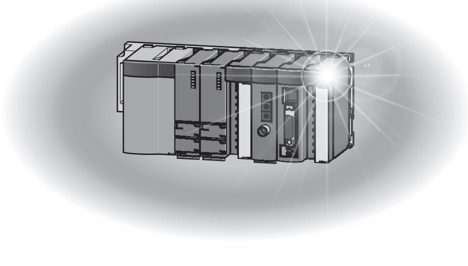
MITSUBISHI ELECTRIC

Programmable Controller

MELSEC Q series

Q Corresponding Ethernet Interface Module User's Manual (Basic)

-QJ71E71-100
-QJ71E71-B5
-QJ71E71-B2



●SAFETY PRECAUTIONS●

(Read these precautions before using this product.)

Before using this product, please read this manual and the relevant manuals carefully and pay full attention to safety to handle the product correctly.

In this manual, the safety precautions are classified into two levels: "⚠WARNING" and "⚠CAUTION".

⚠WARNING	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 minor or moderate injury or property damage.

Under some circumstances, failure to observe the precautions given under "⚠CAUTION" may lead to serious consequences.
Observe the precautions of both levels because they are important for personal and system safety.
Make sure that the end users read this manual and then keep the manual in a safe place for future reference.

[Design Precautions]

⚠WARNING
● For the operating status of each station after a communication failure, refer to manuals relevant to the network. Incorrect output or malfunction due to a communication failure may result in an accident. ● To prevent the malfunction of the programmable controller system due to harmful e-mails, take preventive measures (such as antivirus measures) so that the mail server for this module does not receive harmful e-mails. ● To maintain the safety of the programmable controller system against unauthorized access from external devices via the network, take appropriate measures. To maintain the safety against unauthorized access via the Internet, take measures such as installing a firewall. ● When connecting a peripheral with the CPU module or connecting an external device, such as a personal computer, with an intelligent function module to modify data of a running programmable controller, configure an interlock circuit in the program to ensure that the entire system will always operate safely. For other forms of control (such as program modification or operating status change) of a running programmable controller, read the relevant manuals carefully and ensure that the operation is safe before proceeding. Especially, when a remote programmable controller is controlled by an external device, immediate action cannot be taken if a problem occurs in the programmable controller due to a communication failure. To prevent this, configure an interlock circuit in the program, and determine corrective actions to be taken between the external device and CPU module in case of a communication failure. ● Do not write any data in the system area of the buffer memory in the intelligent function module. Also, do not use any use prohibited signals as an output signal from the programmable controller CPU to the intelligent function module. Doing so may cause malfunction of the programmable controller system.

[Design Precautions]

⚠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 away from each other. Failure to do so may generate noise that may cause malfunctions. ● When changing the operating status of the CPU module (such as remote RUN/STOP) from the external device, select "Always wait for OPEN (Communication possible at STOP time)" for the "Initial timing" setting in the network parameter. The communication line will be closed when "Do not wait for OPEN (Communications impossible at STOP time)" is selected and the remote STOP is executed from the external device. Consequently, the CPU module cannot reopen the communication line, and the external device cannot execute the remote RUN.

[Installation Precautions]

⚠CAUTION

- Use the programmable controller in an environment that meets the general specifications in the user's manual for the CPU module used. Using the programmable controller in any other operating environments may cause electric shocks, fires or malfunctions, or may damage or degrade the module.
- While pressing the installation lever located at the bottom of module, insert the module fixing tab into the fixing hole in the base unit until it stops. Then, securely mount the module with the fixing hole as a supporting point. If the module is not installed properly, it may cause the module to malfunction, fail or fall off. Secure the module with screws especially when it is used in an environment where constant vibrations may occur.
- Be sure to tighten the screws using the specified torque. If the screws loose, it may cause the module to short-circuit, malfunction or fall off. If the screws are tightened excessively, it may damage the screws and cause the module to short-circuit, malfunction or fall off.
- Shut off the external power supply (all phases) used in the system before mounting or removing the module. Failure to do so may cause product damage.
- Do not directly touch any conductive part or electronic component of the module. This may cause the module to malfunction or fail.

[Wiring Instructions]

⚠CAUTION

- Connectors for external devices and coaxial cables must be crimped or pressed with the tool specified by the manufacturer, or must be correctly soldered. Incomplete connections may cause short circuit, fire, or malfunction.
- Shut off the external power supply (all phases) used in the system before connecting the AUI cable.
- Securely connect the connector to the module. Poor contact may cause malfunction.
- Place the cables in a duct or clamp them. If not, dangling cable may swing or inadvertently be pulled, resulting in damage to the module or cables or malfunction due to poor contact.
- 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. If the terminal screws are tightened excessively, it may damage the screws and cause the module to short-circuit, malfunction or fall off.
- When disconnecting the cable from the module, do not pull the cable by the cable part. For the cable with connector, hold the connector part of the cable. For the cable connected to the terminal block, loosen the terminal screw. Pulling the cable connected to the module may result in malfunction or damage to the module or cable.
- 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.
- A protective film is attached to the top of the module to prevent foreign matter, such as wire chips, from entering the module during wiring. A protective film is attached to the top of the module to prevent foreign matter, such as wire chips, from entering the module during wiring. Remove it for heat dissipation before system operation.

[Setup and Maintenance Precautions]

⚠WARNING

- Do not touch any terminal while power is on. Doing so will cause electric shock or malfunction.
- Shut off the external power supply (all phases) used in the system before cleaning the module or retightening the terminal screws, connector screws, or module fixing screws. Failure to do so may result in electric shock or cause the module to fail or malfunction.

[Setup and Maintenance Precautions]

⚠CAUTION

- Do not disassemble or modify the modules. Doing so may cause failure, malfunction, injury, or a fire.
- Shut off the external power supply (all phases) used in the system before mounting or removing the module.
- After the first use of the product, do not mount/remove the module to/from the base unit more than 50 times (IEC 61131-2 compliant). Exceeding the limit of 50 times may cause malfunction.
- Always make sure to touch the grounded metal to discharge the electricity charged in the body, etc., before touching the module. Failure to do so may cause a failure or malfunctions of the module.

[Precautions When Disposing of This Product]

⚠CAUTION

- Dispose of this product as an industrial waste.

●CONDITIONS OF USE FOR THE PRODUCT●

(1) Mitsubishi programmable controller ("the PRODUCT") shall be used in conditions;
i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and
ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.
MITSUBISHI SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI'S USER, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.
("Prohibited Application")
Prohibited Applications include, but not limited to, the use of the PRODUCT in;

- Nuclear Power Plants and any other power plants operated by Power companies, and/or any other cases in which the public could be affected if any problem or fault occurs in the PRODUCT.
- Railway companies or Public service purposes, and/or any other cases in which establishment of a special quality assurance system is required by the Purchaser or End User.
- Aircraft or Aerospace, Medical applications, Train equipment, transport equipment such as Elevator and Escalator, Incineration and Fuel devices, Vehicles, Manned transportation, Equipment for Recreation and Amusement, and Safety devices, handling of Nuclear or Hazardous Materials or Chemicals, Mining and Drilling, and/or other applications where there is a significant risk of injury to the public or property.

Notwithstanding the above restrictions, Mitsubishi may in its sole discretion, authorize use of the PRODUCT in one or more of the Prohibited Applications, provided that the usage of the PRODUCT is limited only for the specific applications agreed to by Mitsubishi and provided further that no special quality assurance or fail-safe, redundant or other safety features which exceed the general specifications of the PRODUCTs are required. For details, please contact the Mitsubishi representative in your region.

INTRODUCTION

Thank you for purchasing the Mitsubishi Electric MELSEC-Q series programmable controllers. This manual describes the operating procedure, system configuration, parameter settings, functions, programming, and troubleshooting of the Ethernet interface modules: QJ71E71-100, QJ71E71-B5, and QJ71E71-B2 (hereafter referred to as E71).

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the MELSEC-Q series programmable controller to handle the product correctly.
When applying the program examples introduced in this manual to the actual system, ensure the applicability and confirm that it will not cause system control problems.

Please make sure that the end users read this manual.

COMPLIANCE WITH THE EMC AND LOW VOLTAGE DIRECTIVES

(1) For programmable controller system

To ensure that Mitsubishi Electric programmable controllers maintain EMC and Low Voltage Directives when incorporated into other machinery or equipment, certain measures may be necessary. Please refer to one of the following manuals.

- QCPU User's Manual (Hardware Design, Maintenance and Inspection)
- Safety Guidelines (This manual is included with the CPU module or base unit.)

The CE mark on the side of the programmable controller indicates compliance with EMC and Low Voltage Directives.

(2) For the product

To ensure that this product maintains EMC and Low Voltage Directives, please refer to one of the manuals listed under (1).

RELEVANT MANUALS

This manual describes the basic specifications, functions, and usage of the E71.

(1) E71 relevant manual

Manual name <manual number, model code>	Description
MELSEC-Q/L Ethernet Interface Module User's Manual (Application) <SH-080010, 13JL89>	E-mail function, communication function (communications via CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, or MELSECNET/10, and communications by using the data link instructions), and file transfer (FTP server) function of the E71
MELSEC-Q/L Ethernet Interface Module User's Manual (Web function) <SH-080180, 13JR40>	Web function of the E71
MELSEC Communication Protocol Reference Manual <SH-080008, 13JF89>	Details of MELSEC communication protocol (MC protocol) that is used for data communication between a target device and a CPU module

(2) Operating manual

Manual name <manual number, model code>	Description
GX Works2 Version1 Operating Manual (Common) <SH-080779, 13JU63>	System configuration, parameter settings, and online operations (common to Simple project and Structured project) of GX Works2
GX Works2 Version 1 Operating Manual (Intelligent Function Module) <SH-080921, 13JU69>	Parameter settings for intelligent function modules, monitoring operations, and the predefined protocol support function of GX Works2
GX Developer Version 8 Operating Manual <SH-080373, 13JU41>	Operating methods of GX Developer, such as programming, printing, monitoring, and debugging

CONTENTS

MANUAL PAGE ORGANIZATION

In this manual, pages are organized and the symbols are used as shown below.

The following illustration is for explanation purpose only, and should not be referred to as an actual documentation.

"" is used for window names and items.

1. shows operating procedures.

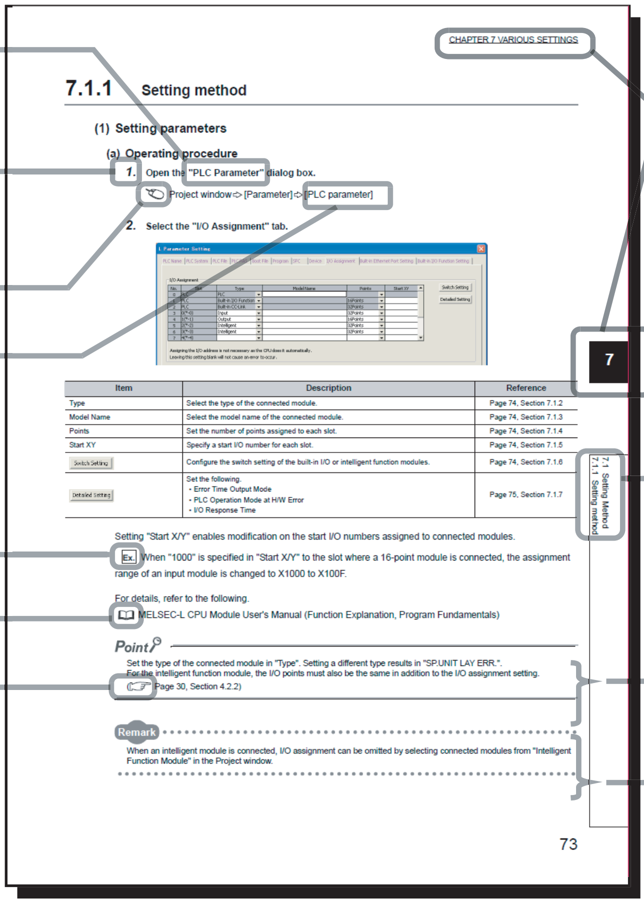
shows mouse operations.*1

[ ] is used for items in the menu bar and the project window.

Ex. shows setting or operating examples.

shows reference manuals.

shows reference pages.



The chapter of the current page is shown.

The section of the current page is shown.

Point shows notes that requires attention.

Remark shows useful information.

*1 The mouse operation example is provided below.

Menu bar

Ex. [Online] ⇨ [Write to PLC...]

Select [Online] on the menu bar, and then select [Write to PLC...].

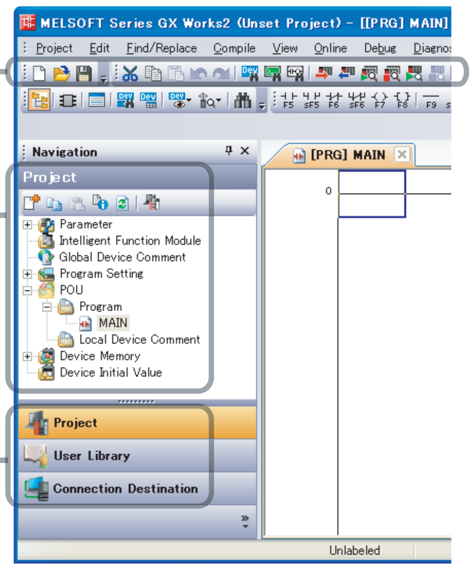
A window selected in the view selection area is displayed.

Ex. Project window ⇨ [Parameter] ⇨ [PLC Parameter]

Select [Project] from the view selection area to open the Project window.

In the Project window, expand [Parameter] and select [PLC Parameter].

View selection area



TERM

Unless otherwise specified, this manual uses the following terms.

Term	Description
ACPU	A generic term for the AnNCPU, AnACPU, and AnUCPU
AnACPU	A generic term for the A2ACPU, A2ACPU-S1, A2ACPUP21/R21, A2ACPUP21/R21-S1, A3ACPU, and A3ACPUP21/R21
AnNCPU	A generic term for the A1NCPU, A1NCPUP21/R21, A2NCPU, A2NCPU-S1, A2NCPUP21/R21, A2NCPUP21/R21-S1, A3NCPU, and A3NCPUP21/R21
AnUCPU	A generic term for the A2UCPU, A2UCPU-S1, A2ASCPU, A2ASCPU-S1, A3UCPU, and A4UCPU
ARP	An abbreviation for Address Resolution Protocol. This protocol is used to obtain the MAC address of Ethernet from an IP address.
BUFRCV	An abbreviation for ZP.BUFRCV
BUFRCVS	An abbreviation for Z.BUFRCVS
BUFSND	An abbreviation for ZP.BUFSND
CLOSE	An abbreviation for ZP.CLOSE
C24	Another name for the Q series serial communication module
DNS	An abbreviation for Domain Name System. This system is mainly used to convert host names on the Internet or domain names used for e-mails to IP addresses.
ECPRTCL	An abbreviation for GP.ECPRTCL
ERRCLR	An abbreviation for ZP.ERRCLR
ERRRD	An abbreviation for ZP.ERRRD
Built-in Ethernet port QCPU	A generic term for the Q03UDVCPU, Q03UDECPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU
E71	A generic term for the Ethernet interface modules: QJ71E71-100, QJ71E71-B5, and QJ71E71-B2
E71-mounted station	An abbreviation for the station where the E71 is mounted
FTP	An abbreviation for File Transfer Protocol. This protocol is used to transfer data files over a network.
GX Developer	The product name of the software package for the MELSEC programmable controllers
GX Works2	
HTTP	An abbreviation for Hyper Text Transfer Protocol. This protocol is used to send and receive content, such as HTML files, between a Web browser and a Web server.
ICMP	An abbreviation for Internet Control Message Protocol. This protocol is used to exchange messages of errors in an IP network or other information related to an Ethernet network.
IP	An abbreviation for Internet Protocol
MAC address	A unique identifier assigned to each external device on a network. This address is also known as an Ethernet hardware address.
MC protocol	An abbreviation for MELSEC Communication Protocol. This protocol is used to access MC protocol supporting modules, such as the C24 and E71, or programmable controllers connected to MC protocol supporting modules from external devices.
MELSECNET/H	An abbreviation for a MELSECNET/H network system
MELSECNET/H remote I/O station	A generic term for the QJ72LP25-25, QJ72LP25G, and QJ72BR15
MELSECNET/10	An abbreviation for a MELSECNET/10 network system
MRECV	An abbreviation for ZP.MRECV
MSEND	An abbreviation for ZP.MSEND
MX Component	An abbreviation for MX Component (SW0D5C-ACT-E or later)
OPEN	An abbreviation for ZP.OPEN

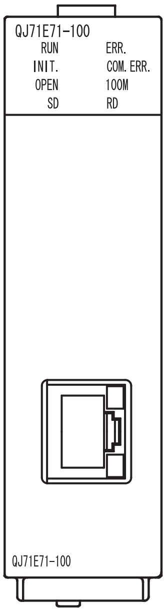
Term	Description
OPS	A generic term for the partner products with built-in EZSocket that supports a redundant system. The E71 communicates with an OPS using a connection specified by a user.
POP3	An abbreviation for Post Office Protocol Ver.3. This protocol is used to transfer e-mails from a mail server to a local computer.
QCPU	A generic term for the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU
QCPU-mounted station	An abbreviation for the programmable controller where the QCPU is mounted
QnACPU	A generic term for the Q2ACPU, Q2ACPU-S1, Q2ASCPU, Q2ASCPU-S1, Q2ASHCPU, Q2ASHCPU-S1, Q3ACPU, Q4ACPU, and Q4ARCPU
READ	An abbreviation for JP.READ and GP.READ
RECV	An abbreviation for JP.RECV and GP.RECV
RECVS	An abbreviation for Z.RECVS
REQ	An abbreviation for J.REQ, JP.REQ, G.REQ, and GP.REQ
SEND	An abbreviation for JP.SEND and GP.SEND
SLMP	An abbreviation for Seamless Message Protocol. This protocol is used to access an SLMP-compatible device or a programmable controller connected to an SLMP-compatible device from an external device.
SMTP	An abbreviation for Simple Mail Transfer Protocol. This protocol is used to transfer e-mails over the Internet.
SREAD	An abbreviation for JP.SREAD and GP.SREAD
SWRITE	An abbreviation for JP.SWRITE and GP.SWRITE
UINI	An abbreviation for ZP.UINI
WRITE	An abbreviation for JP.WRITE and GP.WRITE
ZNRD	An abbreviation for J.ZNRD and JP.ZNRD
ZNWR	An abbreviation for J.ZNWR and JP.ZNWR
Intelligent function module	A module that has functions other than an input or output, such as an A/D converter module and D/A converter module
Subnet mask	A number used to logically divide one network into multiple subnetworks and manage them easily. The following Ethernet network systems can be configured: A small-scale Ethernet network system in which multiple network devices are connected A medium- or large-scale network system in which multiple small-scale network systems are connected via routers or other network communication devices
Predefined protocol support function	A function of GX Works2. This function sets protocols appropriate to each external device and reads/writes protocol setting data from/to the flash ROM of the E71.
Device	A device (X, Y, M, D, or others) in a CPU module
High Performance model QCPU	A generic term for the Q02(H)CPU, Q06HCPU, Q12HCPU, and Q25HCPU
Buffer memory	A memory in an intelligent function module, where data (such as setting values and monitoring values) exchanged with a CPU module are stored
Buffer memory address	An address that indicates the storage location of data assigned to the buffer memory in an intelligent function module
Programming tool	A generic term for GX Works2 and GX Developer
Process CPU	A generic term for the Q02PHCPU, Q06PHCPU, Q12PHCPU, and Q25PHCPU
Basic model QCPU	A generic term for the Q00(J)CPU and Q01CPU
Universal model QCPU	A generic term for the Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU
Redundant CPU	A generic term for the Q12PRHCPU and Q25PRHCPUU
Safety CPU	Another name for the QS001CPU

Term	Description
Dedicated Instruction	An instruction that simplifies programming for using functions of intelligent function modules

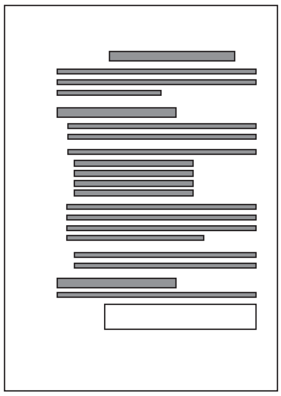
PACKING LIST

The following items are included in the package of this product. Before use, check that all the items are included.

QJ71E71-100

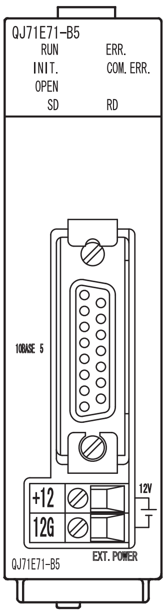


QJ71E71-100

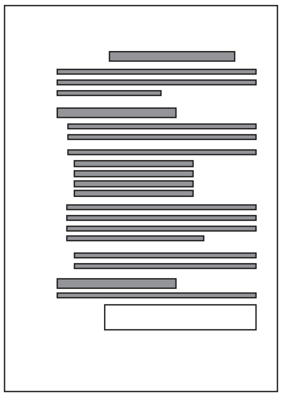


Before Using the Product

QJ71E71-B5

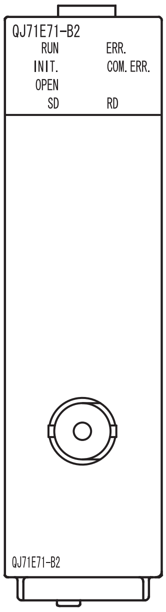


QJ71E71-B5

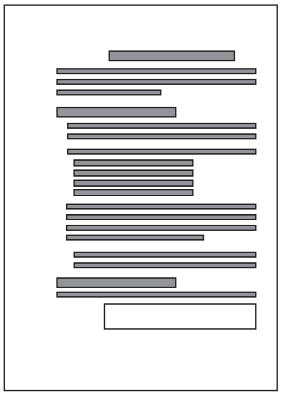


Before Using the Product

QJ71E71-B2

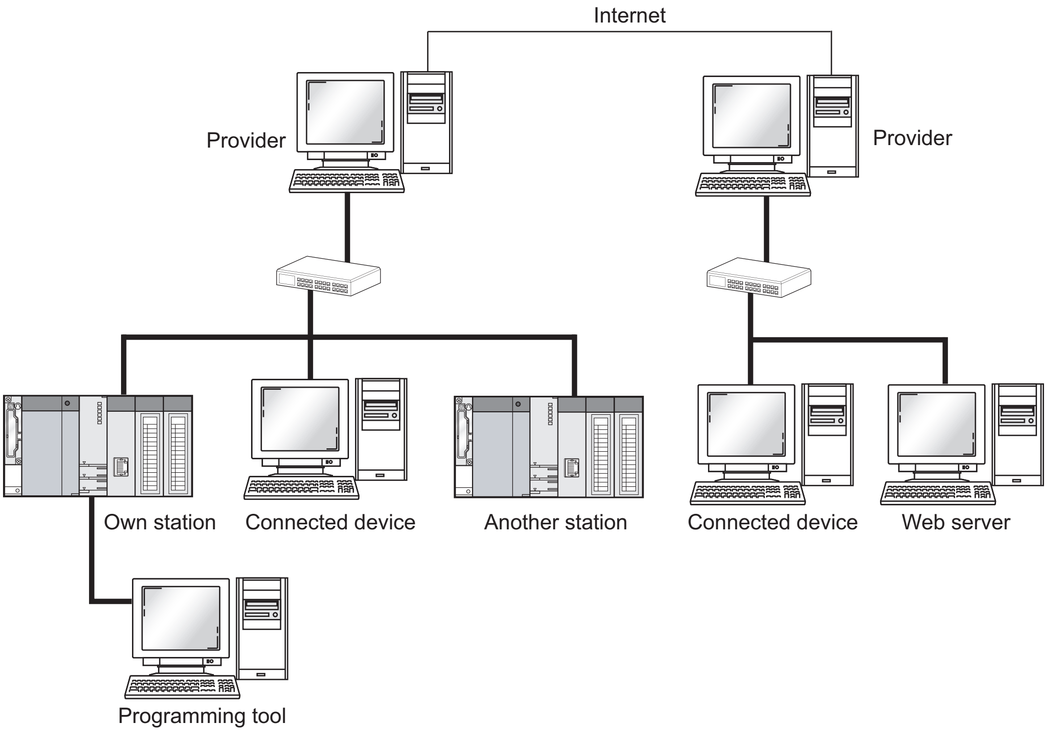


QJ71E71-B2


Before Using the Product

CHAPTER 1 FEATURES

An Ethernet module (hereafter abbreviated as E71) is an interface module on the programmable controller side for connecting a programmable controller system to the host system, such as a personal computer and a workstation, over Ethernet. The module can collect and modify programmable controller data, monitor and control CPU operating status, and exchange data in TCP/IP or UDP/IP.

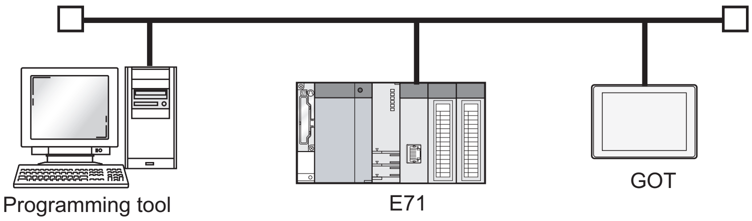


The E71 has the following basic functions.

- Connection with MELSOFT products and a GOT
- Collection and modification of CPU module data from connected devices (MC protocol communications)
- Communications using SLMP
- Data communications using the predefined protocol (predefined protocol support function)
- Exchange of data with connected devices (communications using a fixed buffer and random access buffer)
- Prevention of unauthorized access through the IP filter (IP filter function)
- Prevention of unauthorized access through a remote password (remote password)
- E-mail sending/receiving (e-mail function)
- Data sending/receiving using the Web function

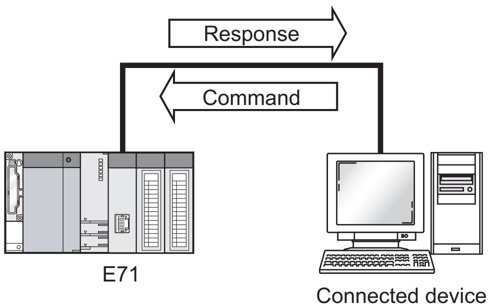
(1) Connection with MELSOFT products and a GOT

In Ethernet, a programming tool can create programming of a programmable controller and monitor a programmable controller (MELSOFT connection), and the GOT can monitor and test a programmable controller. Remote operations making full use of the Ethernet capability, long-distance connectivity and high-speed communications, are achieved.



(2) Collection and modification of CPU module data from connected devices (MC protocol communications)

The MC protocol enables connected devices to access MC protocol supporting modules over Ethernet. The E71 can communicate with a personal computer and HMI (Human Machine Interface) as long as the connected devices can receive/send messages in the MC protocol control procedure. By using a separately sold communication support tool (MX Component), a communication program for the host system can be created without considering detailed protocols (communication procedures). (Page 103, CHAPTER 9)

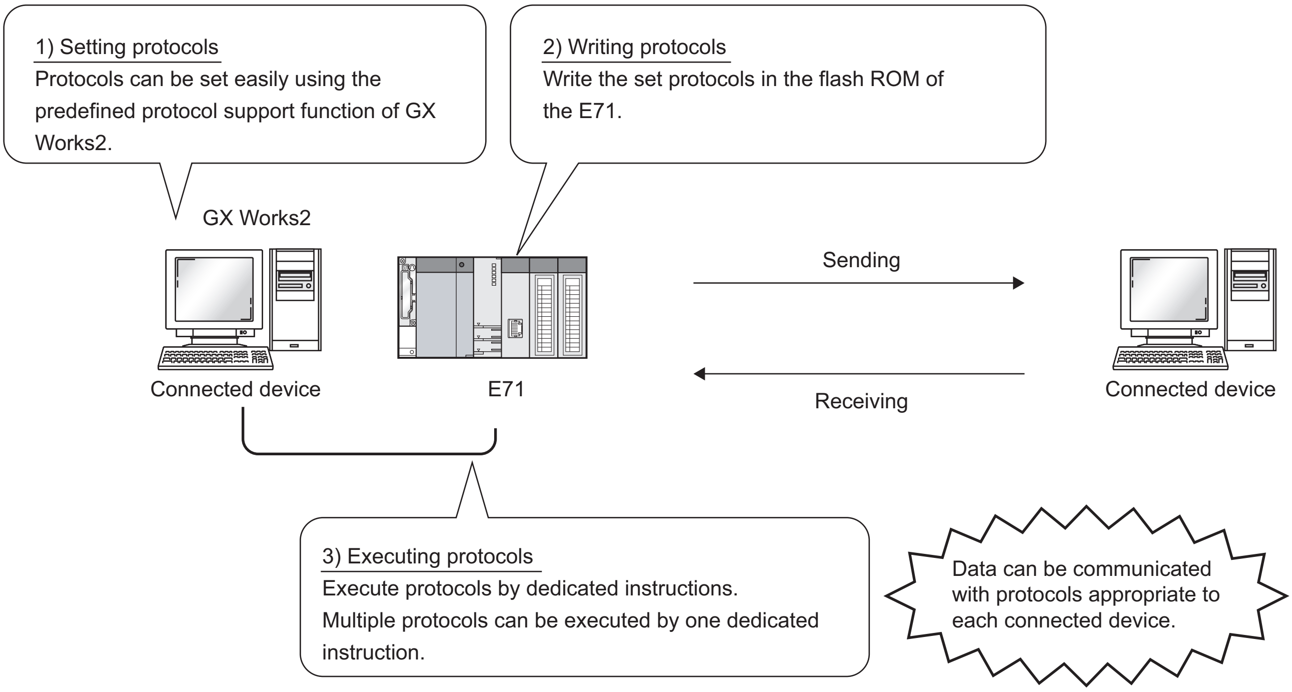


(3) Communications using SLMP

SLMP is a protocol that enables connected devices to access SLMP supporting devices over Ethernet. SLMP communications are available among devices that can receive/send messages in the SLMP control procedure. (Page 107, CHAPTER 10)

(4) Data communications using the predefined protocol (predefined protocol support function)

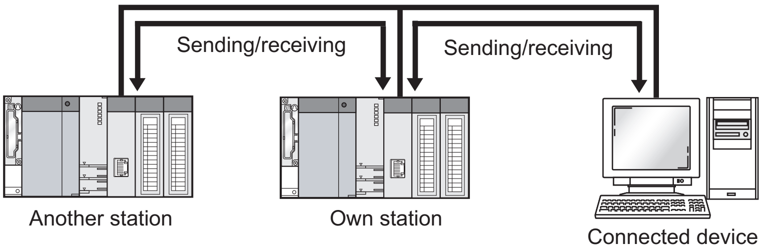
Registering protocol data in advance using GX Works2 allows communications by executing only an ECPRTCL instruction program. In addition, the protocol setting required to communicate with the connected device, such as a measuring instrument or a bar code reader, can be configured easily using the Predefined Protocol Support Function of GX Works2. (Page 112, CHAPTER 11)



(5) Exchange of data with connected devices (communications using a fixed buffer and random access buffer)

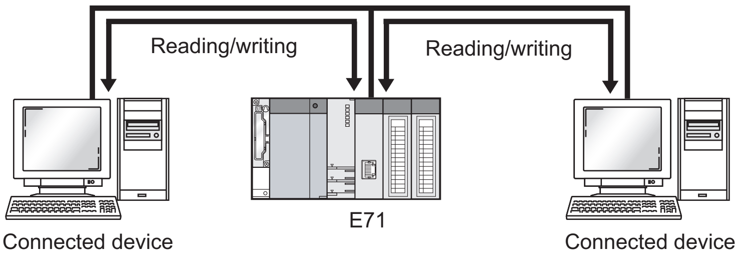
(a) Communications using a fixed buffer

Up to 1K-word data can be exchanged among programmable controllers or between a programmable controller and the host system. While MC protocol communications are passive, communications using a fixed buffer are an active protocol. If an error occurs in equipment or certain conditions are met, the programmable controller can send data to the host system. Using an interrupt program allows the CPU module to quickly read received data. (Page 139, CHAPTER 12)



(b) Communications using a random access buffer

Up to 6K-word data can be communicated. This buffer is useful when the data size is too large for communications using a fixed buffer (capacity: 1K word). (Page 174, CHAPTER 13)



(6) Prevention of unauthorized access through the IP filter

The IP address of the connected device to be allowed or denied access is set in the buffer memory, and access from connected devices is restricted. (Page 195, Section 14.3)

(7) Prevention of unauthorized access through a remote password

This function prevents unauthorized remote access to the CPU module. The E71 checks an entered remote password in data communications from a connected device using remote password-protected connection. (Page 201, Section 14.4)

(8) E-mail sending/receiving (e-mail function)

This function sends and receives e-mails to and from a connected device in a remote location via the Internet.
For details, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Application)

(a) E-mail sending/receiving through the CPU module

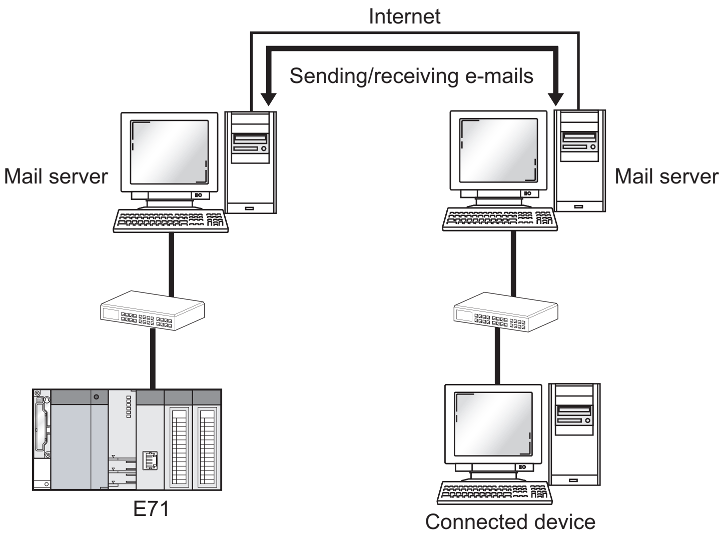
The following data can be sent and received using the MSEND/MRECV instructions.

- The CPU module can receive/send up to 6K-word data from/to a personal computer or other E71 modules as an e-mail attachment.
- The CPU module can send up to 960-word data to a personal computer or portable terminal as the main text of an e-mail.

(b) E-mail sending using the programmable controller CPU monitoring function

Notification conditions (CPU module status or device values) that have been set using parameters are regularly monitored. When the conditions are met, up to 960-word data can be sent by either of the following data formats.

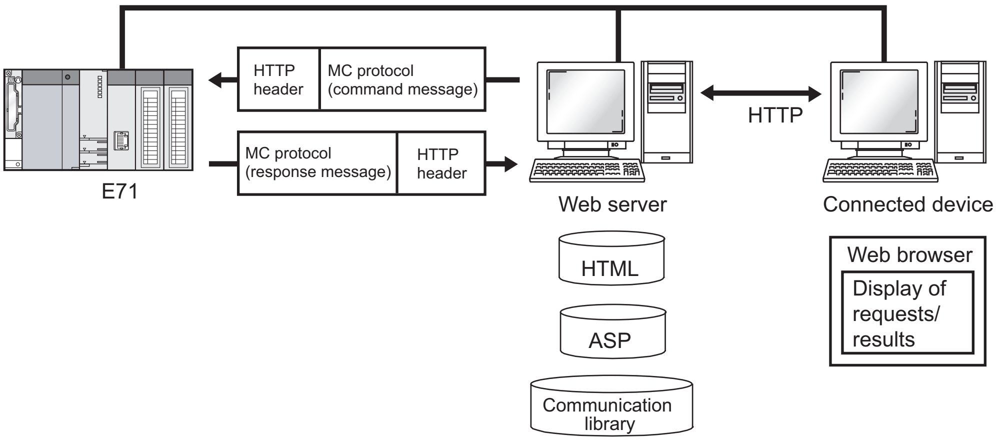
- Attachment
- Main text



(9) Data sending/receiving using the Web function

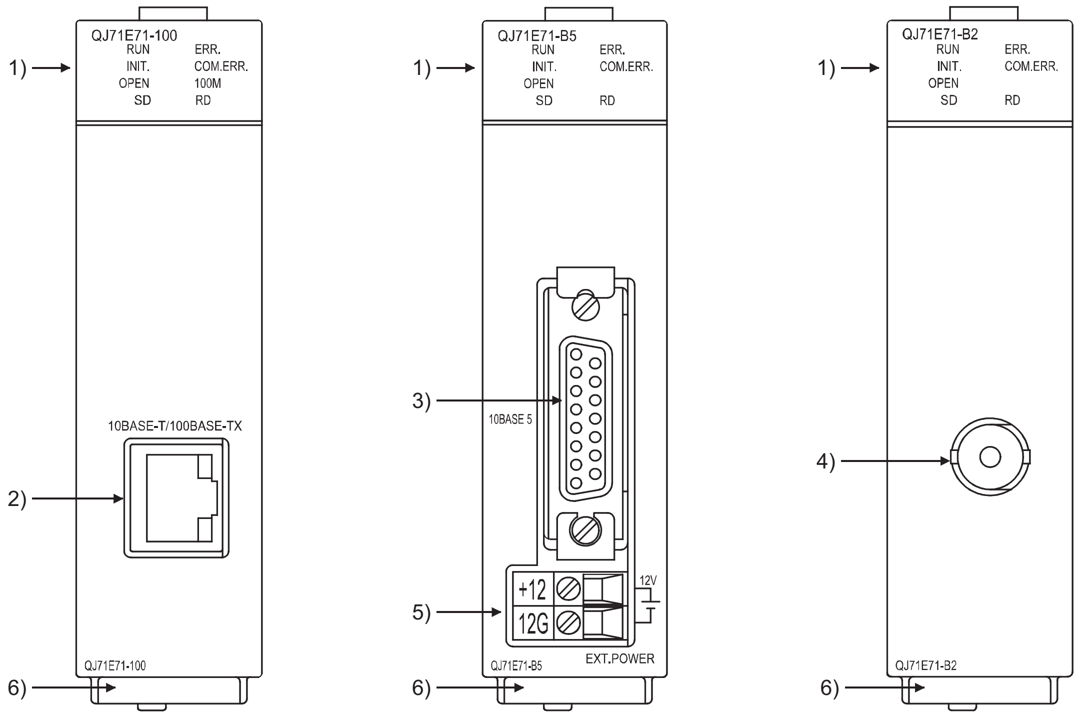
The system administrator can monitor a CPU module in a remote location via the Internet using a commercially available Web browser. For details, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Web function)



CHAPTER 2 PART NAMES

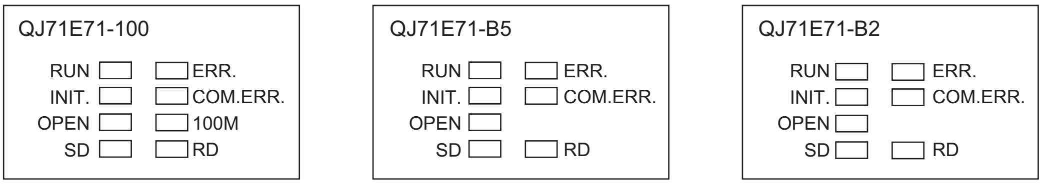
This chapter describes the E71 parts.



No.	Name	Application
1)	LED indicator	Refer to (1) in this chapter.
2)	10BASE-T/100BASE-TX connector (RJ45)*1	A connector to connect the E71 to the 10BASE-T or 100BASE-TX network (The E71 determines whether to use 10BASE-T or 100BASE-TX according to the hub.)
3)	10BASE5 connector	A connector to connect the E71 to the 10BASE5 network (for connecting a 10BASE5 AUI cable (transceiver cable))
4)	10BASE2 connector	A connector to connect the E71 to the 10BASE2 network (for connecting a 10BASE2 coaxial cable)
5)	External power supply terminal	A power supply terminal for supplying power to a transceiver in the 10BASE5 network (13.28VDC to 15.75VDC)
6)	Serial number display	A display indicating the serial number printed on the rating plate

*1 The LED on the connector does not turn on. Depending on the serial number, the connector orientation is left-right reversal.

(1) LED indication



LED name		Description
RUN		Indicates operating status.
	ON	In normal operation
	OFF	An error has occurred. (Page 286, Section 16.4.1)
INIT.		Indicates initial process status.
	ON	Normal completion
	OFF	Not processed
OPEN*1		Indicates open process status.
	ON	An open process normally completed (connection open)
	OFF	An open process not completed (no connection)
SD		Indicates whether data is being sent.
	Flashing	Data being sent
	OFF	Data not sent (Page 287, Section 16.4.3)
ERR.		Indicates whether the setting is correct.
	ON	The setting is incorrect. (Page 286, Section 16.4.2)
	OFF	Correct setting
COM.ERR.*2		Indicates whether a communication failure has occurred.
	ON	A communication failure has occurred. (Page 286, Section 16.4.2)
	OFF	Normal communications in progress
100M		Indicates a transmission speed.
	ON	100Mbps
	OFF	10Mbps or a cable not connected
RD		Indicates whether data is being received.
	ON	Data being received
	OFF	Data not received (Page 287, Section 16.4.4)

*1 The OPEN LED turns on and off depending on the open status of user connections 1 to 16. (The open status of the system connections (e.g. automatic open UDP port) is not included.)

*2 If the COM.ERR. LED is on, it does not turn off even if the error cause is eliminated. For how to turn off the LED, refer to "How to Turn Off the COM.ERR. LED". (Page 340, Section 16.8)

CHAPTER 3 SPECIFICATIONS

This chapter describes the performance specifications, functions, CPU module I/O signals, and buffer memory areas of an E71.

3.1 General Specifications

For the general specifications of an E71, refer to the following.

"Safety Guidelines", the manual included with the CPU module or base unit

3.2 Performance Specifications

The following table lists the performance specifications of an E71.

Item			Specifications			
			QJ71E71-100		QJ71E71-B5	QJ71E71-B2
			100BASE-TX	10BASE-T	10BASE5	10BASE2
Transmission specifications	Data transmission speed		100Mbps (Full-duplex/Half-duplex)	10Mbps (Half-duplex)		
	Flow control		Full-duplex: None*8 Half-duplex: Back pressure congestion control	Back pressure congestion control (Half-duplex)		
	Interface		RJ45 (Fixed to MDI)		AUI	BNC
	Transmission method		Base band			
	Maximum node-to-node distance		-		2500m	925m
	Maximum segment length*6		100m (length between a hub and node)*7		500m	185m
	Maximum number of nodes/connection*6		Cascade connection (maximum of 2 levels*1)	Cascade connection (maximum of 4 levels*1)	100 modules/ segment	30 modules/ segment
	Minimum interval between nodes*6		-		2.5m	0.5m
Sending/receiving data storage memory	Number of simultaneous open connections		16 connections (Connections usable on a program)			
	Fixed buffer		1K word × 16			
	Random access buffer		6K words × 1			
	E-mail	Attachment	6K words × 1			
		Main text	960 words × 1			
Number of occupied I/O points			32 points per slot (I/O assignment: Intelligent 32 points)			
Internal current consumption (5VDC)			0.50A		0.50A	0.60A*2
12VDC external power supply capacity (Transceiver)			-		*3	-
External dimensions			98 (H) × 27.4 (W) × 90 (D) [mm]			
Weight			0.11kg		0.12kg	0.13kg*2

Item			Specifications			
			QJ71E71-100		QJ71E71-B5	QJ71E71-B2
			100BASE-TX	10BASE-T	10BASE5	10BASE2
Transmission specifications sending/receiving data	Data size	Attachment	6K words × 1			
		Main text	960 words × 1			
	Data transfer method		When sending: Send either a file as attachment or main text (select one). When receiving: Receive a file as attachment.			
	Subject		Us-ASCII format or ISO-2022-JP (Base64)			
	Attachment format		MIME format			
	MIME		Version 1.0			
	Data of attachment format		Can be selected from binary, ASCII, and CSV. File name: XXXX.bin (binary), XXXX.asc (ASCII), XXXX.csv (CSV) (CSV: Comma Separated Value)			
	Division of attachment		Cannot be divided (Only one file can be sent/received.)*4			
	When sending (encode)		Subject: Base64/7 bits Main text: 7 bits Attachment: Base64			
	When receiving (decode)		Subject: (Does not decode) Main text: (Cannot be received) Attached file: Base64/7 bits/8 bits/Quoted Printable*5			
	Encryption		None			
	Compression		None			
	Communications with a mail server		SMTP (sending server) Port number = 25, POP3 (receiving server) Port number = 110			
	Operation check mailer		Microsoft® Corporation Internet Explorer 5.0 (Outlook Express 5.5/Outlook Express 5) Netscape® Communications Corporation Netscape® 4.05			

*1 This applies when a repeater hub is used. For the number of levels that can be constructed when a switching hub is used, consult with the manufacturer of the switching hub used.

*2 As described below, a module with a serial number (first five digits) of "05049" or earlier has a different 5VDC internal current consumption value and weight.

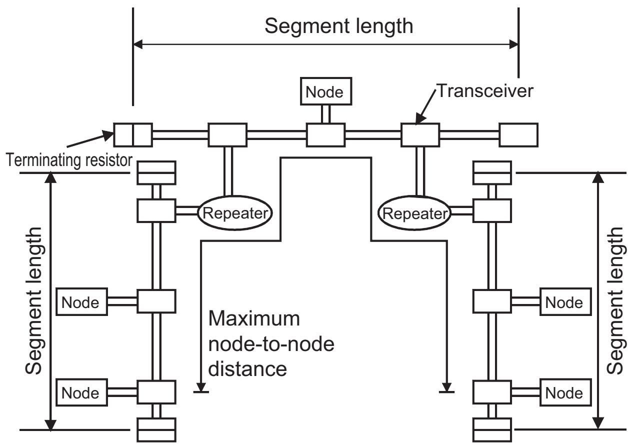
- Internal current consumption (5VDC): 0.70A
- Weight: 0.14kg

*3 The specifications of the transceiver and the AUI cable need to be met. (Page 74, Section 5.2.2)

*4 If divided files are received, only the first file is received and the remaining files are discarded.

*5 If an e-mail is sent from a connected device to the programmable controller side, specify the encoding method (Base64/7 bits/8 bits/Quoted Printable) of the attachment.

*6 The following figure shows segment lengths and node intervals.



*7 For the maximum segment length (the length between hubs), consult the manufacturer of the hub used.

*8 The QJ71E71-100 does not support the flow control of the IEEE802.3x.

3.3 Function List

This section lists the E71 functions.

3.3.1 Function list

The following table lists the functions of the E71.

(1) Basic functions

The following table lists the basic E71 functions explained in this manual.

Function		Description	Reference
Connecting with MELSOFT products and a GOT		An E71 can be connected with MELSOFT products, such as a programming tool and MX Component, and a GOT.	Page 100, CHAPTER 8
MC protocol communications		CPU module data can be read/written from/to connected devices. Access to files can be also performed.	Page 103, CHAPTER 9
Communications using SLMP (only QJ71E71-100)		The connected device can read/write data from/to the buffer memory or device of an SLMP supporting device connected to the shared network with the E71. In addition, the connected device can read/write data from/to the device of the CPU module where an E71 is mounted.	Page 107, CHAPTER 10
Data communications using the predefined protocol (only QJ71E71-100)		Data can be sent/received with protocols appropriate to each connected device. The connected device side protocol can be easily selected, or created/edited from the Predefined Protocol Library of GX Works2.	Page 112, CHAPTER 11
Communications using a fixed buffer	Procedure exists	Any data is sent/received between a CPU module and connected devices using the fixed buffer of an E71.	Page 139, CHAPTER 12
	No procedure		
	Pairing open	Paring receiving/sending connections enables data communications with two connections by performing the open process for one port.	Page 158, Section 12.7
	Broadcast communications	Broadcast communications are enabled with all E71-mounted stations in the same Ethernet network that is connected to the E71 when "No Procedure" communications using a fixed buffer are performed using UDP/IP.	Page 160, Section 12.8
Communications using a random access buffer		Data is read/written from multiple connected devices to the random access buffer of an E71.	Page 174, CHAPTER 13
Router relay function		Data communications are performed through a router and a gateway. This function is not the function where an E71 operates as a router.	Page 189, Section 14.1
Communications using an auto-open UDP port		Communications are enabled without the open/close processes after an E71-mounted station is started up.	Page 194, Section 14.2
IP filter function		The IP address of the connected device to be allowed or denied access is set in the buffer memory, and access from connected devices is restricted.	Page 195, Section 14.3
Remote password		Unauthorized remote access to a CPU module is prevented.	Page 201, Section 14.4
Hub connection status monitor function (only QJ71E71-100)		The current connection status and transmission speed of an E71 and a hub and the number of times that the E71 detected disconnection can be checked.	Page 211, Section 14.5
IP address in use detection function (only QJ71E71-100)		If different stations in the same network use the same IP address, the address in use can be detected.	Page 212, Section 14.6
Network configuration in a redundant system		A network can be configured in a redundant system.	Page 214, Section 14.7
Alive check function		Whether a connected device is normally operating after a connection is established (open process) can be checked.	Page 234, Section 14.8

Function	Description	Reference
Module error collection function (only QJ71E71-100)	An error that has occurred in an E71 can be stored in the CPU module as error history. The history data can be stored on a memory with the backup power feature; therefore error details are held even if the CPU module is reset or the system is powered off.	Page 285, Section 16.3

(2) Special functions

The following special functions are also available. For the functions, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Application)

Function	Description
E-mail function	Data are sent/received using an e-mail. • Data sent/received by a CPU module • Data sent using the programmable controller CPU monitoring function (automatic notification function)
CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications	Data are communicated over multiple network systems where Ethernet and other networks exist together or network systems that relay multiple Ethernet networks.
Communications using data link instructions	Data of a CPU module in other stations can be read/written over Ethernet using data link instructions.
File transfer (FTP server) function	Data can be read/written in files from connected devices using an exclusive FTP command.

(3) Web function

This function allows data to be read/written from/to a remote CPU module over the Internet using a commercially available Web browser. For the function, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Web function)

3.3.2 Use with other functions

The following table lists the relationships between functions that can be used together.

○: Available, ×: Not available or this function does not correspond to the functions in the "Communication function" column.

