CW_SEQ | MPLS Control Word Sequence Number |
| PBB_DA | PBB Destination MAC Address *2 |
| PBB_SA | PBB Source MAC Address *2 |
| PBB_{B\|I}_PCP | PBB Tags PCP |
| PBB_B_VID | PBB B-Tag VID |
| PBB_I_SID | PBB I-Tag SID |

Table 3.3.2-5 String to Specify Offset Location (Continued)

| <CHAR> | Header offset location |
|---|---|
| ETH_DA | Ethernet Destination MAC Address *3 |
| ETH_SA | Ethernet Source MAC Address *3 |
| ETH_TYPE | Ethernet Type |
| VLAN_{1\|2}_PCP | VLAN Tags PCP |
| VLAN_{1\|2}_VID | VLAN Tags VID |
| MPLS_EXP{1\|2\|3} | MPLS Tags Label |
| MPLS_LABEL{1\|2\|3} | MPLS Tags Exp |
| MPLS_TTL{1\|2\|3} | MPLS Tags TTL |
| IPV4_DA | IPv4 Destination Address |
| IPV4_SA | IPv4 Source Address |
| IPV4_TOS | IPv4 TOS |
| IPV4_TTL | IPv4 TTL |
| IPV4_PROT | IPv4 Protocol |
| IPV6_DA | IPv6 Destination Address |
| IPV6_SA | IPv6 Source Address |
| IPV6_TCLASS | IPv6 Traffic Class |
| IPV6_FLABEL | IPv6 Flow Label |
| IPV6_HOP | IPv6 Hop Limit |
| ARP_SMAC | ARP Source MAC Address |
| ARP_SIP | ARP Source IP Address |
| ARP_TMAC | ARP Target MAC Address |
| ARP_TIP | ARP Target IP Address |
| ARP_OPE | ARP Operation |
| ICMPV4_CODE | ICMPv4 Code |
| ICMPV4_EC_ID | ICMPv4 Identifier |
| ICMPV4_EC_SEQ | ICMPv4 Sequence No. |
| ICMPV6_CODE | ICMPv6 Code |
| ICMPV6_EC_ID | ICMPv6 Identifier |
| ICMPV6_EC_SEQ | ICMPv6 Sequence No. |
| ICMPV6_NSNA_TADDR | ICMPv6 Target Address |
| ICMPV6_NSNA_SADDR | ICMPv6 Source Link Layer Address |

* 1: When other than CUSTOM is set with :SOURce:STReam:HEADer, specifying TOP with this command generates the error (–220).

* 2: Can be specified when MPLS_TP and PBB are set to the stream format

* 3: Can be specified when MPLS_TP or PBB are set to the stream format

The same string cannot be set to <field> among Field 3 to 5. If the same string is set to <field> among Field 3 to 5, the error (–220) will be generated.

Changing the stream frame with :SOURce:STReam:HEADer turns OFF the <type> setting.

Specifying a field that does not exist in the frame format set with :SOURce:STReam:HEADer to <format> generates the error (-220).

Setting the stream format to Custom Header with :SOURce:STReam:HEADer changes the <format> setting of this command to TOP.

Setting the stream format to other than Custom Header with :SOURce:STReam:HEADer changes the <format> setting of this command to ETH_TYPE.

<offset>,<length>,<min>,<max>,<step>=<NR1>

| | |
|---|---|
| <offset>: | Bit count from the beginning of the target range<br>Settable values depend on the target range bit count. |
| <length>: | Bit count (1 to 32) to be changed<br>Settable values depend on the target range bit count and <offset> values. |
| <min>: | Field minimum value |
| <max>: | Field maximum value |
| <step>: | Field value change amount |

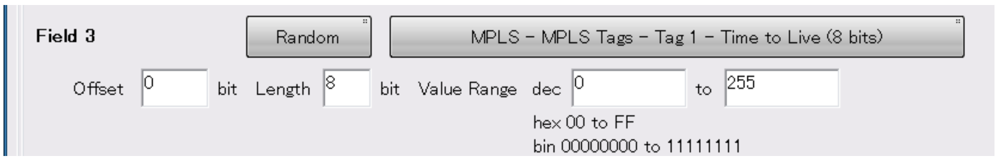
**Response Data**

<CHAR>,<CHAR>,<NR1>,<NR1>,<NR1>,<NR1>,<NR1>

In the order of <type>, <field>, <offset>, <length>, <min>, <max>, and <step>

**Example of Use**

To set the parameters at the Field 3 in the Modifiers tab as shown in the following screen shot.



```
:SOURce:STReam:HEADer:VARiable3
RANDOM,MPLS_TTL1,0,8,0,255,1
:SOURce:STReam:HEADer:VARiable3?
> RANDOM,MPLS_TTL1,0,8,0,255,1
```

## :SOURce:STReam:ID

**Function**

This command sets and queries the stream ID to be edited.
When editing the stream, set ID beforehand.
When connecting multiple units, :SOURce:STReam:ID must be set per unit.

**Syntax**

```
:SOURce:STReam:ID <NR1>
:SOURce:STReam:ID?
```

| | |
|---|---|
| <NR1>: | Stream ID (1 to 16) |

**Response Data**

<NR1>

**Example of Use**

To set stream No. 2 for unit number 1 through 3:
(Stream setting of the Unit No.1)

```
:UENTry:ID 1
:SOURce:STReam:ID 2
```

(Stream setting of the Unit No.2)

```
:UENTry:ID 2
:SOURce:STReam:ID 2
```

(Stream setting of the Unit No.3)

```
:UENTry:ID 3
:SOURce:STReam:ID 2
```

## :SOURce:STReam:RESolve:IP:TARGet

**Function**

This command sets and queries the IP address that should be queried by ARP.

**Syntax**

```
:SOURce:STReam:RESolve:IP:TARGet <CHAR>
:SOURce:STReam:RESolve:IP:TARGet?
```

<CHAR>=GATEWAY | IPADDRESS

| | |
|---|---|
| GATEWAY: | Gateway IP Address |
| IPADDRESS: | Destination IP Address |

**Response Data**

GATEWAY | IPADDRESS

**Example of Use**

```
:SOURce:STReam:RESolve:IP:TARGet IPADDRESS
:SOURce:STReam:RESolve:IP:TARGet?
> IPADDRESS
```

## :SOURce:STReam:RESolve:IPV4:ROUTer

**Function**

This command sets and queries the Gateway IPv4 address that should be queried by ARP.

**Syntax**

```
:SOURce:STReam:RESolve:IPV4:ROUTer <HEX>
:SOURce:STReam:RESolve:IPV4:ROUTer?
```

<HEX>: Gateway IPv4 address (4-byte hexadecimal number)

**Example of Use**

To set the Gateway IP address to 192.168.1.10 (0xC0A8010A).

```
:SOURce:STReam:RESolve:IPV4:ROUTer #HC0A8010A
:SOURce:STReam:RESolve:IPV4:ROUTer?
>#HC0A8010A
```

## :SOURce:STReam:RESolve:IPV6:ROUTer

**Function**

This command sets and queries the Gateway IPv6 address that should be queried by NS.

**Syntax**

```
:SOURce:STReam:RESolve:IPV6:ROUTer <HEX>
:SOURce:STReam:RESolve:IPV6:ROUTer?
```

<HEX>: Gateway IPv6 address (16-byte hexadecimal number)

**Example of Use**

To set the Gateway IP address to 2001::1.

```
:SOURce:STReam:RESolve:IPV6:ROUTer
#H20010000000000000000000000000001
:SOURce:STReam:RESolve:IPV6:ROUTer?
>#H20010000000000000000000000000001
```

## :SOURce:STReam:RESolve:MAC:RETRy

**Function**

This command sets and queries the number of retries to resolve the MAC address.

**Syntax**

```
:SOURce:STReam:RESolve:MAC:RETRy <NR1>
:SOURce:STReam:RESolve:MAC:RETRy?
```

<NR1>=1 to 100

**Response Data**

<NR1>

**Example of Use**

To set the number of retries to resolve the MAC address to 10.

```
:SOURce:STReam:RESolve:MAC:RETRy 10
:SOURce:STReam:RESolve:MAC:RETRy?
> 10
```

## :SOURce:STReam:RESolve:MAC:TIMeout

**Function**

This command sets and queries the timeout for MAC address resolution.

**Syntax**

```
:SOURce:STReam:RESolve:MAC:TIMeout <NR1>
:SOURce:STReam:RESolve:MAC:TIMeout?
```

| | |
|---|---|
| <NR1>: | Timeout 3 to 10 seconds |

**Response Data**

<NR1>

**Example of Use**

```
:SOURce:STReam:RESolve:MAC:TIMeout 10
:SOURce:STReam:RESolve:MAC:TIMeout?
> 10
```

## :SOURce:STReam:RESolve:PING:TRY

**Function**

This command sets and queries the number of times to execute the Ping.

**Syntax**

```
:SOURce:STReam:RESolve:PING:TRY <NR1>
:SOURce:STReam:RESolve:PING:TRY?
```

| | |
|---|---|
| <NR1>: | Number of Ping execution times 1 to 100 |

**Response Data**

<NR1>

**Example of Use**

```
:SOURce:STReam:RESolve:PING:TRY 20
:SOURce:STReam:RESolve:PING:TRY?
> 20
```

## :SOURce:STReam:RESolve:PING:PAYLoad

**Function**

This command sets and queries the payload type to execute the Ping.

**Syntax**

```
:SOURce:STReam:RESolve:PING:PAYLoad ALL0|ALL1|DEFAULT
:SOURce:STReam:RESolve:PING:PAYLoad?
```

| | |
|---|---|
| ALL0: | All 0 |
| ALL1: | All 1 |
| ALTERNATE: | 0/1 pattern |

**Response Data**

ALL0|ALL1|ALTERNATE

**Example of Use**

```
:SOURce:STReam:RESolve:PING:PAYLoad ALL0
:SOURce:STReam:RESolve:PING:PAYLoad?
> ALL0
```

## :SOURce:STReam:RESolve:PING:TIMeout

**Function**

This command sets and queries the Ping timeout.

**Syntax**

```
:SOURce:STReam:RESolve:PING:TIMeout <NR1>
:SOURce:STReam:RESolve:PING:TIMeout?
```

| | |
|---|---|
| <NR1>: | Ping timeout 3 to 10 seconds |

**Response Data**

<NR1>

**Example of Use**

```
:SOURce:STReam:RESolve:PING:TIMeout 10
:SOURce:STReam:RESolve:PING:TIMeout?
> 10
```

## :SOURce:STReam:RESolve:RESult

**Function**

This command queries the result of MAC address resolution or Ping execution.

**Syntax**

```
:SOURce:STReam:RESolve:RESult? <CHAR>
```

<CHAR>=PING | RESOLVE

| | |
|---|---|
| PING: | Ping |
| RESOLVE: | Resolve |

**Response Data**

<NR1>,<NR1>,<NR1>,....<NR1>

| | |
|---|---|
| <NR1>: | Result of MAC address resolution or Ping execution |
| 0: | Success |
| 1: | Excluded or not executed. |
| 2: | Failure |

Response data is 1 when MAC address resolution or Ping execution is in progress.
The number of response data is 16. Every one of them is a result of Streams 1 to 16.

**Example of Use**

```
:SOURce:STReam:RESolve:RESult? RESOLVE
> 0,0,0,0,0,1,2,2,2,2,2,2,2,2,2,2
```

## :SOURce:STReam:RESolve:STARt

**Function**

This command sets Stream(s) to start MAC address resolution or Ping execution.

**Syntax**

```
:SOURce:STReam:RESolve:STARt <CHAR>
```

<CHAR>=ALL|IPV4ALL|IPV6ALL|S1|S2|S3|S4|S5|S6|S7|S8|S9|S10|S11|S12|S13|S14|S15|S16

| | |
|---|---|
| ALL: | All Streams |
| IPV4ALL: | All Streams including IPv4 in the frame configuration |
| IPV6ALL: | All Streams including IPv6 in the frame configuration |
| S1: | Stream 1 |
| S2: | Stream 2 |
| : | : |
| S16: | Stream 16 |

## :SOURce:STReam:RESolve:STARt:EVENt

**Function**

This command queries the start event for MAC address resolution or Ping execution.
If the start event is read, then the response data is 0.

