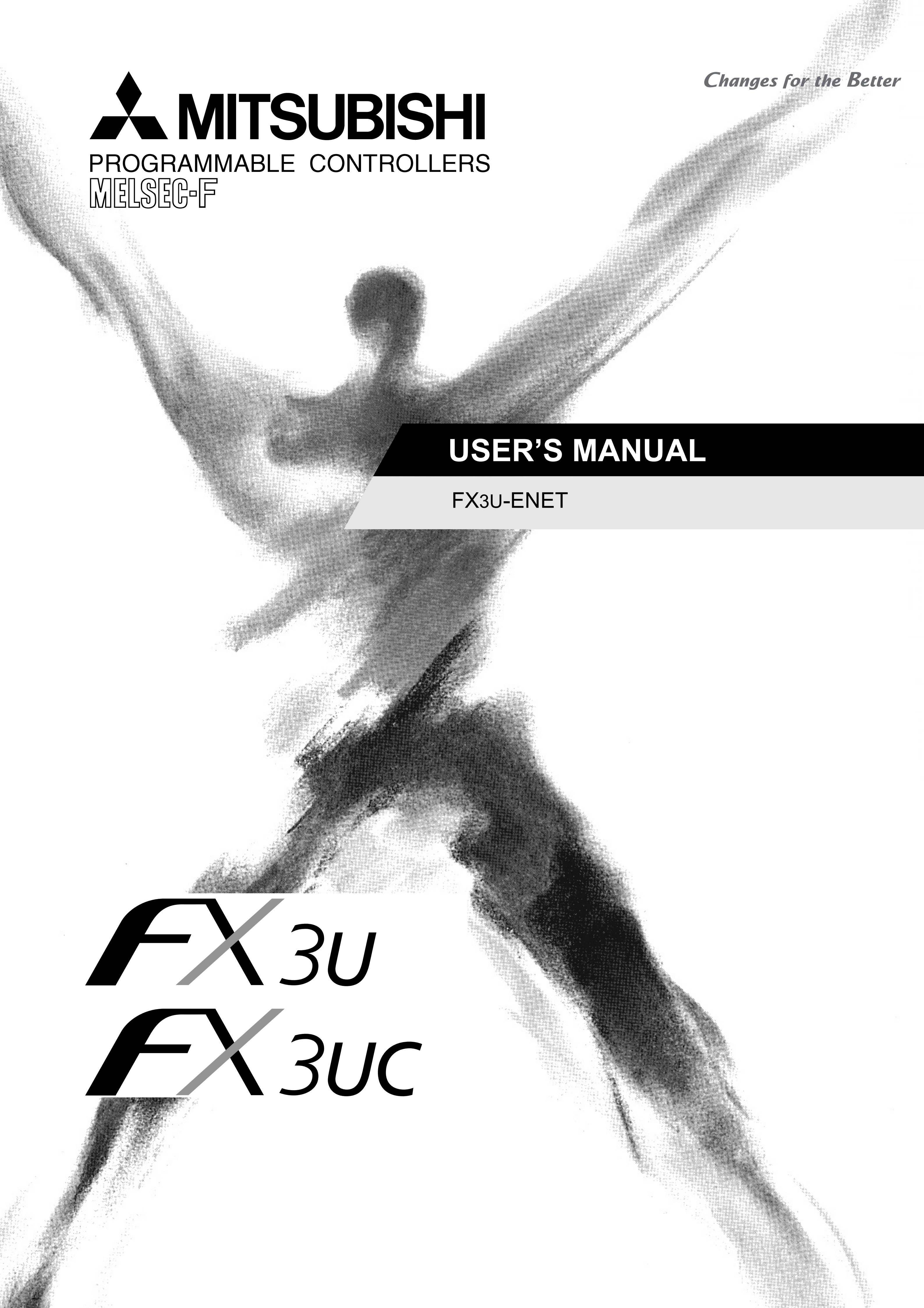
MITSUBISHI

Changes for the Better

PROGRAMMABLE CONTROLLERS

MELSEC-F



USER'S MANUAL

FX3U-ENET

FX3U

FX3UC

• SAFETY PRECAUTIONS •

(Read these precautions before use.)

Before installation, operation, maintenance or inspection of this product, thoroughly read through and understand this manual and all of the associated manuals. Also, take care to handle the module properly and safely.

This manual classifies the safety precautions into two categories: “DANGER” and “CAUTION.”

◇! DANGER	Indicates that incorrect handling may cause hazardous conditions, resulting in death or severe injury.
⚠ CAUTION	Indicates that incorrect handling may cause hazardous conditions, resulting in medium or slight personal injury or physical damage.

Depending on the circumstances, procedures indicated by CAUTION may also cause severe injury.

It is important to follow all precautions for personal safety.

Store this manual in a safe place so that it can be taken out and read whenever necessary. Always forward it to the end user.

[Design Precautions]

◇! DANGER

- To prevent malfunctions of the PLC system that may be caused by illegal e-mail, take proper countermeasures (such as virus detection) so that illegal e-mail is not received by the mail server of this module.
- If it is necessary to ensure the security of the PLC system against unauthorized access from external devices via the Internet, appropriate measures must be incorporated by the user.
- When controlling the PLC (modifying data) while it is in operation by connecting computer peripheral devices to the main unit or connecting personal computers to the extension device, make sure to have an interlock circuit outside of the PLC to ensure safe system operation.

[Design Precautions]

◇! DANGER

- Do not write any data in the "system area" of the buffer memory of the extension device. When writing a value to the buffer memory including "System Area," pay close attention not to change the system bit. If data is written to the "system area" or the "use prohibited" signal is output, there is a risk that the PLC system may malfunction.
- When the program examples introduced in this manual are applied to the actual system, examine the safety of the control in the target system.

⚠ CAUTION

- When the status control (remote RUN/STOP) of the PLC is performed from the external device, select the "Always wait for OPEN" parameter set by an user in advance. (Select with the initial timing setting (BFM#24 b8) in the operational setting.) When "Do not wait for OPEN" is selected, the communication line at remote STOP is closed. The communication line cannot be reopened on the PLC side after that, and the remote RUN from the external device cannot start.

[Installation Precautions]

⚠ CAUTION

- Use the product within the generic environment specifications described in chapter 3 of this manual.
  Never use the product in areas with excessive dust, oily smoke, conductive dusts, corrosive gas (salt air, Cl_2, H_2S, SO_2 or NO_2), flammable gas, vibration or impacts, or exposed to high temperature, condensation, or rain and wind.
  If the product is used in such conditions, electric shock, fire, malfunctions, deterioration or damage may occur.
- When drilling screw holes or wiring, make sure cutting or wire debris does not enter the ventilation slits.
  Failure to do so may cause fire, equipment failures or malfunctions.
- Be sure to remove the dust proof sheet from the PLC's ventilation port when installation work is completed. Failure to do so may cause fire, equipment failures or malfunctions.
- Connect the extension cables securely to their designated connectors.
  Unsecured connection may cause malfunctions.

[Wiring Instructions]

◇ DANGER

- Before mounting or wiring the module, make sure to shut off all of the external power supply.
  Failure to do so may cause electric shocks or damage the module.
- When turning on the power and operating after mounting or wiring the module, make sure to install the accessory terminal covers to the product.
  Otherwise, it may cause electric shocks.

⚠ CAUTION

- Make sure to place the communication cables and the power cables in a duct or fasten them using a clamp.
  If the cables are not placed in a duct or fastened with a clamp, their positions can be unstable, moved and pulled inadvertently. This may damage the module and the cables, or cause the module malfunction due to unsecured cable connections.
- When disconnecting the communication cables and the power cables, do not pull the cables.
  When disconnecting a cable with a connector, hold the connector to the module by hand and pull it out to remove the cable.
  When disconnecting a cable connected to a terminal block, loosen the screws on the terminal block before removing the cable.
  If a cable is pulled while being connected, it may cause the module malfunction or damage the module and the cable.
- Do not bundle the control line and the communication cables together with or lay it close to the main circuit or power line. As a guideline, lay the control line and the communication cables at least 100mm (3.94") or more away from the main circuit or power line.
  Noise may cause malfunctions.
- Before connecting twisted pair cables, the ground pin brush grounded via a resistor enables discharging static electricity on the cables effectively. This can prevent the electrostatic discharge to the product.

[Setup and Maintenance Precautions]

◇! DANGER

- Do not touch any terminal while the PLC’s power is on.
  Doing so may cause electric shock or malfunctions.
- Before cleaning or retightening terminal screws and module mounting screws, externally cut off all phases of the power supply.
  Failure to do so may cause electric shock, fire, malfunctions and product damage.
- Before modifying or disrupting the program, forcible output, RUN and STOP while they are in operation, carefully read through this manual and the associated manuals and ensure the safety of the operation.
  An operation error may damage the machinery or cause accidents.

⚠ CAUTION

- Do not disassemble or modify the PLC.
  Doing so may cause fire, equipment failures, or malfunctions.
  For repair, contact your local Mitsubishi Electric distributor.
- Cut off all phases of the power supply externally before installation or wiring work in order to avoid damage to the product or electric shock.

[Precautions When Disposing of This Product]

⚠ CAUTION

- Dispose of this product as an industrial waste.

CONTENTS

11 TROUBLESHOOTING 11- 1 to 11-45

APPENDIX App- 1 to App-25

About Manuals

This manual explains the mounting of FX3U-ENET, the specifications, broadcast functions and communication methods.
For the instructions and programs of the sequence as well as the parameter settings by FX Configurator-EN, also refer to the following manuals.

Associated Manuals

Manual name	Manual No.	Description
FX3U-ENET User's Manual	JY997D18101 MODEL CODE: 09R716	This manual
FX3U-ENET INSTALLATION MANUAL	JY997D15901	Installation of FX3U-ENET block.
FX3U Series HARDWARE MANUAL	JY997D18801	Extracts the I/O specifications, wiring, and installation of FX3U Series PLC from FX3U Series User's Manual – Hardware Edition.
FX3U Series User's Manual - Hardware Edition	JY997D16501 MODEL CODE: 09R516	Explains FX3U Series PLC specification details for I/O, wiring, installation, and maintenance.
FX3UC(D, DSS) Series HARDWARE MANUAL	JY997D28601	Extracts the I/O specifications, wiring, and installation of FX3UC(D, DSS) Series PLC from FX3UC Series User's Manual – Hardware Edition.
FX3UC-32MT-LT-2 HARDWARE MANUAL	JY997D31601	Extracts the I/O specifications, wiring, and installation of FX3UC-32MT-LT-2 from FX3UC Series User's Manual – Hardware Edition.
FX3UC Series User's Manual - Hardware Edition	JY997D28701 MODEL CODE: 09R519	Explains FX3UC Series PLC specification details for I/O, wiring, installation, and maintenance.
FX3U/FX3UC Series Programming Manual - Basic & Applied Instruction Edition	JY997D16601 MODEL CODE: 09R517	Describes PLC programming for basic/ applied instructions and devices.
FX Configurator-EN Operation Manual	JY997D20501 MODEL CODE: 09R919	The operation method of FX Configurator-EN.

How to obtain manuals

For the necessary product manuals or documents, consult with the Mitsubishi Electric dealer.

How to obtain FX Configurator-EN

The parameter setting software, FX Configurator-EN is not supplied with this product. Consult with the Mitsubishi Electric dealer.

Applicable Standard

Certification of UL, cUL standards

The following product has UL and cUL certification.

UL, cUL File Number: E95239

Models: MELSEC FX$_{3U}$ series manufactured

FX$_{3U}$-ENET

Compliance with EC directive (CE Marking)

This note does not guarantee that an entire mechanical module produced in accordance with the contents of this note will comply with the following standards. Compliance to EMC directive and LVD directive of the entire mechanical module should be checked by the user / manufacturer. For more details please contact the local Mitsubishi Electric sales site.

Requirement for Compliance with EMC directive

The following products have shown compliance through direct testing (of the identified standards below) and design analysis (through the creation of a technical construction file) to the European Directive for Electromagnetic Compatibility (89/336/EEC) when used as directed by the appropriate documentation.

Type: Programmable Controller (Open Type Equipment)

Models: MELSEC FX$_{3U}$ series manufactured

from August 1st, 2005 FX$_{3U}$-ENET

Standard	Remark
EN61131-2:2003 Programmable controllers Equipment requirements and tests	Complies with all relevant aspects of the following standards. **EMI** • Radiated Emissions • Conducted Emissions **EMC** • Radiated electromagnetic field immunity • Fast Transient burst • Electrostatic discharge • High-energy surge • Voltage drops and interruptions • Conducted RF • Power frequency magnetic field

Notes for compliance to EMC regulation.

It is necessary to install the (FX$_{3U}$-ENET) in a shielded metal control panel.
For more details please contact the local Mitsubishi Electric sales site.

● How to use this manual

In this manual, explanations are given for each application of the Ethernet modules (FX3U-ENET).

Please use this manual using the following key items below as a reference.

(1) Features and utility lists

(a) Features and functions

- Chapter 1 describes the features of the Ethernet modules.
- Chapter 3 describes the common functions and specifications of the Ethernet modules.

(b) Items included in this package and network configured items

- Section 2.2 describes the system configuration of the Ethernet module. Parts and components other than those packaged with the module must be purchased separately by the user.

(2) Processing required prior to starting the operation of the Ethernet module

(a) Startup procedure

- Section 4.2 describes an outline of the procedures to do before starting the operation of the Ethernet module.

(b) Connection to the Ethernet network system.

- Section 2.2 describes the devices required to connect to the Ethernet network system.
- Section 4.4 describes the connection methods for each type of interface.

(c) Parameter settings required before starting Ethernet module operation

- Section 3.6 describes the parameter settings required in order to use each function.
  Confirm the required parameters, set them according to the relevant section, and save the setting values in the PLC to which the Ethernet module is installed.
- Section 4.5 describes the types of Ethernet parameter setting in order to use the Ethernet module.

(d) Checking for Ethernet module failures

- Section 4.7 describes the self-diagnostic test for the Ethernet module.

(e) Checking for connection errors with the external devices

- Section 5.4.1 describes how to check for connection errors using the "PING" command.
- Section 5.4.2 describes how to check for connection errors by performing the loopback test through MC protocol-based communication.

(3) Connection between the Ethernet module and external devices

(a) Communication procedures

- Section 5.1 describes an outline of the communication procedures

(b) Connections with the external devices

- Section 5.6 describes the connections (open and close processing) for each communication method (TCP/IP) and the open method (Active, Passive), including programming procedures.

(4) Details on the data communication functions

(a) Communication functions

- Section 1.2 describes an overview of the Ethernet module communication functions and related section numbers that can be referenced for more detailed explanations.

(5) Data communication functions and programming

(a) Reading from and writing to the PLC

- Data is read from and written to the PLC with communication functions using MC protocol.
- Chapter 8 is an overview of the communication functions using MC protocol.
- Chapter 9 is a detail on the communication functions using MC protocol.

(b) Sending and receiving data between the PLC and the external devices

- Data communication between the PLC and external devices is performed with the fixed buffer communication functions.
- Chapters 6 and 7 explains the communication functions and programming using the fixed buffers.

(6) Checking for error occurrences and taking corrective actions

(a) Error codes

- Chapter 11 describes troubleshooting, how to check for errors, and the contents and reference manuals for error codes.

(b) Error code storase area in the buffer memory of the Ethernet module

- Section 11.3 describes the error code storage areas in the buffer memory.

● FX Configurator-EN

Using FX Configurator-EN to set parameters, the sequence programs for communicating with external devices can be simplified.
For details on the parameter settings from FX Configurator-EN, refer to the FX Configurator-EN operation manual.

About the Generic Terms and Abbreviations

This manual uses the following generic terms and abbreviations to describe the Model FX_{3U}-ENET Ethernet interface block.

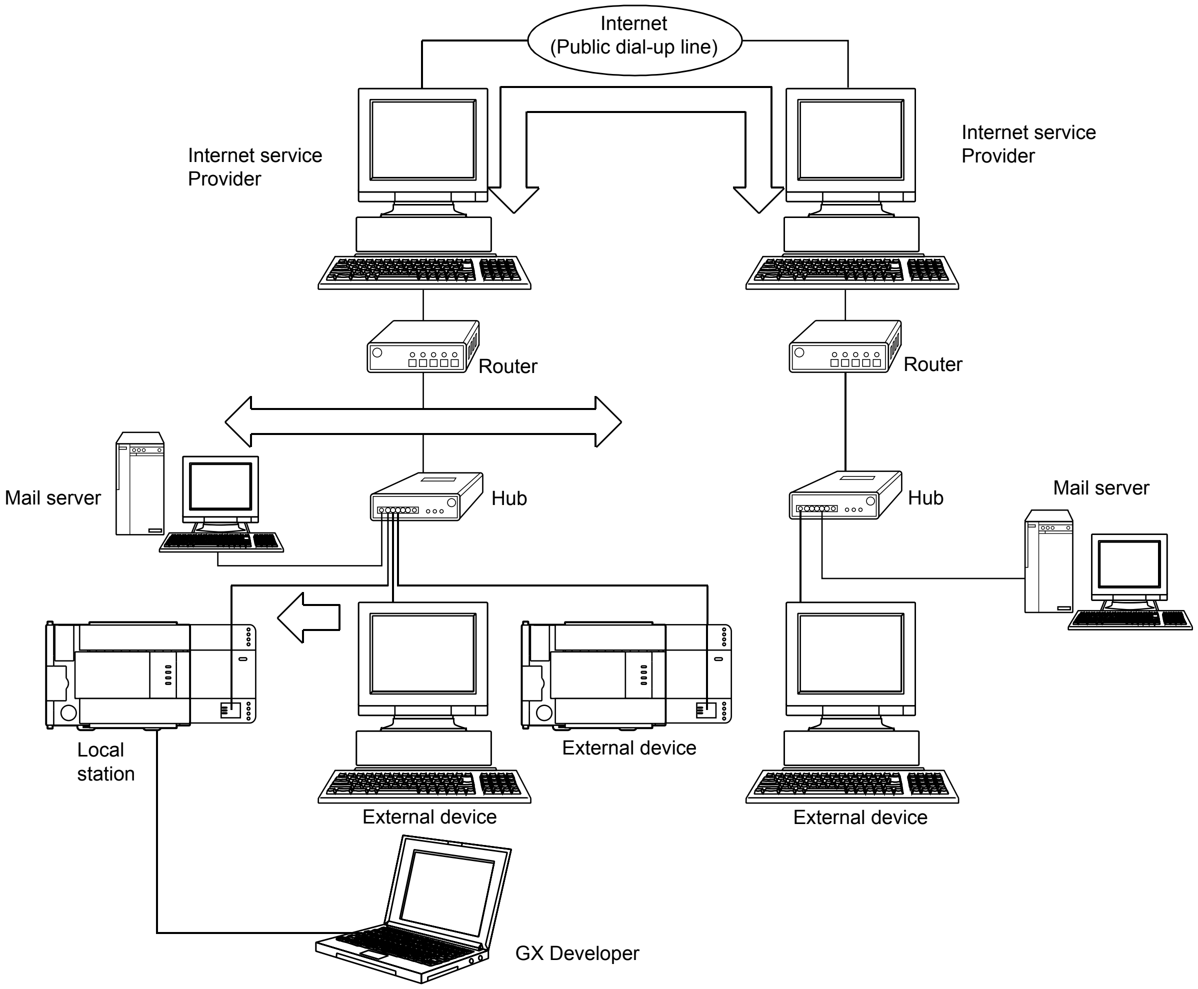
Generic Term/Abbreviation	Description
Ethernet Address	A machine-specific address that is also referred to as the MAC (Media Access Control Address). This is used to identify the addresses of external devices over a network. The Ethernet address of the Ethernet module can be verified on the MAC ADD column of the rating plate.
Ethernet module	Abbreviation for the FX_{3U}-ENET Ethernet Interface block (Described as the Ethernet module or FX_{3U}-ENET in the figures)
Ethernet network	Abbreviation for 10BASE-T and 100BASE-TX networks
GX Developer	Abbreviation for GX Developer (SWnD5C-GPPW-E). (n in the model name is 4 or later)
External device	Generic term for personal computers, computers, workstations (WS) and Ethernet module etc. that are connected by Ethernet for data communication
Personal computer	Generic term for an IBM PC/AT (or IBM compatible) personal computer
Main unit	Generic name for FX Series PLC main unit
FX Configurator-EN (GX Developer)	Software for setting the Ethernet module parameters. The software is described in this manual as "FX Configurator-EN (GX Developer)" because it can be launched both independently as well as from the [Tools] menu in GX Developer.

1 OVERVIEW

This manual provides information on the specifications of the Ethernet interface block, FX3U-ENET (hereinafter called FX3U-ENET or the Ethernet module), as well as the procedures before starting operation, the control procedures and data communication method for communicating with external devices, maintenance, inspection, and troubleshooting.

1.1 Overview of the Ethernet Module

The Ethernet module is an interface module on the PLC side for connecting the FX3U / FX3UC series PLC with the host system, such as a personal computer work station, and other PLCs using the TCP/IP or UDP/IP communication protocol via Ethernet (100BASE-TX, 10BASE-T).



1) Collection and modification of PLC data
(Communication using the MELSEC Communication Protocol (referred to below as the MC Protocol))
2) Transmission and reception of arbitrary data to/from external devices
(Communication using fixed buffers)
3) Data transmission/reception by e-mail
(When using the e-mail function)

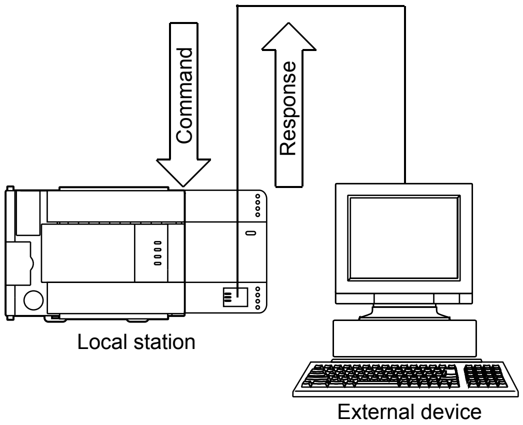
＊ By using the FX Configurator-EN (GX Developer) to set each parameter, the sequence programs for communication can be significantly simplified.

1.2 Features of the Ethernet Module

(1) Data communication using MC protocol (Details are explained in Chapter 8 and Chapter 9)

Using MC protocol, the device data of the PLC can be read from/written to the host system.

This protocol is a passive protocol that communicates data solely according to the requests from the host system. It does not require a sequence program for data communication after a connection is established.

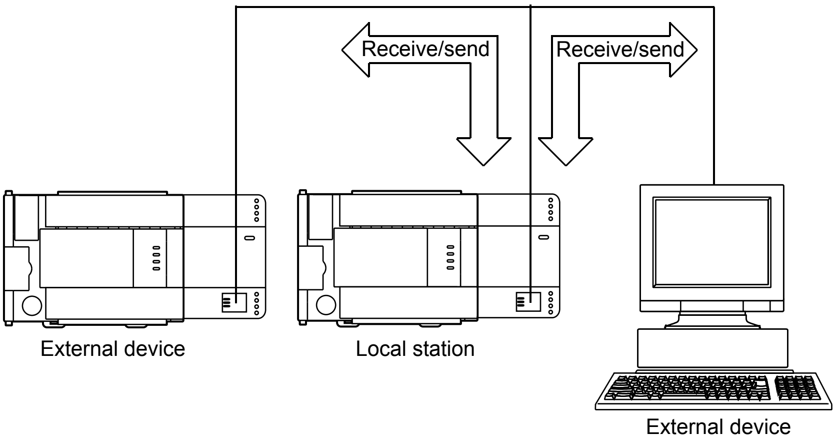


(2) Communication using fixed buffers (Details are explained in Chapters 6 and 7)

With communication using fixed buffers, a maximum of 1023 words of arbitrary data can be sent or received between PLCs or between the PLC and an arbitrary device.

An Ethernet module is provided with 8 fixed buffer data areas each with 1023 word storage space, and each is assigned as either a sending or receiving buffer for an arbitrary device.

Since communication using MC protocol is passive, communication using fixed buffers is an active protocol. Data can be sent from the PLC side to another PLC and an arbitrary device when equipment errors occur or when some specified conditions are satisfied.



(3) Communication by e-mails (Details are explained in Chapter 10)

Data can be sent to and received from an external device at a remote location using e-mail via an Internet line.

(a) Sending/receiving e-mail from the PLC

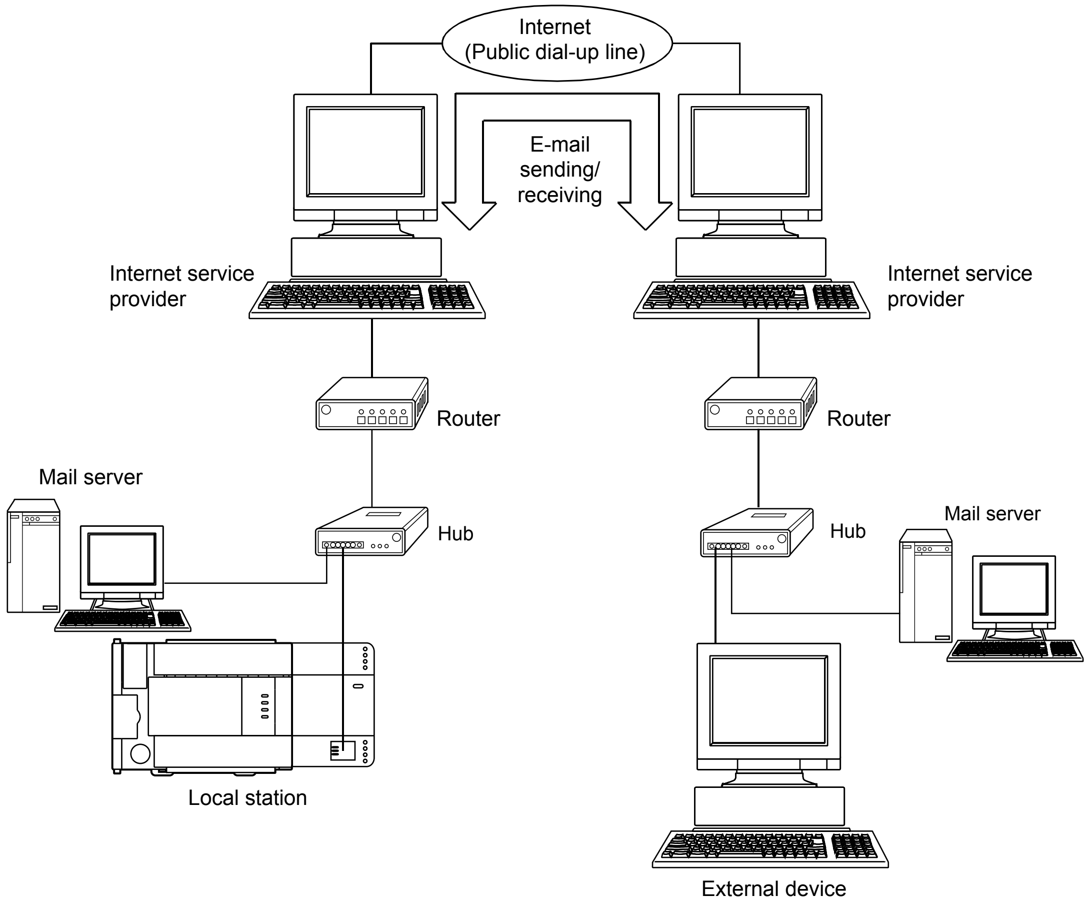
The following data can be sent/received by e-mail.

1) Sending/receiving data as attached files

Up to 2K words of data can be sent to or received from a personal computer or other Ethernet module with mail function as a file attached to an e-mail.

2) Sending data as main text

Up to 256 words of data can be sent to a personal computer or portable terminal in the main text of an e-mail.



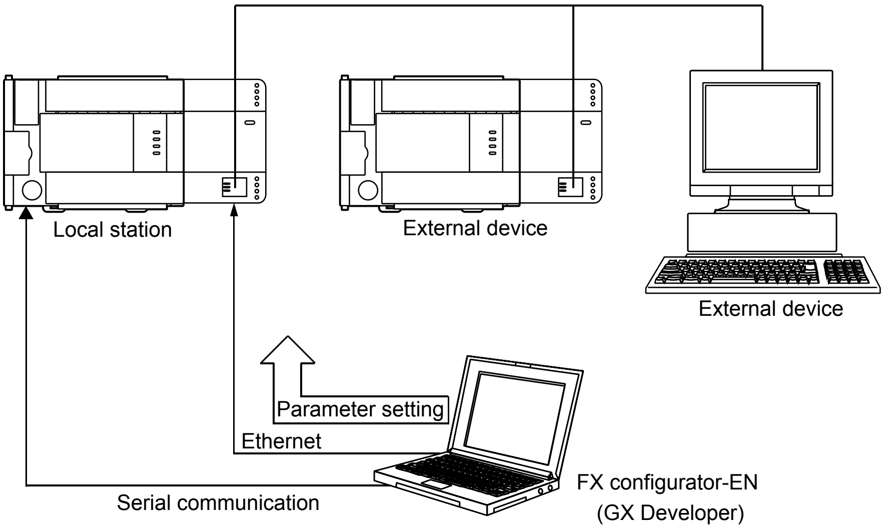
(4) Connecting FX Configurator-EN (Details are explained in the Operating Manuals for FX Configurator-EN)

(a) Simplifying sequence programs using FX Configurator-EN (GX Developer)

FX Configurator-EN (GX Developer) supports the parameter setting function to perform the Ethernet module initialization and the open processing with external devices. By setting up the following parameters with FX Configurator-EN (GX Developer), access is enabled from the external device to the PLC. It can also simplify sequence programs used to perform communication by Ethernet modules.

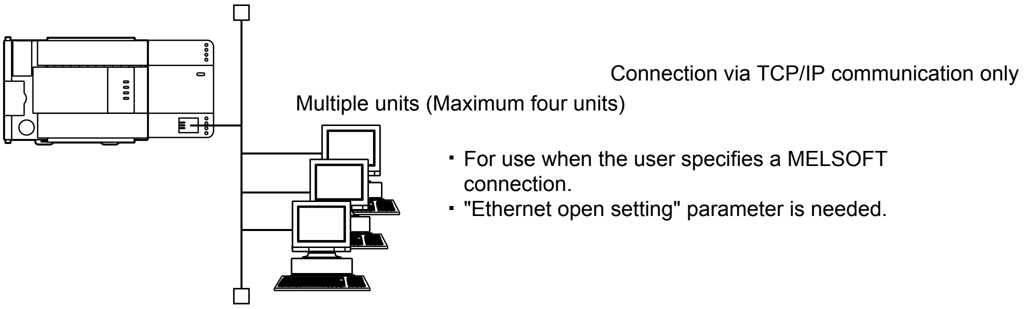
- IP address setting
- Port number setting
- Protocol type setting
- E-mail settings
- Each timer setting

For more information on setting up the Ethernet module, see Section 3.5, “List of Setting Items for Ethernet Modules” and other applicable reference sections.



(b) Connecting multiple MELSOFT products (GX Developer, MX Component)
This product can be connected with one or more MELSOFT product, via TCP/IP communication or UDP/IP communication.

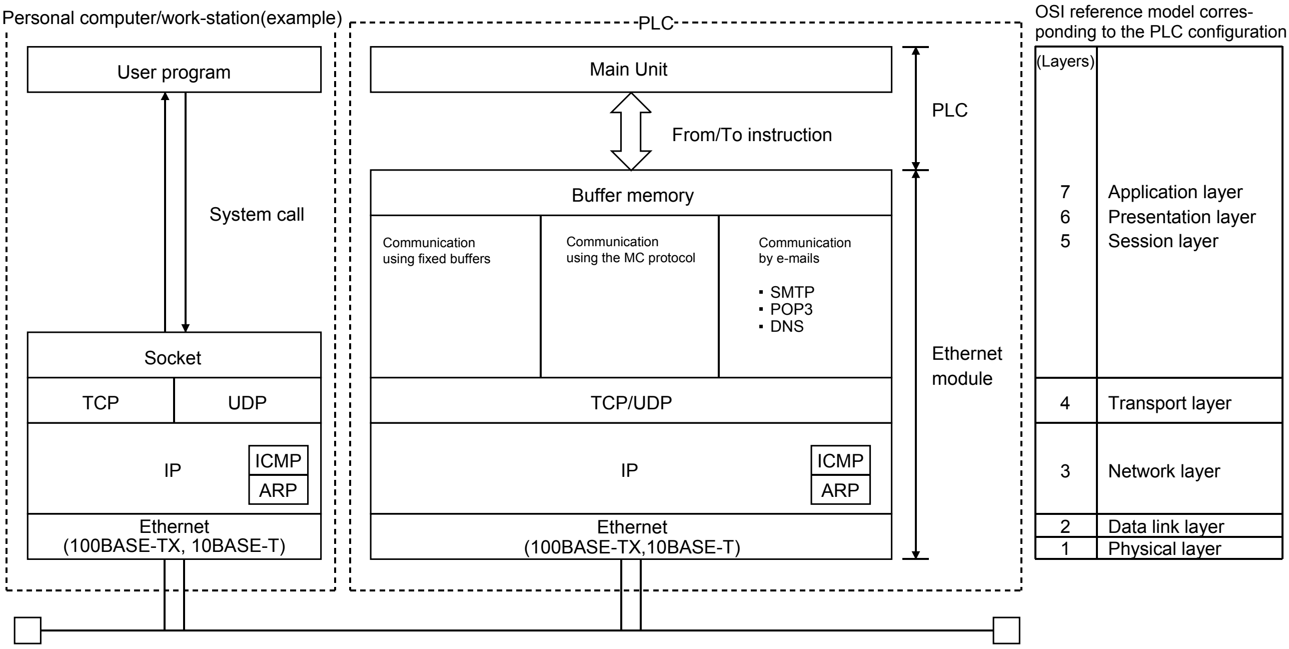
1) Connection via TCP/IP communication
- If a MELSOFT product is to be connected with the FX configurator-EN (GX Developer) open settings, a maximum of four units can be connected with GX Developer.
- If a MELSOFT product is to be connected, the settings of buffer memory for the MELSOFT connection or the following settings using FX Configurator-EN (GX Developer) are required.
For FX Configurator-EN (GX Developer), on the "Open settings" screen, set "TCP" in the protocol field of the connection number to be used, and "MELSOFT connection" in the open method field. (See Section 5.5.)


(External device)
(Protocol used)
Connection via TCP/IP communication only
Multiple units (Maximum four units)

・For use when the user specifies a MELSOFT connection.
・"Ethernet open setting" parameter is needed.

1.3 Software Configuration

The Ethernet modules support the TCP/IP and UDP/IP protocols.



(1) TCP (Transmission Control Protocol)

This protocol guarantees data credibility and reliability in communication between a personal computer/work station and PLC that are connected via network, and provides the following functions:

- Creates a logical connection by establishing a connection (logical line) as if a dedicated line was created between external devices.
- A maximum of 8 connections can be established and communicated to at the same time in the Ethernet module.
- Data reliability is maintained by the sequence control using the sequence numbers, the data retransmission function and the check sum.
- The communication data flow can be controlled by operations using Windows.

(2) UDP (User Datagram Protocol)

This protocol may not guarantee data credibility or reliability in communication between a personal computer/work station and PLC that are connected via network. Thus, even if the data does not reach the target node, it will not be retransmitted.

- Because it is connectionless, communication efficiency is much improved than TCP/IP.
- A check sum is used to increase the reliability of the communication data. When greater reliability must be maintained, a user application or the TCP should be utilized re-try operation.

(3) IP (Internet Protocol)

- Communication data is sent and received in datagram format.
- Communication data can be divided and reassembled.
- Routing option is not supported.

(4) ARP (Address Resolution Protocol)

- This protocol is used to get the Ethernet physical addresses from the IP addresses.

(5) ICMP (Internet Control Message Protocol)

- This protocol is used to exchange errors which occur on an IP network and various information related to the network.
- Provides a function to transmit IP error messages.
- See Appendix for information regarding the types of ICMP supported.

(6) DNS (Domain Name System)

- This system translates IP addresses to names that are easy to remember by the user.

(7) SMTP (Simple Mail Transfer Protocol)

- This protocol transfers mail.

(8) POP3 (Post Office Protocol Ver. 3)

- This protocol transfers mail received by a mail server to a local computer.

(9) Flag bit of TCP/IP (SYN, ACK, PSH, FIN, RST, and URG)

In the communication of TCP/IP, these flag bits indicate the segment where the connect/disconnect, response confirmation, or emergency data are executed.

1) SYN (Synchronized Bit)

When this bit is ON (1), it indicates that the initial sequence number value is set in the sequence number field.
This bit is used when the connection is newly opened.

2) ACK (Corresponding Bit)

When this bit is ON (1), it indicates that ACK (confirmation response number) field is valid.
It also indicates that this segment includes the information on response confirmation.
When this bit is OFF (0), it indicates that ACK (confirmation response number) field is invalid.

3) PSH (Push Bit)

When this bit is ON (1), the host that has received this segment sends the data to the upper application with high priority.
This bit is to be turned ON when the data should be sent to an external device as soon as possible.
When this bit is OFF (0), the timing when the received data is sent to the upper application depends on the TCP layer of the receiving side.

4) FIN (Fin Bit)

When this bit is ON (1), it indicates that there is no more data to be sent from the segment source and that the send source wants to disconnect.
However, data can be received from the external device.
The connection is on until the segment whose FIN bit is ON is received from the external device.

5) RST (Reset Bit)

When this bit is ON (1), the host from which the segment has sent disconnects unilaterally (forcibly).
Disconnection by this method is used when an unrecoverable error with the normal method has occurred or when the host has been restored after being down.

6) URG (Emergency Data Flag)

When this bit is ON (1), it indicates that this data segment includes the emergency data flag.

2 SYSTEM CONFIGURATIONS

This section explains the system configurations that may be combined with the Ethernet modules.

2.1 Applicable Systems

Ethernet modules can be used with the following systems:

(1) Applicable PLC and number of modules that can be mounted

The following table lists the PLC where the Ethernet module can be mounted and the number of modules that can be mounted.

Series name	Compatible version	Number of I/O points occupied	Number of modules that can be mounted
FX3U	System version V2.21 or more	8 points for input or output	1
FX3UC (An FX2NC-CNV-IF or FX3UC-1PS-5V is necessary)	System version V2.21 or more	8 points for input or output	1

(2) Applicable software packages

When using the Ethernet module, the FX configurator-EN (GX Developer) for software packages applicable to the Ethernet module can be used.
Refer to the manual of FX configurator-EN for the details.

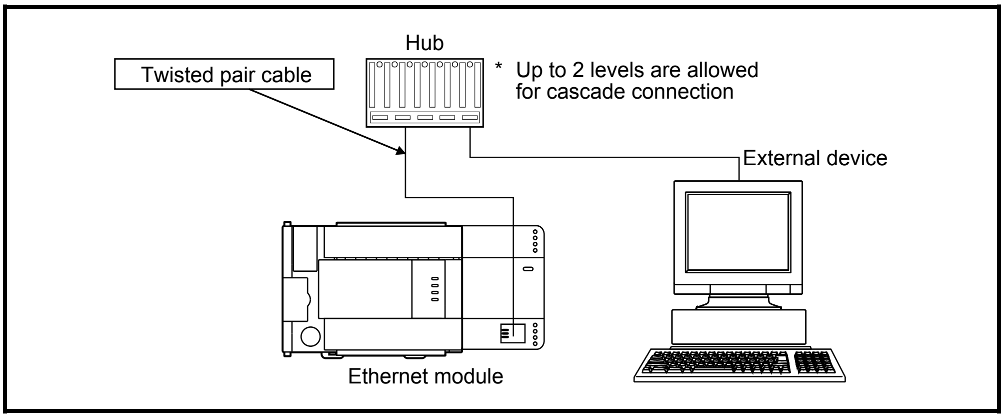
1) The Ethernet module is set with FX configurator-EN (GX Developer) [Configurable by BFM].
2) To use FX configurator-EN, either of the following software should be installed.
   - GX Developer (V 8.25B or later)
   - IEC Developer (V 7.00A or later)
3) For starting FX configurator-EN, the following shows two kinds of methods.
   - Independently starting FX configurator-EN
   - Starting from [Tools] menu of GX Developer
4) The installation form of FX configurator-EN differs depending on sales territory or country of this product.
   Consult with the dealer of this product for details.

2.2 Devices Required for Network Configuration

This section explains the devices that are required to configure a network.
Network installation work requires sufficient safeguard; ask a network specialist for installation.

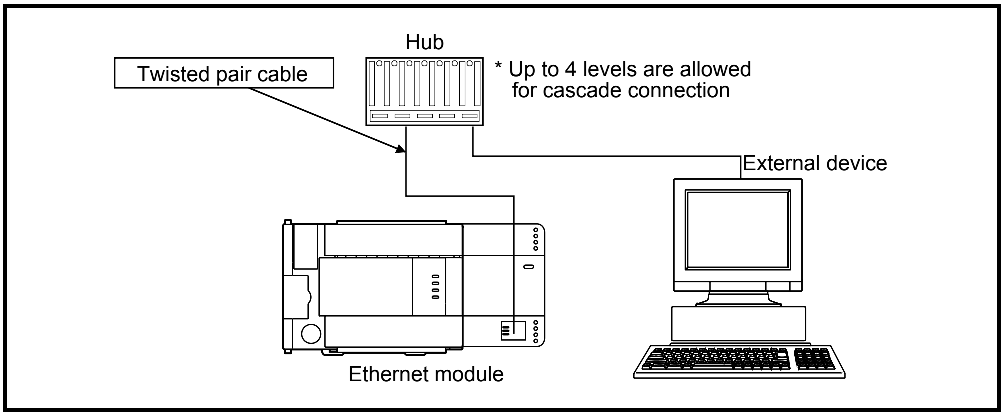
When connecting FX_{3U}-ENET to a network, either a 10BASE-T or 100BASE-TX can be used.
The Ethernet module detects whether it is 10BASE-T or 100BASE-TX, and the full-duplex or half-duplex transmission mode according to the hub.
For connection to the hub without the auto detection function, set the half-duplex mode on the hub side.

(1) Connection using the 100BASE-TX



Use devices that satisfy the standards of IEEE802.3 and 100BASE-TX.
- Shielded twisted pair cable (STP cable), category 5
  ✻ Straight cables can be used.
  (Correct operation is not guaranteed if a crossed cable is used to connect to an external device via the 100BASE-TX of the Ethernet module. However, it is possible to use crossed cables to connect two Ethernet modules for data communication or to connect an Ethernet module.)
- RJ45 jacks
- 100Mbps hub

(2) Connection using the 10BASE-T



Use devices that satisfy the standards of IEEE802.3 and 10BASE-T.

- Shielded twisted pair cable (STP cable), category 3 (4, 5)
  ＊ Straight cables can be used.
  (Correct operation is not guaranteed if a crossed cable is used to connect to an external device via the 10BASE-T of the Ethernet module. However, it is possible to use crossed cables to connect two Ethernet modules for data communication or to connect an Ethernet module.)
- RJ45 jacks
- 10Mbps hub

POINT

- During the high-speed communication (100 M bps) via 100BASE-TX connection, a communication error may occur due to the effect of high frequency noise from devices other than PLC in a given installation environment.
  The following describes countermeasures on the FX$_{3U}$-ENET side to prevent the effect of high frequency noise for construction of a network system.

(1) Wiring connection
- Do not bundle the twisted pair cables with the main circuit and power wires, and do not install them close to each other.
- They should be installed at least 100 mm (3.94 in) away from each other.
- Make sure to place the twisted pair cables in a duct.

(2) Communication method
- Data communication with an external device is performed using TCP/IP communication.
- Increase the number of communication retries as necessary.

[When the error cannot be solved]
- Communication is performed at a data transmission rate of 10 M bps by changing the connection hub for the FX3U-ENET to a hub capable of handling 10 M bps.

3 SPECIFICATIONS

This section explains the Ethernet module performance specifications and transmission specifications.
For more details on the general specifications, refer to the User's Manual for the PLC base module.

3.1 Performance Specifications

The following explains the performance specifications of the Ethernet module.

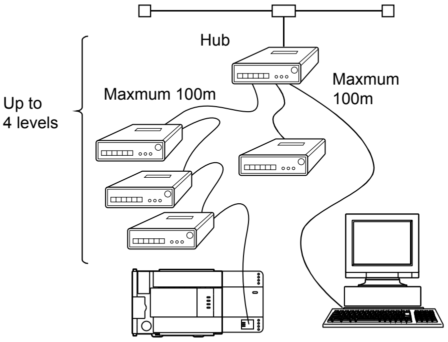
Item			Specification	
Transmission specifications	Data transmission speed		100 M bps	10 M bps
	Communication mode		Full-duplex/Half-duplex	
	Transmission method		Base band	
	Maximum segment length		100 m (328' 1") *1	
	Maximum number of nodes/connection		Cascade connection Maximum 2 stages	Cascade connection Maximum 4 stages
Transmission data storage memory	Number of simultaneously open connections allowed		8 connections (Connections usable by the sequence program)	
	Fixed buffer		1023 word × 8	
	E-mail	Attached file	2048 words × 1	
		Main text	256 words × 1	
Number of I/O occupied points			8 points	
Power supply			24V DC +20%, -15%, ripple (p-p) less than 5%	
External 24V current consumption			240 mA	
Number of connectable units to the main unit			1	
Applicable PLC			FX3U/FX3UC PLC Ver. 2.21 or later	
External dimensions			90 (3.55") (H) × 55 (2.17") (W) × 87 (3.43") (D) [mm] (inches)	
Weight			0.3 kg (0.66 lbs)	
Transmission specifications Transmission and reception data	Data size	Attached file	2048 words x 1	
		Main text	256 words x 1	
	Data transfer method		When sending: Sends either a file as attachment or main text (select one). When receiving: Receives a file as attachment.	
	Subject		US-ASCII format or ISO-2022-JP (Base64)	
	Attached file format		MIME format	
	MIME		Version 1.0	
	Data of attached file format		Binary/ASCII/CSV can be selected. File name: XXXX.bin (binary), XXXX.asc (ASCII), XXXX.csv (CSV) (CSV: Comma Separated Value)	
	Division of attached file		Cannot be divided (only one file can be sent/received) * If any divided files are received, only the first file will be received and the remaining files will be discarded.	
	When sending (encode)		Subject : Base64/7 bits Main text : 7 bits Attached file : Base64	
	When receiving (decode)		Subject : (Does not decode) Main text : (Cannot be received) Attached file : Base64/7 bits/Quoted Printable * If e-mail is sent from the external device to the PLC side, specify the encoding method (Base64/7 bits of the attached file.	
	Encryption		No	
	Compression		No	
	Communication with mail server		SMTP (sending server) Port number = 25 POP3 (receiving server) Port number = 110	
	Operation check mailer		Microsoft® Corporation (Outlook Express 6.0)	

＊1 Length between the Hub and node.

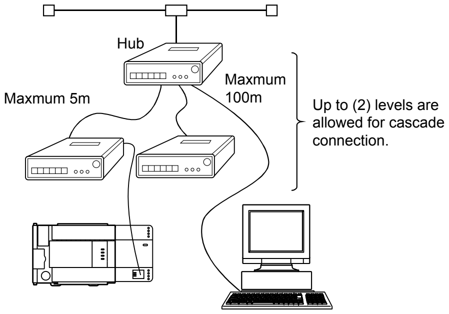
NOTE

The following explains each of the transmission specification items.

[Connecting using the 10BASE-T]



[Connecting using the 100BASE-TX]



3.2 Data Codes for Communication

This section explains the data codes used in the communication between the Ethernet module and the external device or the PLC.

(1) The data codes used while communicating are listed below.

1) Ethernet module ↔ External device
Data can be communicated by selecting either binary code or ASCII code in the data code setting of FX Configurator-EN (GX Developer) or the PLC, as shown below.
For more details about binary code/ASCII code changeover, refer to Section 4.6, "Communication Settings".

Data communication function		Communication data code		Reference chapter
		Binary code	ASCII code	
Communication using MC protocol		○	○	Chapter 8,9
Communication using fixed buffer	Procedure exist	○	○	Chapter 6
	No procedure	○ (＊1)	—	Chapter 7

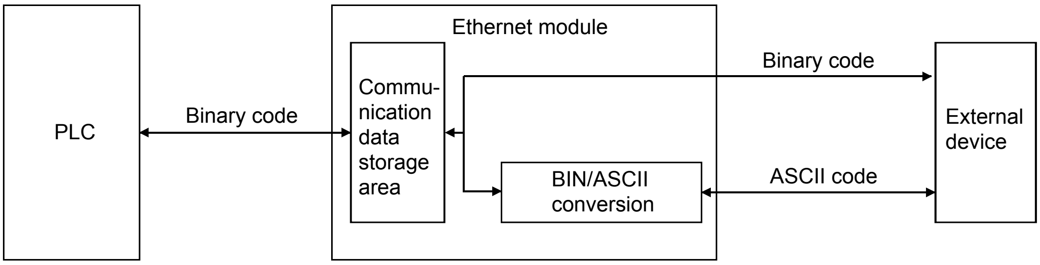
○: Selectable ×: Cannot be communicated

＊1 Communication is performed in binary code regardless of the communication code setting.

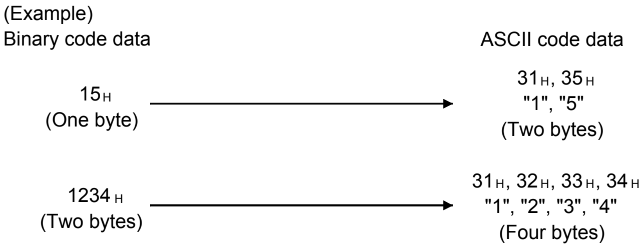
NOTE

When sending/receiving e-mail, communication is performed using the data code handled by each function, regardless of the setting of the communication data code., communication is performed using the data code handled by each function, regardless of the setting of the communication data code.

2) Ethernet module ↔ PLC
Data is sent and received in binary code.



(2) When communicating using ASCII code, 1-byte binary code data is automatically converted into 2-byte ASCII code data and then transmitted.



(3) The size of data that can be communicated at one time between the Ethernet module and an external device is determined by the function used and the selected data code (binary/ASCII) settings.
The following shows the maximum sizes of communication data that can be sent and received at a time with each data communication function.

Data communication function		Exchangeable data size
Communication using MC protocol		The maximum number of point that can be designated with each command/instruction : Maximum of 32 words
Communication using fixed buffer	Procedure exist	1017 words (Binary code) 508 words (ASCII code)
	No procedure	2046 bytes
Sending/receiving by e-mail		Attached file : Maximum of 2048 words, Main text : Maximum of 256 bytes

3.3 Relationship between the External Devices and Additional Functions for Each Communication Function

This section explains which external devices data communication can be performed with and which additional functions can be used for each function.

(1) Communicability with external devices using various functions

The following table lists the communicability with external devices using various functions.

Function	External device				
	Personal computer ↓ FX$_{3U}$-ENET	Personal computer ↑ FX$_{3U}$-ENET	FX$_{3U}$-ENET ↓ ↑ FX$_{3U}$-ENET	FX$_{3U}$-ENET ↓ QJ71E71	QJ71E71 ↓ FX$_{3U}$-ENET
Communication using MC protocol	○	×			
Communication using the fixed buffer	○				
Sending/receiving e-mail	○ (Mail server is required separately)				

○: Can communicate ×: Cannot communicate

FX3U-ENET: FX Series Ethernet interface module

QJ71E71: Q Series Ethernet interface module

(2) Relationship with additional functions

The following table lists the correspondence between functions and their additional functions that can be used.

Communication function		Additional function			Communication method	
		Router relay communication (router relay function)	Existence check of external device	Communication via pairing open	TCP/IP	UDP/IP
Communication using MC protocol		○	○	×	○	○
Communication using the fixed buffer	Procedure exist	○	○	○	○	○
	No procedure	○	○	○	○	○
Sending/receiving e-mail		○	×	×	○	×

○: Available ×: Not available or this function does not correspond to any of the functions in the function column.

3.4 Ethernet Module Function List

This section shows a list of Ethernet module functions.

(1) Basic functions of the Ethernet module

The Ethernet module can perform the communications shown in the table below via TCP/IP or UDP/IP communication.

Function		Description	Reference section
Communication using MC protocol	Subset of a compatible 1E frame	Reads/writes PLC data from/to an external device.	Chapter 8, 9
Communication using the fixed buffer	Procedure exist	Sends/receives arbitrary data between the PLC and the external device using the fixed buffer of the Ethernet module.	Chapter 6
	No procedure		Chapter 7
Sending/receiving e-mail		Sends/receives data via e-mail. • Sending/receiving by the PLC	Chapter 10

(2) Additional functions of the Ethernet module

The following table lists the additional functions of the Ethernet module that can be used.

Function	Description	Reference section
Router relay communication (router relay function)	Performs data communication via a router or gateway. (The router relay function is not a function by which the Ethernet module works as a router.)	Section 5.3
Existence check of external device (Existence check function)	Checks whether or not the external device is working normally after a connection is established (open processing).	Sections 5.2.2 and 5.5
Communication via pairing open	Pairs and then opens a reception connection and a transmission connection (for fixed buffer).	Section 5.7
Connecting MELSOFT products (GX Developer, etc.)	MELSOFT products (GX Developer, etc.) are connected via TCP/IP communication. It is also possible to connect multiple MELSOFT products simultaneously.	Section 1.2 (3) Manual of each MELSOFT product

(3) Status check of the Ethernet module

Checks that the Ethernet module is working and can communicate normally.

Function	Description	Reference section
Self loopback test	Checks the Ethernet module’s sending/receiving function and line connection status.	Section 4.7.1
Hardware test	Tests the RAM and ROM of the Ethernet module.	Section 4.7.2
Communication error storage	When a data communication error occurs, this function stores the error information (error log), including the message subheader and IP address of the external device for a maximum of 16 pairs in the buffer memory.	Chapter 11

3.5 List of Setting Items for Ethernet Modules

The following table lists the parameter setting items that are set using Ethernet Module.

Parameter setting item			Description of setting	Function and parameter setting requirement (* 1)			Refer-ence section
				MC	Fixed	Mail	
Ethernet module settings			Settings for using the Ethernet module as a network module.	○	○	○	Section 4.6
	Operational settings		Set the common items between the modules. These settings are required for the initial processing.	○	○	○	Section 4.6
	Initial settings		Set the data communication timer values.	△ *2	△ *2	△ *2	Section 5.2
			Set the DNS server's IP address.	—	—	△	Section 10.6
	Open settings		Set up the open processing for connection in order to perform data communication with the external device.	○	○	—	Section 5.5
	Router relay parameter		Set the router relay of Ethernet.	△	△	△	Section 5.3
	E-mail settings		Perform settings for sending/receiving e-mail.	—	—	○	Section 10.6
		Send mail address setting	Set the destination mail address.	—	—	○	

○: Must be set when the applicable function is used. △ : Set as needed ×: Setting is not required.

*1 The meanings of the abbreviations used in the table above are as follows:
MC: Communication using MC protocol
Mail: E-mail
Fixed: Communication using the fixed buffer

*2 Setting is not necessary if default values are being used.

POINT

Parameters set in FX Configurator-EN (GX-Developer) are saved in the Ethernet module's flash ROM.

3.6 List of Applications and Assignments of the Buffer Memory

Data transmission/reception between Ethernet Module and PLC is performed via the Ethernet Module buffer memory (hereinafter called BFM).
Reading/writing data in the buffer memory from the PLC must be performed by the FROM/TO instructions (instructions that can access BFM).
In addition, the initial value of the buffer memory or the internally stored value is written when the power is turned on.

(1) Configuration of the buffer memory

Buffer memory consists of a user area and a system area, as listed below.

(a) User areas

1) The areas where the user writes/reads data.
2) A user area consists of a parameter area for initial processing and data communication, an area for data communication, and an area for storing communication status and communication error data.
3) Reading/writing data to the user area should be performed according to the instructions in the corresponding detailed explanation section. Data communication may take longer if continually executed; therefore, execute only when needed.

(b) System areas

The areas used by the Ethernet module

Important

Do not write data in the "system areas" of the buffer memory.
If data is written to any of the system areas, the PLC system may not operate properly.
When writing a value to the buffer memory including "System Area," pay close attention not to change the system bit.

(2) Assignments of the buffer memory

A buffer memory consists of 16 bits per address.

<Bit configuration diagram>

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

The following shows the buffer memory addresses.

(a) Initial processing parameter (BFM #0 to 31)

BFM number	Attribute *1	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save (*2)	Reference section
Decimal (Hexadecimal)							
0 to 1 (0 to 1H)	R/W	Initial processing parameter setting area	Local station Ethernet module IP address (Initial value 192.168.1.254)		3232236030 (C0A801FEH)	○	Section 4.6
2 (2H)	R/W		Special function settings • Router relay function (b5, b4) 00: Do not use (default) 01: Use Bits other than above are reserved for system use.		0 (0000H)	○	Section 5.3
3 (3H)	—		System area		—	—	—
4 (4H)	R/W		Monitoring timer	TCP ULP (existence function) timer value Setting time = setting value × 500 ms	60 (3CH)	○	Section 5.2
5 (5H)	R/W			TCP zero window timer value Setting time = setting value × 500 ms	20 (14H)		
6 (6H)	R/W			TCP resend timer value Setting time = setting value × 500 ms	20 (14H)		
7 (7H)	R/W			TCP end timer value Setting time = setting value × 500 ms	40 (28H)		
8 (8H)	R/W			IP assembly timer value Setting time = setting value × 500 ms	10 (AH)		
9 (9H)	R/W			Response monitoring timer value Setting time = setting value × 500 ms	60 (30H)		
10 (AH)	R/W			Destination existence confirmation starting interval Setting time = setting value × 500 ms	1200 (480H)		
11 (BH)	R/W			Destination existence confirmation interval timer Setting time = setting value × 500 ms	20 (14H)		
12 (CH)	R/W			Destination existence confirmation resend timer	3 (3H)		
13 to 23 (D to 17H)	—		System area		—	—	—

(Continues on the next page)

*1 Attribute R : Read, W : Write, — : Use prohibited

*2 Saving to Flash ROM from FX Configurator-EN (GX Developer) allowed/prohibited
(Settings are saved in the flash ROM of the Ethernet module.)
○: Setting allowed ×: Setting prohibited

(Continued from the previous page)

BFM number Decimal (Hexadecimal)	Attribute	Application		Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
24 (18_H)	R/W	Initial processing parameter setting area (For reinitialization)	Communication condition setting area (Operational Settings)	Communication condition setting (Operational Settings) area • Communication data code setting (b1) 0: Communication in binary code 1: Communication in ASCII code • TCP Existence confirmation setting (b4) 0: Use the Ping 1: Use the KeepAlive • Send frame setting (b5) 0: Ethernet frame 1: IEEE 802.3 frame • Initial timing setting (b8) 0: Do not wait for OPEN (communication impossible at STOP time) 1: Always wait for OPEN (communication possible at STOP time) Bits other than above are reserved for system use.	0 (0_H)	○	Section 4.6
25 (19_H)	R/W	Flash ROM control	Flash ROM writing control area	Flash ROM writing control 0000H: Normal status 9872H: Write command for configuration data to a flash ROM. Automatically returns to 0000H after writing to a flash ROM 9981H: Write command for configuration data to a flash ROM (return to the status at factory shipment). Automatically returns to 0000H after writing to a flash ROM	0 (0000_H)	×	–
26 ($1A_H$)	R/W		Flash ROM writing status	Flash ROM writing status 0000H: Initial status 9981H: During the flash ROM writing 0001H : Flash ROM writing completion 0002H: Flash ROM writing error (Flash ROM fault) The status is updated by write command	0 (0000_H)	×	
27 ($1B_H$)	R	Initial completion		0000H: Initial incompletion 0001H: Initial completion (in conjunction with b0 of BFM#28)	0 (0000_H)	×	Section 5.2.1

(Continues on the next page)

(Continued from the previous page)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
28 (1C$_H$)	R	Module status area	Stores Ethernet module status b0: INIT 1: Initial processing normally completed 0: — b1: System bit b2: 100M/10M 1: 100Mbps 0: 10Mbps/When not connected b3: ERR. 1: Setting abnormal display 0: Setting normal display b4: COM, ERR 1: Communication abnormal display 0: Communication normal display b5: 1: Initial processing abnormally completed 0: — b6: 1: Open processing abnormally completed b7: 1: Link signal ON 0: Link signal OFF b8: Connection No.1 1: Open 0: Closed b9: Connection No.2 1: Open 0: Closed b10: Connection No.3 1: Open 0: Closed b11: Connection No.4 1: Open 0: Closed b12: Connection No.5 1: Open 0: Closed b13: Connection No.6 1: Open 0: Closed b14: Connection No.7 1: Open 0: Closed b15: Connection No.8 1: Open 0: Closed b0 and b5 are cleared at re-initialization request. b6 is cleared at open request.	0 (0000$_H$)	×	Section 4.3
29 (1D$_H$)	R/W	Error code storage	Stores the first error code (when the value of this BFM number is 0000H) Writing 0000H resets the error.	0 (0000$_H$)	×	Section 11.3
30 (1E$_H$)	R	Model name storage	Model name (K7130)	7130 (1BDA$_H$)	×	—
31 (1F$_H$)	—	System area		—	—	—

(b) Communication parameter setting area (BFM #32 to 102)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
32 (20$_{H}$)	R/W	Communication parameters setting area	Connection usage setting area	Connection No. 1 • Usage of fixed buffer (b0) 0: For sending or fixed buffer communication is not executed 1: For receiving • Destination existence confirmation (b1) 0: No confirm 1: Confirm • Pairing open (b7) 0: No pairs 1: Pairs • Communication method (protocol) (b8) 0: TCP/IP 1: UDP/IP • Fixed buffer communication (b9) 0: Procedure exist 1: No procedure • MC Protocol(b10) 0: MC Protocol cannot be used 1: MC Protocol can be used • Open system (b15, b14) 00: Active open or UDP/IP 01: Use prohibited 10: Unpassive open 11: Fullpassive open A002: When MELSOFT communication in use Bits other than above are reserved for system use.	0 (0$_{H}$)	○	Section 5.5
33 (21$_{H}$)	R/W			Connection No. 2 (bit configuration is the same as connection No.1)	(Same as connection No.1)		
34 (22$_{H}$)	R/W			Connection No.3 (bit configuration is the same as connection No.1)	(Same as connection No.1)		
35 (23$_{H}$)	R/W			Connection No.4 (bit configuration is the same as connection No.1)	(Same as connection No.1)		
36 (24$_{H}$)	R/W			Connection No.5 (bit configuration is the same as connection No.1)	(Same as connection No.1)		
37 (25$_{H}$)	R/W			Connection No.6 (bit configuration is the same as connection No.1)	(Same as connection No.1)		
38 (26$_{H}$)	R/W			Connection No.7 (bit configuration is the same as connection No.1)	(Same as connection No.1)		
39 (27$_{H}$)	R/W			Connection No.8 (bit configuration is the same as connection No.1)	(Same as connection No.1)		

(Continues on the next page)

(Continued from the previous page)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name			Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
40 (28$_H$)	R/W	Communication parameters setting area	Communication address setting area	Connection No. 1	Local station Port No.	0 (0$_H$)	○	Section 5.5
41 to 42 (29 to 2A$_H$)	R/W				Destination IP address	0 (0$_H$)	○	Section 5.5
43 (2B$_H$)	R/W				Destination Port No.	0 (0$_H$)	○	Section 5.5
44 to 46 (2C to 2E$_H$)	R/W				Destination Ethernet address	4294967295 (FFFFFFFFFFFF$_H$)	×	–
47 to 53 (2F to 35$_H$)	R/W			Connection No. 2	(Same as connection No. 1)			
54 to 60 (36 to 3C$_H$)	R/W			Connection No. 3	(Same as connection No. 1)			
61 to 67 (3D to 43$_H$)	R/W			Connection No. 4	(Same as connection No. 1)			
68 to 74 (44 to 4A$_H$)	R/W			Connection No. 5	(Same as connection No. 1)			
75 to 81 (4B to 51$_H$)	R/W			Connection No. 6	(Same as connection No. 1)			
82 to 88 (52 to 58$_H$)	R/W			Connection No. 7	(Same as connection No. 1)			
89 to 95 (59 to 5F$_H$)				Connection No. 8	(Same as connection No. 1)			
96 (60$_H$)	–		System area			–	–	–
97 to 99 (61 to 63$_H$)	R		System area (MAC address stored)			(Product characteristic value)	–	–
100 to 102 (64$_H$ to 66$_H$)	–		System area			–	–	–

(c) Communication status storage area (BFM #103 to 226)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name				Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
103 to 104 (67H to 68H)	–	Communication status storage area	System area				–	–	–
105 (69H)	R		Area for initial processing	Initial error code			0 (0H)	×	Section 11.4
106 to 107 (6A to 6BH)	R			Local station IP address			0 *1 (0H)	×	–
108 to 110 (6C to 6EH)	R			Local station Ethernet address			0 *1 (0H)	×	–
111 to 119 (6F to 77H)	–			System area			–	–	–
120 (78H)	R		Connection information area	Connection No. 1	Local station Port No.		0 (10H)	×	–
121 to 122 (79 to 7AH)	R				Destination IP address		0 (0H)	×	–
123 (7BH)	R				Destination Port No.		0 (0H)	×	–
124 (7CH)	R				Open error code		0 (0H)	×	Section 11.4
125 (7DH)	R				Fixed buffer sending error code		0 (0H)	×	Section 11.4
126 (7EH)	R				Connection end code		0 (0H)	×	Section 11.4
127 (7FH)	R				Fixed buffer communication time (unit : 10ms)	Maximum value	0 (0H)	×	–
128 (80H)	R					Minimum value	0 (0H)	×	–
129 (81H)	R					Current value	0 (0H)	×	–
130 to 139 (82 to 8BH)	R			Connection No. 2	(Same as connection No. 1)				
140 to 149 (8C to 95H)	R			Connection No. 3	(Same as connection No. 1)				
150 to 159 (96 to 9FH)	R			Connection No. 4	(Same as connection No. 1)				
160 to 169 (A0 to A9H)	R			Connection No. 5	(Same as connection No. 1)				
170 to 179 (AA to B3H)	R			Connection No. 6	(Same as connection No. 1)				
180 to 189 (B4 to BDH)	R			Connection No. 7	(Same as connection No. 1)				
190 to 199 (BE to C7H)	R			Connection No. 8	(Same as connection No. 1)				
200 (C8H)	R		Area for module status	Stores Ethernet module status (The same specification as BFM#28 "Module status area")			0 *2 (0H)	×	–
201 (C9H)	R			Hub connection status area • Communication mode (b9) 0: Half duplex 1: Full duplex • Hub connection status (b10) 0: Hub not connected/disconnected 1: Hub connected • Data transmission speed (b14) 0: Operating at 10BASE-T 1: Operating at 100BASE-TX Bits other than above are reserved for system use.			–	×	Section 5.8

(Continues on the next page)

(Continued from the previous page)

BFM number	Attribute	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save	Referenc e section
Decimal (Hexadecimal)							
202 (D0H)	R	Communication status storage area	Operation mode settings 0: Online 1: Offline 2: Self loopback test 3: Hardware test 4 to F: System area		0 ＊3 (0H)	×	—
203 (CBH)	R	Communication status storage area	Module status area	Status of settings with FX Configurator-EN • Communication data code setting (b1) 0: Communication in binary code 1: Communication in ASCII code • TCP Existence confirmation setting (b4) 0: Use the Ping 1: Use the KeepAlive • Send frame setting (b5) 0: Ethernet frame 1: IEEE802.3 frame • Initial timing setting (b8) 0: Do not wait for OPEN (Communications impossible at STOP time) 1: Always wait for OPEN (Communication possible at STOP time) Bits other than above are reserved for system use.	4 (04H)	○	Section 4.7
204 tos 226 (E0 to E2H)	—		System area		—	—	—

＊1 Initial processing writes IP and Ethernet address.

＊2 It varies after initialization.

＊3 Values written to BFM #202 will become active the next time the module is powered ON.
(The value returns to 0 when each running mode execution is complete.)