Communication function (Communication method)	CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications	Router relay function	Alive check function	Pairing open	Communications using an auto-open UDP port	IP filter function	Remote password	Broadcast communications
MC protocol communications (TCP/IP, UDP/IP)	○*1	○	○*2	×	○*1	○	○	○*1 *3
Communications using a fixed buffer (TCP/IP, UDP/IP)	×	○	○	○	×	○	○	○*3 *4
Communications using SLMP (TCP/IP, UDP/IP)	×	○	○	×	○	○	○	○*3
Data communications using the predefined protocol (TCP/IP, UDP/IP)	×	○	○	○	×	○	○	○*3
Communications using a random access buffer (TCP/IP, UDP/IP)	×	○	○	×	×	○	○	×
E-mail function (TCP/IP)	×	○	×	×	×	○	×	×
Communications using data link instructions (UDP/IP)	○	○	×	×	○	○	×	○
File transfer (FTP server) function (TCP/IP)	×	○	×	×	×	○	○	×
Web function (TCP/IP)	○	○	×	×	×	○	○	×

*1 These functions cannot be used with an A-compatible 1E frame.
*2 The auto-open UDP port is excluded.
*3 These functions can be used only for UDP/IP communication.
*4 These functions cannot be used for communications using a fixed buffer in the "Procedure Exist" control method.

3.4 List of I/O Signals

The following table lists the I/O signals for an E71. The I/O signal assignment of when the start I/O number of an E71 is 0000 (the module is mounted on the slot 0 of a main base unit) is listed below.

Device number	Signal name	Device number	Signal name
X0	For fixed buffer communication of connection No.1 (ON: Sending normal completion or reception completion, OFF: -)	Y0	Connection No.1 (ON: At sending request or reception complete confirmation signal, OFF: -)
X1	For fixed buffer communication of connection No.1 (ON: Detection of sending error or reception error, OFF: -)	Y1	Connection No.2 (ON: At sending request or reception complete confirmation signal, OFF: -)
X2	For fixed buffer communication of connection No.2 (ON: Sending normal completion or reception completion, OFF: -)	Y2	Connection No.3 (ON: At sending request or reception complete confirmation signal, OFF: -)
X3	For fixed buffer communication of connection No.2 (ON: Detection of sending error or reception error, OFF: -)	Y3	Connection No.4 (ON: At sending request or reception complete confirmation signal, OFF: -)
X4	For fixed buffer communication of connection No.3 (ON: Sending normal completion or reception completion, OFF: -)	Y4	Connection No.5 (ON: At sending request or reception complete confirmation signal, OFF: -)
X5	For fixed buffer communication of connection No.3 (ON: Detection of sending error or reception error, OFF: -)	Y5	Connection No.6 (ON: At sending request or reception complete confirmation signal, OFF: -)
X6	For fixed buffer communication of connection No.4 (ON: Sending normal completion or reception completion, OFF: -)	Y6	Connection No.7 (ON: At sending request or reception complete confirmation signal, OFF: -)
X7	For fixed buffer communication of connection No.4 (ON: Detection of sending error or reception error, OFF: -)	Y7	Connection No.8 (ON: At sending request or reception complete confirmation signal, OFF: -)
X8	For fixed buffer communication of connection No.5 (ON: Sending normal completion or reception completion, OFF: -)	Y8	Connection No.1 (ON: Open request, OFF: -)
X9	For fixed buffer communication of connection No.5 (ON: Detection of sending error or reception error, OFF: -)	Y9	Connection No.2 (ON: Open request, OFF: -)
XA	For fixed buffer communication of connection No.6 (ON: Sending normal completion or reception completion, OFF: -)	YA	Connection No.3 (ON: Open request, OFF: -)
XB	For fixed buffer communication of connection No.6 (ON: Detection of sending error or reception error, OFF: -)	YB	Connection No.4 (ON: Open request, OFF: -)
XC	For fixed buffer communication of connection No.7 (ON: Sending normal completion or reception completion, OFF: -)	YC	Connection No.5 (ON: Open request, OFF: -)
XD	For fixed buffer communication of connection No.7 (ON: Detection of sending error or reception error, OFF: -)	YD	Connection No.6 (ON: Open request, OFF: -)
XE	For fixed buffer communication of connection No.8 (ON: Sending normal completion or reception completion, OFF: -)	YE	Connection No.7 (ON: Open request, OFF: -)
XF	For fixed buffer communication of connection No.8 (ON: Detection of sending error or reception error, OFF: -)	YF	Connection No.8 (ON: Open request, OFF: -)
X10	Open completed for connection No.1 (ON: Open completion signal, OFF: -)	Y10	Use prohibited
X11	Open completed for connection No.2 (ON: Open completion signal, OFF: -)	Y11	Use prohibited

Device number	Signal name	Device number	Signal name
X12	Open completed for connection No.3 (ON: Open completion signal, OFF: -)	Y12	Use prohibited
X13	Open completed for connection No.4 (ON: Open completion signal, OFF: -)	Y13	Use prohibited
X14	Open completed for connection No.5 (ON: Open completion signal, OFF: -)	Y14	Use prohibited
X15	Open completed for connection No.6 (ON: Open completion signal, OFF: -)	Y15	Use prohibited
X16	Open completed for connection No.7 (ON: Open completion signal, OFF: -)	Y16	Use prohibited
X17	Open completed for connection No.8 (ON: Open completion signal, OFF: -)	Y17	COM.ERR. LED Off request (ON: At off request, OFF: -)
X18	Open abnormal detection signal (ON: At off request, OFF: -)	Y18	Use prohibited
X19	Initial normal completion signal (ON: Normal completion, OFF: -)	Y19	Initial request signal (ON: At request, OFF: -)
X1A	Initial abnormal end signal (ON: Abnormal end, OFF: -)	Y1A	Use prohibited
X1B	Use prohibited	Y1B	Use prohibited
X1C	COM.ERR. LED lit confirmation (ON: lit, OFF: -)	Y1C	Use prohibited
X1D	Predefined protocol ready (ON: Ready, OFF: -)	Y1D	Use prohibited
X1E	Use prohibited	Y1E	Use prohibited
X1F	Watchdog timer error detection (ON: Watchdog timer error, OFF: -)	Y1F	Use prohibited

Point

- Do not use any use prohibited signal as I/O signals to the CPU module. Doing so may cause malfunction of the programmable controller system.
- Do not turn off the Open request signal (Y8 to YF) and turn on the COM.ERR. LED Off request (Y17) with the same sequence scan.

Remark

The I/O signals listed in this section are mainly applied when QnA series module programs are used. For the Q series, the I/O signals for intelligent function modules are turned on and off with dedicated instructions. When QnA series module programs are used, it is recommended to replace the signals with the dedicated instructions described in the section that describes the corresponding functions.
For details on I/O signals, refer to the following.

For QnA Ethernet Interface Module User's Manual

3.5 Buffer Memory

This section describes the E71 buffer memory.

3.5.1 Configuration of the buffer memory

This section describes a buffer memory configuration.

(1) Buffer memory address configuration

A buffer memory area consists of 16 bits per address.

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

(2) Buffer memory area configuration

Buffer memory consists of user areas and system areas.

(a) User areas

A user area is an area where a user writes or reads data. A user area consists of a parameter area for the initial process and data communications, an area for data communications, and an area to store communication status and communication error data. Data communications may take long if user areas are used continually; therefore, use them only when needed.

(b) System areas

A system area is an area used by a system.

Point

Do not write any data to the "system area". Doing so may cause malfunction of the programmable controller system.

3.5.2 List of buffer memory addresses

The following table lists the buffer memory addresses of an E71.

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
0 and 1 (0_H and 1_H)	Initial process parameter setting area	Own station E71 IP address		$C00001FE_H$	○
2 and 3 (2_H and 3_H)		System area		-	-
4 (4_H)		Special function settings • Router relay function (b5, b4) 00: Do not use (default) 01: Use • Conversion system setting for CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay function (b7, b6) 00: Automatic response system (default) 01: IP address computation system 10: Table exchange system 11: Use-together system • FTP function setting (b9, b8) 00: Do not use 01: Use (default) Bits other than those described above are reserved for system use.		0100_H	○
5 to 10 (5_H to A_H)		System area		-	-
11 (B_H)		Monitoring timer	TCP ULP timer value (Setting time = setting value × 500 ms)	$3C_H$ (60)	○
12 (C_H)			TCP zero window timer value (Setting time = setting value × 500 ms)	14_H (20)	○
13 (D_H)			TCP resend timer value (Setting time = setting value × 500 ms)	14_H (20)	○
14 (E_H)			TCP end timer value (Setting time = setting value × 500 ms)	28_H (40)	○
15 (F_H)			IP assembly timer value (Setting time = setting value × 500 ms)	A_H (10)	○
16 (10_H)			Response monitoring timer value (Setting time = setting value × 500 ms)	$3C_H$ (60)	○
17 (11_H)			Destination existence confirmation starting interval timer value (Setting time = setting value × 500 ms)	$4B0_H$ (1200)	○
18 (12_H)			Destination existence confirmation interval timer value (Setting time = setting value × 500 ms)	14_H (20)	○
19 (13_H)			Destination existence confirmation resending time	3_H (3)	○

Address Decimal (Hexadecimal)	Application	Name	Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
20 (14_H)	Initial process parameter setting area	Auto-open UDP port number	1388_H	×
21 to 29 (15_H to $1D_H$)		System area	-	-
30 ($1E_H$)	Initial process parameter setting area (For reinitialization)	TCP Maximum Segment Transmission setting area 0_H: Enable TCP Maximum Segment Size Option transmission 8000_H: Disable TCP Maximum Segment Size Option transmission Re-initialization makes the set value effective.	8000_H	×
31 ($1F_H$)		Communication condition setting (Ethernet Operation Setting) area • Communication data code setting (b1) 0: Binary Code 1: ASCII Code • TCP existence confirmation setting (b4) 0: Use the Ping 1: Use the KeepAlive • Send frame setting (b5) 0: Ethernet 1: IEEE 802.3 • Setting of write enable/disable at RUN time (b6) 0: Disable 1: Enable • 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) • Reinitialization specification (b15) 0: Reinitialization process complete (reset by the system) 1: Reinitialization process request (set by the user) Bits other than those described above are reserved for system use.	0_H	○

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
32 (20_H)	Communication parameter setting area	Connection usage setting area	Connection No.1 • Usage of fixed buffer (b0) 0: For sending, or communications using a fixed buffer are not performed 1: For receiving • Destination existence confirmation (b1) 0: No confirm 1: Confirm • Pairing open (b7) 0: Disable 1: Enable • Communication method (protocol) (b8) 0: TCP/IP 1: UDP/IP • Fixed buffer communication procedure (b10, b9) 00: Procedure exists 01: No procedure 10: Predefined protocol • Open system (b15, b14) 00: Active open or UDP/IP 10: Unpassive open 11: Fullpassive open Bits other than those described above are reserved for system use.	0_H	○
33 (21_H)			Connection No.2 (The bit configuration is the same as Connection No.1.)		
34 (22_H)			Connection No.3 (The bit configuration is the same as Connection No.1.)		
35 (23_H)			Connection No.4 (The bit configuration is the same as Connection No.1.)		
36 (24_H)			Connection No.5 (The bit configuration is the same as Connection No.1.)		
37 (25_H)			Connection No.6 (The bit configuration is the same as Connection No.1.)		
38 (26_H)			Connection No.7 (The bit configuration is the same as Connection No.1.)		
39 (27_H)			Connection No.8 (The bit configuration is the same as Connection No.1.)		

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
40 (28_H)	Communication parameter setting area	Communi-cation address setting area	Connection No.1	Own station port No.	0_H	○
41 and 42 (29_H and $2A_H$)				Destination IP address	0_H	○
43 ($2B_H$)				Destination Port No.	0_H	○
44 to 46 ($2C_H$ to $2E_H$)				Destination MAC address	$FFFFFFFFFFFF_H$	×
47 to 53 ($2F_H$ to 35_H)			Connection No.2 (The bit configuration is the same as Connection No.1.)			
54 to 60 (36_H to $3C_H$)			Connection No.3 (The bit configuration is the same as Connection No.1.)			
61 to 67 ($3D_H$ to 43_H)			Connection No.4 (The bit configuration is the same as Connection No.1.)			
68 to 74 (44_H to $4A_H$)			Connection No.5 (The bit configuration is the same as Connection No.1.)			
75 to 81 ($4B_H$ to 51_H)			Connection No.6 (The bit configuration is the same as Connection No.1.)			
82 to 88 (52_H to 58_H)			Connection No.7 (The bit configuration is the same as Connection No.1.)			
89 to 95 (59_H to $5F_H$)			Connection No.8 (The bit configuration is the same as Connection No.1.)			
96 to 102 (60_H to 66_H)		System area			-	-
103 and 104 (67_H and 68_H)	Communication status storage area	System area			-	-
105 (69_H)		Area for the initial process	Initial error code		0_H	×
106 and 107 ($6A_H$ and $6B_H$)			Own station IP address		0_H	×
108 to 110 ($6C_H$ to $6E_H$)			Own station MAC address		0_H	×
111 to 115 ($6F_H$ to 73_H)			System area		-	-
116 (74_H)			Auto-open UDP port number		0_H	×
117 (75_H)			System area		-	-
118 (76_H)			Station number (b0 to b7) Network number of the own station (b8 to b15)		0_H	×
119 (77_H)			Own station group number		0_H	×

3

3.5 Buffer Memory
3.5.2 List of buffer memory addresses

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
120 (78_H)	Communication status storage area	Connection information area	Connection No.1	Own station port No.	0_H	×
121 and 122 (79_H and $7A_H$)				Destination IP address	0_H	×
123 ($7B_H$)				Destination Port No.	0_H	×
124 ($7C_H$)				Open error code	0_H	×
125 ($7D_H$)				Fixed buffer sending error code	0_H	×
126 ($7E_H$)				Connection end code	0_H	×
127 ($7F_H$)				Fixed buffer communication time (Maximum value)	0_H	×
128 (80_H)				Fixed buffer communication time (Minimum value)	0_H	×
129 (81_H)				Fixed buffer communication time (Current value)	0_H	×
130 to 139 (82_H to $8B_H$)			Connection No.2 (The bit configuration is the same as Connection No.1.)			
140 to 149 ($8C_H$ to 95_H)			Connection No.3 (The bit configuration is the same as Connection No.1.)			
150 to 159 (96_H to $9F_H$)			Connection No.4 (The bit configuration is the same as Connection No.1.)			
160 to 169 ($A0_H$ to $A9_H$)			Connection No.5 (The bit configuration is the same as Connection No.1.)			
170 to 179 (AA_H to $B3_H$)			Connection No.6 (The bit configuration is the same as Connection No.1.)			
180 to 189 ($B4_H$ to BD_H)			Connection No.7 (The bit configuration is the same as Connection No.1.)			
190 to 199 (BE_H to $C7_H$)			Connection No.8 (The bit configuration is the same as Connection No.1.)			

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
200 ($C8_H$)	Communication status storage area	Area for module status	LED on/off status (Stores the on/off status of the LEDs on the front of the Ethernet module) • [INIT.]LED (b0) 0: OFF 1: ON (initial process completed) • [OPEN]LED (b1) 0: OFF 1: ON (connection open processing completed) • [ERR.]LED (b3) 0: OFF 1: ON (setting error) • [COM.ERR.]LED (b4) 0: OFF 1: ON (communication error) Bits other than those described above are reserved for system use.	0_H	×
201 ($C9_H$)			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 those described above are reserved for system use.	0_H	×
202 (CA_H)			Switch status • Operational mode setting (b1) 0: Online 1: Offline 2: Self-loopback test 3: Hardware test	0_H	○

<table>
<tr><th>Address
Decimal
(Hexadecimal)</th><th>Application</th><th colspan="3">Name</th><th>Initial value
Hexadecimal
(Decimal)</th><th>Programming tool setting applicability*2</th></tr>
<tr><td>203
(CB$_H$)</td><td rowspan="8">Communication status storage area</td><td>Module status area</td><td colspan="2">Status of settings with a programming tool
• Communication data code setting (b1)
0: Communications in a binary code
1: Communications in an ASCII code
• Initial/open method setting (b2)
0: No parameter setting (start up according to the sequence program)
1: Parameter setting (start up according to the parameters)
• TCP existence confirmation setting (b4)
0: Use the Ping
1: Use the KeepAlive
• Send frame setting (b5)
0: Ethernet
1: IEEE 802.3
• Setting of write enable/disable at RUN time (b6)
0: Disable
1: Enable
• 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 those described above are reserved for system use.</td><td>0$_H$</td><td>○</td></tr>
<tr><td>204
(CC$_H$)</td><td rowspan="7">Area for sending/receiving instructions</td><td colspan="2">System area</td><td>-</td><td>-</td></tr>
<tr><td>205
(CD$_H$)</td><td colspan="2">RECV instruction execution request</td><td>0$_H$</td><td>×</td></tr>
<tr><td>206
(CE$_H$)</td><td colspan="2">System area</td><td>-</td><td>-</td></tr>
<tr><td>207
(CF$_H$)</td><td rowspan="4">Data link instruction execution result</td><td>ZNRD instruction</td><td>0$_H$</td><td>×</td></tr>
<tr><td>208
(D0$_H$)</td><td>System area</td><td>-</td><td>-</td></tr>
<tr><td>209
(D1$_H$)</td><td>ZNWR instruction</td><td>0$_H$</td><td>×</td></tr>
<tr><td>210 to 223
(D2$_H$ to DF$_H$)</td><td>System area</td><td>-</td><td>-</td></tr>
<tr><td>224 to 226
(E0$_H$ to E2$_H$)</td><td rowspan="3">Error log area</td><td colspan="3">System area</td><td>-</td><td>-</td></tr>
<tr><td>227
(E3$_H$)</td><td colspan="3">Number of errors</td><td>0$_H$</td><td>×</td></tr>
<tr><td>228
(E4$_H$)</td><td colspan="3">Error log write pointer</td><td>0$_H$</td><td>×</td></tr>
</table>

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
229 ($E5_H$)	Error log area	Error log block area	Error log block 1	Error code/end code	0_H	×
230 ($E6_H$)				Subheader	0_H	×
231 ($E7_H$)				Command code	0_H	×
232 ($E8_H$)				Connection No.	0_H	×
233 ($E9_H$)				Own station port No.	0_H	×
234 and 235 (EA_H and EB_H)				Destination IP address	0_H	×
236 (EC_H)				Destination Port No.	0_H	×
237 (ED_H)				System area	-	-
238 to 246 (EE_H to $F6_H$)			Error log block 2 (The bit configuration is the same as Error log block 1.)			
247 to 255 ($F7_H$ to FF_H)			Error log block 3 (The bit configuration is the same as Error log block 1.)			
256 to 264 (100_H to 108_H)			Error log block 4 (The bit configuration is the same as Error log block 1.)			
265 to 273 (109_H to 111_H)			Error log block 5 (The bit configuration is the same as Error log block 1.)			
274 to 282 (112_H to $11A_H$)			Error log block 6 (The bit configuration is the same as Error log block 1.)			
283 to 291 ($11B_H$ to 123_H)			Error log block 7 (The bit configuration is the same as Error log block 1.)			
292 to 300 (124_H to $12C_H$)			Error log block 8 (The bit configuration is the same as Error log block 1.)			
301 to 309 ($12D_H$ to 135_H)			Error log block 9 (The bit configuration is the same as Error log block 1.)			
310 to 318 (136_H to $13E_H$)			Error log block 10 (The bit configuration is the same as Error log block 1.)			
319 to 327 ($13F_H$ to 147_H)			Error log block 11 (The bit configuration is the same as Error log block 1.)			
328 to 336 (148_H to 150_H)			Error log block 12 (The bit configuration is the same as Error log block 1.)			
337 to 345 (151_H to 159_H)			Error log block 13 (The bit configuration is the same as Error log block 1.)			
346 to 354 ($15A_H$ to 162_H)			Error log block 14 (The bit configuration is the same as Error log block 1.)			
355 to 363 (163_H to $16B_H$)			Error log block 15 (The bit configuration is the same as Error log block 1.)			
364 to 372 ($16C_H$ to 174_H)			Error log block 16 (The bit configuration is the same as Error log block 1.)			

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
373 to 375 (175_H to 177_H)	System area				-	-
376 and 377 (178_H and 179_H)	Error log area	Status for each protocol	IP	Received IP packet count	0_H	×
378 and 379 ($17A_H$ and $17B_H$)				Received IP packet count discarded due to sumcheck error	0_H	×
380 and 381 ($17C_H$ and $17D_H$)				Sent IP packet total count	0_H	×
382 to 397 ($17E_H$ to $18D_H$)				System area	-	-
398 and 399 ($18E_H$ and $18F_H$)				Simultaneous transmission error detection count	0_H	×
400 to 407 (190_H to 197_H)				System area	-	-
408 and 409 (198_H and 199_H)			ICMP	Received ICMP packet count	0_H	×
410 and 411 ($19A_H$ and $19B_H$)				Received ICMP packet count discarded due to sum check error	0_H	×
412 and 413 ($19C_H$ and $19D_H$)				Sent ICMP packet total count	0_H	×
414 and 415 ($19E_H$ and $19F_H$)				Echo request total count of received ICMP packets	0_H	×
416 and 417 ($1A0_H$ and $1A1_H$)				Echo reply total count of sent ICMP packets	0_H	×
418 and 419 ($1A2_H$ and $1A3_H$)				Echo request total count of sent ICMP packets	0_H	×
420 and 421 ($1A4_H$ and $1A5_H$)				Echo reply total count of received ICMP packets	0_H	×
422 to 439 ($1A6_H$ to $1B7_H$)				System area	-	-
440 and 441 ($1B8_H$ and $1B9_H$)			TCP	Received TCP packet count	0_H	×
442 and 443 ($1BA_H$ and $1BB_H$)				Received TCP packet count discarded due to sum check error	0_H	×
444 and 445 ($1BC_H$ and $1BD_H$)				Sent TCP packet total count	0_H	×
446 to 471 ($1BE_H$ to $1D7_H$)				System area	-	-

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
472 and 473 ($1D8_H$ and $1D9_H$)	Error log area	Status for each protocol	UDP	Received UDP packet count	0_H	×
474 and 475 ($1DA_H$ and $1DB_H$)				Received UDP packet count discarded due to sum check error	0_H	×
476 and 477 ($1DC_H$ and $1DD_H$)				Sent UDP packet total count	0_H	×
478 to 481 ($1DE_H$ to $1E1_H$)				System area	-	-
482 to 491 ($1E2_H$ to $1EB_H$)			System area		-	-
492 and 493 ($1EC_H$ and $1ED_H$)			Receiving error	Framing error count	0_H	×
494 and 495 ($1EE_H$ and $1EF_H$)				Overflow count	0_H	×
496 and 497 ($1F0_H$ and $1F1_H$)				crc error count	0_H	×
498 to 511 ($1F2_H$ to $1FF_H$)			System area		-	-
512 and 513 (200_H and 201_H)	Router relay parameter setting area	Sub-net mask			0_H	○
514 and 515 (202_H and 203_H)		Default router IP address			0_H	○
516 (204_H)		Number of registered routers			0_H	○
517 and 518 (205_H and 206_H)		Router 1	Sub-net address		0_H	○
519 and 520 (207_H and 208_H)			Router IP address		0_H	○
521 to 524 (209_H to $20C_H$)		Router 2 (The bit configuration is the same as Router 1.)				
525 to 528 ($20D_H$ to 210_H)		Router 3 (The bit configuration is the same as Router 1.)				
529 to 532 (211_H to 214_H)		Router 4 (The bit configuration is the same as Router 1.)				
533 to 536 (215_H to 218_H)		Router 5 (The bit configuration is the same as Router 1.)				
537 to 540 (219_H to $21C_H$)		Router 6 (The bit configuration is the same as Router 1.)				
541 to 544 ($21D_H$ to 220_H)		Router 7 (The bit configuration is the same as Router 1.)				
545 to 548 (221_H to 224_H)		Router 8 (The bit configuration is the same as Router 1.)				
549 (225_H)		System area			-	-

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
550 and 551 (226_H and 227_H)	Station No. <-> IP information setting area	System area		-	-
552 (228_H)		Number of conversion table data		0_H	○
553 and 554 (229_H and $22A_H$)		Conversion information No.1	Communication request destination/source stations network number and station number	0_H	○
555 and 556 ($22B_H$ and $22C_H$)			External station Ethernet module IP address	0_H	○
557 and 558 ($22D_H$ and $22E_H$)			System area	-	-
559 to 564 ($22F_H$ to 234_H)		Conversion information No.2 (The bit configuration is the same as Conversion information No.1.)			
to		to			
931 to 936 ($3A3_H$ to $3A8_H$)		Conversion information No.64 (The bit configuration is the same as Conversion information No.1.)			
937 and 938 ($3A9_H$ and $3AA_H$)		Net mask pattern for CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 routing		0_H	○
939 to 943 ($3AB_H$ to $3AF_H$)		System area		-	-
944 to 949 ($3B0_H$ to $3B5_H$)	FTP setting area	FTP login name		"QJ71E71"	○
950 to 953 ($3B6_H$ to $3B9_H$)		Password		"QJ71E71"	○
954 ($3BA_H$)		Command input monitoring timer		708_H	○
955 ($3BB_H$)		Programmable controller CPU monitoring timer		A_H	○
956 to 1663 ($3BC_H$ to $67F_H$)		System area		-	-
1664 (680_H)	Fixed buffer data area	Fixed buffer No.1	Data length	0_H	×
1665 to 2687 (681_H to $A7F_H$)			Fixed buffer data	0_H	×
2688 to 3711 ($A80_H$ to $E7F_H$)		Fixed buffer No.2 (The bit configuration is the same as Fixed buffer No.1.)			
3712 to 4735 ($E80_H$ to $127F_H$)		Fixed buffer No.3 (The bit configuration is the same as Fixed buffer No.1.)			
4736 to 5759 (1280_H to $167F_H$)		Fixed buffer No.4 (The bit configuration is the same as Fixed buffer No.1.)			
5760 to 6783 (1680_H to $1A7F_H$)		Fixed buffer No.5 (The bit configuration is the same as Fixed buffer No.1.)			
6784 to 7807 ($1A80_H$ to $1E7F_H$)		Fixed buffer No.6 (The bit configuration is the same as Fixed buffer No.1.)			
7808 to 8831 ($1E80_H$ to $227F_H$)		Fixed buffer No.7 (The bit configuration is the same as Fixed buffer No.1.)			
8832 to 9855 (2280_H to $267F_H$)		Fixed buffer No.8 (The bit configuration is the same as Fixed buffer No.1.)			

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
9856 to 16383 (2680$_H$ to 3FFF$_H$)	Shared area for random access buffers and e-mail buffers	Shared area for random access buffers and e-mail buffers		0$_H$	×
16384 to 18431 (4000$_H$ to 47FF$_H$)		System area		-	-
18432 to 20479 (4800$_H$ to 4FFF$_H$)	Send/receive area for predefined protocol support function	Send/receive area for predefined protocol support function		0$_H$	○
20480 (5000$_H$)	Connection status storage area	Connection status information area	Open completion signal 0: Open incomplete 1: Open completed • Connection No.1 (b0) • Connection No.2 (b1) to • Connection No.16 (b15)	0$_H$	×
20481 (5001$_H$)			System area	-	-
20482 (5002$_H$)			Open request signal 0: No open request 1: Open being requested • Connection No.1 (b0) • Connection No.2 (b1) to • Connection No.16 (b15)	0$_H$	×
20483 and 20484 (5003$_H$ and 5004$_H$)			System area	-	-
20485 (5005$_H$)		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.16 (b15)	0$_H$	×
20486 (5006$_H$)		Remote password status storage area	Remote password status 0: Unlock status/no remote password setting 1: Lock status • Connection No.1 (b0) • Connection No.2 (b1) to • Connection No.16 (b15)	0$_H$	×
20487 (5007$_H$)	System port information area	Remote password status storage area	Remote password status 0: Unlock status/no remote password setting 1: Lock status • Auto-open UDP port (b0) • MELSOFT application transmission port (UDP) (b1) • MELSOFT application transmission port (TCP) (b2) • FTP transmission port (b3)	0$_H$	×

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
20488 (5008_H)	System port information area	System port use prohibited designation area	System port use prohibited designation 0: Use allowed 1: Use prohibited • Auto-open UDP port (b0) • MELSOFT application transmission port (UDP) (b1) • MELSOFT application transmission port (TCP) (b2)		0_H	×
20489 to 20591 (5009_H to $506F_H$)			System area		-	-
20592 (5070_H)	Monitoring area	Remote password function monitoring area	Remote password mismatch notification accumulated count designation (For user open port) 0: No designation 1 or higher: Notification accumulated count		1_H	×
20593 (5071_H)			Remote password mismatch notification accumulated count designation (For auto-open UDP port, MELSOFT application transmission port (TCP/ UDP) and FTP transmission port) 0: No designation 1 or higher: Notification accumulated count		2_H	×
20594 (5072_H)			Connection No.1	Accumulated count of unlock process normal completion	0_H	×
20595 (5073_H)				Accumulated count of unlock process abnormal end	0_H	×
20596 (5074_H)				Accumulated count of lock process normal completion	0_H	×
20597 (5075_H)				Accumulated count of lock process abnormal end	0_H	×
20598 (5076_H)				Accumulated count of lock process based on close	0_H	×
20599 to 20603 (5077_H to $507B_H$)			Connection No.2 (The bit configuration is the same as Connection No.1.)			
20604 to 20608 ($507C_H$ to 5080_H)			Connection No.3 (The bit configuration is the same as Connection No.1.)			
20609 to 20613 (5081_H to 5085_H)			Connection No.4 (The bit configuration is the same as Connection No.1.)			
20614 to 20618 (5086_H to $508A_H$)			Connection No.5 (The bit configuration is the same as Connection No.1.)			
20619 to 20623 ($508B_H$ to $508F_H$)			Connection No.6 (The bit configuration is the same as Connection No.1.)			
20624 to 20628 (5090_H to 5094_H)			Connection No.7 (The bit configuration is the same as Connection No.1.)			

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
20629 to 20633 (5095$_H$ to 5099$_H$)	Monitoring area	Remote password function monitoring area	Connection No.8 (The bit configuration is the same as Connection No.1.)		
20634 to 20638 (509A$_H$ to 509E$_H$)			Connection No.9 (The bit configuration is the same as Connection No.1.)		
20639 to 20643 (509F$_H$ to 50A3$_H$)			Connection No.10 (The bit configuration is the same as Connection No.1.)		
20644 to 20648 (50A4$_H$ to 50A8$_H$)			Connection No.11 (The bit configuration is the same as Connection No.1.)		
20649 to 20653 (50A9$_H$ to 50AD$_H$)			Connection No.12 (The bit configuration is the same as Connection No.1.)		
20654 to 20658 (50AE$_H$ to 50B2$_H$)			Connection No.13 (The bit configuration is the same as Connection No.1.)		
20659 to 20663 (50B3$_H$ to 50B7$_H$)			Connection No.14 (The bit configuration is the same as Connection No.1.)		
20664 to 20668 (50B8$_H$ to 50BC$_H$)			Connection No.15 (The bit configuration is the same as Connection No.1.)		
20669 to 20673 (50BD$_H$ to 50C1$_H$)			Connection No.16 (The bit configuration is the same as Connection No.1.)		
20674 to 20678 (50C2$_H$ to 50C6$_H$)			Auto-open UDP port (The bit configuration is the same as Connection No.1.)		
20679 to 20683 (50C7$_H$ to 50CB$_H$)			MELSOFT application transmission port (UDP) (The bit configuration is the same as Connection No.1.)		
20684 to 20688 (50CC$_H$ to 50D0$_H$)			MELSOFT application transmission port (TCP) (The bit configuration is the same as Connection No.1.)		
20689 to 20693 (50D1$_H$ to 50D5$_H$)			FTP transmission port (The bit configuration is the same as Connection No.1.)		

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
20694 to 20736 ($50D6_H$ to 5100_H)	HTTP status storage area	System area		-	-
20737 (5101_H)		Error log pointer		0_H	×
20738 (5102_H)		Log counter (HTTP response code 100 to 199)		0_H	×
20739 (5103_H)		Log counter (HTTP response code 200 to 299)		0_H	×
20740 (5104_H)		Log counter (HTTP response code 300 to 399)		0_H	×
20741 (5105_H)		Log counter (HTTP response code 400 to 499)		0_H	×
20742 (5106_H)		Log counter (HTTP response code 500 to 599)		0_H	×
20743 (5107_H)		System area		-	-
20744 (5108_H)		Error log block 1	HTTP response code	0_H	×
20745 and 20746 (5109_H and $510A_H$)			Destination IP address	0_H	×
20747 to 20750 ($510B_H$ to $510E_H$)			Error time	0_H	×
20751 to 20757 ($510F_H$ to 5115_H)		Error log block 2 (The bit configuration is the same as Error log block 1.)			
20758 to 20764 (5116_H to $511C_H$)		Error log block 3 (The bit configuration is the same as Error log block 1.)			
20765 to 20771 ($511D_H$ to 5123_H)		Error log block 4 (The bit configuration is the same as Error log block 1.)			
20772 to 20778 (5124_H to $512A_H$)		Error log block 5 (The bit configuration is the same as Error log block 1.)			
20779 to 20785 ($512B_H$ to 5131_H)		Error log block 6 (The bit configuration is the same as Error log block 1.)			
20786 to 20792 (5132_H to 5138_H)		Error log block 7 (The bit configuration is the same as Error log block 1.)			
20793 to 20799 (5139_H to $513F_H$)		Error log block 8 (The bit configuration is the same as Error log block 1.)			
20800 to 20806 (5140_H to 5146_H)		Error log block 9 (The bit configuration is the same as Error log block 1.)			
20807 to 20813 (5147_H to $514D_H$)		Error log block 10 (The bit configuration is the same as Error log block 1.)			
20814 to 20820 ($514E_H$ to 5154_H)		Error log block 11 (The bit configuration is the same as Error log block 1.)			
20821 to 20827 (5155_H to $515B_H$)		Error log block 12 (The bit configuration is the same as Error log block 1.)			
20828 to 20834 ($515C_H$ to 5162_H)		Error log block 13 (The bit configuration is the same as Error log block 1.)			

Address Decimal (Hexadecimal)	Application	Name	Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
20835 to 20841 (5163$_H$ to 5169$_H$)	Status storage area	Error log block 14 (The bit configuration is the same as Error log block 1.)		
20842 to 20848 (516A$_H$ to 5170$_H$)		Error log block 15 (The bit configuration is the same as Error log block 1.)		
20849 to 20855 (5171$_H$ to 5177$_H$)		Error log block 16 (The bit configuration is the same as Error log block 1.)		
20856 to 20991 (5178$_H$ to 51FF$_H$)	System area		-	-
20992 (5200$_H$)	"Issue system switching request at disconnection detection" status storage area	"Issue system switch in cable disconnection timeout" 0: Not set 1: Set	1$_H$	○
20993 (5201$_H$)		Disconnection detection monitoring time Set time = set value × 500ms (setting range: 0 to 60)	4$_H$	○
20994 (5202$_H$)		System area	-	-
20995 (5203$_H$)		Disconnection detection count	0$_H$	×
20996 to 21007 (5204$_H$ to 520F$_H$)	System area		-	-
21008 (5210$_H$)	"System switching settings when communication error occurs" status storage area	"System switching settings when communication error occurs" (user connection) 0: Not set 1: Set • Connection No.1 (b0) • Connection No.2 (b1) to • Connection No.16 (b15)	0$_H$	○
21009 (5211$_H$)		"System switching settings when communication error occurs" (system connection) 0: Not set 1: Set • Auto-open UDP port (b0) • MELSOFT application transmission port (UDP) (b1) • MELSOFT application transmission port (TCP) (b2) • FTP transmission port (b3) • HTTP port (b4)	0$_H$	○
21010 to 21055 (5212$_H$ to 523F$_H$)	System area		-	-
21056 (5240$_H$)[*1]	Receive buffer status storage area	Receive buffer full detection signal 0: Receive buffer not full 1: Receive buffer full	0$_H$	×
21057 to 21119 (5241$_H$ to 527F$_H$)	System area		-	-

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
21120 (5280_H)	IP address status storage area	Same IP address detection flag 0: Unique IP addresses 1: Same IP address for multiple stations		0_H	×
21121 to 21123 (5281_H to 5283_H)		The MAC address of the station that has been already connected to the network (The address is stored in the station with the same IP address.)		$FFFFFFFFFFFF_H$	×
21124 to 21126 (5284_H to 5286_H)		The MAC address of the station with the IP address used for another station (The address is stored in the station that has been already connected to the network.)		$FFFFFFFFFFFF_H$	×
21127 to 21279 (5287_H to $531F_H$)	System area			-	-
21280 to 21283 (5320_H to 5323_H)	Protocol setting data check area	System area		-	-
21284 (5324_H)		Protocol setting data error information	Protocol No.	0_H	×
21285 (5325_H)			Setting type	0_H	×
21286 (5326_H)			Packet No.	0_H	×
21287 (5327_H)			Element No.	0_H	×
21288 (5328_H)		Number of registered protocols		0_H	×
21289 to 21295 (5329_H to $532F_H$)		System area		-	-
21296 to 21311 (5330_H to $533F_H$)		Protocol registration		0_H	×
21312 to 21695 (5340_H to $54BF_H$)	System area			-	-

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
21696 (54C0$_H$)	Predefined protocol support function execution status check area	Connection No.1	Protocol execution status	0$_H$	×
21697 (54C1$_H$)			System area	-	-
21698 to 21713 (54C2$_H$ to 54D1$_H$)			Received data verification result (receive packet No.1 to 16)	0$_H$	×
21714 (54D2$_H$)			Number of protocol executions	0$_H$	×
21715 (54D3$_H$)			Protocol cancellation specification	0$_H$	○
21716 to 21735 (54D4$_H$ to 54E7$_H$)		Connection No.2 (The bit configuration is the same as Connection No.1.)			
21736 to 21755 (54E8$_H$ to 54FB$_H$)		Connection No.3 (The bit configuration is the same as Connection No.1.)			
21756 to 21775 (54FC$_H$ to 550F$_H$)		Connection No.4 (The bit configuration is the same as Connection No.1.)			
21776 to 21795 (5510$_H$ to 5523$_H$)		Connection No.5 (The bit configuration is the same as Connection No.1.)			
21796 to 21815 (5524$_H$ to 5537$_H$)		Connection No.6 (The bit configuration is the same as Connection No.1.)			
21816 to 21835 (5538$_H$ to 554B$_H$)		Connection No.7 (The bit configuration is the same as Connection No.1.)			
21836 to 21855 (554C$_H$ to 555F$_H$)		Connection No.8 (The bit configuration is the same as Connection No.1.)			
21856 to 21875 (5560$_H$ to 5573$_H$)	Predefined protocol support function execution status check area	Connection No.9 (The bit configuration is the same as Connection No.1.)			
21876 to 21895 (5574$_H$ to 5587$_H$)		Connection No.10 (The bit configuration is the same as Connection No.1.)			
21896 to 21915 (5588$_H$ to 559B$_H$)		Connection No.11 (The bit configuration is the same as Connection No.1.)			
21916 to 21935 (559C$_H$ to 55AF$_H$)		Connection No.12 (The bit configuration is the same as Connection No.1.)			
21936 to 21955 (55B0$_H$ to 55C3$_H$)		Connection No.13 (The bit configuration is the same as Connection No.1.)			
21956 to 21975 (55C4$_H$ to 55D7$_H$)		Connection No.14 (The bit configuration is the same as Connection No.1.)			
21976 to 21995 (55D8$_H$ to 55EB$_H$)		Connection No.15 (The bit configuration is the same as Connection No.1.)			
21996 to 22015 (55EC$_H$ to 55FF$_H$)		Connection No.16 (The bit configuration is the same as Connection No.1.)			
22016 to 22271 (5600$_H$ to 56FF$_H$)	System area			-	-

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
22272 (5700_H)	IP filter function area	IP filter settings	Use of IP filter settings • 0: Do not use • 1: Use		0_H	×
22273 (5701_H)			IP filter function type setting • 0: Allow • 1: Deny		0_H	×
22274 to 22275 (5702_H to 5703_H)			IP address setting 1	Start IP address	0_H	×
22276 to 22277 (5704_H to 5705_H)				End IP address	0_H	×
22278 to 22281 (5706_H to 5709_H)			IP address setting 2 (The bit configuration is the same as IP address setting 1.)		0_H	×
22282 to 22285 ($570A_H$ to $570D_H$)			IP address setting 3 (The bit configuration is the same as IP address setting 1.)		0_H	×
22286 to 22289 ($570E_H$ to 5711_H)			IP address setting 4 (The bit configuration is the same as IP address setting 1.)		0_H	×
22290 to 22293 (5712_H to 5715_H)			IP address setting 5 (The bit configuration is the same as IP address setting 1.)		0_H	×
22294 to 22297 (5716_H to 5719_H)			IP address setting 6 (The bit configuration is the same as IP address setting 1.)		0_H	×
22298 to 22301 ($571A_H$ to $571D_H$)			IP address setting 7 (The bit configuration is the same as IP address setting 1.)		0_H	×
22302 to 22305 ($571E_H$ to 5721_H)			IP address setting 8 (The bit configuration is the same as IP address setting 1.)		0_H	×
22306 to 22307 (5722_H to 5723_H)		IP filter monitoring area	Number of access denied by the IP filter function		0_H	×
22308 to 22309 (5724_H to 5725_H)			IP address denied by the IP filter function		0_H	×
22310 to 22559 (5726_H to $581F_H$)	System area				-	-

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
22560 (5820_H)	Communication status storage area	Connection information area	Connection No.9	Own station port No.	0_H	×
22561 and 22562 (5821_H and 5822_H)				Destination IP address	0_H	×
22563 (5823_H)				Destination Port No.	0_H	×
222564 (5824_H)				Open error code	0_H	×
22565 (5825_H)				Fixed buffer sending error code	0_H	×
22566 (5826_H)				Connection end code	0_H	×
22567 (5827_H)				Fixed buffer communication time (Maximum value)	0_H	×
22568 (5828_H)				Fixed buffer communication time (Minimum value)	0_H	×
22569 (5829_H)				Fixed buffer communication time (Current value)	0_H	×
22570 to 22579 ($582A_H$ to 5833_H)			Connection No.10 (The bit configuration is the same as Connection No.9.)			
22580 to 22589 (5834_H to $583D_H$)			Connection No.11 (The bit configuration is the same as Connection No.9.)			
22590 to 22599 ($583E_H$ to 5847_H)			Connection No.12 (The bit configuration is the same as Connection No.9.)			
22600 to 22609 (5848_H to 5851_H)			Connection No.13 (The bit configuration is the same as Connection No.9.)			
22610 to 22619 (5852_H to $585B_H$)			Connection No.14 (The bit configuration is the same as Connection No.9.)			
22620 to 22629 ($585C_H$ to 5865_H)			Connection No.15 (The bit configuration is the same as Connection No.9.)			
22630 to 22639 (5866_H to $586F_H$)			Connection No.16 (The bit configuration is the same as Connection No.9.)			

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
22640 (5870_H)	E-mail status storage area	Receive	Number of mails remaining on the server		0_H	×
22641 (5871_H)			Dedicated instruction normal completion count		0_H	×
22642 (5872_H)			Dedicated instruction abnormal end count		0_H	×
22643 (5873_H)			Normal receiving count		0_H	×
22644 (5874_H)			Attached file receiving count		0_H	×
22645 (5875_H)			Server inquiry count		0_H	×
22646 (5876_H)			Server communication error count		0_H	×
22647 (5877_H)			Error log write count		0_H	×
22648 (5878_H)			Receiving error log write pointer		0_H	×
22649 (5879_H)			Error log block 1	Error code	0_H	×
22650 ($587A_H$)				Command code	0_H	×
22651 to 22658 ($587B_H$ to 5882_H)				From	0_H	×
22659 to 22662 (5883_H to 5886_H)				Date	0_H	×
22663 to 22692 (5887_H to $58A4_H$)				Subject	0_H	×

Address Decimal (Hexadecimal)	Application	Name		Initial value Hexadecimal (Decimal)	Programming tool setting applicability[*2]
22693 to 22736 ($58A5_H$ to $58D0_H$)	E-mail status storage area	Receive	Error log block 2 (The bit configuration is the same as Error log block 1.)		
22737 to 22780 ($58D1_H$ to $58FC_H$)			Error log block 3 (The bit configuration is the same as Error log block 1.)		
22781 to 22824 ($58FD_H$ to 5928_H)			Error log block 4 (The bit configuration is the same as Error log block 1.)		
22825 to 22868 (5929_H to 5954_H)			Error log block 5 (The bit configuration is the same as Error log block 1.)		
22869 to 22912 (5955_H to 5980_H)			Error log block 6 (The bit configuration is the same as Error log block 1.)		
22913 to 22956 (5981_H to $59AC_H$)			Error log block 7 (The bit configuration is the same as Error log block 1.)		
22957 to 23000 ($59AD_H$ to $59D8_H$)			Error log block 8 (The bit configuration is the same as Error log block 1.)		
23001 to 23044 ($59D9_H$ to $5A04_H$)			Error log block 9 (The bit configuration is the same as Error log block 1.)		
23045 to 23088 ($5A05_H$ to $5A30_H$)			Error log block 10 (The bit configuration is the same as Error log block 1.)		
23089 to 23132 ($5A31_H$ to $5A5C_H$)			Error log block 11 (The bit configuration is the same as Error log block 1.)		
23133 to 23176 ($5A5D_H$ to $5A88_H$)			Error log block 12 (The bit configuration is the same as Error log block 1.)		
23177 to 23220 ($5A89_H$ to $5AB4_H$)			Error log block 13 (The bit configuration is the same as Error log block 1.)		
23221 to 23264 ($5AB5_H$ to $5AE0_H$)			Error log block 14 (The bit configuration is the same as Error log block 1.)		
23265 to 23308 ($5AE1_H$ to $5B0C_H$)			Error log block 15 (The bit configuration is the same as Error log block 1.)		
23309 to 23352 ($5B0D_H$ to $5B38_H$)			Error log block 16 (The bit configuration is the same as Error log block 1.)		
23353 ($5B39_H$)		Send	Dedicated instruction normal completion count	0_H	×
23354 ($5B3A_H$)			Dedicated instruction abnormal end count	0_H	×
23355 ($5B3B_H$)			Number of mails normally completed	0_H	×
23356 ($5B3C_H$)			Attached file sending count	0_H	×
23357 ($5B3D_H$)			Sending to the server count	0_H	×
23358 ($5B3E_H$)			Number of mails abnormally completed	0_H	×
23359 ($5B3F_H$)			Error log write count	0_H	×
23360 ($5B40_H$)			Sending error log write pointer	0_H	×

Address Decimal (Hexadecimal)	Application	Name			Initial value Hexadecimal (Decimal)	Programming tool setting applicability*2
23361 ($5B41_H$)	E-mail status storage area	Send	Error log block 1	Error code	0_H	×
23362 ($5B42_H$)				Command code	0_H	×
23363 to 23370 ($5B43_H$ to $5B4A_H$)				To	0_H	×
23371 to 23374 ($5B4B_H$ to $5B4E_H$)				Date	0_H	×
23375 to 23404 ($5B4F_H$ to $5B6C_H$)				Subject	0_H	×
23405 to 23448 ($5B6D_H$ to $5B98_H$)			Error log block 2 (The bit configuration is the same as Error log block 1.)			
23449 to 23492 ($5B99_H$ to $5BC4_H$)			Error log block 3 (The bit configuration is the same as Error log block 1.)			
23493 to 23536 ($5BC5_H$ to $5BF0_H$)			Error log block 4 (The bit configuration is the same as Error log block 1.)			
23537 to 23580 ($5BF1_H$ to $5C1C_H$)			Error log block 5 (The bit configuration is the same as Error log block 1.)			
23581 to 23624 ($5C1D_H$ to $5C48_H$)			Error log block 6 (The bit configuration is the same as Error log block 1.)			
23625 to 23668 ($5C49_H$ to $5C74_H$)			Error log block 7 (The bit configuration is the same as Error log block 1.)			
23669 to 23712 ($5C75_H$ to $5CA0_H$)			Error log block 8 (The bit configuration is the same as Error log block 1.)			
23713 to 24575 ($5CA1_H$ to $5FFF_H$)		System area			-	-
24576 (6000_H)	Fixed buffer data area	Fixed buffer No.9	Data length		0_H	×
24577 to 25599 (6001_H to $63FF_H$)			Fixed buffer data		0_H	×
25600 to 26623 (6400_H to $67FF_H$)		Fixed buffer No.10 (The bit configuration is the same as Fixed buffer No.9.)				
26624 to 27647 (6800_H to $6BFF_H$)		Fixed buffer No.11 (The bit configuration is the same as Fixed buffer No.9.)				
27648 to 28671 ($6C00_H$ to $6FFF_H$)		Fixed buffer No.12 (The bit configuration is the same as Fixed buffer No.9.)				
28672 to 29695 (7000_H to $73FF_H$)		Fixed buffer No.13 (The bit configuration is the same as Fixed buffer No.9.)				
29696 to 30719 (7400_H to $77FF_H$)		Fixed buffer No.14 (The bit configuration is the same as Fixed buffer No.9.)				
30720 to 31743 (7800_H to $7BFF_H$)		Fixed buffer No.15 (The bit configuration is the same as Fixed buffer No.9.)				
31744 to 32767 ($7C00_H$ to $7FFF_H$)		Fixed buffer No.16 (The bit configuration is the same as Fixed buffer No.9.)				

*1 Available only for the QJ71E71-100.
Availability depends on the QJ71E71-100 version. (Page 351, Appendix 3)

*2 "○" means area which is reflected values of network parameters set by the programming tool and "×" means area which is not reflected them.

CHAPTER 4 PROCEDURES BEFORE OPERATION

4

This chapter describes how to connect an E71 to Ethernet. For how to communicate with connected devices after the E71 is connected, refer to the communication procedure. (Page 82, CHAPTER 7)

Checkbox

☐ **System planning**

Plan the system configuration of the E71.

Page 28, CHAPTER 3, Page 63, CHAPTER 5

▼

☐ **Mounting**

Mount the E71 on the base unit.

Page 26, CHAPTER 2, Page 76, Section 6.1

▼

☐ **Wiring**

Connect an Ethernet cable to the E71.