**Syntax**

```
:SOURce:STReam:RESolve:STARt:EVENt?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | MAC address resolution or Ping execution has not started. Event has been read after the start of MAC address resolution or Ping execution. |
| 1: | MAC address resolution or Ping execution has started. |

**Example of Use**

```
:SOURce:STReam:RESolve:STARt ALL
:SOURce:STReam:RESolve:STARt:EVENt?
> 1
:SOURce:STReam:RESolve:STARt:EVENt?
> 0
```

## :SOURce:STReam:RESolve:STATus

**Function**

This command queries the status of MAC address resolution or Ping execution.

**Syntax**

```
:SOURce:STReam:RESolve:STATus?
```

**Response Data**

0|1|2

| | |
|---|---|
| 0: | End of MAC address resolution or Ping execution. |
| 1: | MAC address resolution or Ping execution in progress. |
| 2: | MAC address resolution or Ping execution ended but results of part or all of Streams are Failure. |

**Example of Use**

```
:SOURce:STReam:RESolve:STARt ALL
:SOURce:STReam:RESolve:STATus?
> 1
```

## :SOURce:STReam:RESolve:STOP

**Function**

To stop MAC address resolution or Ping execution.

**Syntax**

```
:SOURce:STReam:RESolve:STOP
```

**Example of Use**

```
:SOURce:STReam:RESolve:STOP
:SOURce:STReam:RESolve:STATus?
> 2
```

## :SOURce:STReam:RESolve:TYPE

**Function**

This command sets and queries the type of resolution.

**Syntax**

```
:SOURce:STReam:RESolve:TYPE <CHAR>
:SOURce:STReam:RESolve:TYPE?
```

<CHAR>=RES|RES_PING|PING

| | |
|---|---|
| PING: | Ping Only |
| RES: | Resolve Only |
| RES_PING: | Resolve & Ping |

**Response Data**

PING|RES|RES_PING

**Example of Use**

```
:SOURce:STReam:RESolve:TYPE RES
:SOURce:STReam:RESolve:TYPE?
> RES
```

## :SOURce:STReam:STARt[:AUNit]

**Function**

This command starts the stream transmission.
When already sending the stream, it restarts the stream.
Adding :AUNit starts sending the stream for all connected units simultaneously.

**Syntax**

```
:SOURce:STReam:STARt[:AUNit]
```

When the remote command for the stream (:SOURce:STReam:*) is sent, Control Unit at the stream screen is changed to Gap Size (byte).

**Example of Use**

To start the stream transmission:

1. Clear the start event by sending :SOURce:STReam:STARt:EVENt? .
2. Start the stream transmission by sending :SOURce:STReam:STARt .
3. Sending :SOURce:STReam:STARt:EVENt? is repeated until the response data is 1 (started).

## :SOURce:STReam:STARt:EVENt[:AUNit]

**Function**

This command queries the start event of the stream transmission.
When sending the stream, it becomes 1.

**Syntax**

```
:SOURce:STReam:STARt:EVENt[:AUNit]?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | 0 before counter started |
| 1: | Cleared to 0 after reading with this command<br>Adding :AUNit to the header sets the stream transmission of all connected units to 1 when started. |

**Example of Use**

Refer to the example for :SOURce:STReam:STARt[:AUNit].

## :SOURce:STReam:STATus[:AUNit]

**Function**

This command queries the stream sending operation status.
When :AUNit is added to the header, becomes 1 when the stream transmission starts at any of units and becomes 0 when all units stopped.

**Syntax**

```
:SOURce:STReam:STATus[:AUNit]?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Stream transmission stopped |
| 1: | Transmitting stream |

When :AUNit is added to the header, becomes 1 when the stream transmission starts at any of units and becomes 0 when all units stopped.

**Example of Use**

Refer to the example for :SOURce:STReam:STOP[:AUNit].

## :SOURce:STReam:STOP[:AUNit]

**Function**

This command stops the stream transmission. When adding :AUNit, this command stops the stream transmission of all units connected.

**Syntax**

```
:SOURce:STReam:STOP[:AUNit]
```

**Example of Use**

To stop the stream transmission:

1. Stop the stream transmission by sending :SOURce:STReam:STOP .
2. Send repeatedly until the :SOURce:STReam:STATus? response becomes 0 (stopped).

## :SOURce:STReam:TFRame:ENABle

**Function**

This command sets and queries whether to enable/disable the test frame specified by :SOURce:STReam:ID.

**Syntax**

```
:SOURce:STReam:TFRame:ENABle 0|1|OFF|ON
:SOURce:STReam:TFRame:ENABle?
```

| | |
|---|---|
| 0\|OFF: | Disables test frame |
| 1\|ON: | Enables test frame |

**Response Data**

0|1

**Example of Use**

To enable test frame:

```
:SOURce:STReam:TFRame:ENABle ON
:SOURce:STReam:TFRame:ENABle?
> 1
```

## :SOURce:STReam:TFRame:FID

**Function**

This command sets the flow ID of the test frame corresponding to the stream being edited.
It also queries the flow ID settings of the test frame.
The stream for setting the test frame can be set using :SOURce:STReam:ID.

**Syntax**

```
:SOURce:STReam:TFRame:FID <NR1>
:SOURce:STReam:TFRame:FID?
```

| | |
|---|---|
| <NR1>: | Flow ID(0 to 15) |

**Response Data**

<NR1>

**Example of Use**

To set the flow ID of the stream No. 5 to 12:

```
:SOURce:STReam:ID 5
:SOURce:STReam:TFRame:FID 12
:SOURce:STReam:TFRame:FID?
> 12
```

## :SOURce:STReam:TYPE

**Function**

This command sets and queries the stream transmission method.

**Syntax**

```
:SOURce:STReam:TYPE RANDOM|SEQUENTIAL
:SOURce:STReam:TYPE?
```

| | |
|---|---|
| SEQUENTIAL: | Send the stream number from the smallest one repeatedly. |
| RANDOM: | Send the stream number randomly. |

**Response Data**

RANDOM|SEQUENTIAL

## :SOURce:TPATtern:INVert

**Function**

This command inverts the PRBS test pattern settings (transmission).
It also queries the inverse settings of the PRBS test pattern (transmission).
:SOURce:TPATtern:INVert sets the inverse pattern at the reception side.

**Syntax**

```
:SOURce:TPATtern:INVert 0|1|OFF|ON
:SOURce:TPATtern:INVert?
```

| | |
|---|---|
| 0\|OFF: | Inverse pattern Off |
| 1\|ON: | Inverse pattern On |

**Response Data**

0|1

**Example of Use**

Refer to the example for :SENSe:TPATtern:INVert.

## :SOURce:TPATtern:TYPE

**Function**

This command sets and queries the test pattern types of the transmission side (Tx).

**Syntax**

```
:SOURce:TPATtern:TYPE <CHAR>
:SOURce:TPATtern:TYPE?
```

<CHAR>

The text pattern types are as shown below. Select any of them.

PRBS7
PRBS9
PRBS11
PRBS15
PRBS23
PRBS31
ALL0
ALL1
WORD16
SQUARE : Square Wave

| Mapping | Text pattern type |
|---|---|
| 40GbE, 100GbE, OTU4 Ethernet Client | PRBS31, ALL0, ALL1, WORD16 |
| OTU3-PRBS, OTU4-PRBS | PRBS11, PRBS15, PRBS23, PRBS31, WORD16 |
| OTU4-ODTU4.8-ODU2e-PRBS, OTU4-ODTU4.1-ODU0-PRBS | PRBS31, Word16 |
| 100GbE No Frame, 40GbE No Frame, OTU4 No Frame, OTU3 No Frame, | PRBS7, PRBS9, PRBS11, PRBS15, PRBS23, PRBS31, SQUARE |

**Response Data**

ALL0 | ALL1 | PRBS7 | PRBS9 | PRBS11 | PRBS15 | PRBS23 | PRBS31 | SQUARE | WORD16

**Example of Use**

```
:SOURce:TPATtern:TYPE PRBS31
:SOURce:TPATtern:TYPE?
>PRBS31
```

## :SOURce:TPATtern:WORD

**Function**

This command sets the pattern when selecting Word16 at the test pattern of the transmission side (Tx). It also queries the Word16 pattern of the transmission side (Tx).

**Syntax**

```
:SOURce:TPATtern:WORD <HEX>
:SOURce:TPATtern:WORD?
```

| | |
|---|---|
| <HEX>: | Values of Data Field (2 bytes) |

**Response Data**

<HEX>

**Example of Use**

```
:SOURce:TPATtern:TYPE WORD16
:SOURce:TPATtern:WORD #H55AA
:SOURce:TPATtern:WORD?
>#H55AA
```

## :SOURce:TRANsceiver:EMPHasis:FIRSt

**Function**

This command sets Pre-Emphasis (First) of the transceiver.
It also queries the Pre-Emphasis (First) of the transceiver.

**Syntax**

```
:SOURce:TRANsceiver:EMPHasis:FIRSt <lane>,<value>
:SOURce:TRANsceiver:EMPHasis:FIRSt? <lane>
```

<lane>,<value>=<NR1>

<lane>: Lane number

| Application | Range of <lane> |
|---|---|
| 40GbE,OTU3 | 0 to 3 |
| 100GbE,OTU4 | 0 to 19 |
| 40GbE No Frame, OTU3 No Frame | 0 to 3 |
| 100GbE No Frame, OTU4 No Frame | 0 to 19 |

<value>: Setting value (0 to 31)

**Response Data**

<NR1>

## :SOURce:TRANsceiver:EMPHasis:PRE

**Function**

This command sets Pre-Emphasis (Pre) of the transceiver. It also queries the Pre-Emphasis (Pre) settings of the transceiver.

**Syntax**

```
:SOURce:TRANsceiver:EMPHasis:PRE <lane>,<value>
:SOURce:TRANsceiver:EMPHasis:PRE? <lane>
```

<lane>,<value>=<NR1>

<lane>: Lane number
For the setting range of the lane number, refer to the explanation of:SOURce:TRANsceiver :EMPHasis:FIRSt.

<value>: Setting value (–15~+15)

**Response Data**

<NR1>

## :SOURce:TRANsceiver:EMPHasis:SECond

**Function**

This command sets the Pre-Emphasis (Second) of the transceiver. It also queries the Pre-Emphasis (Second) settings of the transceiver.

**Syntax**

```
:SOURce:TRANsceiver:EMPHasis:SECond <lane>,<value>
:SOURce:TRANsceiver:EMPHasis:SECond? <lane>
```

<lane>,<value>=<NR1>

| | |
|---|---|
| <lane>: | Lane number<br>For the setting range of the lane number, refer to the explanation of :SOURce:TRANsceiver :EMPHasis:FIRSt. |
| <value>: | Setting value (–15 to +15) |

**Response Data**

<NR1>

## :SOURce:TRANsceiver:VOD

**Function**

This command sets VOD of the transceiver. It also queries the VOD settings of the transceiver.

**Syntax**

```
:SOURce:TRANsceiver:VOD <lane>,<value>
:SOURce:TRANsceiver:VOD? <lane>
```

<lane>,<value>=<NR1>

| | |
|---|---|
| <lane>: | Lane number<br>For the setting range of the lane number, refer to the explanation of:SOURce:TRANsceiver :EMPHasis:FIRSt. |
| <value>: | Setting value (0 to 6) |

**Response Data**

<NR1>

## :SYSTem:CONFig

### Function

This command switches the application of the MD1260A.

Executing this command restarts the application.

It can be checked with :SYSTem:STATus? command whether the restart has been completed or not.

Commands other than :SYSTem:ERRor? and :SYSTem:STATus? cannot be used during restart.

### Syntax

```
:SYSTem:CONFig <CHAR>[,<CHAR>[,<CHAR>…[,<CHAR>]…]]
```

<CHAR>

Specify the application/mapping of Unit ID 1, 2, ..., 16 to <CHAR>.

Specify NONE to the unit ID not to be connected. However, following NONEs can be omitted.

For <CHAR>, specify any of the following.

| Application/Mapping | <CHAR> |
|---|---|
| None | NONE |
| 40GbE | E40G |
| 100GbE | E100G |
| ODU3-PRBS | OTU3 |
| ODU4-PRBS | OTU4 |
| ODU4-100GbE | OTU4_E |
| ODTU4.8-ODU2e-PRBS | OTU4_ODU2E |
| ODTU4.8-ODU2e-10GbE | OTU4_ODU2E_E |
| ODTU4.1-ODU0-PRBS | OTU4_ODU0 |
| ODTU4.1-ODU0-GbE | OTU4_ODU0_E |
| 40GbE No Frame | E40G_N |
| 100GbE No Frame | E100G_N |
| OTU3 No Frame | OTU3_N |
| OTU4 No Frame | OTU4_N |

### Example of Use

To specify 40GbE to Unit ID 1 and 100GbE to Unit ID 3:

```
:SYSTem:CONFig E40G,NONE,E100G
```

To specify ODU4-100GbE to Unit ID4:

```
:SYSTem:CONFig NONE,NONE,NONE,OTU4_E
```

Use :SOURce:MAPPing to query the application mapping running.

```
:UENTry:ID 1
:SOURce:MAPPing?
> OTU4
:UENTry:ID 2
:SOURce:MAPPing?
```

```
> OTU4
```

To switch the application/mapping of Unit ID1 to Unit ID3:

```
:SYSTem:CONFig OTU4_E,OTU4_ODU2E_E
:SYSTem:ERRor?
> 0,"No error"
```

:SYSTem:STATus? allows checking the restart has been completed.

```
:SYSTem:STATus?
> 0
     :
     :
:SYSTem:STATus?
> 1
```

When the restart is completed, other remote commands can be used for operation.