(d) Error log partition area (BFM #227 to 511)

BFM number	Attribute	Application	Name			Initial value	Flash ROM save	Reference section
Decimal (Hexadecimal)						Decimal (Hexadecimal)		
227 (E3$_H$)	R	Error log area	Number of error occurrences			0 (0$_H$)	×	Section 11.4
228 (E4$_H$)	R		Error log write pointer					
229 (E5$_H$)	R		Error log block area	Error log block 1	Error code/end code	0 (0$_H$)	×	Section 11.4
230 (E6$_H$)	R				Subheader			
231 (E7$_H$)	R				Command code			
232 (E8$_H$)	R				Connection No.			
233 (E9$_H$)	R				Local station Port No.			
234 to 235 (EA to EB$_H$)	R				Destination IP address			
236 (EC$_H$)	R				Destination Port No.			
237 (ED$_H$)	R				System area	—	—	—
238 to 246 (EE to F6$_H$)	R			Error log block 2	(Same as error log block 1)			
247 to 255 (F7 to FF$_H$)	R			Error log block 3	(Same as error log block 1)			
256 to 264 (100 to 108$_H$)	R			Error log block 4	(Same as error log block 1)			
265 to 273 (109 to 111$_H$)	R			Error log block 5	(Same as error log block 1)			
274 to 282 (112 to 11A$_H$)	R			Error log block 6	(Same as error log block 1)			
283 to 291 (11B to 123$_H$)	R			Error log block 7	(Same as error log block 1)			
292 to 300 (124 to 12C$_H$)	R			Error log block 8	(Same as error log block 1)			
301 to 309 (12D to 135$_H$)	R			Error log block 9	(Same as error log block 1)			
310 to 318 (136 to 13E$_H$)	R			Error log block 10	(Same as error log block 1)			
319 to 327 (13F to 147$_H$)	R			Error log block 11	(Same as error log block 1)			
328 to 336 (148 to 150$_H$)	R			Error log block 12	(Same as error log block 1)			
337 to 345 (151 to 159$_H$)	R			Error log block 13	(Same as error log block 1)			
346 to 354 (15A to 162$_H$)	R			Error log block 14	(Same as error log block 1)			
355 to 363 (163 to 16B$_H$)	R			Error log block 15	(Same as error log block 1)			
364 to 372 (16C to 174$_H$)	R			Error log block 16	(Same as error log block 1)			
373 to 375 (175 to 177$_H$)	—		System area			—	—	—

(Continues on the next page)

(Continued from the previous page)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name			Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
376 to 377 (178 to 179H)	R	Error log area	Status for each protocol	IP	Received IP packet count	0 (0H)	×	Section 11.3
378 to 379 (17A to 17BH)	R				Received IP packet count discarded due to sum check error			
380 to 381 (17C to 17DH)	R				Sent IP packet total count			
382 to 397 (17E to 18DH)	—				System area	—	—	—
398 to 399 (18E to 18FH)	R				Simultaneous transmission error detection count Receiving descripter deletion count	0 (0H)	×	Section 11.5 POINT (3)
400 to 407 (190 to 197H)	—				System area	—	—	—
408 to 409 (198 to 199H)	R			ICMP	Received ICMP packet count	0 (0H)	×	
410 to 411 (19A to 19BH)	R				Received ICMP packet count discarded due to sum check error			
412 to 413 (19C to 19DH)	R				Sent ICMP packet total count			
414 to 415 (19E to 19FH)	R				Echo request total count of received ICMP packets			
416 to 417 (1A0 to 1A1H)	R				Echo reply total count of sent ICMP packets			
418 to 419 (1A2 to 1A3H)	R				Echo request total count of sent ICMP packets			
420 to 421 (1A4 to 1A5H)	R				Echo reply total count of received ICMP packets			
422 to 439 (1A6 to 1B7H)	—				System area	—	—	
440 to 441 (1B8 to 1B9H)	R			TCP	Received TCP packet count	0 (0H)	×	
442 to 443 (1BA to 1BBH)	R				Received TCP packet count discarded due to sum check error			
444 to 445 (1BC to 1BDH)	R				Sent TCP packet total count			
446 to 471 (1BE to 1D7H)	—				System area	—	—	
472 to 473 (1D8 to 1D9H)	R			UDP	Received UDP packet count	0 (0H)	×	—
474 to 475 (1DA to 1DBH)	R				Received UDP packet count discarded due to sum check error			
476 to 477 (1DC to 1DDH)	R				Sent UDP packet total count			
478 to 481 (1DE to 1E1H)	—				System area	—	—	
482 to 491 (1E2 to 1EBH)	—		System area			—	—	
492 to 493 (1EC to 1EDH)	R		Receiving error		Framing error count	0 (0H)	×	
494 to 495 (1EE to 1EFH)	R				Overflow count	0 (0H)		
496 to 497 (1F0 to 1F1H)	R				CRC error count	0 (0H)		
498 to 511 (1F2 to 1FFH)	—		System area			—	—	

(e) Router relay parameter setting area (BFM #512 to 1599)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
512 to 513 (200 to 201H)	R/W	Router relay parameter setting area	Sub-net mask	0 (0H)	○	Section 5.3
514 to 515 (202 to 203H)	R/W		Default router IP address			
516 to 1599 (204 to 63FH)	—		System area	—	—	—

(f) Open/close control (BFM #1600 to 1663)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
1600 (640H)	R/W	Re-initialization	• Write value Re-initialization request 0001H : Re-initialization ([COM.ERR.] LED OFF) 0002H : Re-initialization after reading data in Flash ROM ([COM.ERR.] LED OFF) 0005H : Re-initialization ([COM.ERR.] LED retains status.) 0006H : Re-initialization after reading data in Flash ROM ([COM.ERR.] LED retains status.) • Read value 0008H : During re-initialization 8000H : At re-initialization completion C000H : At re-initialization error • In the re-initialization standby, the write value is read directly.	0 (0000H)	×	—
1601 (641H)	R/W	COM.ERR. Off	[COM.ERR.] Off request • Write value 0001H: [COM.ERR.] LED Off request (In standby until extinction) • Read value 0002H: Stores at normal completion 0001H is read while ready until LED OFF operation starts	0 (0000H)	×	—

(Continues on the next page)

(Continued from the previous page)

BFM number	Attribute	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
Decimal (Hexadecimal)							
1602 (643$_H$)	R/W	For open/close control	Open/Close command/status	Connection No.1 Open/Close command • Write value 0001H: Open command 8000H: Close command • Read value b0=0 : Open command enabled b0=1 : Open command disabled 0000H : Close status (Including when FIN is sent from the external device) 0003H : Open request or Listen being processed (b1, b0: ON) 0005H : Open status (b2, b0: ON) 0009H : Close request being processed (b3, b0: ON) 0010H : Open error (Closed status) (b4: ON) 0015H : Open error (Open status) (b5: ON) 0001H : During open standby 8000H : During close standby	0 (0000$_H$)	×	Chapter 5
1603 (644$_H$)	R/W			Connection No.2 Open/Close command	(Same as connection No.1)		
1604 (645$_H$)	R/W			Connection No.3 Open/Close command	(Same as connection No.1)		
1605 (646$_H$)	R/W			Connection No.4 Open/Close command	(Same as connection No.1)		
1606 (647$_H$)	R/W			Connection No.5 Open/Close command	(Same as connection No.1)		
1607 (648$_H$)	R/W			Connection No.6 Open/Close command	(Same as connection No.1)		
1608 (649$_H$)	R/W			Connection No.7 Open/Close command	(Same as connection No.1)		
1609 (650$_H$)	R/W			Connection No.8 Open/Close command	(Same as connection No.1)		

(Continues on the next page)

(Continued from the previous page)

<table>
<tr><th>BFM number
Decimal
(Hexadecimal)</th><th>Attribute</th><th>Application</th><th colspan="2">Name</th><th>Initial value
Decimal
(Hexadecimal)</th><th>Flash ROM save</th><th>Reference section</th></tr>
<tr><td>1610
(651_H)</td><td>R/W</td><td>For fixed buffer communication</td><td>Fixed buffer communication
Transmission/reception request/Execution result</td><td>When sending request or reception complete confirmation signal of connection No.1
• For sending
1) Write value
0001H : Sending request
2) Read value
0001H : Transmission ready
0002H : Between data being sent to while waiting for a response (only when procedure exists)
0004H : Stored when transmission complete If instantly receiving the next data, depending on the timing of FROM instruction, 0001H is stored without reading 0004H
0008H : Stored when transmission error
0000H : Stored when the open processing or close processing is completed
• For receiving
1) Write value
<table>
<tr><th>Write value</th><th>Procedure</th><th>No procedure</th></tr>
<tr><td>0002H</td><td>Response transmission to an external device</td><td>Allows the following data to be received</td></tr>
</table>
2) Read value
<table>
<tr><th>Read value</th><th>Procedure</th><th>No procedure</th></tr>
<tr><td>0001H</td><td colspan="2">At data reception</td></tr>
<tr><td>0004H</td><td>Response transmission complete</td><td>Acceptance for the data reception complete</td></tr>
<tr><td>0008H</td><td colspan="2">Receiving process cannot be normally completed</td></tr>
<tr><td>0000H</td><td colspan="2">Stores after open/close process</td></tr>
</table>
</td><td>0
(0000_H)</td><td>×</td><td>Chapter 6,
Chapter 7</td></tr>
</table>

(Continues on the next page)

(Continues on the next page)

BFM number	Attribute	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
Decimal (Hexadecimal)							
1611 (652$_H$)	R/W		Fixed buffer communication Transmission/reception request/Execution result	When sending request or reception complete confirmation signal of connection No.2	(Same as connection No.1)		
1612 (653$_H$)	R/W			When sending request or reception complete confirmation signal of connection No.3	(Same as connection No.1)		
1613 (654$_H$)	R/W			When sending request or reception complete confirmation signal of connection No.4	(Same as connection No.1)		
1614 (655$_H$)	R/W			When sending request or reception complete confirmation signal of connection No.5	(Same as connection No.1)		
1615 (656$_H$)	R/W			When sending request or reception complete confirmation signal of connection No.6	(Same as connection No.1)		
1616 (657$_H$)	R/W			When sending request or reception complete confirmation signal of connection No.7	(Same as connection No.1)		
1617 (658$_H$)	R/W			When sending request or reception complete confirmation signal of connection No.8	(Same as connection No.1)		
1618 to 1663 (659 to 67F$_H$)	—		System area		—	—	—

(g) Fixed buffer area (BFM #1664 to 9855)

BFM number	Attribute	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
Decimal (Hexadecimal)							
1664 (680$_H$)	R/W	Fixed buffer data area	Fixed buffer No. 1	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
1665 to 2687 (681 to A7F$_H$)	R/W			Fixed buffer data			
2688 (A80)	R/W		Fixed buffer No.2	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
2689 to 3711 (A81 to E7F$_H$)	R/W			Fixed buffer data			
3712 (E80$_H$)	R/W		Fixed buffer No.3	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
3713 to 4735 (E81 to 127F$_H$)	R/W			Fixed buffer data			
4736 (1280$_H$)	R/W		Fixed buffer No.4	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
4737 to 5759 (1281 to 167F$_H$)	R/W			Fixed buffer data			
5760 (1680$_H$)	R/W		Fixed buffer No.5	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
5761 to 6783 (1681 to 1A7F$_H$)	R/W			Fixed buffer data			
6784 (1A80$_H$)	R/W		Fixed buffer No.6	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
6785 to 7807 (1A81 to 1E7F$_H$)	R/W			Fixed buffer data			
7808 (1E80F$_H$)	R/W		Fixed buffer No.7	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
7809 to 8831 (1E81 to 227F$_H$)	R/W			Fixed buffer data			
8832 (2280$_H$)	R/W		Fixed buffer No.8	Data length	0 (0$_H$)	×	Chapter 6, Chapter 7
8833 to 9855 (2281 to 267F$_H$)	R/W			Fixed buffer data			

(h) Mail send parameter settings (BFM #9856 to 14499)

BFM number	Attribute	Application	Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
Decimal (Hexadecimal)						
9856 to 9887 (2680 to 269F$_H$)	R/W	Shared area for e-mail buffers	Local station mail address (Maximum of 64 words)	"\0"	○	Chapter 10
9888 to 9903 (26A0 to 26AF$_H$)	—		System area	—	—	Chapter 10
9904 (26B0$_H$)	R/W		Flag indication whether checks (b15) are received OFF: No check ON: Check Check unit designation (b14 to b12) 0: Hour, 1: Minute, 2: Second Check interval (b11 to b0) For hour : 1 to 24 For minute : 1 to 1440 For second: 30 to 3600	1005$_H$ (5 minutes)	○	Chapter 10
9905 to 9912 (26B1 to 26B8$_H$)	—		System area	—	—	Chapter 10
9913 to 9920 (26B9$_H$ to 26C0)	R/W		Password to the POP3 server (16 characters)	"\0"	○	Chapter 10
9921 (26C1$_H$)	R/W		Method for designating SMTP server 0: Domain name designation 1: IP address designation (Decimal) 2: IP address designation (Hexadecimal)	1	○	Chapter 10
9922 to 9953 (26C2$_H$ to 26E1)	R/W		Domain name of the SMTP server (maximum 64 characters) or IP address ∗1	0.0.0.0	○	Chapter 10
9954 (26E2$_H$)	R/W		Method for designating POP3 server 0: Domain name designation 1: IP address designation (Decimal) 2: IP address designation (Hexadecimal)	1	○	Chapter 10
9955 to 9986 (26E3$_H$ to 2702)	R/W		Domain name of the POP3 server (maximum 64 words) or IP address ∗1	0.0.0.0	○	Chapter 10
9987 to10018 (2703$_H$ to 2722)	R/W		Send destination address 1	"\0"	○	Chapter 10
10019 to 10050 (2723$_H$ to 2742)	R/W		Send destination address 2	"\0"	○	Chapter 10
10051 to 10082 (2743$_H$ to 2762)	R/W		Send destination address 3	"\0"	○	Chapter 10
10083 to 10114 (2763$_H$ to 2782)	R/W		Send destination address 4	"\0"	○	Chapter 10
10115 to 10146 (2783$_H$ to 27A2)	R/W		Send destination address 5	"\0"	○	Chapter 10
10147 to 10178 (27A3$_H$ to 27C2)	R/W		Send destination address 6	"\0"	○	Chapter 10
10179 to 10210 (27C3$_H$ to 27E2)	R/W		Send destination address 7	"\0"	○	Chapter 10
10211 to 10242 (27E3$_H$ to 2802)	R/W		Send destination address 8	"\0"	○	Chapter 10
10243 to 10274 (2803$_H$ to 2822)	R/W		Send destination address 9	"\0"	○	Chapter 10
10275 to 10306 (2823$_H$ to 2842)	R/W		Send destination address 10	"\0"	○	Chapter 10
10307 to 10332 (2843$_H$ to 285C$_H$)	—		System area	—	—	Chapter 10

(Continues on the next page)

(Continued from the previous page)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
10333 (285DH)	R/W	Mail sending command	• Write value Mail sending instruction (0001H) • Read value Returns to “0000H” when mail sending commences Multiple requesting exists.	0 (0000$_{H}$)	×	Chapter 10
10334 (285EH)	R	Mail sending status	Initial status (0000H) Mail sending normally completed (0001H) Mail sending abnormally completed (0002H) While mail being sent (8000H)	0 (0000$_{H}$)	×	Chapter 10
10335 (285FH)	R	Error codes at sending e-mail	When sending mail to a single address, depending on the send (BFM #10333: 0001H), the memory is cleared and the first error code found is stored. When sending mail to multiple addresses, the first error code found is stored, even after errors. " The second error and later errors are not stored.	0 (0000$_{H}$)	×	Chapter 10
10336 (2860H)	R	Sending result of e-mail	Send destination address 1 Success (b0: ON) / Failure (b0: OFF) to Send destination address 10 Success (b9: ON) / Failure (b9: OFF)	0 (0000$_{H}$)	×	Chapter 10
10337 (2861H)	R/W	Sending designation of e-mail	Turns on the bit of the mail address that sends an e-mail. Sending destination address 1 Enable (b0: ON) / Disable (b0: OFF) to Sending destination address 10 Enable (b9: ON) / Disable (b9: OFF) Initialized into 0000H by mail sending instruction, and the bit is updated sequentially depending on the sending result.	0 (0000$_{H}$)	×	Chapter 10
10338 (2862H)	R/W	Attached file format	0000H: Sends the attached file in binary format (maximum 2048 words) 0001H: Sends the attached file in ASCII format (maximum 2048 words) 0002H: Sends the attached file in CSV format (maximum 2048 words) 0004H: Sends the attached file as a text mail (maximum 256 words)	0 (0000$_{H}$)	×	Chapter 10
10339 to 10402 (2863 to 28A2H)	R/W	Subject	Stores a subject (maximum 128 words)	\0	×	–
10403 (28A3H)	R/W	Word count of attached file	Designates the word count of main text (0 to 256)/attached file (0 to 2048)	0 (0000$_{H}$)	×	–
10404 to 12451 (28A4 to 30A3H)	R/W	Main text of attached file	Stores a main text or an attached file	\0	×	–
12452 to 14499 (30A4 to 38A3H)	–	System area		–	–	–

*1 (example)192, 168, 0, 10 →

C0	A8	00	0A
4)	3)	2)	1)

	H	L
BFM 9922	2)	1)
BFM 9923	4)	3)

(i) Mail receive parameter settings (BFM #14500 to 20479)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name	Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
14500 (38A4H)	R/W	Mail receiving command	• Write value Receive mail instruction (0001H) • Read value Initial status (0000H) Returns to "0000H" When mail receiving starts. Multiple requesting exists (0002H)	0 (0000H)	×	Chapter 10
14501 (38A5H)	R	Mail receiving status	Initial status (0000H) Mail receiving normally completed (0001H) Mail receiving abnormally completeds (0002H) While mail being received (8000H)	0 (0000H)	×	Chapter 10
14502 (38A6H)	R	Error status when receiving mail	Stores the error code when mail is received It will be cleared by the mail receiving command (BFM#14500:0001H)	0 (0000H)	×	Chapter 10
14503 to 14534 (38A7 to 38C6H)	R	Message ID	Stores a message ID of received mail (maximum 64 characters)	\0	×	Chapter 10
14535 (38C7H)	R	Character string length of message header	Stores the character string length of the message header	0 (0000H)	×	Chapter 10
14536 to 14727 (38C8 to 3987H)	R	Message header	Stores the message header (maximum 384 characters)	\0	×	Chapter 10
14728 to 14759 (3988 to 39A7H)	R	Send source mail address	Stores the send source mail address	\0	×	Chapter 10
14792 to 14855 (39C8 to 3A07H)	R	Subject	Stores a subject (maximum 128 characters)	\0	×	Chapter 10
14856 to 14871 (3A08 to 3A17H)	R	File name of attached file	Stores a file name of an attached file (maximum 32 characters)	\0	×	Chapter 10
14872 (3A18H)	R	Word count of attached file	Designates the word count of main text/attached file (0 to 2048) The file exceeding 2048 words in discarded.	0 (0000H)	×	Chapter 10
14873 to 16920 (3A19 to 4218H)	R	Main text of attached file	Stores the main text or an attached file	\0	×	Chapter 10
16921 to 20479 (4219 to 4FFFFH)	—	System area		—	—	Chapter 10

(j) Connection status storage area (BFM #20480 to 22639)

BFM number	Attribute	Application	Name		Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
Decimal (Hexadecimal)							
20480 (5000$_H$)	R	Connection status storage area	Connection status information area	Open complete signal 0: Open incomplete 1: Open completed • Connection No. 1 (b0) • Connection No. 2 (b1) to • Connection No. 8 (b7) TCP : 1 by connection establishment UDP : 1 by communication enabled	0 (0$_H$)	×	Section 5.6
20481 (5001$_H$)	–			System area	–	–	–
20482 (5002$_H$)	R			Open request signal 0: No open request 1: Open being requested • Connection No. 1 (b0) • Connection No. 2 (b1) to • Connection No. 8 (b7) TCP Passive : 1 by listen Active : 1 by open process UDP : 1 by communication enabled	0 (0$_H$)	×	Section 5.6
20483 to 20484 (5003 to 5004$_H$)	–		Connection status information area	System area	–	–	–
20485 (5005$_H$)	R		Fixed buffer information area	Fixed buffer reception status signal 0: Data not received 1: Data being received • Connection No. 1 (b0) • Connection No. 2 (b1) to • Connection No. 8 (b7)	0 (0$_H$)	×	Chapter 7
20486 to 20504 (5006 to 5018$_H$)	–		System area		–	–	–
20505 to 20506 (5019 to 501A$_H$)	R/W		DNS server 1 IP address		0 (0$_H$)	○	
20507 to 20508 (501B to 501C$_H$)	R/W		DNS server 2 IP address		0 (0$_H$)	○	
20509 to 20510 (501D to 501E$_H$)	R/W		DNS server 3 IP address		0 (0$_H$)	○	
20511 to 20512 (501F to 0520$_H$)	R/W		DNS server 4 IP address		0 (0$_H$)	○	
20513 to 20591 (5021 to 506F$_H$)	–		System area		–	–	–
20592 to 20693 (5070 to 50D5$_H$)	–	System area			–	–	–
20694 to 20994 (50D6$_H$ to 5202$_H$)	–	"System switchover request when interrupt detection issued" status storage area	System area		–	–	–
20995 (5203$_H$)	R/W		Interrupt detection count		0 (0$_H$)	×	–
20996 to 22639 (5204 to 586F$_H$)	–	System area			–	–	–

(k) E-mail status storage area (BFM #22640 to 31999)

BFM number Decimal (Hexadecimal)	Attribute	Application	Name			Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
22640 (5870$_H$)	R	E-mail status storage area	Receive	Number of mails remaining on the server		0 (0$_H$)	×	Section 11.4
22641 to 22642 (5871 to 5872$_H$)	–			System area		–	–	–
22643 (5873$_H$)	R			Normally Received count		0 (0$_H$)	×	Section 11.4
22644 (5874$_H$)	R			Attached files received count				
22645 (5875$_H$)	R			Server inquiry count				
22646 (5876$_H$)	R			Server communication error count				
22647 (5877$_H$)	R			Error log write count				
22648 (5878$_H$)	R			Receiving error log write pointer				
22649 (5879$_H$)	R			Error log block 1	Error code	0 (0$_H$)	×	Section 11.4
22650 (587A$_H$)	R				Command code			
22651 to 22658 (587B to 5882$_H$)	R				From			
22659 to 22662 (5883 to 5886$_H$)	R				Date ＊1			
22663 to 22692 (5887 to 58A4$_H$)	R				Subject			
22693 to 22736 (58A5 to 58D0$_H$)	R			Error log block 2	(Same as error log block 1)			
22737 to 22780 (58D1 to 58FC$_H$)	R			Error log block 3	(Same as error log block 1)			
22781 to 22824 (58FD to 5928$_H$)	R			Error log block 4	(Same as error log block 1)			
22825 to 22868 (5929 to 5954$_H$)	R			Error log block 5	(Same as error log block 1)			
22869 to 22912 (5955 to 5980$_H$)	R			Error log block 6	(Same as error log block 1)			
22913 to 22956 (5981 to 59AC$_H$)	R			Error log block 7	(Same as error log block 1)			
22957 to 23000 (59AD to 59D8$_H$)	R			Error log block 8	(Same as error log block 1)			
23001 to 23044 (59D9 to 5A04$_H$)	R			Error log block 9	(Same as error log block 1)			
23045 to 23088 (5A05 to 5A30$_H$)	R			Error log block 10	(Same as error log block 1)			
23089 to 23132 (5A31 to 5A5C$_H$)	R			Error log block 11	(Same as error log block 1)			
23133 to 23176 (5A5D to 5A88$_H$)	R			Error log block 12	(Same as error log block 1)			
23177 to 23220 (5A89 to 5AB4$_H$)	R			Error log block 13	(Same as error log block 1)			
23221 to 23264 (5AB5 to 5AE0$_H$)	R			Error log block 14	(Same as error log block 1)			
23265 to 23308 (5AE1 to 5B0C$_H$)	R			Error log block 15	(Same as error log block 1)			
23309 to 23352 (5B0D to 5B38$_H$)	R			Error log block 16	(Same as error log block 1)			

(Continues on the next page)

(Continued from the previous page)

NFM number	Attribute	Application	Name			Initial value Decimal (Hexadecimal)	Flash ROM save	Reference section
Decimal (Hexadecimal)								
23353 to 23354 (5B39 to 5B3A$_H$)	–	E-mail status storage area	Send	System area		–	–	–
23355 (5B3B$_H$)	R			Number of mails normally completed		0 (0$_H$)	×	Section 11.4
23356 (5B3C$_H$)	R			Attached files sent count				
23357 (5B3D$_H$)	R			Sent to the server count				
23358 (5B3E$_H$)	R			Number of mails abnormally completed				
23359 (5B3F$_H$)	R			Error log write count				
23360 (5B40$_H$)	R			Error log write pointer				
23361 (5B41$_H$)	R			Error log block 1	Error code	0 (0$_H$)	×	Section 11.4
23362 (5B42$_H$)	R				Command code			
23363 to 23370 (5B43 to 5B4A$_H$)	R				To			
23371 to 23374 (5B4B to 5B4E$_H$)	R				Date ＊1			
23375 to 23404 (5B4F to 5B6C$_H$)	R				Subject			
23405 to 23448 (5B6D to 5B98$_H$)	R			Error log block 2	(Same as error log block 1)			
23449 to 23492 (5B99 to 5BC4$_H$)	R			Error log block 3	(Same as error log block 1)			
23493 to 23536 (5BC5 to 5BF0$_H$)	R			Error log block 4	(Same as error log block 1)			
23537 to 23580 (5BF1 to 5C1C$_H$)	R			Error log block 5	(Same as error log block 1)			
23581 to 23624 (5C1D to 5C48$_H$)	R			Error log block 6	(Same as error log block 1)			
23625 to 23668 (5C49 to 5C74$_H$)	R			Error log block 7	(Same as error log block 1)			
23669 to 23712 (5C75 to 5CA0$_H$)	R			Error log block 8	(Same as error log block 1)			
23713 to 24575 (5CA1 to 5FFF$_H$)	–		System area			—	—	—
24576 to 31799 (6000 to 7C37H)	–	System area				—	—	—

(Continues on the next page)

＊1 The timestamp for an email that had a communication error with the mail server is stored in BCD code in the following manner.

b15 to b8	b7 to b0
Month (01$_H$ to 12$_H$)	Lower 2-digits of year (00$_H$ to 99$_H$)

b15 to b8	b7 to b0
Hour (00$_H$ to 23$_H$)	Date (01$_H$ to 31$_H$)

b15 to b8	b7 to b0
Second (00$_H$ to 59$_H$)	Minutes (00$_H$ to 59$_H$)

b15 to b8	b7 to b0
Higher 2-digits of year (00$_H$ to 99$_H$)	Day of the week (0 to 6) SUN:0 to SAT:6

(Continued from the previous page)

BFM numebr Decimal (Hexadecimal)	Attribute	Application	Name		Initial value Decimal (Hexadecimal))	Flash ROM save	Reference section
31800 (7C38H)	R/W	Error information clear	Error information to be cleared	Designates the error information to be cleared 0000H : Clears initial error code (BFM#105) 0001H to 0008H : Clears the open error code of connection 1 to 8 (BFM#124, 134, 144, 154, 164, 174, 184, 194). Each of the following connection corresponds. 0001H: Connection 1 0002H: Connection 2 0003H: Connection 3 0004H: Connection 4 0005H: Connection 5 0006H: Connection 6 0007H: Connection 7 0008H: Connection 8 0100H : Clears the error log block area (BFM#227 to 372) to 0. 0101H : Clears communication status (status for each protocol) (BFM#376 to 511) 0102H : Clears communication status (E-mail receive status) (BFM#22641 to 23352) 0103H : Clears communication status (E-mail receive status) (BFM#23353 to 23712) FFFFH : Clears all of the above	0 (0000H)	×	—
31801 (7C39H)	R/W		COM.ERR. Off	Designates the function to be cleared 0000H : [COM.ERR.] LED Off, error code clear FFFFH: Error log clear	0 (0000H)	×	—
31802 (7C40H)	R/W		Executing error clear	Executes error clear • Write value 0001H: Executes error clear • Write value 0002H: Stored at error clear The result is stored in BFM#31803	0 (0000H)		
31803 (7C41H)	R/W		Result of execution error clear	Stores result of execution error clear 0000H: Normally executed Other than 0000H: Error code	0 (0000H)		
31804 to 31999 (7C42 to 7CFFH)	–		System area		—	—	—
32000 to 32639 (7D00 to 7F7FH)	–	System area			—	—	—

4 SETTINGS AND PROCEDURES PRIOR TO OPERATION

This chapter explains the settings and procedures required prior to operating the Ethernet module in a system.

4.1 Loading and Installation

This section explains precautions for Ethernet module handling from unpacking to installation, as well as the installation environment common to all modules.
For more details on module mounting and installation, refer to the Installation Manual supplied with the Ethernet module.

4.1.1 Handling precautions

The following explains precautions for Ethernet module handling:

(1) Since it is made of resin, the Ethernet module case should not be dropped or subjected to any shock.

(2) Tighten the module terminal and fixing screws within the following specified clamping torque range:

Screw location	Tightening torque range
External power supply terminal screw (M3 screw)	0.5 to 0.8 N•m
Unit-affixing screw (M4 screw)	0.78 to 1.08 N•m

DANGER

- Do not touch the terminals and connectors while the power is on.
  Doing so may result in electric shocks and malfunctions.
- Make sure to turn off all phases of the external power supply before cleaning or re-tightening the screws. Failure to do so may cause damages or malfunctions of the module.
  If the screws are loose, it may cause the module to short circuit, malfunction or fall off.
  Tightening the screws excessively may damage the screws and/or the module and cause the module to short circuit, malfunction or fall off.
- Do not write any data in the [system area] of the buffer memory on the module.

⚠CAUTION

- Be careful not to let any foreign matter such as wire chips get inside the module. They may cause fire, as well as breakdowns and malfunctions of the module.
- Never disassemble or modify the module. This may cause breakdowns, malfunctions, injuries or fire.
- When attaching or removing the base module and all expansion module special blocks, make sure to cut all external power supplies. Failure to do so may cause the module to breakdown or malfunction.
- Tighten the terminal screws using the specified torque. If the terminal screws are loose, it may cause the module to short-circuit, malfunction or fall off. Tightening the terminal screws excessively may damage the screws and/or the module and cause the module to short-circuit, malfunction or fall off.
- Do not directly touch the conducting parts and electronic parts of the module. This may cause the module to malfunction or fail.
- When disposing of this product, treat it as industrial waste.
- A protective sheet is pasted on the upper part of the module in order to prevent foreign matter such as wire chips to get inside the module while wiring. Do not remove this protective sheet during wiring work. However, be sure to remove the protective sheet before operating the module to allow heat radiation during operation.
- Before performing control operations (especially data modifications, program modifications and operating-status modifications) on the PLC in operation, be sure to confirm the safety.
- For basic module status control (remote RUN/STOP and similar controls) previous parameter settings should normally be set to open. If these conditions are not met, after a remote STOP, the communication line is closed from another node and cannot be re-opened, so remote RUN cannot start.

4.1.2 Installation environment

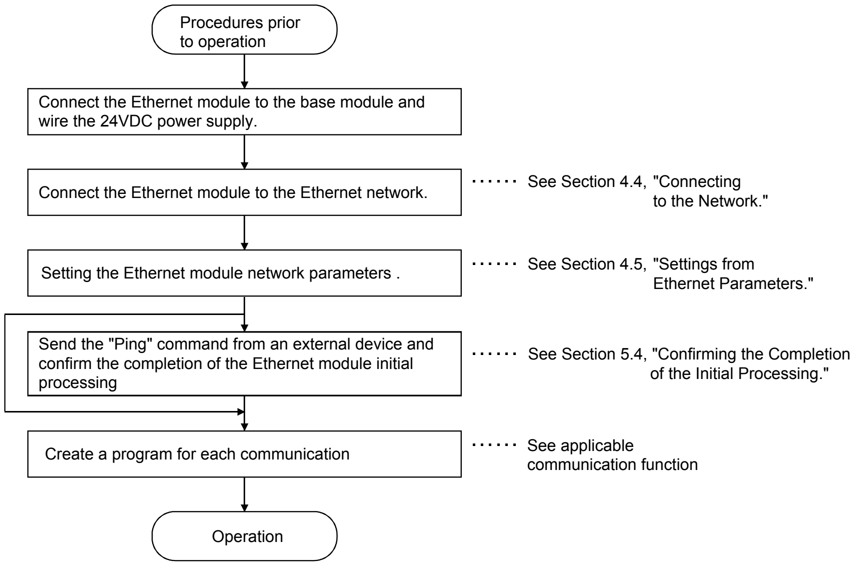
This section explains the installation environment for the PLC. When installing the PLC, the following environments must be avoided:

- Locations where the ambient temperature exceeds the range of 0 to 55 °C.
- Locations where the ambient humidity exceeds the range of 5 to 95 % RH.
- Locations where condensation occurs due to a sudden temperature change.
- Locations where there are corrosive or flammable gases.
- Locations exposed to considerable amounts of conductive powdery substances such as dust and iron filing, oil mist, salt, or organic solvents.
- Locations exposed to direct sunlight.
- Locations exposed to strong electric or magnetic fields.
- Locations where vibrations or impacts are directly applied to the main unit.

⚠CAUTION	• Use the PLC in the operating environment that meets the general specifications described in the user's manual of the PLC to use. Using the PLC in any other operating environments may cause electric shocks, fires or malfunctions, or may damage or degrade the module.

4.2 Settings and Procedures Prior to Starting the Operation

The following shows a flow of the procedure that is required prior to operating:

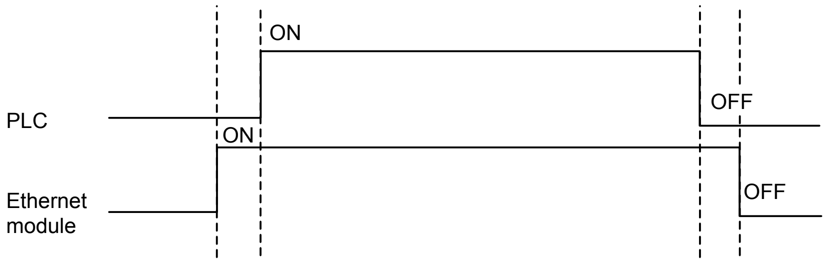

Procedures prior to operation
Connect the Ethernet module to the base module and wire the 24VDC power supply.
Connect the Ethernet module to the Ethernet network.
...... See Section 4.4, "Connecting to the Network."
Setting the Ethernet module network parameters .
...... See Section 4.5, "Settings from Ethernet Parameters."
Send the "Ping" command from an external device and confirm the completion of the Ethernet module initial processing
...... See Section 5.4, "Confirming the Completion of the Initial Processing."
Create a program for each communication
...... See applicable communication function
Operation

POINT

(1) When the parameter settings are added or modified using the FX Configurator-EN (GX Developer), the settings differs depending on the connection condition to the Ethernet module as shown below.

- When directly connected to the PLC
  After the parameter settings are added or modified, re-initialization processing can be automatically performed.
- When connected via Ethernet
  The added or modified parameter settings are not reflected.
  To operate the Ethernet module after the parameter settings are added or modified with FX Configurator-EN (GX Developer), first save the parameter values, second reboot the Ethernet's module power.

(2) Timing for powering ON
Power ON the PLC and the Ethernet module with the timing as shown below.



Power ON at the same time or the Ethernet first.

Power OFF at the same time or the Ethernet afterwards.

Important

(1) Do not write any data in the "System area" of the intelligent function module buffer memory.

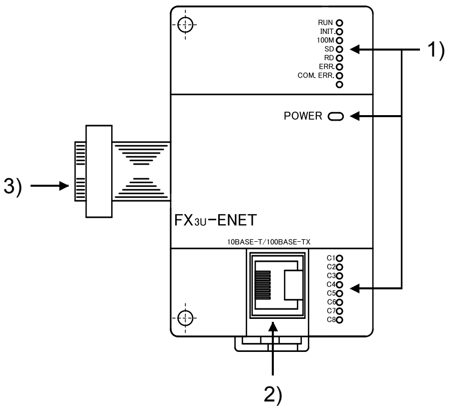
(2) When status control (such as remote RUN/STOP) from a external device is used for the PLC, the user should select "Always wait for OPEN" beforehand using the setting parameters. (Select using initial timing in the operation settings.)
If "Do not wait for OPEN" is selected, the communication line will be closed during remote STOP. After that, communication cannot be reopened from the PLC side and remote run from the external device will not work.

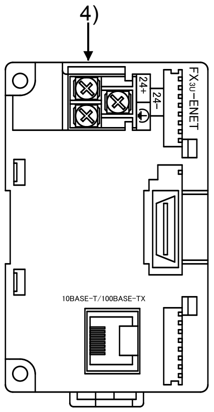
(3) When the Ethernet module is replaced, reset the parameters.

(4) When the Ethernet module is replaced, reboot the external device as well. (If the external device retains the Ethernet address, it may be impossible to continue communication because when a module is replaced the Ethernet address changes.)
In the same way, when the external device (personal computer, etc.)is replaced, reboot the Ethernet module.

4.3 Components of the Ethernet Module

This section shows the components of the Ethernet module.

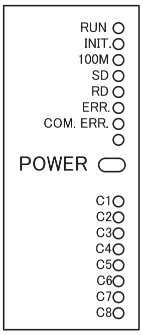




View with the cover removed

	Name	Description
1)	LED display	Refer to the contents of the LED displays (1).
2)	10BASE-T/100BASE-TX connector (RJ45)	Connector for connecting the Ethernet module to the 10BASE-T/100BASE-TX. (Ethernet module chooses between 10BASE-T and 100BASE-TX according to the hub.)
3)	Extension cable	Cables to connect to the PLC
4)	External 24VDC terminal	Power supply terminal for 24VDC

(1) LED display contents (✻1)



LED name	Display description	When the LED is on	When the LED is off
RUN	Normal operation display	Normal (when FROM/TO instruction can be executed from PLC side)	Abnormal (when WDT is in operation)
INIT.	Initial processing status display	Normal completion	Not processed (when execution failed initial processing)
100 M	Transmission speed display	100Mbps	10Mbps/When not connected
SD	Data sending display	Data being sent	Data not being sent
RD	Data receiving status display	Data being received	Data not being received
ERR.	Setting abnormal display	Abnormal ✻2	Normal setting
COM.ERR.	Communication abnormal display	Communication abnormal occurrence ✻3	Normal communication in progress
POWER	Module power status	Power is on	Power is off
C1 to C8	TCP/IP, UDP status of the connections	TCP/IP : Connection Established UDP : Open	TCP/IP : Connection not Established UDP : Closed

✻1 Refer to Section 11.1.1 for causes of error displays and the corresponding corrective actions.

✻2 The [ERR.] LED turns on in the following cases:

- When the parameter written is incorrect.
- When the checksum of the parameter written is not identical.
- When an error has occurred in the Ethernet module and operation is disabled due to the error.

✻3 Refer to Section 11.1.1 for when "COM.ERR." LED is on.

4.4 Connecting to the Network

The following explains how to connect the Ethernet module to the 100BASE-TX/ 10BASE-T networks.
Some precautions that should be observed while connecting the Ethernet module are also shown below. Pay close attention to safety and use the Ethernet module properly.

(1) Sufficient network knowledge and safety precautions are required when installing 100BASE-TX or 10BASE-T networks. Consult a specialist when connecting cable terminals or installing trunk line cables, etc.

(2) Use a connection cable conforming to the standards shown in Section 2.2.

⚠CAUTION

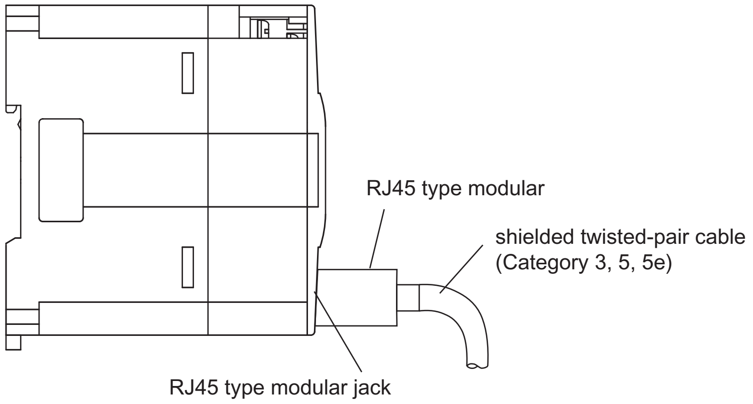
- Do not bundle the control wires and the communication cables with the main circuit and the power wires, and do not install them close to each other. They should be installed at least 100 mm (3.94 in.) away from each other. Failure to do so may generate noise that may cause malfunctions.
- Make sure to place the communication and power cables to be connected to the module in a duct or fasten them using a clamp.
  If the cables are not placed in a duct or fastened with a clamp, their positions may be unstable or moved, and they may be pulled inadvertently. This may damage the module and the cables or cause the module to malfunction because of faulty cable connections.
- When disconnecting the communication and power cables from the module, do not pull the cables by hand.
  When disconnecting a cable with a connector, hold the connector to the module by hand and pull it out to remove the cable.
  When disconnecting a cable without a connector, loosen the screws on the terminal block first before removing the cable.
  If a cable is pulled while connected to the module, it may cause the module to malfunction or damage the module and the cable.

POINT

Cautions regarding powering the hub, PLC and Ethernet module simultaneously.
On some hubs, for a fixed period of time immediately after powering up, even if packets are sent from the Ethernet device, there are cases when packets are not sent to the external device. For this case, create a sequence program that waits a sufficient amount of time after powering up before sending packets.

4.4.1 Connecting to the 10BASE-T/100BASE-TX network

This section explains how to connect the Ethernet module to the 10BASE-T, 100BASE-TX network.
The following shows the connection diagram for the twisted paid cable.



<Operating procedure>

(Step 1) Connect the twisted-pair cable to the hub.

(Step 2) Connect the twisted-pair cable to the Ethernet module.

POINT

(1) The Ethernet module detects whether it is 10BASE-T or 100BASE-TX, and in full-duplex or half-duplex transmission mode automatically according to the hub.
For connection to the hub without the auto detection function, set the half-duplex mode on the hub side.

(2) For 10BASE-T or 100BASE-TX connection required devices and a sample system configuration, refer to Section 2.2 (1) and (2).

4.5 Setting Ethernet Parameters

Setting the Ethernet module as a network module.
The settings found must always be set in order to use the Ethernet module.

(1) "Operational settings" (Details are explained in Section 4.6)

This screen is for setting common items for when other modules use the Ethernet module.
The settings on this screen must always be set, since they are required for the Ethernet initial processing.

(2) "Initial settings" (Details are explained in Section 5.2)

This screen is for setting common timer values for TCP/IP communication to be used in the Ethernet module as well as for setting the DNS server in order to use the e-mail function.
It is not necessary to set the timer values when communicating using the default timer values.

(3) "Open settings" (Details are explained in Section 5.5)

This screen is for setting connection open processing, buffer memory for fixed buffer communication and MC protocol to communicate data with an external device.

(4) "Router relay parameter (Routing information)" (Details are explained in Section 5.3.)

The following setting for data communication with external devices are set on this screen:

- Communicating with external devices connected to other Ethernet networks via a router

(5) "E-mail settings" (Details are explained in Section 10.6)

These settings are for using the e-mail transmission/reception and automatic news (notification) functions.

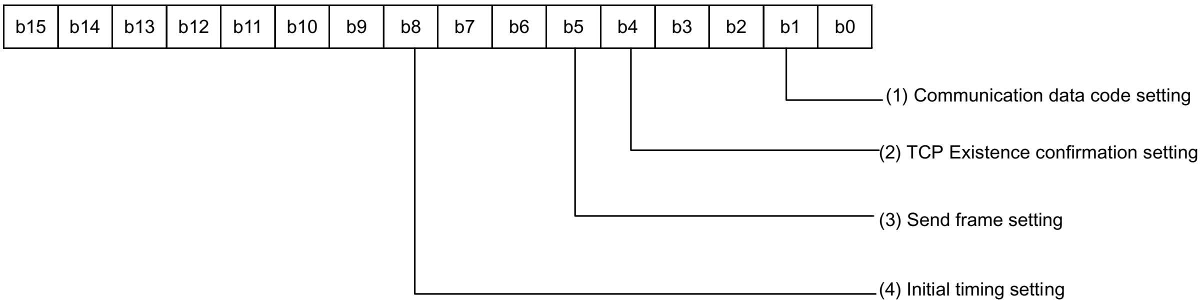
Ethernet parameter settings are written to buffer memory with a ladder program, and the contents are stored in the Ethernet module’s flash ROM.
Also, the contents can be set with FX Configurator-EN (GX Developer).

4.6 Operational Settings

This section explains how to set the operations parameters.

Setting name	BFM number [Decimal] (bit number)	Setting description	Setting range/selection
Communication data code	24 (b1)	Select the communication data code.	OFF : Binary code ON : ASCII code
TCP Existence confirmation setting	24 (b4)	Select the existence check method for TCP communication	ON : Use KeepAlive OFF : Use Ping
Send frame setting	24 (b5)	Select the frame format to send	OFF : Ethernet (V2.0) ON : IEEE 802.3
Initial Timing	24 (b8)	Open setting	OFF : Do not wait for OPEN ON : Always wait for OPEN
IP Address	0 to 1	IP address of the local station.	—

[Initial processing parameter setting area (For re-initialization)]


b15
b14
b13
b12
b11
b10
b9
b8
b7
b6
b5
b4
b3
b2
b1
b0
(1) Communication data code setting
(2) TCP Existence confirmation setting
(3) Send frame setting
(4) Initial timing setting

(1) Communication data code (BFM#24 b1)

(a) Select the format of the communication data when communicating with an external device.

Name of setting	Description of setting
Binary code	Communicate using binary data.
ASCII code	Communicate using ASCII data.

(b) For more details on the data communication codes, see Section 3.2, "Data Codes for Communication."

(2) TCP Existence confirmation setting (BFM #24 b4)

Select the existence check method for TCP communication.
For the existence check function, refer to Section 5.2.2.

Name of setting	Description of setting
Use the KeepAlive	Checks connection status with KeepAlive.
Use the Ping	Checks connection status with Ping.

POINT

(1) "Operational settings" parameters must always be set.
If the settings are changed, the power to the PLC and the Ethernet module must be turned off once and then turned on again.

(2) If re-initialization of the Ethernet module is required due to the occurrence of an error, perform re-initialization using a sequence program.

NOTE

When adjusting settings with FX Configurator-EN (GX Developer), select “Operating Settings” at the initial screen and adjust the settings at the “Ethernet Operating Settings” screen.

(3) Send frame setting (BFM #24 b5)

(a) Select the frame of the Ethernet header for the data link layer to be sent by the Ethernet module.

Setting item	Description of setting
Ethernet (V2.0)	Transmits using an Ethernet frame.
IEEE802.3	Transmits using an IEEE802.3 frame.

(b) When receiving data from the external device, reception should occur regardless of whether the Ethernet frame or IEEE802.3 frame is used

NOTE

(1) Transmission using Ethernet frames is generally recommended.

(2) When communication with the external device fails, check whether or not communication is possible using a PING command.

(4) Initial Timing (BFM#24 b8)

(a) Select the timing to open connections for which TCP-Passive open or UDP open are selected with the "Open settings" parameter (∗1).

∗1 For more details on the open settings, see Section 5.5 "Open Settings".

Name of setting	Description of setting
Do not wait for OPEN (Communication impossible after STOP)	• Execute open/close processing using a sequence program. • Communication cannot be performed while the PLC is in the STOP status.
Always wait for OPEN (Communication possible after STOP)	• Passive open and UDP open connections always wait for open according to the parameter settings (a sequence program for open/close processing is not required) (∗2). • Communication can be performed while the PLC is in the STOP status.

∗2 If the sequence program of the local station's PLC executes the close processing, the station is not placed in the OPEN request wait status after the connection is shut off.

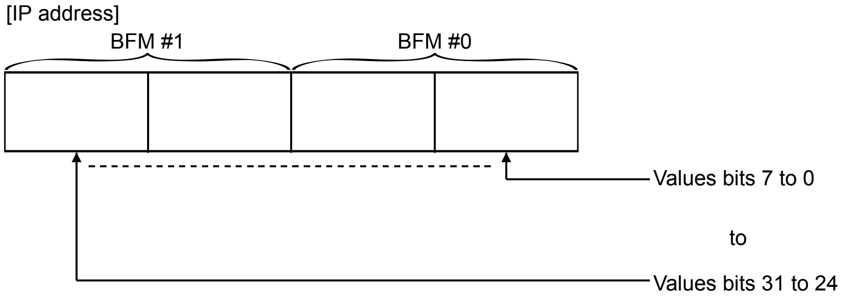
(b) In the following cases, open/close processing is required with instructions written to the buffer memory (BFM #1602 to 1609).

- When "Do not wait for OPEN" is selected in the initial timing settings.
- If the "Open settings" are not performed for a connection.
- If "TCP-Active" is selected in the "Open settings" for a connection.

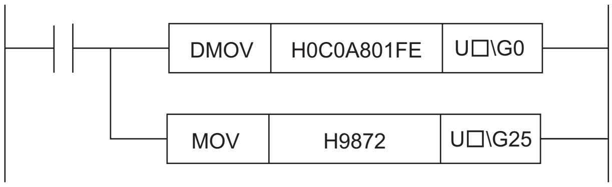
For detail on the open/close processing, see Section 5.6 "Open Processing/Close Processing of the Connection".

(5) IP Address settings – IP address (BFM #0 to 1)

(a) Set the IP address of the local station according to the specified input format (decimal or hexadecimal).
It should be set so that the local station Ethernet module and the communicating external device have the same class and sub-net address (two words).



(Example) Program for "192.168.1.254 (C0 A8 01 FEH)"



```
DMOV  H0C0A801FE  U□\G0
MOV   H9872       U□\G25
```

IP adress "192.168.1.254 (C0 A8 01 FEH)" is stored in BFM #0, #1

Write command for configuration data to a flash ROM.

＊ □ corresponds to the module number.

(b) It is necessary to use the router relay function in order to communicate with an external device on another Ethernet network (different sub-net address).
For more details, see Section 5.3, "Router Relay Parameter".

(c) Set the IP address after consulting a network administrator (the person who plans the network and manages IP addresses).

4.7 Self-Diagnostic Tests

This section explains the self-diagnostic tests for checking the hardware and transmission and reception function of the Ethernet module.

4.7.1 Self loopback test

The following explains the self loopback test that is used to check the hardware including the Ethernet module’s transmission and reception circuit.
The self loopback test transmits a test message within the Ethernet module to check if the same message can be received or not. (Connection to the hub is not necessary.)
The following explains the procedure for performing the self loopback test. The test takes approximately five seconds to complete.
The test result can be determined from the LED displays on the front of the Ethernet module.

Step	Description of operation		Status of LED		
			[RUN]	[C1]	[ERR.]
1	Perform a self loopback test in operation mode (“0002H” is written to BFM #202).		—	—	—
2	The contents of BFM#202 are written to the flash ROM ("9872H” is written to BFM#25).		—	—	—
3	• Turn off the PLC. • Set the RUN/STOP switch to the STOP side • Reboot the PLC’s and the Ethernet module’s power.		—	—	—
4	The test is in execution.		●	●	○
5	The test results are shown on the LED after approximately 5 seconds.	Normal	●	○	○
		Error	●	○	●
6	Turn off the PLC.		—	—	—

●: Lit ○: Off

The following are probable causes of errors.
- Ethernet module hardware error
- Ethernet line error

An error code is stored in the error log area (BFM#229) in the buffer memory of the Ethernet module; the error content can then be checked from FX Configurator-EN. (See Sections 11.2 and 11.3.)

4.7.2 Hardware test (H/W Test)

This section explains the RAM and ROM tests for the Ethernet module. The procedure for the hardware test is as shown in the table below.
The test results are judged from the LED displays on the front of the Ethernet module.

Step	Description of operation		Status of LED		
			[RUN]	[C1]	[ERR.]
1	Perform a hardware test in operation mode. (“0003H” is written to BFM #202)		–	–	–
2	The contents of BFM#202 are written to the flash ROM ("9872H” is written to BFM#25)		–	–	–
3	• Turn off the PLC. • Set the RUN/STOP switch to the STOP side. • Reboot the PLC’s and the Ethernet module’s power.		–	–	–
4	The test is in execution.		●	●	○
5	The test results are shown on the LED after approximately 5 seconds.	Normal	●	○	○
		Error	●	○	●
6	Turn off the PLC and the Ethernet module.		–	–	–

●: Lit ○: Off

The following are probable causes of errors.

- Ethernet module RAM/ROM error

An error code is stored in the error log area (BFM #229) in the buffer memory of the Ethernet module; the error content can then be checked from GX Developer. (See Sections 11.2 and 11.3.)

4.8 Maintenance and Inspection

This section explains the maintenance and inspection as well as the installing and uninstalling of the Ethernet module.

4.8.1 Maintenance and inspection

The Ethernet module does not need to be inspected for anything particular other than checking whether or not the connections of cables and the terminal screws are loose. Maintain and inspect the system according to the same inspection items as described in the user's manual for the PLC in order to use it in optimal operating conditions.

◇DANGER

- Do not touch the terminals and connectors while the power is on.
  Doing so may cause in electric shocks and malfunctions.
- Make sure to turn off all phases of the external power supply before cleaning or re-tightening the screws. Failure to do so may cause damages or malfunctions of the module.
  If the screws are loose, it may cause the module to short-circuit, malfunction or fall off.
  Tightening the screws excessively may damage the screws and/or the module and cause the module to short circuit, malfunction or fall off.

⚠CAUTION

- Be careful not to let any foreign matter such as wire chips get inside the module.
  They may cause fire, as well as breakdowns and malfunctions of the module.
- Never disassemble or modify the module.
  This may cause breakdowns, malfunctions, injuries or fire.

4.8.2 Installing and uninstalling the module

Before installing or uninstalling the Ethernet module, make sure to read Section 4.1, "Handling Precautions" thoroughly, ensure the safety of the installation, and handle the module properly according to the instructions.
The following explains the procedures when installing/uninstalling the Ethernet module.

<Operation procedure when replacing the Ethernet module> (*1)

(Step 1) Use FX Configurator-EN (GX Developer) to write and save the parameters from the Ethernet module. (*2)
(Step 2) Turn off the power supply to the base module, the extension device, and the Ethernet module.
(Step 3) Remove the network cable and Ethernet module.
(Step 4) Set up and start up the new Ethernet module according to Section 4.2, "Settings and Procedures Prior to Starting the Operation." (*3)
(Step 5) Reset the external device. (*4)

<Operation procedure when replacing the FX base module>

(Step 1) Turn off the power supply to the FX base module and the Ethernet module.
(Step 2) Replace the FX base module. (See the Manual of FX PLC.)
(Step 3) Turn off the power.

*1 It is recommended to record and save parameters not only when the Ethernet module is replaced but also when parameters for the Ethernet module are created or modified.

*2 When FX Configurator-EN (GX Developer) is not installed :

1) Monitor the BFM in Ethernet module by GX Developer, and record the setting value.
2) After replacing Ethernet module, write the setting value into BFM by GX Developer.
3) Write the setting value into the Flash ROM by BFM#29.

*3 When the Ethernet module is replaced, use the FX Configurator-EN (GX Developer) to reset the parameters.`

*4 When the Ethernet module is replaced, reset the external device as well. (If the external device retains the Ethernet address, it may be impossible to continue communication because when a module is replaced the Ethernet address changes.)
In the same way, when the external device (personal computer, etc.)is replaced, restart the Ethernet module.

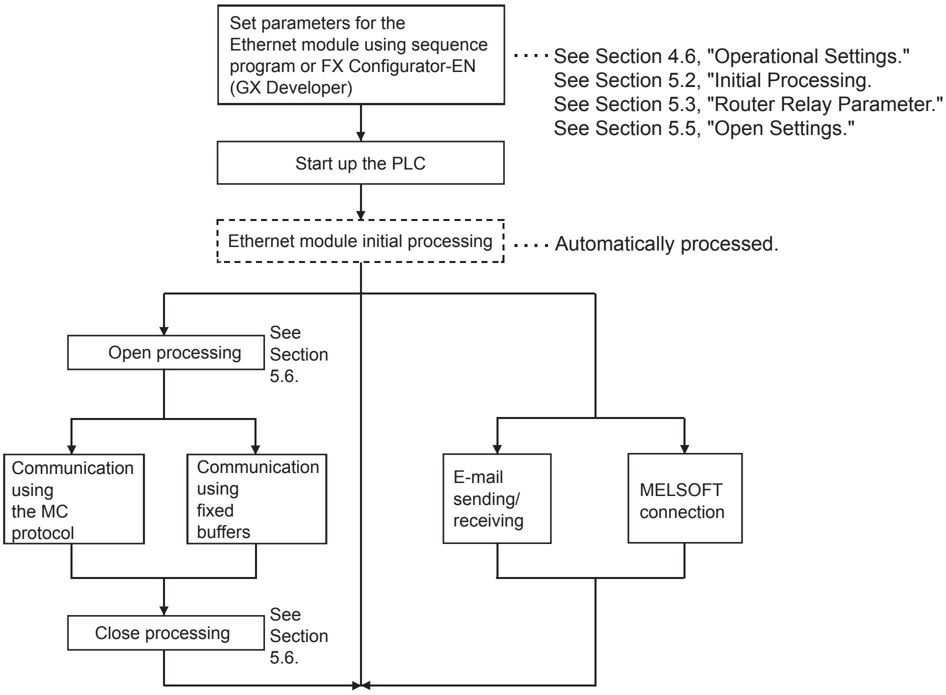
5 COMMUNICATION PROCEDURE

This chapter gives an overview of the communication procedure using the Ethernet module, as well as the required initial processing of the Ethernet prior to data communication and the open processing for communication with an external device.
The sequence program can be effectively simplified by setting the parameters of the Ethernet module supported by FX Configurator-EN (GX Developer).
The following explains the communication procedure using FX Configurator-EN (GX Developer).

5.1 Overview of the Communication Procedure

This section gives an overview of the procedure for performing data communication with external devices via the Ethernet module.

- Start data communication by establishing connections with external devices via the initial or open processing.
- End data communication by closing the connection, and, as a result, terminating all communication processing.
  The following diagram illustrates the communication procedure:
  (There is no connection since UDP/IP does not have a notion of connection.)

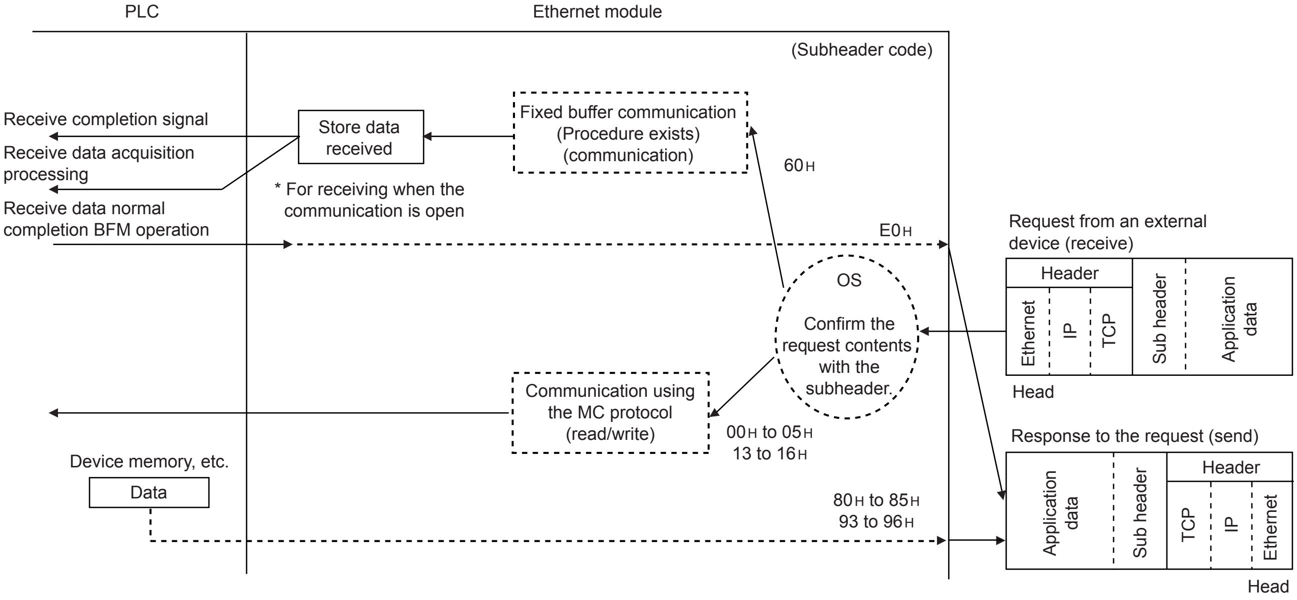

Set parameters for the Ethernet module using sequence program or FX Configurator-EN (GX Developer)
See Section 4.6, "Operational Settings."
See Section 5.2, "Initial Processing.
See Section 5.3, "Router Relay Parameter."
See Section 5.5, "Open Settings."
Start up the PLC
Ethernet module initial processing
Automatically processed.
Open processing
See Section 5.6.
Communication using the MC protocol
Communication using fixed buffers
E-mail sending/ receiving
MELSOFT connection
Close processing
See Section 5.6.

NOTE

Each of the following two types of communication can be performed with an external device opened by the user.

- Communication using MC protocol
- Sending/receiving in fixed buffer communication (procedure exists)

When receiving communication request data from an external device


PLC
Ethernet module
(Subheader code)
Receive completion signal
Receive data acquisition processing
Receive data normal completion BFM operation
Store data received
* For receiving when the communication is open
Fixed buffer communication (Procedure exists) (communication)
60H
E0H
OS
Confirm the request contents with the subheader.
Communication using the MC protocol (read/write)
00H to 05H
13 to 16H
80H to 85H
93 to 96H
Device memory, etc.
Data
Request from an external device (receive)
Header
Ethernet
IP
TCP
Sub header
Application data
Head
Response to the request (send)
Application data
Sub header
Header
TCP
IP
Ethernet
Head

5.2 Initial Processing

This section explains the initial processing of the Ethernet module.

5.2.1 Initial processing

The initial processing enables data communication with an external device by setting the parameters required for data communication via the Ethernet module.
Set the following parameters in the sequence program, write them to the Ethernet module; the initial processing of the Ethernet module is then performed by one of the following methods.

- Select re-initialization
- Re- initialization request (write “0001H” to BFM #1600)
- Power ON the Ethernet module again.

(1) Parameters required for initial processing

- "Operational settings" : (See Section 4.6.)
- "Initial settings" : (See Section 5.2.2.)

(2) Confirmation of initial processing result

Initial processing	Ethernet module			
	INIT.LED	Initial normal completion signal (BFM#28 b0)	Initial abnormal completion signal (BFM#28 b5)	Initial normal completion signal (BFM#27)
At normal completion	●: On	ON	OFF	0001H
At abnormal completion	○: Off	OFF	ON	0000H

If the initial processing does not complete normally, correct the above parameter setting value and write to the Ethernet module. Then re-initialize the Ethernet module.

5.2.2 Initial settings

This section explains the initial settings.

Setting item name		BFM number (Decimal)	Description of setting	Setting range/options
Timer setting	TCP ULP timer	4	Set the time of packet existence at TCP data transmission.	2 to 32767
	TCP zero window timer	5	Set the interval for checking the reception enabled status.	2 to 32767
	TCP resend timer	6	Set the time to resend at TCP data transmission.	2 to 32767
	TCP end timer	7	Set the confirmation wait time at TCP close processing.	2 to 32767
	IP assembly timer	8	Set the wait time for division data packets.	1 to 32766
	Response monitoring timer	9	Set the response wait time.	2 to 32767
	Destination existence confirmation starting interval	10	Set the time to start confirming existence of an external device after communication with it has terminated.	1 to 32767
	Destination existence confirmation interval timer	11	Set the time interval between reconfirming existence.	1 to 32767
	Destination existence confirmation resend timer	12	Set the number of times to reconfirm existence when a response to the existence confirmation is not received.	1 to 32767
DNS setting (＊1)	IP address of DNS server 1	20505 to 20506	Set IP address of DNS server 1.	—
	IP address of DNS server 2	20507 to 20508	Set IP address of DNS server 2.	—
	IP address of DNS server 3	20509 to 20510	Set IP address of DNS server 3.	—
	IP address of DNS server 4	20511 to 20512	Set IP address of DNS server 4.	—

＊1 Details are explained in Section 10.6.

(1) Timer setting – TCP ULP timer (BFM#4)

(a) This item sets the time of packet existence during TCP data sending. This timer is passed through the parameter when TCP opens or data is sent.

(b) Set the value in the range from 2 to 32767.

(c) Timer setting = setting value × 500 ms

(2) Timer setting - TCP zero window timer (BFM#5)

(a) The window indicates the reception buffer on the receiving side.

(b) When there is no more space in the receiving buffer (window size = 0) on the receiving side, data communication has to wait until enough space is made.
When this occurs, the sending side sends a sending window confirmation packet to the receiving side after the TCP zero window timer value has been reached, and confirms the reception enabled status.

(c) Set the value in the range from 2 to 32767.

(d) Timer setting = setting value × 500 ms

(3) Timer setting – TCP resend timer (BFM#6)

(a) Set the resend time if ACK is not returned during TCP opening or data transmission. This timer is also used for the existence time of the ARP function.
(ARP is resent in "TCP resend timer" value/2 if a response is not returned after the sent ARP request.)
It also serves as the minimum setting time for the data link instruction arrival monitoring time.

(b) Set the value in the range from 2 to 32767.

(c) Timer setting = setting value × 500 ms

(4) Timer setting – TCP end timer (BFM#7)

(a) When the TCP connection is closed from the local station, this timer sets the monitoring time for how long the local station waits for a FIN request from an external device after it sends a FIN request and the external device returns an ACK.

(b) If the FIN request cannot be received from the external device before the time designated by the TCP end timer setting, a RST should be sent to the external device to forcibly close the connection.

(c) Set the value in the range from 2 to 32767.

(d) Timer setting = setting value × 500 ms

(5) Timer setting - IP assembly timer (BFM#8)

(a) Communication data may be divided on the IP level due to the buffer restriction at the sending or receiving station.

(b) Set the value in the range from 1 to 32766.

(c) Timer setting = setting value × 500 ms

(6) Timer setting - Response monitoring timer (BFM#9)

(a) This timer setting sets the following times.

1) The time to wait for a response after sending a command.

2) The time to wait for the last message after receiving the first message when the message is divided.

(b) Set the value in the range from 2 to 32767.

(c) Timer setting = setting value × 500 ms

(7) Timer setting - Destination existence confirmation starting interval (BFM#10)

(a) This timer sets the time interval before attempting to confirm the existence of an external side when an open connection for which existence confirmation is required does not respond.

(b) Set the value in the range from 1 to 32767.

(c) Timer setting = setting value × 500 ms

(8) Timer setting - Destination existence confirmation interval timer (BFM#11)

(a) This timer sets the time interval before reconfirming the existence of an external device on an open connection for which existence confirmation is required that does not respond.

(b) Set the value in the range from 1 to 32767.

(c) Timer setting = setting value × 500 ms

(9) Timer setting - Destination existence confirmation resend timer (BFM#12)

(a) This timer sets the number of times to reconfirm existence when there is no response from an external device on an open connection for which existence confirmation is required.

(b) Set the value in the range from 1 to 32767.

(10) DNS setting - Input format (∗1)

(11) DNS setting - IP address of DNS server n (∗1)

∗1 The DNS setting is set when the e-mail sending/receiving function is used. Refer to Chapter 10, "E-mail Function".

POINT

To adjust settings with FX Configurator-EN (GX Developer), select “Initial Settings” from the initial screen, and make the settings at the “Ethernet Initial Settings” screen.

NOTE

(1) Set the value of each timer on the Ethernet module side such that the following relations are met.

- $\left[\begin{array}{l}\text{Response monitoring}\\ \text{timer value}\end{array}\right] \geqq \left[\begin{array}{l}\text{TCP ULP}\\ \text{timer value}\end{array}\right] \geqq \left[\begin{array}{l}\text{TCP end}\\ \text{timer value}\end{array}\right] \geqq \left[\begin{array}{l}\text{TCP resend}\\ \text{timer value}\end{array}\right] > \left[\begin{array}{l}\text{IP assembly}\\ \text{timer value}\end{array}\right]$
- $\left[\begin{array}{l}\text{TCP resend}\\ \text{timer value}\end{array}\right] = \left[\begin{array}{l}\text{TCP zero window}\\ \text{timer value}\end{array}\right]$

Furthermore, when connecting a line using Mitsubishi products, you should make sure that both nodes have the same settings.

(2) Set the value of each timer on the external device side such way that the following relations are met.
Communication errors such as transmission timeouts may occur more frequently if the timer values are not set so that they satisfy the following relationships.

- $\left[\begin{array}{l}\text{TCP resend timer value on}\\ \text{the external device side}\end{array}\right] > \left[\begin{array}{l}\text{TCP resend timer value on}\\ \text{the external device side}\end{array}\right]$
- $\left[\begin{array}{l}\text{Monitoring timer value for the}\\ \text{external device application software}\end{array}\right] > \left\{\left[\begin{array}{l}\text{TCP ULP timer value on}\\ \text{the Ethernet module side}\end{array}\right] \times n^{*1}\right\}$

*1 n is the number of TCP segment transmissions and can be obtained via the following calculation:

$$n = \left[\frac{\text{Size of the message transmitted by the Ethernet module}}{\text{Maximum segment size}}\right] \text{(fractions below decimal point are rounded up)}$$

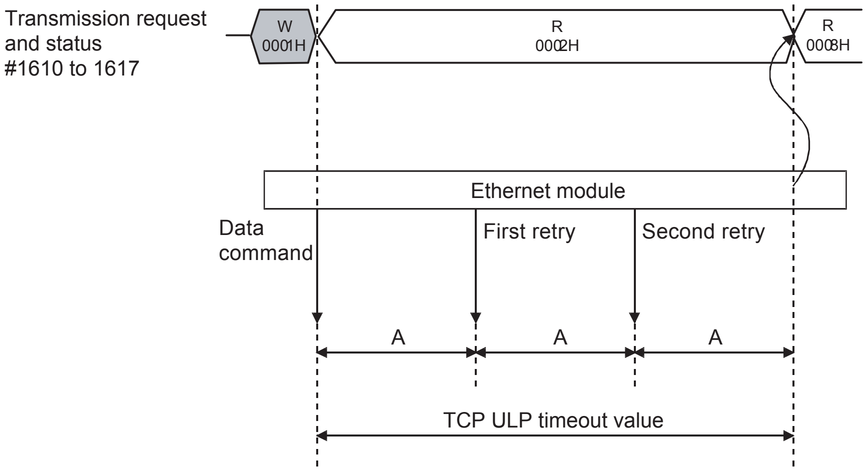
Example1: Number of TCP segment transmissions when communicating via the same line
The maximum segment size is 1460 bytes via the same line (without going through a router) and the number of TCP segment transmissions is as follows:
n = 1, if the size of the message transmitted by the Ethernet module is 1460 bytes or less.
n = 2, if the size of the message transmitted by the Ethernet module is greater than 1460 bytes.

Example2: Number of TCP segment transmissions when communicating via separate lines
The maximum segment size is at least 536 bytes on a separate line (via dialup router, etc.) and the number of TCP segment transmissions is as follows:
n = 1, if the size of the message transmitted by the Ethernet module is 536 bytes or less.
n = 2, if the size of the message transmitted by the Ethernet module is greater than 536 bytes and no more than 1072 bytes.
n = 3, if the size of the message transmitted by the Ethernet module is greater than 1072 bytes and no more than 1608 bytes.

(3) When communication errors occur, the setting value should be changed to increase the number of retries.
The number of retries is obtained by using the following equation:
(In case of the default values, 2 = (60/20) - 1)

- $\left[\begin{array}{c}\text{Number}\\ \text{of retries}\end{array}\right] = \left\{\left[\dfrac{\text{TCP ULP timer value}}{\text{TCP resend timer value}}\right] -1\right\}$

Example: Assuming the values are set in such a way that the number of retries is two, a data transmission error will occur at the timing shown in the figure below if data transmission fails (when communicating using fixed buffer).



A : TCP resend timer value
(The time after which the data should be retransmitted when an "ACK" is not returned after sending data.)

(4) Perform the following setting in order to eliminate the retries explained in (3) (i.e., to set the number of retries to 0).

- $\left[\begin{array}{c}\text{TCP ULP}\\ \text{timer value}\end{array}\right] = \left[\begin{array}{c}\text{TCP end}\\ \text{timer value}\end{array}\right] = \left[\begin{array}{c}\text{TCP resend}\\ \text{timer value}\end{array}\right]$

(Each timer value should be identical.)

(5) The target existence check is a function whereby the Ethernet module checks whether or not a remote device is functioning normally by sending an existence check message and then waiting to see whether a response message is received. It is used if a connection to a remote device is open but communication with the remote device has not been performed for a certain period of time.

(a) The existence check function has two methods of checking: PING and KeepAlive.
The Ethernet module performs each of the existence checks based on the setting values explained in (7) to (9) of this section and the existence check setting of the open settings (refer to Section 5.5 (6)).
The existence check function (Ping or KeepAlive) can be selected at the time of operation setting or re-initialization.
For the operation setting, refer to Section 4.6.
Refer to Section 5.2.3 for the explanation on re-initialization.

1) Checking by KeepAlive

This method is used for a connection opened via the TCP/IP protocol. The Ethernet module performs an existence check by sending an existence check ACK message to a remote device with which communication has not been performed for a certain period of time and waiting to see whether or not a response is received. (∗1)

∗1 The connection is cut off if the remote device does not support the TCP KeepAlive function (response to KeepAlive ACK messages).

2) Checking by PING

This method is used for a connection opened via the TCP/IP or UDP/IP protocol.

The Ethernet module performs an existence check by sending a PING command (using the ICMP echo request/response function) to a remote device with which communication has not been performed for a certain period of time and waiting to see whether or not a response is received. (∗2)

∗2 Note that the Ethernet module automatically returns an echo response packet when it receives a PING echo request command. (It sends a response to the received PING command even if the connection used in the data communication with the remote device is closed.)

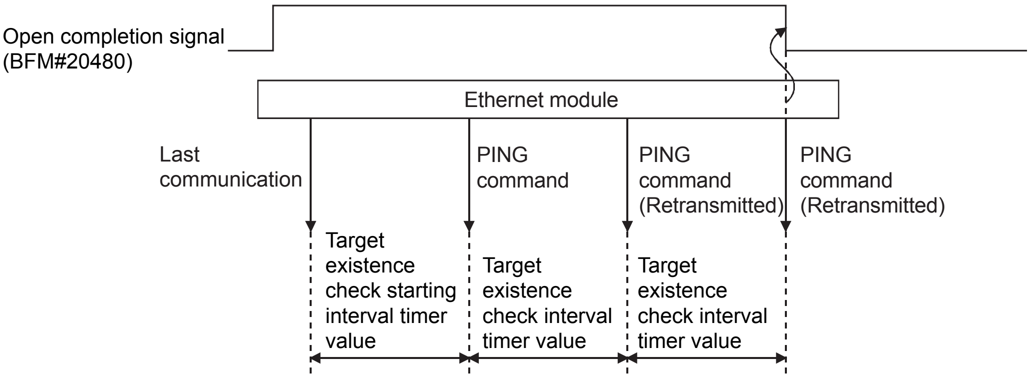
(b) The following actions are taken if a response message cannot be received (an error is detected) from the remote device.

- The corresponding connection will be forcibly closed (the line is disconnected). (∗3)

  ∗3 It is possible to reopen it with a user program.
- The open completion signal (the corresponding bit of BFM#20480) is turned off and, the error code (C035H) is stored in places such as the open error code storage area.

  Also, the open/close status (BFM #1602 to 1609) value becomes "0000H".

Example: Assuming the values are set under the condition that the number of retries is three, the Ethernet module performs target existence check at the timing shown in the figure below. (An example of existence check by PING)


Open completion signal
(BFM#20480)
Ethernet module
Last communication
PING command
PING command (Retransmitted)
PING command (Retransmitted)
Target existence check starting interval timer value
Target existence check interval timer value
Target existence check interval timer value

5.2.3 Re-initialization

Re-initialization is performed in order to place the Ethernet module into its startup status without actually restarting the PLC.
Re-initialization of the Ethernet module can be performed in a sequence program.
The purposes of and how to program the re-initialization of the Ethernet module are explained below.

NOTE

When FX Configurator-EN is being used, re-initialization can be performed after downloading the parameters.
However, if it is connected to the Ethernet module via Ethernet, re-initialization cannot be performed.
Re-initialize from the sequence program.

(1) Purposes of performing re-initialization

(a) To update address information of an external device maintained by the Ethernet module
The Ethernet module maintains the IP address of the external device with which it has been communicating and the corresponding Ethernet address (MAC address). This is done in order to prevent other devices from accessing the PLC illegally using the IP address of an external device with which communication was performed normally. (＊1)
For this reason it is necessary to re-initialize in order to clear the address information of the external device maintained by the Ethernet module in case a module or board on the external device side has been replaced due to failure.
＊1 Ethernet address is unique to a device. No devices share the same Ethernet address.

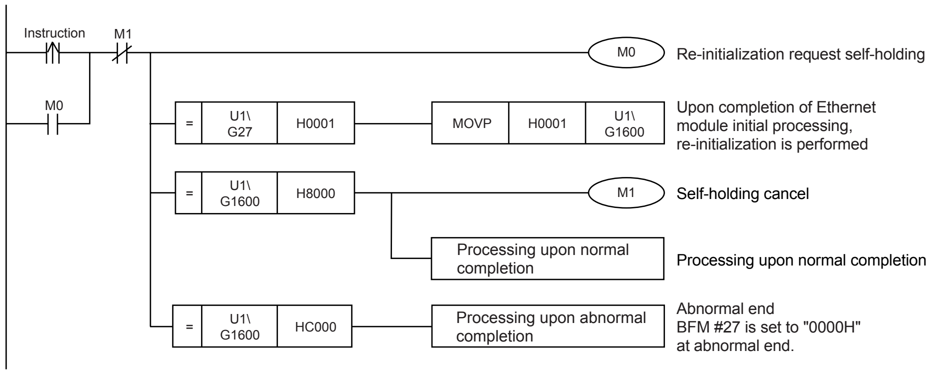
(b) To change the IP address of the Ethernet module of the local station
If the system is changed, it is possible to restart communication with external devices by changing the IP address (for the Ethernet module of the local station) in the operation settings (see Section 4.6).

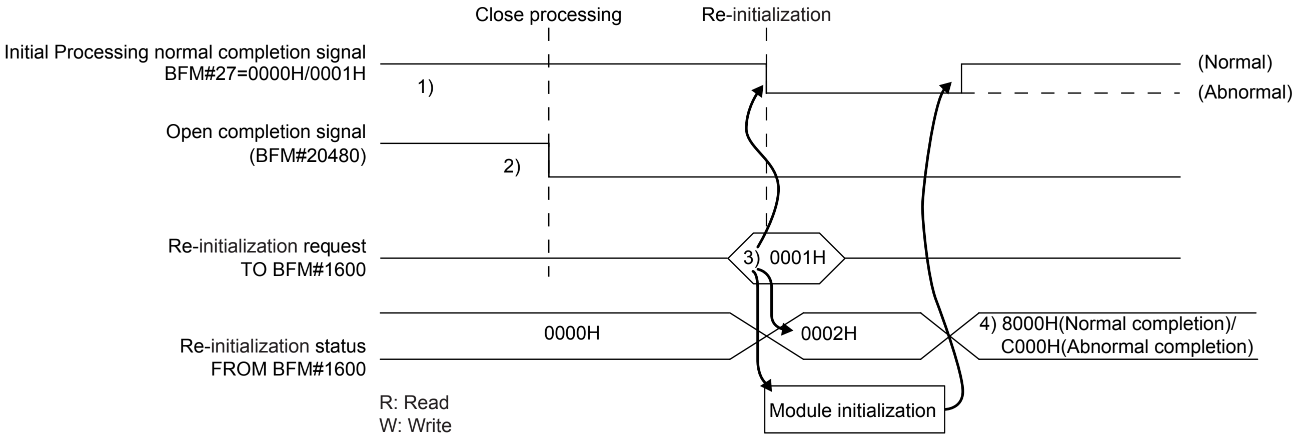
(c) To change the communication condition setting values
It is possible to restart communication with external devices by changing the communication conditions in the operation settings (see Section 4.6).

(2) Programming and timing for re-initialization

The programming and timing for re-initialization is shown below:

<<Re-initialization>>


Instruction
M1
M0
Re-initialization request self-holding
M0
= U1\ G27 H0001
MOVP H0001 U1\ G1600
Upon completion of Ethernet module initial processing, re-initialization is performed
= U1\ G1600 H8000
M1
Self-holding cancel
Processing upon normal completion
Processing upon normal completion
= U1\ G1600 HC000
Processing upon abnormal completion
Abnormal end
BFM #27 is set to "0000H" at abnormal end.


Close processing
Re-initialization
Initial Processing normal completion signal BFM#27=0000H/0001H
1)
(Normal)
(Abnormal)
Open completion signal (BFM#20480)
2)
Re-initialization request TO BFM#1600
3) 0001H
Re-initialization status FROM BFM#1600
0000H
0002H
4) 8000H(Normal completion)/ C000H(Abnormal completion)
R: Read
W: Write
Module initialization

1) It is confirmed whether the initial processing has been completed normally.

2) All data communication currently being performed with external devices is terminated, and close processing will be performed on all connections.

3) "0001H ", "0002H ", "0005H " or "0006H " is written to BFM#1600.
(Initial normal completion signal: BFM#27 "0001H ")
(Open completion signal: BFM#20480 "All OFF(0H)")

4) When re-initialization is completed, the value in BFM#1600 will be 8000H (normal completion) or C000H (abnormal completion).
＊ If the re-initialization is completed abnormally, the error code is stored in the following area.
Initial error code storage area (BFM#105)

(3) Operation for re-initialization

Operation for re-initialization differs as follows, depending on the value written to BFM#1600.

Written value	Operation	
	COM. ERR LED	Re-reading flash ROM[*1]
0001H	Turns off	Not re-read
0002H		Re-read
0005H	No change	Not re-read
0006H		Re-read

*1: When re-reading the flash ROM, the re-initialization is performed in the same status as when the power supply is turned on.

POINT

Please keep the following points in mind when re-initializing the Ethernet module. (Failure to do so may cause errors in the data communication with the external devices.)

(1) Be sure to end all current data communication with external devices and close all connections before re-initializing.

(2) Be sure to reset external devices if the IP address of the Ethernet module has been changed. (If an external device maintains the Ethernet address of a device with which it communicates, the communication may not work after the IP address of the Ethernet module has been changed.)

NOTE

It is possible to change the operational settings when re-initializing by a sequence program.

Parameter setting item		BFM number (decimal)	Parameter changes	Reference section
Operational settings		24	○	Section 4.6
Initial settings	Timer setting	4 to 12	○	Section 5.2
	DNS settings	—	×	Section 5.2
Open settings		32 to 39	○	Section 5.5
Router relay parameter settings		512 to 515	○	Section 5.3
E-mail settings		9856 to 10306	×	Chapter 10
	Send mail address setting	9987 to 10306	×	

○△ : Valid, × : Invalid

5.3 Router Relay Parameter

This section explains the router relay parameter.

Setting item name	BFM number (Decimal)	Description of setting	Setting range/options
Router relay function	2 (b4, b5)	Select whether the router relay function is not used or is used.	00 : Use 01 : Not used
Sub-net mask pattern	512, 513	Set the sub-net mask.	$C0000000_H$ to $FFFFFFFC_H$
Router IP address	514, 515	Set the IP address of the router to be routed through.	Other than 00000000_H and $FFFFFFFF_H$

NOTE

When making settings with the FX Configurator-EN (GX Developer), select [Router Relay Parameters] from the initial screen and adjust the settings at the [Ethernet Router Relay Parameter Settings] screen.

(1) Router relay function (BFM#2 b4,b5)

(a) Set whether the router relay function will be used or not.
The router relay function is not needed when the FX_{3U}-ENET communicates with the target device on the same Ethernet network (the subnet address of the IP address is the same).

(b) The router relay function allows communication with devices on other Ethernet networks via routers and gateways.
(The router relay function does not mean a function with which the FX_{3U}-ENET acts as a router.)

(c) One router can be set for the router relay function.

(2) Subnet mask pattern (BFM#512, 513)

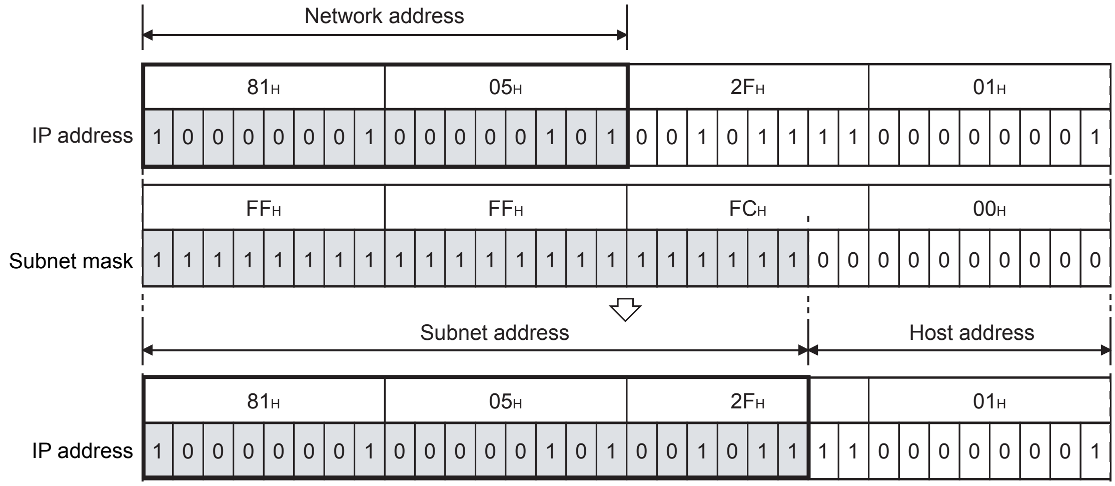
(a) Set the subnet mask. *1 (Setting range: C0000000H to FFFFFFF)
Consult the network administrator for the setting.

(b) When not using the subnet mask, set any of the following table values according to the class.

Class	Mask value
Class A	FF000000H
Class B	FFFF0000H
Class C	FFFFFF00H

*1 Ethernet networks include small-scaled network systems where multiple devices are connected to one Ethernet hub along with medium and large-scaled network systems where multiple small-scaled networks are connected by routers, etc.
The subnet mask logically divides one network, where many devices are connected, into multiple sub-networks to facilitate administration.

(Example) Class B



POINT

(1) All devices on the same sub-network must have a common subnet mask.
(2) When not administrated by the sub-network, the connected devices need not have a subnet mask. (Set the network address of the corresponding class.)

(3) Router IP address (BFM#514, 515)

Set the IP address of the router to be used when FX_{3U}-ENET communicates with the target device on another Ethernet network (refer to (4) below).

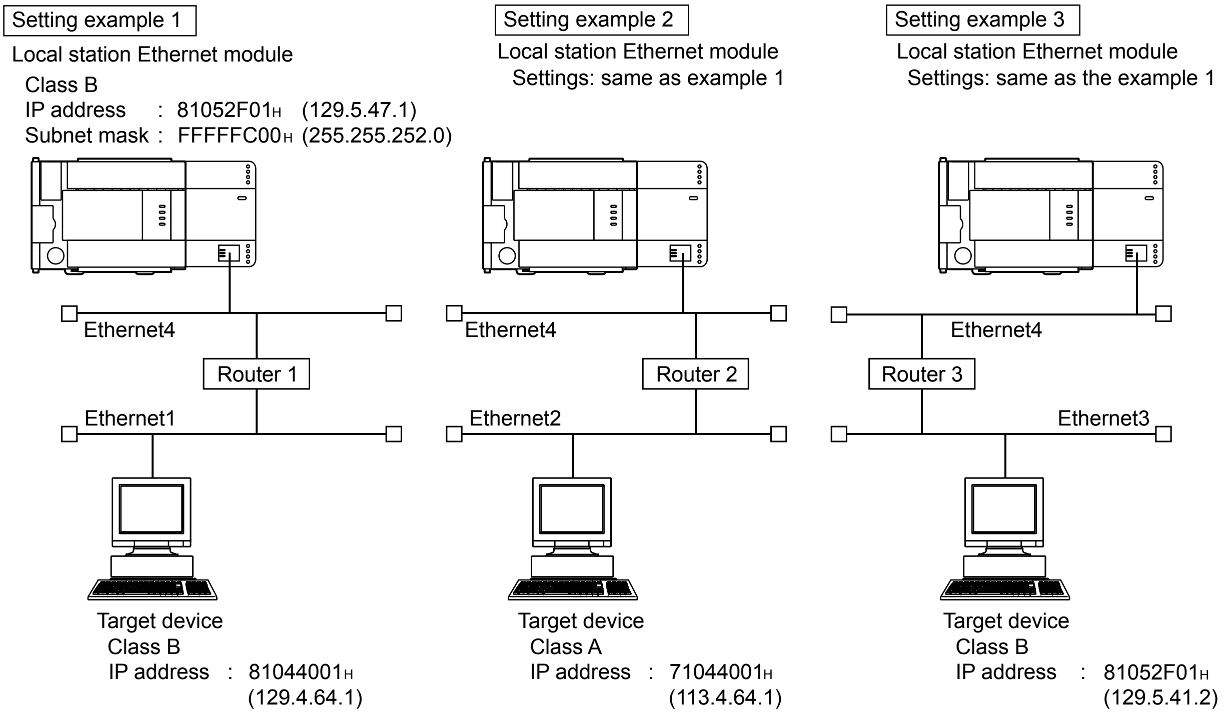
Set the value that satisfies the following conditions.

- Condition 1: The IP address class is either A, B or C.
- Condition 2: The sub-net address of the default router is the same as that of the local station Ethernet module.
- Condition 3: The host address bits are not all "0" or all "1".

(4) Sub-net mask pattern and router IP address

(a) Select the input format (decimal or hexadecimal) for each setting item.