Page 72, Section 5.2, Page 77, Section 6.2

▼

☐ **Setting parameters**

Set the following parameters, which are required to start up the E71, on a programming tool.
- Basic setting
- Ethernet operation setting

Page 84, Section 7.1.2, Page 85, Section 7.1.3

▼

☐ **Restarting the CPU module**

Reflect the parameter settings by powering off and on the CPU module or resetting the CPU module.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

▼

☐ **Network diagnostics**

Check that the cables are connected properly or that communications are performed normally with the set parameters. (PING test, loopback test)

Page 364, Appendix 5

▼

Continued to COMMUNICATION PROCEDURE

Memo

CHAPTER 5 SYSTEM CONFIGURATION

This chapter describes the system configuration of an E71.

5.1 Configuration of an E71-mounted System

This section describes the configuration of the system where an E71 is mounted.

5.1.1 Applicable modules and base units, and the number of connectable modules

(1) Connecting an E71 to a CPU module

For the CPU modules and base units that can be used for an E71 and the number of connectable modules, refer to the following.

User's Manual (Hardware Design, Maintenance and Inspection) for the CPU module used.

(2) Connecting an E71 in a MELSECNET/H remote I/O station

For the base units that can be used for an E71 and the number of connectable modules, refer to the following.

Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

5.1.2 For use with a Basic model QCPU or safety CPU

When an E71 is mounted to a Basic model QCPU or a safety CPU, the available functions are restricted as follows.

○: Available, ×: Not available

Function		Availability	
		Basic model QCPU	Safety CPU
Connection with MELSOFT products and a GOT		○	○
MC protocol communications		○	○
Communications using SLMP		○	○
Data communications using the predefined protocol		○	×
Communications using a fixed buffer		○	○*2
	Receive process using an interrupt program	○*1	×
	Pairing open	○	○
	Broadcast communications	○	○
Communications using a random access buffer		○	×
IP filter function		○	○
Remote password		○*1	○
Router relay function		○	○
Communications using an auto-open UDP port		○	○
Connected device alive check function		○	○
E-mail function		○*1	×
CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications		○	×
Communications using data link instructions		○*3	○*3
	Receive process using an interrupt program	○*1	×
File transfer (FTP server function)		○	×
Web function		○	×

*1 This function can be used in a Basic model QCPU with the function version B or later. For the version of a programming tool, check the corresponding software version. (Page 430, Appendix 11.1)

*2 The connection numbers 1 to 8 only can be specified. If the specified value is out of range, an "OPERATION ERROR" (error code: 4101) occurs.

*3 When the target station of the SREAD/SWRITE instructions is a Basic model QCPU or safety CPU, the read notification device to the target station set to the argument (D3) is ignored. The operation of the SREAD/SWRITE instructions is the same as that of the READ/WRITE instructions. For the SREAD/SWRITE instructions, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Application)

5.1.3 For use in a multiple CPU system

When using an E71 in a multiple CPU system, refer to the following.

QCPU User's Manual (Multiple CPU System)

(1) Precautions

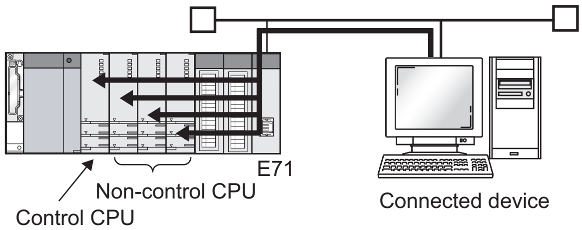
Note the following precautions for when using an E71 in a multiple CPU system.

(a) Writing network parameters

Configure network parameters only on the control CPU of an E71.

(b) Accessing an E71 non-control CPU from a connected device

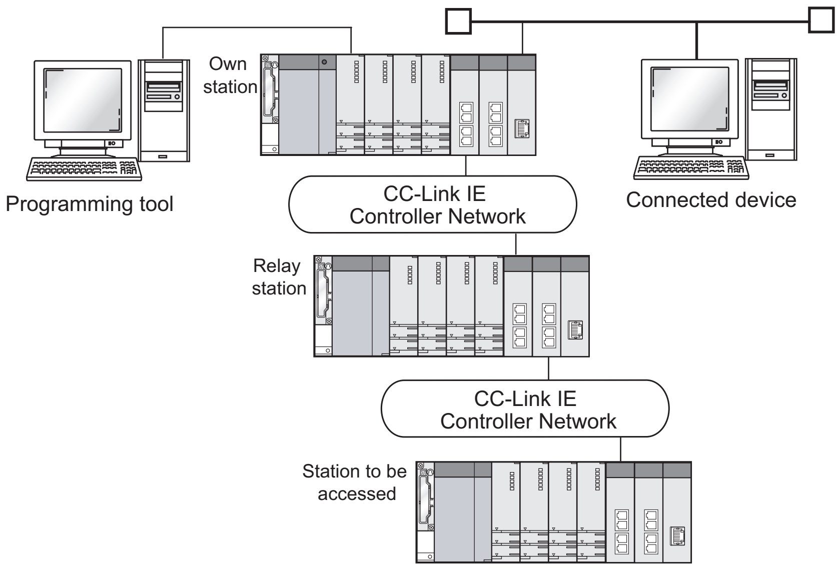
When accessing a non-control CPU in a multiple CPU system, use an E71 with the function version B or later.



(c) Accessing other stations

A control CPU and non-control CPU in a station to be accessed are accessible even when a relay station or station to be accessed is configured in a multiple CPU system. To access a non-control CPU, use modules of the function version B or later in the own station and all relay stations and for all relay modules in stations to be accessed and CPU modules.

Ex. When the relay modules are CC-Link IE Controller Network modules



5.1.4 For use in a redundant system

When using an E71 in a redundant system, refer to the following.

QnPRHCPU User's Manual (Redundant System)

(1) Mounting an E71 on the main base unit of a redundant system

This section provides information on mounting an E71 on the main base unit of a redundant system.

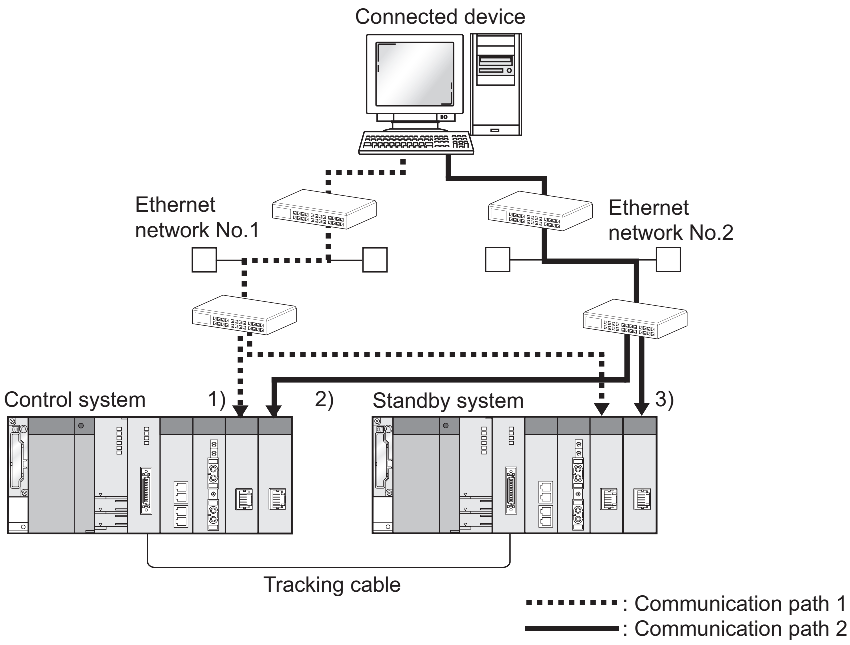
(a) Basic system configuration

Access can be performed from a connected device to the control system and standby system of a redundant system.

(b) System configuration with a communication path backed up

Access can be performed from a connected device to the control system and standby system of the redundant system in the communication path 1 or communication path 2.

Ex. If a communication error occurs during access to the control system through the communication path 1 (1) in the figure below), the control system can be accessed through the communication path 2 (2) in the figure below). If a communication error also occurs in the communication path 2, the system switches between the control system and the standby system, enabling communications with the new system (3) in the figure below).



(c) Restrictions on the use of the functions

When an E71 is mounted on a main base unit, the available functions are restricted as follows.

○: Available, ×: Not available

Function	Availability
Connection with MELSOFT products and a GOT	○
MC protocol communications	○*1
Communications using SLMP	○
Data communications using the predefined protocol	○
Communications using a fixed buffer	○*1
Pairing open	○
Broadcast communications	○*1
Communications using a random access buffer	○*1
IP filter function	○
Remote password	○
Router relay function	○
Communications using an auto-open UDP port	○
Connected device alive check function	○
E-mail function	○*1
CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications	○*1
Communications using data link instructions	○*1
File transfer (FTP server function)	○*1
Web function	○

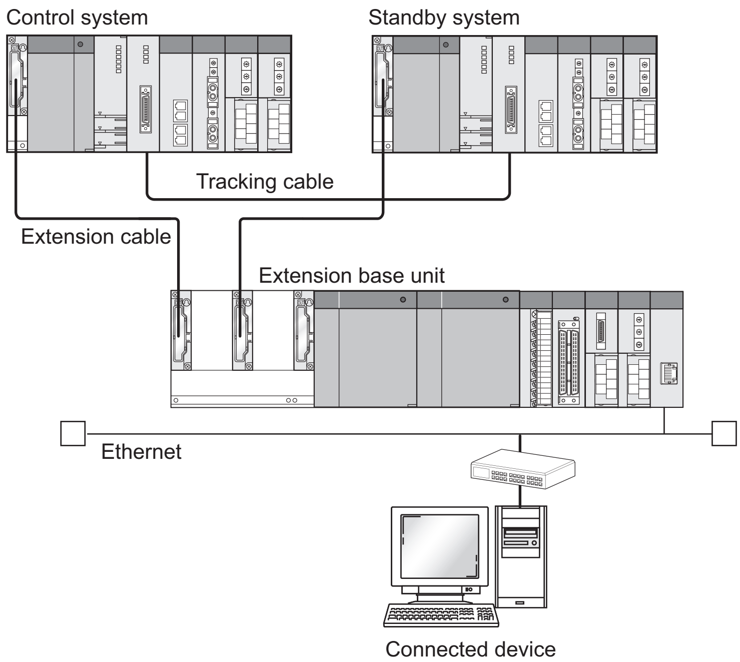
*1 Available with restrictions. For the restrictions, refer to the functions that can be used in a redundant system. (☞ Page 224, Section 14.7.4)

(2) Mounting an E71 on the extension base unit of a redundant system

This section provides information on mounting an E71 on the extension base unit of a redundant system.

(a) System configuration

The following shows the system configuration.



(b) Restrictions on the use of the functions

Restrictions are the same as those of when an E71 is mounted on a main base unit, except for the following restrictions.

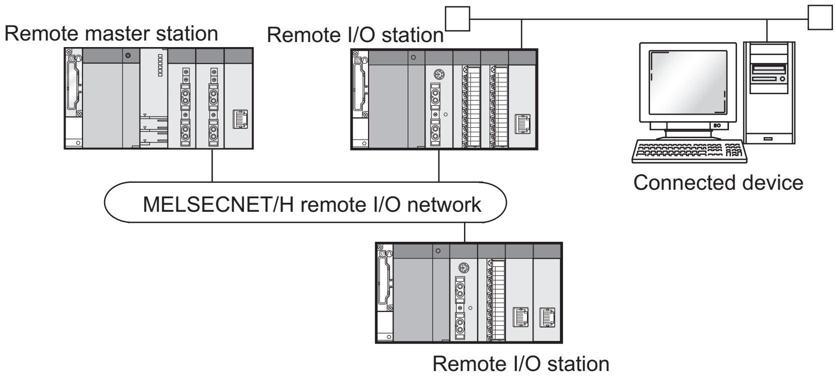
- Because an E71 does not issue a system switching request to the control system CPU module, the system switching does not continue communications. To continue communications even when the E71 experiences a communication error or when the cable is disconnected, mount the E71 on a main base unit.
- Dedicated instructions cannot be used. To use dedicated instructions, mount the E71 on a main base unit.
- The following restrictions apply to communications via modules mounted on an extension base unit.
    - An access destination that can be specified (a control system CPU module or standby system CPU module or a system A CPU module or system B CPU module) varies according to the MC protocol dedicated command.
    - If systems are switched during MC protocol communications or communications using dedicated instructions from other stations, a communication timeout may occur.

5.1.5 For use in a MELSECNET/H remote I/O station

This section provides information on using an E71 in a MELSECNET/H remote I/O station.

(1) System configuration

The following shows the system configuration.



(2) Parameter settings

Configure the network parameters as follows.

(a) Ethernet operation setting

Select "Always wait for OPEN" for the initial timing setting.

(b) Open setting

Select "Unpassive" or "Fullpassive" in the open system for the TCP/IP communication. An auto-open UDP port can be also used for UDP/IP communication.

(3) Open/close processes

Perform the open/close processes on the connected device side.

(4) Restrictions on the use of the functions

The available functions are restricted as follows.

○: Available, ×: Not available

Function		Availability
Initial process	Program setting	×
	Network parameter setting	○
Open/close processes	Program setting	×
	Network parameter setting	○
Connection with MELSOFT products and a GOT		○
MC protocol communications		○ (Refer to clause (5).)
Communications using SLMP		○
Data communications using the predefined protocol		×
Communications using a fixed buffer		×
Pairing open		×
Broadcast communications		×
Communications using a random access buffer		○
IP filter function		○
Remote password		○*1
Router relay function		○
Communications using an auto-open UDP port		○
Connected device alive check function		○
E-mail function		×
CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications		○
Communications using data link instructions		× (Can be relayed)
File transfer (FTP server function)		×
Web function		×

*1 This function can be used in a MELSECNET/H remote I/O station with the function version D or later. For the version of a programming tool, check the corresponding software version. (Page 430, Appendix 11.1)

(5) Access using MC protocol communications

Access to a MELSECNET/H remote I/O station using the MC protocol and access to other stations via a MELSECNET/H remote I/O station are described below.

(a) Compatible frames

Use a QnA-compatible 3E frame or 4E frame for communications. (An A-compatible 1E frame cannot be used.)

(b) Available functions

The following functions can be used in MELSECNET/H remote I/O stations.

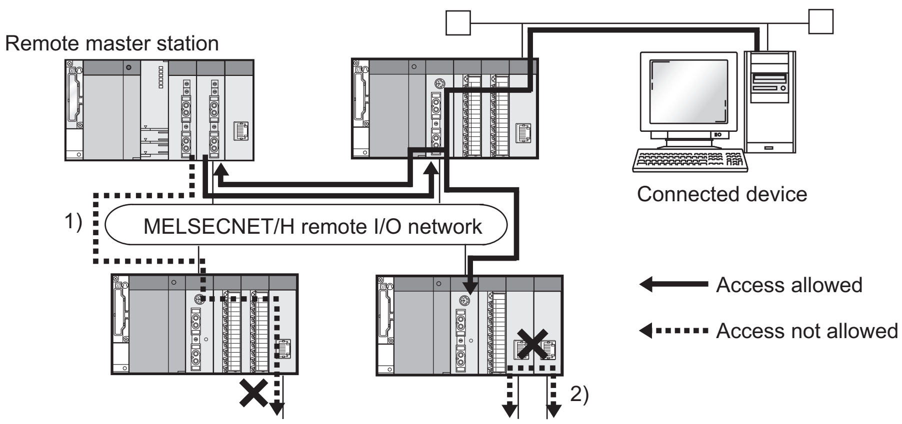
- Reading/writing of device memory*1 *2
- Reading/writing of buffer memory*2
- Reading/writing of intelligent function module buffer memory

*1 For accessible MELSECNET/H remote I/O station devices, refer to the following.
MELSEC Communication Protocol Reference Manual
*2 This function cannot be used in QnA/A series-compatible MELSECNET/10 remote I/O stations.

(c) Access to other stations through MELSECNET/H remote I/O stations

A MELSECNET/H remote master station and MELSECNET/H remote I/O stations can be accessed.

Ex. The MELSECNET/H remote master station and MELSECNET/H remote I/O stations can be accessed from a connected device.



The following methods cannot be used to access other stations through MELSECNET/H remote I/O stations.

- Access to other stations from a MELSECNET/H remote master station via the E71 connected in a MELSECNET/H remote I/O station (1) in the figure above)
- Access to other stations through two E71 modules (2) in the figure above)

5.2 Network Components

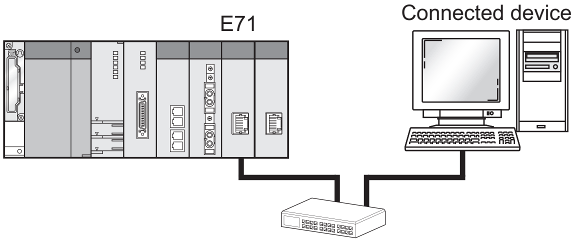
This section describes the devices used for Ethernet.

5.2.1 Configuration devices used for 100BASE-TX/10BASE-T connection

A QJ71E71-100 is used for 100BASE-TX and 10BASE-T connections. An E71 determines whether to use 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 automatic negotiation function, set the half-duplex mode on the hub side.

(1) 100BASE-TX connection

To configure a system for 100BASE-TX connection, use devices that meet the IEEE 802.3 100 BASE-TX standards.



Configuration device		Description
Shielded twisted pair cable (STP)	Straight cable	A Category 5 or higher straight cable is usable.
	Cross cable	System operation using connection with a cross cable is not guaranteed. Note, however, that a Category 5 or 5e cross cable can be used for data communications with an E71 (between QJ71E71-100 modules) or connection with a GOT.
Connector	RJ45 connector	-
Hub	100Mbps hub	Up to two levels of cascade connection can be configured.*1

*1 This applies when a repeater hub is used. For the number of levels that can be constructed when a switching hub is used, consult with the manufacturer of the switching hub used.

(a) Precautions for using a hub

The QJ71E71-100 does not support the flow control of the IEEE802.3x.
Therefore, for the connection using an IEEE802.3x-compatible hub, the data sent from the QJ71E71-100 may be lost and the execution of the dedicated instruction may be timed out if the Ethernet line is heavily loaded.
If the phenomenon mentioned above occurs, reduce the load on the line taking mesures such as adding hubs.

Point

During high-speed communications (100Mbps) using 100BASE-TX connection, a communication error may occur due to high-frequency noise from devices other than a programmable controller in a given installation environment. The following describes countermeasures to be taken on the E71 side to avoid high-frequency noise influence.

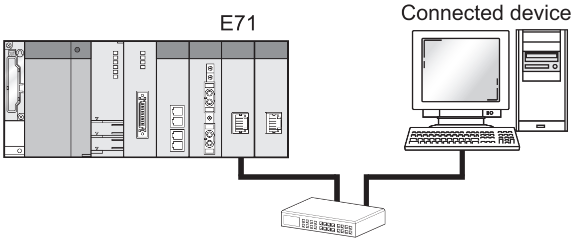
- Wiring connection
  - Do not bundle the cable with the main circuit or power cable or do not place it near those lines.
  - Put the cable in a duct.
- Communication method
  - Use TCP/IP for data communications with connected devices.
  - Increase the number of retries of communications if needed.
- 10Mbps communications
  - Use a 10Mbps hub for the E71 and set the data transmission speed to 10Mbps.
  - Change the transmission speed of the E71 to 10Mbps using the UINI instruction. (Page 277, Section 15.13)

Remark

Consult a network specialist for required devices.

(2) 10BASE-T connection

To configure a system for 10BASE-T connection, use devices that meet the IEEE 802.3 10BASE-T standards.



Configuration device		Description
Unshielded twisted pair cable (UTP) or shielded twisted pair cable (STP)	Straight cable	A Category 3 to 5e straight cable is usable.
	Cross cable	System operation using connection with a cross cable is not guaranteed. Note, however, that a Category 3 to 5e cross cable can be used for data communications with an E71 (between QJ71E71-100 modules) or connection with a GOT.
Connector	RJ45 connector	-
Hub	10Mbps hub	Up to four levels of cascade connection can be configured.*1

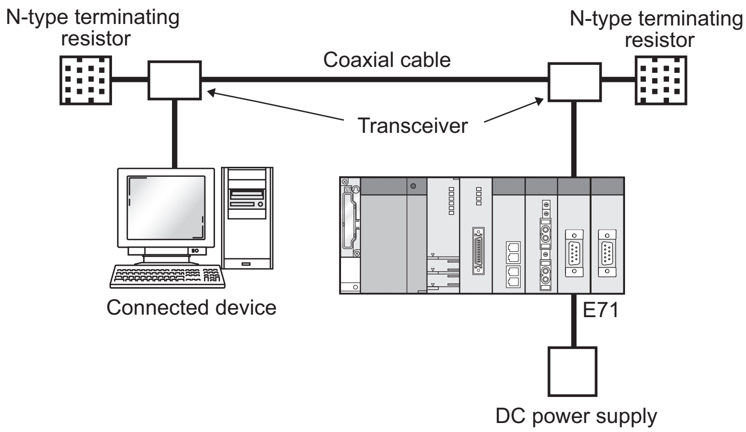
*1 This applies when a repeater hub is used. For the number of levels that can be constructed when a switching hub is used, consult with the manufacturer of the switching hub used.

Remark

Consult a network specialist for required devices.

5.2.2 Configuration devices used for 10BASE5 connection

A QJ71E71-B5 is used for 10BASE5 connection. Use devices that meet the IEEE 802.3 10BASE5 standards.



Configuration device	Description
10BASE5 coaxial cable	The Ethernet standards need to be satisfied.
N-type terminating resistor	
AUI cable (transceiver cable)	
Transceiver	The Ethernet standards need to be satisfied. The SQE TEST (Signal Quality Error TEST) or the heartbeat needs to be supported.
DC power supply (power supply for a transceiver)	The specifications of the transceiver and the AUI cable need to be satisfied.

Point

- The transceiver electrical characteristics are as follows. The power supply for the transceiver needs to be 13.28VDC to 15.75VDC.
  - Input terminal voltage: $12VDC^{-6\%}$ to $15VDC^{+5\%}$
  - AUI cable direct resistance: 40Ω/km or less, maximum length 50m (164ft.)
  - Maximum current consumption: 500mA or less

  The voltage drop (V) of the power supply for the transceiver is calculated as follows.
  Voltage drop (V) = AUI cable direct current resistance (Ω/m) × AUI cable length (m) × 2 (both directions) × transceiver current consumption (A)

  Ex. 2.0(VDC) = 0.04(Ω/m) × 50(m) × 2 × 0.5(A)
  In this case, the recommended power supply for the transceiver is more than 13.28VDC.
  13.28(VDC) = $12VDC^{-6\%}$ × (11.28VDC) + 2.0(VDC)

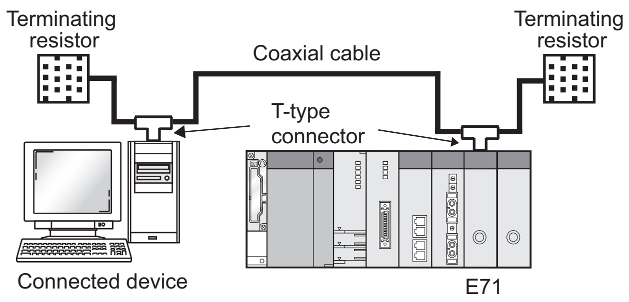
- A communication error may occur due to high-frequency noise from devices other than a programmable controller in a given installation environment. The following describes countermeasures to be taken on the E71 side to avoid high-frequency noise influence.
  - Install a ferrite core. (Page 78, Section 6.2.2)
  - Increase the number of retries of communications for TCP/IP communication.

Remark

Consult a network specialist for required devices.

5.2.3 Configuration devices used for 10BASE2 connection

A QJ71E71-B2 is used for 10BASE2 connection. Use devices that meet the IEEE 802.3 10BASE2 standards.



Configuration device	Description
RG58A/U or RG58C/U (coaxial cable 50Ω)	-
BNC-type terminating resistor	Product equivalent to 221629-4 manufactured by Tyco Electronics AMP K.K.
T-type connector	Product equivalent to UG-274/U(15) manufactured by Hirose Electric Co., Ltd.

Remark

Consult a network specialist for required devices.

CHAPTER 6 INSTALLATION AND WIRING

This chapter describes installation and wiring of the E71.

6.1 Installation

This section describes installation of the E71.

(1) Installation method

For details on installation of the E71, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(2) Handling precautions

The precautions for handling the E71 are described below.

- Do not drop or apply strong shock to the E71 case since it is made from resin.
- Tighten screws such as module fixing screws within the following range.

Screw	Tightening torque range
External power supply terminal screw (M2.5)*1	0.40N•m
Module fixing screw (M3)*2	0.36 to 0.48N•m

*1 This terminal is an external power input terminal for supplying power to a transceiver when the E71 is connected to the 10BASE5 network.

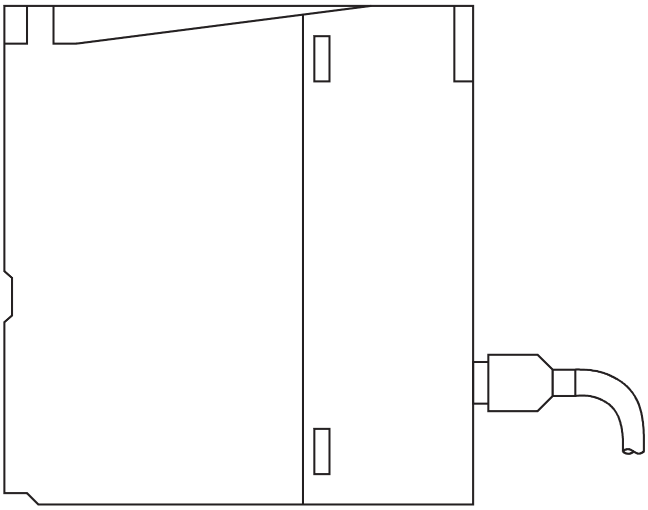
*2 A module can be easily fixed onto the base unit using the hook at the top of the module. However, it is recommended to secure the module with the module fixing screw if the module is subject to significant vibration.

6.2 Wiring

This section describes Ethernet cable wiring and wiring precautions. For a network configuration and cables and hubs used, refer to "SYSTEM CONFIGURATION". (☞ Page 63, CHAPTER 5)

6.2.1 Wiring with the QJ71E71-100

The following describes connection and disconnection of the Ethernet cable.



(1) Connecting the cable

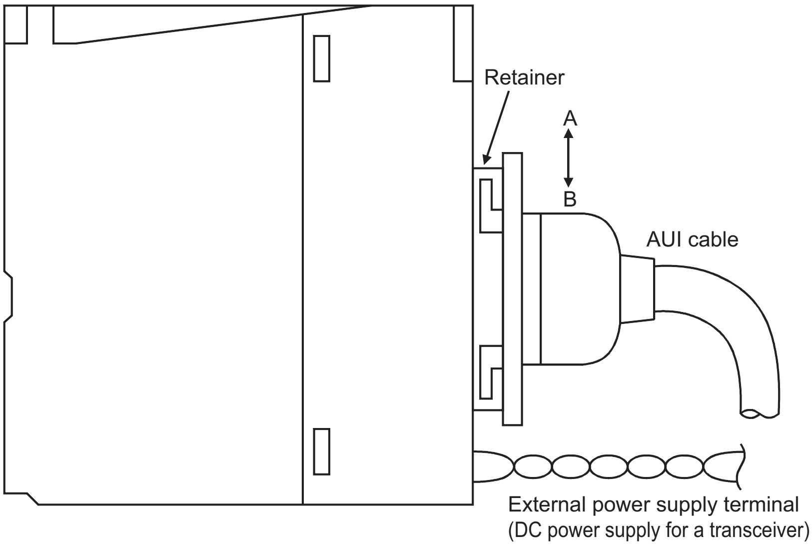
1. **Check the orientation of the connector and insert the Ethernet cable connector into the E71 until it clicks into place.**

(2) Disconnecting the cable

1. **While holding down the Ethernet cable tab, pull out the cable.**

6.2.2 Wiring with the QJ71E71-B5

The following describes connection and disconnection of the AUI cable and connection of the external power supply terminal.



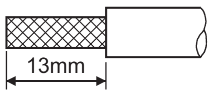
(1) Connecting the cable

1. **Slide the retainer in the orientation A and insert the AUI cable connector all the way in.**

2. **Slide the retainer in the orientation B and make sure that the AUI cable is securely locked into place.**

3. **Connect the external power supply terminal (DC power supply for a transceiver).**

Strip the cable jacket by 13mm.*1 The applicable cable size is 0.13mm^2 to 2.5mm^2 (26 to 14 AWG).



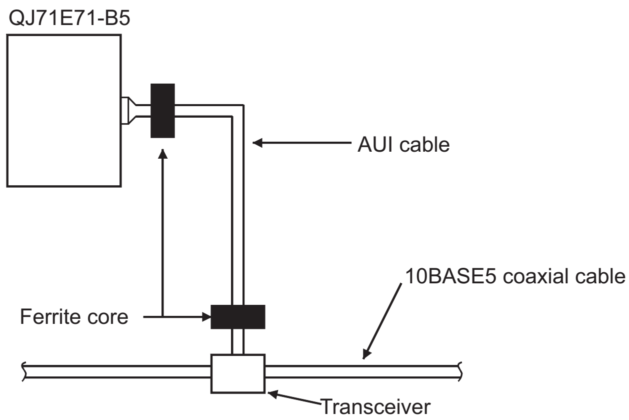
4. **Loosen the terminal screw and insert the cable into the terminal.**

5. **Tighten the terminal screw within the torque range listed in Section 6.1 (2).**

*1 If the wire strip length is too long, the conductive part is exposed and it may increase the risk of electric shock or short-circuit between the adjacent terminals. If the wire strip length is too short, it may result in poor contact.

Point

To prevent the influence of high-frequency noise using ferrite cores, install them on the E71 side, connected device side, and transceiver side on the AUI cable. (Ferrite core used for the tests conducted by Mitsubishi Electric Corporation: ZCAT 2032-0930 from TDK Corporation)

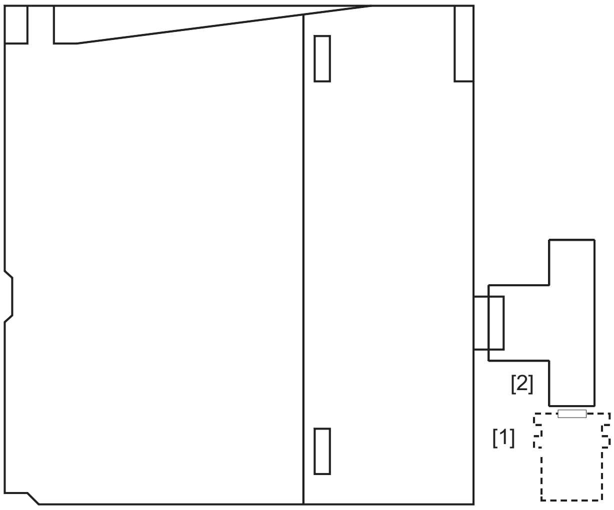


(2) Disconnecting the cable

1. **Slide the retainer in the orientation A, hold the connector on the AUI cable, and pull out the cable.**
2. **Slide the retainer in the orientation B.**

6.2.3 Wiring with the QJ71E71-B2

The following describes connection and disconnection of the coaxial cable.



(1) Connecting the cable

1. **Align the groove [1] with the groove [2] and push the connector in.**
2. **While pushing the connector in, rotate it one-quarter clockwise until it securely locks into place.**
3. **Check that the connector is locked.**

(2) Disconnecting the cable

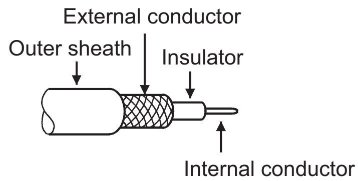
1. **Rotate the connector one-quarter counterclockwise and pull out the connector.**

Remark

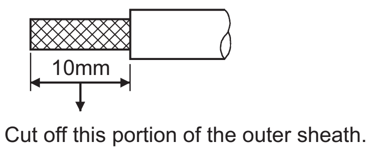
- The following shows the composition of the BNC connector and coaxial cable.

• Parts comprising the BNC connector

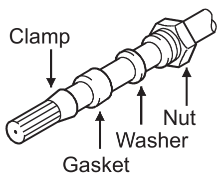
• Structure of the coaxial cable



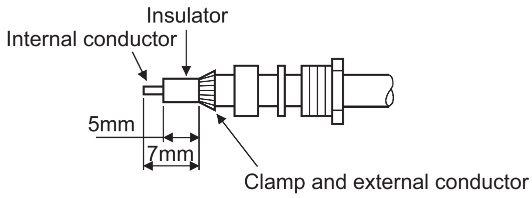
- The following shows how to attach the BNC connector and the coaxial cable.



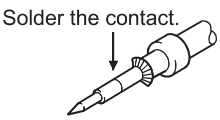
1. Cut off the outer sheath of the coaxial cable by the length shown in the figure to left. Take care not to damage the external conductor.



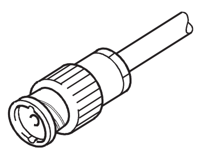
2. Fix the nut, washer, gasket, and clamp to the coaxial cable as shown in the figure to left and unfasten the external conductor.



3. Cut the external conductor, insulator, and internal conductor by the length shown in the figure to left. Cut off the external conductor to the same length as that of the tapered section of the clamp, and smooth it down to the clamp.



4. Solder the contact to the internal conductor. Note the following.
   - Make sure that the soldered part does not swell.
   - Make sure that there is no gap between the connector and the cable insulator and they do not cut into each other.
   - To prevent the insulator from deformation, solder the contact quickly.



5. Insert the contact assembled in the step 4. into the plug shell and screw the nut into the plug shell.

CHAPTER 7 COMMUNICATION PROCEDURE

This chapter describes the procedure for communicating with connected devices after the E71 is connected to Ethernet. For the procedure for connecting the E71 to Ethernet, refer to "PROCEDURES BEFORE OPERATION". (Page 61, CHAPTER 4)

Continued from PROCEDURES BEFORE OPERATION

▼

Parameter settings

Set parameters other than the basic setting and Ethernet operation setting when required.

Page 83, Section 7.1, Parameter setting of each function

▼

Starting the CPU module

Update the settings by powering off and on the CPU module or resetting the CPU module.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

▼

Initial process

The process is automatically performed with default settings of the parameters.

▼

Open process

Establish connections with connected devices.

Page 89, Section 7.2, Page 97, Section 7.3

Data communications using each function

▼

Close process

Disconnect (cancel) connections with the connected devices where connections have been established by the open process.

Page 89, Section 7.2, Page 97, Section 7.3

7.1 Setting Parameters Required for Communications

This section describes how to set parameters to communicate between the E71 and connected devices.

7.1.1 Parameter list

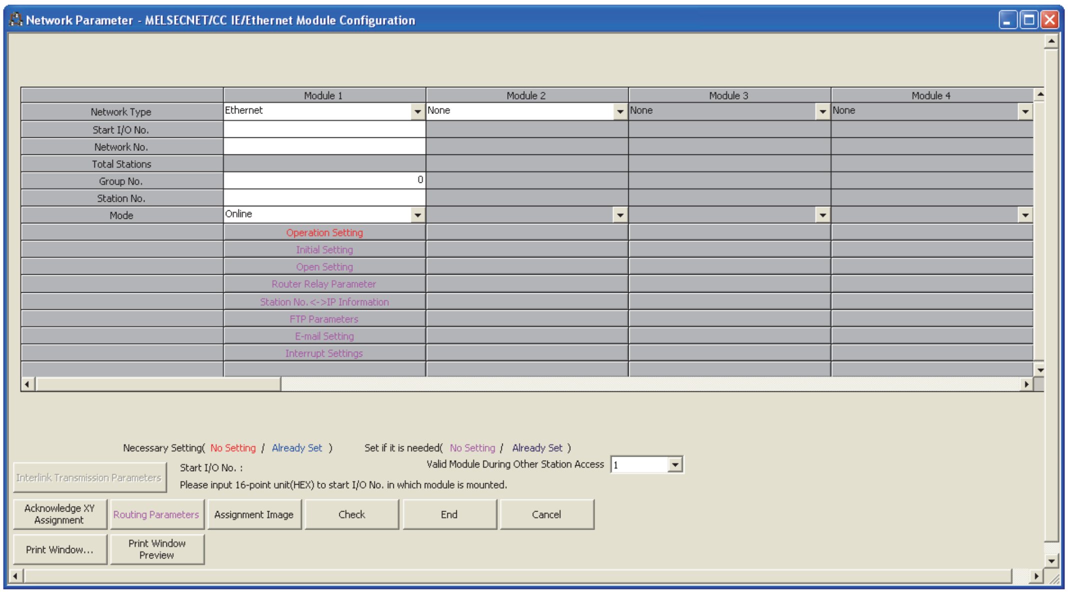
The following table lists parameters set through a programming tool.

Item			Description	Reference
Network Parameter	Basic setting	Network Type	Configure settings to use the E71 as a network module.	Page 84, Section 7.1.2
		Start I/O No.		
		Network No.		
		Station No.		
		Mode		
	Operation Setting		Configure the settings, such as an IP address, to connect the E71 to Ethernet.	Page 85, Section 7.1.3
	Initial Setting		Set a data communication timer value.	Page 354, Appendix 4.1
	Open Setting		Configure connection settings.	Page 87, Section 7.1.4, Parameter setting of each function
	Router Relay Parameter		Configure settings to communicate with connected devices on Ethernet via a router and a gateway.	MELSEC-Q/L Ethernet Interface Module User's Manual (Application)
	Station No. <-> IP Information		Configure settings (e.g. network number and station number) to communicate with other network modules.	
	FTP Parameters		Configure the settings of the file transfer (FTP) function.	
	E-mail Setting		Configure the settings of the e-mail function and the automatic notification function.	
	Interrupt Settings		Set the control number (SI) on the E71 side to request an interrupt to the CPU module.	Page 150, Section 12.5.1
	Redundant settings		Configure settings to use the E71 on the main base unit in a redundant system.	Page 221, Section 14.7.3
	Routing Parameters		Set the communication path to communicate with a station having a different network number.	MELSEC-Q/L Ethernet Interface Module User's Manual (Application)
	Group Settings		When two communication paths are configured between the connected device and the E71 on the main base unit in a redundant system, configure settings to disable system switching even if an error occurs in either communication path.	QnPRHCPU User's Manual (Redundant System)
	Valid Module During Other Station Access		Select a relay module when an access request with no network number specified is issued from another station.	-
Predefined Protocol Support Function			The connected device side protocol can be easily selected, or created/edited from the Predefined Protocol Library of GX Works2.	Page 112, CHAPTER 11
Remote Password			Select the connection protected by a remote password.	Page 201, Section 14.4

7.1.2 Basic setting

Set items, such as a network number and station number.

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ Select "Ethernet" under "Network Type".



Item	Description	Setting range
Network Type	Select "Ethernet".	-
Start I/O No.	Set the start I/O number of the E71 in units of 16 points.	Within the number of I/O points of the CPU module
Network No.	Set the network number of the E71.	1 to 239
Group No.	Set the group number of the E71.	0 to 32
Station No.	Set the station number of the E71.	1 to 64
Mode	Select the operation mode of the E71.	• Online • Offline • Self-Loopback Test • H/W Test

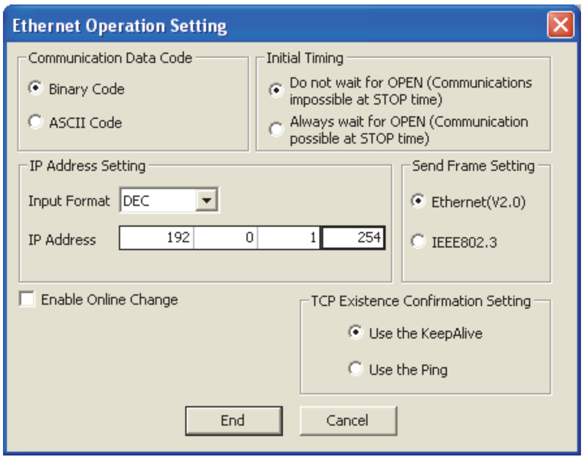
Remark

For settings to use the E71 in a redundant system, refer to the parameter settings for the redundant system function. (Page 221, Section 14.7.3)

7.1.3 Ethernet Operation Setting

Configure the settings, such as an IP address, to connect the E71 to Ethernet.

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ Select "Ethernet" under "Network Type". ⇨ "Operation Setting"



Item		Description	Setting range
Communication Data Code		Select the communication data code for the connected device.	• Binary Code • ASCII Code
Initial Timing		Page 86, Section 7.1.3 (1)	• Do not wait for OPEN (Communications impossible at STOP time) • Always wait for OPEN (Communication possible at STOP time)
IP Address Setting	Input Format	Select the IP address input format.	• DEC • HEX
	IP Address	Set the IP address of the own station. Make sure that the E71 on the own station and the connected device to be communicated with have the same class and subnet address. Consult with the network administrator for the IP address setting. Page 86, Section 7.1.3 (2)	0_H to $FFFFFFFF_H$
Send Frame Setting		Select the frame of the Ethernet header for the data link layer to be sent by the E71.	• Ethernet(V2.0) • IEEE802.3
Enable Online Change		Select whether to enable connected devices to write data in MC protocol communications while the CPU module is in RUN.	• Selected: Enable • Not selected: Disable
TCP Existence Confirmation Setting*1		Select an alive check method in TCP/IP communications.	• Use the KeepAlive • Use the Ping

*1 Do not use a programming tool that supports this setting together with one not supporting the setting. (Doing so may change the setting to "Use the Ping".) This setting is ignored if the E71 does not support the alive check function using KeepAlive. (The PING command is used for alive check.)

Remark

For settings to use the E71 in a redundant system, refer to the parameter settings for the redundant system function. (Page 221, Section 14.7.3)

(1) Initial Timing

This setting configures the open timing of the connection where "TCP" (Passive open) or "UDP" has been selected under "Open System" in the open setting. (Page 87, Section 7.1.4)

(a) Do not wait for OPEN (Communications impossible at STOP time)

Connections are opened or closed using a program. Communications cannot be performed while the switch on the CPU module is set to STOP.

(b) Always wait for OPEN (Communication possible at STOP time)

Connections always wait for the open status according to the "Open System" parameter. (This eliminates the need to open/close connections using a program.*1) Communications can be performed while the switch on the CPU module is set to STOP.

*1 If the program of the CPU module on the own station closes a connection, the station does not enter the OPEN request wait status after the connection is disconnected.

Point

When remotely controlling the CPU module from a connected device, select "Always wait for OPEN (Communication possible at STOP time)". If "Do not wait for OPEN (Communications impossible at STOP time)" is selected, the communication line is closed during remote STOP. After the line is closed, the line cannot be reopened on the CPU module side, and remote RUN cannot be performed from the connected device.

(2) IP address setting

Set the IP address of the own station so that it has the same class and subnet address as that of the connected device to be communicated.

Ex. When a subnetwork is not used

- IP address of the own station: 129.5.1.1
- IP address of the connected device: 129.5.47.1
- Subnet mask of the connected device: None

In the above case, the IP address of the connected device is class B.
Since class B has a default subnet mask of 255.255.0.0, set the IP address of the own station so that the first and second octets are the same as those of the connected device.

Ex. When a subnetwork is used

- IP address of the own station: 129.5.47.5
- IP address of the connected device: 129.5.47.1
- Subnet mask of the connected device: 255.255.255.0

In the above case, although the IP address of the connected device is class B, the first to third octets are the subnet address because the subnet mask is set to the first to third octets.
Therefore, set the IP address of the own station so that the first to third octets are the same as those of the connected device.

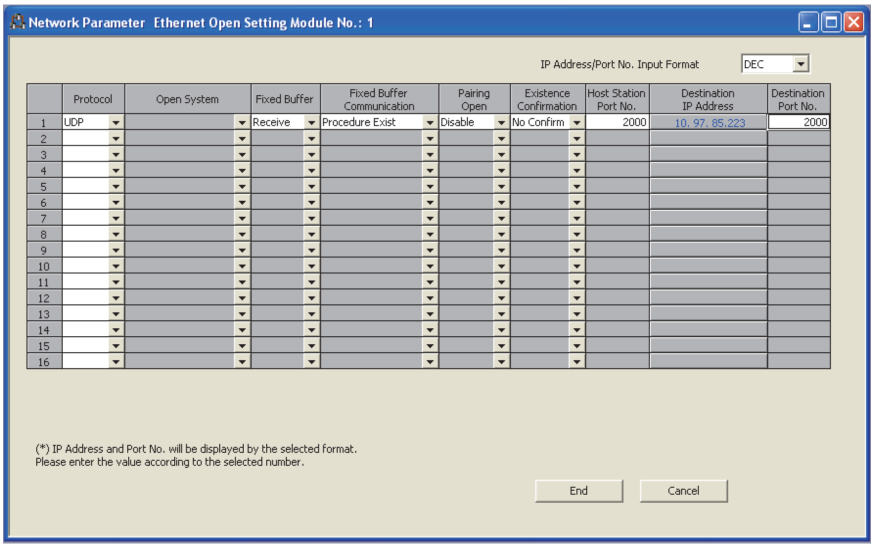
Remark

- The IP address pattern for each class is as follows.
  Class A: 0.x.x.x to 127.x.x.x, Class B: 128.x.x.x to 191.x.x.x, Class C: 192.x.x.x to 223.x.x.x
- The host address for each class is the part shown with "0".
  Class A: 255.0.0.0, Class B: 255.255.0.0, Class C: 255.255.255.0

7.1.4 Open Setting

Configure settings to open connections for data communications with connected devices.

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ Select "Ethernet" under "Network Type". ⇨ "Open Setting"



Item	Description	Setting range
IP Address/Port No. Input Format	Select the input format for the IP address and port number.	• DEC • HEX
Protocol	Select the communication protocol for the connected device. (Page 89, Section 7.2, Page 97, Section 7.3)	• TCP • UDP
Open System	Set this item when "TCP" is selected under "Protocol". (Page 90, Section 7.2.2, Page 91, Section 7.2.3, Page 93, Section 7.2.4)	• Active • Unpassive • Fullpassive • MELSOFT Connection
Fixed Buffer	For communications using a fixed buffer, select whether to use the buffer for sending or for receiving in a connection to the connected device. When not performing communications using a fixed buffer, select "Send". (Page 139, CHAPTER 12)	• Send • Receive
Fixed Buffer Communication	Select a communication method for communications using a fixed buffer. (Page 139, CHAPTER 12)	• Procedure Exist • No Procedure • Predefined protocol
Pairing Open	Configure the pairing open setting for communications using a fixed buffer. (Page 158, Section 12.7)	• Disable • Enable
Existence Confirmation	Select whether to use the alive check function. (Page 234, Section 14.8)	• No Confirm • Confirm
Host Station Port No.	Set the E71 connection port numbers.	401_H to 1387_H, $138B_H$ to $FFFE_H$[*1]
Destination IP Address	Set the IP address of the connected device.	0_H to $FFFFFFFF_H$
Destination Port No.	Set port numbers for connections of connected devices.	1_H to $FFFF_H$[*2]

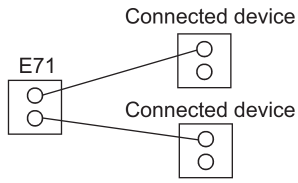
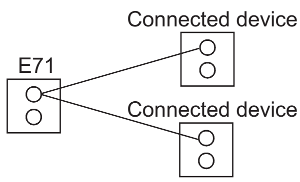
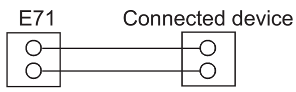
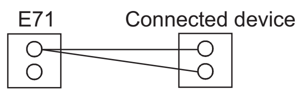
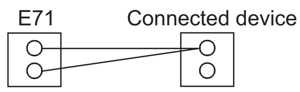
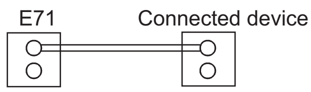
*1 1388_H to $138A_H$ cannot be specified. (Page 350, Appendix 2)

*2 The range of 1_H to 400_H is available only for the QJ71E71-100 with the serial number (first five digits) of "15042" or later.

Point

- Note the following points when setting port numbers.

○: Can be set, ×: Cannot be set

Connection status	Description	Communication protocol	
		TCP/IP	UDP/IP
	When connecting multiple connected devices, set multiple own station port numbers.	○	○
	When connecting multiple connected devices, set a single own station port number. (However, connections need to be opened by the number of connections.) This setting cannot be configured when the own station is Unpassive.	○	×
	When connecting multiple ports on a connected device, set multiple port numbers of the E71.	○	○
	When connecting multiple ports on a connected device, set a single port number of the E71. (However, connections need to be opened by the number of connections.) This setting cannot be configured when the own station is Unpassive.	○	×
	When connecting the same port on a connected device, set multiple port numbers of the E71. (However, connections need to be opened by the number of connections.)	○	○
	Multiple ports can be set for the same port on a connected device and on the E71 only when paring open has been set.	○	○

- Consult with the network administrator for setting the port numbers of the E71 and the connected device and the IP address of the connected device.

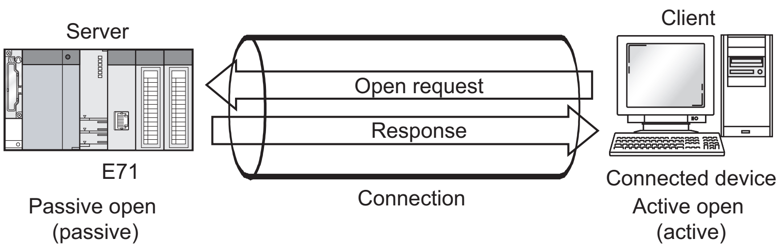
7.2 TCP/IP Communications

This section describes TCP/IP communications.