```
:UENTry:ID 1
:SOURce:MAPPing?
> OTU4_E
:UENTry:ID 2
:SOURce:MAPPing?
> OTU4_ODU2E_E
```

To set the number of lanes for 100GbE No Frame or OTU4 No Frame application, use :SOURce:MAPPing.

## :SYSTem:DATE

**Function**

This command queries the date and time. The time is returned by :SYSTem:TIME?.

**Syntax**

```
:SYSTem:DATE?
```

**Response Data**

<year>, <month>, <day> = <NR1>

Returns year, month, and date in comma-separated order

**Example of Use**

```
:SYSTem:DATE?;:SYSTem:TIME?
> 2010,10,7;12,40,3
```

(12:40:03 7 June 2010)

## :SYSTem:ERRor

### Function

This command queries the error information saved in the error event queue. One data item is read from the event queue each time.

### Syntax

```
:SYSTem:ERRor?
```

### Response Data

<error_event_number>,"<error_event_description>"
<error_event_number> = <NR1>
The range of the error code is from –32768 to +32767.
0 indicates that no errors or events have occurred.

<error_event_description> = <STRING>
<error_event_number>: This is the error messages corresponding to the value of <error_event_number>. The maximum length of this character string is 255.

A maximum of 128 error events can be saved in the event queue.
When the error event queue is full, nothing more can be stored so the the generated error information is discarded without discarding error information saved in the event queue. When this happens, the –350 (Queue overflow) error information is stored at the event 128 in the queue.

### Example of Use

When sending a command that the application does not support, the –113 ( Undefined header) error is returned.
Example: When sending command not supported by 100GbE application:

```
:CALCulate:MONitor:OTU:COLumn 1
:SYSTem:ERRor?
> -113,"Undefined header"
```

When specifying a parameter that the application does not support, –220,"Parameter error" is returned.
Example: When sending parameter not supported by 100GbE application:

```
:SOURce:EALarm:TYPE FAS_MLD
:SYSTem:ERRor?
>-220,"Parameter error"
```

When specifying parameter that cannot be executed in the current application status, –310,"System error" is returned.

Example: When specifying non-existent command that cannot be loaded by :MMEMory:RECall

```
:MMEMory:RECall "nonexistent_file"
:SYSTem:ERRor?
> -310,"System error"
```

In these cases, the error can be confirmed by sending SYSTem:ERRor? immediately after the sent command. Refer to Section 2.6 Confirming Message Execution Status and Appendix B Error Codes.

However, since the errors are saved in the queue, if the queue is not checked each time it is emptied, sometimes it is not possible to capture immediately preceding command errors, as follows:

```
:SOURce:EALarm:TYPE FAS_MLD
:MMEMory:RECall "nonexistent_file"
:SYSTem:ERRor?
> -220,"Parameter error"← :SOURce:EALarm:TYPE error
:SYSTem:ERRor?
> -310,"System error"← :MMEMory:RECall error
:SYSTem:ERRor?
> 0,"No error"← becomes "No error" when queue empty
:SYSTem:ERRor?
> 0,"No error"
```

## :SYSTem:PRINt:COPY

**Function**

This command copies the screen.

**Syntax**

```
:SYSTem:PRINt:COPY
```

The currently displayed screen is output as a PNG format file. The screen copy file is created in the following path.
C:\Documents and Settings\Administrator
\My Documents\Anritsu\MD1260A\UserData\Screen Copy

## :SYSTem:STATus

**Function**

This command queries whether the application is running.
Use it after the application is set with :SYSTem:CONFig.

**Syntax**

```
:SYSTem:STATus?
```

**Response Data**

0|1

| | |
|---|---|
| 0: | Stopped or starting |
| 1: | Running |

## :SYSTem:TERMination

**Function**

This command sets and queries the terminator types of the response data

**Syntax**

```
:SYSTem:TERMination 0|1
:SYSTem:TERMination?
```

Displays terminator types

| | |
|---|---|
| 0: | LF |
| 1: | CR+LF |

**Response Data**

0|1

The equipment start status is 0 (LF). Since this setting is not backed-up, the start status is always (LF). In addition, this setting cannot be changed with the *RST command.

With the GPIB option, EOI (GPIB bus line signal) is generated simultaneously.

**Example of Use**

Refer to Section 2.4 Checking Connection for Ethernet.

## :SYSTem:TIME

**Function**

This command queries the time. The date is queried by :SYSTem:DATE?.

**Syntax**

```
:SYSTem:TIME?
```

**Response Data**

<hour>,<min>,<sec> = <NR1>

Returns hour, minute, and second in comma-separated order

**Example of Use**

Refer to:SYSTem:DATE.

## :SYSTem:VERSion

**Function**

This command queries the SCPI version conforming to the MD1260A software.

**Syntax**

```
:SYSTem:VERSion?
```

**Response Data**

1999.0

1999.0 is returned in the SCPI version. (year: 1999, revision number : 0)

## :UENTry:ID

**Function**

This command sets and queries the Unit ID for the target operation.

**Syntax**

```
:UENTry:ID <NR1>
:UENTry:ID?
```

| | |
|---|---|
| <NR1>: | Unit ID from Unit 1 to Unit 16 (1~16) |

**Response Data**

<NR1>

The –220 error is returned for out of range or when the set Unit ID is not connected.

**Example of Use**

To set Unit ID for target operation to ID 16:

```
:UENTry:ID 16
:UENTry:ID?
> 16
```

## :UENTry:LIST

**Function**

This command queries operable Unit IDs.

**Syntax**

```
:UENTry:LIST?
```

**Response Data**

<num of unit>, <unit id> [, <unit id>, <unit id>,....]= <NR1>

| | |
|---|---|
| <num of unit>: | Count of operable units |
| <unit id>: | ID of Unit ID |

When there are multiple operable units, the Unit IDs are arranged in the following order.
Master Unit ID
Slave IDs (from smallest)

**Example of Use**

```
:UENTry:LIST?
> 3,1,2,16
```

# *Chapter 4 Operation Record Function*

This chapter explains the function for making a remote control program.

## 4.1 Outline of Operation Record Function

The operation record function supports making a remote control program.
When setting the parameter etc. on the MD1260A screen while executing the operation record function, a remote control command corresponding to the screen operation is saved to the text file.
When using the operation record function, the command querying the setting details and measurement results can be saved to the text file as well.

```
*** Operation Log Start
:UENTry:ID 1
*** Test Pattern Setting
:SOURce:STReam:HEADer:ETHer:SA
#H0123456789AB
:SOURce:STReam:HEADer:ETHer:TYPE #H1234
:SOURce:STReam:ERRor:TYPE FCS
:SOURce:STReam:FSIZe:TYPE FIXED
:SOURce:STReam:FSIZe:VALue 60
:SOURce:STReam:FSIZe:RANGe 60,16376