(5) Setting examples


Setting example 1
Local station Ethernet module
Class B
IP address : $81052F01_H$ (129.5.47.1)
Subnet mask : $FFFFFC00_H$ (255.255.252.0)
Ethernet4
Router 1
Ethernet1
Target device
Class B
IP address : 81044001_H (129.4.64.1)
Setting example 2
Local station Ethernet module
Settings: same as example 1
Ethernet4
Router 2
Ethernet2
Target device
Class A
IP address : 71044001_H (113.4.64.1)
Setting example 3
Local station Ethernet module
Settings: same as the example 1
Ethernet4
Router 3
Ethernet3
Target device
Class B
IP address : $81052F01_H$ (129.5.41.2)

(Setting example 1) When the network addresses of the local station Ethernet module and target device differ

Network address

Local station Ethernet module IP address (Class B)	81H	05H	2FH	01H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 1	0 0 1 0 1 1 1 1	0 0 0 0 0 0 0 1
Target device 1) IP address (Class B)	81H	04H	40H	01H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 0	0 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1

⇩ Set the network address of the target device 1)

Network address

Subnet address set value	81H	04H	00H	00H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0

(Setting example 2) When the classes of the local station Ethernet module and target device differ

Network address

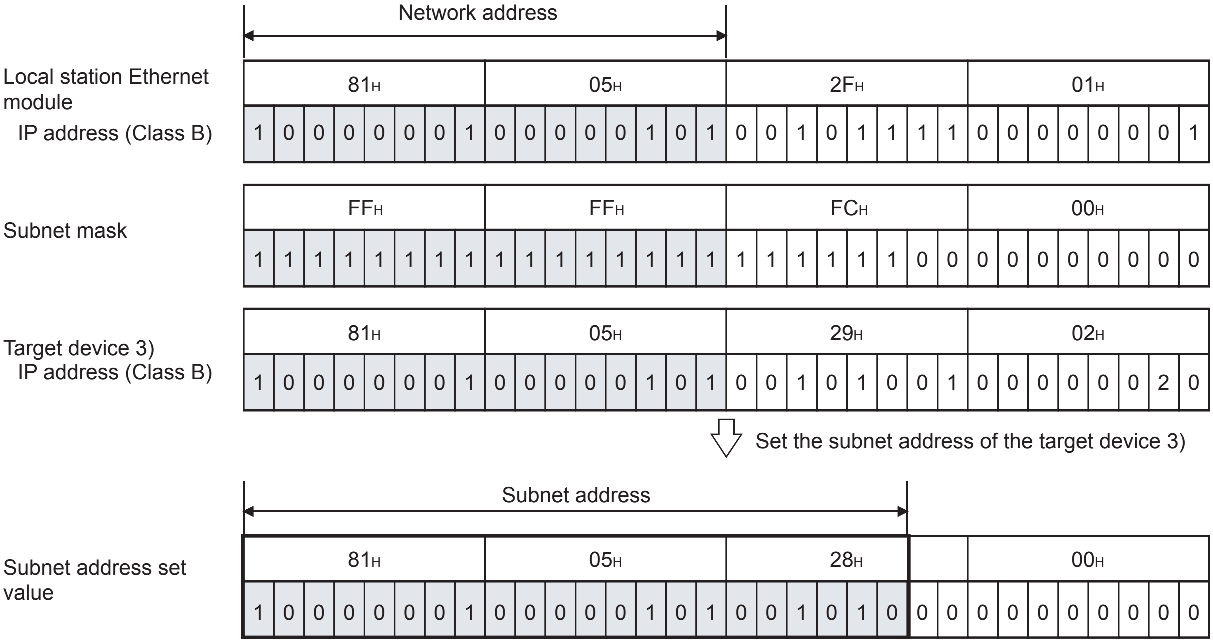
Local station Ethernet module IP address (Class B)	81H	05H	2FH	01H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 1	0 0 1 0 1 1 1 1	0 0 0 0 0 0 0 1
Target device 2) IP address (Class A)	71H	04H	40H	01H
	0 1 1 1 0 0 0 1	0 0 0 0 0 1 0 0	0 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1

⇩ Set the network address of the target device 2)

Network address

Subnet address set value	71H	00H	00H	00H
	0 1 1 1 0 0 0 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0

(Setting example 3) When the network addresses of the local station Ethernet module and target device are the same



Network address

Local station Ethernet module IP address (Class B)	81_H	05_H	$2F_H$	01_H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 1	0 0 1 0 1 1 1 1	0 0 0 0 0 0 0 1
Subnet mask	FF_H	FF_H	FC_H	00_H
	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 1 1 1 1 1 0 0	0 0 0 0 0 0 0 0
Target device 3) IP address (Class B)	81_H	05_H	29_H	02_H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 1	0 0 1 0 1 0 0 1	0 0 0 0 0 0 2 0

Set the subnet address of the target device 3)

Subnet address

Subnet address set value	81_H	05_H	28_H	00_H
	1 0 0 0 0 0 0 1	0 0 0 0 0 1 0 1	0 0 1 0 1 0 0 0	0 0 0 0 0 0 0 0

5.4 Confirming the Completion of the Initial Processing

The initial processing for the Ethernet module completes by selecting re-initialization, then turning the power ON again, or re-initializing via buffer memory operations after parameters are written to the Ethernet module.
(When the processing is completed normally, the [INIT.] LED on the front of the Ethernet module turns on.)

- "Operational settings" parameter
- "Initial settings" parameter

This section explains how to check the completion of the initial processing.

POINT

The status of the Ethernet module becomes communication enabled when the initial processing is completed normally. See reference sections for each communication function to perform communication.
When the initial processing has not been completed normally, do the following to check the error contents, take corrective actions, then execute the initial processing again.

- Check the error code using the "Parameter status" of the Ethernet diagnostics. (See Section 11.2.)
- Check the contents of the error corresponding to the error code, then take corrective actions. (See Section 11.4.3.)

5.4.1 PING command (Personal computer → Ethernet module)

The following example illustrates how to confirm the completion of the initial processing by issuing the PING command to the local station's Ethernet module from an external device connected on the same Ethernet network. (In the example, the confirmation is made between devices whose IP address class and sub-net address are identical.)

<Designation method>
ping IP address

<Example>
IP address of the Ethernet module: 192.0.1.254

Example of screen at normal completion

```
C:\>ping 192.0.1.254･･･Execute the ping command

Pinging 192.0.1.254 with 32 bytes of data:

Reply from 192.0.1.254: bytes=32 time=1ms TTL=128
Reply from 192.0.1.254: bytes=32 time<10ms TTL=128
Reply from 192.0.1.254: bytes=32 time<10ms TTL=128
Reply from 192.0.1.254: bytes=32 time<10ms TTL=128

Ping statistics for 192.0.1.254:
        Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)
Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>_
```

Example of screen at abnormal completion

```
C:\>ping 192.0.1.254･･･Execute the ping command

Pinging 192.0.1.254 with 32 bytes of data:

Request timed out:
Request timed out:
Request timed out:
Request timed out:

Ping statistics for 192.0.1.254:
 Packets: Sent = 4, Received = 0, Lost = 4 (100% loss)
Approximate round trip times in milli-seconds:
 Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>_
```

<When the PING command does not respond successfully>
Check the following items and send the PING command again.

- Check the Ethernet module's attachment.
- Check the connection to the Ethernet network.
- Check the contents of each parameter written to the Ethernet module.
- Check the operation condition of the Ethernet module (are there any irregularities?).
- Check the IP address of the Ethernet module dictated by the PING command.

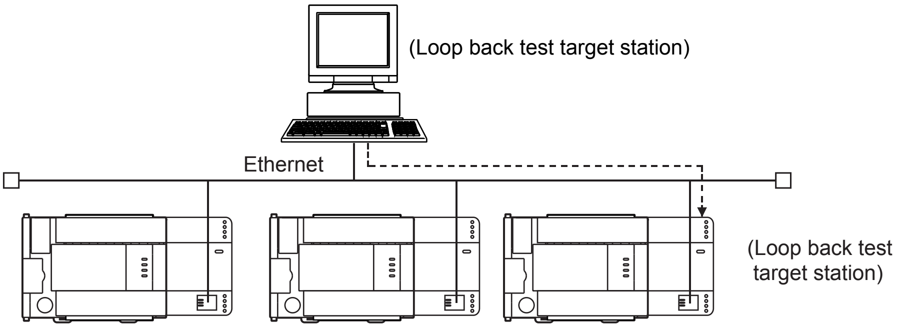
5.4.2 Loop back test (Communication using MC protocol)

The loop back test can be performed with communication using MC protocol in order to check the completion status of the initial processing for the target Ethernet module.
The following is an overview of the loop back test for communication using MC protocol.
See Chapter 9 for details.

(1) Loop back test for communication using MC protocol
The loop back test is a function designed to check the following:
- Whether a line has been properly connected to the test target Ethernet module.
- Whether the parameters for the Ethernet module have been correctly set.
- Whether the initial processing for the Ethernet module has been completed normally.
- Whether the program for the external device is running correctly.

(2) It is necessary to connect lines when performing communication using MC protocol with the user port on the Ethernet module side.
Perform the open processing for the connection to be used on the Ethernet module side.

(3) This function can only be used for the Ethernet module of the local station. The function cannot be used for the Ethernet module of another station via a network system.


(Loop back test target station)
Ethernet
(Loop back test target station)

5.5 Open Settings

This section explains the open setting.
A sequence program can perform open processing (establishing connection) with up to a maximum of 8 external device stations
Once a connection is established with an external device, it is possible to communicate using MC protocol, fixed buffer communication.
Thus, the open processing is required even when communicating using MC protocol.

NOTE

When adjusting settings in FX Configurator-EN (GX Developer), select the [Open Settings] from the initial screen and adjust the settings at the [Ethernet Open Settings] screen.

Setting item name	BFM number (Decimal)	Description of setting	Setting range/options
Protocol	32 to 39 (b8)	Set the communication method (protocol).	OFF : TCP/IP ON : UDP/IP
Open system	32 to 39 (b15, b14)	Select the connection open system.	00 : Active open 10 : Unpassive open 11 : Fullpassive open A002 : MELSOFT connection ＊1
Fixed buffer	32 to 39 (b0)	Select the usage of the fixed buffer.	OFF : Send ON : Receive
Fixed buffer communication	32 to 39 (b10, b9)	Select which protocol is used for fixed buffer communication.	00 : Procedure exist 10 : MC (App) 01 : No procedure
Pairing open	32 to 39 (b7)	Select whether paring open is used or not.	OFF : Pairs ON : No pairs
Existence confirmation	32 to 39 (b1)	Select whether the continued existence of a destination station for a connection should be confirmed or not.	OFF : No confirm ON : Confir
Local station Port No.	40, 47, 54, 61, 68, 75, 82, 89	Set the local station's port No.	1025 to 5548 or 5552 to 65534
Destination IP address	41 to 42, 48 to 49, 55 to 56, 62 to 63, 69 to 70, 76 to 77, 83 to 84, 90 to 91	Set the IP address of an external device.	1_H to $FFFFFFFE_H$
Dest. Port No.	43, 50, 57, 64, 71, 78, 85, 92	Set the port No. of an external device.	401_H to $FFFF_H$

＊1 Select $A002_H$ in BFM#32 to 39 for MELSOFT connection.

(1) Protocol (connection numbers 1 to 8; BFM#32 to 39 b8)

(a) Select the protocol for each connection.

Name of setting	Description of setting
TCP	Communicate using TCP/IP.
UDP	Communicate using UDP/IP.

(b) For protocols (TCP/UDP), see 1.3, "Software Configuration".

(2) Open system (connection numbers 1 to 8; BFM#32 to 39 b15, b14)

(a) Select the connection open system for each connection for which "TCP" is selected in "(1) Protocol". If "UDP" is selected, the specification of this item is not required.

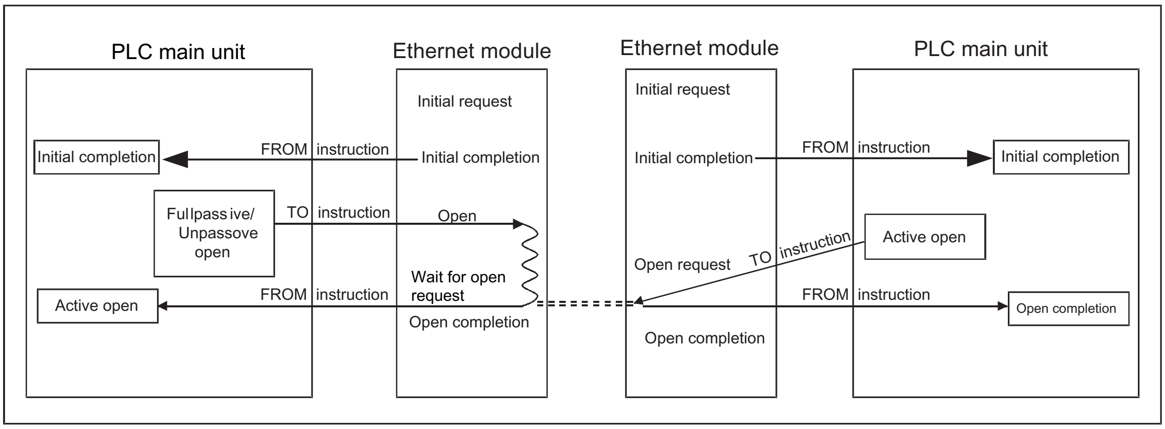
Name of setting	Description of setting
Active	Perform active open processing to an external device that waits for a passive open (Fullpassive/Unpassive) on the TCP connection.
Unpassive	Perform passive open processing on the TCP connection addressing all the devices connected to a network. (The local station is placed in the wait status to wait for an Active open request to be sent.
Fullpassive	Perform passive open processing on the TCP connection, only addressing specific devices. (The local station is placed in the wait status to wait for an Active open request to be sent.) The local station waits for an Active open request from the opposite station set in "(8) Destination IP address".
MELSOFT connection ＊1＊2＊3	Used to connect MELSOFT products via TCP/IP communication. Perform passive open processing on the TCP connection, addressing all the MELSOFT products connected to a network. (The local station is placed in the wait status for an Active open request to be sent.)

＊1 Regardless of the initial timing setting in the operation setting (refer to Section 4.7), this connection will always wait for the open status.

＊2 The set connection is dedicated to data communication with the MELSOFT products.

＊3 When simultaneously connecting to multiple MELSOFT products, set the connections to be as many as there are MELSOFT products.
A maximum of four connections can be set. However, when setting [Usable] in the open settings for [MC Protocol], the number of usable connections decrease by the number set in [MC Protocol].

The number of connections to MELSOFT products +
Number of connections which made MC protocol operational ≦ 4



(3) Fixed buffer (connection numbers 1 to 8; BFM#32 to 39 b0)

(a) Here it is selected whether the fixed buffer corresponding to each applicable connection number will be used for sending or receiving when communicating using the fixed buffers.

Name of setting	Description of setting
Send	For sending or fixed buffer communication is not executed.
Receive	For receiving.

(b) When both sending and receiving are performed with an external device using fixed buffer communication, one buffer for sending and one for receiving are required. Thus, two connections should be set.

(c) Whether the fixed buffers are set for sending or receiving, external devices can communicate using MC protocol.

(4) Fixed buffer communication (connection numbers 1 to 8; BFM#32 to 39 b9 to b10)

(a) For this item, select the communication method when communicating using the fixed buffers.

Name of setting	Description of setting
Procedure exist	• In fixed buffer communication, data is communicated in 1:1 by handshaking with the external device.
Procedure exist (MC)	• In fixed buffer communication, 1:1 data communication is performed by handshaking with the external device. • Communication using MC protocol can be performed.
No procedure	• The no procedure fixed buffer communication uses dedicated connections. • The PLC and external devices communicate data in 1:1 • The handshaking with an external device must be performed using a sequence program.

(b) When communicating using MC protocol, adjust the setting of the fixed buffer communication procedure to [Procedure exist (MC)].
In the settings other than [Procedure exist (MC)], communication using MC protocol cannot be performed.

(5) Pairing open (connection numbers 1 to 8; BFM#32 to 39 b7)

(a) Select whether or not the Ethernet module's receiving and sending connections are made into one pair and connected to one port of an external device when using fixed buffer communication (either of the procedure exist or no procedure can be designated).
For more detail, see [Section 5.7 Pairing Open].

Name of setting	Description of setting
No pairs	Does not use the pairing open method.
Pairs	Uses the pairing open method.

(6) Existence confirmation (connection numbers 1 to 8; BFM#32 to 39 b1)

(a) This setting selects whether or not the Ethernet module should confirm that an external device still operates normally when there is no communication for a fixed period of time. The open processing for the connection with the external device must have been completed.

Name of setting	Description of setting
No confirm	Do not confirm the existence of the external device.
Confirm	Confirm the existence of the external device. For details on the settings of the existence confirmation time and others, see Section 5.2, "Initial Settings".

(b) When an error occurs in the existence confirmation, the Ethernet module performs the following processing.

- Forcibly closes the line and stores the error information in the error log area (BFM#227 to 511) of the buffer memory.
- Turns off the open completion signal (BFM#20480) and stores the open error code in BFM#124, 134, 144, 154, 164, 174, 184, 194.

(c) If the external device will be changed while a UDP/IP connection is open, "No confirm" should be selected.
If "Confirm" is selected, the Ethernet module will confirm the existence of the first destination after the UDP/IP connection is opened. Existence confirmation is not performed for the changed destination, i.e. the newly selected external device.

(7) Local station Port No. (connection numbers 1 to 8; BFM#40, 47, 54, 61, 68, 75, 82, 89)

(a) In this item the port number of each connection for the Ethernet module is set in Decimal.

(b) The setting values are designated in the range from 1025 to 5548 and from 5552 to 65534. Set port numbers that are not already used by other ports.

(c) Set the port numbers for the Ethernet module upon consulting a network administrator.

(8) Destination IP address (connection numbers 1 to 8; BFM#41 to 42, 48 to 49, 55 to 56, 62 to 63, 69 to 70, 76 to 77, 83 to 84, 90 to 91)

(a) Set the IP addresses (two words) for external devices in the chosen input format (decimal/hexadecimal).

(b) The IP addresses of external devices must be given values other than 0_H.

(c) Set the IP addresses of external devices upon consulting a network administrator.

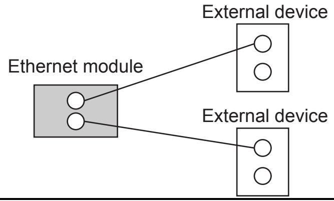
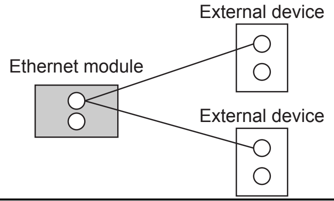
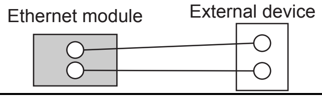
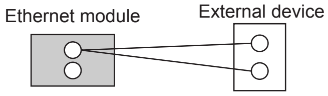
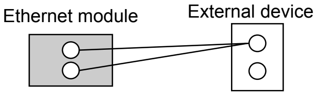
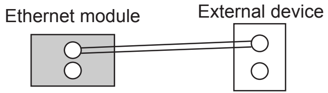
(9) Destination Port No.
(connection numbers 1 to 8; BFM#43, 50, 57, 64, 71, 78, 85, 92)

(a) Set the port numbers of the external devices for each connection in decimal.

(b) The port numbers of the external devices are set in the range from 1025 to 65535.

(c) Set the port numbers for the external device upon consulting a network administrator.

(d) The following table provides some precautions that should be observed when setting port numbers.
(□ in the diagram indicates a device and ○ indicates a port number.)

Status of connection establishment (○: Port (indicates port number))	Description of connection	Communication protocol	
		TCP	UDP
	When connecting to multiple external devices, set multiple port numbers for the local station.	○	○
	When connecting to multiple external devices, set a single port number for the local station. (However, it is necessary to open a connection for each external device) This cannot be chosen when the local station is Unpassive.	○	×
	When connecting to multiple ports of an external device, set multiple port numbers for the Ethernet module.	○	○
	When connecting to multiple ports of an external device, set a single port number for the Ethernet module. (However, it is necessary to open a connection for each external device port) This cannot be chosen when the local station is Unpassive.	○	×
	When connecting to the same port of an external device, set multiple port numbers for the Ethernet module. (However, it is necessary to open a connection for each Ethernet module port)	○	○
	Setting multiple ports numbers for the same port of an external device and the Ethernet module is possible only when paring open is set.	○	○

Important

Parameters for connections that communicate by Passive open and UDP open must always be set from this screen when "Always wait for OPEN (communication possible at STOP time)" is selected in the Operational settings (see Section 4.6).

POINT

Set parameters according to the open method to be used for open connection.

Communication system open system Parameter		TCP				UDP	
		Active		Passive			
		ARP function of external device		Un-passive	Full-passive	ARP function of external device	
		Yes	No			Yes	No
Communication address	Local station Port No.	○	○	○	○	○	○
	Destination IP address	○	○	×	○	○	○
	Destination Port No.	○	○	×	○	○	○
	Destination Ethernet address (＊2)	○ (＊1)	○	×	×	○ (＊1)	○

＊1 Use the default value (FFFFFFFFFFFF$_H$) or "0".

＊2 When using the "Open settings" of FX Configurator-EN (GX Developer), the default value is used.

5.6 Open Processing/Close Processing of the Connection

This section explains the open processing/close processing using sequence programs.

(1) Open processing

(a) The purpose of the open processing is to establish a connection with an external device in order to perform the following forms of data communication.
They can all be performed with an external device opened by the user.
- Communication using MC protocol
- Sending/receiving using the fixed buffers (Procedure exists)

(b) When the following is set by the parameter settings, the open processing should be performed in a sequence program.
1) In the Operational settings (Section 4.6)
When "Do not wait for OPEN" is set in the "Initial timing setting".
2) In the open settings (Section 5.5)
When "Active" is set in the "OPEN system" setting.

(c) In order to perform open processing, the initial processing must be complete (BFM #28b0: ON) and the hub communication status (BFM #28b7: ON) must be on.

(d) A connection with an external device must be established (open processing) when communicating using either the MC protocol, fixed buffers. (＊1)
Two types of data communication mentioned above can also be performed with an external device opened by the user.
＊1 Since the Ethernet module recognizes the external device by the IP address, the open processing is required for UDP communication.

(e) Up to a maximum of 8 connections can be opened to external devices.
However, two buffers are required when communicating with the same external device using fixed buffer communication, so in this case the number of external devices that can be communicated with will be lower.

POINT

During communication using MC protocol, if data communication is to continue even after the Ethernet module installed PLC, has been placed in the STOP status, enable "Always wait for OPEN (communication possible at STOP time)" under "Initial timing setting" (see 4.6, "Operation Settings").

(2) Close processing

(a) The purpose of the close processing is to disconnect (cancel) the connection with the external device established by open processing mentioned previously.

(b) The close processing is used when terminating a connection with an external device, changing an external device of a connection, changing communication conditions, etc.

(c) Perform the close processing for connections established by open processing using sequence programs.

(d) Determine the timing of close processing with the external device.

The examples in the following sections describe the procedures for establishing a connection from the Ethernet module to an external device and subsequently closing it again by open and close processing for connection number 1.

• TCP/IP Active open	: See 5.6.1, "Active open processing/close processing".
• TCP/IP Passive open	: See 5.6.2, "Passive open processing/close processing".
• UDP/IP Open	: See 5.6.3, "UDP/IP open processing/close processing".

POINT

Except when the close processing is requested, the open completion signal (applicable bit of BFM#20480) automatically turns off and the communication line is closed in the following cases:

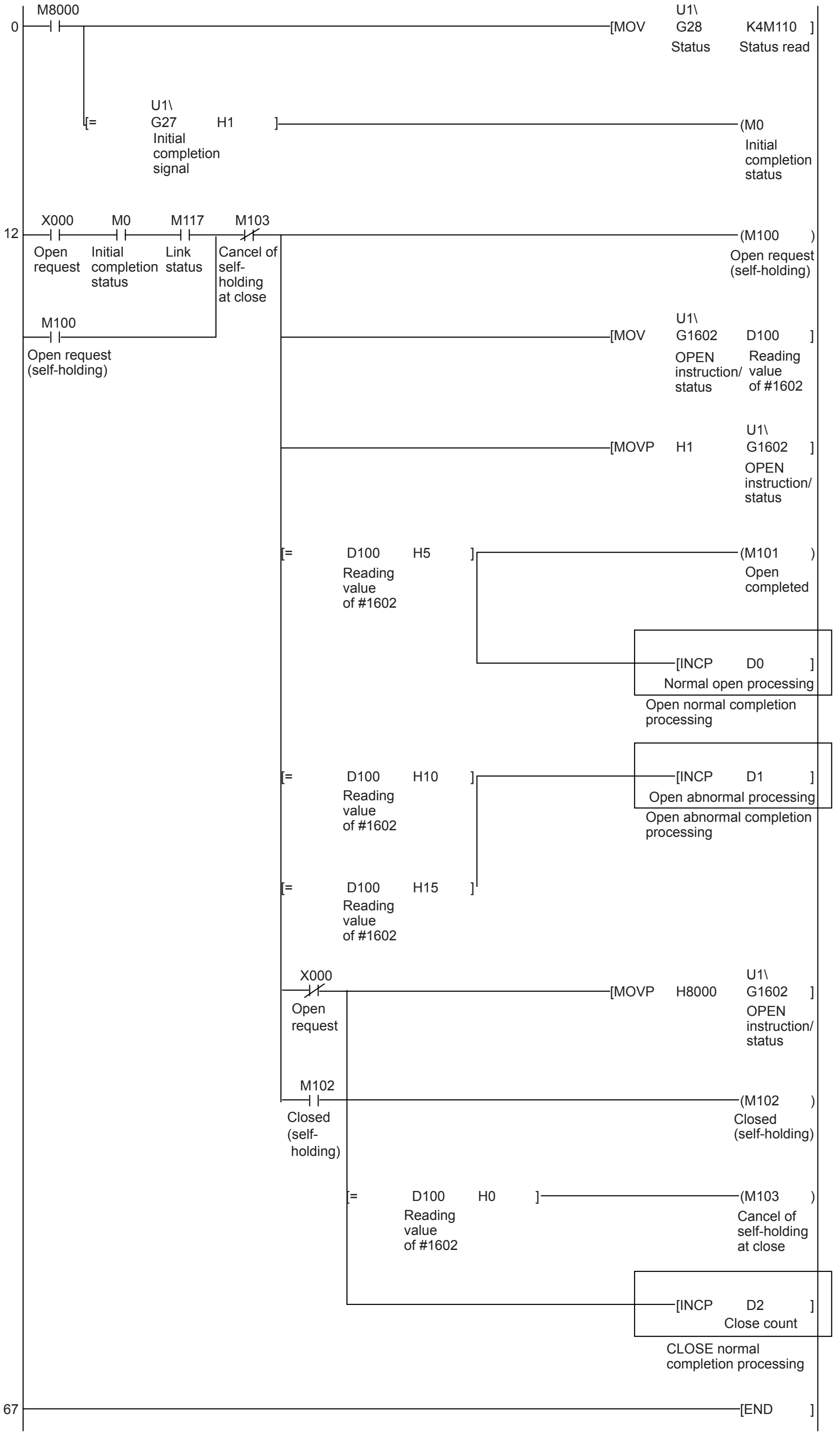
(1) When the existence confirmation function times out (see Section 5.5).

(2) When a close is received from an external device.

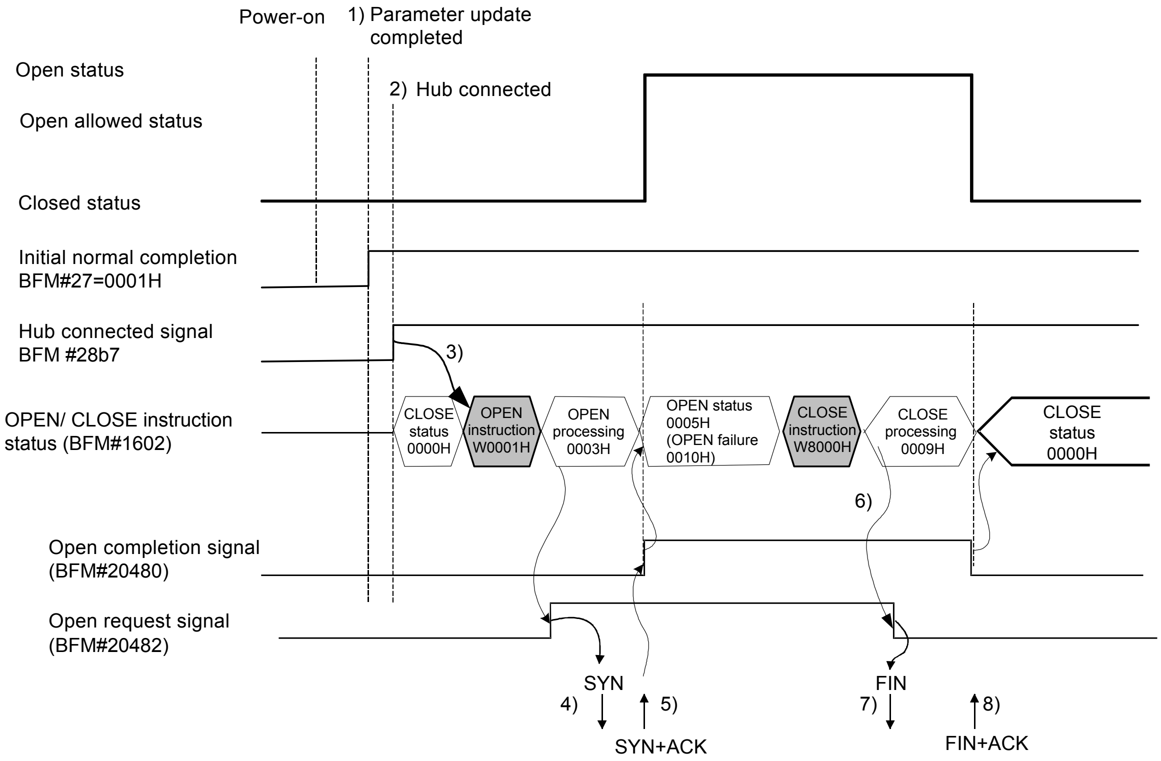
(3) When the Active open request is received again from the external device in the open completion status of TCP.
After returning ACK to the external device, the Ethernet module closes the connection when the RST command is received from the external device.
However, when receiving the Active open request again from the external device with a different IP address or port No., the Ethernet module only sends the RST command. (It does not close the connection.)

5.6.1 Active open processing/close processing

This section explains the procedure for opening and closing a connection with an external device from the Ethernet module.



User applications are described at the area enclosed with □.


Power-on
1) Parameter update completed
2) Hub connected
Open status
Open allowed status
Closed status
Initial normal completion BFM#27=0001H
Hub connected signal BFM #28b7
3)
OPEN/ CLOSE instruction status (BFM#1602)
CLOSE status 0000H
OPEN instruction W0001H
OPEN processing 0003H
OPEN status 0005H (OPEN failure 0010H)
CLOSE instruction W8000H
CLOSE processing 0009H
CLOSE status 0000H
6)
Open completion signal (BFM#20480)
Open request signal (BFM#20482)
SYN
4)
5)
SYN+ACK
FIN
7)
8)
FIN+ACK

1) After communicating the parameter settings, confirm the normal completion of the Ethernet module initial processing.
(Initial completion signal BFM#27 "0001H")

2) Confirm that the Ethernet module and hub are connected. (BFM#28 b7:ON)

3) Start the open processing writing with pulse execution instruction 0001H to BFM#1602.

4) The Ethernet module executes the open processing.
   - Sends the open request (SYN).

5) When the open processing completes normally
   - Open completion signal BFM#20480 b0 : ON
   - BFM#1602 "0005H"

   Data communication is enabled.

   When the open processing completes abnormally (∗1)
   - The open error code is stored in the buffer memory. (∗2)
   - Open error detection (BFM#1602 "0010H" or "0015H")
   - Area for module status (BFM#28 b6:ON)

6) Start the close processing writing 8000H to BFM#1602.

7) The Ethernet module executes the close processing.
   - Sends the close request (FIN).

8) When the close processing completes normally (∗3)
   - Open completion signal (BFM#2048 b0) : OFF
   - Open/Close instruction status (BFM#1602 "0000 H ")

POINT

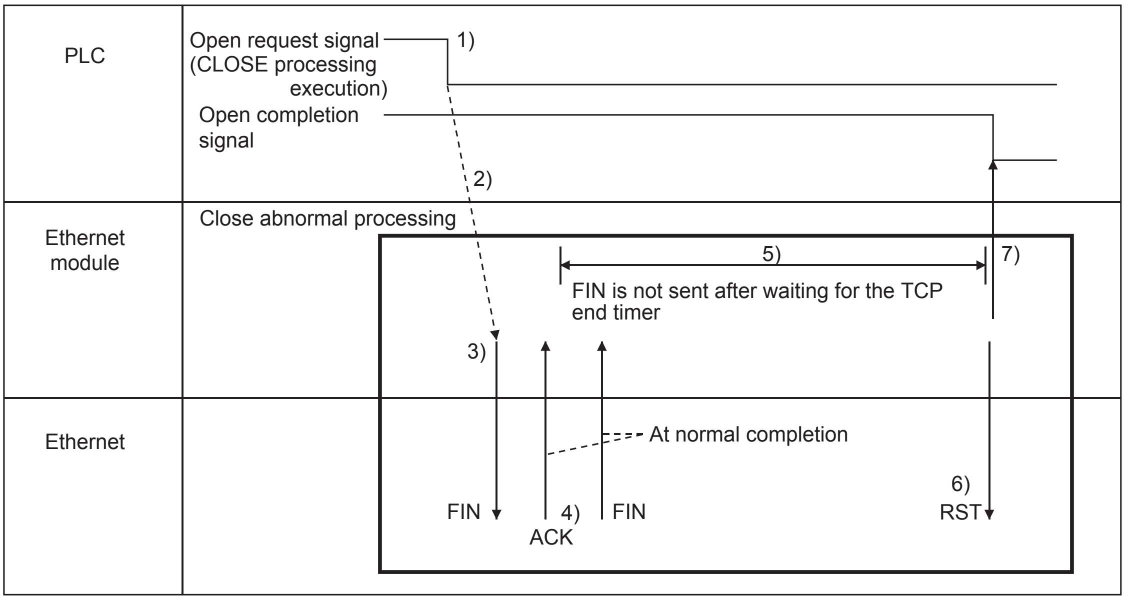
This example uses connection number 1 for explanation. Use the corresponding BFM No. and bits for other connection numbers.

＊1 Processing when the open processing is abnormally completed (TCP)
When opening from the Ethernet module under a normal situation, if the Ethernet module sends a SYN, the external device returns an ACK and a SYN.
However, if the Ethernet module sends a SYN and then the external device returns a RST, the open abnormal completion (BFM#28 b6) is immediately turned on and open processing ends.

＊2 The open status and error code at abnormal end can be confirmed with the following buffer memory.

- Each connection open abnormal code area of the communication status storage area
  (Connection numbers 1 to 8; BFM#120 to 199)
- Error log area (BFM#227 to 372)
- Error codes stored in the open error code area are cleared when writing "0001H" to the BFM #1602 to 1609 to perform the open operation.

＊3 Processing when the close processing is abnormally completed (TCP)
When closing normally from the Ethernet module, the Ethernet module sends a FIN request and the external device returns an ACK and a FIN.
However, if an ACK and a FIN are not returned because the external device is faulty, the Ethernet module forcibly disconnects the connection (sends a RST message).


PLC
Open request signal (CLOSE processing execution)
1)
Open completion signal
2)
Ethernet module
Close abnormal processing
5)
7)
FIN is not sent after waiting for the TCP end timer
3)
Ethernet
At normal completion
6)
FIN
4)
ACK
FIN
RST

1) The open request signal turns off by writing "8000H" to the BFM#1602.
2) The Ethernet module executes the close processing.
3) The Ethernet module sends a FIN request to the external device.
4) The external device sends back FIN and ACK messages in reply to the FIN request sent by the Ethernet module.
   (When the reply is not returned, the Ethernet module sends the FIN request again.)
5) The Ethernet module waits for the external device to send an ACK and a FIN.
   (The module waits for the amount of time set in the TCP end timer value. For details on how to set it, see Section 5.2, "Initial Settings".)
   If the ACK and FIN messages are received at this point, it returns an ACK as in the normal processing.
6) If an ACK and a FIN are not received within the time designated by the TCP end timer, an RST message is sent to the external device.
7) The Ethernet module determines that the close procedure is completed and turns off the open completion signal regardless of the status of the external device.

NOTE

(1) When the procedure above is performed, the Ethernet module determines that the closing of the external device is executed normally, thus the close processing result is not stored in the error log area.

(2) The procedure described above is a special function of the Ethernet module; it is not available for general TCP/IP protocols.

Program example

This example explains a program for open processing/close processing when Active open is selected in the Open system setting.

(1) Execution environment for the program example

(a) The Ethernet module is connected to the second special module.

(b) The unit number specified at the sequence program and FX Configurator-EN (GX Developer) is 1.

(c) Make Operation settings in the following manner.

Communication data code	: Binary code (BFM#24 b1 : OFF)
Initial timing	: Do not wait for OPEN (Communications impossible at STOP time) (BFM#24 b8 : OFF)
IP address	: 10.97.85.222 (0A.61.55.DE$_H$) (BFM#106 to 108)

(d) Make open settings in the following manner.

Protocol	: TCP (BFM#32 b8 : OFF)
Open system	: Active (BFM#32 b14, b15 : 00$_H$)
Fixed buffer	: Send (BFM#32 b0 : OFF)
Fixed buffer communication procedure	: Procedure exist (BFM#32 b9 : OFF)
Pairing open	: Disable (BFM#32 b7 : OFF)
Existence confirmation	: No confirm (BFM#32 b1 : OFF)
Host station Port No.	: 4096 (BFM#40)
Transmission target device IP address	: 10.97.85.223 (0A.61.55.DF$_H$) (BFM#41 to 42)
Transmission target device Part No.	: 8192 (BFM#43)

(e) The following contact signals are used in the program.

Ethernet module status	: M110 to M115
Connection No.1 open/close status and instruction	: D100

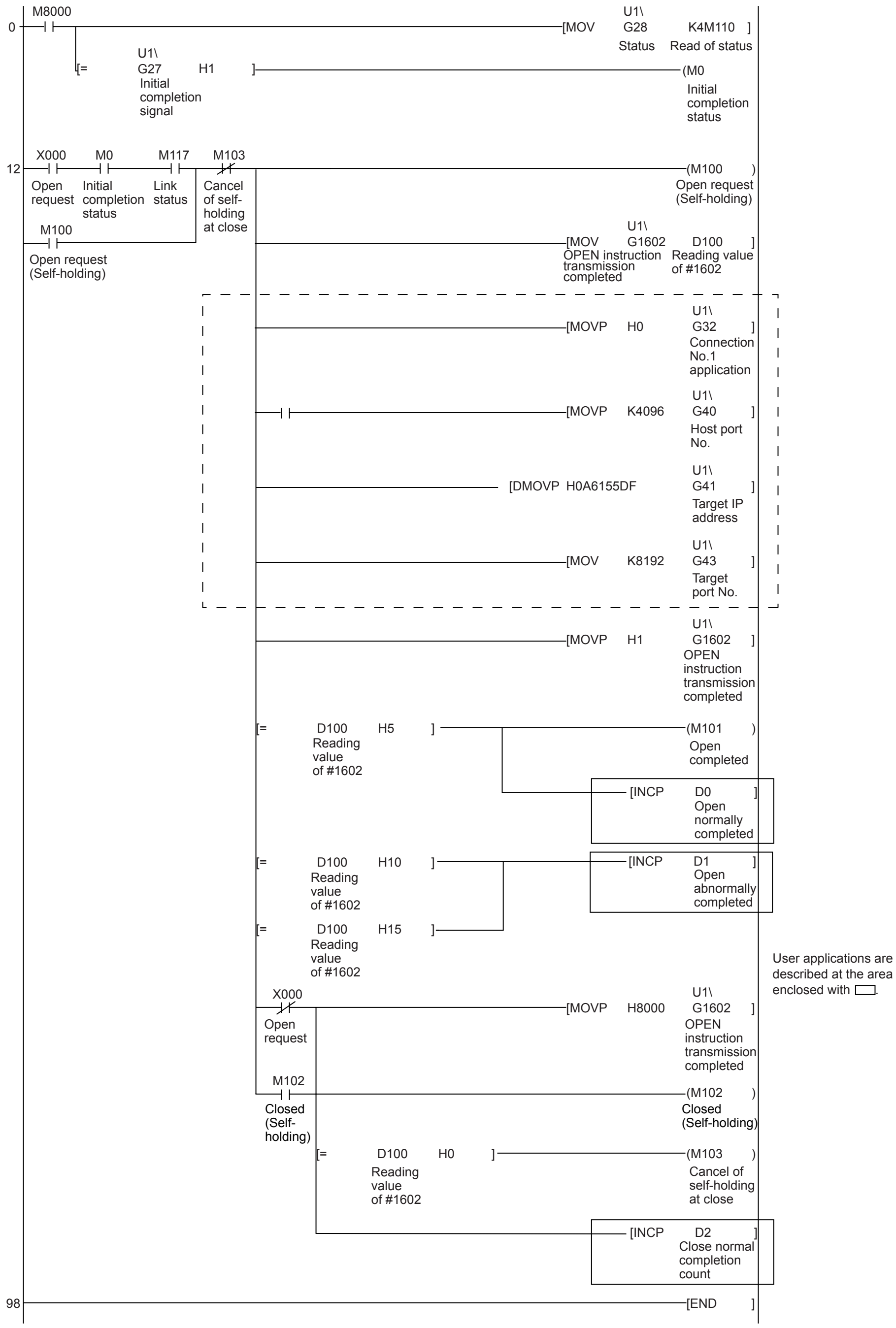
(f) The area enclosed with ⬚ in the program example should be used when the [Open settings] Ethernet module parameters are not set for FX Configurator-EN (GX Developer).
This part of the program is not required when the [Open settings] parameters are used for FX Configurator-EN (GX Developer).

(2) Outline of the program example

(a) After each parameter is set in FX Configurator-EN (GX Developer) or the sequence program and written to the Ethernet module, re-initialize or power ON the module again and confirm the completion of the initial processing via buffer memory operations.

(b) The Ethernet module performs the open processing for connection No. 1 to the external device set in the [Open settings] or control data.

(c) The close processing for connection No. 1 is performed according to the close instruction to the Ethernet module or the close request from the external device.

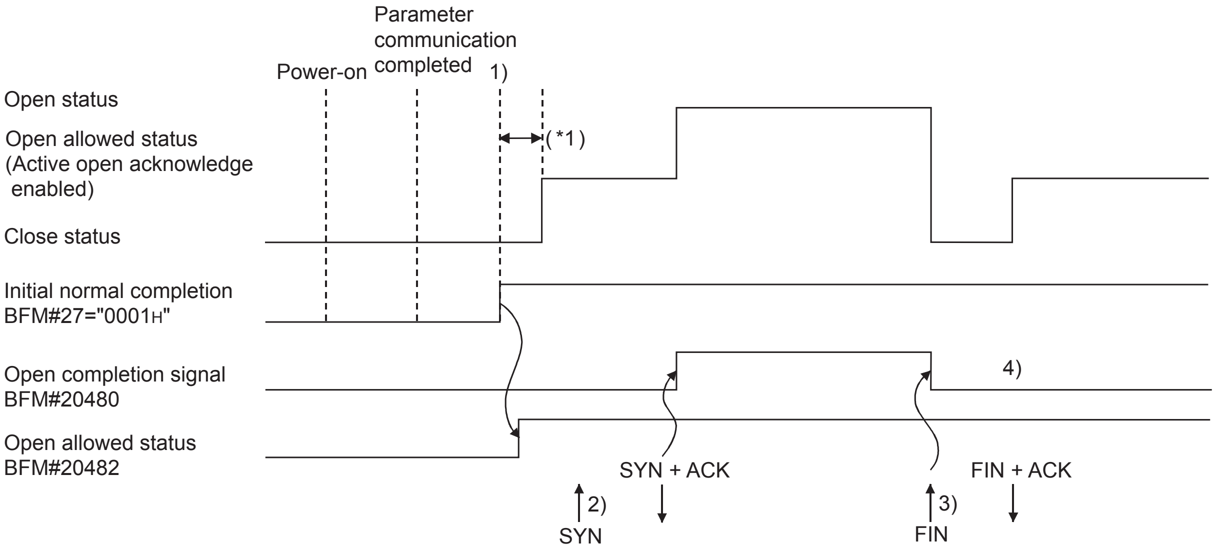

0
M8000
[MOV U1\ G28 Status K4M110] Read of status
[= U1\ G27 Initial completion signal H1]
(M0 Initial completion status
12
X000 Open request
M0 Initial completion status
M117 Link status
M103 Cancel of self-holding at close
(M100) Open request (Self-holding)
M100 Open request (Self-holding)
[MOV U1\ G1602 OPEN instruction transmission completed D100] Reading value of #1602
[MOVP H0 U1\ G32] Connection No.1 application
[MOVP K4096 U1\ G40] Host port No.
[DMOVP H0A6155DF U1\ G41] Target IP address
[MOV K8192 U1\ G43] Target port No.
[MOVP H1 U1\ G1602] OPEN instruction transmission completed
[= D100 Reading value of #1602 H5]
(M101) Open completed
[INCP D0] Open normally completed
[= D100 Reading value of #1602 H10]
[INCP D1] Open abnormally completed
[= D100 Reading value of #1602 H15]
User applications are described at the area enclosed with □.
X000 Open request
[MOVP H8000 U1\ G1602] OPEN instruction transmission completed
M102 Closed (Self-holding)
(M102) Closed (Self-holding)
[= D100 Reading value of #1602 H0]
(M103) Cancel of self-holding at close
[INCP D2] Close normal completion count
98
[END]

5.6.2 Passive open processing/close processing

This section explains the procedure for opening and closing a connection with the Ethernet module via an external device.
The operations of Passive open processing/close processing differ depending on whether "Always wait for OPEN" or "Do not wait for OPEN" is selected, as shown in this section.

(1) When "Always wait for OPEN" is selected in the operational setting

This is an explanation of (BFM #24 b8: ON) open processing/close processing when initial timing settings for Operational Settings are set to be always wait OPEN (Communication Possible at STOP).
In this case, sequence programs for open processing and close processing are not required because the Ethernet module keeps the connection in the always wait for the OPEN status according to the [Open settings] parameter setting.
For detail on the [Open settings] parameter, see Section 5.5, "Open Settings".


Power-on
Parameter communication completed
1)
(*1)
Open status
Open allowed status (Active open acknowledge enabled)
Close status
Initial normal completion BFM#27="0001H"
Open completion signal BFM#20480
Open allowed status BFM#20482
4)
SYN + ACK
FIN + ACK
2)
SYN
3)
FIN

＊1 An open request (SYN) received after the normal completion of an initial processing and before the Ethernet module is placed in the open acknowledge enabled status generates an error, and the Ethernet module sends a connection forced close (RST).

1) After the parameters are sent, the normal completion of the Ethernet module initial processing is confirmed (Initial normal completion signal (BFM#27 "0001H")
After the initial processing is normally completed, the connection is placed in the open allowed status and the Ethernet module waits for an open request from the external device.

2) The Ethernet module starts the open processing upon receiving the open request (SYN) from the external device.
When the open processing is normally completed (Open/close status (BFM#1602 "0005H")), the open completion signal (BFM#20480 b0) turns on and data communication is enabled.

3) The Ethernet module starts the close processing upon receiving the close request (FIN) from the external device.
When the close processing is completed (BFM#1602 "0000H"), the open completion signal (BFM#20480 b0) turns off and data communication is disabled.

4) After the Ethernet module’s internal processing is Rissn/completed, the connection returns to the open acknowledge enabled status.

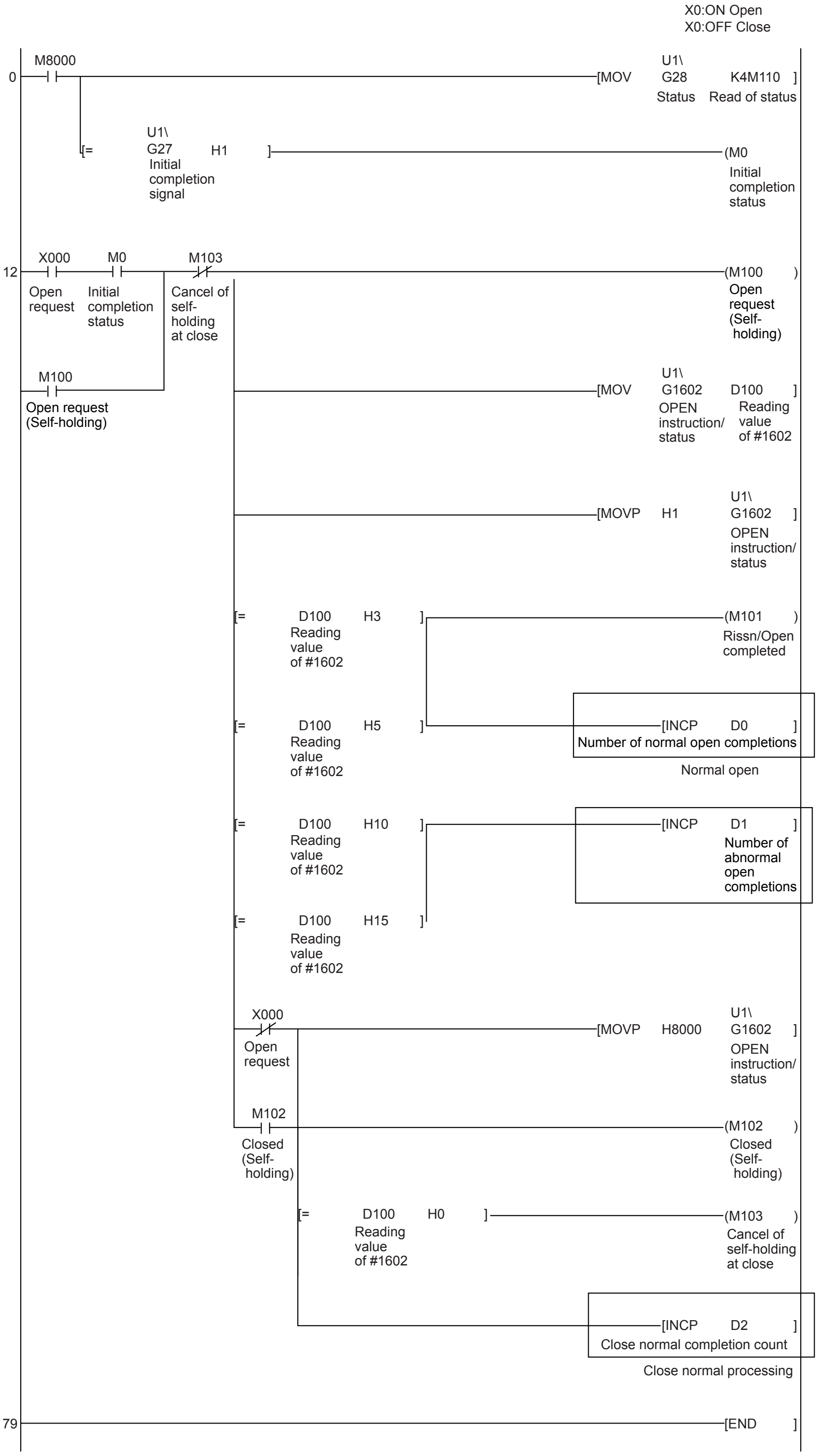
NOTE

For Passive open connections for which [Always wait for OPEN (Communication possible at STOP time)] is selected in [Operational settings], the connection open/close processing of the Ethernet module side is performed according to the open/close request from the external device.
When the close processing is performed from the Ethernet module (this station), the applicable connection will not return to the open acknowledge enabled status after the close processing, even if [Always wait for OPEN (communication possible at STOP time)] is selected in [Operational settings].
(It requires the same open processing and close processing as the connection for Passive open for which [Do not wait for OPEN (communication impossible at STOP)] is selected.)

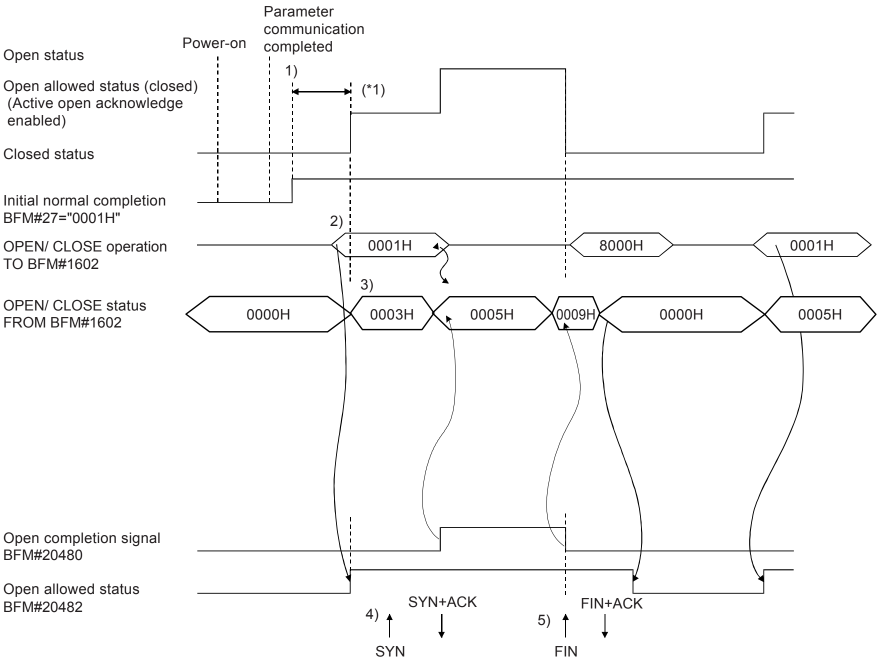
(2) When "Do not wait for OPEN" is selected in the operational setting

When initial timing settings of operational settings are at [Do not wait for OPEN (communication impossible at STOP time)], the (BFM #24 b8: OFF) open processing/close processing is performed as explained below.

Since open processing/close processing is executed by a sequence program, an external device can be changed while the connection is established.



User applications are described at the area enclosed with □.


Parameter communication completed
Power-on
Open status
Open allowed status (closed) (Active open acknowledge enabled)
Closed status
Initial normal completion BFM#27="0001H"
OPEN/ CLOSE operation TO BFM#1602
OPEN/ CLOSE status FROM BFM#1602
Open completion signal BFM#20480
Open allowed status BFM#20482
1)
(*1)
2)
3)
0001H
8000H
0001H
0000H
0003H
0005H
0009H
0000H
0005H
4)
SYN+ACK
SYN
5)
FIN+ACK
FIN

1) After communicating the parameter settings, confirm the normal completion of the Ethernet module initial processing.
(Initial normal completion signal (BFM#27 "0001H":)

2) Start the open processing by writing "0001H" to the BFM#1602.
(Open request signal (BFM#20482 b0) : ON

3) OPEN possible (Listen status) from external device. (Open/close status BFM #1602 “0003H”)

4) The Ethernet module starts the open processing upon receiving the open request (SYN) from the external device.

At normal completion
- Open completion signal (#BFM20480 b0) : ON
- BFM#1602 : 0005H

Data communication is enabled.

At abnormal completion
- Open completion signal : OFF
- The open error code is stored in the buffer memory
- Open abnormal detection signal (BFM28 b6) : ON
- OPEN/CLOSE instruction status (BFM#1602 "0010H" or "0015H")

5) The Ethernet module starts the close processing upon receiving the close request (FIN) from the external device.
When the close processing is completed, the open completion signal turns off and data communication is disabled.

POINT

This example uses connection number 1 for explanation. Use the appropriate BFM signals and bits for other connection numbers.

＊1 If an open request (SYN) received after the normal completion of an initial processing and before the Ethernet module is placed in the open acknowledge enabled status, an error is generated, and the Ethernet module sends a connection forced close (RST).

NOTE

(1) If the settings of the connection need modifying, the modifications should be done before writing the open instruction to the buffer memory.

(2) Once open processing is executed, an open request cannot be canceled before the open processing is completed.
Write the close instruction to the buffer memory after the open processing has been completed.

Program example

This example explains a program for open processing/close processing when Unpassive open is selected in the open system setting.

(1) Execution environment for the program example

(a) The Ethernet module is connected to the second special module.

(b) The unit number specified at the sequence program and in FX Configurator-EN (GX Developer) is 1.

(c) Adjust Operation settings in the following manner.

Communication data code	: Binary code (BFM#24 b1 : OFF)
Initial timing	: Do not wait for OPEN (Communications impossible at STOP time) (BFM#24 b8 : OFF)
IP address	: 10.97.85.223 (0A.61.55.DF$_H$) (BFM#106 to 107)

(d) Make open settings in the following manner.

Protocol	: TCP (BFM#32 b8 : OFF)
Open system	: Unpassive (BFM#32 b14, b15 : 10$_H$)
Fixed buffer	: Send (BFM#32 b0 : ON)
Fixed buffer communication procedure	: Procedure exist (BFM#32 b9 : OFF)
Pairing open	: Disable (BFM#32 b7 : OFF)
Existence confirmation	: No confirm (BFM#32 b1 : OFF)
Host station Port No.	: 8192 (BFM#40)

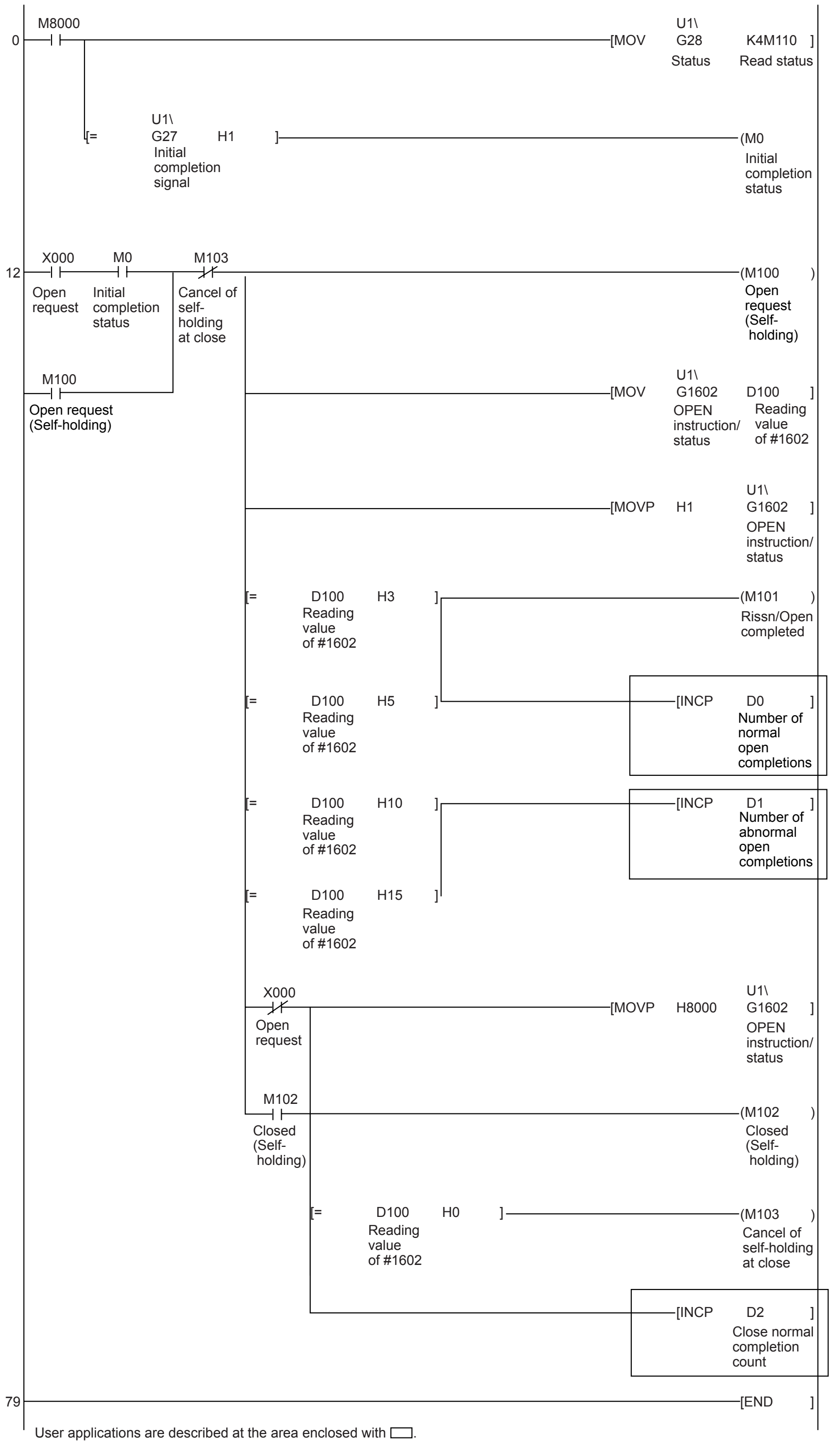
(e) The following contact signals are used in the program.

X000 Open instruction
M100 Open request self-hold
M101 Open complete
M102 Closed indicator self-hold
M103 Cancel self-hold when closed

(2) Outline of the program example

(a) The open processing for connection No. 1 of the Ethernet module is performed.
After the completion of the open processing, connection No. 1 waits for the open request from the external device.

(b) The close processing for connection No. 1 is performed according to the close instruction (X000=OFF) to the Ethernet module.



User applications are described at the area enclosed with □.

5.6.3 UDP/IP open processing/close processing

The following explains the UDP/IP open processing.

The operations of the open processing/close processing in UDP/IP differ depending on whether “Always wait for OPEN” or “Do not wait for OPEN” is selected in the initial timing settings (BFM #24 b8) from operational settings.

(1) When "Always wait for OPEN" is selected in the operational settings (Communications possible at STOP time)

According to the Open settings (BFM#32), a selected connection for the UDP/IP communication is established automatically after the Ethernet module installed station has been restarted, and data transmission/reception is enabled.
Sequence programs for open processing and close processing are not required.
For details on the [Open settings] parameter, see Section 5.5, "Open Settings".

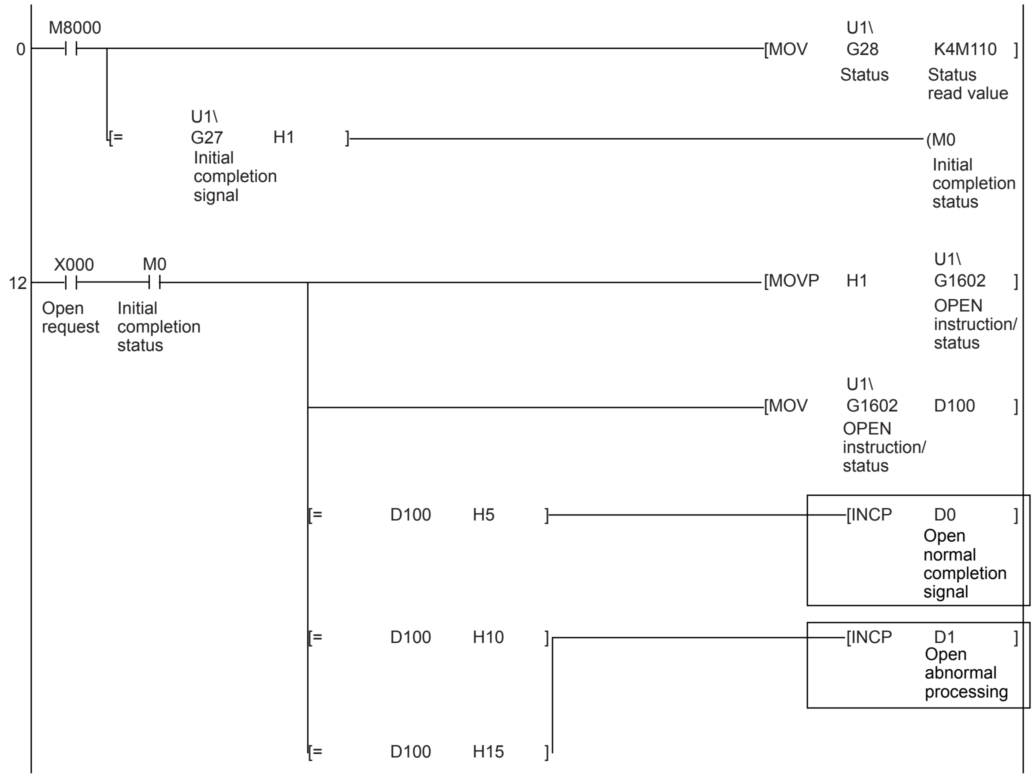
NOTE

When performing the open and close processing by writing to the buffer memory from the Ethernet module side, even if [Always wait for OPEN (Communications impossible at STOP time)] is selected at [Operational settings], it is required to use the sequence program to process all that follows the processing for the applicable connection.
If "Do not wait for OPEN (Communication impossible at STOP time)" is selected, open processing and close processing need to be done in the same manner as a connection.

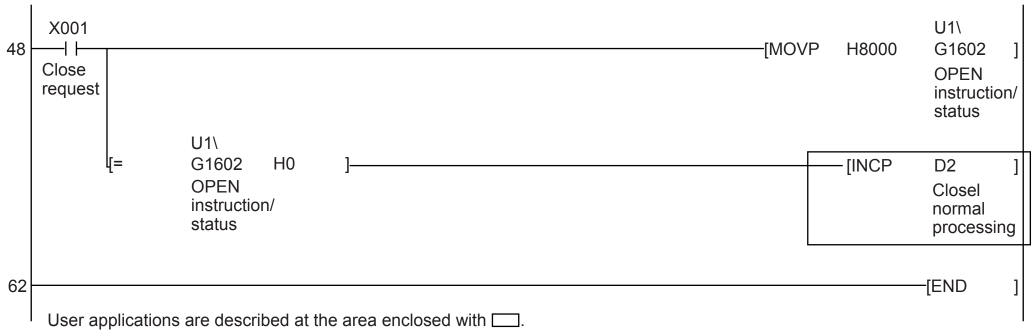
(2) When "Do not wait for OPEN" is selected in the operational settings (Communications impossible at STOP time)

In this case, the open processing and close processing in the sequence program shown in the next page are required. Data transmission and reception are enabled after the open processing is normally completed.
Perform the open processing and close processing by writing "0001H" in BFM#1602 to 1609.

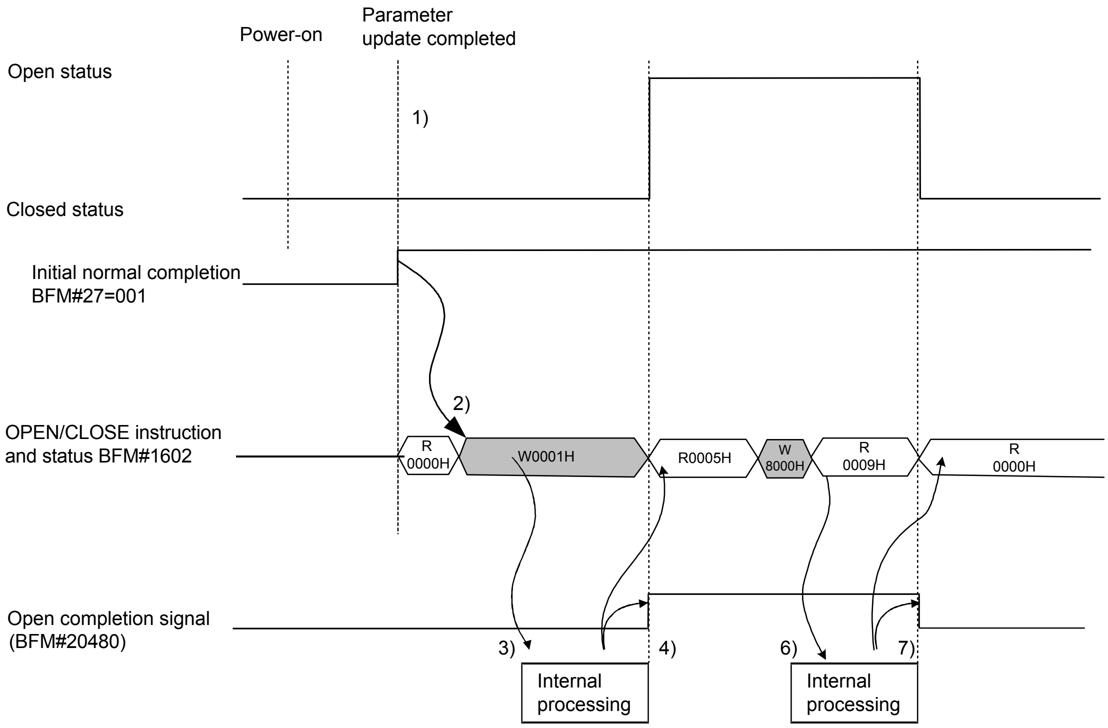
<<Open processing for connection No.1>>



<<Close processing for connection No.1>>



User applications are described at the area enclosed with □.



1) After communicating the parameter settings, confirm the normal completion of the Ethernet module initial processing.
(Initial normal completion signal (BFM#27"0001H"))

2) Start the open processing by writing 0001H to the BFM#1602.
(Open request signal (BFM#20482 b0) : ON

3) The Ethernet module executes the open processing. (Internal processing only)

4) When the open processing completes normally
- Open completion signal (BFM#20482 b0) : ON
- Reading value of BFM#1602 : 0005H

Data communication is enabled.

When the open processing completes abnormally
- Open status (BFM#1602) : Value other than 0010H or 0015H
- Open abnormal detection signal (BFM#28 b6) : ON

5) Start the close processing by writing 8000H to the BFM#1602.

6) The Ethernet module executes the close processing. (Internal processing only)

7) When the close processing completes normally
- Open/Close instruction status (BFM#1602) : 0000H

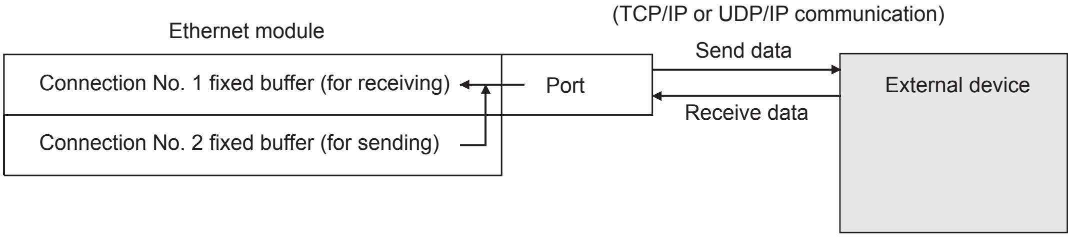
5.7 Pairing Open

The following explains communication using the paring open method via the Ethernet module.

5.7.1 Pairing open

Pairing open is an opening method for establishing a connection in which the local station and the external device use a single port for each, by pairing the receiving and sending connections through fixed buffer communication (both the procedure exists and no procedure are allowed) of the Ethernet module.
By designating the paring open method, data communication can be performed with two connections by performing open processing for only one port.
Communications using MC protocol can also be performed using these paring-opened connections.
The procedure for performing the open/close processing for pairing open is explained below.

[Example]


(TCP/IP or UDP/IP communication)
Ethernet module
Send data
Connection No. 1 fixed buffer (for receiving)
Port
External device
Receive data
Connection No. 2 fixed buffer (for sending)

POINT

(1) When setting the pairing open method, the fixed buffer of the applicable connection number (for receiving only) and the fixed buffer of the next consecutive connection number (for sending only) are paired in the order of receiving, then sending.
For the applicable connection (for receiving only), choose connection No. 1 to 7.

(2) The range of external devices that can be communicated by the pairing open method are limited to devices on the Ethernet to which the Ethernet module is connected and devices connected with the router relay function (see Section 5.3, "Router Relay Parameter").

(3) By the open/close processing of the applicable connection (for receiving only) for which the pairing open method has been set, the open/close processing of the next connection (for sending only) will automatically be performed.

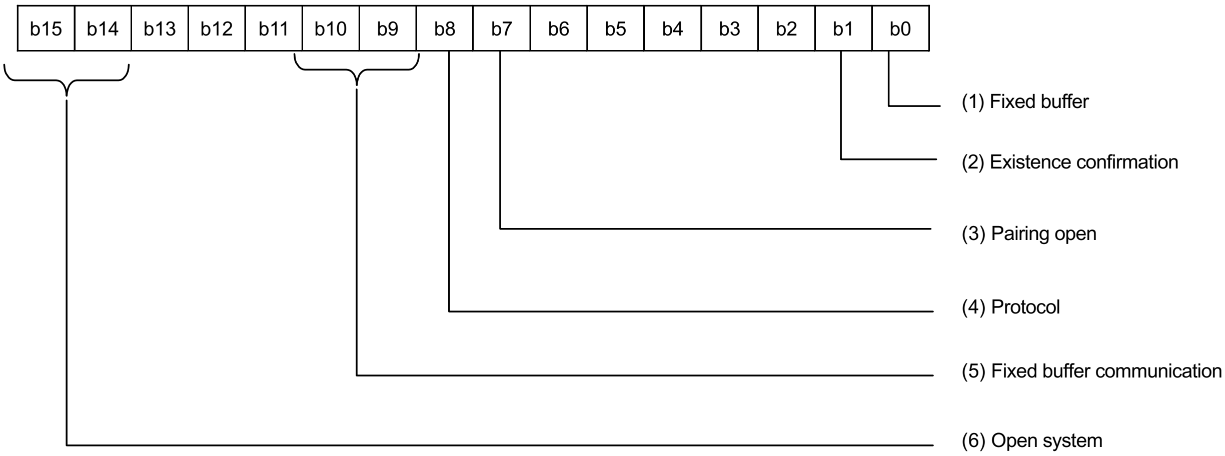
5.7.2 Example of pairing open settings

This section explains the settings in order to communicate in the pairing open method by giving an example.

- Connection No. 1 and 2 are used.

(When connection No. 1 is specified at pairing open settings, connection No. 2 is automatically used as pairing.
Therefore, connection No. 8 cannot be specified as pairing.)

[BFM#32 Connection usage setting area]



(1) Fixed buffer (BFM#32 b0)
In the open pairing, the applicable connection No. and the subsequent connection No. are paired. Set the applicable connection No. to "Receive" and the next connection number to "Send".

(2) Existence confirmation (BFM#32 b1)
If existence confirmation is going to be executed, set the receiving connection to "Confirm" and the sending connection to "No confirm".
If existence confirmation is not going to be executed, select "No confirm" for both.

(3) Pairing open (BFM#32 b7)
Set the receiving connection to "Pairs" and the sending connection to "No pairs".

(4) Protocol (BFM#32 b8)
Both "TCP/IP" and "UDP/IP" are allowed.

(5) Fixed buffer communication (BFM#32 b9, b10)
Both "Procedure exist" and "No procedure" can be selected.

(6) Open system (BFM#32 b14, b15)
All the open systems - "Active", "Unpassive", and "Fullpassive" - can be set.

(7) Local station Port No. (BFM#40, 47)

Set this for the receiving connection for receiving only. (Setting is not required for the sending connection.)

Set the port number upon consulting a network administrator.

(8) Destination IP address (BFM#41 to 42, 48 to 49)

(a) If the setting is not required

- Open system: At [Unpassive] setting

(b) If the setting is required

Set for the receiving connection only upon consulting a network administrator.

- Open system: At [Active] and [Fullpassive] setting
- Protocol: At [UDP/IP] setting

(9) Destination Port No. (BFM#43, 50)

(a) If the setting is not required

- Open system: At [Unpassive] setting

(b) If the setting is required

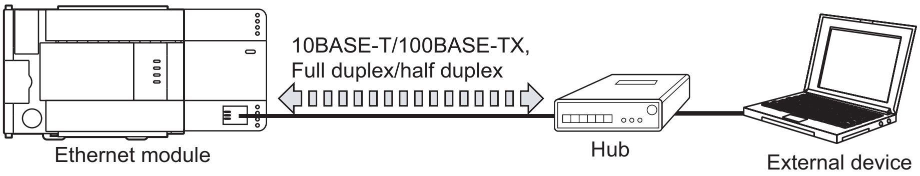
- Open system: At [Active] and [Fullpassive] setting
- Protocol: At [UDP/IP] setting

NOTE

When making settings in the FX Configurator-EN (GX Developer), select [Open Settings] from the initial screen and at the [Ethernet Open Settings] screen, set "Pairing Open” to “Yes”.

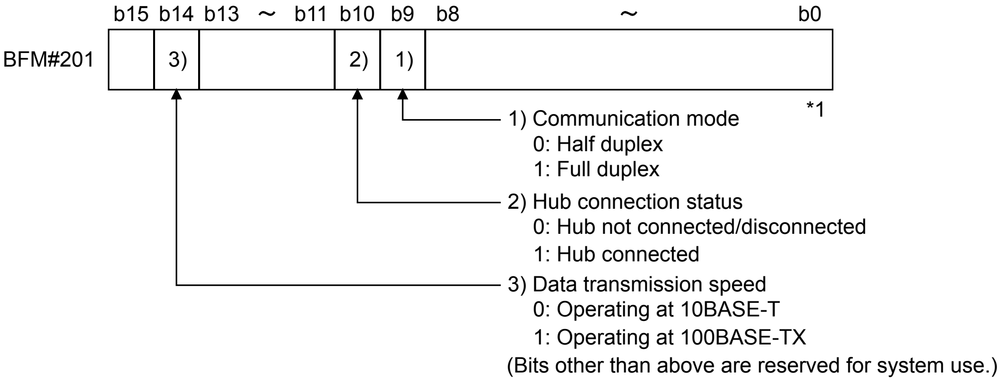
5. 8 Hub Connection Status Monitor Function

The current connection status of the Ethernet module and hub, the transmission speed, and the number of times the Ethernet module detected disconnection can be checked at the following buffer memory addresses.



(1) Hub connection status area (BFM#201)

Stores the current connection status of the FX3U-ENET and hub and the transmission speed.



1) Communication mode
0: Half duplex
1: Full duplex

2) Hub connection status
0: Hub not connected/disconnected
1: Hub connected

3) Data transmission speed
0: Operating at 10BASE-T
1: Operating at 100BASE-TX

(Bits other than above are reserved for system use.)

*1 When a switching hub is used, it may not be displayed correctly.

(2) Disconnection detection count storage area (BFM#20995)

(a) Stores the number of disconnection detection times after initial processing is completed.
Disconnection is detected in any of the following cases.

- Disconnection between Ethernet module and hub
- Cable removal from hub side connector
- Hub power-off
- Cable removal from Ethernet module side connector

(b) If an error has occurred 65536 times or more, a count stops at FFFFH (65535).
Write "0" to this area using a sequence program to clear the stored value.

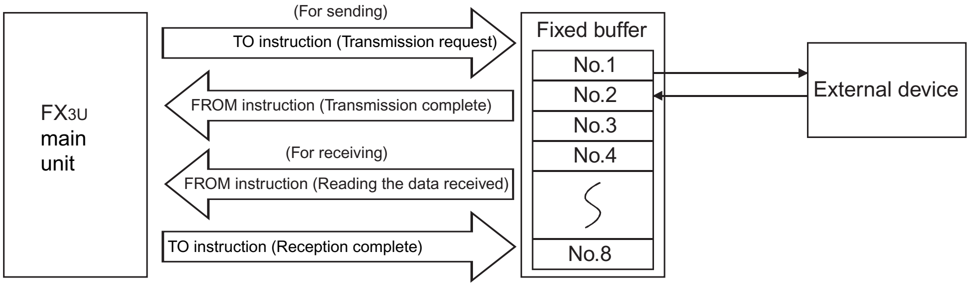
6 FIXED BUFFER COMMUNICATION (WITH THE PROCEDURE EXIST CONTROL METHOD)

This chapter explains how the PLC and external device communicate in a 1:1 mode using the fixed buffers (with the procedure exist control method) of the Ethernet module.

6.1 Control Method

The following explains how communication is performed using the fixed buffers and the procedure exist control method.
In communication using the fixed buffers, data transmission from the PLC and the external device is executed through handshaking.

(1) The data flow during communication is as follows.


FX3U main unit
(For sending)
TO instruction (Transmission request)
FROM instruction (Transmission complete)
(For receiving)
FROM instruction (Reading the data received)
TO instruction (Reception complete)
Fixed buffer
No.1
No.2
No.3
No.4
No.8
External device

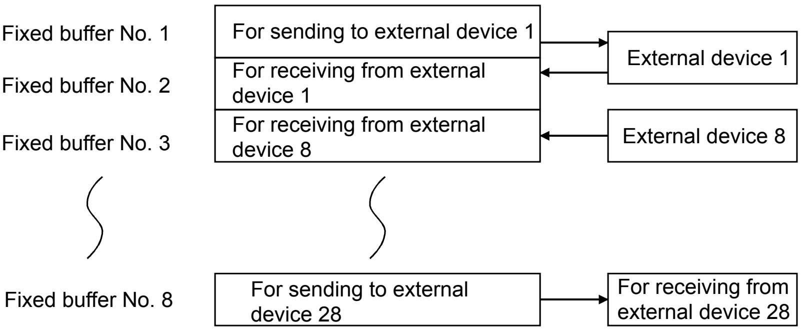
6

(2) Data can be communicated with the following external devices.
- Device on the Ethernet to which the Ethernet module is connected.
- Devices connected with the router relay function (see Section 5.3)

As shown in the diagram below, when using each fixed buffer (No. 1 to 8), the destination devices and usage conditions (for sending/receiving, procedure exist/no procedure, etc.) should be set when the connection via the Ethernet module is opened to fix the external device for each buffer.