7.2.1 Establishing a connection

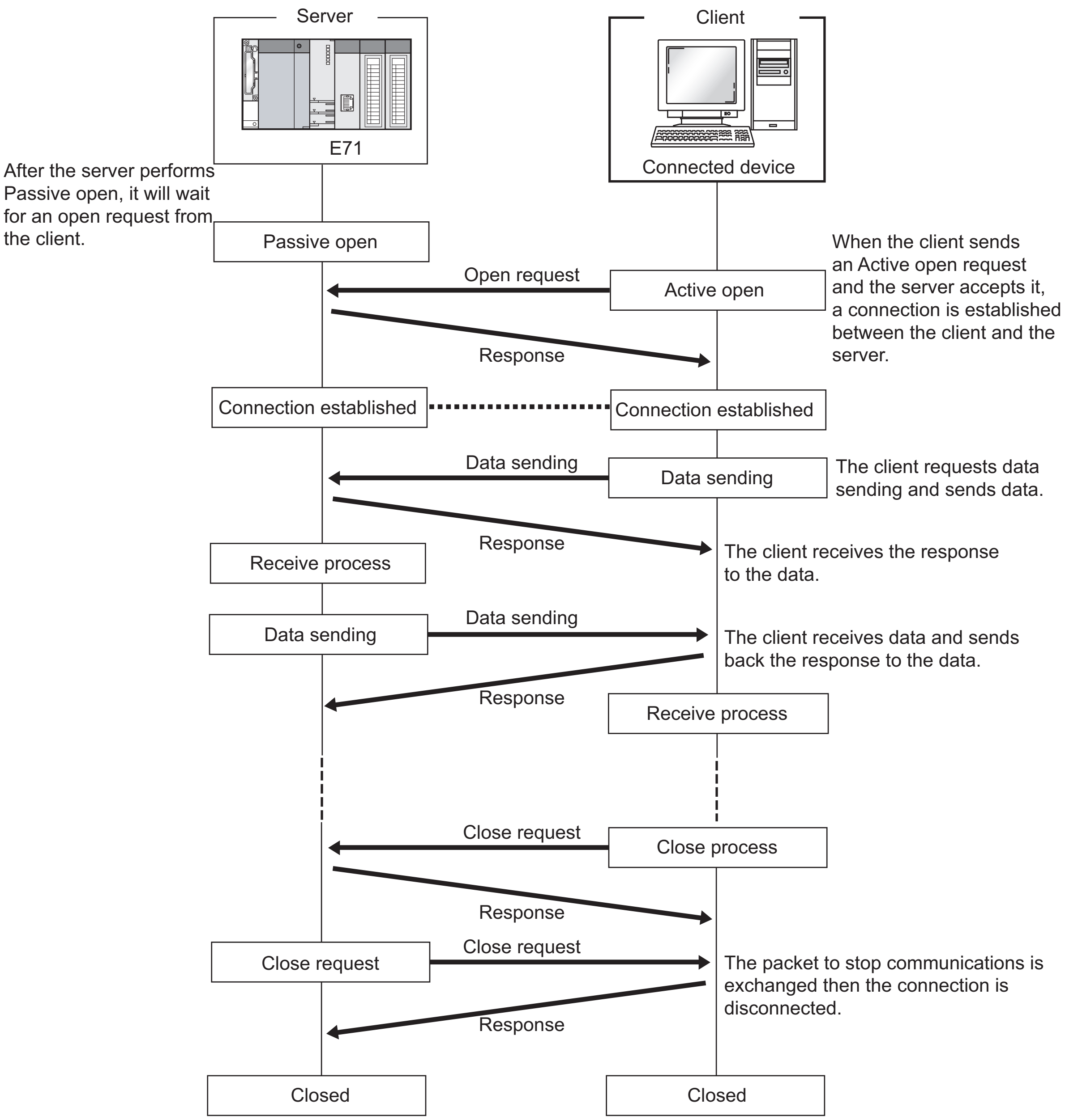
TCP/IP communications require establishing connections between communication devices. When the server-side device is in standby status after performing a Passive open process, a connection is established after the client-side device issues an open request (Active open process) to the server and a response is received.
In TCP/IP communications, the system establishes a connection and checks whether each data has arrived at the recipient normally, thereby ensuring data reliability. Compared to UDP/IP communications, however, TCP/IP places a greater load on the line.

Ex. When the E71 is in Passive open status



7.2.2 Communication process

This section describes the process from establishing a connection to terminating communications.



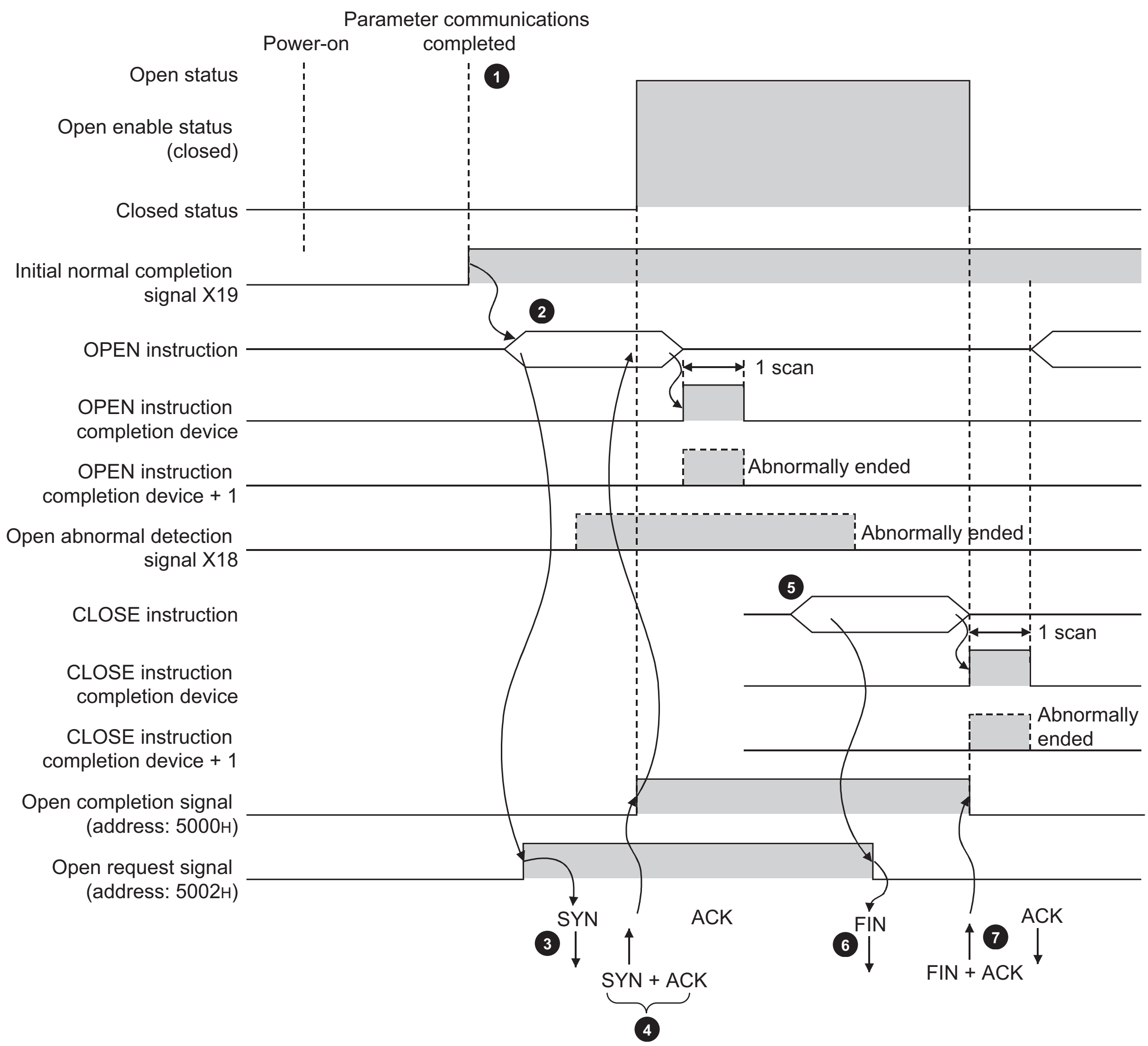
Point

After the connected device sends a close request to the E71, wait for 500ms or more before performing an open process again.

7.2.3 Active open procedure

Active open is a connection method that performs an active open process on a connected device in Passive open wait status. The procedure that the E71 performs an Active open process is as follows. For the OPEN/CLOSE instructions, refer to "DEDICATED INSTRUCTIONS". (☞Page 236, CHAPTER 15)

Ex. Open/close processes for the connection number 1



1. After parameter communications, check that the initial process of the E71 is normally completed.
(Initial normal completion signal (X19): ON)
2. Start the open process using the OPEN instruction. (Open request signal (address: 5002_H (b0)): ON)
3. The E71 performs the open process. (The E71 sends an open request (SYN) to the connected device.)
4. When the open process is normally completed, data communications are enabled.*1
5. Start a close process using the CLOSE instruction. (Open request signal: OFF)
6. The E71 performs the close process. (The E71 sends a close request (FIN) to the connected device.)
7. When the close process is normally completed, data communications terminate.*2

*1 If the E71 sends an SYN and the connected device returns a RST, Open abnormal completion (X18) turns on immediately and the open process terminates. (Open abnormal end)

*2 If an ACK and a FIN are not returned within the time specified by the TCP end timer value, the E71 forcibly closes the connection (sends a RST). (Close abnormal end)

7.2.4 Passive open procedure

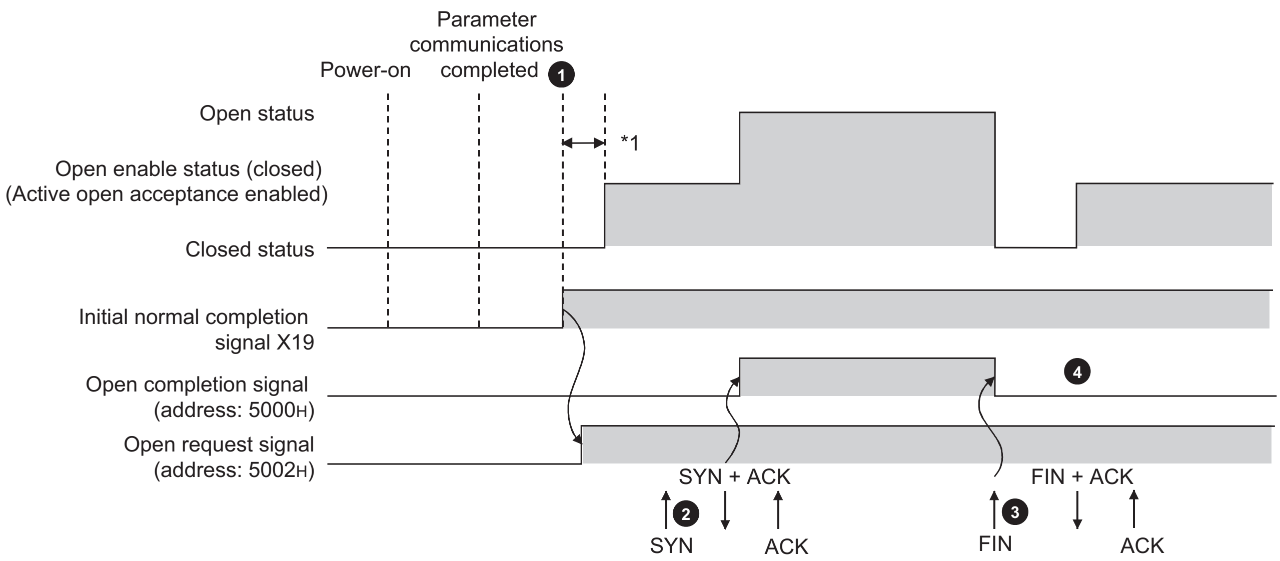
The E71 has the following two connection methods for Passive open.

- Unpassive: Performs a passive open process on connections for all devices connected to the network, regardless of the IP address and port number of the destination device.
- Fullpassive: Performs a passive open process on connections for the connected device specified by the IP address and port number.

Under the Passive open method, the open/close processes are performed using the procedures described below, according to the Ethernet operation setting.

(1) When "Always wait for OPEN" is selected for the Ethernet operation setting

As the E71 is set to always wait for open, a connection is established when an Active open is requested from the connected device. This eliminates the need for open/close processing programs on the E71 side.



1. After parameter communications, check that the initial process of the E71 is normally completed.
(Initial normal completion signal (X19): ON)
After the initial process is normally completed, the connection enters open enable status and the E71 waits for an open request from the connected device.
2. The E71 starts the open process upon receiving the open request (SYN) from the connected device. When the open process is normally completed, Open completion signal (address: 5000_H (b0)) turns on and data communications are enabled.
3. The E71 starts the close process upon receiving the close request (FIN) from the connected device. When the close process is completed, Open completion signal (address: 5000_H (b0)) turns off and data communications are disabled.
4. After the internal process of the E71 is completed, the connection returns to the open acceptance enable status.

*1 An open request (SYN) received after the normal completion of an initial process and before the E71 is in the open acceptance enable status will cause an error, and the E71 sends a connection forced close (RST).

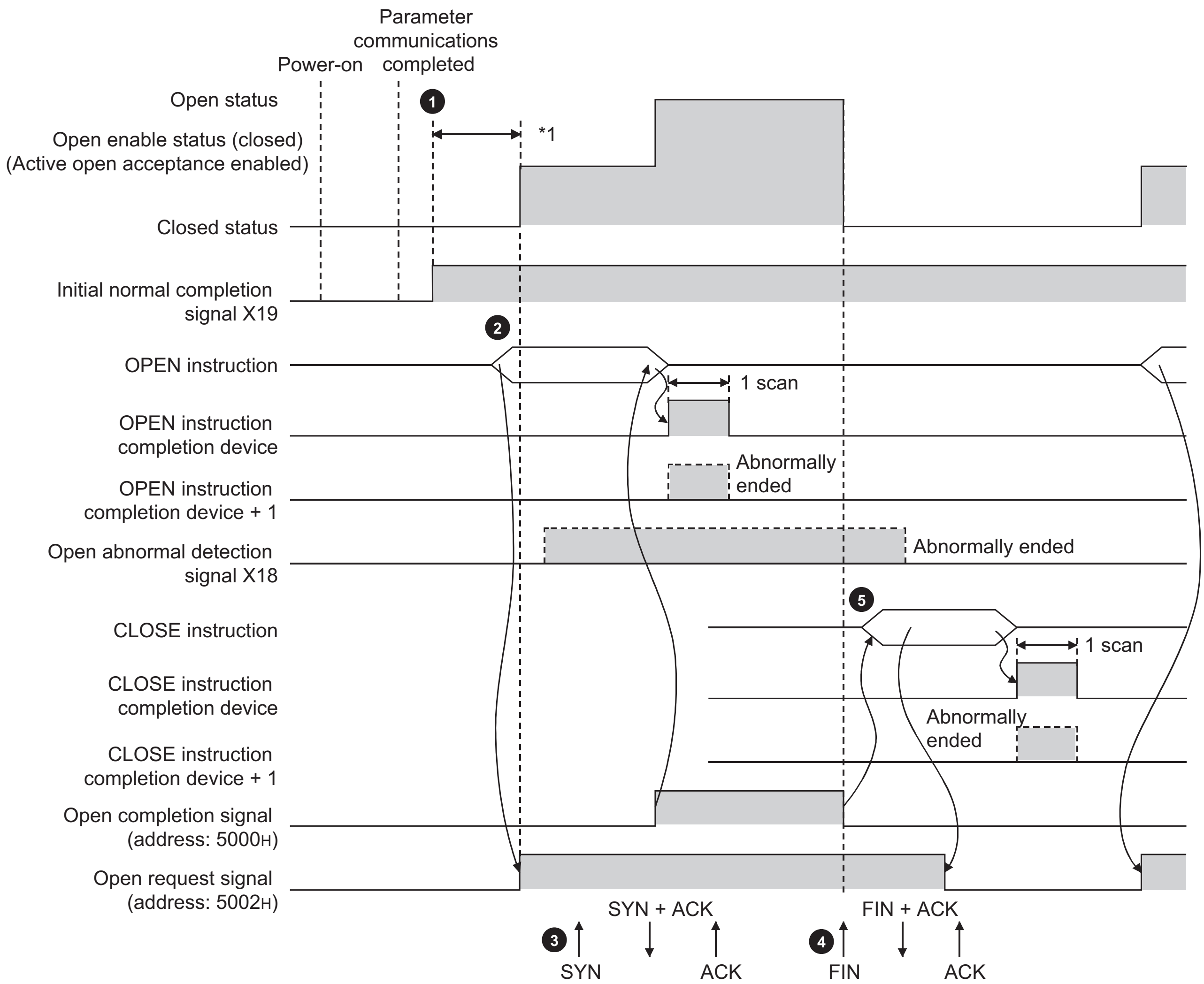
Point

When the open/close processes are performed from the E71 side using a dedicated instruction, even if "Always wait for OPEN" has been selected for the Ethernet operation setting, the connection will not return to the open acceptance enable status after the close process is completed.

(2) When "Do not wait for OPEN" is selected for the Ethernet operation setting

Executing the OPEN/CLOSE instructions is required on the E71 to make the E71 enter open/close wait status before receiving an open/close request from the connected device. When the open process is normally completed, data sending and receiving are enabled. For the OPEN/CLOSE instructions, refer to "DEDICATED INSTRUCTIONS". (Page 236, CHAPTER 15)

Ex. Open/close processes for the connection number 1



1. After parameter communications, check that the initial process of the E71 is normally completed. (Initial normal completion signal (X19): ON)
2. Start the open process using the OPEN instruction. (Open request signal (address: 5002_H (b0)): ON)
3. The E71 starts the open process upon receiving the open request (SYN) from the connected device. When the open process is normally completed, Open completion signal (address: 5000_H (b0)) turns on and data communications are enabled.
4. The E71 starts the close process upon receiving the close request (FIN) from the connected device. When the close process is completed, Open completion signal (address: 5000_H (b0)) turns off and data communications are disabled.
5. Start the close process using the CLOSE instruction. (Open request signal: OFF)

*1 An open request (SYN) received after the normal completion of an initial process and before the E71 is in the open acceptance enable status will cause an error, and the E71 sends a connection forced close (RST).

Point

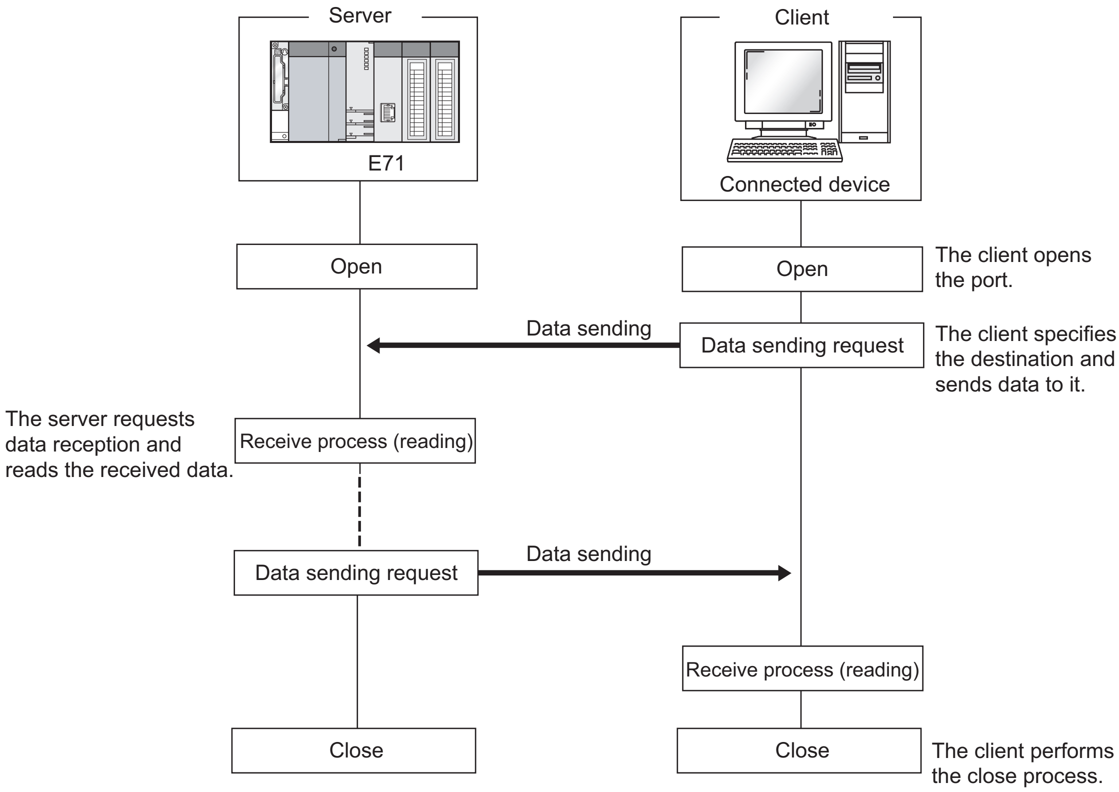
- Change connection settings before executing the OPEN instruction.
- Once an open process is performed, an open request cannot be canceled before the open process is completed. Perform a close process (Execute the CLOSE instruction) after the open process is completed.

7.3 UDP/IP Communications

This section describes UDP/IP communications. In UDP/IP communications, the system does not establish a connection and does not check whether each data has arrived at the destination normally, thereby reducing the load on the line. However, UDP/IP communications do not guarantee data reliability as the TCP/IP communications do.

7.3.1 Communication process

Unlike TCP/IP communications, UDP/IP communications do not require connections to be established with connected devices.



Point

After the connected device sends a close request to the E71, wait for 500ms or more before performing an open process again.

7.3.2 Open procedure

The open/close processes are performed by the procedures described below, according to the Ethernet operation setting.

(1) When "Always wait for OPEN" is selected for the Ethernet operation setting

After the E71-mounted station has been started up, the connection in UDP/IP communications automatically opens and data sending/receiving are enabled. Creating open/close processing programs is not required.

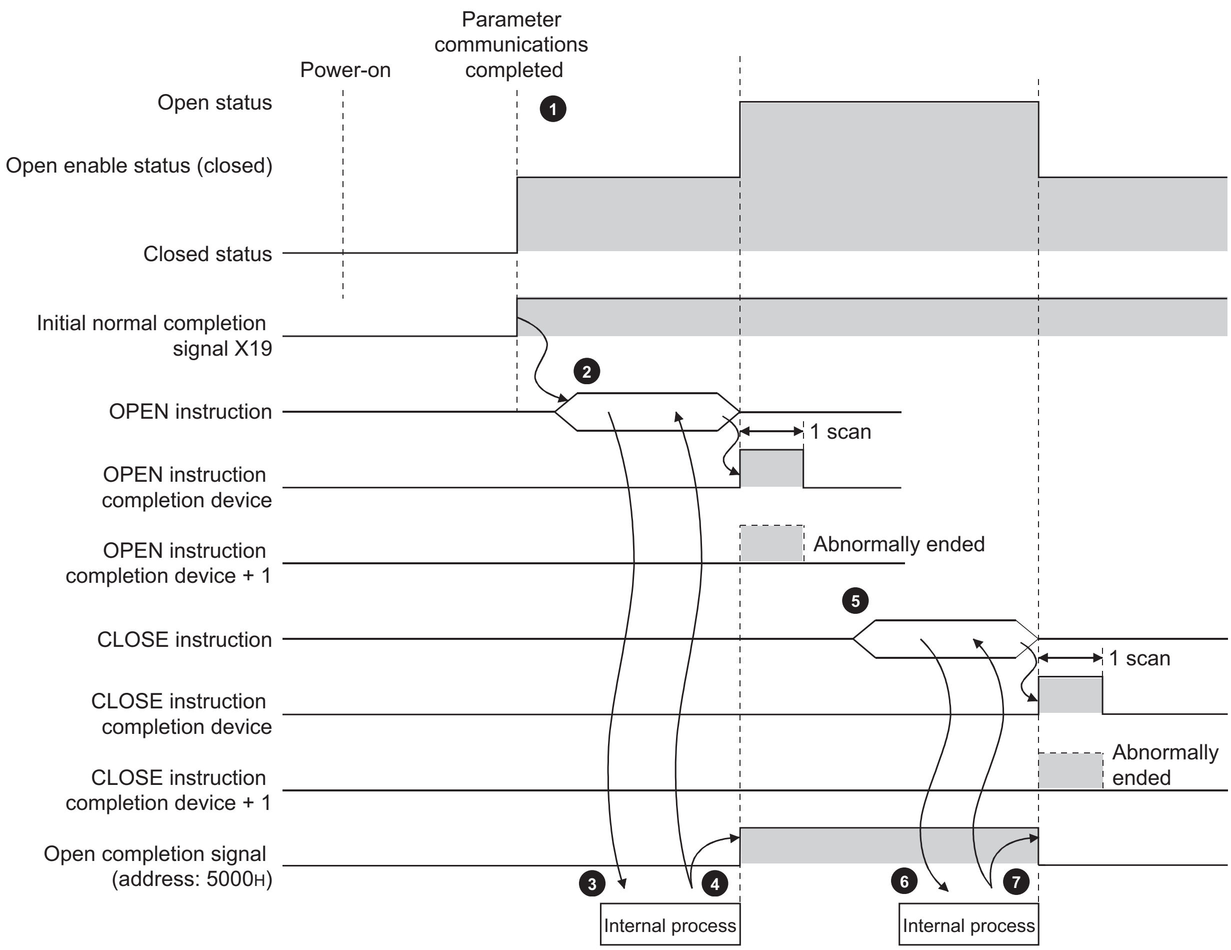
Point

When open/close processes are performed from the E71 side using a dedicated instruction, even if "Always wait for OPEN" has been selected for the Ethernet operation setting, all open/close processes after the connection to the connected device is established must be performed in a program.

(2) When "Do not wait for OPEN" is selected for the Ethernet operation setting

Executing the OPEN/CLOSE instructions is required on the E71 to make the E71 enter open/close wait status open/close wait status before receiving an open/close request from the connected device. When the open process is normally completed, data sending and receiving are enabled. For the OPEN/CLOSE instructions, refer to "DEDICATED INSTRUCTIONS". (Page 236, CHAPTER 15)

Ex. Open/close processes for the connection number 1



1. After parameter communications, check that the initial process of the E71 is normally completed. (Initial normal completion signal (X19): ON)
2. Start the open process using the OPEN instruction. (Open request signal (address: 5002_H (b0)): ON)
3. The E71 performs the open process. (Internal process only)
4. Data communications are enabled when the open process is normally completed.
5. Start the close process using the CLOSE instruction. (Open request signal: OFF)
6. The E71 performs the close process. (Internal process only)
7. When the close process is normally completed, data communications terminate.

CHAPTER 8 CONNECTING MELSOFT PRODUCTS AND A GOT

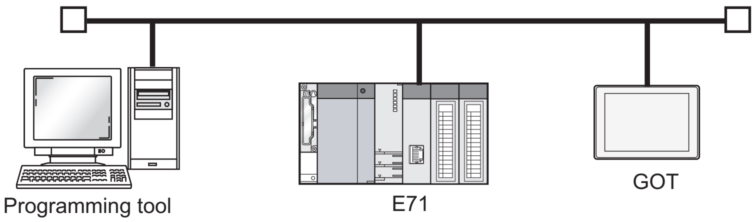
This chapter describes the connection of the E71 with MELSOFT products (such as a programming tool and MX Component) and the GOT.

8.1 Applications

This section describes applications according to a connection type.

(1) Programming and monitoring over Ethernet

In Ethernet, a programming tool can create programming of a programmable controller and monitor a programmable controller (MELSOFT connection), and the GOT can monitor and test a programmable controller. Remote operations making full use of the Ethernet capability, long-distance connectivity and high-speed communications, are achieved.



(2) Connecting multiple products

Multiple MELSOFT products and GOTs can be connected using MELSOFT application communication ports.

Point

The connection used in the connection with MELSOFT products is only for data communications with the MELSOFT products and cannot be used for data communications with connected devices other than MELSOFT products.

(a) Connection using the TCP/IP communications

The E71 can connect to up to 17 MELSOFT products using one dedicated system connection and 16 user connections. GOTs cannot be connected using the TCP/IP communications. (Use the UDP/IP communications.)

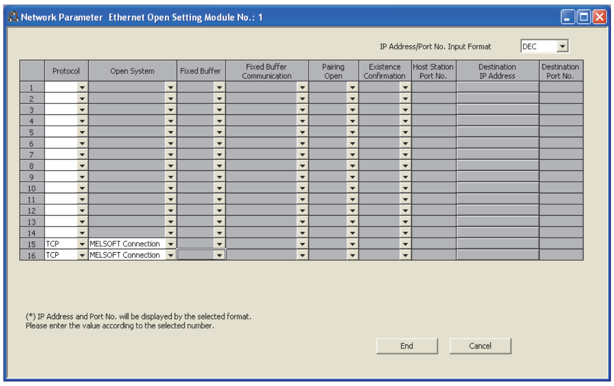
(b) Connection using the UDP/IP communications

The E71 can connect a MELSOFT product and a GOT using one dedicated system connection.

8.2 Data Communication Procedure

This section describes the data communication procedure in the MELSOFT connection.

Ex. Connecting the E71 and a personal computer (GX Works2) over Ethernet



1. **Write the parameter settings (Page 87, Section 7.1.4) in the CPU module.**

In the following cases, the open setting is not required.

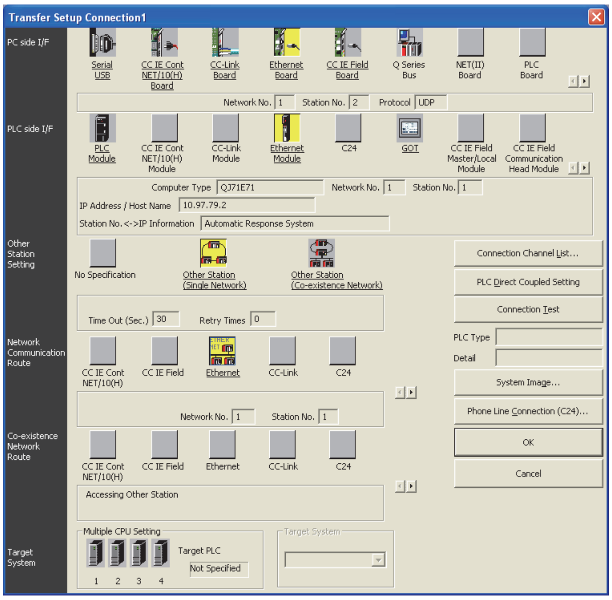
- Only one product is connected in the TCP/IP communications.

  One product can be connected without the open setting using the dedicated system connection.

- The network is running in the UDP/IP communications.

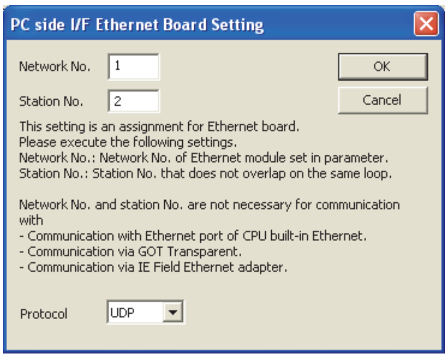
  Multiple products can be connected without the open setting using the dedicated system connection.

To connect multiple products in the TCP/IP communications, set the connection for the second product or more in the open setting.

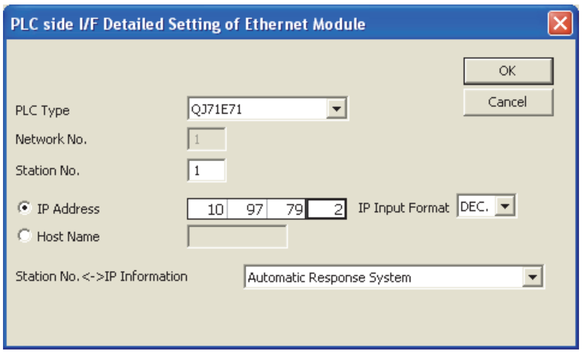


2. **Set the connection target in the programming tool.**

Connection destination window ⇨ [Connection1]



3. **Select "Ethernet Board" under "PC side I/F" and double-click the item. The window shown to left appears. Set the network number according to the network parameter of the E71. Assign a unique station number.**



4. **Select "Ethernet Module" under "PLC side I/F" and double-click the item. The window shown to left appears. Set the station number and the IP address according to the network parameters.**

5. **Set "Other Station Setting" and "Network Communication Route" when required.**

6. **A Passive open process is performed on all the MELSOFT products connected to the network. (The E71 will wait for an Active open request to itself.)**

CHAPTER 9 MC PROTOCOL COMMUNICATIONS

Using MC protocol communications, connected devices that can send and receive data in accordance with the MC protocol can access a CPU module. Because an E71 processes and sends/receives data based on commands from connected devices, no programs for data communications are required on the programmable controller side.
For MC protocol communications, refer to the following.

MELSEC Communication Protocol Reference Manual

9.1 Applications

Applications of MC protocol communications are as follows.

(1) Data read/write

Data can be read/written from/to a CPU module device and intelligent function module buffer memory. This allows a connected device to monitor the CPU module operation, perform data analysis, and manage production control.

(2) File read/ write

Files such as a program and parameter stored in a CPU module can be read/written. This allows a connected device to manage CPU module files and switch execution programs.

(3) Remote control of a CPU module

The remote operation allows a connected device to remotely operate a CPU module.

(4) Turning off the COM.ERR. LED

The COM.ERR. LED on an E71 can be turned off from a connected device.

(5) Remote password lock/unlock

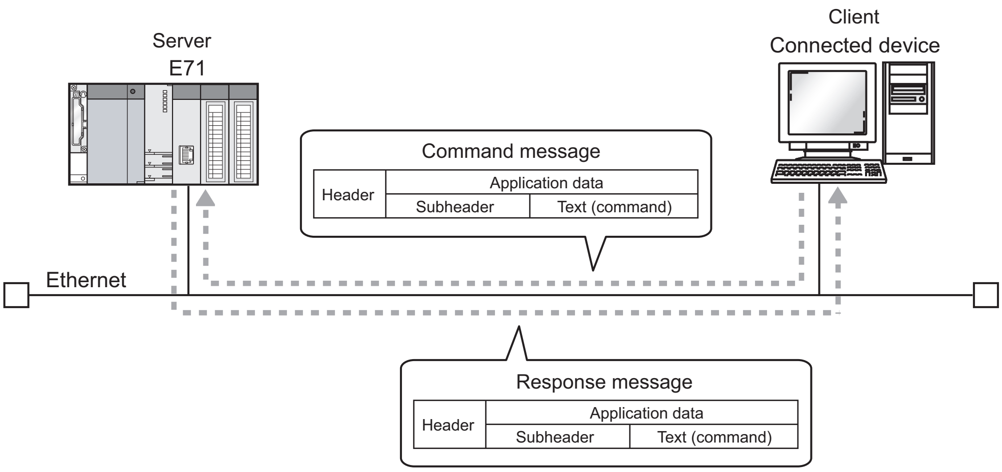
A remote password can be locked/unlocked from a connected device.

(6) Access to a programmable controller in other stations over other networks

In a system containing CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10, and Ethernet, access to a programmable controller in other stations can be performed over each network from a connected device.

9.2 Communication Structure

When a connected device sends a message to an E71 in the MC protocol message format, the E71 performs the process according to the message. During communications, a CPU module including the E71 operates as a server, and the connected device (a personal computer or other terminals) operates as a client. Based on the command message received from the client, the server (the E71) automatically sends an appropriate response message.



The frames that can be used with an E71 are as follows.

- 4E frame
- QnA-compatible 3E frame
- A-compatible 1E frame

Remark

When the connected device is a personal computer, using a separately sold communication support tool (MX Component) allows a communication program for the connected device to be created without considering the MC protocol message format or sending/receiving procedures. (Page 391, Appendix 9)

9.3 Data Communication Procedure

The following is a data communication procedure for MC protocol communications.

1. **Set the parameters. (☞ Page 106, Section 9.4)**
2. **Perform the open process and establish a connection between the E71 and the connected device. (☞ Page 89, Section 7.2, Page 97, Section 7.3)**
3. **Once the connection is established, the connected device sends an MC protocol message.**
4. **The connection is closed after the communications are completed.**

Point

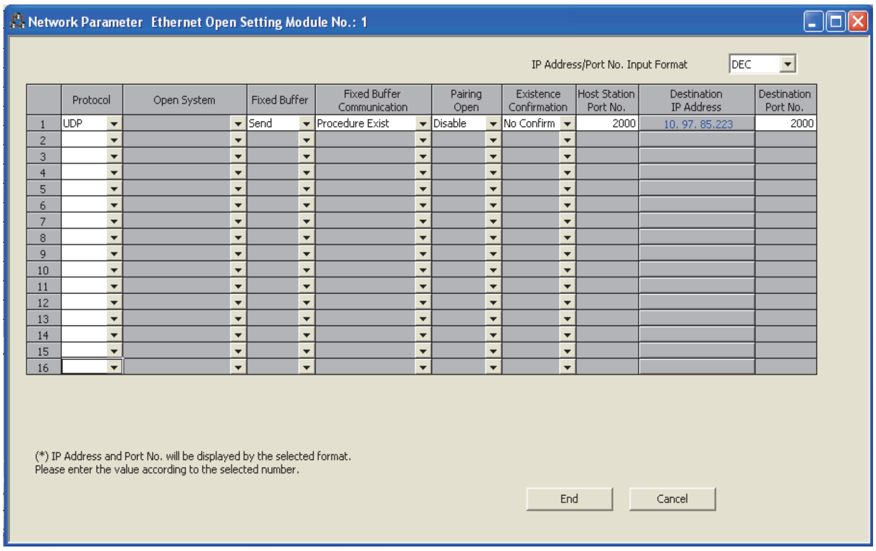
In the cases described below, an E71 performs a remote password check when the connected device accesses the programmable controller. If communications cannot be performed, unlock the remote password. (☞ Page 202, Section 14.4.2)

- When a remote password has been set in the CPU module
- When the connection for data communications with the connected device is to be subjected to the remote password check

9.4 Parameter Setting

Set the following parameters for MC protocol communications.

- Basic setting (Page 84, Section 7.1.2)
- Ethernet operation setting (Page 85, Section 7.1.3)
- Open setting (Page 87, Section 7.1.4)



Item	Description	Setting range
Fixed Buffer Communication	Select the communication method for communications using a fixed buffer. For MC protocol communications, select "Procedure Exist".	Procedure Exist

CHAPTER 10 SLMP COMMUNICATIONS

SLMP is a protocol that enables connected devices to access SLMP supporting devices over Ethernet.
SLMP communications are available among devices that can receive/send messages in the SLMP control procedure.
This function is available only in the QJ71E71-100 with the serial number (first five digits) of "15042" or later.

For SLMP communications, refer to the following.
SLMP Reference Manual

10.1 Applications

Applications of SLMP communications are as follows.

(1) Data read/write

Data can be read/written from/to a device of the CPU module where an E71 is mounted and intelligent function module buffer memory. This allows a connected device to monitor the operation of the CPU module where an E71 is mounted, perform data analysis, and manage production control.

(2) Remote control of a CPU module

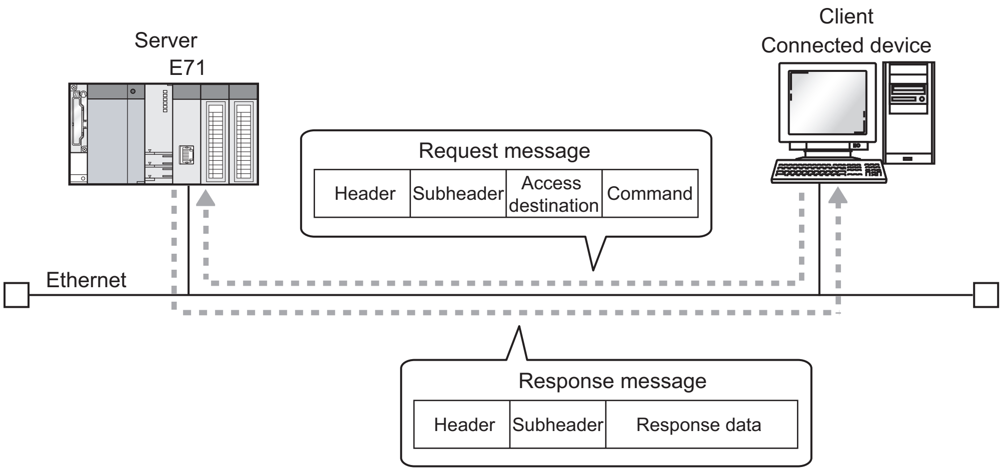
The remote operation allows a connected device to remotely operate a CPU module where an E71 is mounted.

(3) Turning off the COM.ERR. LED

The COM.ERR. LED on an E71 can be turned off from a connected device.

10.2 Communication Structure

When a connected device sends a message to an E71 in the SLMP message format, the E71 performs the process according to the message. During communications, a CPU module including the E71 operates as a server, and the connected device (a personal computer or other terminals) operates as a client. Based on the request message received from the client, the server (the E71) automatically sends an appropriate response message.



10.3 Data Communication Procedure

The following is a data communication procedure for SLMP communications.

1. **Set the parameters. (Page 109, Section 10.4)**
2. **Perform the open process and establish a connection between the E71 and the connected device. (Page 89, Section 7.2, Page 97, Section 7.3)**
3. **Once the connection is established, the connected device sends an SLMP message.**
4. **The connection is closed after the communications are completed.**

Point

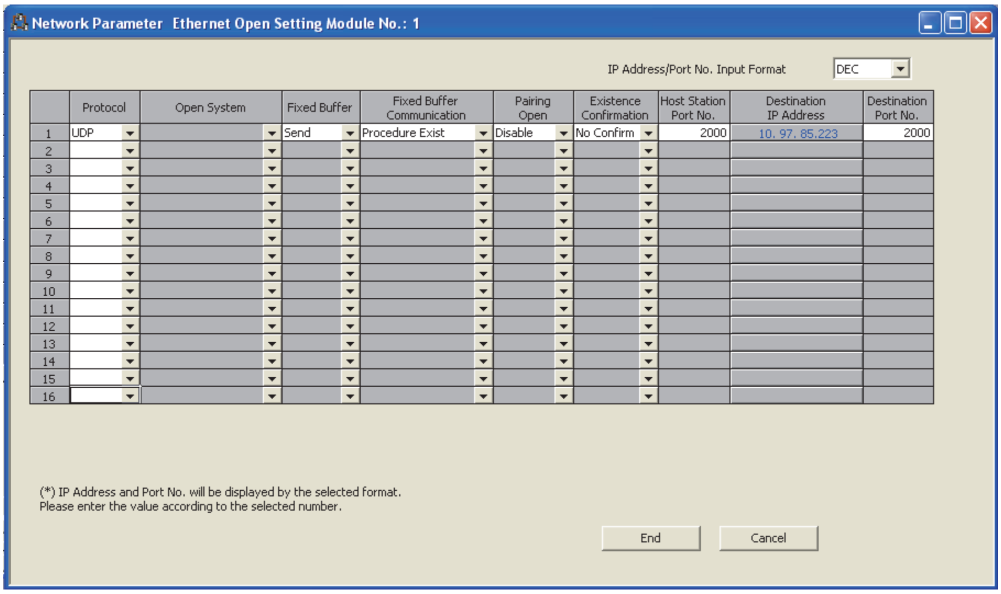
In the cases described below, an E71 performs a remote password check when the connected device accesses the programmable controller. If communications cannot be performed, unlock the remote password. (Page 202, Section 14.4.2)

- When a remote password has been set in the CPU module
- When the connection for data communications with the connected device is to be subjected to the remote password check

10.4 Parameter Setting

Set the following parameters for SLMP communications.

- Basic setting (Page 84, Section 7.1.2)
- Ethernet operation setting (Page 85, Section 7.1.3)
- Open setting (Page 87, Section 7.1.4)



Item	Description	Setting range
Fixed Buffer Communication	Select the communication method for communications using a fixed buffer. For SLMP communications, select "Procedure Exist".	Procedure Exist

10.5 Available command list

The following table lists the commands that can be executed from a connected device to the E71.
The □ part in the Subcommand column varies depending on the specified device.
For details on each command, refer to the following.

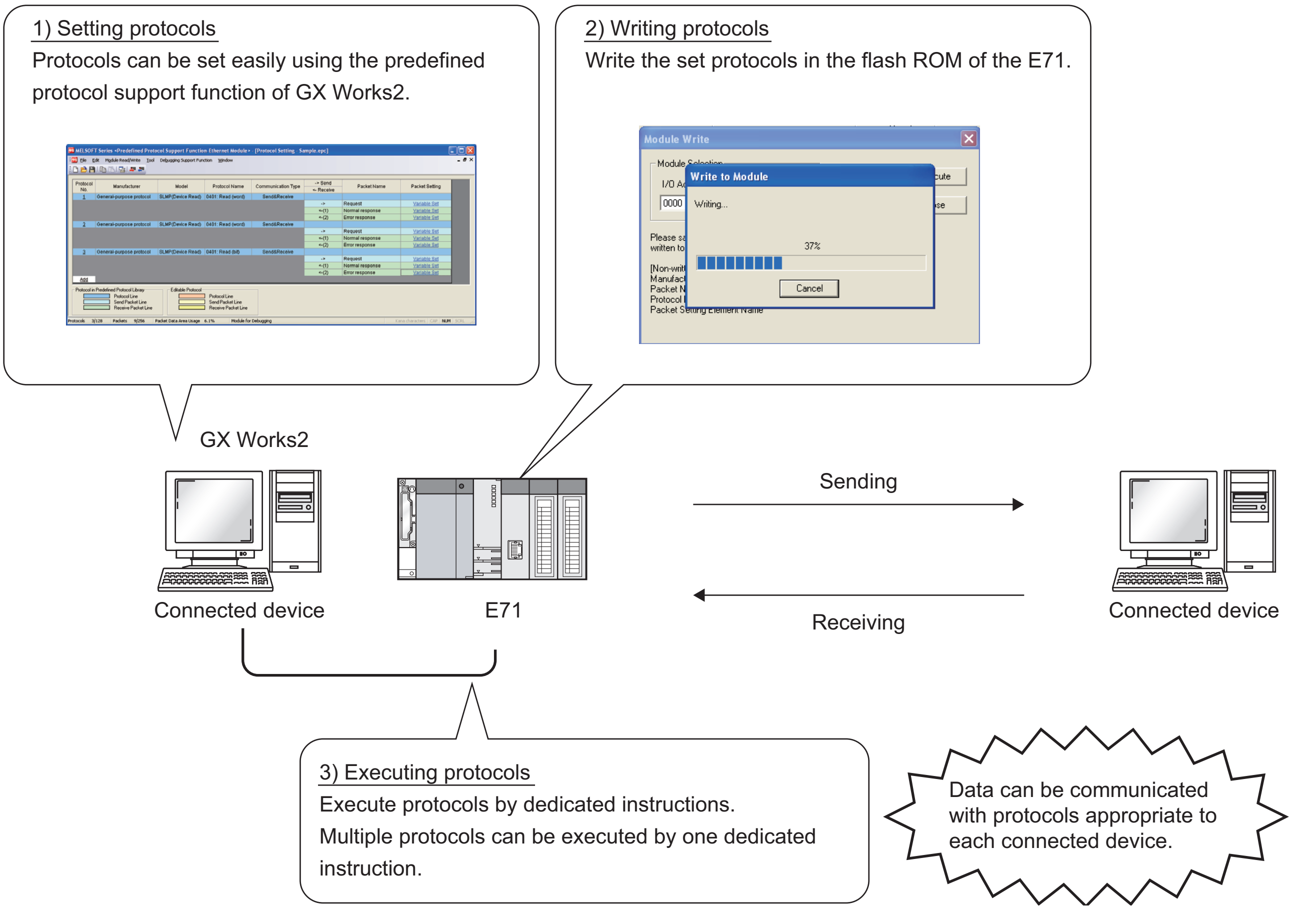
SLMP Reference Manual

Item		Command	Subcommand	Description
Type	Operation			
Device	Read	0401	00□1	Reads values in 1-point units from bit devices (consecutive device numbers).
			00□0	• Reads values in 16-point units from bit devices (consecutive device numbers). • Reads values in 1-word units from word devices (consecutive device numbers).
	Write	1401	00□1	Writes values to bit devices (consecutive device numbers) in 1-point units.
			00□0	• Writes values to bit devices (consecutive device numbers) in 16-point units. • Writes values to word devices (consecutive device numbers) in 1-word units.
	Read Random	0403	00□0	Specifies a device number to read the device value. Discrete device numbers can be used. • Reads bit devices in 16- or 32-point units. • Reads word devices in 1- or 2-word units.
	Write Random	1402	00□1	Specifies a device number in 1-point units to write a value to a bit device. Discrete device numbers can be used.
			00□0	• Specifies a device number in 16-point units to write a value to a bit device. Discrete device numbers can be used. • Specifies a device number in 1- or 2-word units to write a value to a word device. Discrete device numbers can be used.
	Entry Monitor Device	0801	00□0	Registers a device to be read by Execute Monitor (command: 0802).
	Execute Monitor	0802	0000	Reads the value of the device registered by Entry Monitor Device (command: 0801).
	Read Block	0406	00□0	Specifies n points (1 point: 16 bits) of word or bit devices as one block, and reads multiple blocks of data. Discrete device numbers can be used.
	Write Block	1406	00□0	Specifies n points (1 point: 16 bits) of word or bit devices as one block, and writes multiple blocks of data. Discrete device numbers can be used.
Memory	Read	0613	0000	Reads buffer memory data of the E71.
	Write	1613	0000	Writes data to the buffer memory of the E71.
Extend Unit	Read	0601	0000	Reads buffer memory data of the intelligent function module.
	Write	1601	0000	Writes data to the buffer memory of the intelligent function module.
Remote Control	Remote Run	1001	0000	Executes remote RUN to the CPU module where an E71 is mounted.
	Remote Stop	1002	0000	Executes remote STOP to the CPU module where an E71 is mounted.
	Remote Pause	1003	0000	Executes remote PAUSE to the CPU module where an E71 is mounted.
	Remote Latch Clear	1005	0000	Executes remote latch clear to the CPU module where an E71 is mounted.
	Remote Reset	1006	0000	Executes remote RESET to the CPU module where an E71 is mounted.
	Read Type Name	0101	0000	Reads the model name and code of the CPU module where an E71 is mounted.
Remote Password	Lock	1631	0000	Specifies the remote password to disable the communication with other devices. (The locked state is activated from the unlocked state.)
	Unlock	1630	0000	Specifies the remote password to enable communication with other devices. (The unlocked state is activated from the locked state.)

Item		Command	Subcommand	Description
Type	Operation			
File	Read Directory/File	1810	0000	Reads file list information from the CPU module where an E71 is mounted.
	Search Directory/File	1811	0000	Reads the file number of the specified file from the CPU module where an E71 is mounted.
	New File	1820	0000	Reserves storage area for the specified file of the CPU module where an E71 is mounted.
	Delete File	1822	0000	Deletes a file in the CPU module where an E71 is mounted.
	Copy File	1824	0000	Copies the specified file in the CPU module where an E71 is mounted.
	Change File State	1825	0000	Changes the attributes of the file in the CPU module where an E71 is mounted.
	Change File Date	1826	0000	Changes the creation date of the file in the CPU module where an E71 is mounted.
	Open File	1827	0000	Locks a file so that the data of the file is not changed by other devices in the CPU module where an E71 is mounted.
	Read File	1828	0000	Reads the data of a file from the CPU module where an E71 is mounted.
	Write File	1829	0000	Writes the data to a file in the CPU module where an E71 is mounted.
	Close File	182A	0000	Cancels the file lock by the open process.
Self Test		0619	0000	Tests whether the communication with the E71 is normally performed or not.
Clear Error		1617	0000	Turns off the COM.ERR. LED on the E71.