*** Operation Log End
```

**Figure 4.1-1 File Creation Using Operation Record Function**

The measurement result file is saved in the following folder.
C:\ Documents and Settings\Administrator\My Documents \Anritsu\MD1260A\UserData\Operation Log

The file name is as follows.
'OperationLog' +'_' +date +hour, min, second+ millisecond
The file name saved to 55.324 at 16:40:55, October 7, 2010 is as follows.
OperationLog_20101007T164055324.txt

### Restrictions

For the operation record function, there are following restrictions.

- The setting dialog operation without the OK button is not recorded. Example: [Write] and [Read] on the MDIO screen
- The panel operation without the relevant remote commands is not recorded.
  Refer to Section 3.2.7 "Panel operation not controlled remotely".

## 4.2 Starting/Stopping Operation Record Function

Starting operation record function

After starting the application, the operation record function is started according to the following procedure.

1. Touch [System Menu].
2. Touch [Operation REC]. Operation Record Panel opens. The record of a remote command corresponding to the screen operation starts at the time of this. However, the operation for Operation Record Panel, and the operation without corresponding remote command are not recorded in the file.
3. Touching System Menu, setting area, operation area, or button on the top menu operates the MD1260A. The remote command corresponding to the operation is saved in the file.
4. When Operation Record Panel is hided, touch [Operation REC] of [System Menu].
5. When recording the comments corresponding to the screen operation, touch the comment text box of Operation Record Panel and enter the characters. When touching [Insert], the comment is saved in the file.

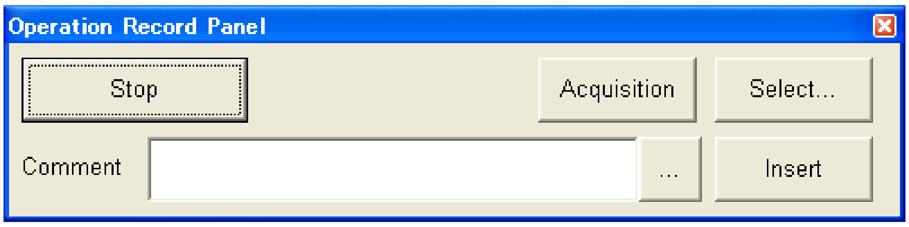


**Figure 4.2-1 Operation Record Panel**

When saving the the Query command to acquire the measurement result to the file, the following operation is performed.

1. Touch [Select] of Operation Record Panel. The Select screen opens.
2. Touch the measurement item to be read.
3. Touch [OK].
4. Touch [Acquisition] of Operation Record Panel.

   A remote command reading the measurement item is saved to the file.

Stopping operation record function

Touch [Stop] of Operation Record Panel.

Or, touch the close button of Operation Record Panel.

# *Appendix A Measurement Item List*

This appendix explains the compatibility of measurement items displayed on-screen and remote commands.

"IDs" in tables in Section A.3 "40/100GbE Application", Section A.4 "OTU3/OTU4 Application", and Section A.5 "No Frame Application" are character strings that can be specified with parameters of commands, :CALCulate:DATA and MMEMory:LOG:ITEM.

## A.1 Response Data Format

[Response Data Format] in the following table indicates the ID of the measurement items specified by the `:CALCulate:DATA?` as well as the format of the response data. The format of the response described here is one of the following.

**Table A.1-1 Response Data Format**

| Format | Explanation |
|---|---|
| `<status>` | 0\|1\|2<br>Returns value corresponding to color of status LED displayed on operation screen<br>0 Normal (green)<br>1 Abnormal (red)<br>2 History (orange) |
| `<NR1>` | Displays integer value<br>Example: 123, –500 |
| `<NR2>` | Displays numeric value in fixed-point decimal<br>Example: 12.345, –500.0 |
| `<NR3>` | Displays numeric value in floating-point decimal<br>Example: 0.00E-11, 3.05E–11 |

The [---------] character string is displayed on-screen when the measurement value is disabled for any of the measurement items. In addition, a character string in the format >Max. value, <Min. value is displayed (e.g. `>200.0`, `<-200.0`) to indicate an out-of-range value.

When there are multiple measurement items per lane, the data is returned for each lane separated by a comma (,).

***Note:***

Before capturing the counter value using :CALCulate:DATA, set the type of counter data captured by :CALCulate:DATA:TYPE to either Current or Accumulated.

## A.2 Common Items

**Table A.2-1 List of Common Items**

| Measurement Items | Command | Response Data Format |
|---|---|---|
| Stream Status Display (LED) | `:SOURce:STReam:STATus?` | <status> |
| Error/Alarm Insertion Status Display (LED) | `:SOURce:EALarm:STATus?` | <status> |
| Counter Status Display (LED) | `:CALCulate:COUNter:STATus?)` | <status> |
| Error/Alarm Status Display (LED) | `:CALCulate:EALARM?` | <status> |
| Log Status Display (LED) | `:MMEMory:LOG:STATus?` | <status> |

## A.3 40/100GbE Application

**Table A.3-1 Measurement Items and ID List**

| Tab | Measurement Item | ID | Response Data Format |
|---|---|---|---|
| (Common) | Elapsed Time (s) | `ELAPSED` | <NR1> |
| | Transmit Duration (ns) | `TX_TIME` | <NR2> |
| Test Frame | Tx Test Frame | `TX_TFRAME` | <NR1> |
| | Rx Test Frame | `RX_TFRAME` | |
| | Tx Rate (Mbit/s) | `TX_TBPS` | <NR1> |
| | Rx Rate (Mbit/s) | `RX_TBPS` | |
| | Sequence Error | `RX_SEQ_ERR` | <NR1> |
| | Current Latency (μs) *1 | `LAT_CURR` | <NR2> |
| | Maximum Latency (μs) *2 | `LAT_MAX` | <NR2> |
| | Minimum Latency (μs) *2 | `LAT_MIN` | <NR2 |
| | Tx Test Frame (Other) | `TX_TFRAME_O` | <NR1> |
| | Rx Test Frame (Other) | `RX_TFRAME_O` | |
| | Tx Rate (Mbit/s) (Other) | `TX_TBPS_O` | <NR1> |
| | Rx Rate (Mbit/s) (Other) | `RX_TBPS_O` | |
| Distribution | Tx Frame Size Distribution | `TX_FSIZE` | <NR1> |
| | Rx Frame Size Distribution | `RX_FSIZE` | |
| | Tx Gap Size Distribution | `TX_GSIZE` | <NR1> |
| | Rx Gap Size Distribution | `RX_GSIZE` | |

*1: Before capturing the value of LAT_CURR, send :CALCulate:DATA:TYPE CURRENT.

*2: Before capturing the values of LAT_MAX and LAT_MIN, send :CALCulate:DATA:TYPE ACCUM.

Table A.3-1 Measurement Items and ID List (Cont'd)

| Tab | Measurement Item | ID | Response Data Format |
|---|---|---|---|
| All Lanes | Tx Clock Status | CSLOSS | <status> |
| | Rx Clock Status | CUNLOCK | <status> |
| | Tx Frequency (Hz) | TX_FREQ | <NR1> |
| | Rx Frequency (Hz) | RX_FREQ | |
| | Tx Difference (ppm) | TX_FREQ_D | <NR1> |
| | Rx Difference (ppm) | RX_FREQ_D | |
| | Tx LF | TX_LF | <NR1> |
| | Rx LF | RX_LF | |
| | Tx RF | TX_RF | <NR1> |
| | Rx RF | RX_RF | |
| | Error Signals | RX_ERR_SIG | <NR1> |
| | Tx Errored Bytes | TEBYTE | <NR1> |
| | Rx Errored Bytes | REBYTE | |
| | Tx Good Bytes | TGBYTE | <NR1> |
| | Rx Good Bytes | RGBYTE | |
| | Tx FCS Errors | TFCS | <NR1> |
| | Rx FCS Errors | RFCS | |
| | Tx Fragments | TFRAGMENT | <NR1> |
| | Rx Fragments | RFRAGMENT | |
| | Tx Oversize & FCS Errors | TOAF | <NR1> |
| | Rx Oversize & FCS Errors | ROAF | |
| | Tx Undersize | TUNDERSIZE | <NR1> |
| | Rx Undersize | RUNDERSIZE | |
| | Tx Oversize | TOVERSIZE | <NR1> |
| | Rx Oversize | ROVERSIZE | |
| | Tx Good Frames | TGFRAME | <NR1> |
| | Rx Good Frames | RGFRAME | |
| | Tx Rate (bit/s) | TBPS | <NR1> |
| | Rx Rate (bit/s) | RBPS | |

*1: Before capturing the value of LAT_CURR, send :CALCulate:DATA:TYPE CURRENT.

*2: Before capturing the values of LAT_MAX and LAT_MIN, send :CALCulate:DATA:TYPE ACCUM.

Table A.3-1 Measurement Items and ID List (Cont'd)

| Tab | Measurement Item | ID | Response Data Format |
|---|---|---|---|
| All Lanes (Cont'd) | Tx Rate (%) | TRATE | <NR2> |
| | Rx Rate (%) | RRATE | |
| | Pause Frame | RX_PAUSE | <NR1> |
| | Trigger Condition | RX_TRIGGER | <NR1> |
| | Tx Broadcast | TBCAST_FRAME | <NR1> |
| | Rx Broadcast | RBCAST_FRAME | |
| | Tx Broadcast Bytes | TBCAST_BYTE | <NR1> |
| | Rx Broadcast Bytes | RBCAST_BYTE | |
| | Tx Multicast Frame | TMCAST_FRAME | <NR2> |
| | Rx Multicast Frame | RMCAST_FRAME | |
| | Tx Multicast Bytes | TMCAST_BYTE | <NR2> |
| | Rx Multicast Bytes | RMCAST_BYTE | |
| | Tx MPLS-TP | TMPLSTP | <NR2> |
| | Rx MPLS-TP | RMPLSTP | |
| | Tx PBB | TPBB | <NR1> |
| | Rx PBB | RPBB | |
| | Tx ARP Request | TARP_REQ | <NR1> |
| | Rx ARP Request | RARP_REQ | |
| | Tx ARP Reply | TARP_REP | <NR1> |
| | Rx ARP Reply | RARP_REP | |
| | Tx PINGv4 Request | TPINGV4_REQ | <NR1> |
| | Rx PINGv4 Request | RPINGV4_REQ | |
| | Tx PINGv4 Reply | TPINGV4_REP | <NR1> |
| | Rx PINGv4 Reply | RPINGV4_REP | |
| | Tx NDP (NS) | TNDP_NS | <NR1> |
| | Rx NDP (NS) | RNDP_NS | |
| | Tx NDP (NA) | TNDP_NA | <NR1> |
| | Rx NDP (NA) | RNDP_NA | |
| | Tx PINGv6 Request | TPINGV6_REQ | <NR1> |
| | Rx PINGv6 Request | RPINGV6_REQ | |
| | Tx PINGv6 Reply | TPINGV6_REP | <NR1> |
| | Rx PINGv6 Reply | RPINGV6_REP | |
| | Bit Errors (bit) | BER_CNT | <NR1> |
| | Bit Errors (rate) | BER_RATE | <NR3> |
| | Pattern Sync Loss | PSLOSS | <NR2> |

**Table A.3-1 Measurement Items and ID List (Cont'd)**

| Tab | Measurement Item | ID | Response Data Format |
|---|---|---|---|
| Individual | Alignment Status | ALIGNMENT | <status> |
| | High-BER | HIBER | <status> |
| | Invalid Block (Count) | IBLOCK | <NR1> |
| | (Rate) | IBLOCK_RATE | <NR3> |
| | Sync Header Lock | SHLOCK | <status> |
| | Alignment Marker Lock | AMLOCK | <status> |
| | Skew Stability | SSTAB | <status> |
| | Marker Map | MMAP | <NR1> |
| | Relative Skew (ns) | RSKEW | <NR2> |
| | Invalid Sync Header (Count) | ISH | <NR1> |
| | (Rate) | ISH_RATE | <NR3> |
| | Invalid Alignment Marker (Count) | IAM | <NR1> |
| | (Rate) | IAM_RATE | <NR3> |
| | BIP Error (bit) (Count) | BIP_CNT | <NR1> |
| | (Rate) | BIP_RATE | <NR3> |
| Opt | LOS | LOS | <status> |
| | Programmable Alarm 1 | PALARM1 | |
| | Programmable Alarm 2 | PALARM2 | <status> |
| | Programmable Alarm 3 | PALARM3 | |
| | Global Alarm | GALARM | <status> |
| | Optical Power (dBm) | OPOWER | <NR1> |

## A.4 OTU3/OTU4 Application

When the measurement results for the number of frames and time exist, add _S to the parameter to read the measurement results in seconds.

Example: To read the number of frames that generated PT-TIM: PM_TIM

To read the PM-TIM generation time (in seconds): PM_TIM_S

When the measurement results for the number of errors and generation rate exist, add _R to the parameter to read the measurement results for the generation rate.

Example: To read the number of PM-BIP8 generation: PM_BIP8

To read the rate of PM-BIP8 generation: PM_BIP8_R

When the high order (OTU4) and low order (ODU2e and ODU0) have the same measurement items, add _L to the parameters to read the measurement results for the low order.

Example: To read the number of PM-BIP8 generation for OTU4: PM_BIP8

To read the rate of PM-BIP8 generation for ODU2e or ODU0: PM_BIP8_L

**Table A.4-1 Measurement Items and ID List**

| Tab | | Measurement Item | ID | Response Data Format |
|---|---|---|---|---|
| (Common) Summary | | Elapsed Time (second) | `ELAPSED` | <NR1> |
| | | Clock Source Loss | `CSLOSS` | <status> |
| | | CDR Unlock | `CUNLOCK` | <status> |
| | | Rx Frequency (Hz) | `RX_FREQ` | <NR1> |
| | | Rx Frequency Difference (ppm) | `RX_FREQ_D` | <NR1> |
| | | LOS | `LOS` | <status> |
| Statistics | | | | |
| | LLD | ILA/OLA | `ILA` | <status> |
| | | Skew Stability | `SSTAB` | <status> |
| | | Marker Map | `MMAP` | <NR1> |
| | | Relative Skew (ns) | `RSKEW` | <NR2> |
| | | LOF Lane (ms) | `LOF_LLD` | <NR2> |
| | | OOF (frame) | `OOF_LLD` | <NR1> |
| | | LOR (ms) | `LOR` | <NR2> |
| | | OOR (frame) | `OOR` | <NR1> |
| | | FAS-LLD | `FAS_LLD` | <NR1> |
| | OTU3, OTU4 | LOF | `LOF` | <NR2> |
| | | OOF (second) | `OOF_S` | <NR2> |
| | | (frame) | `OOF` | <NR1> |
| | | LOM | `LOM` | <NR2> |
| | | OOM (second) | `OOM_S` | <NR2> |
| | | (frame) | `OOM` | <NR1> |
| | | SM-TIM | `SM_TIM` | <NR1> |
| | | SM-BIAE (second) | `SM_BIAE_S` | <NR2> |
| | | (frame) | `SM_BIAE` | <NR1> |
| | | SM-BDI (second) | `SM_BDI_S` | <NR2> |
| | | (frame) | `SM_BDI` | <NR1> |
| | | SM-IAE (second) | `SM_IAE_S` | <NR2> |
| | | (frame) | `SM_IAE` | <NR1> |
| | | FAS | `FAS` | <NR1> |

Table A.4-1 Measurement Items and ID List (Cont'd)

| Tab | | Measurement Item | ID | Response Data Format |
|---|---|---|---|---|
| Statistics (Cont'd) | | | | |
| | OTU3, OTU4 (Cont'd) | SM-BIP8 (count) | SMBIP8 | <NR1> |
| | | (rate) | SMBIP8_R | <NR3> |
| | | SM-BEI (count) | SMBEI | <NR1> |
| | | (rate) | SMBEI_R | <NR3> |
| | | FEC-Uncorr EBs | FUEB | <NR1> |
| | | FEC-Corr Errors | FCB | <NR1> |
| | | FEC-Corr 1s to 0s | FCB1S | <NR1> |
| | | FEC-Corr 0s to 1s | FCB0S | <NR1> |
| | OTU3, OTU4 | MSIM | MSIM | <status> |
| | | ODU-AIS (second) | AIS_ODU_S | <NR2> |
| | | (frame) | AIS_ODU | <NR1> |
| | | ODU-OCI (second) | OCI_S | <NR2> |
| | | (frame) | OCI | <NR1> |
| | | ODU-LCK (second) | LCK_S | <NR2> |
| | | (frame) | LCK | <NR1> |
| | | PM-TIM (frame) | PM_TIM | <NR1> |
| | | PM-BDI (second) | PM_BDI_S | <NR2> |
| | | (frame) | PM_BDI | <NR1> |
| | | Client-AIS (second) | AIS_C_S | <NR2> |
| | | (frame) | AIS_C | <NR1> |
| | | PLM | PLM | <NR1> |
| | | CSF | CSF | <NR1> |
| | | PM-BIP8 (count) | PM_BIP8 | <NR1> |
| | | (rate) | PM_BIP8_R | <NR3> |
| | | PM-BEI (count) | PM_BEI | <NR1> |
| | | (rate) | PM_BIP8_R | <NR3> |
| | TCM | TCMi-TIM | TCM{1\|2\|3\|4\|5\|6}_TIM_S | <NR2> |
| | | TCMi-BIAE (second) | TCM{1\|2\|3\|4\|5\|6}_BIAE_S | <NR2> |
| | | (frame) | TCM{1\|2\|3\|4\|5\|6}_BIAE | <NR1> |
| | | TCMi-BDI (second) | TCM{1\|2\|3\|4\|5\|6}_BDI_S | <NR2> |
| | | (frame) | TCM{1\|2\|3\|4\|5\|6}_BDI | <NR1> |
| | | TCMi-IAE (second) | TCM{1\|2\|3\|4\|5\|6}_IAE_S | <NR2> |
| | | (frame) | TCM{1\|2\|3\|4\|5\|6}_IAE | <NR1> |
| | | TCMi-LTC (second) | TCM{1\|2\|3\|4\|5\|6}_LTC_S | <NR2> |
| | | (frame) | TCM{1\|2\|3\|4\|5\|6}_LTC | <NR1> |

Table A.4-1 Measurement Items and ID List (Cont'd)

| Tab | | Measurement Item | ID | Response Data Format |
|---|---|---|---|---|
| Statistics (Cont'd) | | | | |
| | TCM(Cont'd) | TCMi-BIP8 (count) | TCM{1\|2\|3\|4\|5\|6}_BIP8 | <NR1> |
| | | (rate) | TCM{1\|2\|3\|4\|5\|6}_BIP8_R | <NR3> |
| | | TCMi-BEI (count) | TCM{1\|2\|3\|4\|5\|6}_BEI | <NR1> |
| | | (rate) | TCM{1\|2\|3\|4\|5\|6}_BEI_R | <NR3> |
| | ODU2e, ODU0 | LOFLOM | LOFLOM | <NR2> |
| | | OOF (second) | OOF_L_S | <NR2> |
| | | (frame) | OOF_L | <NR1> |
| | | OOM (second) | OOM_L_S | <NR2> |
| | | (frame) | OOM_L | <NR1> |
| | | ODU-AIS (second) | AIS_ODU_L_S | <NR2> |
| | | (frame) | AIS_ODU_L | <NR1> |
| | | ODU-OCI (second) | OCI_L_S | <NR2> |
| | | (frame) | OCI_L | <NR1> |
| | | ODU-LCK (second) | LCK_L_S | <NR2> |
| | | (frame) | LCK_L | <NR1> |
| | | PM-TIM (second) | PM_TIM_L_S | <NR2> |
| | | (frame) | PM_TIM_L | <NR1> |
| | | PM-BDI (second) | PM_BDI_L_S | <NR2> |
| | | (frame) | PM_BDI_L | <NR1> |
| | | PLM | PLM_L | <NR1> |
| | | CSF | CSF_L | <NR1> |
| | | FAS | FAS_L | <NR1> |
| | | PM-BIP8 (count) | PM_BIP8_L | <NR1> |
| | | (rate) | PM_BIP8_L_R | <NR3> |
| | | PM-BEI (count) | PM_BEI_L | <NR1> |
| | | (rate) | PM_BEI_L_R | <NR3> |

Table A.4-1 Measurement Items and ID List (Cont'd)

| Tab | | Measurement Item | ID | Response Data Format |
|---|---|---|---|---|
| Statistics (Cont'd) | | | | |
| | GMP | Tx Inc 1 | GMP_INC1_TX | <NR1> |
| | | Rx Inc 1 | GMP_INC1 | |
| | | Tx Inc 2 | GMP_INC2_TX | <NR1> |
| | | Rx Inc 2 | GMP_INC2 | |
| | | Tx Dec 1 | GMP_DEC1_TX | <NR1> |
| | | Rx Dec 1 | GMP_DEC1 | |
| | | Tx Dec 2 | GMP_DEC2_TX | <NR1> |
| | | Rx Dec 2 | GMP_DEC2 | |
| | | Tx Inc >2 | GMP_INC3_TX | <NR1> |
| | | Rx Inc >2 | GMP_INC3 | |
| | | Tx Dec >2 | GMP_DEC3_TX | <NR1> |
| | | Rx Dec >2 | GMP_DEC3 | |
| | | Rx Inc Over | GMP_INC_O | <NR1> |
| | | Rx Dec Over | GMP_DEC_O | |
| | | Offset (ppm) | FREQ_D_P | <NR2> |
| | | CRC8 Error | GMP_CRC8 | <NR1> |
| | | CRC5 Error | GMP_CRC5 | <NR1> |
| | GMP ( ODU2e, ODU0) | Tx Inc 1 | GMP_INC1_TX_L | <NR1> |
| | | Rx Inc 1 | GMP_INC1_L | |
| | | Tx Inc 2 | GMP_INC2_TX_L | <NR1> |
| | | Rx Inc 2 | GMP_INC2_L | |
| | | Tx Dec 1 | GMP_DEC1_TX_L | <NR1> |
| | | Rx Dec 1 | GMP_DEC1_L | |
| | | Tx Dec 2 | GMP_DEC2_TX_L | <NR1> |
| | | Rx Dec 2 | GMP_DEC2_L | |
| | | Tx Inc >2 | GMP_INC3_TX_L | <NR1> |
| | | Rx Inc >2 | GMP_INC3_L | |
| | | Tx Dec >2 | GMP_DEC3_TX_L | <NR1> |
| | | Rx Dec >2 | GMP_DEC3_L | |
| | | Rx Inc Over | GMP_INC_O_L | <NR1> |
| | | Rx Dec Over | GMP_DEC_O_L | |
| | | Offset (ppm) | FREQ_D_P_L | <NR2> |
| | | CRC8 Error | GMP_CRC8_L | <NR1> |

Table A.4-1 Measurement Items and ID List (Cont'd)

| Tab | | Measurement Item | ID | Response Data Format |
|---|---|---|---|---|
| Statistics (Cont'd) | | | | |
| | GFP-T | Invalid GFP-T Frame | GFP_INV_FRM | <status> |
| | | CSF Signal (second) | GFP_CSF_SIG_S | <NR2> |
| | | (frame) | GFP_CSF_SIG | <NR1> |
| | | CSF Sync (second) | GFP_CSF_SYNC_S | <NR2> |
| | | (frame) | GFP_CSF_SYNC | <NR1> |
| | | SSF (second) | GFP_SSF | <NR2> |
| | | Superblock CRC | GFP_SCRC | <NR1> |
| | | Correctable cHEC | GFP_CHEC_C | <NR1> |
| | | Uncorrectable cHEC | GFP_CHEC_U | <NR1> |
| | | Correctable tHEC | GFP_THEC_C | <NR1> |
| | | Uncorrectable tHEC | GFP_THEC_U | <NR1> |
| | PCS Lane | Marker Lock | AMLOCK | <status> |
| | | Marker Map | MMAP | <NR1> |
| | | BIP Error | BIP_CNT | <NR1> |
| | | Invalid Alignment Marker | IAM | <NR1> |
| | Ethernet | Alignment Status | ALIGNMENT | <status> |