(a) At TCP/IP communication
It is allowed to change external devices only when the open completion signal of the applicable connection is off.

(b) At UDP/IP communication
External devices can be changed regardless of the status of the applicable connection.
("Destination IP address" and "Destination Port No." in the communication address setting area can be changed. However, the "Local station Port No." cannot be changed.)
When changing external devices, do not use the "Pairing open" and "Existence confirmation" functions.


Fixed buffer No. 1
Fixed buffer No. 2
Fixed buffer No. 3
Fixed buffer No. 8
For sending to external device 1
For receiving from external device 1
For receiving from external device 8
For sending to external device 28
External device 1
External device 8
For receiving from external device 28

POINT

In communication where the procedure exist control method is selected, the data can be communicated by the following methods after the open processing is completed.

- Fixed buffer communication with the procedure exist control method (sending or receiving)
- Communication using MC protocol
  (When adjusting settings in the FX Configurator-EN (GX Developer), select [Procedure exists (MC)] in [Ethernet Settings] → [Open settings]→ [Open system].

(3) At data sending/receiving, the Ethernet module processes the following.

(a) When sending data
In response to the fixed buffer communication transmission request (write "0001H" to BFM #1610 to 1617), the Ethernet module sends the data of the fixed buffer stored in the fixed buffer area (BFM #1664 to 9855) to the the specified external device (BFM #40 to 95) (∗1)

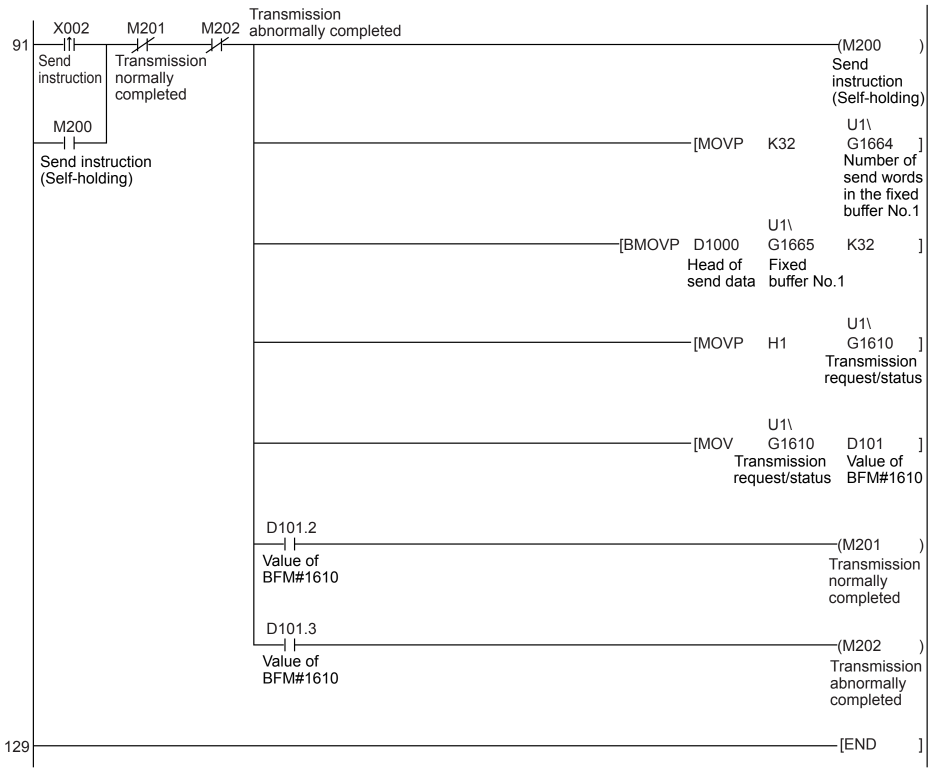
(b) When receiving data
The Ethernet module processes the received data if the data is received from an external device set in the communication setting area that corresponds to fixed buffer No. n. (∗1)
If data is received from an external device not set in the connection information area of the buffer memory, the Ethernet module ignores the received data to the PLC side.

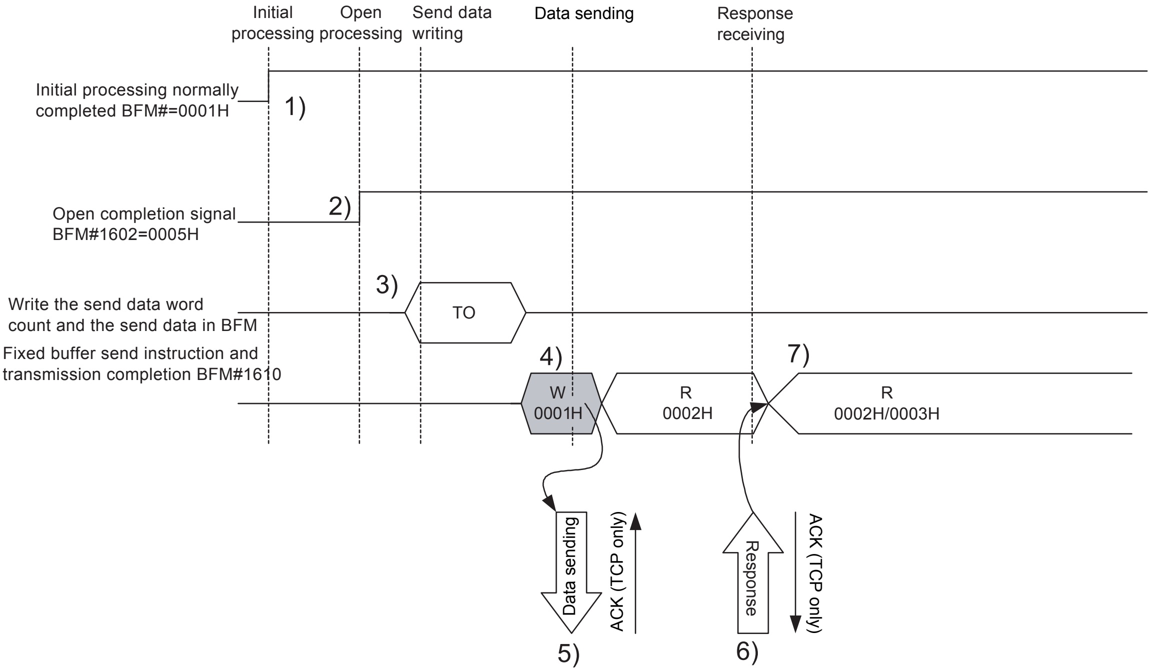
∗1 In case of TCP/IP Unpassive open, data is communicated with an external device stored in the connection information area corresponding to fixed buffer No. n.

6.2 Sending Control Method

This section explains the control method when data is sent from the Ethernet module to an external device using the fixed buffer No. 1 and the area corresponding to connection No. 1 as an example.

* <<Sending 32 words from D1000 to the external device>>


91
X002
Send instruction
M201
Transmission normally completed
M202
Transmission abnormally completed
(M200)
Send instruction (Self-holding)
M200
Send instruction (Self-holding)
[MOVP K32 U1\ G1664]
Number of send words in the fixed buffer No.1
[BMOVP D1000 U1\ G1665 K32]
Head of send data
Fixed buffer No.1
[MOVP H1 U1\ G1610]
Transmission request/status
[MOV U1\ G1610 D101]
Transmission request/status
Value of BFM#1610
D101.2
Value of BFM#1610
(M201)
Transmission normally completed
D101.3
Value of BFM#1610
(M202)
Transmission abnormally completed
129
[END]


Initial processing
Open processing
Send data writing
Data sending
Response receiving
Initial processing normally completed BFM#=0001H
1)
Open completion signal BFM#1602=0005H
2)
Write the send data word count and the send data in BFM
3)
TO
Fixed buffer send instruction and transmission completion BFM#1610
4)
W 0001H
R 0002H
7)
R 0002H/0003H
Data sending
ACK (TCP only)
5)
Response
ACK (TCP only)
6)

1) Confirm the normal completion of the initial processing.

2) Confirm the normal completion of the open processing of connection No. 1

3) Write the send data word count and the send data in the fixed buffer data area (communication address setting area: BFM#1664 to 2687) by TO instruction.

4) Write "0001$_H$" in the fixed buffer communication transmission request (BFM#1610).

5) The size of send data in the fixed buffer (No. 1) area designated by the send data length is sent to the designated external device (set in the open processing).
The value in BFM#1610 becomes "0002$_H$".

6) Upon receiving the data from the Ethernet module, the external device returns a "Response" to the Ethernet module.

7) Upon receiving the "Response" from the external device, the Ethernet module ends the data transmission.
If the "Response" is not returned within the response monitoring time values (see Section 5.2), a data send error occurs.

At normal completion
Fixed buffer transmission result (BFM#1610) : 0004$_H$

At abnormal completion
Fixed buffer transmission result (BFM#1610) : 0008$_H$
If the data transmission is abnormally completed, execute the send process again writing "0001$_H$" in BFM#1610.

POINT

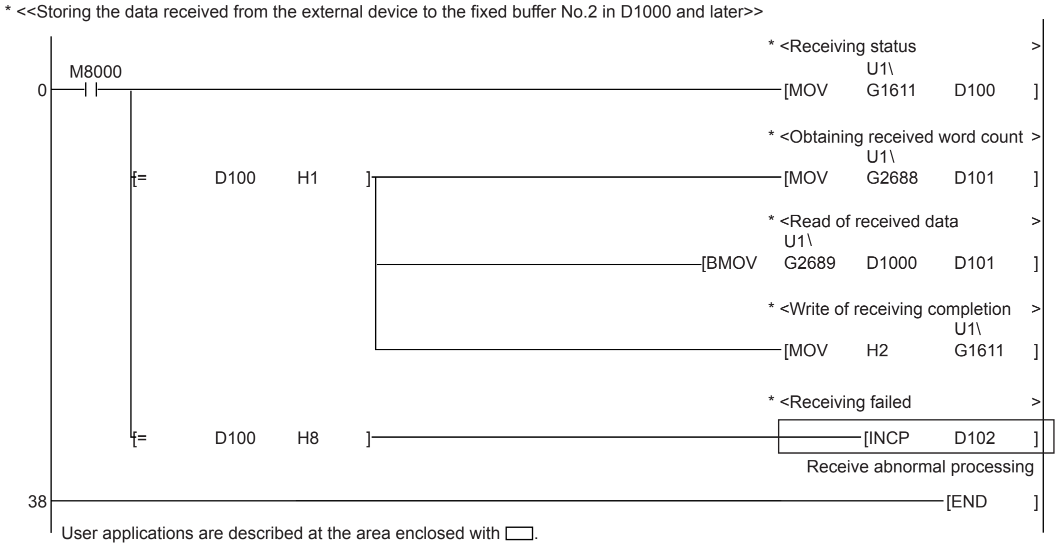
The destination setting (see Section 5.5) for a connection whose parameters are set in FX Configurator-EN (GX Developer) becomes valid when the open completion signal (BFM#20480 corresponding bit) of the Ethernet module switches from off to on.
The setting also becomes valid when the each value in BFM#1602 to 1610 is "0005$_H$" for Connection No.1 to 8.

6.3 Receiving Control Method

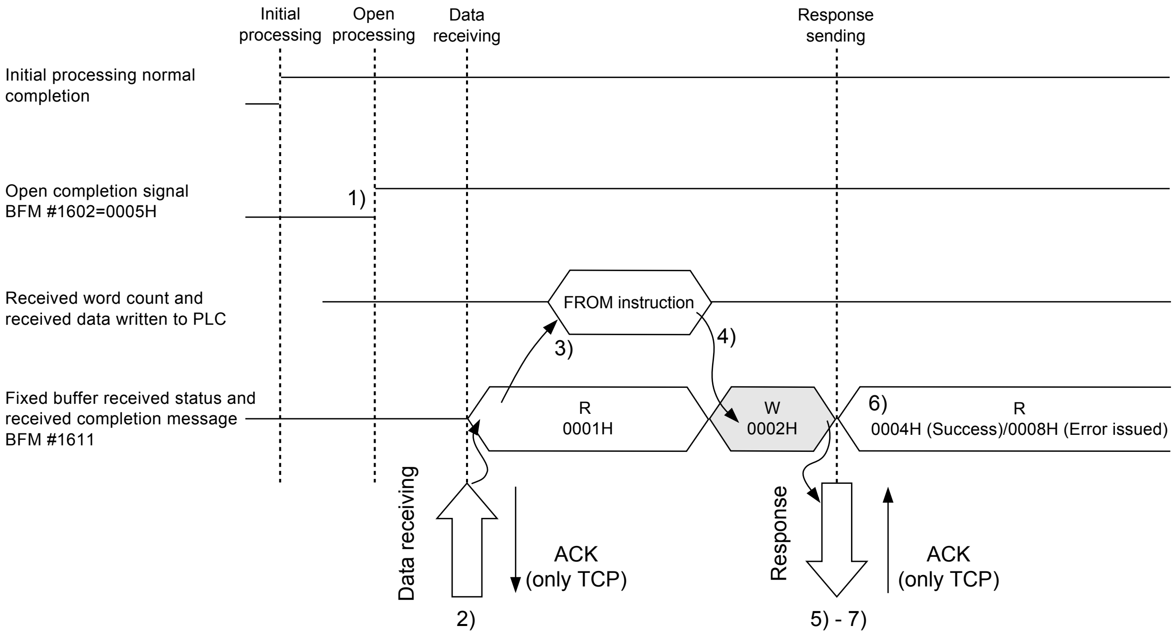
This section explains the control method when the Ethernet module receives data from an external device.

6.3.1 Receive processing with the main program

This section explains the receiving process to be performed with the main program, using an example in which the fixed buffer No. 2 and the area corresponding to connection No. 2.



```
* <<Storing the data received from the external device to the fixed buffer No.2 in D1000 and later>>
                                                            * <Receiving status             >
      M8000                                                            U1\
 0 ---| |------------------------------------------------------[MOV    G1611     D100      ]

                                                            * <Obtaining received word count >
                                                                       U1\
        [=     D100     H1     ]-------------------------------[MOV    G2688     D101      ]

                                                            * <Read of received data        >
                                                              U1\
                                         ---------------[BMOV  G2689    D1000     D101      ]

                                                            * <Write of receiving completion >
                                                                                 U1\
                                       ------------------------[MOV    H2        G1611     ]

                                                            * <Receiving failed             >
        [=     D100     H8     ]---------------------------------------[INCP     D102      ]
                                                                 Receive abnormal processing
38 --------------------------------------------------------------------[END               ]
    User applications are described at the area enclosed with □.
```


Initial processing
Open processing
Data receiving
Response sending
Initial processing normal completion
Open completion signal BFM #1602=0005H
1)
Received word count and received data written to PLC
FROM instruction
3)
4)
Fixed buffer received status and received completion message BFM #1611
R
0001H
W
0002H
6)
R
0004H (Success)/0008H (Error issued)
Data receiving
ACK (only TCP)
2)
Response
ACK (only TCP)
5) - 7)

1) Confirm the normal completion of the open processing of connection No. 2.

2) Upon receiving data from the designated external device (set in the open processing), the Ethernet module processes the following.
   - Stores the received data to the fixed buffer (No. 2) area.
     Receive data length : The head address area of the target fixed address
     Receive data : Area beginning from the head address of the target fixed buffer + 1
   - Fixed buffer receive status signal (BFM#20485 b1) : ON
   - Fixed buffer communication receive result (BFM#1611) : 0001H

3) Execute the FROM instruction to read the received word length and received data to the PLC.

4) Execute the TO instruction to write "0002H" in the fixed buffer communication receive result (BFM#1611).

5) Return "Response" to communication destination.

6) The receive processing is performed.

   At normal completion
   - Fixed buffer transmission results (BFM#1611) : 0004H

   At abnormal completion
   - Fixed buffer transmission results (BFM#1611) : 0008H

POINT

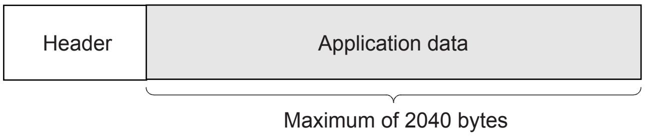
(1) The destination setting (see Section 5.5) for a connection whose parameters are set with FX Configurator-EN (GX Developer) becomes valid when the open completion signal (BFM#20480… corresponding bit) of the Ethernet module switches from off to on.

(2) Execute data receiving process when the corresponding connection’s bit in the fixed buffer receive status signal storage area (BFM#20485) of the buffer memory switches from off to on or when the value "0001H" is stored in the fixed buffer communication receive results (BFM#1610 to #1617).

(3) At abnormal data receiving, each buffer memory status is shown in the following.
   (For connection No. 2)
   - The fixed buffer receive completion signal (BFM#20485 b1) remains off.
   - The fixed buffer communication receive result (BFM#1611) remains "0008H".
   - Data is not stored in the fixed buffer (No.2) area.

6.4 Data Format

When communicating between the Ethernet module and an external device, the data format explained below is used.
The communication data consists of a "header" and "application data" as follows:



6.4.1 Header

The header for TCP/IP or UDP/IP is used. In case of the Ethernet module, the Ethernet module adds and deletes the header. Thus, the user does not need to set it.
(Details of the size of the header section)

1) In case of TCP/IP

Ethernet 14 bytes	IP 20 bytes	TCP 20 bytes

2) In case of UDP/IP

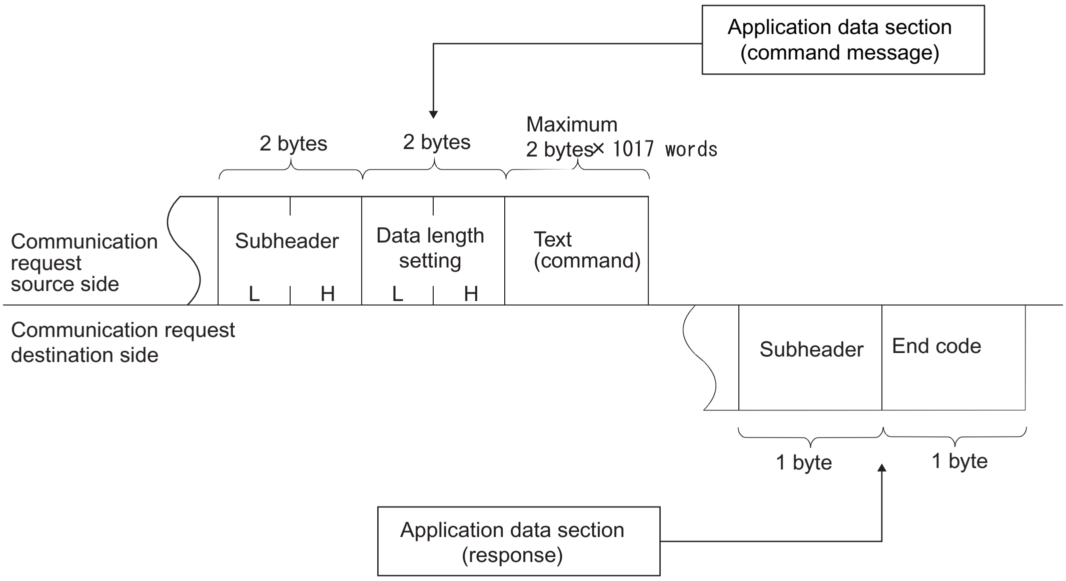
Ethernet (14 bytes)	IP (20 bytes)	UDP (8 bytes)

6.4.2 Application data

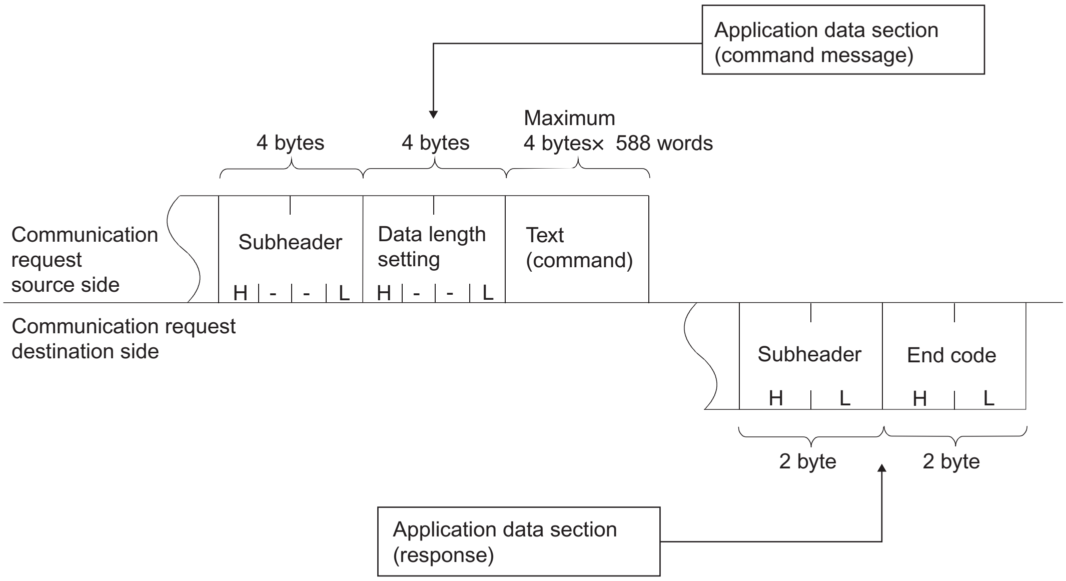
As shown below, the data code in the application data can be expressed in either binary or ASCII code. Switching between binary code and ASCII code is performed via the communication data code setting (BFM #24 b1 OFF: binary, ON: ASCII code). (When setting using the FX Configurator-EN (GX Developer), select [Operational Settings] from the initial screen, and adjust the settings from the [Operational Settings] screen.)
For more details, see Section 4.6, "Operational Settings".

(1) Format

(a) Communication using binary code


Application data section (command message)
2 bytes
2 bytes
Maximum 2 bytes× 1017 words
Communication request source side
Subheader
L
H
Data length setting
L
H
Text (command)
Communication request destination side
Subheader
End code
1 byte
1 byte
Application data section (response)

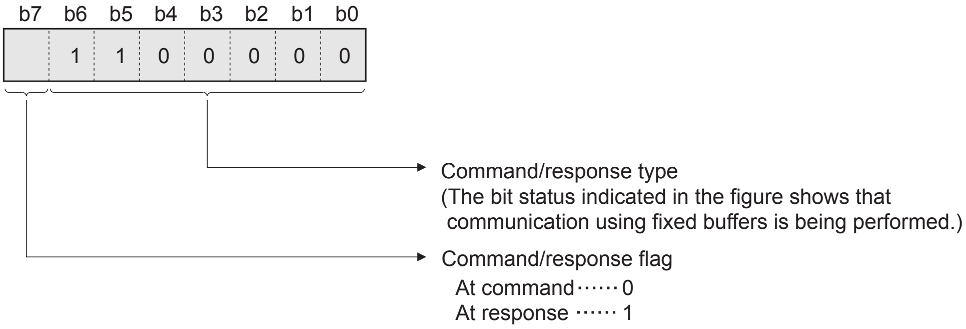
(b) Communication using ASCII code

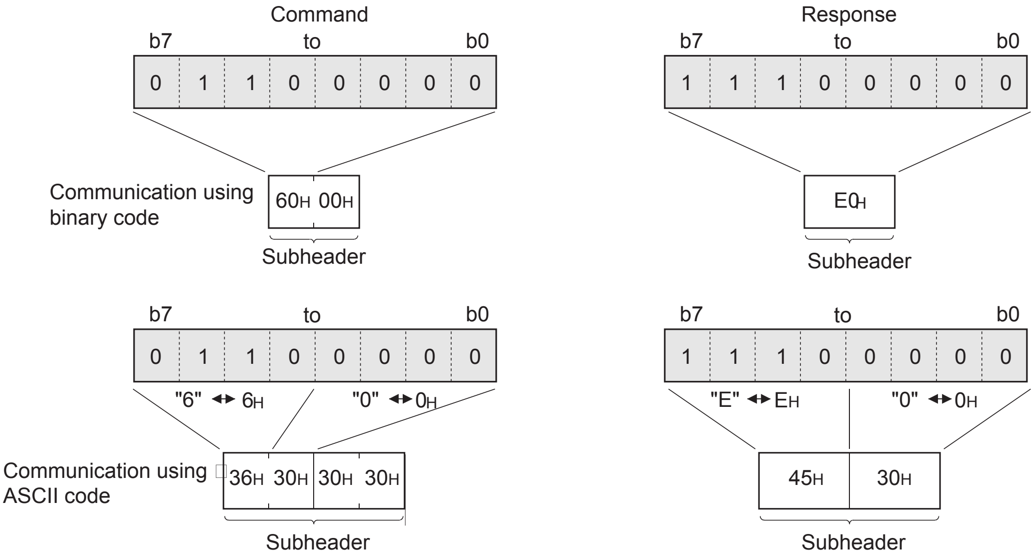

Application data section (command message)
4 bytes
4 bytes
Maximum 4 bytes× 588 words
Communication request source side
Subheader
H - - L
Data length setting
H - - L
Text (command)
Communication request destination side
Subheader
H L
End code
H L
2 byte
2 byte
Application data section (response)

(2) Subheader

The format of the subheader is as shown below.

The user does not need to set the subheader when using the Ethernet module since the Ethernet module adds and deletes it.





(3) Data length setting

This value indicates the data size of the text (command) section.

POINT

The data length can be designated in the following range:

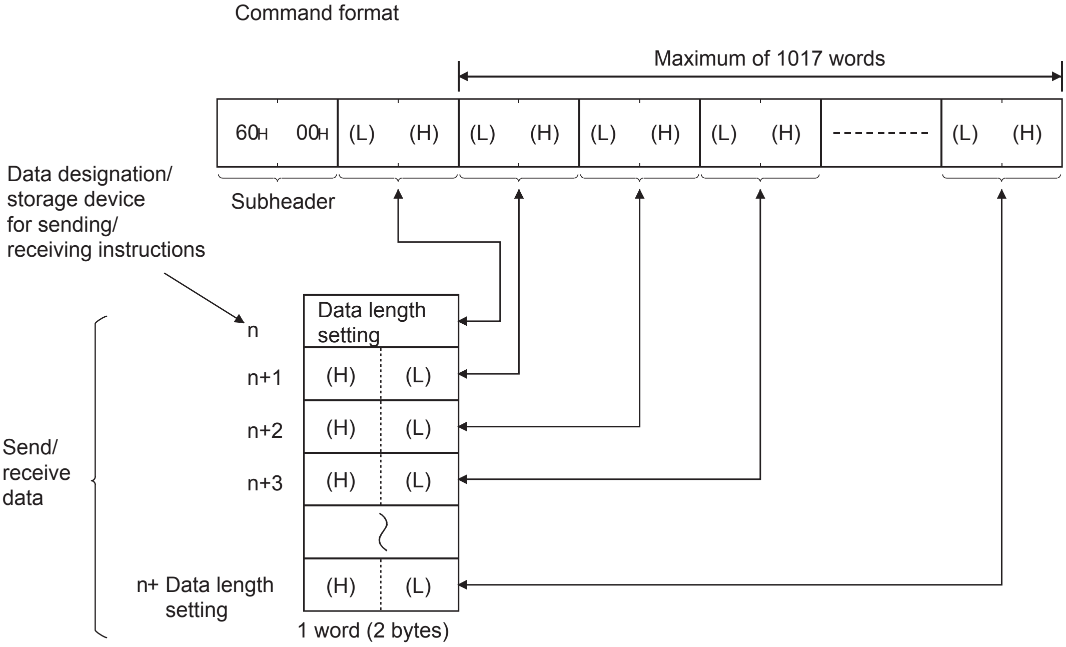
- Communication using binary code: Maximum of 1017 words
- Communication using ASCII code: Maximum of 508 words (*1)

＊1 Since data is sent/received as ASCII data, the communication data size is approximately half of the data size when using binary code.

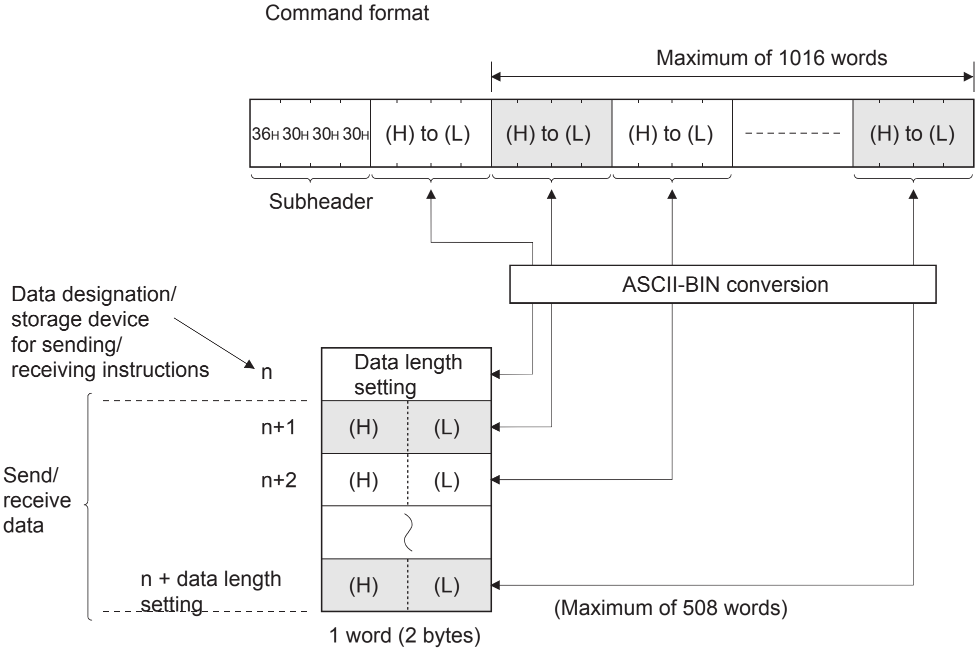
(4) Text (command)

The format of the command/response when communicating using fixed buffers is configured as follows.

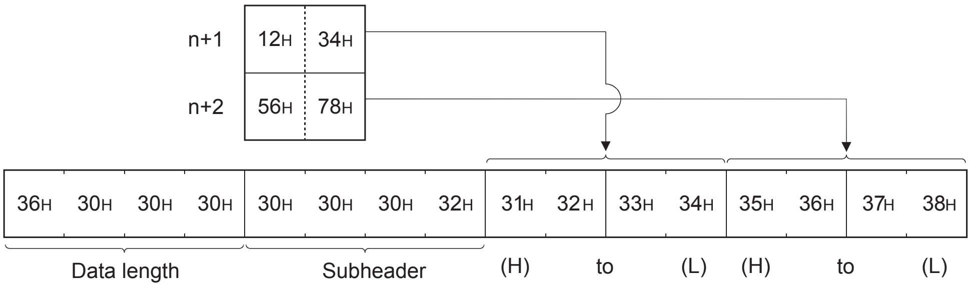
(a) Communication using binary code



(b) Communication using ASCII code



(Example)



(5) End codes

For more details on the end codes added to a response when communicating using fixed buffers, see Section 11.4.1.

End codes are stored in the communication status storage area of the buffer memory.

6.5 Programming

This section explains the programming method when the Ethernet module communicates with an external device using fixed buffers and the procedure exist control method.

6.5.1 Precautions when creating programs

(1) In order to communicate using the fixed buffer, the initial processing and the connection open processing must be completed.

(2) The contents of the parameter settings have been loaded into the Ethernet module when the Ethernet module open completion signal (BFM#20480 … corresponding bit) switches from off to on.

(3) The data length (word count) is established using the instruction for procedure exist communication control method.
If the send data length exceeds this range at data sending, a communication error occurs and the data is not sent.

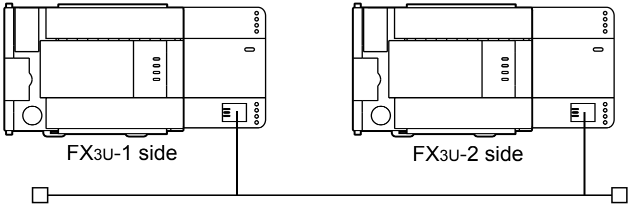
(4) Reading or writing to the buffer memory using fixed buffer communication.
- Data sending
  Write the send data in the buffer memory (BFM#1664 to #9855).
  Write "0001H" in BFM#1610 to #1617 to send the data.
- Data receiving
  Check the data received (BFM#1610 to #1617: "0001H") and read the received data to the PLC (BFM#1664 to #9855).
  After reading out, write "0002H" in BFM#1610 to #1617 to send the response.

(5) The following should be observed when using a connection opened by UDP.
External devices can be switched by modifying the setting values in the communication address setting area of the communication parameter setting area before sending/receiving data. Thus, data can be sent to multiple external devices sequentially. When sending/receiving, make sure to switch between external devices properly so that no communication problems occur.

(6) For data (command) transmission, the next data (command) should be after the data commnication is complete (such as after receiving a response) for the transmission of the previous data (command).

6.5.2 Fixed buffer communication program example (with the procedure exist control method)

This section explains the programming method in order to communicate data (procedure exist control method) with an external device using the fixed buffers.



(1) Execution environment for the program example

(a) Send program (FX3U-1 station side)

1) Connection No. 1 is used for fixed buffer sending.
2) The communication parameter settings are assumed to have been set as described in Section 5.6.1, "Active open processing/close processing".
3) Fixed buffer No. 1 send data
   : Stored in D1000 to D1031
4) Fixed buffer No. 1 send instruction complete device
   : M201
5) Fixed buffer No. 1 send instruction abnormal complete device
   : M202
6) Fixed buffer No. 1 send instruction complete status
   : D101

(b) Receive program (FX3U-2 station side)

1) Connection No. 1 is used to process the fixed buffer receiving in the main program.
2) Connection No. 2 is used to process the fixed buffer receiving in the interrupt program.
3) The Ethernet module is mounted as the second special module.
4) The unit number specified at the sequence program and FX Configurator-EN (GX Developer) is 1.
5) Make Operational Settings in the following manner.

Communication data code	: Binary code (BFM#24 b1 : OFF)
Initial timing	: Always wait for OPEN (Communications possible at STOP time) (BFM#24 b8 : OFF)
IP address	: 10.97.85.223 (0A.61.55.DFH) (BFM#106 to 107)

6) Adjust Open settings in the following manner.

Protocol	: TCP (BFM#32 b8 : OFF)
Open system	: Unpassive (BFM#32 b14, b15 : 10H)
Fixed buffer	: Receive (BFM#32 b0 : ON)
Fixed buffer communication procedure	: Procedure exist (BFM#32 b9 : OFF)
Pairing open	: Disable (BFM#32 b7 : OFF)
Existence confirmation	: No confirm (BFM#32 b1 : OFF)
Connection No.1 Host station Port No.	: 8192 (BFM#40) (For use in main program)
Connection No.2 Host station Port No.	12288 (BFM#47) (For use in embedded : program)

7) Fixed buffer No. 1 receive data
   : Stored in D500 to D503

8) Fixed buffer No. 2 receive data
: Stored in D700 to D703

9) Fixed buffer No. 1 receive instruction complete device
: M500

10) Fixed buffer No. 1 receive instruction abnormal complete device
: M501

11) Fixed buffer No. 1 send instruction complete status
: D5001

12) Fixed buffer No. 1 receive status signal
: M40

POINT

Make sure to have sufficient device memory according to the maximum data length sent in order to prevent device areas used for other purposes from being overwritten by the received data.

(2) Outline of the program example

(a) Send program (Ethernet module-1st station side)

1) After setting each parameter with FX Configurator-EN (GX Developer) or a sequence program and writing to the Ethernet module, select re-initial, power ON the module again, or confirm that the initial processing is completed via buffer memory operations.

2) Perform open processing (Active open) of connection No. 1. (＊1)

3) Communicate with the PLC using fixed buffer communication (procedure exist sending).

4) After sending is complete, perform close processing of connection No. 1. (＊1)

＊1 Use the program example described in Section 5.6.1, "Active open processing/close processing" for the sequence program that executes the open processing/close processing.

(b) Receive program (Ethernet module -2nd station side)

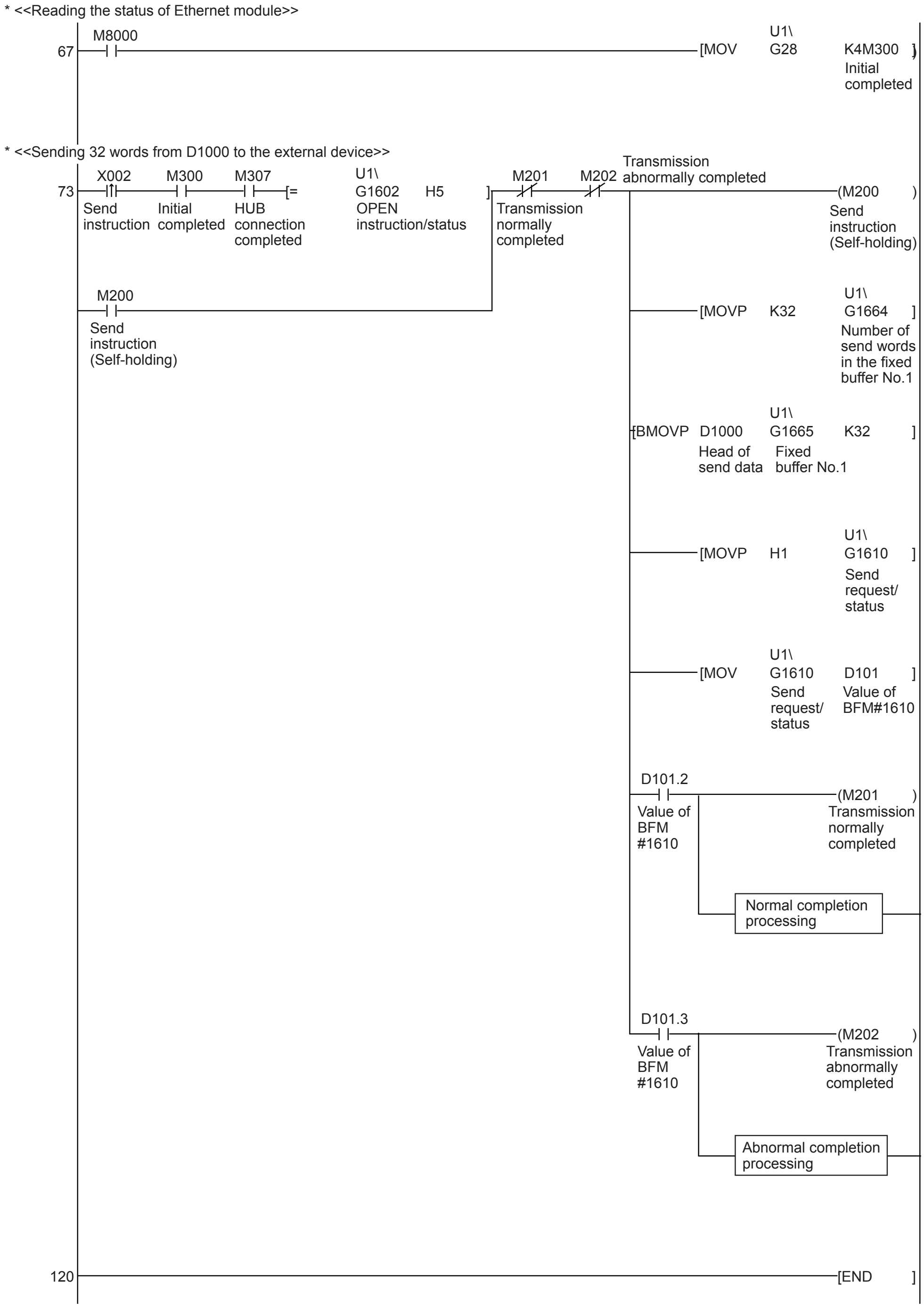
1) After setting each parameter with FX Configurator-EN (GX Developer) or a sequence program and writing to the Ethernet module, select re-initial, power ON the module again, or confirm that the initial processing is completed via buffer memory operations.
If the initial processing is normally completed, connection No. 1 and 2 wait for an Active open request from the external device.

2) Transfer data from the external device using fixed buffer communication (procedure exist sending).

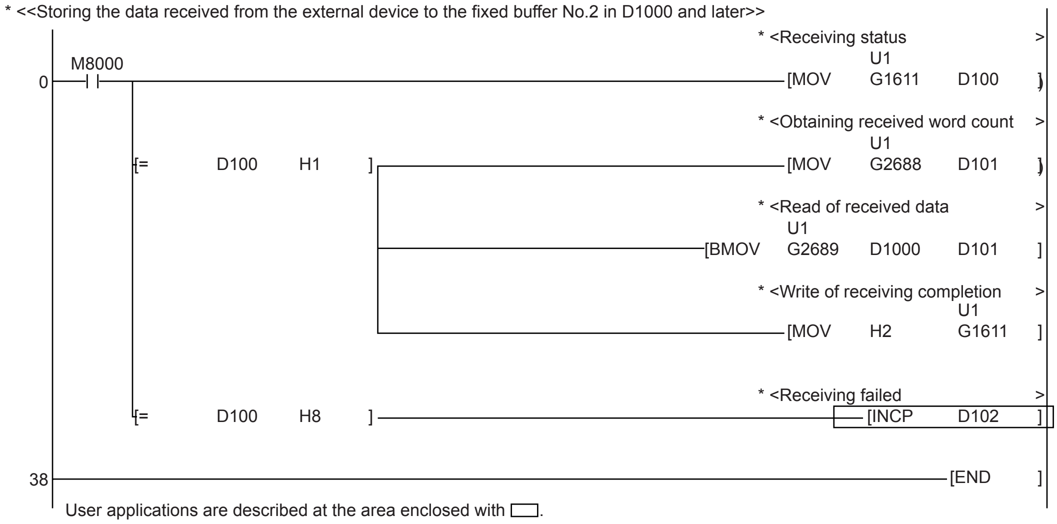
3) The data received by the corresponding fixed buffer data area in the Ethernet module is read to the PLC.

(Send program)

Open processing program (See Section 5.6.1)


* <<Reading the status of Ethernet module>>
67
M8000
[MOV U1\ G28 K4M300]
Initial completed
* <<Sending 32 words from D1000 to the external device>>
73
X002
Send instruction
M300
Initial completed
M307
HUB connection completed
[= U1\ G1602 H5]
OPEN instruction/status
M201
Transmission normally completed
M202
Transmission abnormally completed
(M200)
Send instruction (Self-holding)
M200
Send instruction (Self-holding)
[MOVP K32 U1\ G1664]
Number of send words in the fixed buffer No.1
[BMOVP D1000 U1\ G1665 K32]
Head of send data
Fixed buffer No.1
[MOVP H1 U1\ G1610]
Send request/ status
[MOV U1\ G1610 D101]
Send request/ status
Value of BFM#1610
D101.2
Value of BFM #1610
(M201)
Transmission normally completed
Normal completion processing
D101.3
Value of BFM #1610
(M202)
Transmission abnormally completed
Abnormal completion processing
120
[END]

(Receive program)



```
* <<Storing the data received from the external device to the fixed buffer No.2 in D1000 and later>>
                                                              * <Receiving status              >
                                                                                U1
   M8000
0 ─┤├─┬───────────────────────────────────────────────────[MOV    G1611    D100    ]
      │
      │                                                       * <Obtaining received word count >
      │                                                                         U1
      ├[=    D100    H1    ]─┬────────────────────────────[MOV    G2688    D101    ]
      │                      │
      │                      │                                * <Read of received data         >
      │                      │                                            U1
      │                      ├───────────────────────[BMOV    G2689    D1000    D101    ]
      │                      │
      │                      │                                * <Write of receiving completion >
      │                      │                                                           U1
      │                      └────────────────────────────[MOV    H2       G1611   ]
      │
      │                                                       * <Receiving failed              >
      └[=    D100    H8    ]──────────────────────────────────────[INCP    D102    ]

38 ──────────────────────────────────────────────────────────────────────[END     ]
```

User applications are described at the area enclosed with □.

7 FIXED BUFFER COMMUNICATION (WITH THE NO PROCEDURE CONTROL METHOD)

This chapter explains how the PLC and external device communicate using the fixed buffers (with the no procedure control method) via the Ethernet module.

POINT

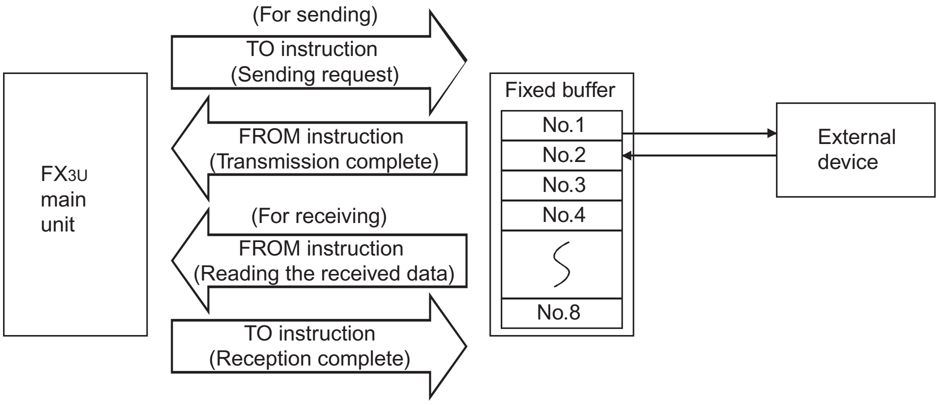
The following points describe the difference from the "Procedure exist" using fixed buffer communication:

1) It is possible to send and receive data which match the message format of the external device.
At data sending, subheader, data length, etc. are not included in the application data field of a message; only the data in the fixed buffer is sent.
Upon data reception, all the data in the message excluding the header is stored in the fixed buffer.
2) A response to data receiving is not sent.
3) Communication is performed using binary code regardless of the communication data code settings (see Section 4.6, "Operational Settings").
4) The maximum application data area is 2046 bytes per communication.
5) The applicable connection is dedicated to the no procedure fixed buffer communication.
As with the procedure fixed buffer communication, and communication using MC protocol cannot be performed at the same time as the no procedure fixed buffer communication.

7.1 Control Method

The following explains how communication is performed using the fixed buffers and the no procedure control method.
In communication using the fixed buffers, data transmission from the PLC and the external device is executed using the no procedure control method.

(1) The data flow during communication is as follows.



(2) Data can be communicated with the following external devices.
- Device on the Ethernet to which the Ethernet module is connected.
- Devices connected with the router relay function (see Section 5.3)

As shown in the diagram below, when using each fixed buffer (No. 1 to 8), the destination devices and usage conditions (for sending/receiving, procedure exist/no procedure, etc.) should be set when the connection via the Ethernet module is opened to fix the external device for each buffer.

(a) At TCP/IP communication

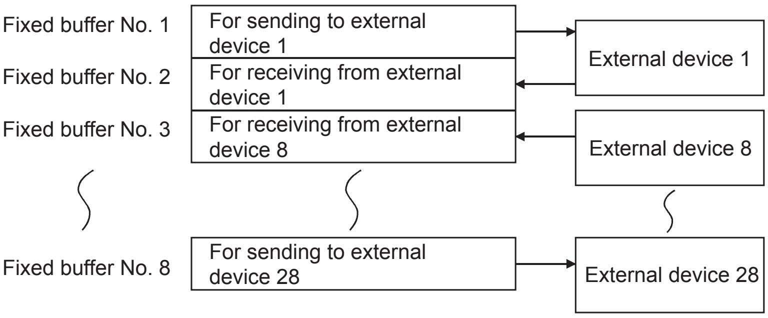
It is allowed to change external devices only when the open completion signal of the applicable connection is off.

(b) At UDP/IP communication

External devices can be changed regardless of the status of the applicable connection.

("Destination IP address" and "Destination Port No." in the communication address setting area can be changed. However, "Local station Port No." cannot be changed.)

When changing external devices, do not use the "Pairing open" and "Existence confirmation" functions.


Fixed buffer No. 1
For sending to external device 1
Fixed buffer No. 2
For receiving from external device 1
Fixed buffer No. 3
For receiving from external device 8
Fixed buffer No. 8
For sending to external device 28
External device 1
External device 8
External device 28

POINT

The connections for which no procedure is selected are dedicated to the fixed buffer sending or receiving after the completion of open processing.

(3) At data sending/receiving, the Ethernet module processes the following.

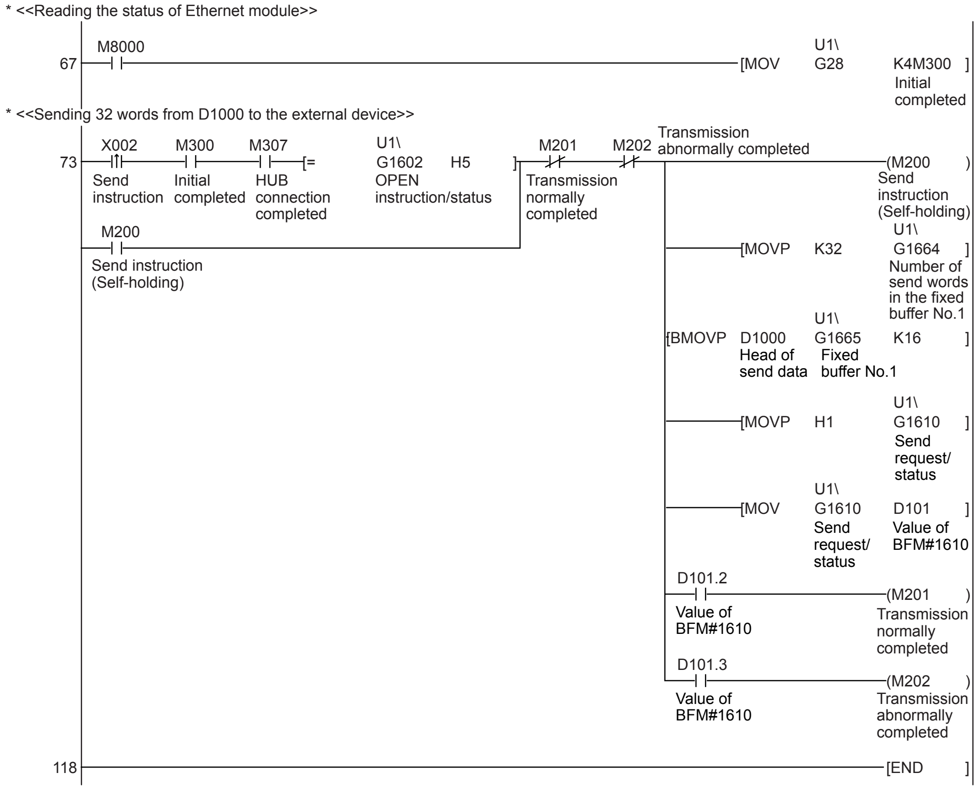
1) When sending data
The fixed buffer communication transmission request (writing "0001H " in BFM#1610 to #1617) allows the fixed buffer data stored in the fixed buffer data area (BFM#1664 to #9855) to be sent to the specified external device (BFM#40 to #95).

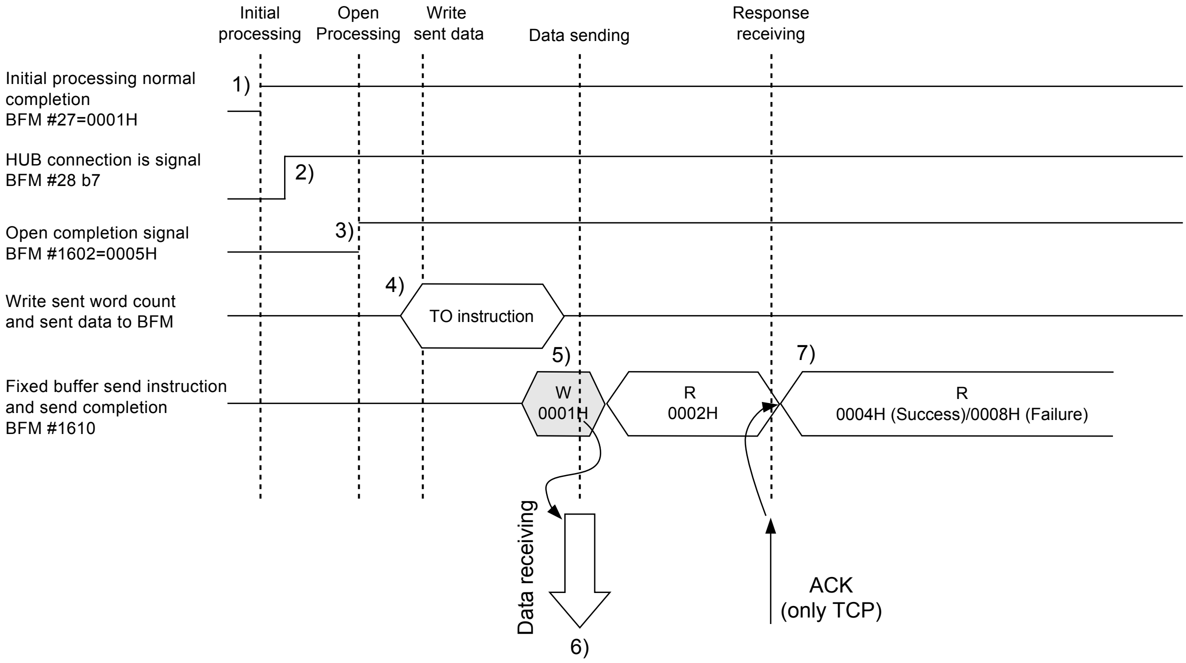
2) When receiving data
The Ethernet module processes the received data if it is received from an external device set in the communication setting area that corresponds to fixed buffer No. n. (*1)
Also, when the Ethernet module stores the received data in the corresponding fixed buffer in the receive processing, it updates the destination IP address and destination port No. in the corresponding fixed buffer connection information area (BFM#120 to #199)
If data is received from an external device not set in the connection information area of the buffer memory, the Ethernet module ignores the received data.

*1 In case of TCP/IP Unpassive open, data is communicated with an external device stored in the connection information area of the buffer memory.

7.2 Sending Control Method

This section explains the control method when data is sent from the Ethernet module to an external device using fixed buffer No. 1 and the area corresponding to connection No. 1 as an example.




Initial processing
Open Processing
Write sent data
Data sending
Response receiving
Initial processing normal completion BFM #27=0001H
HUB connection is signal BFM #28 b7
Open completion signal BFM #1602=0005H
Write sent word count and sent data to BFM
Fixed buffer send instruction and send completion BFM #1610
1)
2)
3)
4)
TO instruction
5)
W 0001H
R 0002H
7)
R 0004H (Success)/0008H (Failure)
Data receiving
6)
ACK (only TCP)

1) Confirm normal completion of the initial processing.

2) Confirm that the Ethernet module is connected to the hub. (BFM#28 b7:ON)

3) Confirm the normal completion of the open processing for connection No. 1.

4) Write the word count for send data and the send data to the fixed buffer data area (BFM#1664 to #2687) by TO instruction.

5) Write "0001H" in the fixed buffer transmission request (BFM#1610).

6) Only the size of the send data in the fixed buffer (No. 1) designated by the send data length is sent to the designated external device (The value of BFM#1610 becomes "0002H ".).

7) The Ethernet module terminates the data transmission.

At normal completion

- Fixed buffer transmission results (BFM#1610) : 0004H

At abnormal completion :

- Fixed buffer transmission results (BFM#1610) : 0008H

If the data transmission is abnormally completed, rewrite "0001H" in BFM#1610 and execute the send processing again.

POINT

The following precaution should be observed when communicating using UDP/IP:

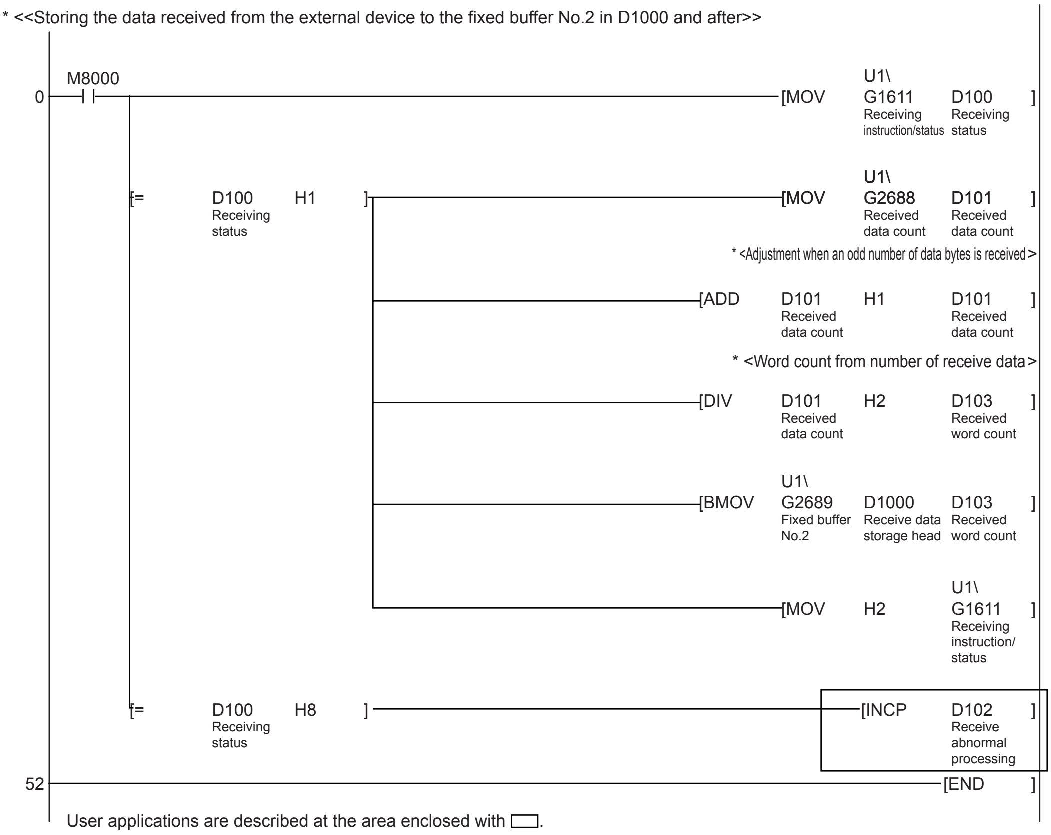
- When the Ethernet module’s internal processing is normally completed, data send processing ends even if the communication line between the PLC and an external device is disconnected because of cable disconnection, etc. It is recommended to send/receive data using a user defined communication procedure.

7.3 Receiving Control Method

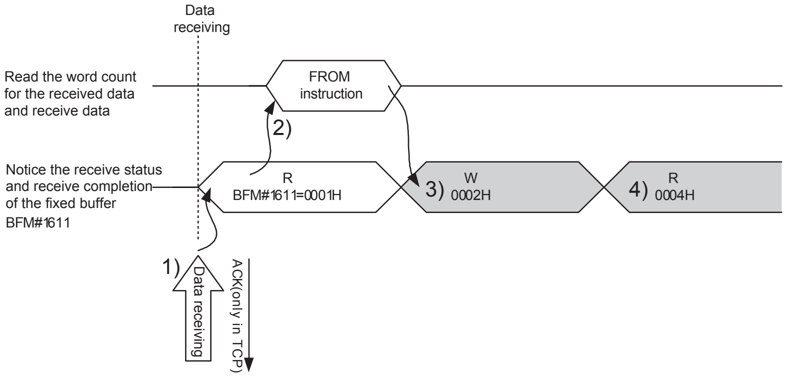
This section explains the control method when the Ethernet module receives data from an external device.

7.3.1 Receive processing with the main program

This section explains the receiving process to be performed with the main program, using an example in which the fixed buffer No. 2 and the area corresponding to connection No. 2.



User applications are described at the area enclosed with □.


Data receiving
Read the word count for the received data and receive data
FROM instruction
2)
Notice the receive status and receive completion of the fixed buffer BFM#1611
R BFM#1611=0001H
3) W 0002H
4) R 0004H
1)
Data receiving
ACK(only in TCP)

1) Upon receiving data from the designated external device (set in the open processing), the Ethernet module processes the following.
   - Stores the received data to the fixed buffer (No. 2) area.
     (Area beginning from the head address of the target fixed buffer + 1)
   - Stores the data length to the head address area of the target fixed address (∗1)
   - Fixed buffer communication receive results (BFM#1611) : 0001H
   - Fixed buffer receive status signal (BFM#20485 b1) : ON

   ∗1 The received data length is expressed by a byte count.
   When an odd number of data bytes is received, the last byte of received data is stored at the lower byte of the last data storage area. (The higher byte becomes a non-constant value.)

2) Execute the FROM instruction to read out the receive data length and receive data to the PLC.

3) Execute the TO instruction to write "0002H" in the fixed buffer communication receive results (BFM#1611).

4) Result of the receiving processing is indicated.

POINT

(1) The destination setting (see Section 5.5) for a connection whose parameters are set in FX Configurator-EN (GX Developer) becomes valid when the open completion signal (BFM#20480 … corresponding bit) of the Ethernet module switches from off to on.

(2) When data is received and the data in the buffer memory's fixed buffer is set, BFM #1610 to 1617 is set to "0001H".

(3) At abnormal data receiving, each buffer memory status becomes as follows.
   - Fixed buffer receive completion signal (BFM#20485 b1) remains off.
   - Fixed buffer communication receive results (BFM#1611): 0008H
   - Data is not stored in the fixed buffer (No.2) area.

7.4 Data Format

When communicating between the Ethernet module and an external device, the data format explained below is used.
The communication data consists of a "header" and "application data" as shown below.

Header	Application data

(1) Header

The header for TCP/IP or UDP/IP is used. In case of the Ethernet module, the Ethernet module adds and deletes the header. Thus, the user does not need to set it.
(Details of the size of the header section)

1) In case of TCP/IP

Ethernet 14 bytes	IP 20 bytes	TCP 20 bytes

2) In case of UDP/IP

Ethernet 14 bytes	IP 20 bytes	UDP 8 bytes

(2) Application Data

The data code in the application data is expressed in binary code.
Communication is performed using binary code, regardless of the set communication data (see Section 4.6).

Text (command)

Maximum of 2064 bytes

NOTE

The subheader and data length that are added for communications using the fixed buffers in the procedure exist control method are not present for communications in the no procedure control method. All data is treated as valid text.

7.5 Programming

This section explains the programming method when the Ethernet module communicates with an external device using fixed buffers and the no-procedure control method.

7.5.1 Precautions when creating programs

(1) In order to communicate using the fixed buffers, the initial processing and the connection open processing must be completed.

(2) The contents of the parameter settings have been loaded into the Ethernet module when the Ethernet module open completion signal (BFM#20480... corresponding bit) switches from off to on.

(3) The data length (byte count) is established using the instruction for no procedure communication.
If the send data length exceeds this range at data sending, a communication error occurs and the data is not be sent.

(4) Reading or writing to the buffer memory using fixed buffer communication.
- When sending data
  Write the send data to the buffer memory (BFM#1664 to #9855).
  Write "0001H" in BFM#1610 to #1617 before sending the data.
- When receiving data
  Check that data has been received (BFM#1610 to 1617: "0001H") and read out the received data (BFM#1664 to #9855) to the PLC.

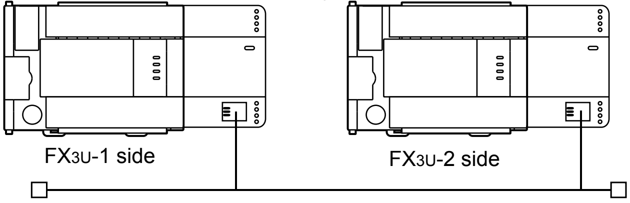
(5) The following should be observed when using a connection opened by UDP.
- External devices can be switched by modifying the setting values in the communication address setting area of the communication parameter setting area before sending/receiving data. Thus, data can be sent to multiple external devices sequentially. When sending/receiving, make sure to switch between external devices properly so that no communication problems occur.

(6) The connections for which no procedure is selected are dedicated to the no procedure fixed buffer sending/receiving. Thus, communications using fixed buffers with the procedure exist control method, random access buffers, and the MC protocol cannot be performed at the same time with communication using fixed buffers with the no procedure control method.

(7) Message data length is not included in a packet when communicating with the no procedure.
The Ethernet module stores the size of the received message (packet) in the receive data length storage area and turns on the fixed buffer receiving status signal (BFM#20485 … corresponding bit).
It is recommended to employ a check system, such as including the data length and data type code in the application data of a message, so that the byte count and data type in the application data can be identified by the receiving side.

7.5.2 Fixed buffer communication program example (with the no procedure control method)

This section explains the programming method in order to communication data (through the no procedure control method) with an external device using the fixed buffers.



(1) Execution environment for the program example

(a) Send program (FX3U-1st station side)

1) Connection No. 1 is used for fixed buffer sending.
2) The communication parameters settings are assumed to have been set as described in Section 5.6.1, "Active open processing/close processing" except the "Fixed buffer communication" parameter. The "Fixed buffer communication" setting should be changed from "Procedure exist" to "No procedure".
3) Fixed buffer No. 1 send data
   : Stored in D1000 to D1015
4) Fixed buffer No. 1 send Send instruction complete device
   : M201
5) Fixed buffer No. 1 send Send instruction abnormal complete device
   : M202
6) Fixed buffer No. 1 send Send instruction complete status
   : D101

(b) Receive program (FX3U -2nd station side)

1) Connection No. 2 is used to process the fixed buffer receiving.
2) The Ethernet module is mounted as the second special module.
3) The unit number specified at the sequence program and FX Configurator-EN (GX Developer) is 1.
4) Make Operation settings in the following manner.

Communication data code	:	Binary code (BFM#24 b1 : OFF)
Initial timing	:	Do not wait for OPEN (Communications impossible at STOP time) (BFM#24 b8 : OFF)
IP address	:	10.97.85.223 (0A.61.55.DFH)

5) Adjust open settings in the following manner.

Protocol	: TCP (BFM#32 b8 : OFF)
Open system	: Unpassive (BFM#32 b14, b15 : 10H)
Fixed buffer	: Receive (BFM#32 b0 : ON)
Fixed buffer communication procedure :	No procedure exist (BFM#32 b9 : ON)
Pairing open	: Disable (BFM#32 b7 : OFF)
Existence confirmation	: No confirm (BFM#32 b1 : OFF)
Host station Port No.	: 8192 (BFM#40)

6) Fixed buffer No.1 receive data.
: Stored in D1000 to D1015

7) Fixed buffer No.1 receive instruction complete status.
: D100

POINT

Make sure to have sufficient device memory according to the maximum data length sent in order to prevent device areas used for other purposes from being overwritten by the received data.

(2) Outline of the program example

(a) Send program (Ethernet-1st station side)

1) After setting each parameter with FX Configurator-EN (GX Developer) or a sequence program and writing to the Ethernet module, power ON the module again, or confirm that the initial processing is completed via buffer memory operations.

2) Perform open processing (Active open) for connection No. 1. (∗1)

3) Communicate data from the PLC using fixed buffer communication (no procedure sending).

4) After data sending is complete, perform close processing for connection No. 1. (∗1)

∗1 Use the program example described in Section 5.6.1, "Active open processing/close processing" for the sequence program that executes the open processing/close processing.
Make sure to change the "Fixed buffer communication" setting from "Procedure exist" to "No procedure".

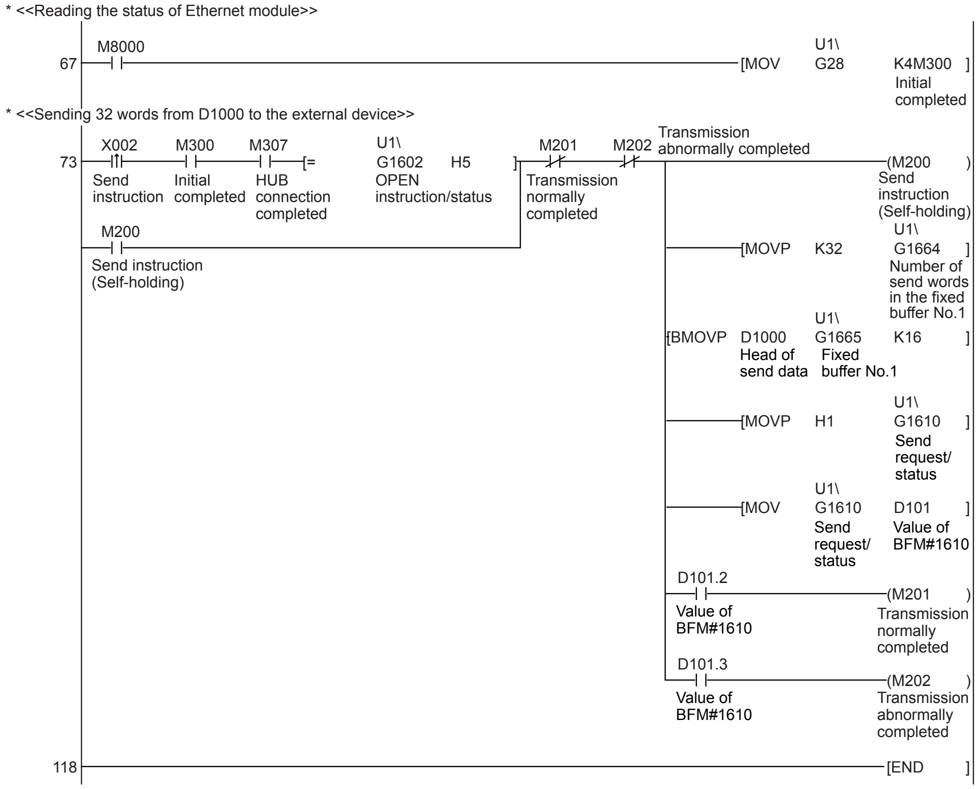
(b) Receive program (Ethernet-2nd station side)

1) After setting each parameter with FX Configurator-EN (GX Developer) or a sequence program and writing to the Ethernet module select re-initial and power ON the module again, or confirm that the initial processing is completed via buffer memory operations.
If the initial processing is normally completed, connection No. 1 waits for an Active open request from the external device.

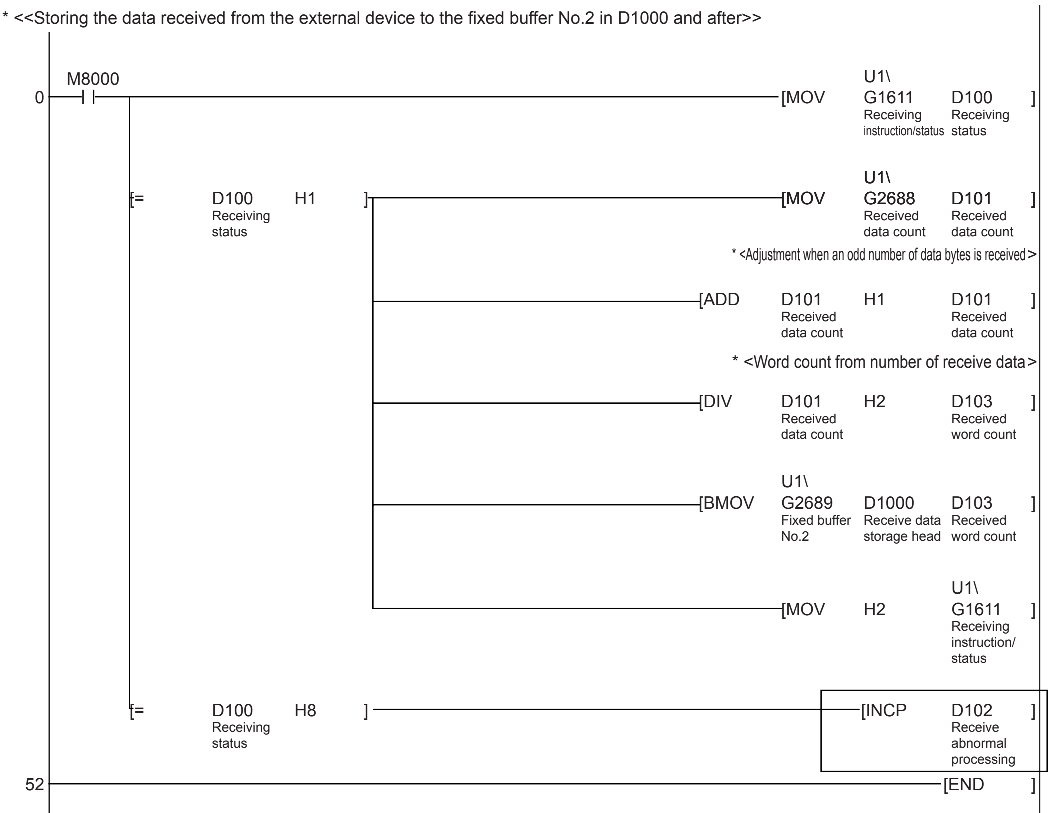
2) Transfer data from the external device using fixed buffer communication (no procedure sending).

3) The data received by the corresponding fixed buffer data area in the Ethernet module is read to the PLC.

(Send program)



(Receive program)


* <<Storing the data received from the external device to the fixed buffer No.2 in D1000 and after>>
0
M8000
[MOV U1\ G1611 Receiving instruction/status D100 Receiving status]
[= D100 Receiving status H1]
[MOV U1\ G2688 Received data count D101 Received data count]
* <Adjustment when an odd number of data bytes is received>
[ADD D101 Received data count H1 D101 Received data count]
* <Word count from number of receive data>
[DIV D101 Received data count H2 D103 Received word count]
[BMOV U1\ G2689 Fixed buffer No.2 D1000 Receive data storage head D103 Received word count]
[MOV H2 U1\ G1611 Receiving instruction/ status]
[= D100 Receiving status H8]
[INCP D102 Receive abnormal processing]
52
[END]

User applications are described at the area enclosed with ▭.

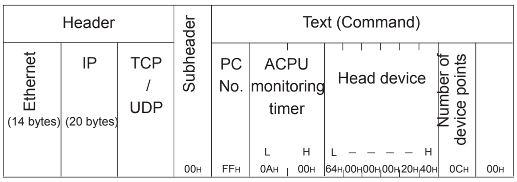
8 COMMUNICATION USING MC PROTOCOL

This chapter gives an overview of the MC protocol.

NOTE

The frame type of MC protocol (data communication messages) used by the external device to access the PLC via this product is equivalent to a compatible 1E frame.

(Example)



(Command message for the A compatible 1E frame)

8.1 Data Communication Function

The MC protocol is the abbreviated name of the MELSEC protocol that is a communication system for the PLCs. Using this protocol, the external devices can read or write device data from/to the PLC via the Ethernet module.
Any external devices on which application programs can be installed and which can send and receive data in accordance with the MELSEC PLC protocol can access the PLC using MC Protocol.

8.1.1 Accessing the PLC using MC protocol

This section explains the main functions for accessing the PLC using MC protocol.
On the PLC side, the Ethernet module sends and receives data based on the instructions (protocol) from the external devices, so the PLC side does not require sequence programs for data communication.

(1) Data read/write

This function reads/writes data from/to the PLC device memory to which the Ethernet module is connected.
By reading and writing data, the PLC operation monitoring, dzata analysis and production management can be performed on the external device side.

(2) Remote control of the PLC

This function executes remote RUN/STOP operations.
Remote operations of the PLC can be performed from the external device side.

	Function		
Communication using MC protocol	Communication using A compatible 1E frames		Communication using ASCII code Communication using binary code *1
	Device memory read/write	Batch read/write in bit/word units	
	Status control of the PLC (remote RUN/STOP, etc.)		

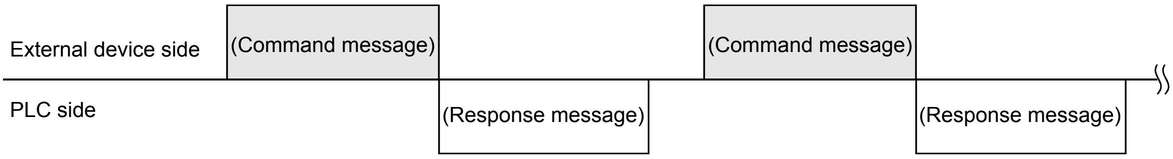
＊1 Time required for communication in binary code is shorter because the amount of the communication data is approximately a half of what is required for communication in ASCII code data.

8.1.2 How to Read the Control Procedures of the MC Protocol

This section explains the control procedures when an external device accesses the PLC using MC protocol.

(1) Transmission of command messages

Data communication through the MC protocol is performed using half-duplex communication.
When accessing the PLC, send the next command message after receiving a response message from the PLC side for the previous command message transmission.

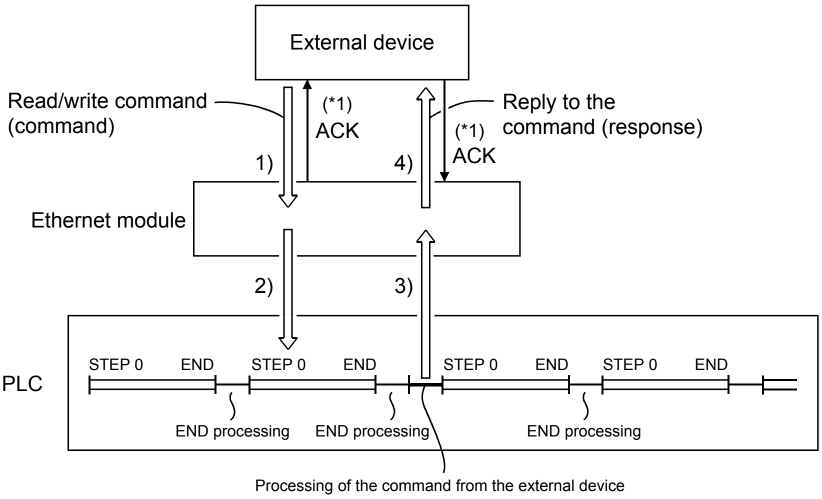


(2) When a normal completion response message to a command message cannot be received

1) If an abnormal completion response message is received
Handle the case according to the error code in the response message.
2) If the response message cannot be received or nothing can be received
Send the command message again after the monitoring time of the response monitor timer.
✻ Modify the value of the monitoring time as needed.

8.1.3 Access Timing on the PLC Side

The following diagram illustrates the access timing on the PLC side when an external device accesses the PLC via the Ethernet module.



1) A command message is transmitted from the external device to the PLC side.
2) Upon receiving a command message from the external device, the FX3U-ENET on the PLC side sends a data read/write request to the PLC according to the content of the command.
3) The PLC reads/writes data according to the request from the external device at the time of executing the sequence program's END instruction, and passes the processing result to the Ethernet module.
4) Upon receiving the processing result from the PLC, the Ethernet module sends a response message including the processing result to the external device that originated the request.

＊1 The ACK response shown in the diagram is sent/received between the Ethernet module and the external device respectively when the access via the Ethernet module is performed by TCP/IP communication.
It is different from the response to the processing requested by the external device in the command message (the processing result).
When access via the Ethernet module is performed by UDP/IP, the ACK response indicated by ＊1 is not sent.

NOTE

Regarding the scan time of the PLC

1) While the PLC is performed by required command, access command to the Ethernet module and the PLC is processed during every END processing. (The scan time becomes longer according to the amount of time it takes to process the command.)
2) When multiple external devices issue access requests to a single station at the same time, the processing requested from the external devices may have to wait for multiple END processing executions depending on the timing of the requests.

8.1.4 PLC setting for performing data communication

Data communication via MC protocol is enabled by adjusting the following settings using FX Configurator-EN (GX Developer) and writing the parameters to the Ethernet module.