CHAPTER 11 DATA COMMUNICATIONS USING THE PREDEFINED PROTOCOL

Data can be transferred between a connected device and the CPU module with a protocol appropriate to the connected device (such as a measuring instrument or a bar code reader).
Device or buffer memory data can be taken into communication packets, and thereby this protocol communications are suitable for data that may change in each communication.
The protocol setting required to communicate with the connected device can be configured in GX Works2.
Protocols can be selected from the Predefined Protocol Library or can be created and edited.
This function is available only in the QJ71E71-100 with the serial number (first five digits) of "15042" or later.



For the protocol setting method, refer to the following.

📖 GX Works2 Version 1 Operating Manual (Intelligent Function Module)

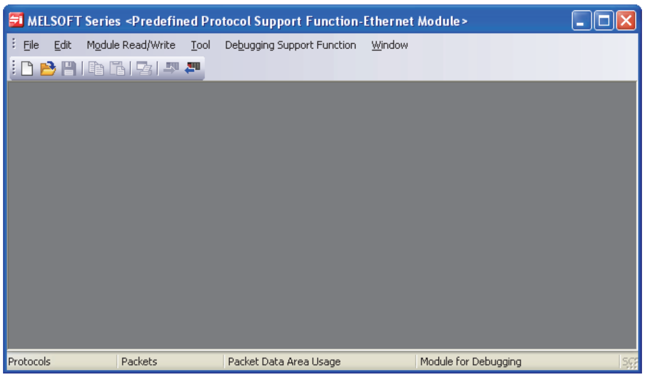
Point

- The followings are the maximum numbers of protocols and packets that can be registered.
  - Protocols: Up to 128
  - Packets: Up to 256
  - Packet data area size: Up to 12288 bytes

  If once the number of packets reaches the upper limit, protocols cannot be added even though the number of protocols has not reached the upper limit.
  In addition, if once the packet data area size reaches the upper limit, protocols and packets cannot be added even though the numbers of protocols and packets have not reached the upper limit.
  For details, refer to the following.
  GX Works2 Version 1 Operating Manual (Intelligent Function Module)
- From Connection No.1 to 16 are available.
- The Communication Data Code becomes Binary Code regardless of the setting.

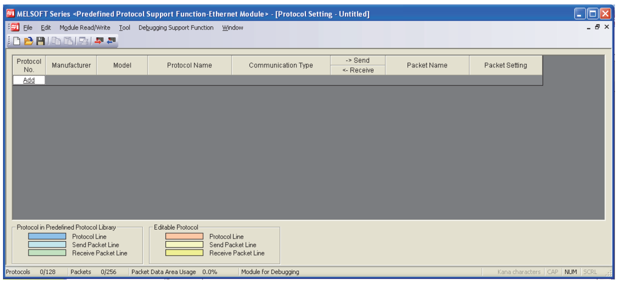
11.1 Data Communication Procedure

By using the predefined protocol support function, data can be communicated with the connected device in the following procedure.



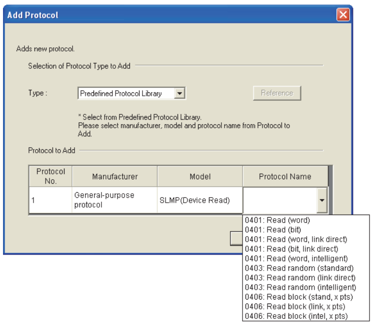
1. Display the "Predefined Protocol Support Function" window.

[Tool] ➪ [Predefined Protocol Support Function] ➪ [Ethernet Module...]



2. Create a new file.

[File] ➪ [New...] ➪ "Add"



3. Select "Predefined Protocol Library" or "Add New" in the "Add Protocol" window.
When "Predefined Protocol Library" is selected, select the protocol from Predefined Protocol Library registered in GX Works2.

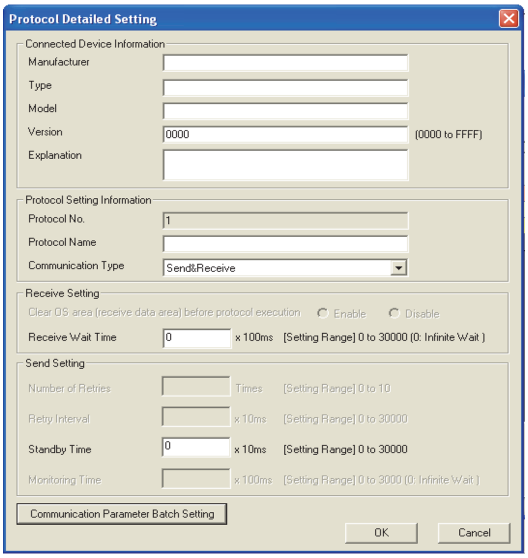
Point

If "User Protocol Library" is selected in the "Add Protocol" window, protocols saved by user can be read.
For details on the User Protocol Library, refer to the following.
GX Works2 Version 1 Operating Manual (Intelligent Function Module)

4. Set the items required for the data communications.

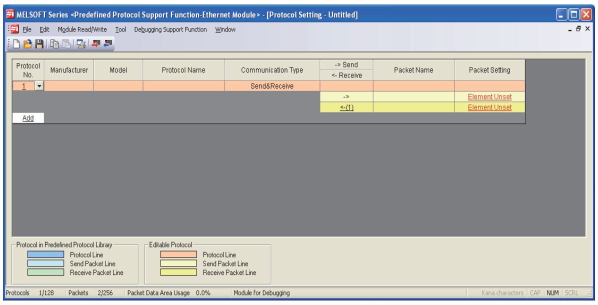
- Set communication parameters in the "Protocol Detailed Setting" window.

  "Protocol Detailed Setting" window ➪ Select a protocol ➪ [Edit] ➪ [Protocol Detailed Setting...]

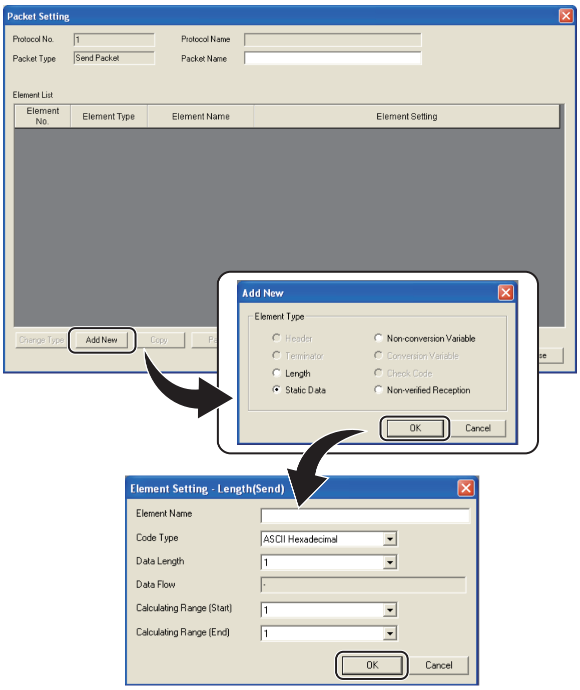


- Set the configuration of packets to be sent and received in the "Packet Setting" window.

  "Protocol Detailed Setting" window ➪ [Variable Unset] or [Element Unset]



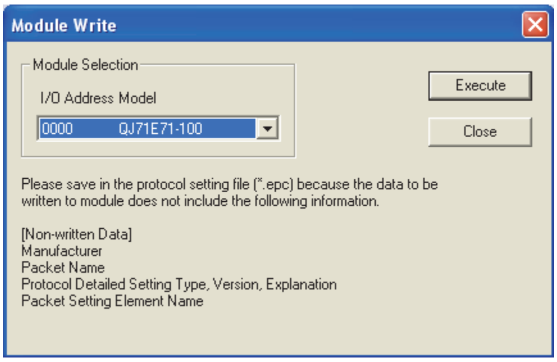
↓



5. Write the protocol setting data to the flash ROM.

[Online] ⇨ [Write to PLC...]

Select a target module and write the protocol setting data to the flash ROM.*1



*1 When the initial process is not completed, the protocol setting data cannot be written to the flash ROM. Before writing the data, set the network parameters and check that the initial process is completed.
(The initial process completion can be checked using Initial normal completion signal (X19).)

Point

- When writing the data to the Redundant CPU, pay attention to the followings:
  - To write the data, connect directly to the CPU module or connect via an intelligent function module on a main base unit; connection via an intelligent function module on an extension base is not available.
  - Select "Not specified" for Target System in the Transfer Setup of GX Works2. When the Target System is specified, an error occurs in the CPU module.
  - A system switching during data writing may cause the data not to be written to the flash ROM.
    Change the operation mode of the Redundant CPU to separate mode before writing to prevent such situation from occurring.
    If a system switching has occurred during data writing, write the data to the flash ROM again.
- The following data cannot be read out because they are not written to the E71. However, when the protocols are selected from the Predefined Protocol Library, the followings can be displayed.
  - Manufacturer
  - Packet Name
  - Protocol Detailed Setting Type, Version, Explanation
  - Packet Setting Element Name
- Module Selection using the Predefined Protocol Support Function is available only when "Not Specification" is set in the Transfer Setup of GX Works2.
  When other station is specified, the E71 specified in the Transfer Setup of GX Works2 becomes the target module.

6. Execute the protocol by using a dedicated instruction.

Execute the protocol written to the flash ROM by using the dedicated instruction (ECPRTCL instruction).

Point

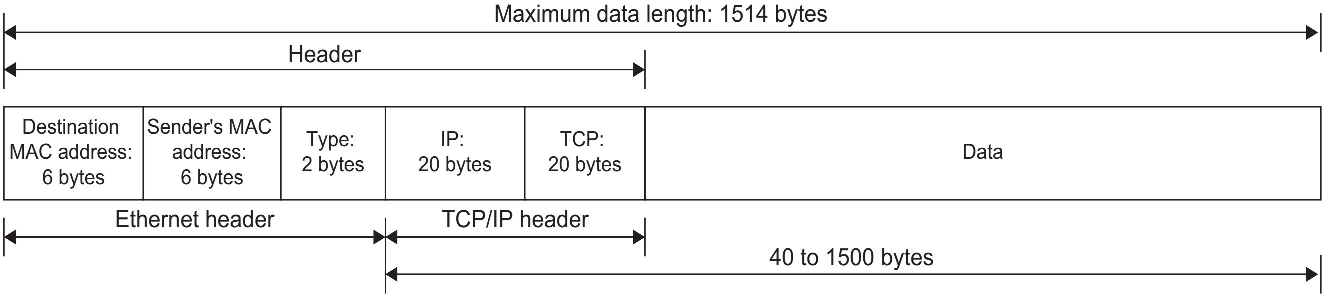
- Protocols can be executed only when the target connection is in the following status:
  - When Open completion signal is on
  - When "Predefined protocol" is set to Fixed Buffer Communication in the open setting

  If the protocol is executed to the connection where the above conditions are not satisfied, the ECPRTCL instruction ends abnormally.

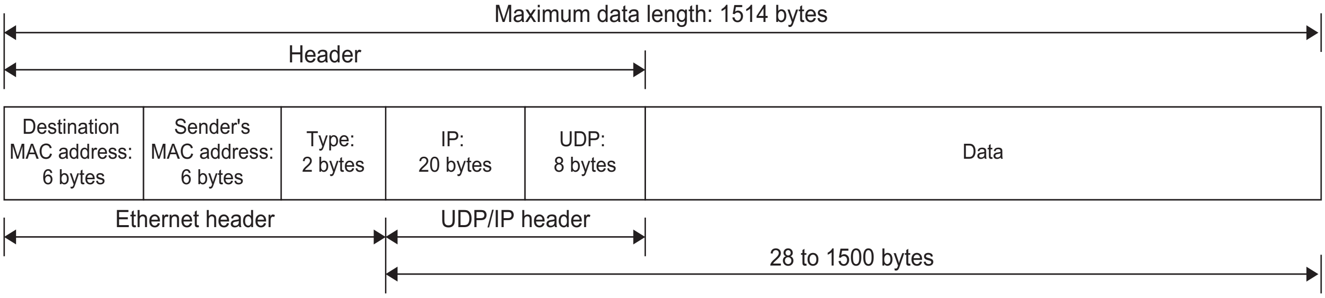
11.2 Communication Type of Protocols

Receive and send packets from/to the connected device for the process execution are registered in a protocol. Packet elements set using the predefined protocol support function correspond to the data part of the send/receive packets. The following shows an example of packet configuration. For details on the packet elements, refer to Page 387, Appendix 8.3.

(a) TCP/IP



(b) UDP/IP



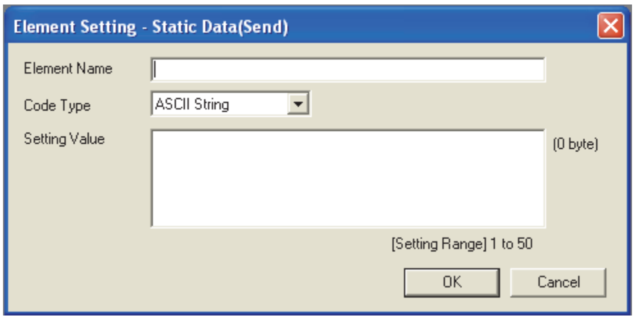
Communications with the connected device using the predefined protocol function are performed with the following procedure (communication types). For the operations of each communication type, refer to Page 380, Appendix 8.1.

Communication type name	Description
Send Only	Sends a send packet once.
Receive Only	Receives a packet if it matches any of up to 16 defined receive packets.
Send&Receive	Sends a send packet, then receives a packet if it matches any of up to 16 defined receive packets.

11.3 Packet Elements

A packet consists of packet elements.
Up to 32 elements can be placed in a packet, and the maximum data length is 2046 bytes per packet.
The following shows the details of the packet elements.
For examples of packet element data, refer to Page 387, Appendix 8.3.

(1) Static Data



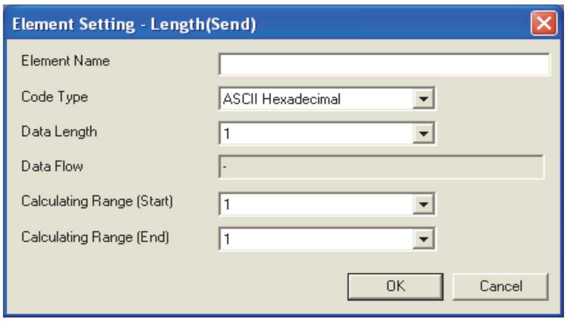
This element is used when a specific code or character string such as a command exists in a packet.

- When sending: The specified code and character string are sent.
- When receiving: Receive data are verified.

Multiple Static Data elements can be placed anywhere in the data part.
The following table lists the items.

Item	Description	Remark
Element Name	Set name of the element.	-
Code Type	Select a data type of the setting value. ASCII String/ASCII Control Code/HEX	-
Setting Value	Set data within 1 to 50 bytes. Code type and setting range are as follows: • ASCII String: 20_H to $7F_H$ • ASCII Control Code: Control code of 00_H to $1F_H$ and $7F_H$ • HEX: Hexadecimal data of 00_H to FF_H	Setting example ASCII String: "ABC" ASCII Control Code: STX HEX: FFFF

(2) Length



This element is used when an element indicating the data length is included in a packet.

- When sending: The data length of the specified range is calculated and the result is added to a send packet.
- When receiving: The data (setting value) corresponds to the Length in the receive data is verified as the data length of the specified range.

Length elements can be placed anywhere in the data part.

The following table lists the items.

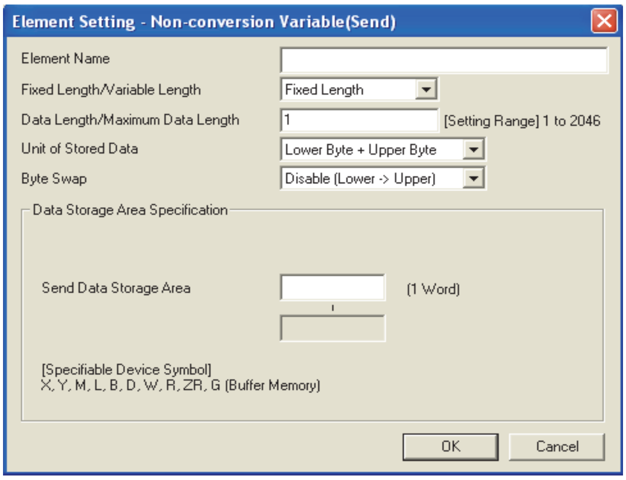
Item		Description		Remark
Element Name		Set name of the element.		-
Code Type		Select a format of the data length. ASCII Hexadecimal/HEX		-
Data Length		Select the data length on the line. The range is 1 to 4.		-
Data Flow		Forward Direction (Upper Byte→Lower Byte)	When sending: The calculated Length is sent from the upper byte. When receiving: The data is received from the upper byte.	Not settable when Data Length is 1 byte
		Reverse Direction (Lower Byte→Upper Byte)	When sending: The calculated Length is sent from the lower byte. When receiving: The data is received from the lower byte.	
		Byte Swap (by Word)*1	When sending: The calculated Length is sent swapping the upper byte and lower byte by word. When receiving: The data is received swapping the upper byte and lower byte by word.	
Calculating Range	Start	Select the start element number of the calculating range. The range is 1 to 32.		-
	End	Select the end element number of the calculating range. The range is 1 to 32.		

*1 Available only when the Data Length is 4 bytes.

Point

- Multiple Length elements can be placed in a packet.
- When there is no element other than a Length element, an element error occurs.
  (To use a Length element, one or more element(s) other than Length is/are required.)
- When the number of digits of calculation result is greater than that specified in "Data Length", digits greater than the specified digit are omitted (ignored).
  For example, if the data length is 2 bytes and the data size calculation results are "123" bytes, the data length will be "23".
- When a Non-conversion Variable (Variable length) or Non-verified Reception (Variable number of characters) is placed behind a Length and is not included in the Length calculating range, place Static Data immediate after the Non-conversion Variable or Non-verified Reception.
- When "Code Type" is "ASCII Hexadecimal", a corresponding packet is regarded as a mismatch packet if a string except for "0" to "9", "A" to "F", and "a" to "f" is received.
- When sending data converted to ASCII characters, use "0" to "9" or "A" to "F".
- When multiple Length elements are placed, the calculating range cannot be set as follows:
  - A Length calculating range overlaps the others.
  - The calculating range of a Length is greater than that of the Length placed before.
- Length element cannot be placed at the end of the packet elements.

(3) Non-conversion Variable



This element is used to send data in the device memory of a CPU module or buffer memory as a part of a send packet, or to store a part of a receive packet to the device memory of a CPU module device or buffer memory. The following table lists the items.

Item	Description	
Element Name	Set name of the element.	
Fixed Length/Variable Length	Fixed Length	Sends and receives the data whose length is fixed.
	Variable Length	When sending: The data length at the time of the protocol execution is specified and the data is sent. When receiving: The data of which the length is variable is received.
Data Length/Maximum Data Length	Set the length of data to be sent and received. (For the variable length data, set the maximum data length that can be specified to the data length storage area.) The range is 1 to 2046.	
Unit of Stored Data	Lower Byte + Upper Byte	When sending: Each one word (2 bytes) data in the data storage area is sent in the order of the lower byte to the upper byte. When receiving: The receive data is stored to the data storage area in the order of the lower byte to the upper byte.
	Lower Bytes Only	When sending: Each lower byte data in the data storage area is sent. The E71 ignores data in the upper byte. When receiving: The receive data is stored to each lower byte in the data storage area. The E71 stores 00_H in the upper byte.
Byte Swap	Disable/Enable	When sending: When "Enable" is selected, data in the upper byte and lower byte are swapped by word (2 bytes) and sent. When "Unit of Stored Data" is "Lower Byte + Upper Byte" and "Data Length" is an odd number of bytes, the upper byte is sent at transmission of the last byte. When "Unit of Stored Data" is "Lower Bytes Only" and "Data Length" is an odd number of bytes, data without any byte swap is sent at transmission of the last byte. When receiving: When "Enable" is selected, data in the upper byte and lower byte are swapped by word (2 bytes) and sent. When "Unit of Stored Data" is "Lower Byte + Upper Byte" and "Data Length" is an odd number of bytes, the last byte is stored to the upper byte. When "Unit of Stored Data" is "Lower Bytes Only" and "Data Length" is an odd number of bytes, the last byte is stored without any byte swap.

Item	Description
Data Storage Area Specification	Specify a start device to store variable value. Available devices are as follows: • Internal user*1*2 Input relay (X) Output relay (Y) Internal relay (M) Latch relay (L) Link relay (B) Data register (D) Link register (W) • File register*2 File register (R, ZR) • Buffer memory G device (G) (Send/receive area for predefined protocol support function (address: 4800_H to $4FFF_H$)) *1: Do not set local devices. *2: Set within the device range specified in the "Device" window of PLC Parameter.

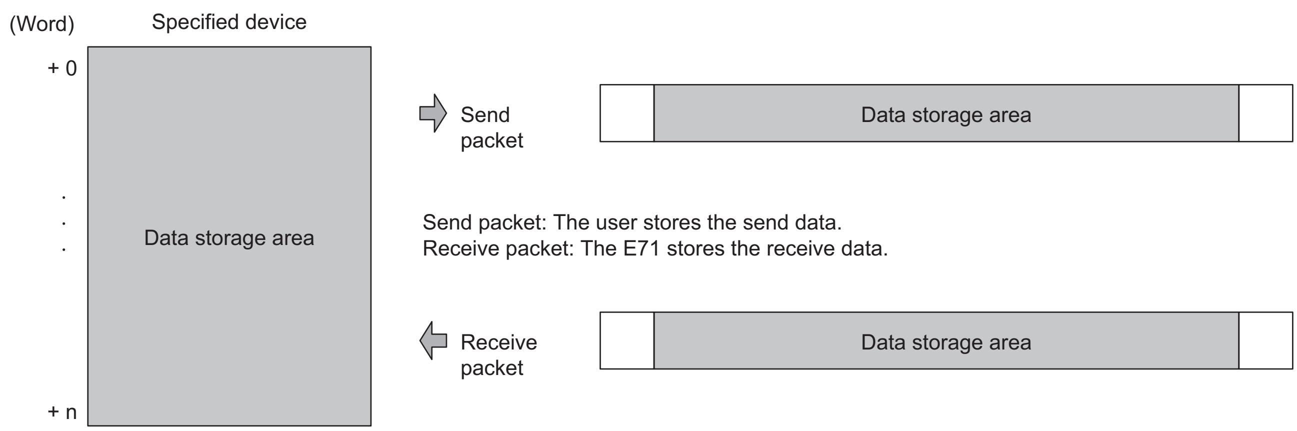
The following describes the configuration of the data storage area.

(a) When "Fixed Length/Variable Length" is "Fixed Length"

An area starting from the device number which is specified in the "Element Setting" window is considered as the data storage area.

The data storage area to be occupied varies depending on the setting of "Unit of Stored Data".

- When "Lower Byte + Upper Byte" is selected, the same size as the data length is occupied. (However, when the data length of a send packet is an odd number, the upper byte (lower byte for "Byte Swap") of the last device is not sent. When the data length of a receive packet is an odd number, the last data is stored with one byte of 00_H.)
- When "Lower Bytes Only" is selected, twice the size of the data length is occupied.

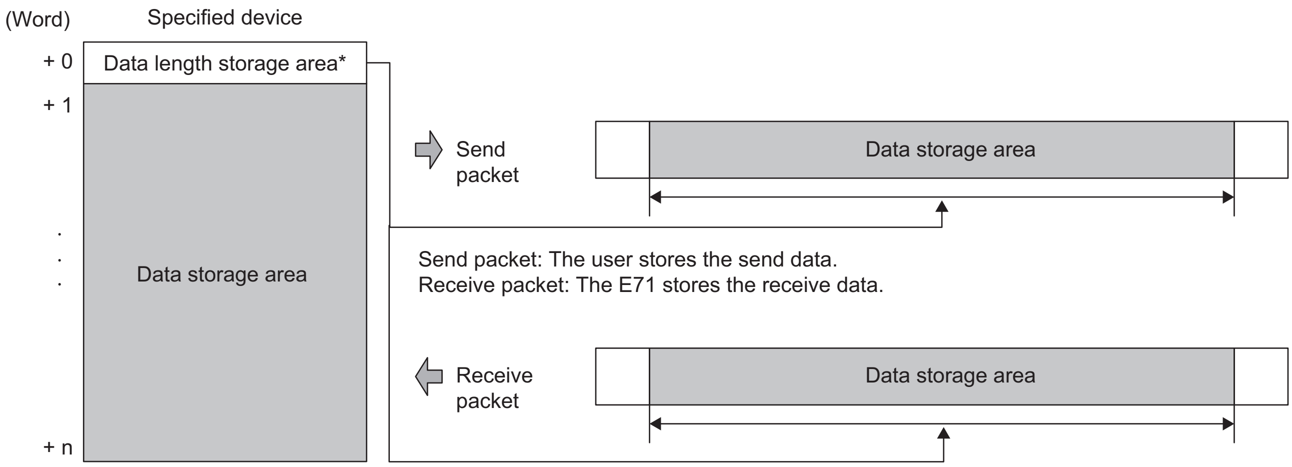


(b) When "Fixed Length/Variable Length" is "Variable Length"

An area starting from the device number which is specified in the "Element Setting" window +1 is considered as the data storage area.

The data storage area to be occupied varies depending on the setting of "Unit of Stored Data".

- When "Lower Byte + Upper Byte" is selected, the same size as the data length + one word (length for the data length storage area) are occupied. (However, when the data length of a send packet is an odd number, the upper byte (lower byte for "Byte Swap") of the last device is not sent. When the data length of a receive packet is an odd number, the last data is stored with one byte of 00_H.)
- When "Lower Bytes Only" is selected, twice size of the data length + one word (length for the data length storage area) are occupied.

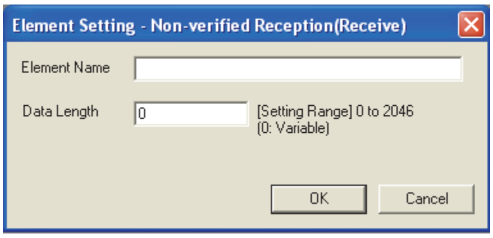


*: The unit of the data length is fixed to byte.

Point

- Multiple Non-conversion Variable elements can be placed in a packet.
- When "Fixed Length/Variable Length" is "Variable Length" and the configuration is set as follows, an error occurs:
  - An element other than Static Data is placed behind a Non-conversion Variable element when Non-conversion Variable is out of the Length calculating range or when there is no Length element (except for when Non-conversion Variable is placed at the end of the packet elements).
  - Multiple Non-conversion Variable elements are placed in the Length calculating range, while a Length element is not placed.
  - A Non-conversion Variable element is placed before a Length element in the Length calculating range.

(4) Non-verified Reception



This element is used when receive data include unnecessary data.
The E71 skips characters as many as the specified number if a receive packet includes a Non-verified Reception.
The following table lists the items.

Item	Description		Remark
Element Name	Set name of the element.		-
Data Length	0 (Variable number of characters)	Set this item when the number of characters not to be verified varies in each communication.	-
	1 to 2046 (Specified number of characters)	Set the number of characters not to be verified.	

Point

- Multiple Non-verified Reception elements can be placed in a packet.
- When "Data Length" is set to 0 and the configuration is set as follows, an error occurs:
  - An element other than Static Data is placed behind a Non-verified Reception element when Non-verified Reception is out of the Length calculating range or when there is no Length element (except for when Non-verified Reception is placed at the end of the packet elements).
  - Multiple Non-verified Reception elements are placed in the Length calculating range, while a Length element is not placed.
  - A Non-verified Reception element is placed before a Length element in the Length calculating range.

11.4 Execution Condition of Predefined Protocol Communication

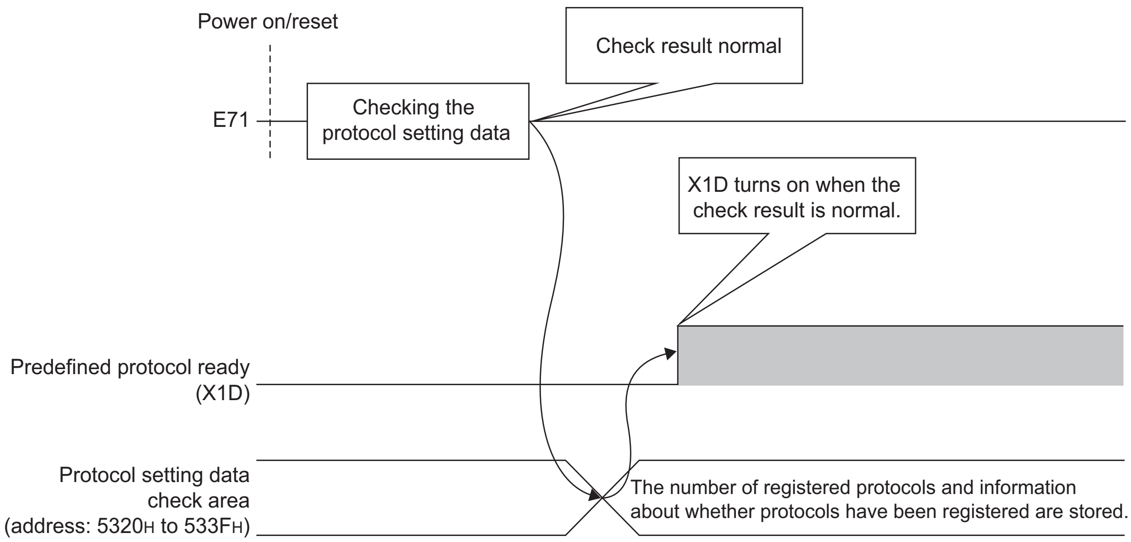
Communications using the predefined protocol can be executed when Predefined protocol ready (X1D) is on.
This section describes the operations of Predefined protocol ready (X1D).

(1) When the power supply is on or reset

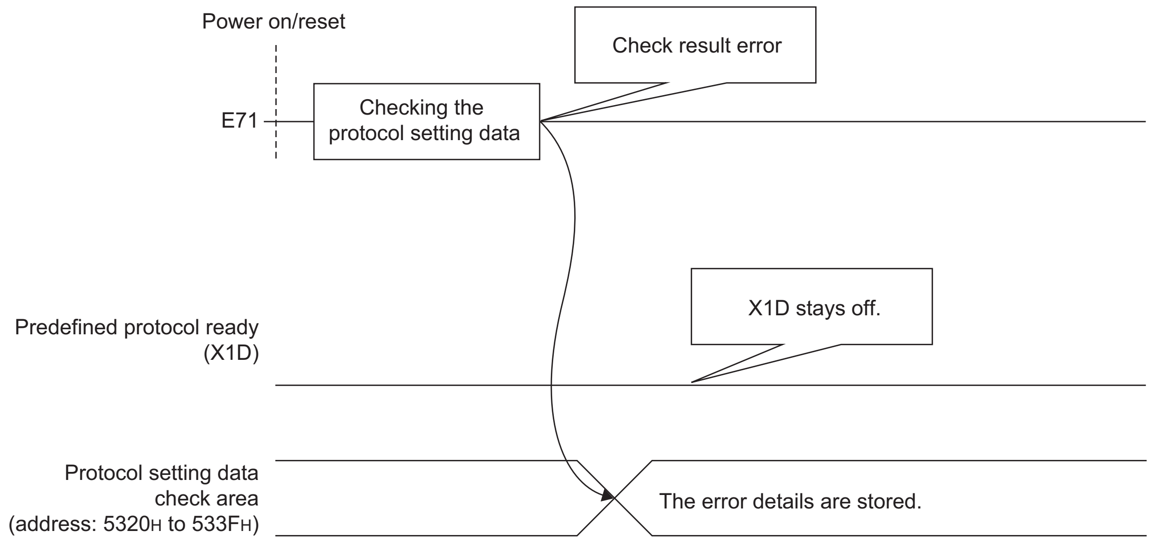
The E71 checks the written protocol setting data when the power supply is on or reset.
If the protocol setting data is set correctly, the E71 turns on Predefined protocol ready (X1D) then the protocols can be executed.
Predefined protocol ready (X1D) is used as an interlock signal when the protocols are executed.
If an error has occurred in the protocol setting data, Predefined protocol ready (X1D) stays off and the error details are stored to the protocol setting data check area (address: 5320_H to $533F_H$).
If no protocol setting data has been set, the E71 does not check the data and Predefined protocol ready (X1D) stays off.

The information about whether protocol setting data has been registered can be checked in Number of registered protocols (address: 5328_H) and Protocol registration (address: 5330_H to $533F_H$).

(a) If the protocol setting data is set correctly



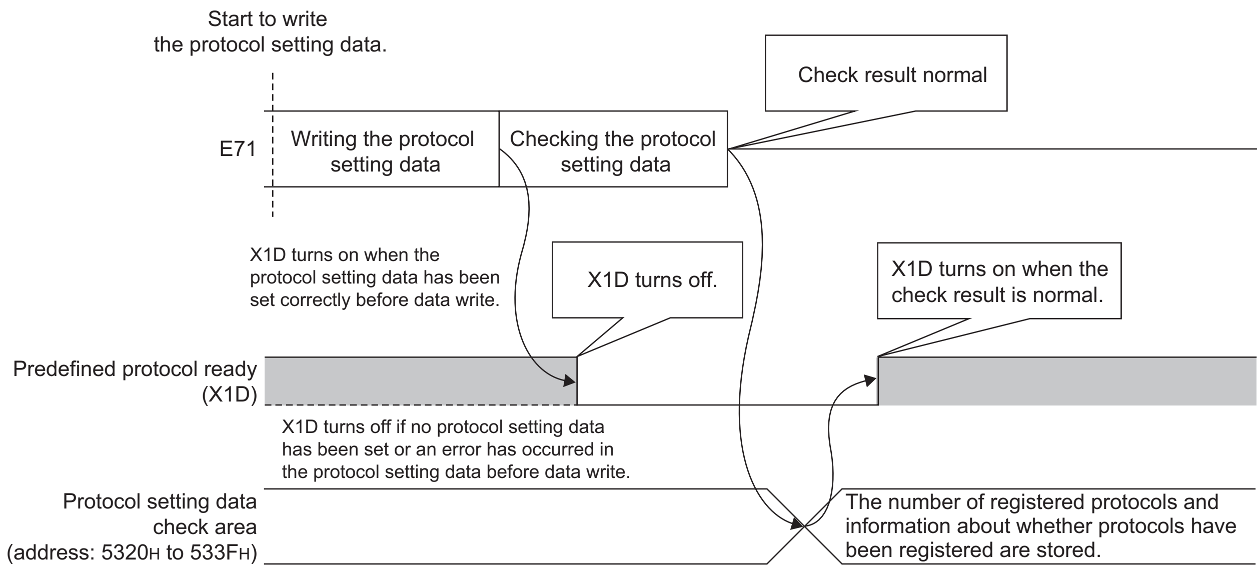
(b) If an error has occurred in the protocol setting data



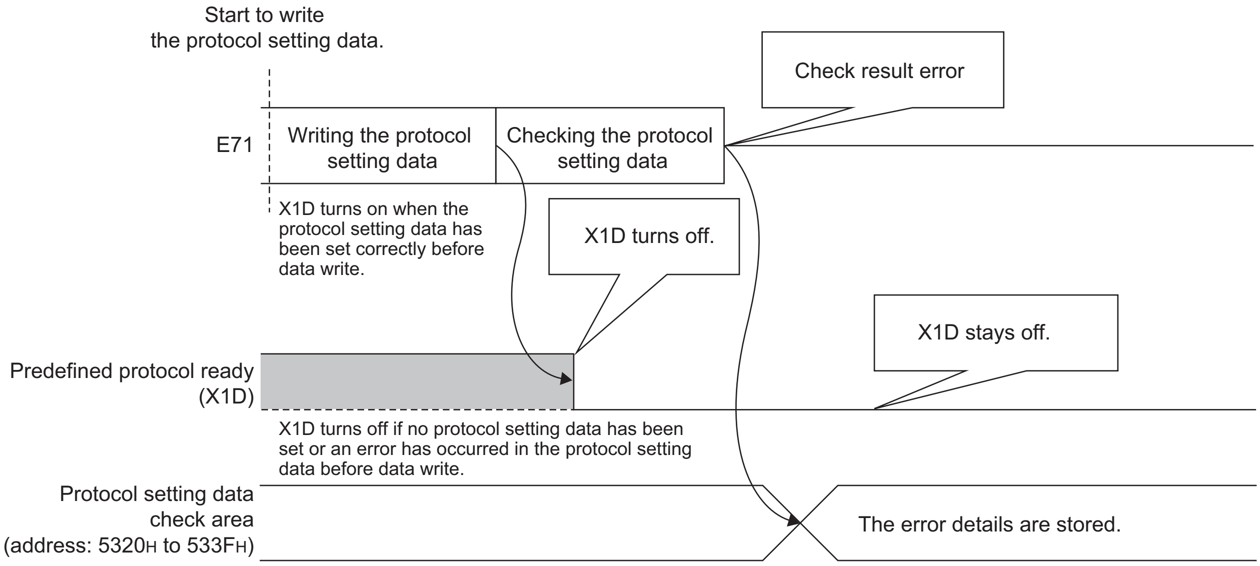
(2) When the protocol setting data is written

When the protocol setting data writing from GX Works2 is completed, Predefined protocol ready (X1D) turns off. After the completion, the E71 checks the protocol setting data. If the protocol setting data is set correctly, the E71 turns on Predefined protocol ready (X1D).
If an error has occurred in the protocol setting data, Predefined protocol ready (X1D) stays off and the error details are stored to the protocol setting data check area (address: 5320_H to $533F_H$).

(a) If the protocol setting data is set correctly



(b) If an error has occurred in the protocol setting data



(3) When the UINI instruction is being executed

The E71 does not check the protocol setting data when the UINI instruction is being executed.
Predefined protocol ready (X1D) keeps its status even when the instruction is executed.

(4) The operations of X1D and COM.ERR.LED

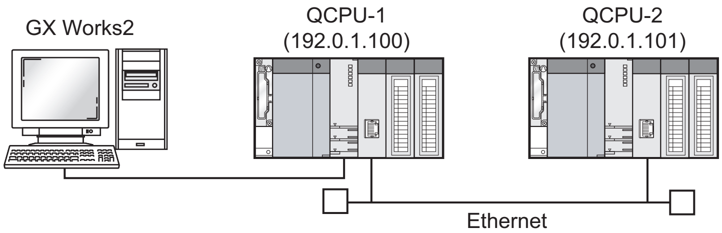
If an error has occurred in the protocol setting data when the power supply is on or reset, or at the data check after the data writing, Protocol setting data error (error code: $C402_H$) occurs.
Predefined protocol ready (X1D) turns off and the COM.ERR.LED turns on at this time.

11.5 Example of Predefined Protocol Communications

This section describes an example of communications using the predefined protocol under the following system configuration.

11.5.1 System configuration example

(1) System configuration



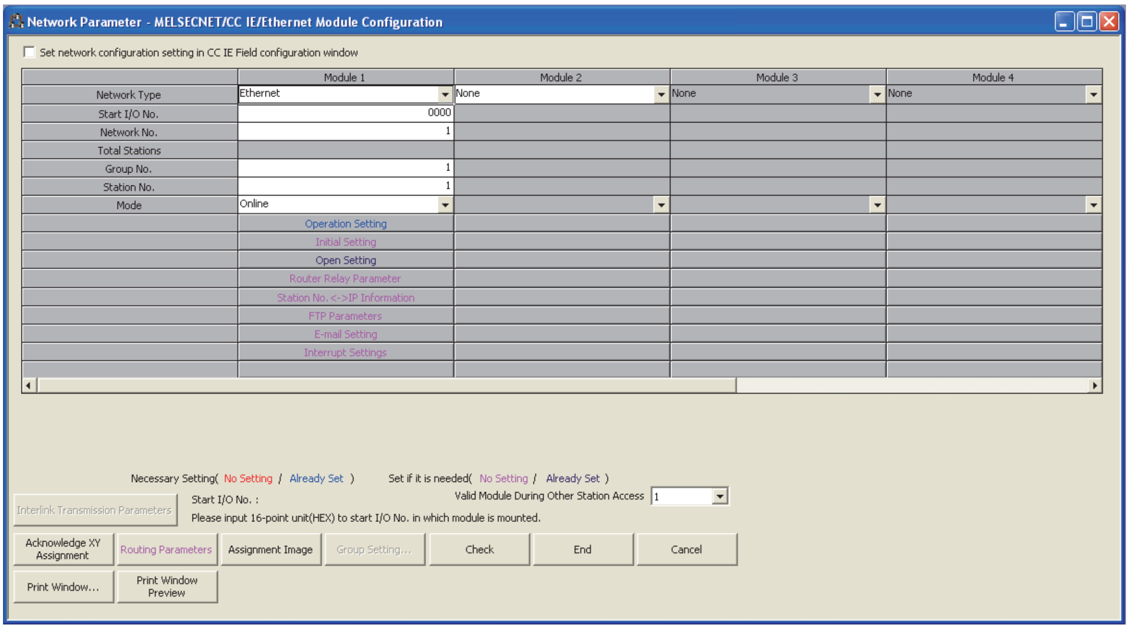
11.5.2 Parameter Setting

(1) Sending side (QCPU-1 side)

The following is an example of the parameter settings of the sending side (QCPU-1 side).

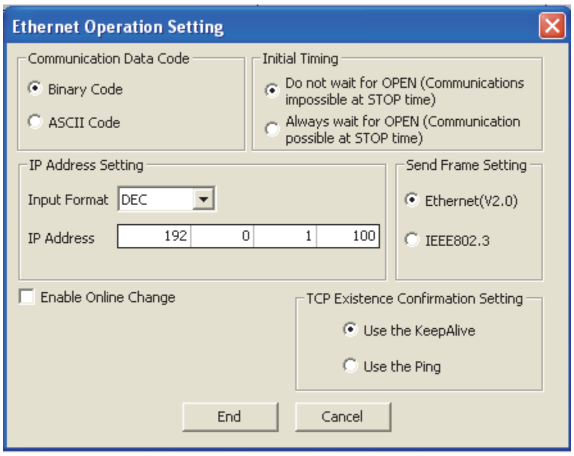
(a) Basic setting

The following is an example of the basic setting.



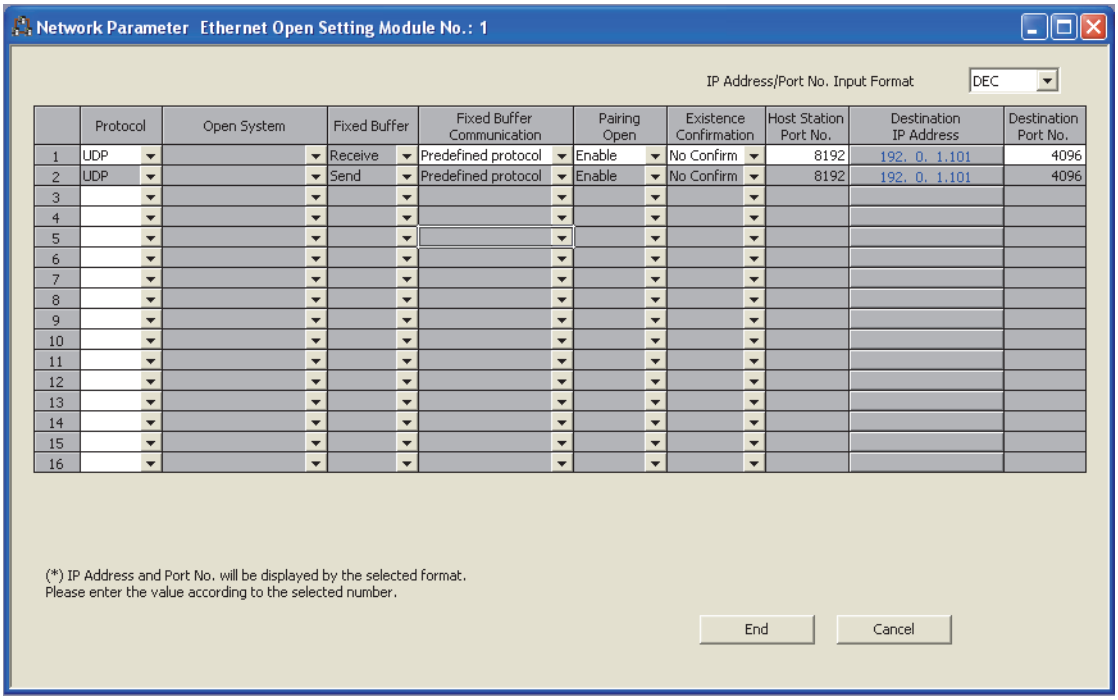
(b) Ethernet operation setting

The following is an example of the Ethernet operation setting.



(c) Open setting

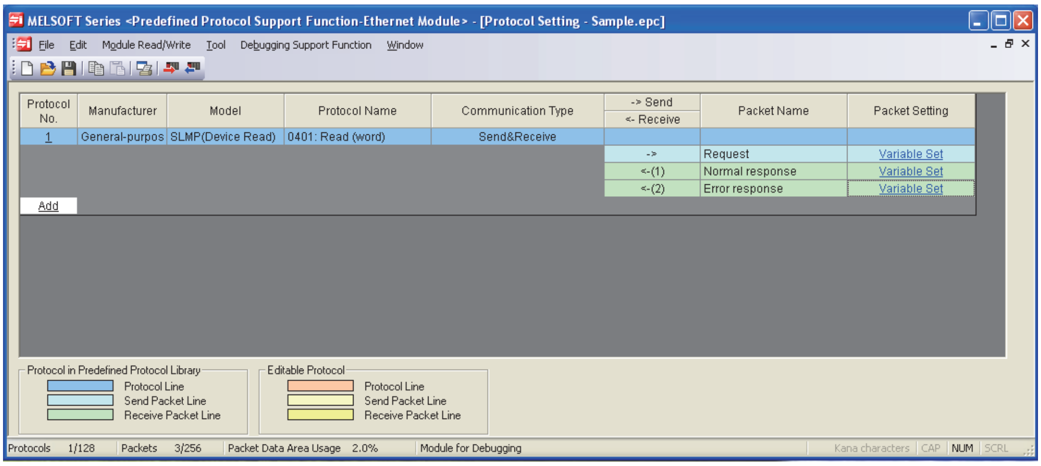
The following is an example of the open setting.



(d) Protocol setting data

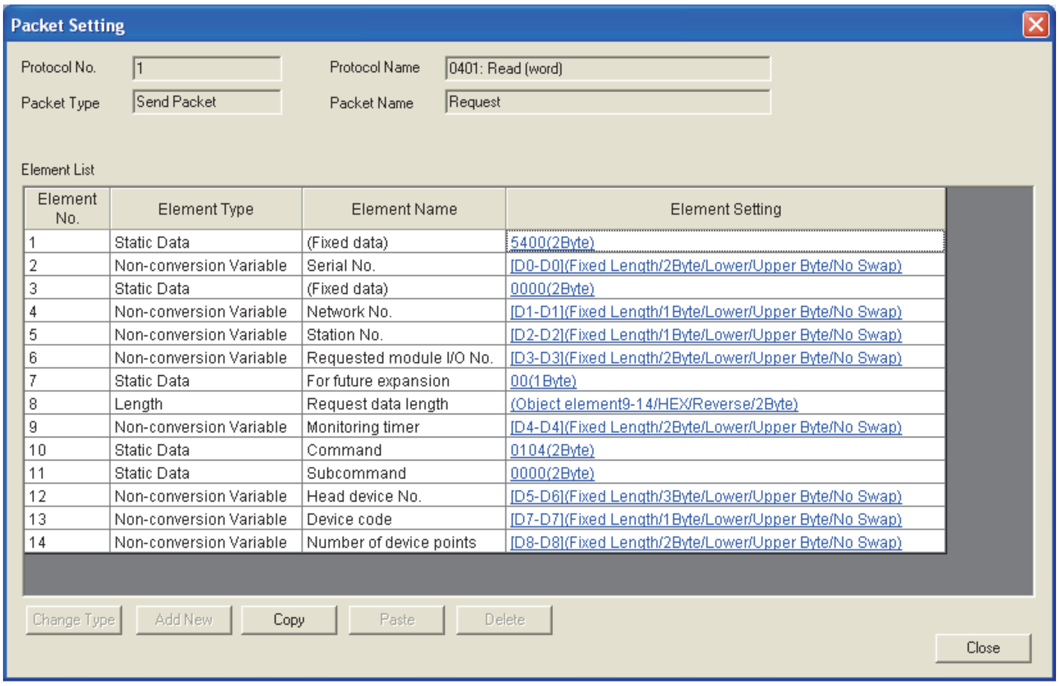
The value in D100 to D109 of the destination station can be read by using SLMP (Device Read) command in the Predefined Protocol Library.
The followings are setting examples of the protocol setting data using the Predefined Protocol Support Function.