| | | High-BER | HIBER | <status> |
| | | Sync Header Lock | SHLOCK | <status> |
| | | Invalid Sync Header | ISH | <NR1> |
| | | Invalid Alignment Marker | IAM | <NR1> |
| | | Invalid Block | IBLOCK | <NR1> |
| | | 66B Error | ERR_66B | <NR1> |
| | | 10B Error | ERR_10B | <NR1> |
| | | Tx LF | TX_LF | <NR1> |
| | | Rx LF | RX_LF | |
| | | Tx RF | TX_RF | <NR1> |
| | | Rx RF | RX_RF | |
| | | Error Signals | RX_ERR_SIG | <NR1> |
| | | Tx Errored Bytes | TEBYTE | <NR1> |
| | | Rx Errored Bytes | REBYTE | |
| | | Tx Good Bytes | TGBYTE | <NR1> |
| | | Rx Good Bytes | RGBYTE | |

Table A.4-1 Measurement Items and ID List (Cont'd)

| Tab | | Measurement Item | ID | Response Data Format |
|---|---|---|---|---|
| Statistics (Cont'd) | | | | |
| | Ethernet (Cont'd) | Tx FCS Errors | TFCS | <NR1> |
| | | Rx FCS Errors | RFCS | |
| | | Tx Fragments | TFRAGMENT | <NR1> |
| | | Rx Fragments | RFRAGMENT | |
| | | Tx Oversize & FCS Errors | TOAF | <NR1> |
| | | Rx Oversize & FCS Errors | ROAF | |
| | | Tx Undersize | TUNDERSIZE | <NR1> |
| | | Rx Undersize | RUNDERSIZE | |
| | | Tx Oversize | TOVERSIZE | <NR1> |
| | | Rx Oversize | ROVERSIZE | |
| | | Tx Good Frames | TGFRAME | <NR1> |
| | | Rx Good Frames | RGFRAME | |
| | | Tx Rate (bit/s) | TBPS | <NR1> |
| | | Rx Rate (bit/s) | RBPS | |
| | | Tx Rate (%) | TRATE | <NR2> |
| | | Rx Rate (%) | RRATE | |
| | Test Pattern | Pattern Sync Loss (second) | PSLOSS | <NR2> |
| | | Bit Errors (bit) | BER_CNT | <NR1> |
| | | Bit Errors (rate) | BER_RATE | <NR3> |
| Delay*1 | | PM Current (µs) | PM_DELAY | <NR2> |
| | | TCMi Current (µs) | TCM{1\|2\|3\|4\|5\|6}_DELAY | <NR2> |
| APS*2 | | Current (ms) | APS_TIME | <NR2> |

*1: The values of Max, Min, Average, History, Tx Delay Frame, and Rx Delay Frame cannot be read with the remote control.

*2: The values of Max, Min, Average, and History cannot be read with the remote control.

* For the IDs of Opt tab, refer to Section A.3 "40/100GbE Application".

## A.5 No Frame Application

**Table A.5-1 Measurement Items and ID List**

| Tab | Measurement Item | ID | Response Data Format |
|---|---|---|---|
| Statistics | Elapsed Time (s) | `ELAPSED` | <NR1> |
| | Clock Source Loss | `CSLOSS` | <status> |
| | Pattern Sync Loss (s) | `PSLOSS` | <NR2> |
| | Bit Errors (Count) | `BER_CNT` | <NR1> |
| | Bit Errors (rate) | `BER_RATE` | <NR3> |
| | Frequency (Hz) | `RX_FREQ` | <NR1> |
| | Frequency (ppm) | `RX_FREQ_D` | <NR1> |
| | CDR Unlock | `CUNLOCK` | <status> |
| | Clock Count (bit) * | `CLK_CNT` | <NR1> |

* Not displayed on the MD1260A screen
Clock Count (bit) is the number of received bits to be used for Bit Error Rate calculation.

$$\mathrm{BER} = \frac{ErrorBitCount}{\mathrm{Re}\,ceivedBitCount}$$

For the IDs of Opt tab, refer to Section A.3 "40/100GbE Application".

# *Appendix B   Message Codes*

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 3.3 Explanation of Messages
- When sending the command not corresponding to the application
- When ? is not appended to a query.

### 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 200 is set even though the setting range of the transmission clock frequency offset is –20 to +120
- When message cannot be executed in current equipment status
  Example: When sending message for setting the MDIO register of CFP to instrument without CFP function
- When a parameter not specified by the application is set.

### Device dependant error

Bit 3 of the standard event status register is set when the following errors occur.

- When a non-existent file or a file not loaded by :MMEMory:RECall is set
- When an extension is included in the parameter file name at :MMEMory:RECall

# *Appendix C Change Information on Command*

This appendix explains the command changed along with upgrading the software.

## C.1 Change Information of Ver.2.0

### C.1.1 No used command

Not used and alternative commands are listed in the following table.

**Table C.1.1-1 Not Used and Alternative Command**

| No used command | Alternative command |
|---|---|
| :CALCulate:COUNter:ETHer:BERRors | :ROUTe:BERT |
| :SOURce:STReam:DURation:FRAMes | :SOURce:STReam:DURation:REPeat:COUNt |
| :SOURce:STReam:HEADer:ETHer:SA | :SOURce:STReam:HEADer:PATTern |
| :SOURce:STReam:HEADer:ETHer:DA | :SOURce:STReam:HEADer:PATTern |
| :SOURce:STReam:HEADer:ETHer:TYPE | :SOURce:STReam:HEADer:PATTern |

:CALCulate:COUNter:ETHer:BERRors
When setting/querying the execution of the counter of the Bit Errors Count in the 40GbE/100GbE application, use :ROUTe:BERT.

:SOURce:STReam:DURation:FRAMes
When setting/querying the execution of the number of transmission frames in the 40GbE/100GbE application, use : SOURce:STReam:DURation:REPeat:COUNt.

:SOURce:STReam:HEADer:ETHer:DA
:SOURce:STReam:HEADer:ETHer:SA
:SOURce:STReam:HEADer:ETHer:TYPE
When setting/ querying the Header MAC address, MAC address transmission source, and the Type field value of the Ethernet frame to be sent, use : SOURce:STReam:HEADer:PATTern.

## C.1.2 Parameter Changed Command

The following command parameters were changed.

:CALCulate:DATA

The parameter is added and deleted to set it to < item >.

Added parameter:

TX_LF,RX_LF,TX_RF,RX_RF,RX_ERR_SIG,TX_TIME,RX_PAUSE,RX_TRIGGER,TX_FSIZE,RX_FSIZE,TX_GSIZE,RX_GSIZE,TX_TFRAME,RX_TFRAME,RX_SEQ_ERR,LAT_CURR,LAT_MAX,LAT_MIN,TX_FREQ,TX_FREQ_D

Deleted parameter:

TXRX

Appendix A measurement item list

:SOURce:STReam:CONTrol:VALue

Parameter format: changes from < NR1> to < NR2>.

:SOURce:STReam:DURation:TYPE

Before changing : SOURce:STReam:DURation:TYPE FRAME
After changing : SOURce:STReam:DURation:TYPE REPEAT

:SOURce:EALarm:LANE

When sending : SOURce:EALarm:LANE? at the measurement of 10 lanes, the response data is changed from 20 to 10 digits.

Before changing 20 digits, Example : #B10000000000000000000
After changing 10 digits, Example: #B1000000000

## C.1.3 Added Command

The following commands were added.

Capture command
:CALCulate:CAPTure:DATA:RXC
:CALCulate:CAPTure:DATA:RXD
:CALCulate:CAPTure:DATA:SIZE
:CALCulate:CAPTure:DATA:TRIGger:POSition
:CALCulate:CAPTure:STARt
:CALCulate:CAPTure:STARt:EVENt
:CALCulate:CAPTure:STATus
:CALCulate:CAPTure:STOP
:CALCulate:CAPTure:TRIGgered
:CALCulate:TRIGger:CONDition
:MMEMory:STORe:CAPTure
:MMEMory:STORe:CAPTure:ITEM

Lane Mapping command
:SOURce:MAPPing:LANE

Port command
:ROUTe:FCONtrol
:ROUTe:LFS:REPLy

Relative Skew command
:SOURce:SKEW:TYPE

Stream command
:SOURce:STReam:BURSt:CONTrol:VALue
:SOURce:STReam:BURSt:ENABle
:SOURce:STReam:BURSt:SIZE
:SOURce:STReam:COUNt
:SOURce:STReam:DURation:REPeat:COUNt
:SOURce:STReam:ENABle
:SOURce:STReam:ERRor:TYPE
:SOURce:STReam:FSIZe
:SOURce:STReam:HEADer:PATTern
:SOURce:STReam:HEADer:VARiable1:RANGe
:SOURce:STReam:HEADer:VARiable1:TYPE
:SOURce:STReam:HEADer:VARiable2:RANGe
:SOURce:STReam:HEADer:VARiable2:TYPE

:SOURce:STReam:ID
:SOURce:STReam:TFRame:ENABle
:SOURce:STReam:TFRame:FID
:SOURce:STReam:TYPE

Test Frame command
:CALCulate:COUNter:GAP
:CALCulate:COUNter:SERRor

Summary status command
:CALCulate:LINK[:AUNit]

Multiport function command
:CALCulate:COUNter:STARt:AUNit
:CALCulate:COUNter:STARt:EVENt:AUNit
:CALCulate:COUNter:STATus:AUNit
:CALCulate:COUNter:STOP:AUNit
:CALCulate:EALarm:AUNit
:CALCulate:LINK:AUNit
:MMEMory:INITialize
:MMEMory:RECall:AUNit
:MMEMory:STORe:AUNit
:SOURce:EALarm:STARt:AUNit
:SOURce:EALarm:STARt:EVENt:AUNit
:SOURce:EALarm:STATus:AUNit
:SOURce:EALarm:STOP:AUNit
:SOURce:STReam:STARt:AUNit
:SOURce:STReam:STARt:EVENt:AUNit
:SOURce:STReam:STATus:AUNit
:SOURce:STReam:STOP:AUNit
:UENTry:ID
:UENTry:LIST

## C.2 Change Information of Ver. 3.0

### C.2.1 No used command

The following command has been discarded.

:CALCulate:LINK[:AUNit]
To query the status area Link lamp status,
use :CALCulate:DATA? LINK command.

### C.2.2 Changed Command

The following command has been changed.

Table C.2.2-1 Changed Command

| Before change | After change |
|---|---|
| :CALCulate:CAPTure:TRIGgered | :CALCulate:CAPTure:TRIGger |

### C.2.3 Parameter Changed Command

The following command parameters have been changed.

Refer to Table 3.1-1 "Command Description Method" for the parameter description method.

:CALCulate:MONitor:OTU:DATA
Parameters to specify the column position and data length have been added.

:CALCulate:DATA
The parameters to set to <item> have been added and deleted.
Refer to Appendix A "Measurement Item List."
Replaced parameters:

MLOFOTL_SEC → LOF_LLD
MOOF_FRM → OOF_LLD
MFAS_CNT → FAS_LLD
MOOR_FRM → OOR
OLOF_SEC → LOF
OOOF_FRM → OOF
OOOF_SEC → OOF_S
OFAS_CNT → FAS
OLOM_SEC → LOM
OOOM_FRM → OOM
OOOM_SEC → OOM_S
SMBIP8_CNT → SM_BIP8

SMBIP8_RATE → SM_BIP8_R
PMBIP8_CNT → PM_BIP8
PMBIP8_RATE → PM_BIP8_R

It is possible to be controlled by the parameter before changed for keeping compatibility, but do not use this parameter.

Added parameters:

AIS_C[_S],AIS_ODU[_L][_S],CSF,ERR_10B,
ERR_66B,FAS_L,FREQ_D_P[_L],GFP_CHEC_C,
GFP_CHEC_U,GFP_CSF,GFP_CSF_SIG,GFP_CSF_SYNC,
GFP_INV_FRM,GFP_SCRC,GFP_SSF,GFP_THEC_C,
GFP_THEC_U, GMP_CRC5,GMP_CRC8[_L],GMP_DEC_O[_L],
GMP_DEC{1|2|3}[_L],GMP_DEC{1|2|3}_TX[_L],
GMP_INC_O[_L],GMP_INC{1|2|3}[_L],GMP_INC{1|2|3}_TX[_L],
LCK[_L][_S],LOFLOM,MSIM,OCI[_L][_S],OOF_L,OOM_L,
PLM[_L],PM_BDI[_L][_S],PM_BEI[_L][_R],PM_BIP8_L,
PM_TIM[_L][_S],SM_BDI[_S],SM_BEI[_R],SM_BIAE[_S],
SM_IAE[_S],SM_TIM,TCM{1|2|3|4|5|6}_BDI[_S],
TCM{1|2|3|4|5|6}_BEI[_R],TCM{1|2|3|4|5|6}_BIAE[_S],
TCM{1|2|3|4|5|6}_BIP8[_R],TCM{1|2|3|4|5|6}_IAE[_S],
TCM{1|2|3|4|5|6}_LTC[_S],TCM{1|2|3|4|5|6}_TIM

:MMEMory:CATalog

Replaced parameters:

100GE → E100G
100GE_N → E100G_N
40GE → E40G
40GE_N → E40G_N

Added parameters: OTU4_E,OTU4_ODU0,
OTU4_ODU0_E,OTU4_ODU2E,OTU4_ODU2E_E

Deleted parameters: ALL

:ROUTe:MODE

Added parameters: THROUGH

:SOURce:MAPPing

Replaced parameters:

100GE → E100G
100GE_10N → E100G_10N
100GE_20N → E100G_20N
40GE → E40G
40GE_N,ALL → E40G_N

Added parameters:

NONE,OTU4_E,OTU4_ODU0,
OTU4_ODU0_E,OTU4_ODU2E,OTU4_ODU2E_E

:SOURce:EALarm:TIMing:RATE

The response format has been changed to <NR3>.

:SOURce:EALarm:TIMing:TYPE

Replaced parameters: CRATE,RRATE → RATE

:SOURce:EALarm:TYPE

Replaced parameters:

FAS_MLD → FAS_LLD
FAS_OTU → FAS
LOFOTL → OOF_LLD
LOF_OTU → OOF
LOM → OOM
OOR → LOR

Added parameters:

AIS_C,AIS_ODU[_L],BIT_ALL_C,
BIT_ALL_U,CSF,ERR_10B,ERR_66B,GFP_CHEC,GFP_SCRC,
GFP_THEC,GMP_CRC5,GMP_CRC8,GMP_JC1,GMP_JC12,
GMP_JC2,LCK[_L],OCI[_L],PM_BDI[_L],PM_BEI[_L],
PM_BIP8_L,PM_TIM[_L],SM_BDI,SM_BEI,SM_BIAE,SM_IAE,
SM_TIM,TCM{1|2|3|4|5|6}_BDI,TCM{1|2|3|4|5|6}_BDI,
TCM{1|2|3|4|5|6}_BIAE,TCM{1|2|3|4|5|6}_BIP8,
TCM{1|2|3|4|5|6}_IAE,TCM{1|2|3|4|5|6}_LTC,
TCM{1|2|3|4|5|6}_TIM

## C.2.4 Added Commands

The following commands are added.

Command related to APS
:CALCulate:APS:ERRor:FREE
:CALCulate:APS:STARt[:AUNit]
:CALCulate:APS:STARt:EVENt[:AUNit]
:CALCulate:APS:STATus[:AUNit]
:CALCulate:APS:STOP[:AUNit]
:CALCulate:APS:TRIGger

Command related to Capture
:CALCulate:CAPTure:DATA
:CALCulate:CAPTure:TRIGger:Manual
:CALCulate:CAPTure:TRIGger:POSition
:CALCulate:CAPTure:TRIGger:TYPE
:CALCulate:CAPTure:TYPE

Command related to Clock
:SOURce:CLOCk:PAYlaod:OFFSet[:L]

Command related to Counter
:SENSe:PLM[:L]:Pattern
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}
:SENSe:TIM:{SM|PM[:L]|TCM{1|2|3|4|5|6}}:PATTern:{DAPI|SAPI}

Command related to Data Monitor
:CALCulate:MONitor:OTU:TYPE

Command related to Delay
:CALCulate:DELay:STARt[:AUNit]
:CALCulate:DELay:STARt:EVENt[:AUNit]
:CALCulate:DELay:STATus[:AUNit]
:CALCulate:DELay:STOP[:AUNit]

Command related to Error/Alarm
:SOURce:EALarm:BIT
:SOURce:EALarm:FAS:EXCLude
:SOURce:EALarm:SUBRow

Command related to GFP-T
:SOURce:GFP:CSF:REPLacement
:SOURce:GFP:{PTI|UPI}

Command related to OH Preset

:SOURce:OTN:OH[:L]

Command related to Port

:ROUTe:ETHer:NEGptiation:AUTO

:ROUTe:MODE:THRough:OVERwirte:RANGe

:ROUTe:MODE:THRough:TYPE

:SOURce:OTN:FEC

Command related to Selector

:SYSTem:CONFig

:SYSTem:STATus

Command related to Stream

:SOURce:STReam:DURation:FRAMes

:SOURce:STReam:HEADer:ETHer:DA

:SOURce:STReam:HEADer:ETHer:SA

:SOURce:STReam:HEADer:ETHer:TYPE

Command related to TP/TS

:SENSe:ODTU:MAIN:DETect

:SENSe:ODTU:MAIN:TS

:SOURce:ODTU:DUMMy:PATTern

:SOURce:ODTU:MAIN:TP

:SOURce:ODTU:MAIN:TS

## C.3 Change Information of Ver.3.1

### C.3.1 Added Commands

The following commands are added.

Command related to Opt
:SOURce:CFP:OPTical:OFF
:SOURce:CFP:OPTical:ON
:SOURce:CFP:OPTical:STATus

Command related to Port
:ROUTe:MPLStp:CWORd

Command related to Stream
:MMEMory:STReam:RECall
:SOURce:STReam:HEADer
:SOURce:STReam:HEADer:VARiable{3|4|5}

Command related to Test Frames
:CALCulate:COUNter:FLOW:FIELd
:CALCulate:COUNter:FLOW:FIELd:ID
:CALCulate:COUNter:FLOW:FIELd:NFID
:CALCulate:COUNter:FLOW:TYPE

### C.3.2 Changed Command

The following commands have been changed.

:SOURce:STReam:HEADer:VARiable1:TYPE

| Before change | After change |
|---|---|
| Sets the Ethernet Destination MAC Address | Sets the Modifiers Field1 Destination MAC Address |

:SOURce:STReam:HEADer:VARiable2:TYPE

| Before change | After change |
|---|---|
| Sets the Ethernet Source MAC Address | Sets the Modifiers Field2 Destination MAC Address |

## C.3.3 Parameter Changed Command

The following command parameters have been changed.

Refer to Table 3.1-1 "Command Description Method" for the parameter description method.

:CALCulate:DATA

The parameters to set to <item> have been added.
Refer to Appendix A "Measurement Item List".

Added parameters:

CLK_CNT,RBCAST_BYTE,RBCAST_FRAME,
RMCAST_BYTE,RMCAST_FRAME,RMPLSTP,RX_TBPS,
RX_TBPS_O,RX_TFRAME_O,TBCAST_BYTE,TBCAST_FRAME,
TMCAST_BYTE,TMCAST_FRAME,TMPLSTP,TX_TBPS,
TX_TBPS_O,TX_TFRAME_O

:SOURce:STReam:HEADer:VARiable1:TYPE
:SOURce:STReam:HEADer:VARiable2:TYPE

Added parameters:

DECREMENT

## C.4 Change Information of Ver.3.2

### C.4.1 Added Commands

The following commands are added.

Command related to Port
:ROUTe:VLAN:NUM
:ROUTe:VLAN:VALue
:SENSe:OTN:FEC

Command related to Stream
:SOURce:STReam:RESolve:IP:TARGet
:SOURce:STReam:RESolve:IPV4:ROUTer
:SOURce:STReam:RESolve:IPV6:ROUTer
:SOURce:STReam:RESolve:MAC:RETRy
:SOURce:STReam:RESolve:MAC:TIMeout
:SOURce:STReam:RESolve:PING:TRY
:SOURce:STReam:RESolve:PING:PAYLoad
:SOURce:STReam:RESolve:PING:TIMeout
:SOURce:STReam:RESolve:RESult
:SOURce:STReam:RESolve:STARt
:SOURce:STReam:RESolve:STARt:EVENt
:SOURce:STReam:RESolve:STATus
:SOURce:STReam:RESolve:STOP
:SOURce:STReam:RESolve:TYPE

Command related to Protocol
:PROTocol:ARPNa:REPLy:ENABle
:PROTocol:GARPns:DURation:TYPE
:PROTocol:GARPns:ENABle
:PROTocol:GARPns:INTerval
:PROTocol:GARPns:STARt
:PROTocol:GARPns:STATus
:PROTocol:GARPns:STOP
:PROTocol:GARPns:TYPE
:PROTocol:PING:COUNt
:PROTocol:PING:DST:IPV4
:PROTocol:PING:DST:IPV6
:PROTocol:PING:DST:MAC
:PROTocol:PING:IPMode
:PROTocol:PING:MACResolve
:PROTocol:PING:PAYLoad
:PROTocol:PING:PSIZe:VALUe
:PROTocol:PING:REPLy:ENABle

:PROTocol:PING:RESult
:PROTocol:PING:SRC:IPV4
:PROTocol:PING:SRC:IPV6
:PROTocol:PING:STARt
:PROTocol:PING:STARt:ENABle
:PROTocol:PING:STATus
:PROTocol:PING:STOP
:PROTocol:PING:TIMeout
:PROTocol:PING:VLAN

## C.4.2 Parameter Changed Command

The following command parameters have been changed.

Refer to Table 3.1-1 "Command Description Method" for the parameter description method.

:CALCulate:COUNter:FLOW:FIELd
The parameters to set to <field> have been added.
Added parameters:
ARP_SMAC, ARP_SIP, ARP_TMAC, ARP_TIP, ARP_OPE,
ICMPV4_CODE, ICMPV4_EC_ID ICMPV4_EC_SEQ,
ICMPV6_CODE, ICMPV6_EC_ID ICMPV6_EC_SEQ,
ICMPV6_NSNA_TADDR ICMPV6_NSNA_SADDR

:CALCulate:DATA
The parameters to set to <item> have been added.
Refer to Appendix A "Measurement Item List".

:SOURce:STReam:HEADer
The parameters to set to <field> have been added.
Added parameters:
ARP,ICMPV4_ECHO,ICMPV6_ECHO,ICMPV6_NS,ICMPV6_NA

:SOURce:STReam:HEADer:VARiable{3|4|5}
The parameters to set to <field> have been added.
Added parameters:
ARP_SMAC, ARP_SIP, ARP_TMAC, ARP_TIP, ARP_OPE,
ICMPV4_CODE, ICMPV4_EC_ID ICMPV4_EC_SEQ,
ICMPV6_CODE, ICMPV6_EC_ID ICMPV6_EC_SEQ,
ICMPV6_NSNA_TADDR ICMPV6_NSNA_SADDR

# *Appendix D Correspondence between Application and Command*

The following table shows the correspondence between application and command.

✓: Supported

–: Not supported