1) Initial settings (see Section 5.2)

2) Open settings (see Section 5.5)

8.1.5 Precautions on Data Communication

This section explains some precautions that should be observed when performing data communication between an external device and the Ethernet module.

(1) Precautions that should be observed when performing data communication via the Ethernet module

(a) Perform read/write only when the buffer memory status of Ethernet module is the following.
Initial normal completion signal (BFM# 27 [0001H])
Open completion signal of the connection is used (BFM#20480·corresponding bit)
If the buffer memory status is the above, it is possible to communicate from an external device using MC protocol regardless of whether or not a sequence program is used.

(b) Use the automatic open UDP port. Alternatively, use the passive open connection set to "Always wait for OPEN" (BFM#24 b8:ON) in the initial timing setting in the operation settings.

(c) When not selecting the open 0 connection's communication parameter setting, "MC Protocol Usable" (BFM #32 b10: ON), communication using MC protocol cannot be performed.
To set this using FX-Configurator-EN (GX Developer), select the open settings from the initial screen, and select [Procedure exists (MC)] in [Fixed buffer communication procedure].

(d) Replacing the Ethernet module
The Ethernet address (MAC address) differs depending on the device.
When the Ethernet module is replaced due to breakdown, reboot any external devices also.
Similarly, reboot the Ethernet module even when an external device is replaced (such as a personal computer).

MEMO

9 WHEN COMMUNICATING DATA USING MC PROTOCOL

This chapter explains the message format, how to designate data items in a message and restrictions when communicating data via the Ethernet module using MC protocol and A compatible 1E frames.

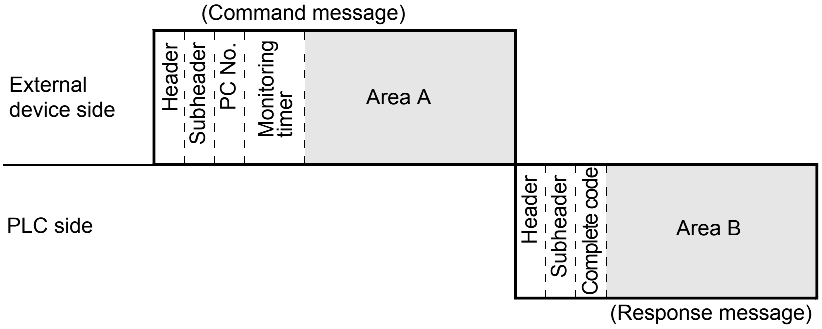
9.1 Message Formats and Control Procedures

This section explains the message format and control procedure for each command when data communication is performed using A compatible 1E frames.
The MC protocol for the Ethernet module is a subset of A compatible 1E frames.
Both TCP/IP and UDP can be used as lower layer protocol and support both ASCII code and binary code.

9.1.1 How to read the command reference section

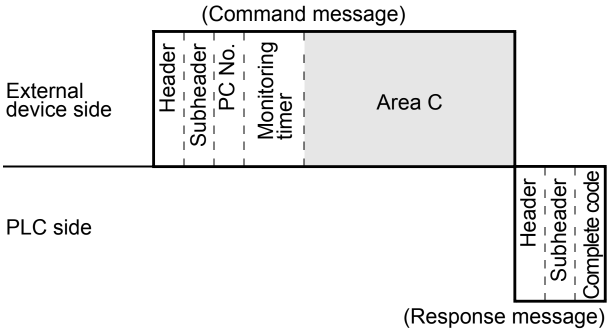
The following explains how to read the message explanation diagrams shown in each of the command description Sections 9.3 through 9.5.

(1) When an external device reads data from the PLC



1) Area A indicates transmission from the external device to the PLC.
2) Area B indicates transmission from the PLC to the external device.
3) The program of the external device is generated so that the data is transmitted sequentially from left to right. (For example: in case of area A, data should be sequentially sent from Header to the right.)

(2) When an external device writes data to the PLC



1) Area C indicates transmission from the external device to the PLC.
2) The program of the external device is generated so that the data is transmitted sequentially from left to right. (For example: in case of area C, data should be sequentially sent from Header to the right.)

POINT

When the PLC receives a command message from an external device, it completes processing of the data in area A/C, then sends a response message and waits for the next command message (neutral state).

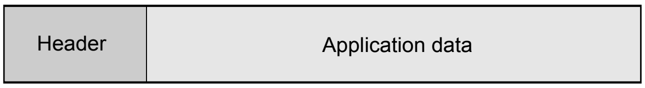
9.1.2 Message format and control procedure

This section explains the message format and control procedure.

(1) Message format

This section explains the message format for transmission between the Ethernet module and an external device.

The communication data consists of "header" and "application data" as shown below.



(a) Header

The header for TCP/IP or UDP/IP is used. The user does not need to specify it; the Ethernet module attaches it.

(b) Application data

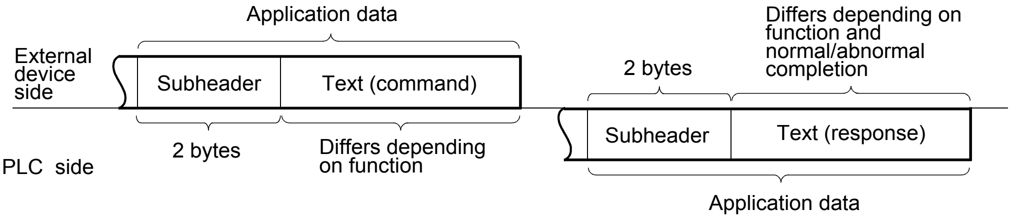
The application data is largely divided into "subheader" and "text" as shown below.

The subheader represents command/response and the setting value is predetermined.

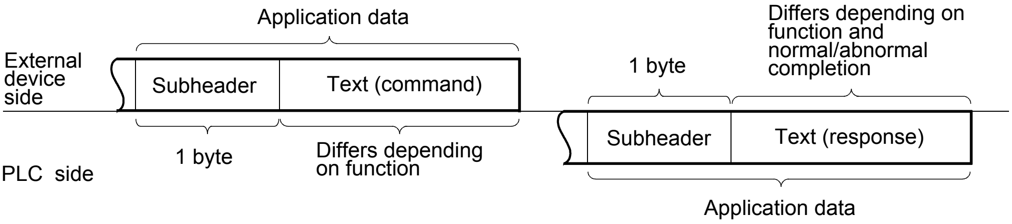
The text contains the request data (command) and response data (response) for each function and this data is determined by the prescribed format (for a more detailed description, see sections 9.3 and later).

(c) Format in the application data field

- Communication in ASCII code



- Communication in binary code



NOTE

When communicating in the MC protocol, the user does not need to specify a response for a command from an external device; the Ethernet module generates it and then responds.

(2) Control procedure

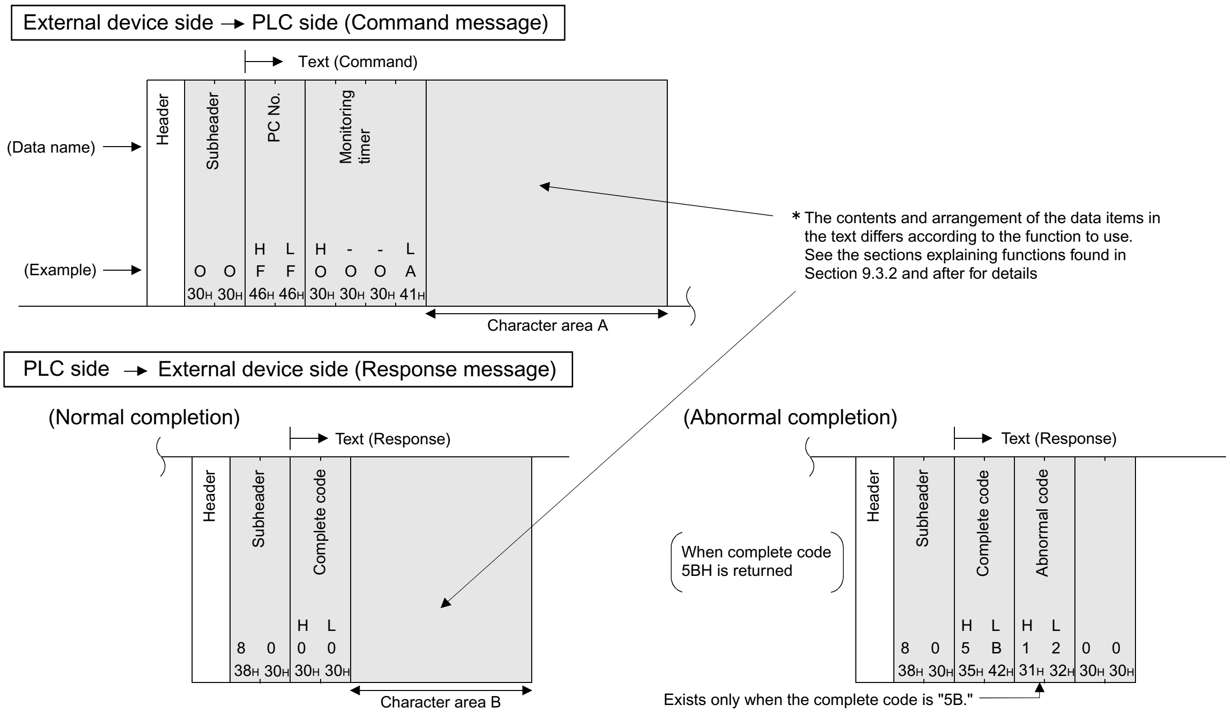
The following diagrams illustrate the control procedure for communicating with the MC protocol and the order of data items in the application data field.

The header section shown in the message explanation diagram of this section corresponds to the ✻ portion of the message explanation diagrams indicated in Section 9.3.2 and later.

See Section 9.1.3 regarding the content of data items in the message format and data specification method.

(a) Communication in ASCII code

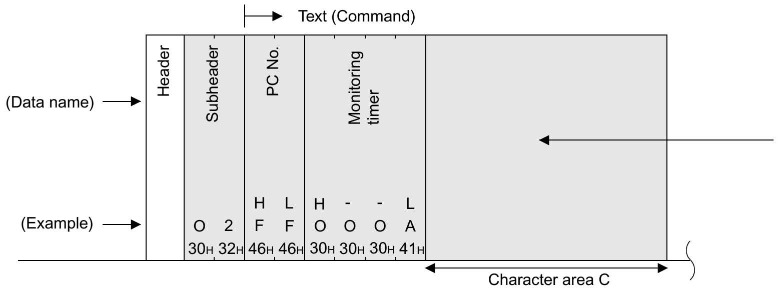
1) When reading data from the local station PLC at the external device side



* The contents and arrangement of the data items in the text differs according to the function to use. See the sections explaining functions found in Section 9.3.2 and after for details

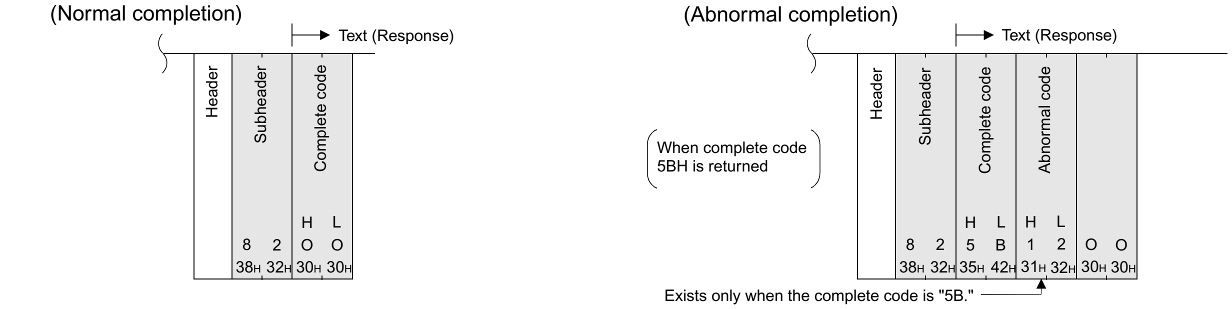
2) When writing data to the local station PLC from the external device side

External device side → PLC side (Command message)



* The contents and arrangement of the data items in the text differs according to the function to use. See the sections explaining functions found in Section 9.3.2 and after for details

PLC side → External device side (Response message)


(Normal completion)

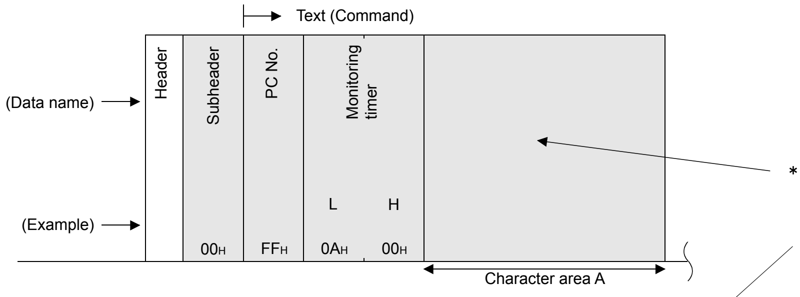
(Abnormal completion)

Exists only when the complete code is "5B."

(b) Communication in binary code

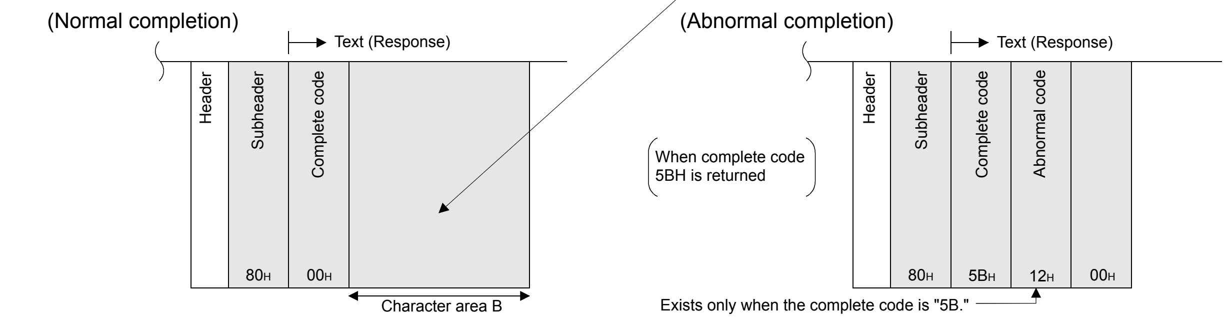
1) When reading data from the local station PLC at the external device side

External device side → PLC side (Command message)



* The contents and arrangement of the data items in the text differs according to the function to use. See the sections explaining functions found in Section 9.3.2 and after for details

PLC side → External device side (Response message)

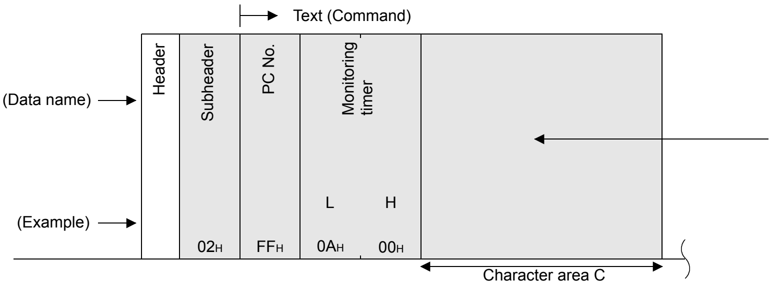

(Normal completion)

(Abnormal completion)

Exists only when the complete code is "5B."

2) When writing data to the local station PLC from the external device side

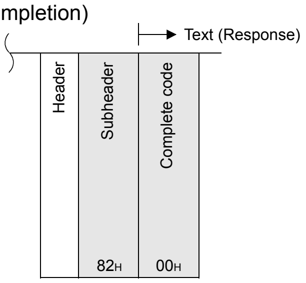
External device side → PLC side (Command message)



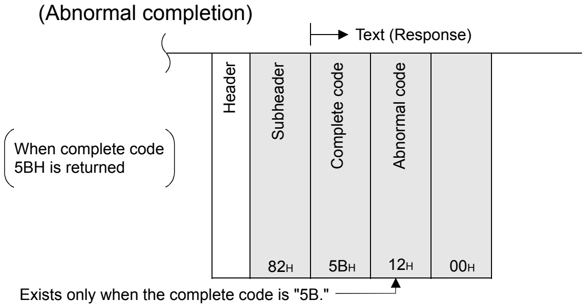
* The contents and arrangement of the data items in the text differs according to the function to use. See the sections explaining functions found in Section 9.3.2 and after for details

PLC side → External device side (Response message)

(Normal completion)



(Abnormal completion)


Text (Response)
When complete code 5BH is returned
Header
Subheader
Complete code
Abnormal code
82H
5BH
12H
00H
Exists only when the complete code is "5B."

9.1.3 Contents of data designation items

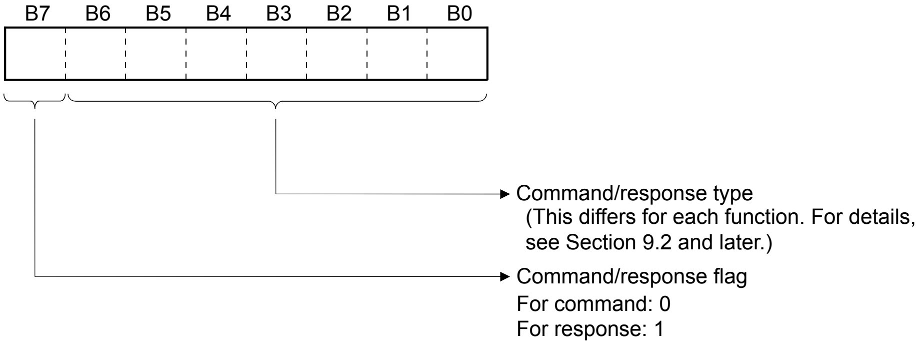
This section explains the data items of commands and responses when communicating using MC protocol.
In the response that is returned by the Ethernet module to an external device, the data is automatically specified by the Ethernet module; the user does not need to specify it.

(1) Header

The header for TCP/IP or UDP/IP is used. The user does not need to specify it at the PLC side; the Ethernet module attaches it.

(2) Subheader

The format of the subheader is configured as follows.



(3) PC number

The PC number is fixed at a value of “FFH”.

(a) When communicating in binary code, the PC number is expressed in binary value.

(b) When communicating in ASCII code, the PC number is expressed in hexadecimal ASCII code.

(4) Monitoring timer

This is a timer for setting the period of time that the Ethernet module should wait after outputting a read/write request to the PLC until the result is returned.

(a) Specify the value as shown below.

0000H (0):	:	Waits infinitely (＊1)
0001 to FFFFH (1 to 65535)	:	Wait time (in units of 250 ms)

＊1 It keeps waiting until a response is returned from the PLC.

(b) For normal data communications, using a value within the setting range of 1 to 40 (0.25 to 10s) is recommended.

(5) Character field (command)

This data contains the Ethernet module commands that indicate functions for when an external device reads/writes data from/to the target PLC station.
The contents and order of data in the Character field (command) field differ depending on the function used.
The order of data items for each function is explained in each function's reference section in Section 9.3.2. and later.

(6) Character field (response)

This data contains data read/processing result when an external device reads/writes data from/to the target PLC station.
The contents and order of data in the Character field (response) differ depending on the function used.
The order or data items at normal completion for each function is explained in each function's reference section in Section 9.3.2. and later.

(7) Complete code

The result of processing when an external device reads/writes data from/to the target PLC station is indicated by the following values.

00H	:	Normal completion
Other than 00H	:	Abnormal completion (50H to 60H)

(a) When communicating in binary code, the complete code is expressed in binary values.

(b) When communicating in ASCII code, the complete code is expressed in hexadecimal ASCII code.

(c) When the complete code indicates abnormal completion, check the content and take an action according to the troubleshooting section of Chapter 11. When the complete code is 5BH/"5B," the abnormal code data (10H to 18H) and 00H/"00" are included immediately after.

(8) Abnormal code

This value indicates the nature of the error when the processing result of reading/writing data from/to the target PLC station by an external device is faulty and the complete code is 5BH/"5B." (Abnormal code: 10H to 18H)

(a) When communicating in binary code, the abnormal code is expressed in binary values.

(b) When communicating in ASCII code, the abnormal code is expressed in hexadecimal ASCII code.

(c) Check the content and take an action according to the troubleshooting section of Chapter 11.

POINT

The data code (ASCII/binary) when sending/receiving commands and responses between the Ethernet module and an external device is determined in the operation settings in FX Configurator-EN (GX Developer).
The external device communicating with Ethernet module should send the values specified in each data item in the commands and responses in the above setting using the code shown below. Also, it should receive the corresponding values in the code shown below.

In the explanation hereafter in this section, the values specified in each item in a command and response are shown in binary values.

(1) Communication in binary code
Unless specifically stated, the value shown in each explanation is sent/received in the designated order (L to H) since it is in binary.

(2) Communication in ASCII code
Unless specifically stated, the value shown in each explanation is converted to hexadecimal ASCII code and sent/received in the designated order (H to L).

NOTE

The following example shows the designation of the subheader to the monitoring timer when communicating using MC protocol under the following conditions.

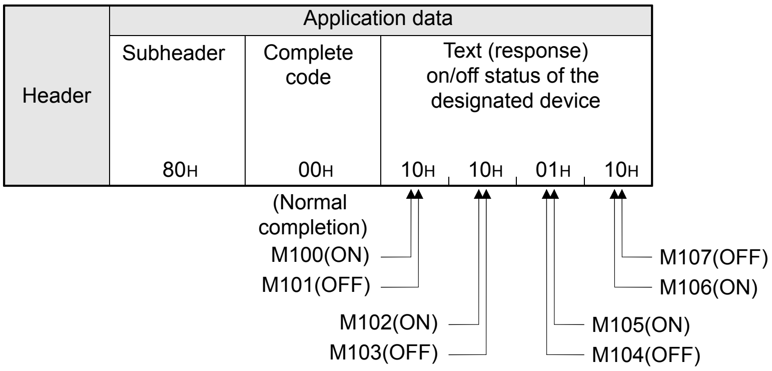
				(Designated value)
• Target station	:	FX3U/FX3UC series to which FX3U-ENET is connected	:	FFH
• Function used	:	Device memory batch read (bit units)	:	00H
• Monitoring timer value	:	2500 ms	:	000AH

(1) Format when communicating in binary code

(a) The order when sending a command (external device → Ethernet module)

Header	Application data											
	Subheader	PC number	Monitoring timer		Text (command)							
					(Head device number)				(Device name)		Number of device points	
			(L)	(H)	(L)	–	–	(H)	(L)	(H)		
	00H	FFH	0AH	00H	64H	00H	00H	00H	20H	4DH	08H	00H
		(Local station)	(2500 ms)		(100)				(M)		(8 points)	

(b) The order when receiving a response (external device ← Ethernet module)



Header	Application data					
	Subheader	Complete code	Text (response) on/off status of the designated device			
	80H	00H	10H	10H	01H	10H
		(Normal completion)				

(2) Format when communicating in ASCII code

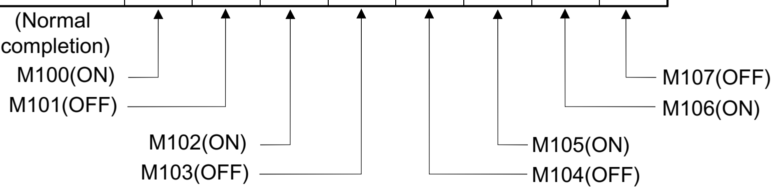
(a) The order when sending a command (external device → Ethernet module)

Header	Application data							
	Subheader		PC number		Monitoring timer			
			(H)	(L)	(H)		(L)	
	"0"	"0"	"F"	"F"	"0"	"0"	"0"	"A"
	30H	30H	46H	46H	30H	30H	30H	41H
			(Local station)		(2500 ms)			

Application data															
Text (command)															
(Device name)				(Head device number)								Number of device points			
(H)	–	–	(L)	(H)	–	–	–	–	–	–	(L)	(H)	(L)		
"4"	"D"	"2"	"0"	"0"	"0"	"0"	"0"	"0"	"0"	"6"	"4"	"0"	"8"	"0"	"0"
34H	44H	32H	30H	30H	30H	30H	30H	30H	30H	36H	34H	30H	38H	30H	30H
(M)				(100)								(8 points)			

(b) The order when receiving a response (external device ← Ethernet module)

Header	Application data											
	Subheader		Complete		Text (response) ON/OFF status of the designated device							
			(H)	(L)	(H)	–	–	–	–	–	–	(L)
	8	"0"	"0"	"0"	"1"	"0"	"1"	"0"	"0"	"1"	"1"	"0"
	38H	30H	30H	30H	31H	30H	31H	30H	30H	31H	31H	30H
			(Normal completion)									



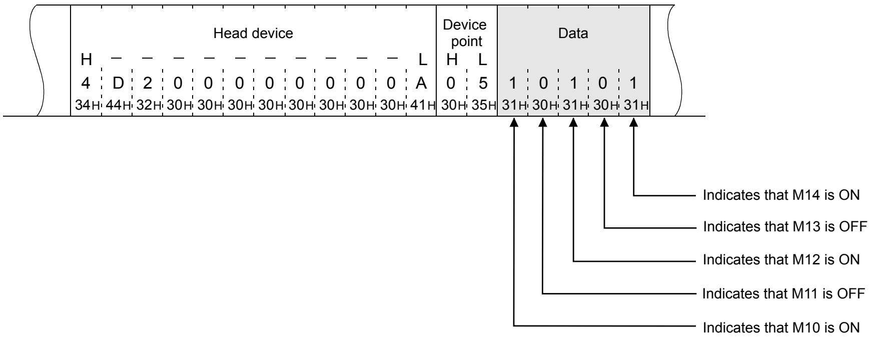
9.1.4 Character area transmission data

This section explains how to transmit the bit device data and word device data handled in the character areas, as well as the order of transmission when communicating data between an external device and the PLC by each command.
The transmission data shown in the examples is contained in character area B in case of reading and monitoring, and in character area C in case of writing, testing, and monitor data registration.

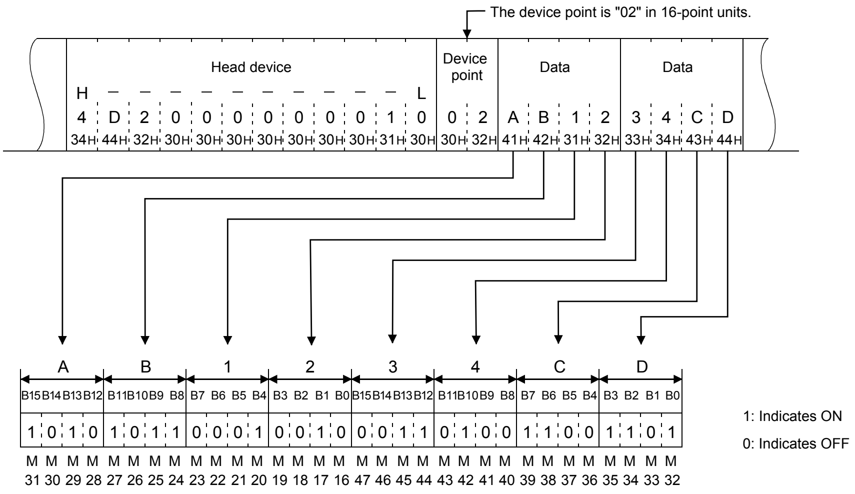
(1) Data communication using ASCII code

(a) When reading to or writing from a bit device memory
The bit device memory can be read and written in bit units (one device point) or word units (16 device points).
How data is transmitted in each case is explained below.

1) Bit units (one point)
In case of bit units, the bit device memory is handled from the designated head device for the number of designated device points sequentially from the left. They are expressed as "1" (31H) if the device is on or "0" (30H) if the device is off.
(Example)
Indicating the ON/OFF status of five points from M10



2) Word units (16 points)
When the bit device memory is handled as word units, each word is expressed in hexadecimal values in 4-bit units sequentially from the higher bit.
(Example)
Indicating the ON/OFF status of 32 points from M16

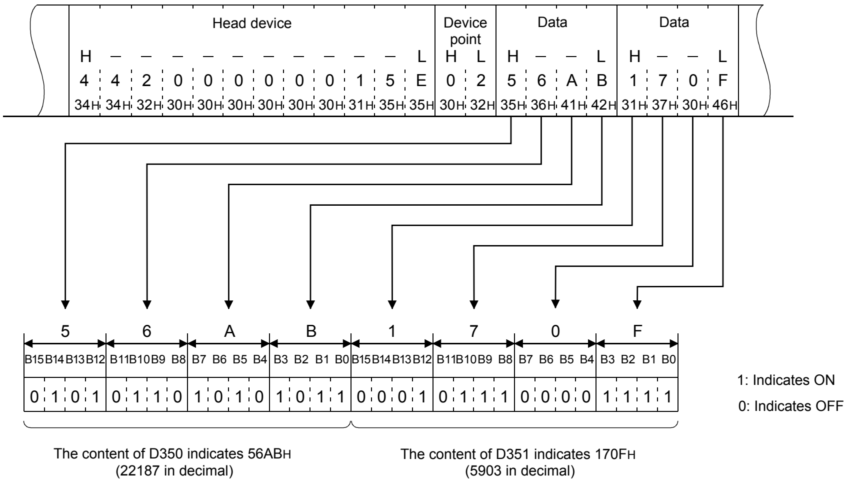

The device point is "02" in 16-point units.
Head device
Device point
Data
Data
H – – – – – – – – – – – L
4 D 2 0 0 0 0 0 0 0 1 0
34H 44H 32H 30H 30H 30H 30H 30H 30H 30H 31H 30H
0 2
30H 32H
A B 1 2
41H 42H 31H 32H
3 4 C D
33H 34H 43H 44H
A B 1 2 3 4 C D
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
1 0 1 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1
M31 M30 M29 M28 M27 M26 M25 M24 M23 M22 M21 M20 M19 M18 M17 M16 M47 M46 M45 M44 M43 M42 M41 M40 M39 M38 M37 M36 M35 M34 M33 M32
1: Indicates ON
0: Indicates OFF

(b) When reading from or writing to a word device memory

Each word of the word device memory is expressed in hexadecimal values in 4-bit units sequentially from the higher bit.

(Example)

Indicating the storage contents of data registers D350 and D351


Head device
Device point
Data
Data
H — — — — — — — — — — — L
H L
H — — L
H — — L
4 4 2 0 0 0 0 0 0 1 5 E
0 2
5 6 A B
1 7 0 F
34H 34H 32H 30H 30H 30H 30H 30H 30H 31H 35H 35H
30H 32H
35H 36H 41H 42H
31H 37H 30H 46H
5 6 A B 1 7 0 F
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0
0 1 0 1 0 1 1 0 1 0 1 0 1 0 1 1
0 0 0 1 0 1 1 1 0 0 0 0 1 1 1 1
1: Indicates ON
0: Indicates OFF
The content of D350 indicates 56ABH (22187 in decimal)
The content of D351 indicates 170FH (5903 in decimal)

POINT

(1) When designating alphabet letters in the character area, use upper case code.

(2) When data other than an integer value (real number, character string, etc.) was stored in a word device memory to be read, the stored value is read as an integer value.

(Example 1)

When the real number (0.75) is stored in D0 and D1, it is read as the following integer values:

D0 = 0000H, D1 = 3F40H

(Example 2)

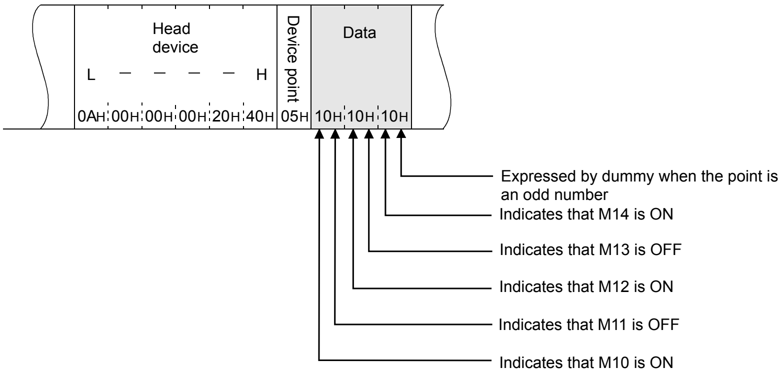
When the character string ("12AB") is stored in D2 and D3, it is read as the following integer values:

D2 = 3231H, D3 = 4241H

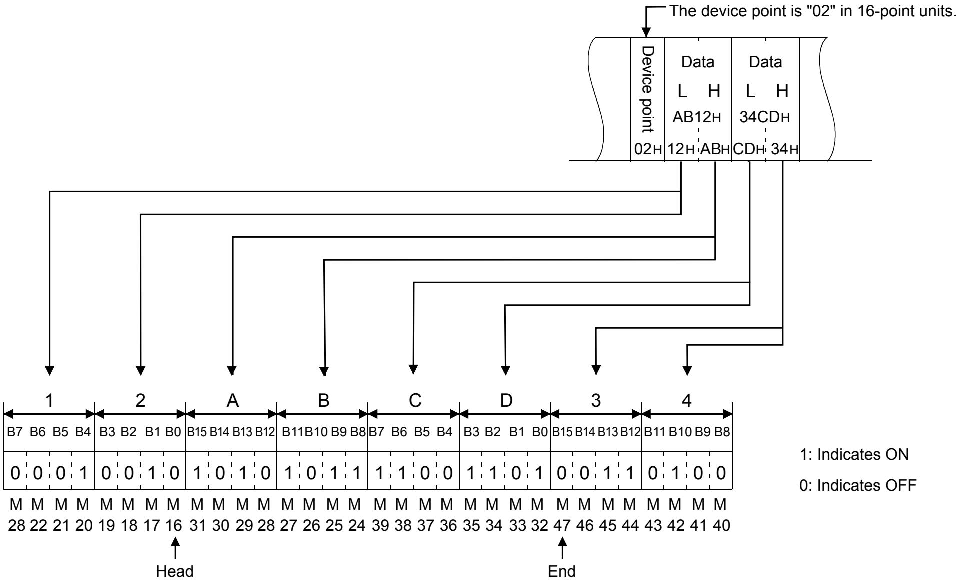
(2) Data communication using binary code

(a) When reading to or writing from the bit device memory
The bit device memory can be read and written in bit units (one device point) or word units (16 device points).
How data is transmitted in each case is explained below.

1) Bit units (one point)
In case of bit units, four bits designate one point and the bit device memory is handled from the designated head device for the number of designated device points sequentially from the left. They are expressed as "1" if the device is ON or "0" if the device is OFF.
(Example)
Indicating the ON/OFF status of five points from M10



2) Word units (16 points)
In case of word units, one bit designates one point and the bit device memory is handled from the designated head device for the number of designated device points sequentially from the left. They are expressed in 16-point units in the order, low byte (L: bits 0 to 7) to high byte (bits 8 to 15).
(Example)
Indicating the ON/OFF status of 32 points from M16

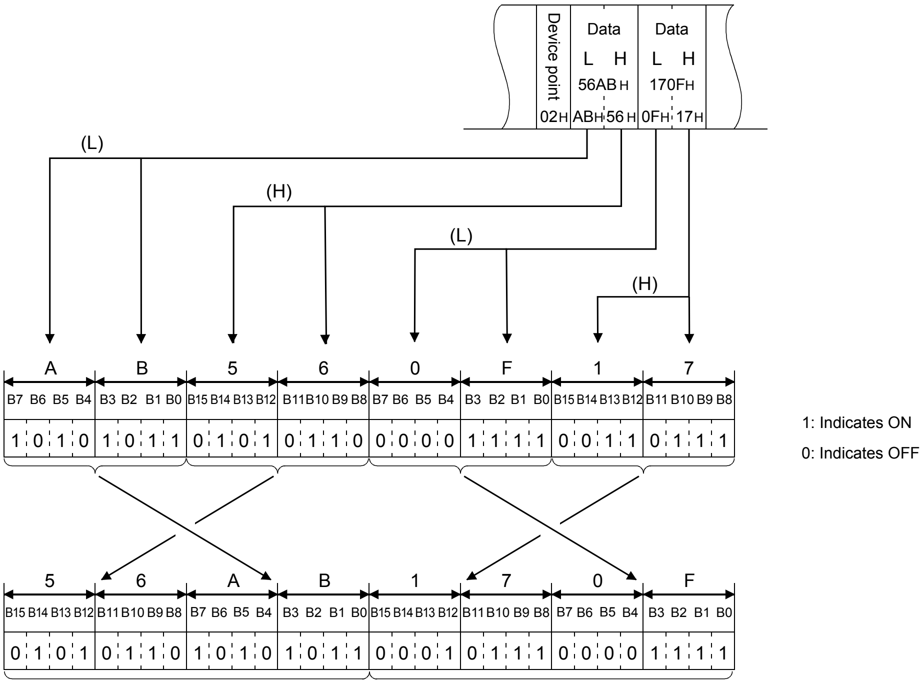

The device point is "02" in 16-point units.
Device point
Data
L H
AB12H
Data
L H
34CDH
02H
12H ABH CDH 34H
1 2 A B C D 3 4
B7 B6 B5 B4 B3 B2 B1 B0 B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 B15 B14 B13 B12 B11 B10 B9 B8
0 0 0 1 0 0 1 0 1 0 1 0 1 0 1 1 1 1 0 0 1 1 0 1 0 0 1 1 0 1 0 0
M28 M22 M21 M20 M19 M18 M17 M16 M31 M30 M29 M28 M27 M26 M25 M24 M39 M38 M37 M36 M35 M34 M33 M32 M47 M46 M45 M44 M43 M42 M41 M40
Head
End
1: Indicates ON
0: Indicates OFF

(b) Reading from or writing to a word device memory

Each word of a word device memory is designated by 16 bits and the designated number of points from the designated head device are sequentially expressed in one-point units in the order, low byte (L: bits 0 to 7) to high byte (H: bits 8 to 15).

(Example)

Indicating the storage contents of data registers D350 and D351



The content of D350 indicates 56ABH
(22187 in decimal)

The content of D351 indicates 170FH
(5903 in decimal)

POINT

(1) When data other than an integer value (real number, character string, etc.) was stored in a word device memory to be read, the module reads the stored value as an integer value.

(Example 1)

When the real number (0.75) is stored in D0 and D1, it is read as the following integer values:

D0 = 0000H, D1 = 3F40H

(Example 2)

When the character string ("12AB") is stored in D2 and D3, it is read as the following integer values:

D2 = 3231H, D3 = 4241H

9.2 List of Commands and Functions for The MC protocol

The following table lists the commands and functions when an external device accesses the PLC.

– Function			Command/response type	Processing	Number of points processed per communication
Device memory	Batch read	Bit units	00$_H$	Reads bit devices (X, Y, M, S, T, C) in 1-point units.	256 points
		Word units	01$_H$	Reads bit devices (X, Y, M, S, T, C) in 16-point units.	32 words (512 points)
				Reads word devices (D, R, T, C) in 1-point units. (＊1)	64 points
	Batch write	Bit units	02$_H$	Writes to bit devices (X, Y, M, S, T, C) in 1-point units.	160 points
		Word units	03$_H$	Writes to bit devices (X, Y, M, S, T, C) in 16-point units.	10 words (160 points)
				Writes to word devices (D, R, T, C) in 1-point units.	64 points
	Test (Random write)	Bit units	04$_H$	Sets/resets bit devices (X, Y, M, S, T, C) in 1-point units by arbitrarily designating the devices and device number.	80 points
		Word units	05$_H$	Sets/resets bit devices (X, Y, M, S, T, C) in 16-point units by arbitrarily designating the devices and device number.	10 words (160 points)
				Sets/resets word devices (D, R, T, C) in 1-point units by arbitrarily designating the devices and device numbers. Not applicable for 32 bit devices from C200 to C255.	10 points
PLC control	Remote RUN		13$_H$	Requests the remote RUN/STOP to PLC.	
	Remote STOP		14$_H$		
	PLC model name read		15$_H$	Reads PLC model name code.	
Loop test			16F$_H$	Returns the character received from the other node to that node with the data as it is.	254 bytes

＊1 When the double word data is read using the user program, etc., read out the 32bit data at once.

9.3 Device Memory Read/Write

This section explains the designations in the control procedure when reading from and writing to the device memory by providing an example.

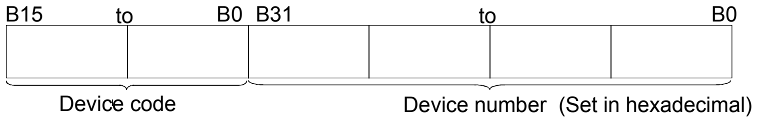
9.3.1 Commands and device range

(1) Commands used for reading from and writing to the device memory

Item		Command/response type	Processing	Number of points processed per communication
Batch read	Bit units	00H	Reads bit devices (X, Y, M, S, T, C) in 1-point units.	256 points
	Word units	01H	Reads bit devices (X, Y, M, S, T, C) in 16-point units.	32 words (512 points)
			Reads word devices (D, R, T, C) in 1-point units.	64 points
Batch write	Bit units	02H	Writes to bit devices (X, Y, M, S, T, C) in 1-point units.	160 points
	Word units	03H	Writes to bit devices (X, Y, M. S, T, C) in 16-point units.	10 words (160 points)
			Writes to word devices (D, R, T, C) in 1-point units.	64 points
Test (random write)	Bit units	04H	Sets/resets bit devices (X, Y, M, S, T, C) in 1-point units by arbitrarily designating the devices and device number.	80 points
	Word units	05H	Sets/resets bit devices (X, Y, M, S, T, C) in 16-point units by arbitrarily designating the devices and device numbers.	10 words (160 points)
			Writes to word devices (D, R, T, C, etc.) in 1-point units by arbitrarily designating the devices and device numbers. Not applicable for 32 bit devices from C200 to C255.	10 points

(2) Designation method and accessible range of devices.

(a) In device read/write, each device is set by a device code and number as shown in the diagram below.



[Example] In case of D100

D100 = 4420 00000064_H

→ Device number (hexadecimal)

→ Device code

(b) The following table outlines the device codes and numbers.

Device list

Device		Device code	Device range	Device number
Data register		D (44_H, 20_H)	D0 to D7999	0000_H to $1F3F_H$
			D8000 to D8511	$1F40_H$ to $213F_H$
Extension register		R (52_H, 20_H)	R0 to R32767	0000_H to $7FFF_H$
Timer	Current value	TN (54_H, $4E_H$)	T0 to T511	0000_H to $01FF_H$
	Contact	TS (54_H, 53_H)	T0 to T511	0000_H to $01FF_H$
Counter	Current value	CN (43_H, $4E_H$)	C0 to C199	0000_H to $00C7_H$
			C200 to C255	$00C8_H$ to $00FF_H$
	Contact	CS (43_H, 53_H)	C0 to C199	0000_H to $00C7_H$
			C200 to C255	$00C8_H$ to $00FF_H$
Input		X (58_H, 20_H)	X0 to X377	0000_H to $00FF_H$
Output		Y (59_H, 20_H)	Y0 to Y377	0000_H to $00FF_H$
Internal relay		M ($4D_H$, 20_H)	M0 to M7679	0000_H to $1DFF_H$
			M8000 to M8511	$1F40_H$ to $213F_H$
State		S (53_H, 20_H)	S0 to 4095	0000_H to $0FFF_H$

POINT

(1) Bit devices and word devices are classified according to the following.

Bit devices	:	X, Y, M, S, T (contact), C (contact)
Word devices	:	T (current value), C (current value), D, R

(2) When word units are designated, always make the head device number of a bit device a multiple of 16 (0, 16... in decimal representation).
For X and Y is expressed in octadecimal, the device numbers that can be designated are X00, X20, X40,,, or Y00, Y20, Y40,,,.
Also, special M relays M8000 or later can be specified (8000 + multiples of 16).

(3) The special relays (M8000 to M8511) and special registers (D8000 to D8511) are divided into read only, write only and system use registers.
If writing takes place outside the writing enabled range, a PLC error may occur.
See the Programming Manual for detailed descriptions of the special relays and the special registers.

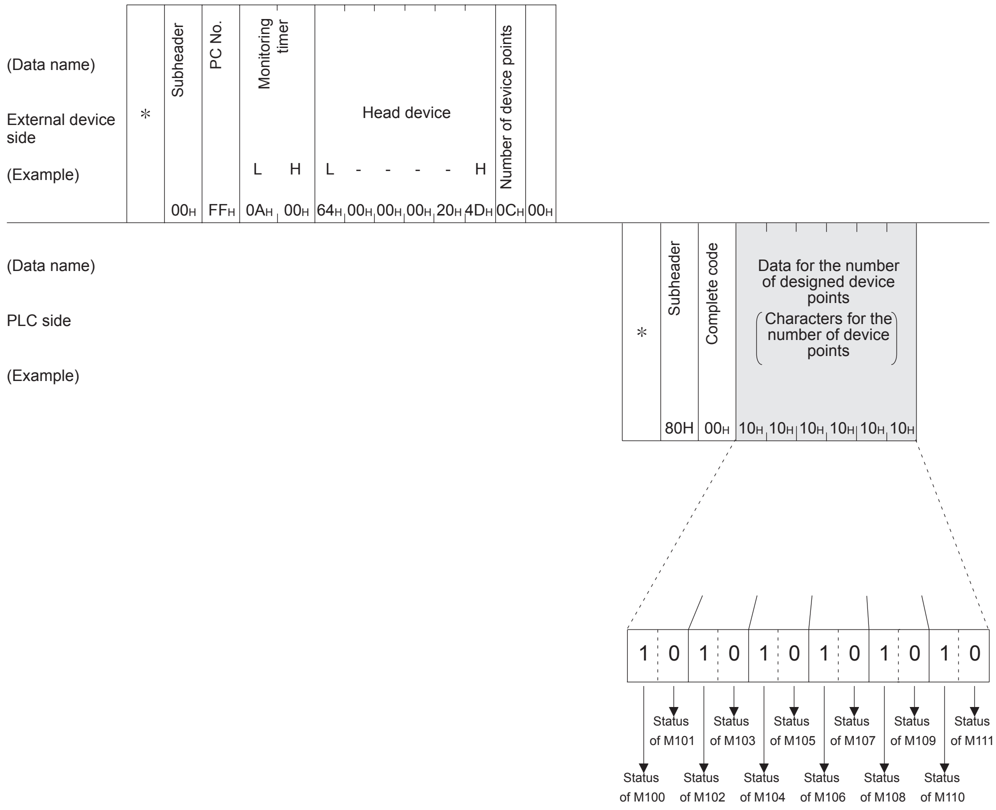
9.3.2 Batch read in bit units (command: 00)

The examples shown in this section explain the command/response format when batch-reading the bit device memory.
For more details on the order and contents of data items of the areas marked by "*" shown in the control procedure diagram, see Section 9.1.2.

[Control procedure]

Reading the on/off status of M100 to M111 of the PLC on which the FX3U-ENET is loaded.

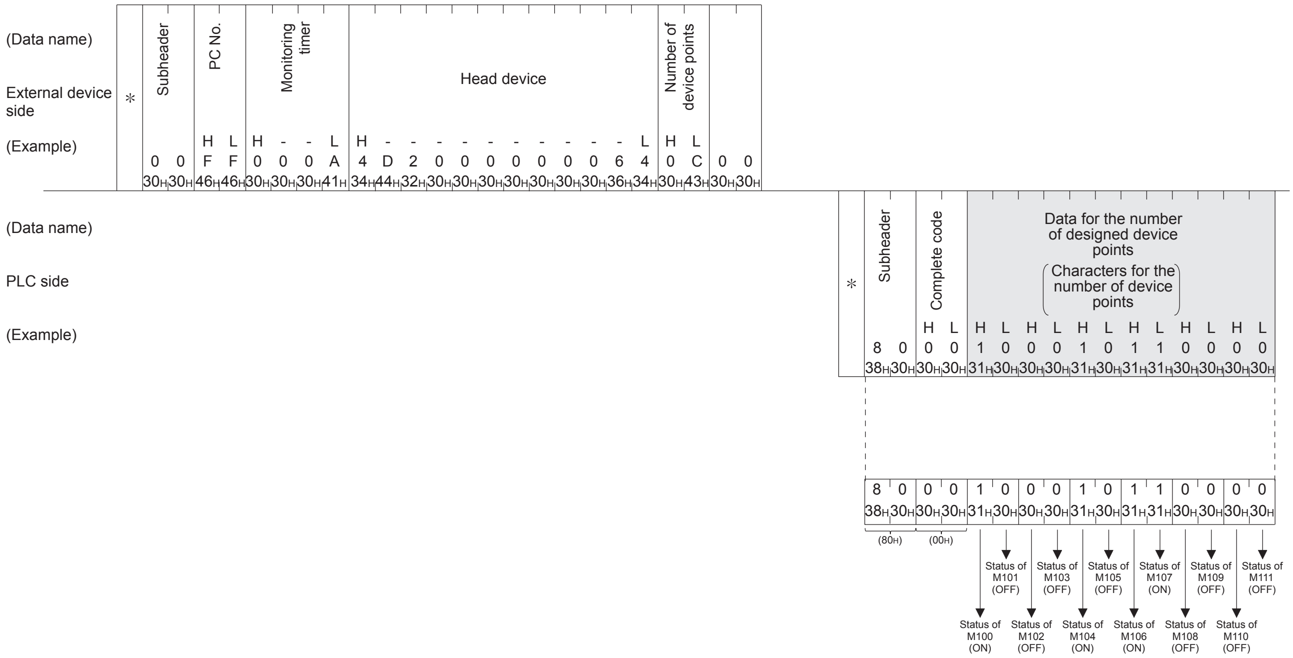
(1) Communication in binary code



NOTE

Use the designation "00H" when the number of device points is 256 points.

(2) Communication in ASCII code



NOTE

(1) Use the designation "00H" when the number of device points is 256 points.

(2) If the number of device points designated is an odd number, one byte of dummy data (30H) will be added to the response data. For example, if three points are read, data for four points is returned. The last byte is dummy data.

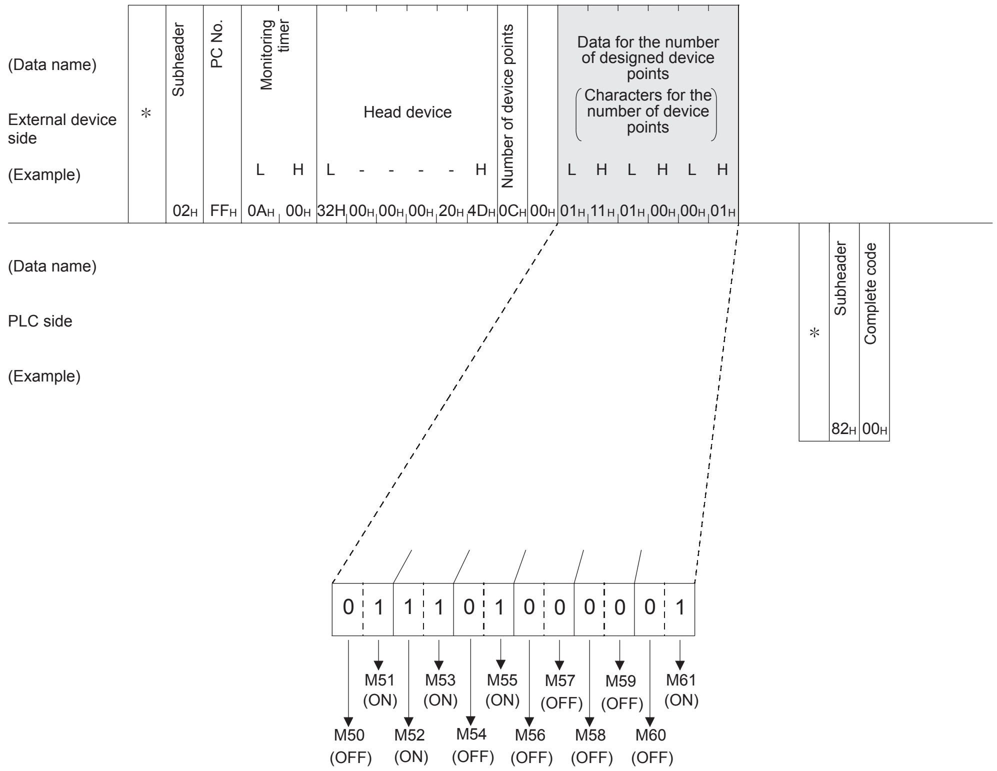
9.3.3 Batch write in bit units (command: 02)

The examples shown in this section explain the command/response format when batch writing to the bit device memory.
For more details on the order and contents of data items of the areas marked by "*" shown in the control procedure diagram, see Section 9.1.2.

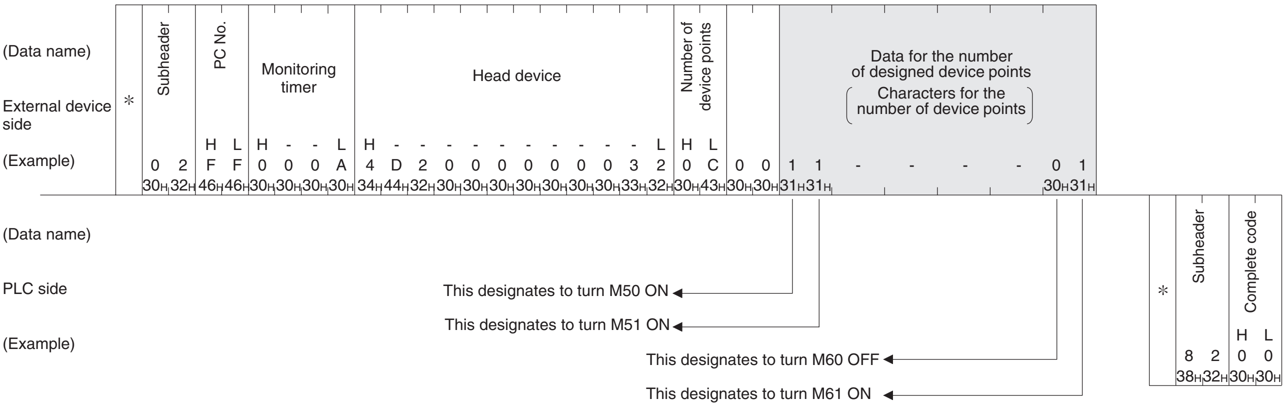
[Control procedure]

Writing the on/off status of M50 to M61 of the PLC on which the Ethernet module is loaded.

(1) Communication in binary code



(2) Communication in ASCII code


(Data name)
External device side
(Example)
*
Subheader
PC No.
Monitoring timer
Head device
Number of device points
Data for the number of designed device points
Characters for the number of device points
H L H - - L H - - - - - - - - - - L H L
0 2 F F 0 0 0 A 4 D 2 0 0 0 0 0 0 0 3 2 0 C 0 0 1 1 - - - - 0 1
30H 32H 46H 46H 30H 30H 30H 30H 34H 44H 32H 30H 30H 30H 30H 30H 30H 30H 33H 32H 30H 43H 30H 30H 31H 31H 30H 31H
(Data name)
PLC side
(Example)
This designates to turn M50 ON
This designates to turn M51 ON
This designates to turn M60 OFF
This designates to turn M61 ON
*
Subheader
Complete code
H L
8 2 0 0
38H 32H 30H 30H

NOTE

If the number of device points to be set is an odd number, add one byte of dummy data (30H) at the end of data written. For example, when writing three points, add the dummy data (30H) at the end.

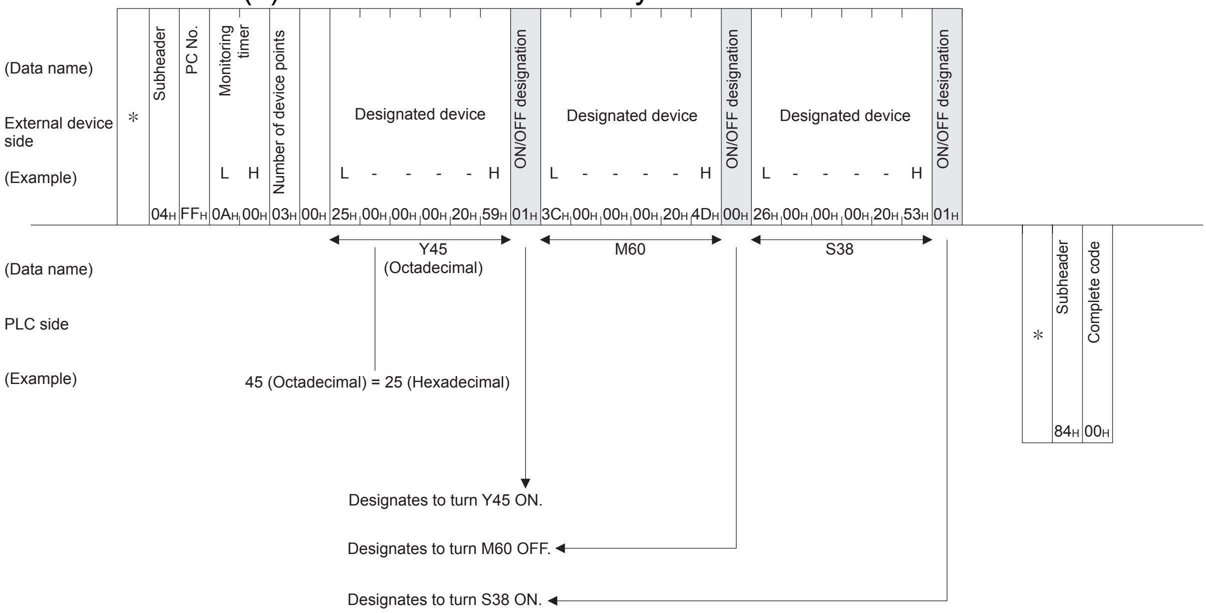
9.3.4 Test in bit units (random write) (command: 04)

The examples shown in this section explain the command/response format when writing data by designating bit device memories arbitrarily.
For more details on the order and contents of data items of the areas marked by "＊" shown in the control procedure diagram, see Section 9.1.2.

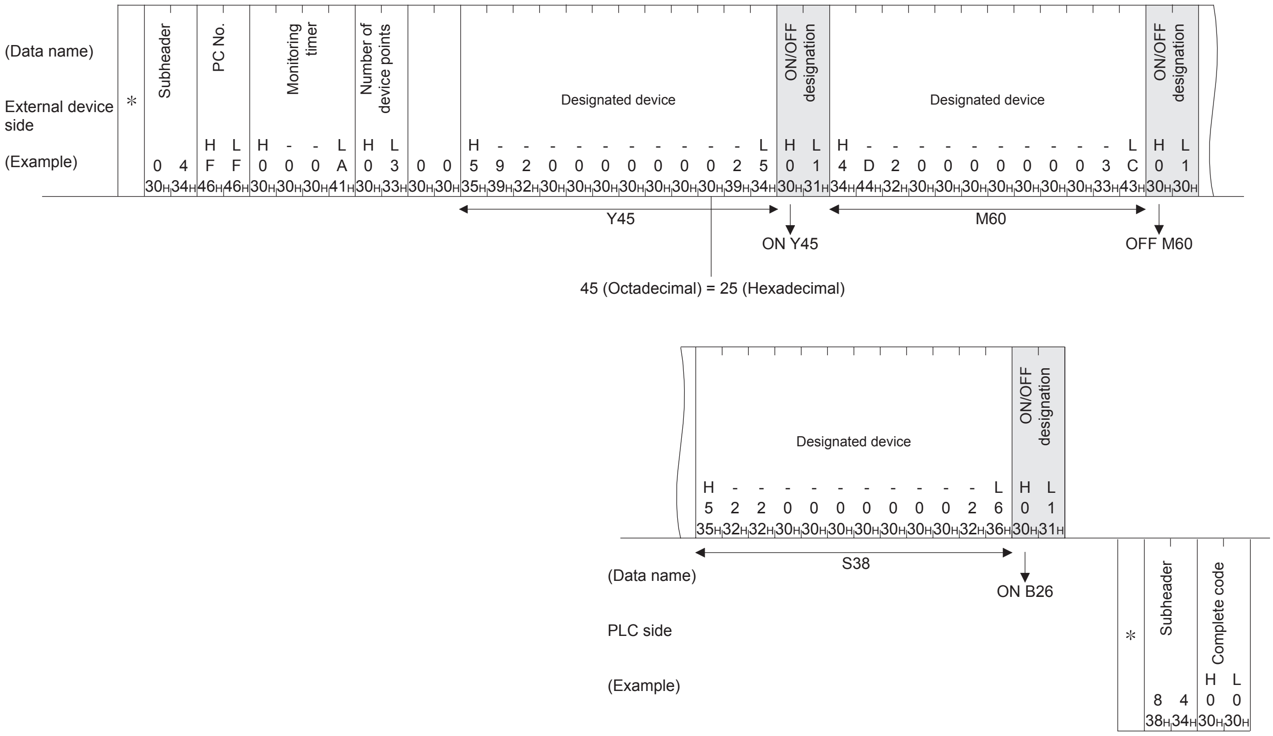
[Control procedure]

Specifying Y45 to ON, M60 to OFF, and S38 to ON at the PLC on which the Ethernet module is loaded.

(1) Communication in binary code



(2) Communication in ASCII code



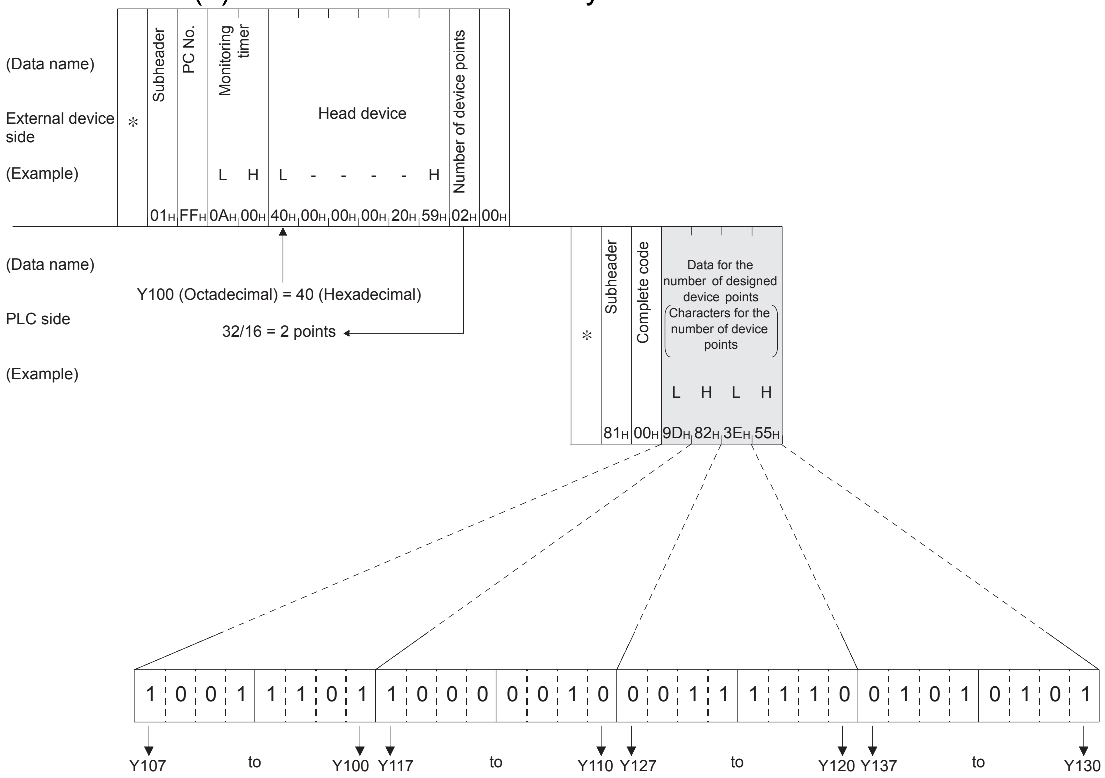
9.3.5 Batch read in word units (command: 01)

The examples shown in this section explain the command/response format when batch reading the word device memory and the bit device memory (16 point units).
For more details on the order and contents of data items of the areas marked by "✻" shown in the control procedure diagram, see Section 9.1.2.

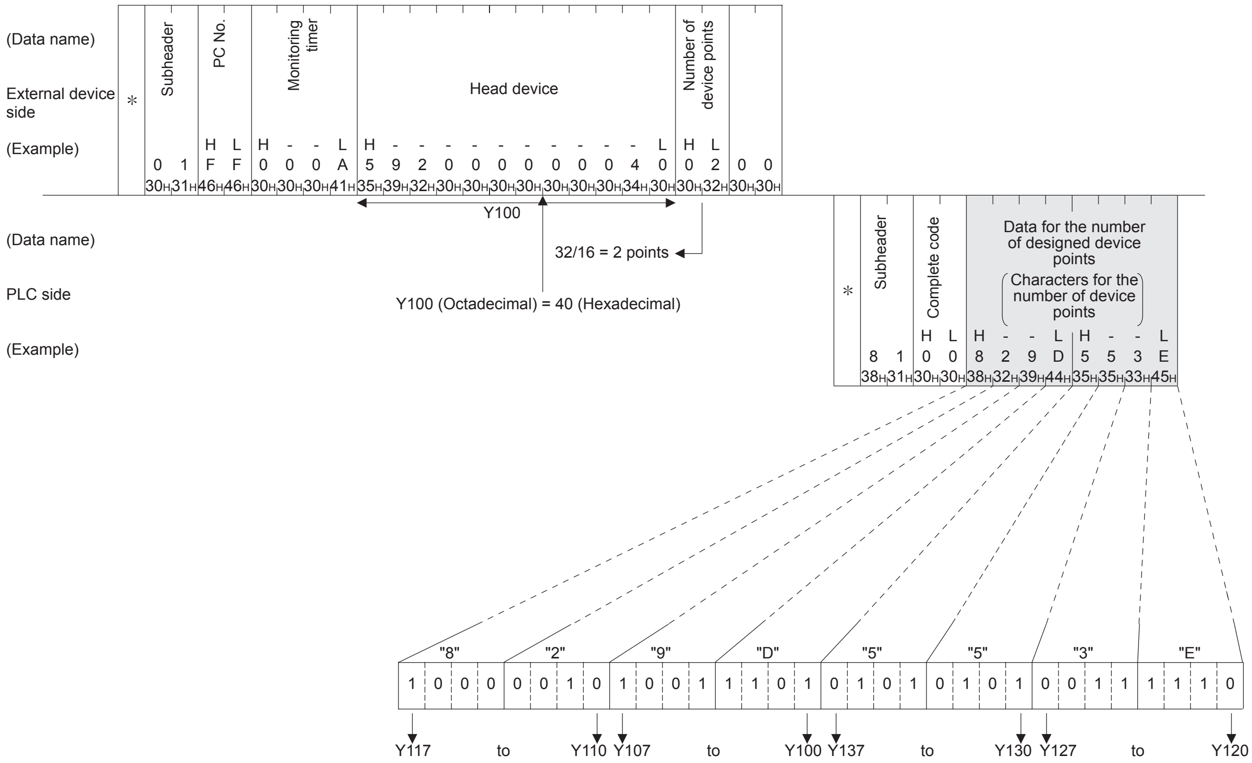
[Control procedure]

Reading the on/off status of Y100 to 137 (32 points) of the PLC on which the Ethernet module is loaded.

(1) Communication in binary code



(2) Communication in ASCII code

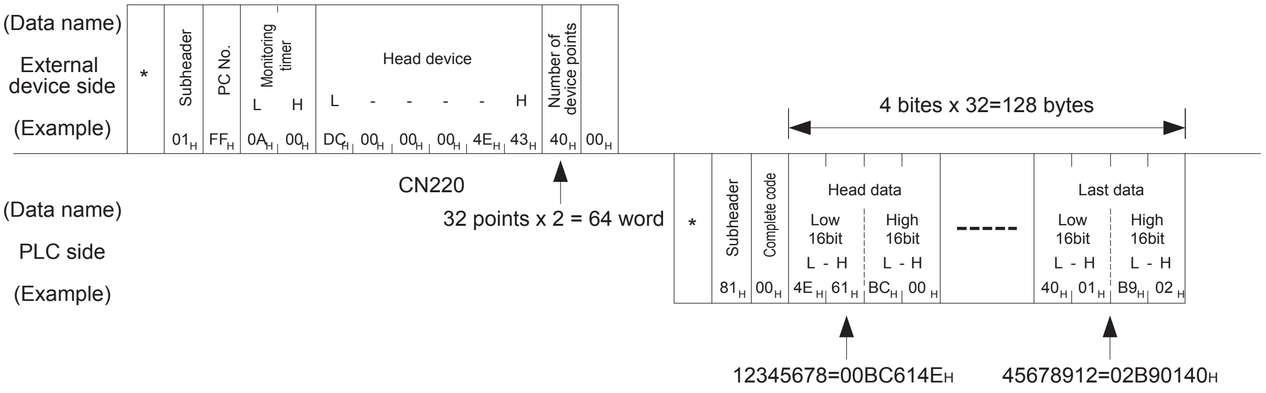

(Data name)
External device side
(Example)
*
Subheader
PC No.
Monitoring timer
Head device
Number of device points
H L H - - L H - - - - - - - - - - - L H L
0 1 F F 0 0 0 A 5 9 2 0 0 0 0 0 0 0 4 0 0 2 0 0
30H 31H 46H 46H 30H 30H 30H 41H 35H 39H 32H 30H 30H 30H 30H 30H 30H 30H 34H 30H 30H 32H 30H 30H
Y100
32/16 = 2 points
Y100 (Octadecimal) = 40 (Hexadecimal)
(Data name)
PLC side
(Example)
*
Subheader
Complete code
Data for the number of designed device points
(Characters for the number of device points)
H L H - - L H - - L
8 1 0 0 8 2 9 D 5 5 3 E
38H 31H 30H 30H 38H 32H 39H 44H 35H 35H 33H 45H
"8" "2" "9" "D" "5" "5" "3" "E"
1 0 0 0 0 0 1 0 1 0 0 1 1 1 0 1 0 1 0 1 0 1 0 1 0 0 1 1 1 1 1 0
Y117 to Y110 Y107 to Y100 Y137 to Y130 Y127 to Y120

NOTE

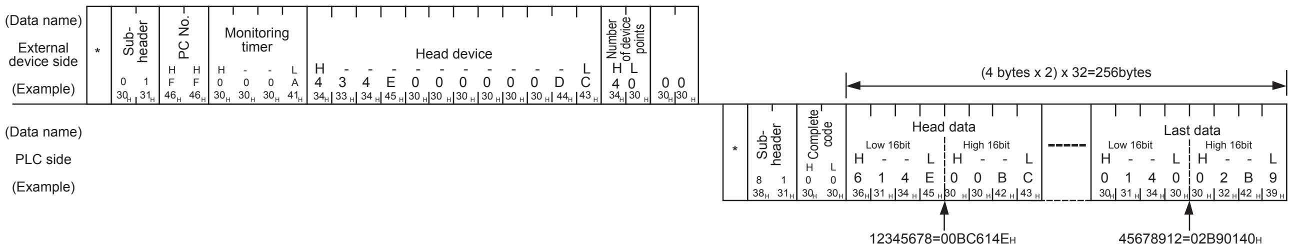
C200 to C255 (Current Values) Batch Read
Since C200 to C255 are 32 bit devices, the device points should be specified as twice as much as the points actually read.
Also, make sure that the device points are specified by an even number.
If they are specified by an odd number, the batch read cannot be performed. (Error code 57H will be stored)

Example) When 32 devices are read at one access from C220 to 251 in the PLC on which the Ethernet module is mounted.

(1) When communicating in binary code



(2) When communicating in ASCII code



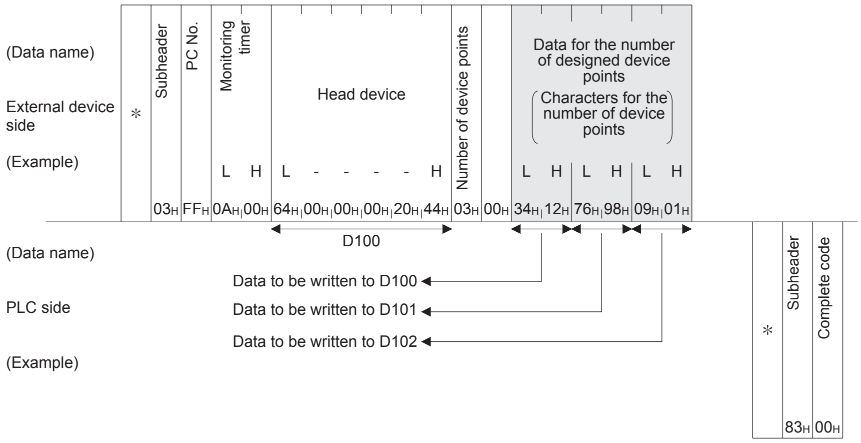
9.3.6 Batch write in word units (command: 03)

The examples shown in this section explain the command/response format when batch writing to a word device memory and bit device memory (16 point units).
For more details on the order and contents of data items of the areas marked by "✻" shown in the control procedure diagram, see Section 9.1.2.

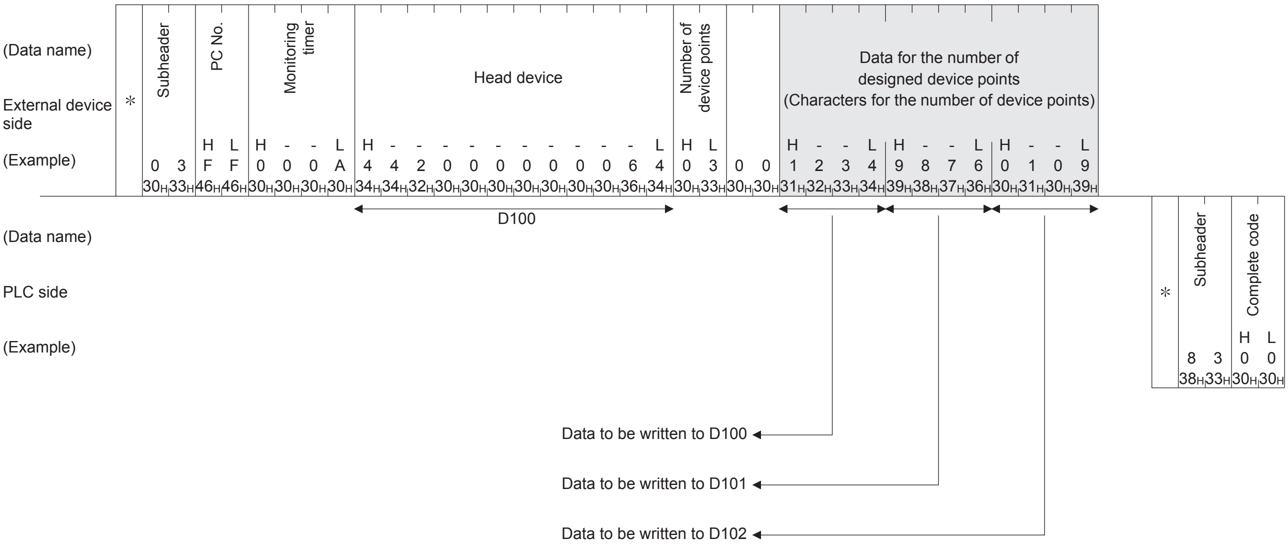
[Control procedure]

Writing data to D100 to 102 of the PLC on which the Ethernet module is loaded.

(1) Communication in binary code


(Data name)
External device side
(Example)
✻
Subheader
PC No.
Monitoring timer
Head device
Number of device points
Data for the number of designed device points
(Characters for the number of device points)
L H L - - - - H L H L H L H
03H FFH 0AH 00H 64H 00H 00H 00H 20H 44H 03H 00H 34H 12H 76H 98H 09H 01H
D100
(Data name)
PLC side
(Example)
Data to be written to D100
Data to be written to D101
Data to be written to D102
✻
Subheader
Complete code
83H 00H

(2) Communication in ASCII code

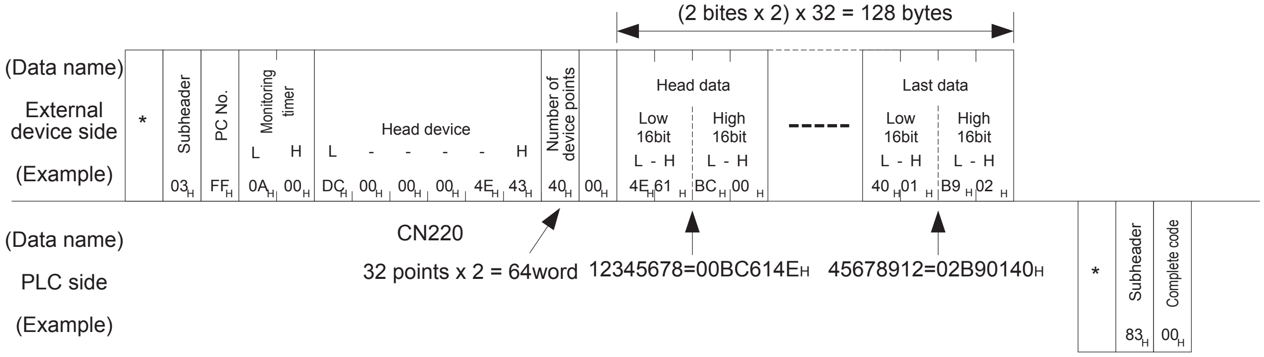


NOTE

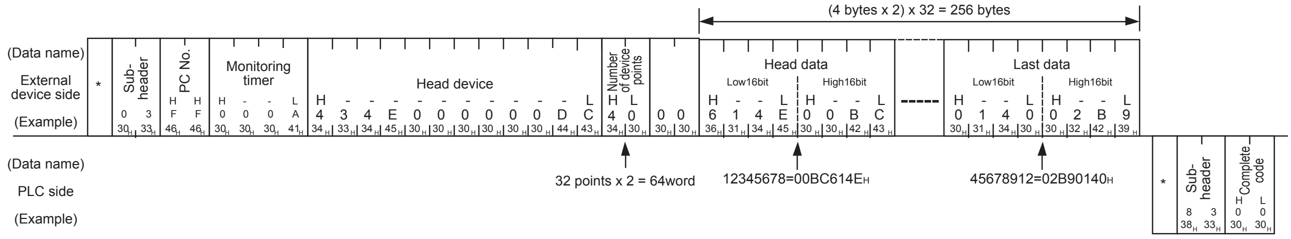
C200 to C255 (Current Values) Batch Write
Since C200 toC255 are 32 bit devices, the device points should be specified twice as much as the points actually read.
Also, make sure that the device points are specified by even number.
If they are specified by odd number, the batch read cannot be performed. (Error code 57H will be stored)

Example) When 32 points are written at one access from C220 to 251 in the PLC on which the Ethernet module is mounted.