[Protocol Setting]

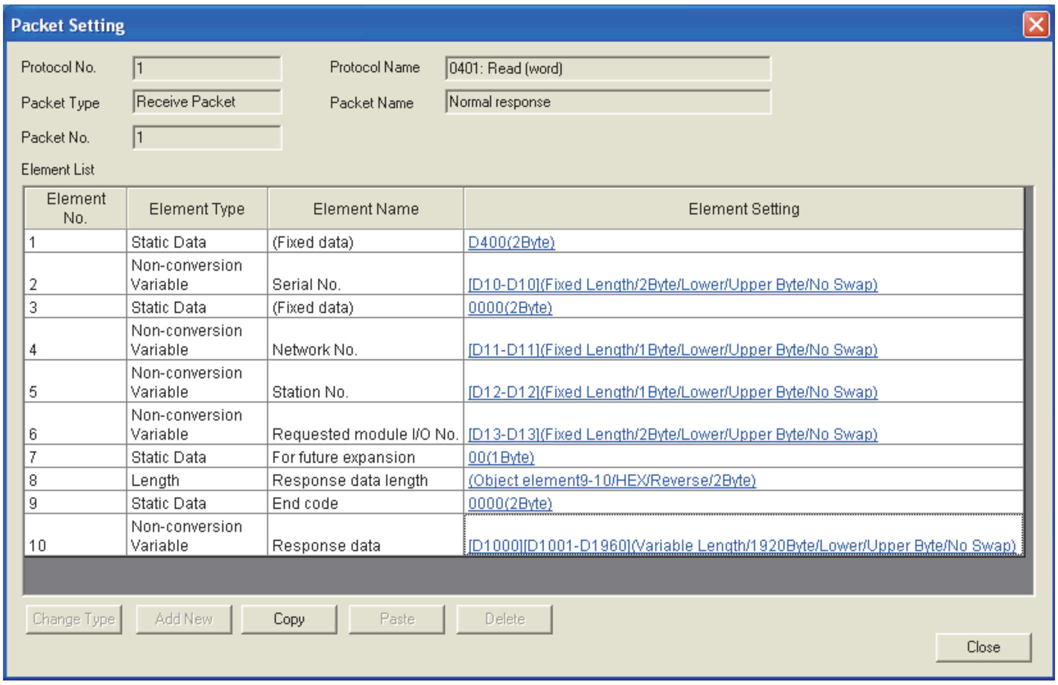


[Packet Setting]

- Send Packet (Request)



• Receive Packet (Normal response)



Packet Setting

Protocol No. 1 | Protocol Name 0401: Read (word)

Packet Type Receive Packet | Packet Name Normal response

Packet No. 1

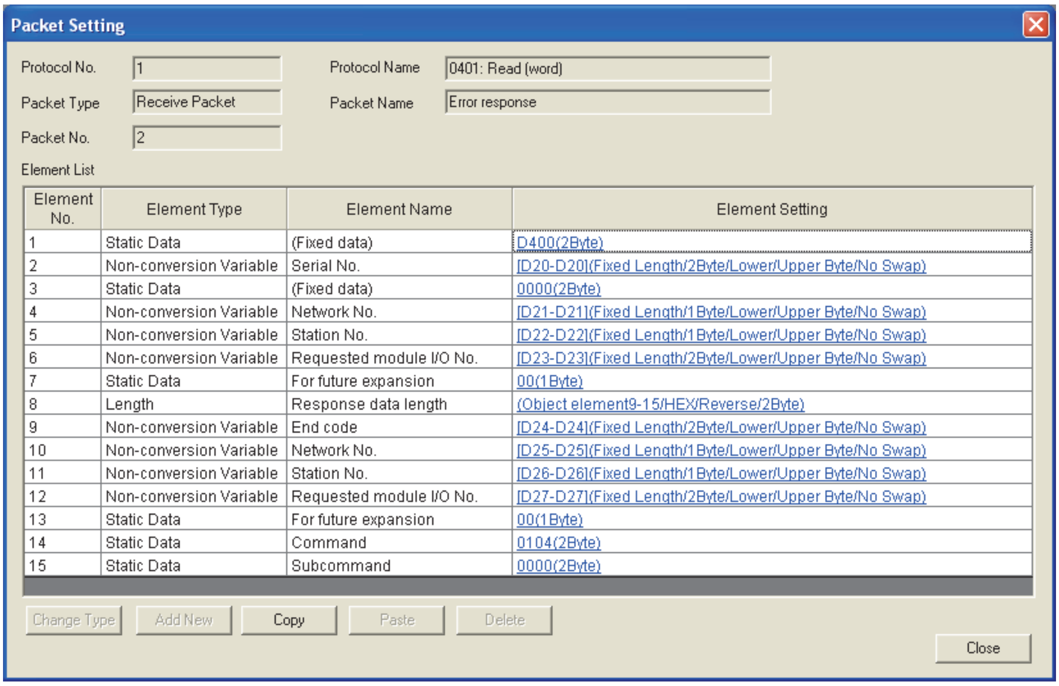
Element List

Element No.	Element Type	Element Name	Element Setting
1	Static Data	(Fixed data)	D400(2Byte)
2	Non-conversion Variable	Serial No.	[D10-D10](Fixed Length/2Byte/Lower/Upper Byte/No Swap)
3	Static Data	(Fixed data)	0000(2Byte)
4	Non-conversion Variable	Network No.	[D11-D11](Fixed Length/1Byte/Lower/Upper Byte/No Swap)
5	Non-conversion Variable	Station No.	[D12-D12](Fixed Length/1Byte/Lower/Upper Byte/No Swap)
6	Non-conversion Variable	Requested module I/O No.	[D13-D13](Fixed Length/2Byte/Lower/Upper Byte/No Swap)
7	Static Data	For future expansion	00(1Byte)
8	Length	Response data length	(Object element9-10/HEX/Reverse/2Byte)
9	Static Data	End code	0000(2Byte)
10	Non-conversion Variable	Response data	[D1000][D1001-D1960](Variable Length/1920Byte/Lower/Upper Byte/No Swap)

Change Type | Add New | Copy | Paste | Delete

Close

• Receive Packet (Error response)



Packet Setting

Protocol No. 1 | Protocol Name 0401: Read (word)

Packet Type Receive Packet | Packet Name Error response

Packet No. 2

Element List

Element No.	Element Type	Element Name	Element Setting
1	Static Data	(Fixed data)	D400(2Byte)
2	Non-conversion Variable	Serial No.	[D20-D20](Fixed Length/2Byte/Lower/Upper Byte/No Swap)
3	Static Data	(Fixed data)	0000(2Byte)
4	Non-conversion Variable	Network No.	[D21-D21](Fixed Length/1Byte/Lower/Upper Byte/No Swap)
5	Non-conversion Variable	Station No.	[D22-D22](Fixed Length/1Byte/Lower/Upper Byte/No Swap)
6	Non-conversion Variable	Requested module I/O No.	[D23-D23](Fixed Length/2Byte/Lower/Upper Byte/No Swap)
7	Static Data	For future expansion	00(1Byte)
8	Length	Response data length	(Object element9-15/HEX/Reverse/2Byte)
9	Non-conversion Variable	End code	[D24-D24](Fixed Length/2Byte/Lower/Upper Byte/No Swap)
10	Non-conversion Variable	Network No.	[D25-D25](Fixed Length/1Byte/Lower/Upper Byte/No Swap)
11	Non-conversion Variable	Station No.	[D26-D26](Fixed Length/1Byte/Lower/Upper Byte/No Swap)
12	Non-conversion Variable	Requested module I/O No.	[D27-D27](Fixed Length/2Byte/Lower/Upper Byte/No Swap)
13	Static Data	For future expansion	00(1Byte)
14	Static Data	Command	0104(2Byte)
15	Static Data	Subcommand	0000(2Byte)

Change Type | Add New | Copy | Paste | Delete

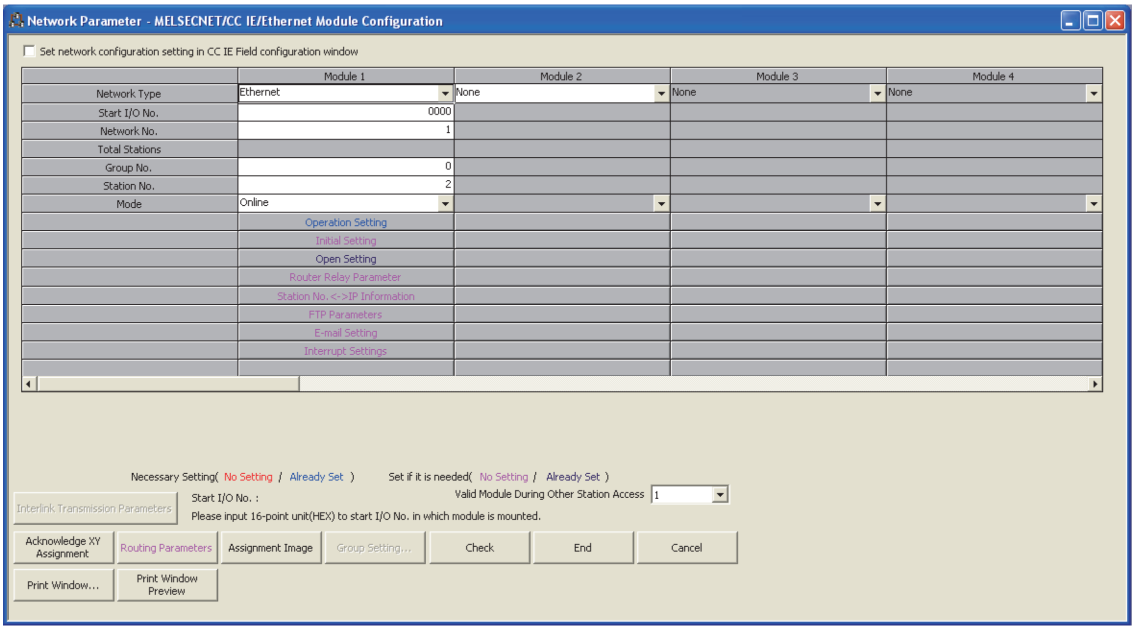
Close

(2) Receiving side (QCPU-2 side)

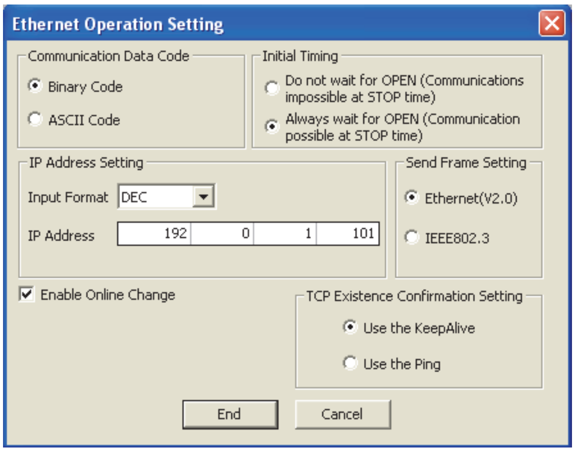
The following is an example of the parameter settings of the receiving side (QCPU-2 side).

(a) Basic setting

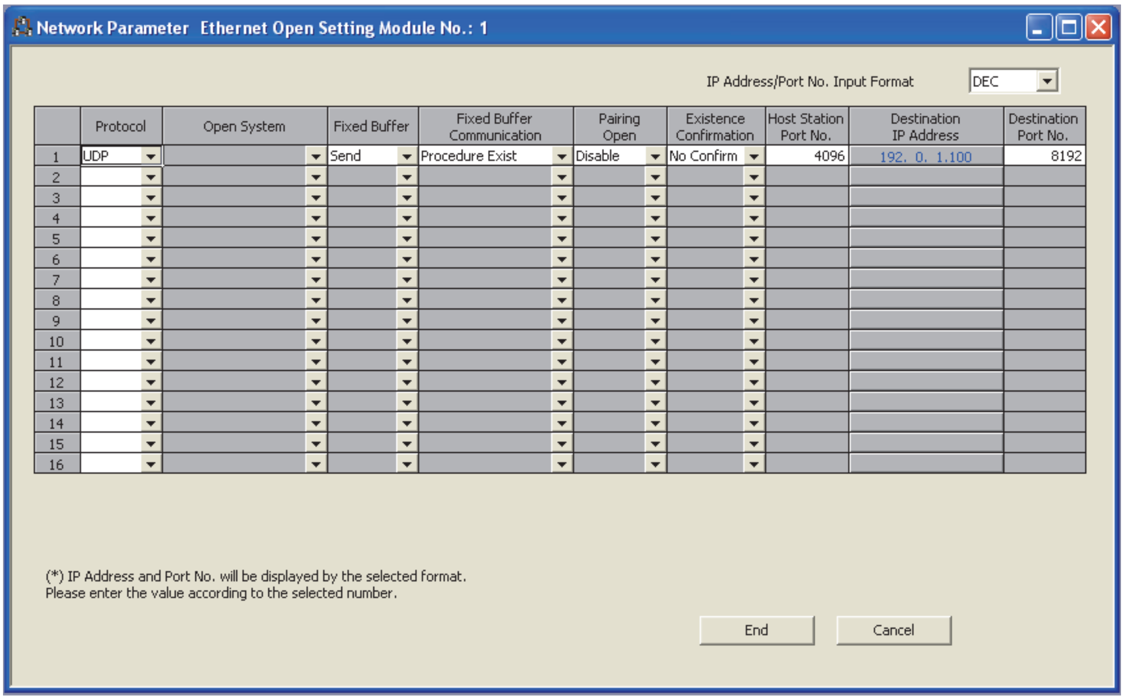
The following is an example of the basic setting.



(b) Ethernet operation setting



(c) Open setting



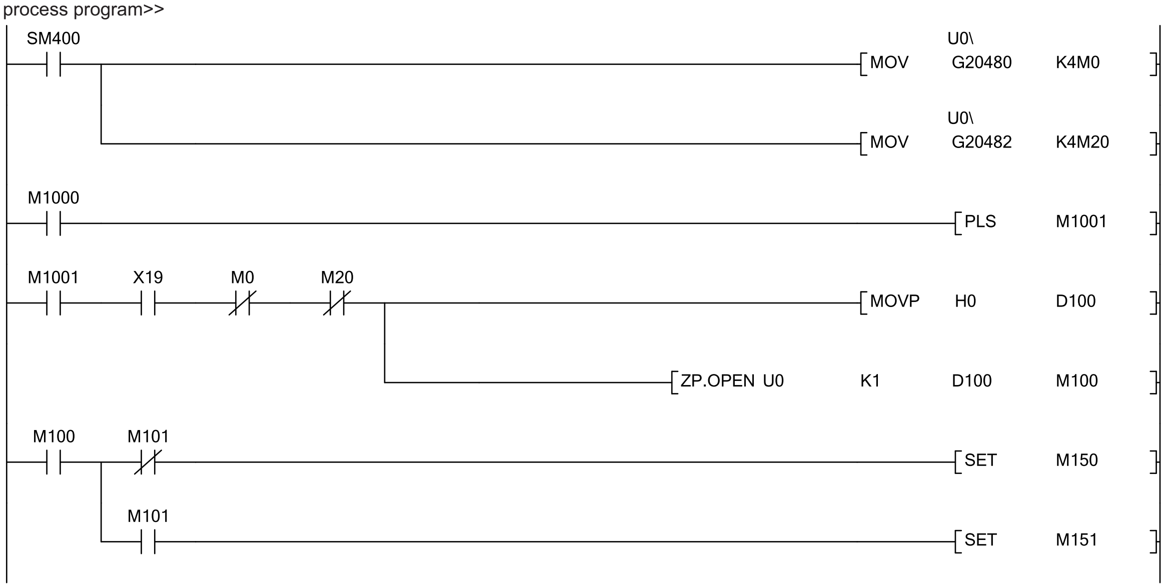
11.5.3 Program example

The following is a program example that specifies Connection No.1 and executes a protocol by using the ECPRTCL instruction.

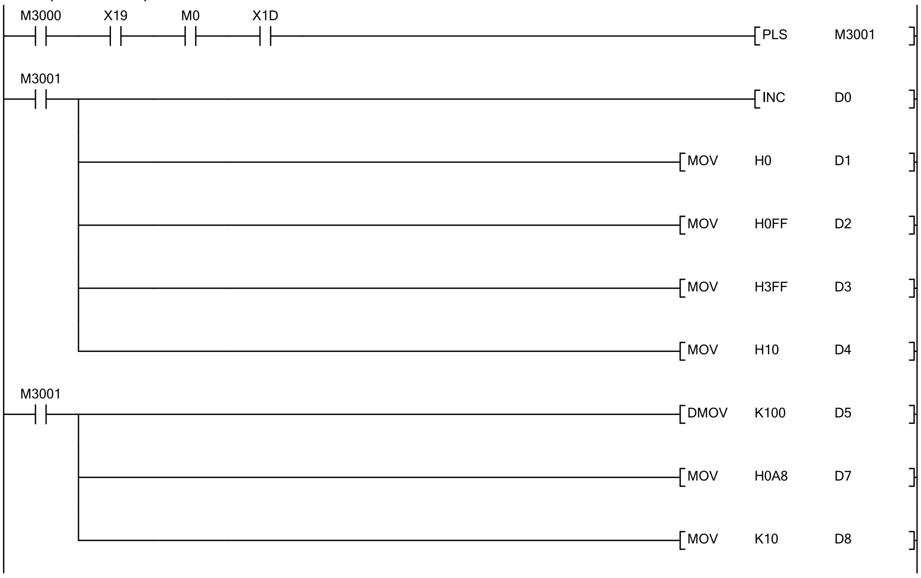
(1) Sample program

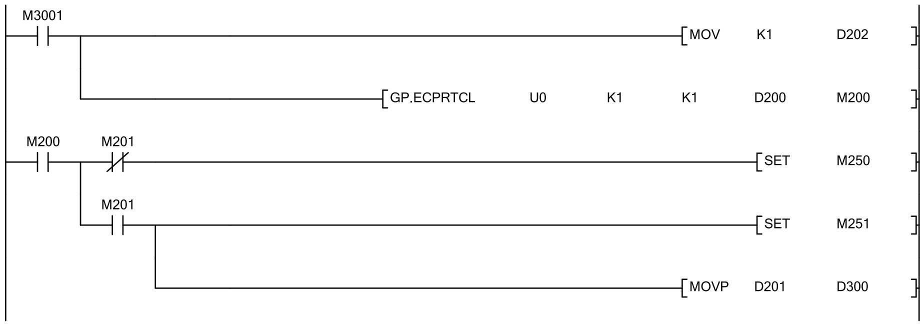
(a) Sending side (QCPU-1 side)

<<Open process program>>

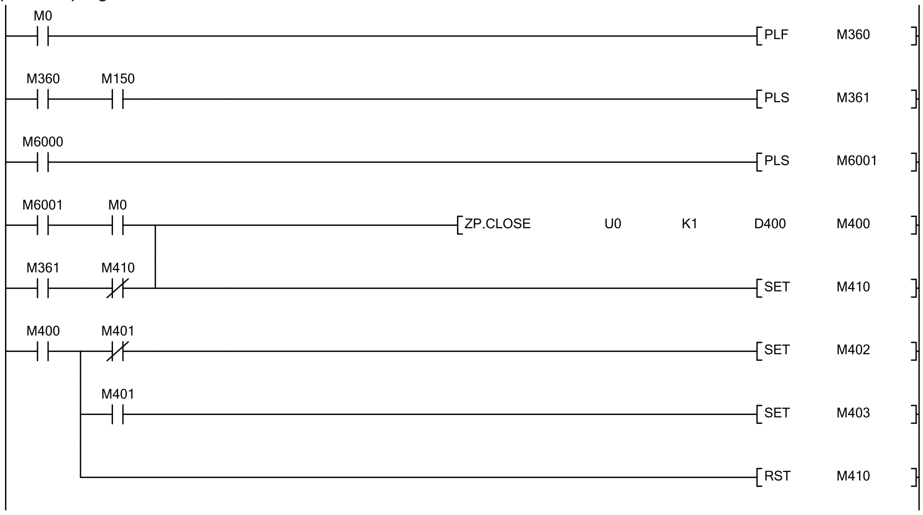


<<Program for predefined protocol communications>>




M3001
MOV K1 D202
GP.ECPRTCL U0 K1 K1 D200 M200
M200
M201
SET M250
M201
SET M251
MOVP D201 D300

<<Close process program>>



CHAPTER 12 COMMUNICATIONS USING A FIXED BUFFER

This chapter describes communications using a fixed buffer.

12.1 Applications

In communications using a fixed buffer, a programmable controller can actively send data; therefore, data can be sent from the programmable controller to the host system when an error occurs in machine equipment or some other conditions are satisfied. A maximum of 1K word of data can be sent or received between programmable controllers or between a programmable controller and the host system.

12.1.1 Differences between the "Procedure Exist" and "No Procedure" control methods

There are two control methods for communications using a fixed buffer; "Procedure Exist" control method and "No Procedure" control method. This section describes the differences between the "Procedure Exist" and "No Procedure" control methods.

Item	Difference	
	Procedure Exist	**No Procedure**
Message format	Data is sent/received in a fixed data format.	Data is sent/received in the message format of the connected device.
A response to data receiving	A response to data receiving is sent.	A response to data receiving is not sent.
Data code	Communications can be performed using a binary code or ASCII code.	Communications are performed using only a binary code.
Data length specified using a dedicated instruction	Specified by the number of words.	Specified by the number of bytes.
Amount of application data per communication	Maximum of 1017 words	Maximum of 2046 bytes

Point

A connection with a connected device is dedicated for "No Procedure" communications using a fixed buffer. MC protocol communications and "Procedure Exist" communications using a fixed buffer and random access buffer cannot be performed at the same time as "No Procedure" communications using a fixed buffer.

12.2 Communication Structure

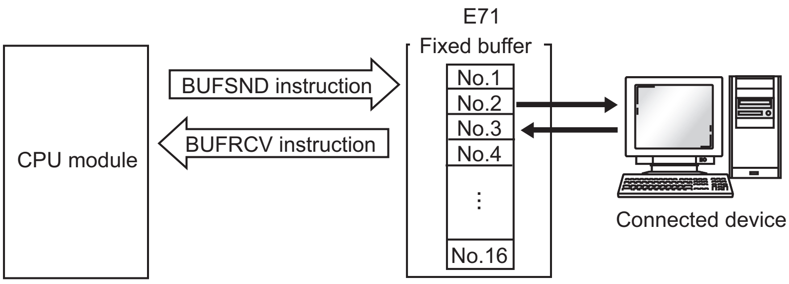
This section describes the structure of communications using a fixed buffer.

(1) Data flow

In communications using a fixed buffer, data is sent/received using dedicated instructions.

- Sending data: BUFSND instruction
- Receiving data: BUFRCV instruction or BUFRCVS instruction

When the "Procedure Exist" control method is used, the CPU module and the connected device communicate on a 1:1 basis. The CPU module sends/receives data to/from the connected device while handshaking with the connected device.
When the "No Procedure" control method is used, the CPU module sends/receives data to/from the connected device in a non-procedural manner.

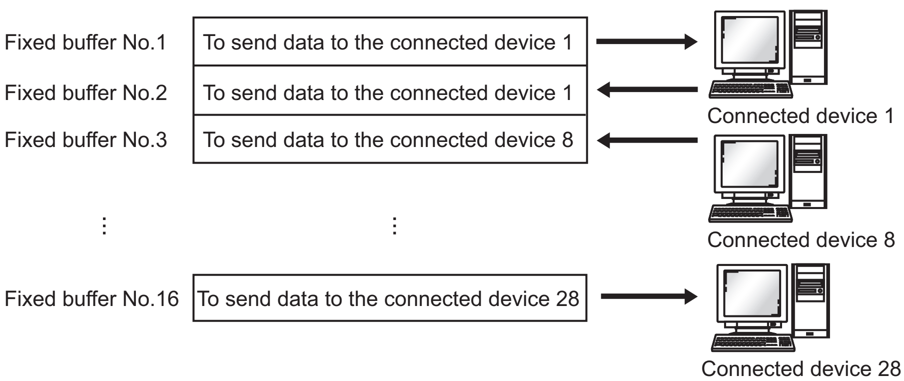


(2) Connected devices where data communications can be performed

Data communications can be performed with the following connected devices.

- Devices in Ethernet where an E71 is connected
- Devices connected with the router relay function

When using each fixed buffer (numbers 1 to 16) as shown in the figure below, set the destination devices and usage conditions (e.g. for sending/receiving and "Procedure Exist"/"No Procedure") in the open setting to fix a connected device to each buffer.



Note the following points when changing connected devices.

- During TCP/IP communications, a connected device can be changed only when the connection with the connected device is not established (when Open completion signal is off).
- During UDP/IP communications, a connected device can be changed regardless of the status of connection with the connected device.

Point

- When changing connected devices, do not use the pairing open or alive check function.
- When "No Procedure" is selected for a connection, the connection is dedicated to "No Procedure" communications using a fixed buffer after the open process is completed. When "Procedure Exist" is selected for a connection, the following data communications can be performed after the open process is completed.
  - MC protocol communications
  - Communications using a fixed buffer using the "Procedure Exist" control method
  - Communications using a random access buffer

(3) Processes during data sending/receiving

(a) When data is sent

When a CPU module executes the BUFSND instruction, the E71 sends data of the applicable fixed buffer number n to the connected device that is specified in Communication address setting area (addresses: 28_H to $5F_H$ and 5038_H to $506F_H$) corresponding to the fixed buffer number n.*1

(b) When data is received

An E71 processes the data that is received from the connected device set in the communication address setting area corresponding to the fixed buffer number n.*1

In addition, when an E71 stores the received data in the corresponding fixed buffer in the receive process, it updates the IP address and destination port number of the connected device in Connection information area (addresses: 78_H to $C7_H$ and 5820_H to $586F_H$) corresponding to the fixed buffer number n.

If data is received from a connected device not set in the connection information area of the buffer memory, an E71 ignores the received data.

*1 In case of the TCP/IP Unpassive open, data is sent to or received from a connected device stored in the connection information area of the buffer memory.

Point

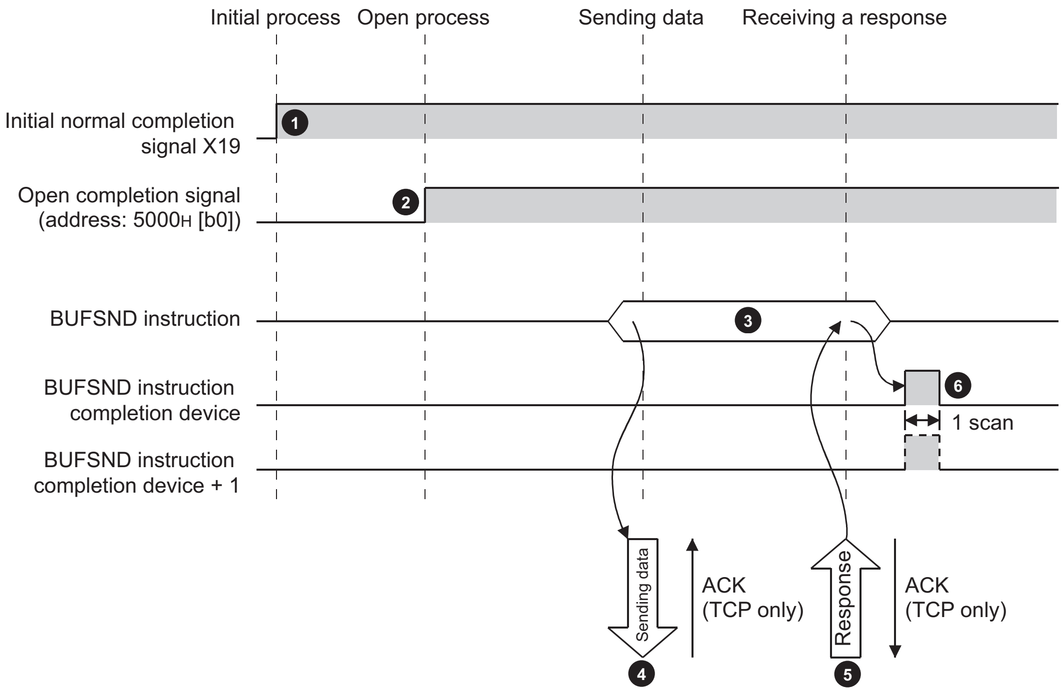
When received data is stored in the corresponding fixed buffer in the receive process of broadcast communications, an E71 updates the destination IP address and port number in Connection information area (address: 78_H to $C7_H$ and 5820_H to $586F_H$) corresponding to the fixed buffer number n.

12.3 Data Sending Procedure

This section describes how an E71 sends data to the connected device.

(1) Procedure exists

The following is the data send process performed in the fixed buffer number 1 area for the connection number 1.



1. Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)
2. Establish a connection between the E71 and the connected device and confirm the normal completion of the open process of the connection number 1.
3. Execute the BUFSND instruction. (The data is sent.)
4. The send data in the fixed buffer number 1 is sent to the connected device by the send data length.
5. When the connected device receives the data from the E71, the connected device sends a response to the E71.
6. When the E71 receives the response from the connected device, the E71 stops sending data. If a response is not sent within the response monitoring timer value, a data send error occurs.*1
   After the data send process abnormally ends, execute the BUFSND instruction again to repeat the send process.

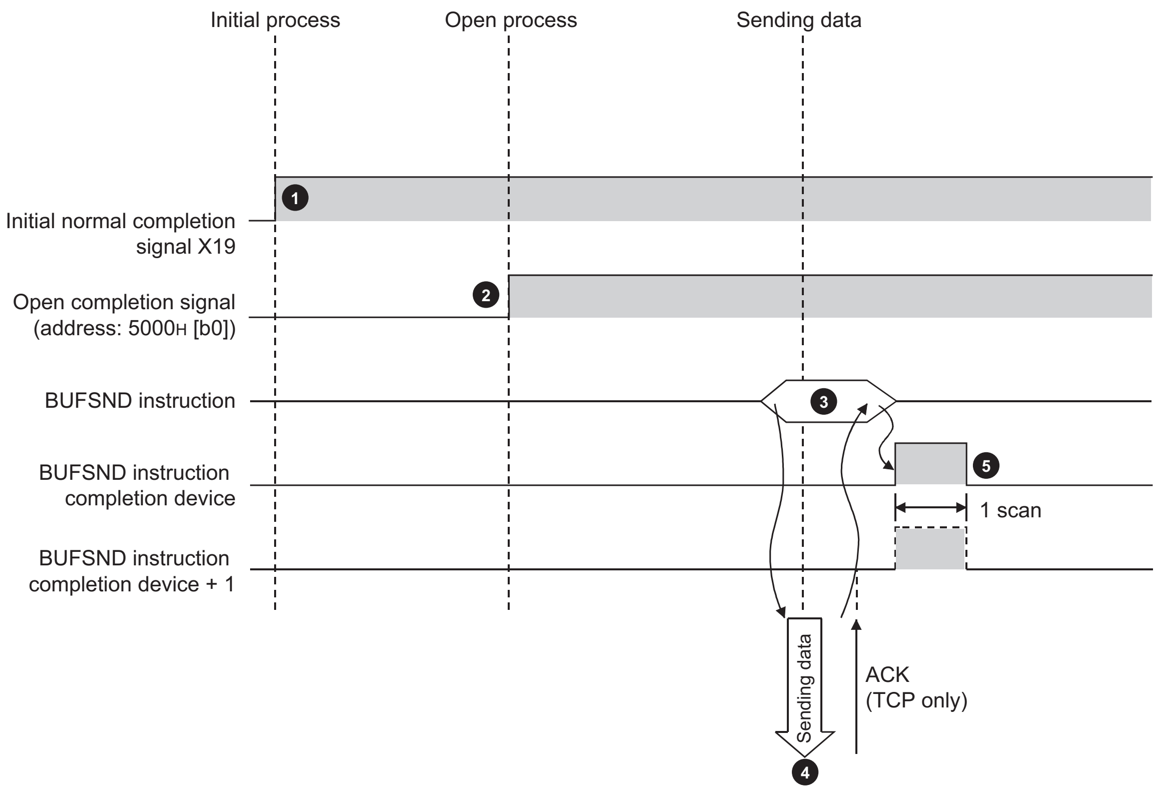
*1 Refer to the section describing the initial setting when adjusting the monitoring timer value. (Page 354, Appendix 4.1)

Point

- The items configured in the open setting become enabled when Open completion signal of an E71 is started up.
- Do not send data (command) before data communications are completed (or a response is received) in response to the previous data (command).
- When an E71 communicates data with multiple connected devices, data can be sent sequentially; however, switching the connected devices before sending/receiving data is recommended to prevent communication problems. When a connection opened using the UDP/IP protocol is used, change the setting values in the communication address setting area before sending or receiving data to switch the connected devices.

(2) No procedure

The following is the data send process performed in the fixed buffer number 1 area for the connection number 1.



1. Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)
2. Establish a connection between the E71 and the connected device and confirm the normal completion of the open process of the connection number 1.
3. Execute the BUFSND instruction. (The data is sent.)
4. The send data in the fixed buffer number 1 is sent to the connected device by the send data length.
5. The E71 stops sending data. After the data send process abnormally ends, execute the BUFSND instruction again to repeat the send process.

Point

During UDP/IP communications, when the internal process of an E71 is normally completed, the data send process ends even if the communication line between a CPU module and the connected device is not connected due to cable disconnection or other causes. It is recommended to set an original communication procedure to send/receive data.

12.4 Data Receiving Procedure

This section describes how an E71 receives data from the connected device. The following methods for receiving data are offered:

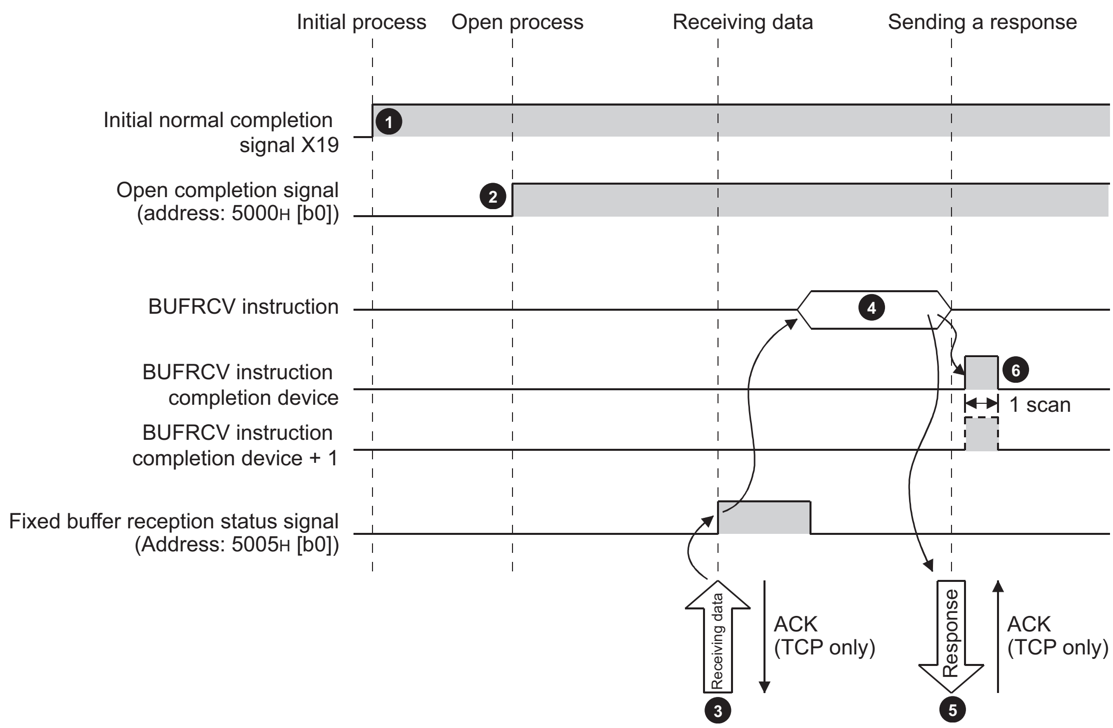
- Data receiving using the main program (BUFRCV instruction)
- Data receiving using an interrupt program (BUFRCVS instruction)

12.4.1 Data receiving using the main program (BUFRCV instruction)

A receive process in the main program is performed using the BUFRCV instruction.

(1) Procedure exists

The following is the data receive process performed in the fixed buffer number 1 area for the connection number 1.



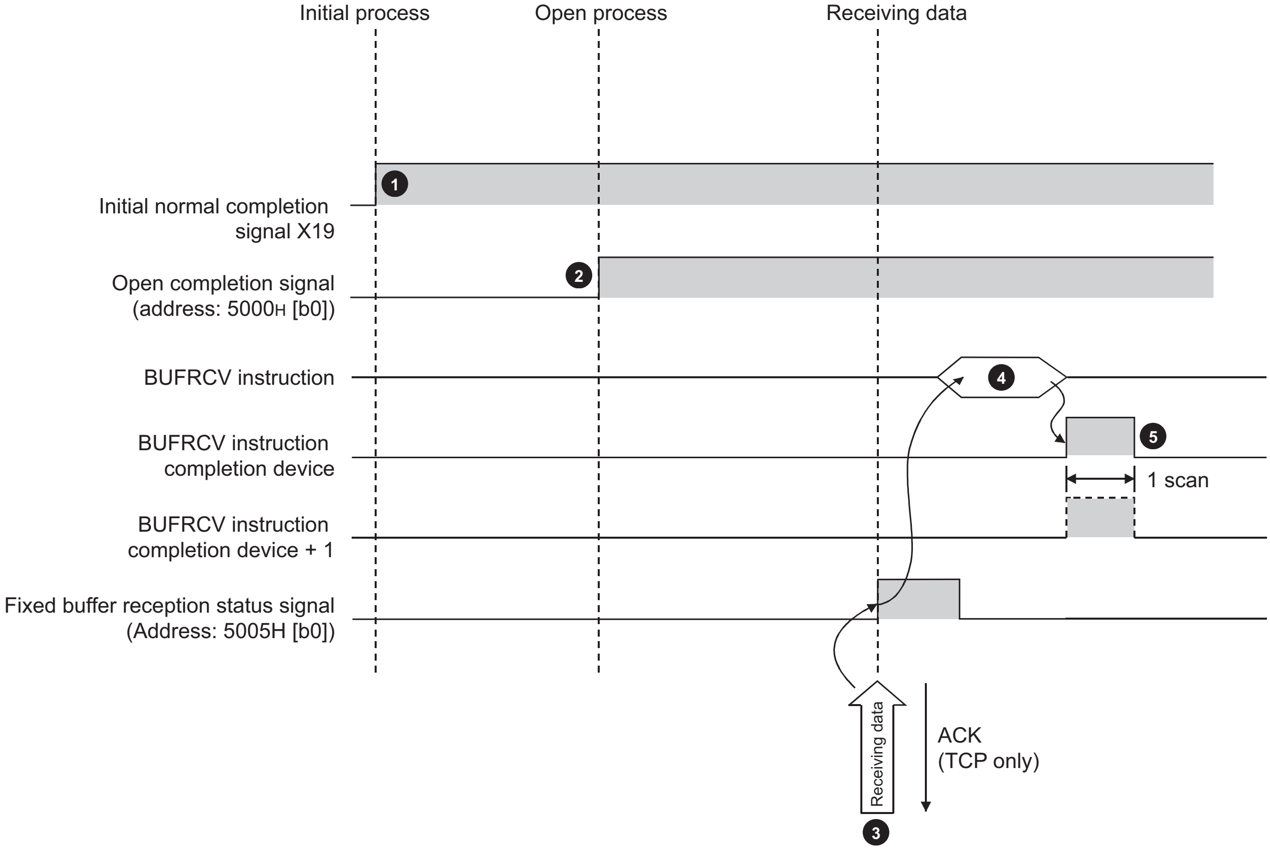
1. Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)
2. Establish a connection between the E71 and the connected device and confirm the normal completion of the open process of the connection number 1.
3. Receive data from the destination.
   - Fixed buffer reception status signal (address: 5005_H (b0)): ON
4. Execute the BUFRCV instruction to read the receive data length and receive data from the fixed buffer number 1.
   - Fixed buffer reception status signal (address: 5005_H (b0)): OFF
5. When the receive data length and the receive data are completely read out, a response is sent to the destination.
6. End the receive process. If the data receive process is abnormally ended, execute the BUFRCV instruction again to repeat the receive process.

Point

- The items configured in the open setting become enabled when Open completion signal of an E71 is started up.
- When the buffer memory area that stores Fixed buffer reception status signal is turned off and on, execute the BUFRCV instruction.
- When abnormal data is received, Fixed buffer reception status signal does not turn on. In addition, data is not stored in the fixed buffer number 1 area.

(2) No procedure

The following is the receive process performed in the fixed buffer number 1 area for the connection number 1.



1. Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)
2. Establish a connection between the E71 and the connected device and confirm the normal completion of the open process of the connection number 1.
3. Receive data from the connected device.
   • Fixed buffer reception status signal (address: 5005_H (b0)): ON
4. Execute the BUFRCV instruction to read the receive data length and receive data from the fixed buffer number 1.
   • Fixed buffer reception status signal (address: 5005_H (b0)): OFF
5. End the receive process. If the data receive process is abnormally ended, execute the BUFRCV instruction again to repeat the receive process.

Point

- The items configured in the open setting become enabled when Open completion signal of an E71 is started up.
- When the buffer memory area that stores Fixed buffer reception status signal is turned off and on, execute the BUFRCV instruction.
- When abnormal data is received, Fixed buffer reception status signal does not turn on. In addition, data is not stored in the fixed buffer number 1 area.

12.4.2 Data receiving using an interrupt program (BUFRCVS instruction)

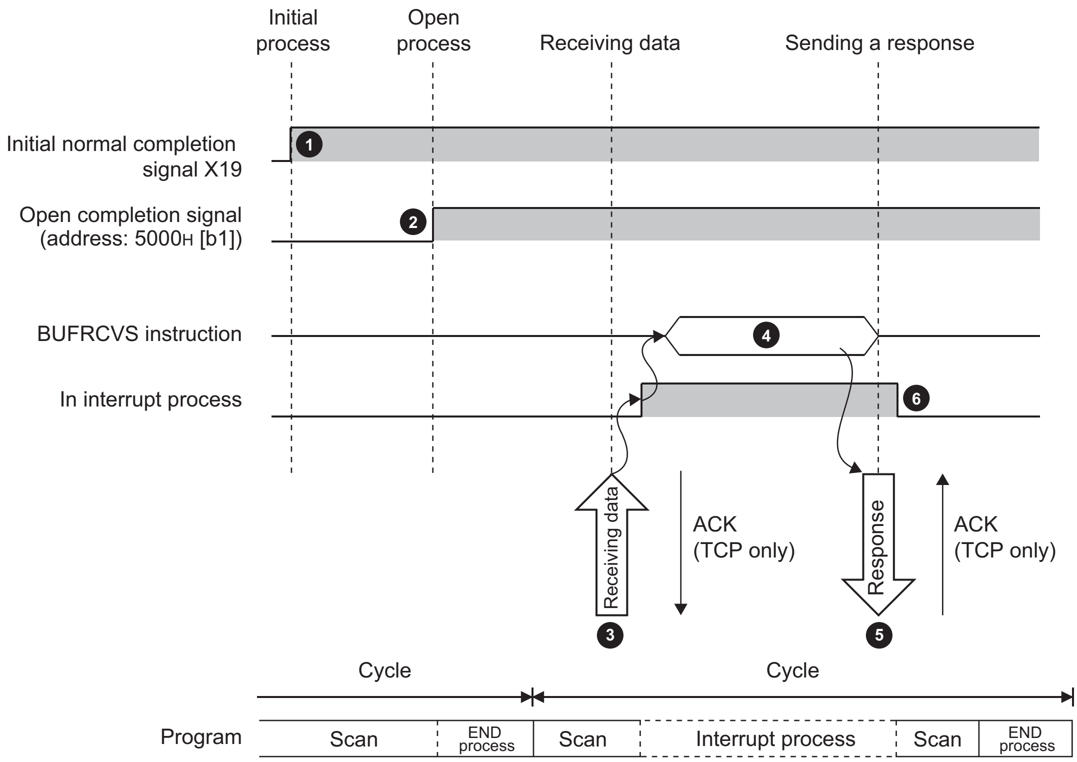
A receive process in an interrupt program is performed using the BUFRCVS instruction. An interrupt program is started up when data is received from a connected device, and receive data for a CPU module can be read.

The following settings are required to use an interrupt program: (Page 150, Section 12.5.1)

- Interrupt pointer setting
- Interrupt settings

(1) Procedure exists

The following is the receive process performed in the fixed buffer number 2 area for the connection number 2.

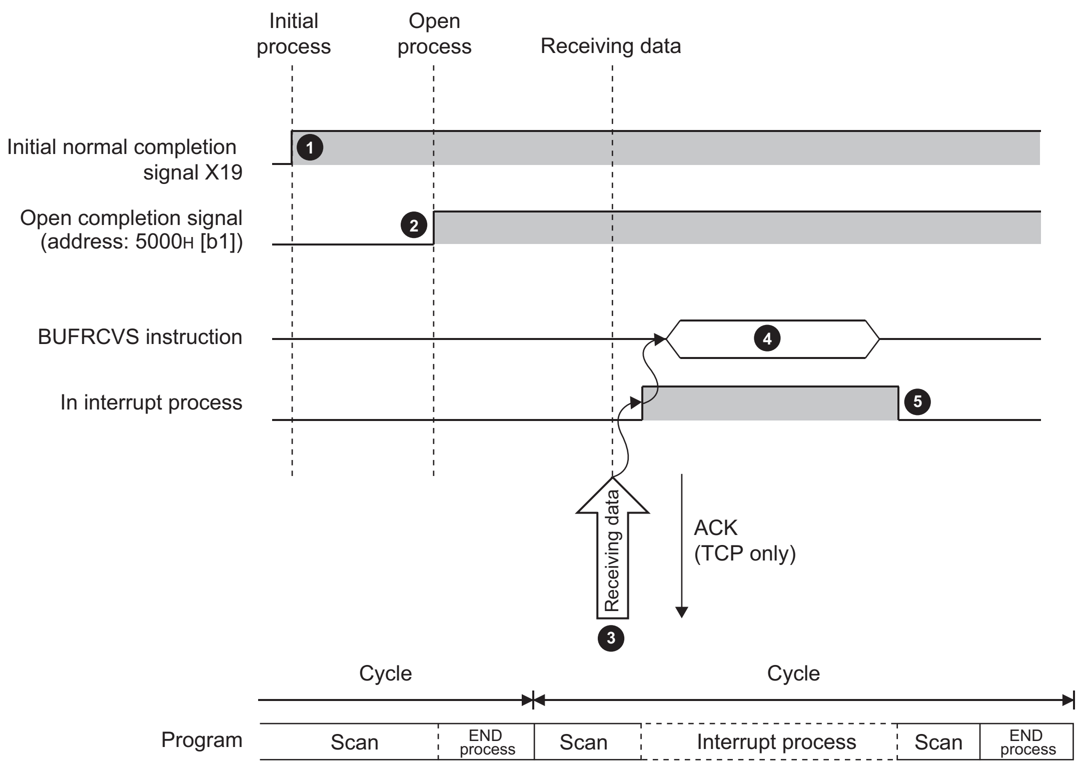


1. Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)
2. Establish a connection between the E71 and the connected device and confirm the normal completion of the open process of the connection number 2.
3. Receive data from the destination.
   - Fixed buffer reception status signal (address: 5005_H (b1)): ON
   - Request the CPU module to start up the interrupt program.
4. The interrupt program starts up. Execute the BUFRCVS instruction to read the receive data length and receive data from the fixed buffer number 2.
5. When the receive data length and the receive data are completely read out, a response is sent to the destination.[*1]
6. End the interrupt program to restart the main program.

*1 If the data receive process is abnormally ended, the E71 does not return a response.

(2) No procedure

The following is the receive process performed in the fixed buffer number 2 area for the connection number 2.



❶ Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)

❷ Establish a connection between the E71 and the connected device and confirm the normal completion of the open process of the connection number 1.

❸ Receive data from the destination.

- Request the CPU module to start up the interrupt program.

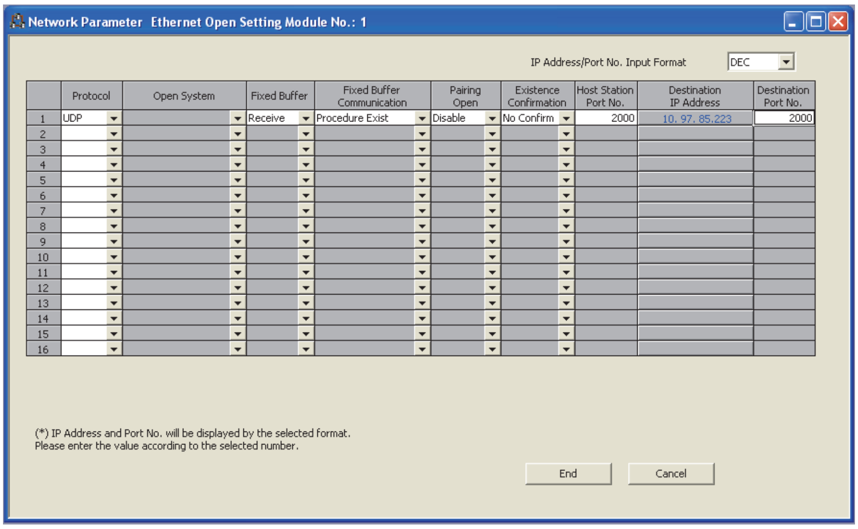
❹ The interrupt program starts up. Execute the BUFRCVS instruction to read the receive data length and receive data from the fixed buffer number 1.

❺ End the interrupt program to restart the main program.

12.5 Parameter Setting

Set the following parameters to perform communications using a fixed buffer.

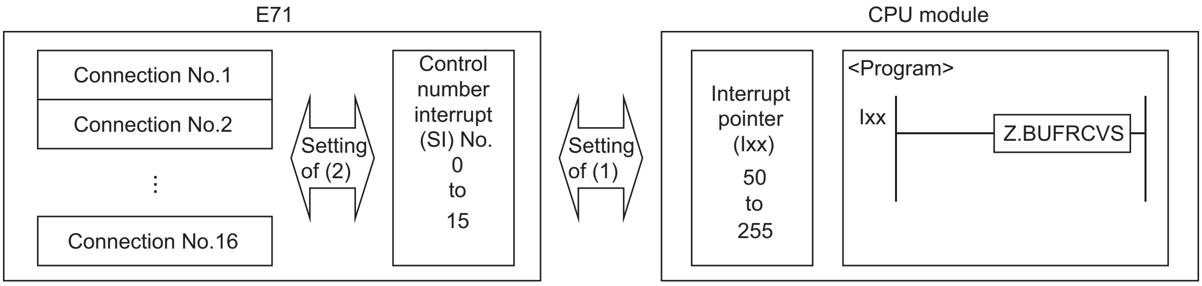
- Basic setting (Page 84, Section 7.1.2)
- Ethernet operation setting (Page 85, Section 7.1.3)
- Open setting (Page 87, Section 7.1.4)



Item	Description	Setting range
Fixed Buffer	Select whether to use the fixed buffer for connection with the connected device for sending or receiving.	• Send • Receive
Fixed Buffer Communication	Select the communication method for communications using a fixed buffer.	• Procedure Exist • No Procedure
Pairing Open	Select whether to disable or enable the pairing open for communications using a fixed buffer. (Page 158, Section 12.7)	• Disable • Enable
Existence Confirmation	Select whether or not to use the alive check function. (Page 234, Section 14.8) Select "No Confirm" to send data by broadcast communications during "No Procedure" communications using a fixed buffer.	• No Confirm • Confirm

12.5.1 Parameter setting when using an interrupt program

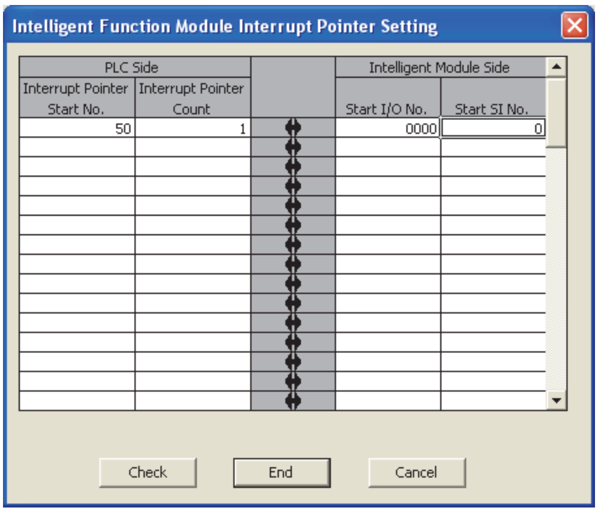
Configure the parameter setting on a programming tool to start up an interrupt program.