**Table D-1 Correspondence between Application and Command**

| Application / Command | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :CALCulate:APS:ERRor:FREE | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:APS:STARt[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:APS:STARt:EVENt[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:APS:STATus[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:APS:STOP[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:APS:TRIGger | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:DATA | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:DATA:RXC | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:CAPTure:DATA:RXD | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:CAPTure:DATA:SIZE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:DATA:TRIGger:POSition | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:STARt | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:STARt:EVENt | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:STATus | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:STOP | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:TRIGger:MANual | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:TRIGger:POSition | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:TRIGger:TYPE | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:TRIGger | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:CAPTure:TYPE | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:COUNter:ETHer:OVERsize | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :CALCulate:COUNter:FLOW:FIELd | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:COUNter:FLOW:FIELd:ID | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |

Table D-1 Correspondence between Application and Command (Cont'd)

| Command \ Application | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :CALCulate:COUNter:FLOW:FIELd:NFID | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:COUNter:FLOW:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:COUNter:GAP | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:COUNter:SERRor | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :CALCulate:COUNter:STARt[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:COUNter:STARt:EVENt[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:COUNter:STATus[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:COUNter:STOP[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:DATA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:DATA:TYPE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:DELay:STARt[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:DELay:STARt:EVENt[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:DELay:STATus[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:DELay:STOP[:AUNit] | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:EALarm[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :CALCulate:MONitor:OTU:COLumn | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:MONitor:OTU:DATA | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:MONitor:OTU:TYPE | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :CALCulate:TRIGger:CONDition | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :MDIO:READ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MDIO:WRITe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:CATalog | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:INITialize | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:LOG:FNAMe | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:LOG:ITEM | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:LOG:PREFix | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:LOG:STARt | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:LOG:STATus | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:LOG:STOP | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Table D-1 Correspondence between Application and Command (Cont'd)

| Command \ Application | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :MMEMory:LOG:TIMing | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:RECall[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:STORe[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :MMEMory:STORe:CAPTure | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :MMEMory:STORe:CAPTure:ITEM | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :MMEMory:STReam:RECall | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:ARPNa:REPLy:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:DURation:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:INTerval | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:STARt | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:STATe | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:STOP | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:GARPns:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:COUNt | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:DEST:IPV4 | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:DEST:IPV6 | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:DEST:MAC | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:IPMode | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:PAYLoad | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:PSIZe:VALUe | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:REPLy:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:RESult | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:SRC:IPV4 | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:SRC:IPV6 | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:SRC:MAC | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:STARt | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:STARt:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:STATus | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:STOP | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |

**Table D-1 Correspondence between Application and Command (Cont'd)**

| Command \ Application | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :PROTocol:PING:TIMeout | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :PROTocol:PING:VLAN | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :ROUTe:BERT | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :ROUTe:ETHer:NEGotiation:AUTO | – | – | – | – | – | – | – | ✓ | – | – | – | – | – |
| :ROUTe:FCONtrol | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :ROUTe:LFS:REPLy | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :ROUTe:MODE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :ROUTe:MODE:THRough:OVERwrite:RANGe | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :ROUTe:MODE:THRough:TYPE | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :ROUTe:MPLStp:CWORd | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :ROUTe:VLAN:NUM | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :ROUTe:VLAN:VALue | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SENSe:MAPPing | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SENSe:ODTU:MAIN:DETect | – | – | – | – | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SENSe:ODTU:MAIN:TS | – | – | – | – | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SENSe:OTN:FEC | – | – | ✓ | – | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SENSe:PLM[:L]:PATTern | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SENSe:TIM:{SM\|PM[:L]\|TCM{1\|2\|3\|4\|5\|6}} | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SENSe:TIM:{SM\|PM[:L]\|TCM{1\|2\|3\|4\|5\|6}}:PATTern:{DAPI\|SAPI} | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SENSe:TPATtern:INVert | – | – | ✓ | – | ✓ | – | ✓ |  | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SENSe:TPATtern:TYPE | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SENSe:TPATtern:WORD | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SENSe:TRANsceiver:EQUalizer:CONTrol | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SENSe:TRANsceiver:EQUalizer:DCGain | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

**Table D-1 Correspondence between Application and Command (Cont'd)**

| Application<br>Command | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :SOURce:CFP:OPTical:OFF | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CFP:OPTical:ON | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CFP:OPTical:STATus | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CLOCk | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CLOCk:FREQuency:OFFSet | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CLOCk:OUTPut:DIVide | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CLOCk:OUTPut:M10 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:CLOCk:PAYLoad:OFFSet[:L] | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:EALarm:BIT | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:EALarm:FAS:EXCLude | – | – | ✓ | – | – | – | – | – | ✓ | – | – | – | – |
| :SOURce:EALarm:LANE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:EALarm:STARt[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:EALarm:STARt:EVENt[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:EALarm:STATus[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:EALarm:STOP[:AUNit] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:EALarm:SUBRow | – | – | ✓ | – | – | – | – | – | ✓ | – | – | – | – |
| :SOURce:EALarm:TIMing:BURSt | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:EALarm:TIMing:ERRor | ✓ | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:EALarm:TIMing:NORMal | ✓ | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:EALarm:TIMing:RATE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:EALarm:TIMing:TYPE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:EALarm:TYPE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:GFP:CSF:REPLacement | – | – | – | – | – | – | – | ✓ | – | – | – | – | – |
| :SOURce:GFP:{PTI\|UPI} | – | – | – | – | – | – | – | ✓ | – | – | – | – | – |
| :SOURce:MAPPing | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:MAPPing:LANE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:ODTU:DUMMy:PATTern | – | – | – | – | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:ODTU:MAIN:TP | – | – | – | – | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:ODTU:MAIN:TS | – | – | – | – | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |

Table D-1 Correspondence between Application and Command (Cont'd)

| Application / Command | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :SOURce:OTN:FEC | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:OTN:OH[:L] | – | – | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:SKEW:BIT | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:SKEW:LANE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:SKEW:NS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:SKEW:TYPE | ✓ | – | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – | – |
| :SOURce:STReam:BURSt:CONTrol:VALue | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:BURSt:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:BURSt:SIZE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:CONTrol:RANGe | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:CONTrol:TYPE | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:CONTrol:VALue | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:COUNt | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:DURation:FRAMes | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:DURation:REPeat:COUNt | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:DURation:TYPE | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:ERRor:TYPE | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:FSIZe | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:FSIZe:RANG | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:FSIZe:TYPE | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:FSIZe:VALue | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:HEADer | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:HEADer:ETHer:DA | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:HEADer:ETHer:SA | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:HEADer:ETHer:TYPE | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:HEADer:PATTern | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |

**Table D-1 Correspondence between Application and Command (Cont'd)**

| Application / Command | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :SOURce:STReam:HEADer:VARiable1:RANGe | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:HEADer:VARiable1:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:HEADer:VARiable2:RANGe | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:HEADer:VARiable2:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:HEADer:VARiable{3\|4\|5} | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:ID | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:IP:TARGet | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:IPV4:ROUTer | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:IPV6:ROUTer | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:MAC:RETRy | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:MAC:TIMeout | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:PING:TRY | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:PING:PAYLoad | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:PING:TIMeout | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:RESult | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:STARt | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:STARt:EVENt | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:STATus | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:STOP | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:RESolve:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:STARt[:AUNit] | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:STARt:EVENt[:AUNit] | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:STATus[:AUNit] | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:STOP[:AUNit] | ✓ | ✓ | – | ✓ | – | ✓ | – | ✓ | – | – | – | – | – |
| :SOURce:STReam:TFRame:ENABle | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:STReam:TFRame:FID | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |

**Table D-1 Correspondence between Application and Command (Cont'd)**

| Application / Command | 100GbE | 40GbE | ODU4-PRBS | ODU4-100GbE | ODU4-ODTU4.8-ODU2e-PRBS | ODU4-ODTU4.8-ODU2e-10GbE | ODU4-ODTU4.1-ODU0-PRBS | ODU4-ODTU4.1-ODU0-GbE | ODU3- PRBS | 100GbE No Frame | 40GbE No Frame | OTU4 No Frame | OTU3 No Frame |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| :SOURce:STReam:TYPE | ✓ | ✓ | – | – | – | – | – | – | – | – | – | – | – |
| :SOURce:TPATtern:INVert | – | – | ✓ | – | ✓ | – | ✓ |  | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:TPATtern:TYPE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:TPATtern:WORD | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | – | – | – | – |
| :SOURce:TRANsceiver:EMPHasis:FIRSt | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:TRANsceiver:EMPHasis:PRE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:TRANsceiver:EMPHasis:SECond | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SOURce:TRANsceiver:VOD | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:CONFig | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:DATE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:ERRor | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:PRINt:COPY | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:STATus | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:TERMination | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:TIME | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :SYSTem:VERSion | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :UENTry:ID | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| :UENTry:LIST | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

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.

# *Appendix E Sample Program*

This appendix describes the sample program using the Tera Term macro function.

## E.1 Executing sample Programs

1. Start the test editor such as the Windows memo pad.
2. Copy the sample program in this manual.
3. Past the copied sample program to the test editor.
4. The file can be saved in Tera Term macro format (with ttl extension).
5. Start Tera Term.
6. Confirm that it can be communicated with the MD1260A referring to Section 2.7.1 When using Ethernet (Windows 7/Vista) "
7. Click [Control] →[Macro] from the menu of Tera Term.
8. Open the file selection window.
   Select the file saved at step 4.

For the other execution method of macro, refer to the help of Tera Term.

## E.2 Example 1: Setting Stream and Reading Counter

This sample program sets the 100 GbE or 40 GbE application stream and reads out the counter value.

Processing Flow

1. The Unit ID defines One MD1260A as the control target.
2. The application is queried by :SOURce:MAPPing?.
   When the application is 100GbE or 40 GbE, the results of the message box are displayed.
3. Set a port as follows.
   Mode: Normal
   Bert: Off
   LFS Reply: On
   Flow Control:On
   Rx MPLS-TP Control Word : Off
4. Query the link lamp in the status area.
5. Set the stream 1 through 16 as follows.
   Frame size: Random, 128 to 1600 (bytes)
   Gap size: Random, 2000 to 20000 (bytes)
6. Enable the stream1 through 16 and send the stream.
7. Start counting.
8. Query whether to start counting every one second.
9. Query the time from the start of counting every one second.
10. Query the following data after the time from the start of counting indicates more than 10 seconds.
    Opt tab: LOS
    Statistics tab: Clock Source Loss, Rx Frequency (Hz), Rx Frequency (ppm), Bit Errors (rate), Bit Errors (Count)