(1) When communicating in binary code



(2) When communicating in ASCII code



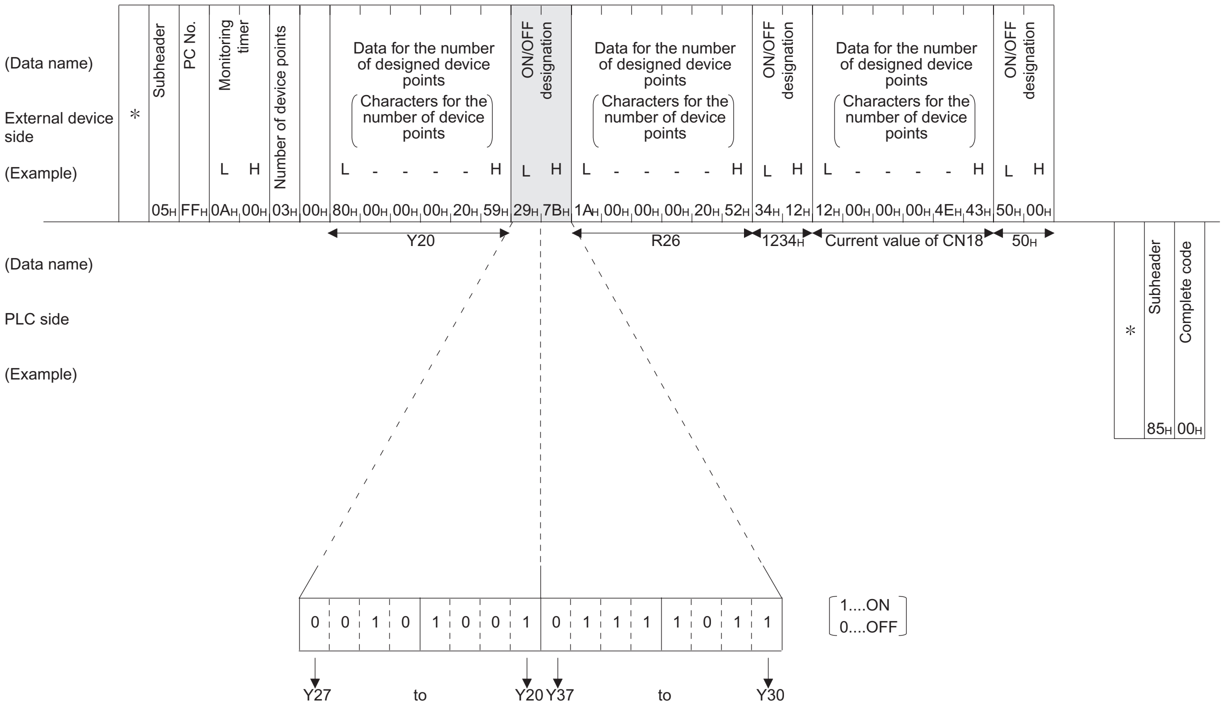
9.3.7 Test in word units (random write) (command: 05)

The examples shown in this section explain the command/response format when writing data by designating word device memories and bit device memories (16 point units) arbitrarily.
For more details on the order and contents of data items of the areas marked by "✻" shown in the control procedure diagram, see Section 9.1.2.

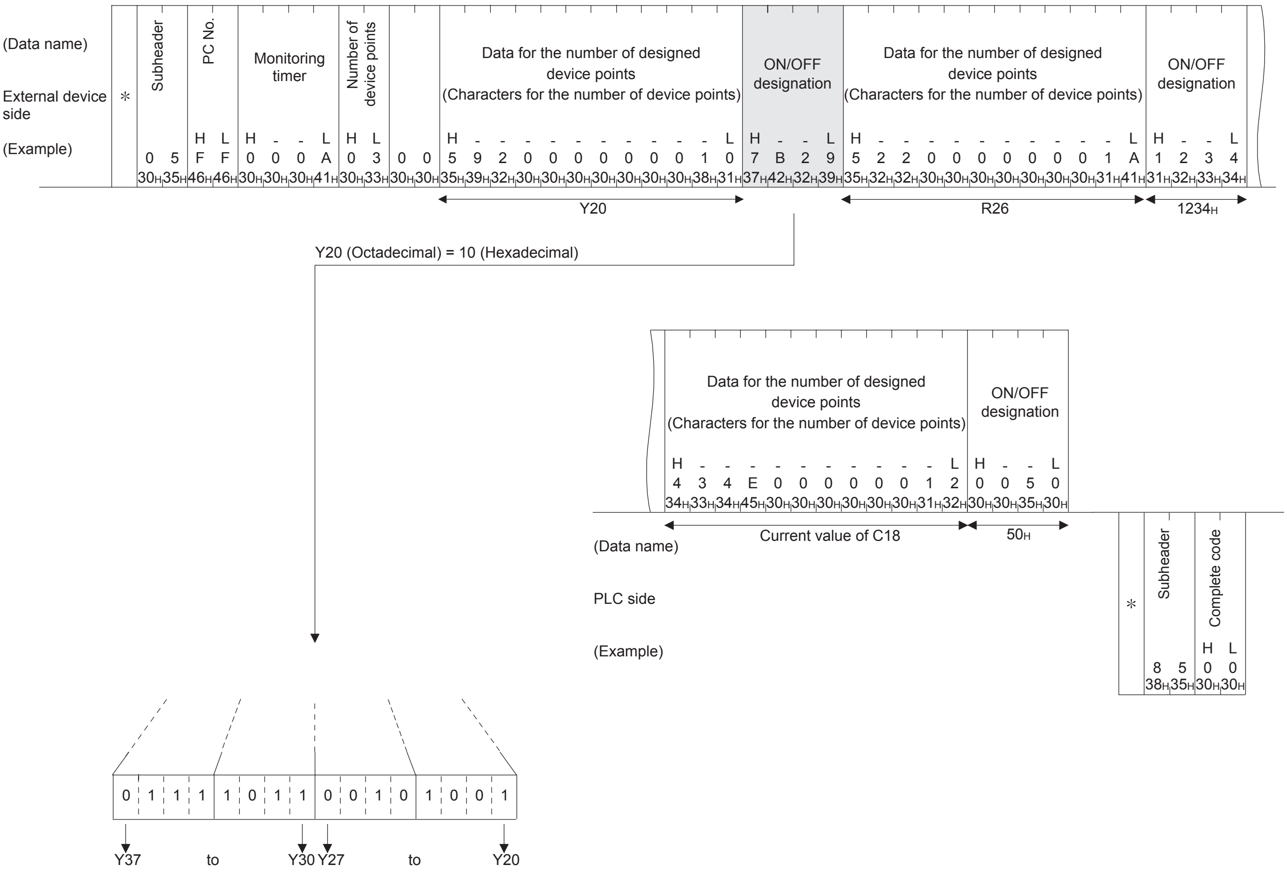
[Control procedure]

Specifying Y20 to 37 to on/off, R26 to "1234H," and the current value of C18 to "50H" at the PLC on which the Ethernet module is loaded.

(1) Communication in binary code



(2) Communication in ASCII code


(Data name)
External device side
(Example)
*
Subheader
PC No.
Monitoring timer
Number of device points
Data for the number of designed device points (Characters for the number of device points)
ON/OFF designation
Data for the number of designed device points (Characters for the number of device points)
ON/OFF designation
H L H - - L H L H - - - - - - - - - - L H - - L H - - - - - - - - - - L H - - L
0 5 F F 0 0 0 A 0 3 0 0 5 9 2 0 0 0 0 0 0 0 1 0 7 B 2 9 5 2 2 0 0 0 0 0 0 0 1 A 1 2 3 4
30H 35H 46H 46H 30H 30H 30H 41H 30H 33H 30H 30H 35H 39H 32H 30H 30H 30H 30H 30H 30H 30H 38H 31H 37H 42H 32H 39H 35H 32H 32H 30H 30H 30H 30H 30H 30H 30H 31H 41H 31H 32H 33H 34H
Y20
R26
1234H
Y20 (Octadecimal) = 10 (Hexadecimal)
Data for the number of designed device points (Characters for the number of device points)
ON/OFF designation
H - - - - - - - - - - L H - - L
4 3 4 E 0 0 0 0 0 0 1 2 0 0 5 0
34H 33H 34H 45H 30H 30H 30H 30H 30H 30H 31H 32H 30H 30H 35H 30H
Current value of C18
50H
(Data name)
PLC side
(Example)
*
Subheader
Complete code
H L
8 5 0 0
38H 35H 30H 30H
0 1 1 1 1 0 1 1 0 0 1 0 1 0 0 1
Y37
to
Y30
Y27
to
Y20

9.4 Remote RUN/STOP, PLC model name code read

This function is used to remotely RUN/STOP a PLC and read the model name of a PLC from an external device.

9.4.1 Commands and control contents

This section describes the commands and control contents of the control procedure when controlling the status of the PLC.

(1) Commands

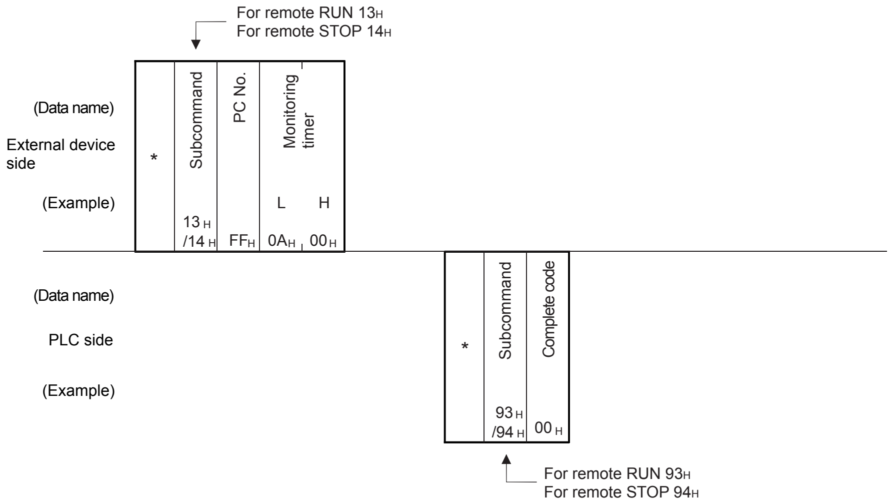
Function	Command / Response type	Processing
Remote RUN	13_H	Requests remote RUN (execute operation) Executes a forced RUN regardless of the RUN/STOP switch status of PLC.
Remote STOP	14_H	Requests remote STOP (stop operation) Executes a forced STOP regardless of the RUN/STOP switch status of PLC.
PLC model name read	15_H	Request to read the model name of the PLC.

9.4.2 Remote RUN (command: 13) / Remote STOP(Command: 14)

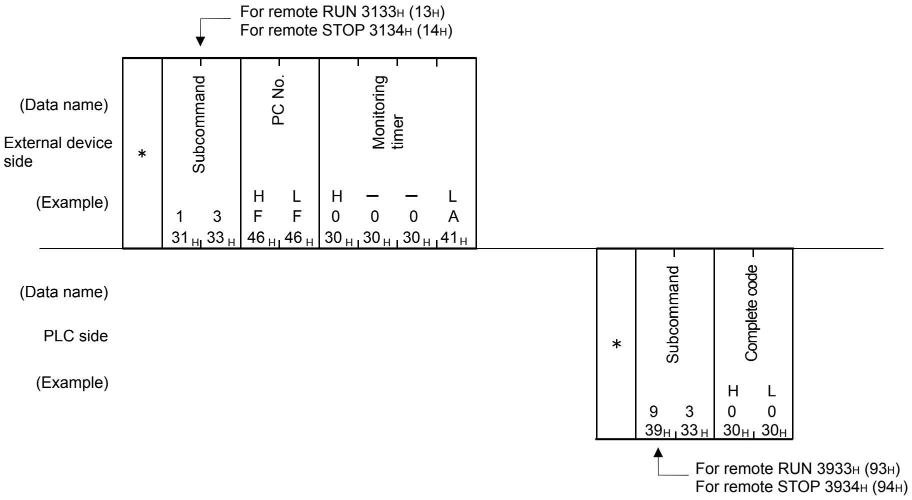
The examples shown in this section explain the control procedure of remote RUN.
The order and content of data items of the areas marked by "＊" shown in the control procedure diagram differ depending on the module used as well as the frame and format used for communication.
See the detailed information described in Section 9.1.2.

[Control procedure]

(1) Executing remote RUN while communicating in binary code



(2) Executing remote RUN while communicating in ASCII code



POINT

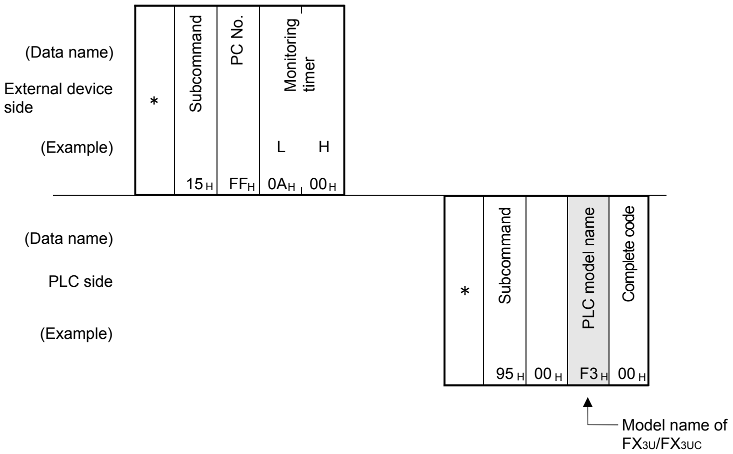
- When executing the remote RUN/STOP using other external devices or computer link, the remote RUN/STOP is executed in the order of execution.
- When the power of PLC is turned on, then off, and then on again with the remote RUN/STOP executed, the remote RUN/STOP status is canceled and the setting of the PLC's RUN/STOP switch becomes valid.

9.4.3 PLC model name read (command: 15)

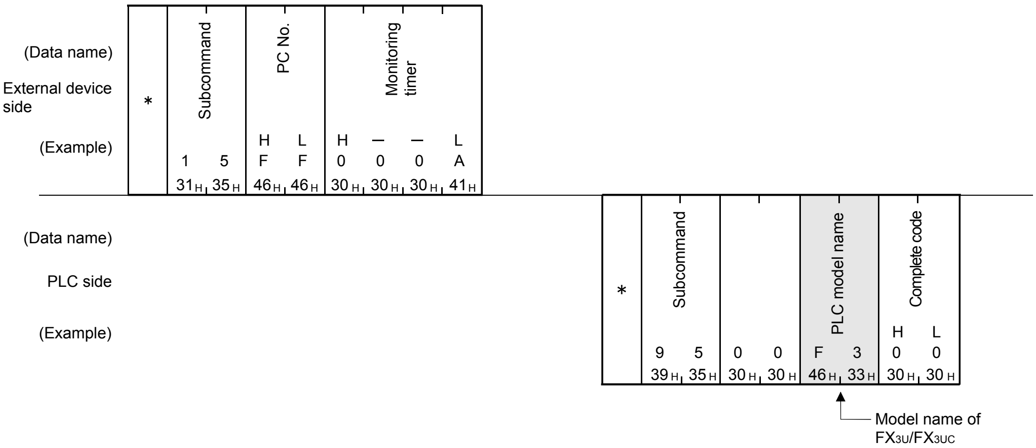
The examples shown in this section explains the control procedure for reading the model name of the PLC.
The order and content of data items of the areas marked by "*" shown in the control procedure diagram differ depending on the module used as well as the frame and format used for communication.
See the detailed information described in Section 9.1.2.

[Control procedure]

(1) Data communication in binary code



(2) Data communication in ASCII code



POINT

Distinguish the PLC model name by the model name code.
“F3H”, which is the model code of FX3U, is read as the PLC model name.

9.5 Loopback Test

A loopback test checks whether or not the communication function between an external device and the Ethernet module operates normally. The examples show the control procedure using this function.

POINT

(1) A loopback test checks whether or not the connection between the external device and the Ethernet module is correct and that the data communication function operates properly when the FX3U-ENET is started up or when a problem occurs.

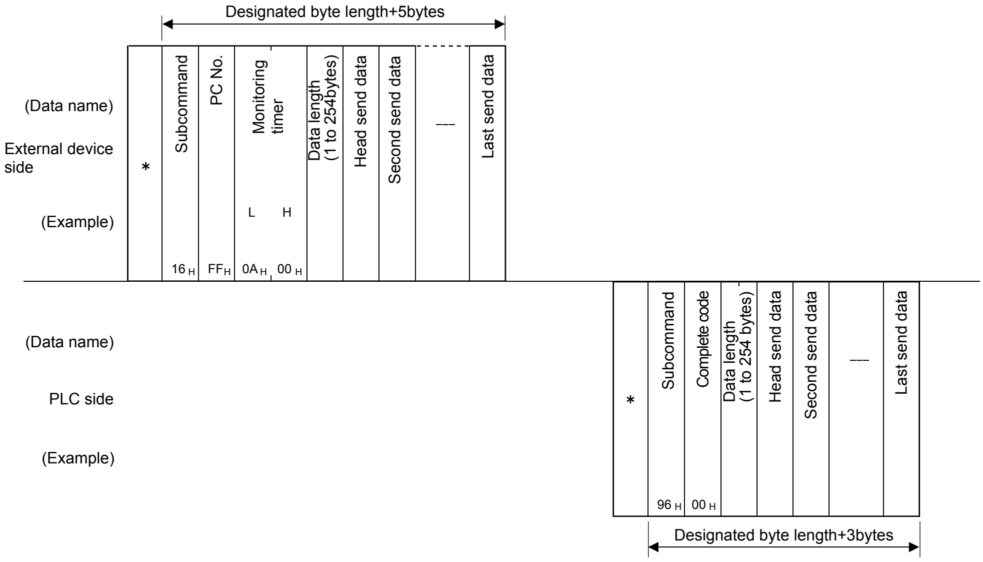
(2) Use the following data for loopback tests.
Send the head part first.
・When communicating in binary code
Maximum of 254 bytes of numerical values (00H to FFH).
・When communicating in ASCII code
Maximum of 254 single-byte characters (“0” to “9”, “A” to “F”).

9.5.1 Loopback test (command: 16)

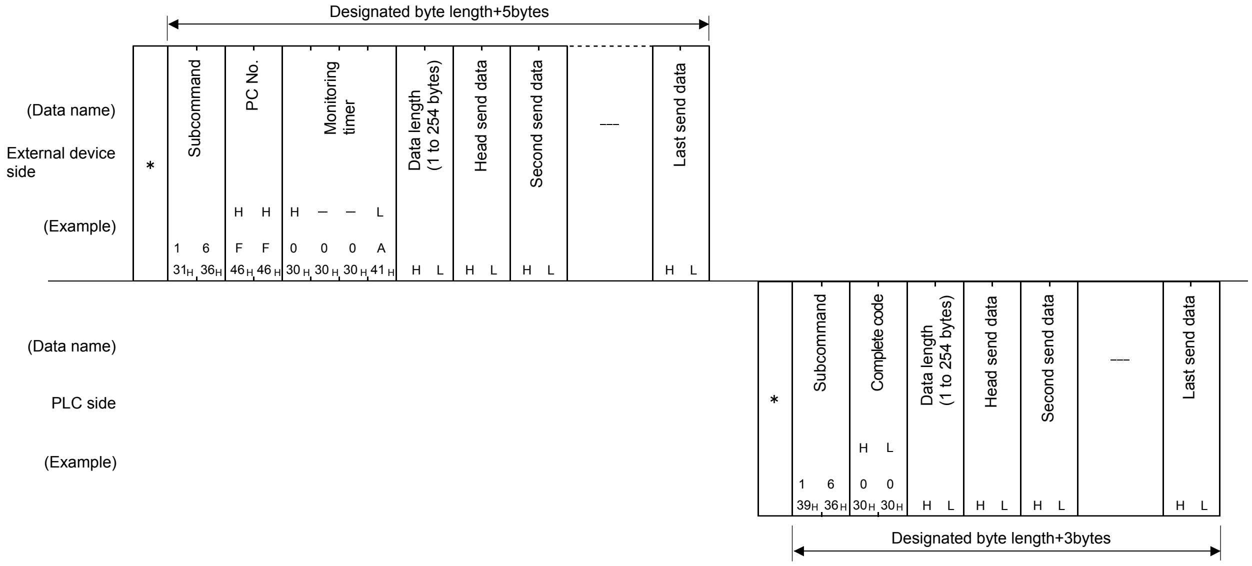
The examples shown in this section explain the control procedure for an external device performing a loopback test with the Ethernet unit.
The order and content of data items of the areas marked by "✻" shown in the control procedure diagram differ depending on the module used as well as the frame and format used for communication.
See the detailed information described in Section 9.1.2.

[Control procedure]

(1) Performing a loopback test while communicating in binary code



(2) Performing a loopback test while communicating in ASCII code



POINT

The number of the loopback data transmitted by an external device is returned to the external device it is.

10 USING THE E-MAIL FUNCTION

This chapter explains e-mail functions of the Ethernet module.

10.1 E-mail Function

The e-mail function sends and receives e-mail to/from PCs or PLC in remote locations via the network.
Executed by operating the buffer memory in a sequence program.

10.1.1 E-mail send and reception by the PLC

(1) Sending/receiving e-mail by the PLC
In sending/receiving e-mail by the PLC, the PLC information of an Ethernet module installed station is sent to another Ethernet module or an external device such as a PC in the main text or attached file of an e-mail. E-mail sent from other Ethernet modules and PCs can also be received.

Sending e-mail by the PLC		External device			Remark
		Ethernet module	PC	Portable terminal (device that cannot handle attached files)	
Subject		○ (∗1)	○	○	—
Attached file	Binary format	○	○	×	Maximum 2k words
	Binary to ASCII conversion	×			
	Binary to CSV conversion				
Main text	Designating using ASCII data	×	○	○	Maximum 256 words.

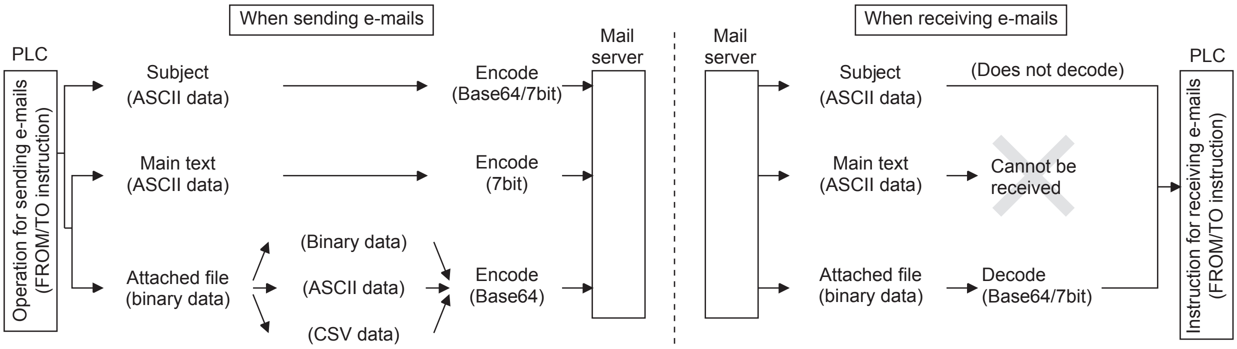
∗1 The Ethernet module receives the Subject without decoding.

(a) Sending e-mail from the PLC
Data is transmitted via the main text or attached file of an e-mail.
1) For an attached file, device data and other information is sent by converting into binary/ASCII/CSV data format.
2) For the main text, ASCII code data (character string data) created by a sequence program is sent.

(b) Receiving e-mail to the PLC
Receives attached file data of e-mail. The main text data of an e-mail cannot be received.

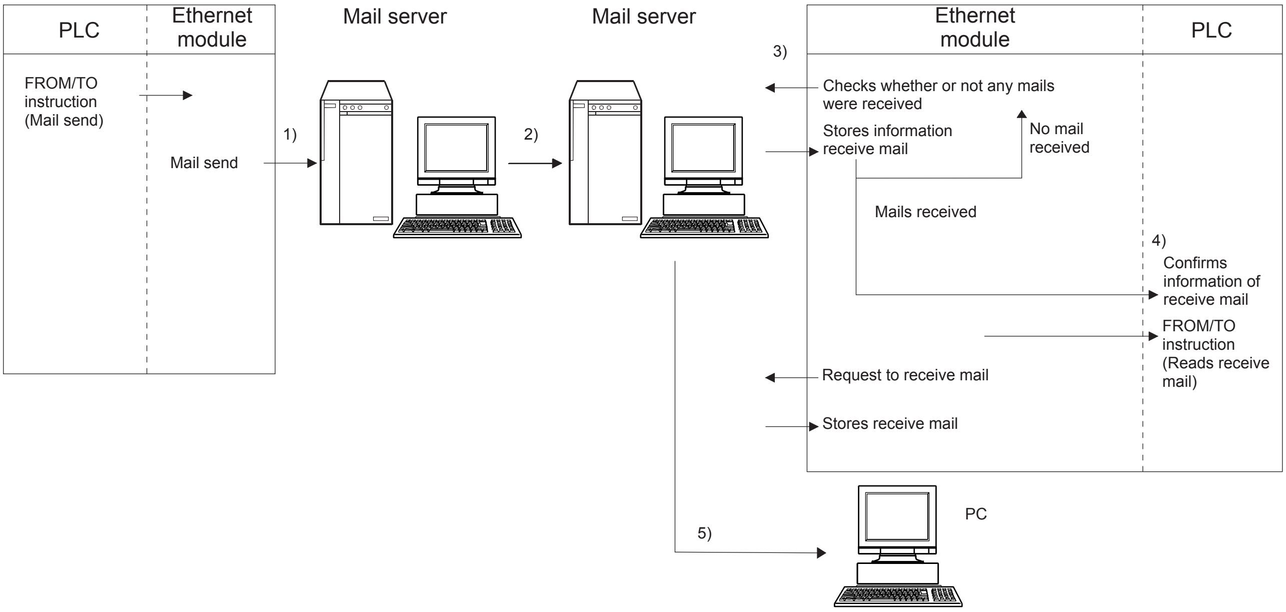
(2) Data conversion method used by the Ethernet module

The following shows the data conversion method used by the Ethernet module.



(3) Flow of the e-mail

The following diagram illustrates the flow of e-mail sent by the PLC to the external device:


PLC
Ethernet module
FROM/TO instruction (Mail send)
Mail send
1)
Mail server
2)
Mail server
3)
Ethernet module
PLC
Checks whether or not any mails were received
Stores information receive mail
No mail received
Mails received
4)
Confirms information of receive mail
FROM/TO instruction (Reads receive mail)
Request to receive mail
Stores receive mail
5)
PC

1) Send an e-mail to the mail server of the local station's Ethernet module.
2) The mail server of the local station's Ethernet module sends the e-mail to the receive mail server of the send destination device.

[When receiving by the Ethernet module]

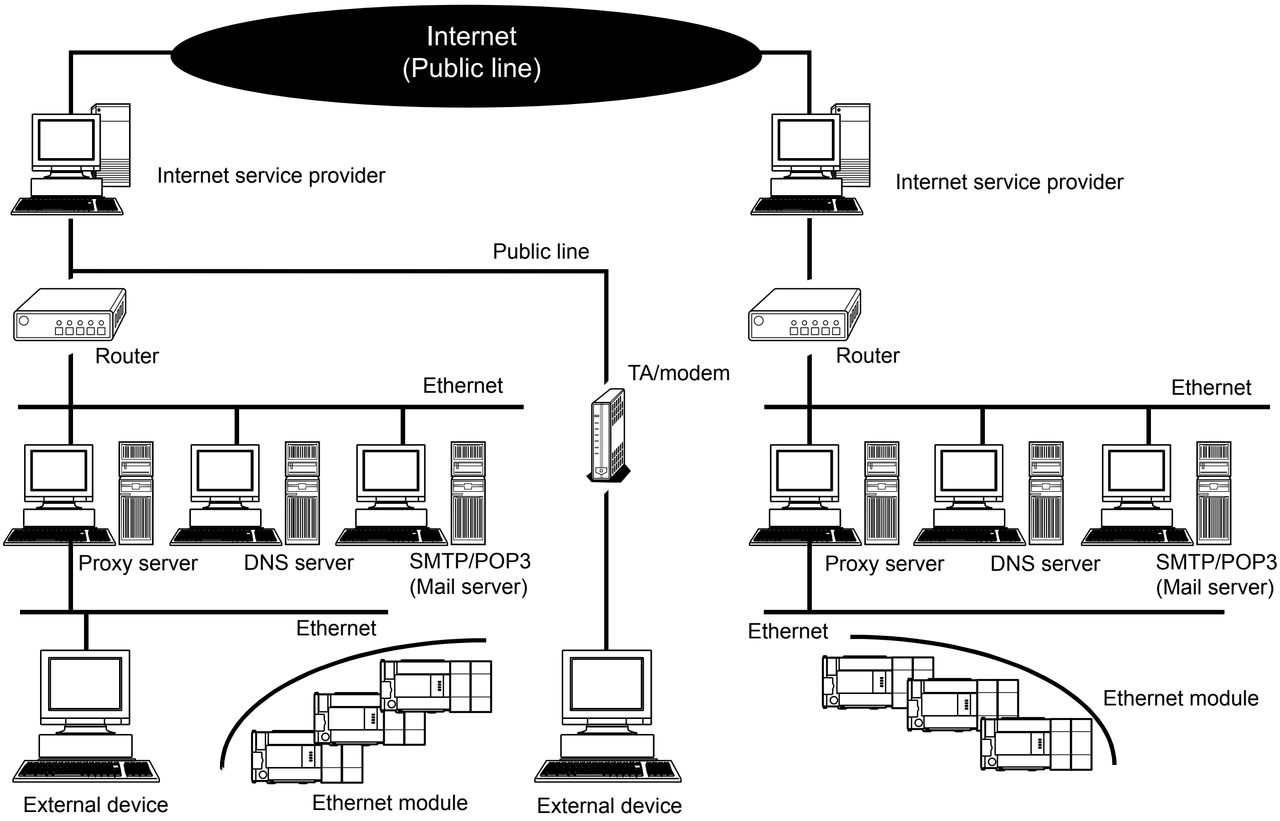
3) The Ethernet module confirms mail receiving to be active or not for the time set in the parameters, and stores the results in the buffer memory.
4) By executing the mail receiving command issued by the FROM/TO instruction, the Ethernet module reads e-mails from the e-mail server to the mail buffer memory.
   ＊ When receiving e-mails, the Ethernet module reads the number of remaining receive mails on the mail server and stores this number in the buffer memory.

[When receiving by a PC]

5) Mails can be received using applications such as e-mail software.

10.2 Configuration and Environment of the Applicable System

This section explains the environment and system configuration for using the e-mail function from the Ethernet module.



(1) E-mails can be sent and received in an environment where the Internet service is available as shown above.

(2) Set each Internet mail setting value for the Ethernet module and PC following the directions from the system administrator.

(3) Each of the Ethernet module parameters for using the e-mail function is set with FX Configurator-EN (GX Developer) or a sequence program.

10.3 Precautions for Using the E-mail Function

The following explains precautions when using the e-mail function of the Ethernet module.

(1) Precautions regarding the system

(a) Design the system so that the entire system operates normally at all times when sending/receiving e-mails to/from a running system and controlling the PLC.

(b) In order to avoid PLC system malfunctions caused by receptions of illegal e-mails from the outside sources, take precautions in preventing illegal e-mails from being received on the mail served on the Ethernet module side (using an anti-virus program, etc).

(2) Precautions regarding the external device

(a) E-mails can be sent to PLC stations in which Q series or FX3U series Ethernet modules are installed.

(b) To send files to the Ethernet module as attachments to e-mail, specify the encoding method (Base 64/7 bits/Quoted Printable) of the attached files.

(3) Common precautions

(a) The Ethernet module stores the data of attached files received from the external device in the device specified by the FROM/TO instruction without converting it from ASCII to binary.

(b) The maximum sizes of data that can be sent/received by the Ethernet module are as follows:

- Data size of attached files: Up to 2048 words
- Data size of main text: Up to 256 words (Sending only)

(c) Mail that is sent and received do not support encrypted data, compressed data, and data conversion.

(d) If the external device cannot be found when sending an e-mail from the Ethernet module, the error code can be checked through the reception processing with the MRECV instruction.

＊ If an external device cannot be found when sending e-mail using the operation of buffer memory, failure of e-mail transmission may not be recognized, depending on the operation of the mail server. Be sure to verify the receiver's e-mail address in advance.

＊ Error codes are stored within BFM#10335. (For details on error codes, refer to section 11.4.)

(e) The e-mail function is supported for SMTP and POP3 servers.

(f) When e-mail communication errors occur, check the error codes stored in the mail send/reception error log area of the buffer memory.

(g) If e-mails cannot be received, try one of the following.
  1) Execute the mail receive instruction (BFM#14500:0001H) once.
  2) Shorten the "Inquiry interval" time (BFM#9904) in the e-mail settings.
  3) Check the number of incoming mails remaining on the mail server.
     (Can be checked with BFM#22640.)

(h) When the receive data of an e-mail sent is abnormal (garbled characters, etc.), review the transmission data format (binary/ASCII/CSV) designated with BFM#10338.
  * The Subject should be in ASCII code data format. (It is not converted to ASCII format.)

(i) Confirm with a system administrator regarding the minimum time intervals for accessing the mail server when sending e-mail, reading received e-mail, and inquiring whether or not there is any received e-mail.
Depending on the mail server's security setting, frequent access may be prohibited.

10.4 E-mail Specifications

The following table lists the specifications of the e-mail function.

Item			Specification
Transmission specifications Transmission and reception data	Data size	Attached file	2048 words × 1
		Main text	256 words × 1
	Data transfer method		When sending: Sends either a file as attachment or main text (select one). When receiving: Receives file as attachment.
	Subject		US-ASCII format or ISO-2022-JP (Base 64)
	Attached file format		MIME format
	MIME		Version 1.0
	Data of attached file format		Binary/ASCII/CSV can be selected. File name: XXXX.bin (binary), XXXX.asc (ASCII), XXXX.csv (CSV) (CSV: Comma Separated Value)
	Division of attached file		Cannot be divided (only one file can be sent/received) ＊ If any divided files are received, only the first file will be received and the remaining files will be discarded.
	When sending (encode)		Subject: Base 64/7 bits Main text: 7 bits Attached file: Base 64
	When receiving (decode)		Subject: (Does not decode) Main text: (Cannot be received) Attached file: Base 64/7 bits/Quoted Printable ＊ If e-mail is sent from the external device to the PLC side, specify the encoding method (Base 64/7 bits/Quoted Printable) of the attached file.
	Encryption		No
	Compression		No
	Communication with mail server		SMTP (sending server) Port number = 25 POP3 (receiving server) Port number = 110
	Operation check mailer		Microsoft® Corporation Outlook Express 6

NOTE

Mail terminology list:

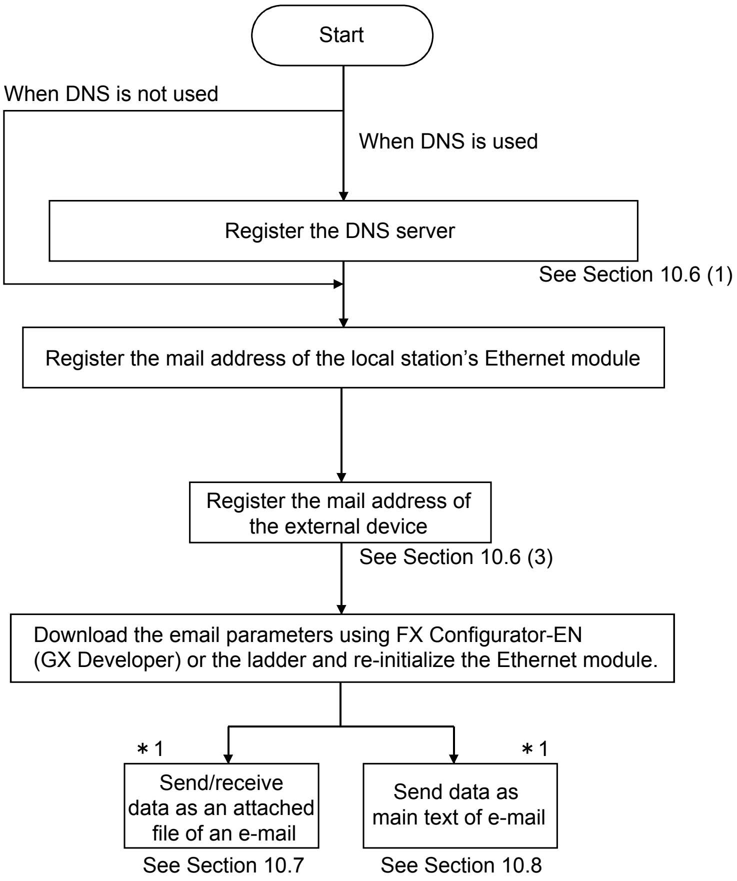
DNS server: Server that manages networks

SMTP server: Server that delivers (transfers) e-mail between mail servers

POP3 server: Server that transfers received e-mail addressed to subscribers to the corresponding subscribers

10.5 Processing Procedure of the E-mail Function

The following flowchart illustrates the procedure that is required before sending and receiving e-mails:


Start
When DNS is not used
When DNS is used
Register the DNS server
See Section 10.6 (1)
Register the mail address of the local station's Ethernet module
Register the mail address of the external device
See Section 10.6 (3)
Download the email parameters using FX Configurator-EN (GX Developer) or the ladder and re-initialize the Ethernet module.
∗1
∗1
Send/receive data as an attached file of an e-mail
See Section 10.7
Send data as main text of e-mail
See Section 10.8

∗1 The open and close processing of the communication line for sending/receiving e-mails is automatically executed by the Ethernet module.

10.6 E-mail Settings

This section explains the settings required to send and receive e-mails.

(1) DNS settings
Specify the IP address of the DNS server when setting the mail server name using the domain name from the e-mail settings (see (2)).

Setting name		BFM number (Decimal)	Description of setting	Setting range/options
DNS setting	IP address of DNS server 1	20505 to 20506	Set IP address of DNS server 1.	—
	IP address of DNS server 2	20507 to 20508	Set IP address of DNS server 2.	—
	IP address of DNS server 3	20509 to 20510	Set IP address of DNS server 3.	—
	IP address of DNS server 4	20511 to 20512	Set IP address of DNS server 4.	—

(a) DNS settings
Designate the IP addresses of the domain name servers (DNS) used by the local station's Ethernet module or designated by a system administrator.

1) Designate the IP addresses of the DNS servers 1 to 4.

POINT

(1) The DNS servers manage networks.
DNS settings are required to allow search for the SMTP server or POP3 server from a domain name.

(2) Adjust the DNS settings when the mail server name shown in item (2) is specified with a domain name.
Setting is not required when it is specified with an IP address.

(3) When obtaining the IP addresses from a domain name, the DNS servers are searched sequentially starting from the first DNS server.

NOTE

To adjust settings using FX Configurator (GX Developer), select [Initial Settings] from the initial screen and adjust the settings at the initial settings screen.

(2) E-mail settings

Set in order to use the e-mail function.

[Setting item]

Setting name			BFM number (Decimal)	Description of setting item	Setting range/option
General setting	Password		9913 to 9920	Set the password to the mail server (16 characters or fewer)	—
	Mail address		9856 to 9887	Set the mail address for the Ethernet module (64 characters or fewer)	—
	Check of mail		9904 (b15)	Select whether or not to make inquiries to the incoming mail server regarding received mail	ON:Inquire OFF: Do not inquire
	Time interval for inquiries		9904 (b14 to b0)	Set the time interval and unit for making inquiries to the incoming mail server	Check unit designation (b12 to b14) 0: Hour, 1: Minute, 2: Second Check interval (b11 to b0) For hour : 1 to 24 For minute : 1 to 1440 For second : 30 to 3600
Mail server name	Mail send setting	Specification method	9921	Select send mail server specification method	0: Domain name specification 1: Decimal 2: Hexadecimal
		STMP server name / IP address	9922 to 9953	Set the IP address of send mail server or send mail server name	00000001$_{H}$ to FFFFFFFE$_{H}$
	Mail receive setting	Specification method	9954	Select receive mail server specification method	0: Domain name specification 1: Decimal 2: Hexadecimal
		POP server name / IP address	9955 to 9986	Set the IP address of the receive mail server or receive mail server name	00000001$_{H}$ to FFFFFFFE$_{H}$

(a) General settings

Designate the mail setting values that have been registered with the Internet service provider of the local station's Ethernet module or the mail registration information designated by the system administrator.

1) Password:

Designate the mail password of the local station's Ethernet module.

2) Mail address:

Designate the mail address of the local station's Ethernet module.

3) Check for received mail:

Select whether or not to check for received mail and set the checking time interval when incoming mail is checked.

When mail check is designated, set the time interval for inquiring the mail server whether there is mail or not.

The following table shows each setting range of the inquiry time intervals for checking mail. (30 s to 24 h)

Time units	Setting range (default: 5 min)
h	1 to 24
min	1 to 1440
s	30 to 3600

POINT

If the inquiry time interval from the PLC or other module to the server is short because of the POP3 server specifications, access may be restricted (lock status) on the server side.
Check the POP3 server specifications, and set the inquiry time interval accordingly.
(It is recommended to set the value of the inquiry time interval to the default (5 minutes) or more.)

(b) Mail server name
Set the domain name or IP addresses of the mail servers designated by the system administrator, used by the local station's Ethernet module.

1) Designate the domain name of the mail server.
2) Select the input format (decimal/hexadecimal) of the IP addresses.
3) Set the IP address of the outgoing mail server (SMTP).
4) Set the IP address of the incoming mail server (POP3).

POINT

- The SMTP server delivers (transfers) e-mails between mail servers.
- The POP3 server transfers received e-mails addressed to users to the corresponding users.
- The mail server names must be designated in order to use the e-mail function. (See POINT of (1).)

NOTE

To adjust the settings in FX Configurator-EN (GX Developer), select [Email Settings] from the initial screen and adjust the settings from the email settings screen.

(3) Send mail address setting
Register the e-mail address of the external devices where e-mail is to be sent.

[Setting item]

Setting Item name	BFM number (Decimal)	Description of setting	Setting range/Selections
Send mail address 1 to Send mail address 10	9987 to 10307	Set the mail address of the transmission destination (1 to 10)	—

(a) Send mail address

1) Designate the mail addresses of a maximum of 10 external devices to which mail is being sent from the local station's Ethernet module. (Only one e-mail address can be specified for each area.)
2) When sending e-mail by the Ethernet module, specify the send e-mail address by the bit of BFM#10337.

NOTE

To adjust the settings in FX Configurator-EN (GX Developer), select [Email Settings] from the initial screen and adjust the settings from the email settings screen.

10.7 Sending/Receiving E-mail (Attached Files) by the PLC

This section explains how to send and receive attached files by e-mail.

10.7.1 When sending data as an attached file

The following explains the buffer memory addresses and program for sending e-mail. The following instruction shows how to store data in an attached file and send it to an external device.

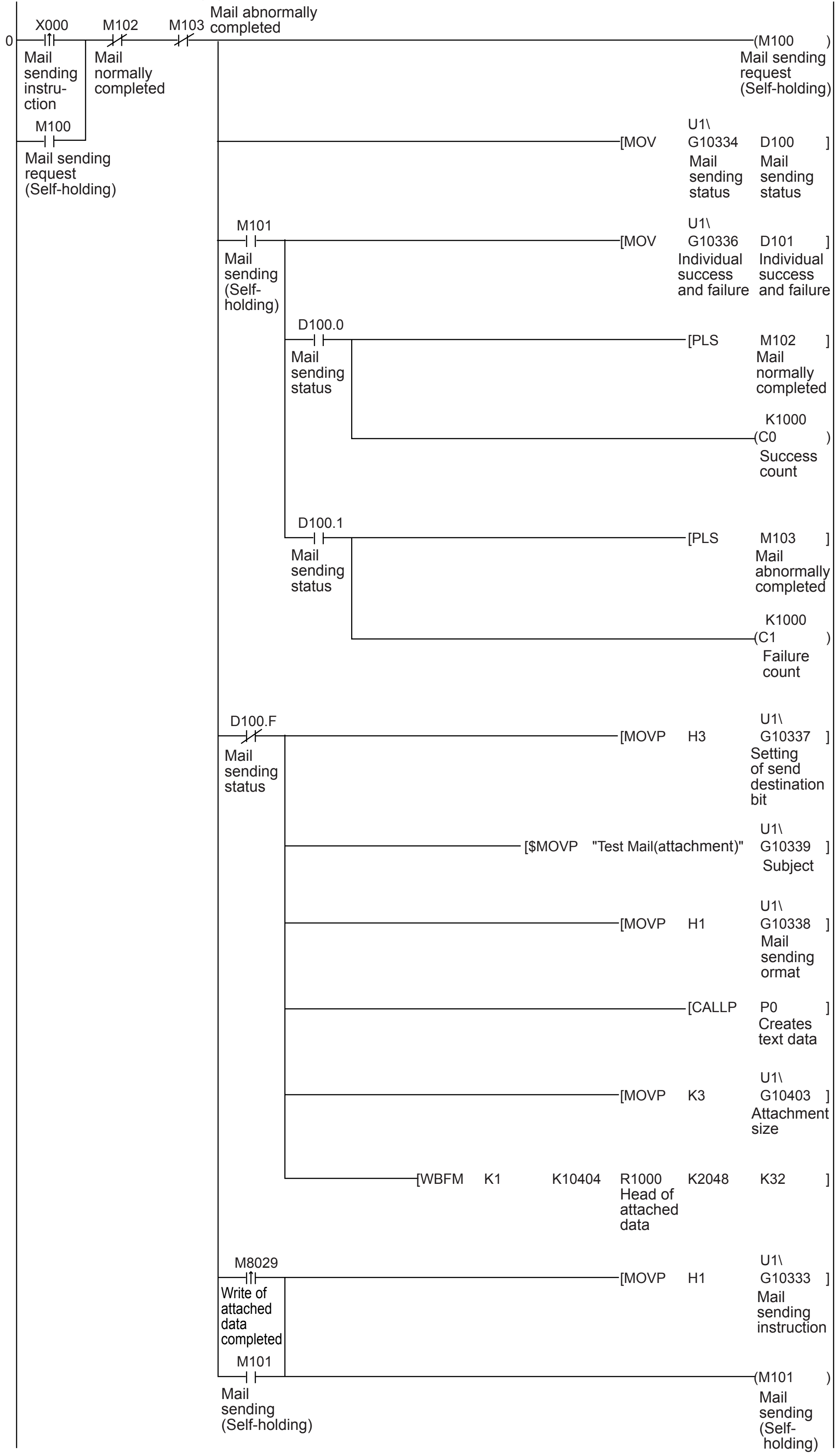
1) Check whether mail can be sent or not.
   (BFM#10334…b15: OFF)
2) Designate the destination of the mail.
   Turn the bit corresponding to the e-mail destination address to ON in BFM#10337.
   To send to all of the mail addresses registered, write to “3FF$_H$”.
3) Write the mail data to the following buffer memory addresses.
   (Subject: BFM#10339 to 10402)
   (Attached file format: BFM#10338)
   (Word count of attached file: BFM#10403)
   (Contents of main text or attached file: BFM#10404 to 12451)
4) Write “0001$_H$” in BFM#10333 to send the mail.
5) Read BFM#10336 to check if mail was correctly sent.

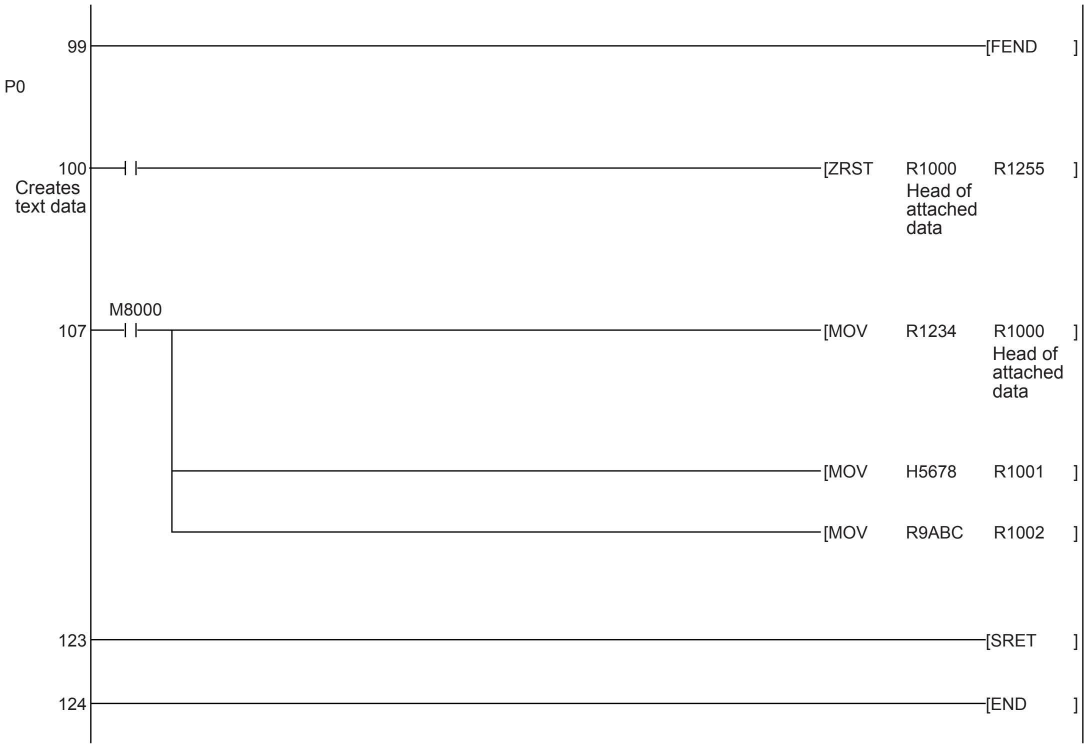
The table below shows the buffer memory to be used for sending mail.

BFM number	Items	Contents
Decimal (Hexadecimal)		
10333 (285DH)	Mail sending command	Start mail sending (0001H) Ignore (0000H)
10334 (285EH)	Mail sending status	Initial status (0000H) Mail sending normally completed (0001H) Mail sending abnormally completed (0002H) Mail being sent (8000H)
10336 (2860H)	Result of sending e-mail	Sending destination address 1 Success (b0: ON) / Failure (b0: OFF) to Sending destination address 10 Success (b9: ON) / Failure (b9: OFF)
10337 (2861H)	Designation Address of e-mail	Turns the bit of the e-mail destination address to ON. Sending destination address 1 Enable (b0: ON) / Disable (b0: OFF) to Sending destination address 10 Enable (b9: ON) / Disable (b9: OFF)
10338 (2862H)	Attached file format	Sends the attached file in binary format (0000H) Sends the attached file in ASCII format (0001H) Sends the attached file in CSV format (0002H) Sends the attached file as a text mail (0004H)
10339 to 10402 (2863 to 28A2H)	Subject	Subject consists of the send destination number and the character data designated by the sequence program. (It is not converted to ASCII format.) * The Subject should be designated in ASCII characters in the sequence program. * When the subject consists of less than 128 characters, write "\0" at the end of the subject
10403 (28A3H)	Word count of attached file	Designates the number of words of main text/attached file (0 to 2048)
10404 to 12451 (28A4 to 30A3H)	Main text or attached file	Stores a main text or an attached file

(1) Send using the sequence program

The following explains how an e-mail is sent by the sequence program. See (3) sending data for detail.

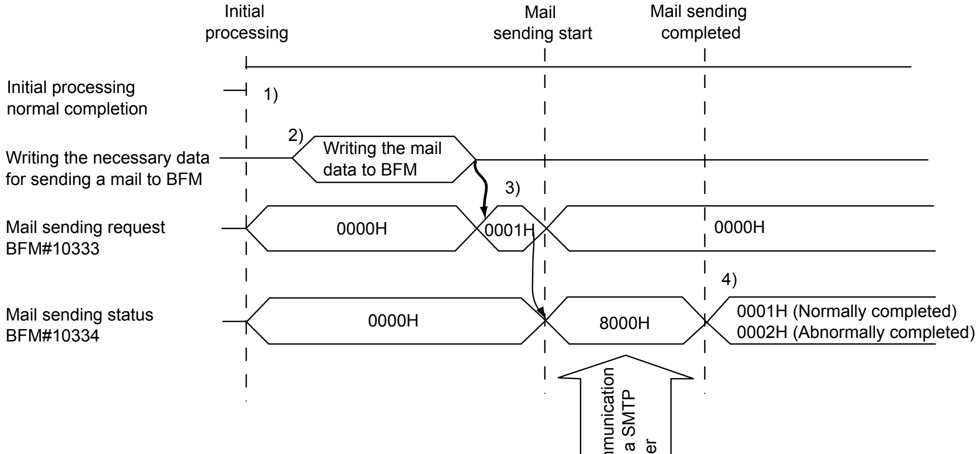

0
X000 Mail sending instru-ction
M102 Mail normally completed
M103 Mail abnormally completed
(M100) Mail sending request (Self-holding)
M100 Mail sending request (Self-holding)
[MOV U1\G10334 Mail sending status D100 Mail sending status]
M101 Mail sending (Self-holding)
[MOV U1\G10336 Individual success and failure D101 Individual success and failure]
D100.0 Mail sending status
[PLS M102 Mail normally completed]
K1000 (C0) Success count
D100.1 Mail sending status
[PLS M103 Mail abnormally completed]
K1000 (C1) Failure count
D100.F Mail sending status
[MOVP H3 U1\G10337 Setting of send destination bit]
[$MOVP "Test Mail(attachment)" U1\G10339 Subject]
[MOVP H1 U1\G10338 Mail sending ormat]
[CALLP P0 Creates text data]
[MOVP K3 U1\G10403 Attachment size]
[WBFM K1 K10404 R1000 Head of attached data K2048 K32]
M8029 Write of attached data completed
[MOVP H1 U1\G10333 Mail sending instruction]
M101 Mail sending (Self-holding)
(M101) Mail sending (Self-holding)



NOTE

When an error occurs while sending e-mail from the sending mail server to an external device, an error code is stored in the receiving error log area.

(2) Execution timing of the instruction


Initial processing
Mail sending start
Mail sending completed
Initial processing normal completion
1)
2)
Writing the necessary data for sending a mail to BFM
Writing the mail data to BFM
3)
Mail sending request BFM#10333
0000H
0001H
0000H
4)
Mail sending status BFM#10334
0000H
8000H
0001H (Normally completed)
0002H (Abnormally completed)
Communication with a SMTP server

1) Check whether the initial processing has been completed normally.

2) Write mail data to the buffer memory.
(Mail sending destination: BFM#10337…b0 to b9)
(Subject: BFM#10339 to 10402)
(Attached file format: BFM#10338)
(Word count of attached file: BFM#10403)
(Contents of main texts or attached files: BFM#10404 to 12451)

3) Write "0001$_H$" in BFM#10333.
The e-mail is sent to the mail server.

4) When sending to the mail server is complete, the result is stored in BFM#10334.
＊For details on causes of errors, see "Troubleshooting" in Section 11.

＊1 Specify the attached file format by writing a parameter in BFM#10338 before sending e-mail.

0000H: Sends the attached file in binary format
0001H: Sends the attached file in ASCII format
0002H: Sends the attached file in CSV format
0004H: Sends the attached file as a text mail

(a) Designate binary format when communicating between Ethernet modules.
The name of the attached file is mmddhhss.bin (system date.bin).
mm: Month dd: Date hh: Hour ss: minute

(b) Designate either binary format, ASCII format or CSV format when a PC/UNIX receives e-mail sent from an Ethernet module.

1) When binary format is designated
The name of the attached file is mmddhhss.bin (system date.bin).
mm: month dd: day hh: hour ss: minute

2) When ASCII format is designated
The name of the attached file is mmddhhss.asc (system date.asc).
mm: month dd: day hh: hour ss: minute

3) When CSV format is designated
The name of the attached file is mmddhhss.csv (system date.csv).
mm: month dd: day hh: hour ss: minute

＊ For details on how to receive e-mails, see (3).

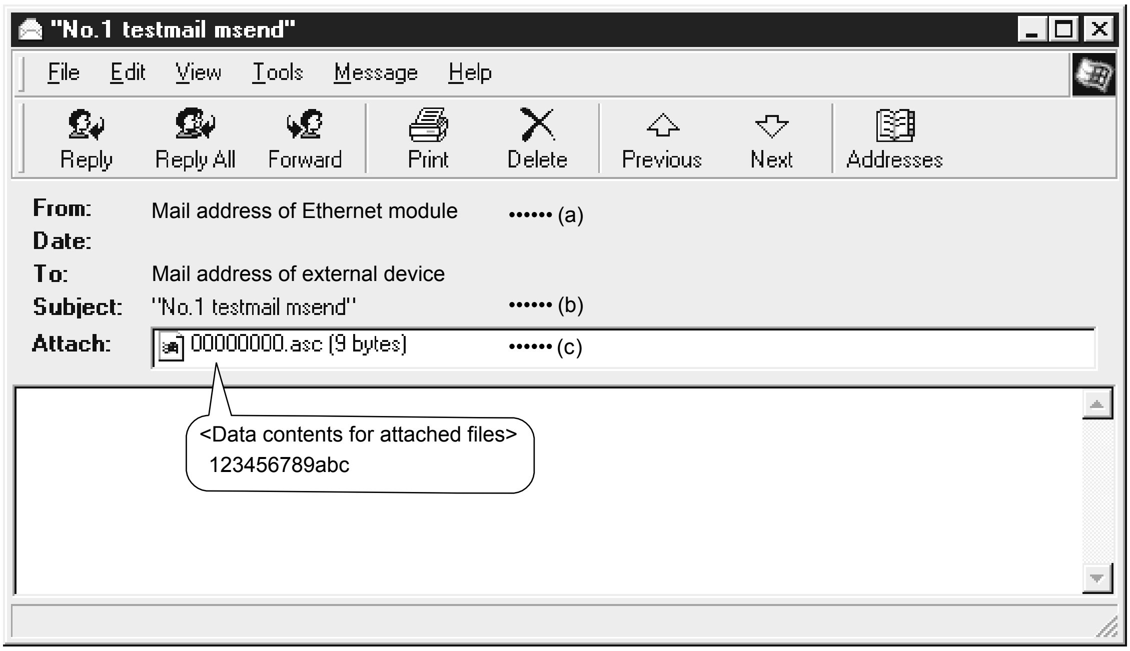
＊ The system date for the file name is the data managed by the PLC in which the Ethernet module is installed.

(3) Receiving with a personal computer

The following explains how a PC/UNIX receives e-mails sent from the Ethernet module.

After setting an environment for using the Internet with application software that processes e-mails, e-mails can be received on a receive message screen as shown below.

(Example) The following shows an example of the received e-mail. (The screen display shows Microsoft® Corporation's Outlook Express 6.)



From: Mail address of Ethernet module ······ (a)
Date:
To: Mail address of external device
Subject: "No.1 testmail msend" ······ (b)
Attach: 00000000.asc (9 bytes) ······ (c)

<Data contents for attached files>
123456789abc

(a) From

Displays the mail address of the Ethernet module.

NOTE

By entering the mail addresses of the send destination devices in the address book, the registered names (display names) can be displayed and makes management easier.

(b) Subject

This indicates the transmission destination No. (*1) and the Subject written by a sequence program.

*1 The Ethernet module automatically adds this number, sequentially starting from 1.
(After the maximum number (99999) is reached, the number starts again from 0.)

(c) Attached file

This indicates the name of the attached file sent by the Ethernet module.

POINT

Data in the attached file in CSV format can be read in cell units in Excel, etc., for each word of data.

10.7.2 When receiving data in an attached file

This section explains the buffer memory addresses and a program to receive e-mail from external devices.

The following instruction shows how to read the received attached file.

1) Read the value in BFM#22640 to check if there is incoming mail in the mail server.
2) Write "0001$_H$" in BFM#14500 to receive a mail.
(Only e-mails with an attached file are possible to be read.)
3) The received mail data is written in the buffer memory address.
4) The received mail is deleted from the mail server.
The number of receivable mail is reconfirmed and stored in BFM#22640.

The table below shows the buffer memory to be used for receiving mail.

BFM number	Item	Description
Decimal (Hexadecimal)		
22640 (5870H)	E-mail status	Stores the number of receivable mail on the server
14500 (38A4H)	Mail receiving command	Start mail receiving (0001H), Ignore (0000H)
14501 (38A5H)	Mail receiving status	Initial status (0000H) Mail receiving normally completed (0001H) Mail receiving abnormally completed (0002H) Mail being received (8000H)
14502 (38A6H)	Error status while receiving mail	Stores the error code while receiving mail
14503 to 14534 (38A7 to 38C6H)	Message ID	Stores the message ID of the received mail
14535 (38C7H)	Character string length of message header	Stores the character string length of message header
14536 to 14727 (38C8 to 3987H)	Message header	Stores the message header (maximum 384 characters)
14728 to 14759 (3988 to 39A7H)	Send source mail address	Stores the source mail address (size)
14792 to 14855 (39C8 to 3A07H)	Subject	Stores the subject (maximum 128 characters)
14856 to 14871 (3A08 to 3A17H)	File name of attached file	Stores the file name of an attached file (maximum 32 characters)
14872 (3A18H)	Word count of attached file	Stores the word count of main text/attached file (0 to 2048)
14873 to 16920 (3A19 to 4218H)	Main text or attached file	Stores the main text or the attached file (size)

POINT

(1) The Ethernet module checks whether there is incoming mail in the mail server at the time interval set in the Mail parameter (BFM#9904).
If there is incoming mail, the reception information is stored in the following areas of the buffer memory.
- The number of received mail on the mail server is stored in the received mail count storage area (BFM#22640).

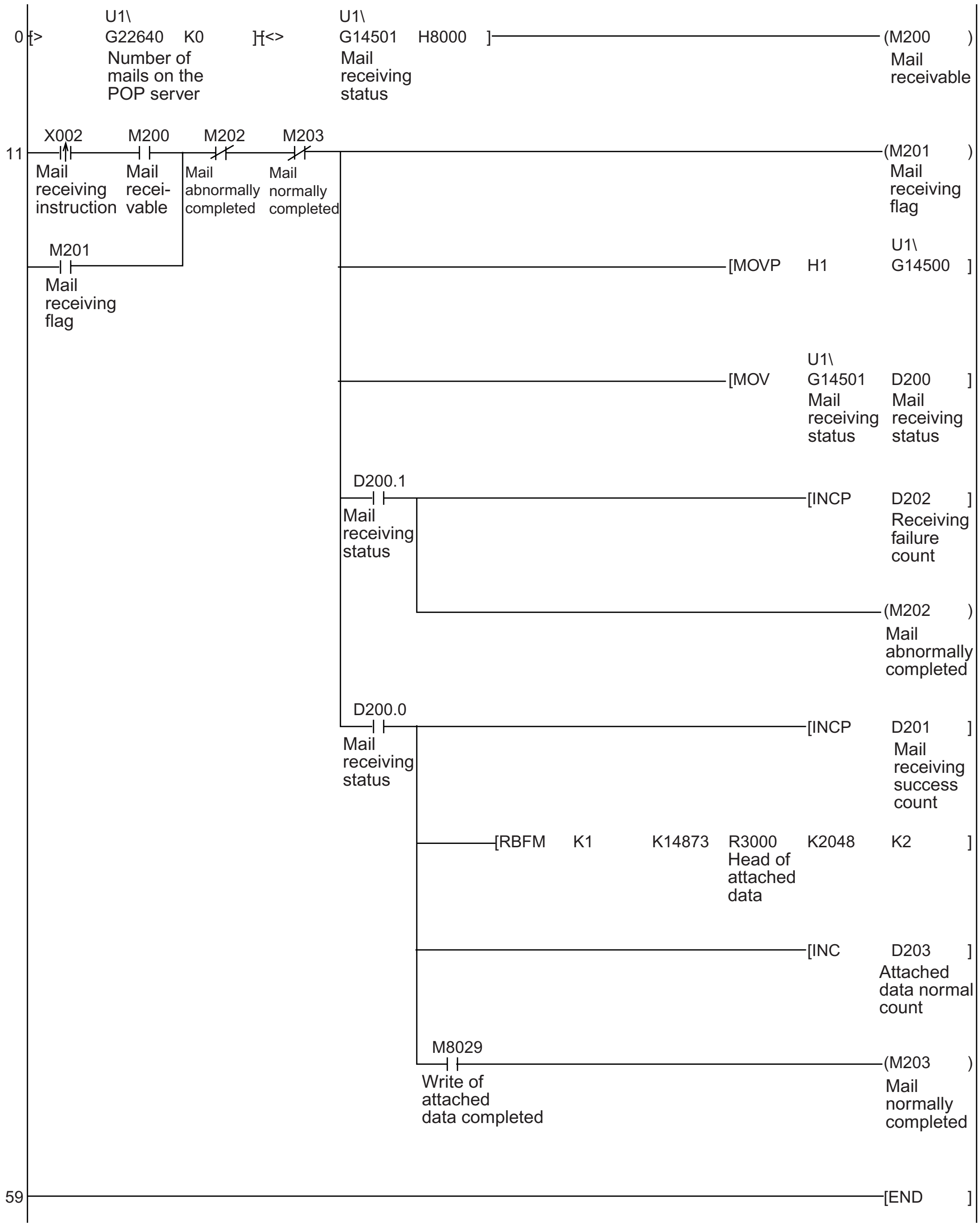
(2) When there is mail stored on the mail server, execute the FROM/TO instruction to receive them.
Received mail accumulate on the mail server if the mail is not received.

(3) E-mails that have been read from the mail server are deleted.

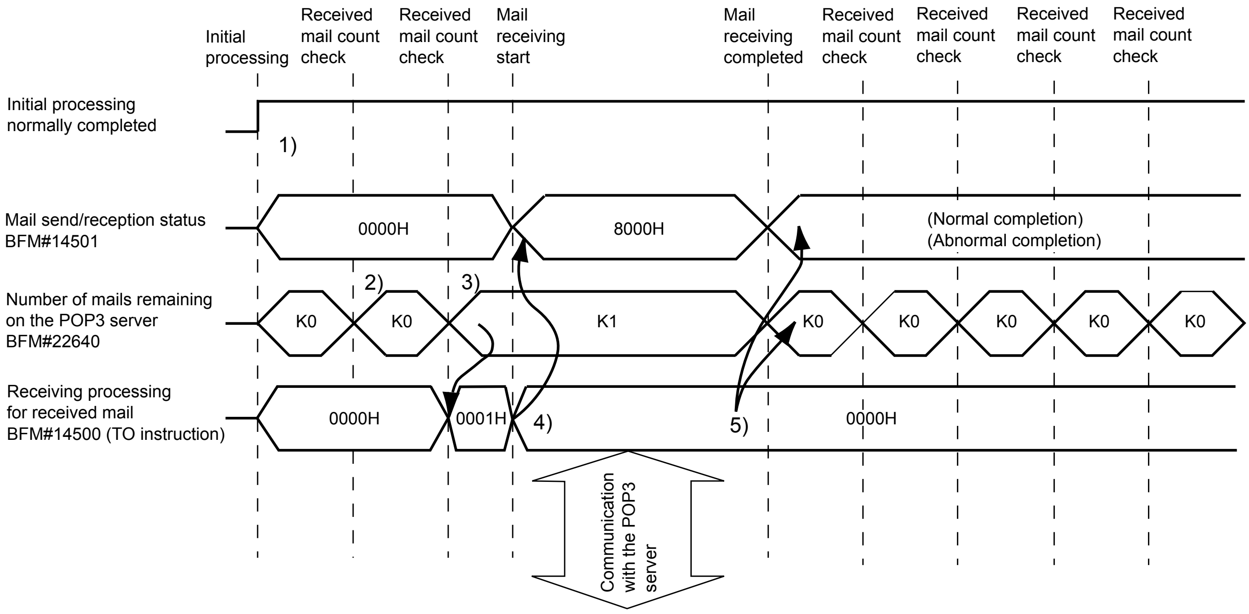
(4) If the Ethernet module has received e-mail longer than 2k words, it stores the data equivalent to the reception data length and ignores the remaining data.

(1) Reception using the sequence program

The following explains the designation method in a sequence program.



(2) Execution timing of the instruction


Initial processing
Received mail count check
Received mail count check
Mail receiving start
Mail receiving completed
Received mail count check
Received mail count check
Received mail count check
Received mail count check
Initial processing normally completed
1)
Mail send/reception status BFM#14501
0000H
8000H
(Normal completion)
(Abnormal completion)
2)
3)
Number of mails remaining on the POP3 server BFM#22640
K0
K0
K1
K0
K0
K0
K0
K0
Receiving processing for received mail BFM#14500 (TO instruction)
0000H
0001H
4)
5)
0000H
Communication with the POP3 server

1) Check whether the initial processing has been completed normally or not.

2) Read the value in BFM#22640 to see whether there is incoming mail in the mail server.

3) Write “0001$_H$” in BFM#14500 to read e-mail from the server.
＊E-mail read from the server do not remain on the server.

4) The received mail data is written in the buffer memory address.
Mail receiving status (BFM#14501) becomes “8000$_H$” (mail receiving).

5) The received mail is deleted from the server and the number of receivable mail is stored in BFM#22640 again.

6) When there is receivable mail on the server, read them by writing “0001$_H$" in BFM#14500.

POINT

(1) Reading receive mails to the Ethernet module
The Ethernet module automatically checks the mail server to see whether there is incoming mail at the inquiry interval set with Mail parameter (BFM#9904).
If there is incoming mail, the reception information is stored in BFM#22640; the FROM/TO instruction should be executed based on this information to read receive mail.

(2) When there are two or more received mail on the mail server
After reading a mail, the received mail information on the mail server is read again and stored in BFM#22640.
Following this information, execute the FROM/TO instruction for reading the received mail.
In addition, note that the second mail overwrites the first mail.

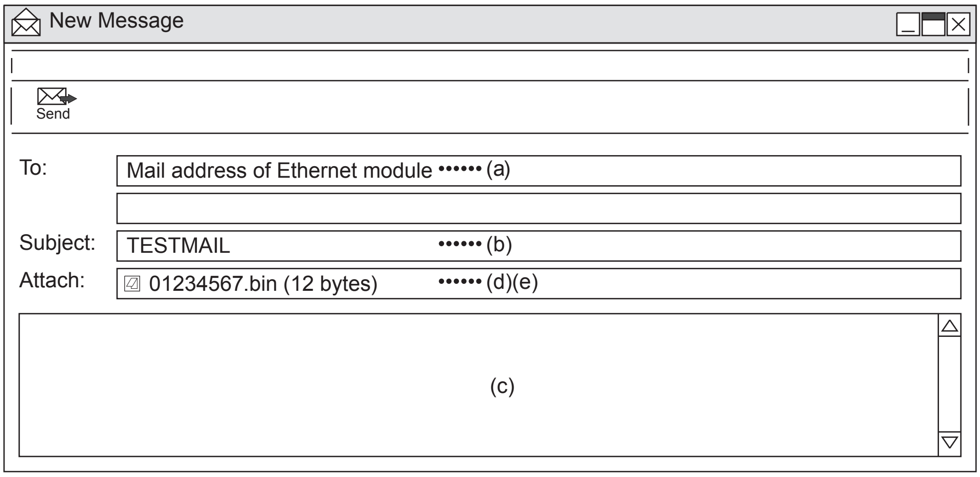
(3) Sending from a personal computer

This section explains how to send e-mails from a PC/UNIX to the Ethernet module.

Once an environment that is capable of using the Internet through an application program that handles e-mail is set up, e-mail can be created and sent on the New Message screen.

POINT

As for an e-mail program that sends e-mail to the Ethernet module, use a program that can designate the encoding method (Base 64/7 bits/Quoted printable) for attached files.



To: Mail address of Ethernet module •••••• (a)

Subject: TESTMAIL •••••• (b)

Attach: ☑ 01234567.bin (12 bytes) •••••• (d)(e)

(c)

(a) To

Mail address of the Ethernet module.

(b) Subject

Subject of the e-mail to be sent to the Ethernet module in ASCII format.

(c) Main text

Do not assign data (the Ethernet module ignores it.)

(d) Attach file

Data can be sent as an attached file.

Send the attached file data as binary data.

* The Ethernet module does not convert the received attached file data from ASCII to binary.

(e) Attached file name

Designate the name of the attached file using alphanumeric characters. (It is recommended to use the name within eight characters.) Designate either ".bin" or ".asc" for the attached file extension. The stored data will be received as binary data.

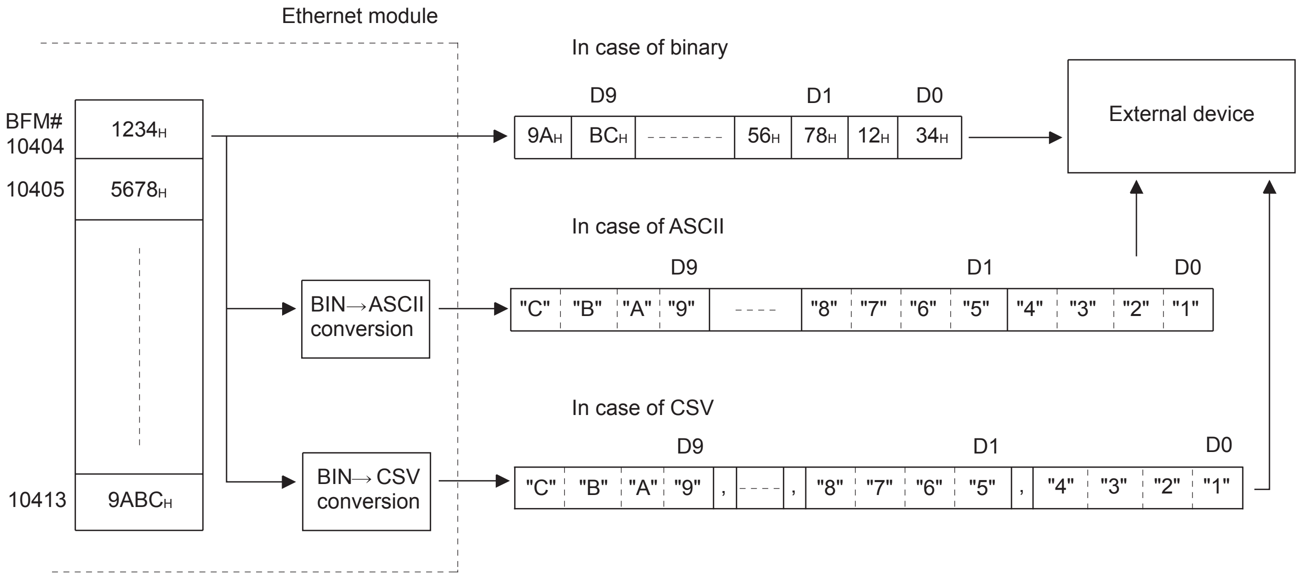
NOTE

Designate a single attached file when sending e-mail to the Ethernet module.
If more than one files are attached, the destination Ethernet module will store only the first file and ignore the rest of the attached files.

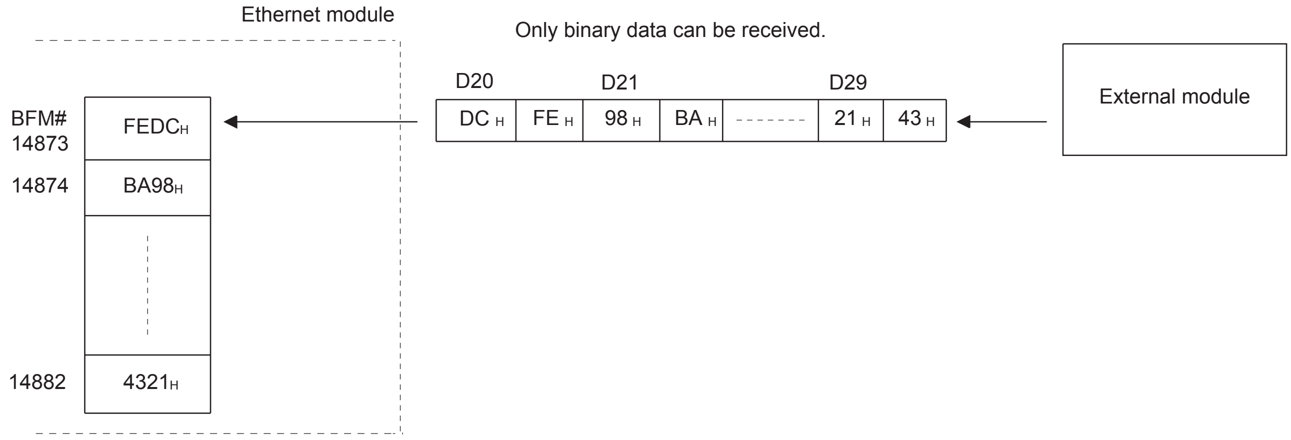
10.7.3 Contents of the attached files

The following explains the contents of the attached file data.
The data format of the attached file is ASCII code (ASCII format, CSV format) or binary code (binary format).
The following example shows the transmission contents for each data format (binary/ASCII/CSV) when the Ethernet module sends the data register value as an attached file.

(1) When sending from the Ethernet module



(2) When receiving by the Ethernet module



(3) Data configuration in CSV format

Data in CSV format can be used as control data for the cell units in Excel, etc. Data in CSV format handled by the Ethernet module is the data expressed as follows.

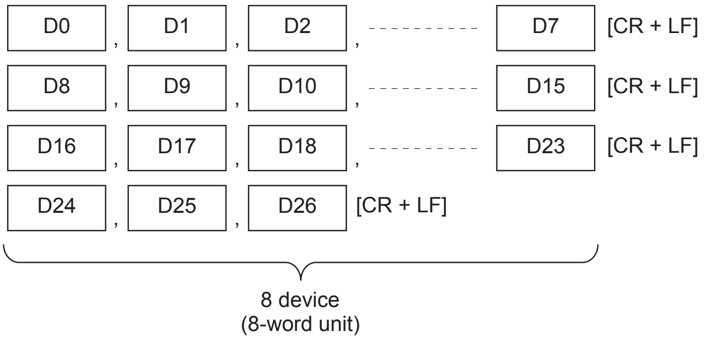
- 1 word (16 bits) binary data is converted to four digit ASCII code when expressed as a hexadecimal. (4 bytes of ASCII code)
- When handling multiple word data, insert comma data between each word (4 bytes of ASCII code).

Data is sent in the following arrangement.

Arrangement of data

1) In addition to the data conversion and insertion described above, CR (code: $0D_H$) and LF (code: $0A_H$) are inserted for each 8 words of data.

2) The arrangement of data sent in CSV format is shown below.


D0 , D1 , D2 , ---------- D7 [CR + LF]
D8 , D9 , D10 , ---------- D15 [CR + LF]
D16 , D17 , D18 , ---------- D23 [CR + LF]
D24 , D25 , D26 [CR + LF]
8 device
(8-word unit)

10.8 Sending E-mail (Main Text) by the PLC

This section explains how to send the main text of e-mail.

10.8.1 When sending data as main text of e-mail

The following explains the buffer memory address and sequence program for sending e-mail.

This section explains how to store the data in main text and send it to the external device.

1) Check whether mail can be sent or not.
(BFM#10334…b15: OFF)

2) Destination of the mail.
Turn the bit corresponding to the e-mail destination address to ON in BFM#10337.
To send to all of the mail addresses registered (maximum 10 addresses), write $3FF_H$.

3) Write the mail data to the following buffer memory addresses.
(Subject: BFM#10339 to 10403)
(Attached file format: Set “0004H” in BFM#10338)
(Mail text/word count of attached file : BFM#10403)
(Contents of main text: BFM#10404 to BFM#12451)

4) Write “0001_H” to BFM#10333 to send the mail.

5) Read the value in BFM#10336 to check if mail was correctly sent.

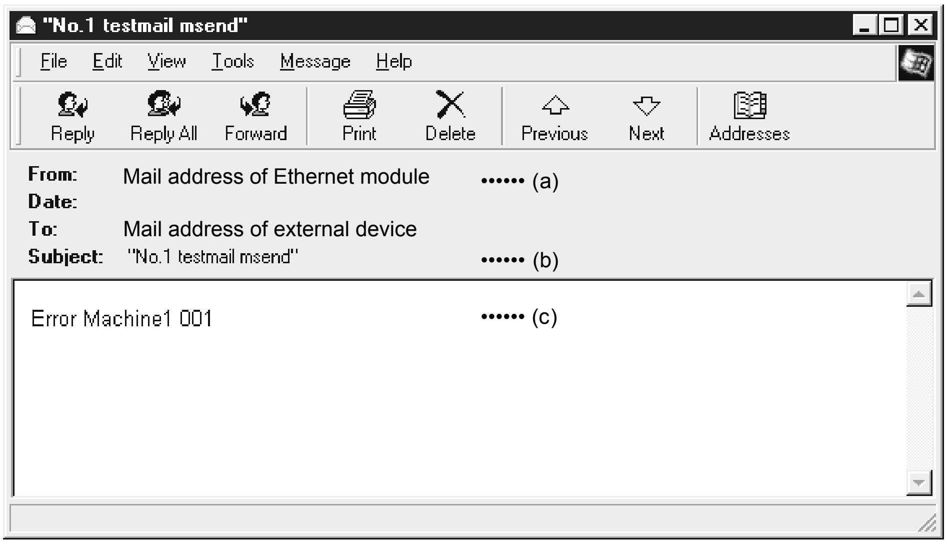
The table below shows the buffer memory to be used for sending mail.