(1) Interrupt pointer setting

The following is a setting example.

Project window ⇨ [Parameter] ⇨ [PLC Parameter] ⇨ [PLC System] ⇨ Interrupt Pointer Setting button

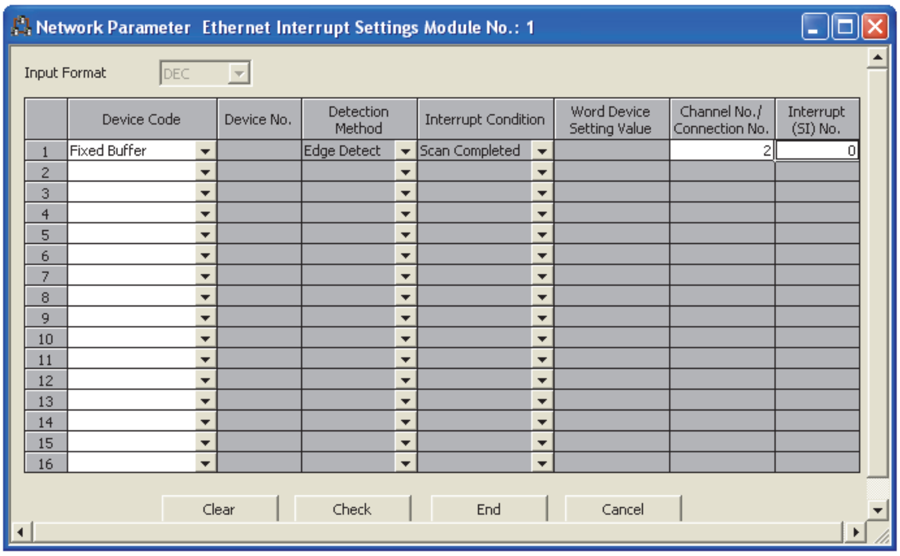


Item		Description	Setting range
PLC Side	Interrupt Pointer Start No.	Set the start number of the interrupt pointer used in a program.	50 to 255
	Interrupt Pointer Count	Set the number of interrupt pointers used in a program.	1 to 16
Intelligent Module Side	Start I/O No.	Set the start I/O number of an E71.	0000_H to $0FE0_H$
	Start SI No.	Among maximum 16 interrupt (SI) numbers (control numbers on the Ethernet module side) to be set in the interrupt program, set the smallest numbers 0 to 15.	0 to 15

(2) Interrupt settings

The following is a setting example.

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ [Interrupt Settings]



Item	Description	Setting range
Input Format	The input format of each data is decimal number only.	-
Device Code	Select a device code. Select "Fixed Buffer" in this setting.	Fixed Buffer
Channel No./Connection No.	Set the connection number of the fixed buffer to trigger the interrupt program.	1 to 16
Interrupt (SI) No.	Set the interrupt control number for the E71 for when an interrupt request is issued from the E71 to the CPU module.*1	0 to 15

*1 Do not set the same interrupt (SI) numbers as those to be set for an interrupt of other communications using a fixed buffer or the RECV instruction.

Point

Interrupt (SI) numbers can be assigned by the user to receive up to 16 communications using a fixed buffer and communications using the RECV instruction in an interrupt program. The user must manage the interrupt (SI) numbers assigned for receiving data.

12.6 Data Format

Communication data consists of a header and application data.

12.6.1 Header

The header for TCP/IP or UDP/IP is used. Because an E71 automatically adds and deletes a header, the user setting is not required.

(1) Detailed header sizes

(a) TCP/IP

Ethernet (14 bytes)	IP (20 bytes)	TCP (20 bytes)

(b) UDP/IP

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

12.6.2 Application data

This section describes the application data configuration.

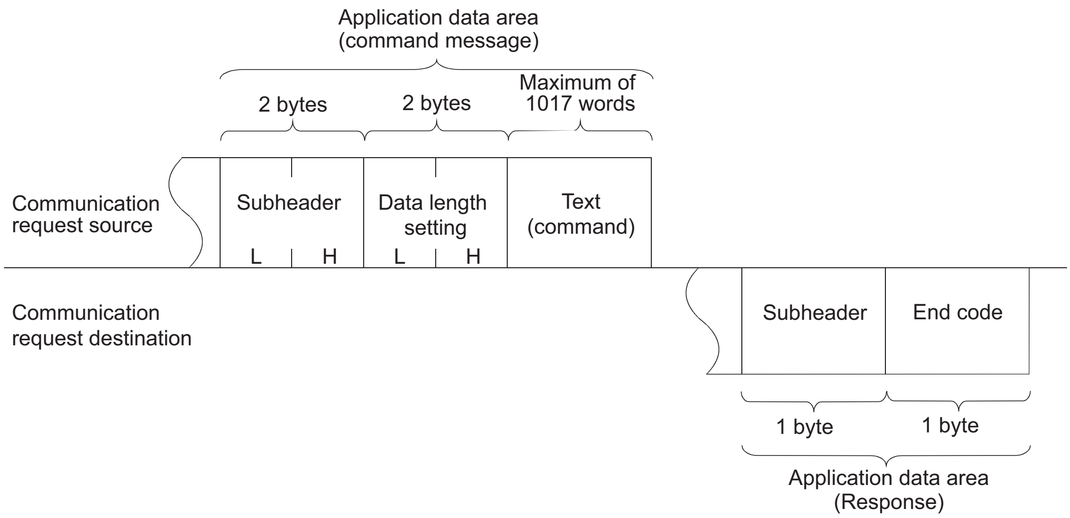
(1) Procedure exists

In the application data for communications with the "Procedure Exist" control method, the following data code is expressed in a binary code or an ASCII code. To switch between a binary code and an ASCII code, use the Ethernet operation setting.

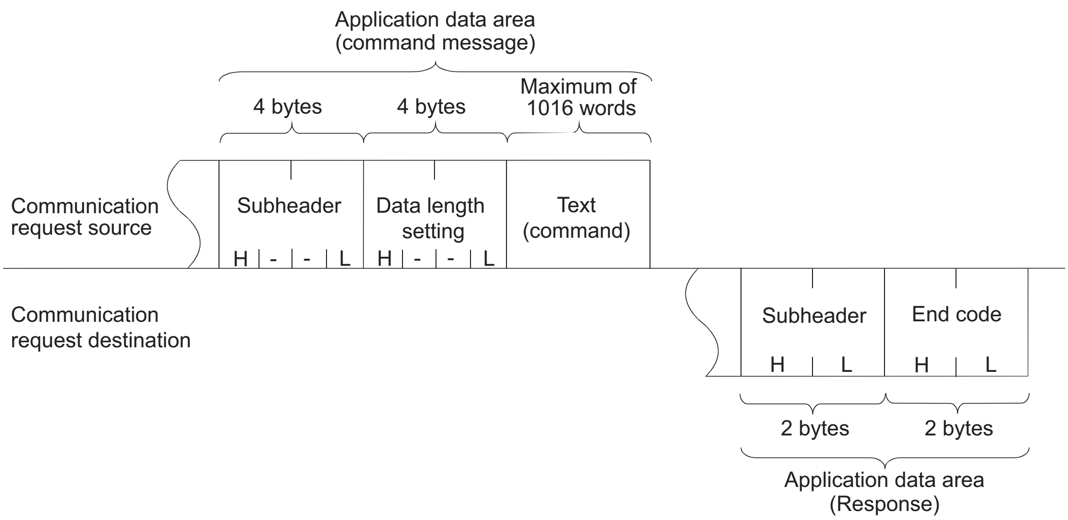
Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ "Operation Setting"

(a) Format

- Communications using a binary code

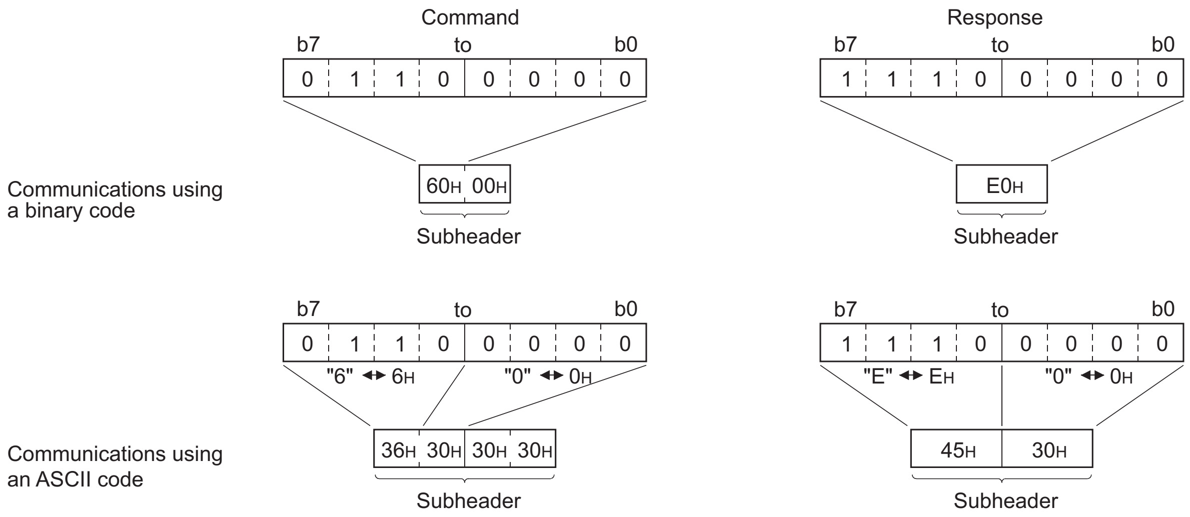


- Communications using an ASCII code



(b) Subheader

Because an E71 automatically adds and deletes a header, the user setting is not required.



(c) Data length setting

The following are the data sizes of a text (command).

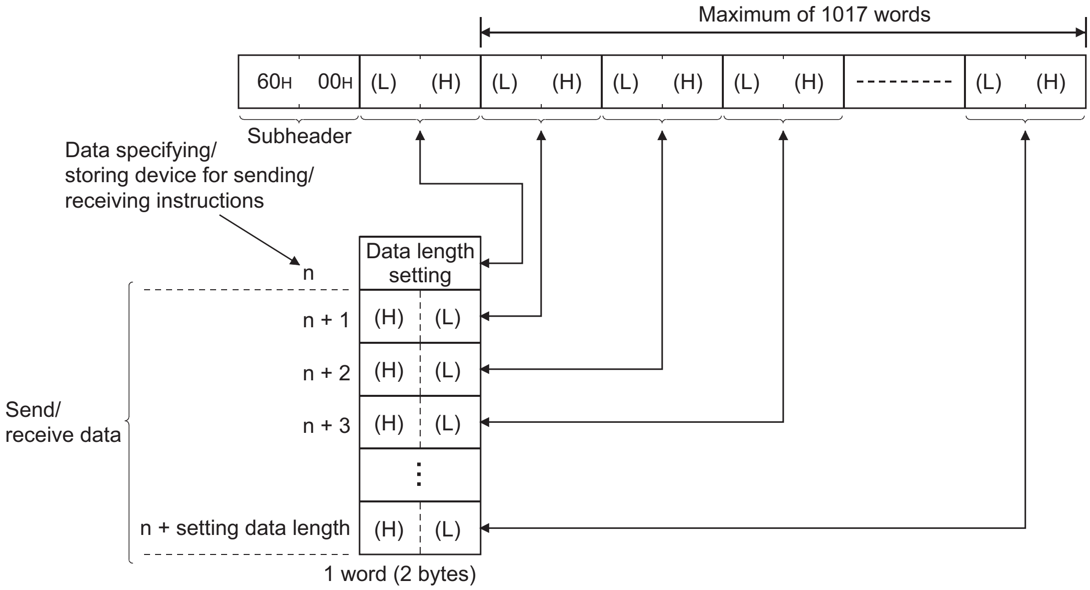
- Communications using a binary code: Maximum of 1017 words
- Communications using an ASCII code: Maximum of 508 words

(d) Text (command)

The following is the format of a command/response.

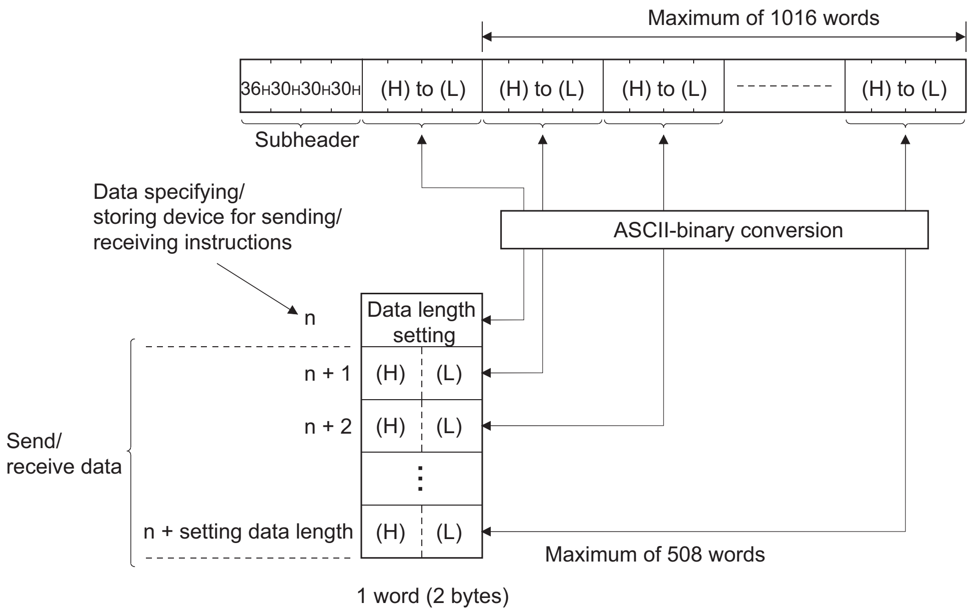
- Communications using a binary code

Command format



- Communications using an ASCII code

Command format



(e) End codes

An error code is stored in the end code added to a response. For the error codes, refer to the error code list.
(☞Page 310, Section 16.6.1)
End codes are stored in the complete status area (in the control data) of the BUFSND and BUFRCV instructions, as well as the communication status storage area of the buffer memory.

Point

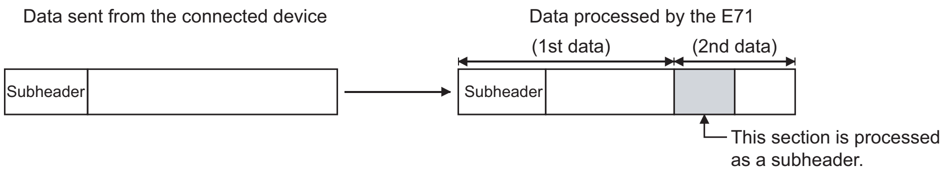
If an error code related to MC protocol communications or communications using a random access buffer is stored during communications using a fixed buffer, it may be caused by the following.

Cause	Action
The data length value specified in the application data of the message sent from a connected device to the E71 is different from the actual text data size.	Specify the actual text data size for the data length in the application data. (Refer to "Remark" in this page.)
The subheader of the message sent from a connected device to the E71 is incorrect.	Correct the subheader specified in the application data.

Remark

Depending on the restrictions of the buffers of the own station and destination station, data may be divided for communications. Data received separately is restored (reassembled) by the E71 for communications. The received data is restored (reassembled) based on the data length in the communication data. The E71 performs the following processes if data in the communication data is incorrect.

- If the data length specified immediately after the subheader is smaller than the amount of text data received
  1) The data immediately after the text data equivalent to the data length specified immediately after the subheader is regarded as the next message.
  2) Because the header of each message is a subheader, an E71 processes data according to the code in the subheader.
  3) If the subheader contains a code that is not recognized by an E71, the E71 sends a response notifying about an abnormal end to the connected device.

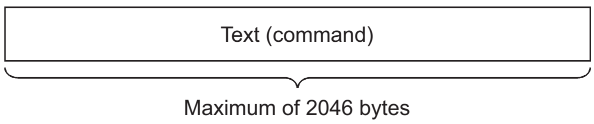


  In the figure above, the E71 returns a response containing a code obtained by changing the most significant bit of the code processed as a subheader to 1. For example, if the subheader of a command is 65_H, the subheader of the response is $E5_H$.

- If the data length specified immediately after the subheader is larger than the amount of text data received
  1) The E71 waits for the remaining data.
  2) When the remaining data is received within the response monitoring timer value, the E71 processes data according to the code in the subheader.
  3) If the remaining data could not be received within the response monitoring timer value, the E71 performs the following processes.
    - Sends the ABORT (RST) instruction to the connected device and closes the line.
    - Notifies the CPU module that an open error has occurred by turning on Open abnormal detection signal (X18)
    - Stores the error code in the open error code storage area. (The error code is not stored in the error log storage area.)

(2) No procedure

In the application data for communications in the "No Procedure" control method, the following data code is expressed in a binary code. Communications are performed using a binary code regardless of the communication data code setting.



Point

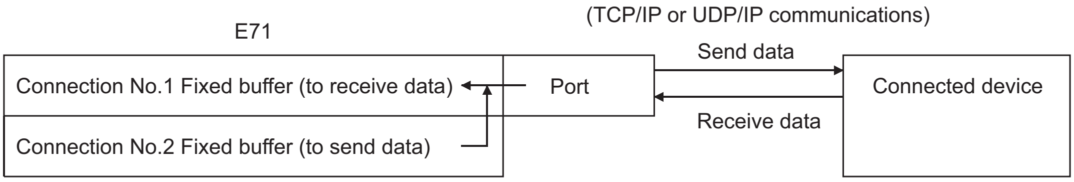
In communications in the "No Procedure" control method, a subheader or data length setting is not added unlike the "Procedure Exist" control method; therefore, all data are handled as a valid text. In addition, the E71 turns on Fixed buffer reception status signal after storing the size of the received message (packet) in the data length storage area. It is recommended to set a checking method. For example, include data length and a data type code in the application data of a message so that the number of bytes and data type of the application data can be identified on the receiving side.

12.7 Pairing Open

Pairing open is an opening method that connects the own station with the connected device using only one port by pairing the receive and send connections for communications using a fixed buffer.

12.7.1 Applications

Enabling the pairing open allows data communications to be performed with two connections by performing the open process for one port. MC protocol communications and communications using a random access buffer can be also performed using paring-opened connections.

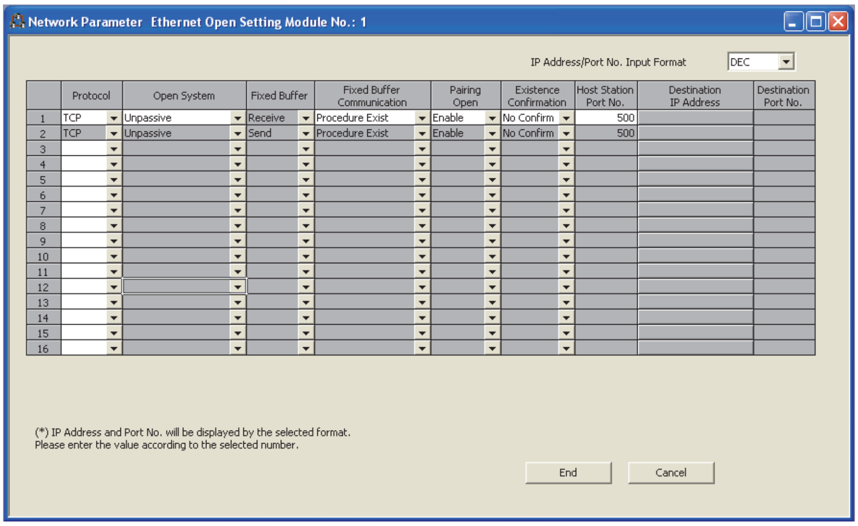


Point

- Connected devices where communications can be performed using the pairing open method are those in Ethernet where an E71 is connected and those connected using the router relay function.
- The open/close processes in the receive connection where the pairing open has been enabled automatically perform the open/close processes in the next connection (send connection).

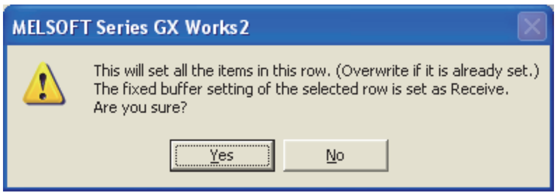
12.7.2 Parameter setting

The following is the pairing open setting.



Item	Description	Setting range
Pairing Open	Set "Pairing Open" of the receive connection to "Enable". The next connection is automatically set as a send connection.*1	-

*1 When "Enable" has been set in "Pairing Open" for a send connection, the following window appears.



When "Yes" is selected, the applicable connection becomes a receive connection, and the next connection is automatically set as a send connection. (When "Enable" has already been set in "Pairing Open", the setting is overwritten.)

Point

Because the fixed buffer of the connection with the connected device (dedicated for receiving) and the fixed buffer of the next connection (dedicated for sending) are paired, set connection numbers 1 to 7 and 9 to 15 as receive connections. (Connection numbers 8 and 16 cannot be set.)

12.8 Broadcast Communications

Broadcast communications mean that the same data is sent to all E71-mounted stations in the same Ethernet network and to the connected devices without specifying destinations. Broadcast communications can be performed when the following conditions are met.

- When UDP/IP protocol is used
- When "No Procedure" communications using a fixed buffer or data communications using the predefined protocol are performed

Point

- When connected devices in the same Ethernet network do not need to receive communication messages by broadcast communications, a discarding process is required.
- The user needs to determine the port numbers dedicated for data sending/receiving for broadcast communications.
- Sending/receiving procedures to perform the broadcast communications in communications using the predefined protocol are same as that in communications using a fixed buffer except for setting "Predefined protocol" under “Fixed Buffer Communication” of the open setting.

12.8.1 Sending/receiving procedures

(1) Sending procedure

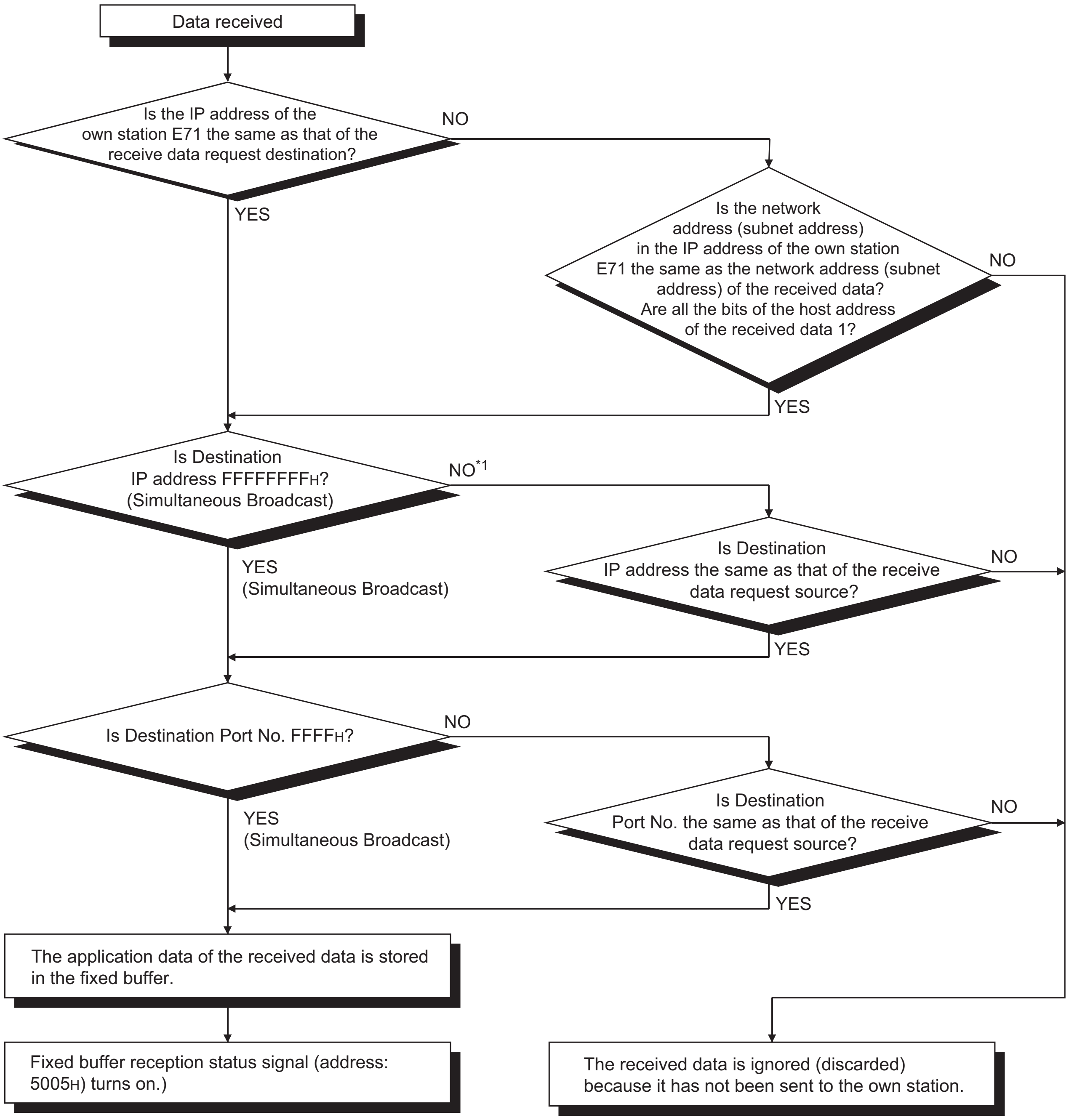
To send data by broadcast communications, set the IP addresses of the connected devices where the E71 sends data to $FFFFFFFF_H$ and perform the open process. The E71 sets all the host address bits to 1 and sends data over the Ethernet network.

(2) Receiving procedure

To receive data as data for broadcast communications, set the IP addresses of the connected devices that send data to the E71 to $FFFFFFFF_H$ and the port numbers to $FFFF_H$ and perform the open process.

Remark

The following is the outline of the internal process of an E71 when data is received in the "No Procedure" control method or by broadcast communications. (The values specified in the buffer memory addresses 0_H, 1_H and 28_H to $5F_H$ are used for the IP addresses and port numbers of the E71 in the own station as well as those of connected devices.)



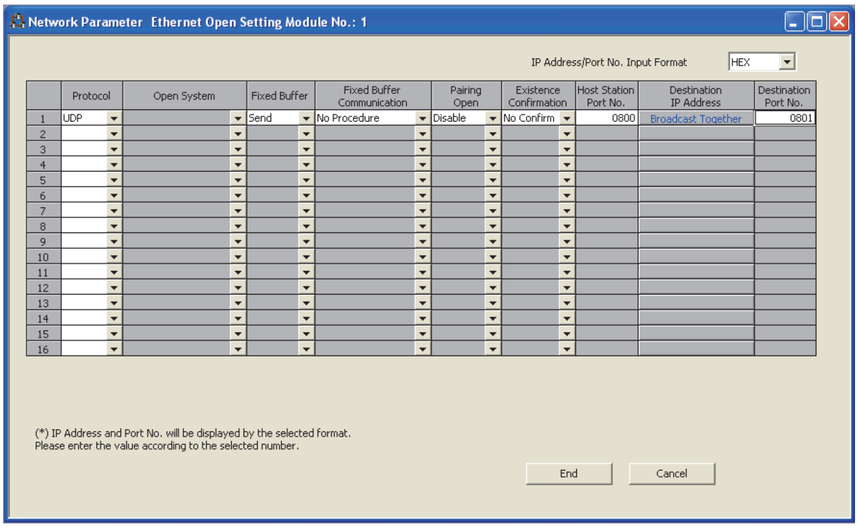
*1 When all the bits of the area expressing the host address of the IP address of the receive data request destination are 1, proceed to "YES".

12.8.2 Parameter setting

The following is the parameter setting for broadcast communications.

(1) When sending data

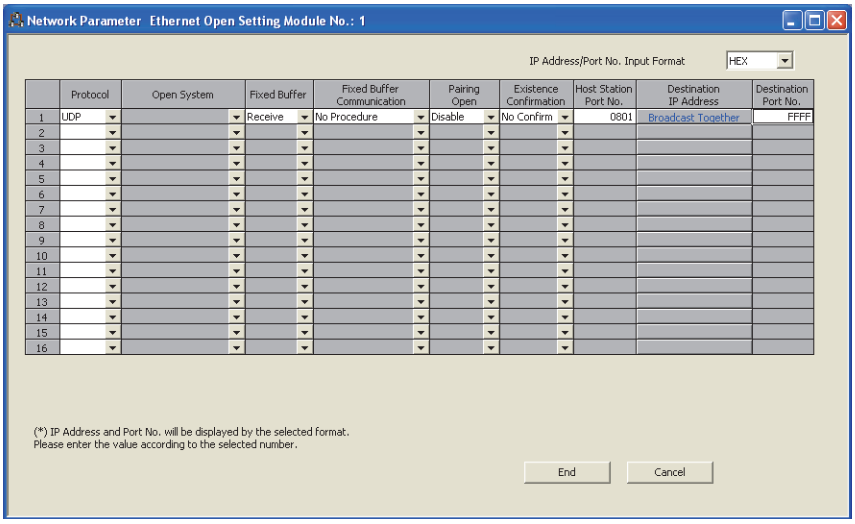
The following is the parameter setting for data sending.



Item	Description	Setting range
Protocol	Select "UDP/IP".	-
Fixed Buffer	Select "Send".	-
Fixed Buffer Communication	Select "No Procedure".	-
Existence Confirmation	Select "No Confirm".	-
Destination IP Address	Set $FFFFFFFF_H$.	-

(2) When receiving data

The following is the parameter setting for data receiving.



Item	Description	Setting range
Protocol	Select "UDP/IP".	-
Fixed Buffer	Select "Receive".	-
Fixed Buffer Communication	Select "No Procedure".	-
Existence Confirmation	Select "No Confirm".	-
Destination IP Address	Set $FFFFFFFF_H$.	-
Destination Port No.	Set $FFFF_H$.	-

12.8.3 Precautions

This section describes the precautions for broadcast communications.

(1) Port number

To perform broadcast communications, the user needs to determine the port numbers dedicated for data sending/receiving for broadcast communications and specify these port numbers.

(2) Size of send/receive data per sending/receiving

A maximum of 2046 bytes of data in the application data can be processed per sending/receiving. To send data with a size exceeding 2047 bytes, divide the data into smaller chunks at the send source.

Remark

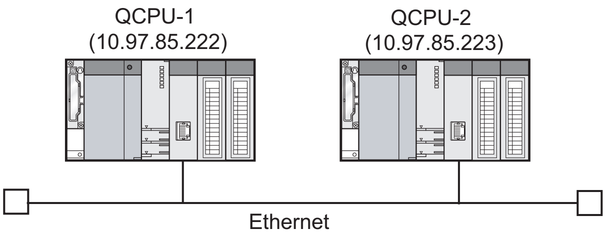
An E71 temporarily stores a subsequent receive data in its internal buffer for the operating system until the current receive process is completed. If data with a size exceeding the capacity of the internal buffer (approximately 40KB) is received by broadcast communications, the data is discarded. In communications using a fixed buffer in the "Procedure Exist" control method, the E71 sends a command message to the connected device, waits for a response message, then sends the next command message. Therefore, the user need not take account of the above-mentioned internal buffer for the operating system.

12.9 Example of Communications Using a Fixed Buffer

This section describes an example of communications using a fixed buffer in the "Procedure Exist" control method between an E71 and the connected device.

12.9.1 System configuration

The following system configuration is used for explanation purpose.



12.9.2 Parameter setting

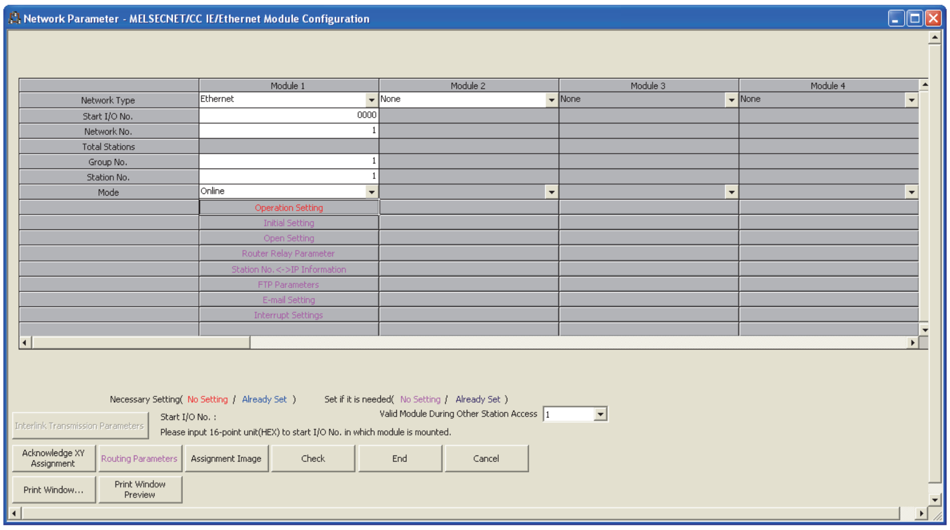
This section describes a parameter setting example.

(1) Sending side (QCPU-1 side)

The following is an example of the parameter settings of the sending side (QCPU-1 side).

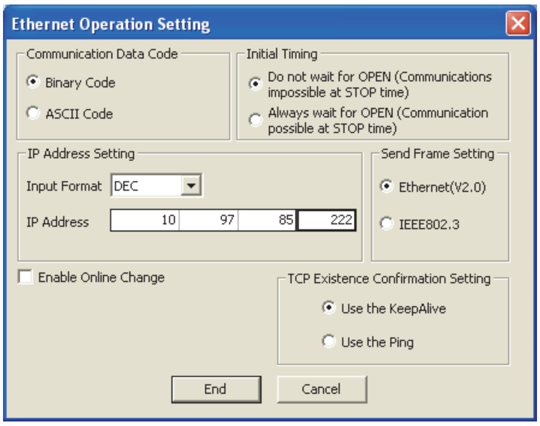
(a) Basic setting

The following is an example of the basic setting.



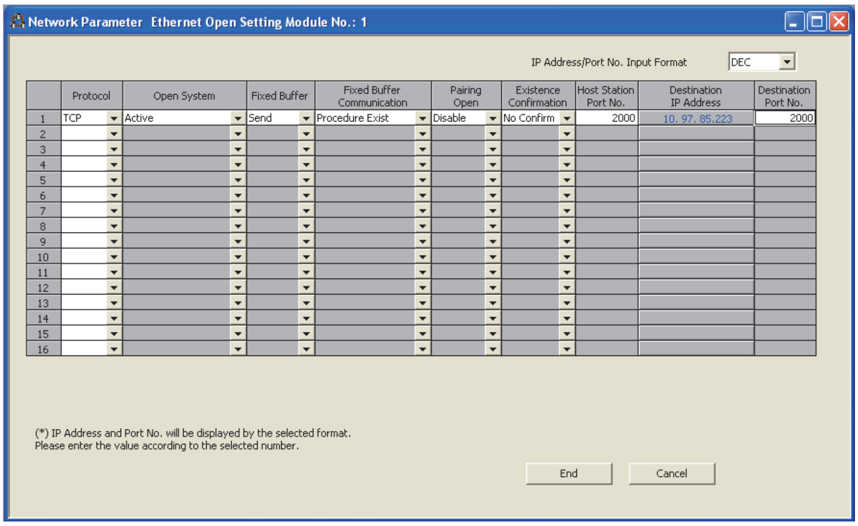
(b) Ethernet operation setting

The following is an example of the Ethernet operation setting.



(c) Open setting

The following is an example of the open setting.

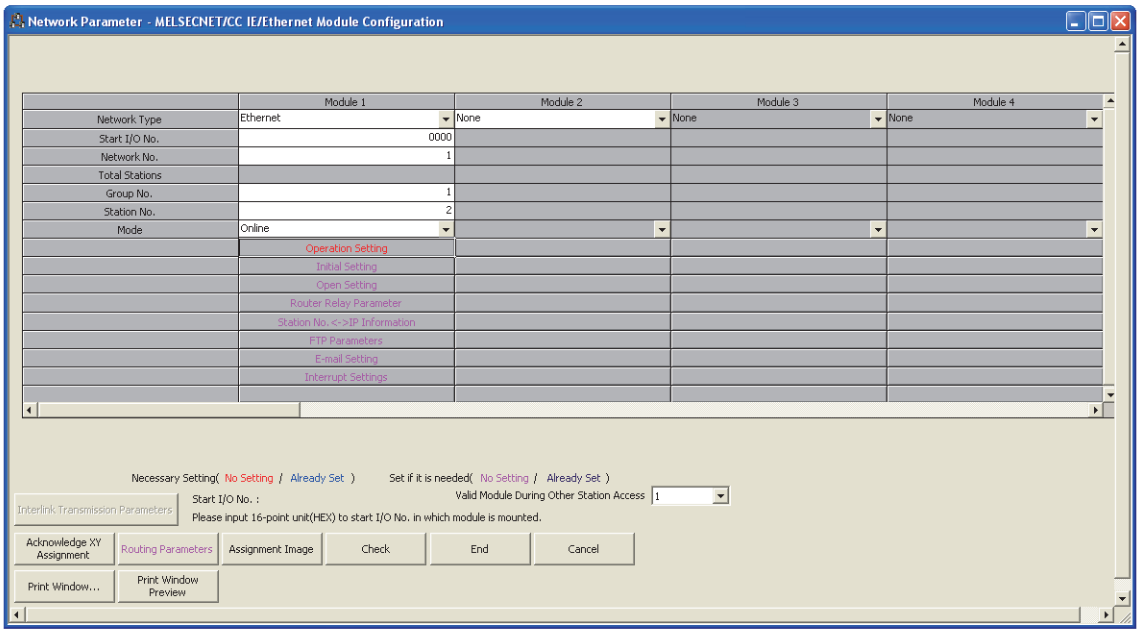


(2) Receiving side (QCPU-2 side)

The following is an example of the parameter settings of the receiving side (QCPU-2 side).

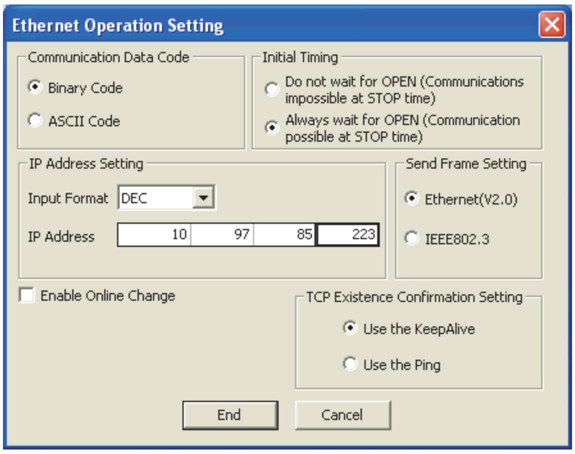
(a) Basic setting

The following is an example of the basic setting.



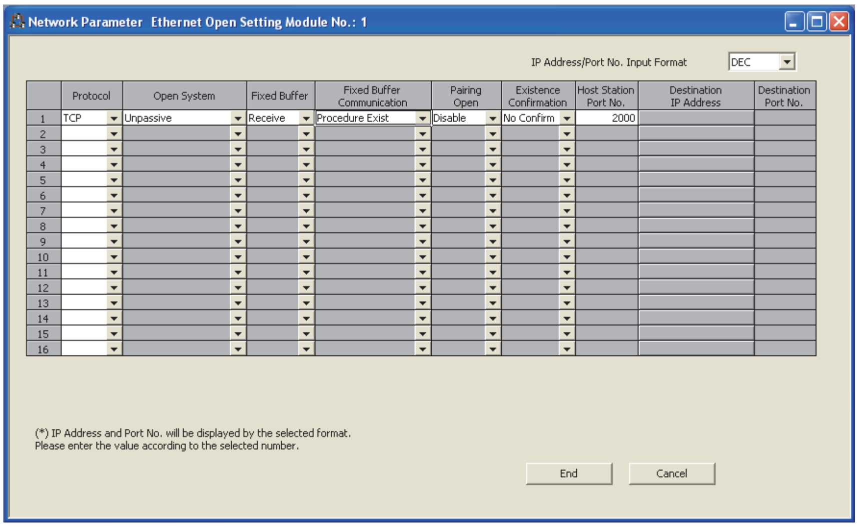
(b) Ethernet operation setting

The following is an example of the Ethernet operation setting.



(c) Open setting

The following is an example of the open setting.



12.9.3 Program

(1) Sample program procedures

This section describes the procedures for a sample program.

(a) Sending side (QCPU-1 side)

The following describes how communications are processed on the sending side (QCPU-1 side).

1. **Set each parameter on a programming tool and write the set parameters to the CPU module. Then reset the CPU module and confirm that the initial process is completed.**
2. **Perform the open process (Active open) of the connection number 1.**
3. **Perform communications using a fixed buffer from the CPU module using the BUFSND instruction.**
4. **After the data is sent, perform the close process of the connection number 1.**

(b) Receiving side (QCPU-2 side)

The following describes how communications are processed on the receiving side (QCPU-2 side).

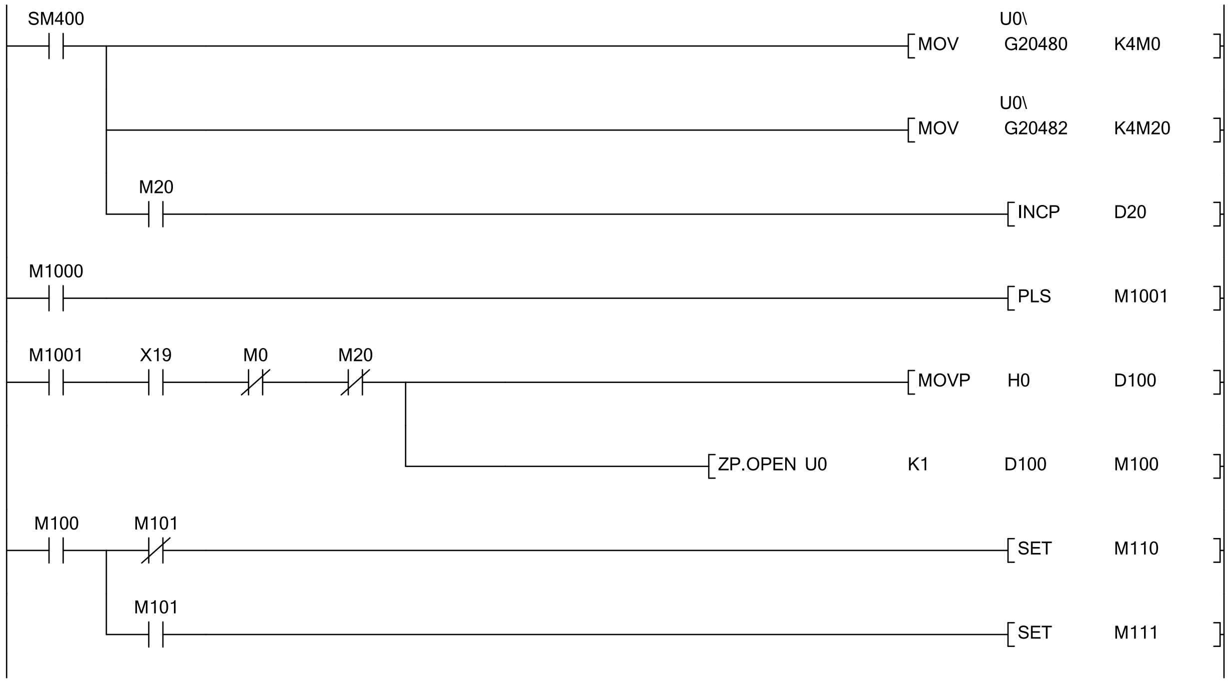
1. **Set each parameter on a programming tool and write the set parameters to the CPU module. Then reset the CPU module and confirm that the initial process is completed. When the initial process is normally completed, the connection number 1 waits for an Active open request from the connected device.**
2. **Perform communications using a fixed buffer from the CPU module using the BUFRCV instruction.**
3. **Data received in the corresponding fixed buffer data area in the E71 is read to the CPU module.**

(2) Sample program

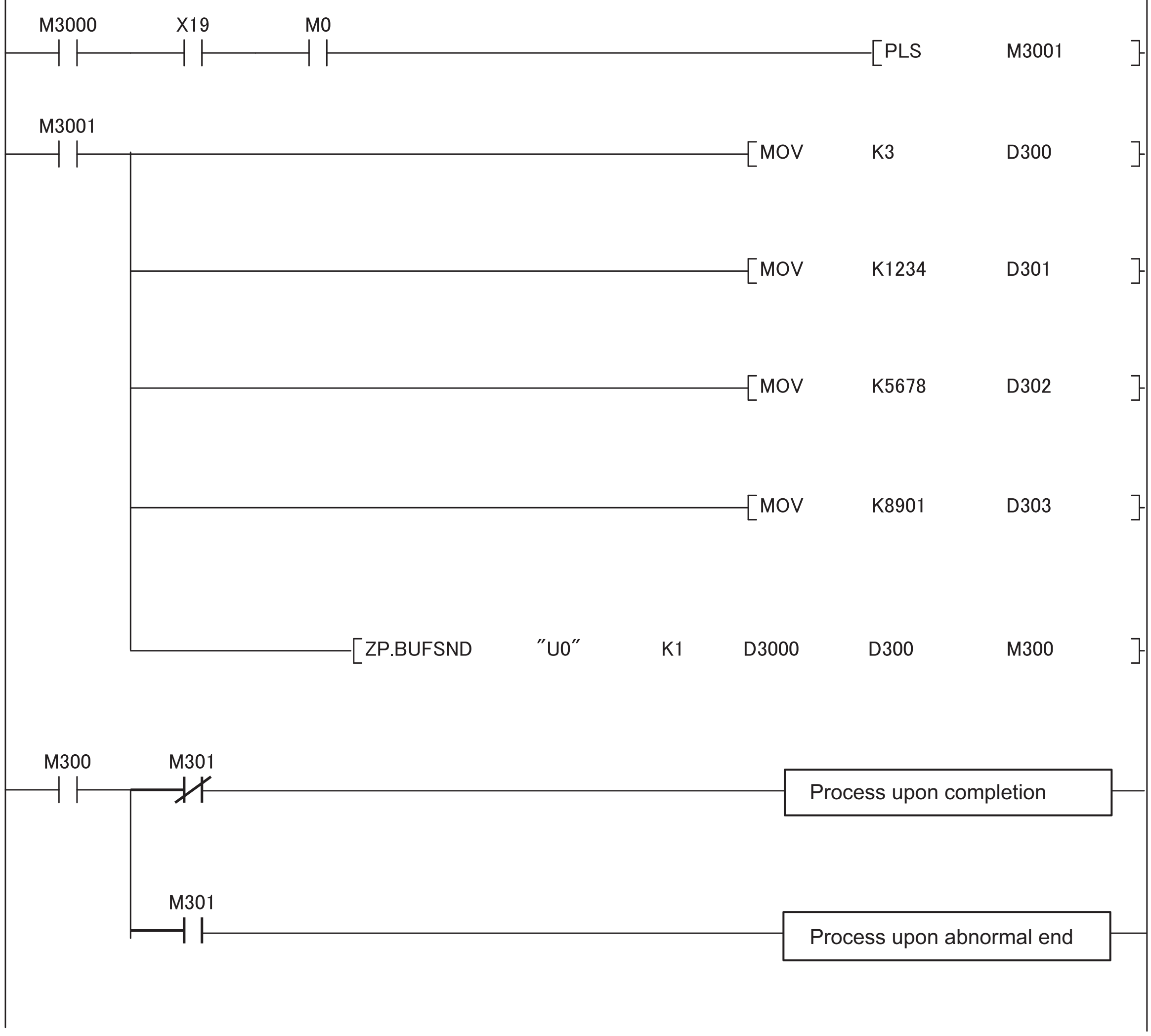
(a) Program on the sending side (QCPU-1 side)

The following is a program on the sending side (QCPU-1 side).