```
; sample program for MD1260A ver 3.1
; Anritsu Corporation April,2012
;
; set local echo to on
setecho 1
flushrecv
; specify module number of MD1260A
sendln ':UENT:ID 1'

; time out 3 second
timeout=3

; query application
sendln ':SOURce:MAPPing?'
waitln 'E100G' 'E40G'

result_mapp=result
if result_mapp=0 goto _timeout
if result_mapp=3 then
  messagebox  'Application is not 40GbE nor 100GbE' 'MD1260A Application'
  end
endif

call check_error_code

; set port as normal mode
sendln ':ROUTe:MODE NORMAL'
call check_error_code
; set frame BERT off
sendln ':ROUTe:BERT 0'
call check_error_code
; set LFS reply on
sendln ':ROUTe:LFS:REPLy 1'
call check_error_code
; set flow control on
sendln ':ROUTe:FCONtrol 1'
call check_error_code
; set Rx MPLS-TP Control Word to off
sendln ':ROUTe:MPLStp:CWORd 0'
call check_error_code
pause 2

call check_link
```

```
; stream configuration
for id 1 16
  int2str id_str id
  spcify_id=':SOURce:STReam:ID '
  strconcat spcify_id id_str
  sendln spcify_id
  call check_error_code
  sendln ':SOURce:STReam:CONTrol:TYPE RANDOM'
  call check_error_code
  sendln ':SOURce:STReam:CONTrol:RANGe 2000,200000'
  call check_error_code
  sendln ':SOURce:STReam:FSIZe:TYPE RANDOM'
  call check_error_code
  sendln ':SOURce:STReam:FSIZe:RANGe 128,1600'
  call check_error_code
next

sendln ':SOURce:STReam:ENABle 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16'
call check_error_code
sendln ':SOURce:STReam:STARt'
call check_error_code

; check counter status to clear buffer
sendln ':CALCulate:COUNter:STARt:EVENt?'
pause 1  ; wait 1 second
waitln '0' '1'
call check_error_code

; start counter
sendln ':CALCulate:COUNter:STARt'
call check_error_code
pause 3

; query counter status
for i 1 10
  sendln ':CALCulate:COUNter:STARt:EVENt?'
  pause 1; wait 1 second
  waitln '0' '1'
  counter_start=result
  if result=0 goto _timeout
  if result=2 break
  call check_error_code
next
```