BFM number Decimal (Hexadecimal)	Item	Description
10333 (285DH)	Mail sending command	Start mail sending (0001H) Ignored (0000H)
10334 (285EH)	Mail sending status	Initial status (0000H) Mail sending normal completed (0001H) Mail sending abnormal completed (0002H) Mail being sent (8000H)
10336 (2860H)	Sending result of e-mail	Sending destination address 1 Success (b0: ON) / Failure (b0: OFF) to Sending destination address 10 Success (b9: ON) / Failure (b9: OFF)
10337 (2861H)	Destination Address of e-mail	Turns the bit of mail destination address to ON. Sending destination address 1 Enable (b0: ON) / Disable (b0: OFF) to Sending destination address 10 Enable (b9: ON) / Disable (b9: OFF)
10338 (2862H)	Attached file format	Write 0004H for sending the attached file as main text.
10339 to 10402 (2863 to 28A2H)	Subject	Stores a subject (maximum 128 characters) Subject consists of the send destination number and the character data designated by the sequence program. (It is not converted to ASCII format.) ✻The Subject should be designated in ASCII characters in the sequence program. ✻When the subject consists of less than 128 characters, write "\0" at the end of the subject.
10403 (28A3H)	Word count of attached file	Designates the word count of main text (0 to 256)
10404 to 12451 (28A4 to 30A3H)	Main text data	Stores a main text or an attached file Main texts are not converted to ASCII format by the Ethernet module. ✻It should be designated in ASCII characters in the sequence program. ✻The following binary code data is processed as control codes: 0D0AH: Line feed code CR + LF 00H: End of main text ✻It is recommended to use 78 characters or less in a single line of main text. (Be sure to add a line feed code CR + LF ($0D0A_H$) at the end of the main text lines.)

(1) Send using the sequence program
Method for sending e-mails by the sequence program is similar to the one by an attached file.
See Section 10.7.1 for sample programs for sending e-mail from PLC.

(2) Receiving by a personal computer
The following explains how a PC/UNIX/portable terminal receives e-mail sent from the Ethernet module.
After setting an environment for using the Internet with application software that processes e-mails, e-mails can be received on a receive message screen as shown below.

(Example) The following shows an example of a received e-mail.
(The screen display shows Microsoft® Corporation's Outlook Express 6.)



From: Mail address of Ethernet module •••••• (a)
Date:
To: Mail address of external device
Subject: "No.1 testmail msend" •••••• (b)

Error Machine1 001 •••••• (c)

(a) From
Displays the e-mail address of the Ethernet module.

NOTE

By entering the mail addresses of the send destination devices in the address book, the registered names (display names) can be displayed and makes management easier.

(b) Subject
This indicates the transmission destination No. (＊1) and the Subject written by a sequence program.
＊1 The Ethernet module automatically adds this number, sequentially, starting from 00001.
(After the maximum number (99999) is reached, the number starts again from 00000.)

(c) Main text of e-mail
This indicates the main text of the e-mail sent by the Ethernet module.

11 TROUBLESHOOTING

This section explains the contents of the errors that may occur during communication between the Ethernet module and an external device as well as the troubleshooting procedures.
The following are methods for checking if there is an error on the Ethernet module side and the contents of the error.
Use one of the following methods to check if there is an error and its content, then take corrective actions.

(1) Check using the display LED on the front of the Ethernet module (See Section 11.1.)

The display LED on/off status can be used to check if an error is occurring in the Ethernet module.

(2) Check through FX Configurator-EN (GX Developer)

FX Configurator-EN (GX Developer) can be used to check various conditions of the Ethernet module as well as the error code corresponding to the contents of the error occurring and to perform tests.

(a) Ethernet diagnostics (using the dedicated screen)

1) PING test (See Section 5.4.1.)

2) COM. ERR off (See Sections 11.1.2 and 11.2.1.)

(3) Check through GX Developer

(a) Buffer memory batch monitor

The error code can be checked by monitoring the buffer memory of the Ethernet module.

(4) Check the contents of the error using the error code (See Section 11.4.)

The contents of the error can be checked using the error code confirmed on the dedicated screen above or by monitoring the buffer memory by referring to Section 11.4.

NOTE

If line errors and other errors occur when connecting devices of multiple manufacturers, the users needs to isolate the malfunctioning parts using line analyzers, etc.

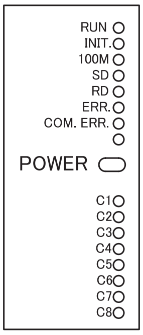
11.1 How to Check Errors Using LED Displays

This section describes the errors that can be checked with the LED displays on the front of the Ethernet module.

11.1.1 Checking error display

The following can be checked with the LED displays on the front of the Ethernet module.

<Ethernet module LED>



	LED name	Status to check	Cause/corrective action
1	RUN	Turns off after powering on the Ethernet module.	1) Watchdog timer error • A watchdog timer (approximately 600 ms) error is detected by the self diagnosis function of the Ethernet module. 2) Base module power is OFF, or the extension cable is not connected.
2	INIT.	Off	1) Failed to re-initialization due to a parameter error • Check/correct the parameter setting values for the Ethernet modules using a sequence program or the FX Configurator-EN (GX Developer) and re-download. 2) Base module power is OFF, or the extension cable is not connected.
3	100M	Does not turn on	1) Using 10Mbps HUB • Initial processing. • Replace with 100Mbps HUB. 2) Does not turn on using 100Mbps HUB • Check the connection of the connectors on the Ethernet side and the HUB side. • Replace the cable.
4	ERR.	Turns on after powering on the Ethernet module.)	1) Module parameter setting error • Check/correct the parameter setting values for the Ethernet modules using the sequence program or the FX Configurator-EN (GX Developer) and re-download. 2) PLC error • When the PLC [RUN] LED is off/flashing, or the [ERR.] LED is on, check the content of the error occurring in the PLC and correct the problem. 3) Ethernet module error (H/W error) 4) Base module power is OFF, or the extension cable is not connected.
5	COM.ERR	Turns on after powering on the Ethernet module. Or turns on temporarily	1) Check the contents of the error using the error codes stored by the error detection of the following processing and remove the causes. • Initial processing • Fixed buffer send processing • E-mail send/receive processing • Open processing • MC protocol communication processing • Other processing (processing for which error codes are stored in the error log area) 2) For a list of error codes, see Section 11.4.

(Continues on the following page)

(Continued from the previous page)

	LED name	Status to check	Cause/corrective action
6	SD	The [SD] LED does not flash at data sending.	1) [ERR.] or [COM.ERR] LED turns on. • Remove the factors that turn on the [ERR.] or [COM.ERR] LED. 2) Program reviewing is required • Review the sequence program for sending.
7	RD	[RD] LED stays off and data cannot be received.	1) [ERR.] or [COM.ERR] LED turns on. • Remove the factors that turn on the [ERR.] or [COM.ERR] LED. 2) Poor cable connection • Check the connection of the cable. (∗ 1) 3) Local station IP address setting error • If the cable connection is all right, review each setting value of the local station IP address, router setting, and sub-net mask settings using FX Configurator-EN (GX Developer). 4) Program reviewing is required • Review the sequence program for sending.
8	C1 to C8	Does not turn on	1) Poor cable connection • Check the connection of the cable. 2) Program or parameter reviewing is required • Check the program or parameter. • Check error contents with the diagnosis function of FX-Configurator-EN (GX Developer).

∗ 1 Confirm the completion for the initial processing and check whether or not there is any problem in the cable connection and the Ethernet lines.
See Section 5.4, "Confirming the completion of the Initial Processing" for details on confirming the completion for the initial processing. (Perform either one of the "Confirming the completion of the initial processing completion" actions described in Section 5.4.)

POINT

The on/off status of the [INIT], [OPEN], [ERR.] and [COM.ERR] LEDs is stored in the module status area (BFM#28) of the buffer memory.
For more details, see Section 3.6, "List of Applications and Assignments of the Buffer Memory".

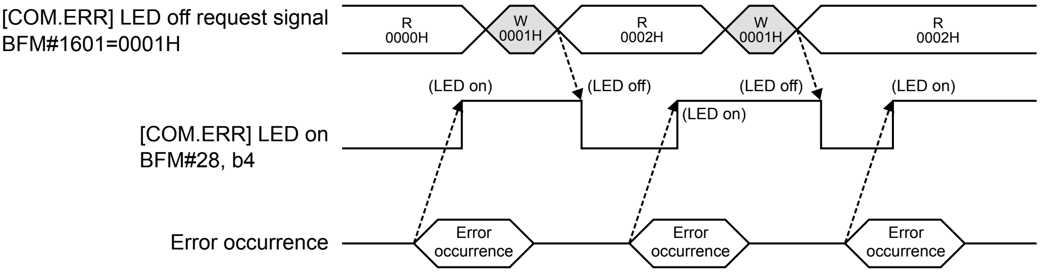
11.1.2 How to turn off COM.ERR LED and to read/clear error information

This section explains how to turn off [COM.ERR] LED and to read/clear error information using a sequence program.

(1) How to turn off [COM.ERR] LED using input/output signals

The [COM.ERR] LED on the front of the Ethernet module is turned on when a communication error occurs in an external device. (BFM#28...b4: ON)

(a) The [COM.ERR] LED is turned off by writing "0001H" to the off request signal (BFM#1601).



(b) The off processing is performed by writing "0001H" to BFM#1601.

(c) The error information in the error log area of the buffer memory is not cleared (deleted) even if "0001H" is written to BFM#1601.

(2) How to turn off [COM.ERR] LED on the "Ethernet diagnostics" screen of FX Configurator-EN (See Section 11.2.2.)

(a) Clicking on the COM.ERR off button turns the [COM.ERR] LED off.

(b) The error information in the error log area of the buffer memory is not cleared (deleted).

(3) Reading buffer memory to read error information

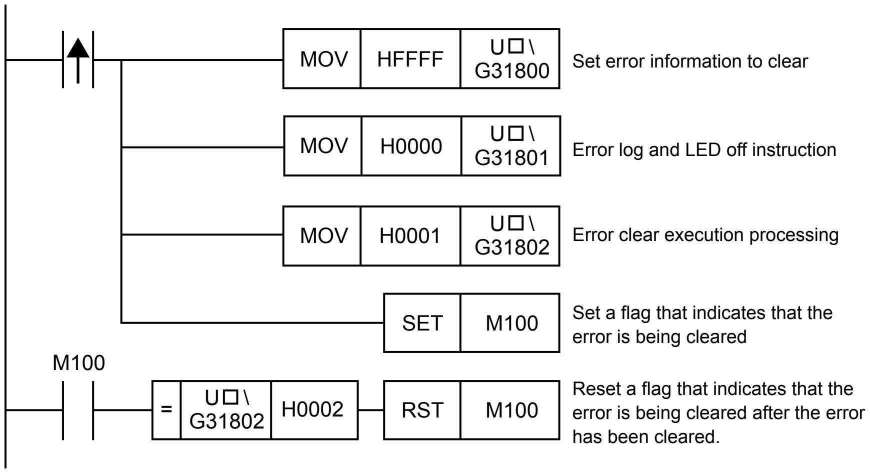
Read the error information stored in the buffer memory of Ethernet module.

The “Initial alarm code” and each connection’s “Open alarm code” are stored in the following buffer memory.

Initial error code	BFM#105
Error code for connection 1	BFM#124
Error code for connection 2	BFM#134
Error code for connection 3	BFM#144
Error code for connection 4	BFM#154
Error code for connection 5	BFM#164
Error code for connection 6	BFM#174
Error code for connection 7	BFM#184
Error code for connection 8	BFM#194

(4) How to clear error information by writing to buffer memory

Writing to BFM#31800 to 31802 allows clearing the error information and turning off COM.ERR.

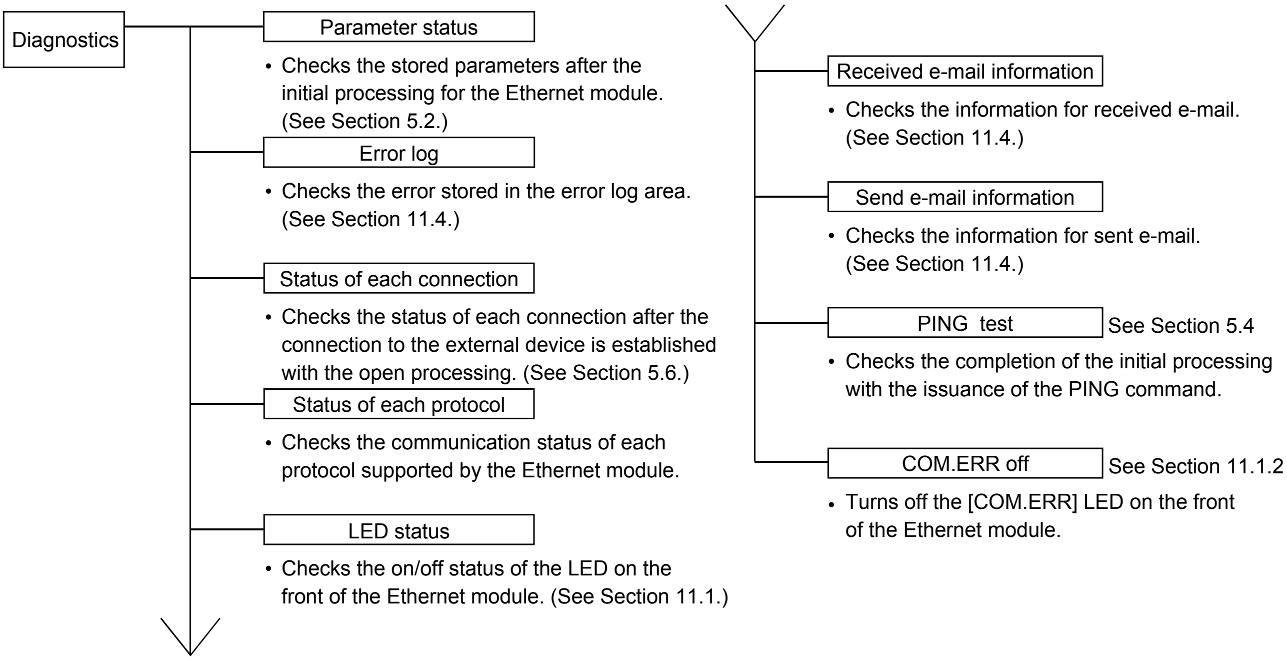

MOV
HFFFF
U□\ G31800
Set error information to clear
MOV
H0000
U□\ G31801
Error log and LED off instruction
MOV
H0001
U□\ G31802
Error clear execution processing
SET
M100
Set a flag that indicates that the error is being cleared
M100
=
U□\ G31802
H0002
RST
M100
Reset a flag that indicates that the error is being cleared after the error has been cleared.

11.2 How to Check an Error Through FX Configurator-EN (GX Developer)

The status of the various settings for the Ethernet module can be checked using the FX Configurator-EN (GX Developer) functions.

(1) Ethernet diagnostics (See Section 11.2.2.)

The block status of an Ethernet module, parameter settings, communication status, error log and others can be checked using the diagnostic function.
The following are the functions of the diagnostics.



Parameter status
- Checks the stored parameters after the initial processing for the Ethernet module. (See Section 5.2.)

Error log
- Checks the error stored in the error log area. (See Section 11.4.)

Status of each connection
- Checks the status of each connection after the connection to the external device is established with the open processing. (See Section 5.6.)

Status of each protocol
- Checks the communication status of each protocol supported by the Ethernet module.

LED status
- Checks the on/off status of the LED on the front of the Ethernet module. (See Section 11.1.)

Received e-mail information
- Checks the information for received e-mail. (See Section 11.4.)

Send e-mail information
- Checks the information for sent e-mail. (See Section 11.4.)

PING test See Section 5.4
- Checks the completion of the initial processing with the issuance of the PING command.

COM.ERR off See Section 11.1.2
- Turns off the [COM.ERR] LED on the front of the Ethernet module.

(2) Buffer memory batch monitor (See Section 11.3.)

The buffer memory of an Ethernet module is monitored.

POINT

See Section 11.2.1 for the buffer memory that can be checked on the "Ethernet diagnostics" screen.

11.2.1 Buffer memory that can be monitored with the FX Configurator-EN (GX Developer) diagnostic function

The following is a list of the buffer memory addresses that can be displayed on the "Ethernet diagnostics" screen of FX Configurator-EN (GX Developer).

BFM address Decimal (hexadecimal)	Applicable buffer memory		Ethernet diagnostics display screen	Display contents	
32 (20$_H$)	Connection No.1	D Destination existence confirmation (b1)	Status of each connection	Connection No.1	Existence confirmation
		Pairing open (b7)			Paring open
		Communication system (protocol) (b8)			Protocol
		Open system (b15, b14)			Open system
33 to 39 (21$_H$ to 27$_H$)	Connection No.2 to 8 (same as connection No.1)			Connection No.2 to 8 (same as connection No.1)	
105 (69$_H$)	Initial error code		Parameter status	Module information	Initial error code
106 to 107 (6A$_H$ to 6B$_H$)	Local station IP address				IP address
108 to 110 (6C$_H$ to 6E$_H$)	Local station Ethernet address				Ethernet address
120 (78$_H$)	Connection No.1	Local station Port No. (Decimal)	Status of each connection	Connection No.1	Local station Port No.
121 to 122 (79$_H$ to 7A$_H$)		Destination IP address			Destination IP address
123 (7B$_H$)		Destination Port No.			Destination Port No.
124 (7C$_H$)		Open error code			Open error code
125 (7D$_H$)		Fixed buffer sending error code			Fixed buffer transfer error code
126 (7E$_H$)		Connection end code			Connection end code
130 to 199 (82$_H$ to C7$_H$)	Connection No.2 to 8 (same as connection No.1)			Connection No.2 to 8 (same as connection No.1)	
200 (C8$_H$)	LED on/off status	[INIT.]LED (b0)	LED status	LED display status	INIT.
		[C1] to [C8] LED (b8 to b15)			C1 toC8
		[ERR.]LED (b3)			ERR.
		[COM.ERR] LED (b4)			COM.ERR
227 (E3$_H$)	Number of error occurrence		Error log	Latest	Number of error occurrences
229 (E5$_H$)	Error log block 1	Error code/End code			Error code/End code
230 (E6$_H$)		Subheader			Subheader
231 (E7$_H$)		Command code			Command code
232 (E8$_H$)		Connection No.			Connection No.
233 (E9$_H$)		Local station Port No. (Decimal)			Local station Port No.
234 to 235 (EA$_H$ to EB$_H$)		Destination IP address			Destination IP address
236 (EC$_H$)		Destination Port No.			Destination Port No.
238 to 372 (EE$_H$ to 174$_H$)	Error log block 2 to 16 (same as error log block 1)			No.2 to 16 (same as connection No.1)	
376 to 377 (178$_H$ to 179$_H$)	IP	Received IP packet count	Status of each protocol	IP packet	Total number of echo reply sends
378 to 379 (17A$_H$ to 17B$_H$)		Received IP packet count discarded due to sum check error			Total number echo request sends
380 to 381 (17C$_H$ to 17D$_H$)		Sent IP packet total count			Total number echo reply receives
408 to 409 (198$_H$ to 199$_H$)	ICMP	Received ICMP packet count		ICMP packet	Total number of receives
410 to 411 (19A$_H$ to 19B$_H$)		Received ICMP packet count discarded due to Sum check error			Total number of Sum check error annulments
412 to 413 (19C$_H$ to 19D$_H$)		Sent ICMP packet total count			Total number of sends
414 to 415 (19E$_H$ to 19F$_H$)		Echo request total count of received ICMP packets			Total number echo request receives
416 to 417 (1A0$_H$ to 1A1$_H$)		Echo reply total count of sent ICMP packets			Total number of echo reply sends
418 to 419 (1A2$_H$ to 1A3$_H$)		Echo request total count of sent ICMP packets			Total number of echo request sends
420 to 421 (1A4$_H$ to 1A5$_H$)		Echo reply total count of received ICMP packets			Total number of echo reply receives

BFM address Decimal (hexadecimal)	Applicable buffer memory			Ethernet diagnostics display screen	Display contents	
440 to 441 (1B8H to 1B9H)	TCP	Received TCP packet count		Status of each protocol	TCP packet	Total number of receives
442 to 443 (1BAH to 1BBH)		Received TCP packet count discarded due to Sum check error				Total number of Sum check error annulments
444 to 445 (1BCH to 1BDH)		Sent TCP packet total count				Total number of sends
472 to 473 (1D8H to 1D9H)	UDP	Received UDP packet total count			UDP packet	Total number of receives
474 to 475 (1DAH to 1DBH)		Received UDP packet count discarded due to Sum check error				Total number of Sum check error annulments
476 to 477 (1DCH to 1DDH)		Sent UDP packet total count				Total number of sends
22640 (5870H)	Receive	Number of mails remaining on the server		Received e-mail information	Number of mails remaining on the server	
22643 (5873H)		Number of normal receives			Number of normal receives	
22644 (5874H)		Number of attached file receives			Number of attached file receives	
22645 (5875H)		Number of server inquiries			Number of server inquiries	
22646 (5876H)		Number of server transfer errors			Number of server transfer errors	
22647 (5877H)		Number of error log writes			Number of error log writes	
22649 (5879H)		Error log block 1	Error code		Latest error log	Error code
22650 (587AH)			Command code			Command code
22651 to 22658 (587BH to 5882H)			From			From
22659 to 22662 (5883H to 5886H)			Receive data			Date
22663 to 22692 (5887H to 58A4H)			Subject			Subject
22693 to 23352 (58A5H to 5B38H)		Error log block 2 to 16 (same as error log block 1)			Error log block 2 to 16 (same as error log block 1)	
23355 (5B3BH)	Send	Number of mails normally completed		Send e-mail information	Number of mails normally completed	
23356 (5B3CH)		Number of attached file sends			Number of attached file sends	
23357 (5B3DH)		Number of sends to the server			Number of sends to the server	
23358 (5B3EH)		Number of mails abnormally completed			Number of mails abnormally completed	
23359 (5B3FH)		Number of error log writes			Number of error log writes	
23361 (5B41H)		Error log block 1	Error code		Latest error log	Error code
23362 (5B42H)			Command code			Command code
23363 to 23370 (5B43H to 5B4AH)			To			To
23371 to 23374 (5B4BH to 5B4EH)			Receive data			Send date
23375 to 23404 (5B4FH to 5B6CH)			Subject			Subject
23405 to 23712 (5B6DH to 5CA0H)		Error log block 2 to 16 (same as error log block 2 to 16)			Error log block 2 to 16 (same as error log block 1)	

11.2.2 Ethernet diagnostics

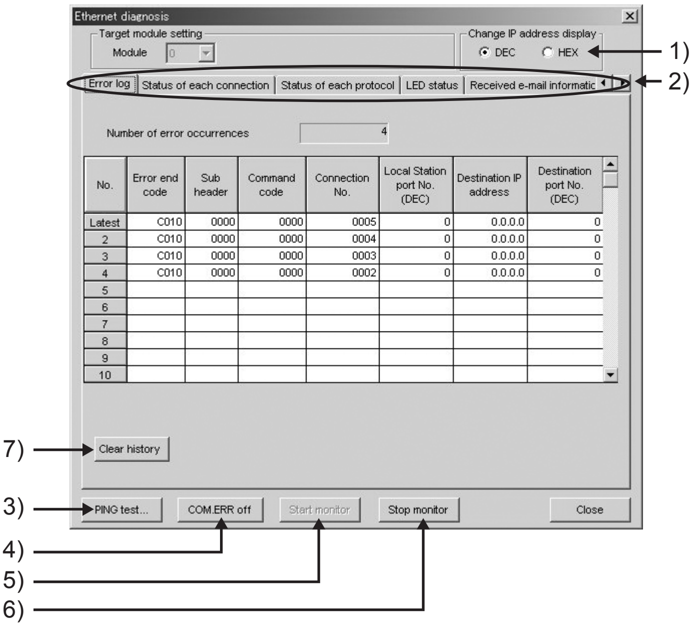
[Purpose]

The block status of an Ethernet module, parameter settings, communication status, error log and others can be checked using the Ethernet diagnostic function of FX Configurator-EN.

[Operating procedure]

FX Configurator-EN → Diagnostics → Diagnostics

[Ethernet diagnostics screen]



Module 0

● DEC ○ HEX

Number of error occurrences 4

No.	Error end code	Sub header	Command code	Connection No.	Local Station port No. (DEC)	Destination IP address	Destination port No. (DEC)
Latest	C010	0000	0000	0005	0	0.0.0.0	0
2	C010	0000	0000	0004	0	0.0.0.0	0
3	C010	0000	0000	0003	0	0.0.0.0	0
4	C010	0000	0000	0002	0	0.0.0.0	0
5							
6							
7							
8							
9							
10							

[Explanation of items]

No.	Item	Description	Setting range
1	Change IP address display	Switches the IP address display between decimal and hexadecimal.	Decimal/hexadecimal
2	Selection from the various information monitors	Various types of information for the Ethernet module can be monitored. (See Section 11.2.1 for the buffer memory corresponding to the display contents.)	—
3	PING test	Performs the PING test on the external device. (See Section 5.4.1, Section 5.4.2.)	
4	COM. ERR off	Clicking this button turns the [COM.ERR] LED off. (See Section 11.1.2.)	
5	Start monitor	Clicking this button executes the Ethernet diagnostics. The display is updated during monitoring.	
6	Stop monitor	Clicking this button stops the Ethernet diagnostics. The display is retained while monitoring is stopped.	
7	Clear history	Clears the log.	

11.3 Checking the error information by the buffer memory batch monitoring function

It is explained here how the Ethernet module errors can be checked from GX Developer.
Error codes stored in the buffer memory of the Ethernet module can be monitored using the "Buffer memory batch monitoring" function of GX Developer.

[Operating procedure]

(Step 1) Select [Online] – [Monitor] – [Buffer memory batch] from the GX Developer menu bar, and start the "Buffer memory batch monitoring" screen.

(Step 2) Enter [Module start address:].
Assign a special module number to each base module following the order that 0 is assigned to the rightmost module, 1 to the second rightmost module, and goes same up to 7.
However, for FX3UC-32MT-LT which incorporates the CCLink/LT function, the first special module will be assigned with No.1.

(Step 3) Enter [Buffer memory start address:].
Enter the buffer memory address to be monitored, using the selected input format (decimal/hexadecimal).
For a list of the buffer memory addresses where error codes are stored, see Section 11.4, "Error Code List".
(Example)
When monitoring the initial abnormal code (BFM #105):
Enter "105" + "decimal"

(Step 4) Click the Start Monitor button.
The contents of the buffer memory after the specified address are displayed.
(In case of the above example, the contents of 105_H and succeeding addresses are displayed.)

POINT

To see if the PLC base module recognizes the FX3U-ENET or not, monitor BFM #30.
If BFM #30’s value is K7130, the FX3U-ENET is being recognized.

NOTE

The display format can by modified as follows:

Monitor format	:	Bits & words/ Multiple bit points/ Multiple word points
Display	:	16-bit integer/32-bit integer/real number/ASCII character
Numerical value	:	Decimal/hexadecimal

For details, refer to the "Operating Manual" for GX Developer.

11.4 Error Code List

This section explains the error codes (abnormal codes) for the errors that may occur in each processing when communicating data between the Ethernet module and an external device as well as those generated by processing requests from the local station's FX series PLC.

11.4.1 Type of error incident

The details of errors and error handling procedures are described.

	Type of error	Description	Error code storage buffer memory	Explanation
1	Errors occurring in initial processing	• Setting value error • Initial processing error	BFM#105............ Initial error code (Communication status storage area)	Section 11.4.4
2	Errors occurring in open processing	• Setting value error • Open processing error	BFM#124............ Open error code (Communication status storage area)	
3	Errors occurring in fixed buffer sending to an external device	• Designated data error • Sending error	BFM#125.............Fixed buffer sending error code BFM#126.............Connection end code (Communication status storage area)	
4	Errors occurring in fixed buffer communication with an external device	• Designated data error • Communication error (exclude 3 above)	BFM#126..............Connection end code (Communication status storage area)	
5	Errors returned to an external device when communicating with the external device	• Errors returned in fixed buffer communication (end code)		Section 11.4.2
		• Error returned in communication using MC protocol	End codes when A compatible 1E frame commands are used	Section 11.4.2
			Error codes when A compatible 1E frame commands are used	Section 11.4.3
6	Errors occurring while communicating with an external device (including the causes shown in the description column), and whose error codes are stored in the error log area.	• Designated data errors • Errors whose source cannot be confirmed • Errors that occurred while communicating using MC protocol	BFM#229 (Error log area)	Section 11.4.4
7	Errors occurring when receiving e-mail	• Setting data error • Receiving error	BFM#22640 Receive (E-mail status storage area)	Section 11.4.4
8	Errors occurring when sending e-mail	• Setting data error • Sending error	BFM#23355 Send (E-mail status storage area)	

(1) Initial error code (BFM#105)

(a) This address stores the error codes generated when the initial processing is executed.

(b) Error codes are stored as binary values when the initial abnormal completion signal (BFM#28...b5) is on.

(c) An error code is cleared when the initial normal completion signal (BFM#27"0001H", BFM#28...b0: ON) is turns on, but can also be cleared by the following operations.

1) Writing the value in BFM#1600 to perform re-initialization or turning off the PLC and Ethernet module

2) Re-downloading the parameter with FX Configurator-EN (GX Developer) or sequence pragram

(2) Open error code (connection numbers: 1 to 8; BFM#124 to 194)

(a) These addresses store the result of the open processing for the applicable connections.

(b) The results of the open processing are stored in binary values.

0	:	Normal completion
Other than 0	:	Abnormal completion (BFM#28...b0: ON)

(c) An error code is cleared by the following operations.

1) Reopening the connection that caused an open error

2) Performing the re-initial processing by writing the value to BFM#1600

(3) Fixed buffer sending error code (connection numbers: 1 to 8; BFM#125 to 195)

(a) These addresses store error codes generated when an error in data sending to an external device occurs during fixed buffer communication using the applicable connection.

(b) A sending error code is cleared when the next data sending is normally completed.

(4) Connection end code (connection numbers: 1 to 8; BFM#126 to 196)

(a) These addresses store end codes returned from an external device as a response during the fixed buffer communication using the applicable connection.

(b) Determine how to handle the end codes in the responses by arranging with the particular external device.

(5) Error log area (BFM#227 to 372)

This area stores the following errors.

- Errors whose source cannot be confirmed
- Errors that occurred during communications using the random access buffer
- Errors that occurred during communications using MC protocol

(a) Number of error occurrences (BFM#227)

1) This address stores the number of errors registered in the error log block area.

2) When errors occur more than 65536 times, the count is stopped at FFFF$_H$ (65535).

(b) Error log write pointer (BFM#228)

1) This address stores the error log block number where the latest error logging is registered.

0	:	No error. (No registration of error log)
1 or more	:	Error log block number where the latest error logging is registered

✻ If the pointer value is "16", it means that the latest error logging is registered in the error log block 16 area.

2) When 17 or more errors occur, the registration of error logging starts from error log block 1 area again.

POINT

(1) An error log block area consists of sixteen error log blocks that have the same data order.

(2) The error information continues to be stored into the following areas even if the count of the error occurrences is stopped and no longer stored:

- Error log write pointer storage area
- Error log block

(c) Error log block – Error code/end code (starting from BFM#229)

1) In the error code/end code area, error codes are stored that indicate error content. (See Section 11.3.)

(d) Error block – Subheader (starting from BFM#230)

1) The subheader code of a faulty message is stored in bits 0 to 7 of the corresponding area. ("0" is stored in bits 8 to 15).

2) For errors below the TCP and UDP levels, "0" is stored.

(e) Error log block – Command code (starting from BFM#231)

1) This block stores the command code of a faulty message, or each lower byte value of request type and sub-request type of a data link instruction.

b15	to	b0
	Command code	

2) "0" is stored in the following case.

- For messages not containing a command code
- For errors below the TCP and UDP levels (because their commands are unknown)

(f) Error log block – Connection No. (starting from BFM#232)

1) The faulty connection No. is stored in bits 0 to 7 of the corresponding area. ("0" is stored in bits 8 to 15).

2) For errors below the TCP and UDP levels, "0" is stored.

(g) Error log block – Local station Port No. (starting from BFM#233)

1) This block stores the local station's port No. when an error occurred.

2) "0" is stored for errors below the TCP and UDP levels.

(h) Error log block – Destination IP address (starting from BFM#234)

1) This block stores the IP address of an external device when an error occurred.

2) "0" is stored in the following cases.
- For errors below the IP level
- When an error response was relayed through the PLC.

(i) Error log block – Destination Port No. (starting from BFM#236)

1) This block stores the external device's port No. when an error occurred.

2) "0" is stored for errors below the TCP and UDP levels.

(j) Status for each protocol (BFM#376 to 511)

1) This address stores the count of the occurrence for the applicable contents of each protocol's condition.
(The value counted by the Ethernet module.)

2) When the count value exceeds two words, the count stops at $\text{FFFFFFFF}_{\text{H}}$ (4294967295).

POINT

Values stored in the buffer memory are cleared when the Ethernet module's local station is powered on. (They are not cleared during the initial processing.)
Normally, it is not necessary to read this area; so, read it as needed during maintenance.

(6) E-mail status storage area (BFM#22640 to 24575)

＊ When the storage count exceeds $FFFF_H$ times, the count starts from 0_H again.

(a) E-mail status storage area for reception (BFM#22640 to 23352)

1) Number of mails remaining on the server (BFM#22640)

- This area stores the number of mails remaining when the Ethernet module inquires to the receiving mail server.

  0 : No received mail on the server
  1 to 15 : The number of mails remaining in the server
  16 The number of mails in the server is 16 or more

2) Normal receiving count (BFM#22643)

- This address stores a cumulative count when the Ethernet module transferred received mails to the mail buffer data area.

  0 : No mail is transferred.
  1 or more : The number of normal mail transfer completions

3) Attached file receiving count (BFM#22644)

- This address stores a cumulative count of how many times the Ethernet module received mails with files attached.

  0 : No mail received with files attached
  1 or more : The number of normal completions of mail receiving with files attached

4) Server inquiry count (BFM#22645)

- This address stores a cumulative count of inquiries made to the receiving mail server according to the parameter settings. (See Chapter 10)

  0 : No inquiry was made to the server.
  1 or more : Cumulative count of inquiries to the server

5) Server communication error count (BFM#22646)

- This address stores a cumulative count of communication error occurrences that are returned when making inquiries to the receiving mail server.

  0 : No communication error between servers, or no inquiry has been made
  1 or more : Cumulative count of communication error occurrences

6) Error log write count (BFM#22647)

- This address stores a cumulative count of registrations made to the receiving error log block area.

  0 : No error, or no inquiry has been made to the server.
  1 or more : Cumulative count of registrations made to the error log block area

7) Receiving error log write pointer (BFM#22648)
- This address stores the error log block number where the latest error logging is registered.
  0 : No error. (No registration of error log)
  1 or more : Error log block number where the latest error logging is registered
  ＊ If the pointer value is "16", it means that the latest error logging is registered in the error log block 16 area.
- When 17 or more receiving errors occur, the registration of error logging starts from error log block 1 area again.

POINT

The error log block area consists of 16 error log blocks with the same data order.

8) Error log block – Error code (BFM#22649 onwards)
- This block stores the error codes that indicate the contents of errors. (See Section 11.4.)

9) Error log block – Command code (BFM#22650 onwards)
- This block stores the system command codes for the error causing messages.

10) Error log block – From (BFM#22651 onwards)
- This block stores eight words from the beginning of the sending source mail address of an error causing e-mail during communication with the mail server, in ASCII code characters.
  (Example)
  If the sending source mail address was
  "use@from.add.sample.co.jp",
  "use@from.add.sam" is stored as ASCII code characters.

11) Error log block – Date (BFM#22659 onwardss)
- This block stores the time and date on which the e-mail is received in BCD code.

b15 to b8	b7 to b0
Month (01H to 12H)	Lower 2-digits of year (00H to 99H)

b15 to b8	b7 to b0
Hour (00H to 23H)	Date (01H to 31H)

b15 to b8	b7 to b0
Second (00H to 59H)	Minute (00H to 59H)

b15 to b8	b7 to b0
Higher 2-digits of year (00H to 99H)	Day of the week (0 to 6) SUN.: 0 to SAT.: 6

12) Error log block – Subject (BFM#22648)
- This block stores 30 words from the beginning of the Subject of the e-mail.
- A Subject is not stored successfully if it contains characters other than alphanumeric and ASCII code.

(b) E-mail status storage area for sending (BFM#23355 to 23712)

1) Number of mails normally completed (BFM#23355)

- This address stores a cumulative count of how many times the Ethernet module transferred send mails to the Send mail server.

  0 : No mail was sent.
  1 or more : The number of normal completions of mail sending

2) Attached file sending count (BFM#23356)

- This address stores a cumulative count of how many times the Ethernet module sent mails with files attached.

  0 : No mail with files attached has been sent.
  1 or more : The number of normal completions of mail sending with files attached

3) Sending to the server count (BFM#23357)

- This address stores a cumulative count of sending to the send mail server.

  0 : No mail was sent to the server.
  1 or more : Cumulative count of sending to the server

4) Number of mails abnormally completed (BFM#23358)

- This address stores a cumulative count of communication error occurrences that are returned when requesting sending to the transmitting mail server.

  0 : No communication error between servers, or no transmission has been made
  1 or more : Cumulative count of communication error occurrences

5) Error log write count (BFM#23359)

- This address stores a cumulative count of registrations made to the sending error log block area.

  0 : No error, or no inquiry has been made to the server.
  1 or more : Cumulative count of registrations made to the error log block area

6) Sending error log write pointer (BFM#23360)
- This address stores the error log block area number where the latest sending error logging is registered.

0	:	No error. (No registration of sending error log)
1 or more	:	Error log block number where the latest sending error logging is registered

＊ If the pointer value is "8", it means that the latest error logging is registered in the sending error log block 8 area.

- When 9 or more sending errors occur, the registration of sending error logging starts from sending error log block 1 area again.

POINT

The sending error log block area consists of eight error log blocks that have the same order of data items.

7) Error log block – Error code (BFM#23361 onwards)
- This block stores the error codes that indicate the contents of errors. (See Section 11.4.)

8) Error log block – Command code (BFM#23362 onwards)
- This block stores the system command codes for the error causing messages.

9) Error log block – To (BFM#23363 onwards)
- This block stores eight words from the beginning of the sending source mail address of an error causing e-mail during communication with the mail server, in ASCII code characters.

(Example)

If the sending source mail address was "use@from.add.sample.co.jp", "use@from.add.sam" is stored as ASCII code characters.

10) Error log block – Date (BFM#23371 onwards)
- This block stores the time and date on which the e-mail is sent in BCD code.
- The order of date and time to be stored is the same as for the date and time of e-mail reception shown in (a) 11).

11) Error log block – Subject (BFM#23375 onwards)
- This block stores 15 words from the beginning of the Subject of the e-mail.

11.4.2 End codes (Complete codes) returned to an external device during data communication

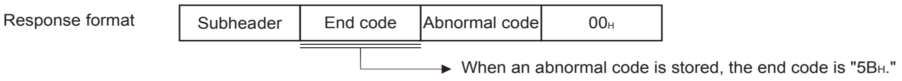
This section explains the end codes (complete codes) that are added to responses when communicating using the fixed buffer or the MC protocol.
For more details on the error codes that are added to responses during communication using A compatible 1E frames through the MC protocol, see Section 11.4.3.
For more details on the end codes (error codes) that are stored in the buffer memory of the Ethernet module, see Section 11.4.4.

<table>
<tr><th rowspan="2">End code</th><th rowspan="2">Description</th><th rowspan="2">Processing</th><th colspan="2">Communication function</th></tr>
<tr><th>Fixed</th><th>MC</th></tr>
<tr><td>00_H</td><td>• Normal completion</td><td>—</td><td>○</td><td>○</td></tr>
<tr><td>50_H</td><td>• Codes for command/response type of subheader are not within the specifications.
<table>
<tr><th>Communication processing</th><th>Command/ response type</th></tr>
<tr><td>Communication using fixed buffer</td><td>60_H</td></tr>
<tr><td>Communication using MC protocol</td><td>00_H to 05_H
13_H to 16_H</td></tr>
</table>
• In communication using the fixed buffer, if the data length setting is less than the actual data count, the remaining data is determined as the second data and processed. In this case, a subheader undefined command type error may occur.</td><td>• Check and correct command/response type set by an external device. (The Ethernet module automatically adds command/response type; the user does not need to set these.) See NOTE in Section 11.4.4.
• Check and correct the data length.</td><td>○</td><td>○</td></tr>
<tr><td>54_H</td><td>• When “ASCII code communication” is selected (BFM #24b1: ON) in the [Communication data code settings] of operational settings parameters of FA Configurator-EN, ASCII code data that cannot be converted to binary code was received from an external device.</td><td>• Check and correct the send data of the external device.</td><td>○</td><td>○</td></tr>
<tr><td>56_H</td><td>• Device designation from an external side is incorrect.</td><td>• Correct the device designated.</td><td></td><td>○</td></tr>
<tr><td rowspan="2">57_H</td><td>• The number of points for a command designated by an external device exceeds the maximum number of processing points for each processing (number of processes that can be executed per communication).
• Head device number to the designated points exceeds the maximum addresses (device number).
• When performing batch read/write operations on C200 to C255, the address (device numbers), which was designated with the odd number of points, is exceeded.</td><td>• Correct the designated points or device number.</td><td></td><td>○</td></tr>
<tr><td>• Byte length of a command does not conform to the specifications.
• When writing data, the set number of data points written is different from the value of the designated number.</td><td>• Check the data length of the command and redo the data setting.</td><td></td><td>○</td></tr>
<tr><td rowspan="2">58_H</td><td>• Head device number of a command designated by an external device is set outside the range that can be designated.</td><td>• Designate the appropriate values within the range that are allowed for each processing.</td><td></td><td>○</td></tr>
<tr><td>• A word device is designated in a command for bit devices.
• The head number of bit devices is designated by a value other than a multiple of 16 in a command for word devices.</td><td>• Correct the command or the designated device.</td><td></td><td>○</td></tr>
<tr><td>$5B_H$</td><td>• The PLC and the Ethernet module cannot communicate.
• The PLC cannot process requests from an external device.</td><td>• Fix the faulty parts by referring to the abnormal codes appended to the end codes (see Section 11.4.3).</td><td></td><td>○</td></tr>
<tr><td>60_H</td><td>• Communication time between the Ethernet module and the PLC exceeded PLC monitoring timer value.</td><td>• Increase the monitoring timer value.</td><td></td><td>○</td></tr>
</table>

Fixed : Fixed buffer communication MC : Communication using MC protocol

11.4.3 Abnormal codes returned during communication using MC protocol

This section explains the abnormal codes (error codes) that are added to responses when communicating using MC protocol (An abnormal code is added only when an end code is "5B".)
For more details on the end codes (error codes) that are added to responses, see Section 11.4.2.
For more details on the end codes (error codes) that are stored in the buffer memory of the Ethernet module, see Section 11.4.4.



Error code (hexadecimal)	Error	Description of error	Corrective action
10H	PC number error	The PC number designated was not “FF”.	(1) Set the PC number to “FF"..
11H	Mode error	Poor communication between the Ethernet module and the PLC (1) After the Ethernet module receives a request successfully from an external device, the Ethernet module and the PLC could not communicate for some reason (noise, etc.).	(1) Communicate again. If an error occurs again, check noise, etc. and replace the Ethernet module, then communicate again.
18H	Remote error	Remote RUN/STOP not accessible. The PLC base module status is CPU ERR or PROG ERR.	(1) Check whether or not the base module has a CPU ERR (light on) or PROG ERR (flashing light).

11.4.4 Error codes stored in the buffer memory

When an error occurs at each data communication processing between the Ethernet module and an external device, the error code (abnormal code) is stored in the buffer memory of the Ethernet module. This section explains the contents of this type of errors and error handling procedures.

The "Storage destination" column in the error code list indicates the buffer memory where the applicable error code is stored.

The names used for the explanation indicated in the "Storage destination" column corresponding to the buffer memory error code storage areas shown in the table below. (Error codes whose storage destination is not written are returned to the external device.)

Note that the buffer memory may store error codes of the messages returned from the external device. For error codes not shown in this manual, refer to the manual of the external device and check the returned messages.

Name used for explanation	Buffer memory	Buffer memory address
Initial	Initial abnormal code area	BFM#105
Open	Open abnormal code area	BFM#124…
Fixed sending	Fixed buffer sending abnormal code area	BFM#125…
Connection	Connection end code/error log area	BFM#126…
Error code	Error code/end code area	BFM#229…
E-mail log	E-mail error log area	BFM#22649…

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	E-mail log BFM# 22649
02H	The content of these errors and the error handling for each error code is the same as the end codes (02H to 0060H) returned to the external device. Check the explanation of the applicable code shown in Section 11.4.1, and take actions accordingly.						○	
0050H						○	○	
0051H							○	
0052H					○	○	○	
0054H						○	○	
0055H						○	○	
0056H						○	○	
0057H						○	○	
0058H						○	○	
0059H						○		
005BH	Read and handle the error code and end code area.					○	○	
0060H						○	○	

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	E--mail log BFM# 22649
C001$_{H}$	• At initial processing, the IP address setting value of the Ethernet module is incorrect. • When using the router relay function, the setting value of the sub-net mask field is incorrect.	• Correct the IP address. Set the class to A/B/C. • Correct the sub-net mask.	○				○	
C002$_{H}$	At initial setting, some of the various timer setting values are outside the allowable range.	• Review and correct the necessary timer values at initial processing.	○				○	
C004$_{H}$	The setting value of the sub-net mask field is incorrect.	• Correct the sub-net mask and execute the initial processing again.	○				○	
C005$_{H}$	• The setting value of the default router IP address for the router relay function is incorrect. • Network address (network address after sub-net mask) of the default router IP address is different from the network address of the local station's Ethernet module IP address.	• Correct the default router IP address and execute the initial processing again. • Set the network address to the same network address as the local station's Ethernet module IP address.	○				○	
C006$_{H}$	The setting value of the sub-net address for the router relay function is incorrect.	• Correct the sub-net address and execute the initial processing again.	○				○	
C007$_{H}$	• The setting value of the router IP address for the router relay function is incorrect. • Network address (network address after sub-net mask) of router IP address is different from the network address of the local station's Ethernet module IP address.	• Correct the router IP address and execute the initial processing again. • Set the network address to the same network ID as the local station's Ethernet module IP address.	○				○	
C010$_{H}$	At open processing, the setting value of the Ethernet module port number is incorrect.	• Correct the port number.	○	○			○	
C011$_{H}$	At open processing, the setting value of an external device's port number is incorrect.	• Correct the port number.		○	○		○	
C012$_{H}$	The port number set is used in a connection already opened by TCP/IP.	• Review and correct the port numbers of the Ethernet module and external device.		○			○	
C013$_{H}$	The port number used in a connection already opened is set in UDP/IP open processing.	• Review and correct the port number of the Ethernet module.		○			○	
C014$_{H}$	Initial processing and open processing of the Ethernet module is not completed.	• Execute the initial processing and open processing.			○		○	
C015$_{H}$	At open processing, the setting value of an external device's IP address is incorrect.	• Correct the IP address. Set the class to A/B/C.		○	○		○	
C016$_{H}$	The pairing open processing of the connection (or the next connection) designated for pairing open has already been completed.	• Check that the open processing of both of the target connections for pairing open is not executed. • Review the combination for pairing open.		○			○	
C017$_{H}$	A connection could not be established during the open processing of a TCP connection.	• Check the operation of the external device. • Check the open processing of the external device. • Correct the open settings of the communication parameters. • Review the Ethernet module's port number and the IP address/port number and open system of the external device. • Check that the connection cable is not dislocated.		○			○	
C018$_{H}$	The setting value of an external device's IP address is incorrect. ＊ When TCP is used, FFFFFFFF$_{H}$ cannot be set as an IP address.	• Correct the IP address.		○			○	

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	E-mail log BFM# 22649
C020$_H$	Data length exceeds the allowable range.	• Correct the data length. • If the data transmitted is larger than the allowable size, divide and then send it.			○		○	
C021$_H$	An abnormal end response was received after a transmission using fixed buffers.	• Read the end code of the response from the connection end code/error log area, and handle as needed.			○		○	
C022$_H$	• A response could not be received within the response monitoring timer value. • The applicable connection was closed while waiting for a response.	• Check the operation of the external device. • Review and correct the response monitoring timer value. • Check the open status of the applicable connection.			○		○	
C023$_H$	• The open processing for the applicable connection is not completed. • The applicable connection is closed.	• Execute the open processing of the applicable connection.			○		○	
C030$_H$	A sending error occurred.	• Check the operation of the external device. • Send after an arbitrarily selected time has elapsed because packets may be congested on the line. • Check that the connection cable is not dislocated.		○	○		○	
C032$_H$	A TCP ULP timeout error occurred in the TCP/IP communication. (An external device does not return ACK.)	• Check the operation of the external device. • Correct the TCP ULP timeout value and execute the initial processing again. • Send after an arbitrarily selected time has elapsed because packets may be congested on the line. • Check that the connection cable is not dislocated.		○	○		○	
C033$_H$	An external device side with the set IP address does not exist.	• Review and correct the external device's IP address and the Ethernet address. • If the external device has the ARP function, set the default value. If not, set the Ethernet address of the external device. • Check the operation of the external device. • Send after an arbitrarily selected time has elapsed because packets may be congested on the line. • Check that the connection cable is not dislocated. • Just after powering ON, there are cases when the connected hub is not booted, so send will occur after an arbitrary time period has elapsed.		○	○		○	
C035$_H$	The existence of an external device could not be confirmed within the response monitoring timer value.	• Check the operation of the external device. • Review and correct each setting value for the existence confirmation. • Check that the connection cable is not dislocated.		○	○		○	
C040$_H$	• Not all the data could be received within the response monitoring timer value. • Sufficient data for the data length could not be received. • The remaining part of a message divided at the TCP/IP level could not be received within the response monitoring timer value.	• Review and correct the data length of the communication data. • Review and correct each setting value at the initial processing because the packets may be congested on the line. • Send the same data from the external device again.		○	○		○	

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial	Open	Fixed sending	Connection	Error code	E-mail log
			BFM# 105	BFM# 124	BFM# 125	BFM# 126	BFM# 229	BFM# 22649
C041$_{H}$	When TCP is used, the checksum of the receive data is incorrect.	• Review the checksum on the external device side and send the correct value. • Investigate the conditions of the line (noise, distance between the line and power line, contact of each device, etc.)			○		○	
C042$_{H}$	When UDP is used, the checksum of the receive data is incorrect.				○		○	
C043$_{H}$	The checksum in header of IP packet received is incorrect.				○		○	
C044$_{H}$ to C048$_{H}$	An error packet of ICMP was received.	• Check the operation of the external device. • Check that the connection cable is not dislocated.			○		○	
C049$_{H}$	An error packet of ICMP was received.	• Check the operation of the external device. • Send after an arbitrarily selected time has elapsed because packets may be congested on the line. • Check that the connection cable is not dislocated. • Correct the IP assembly timer value of the external device existence timer timeout.			○		○	
C04A$_{H}$	An error packet of ICMP was received. (An IP assembly timeout error occurred in an external device.)				○		○	
*1 C04B$_{H}$	An IP assembly timeout error occurred. (The remaining part of divided data could not be received and a timeout occurred.)	• Check the operation of the external device. • Send after an arbitrarily selected time has elapsed because packets may be congested on the line. • Check that the connection cable is not dislocated. • Correct the IP assembly timer value and execute the initial processing again.			○		○	
C04C$_{H}$	Cannot send because there is no space in the internal buffer, e.g. the IP header buffer.	• Send the same data again and check that the response is received.			○		○	
C050$_{H}$	ASCII code data that cannot be converted to binary code is received when ASCII code communication is set in the operational settings of the Ethernet module.	• Select binary code communication in the operational settings, and restart the Ethernet module. • Correct the data sent from the external side and send again.			○		○	
C051$_{H}$ to C054$_{H}$	The number of read/write points is outside the allowable range.	• Correct the number of read/write points and send to the Ethernet module again.			○		○	
C056$_{H}$	• Read/write request exceeds the maximum address. • Address is 0.	• Correct the head address or the number of read/write points and send to the Ethernet module again. (The maximum address must not be exceeded.)			○		○	
C059$_{H}$	Incorrect designation of command and subcommand.	• Review the content of the request.			○		○	
C05A$_{H}$	The Ethernet module cannot read from/write to the designated device.	• Examine the device to be read/written.			○		○	
C05B$_{H}$					○		○	
C05C$_{H}$	The content of a request is incorrect. (Requesting read/write in bit units to word devices.)	• Correct the content of the request and send to the Ethernet module again. (Correction of subcommand, etc.)			○		○	
C05E$_{H}$	Communication time between the Ethernet module and PLC exceeded the monitoring timer.	• Increase the monitoring timer value. • Check whether or not the PLC operates normally.			○		○	

*1 When this error occurs, the COM.ERR light turns on, but turns off again about one second later.

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial	Open	Fixed sending	Connection	Error code	E-mail log
			BFM# 105	BFM# 124	BFM# 125	BFM# 126	BFM# 229	BFM# 22649
C05FH	The request could not be executed on the target PLC.	• Correct the network number and PC number. • Correct the content of the read/write request.			○		○	
C060H	The content of a request is incorrect. (Incorrect data was designated for bit devices, etc.)	• Correct the content of the request and send to the Ethernet module again. (Correction of data, etc.)			○		○	
C072H	The content of a request is incorrect. (Requested read/write in bit units to word devices.)	• Check whether the content can be requested to the target PLC. • Correct the content of the request and send it to the Ethernet module again.			○		○	
C073H	Request was not supported by the Ethernet module of the target PLC.	• Review the content of the request.			○		○	
C086H	A message exceeding the receive message size was received.	• Correct the send message size from the requesting source.					○	
C091H	System Error (Flash ROM write error)	(＊1)						
C093H	System Error (Sum of the parameters written to flash ROM is inconsistent)	(＊1)						
C0B5H	Data that could not be processed by the PLC/Ethernet module was designated.	• Review the content of the request. • Cancel the current request.					○	
C0B9H	The open processing of the applicable connection is has not been completed.	• Execute the open processing. • Check the operation of the external device.			○		○	
C0BAH	Cannot acknowledge sending request since the close processing is being executed via the CLOSE instruction.	• Execute the open processing and make a sending request.			○		○	
C0BBH	System error • The OS detected any error.	(＊1)						
C0BCH	Designated communication line is closed.	• Open the communication line. • Review the target connection number.			○	○	○	
C0BDH	Cannot send by acknowledging continuous requests.	• Check whether or not requests are made continuously without waiting for responses.			○	○	○	
C0BEH C0BFH	System error • The OS detected any error.	(＊1)						
C0C0H	The receiving completion confirmation signal was turned on when the receiving completion signal was not on.	• Review and correct the program.					○	
C0C1H	The transmission interval of UDP is too short.	• Check whether or not sending requests are repeated. • Make the sending interval longer.					○	
C0C2H to C0C3H	System error • The OS detected any error.	(＊1)						
C0C4H	Re-initialization was started during communication	• Execute re-initialization after closing all connections.			○			○
C0C5H	• A sending request was made to an external device whose class/network address are different from those of the local station when the router relay function is not used. • The setting of the router relay parameter is incorrect.	• Execute the initial processing by setting that the router relay function should be used. • Set the correct data for the router relay parameter. • Correct the IP address of the external device and execute the open processing. • Check that the network address is correct. When it is changed, execute the initial processing again.		○	○		○	
C0C6H	System error • The OS detected any error.	(＊1)						

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	E-mail log BFM# 22649
C0C7$_H$	An Ethernet module system error occurred.	• Execute the initial processing again. • Execute the processing by referring Section 11.5 POINT (3).	○		○		○	
C0C8$_H$ to C0CA$_H$	System error • The OS detected any error.	(＊1)						
C0CB$_H$	Another sending request was made when the sending processing has not been completed.	• Make the next sending request after the previous sending is completed.			○		○	
C0CC$_H$ C0CF$_H$	System error • The OS detected any error.	(＊1)						
C0D9$_H$	Incorrect subcommand value was designated.	• Correct the designated value for the subcommand.					○	
C0DA$_H$	A response to the PING test could not be received within the time of the communication time check.	• Review the IP address/host name of the Ethernet module for the PING test. • Change the status of the Ethernet module for the PING to allow communication (to the status after the initial processing is completed).						
C0DB$_H$	There is an error in the IP address/host name of the Ethernet module for the PING test.	• Review the IP address/host name of the Ethernet module for the PING test.						
C0DC$_H$ C0DD$_H$	System error • The OS detected any error.	(＊1)						
C0DF$_H$	System error • The OS detected any error.	(＊1)						
C0E0$_H$ to C0EF$_H$	An error was detected in the PLC.	• Check the connection cables between the PLC and extended I/Os or special module. • Check that no error has occurred in the PLC. If an error is found, take corrective actions according to error description of the PLC. • Replace the PLC and/or special function block.					○	
C0F0$_H$	An Ethernet module RAM abnormality was detected in the hardware test.	• Conduct a hardware test again. If an abnormality is detected again, the Ethernet module hardware may be faulty. Consult your nearest branch office or dealer with the details of the errors.					○	
C0F1$_H$	An Ethernet module ROM abnormality was detected in the hardware test.	• Conduct a hardware test again. If an abnormality is detected again, the Ethernet module hardware may be faulty. Consult your nearest branch office or dealer with the details of the errors.					○	
C0F4$_H$ to C0F6$_H$	System error • The OS detected any error.	(＊1)						
C0F8$_H$	Attempted to access to a buffer memory No. for which no base module of the PLC exists.	• Check the sequence program again. • Check whether or not there is a contact failure on the extension connector. • Check whether or not there are any factors possibly causing noise. • The Ethernet module hardware may be faulty. Consult your nearest dealer with the details of the error	○	○	○	○	○	○

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	E-mail log BFM# 22649
C0F9$_H$	Handshake error on TO instruction with a base module	• Check whether or not there is a contact failure on the extension connector. • Check whether or not there are any factors possibly causing noise. • The Ethernet module hardware may be faulty. Consult your nearest dealer with the details of the errors.	○	○	○	○		
C0FA	Base module 5V error • There is no power to the base module • The extension connector is not connected • The Ethernet module was turned on before the base module.	• Check that power to the base module is ON. • Check that there are no connection errors on the extension connector. • Turn on the base module and Ethernet module at the same time. • There may be a hardware problem with the Ethernet module or the base module. Please inform your nearest branch office or agent with a detailed description of the problem.					○	
C0FB	Error detected at base module • An error is occurring at the base module • The extension connector is not connected	• Check the sequence program. • Check that there are no connection errors with the extension connector. • There may be a hardware problem with the Ethernet module or the base module. Please inform your nearest branch office or agent with a detailed description of the problem..					○	
C100$_H$	System error • The OS detected any error.	(＊1)						
C101$_H$	A response could not be received from the DNS client.	• Check the address of the DNS server. • Check whether or not it is possible to communicate with the DNS server using the Ping command. • Check that the IP addresses of the local station and DNS server are in the same class. (If the class is different, check the router setting.)						○
C102$_H$	A response from the SMTP layer could not be received.	• Check that the SMTP server name is registered in DNS. • Delete the SMTP server name, change to the IP address setting, and check the operation. • Check whether or not it is possible to communicate with the SMTP server using the Ping command.						○
C103$_H$	DNS settings incorrect.	• Check the DNS mail address. • Check the content of the DNS setting.						○
C104$_H$ to C106$_H$ C110$_H$	System error • The OS detected any error.	(＊1)						
C111$_H$	A response could not be received from the DNS client.	• Check cable, hub, etc. • Check whether or not it is possible to communicate with the DNS server using the Ping command.						○
C112$_H$	A response from the POP3 layer could not be received.	• Check that the POP3 server name is registered in DNS. • Delete the POP3 server name, change to the IP address setting, and check the operation. • Check whether or not it is possible to communicate with the POP3 server using the Ping command.						○

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	E-mail log BFM# 22649
C113H	An e-mail was received that did not have an attached file. (This will generate when the attached file is not read normally.)	• Designate an attached file on the sending side. • Check the program on the sending side. • Check that the sending side has the same e-mail specifications as the Ethernet module. (encode/decode, file format, etc.) • A server with unknown destination was received from the SMTP server. • HTML-formatted mail was received.						○
C114H	An e-mail was received whose attached file name was invalid.	• Check on the sending side whether the extension of the attached file is “bin” or "asc". • Check whether or not the mail is compressed or encrypted. • A server with unknown destination was received from the SMTP server.						○
C115H to C118H	System error • The OS detected any error.	(＊1)						
C119H	There is no received mail.	• Check the mail information storage area of the buffer memory (BFM#22640), then read any received mail that is on the server.						
C11AH	Failed to convert an e-mail to be received.	• Check whether or not the mail is compressed or encrypted. • Check that the sending side has the same e-mail specifications as the Ethernet module. (encode/decode, file format, etc.) • Check whether or not the file was divided on the sending side.						○
C11BH	An e-mail was sent and an error mail was received from the mail server of the destination.	• A server with unknown destination was received from the SMTP server. (Received mail is stored in the mail buffer.) • Check that the part before "@" is correct in the mail address setting of the parameter settings. • Check that the part before "@" is registered to the destination mail server.						○
C11CH	Mail address not found.	• Check whether the mail setting of the parameter setting is correct or not. • When the mail server and Ethernet module are connected via the router, check whether the router setting is correct or not. • Test-send a mail to the address where it will be received without fail. When the test is normally completed, recheck whether the domain name after "@" is correct or not.						○
C11DH	The size of the attached file exceeded the allowable size.	• Check that the attached file is less than 2 k word. • Check that the sending side did not divide the file.						○
C120H	Could not open the SMTP server.	• Check that the port number of the SMTP server = 25. • Check whether or not it is possible to communicate with the SMTP server using the Ping command.						○
C121H	Cannot communicate with the SMTP server. (Error response)	• Check if the SMTP server is busy.						○
C122H	Cannot communicate with the SMTP server. (Abort)	• Check if the SMTP server is busy.						○
C123H	Cannot communicate with the SMTP server. (Reset response)	• Check if the SMTP server is busy.						○

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial	Open	Fixed sending	Connection	Error code	E-mail log
			BFM# 105	BFM# 124	BFM# 125	BFM# 126	BFM# 229	BFM# 22649
C124$_H$	A response from the SMTP server timed out.	• Check whether or not the SMTP server is faulty. • Check whether or not there is too much load on the network.						○
C125$_H$	Forcefully disconnected from the SMTP server.	• Check whether or not the SMTP server is faulty.						○
C126$_H$	Could not close the SMTP server.	• Check whether or not the SMTP server is faulty. • Check whether or not there is too much load on the network.						○
C127$_H$	Closing the SMTP server gave an error response.	• Check whether or not the SMTP server is faulty.						○
C130$_H$	Communication channel is closed because the service is not available.	• Check the status of the SMTP server.						○
C131$_H$	The SMTP server was performing processing and an error response was received.	• Check if a user name not registered in the server was designated. • Send again after arbitrary set time has passed.						○
C132$_H$	The SMTP server was performing processing and an error response was received. (Local error)	• Check the status of the SMTP server.						○
C133$_H$	The SMTP server was performing processing and an error response was received. (Insufficient memory area)	• Check the status of the SMTP server.						○
C134$_H$ to C137$_H$	System error • The OS detected any error.	(＊1)						
C138$_H$	The SMTP server was performing processing and an error response was received. (Mailbox not found)	• Check that the Ethernet module's mail address is set correctly.						○
C139$_H$	System error • The OS detected any error.	(＊1)						
C13A$_H$	The SMTP server was performing processing and an error response was received. (Exceeded the allocation of memory area)	• Check the status of the SMTP server.						○
C13B$_H$	The SMTP server was performing processing and an error response was received. (Illegal mail box name)	• Check that the Ethernet module's mail address is set correctly.						○
C13C$_H$	System error • The OS detected any error.	(＊1)						
C140$_H$	Could not open the POP3 server.	• Check that the port number of the POP3 server = 110. (For the Ethernet module, this is fixed to 110.) • Check whether or not it is possible to communicate with the POP3 server using the Ping command.						○
C141$_H$	Cannot communicate with the POP3 server. (Error response)	• Check if the POP3 server is busy.						○
C142$_H$	Cannot communicate with the POP3 server. (Abort)	• Check if the POP3 server is busy.						○
C143$_H$	Cannot communicate with the POP3 server. (Reset response)	• Check if the POP3 server is busy.						○
C144$_H$	Could not receive a response from the POP3 server.	• Check whether or not the POP3 server is faulty. • Check whether or not there is too much load on the network. • Check that the correct password is being sent to the POP3 server.						○

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial	Open	Fixed sending	Connection	Error code	E-mail log
			BFM#	BFM#	BFM#	BFM#	BFM#	BFM#
			105	124	125	126	229	22649
C145H	Forcibly disconnected from the POP3 server.	• Check whether or not the POP3 server is faulty.						○
C146H	Could not close the POP3 server.	• Check whether or not the POP3 server is faulty. • Check whether or not there is too much load on the network.						○
C147H	Closing the POP3 server gave an error response.	• Check whether or not the POP3 server is faulty.						○
C150H	POP3 server verification error.	• Check the status of the POP3 server.						○
C151H	The Ethernet module's mail address (e-mail address setting parameter) is different from the account name in the mailbox on the server side.	• Check the account name in the mailbox on the server side and correct the mailbox account set for the Ethernet module.						○
C152H	The Ethernet module's password (e-mail setting parameter) is different from the password on the server side.	• Check the password on the server side and correct the password set for the Ethernet module.						○
C153H	An error occurred when getting the received mail list. (Failed to obtain the list of arrived mail at the POP3 server.)	• Reset the server inquiry time to the default value and reset or re-initialize the PLC.						○
C154H	An error occurred when receiving a mail. (Cannot read e-mail from the POP3 server.)	• Check whether or not the mail is compressed or encrypted. • Check that the sending side has the same e-mail specifications as the Ethernet module. (encode/decode, file format, etc.)						○
C160H	Received a response from the DNS server after timeout.	• Check whether or not there is too much load on the network. • Check the status of the DNS server.						○
C161H	Could not receive a response from the DNS server.							○
C162H	An error is returned from DNS server.	• Check if the DNS server's IP address setting is correct or not. • Check if the mail server name setting (SMTP server name, POP server name) is correct or not. • Check with the network administrator or similar person that the DNS function of the server set in the DNS setting" is being performed.						○
C163H								
C171H to C17FH								
C180H	Multiple instructions for mail communication • Mail receiving instruction was issued while receiving a mail. • Mail sending instruction was issued while sending a mail.	• Issue the instruction for sending or receiving a mail after checking the mail send or receive status.						○
C1A0H	An illegal request was made.	• Execute again. If the same error occurs, the Ethernet module's hardware may be faulty. Consult your nearest dealer with the details of the error.						
C1A2H	A response to a request could not be received.	• Review and correct the response wait time.						
C1A5H	The designation of the target station or clear target were incorrect.	• Correct the designated value of the target station or clear target.						
C1A6H	Incorrect connection number was designated.	• Designate the connection number within the range of 1 to 8.						
C1A7H	Incorrect network number was designated.	• Correct the designated value of the network number.						
C1ACH	Incorrect resent count was designated.	• Correct the designated value of the resent count.						

Error code (abnormal code)	Description of error	Error handling	Storage destination					
			Initial BFM# 105	Open BFM# 124	Fixed sending BFM# 125	Connection BFM# 126	Error code BFM# 229	zE-mail log BFM# 22649
C1AD$_H$	Incorrect data length was designated.	• Correct the designated value of the data length.						
C1AE$_H$	Incorrect mail sending/receiving data length and header length were designated.	• Correct the designated values of sending/receiving data length and header length. • Sending/receiving data length should be equal or longer than the header length.						
C1AF$_H$	Incorrect port number was designated.	• Correct the designated value of the port number.						
C1B0$_H$	The open processing of the designated connection is already completed.	• Execute the open processing after executing the close processing.						
C1B1$_H$	The open processing of the designated connection has not been completed.	• Execute the open processing.						
C1B2$_H$	The OPEN/CLOSE instruction is being executed in the designated connection.	• Execute after the OPEN/CLOSE instruction is completed.						
C1B3$_H$	Another sending/receiving instruction is being executed on the designated channel.	• Change the channel number. • Execute after the sending/receiving instruction is completed.						
C1B6$_H$	Incorrect mail destination number was designated.	• Review the designated value of the mail designation number. • Review the sending mail address setting parameter.						
C203$_H$	System error • The OS detected any error.	(＊1)						
C206$_H$	System error • The OS detected any error.	(＊1)						
C280$_H$	The set limit number for MELSOFT connections is exceeded	• Modify your open settings so that there are a total of 4 or fewer MELSOFT connections/MC protocol connections..					○	
C300$_H$	A response could not be received within the response monitoring timer value.	• Check the operation of the external device. • Review and correct the response monitoring timer value.					○	

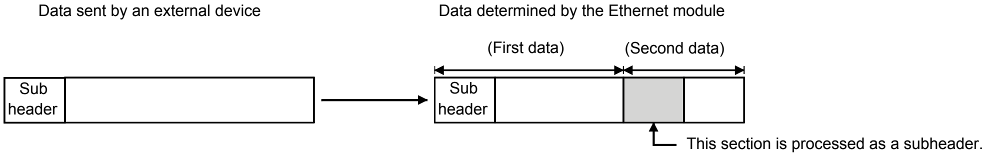
＊1 Take corrective action in the following procedure.

1) Check whether the expansion connectors are connected correctly.
2) Check whether the operating environment of the Ethernet module is within the general specifications range of the module.
3) Check whether the power supply capacity is sufficient or not.
4) Check whether the hardware of the Ethernet module, power supply block and PLC are normal according to the manuals of the corresponding modules.
   If any module is faulty, please request your local Mitsubishi service center or representative to repair.
5) If the problem cannot be solved through the above steps, please consult your local Mitsubishi service center or representative, explaining the operation/communication conditions at error occurrence and the information stored in the error log area within the buffer memory of the Ethernet module.

NOTE

Depending on the restrictions of the buffers of the local station and external device, data may be divided for communication.
Data received separately is restored (reassembled) by the Ethernet module to be communicated using the fixed buffer, etc. The received data is restored (reassembled) based on the data length in the communication data.
The Ethernet module performs the following processing if data among the communication data is incorrect.

(1) When communication is performed using fixed buffer (with procedure)
   (a) If the data length specified immediately after the subheader is smaller than the amount of text data received
      1) The data immediately following the part of the text data equivalent to the data length specified immediately after the subheader will be regarded as the next message.
      2) Since the header of each message will be a subheader, the Ethernet module performs processing according to the code in the subheader.
      3) If the subheader contains a code that is not recognized by the Ethernet module, the Ethernet module will send a response to the external device notifying it of abnormal completion.


Data sent by an external device
Data determined by the Ethernet module
(First data)
(Second data)
Sub header
Sub header
This section is processed as a subheader.

At this point, the Ethernet module returns a response containing a code obtained by changing the most significant bit of the code processed as subheader to 1.
For example, if the subheader field of a command is 65H, the subheader of the response will become E5H.

(b) If the data length specified immediately after the subheader is larger than the amount of text data received

1) The Ethernet module waits for the reception of the remaining missing data.

2) If the remaining data could be received within the time allotted by the response monitoring timer, the Ethernet module performs processing according to the code in the subheader.

3) If the remaining data could not be received within the time allotted by the response monitoring timer, the Ethernet module performs the following processing.

- Sends the ABORT (RST) instruction to the external device and closes the line.
- Notifies the PLC side about occurrence of an open error via the open error detection signal (BFM#28.b6=ON)
- Stores the error code in the open error code storage area. (The error code is not stored in the error log storage area.)

POINT

Designate the actual data size of the text field for "data length" specified in the application data field of a message sent from the external device to the Ethernet module.

The Ethernet module never sends text whose size is different from the specified data length to the external device.

(2) When communication is performed using fixed buffer (non-procedure)

Since no message data length is specified in non-procedure communication, the data received is stored in the receive buffer area as is.

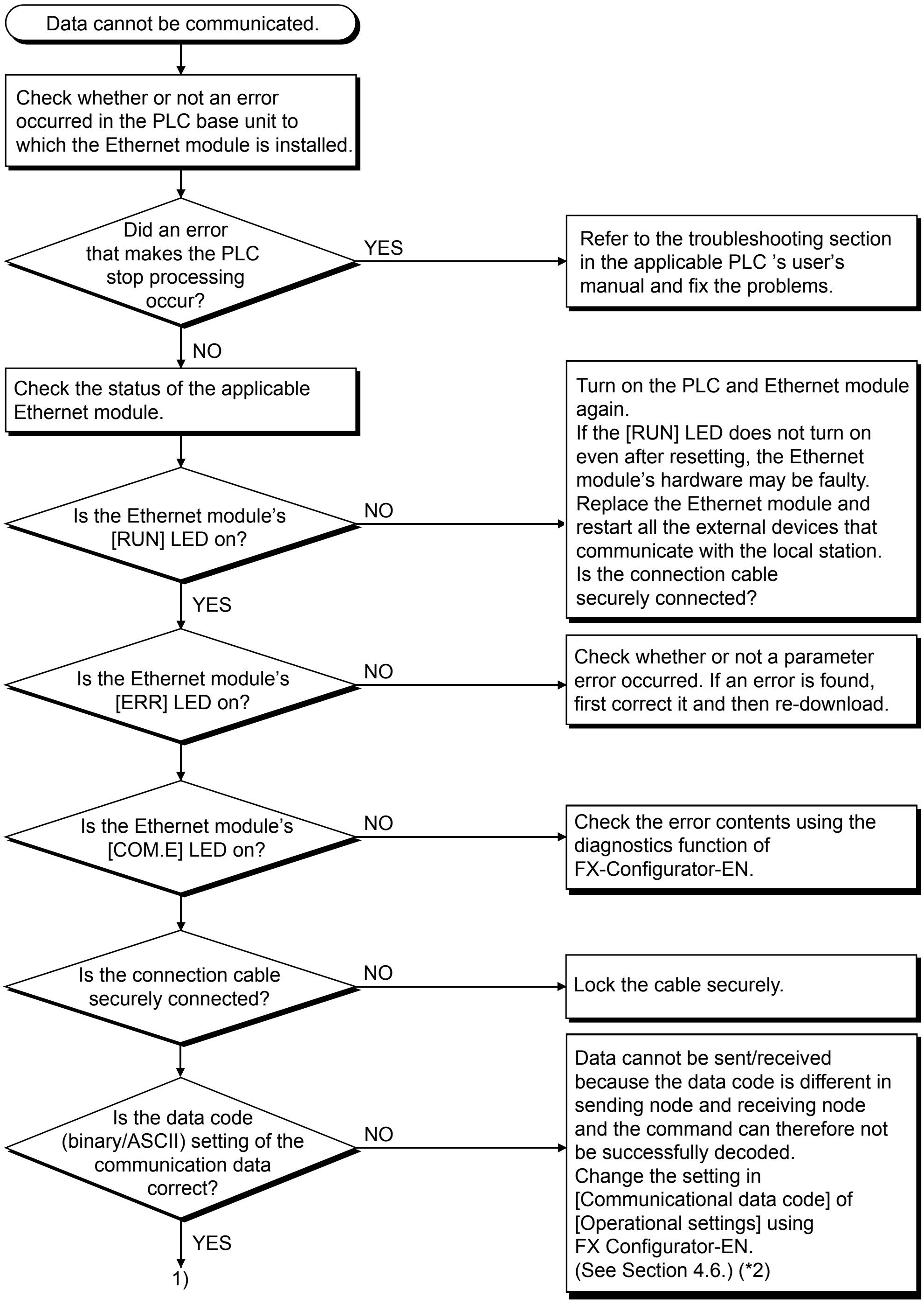
It is recommended to set up some method of checking that the data is received correctly. This can be achieved for instance by including the data length and data type code in the application data of the message, so that the number of bytes and type of application data can be identified on the receiving side.

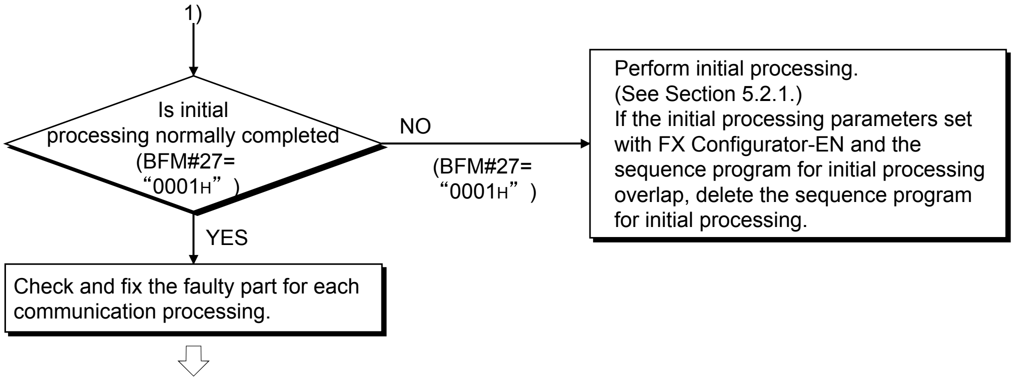
11.5 Troubleshooting Flowchart

This section explains some simple troubleshooting procedures when the Ethernet module and an external device have communication problems in a flowchart format. (*1)

POINT

If trouble occurs when using the Ethernet module, check the block status, error status and others for the Ethernet module using the Ethernet diagnostic function of FX Configurator-EN described in Section 11.2.2.


Data cannot be communicated.
Check whether or not an error occurred in the PLC base unit to which the Ethernet module is installed.
Did an error that makes the PLC stop processing occur?
YES
Refer to the troubleshooting section in the applicable PLC 's user's manual and fix the problems.
NO
Check the status of the applicable Ethernet module.
Is the Ethernet module's [RUN] LED on?
NO
Turn on the PLC and Ethernet module again.
If the [RUN] LED does not turn on even after resetting, the Ethernet module's hardware may be faulty.
Replace the Ethernet module and restart all the external devices that communicate with the local station.
Is the connection cable securely connected?
YES
Is the Ethernet module's [ERR] LED on?
NO
Check whether or not a parameter error occurred. If an error is found, first correct it and then re-download.
Is the Ethernet module's [COM.E] LED on?
NO
Check the error contents using the diagnostics function of FX-Configurator-EN.
Is the connection cable securely connected?
NO
Lock the cable securely.
Is the data code (binary/ASCII) setting of the communication data correct?
NO
Data cannot be sent/received because the data code is different in sending node and receiving node and the command can therefore not be successfully decoded.
Change the setting in [Communicational data code] of [Operational settings] using FX Configurator-EN.
(See Section 4.6.) (*2)
YES
1)

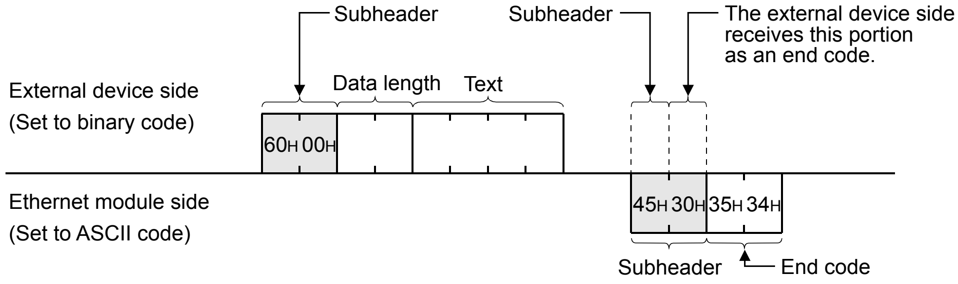

1)
Is initial processing normally completed (BFM#27= “0001H”)
NO
(BFM#27= “0001H”)
Perform initial processing.
(See Section 5.2.1.)
If the initial processing parameters set with FX Configurator-EN and the sequence program for initial processing overlap, delete the sequence program for initial processing.
YES
Check and fix the faulty part for each communication processing.

Sending error in fixed buffer communication ··········· See Section 11.5.1.
Receiving error in fixed buffer communication ········· See Section 11.5.2.
Error in the MC protocol communication ··············· See Section 11.5.3.
Send error during communication by e-mail ··········· See Section 11.5.4.
Receive error during communication by e-mail ········· See Section 11.5.5.

＊1 See Section 11.1 when the value of BFM#27 is "0000H" or when the display LED COM.ERR (communication error detection display) lights up.
(Check an error code that is stored in the buffer memory in reaction to an error occurrence, then check the contents of the error and take corrective actions by referring to Section 11.4.)

＊2 Error codes not found in the error code list may be returned to the external device side if the communication data settings on the Ethernet module side (see Section 4.6) and the data code settings on the external device side are different.
The Ethernet module cannot decode commands correctly if data with different data codes is received. The Ethernet module returns error responses according to the communication data code settings.

[example] When communication is performed using a fixed buffer.



＊3 If the FG terminals of the Ethernet module are not set, the communication line is closed (disconnected) due to the effects of noise, and as a result communication with an external device may not be performed.

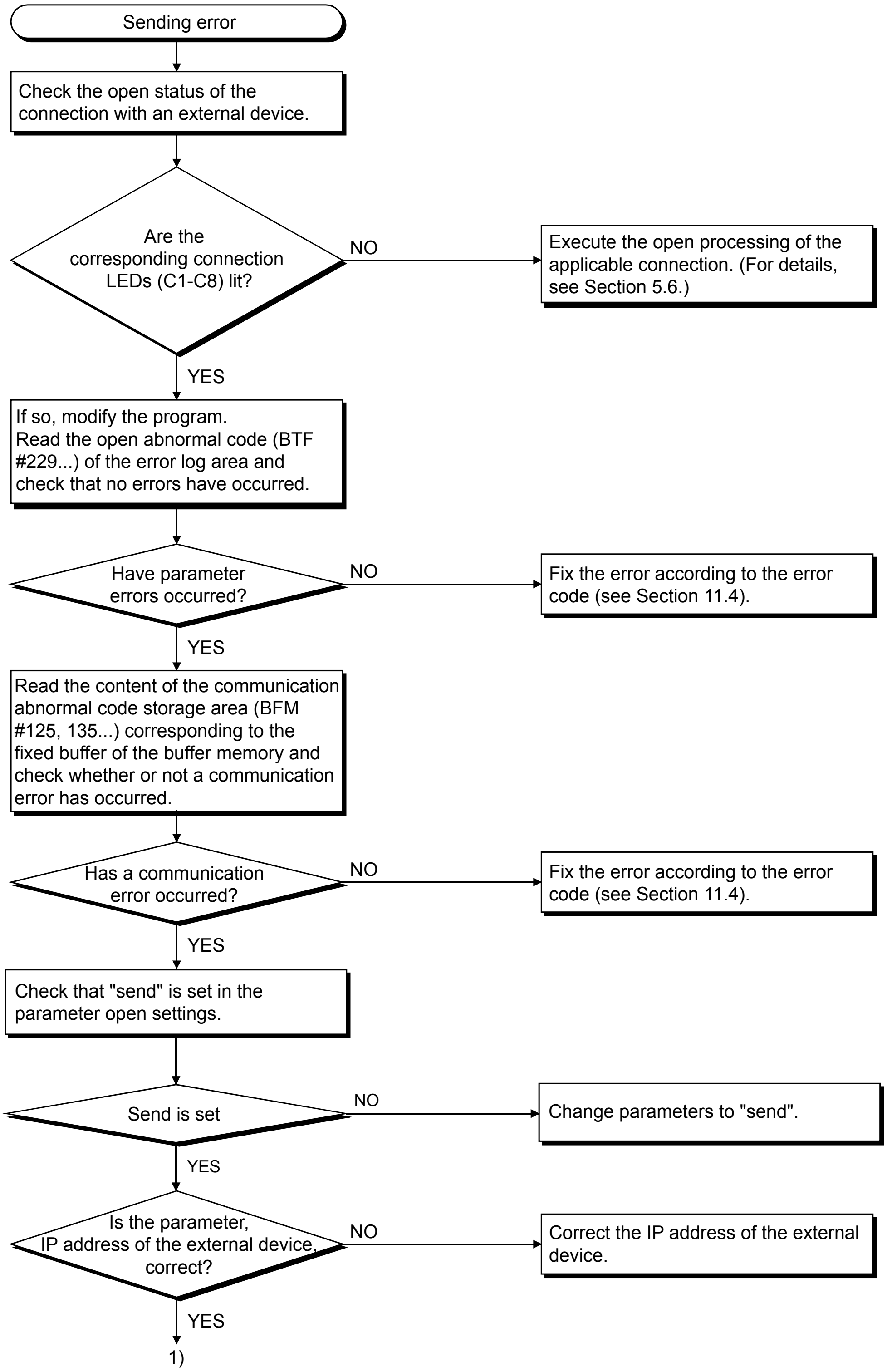
POINT

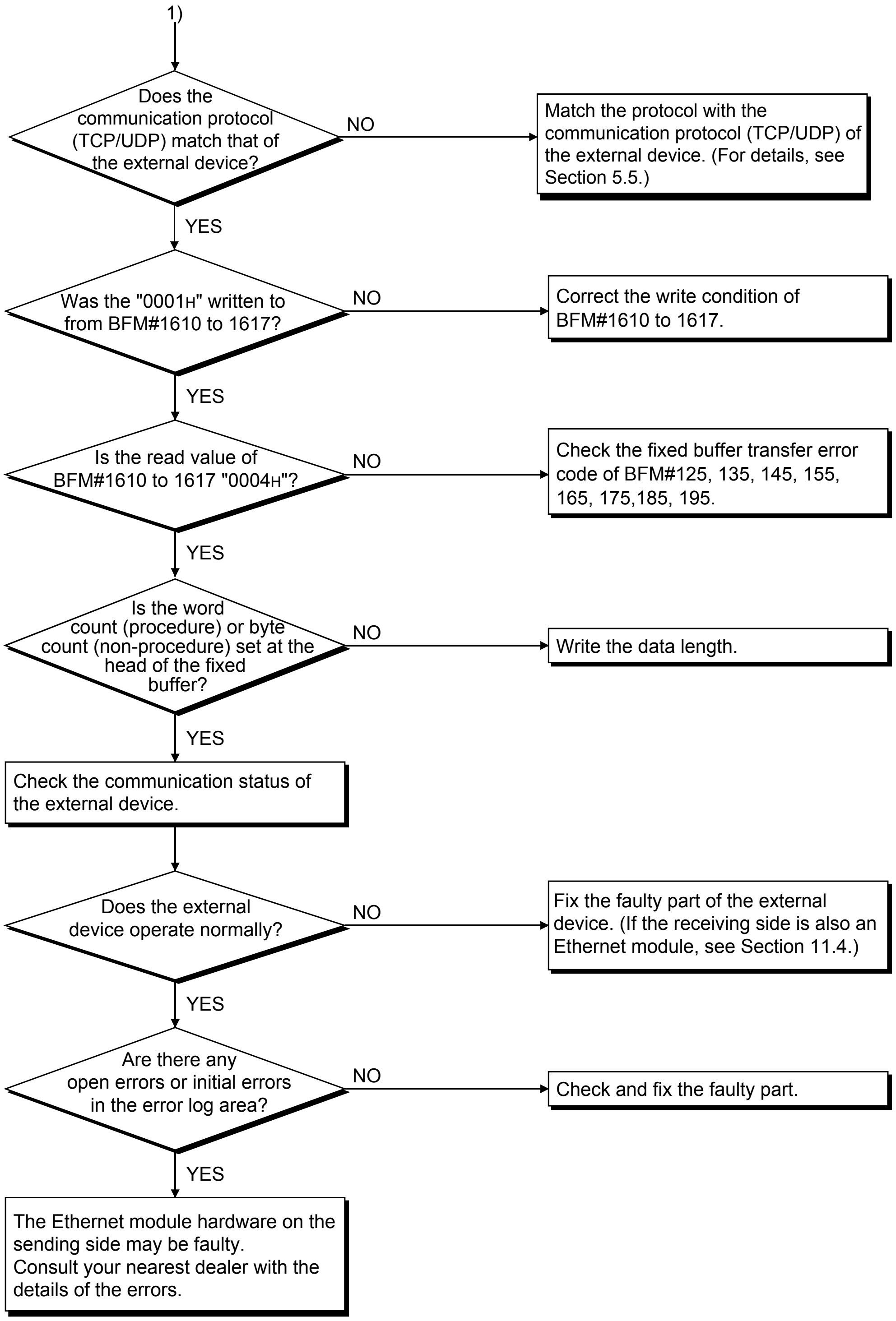
(1) When the Ethernet module is replaced due to an error, reboot the following external devices and restart data communication:
(If the external device retains the Ethernet address, it may be impossible to continue communication because when a module is replaced the Ethernet address changes.)
In the same way, when the external device (personal computer, etc.)is replaced, restart the Ethernet module.
- All external devices that communicated with the station whose Ethernet module was replaced.
- All external devices that communicated with other station's PLC via a station whose Ethernet module was replaced.

(2) When connecting a device to the Ethernet module, see the following sections for the required devices and connection method:
Section 2.2: Devices Required for Network Configuration
Section 4.4: Connecting to the Network

(3) If the Ethernet module could not receive messages sent from external devices frequently, check the values stored in the following buffer memory.
(a) Simultaneous transmission error detection count storage area (BFM#398 to 399) and Error/End code storage area for each error log block (BFM#229…)
When the error detection count number is high or when the error code C0C7H has been stored, high load may be applied to the Ethernet connection line to transfer data among connected devices.
To reduce the load to the Ethernet line, it is necessary to take corrective measures such as dividing the network or reducing the data transmission frequency.
Consult your network administrator and take appropriate measures.

11.5.1 Sending errors during fixed buffer communication (common to procedure exist and no procedure)


Sending error
Check the open status of the connection with an external device.
Are the corresponding connection LEDs (C1-C8) lit?
NO
Execute the open processing of the applicable connection. (For details, see Section 5.6.)
YES
If so, modify the program.
Read the open abnormal code (BTF #229...) of the error log area and check that no errors have occurred.
Have parameter errors occurred?
NO
Fix the error according to the error code (see Section 11.4).
YES
Read the content of the communication abnormal code storage area (BFM #125, 135...) corresponding to the fixed buffer of the buffer memory and check whether or not a communication error has occurred.
Has a communication error occurred?
NO
Fix the error according to the error code (see Section 11.4).
YES
Check that "send" is set in the parameter open settings.
Send is set
NO
Change parameters to "send".
YES
Is the parameter, IP address of the external device, correct?
NO
Correct the IP address of the external device.
YES
1)


1)
Does the communication protocol (TCP/UDP) match that of the external device?
NO
Match the protocol with the communication protocol (TCP/UDP) of the external device. (For details, see Section 5.5.)
YES
Was the "0001H" written to from BFM#1610 to 1617?
NO
Correct the write condition of BFM#1610 to 1617.
YES
Is the read value of BFM#1610 to 1617 "0004H"?
NO
Check the fixed buffer transfer error code of BFM#125, 135, 145, 155, 165, 175,185, 195.
YES
Is the word count (procedure) or byte count (non-procedure) set at the head of the fixed buffer?
NO
Write the data length.
YES
Check the communication status of the external device.
Does the external device operate normally?
NO
Fix the faulty part of the external device. (If the receiving side is also an Ethernet module, see Section 11.4.)
YES
Are there any open errors or initial errors in the error log area?
NO
Check and fix the faulty part.
YES
The Ethernet module hardware on the sending side may be faulty. Consult your nearest dealer with the details of the errors.

POINT

If the external device communicates normally, the following conditions occur.

For TCP communication

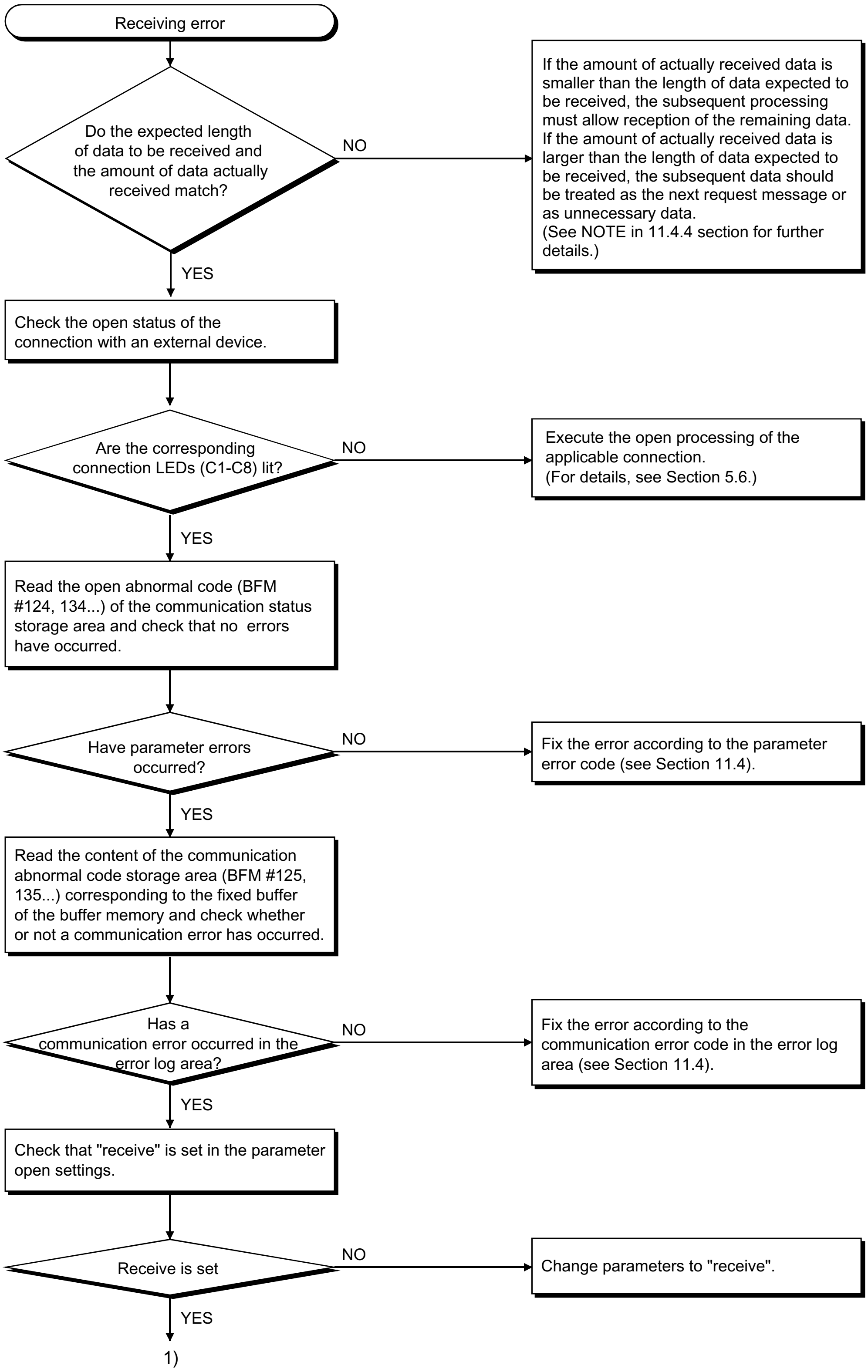
When the external device's open status is complete, LEDs C1-C8, which are in correspondence with connections 1-8, turn on.

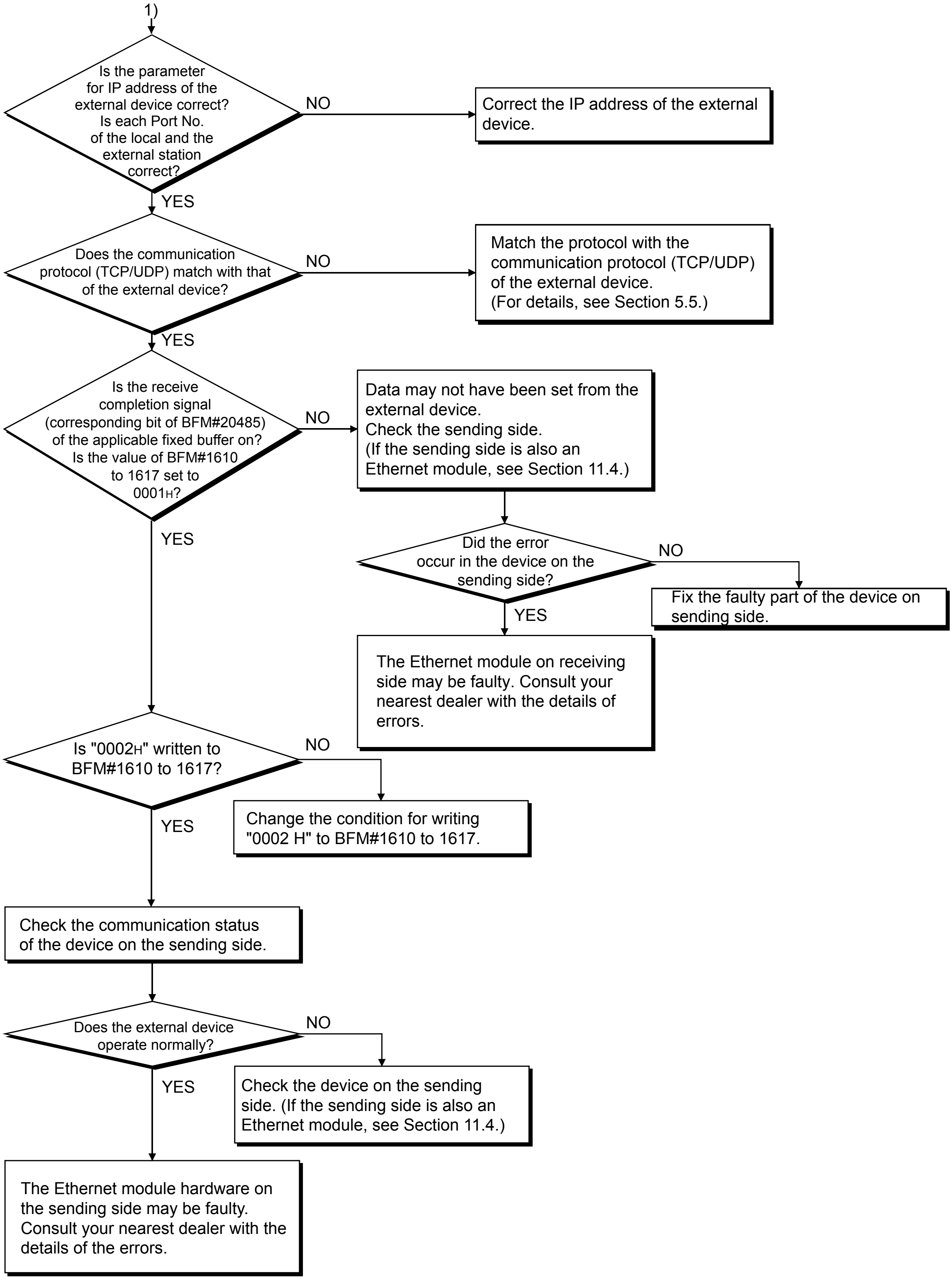
For UDP communication

When external device communication is possible, LEDs C1-C8, which are in correspondence with connections 1-8, turn on.

11.5.2 Receiving errors during fixed buffer communication (common to procedure exist and no procedure)

> **POINT**
>
> Perform error code reading while the connection is open.


Receiving error
Do the expected length of data to be received and the amount of data actually received match?
NO
If the amount of actually received data is smaller than the length of data expected to be received, the subsequent processing must allow reception of the remaining data. If the amount of actually received data is larger than the length of data expected to be received, the subsequent data should be treated as the next request message or as unnecessary data. (See NOTE in 11.4.4 section for further details.)
YES
Check the open status of the connection with an external device.
Are the corresponding connection LEDs (C1-C8) lit?
NO
Execute the open processing of the applicable connection. (For details, see Section 5.6.)
YES
Read the open abnormal code (BFM #124, 134...) of the communication status storage area and check that no errors have occurred.
Have parameter errors occurred?
NO
Fix the error according to the parameter error code (see Section 11.4).
YES
Read the content of the communication abnormal code storage area (BFM #125, 135...) corresponding to the fixed buffer of the buffer memory and check whether or not a communication error has occurred.
Has a communication error occurred in the error log area?
NO
Fix the error according to the communication error code in the error log area (see Section 11.4).
YES
Check that "receive" is set in the parameter open settings.
Receive is set
NO
Change parameters to "receive".
YES
1)



1)
Is the parameter for IP address of the external device correct? Is each Port No. of the local and the external station correct?
NO
Correct the IP address of the external device.
YES
Does the communication protocol (TCP/UDP) match with that of the external device?
NO
Match the protocol with the communication protocol (TCP/UDP) of the external device. (For details, see Section 5.5.)
YES
Is the receive completion signal (corresponding bit of BFM#20485) of the applicable fixed buffer on? Is the value of BFM#1610 to 1617 set to 0001H?
NO
Data may not have been set from the external device. Check the sending side. (If the sending side is also an Ethernet module, see Section 11.4.)
Did the error occur in the device on the sending side?
NO
Fix the faulty part of the device on sending side.
YES
The Ethernet module on receiving side may be faulty. Consult your nearest dealer with the details of errors.
YES
Is "0002H" written to BFM#1610 to 1617?
NO
Change the condition for writing "0002 H" to BFM#1610 to 1617.
YES
Check the communication status of the device on the sending side.
Does the external device operate normally?
NO
Check the device on the sending side. (If the sending side is also an Ethernet module, see Section 11.4.)
YES
The Ethernet module hardware on the sending side may be faulty. Consult your nearest dealer with the details of the errors.

POINT

If the external device communicates normally, the following conditions occur.

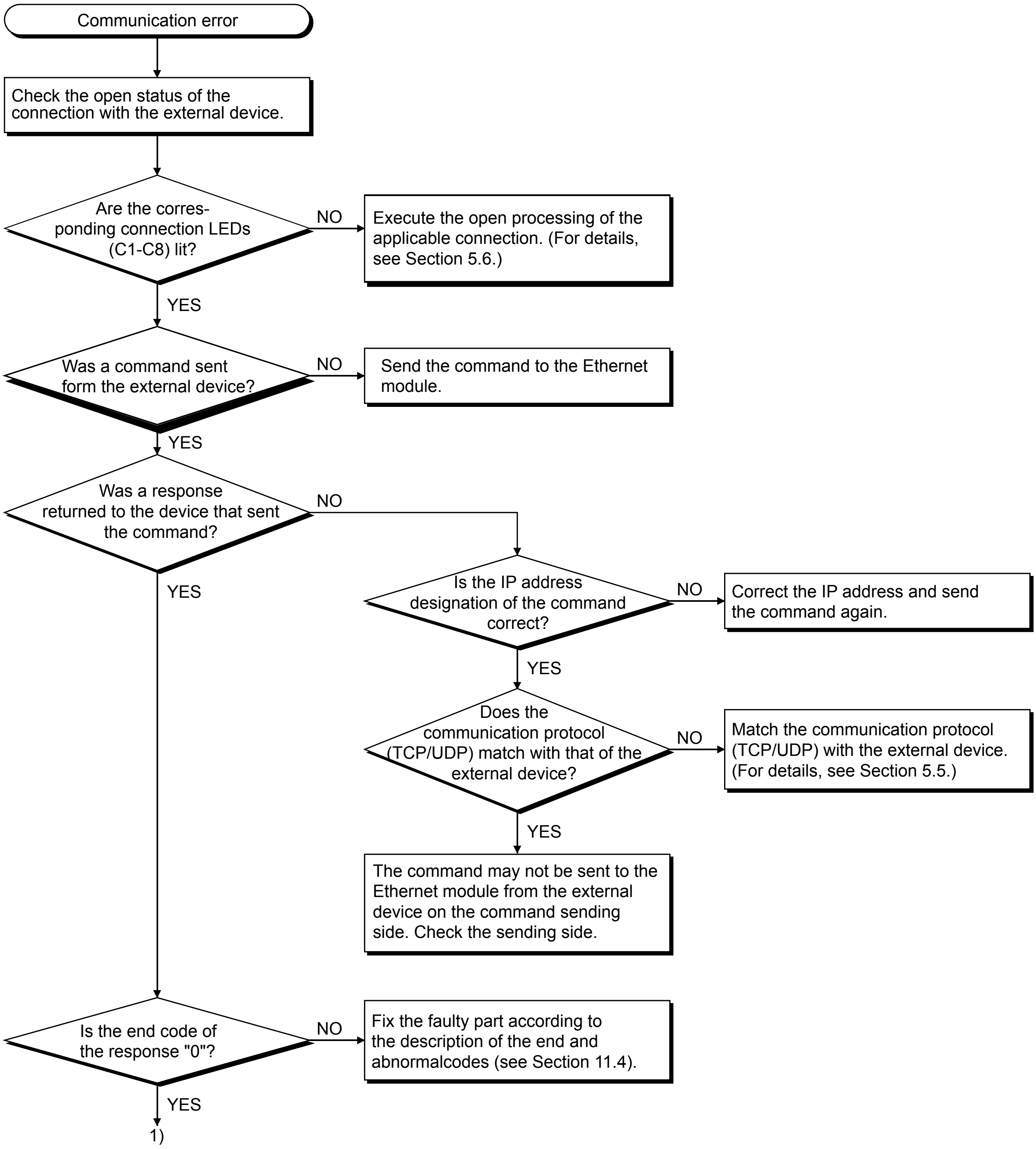
For TCP communication

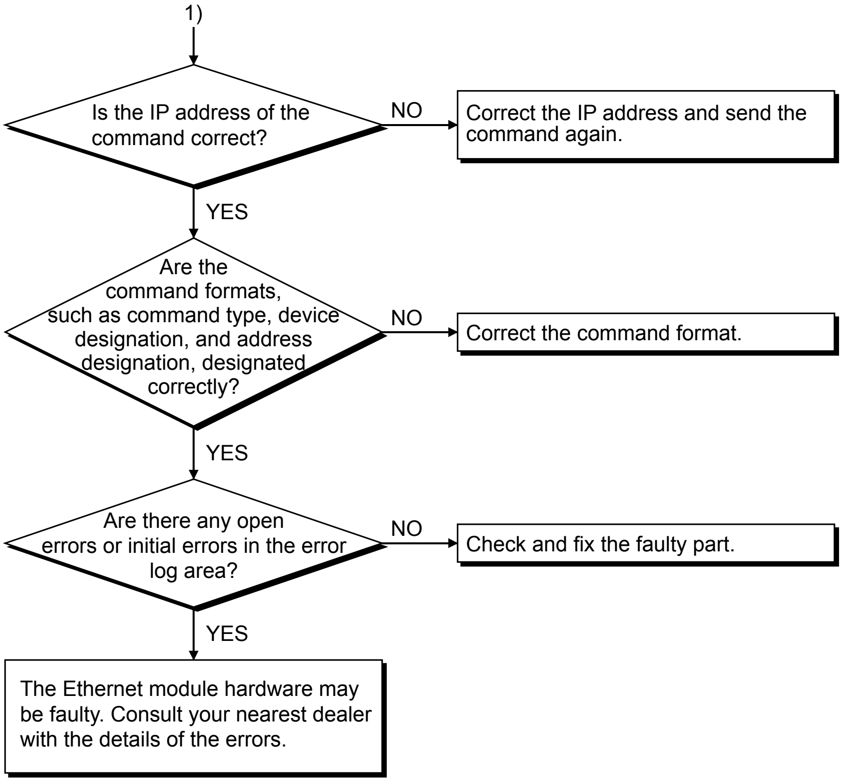
When the external device’s open status is complete, LEDs C1-C8, which are in correspondence with connections 1-8, turn on.

For UDP communication

When external device communication is possible, LEDs C1-C8, which are in correspondence with connections 1-8 turn on.

11.5.3 Errors in communication using MC protocol


Communication error
Check the open status of the connection with the external device.
Are the corres-ponding connection LEDs (C1-C8) lit?
NO
Execute the open processing of the applicable connection. (For details, see Section 5.6.)
YES
Was a command sent form the external device?
NO
Send the command to the Ethernet module.
YES
Was a response returned to the device that sent the command?
NO
Is the IP address designation of the command correct?
NO
Correct the IP address and send the command again.
YES
Does the communication protocol (TCP/UDP) match with that of the external device?
NO
Match the communication protocol (TCP/UDP) with the external device. (For details, see Section 5.5.)
YES
The command may not be sent to the Ethernet module from the external device on the command sending side. Check the sending side.
YES
Is the end code of the response "0"?
NO
Fix the faulty part according to the description of the end and abnormalcodes (see Section 11.4).
YES
1)


1)
Is the IP address of the command correct?
NO
Correct the IP address and send the command again.
YES
Are the command formats, such as command type, device designation, and address designation, designated correctly?
NO
Correct the command format.
YES
Are there any open errors or initial errors in the error log area?
NO
Check and fix the faulty part.
YES
The Ethernet module hardware may be faulty. Consult your nearest dealer with the details of the errors.

POINT

- If the external device communicates normally, the following conditions occur.

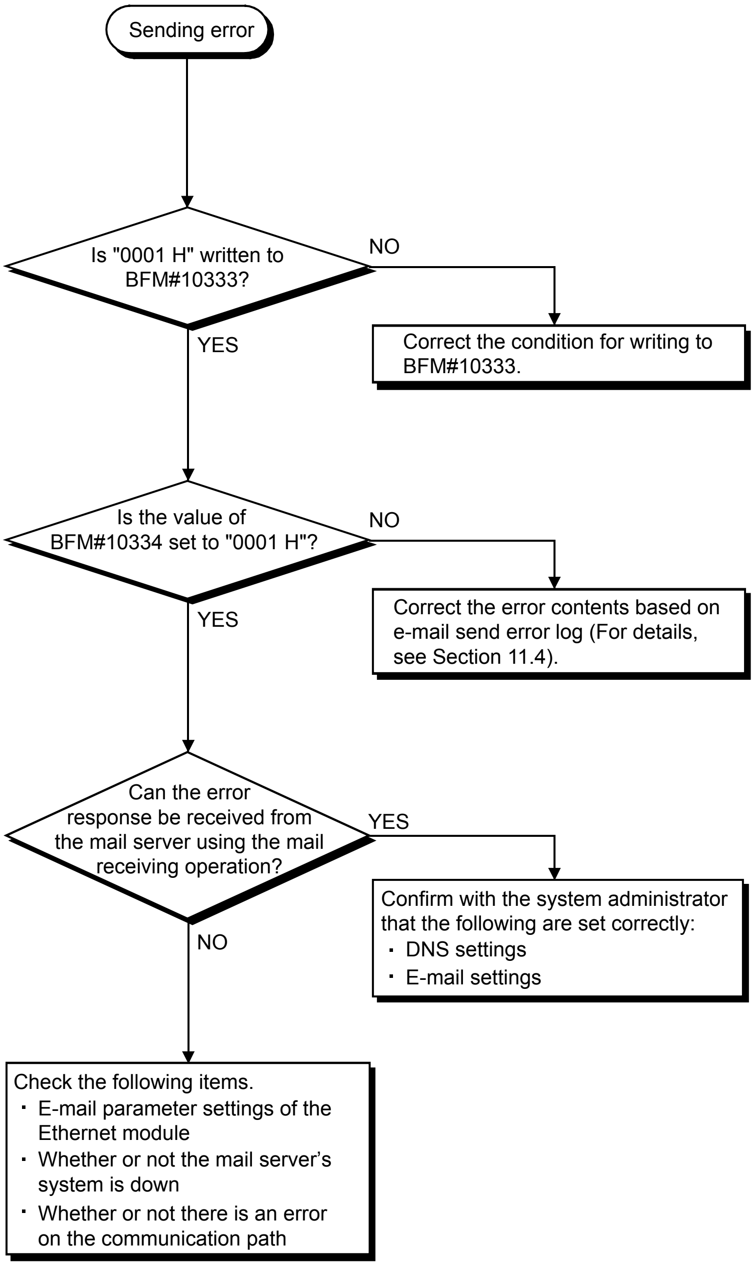
  For TCP communication

  When the external device's open status is complete, LEDs C1-C8, which are in correspondence with connections 1-8, turn on.

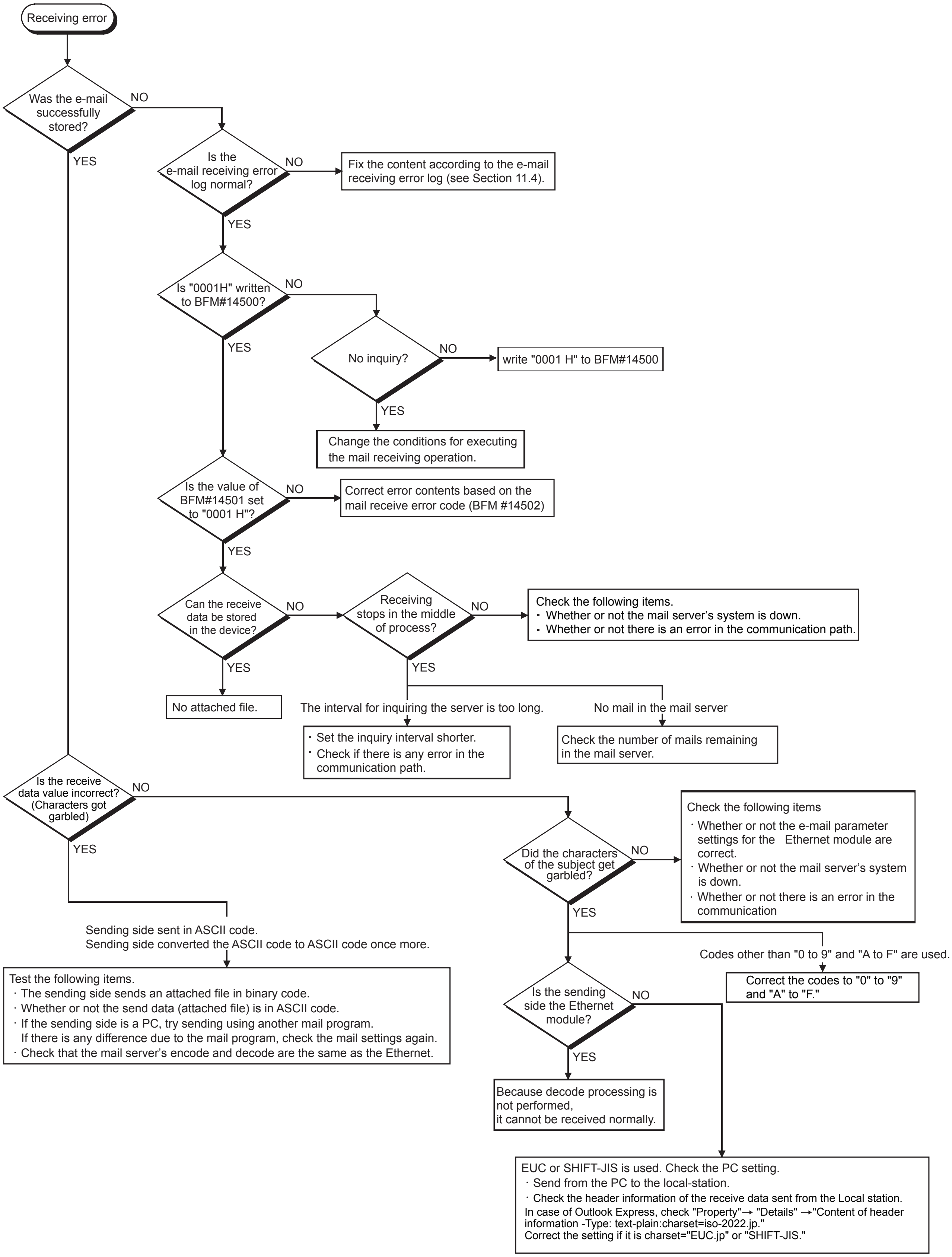
  For UDP communication

  When external device communication is possible, LEDs C1-C8, which are in correspondence with connections 1-8, turn on.
- For connections that communicate using MC protocol, [MC protocol usable] (BFM #32 b9, b10: 10) at the communication parameters, or [Procedure (MC)] at [Open Settings] in the FX Configurator-EN (GX Developer) must be selected.

11.5.4 Sending errors during e-mail communication


Sending error
Is "0001 H" written to BFM#10333?
NO
Correct the condition for writing to BFM#10333.
YES
Is the value of BFM#10334 set to "0001 H"?
NO
Correct the error contents based on e-mail send error log (For details, see Section 11.4).
YES
Can the error response be received from the mail server using the mail receiving operation?
YES
Confirm with the system administrator that the following are set correctly:
・ DNS settings
・ E-mail settings
NO
Check the following items.
・ E-mail parameter settings of the Ethernet module
・ Whether or not the mail server's system is down
・ Whether or not there is an error on the communication path

11.5.5 Receiving errors during e-mail communication


Receiving error
Was the e-mail successfully stored?
NO
YES
Is the e-mail receiving error log normal?
NO
Fix the content according to the e-mail receiving error log (see Section 11.4).
YES
Is "0001H" written to BFM#14500?
NO
YES
No inquiry?
NO
write "0001 H" to BFM#14500
YES
Change the conditions for executing the mail receiving operation.
Is the value of BFM#14501 set to "0001 H"?
NO
Correct error contents based on the mail receive error code (BFM #14502)
YES
Can the receive data be stored in the device?
NO
Receiving stops in the middle of process?
NO
Check the following items.
・ Whether or not the mail server's system is down.
・ Whether or not there is an error in the communication path.
YES
No attached file.
YES
The interval for inquiring the server is too long.
No mail in the mail server
・ Set the inquiry interval shorter.
・ Check if there is any error in the communication path.
Check the number of mails remaining in the mail server.
Is the receive data value incorrect? (Characters got garbled)
NO
YES
Did the characters of the subject get garbled?
NO
Check the following items
· Whether or not the e-mail parameter settings for the Ethernet module are correct.
· Whether or not the mail server's system is down.
· Whether or not there is an error in the communication
YES
Codes other than "0 to 9" and "A to F" are used.
Correct the codes to "0" to "9" and "A" to "F."
Sending side sent in ASCII code.
Sending side converted the ASCII code to ASCII code once more.
Test the following items.
· The sending side sends an attached file in binary code.
· Whether or not the send data (attached file) is in ASCII code.
· If the sending side is a PC, try sending using another mail program.
If there is any difference due to the mail program, check the mail settings again.
· Check that the mail server's encode and decode are the same as the Ethernet.
Is the sending side the Ethernet module?
NO
YES
Because decode processing is not performed, it cannot be received normally.
EUC or SHIFT-JIS is used. Check the PC setting.
· Send from the PC to the local-station.
· Check the header information of the receive data sent from the Local station.
In case of Outlook Express, check "Property"→ "Details" →"Content of header information -Type: text-plain:charset=iso-2022.jp."
Correct the setting if it is charset="EUC.jp" or "SHIFT-JIS."

APPENDIX

Appendix 1 Processing Time

Calculate the minimum processing time for each function using the expressions below. Note that the processing time may become longer depending on the load factor on the network (how congested the line is), the window size of each connected device, the number of connections used concurrently, and how the system is configured. Use the values obtained from the expressions below as a guideline for the processing time when communication is performed using only one connection.

(1) Minimum processing time of communication using the fixed buffer (communication between the Ethernet blocks)

(a) Communication using the fixed buffer (Procedure exist)

$Tfs = St + Ke + (Kdf \times Df) + Sr$

Tfs : Time from the start of sending to the completion of sending (unit: ms)
St : Sending station scan time
Ke, Kdf : Constant (see the table below)
Df : Word count of send data
Sr : Receiving station scan time

	Communication using TCP/IP		Communication using UDP/IP	
	Ke	Kdf	Ke	Kdf
Data communication using binary code	12	0.0065	10	0.0069
Data communication using ASCII code	12	0.030	10	0.029

(b) Communication using the fixed buffer (No procedure)

$Tfs = St + Ke + (Kdf \times Df)$

Tfs : Time from the start of sending to the completion of sending (unit: ms)
St : Sending station scan time
Ke, Kdf : Constant (see the table below)
Df : Byte count of send data

	Communication using TCP/IP		Communication using UDP/IP	
	Ke	Kdf	Ke	Kdf
Data communication using binary code	7	0.0018	4	0.0014

[Calculation example]

Calculate the time from the start of sending to the completion of sending (unit: ms) when the FX3U-ENET communicate using TCP/IP and send 32 words of binary code data using fixed buffer communication (procedure exist).

- Assume that the scan time on the receiving side is 30 ms and the scan time on the transmission side is 25 ms:

  $67\ (\text{ms}) \fallingdotseq 30 + 12 + (0.0065 \times 32) + 25$

(2) Minimum processing time of communication using MC protocol (batch read and batch write)

Tfs = Ke + (Kdt × Df) + Scr × number of scans required for processing + ACK processing time of external device

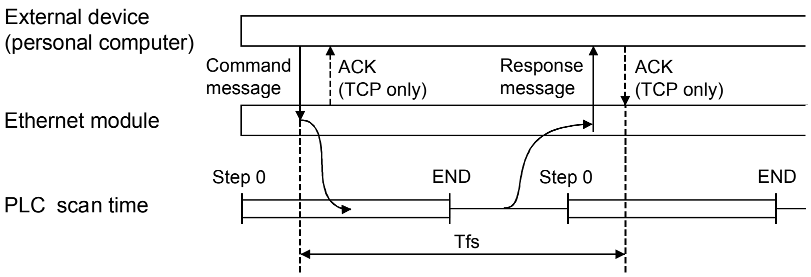
Tfs : Time from when Ethernet block receives request data from personal computer until it completes processing (unit: ms) ＊1

Ke, Kdt : Constant (refer to the table below)

Df : Number of request data words + number of response data words (Application data part)

Scr : Scan time

＊1 The timing sequence from when the Ethernet block receives the request data from the personal computer until it completes processing is shown below.



		Communication using TCP/IP		Communication using UDP/IP	
		Ke	Kdt	Ke	Kdt
Batch read	Data communication using binary code	14	0.009	13	0.008
	Data communication using ASCII code	18	0.015	13	0.017
Batch write	Data communication using binary code	14	0.009	13	0.008
	Data communication using ASCII code	16	0.027	14	0.027

[Calculation example 1]

Calculate the time the FX3U-ENET takes to complete the processing of a data request from a PC after receiving it, when the FX3U-ENETand the PC perform TCP/IP communication and read 32 points of ASCII code data from the data register (D) of the local station (unit: ms) using MC protocol communication.

- Assume that the scan time of the FX3U-ENET installed station is 40 ms:

  59.17 + ACK processing time of the external device (ms) $\fallingdotseq$ 18 + (0.015 × (12 + 66)) + 40 × 1 + ACK processing time of the external device

  Command data length = 12 words
  Response data length = 66 words

[Calculation example 2]

Calculate the time the FX3U-ENETtakes to complete the processing of a data request from a PC after receiving it, when the FX3U-ENETand the PC performs TCP/IP communication and writes 32 points of ASCII code data to the data register (D) of the local station (unit: ms) using MC protocol communication.

- Assume that the scan time of the FX3U-ENET installed station is 40 ms:

  58.11 (ms) $\fallingdotseq$ 16 + (0.027 × (76 + 2)) + 40 × 1

  Command data length = 76 words
  Response data length = 2 words

Appendix 2 ASCII Code List

MSD		0	1	2	3	4	5	6	7
LSD		000	001	010	011	100	101	110	111
0	0000	NUL	DLE	(SP)	0	@	P	`	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	“	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	‘	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
A	1010	LF	SUB	*	:	J	Z	j	z
B	1011	VT	ESC	+	;	K	[	k	{
C	1100	FF	FS	,	<	L	\	l	|
D	1101	CR	GS	–	=	M	]	m	}
E	1110	SO	RS	.	>	N	^	n	~
F	1111	SI	US	/	?	O	_	o	DEL

Appendix 3 References

For details on TCP/IP, refer to the DDN Protocol Handbook (3 volumes).

Publisher

DDN Network Information Center
SRI International
333 Ravenswood Avenue, EJ291
Menlo Park, California 94025

RFC Number

TCP	RFC793
UDP	RFC768
IP	RFC791
ICMP	RFC792
ARP	RFC826

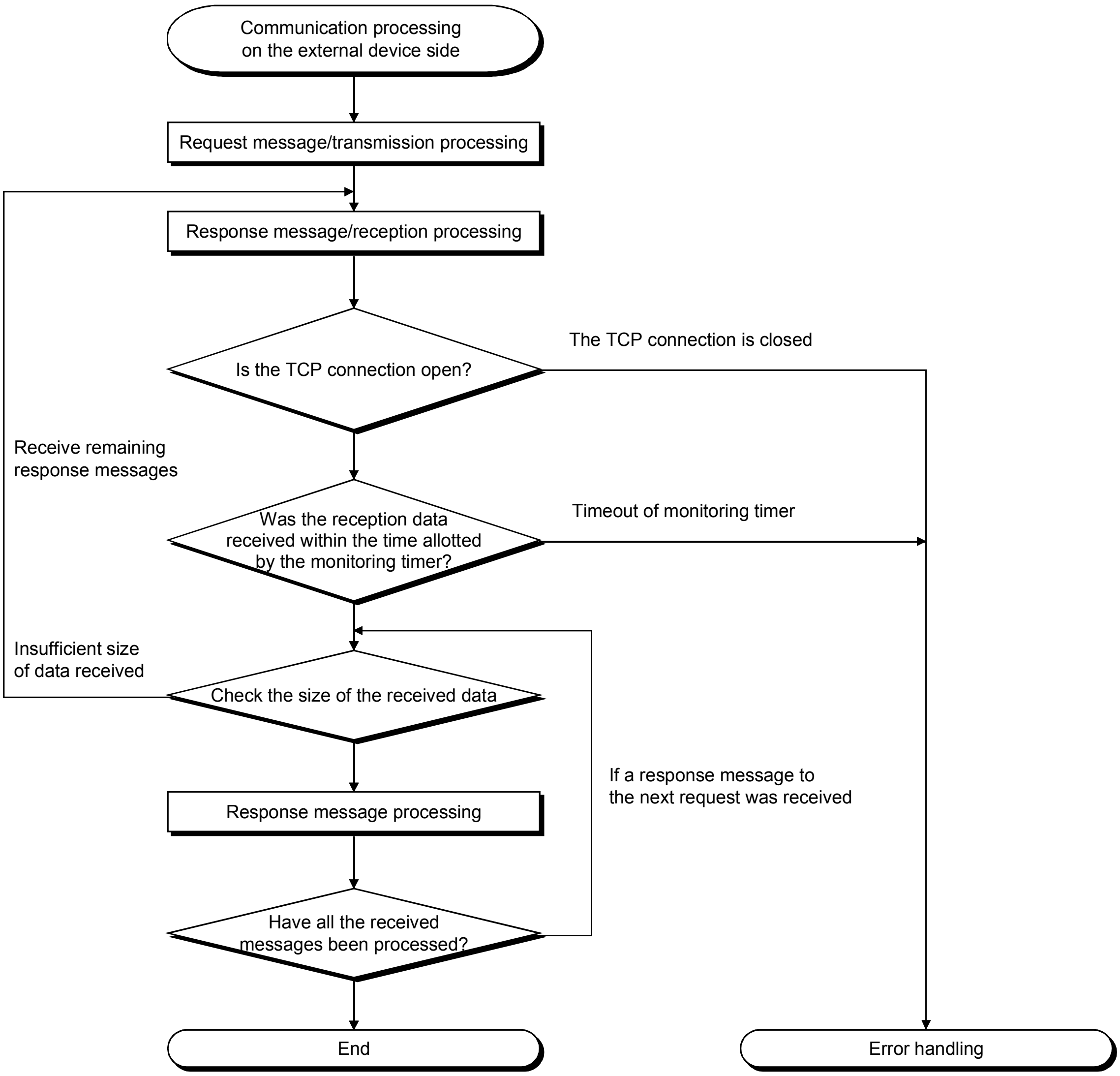
Appendix 4 Program Examples

The program examples presented in this section test the connection between the Ethernet block and an external device (IBM PC/AT) that is connected to the same Ethernet.

For each program, only the minimum programming that is required to perform the communication test is done. Modify the IP address, port number and other values according to your system configuration. In addition, abnormal handling may be added.

(1) Receive processing of target device

A receive processing example on the target device side is shown below.


Communication processing on the external device side
Request message/transmission processing
Response message/reception processing
Is the TCP connection open?
The TCP connection is closed
Receive remaining response messages
Was the reception data received within the time allotted by the monitoring timer?
Timeout of monitoring timer
Insufficient size of data received
Check the size of the received data
If a response message to the next request was received
Response message processing
Have all the received messages been processed?
End
Error handling

Background

For Ethernet communications, the TCP socket functions are used inside the personal computer. However, these functions do not have any limits. Therefore, when the "send" function is executed once to transmit data, the receiving end (node) needs to execute the "recv" function once or more in order to read the data ("send" and "recv" is not proportional to 1:1 execution). For this reason, the receiving procedure explained above, is required.

Appendix 4.1 Program example for communication using MC protocol –1

The following explains a program, its execution environment and the contents of data communication .

(1) Execution environment of the program example

(a) PLC side

1) Base module of the Ethernet installed station : FX3U/FX3UC
2) Ethernet block No. : No.1
3) Ethernet block IP address : 172.16.56.99 (AC.10.38.63H)
4) Ethernet block port number : 10000
5) FX Configurator-EN (GX Developer) setting
   - Operational settings : See "(3) FX Configurator-EN (GX Developer) setting (a)" on the next page
   - Open settings : See "(3) FX Configurator-EN (GX Developer) setting (b)" on the next page

(b) External device side

1) Operation environment : Microsoft® Windows® 2000 Operating system
2) Ethernet interface board model name : WINSOCK compatible board
3) Library : WSOCK32.LIB
4) Software development environment : Microsoft® Corporation Visual C++ .NET
5) Ethernet address : Setting not required because the ARP function is available
6) IP address : Receive at Active open
7) Port number : Receive at Active open

(c) Communication protocol : TCP/IP

(2) Outline of the program example

(a) Sequence program on the PLC side
Parameters are set from FX Configurator-EN (GX Developer).
(Sequence program is not required)

(b) Program on the external device side
Executes the following read/write data communication with the PLC using the library mentioned above.
- Write in word units (for 5 points from D0 to D4)
- Read in word units (for 5 points from D0 to D4)

(3) FX Configurator-EN (GX Developer) settings

Set the PLC parameters as follows.

(a) Operation settings

Communication data code	: ASC II (BFM#24 b1: ON)
Initial timing	: Always wait for OPEN (Communication possible at STOP time) (BFM#24 b8: ON)
IP address	: 172.16.56.99 (AC.10.38.63H) (BFM#106 to 107)

(b) Open settings

Protocol	: TCP (BFM#32 b8: OFF)
Open system	: Unpassive (BFM#32 b14, b15: 10H)
Fixed buffer	: Send (BFM#32 b0: OFF)
Fixed buffer communication procedure	: Procedure exist (MC) (BFM#32 b9: OFF, b10: ON)
Pairing open	: Disabie (BFM#32 b7: OFF)
Existence confirmation	: No confirm (BFM#32 b1: OFF)
Host station Port No.	: 10000 (BFM#40)

(4) Program on the external device side

The program example of the external device shown below accesses the FX3U of the station in which the Ethernet module is installed.
When this program is executed, the contents of the following communication messages are displayed in sequence:

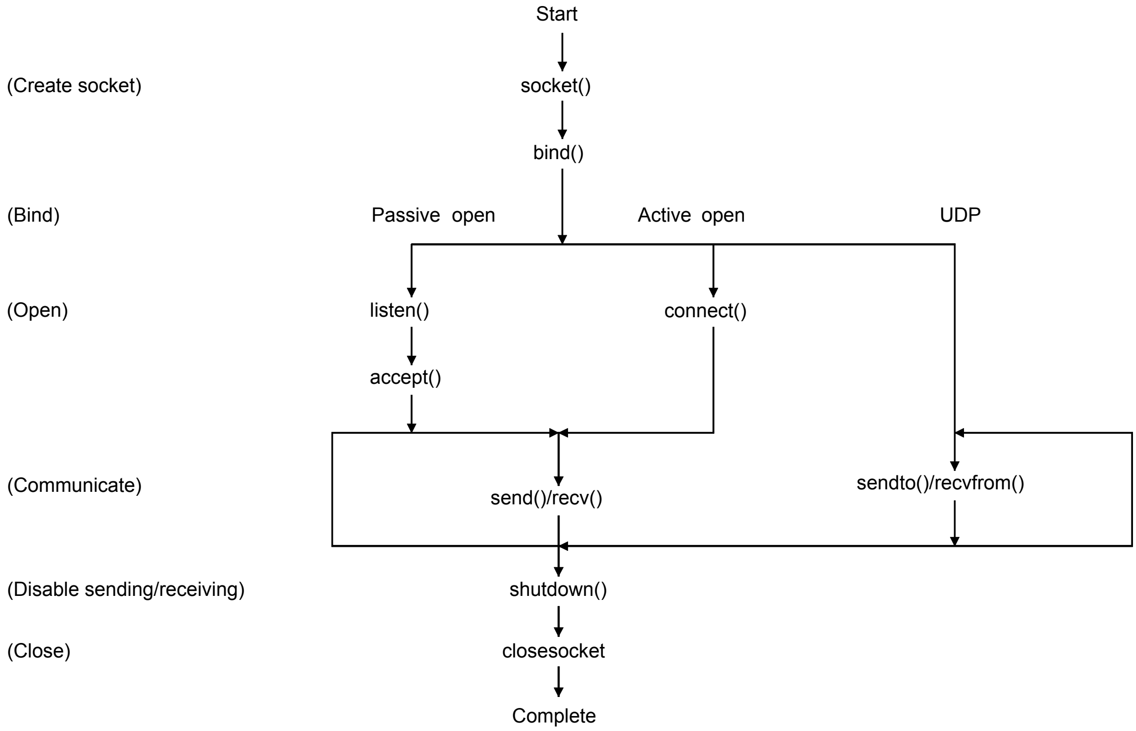
1) Batch write command message in word units
2) Batch write response message in word units
3) Batch read command message in word units
4) Batch read response message in word units

REMARK

(1) The following explains an outline of the compiling procedure for a program created using Microsoft® Corporation Visual C++ .NET

1) Start Visual C++.
2) Prepare for program creation.
   Select "New Project" from the project tab, select "Win32 console project", and create an empty project with the name "ENETSAMP".
3) Add ENETSAMP.C to the project and create a program.
   (See the program example on the next page.)
4) Compile the created file from the compile screen of the build menu.
5) From the project menu property setting screen, add WSOCK32.LIB to the linker addition dependency files.
6) Create an executable file (ENETSAMP.EXE) on the build screen of the build menu.
7) End Visual C++.NET.
8) Execute ENETSAMP.EXE.

(2) Outline of the procedure for calling the socket routine



```
/ ＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊ /
/ ＊＊                                                  ＊＊ /
/ ＊＊   Sample program                                 ＊＊ /
/ ＊＊                                                  ＊＊ /
/ ＊＊   This program is a sample program to conduct a ＊＊ /
/ ＊＊   connection test between the Ethernet block and ＊＊ /
/ ＊＊   target device.                                 ＊＊ /
/ ＊＊   This program accesses the data register (D) of ＊＊ /
/ ＊＊   the Base Module installed together with the    ＊＊ /
/ ＊＊   Ethernet block.                                ＊＊ /
/ ＊＊                                                  ＊＊ /
/ ＊＊   Copyright(C) 2005 Mitsubishi Electric          ＊＊ /
/ ＊＊   Corporation                                    ＊＊ /
/ ＊＊   All Rights Reserved                            ＊＊ /
/ ＊＊                                                  ＊＊ /
/ ＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊ /


#include <stdio.h>
#include <winsock.h>
#define FLAG_OFF 0                      // Completion flag OFF
#define FLAG_ON 1                       // Completion flag ON
#define SOCK_OK 0                       // Normal completion
#define SOCK_NG -1                     // Abnormal completion
#define BUF_SIZE 4096                  // Receive buffer size
#define ERROR_INITIAL 0                 // Initial error
#define ERROR_SOCKET 1                  // Socket creation error
#define ERROR_BIND 2                    // Bind error
#define ERROR_CONNECT 3                 // Connection error
#define ERROR_SEND 4                    // Send error
#define ERROR_RECEIVE 5                 // Receive error
#define ERROR_SHUTDOWN 6                // Shutdown error
#define ERROR_CLOSE 7                   // Line close error
//Definitions for checking the receiving sizes
//#define   RECV_ANS_1 4        // Receiving size of response message in reply to device write (1E frame)
//#define   RECV_ANS_2 24       // Receiving size of response message in reply to device read (1E frame)

typedef struct sck_inf{
        struct in_addr my_addr;
        unsigned short my_port;
        struct in_addr FX_IP_addr;
        unsigned short FX_port;
};

int nErrorStatus;                               // Error information storage variable
int Dmykeyin;                                   // Dummy key input
int Closeflag;                                  // Connection completion flag
SOCKET socketno;
```

```
int main()
{
        WORD wVersionRequested=MAKEWORD(1,1);           // Winsock Ver 1.1 request
        WSADATA wsaData;
        int length;                                     // Communication data length
        unsigned char s_buf[BUF_SIZE];                  // Send buffer
        unsigned char r_buf[BUF_SIZE];                  // Receive buffer
        int rbuf_idx;                                   // Receive data storage head index
        int recv_size;                                  // Number of receive data
        struct sck_inf sc;
        struct sockaddr_in hostdata;                    // External device side data
        struct sockaddr_in FX3UENET;                    // Ethernet block side data
        void Sockerror(int);                            // Error handling function
        unsigned long ulCmdArg ;                        // Non-blocking mode setting flag

        sc.my_addr.s_addr=htonl(INADDR_ANY);            // External device side IP address
        sc.my_port=htons(0);                            // External device side port number
        sc.FX_IP_addr.s_addr=inet_addr("172.16.56.99"); // Ethernet block side IP address
(AC103863h)
        sc.FX_port=htons(10000);                        // Ethernet block side port number
        Closeflag=FLAG_OFF;                             // Connection completion flag off
        nErrorStatus=WSAStartup(wVersionRequested,&wsaData); // Winsock Initial processing
        if (nErrorStatus!=SOCK_OK) {
                Sockerror(ERROR_INITIAL);               // Error handling
                return (SOCK_NG);
        }
        printf ("Winsock Version is %ld.%ld\n",HIBYTE(wsaData.wVersion),LOBYTE(wsaData.wVersion));
        printf ("FX3U-ENET Test Start\n");
        socketno=socket(AF_INET,SOCK_STREAM,0);         // Create socket for TCP/IP
        if (socketno==INVALID_SOCKET){
                Sockerror (ERROR_SOCKET);               // Error handling
                return(SOCK_NG);
        }
        hostdata.sin_family=AF_INET;
        hostdata.sin_port=sc.my_port;
        hostdata.sin_addr.s_addr=sc.my_addr.s_addr;

        if(bind(socketno,(LPSOCKADDR)&hostdata,sizeof(hostdata))!=SOCK_OK){
                                                        // Bind
                Sockerror(ERROR_BIND);                  // Error handling
                return(SOCK_NG);
        }
        FX3UENET.sin_family=AF_INET;
        FX3UENET.sin_port=sc.FX_port;
        FX3UENET.sin_addr.s_addr=sc.FX_IP_addr.s_addr;
        if(connect(socketno,(LPSOCKADDR)&FX3UENET,sizeof(FX3UENET))!=SOCK_OK){
                                                                  // Connection (Active open)
                Sockerror(ERROR_CONNECT);                         // Error handling
                return(SOCK_NG);
        }
```

```
Closeflag=FLAG_ON;                                          // Connection completion flag ON
// Go to non-blocking mode
ulCmdArg = 1;
ioctlsocket(socketno, FIONBIO, &ulCmdArg);                  // Set to non-blocking mode
strcpy(s_buf, "03FF000A4420000000000500112233445566778899AA");
                                                        // D0 to D4 batch write request (1E frame)
length=(int)strlen(s_buf);
if(send(socketno,s_buf,length,0)==SOCKET_ERROR){            // Data sending
            Sockerror(ERROR_SEND);                          // Error handling
            return (SOCK_NG);
}
printf("\n send data\n%s\n",s_buf);
// Perform receiving size check and receiving processing simultaneously
rbuf_idx = 0;                           // Receive data storage head index initialization
recv_size = 0;                               // Initialize the number of receive data
while(1) {
            length = recv(socketno, &r_buf[rbuf_idx], (BUF_SIZE - rbuf_idx), 0);
                                                            // Response data receiving
            if(length == 0) {                               // Is connection cut off?
               Sockerror(ERROR_RECIEVE);                    // Error handling
               return (SOCK_NG);
            }

            if(length == SOCKET_ERROR) {
               nErrorStatus = WSAGetLastError();
               if(nErrorStatus != WSAEWOULDBLOCK) {
                     Sockerror(ERROR_RECIEVE);              // Error handling
                     return (SOCK_NG);
               } else {
                     continue;                              // Repeat until messages are received
               }
            } else {
               rbuf_idx += length;                          // Update the receive data storage
                                                            // position
               recv_size += length;                         // Update the number of receive data
               if(recv_size >= RECV_ANS_1)                  // Have all response messages been
                                                            // received?
                     break;                                 // Stop repeating as messages have
                                                            // been received
            }
}
r_buf[rbuf_idx] = '\0' ;                                    // Set NULL at the end of receive data
printf("\n receive data\n%s\n",r_buf);
strcpy(s_buf, "01FF000A4420000000000500");                  // D0 to D4 batch read request
                                                            // (1E frame)
length=(int)strlen(s_buf);
if(send(socketno,s_buf,length,0)==SOCKET_ERROR){            // Data sending
            Sockerror(ERROR_SEND);                          // Error handling
            return (SOCK_NG);
}
printf("\n send data\n%s\n",s_buf);
// Perform receiving size check and receiving processing simultaneously
rbuf_idx = 0;                                               // Receive data storage head index
                                                            // initialization
recv_size = 0;                                              // Initialize the number of receive data
```

```
        while(1) {
                length = recv(socketno, &r_buf[rbuf_idx], (BUF_SIZE - rbuf_idx), 0);
                                                            // Response data receiving
                if(length == 0) {                           // Is connection cut off?
                   Sockerror(ERROR_RECIEVE);                // Error handling
                   return (SOCK_NG);
                }

        if(length == SOCKET_ERROR) {
                nErrorStatus = WSAGetLastError();
                if(nErrorStatus != WSAEWOULDBLOCK) {
                   Sockerror(ERROR_RECIEVE);                // Error handling
                   return (SOCK_NG);
                } else {
                   continue;                                // Repeat until messages are received
                }
        } else {
                rbuf_idx += length;                         // Update the receive data storage
                                                            // position
                recv_size += length;                        // Update the number of receive data
                if(recv_size >= RECV_ANS_2)                 // Have all response messages been
                                                            // received?
                   break;                                   // Stop repeating as messages have
                                                            // been received
               }
}
r_buf[rbuf_idx] = '\0' ;                                    // Set NULL at the end of receive data
printf("\receive data\n%s\n", r_buf);
        if(shutdown(socketno,2)!=SOCK_OK){                  // Processing to disable
                                                            // sending/receiving
                Sockerror(ERROR_SHUTDOWN);                  // Error handling
                return(SOCK_NG);
        }
        if(closesocket(socketno)!=SOCK_OK){                 // Close processing
                Sockerror(ERROR_CLOSE);                     // Error handling
                return(SOCK_NG);
        }
        Closeflag=FLAG_OFF;                                 // Connection completion flag off
        WSACleanup();                                       // Release Winsock.DLL
        printf("\nFX3U-ENET Test End.\n\n Normally completed. \n");
        printf("Press any key to exit the program.\n");
        Dmykeyin=getchar();                                 // Wait for key input
        return(SOCK_OK);
}

void Sockerror(int error_kind)                              // Error handling function
{
        if(error_kind==ERROR_INITIAL){
                printf("Initial processing is abnormal.");
        }
        else{
                nErrorStatus=WSAGetLastError();
                switch(error_kind){
                case ERROR_SOCKET:
                        printf("Failed to create socket.");
                        break;
                case ERROR_BIND:
                        printf("Failed to bind.");
                        break;
                case ERROR_CONNECT:
                        printf("Failed to establish connection.");
                        break;
                case ERROR_SEND:
                        printf("Sending failed.");
                        break;
                case ERROR_RECIEVE:
                        printf("Receiving failed.");
                        break;
                case ERROR_SHUTDOWN:
                        printf("Failed to shutdown.");
                        break;
                case ERROR_CLOSE:
                        printf("Failed to close normally.");
                        break;
                }
        }
        printf("Error code is %d.\n", nErrorStatus);
        if(Closeflag==FLAG_ON){
                nErrorStatus=shutdown(socketno,2);              // Shutdown processing
                nErrorStatus=closesocket(socketno);             // Close processing
                Closeflag=FLAG_OFF;                             // Connection completion flag off
        }
        printf("Press any key to exit the program.\n");
        Dmykeyin=getchar();                                     // Wait for a key input
        WSACleanup();                                           // Release Winsock.DLL
        return;
}
```

Appendix 4.2 Program example of communication using MC protocol –2

This section explains an example of an external device program that reads/writs data from the PLC.
A sample program, its execution environment and contents of data communication are shown below.

(1) Execution environment of the program example

1) The settings of the PLC side are the same as those of the execution environment described in Section 4.1 (1) (a) and (3) of Appendix.

2) The settings of the external device side are the same as those of the execution environment described in Section 4.1 (1) (b) of Appendix, except for the following including the software development:
- Software development environment: Microsoft® Corporation Visual Basic .NET
- Arbitrary numbers are assigned for the IP address and port number.

3) The communication protocol is TCP/IP

(2) Outline of the program example

With the A-compatible 1E frame command (01: batch read in word units), this program reads data from D0 to D4 (five points) of the FX3U-CPU of the station on which the Ethernet block is mounted.

(3) Outline of the sample program

(a) Create a new project and form.

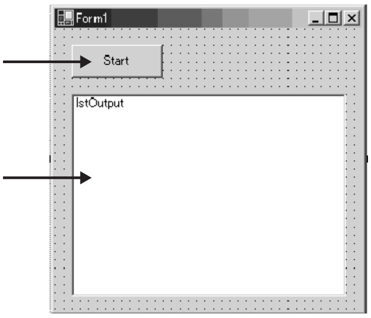
(b) Create the (example) window shown in (4) below using " Button" and "List Box" in the toolbox.

(c) Create the program shown in (5).

(4) Window example (Form 1)

(Object name)
Start
Connect to the Ethernet module and perform communications

1st Output
Display data that was read from the Ethernet module



(5) Sample program (Form 1)

Parts in italic are created automatically by Visual Basic .NET, so no input is required.
Only input where written in bold.

```
Option Explicit On
Option Strict On

Imports System.Net.Sockets

Public Class Form1
    Inherits System.Windows.Forms.Form

#Region " Windows Code created by Form Designer "

    Public Sub New()
        MyBase.New()

        ' This call is necessary for Windows Form Designer.
        InitializeComponent()

        ' InitializeComponent() Initialization is added after the call.

    End Sub

    ' The Form overwrites dispose to execute after-processing in the component list.
    Protected Overloads Overrides Sub Dispose(ByVal disposing As Boolean)
        If disposing Then
            If Not (components Is Nothing) Then
                components.Dispose()
            End If
        End If
        MyBase.Dispose(disposing)
    End Sub

    ' Necessary for Windows Form Designer.
    Private components As System.ComponentModel.IContainer
    Memo: The following procedure is necessary for Windows Form Designer.
    Change by using Windows Form Designer.
    Do not use code editor to change.
    Friend WithEvents Start As System.Windows.Forms.Button
    Friend WithEvents lstOutput As System.Windows.Forms.ListBox
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()
        Me.Start = New System.Windows.Forms.Button
        Me.lstOutput = New System.Windows.Forms.ListBox
        Me.SuspendLayout()
        '
        'Start
        '
        Me.Start.Location = New System.Drawing.Point(16, 16)
        Me.Start.Name = "Start"
        Me.Start.Size = New System.Drawing.Size(88, 32)
        Me.Start.TabIndex = 0
        Me.Start.Text = "Start"
        '
        'lstOutput
        '
        Me.lstOutput.ItemHeight = 12
        Me.lstOutput.Location = New System.Drawing.Point(16, 64)
        Me.lstOutput.Name = "lstOutput"
        Me.lstOutput.Size = New System.Drawing.Size(264, 196)
        Me.lstOutput.TabIndex = 1
        '
        'Form1
        '
        Me.AutoScaleBaseSize = New System.Drawing.Size(5, 12)
        Me.ClientSize = New System.Drawing.Size(296, 273)
        Me.Controls.Add(Me.lstOutput)
        Me.Controls.Add(Me.Start)
        Me.Name = "Form1"
        Me.Text = "Form1"
        Me.ResumeLayout(False)

    End Sub

#End Region

    Private Sub Start_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
    Start.Click
        Dim IpAddress As String
        Dim PortNum As Integer
        Dim Client As TcpClient
        Dim Buffer() As Byte
        Dim InBuff(1532) As Byte
        Dim TxCommand As String
        Dim RxResponse As String
        Dim Temp As String
        Dim j As Integer
        Dim Dreg(5) As Double
        Dim DregStr$
        Dim SubHeader$

        ' IP Address specification
        IpAddress = "172.16.56.99"
        ' Port Number specification
        PortNum = 10000

        Client = New TcpClient
        'Line connection processing
        Try
            Client.Connect(IpAddress, PortNum)
        Catch ex As Exception
            MsgBox("Connection with the server failed, and the following code was returned:
            "& ex.Message, 0, "connection error")
            Exit Sub
        End Try

        'Read D0 to D4 (5 points) with the A-compatible 1E frame command.
        TxCommand = "01ff000a4420000000000500"
        Buffer = System.Text.Encoding.Default.GetBytes(TxCommand.ToCharArray)
        'Sending a read command
        Client.GetStream().Write(Buffer, 0, Buffer.Length)
        'Waiting for a response from an Ethernet block
        While Not Client.GetStream().DataAvailable()
            Application.DoEvents()
        End While
        If Client.GetStream().DataAvailable() Then
            Client.GetStream().Read(InBuff, 0, InBuff.Length)
            RxResponse = System.Text.Encoding.Default.GetString(InBuff)
            SubHeader = Mid$(RxResponse, 3, 2)
            If SubHeader = "00" Then  'Normal response
                Temp = ""          'Initialization of an output character string
                For j = 0 To 4
                    DregStr$ = Mid(RxResponse, j * 4 + 5, 4)
                    Dreg(j) = Val("&H" + DregStr$)
                    Temp = Temp + Format(Dreg(j), "#####0") + "  "
                Next
                lstOutput.Items.Insert(lstOutput.Items.Count, Temp)
            ElseIf SubHeader = "5B" Then          ' In an abnormal response, an abnormal code is added.
                Temp = "Terminate Code = " & SubHeader & " Error Code = " & Mid$(RxResponse, 5, 2)
                lstOutput.Items.Insert(lstOutput.Items.Count, Temp)
            Else
                Temp = "Terminate Code = " & SubHeader
                lstOutput.Items.Insert(lstOutput.Items.Count, Temp)
```

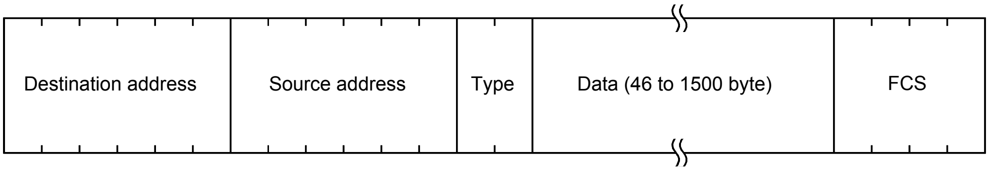
```
                End If
                lstOutput.SelectedIndex = lstOutput.Items.Count - 1
            End If
            ' Line disconnection processing
            Client.GetStream().Close()
            Client.Close()

        End Sub
    End Class
```

Appendix 5 Differences between the Ethernet and the IEEE802.3

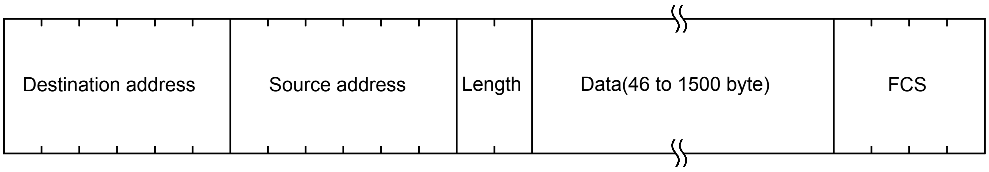
The following shows the Ethernet header in the data link layer supported by the Ethernet block.

Ethernet data link layer header	Ethernet block
Ethernet frame (V 2.0) specification	○
IEEE802.3 (ISO/IEC8802.3) frame specification	○

(1) Ethernet



(2) IEEE802.3



Appendix 6 ICMP Protocol Supported by the Ethernet Module

The following table outlines the types of ICMP supported by the Ethernet block and the processing performed by the Ethernet block.

ICMP Type	ICMP name/description	Processing by the Ethernet block
0	Echo Reply Result of IP packet loopback	Upon receiving an Echo Request, the Ethernet block sends this message.
8	Echo Request Requests loopback of IP packet	If destination existence confirmation is set in the buffer memory, the Ethernet module sends this message when attempting to confirm the existence of the target. (∗[1])
Others	—	Ignored by the Ethernet module. (Not supported)

∗1 The Ethernet module can simultaneously receive two ICMP ECHO requests (type 8, Ping message), which are used for existence confirmation, etc., and handles them accordingly.
When three or more ICMP ECHO requests are received at the same time, the third and succeeding requests will be ignored.
If a response is not returned to the external device when an ICMP ECHO request is sent to the Ethernet module, send an ICMP ECHO request to the Ethernet module again.
The Ethernet module is able to receive a maximum of 1460-byte ICMP message at one time.
Do not send an ICMP message request exceeding 1460 bytes to the Ethernet module.

Appendix 7 Setting Value Recording Sheets

This section provides setting value recording sheets for parameters set with FX Configurator-EN (GX Developer) Make copies as needed.

Setting value recording sheet No.	FX configurator-EN (GX Developer) setting screen
Recording sheet 1	Ethernet operations
Recording sheet 2	Ethernet initial settings
Recording sheet 3	Ethernet open settings
Recording sheet 4	Setting the Ethernet relay parameter
Recording sheet 5	Ethernet E-mail settings
	Ethernet mail address settings

Recording sheet 1

[Block number]

FX configurator-EN (GX Developer) setting screen	Data item		Setting data		
			Setting value		Remark
Ethernet Operations	Communication data code setting			Binary code communication	
				ASCII code communication	
	Initial Timing			Do not wait for OPEN (Communications impossible at STOP time)	
				Always wait for OPEN (Communication possible at STOP time)	
	IP address setting	Input format		Decimal	IP address Input format
				Hexadecimal	
		IP address	. . .		Adjust to input format
	Send frame setting			Ethernet (V2.0)	
				IEEE802.3	
	TCP Existence Confirmation setting			Use the KeepAlive	
				Use the Ping	

Recording sheet 2

[Block number]

FX configurator-EN (GX Developer) setting screen	Data item		Setting data	
			Setting value	Remark
Ethernet Initial settings	Timer setting	TCP ULP Timer		Default: 60 (500 ms)
		TCP zero window Timer		Default: 20 (500 ms)
		TCP resend timer		Default: 20 (500 ms)
		TCP end timer		Default: 40 (500 ms)
		IP assembly timer		Default: 10 (500 ms)
		Response monitoring timer		Default: 60 (500 ms)
		Destination existence confirmation starting interval		Default: 1200 (500 ms)
		Destination existence confirmation Interval timer		Default: 20 (500 ms)
		Destination existence confirmation resend timer		Default: 3 (times)
	DNS setting	Input format	□ Decimal □ Hexadecimal	DNS server IP address Input format
		DNS server 1 IP address	. . .	Adjust to input format
		DNS server 2 IP address	. . .	
		DNS server 3 IP address	. . .	
		DNS server 4 IP address	. . .	

Recording Sheet 3

[Block number]

FX configurator-EN (GX Developer) setting screen	Data item		Setting data		
			Setting value		Remark
Ethernet open settings	Connection No.	Protocol		TCP	
				UDP	
		Open system		Active	Setting not required if UDP is selected.
				Unpassive	
				Fullpassive	
		Fixed buffer		Send	
				Receive	
		Fixed buffer communication		Procedure exist	
				No procedure	
		Paring open		No pairs	
				Pairs	
		Existence confirmation		No confirm	
				Confirm	
		Local station Port No.			Input format: Hexadecimal
		Destination IP address		Decimal	Destination IP address input format
				Hexadecimal	
			. . .		Adjust to input format
		Destination Port No.			Input format: Hexadecimal
	Connection No.	Protocol		TCP	
				UDP	
		Open system		Active	
				Unpassive	
				Fullpassive	
		Fixed buffer		Send	
				Receive	
		Fixed buffer communication		Procedure exist	
				No procedure	
		Paring open		No pairs	
				Pairs	
		Existence confirmation		No confirm	
				Confirm	
		Local station Port No.			Input format: Hexadecimal
		Destination IP address		Decimal	Destination IP address input format
				Hexadecimal	
			. . .		Adjust to input format
		Destination Port No.			Input format: Hexadecimal

Recording sheet 4

[Block number]

FX configurator-EN (GX Developer) setting screen	Data item	Setting data	
		Setting value	Remark
Setting the Ethernet relay parameter	Sub-net mask pattern	. . .	Adjust to input format
	Default router IP address	. . .	Adjust to input format
	Input format	□ Decimal	Router information Input format
		□ Hexadecimal	

Recording sheet 5

[Block number]

FX configurator-EN (GX Developer) setting screen	Data item			Setting data		
				Setting value		Remark
Ethernet e-mail settings	General settings	Password				
		Mail address				
		Check of received mail		□	Check received mails	Check: Check mark
		Interval of inquiry				Set interval value to check received mail.
				□	s	Select unit for interval to check received mail.
				□	min	
				□	h	
	Mail Server name	Send setting	SMTP Server name			
			IP address	□	Decimal	IP address input format
				□	Hexadecimal	
					. . .	Adjust to input format
			POP Server name			
			IP address	□	Decimal	IP address input format
				□	Hexadecimal	
					. . .	Adjust to input format
Ethernet send mail address settings	No.1	Send mail address				
	No.2	Send mail address				
	No.3	Send mail address				
	No.4	Send mail address				
	No.5	Send mail address				
	No.6	Send mail address				
	No.7	Send mail address				
	No.8	Send mail address				
	No.9	Send mail address				
	No.10	Send mail address				

MEMO

Revised History

Date	Revision	Description
10/2005	A	First Edition
9/2007	B	- The FX3UC Series was added - Clerical error correction
9/2008	C	Clerical error correction

USER'S MANUAL

FX3U-ENET

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE: TOKYO BULDING, 2-7-3 MARUNOUCHI, CHYODA-KU, TOKYO 100-8310, JAPAN
HIMEJI WORKS: 840, CHIYODA CHO, HIMEJI, JAP*AN

MODEL	FX3U-ENET-U-E
MODEL CODE	09R716

JY997D18101C
(MEE)

Effective Sep. 2008
Specifications are subject to change without notice.