```
if counter_start=1 then
  messagebox  'Counter is not ready' 'Warning !'
endif

; waiting until elapsed time reaches 10 second
for i 1 20
  sendln ':CALCulate:DATA? ELAPSED'
  pause 1; wait 1 second
  recvln
  recvln
  ;call check_response ; debug
  if result=0 goto _timeout
  if result=1 then
          str2int e_time inputstr
          if e_time>9 break
  endif
  call check_error_code
next

; data acquisition
sendln ':CALCulate:DATA? LOS,CSLOSS,RX_FREQ,RX_FREQ_D,BER_CNT,BER_RATE'
call check_error_code
call check_alarm

messagebox  'Macro end successfully' 'Finish'

End

;       ------------ subroutines -----------
:_timeout
  messagebox  'No response from MD1260A.' 'Time out!'
  call check_error_code
  End

:check_link
  ; check status lamp of link
  sendln ':CALCulate:DATA? LINK'
  waitln '0' '1' '2'
  link_state=result
  ; in case of timeout
  if result=0 goto _timeout
  ; in case of link up
  ;if result=1  messagebox  'Link lamp of status area is  lit' 'Link Up'
  ; in case of link down
  if result=2 messagebox  'Link lamp of status area is not lit' 'Link Down'

  call check_error_code

return

:check_alarm
  ; in case of link, check status lamp of error
  sendln ':CALCulate:EALarm?'
  waitln '0' '1' '2'
  ; in case of timeout
  if result=0  then
         goto _timeout
  ; in case of no error
  elseif result=1  then
         messagebox  'Error lamp of status area is not lit' 'No error occurring'
  ; in case of error occurring
  elseif result=2 then
         messagebox  'Error lamp of status area is red' 'Error occurring'
         end
  elseif result=3  then
         messagebox  'Error lamp of status area is orange' 'No error occurring
currently'
  endif

  call check_error_code

return

:check_error_code
  ; query error
  sendln ':SYSTem:ERRor?'
  waitln 'No error'

  ; in case of timeout
  if result=0 goto _timeout
  ; in case of error occurring
  if result=2 then
         e_message='Error code = '
         strconcat e_message inputstr
         messagebox  e_message 'Command Error occurred'
         end
  endif
  ; in case of no error

Return

:check_response
  ;for debug
  messagebox inputstr 'debug1'
  int2str result_str result
  messagebox result_str 'debug2'
  return
```

## E.3 Example 2: Reading out Error Insertion and Counter

This sample program sets the errors of the 100 GbE or 40 GbE application and reads out the counter value.

Processing Flow

1. The Unit ID defines One MD1260A as the control target.
2. The application is queried by :SOURce:MAPPing?.
   When the application is not OTU4 or OTU3, the message box is displayed and the processing is suspended.
3. Set the errors as follows.
   Type: ErrorODU4-PM-BIP8
   Timing: Rate, 10E–8
4. Start counting.
5. Query whether to start counting every one second.
6. Query the time from the start of counting every one second.
7. Query the following data after the time from the start of counting indicates more than 20 seconds.
   Summary tab: LOS, Clock Source Loss, Rx Frequency (Hz),
   Rx Frequency (ppm)
   Statistics tab: PM-BIP8 (count)

```
; sample program for MD1260A ver 3.0
; Anritsu Corporation June,2011
;; set local echo to on
setecho 1
flushrecv
; specify module number of MD1260A
sendln ':UENT:ID 1'

; time out 3 second
timeout=3

; query application
sendln ':SOURce:MAPPing?'
waitln 'OTU3' 'OTU4'

result_mapp=result
if result_mapp=0 goto _timeout
if result_mapp=3 then
  messagebox  'Application is not OTU3 nor OTU4.' 'MD1260A Application'
  end
endif

call check_error_code

; set port parameter
sendln ':ROUTe:MODE NORMAL'
call check_error_code
sendln ':SOURce:OTN:FEC OFF'
call check_error_code

pause 1

; stream configuration
sendln ':SOURce:EALarm:TYPE PM_BIP8'
call check_error_code
sendln ':SOURce:EALarm:TIMing:TYPE RATE'
call check_error_code
sendln ':SOURce:EALarm:TIMing:RATE R1E_8'
call check_error_code
sendln ':SOURce:EALarm:STARt:EVENt?'; clear event buffer
pause 1  ; wait 1 second
waitln '0' '1'

call check_error_code
```

```
; start error insertion
sendln ':SOURce:EALarm:STARt'
call check_error_code

pause 2

; query error insertion status
for i 1 10
  sendln ':SOURce:EALarm:STARt:EVENt?'
  pause 1; wait 1 second
  waitln '0' '1'
  error_ins_start=result
  if result=0 goto _timeout
  if result=2 break
  call check_error_code
next

if error_ins_start=1 then
  messagebox  'Error insertion does not start' 'Warning !'
endif

; check counter status to clear buffer
sendln ':CALCulate:COUNter:STARt:EVENt?'

waitln '0' '1'
call check_error_code

; start counter
sendln ':CALCulate:COUNter:STARt'
call check_error_code
pause 3

; query counter status
for i 1 10
  sendln ':CALCulate:COUNter:STARt:EVENt?'
  pause 1; wait 1 second
  waitln '0' '1'
  counter_start=result
  if result=0 goto _timeout
  if result=2 break
  call check_error_code
next
```

```
if counter_start=1 then
  messagebox  'Counter is not ready' 'Warning !'
endif

; waiting until elapsed time reaches 15 second
for i 1 20
  sendln ':CALCulate:DATA? ELAPSED'
  pause 1; wait 1 second
  recvln
  recvln
  ;call check_response ; debug
  if result=0 goto _timeout
  if result=1 then
          str2int e_time inputstr
          if e_time>14 break
  endif
  call check_error_code
next

; data acquisition
sendln ':CALCulate:DATA? LOS,CSLOSS,RX_FREQ,RX_FREQ_D,PM_BIP8'
call check_error_code
call check_alarm

messagebox  'Macro end successfully' 'Finish'
End

;       ------------ subroutins -----------

:check_response
  ;for debug
  int2str result_str result
  messagebox result_str 'debug1'

  return

:_timeout
messagebox  'No response from MD1260A.' 'Time out!'
End

:check_alarm
  ; in case of link, check status lamp of error
  sendln ':CALCulate:EALarm?'
  waitln '0' '1' '2'
  ; in case of timeout
  if result=0  then
         goto _timeout
  ; in case of no error
  elseif result=1  then
         messagebox  'Error lamp of status area is not lit' 'No error occurring'
  ; in case of error occurring
  elseif result=2 then
         messagebox  'Error lamp of status area is red' 'Error occurring'
         end
  elseif result=3  then
         messagebox  'Error lamp of status area is orange' 'No error occurring
currently'
  endif

  call check_error_code

return

:check_error_code
  ; query error
  sendln ':SYSTem:ERRor?'
  waitln 'No error'

  ; in case of timeout
  if result=0 goto _timeout
  ; in case of error occurring
  if result=2 then
         e_message='Error code = '
         strconcat e_message inputstr
         messagebox  e_message 'Command Error occurred'
         end
  endif

  ; in case of no error

return
```

## E.4 Example 3: Saving Captured Data

This sample program saves the data captured by the 100 GbE or 40 GbE application.

Processing Flow

1. The Unit ID defines One MD1260A as the control target.
2. Query the application by :SOURce:MAPPing?.
   When the application is not OTU4 or OTU3, the message box is displayed and the processing is suspended.
3. Query the link lamp in the status area after setting to the normal mode.
4. Query the Error/Alarm lamp in the status area if the link by lamp lights green.
5. Set a port if the Error/Alarm by lamp does not light red.
   Bert: Off
   LFS Reply: On
   Flow Control: On
   Rx MPLS-TP Control Word : Off
6. Set the lane mapping.
7. Set the stream 1 through 16 as follows.
   Frame size: Random, 128 to 1600 (bytes)
   Gap size: Random, 2000 to 20000 (bytes)
8. Enable the stream 1 through 16 as follows.
9. Read out the counter status.
10. Start counting.
11. Read out the counter status.
12. After the captured data is obtained, save data with the file name including the date and time.

```
; sample program for MD1260A ver 3.1
; Anritsu Corporation April,2012
;
; set local echo to on
setecho 1
flushrecv
; specify module number of MD1260A
sendln ':UENT:ID 1'

; time out 3 second
timeout=3

; query application
sendln ':SOURce:MAPPing?'
waitln 'E100G' 'E40G'

result_mapp=result
if result_mapp=0 goto _timeout
if result_mapp=3 then
  messagebox  'Application is not 40GbE nor 100GbE' 'MD1260A Application'
  end
endif

call check_error_code

; set port as normal mode
sendln ':ROUTe:MODE NORMAL'
call check_error_code
; set frame BERT off
sendln ':ROUTe:BERT 0'
call check_error_code
; set LFS reply on
sendln ':ROUTe:LFS:REPLy 1'
call check_error_code
; set flow control on
sendln ':ROUTe:FCONtrol 1'
call check_error_code
; set Rx MPLS-TP Control Word to off
sendln ':ROUTe:MPLStp:CWORd 0'
call check_error_code

pause 2

call check_link
call check_alarm

; set lane mapping
if result_mapp=2 sendln ':SOURce:MAPPing:LANE 0,1,2,3'
if result_mapp=1 sendln ':SOURce:MAPPing:LANE
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19'
call check_error_code

; stream configuration
for id 1 16
int2str id_str id
spcify_id=':SOURce:STReam:ID '
strconcat spcify_id id_str
sendln spcify_id
call check_error_code
sendln ':SOURce:STReam:CONTrol:TYPE RANDOM'
call check_error_code
sendln ':SOURce:STReam:CONTrol:RANGe 2000,200000'
call check_error_code
sendln ':SOURce:STReam:FSIZe:TYPE RANDOM'
call check_error_code
sendln ':SOURce:STReam:FSIZe:RANGe 128,1600'
call check_error_code
next
sendln ':SOURce:STReam:ENABle 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16'
call check_error_code

; set trigger to oversize
sendln ':CALCulate:TRIGger:CONDition #B00000001'
pause 1  ; wait 1 second
call check_error_code

; clear event buffer
sendln ':CALCulate:CAPTure:STARt:EVENt?'
call check_error_code

; start capture
sendln ':CALCulate:CAPTure:STARt'
call check_error_code
flushrecv

for i 1 10
  pause 1
  sendln ':CALCulate:CAPTure:STARt:EVENt?'
```

```
    waitln '0' '1'
    capture_result=result
    if result=0 goto _timeout
    if result=2 break
    call check_error_code

  next

  if capture_result=1 then
    messagebox  "Capture does not start!" 'Capture error'
    end
  endif

  ; stream start
  sendln ':SOURce:STReam:STARt'
  call check_error_code

  ; wait trigger of capture
  for i 1 10
    pause 1
    sendln ':CALCulate:CAPTure:STATus?'
    waitln '0' '1'
    capture_result=result
    ;call check_response
    if result=0 goto _timeout
    if result=1 break
    call check_error_code
  next

  if capture_result=2 then
    messagebox  "Capture trigger doesn't raise!" 'Capture error'
    end
  endif

  ; save capture data to file
  getdate filename "macro-%Y%m%d-%H%M%S"
  header=':MMEMory:STORe:CAPTure '
  strconcat header filename

  sendln ':MMEMory:STORe:CAPTure:ITEM LIBPCAP'
  call check_error_code
  sendln header
  call check_error_code
```

```
messagebox  'Macro end successfully' 'Finish'

End

:_timeout
messagebox  'No response from MD1260A.' 'Time out!'
End

:check_link
  ; check status lamp of link
  flushrecv
  sendln ':CALCulate:DATA? LINK'
  waitln '0' '1' '2'
  call check_response
  link_state=result
  ; in case of timeout
  if result=0 goto _timeout
  ; in case of link up
  ;if result=1  messagebox  'Link lamp of status area is  lit' 'Link Up'
  ; in case of link down
  if result=2 messagebox  'Link lamp of status area is not lit' 'Link Down'

  call check_error_code

return

:check_alarm
  ; in case of link, check status lamp of error
  sendln ':CALCulate:EALarm?'
  waitln '0' '1' '2'

  ; in case of timeout
  if result=0  then
         goto _timeout
  ; in case of no error
  elseif result=1  then
         ;messagebox  'Error lamp of status area is not lit' 'No error occurring'
  ; in case of error occurring
  elseif result=2 then
         messagebox  'Error lamp of status area is red' 'Error occurring'
         end
  elseif result=3  then
         messagebox  'Error lamp of status area is orange' 'No error occurring
currently'
```

```
    endif

    call check_error_code

  return

  :check_error_code
    ; query error
    sendln ':SYSTem:ERRor?'
    waitln 'No error'

    ; in case of timeout
    if result=0 goto _timeout
    ; in case of error occurring
    if result=2 then
           e_message='Error code = '
           strconcat e_message inputstr
           messagebox  e_message 'Command Error occurred'
           end
    endif

    ; in case of no error

  return

  :check_response

    ;for debug
    messagebox inputstr 'debug1'
    int2str result_str result
    messagebox result_str 'debug2'

    return
```

# Index

References are to page numbers.