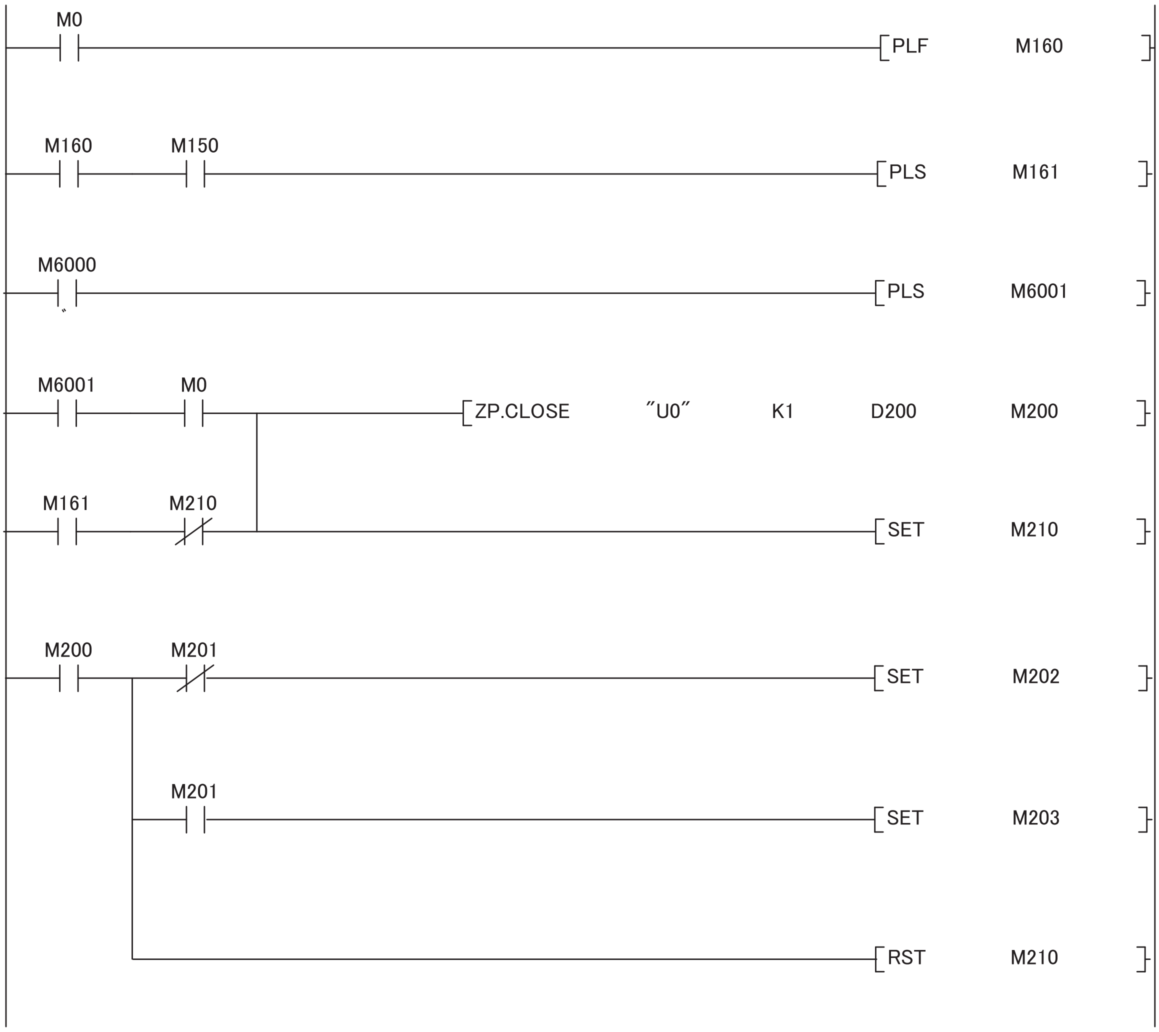
<<Open process program>>



<<Fixed buffer No.1 send program>>



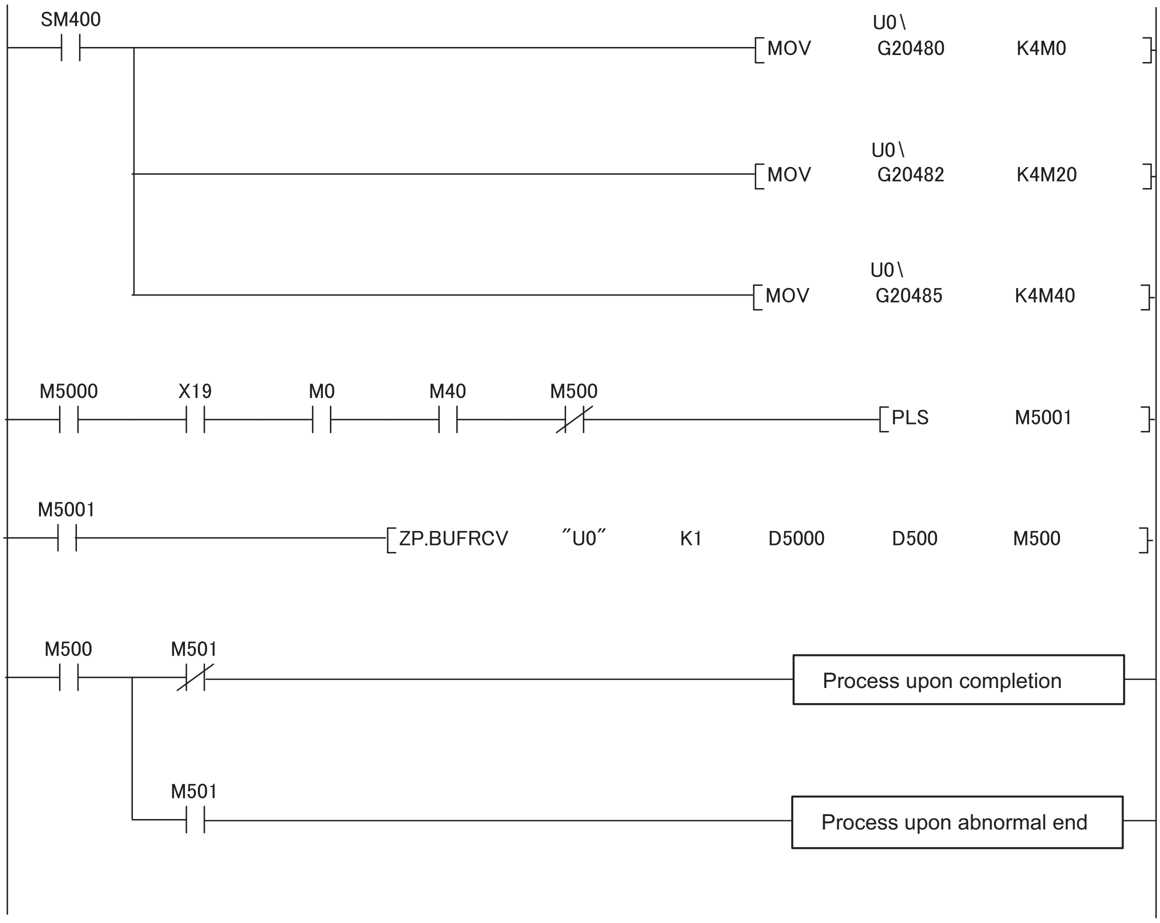
<<Close processing program>>



(b) Receiving side (QCPU-2 side)

The following is a program on the receiving side (QCPU-2 side).

<<Fixed buffer No.1 receive program>>



Point

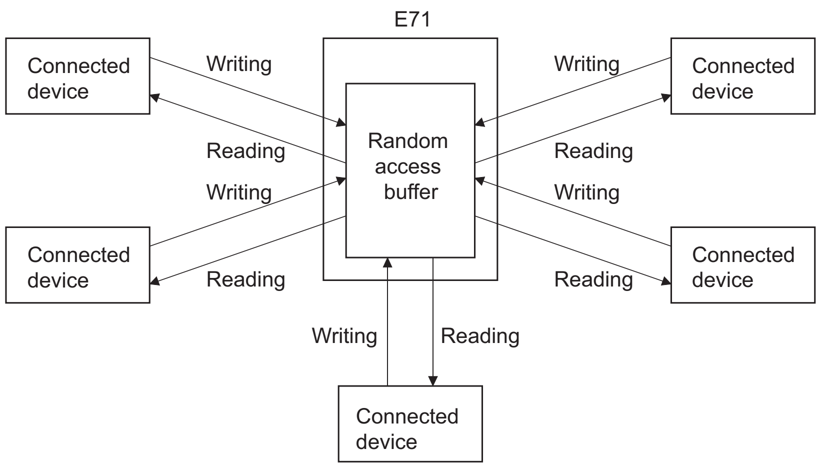
- Secure sufficient device areas according to the maximum length of data sent from the send source to prevent the device areas used for other purposes from being overwritten by the receive data.
- To receive data at shorter intervals than the scan time of the CPU module, add the normally closed contact for the completion device (M500) of the BUFRCV instruction to the execution conditions of the BUFRCV instruction as shown in the above program. When there is no normally closed contact for the completion device (M500), the receive direction (M5001) of the BUFRCV instruction is not turned off and on and the BUFRCV instruction may not be executed.

CHAPTER 13 COMMUNICATIONS USING A RANDOM ACCESS BUFFER

This section describes communications using a random access buffer.

13.1 Applications

In communications using a random access buffer, data can be freely read from and written to any connected device (not including an E71). A random access buffer is used as a common buffer area for all the connected devices in the Ethernet network.



Point

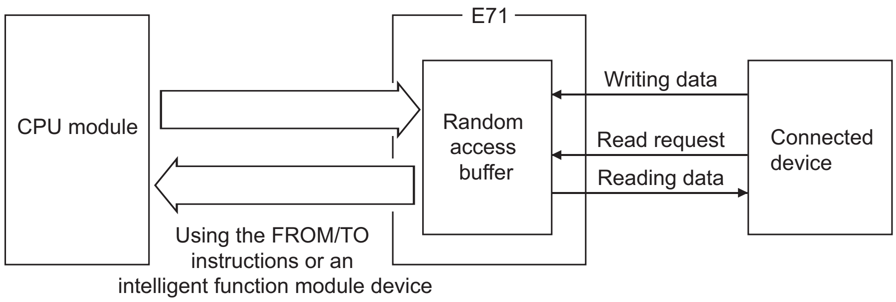
Data communications (sending/receiving) by e-mail using a CPU module cannot be used together with communications using a random access buffer. Only one of the functions can be used at one time. (The e-mail sending function using the E71 programmable controller CPU monitoring function can be used together with communications using a random access buffer.)

13.2 Communication Structure

This section describes the structure of communications using a random access buffer.

(1) Data flow

The data flow in communications using a random access buffer is as follows. The FROM/TO instructions or intelligent function module devices are used to access a random access buffer from a CPU module.



Point

- Communications can be performed only with connected devices where communications using a fixed buffer in the "Procedure Exist" control method are set using the connection with Open completion signal of an E71 on.
- Communications are performed asynchronously with the program. If synchronization is required, use communications using a fixed buffer.

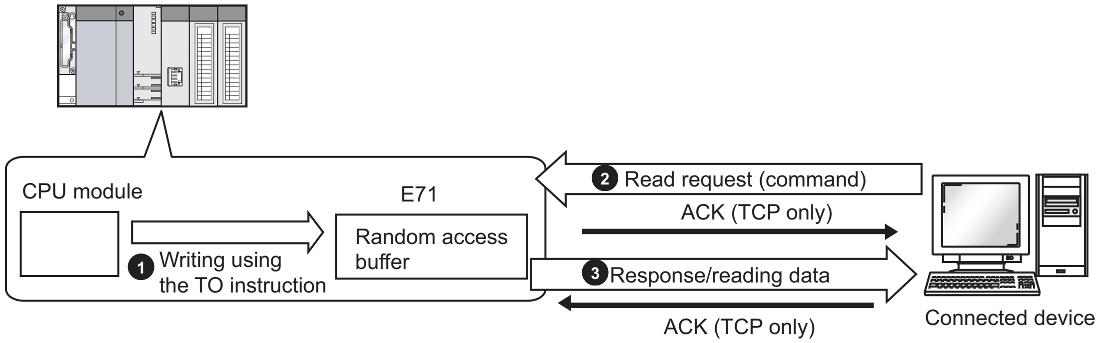
(2) Connected devices where data communications can be performed

Data communications can be performed with the following connected devices:

- Devices in Ethernet where an E71 is connected
- Devices connected using the router relay function

13.2.1 How data is read from a connected device

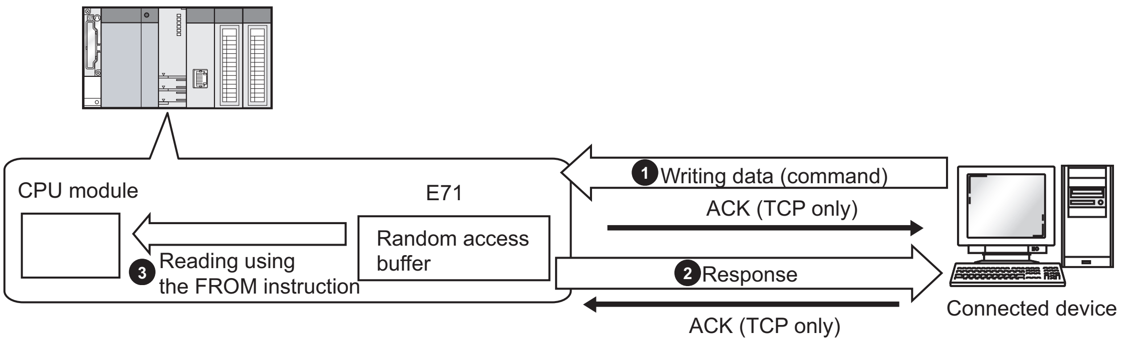
The following figure shows how an E71 sends data in response to a read request sent from a connected device.



1. Write data in the random access buffer of the E71 using the TO instruction.
2. Asynchronously with the process 1 above, the connected device sends a read request to the E71. (The E71 side: command receiving)
3. Upon receiving the read request from the connected device, the E71 sends the data written in the random access buffer to the connected device that sent the read request. (The E71 side: response sending)

13.2.2 How a connected device writes data

The following figure shows how a connected device writes data to the E71 random access buffer.

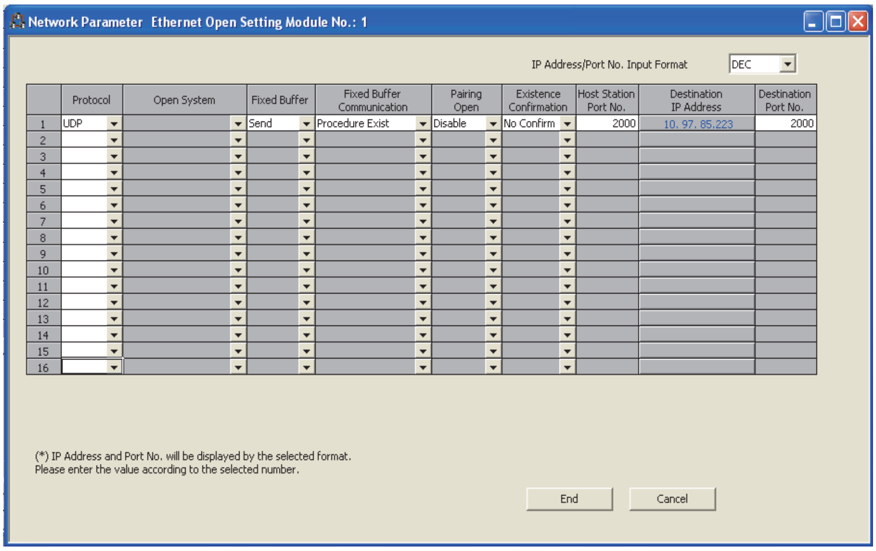


1. The connected device writes data to the random access buffer of the E71. (The E71 side: command receiving)
2. The E71 processes the write request from the connected device and returns the writing result to the connected device that sent the write request. (The E71 side: response sending)
3. Asynchronously with the processes 1 and 2 above, the data written in the random access buffer is read using the FROM instruction.

13.3 Parameter Setting

Set the following parameters to perform communications using a random access buffer.

- Basic setting (Page 84, Section 7.1.2)
- Ethernet operation setting (Page 85, Section 7.1.3)
- Open setting (Page 87, Section 7.1.4)



Item	Description	Setting range
Fixed Buffer Communication Procedure	Select the communication method for communications using a fixed buffer. Select "Procedure Exist" for communications using a random access buffer.	Procedure Exist

13.4 Data Format

Communication data consists of a header and application data.

13.4.1 Header

The header for TCP/IP or UDP/IP is used. Because an E71 automatically adds and deletes a header, the user setting is not required.

(1) Detailed header sizes

(a) TCP/IP

Ethernet (14 bytes)	IP (20 bytes)	TCP (20 bytes)

(b) UDP/IP

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

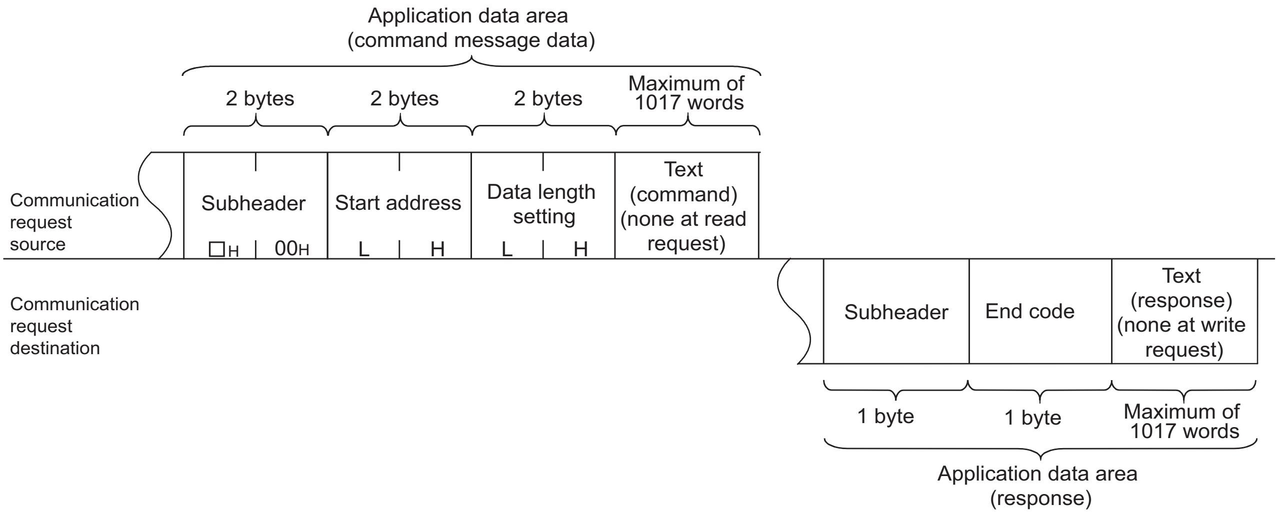
13.4.2 Application data

In the application data for communications with the "Procedure Exist" control method, the following data code is expressed in a binary code or an ASCII code. To switch between a binary code and an ASCII code, use the Ethernet operation setting.

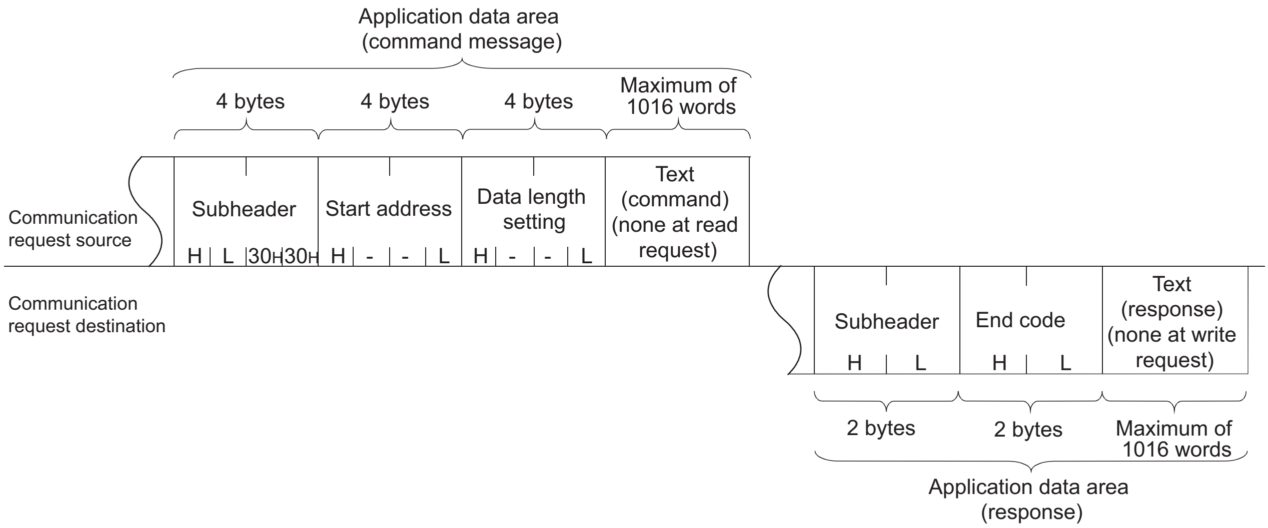
Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ "Operation Setting"

(1) Format

• Communications using a binary code

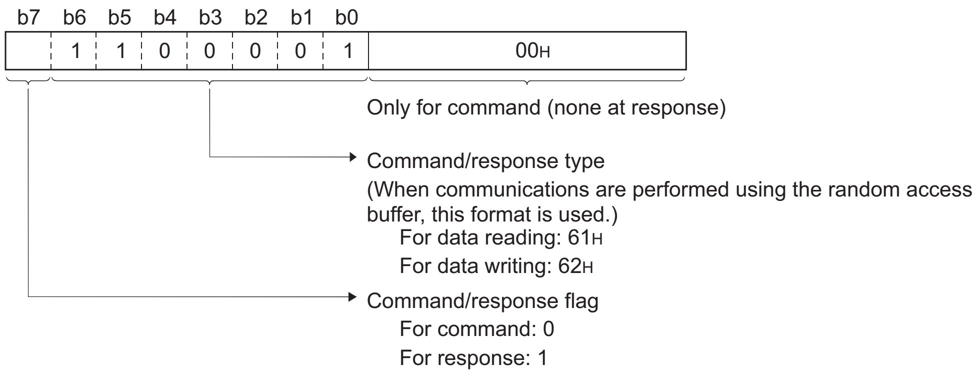


• Communications using an ASCII code

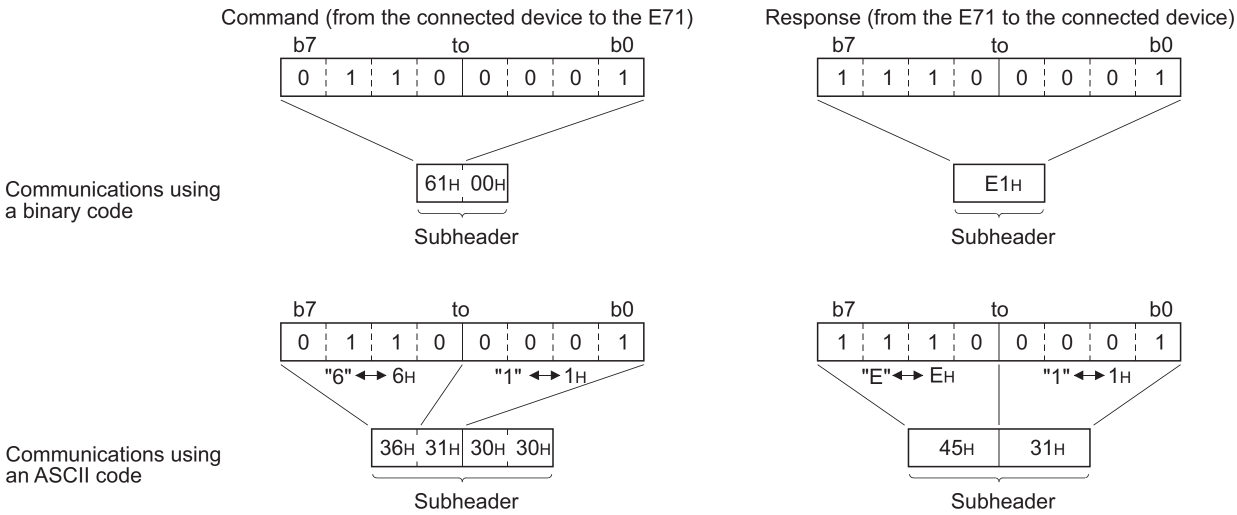


(2) Subheader

Because an E71 automatically adds and deletes a header, the user setting is not required.



(a) When reading

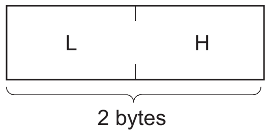


(3) Start address

Use a logical address to set the start address of a random access buffer that reads/writes data. (Page 187, Section 13.6)

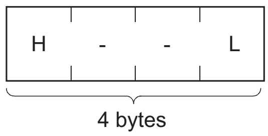
(a) Communications using a binary code

Specify the start address using a binary value.



(b) Communications using an ASCII code

Specify the start address using an ASCII code value expressed in hexadecimal.

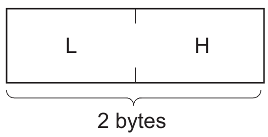


(4) Data length setting

Set the number of words of the read/written data in the random access buffer range.

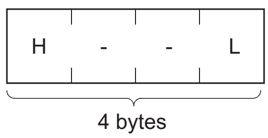
(a) Communications using a binary code

Specify the number of words using a binary value. (Maximum of 1017 words)



(b) Communications using an ASCII code

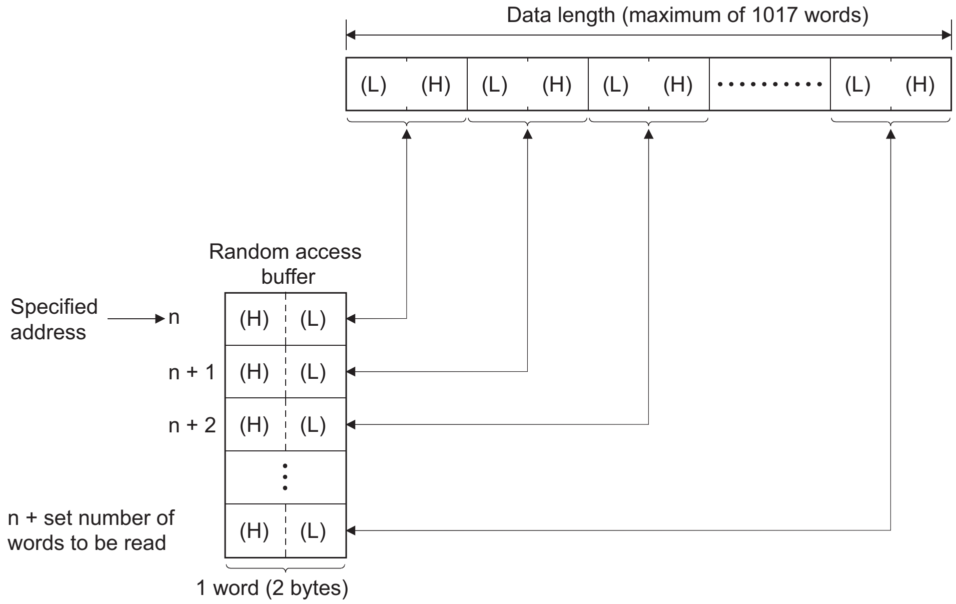
Specify the number of words using an ASCII code value expressed in hexadecimal. (Maximum of 508 words)



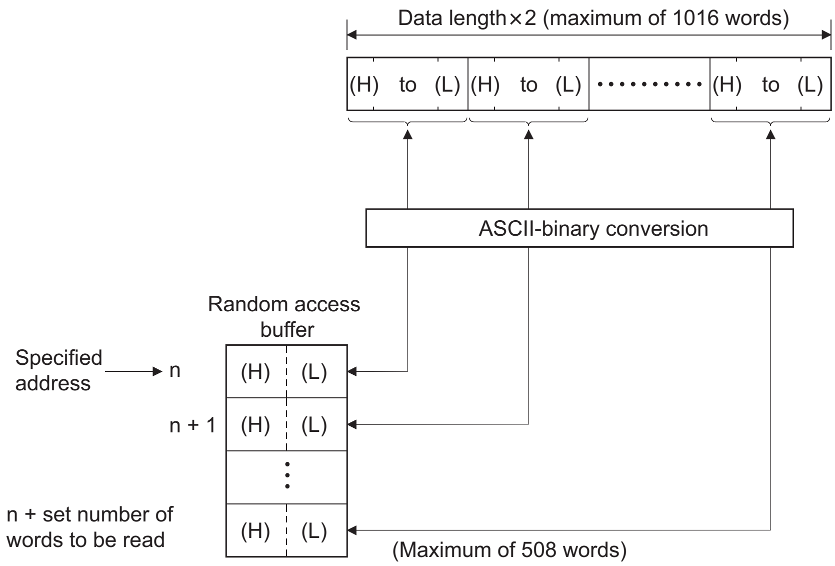
(5) Text

Text is a data written to and read from a random access buffer.

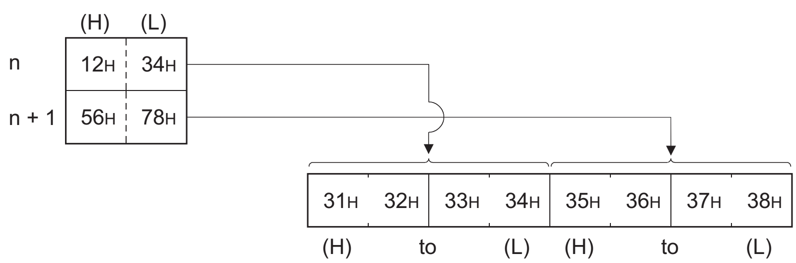
(a) Communications using a binary code



(b) Communications using an ASCII code



Ex.



(6) End code

An error code is stored in the end code added to a response. For the error codes, refer to the error code list.
(Page 310, Section 16.6.1)
End codes are stored in the communication status storage area of the buffer memory.

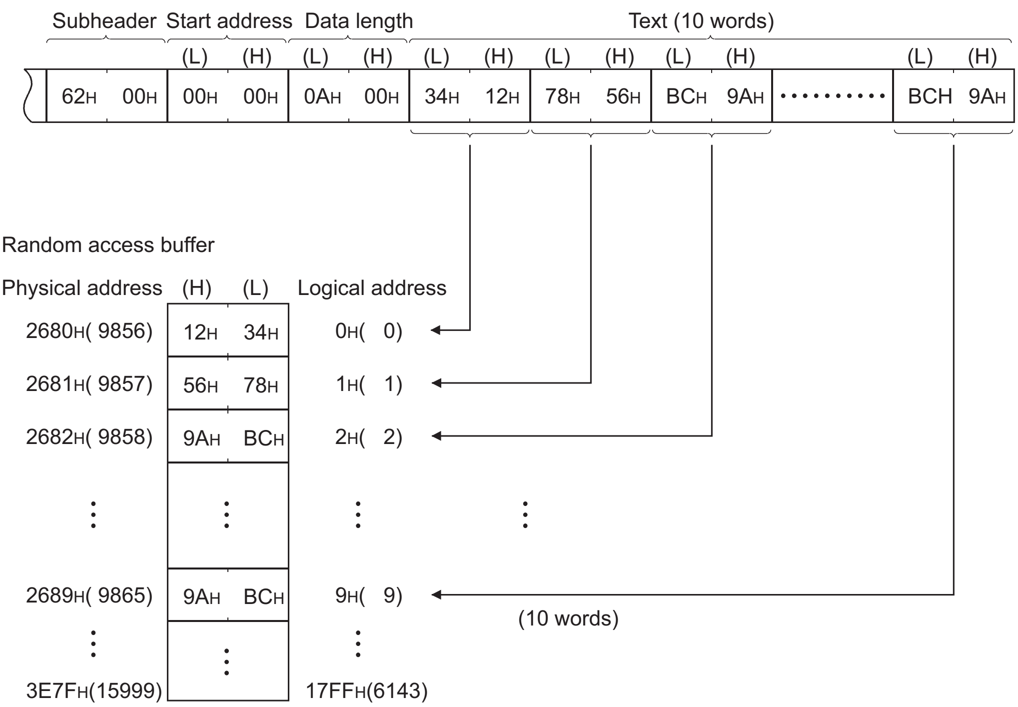
13.4.3 Examples of command and response formats

This section describes examples of command and response formats.

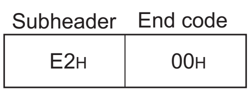
(1) Writing data to a random access buffer upon a write request from a connected device

(a) Communications using a binary code

- Command format (from the connected device to the E71)

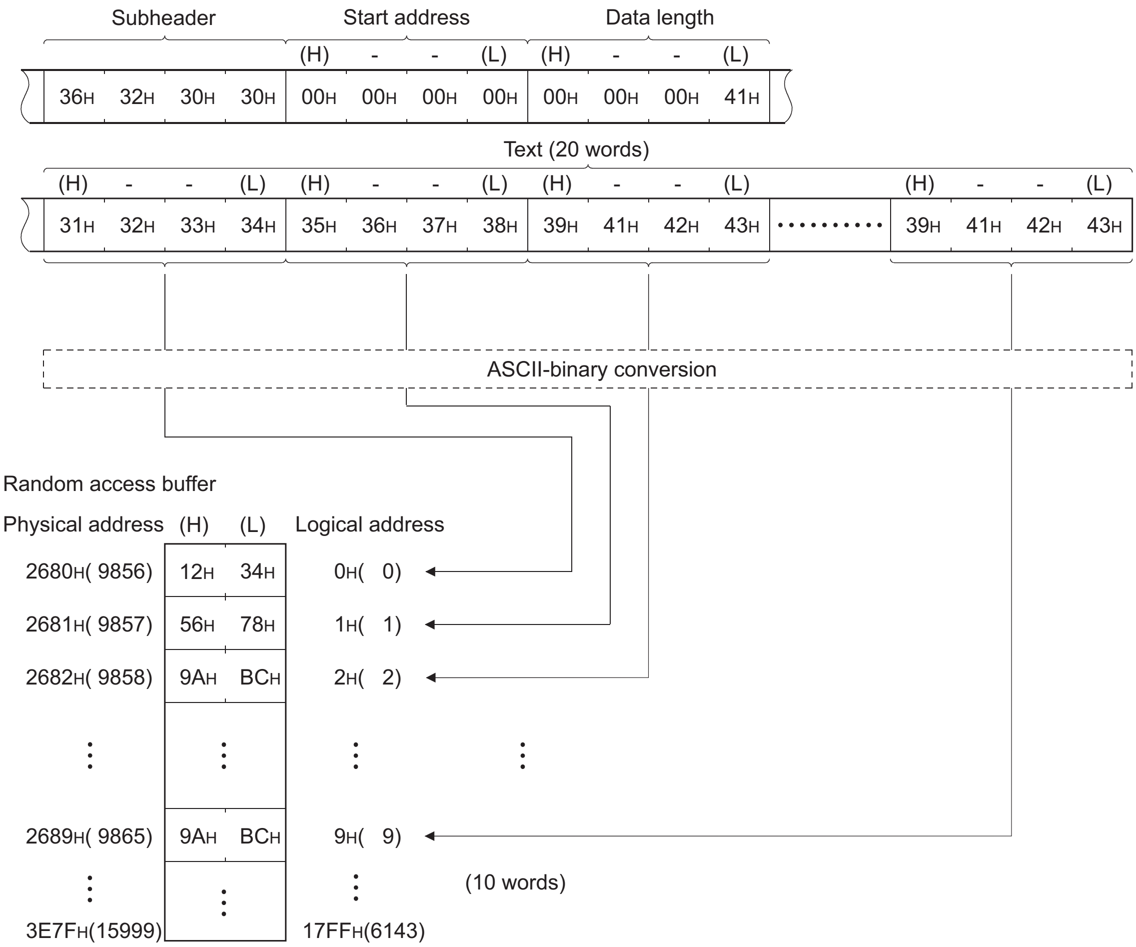


- Response format (from the E71 to the connected device)



(b) Communications using an ASCII code

- Command format (from the connected device to the E71)



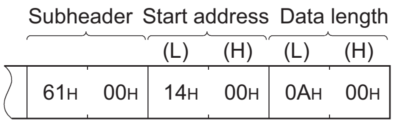
- Response format (from the E71 to the connected device)

Subheader		End code	
45H	32H	30H	30H

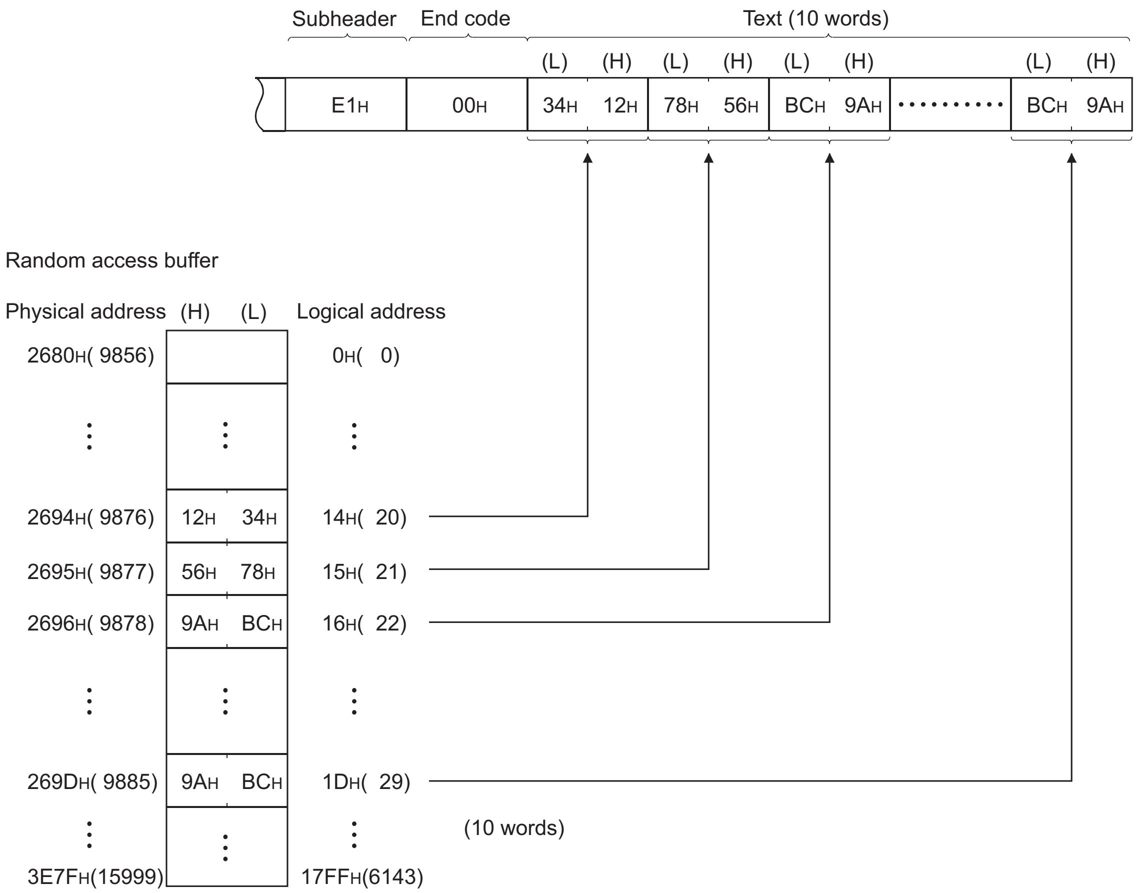
(2) Reading data from a random access buffer upon a read request from a connected device

(a) Communications using a binary code

• Command format (from the connected device to the E71)

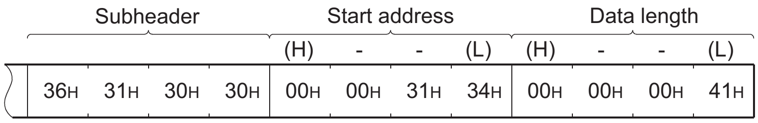


• Response format (from the E71 to the connected device)

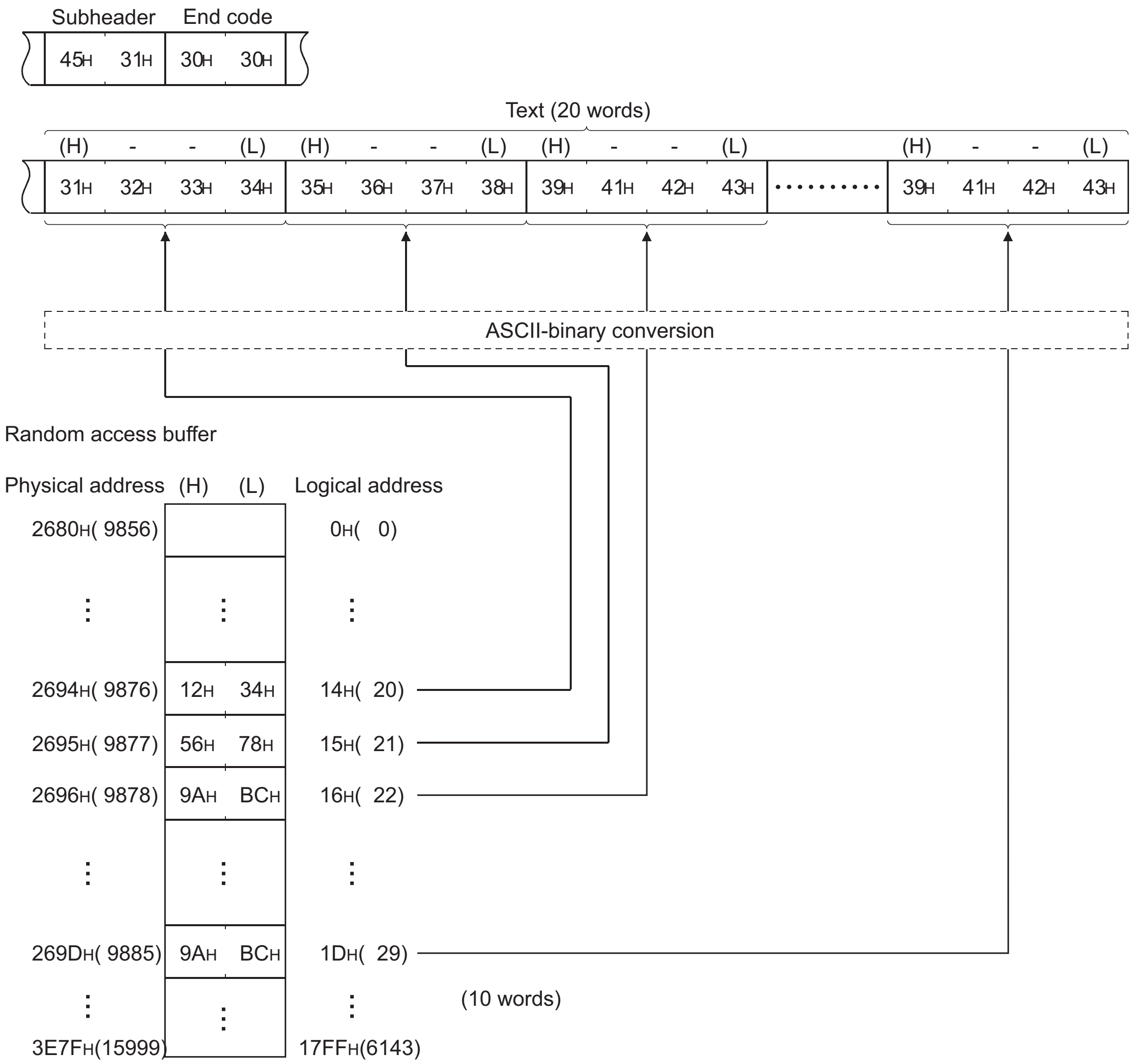


(b) Communications using an ASCII code

- Command format (from the connected device to the E71)



- Command format (from the connected device to the E71)



13.5 Precautions when Creating Programs

This section describes the precautions when creating programs for communications using a random access buffer.

(1) Completion of the initial and open processes

The initial process and the connection open process need to be completed.

(2) Send request from a CPU module

A CPU module cannot issue send requests. In addition, reception completion in a CPU module is not confirmed. When it is necessary to synchronize data sending/receiving between the CPU module and the connected device, use communications using a fixed buffer.

(3) Random access buffer address

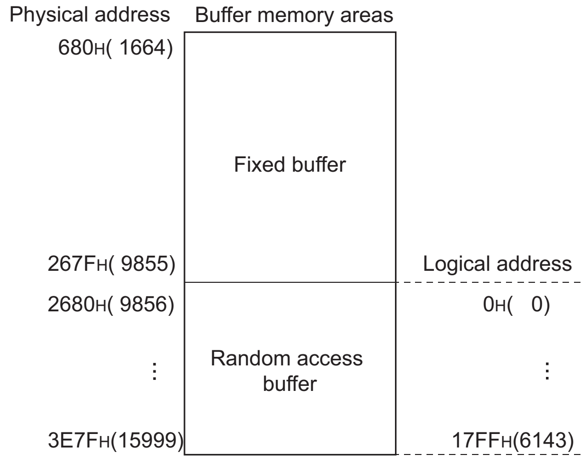
The address specified by the connected device and the address specified using the FROM/TO instructions are different. For details, refer to the physical and logical addresses of a random access buffer. (Page 187, Section 13.6)

13.6 Physical and Logical Addresses of a Random Access Buffer

This section describes the start address of the E71 random access buffer (without battery backup), which is specified in commands.

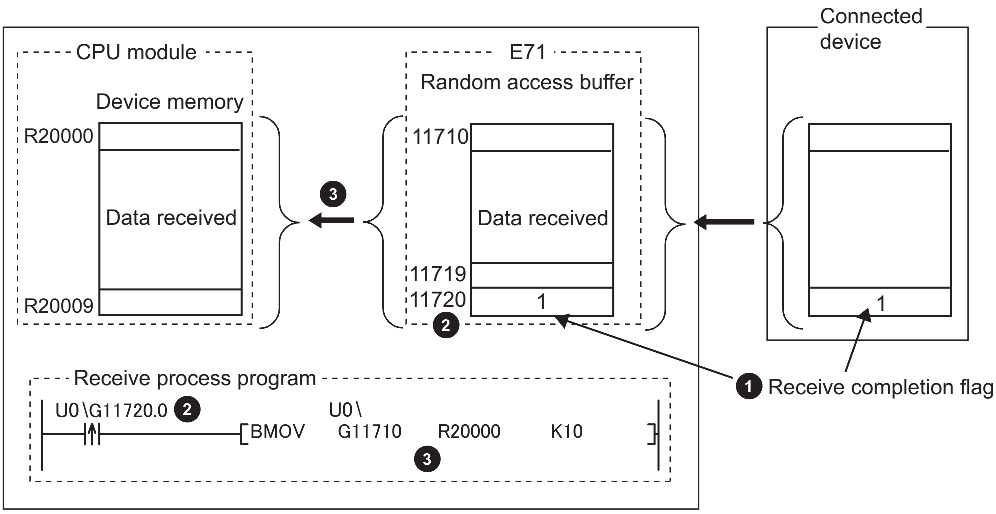
Note that when designating the start address of a random access buffer, the address specified by a connected device is different from that specified using the FROM/TO instructions.

- Physical address: Address specified with the FROM/TO instructions of a program
- Logical address: Address specified by a connected device as the start address item in the command



13.7 Example of Communications Using a Random Access Buffer

The following figure shows an example of a write operation from a connected device.



1. The receive completion flag is provided at the end of the received data.
2. The receive completion flag is monitored.
3. When the receive completion flag turns on, the data is transmitted from the random access buffer to the CPU module.

CHAPTER 14 OTHER FUNCTIONS

This chapter describes basic functions of the E71 other than those described in the preceding chapters.

14.1 Router Relay Function

This section describes the router relay function.

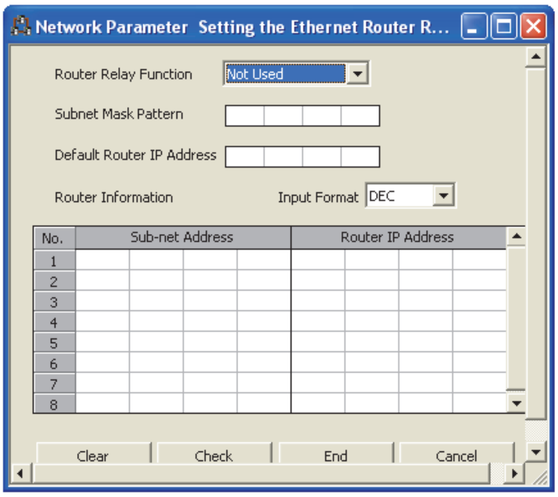
14.1.1 Applications

This function allows the E71 to communicate with connected devices on other Ethernet networks via a router and gateway. This function does not allow the E71 to operate as a router. One default router and up to eight routers can be set.

14.1.2 Parameter settings

Configure the settings in the following window.

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ Select "Ethernet" under "Network Type". ⇨ "Operation Setting"



Item		Description	Setting range
Router Relay Function		Select whether to use the router relay function.	Not Used/Use
Subnet Mask Pattern		Refer to (1) in this section.	$C0000000_H$ to $FFFFFFFC_H$
Default Router IP Address		Refer to (2) in this section.	A value other than 00000000_H and $FFFFFFFF_H$
Input Format		Select the input format of router information.	DEC/HEX
Router Information	Sub-net Address	Refer to (3) (a) in this section.	A value other than 00000000_H and $FFFFFFFF_H$
	Router IP Address	Refer to (3) (b) in this section.	A value other than 00000000_H and $FFFFFFFF_H$

(1) Subnet mask pattern

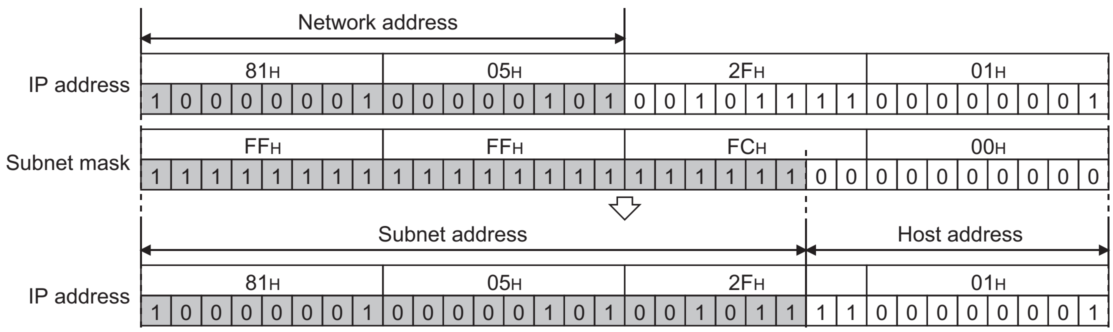
Set the subnet mask. (Consult with the network administrator for the setting.) When not using the subnet mask, set any of the following values according to the class.

Class	Mask value
Class A	$FF000000_H$
Class B	$FFFF0000_H$
Class C	$FFFFFF00_H$

(a) Setting example

The following shows a Class B setting example.

Ex. Class B



Point

- All devices on the same subnetwork must have the common subnet mask.
- When not managed by the subnetwork, the connected devices need not have subnet masks. (Set the network address of the corresponding class.)

(2) Default router IP address

Set the IP address of the router (default router) when the E71 communicates with the connected devices on other Ethernet networks via a router other than the one specified in the router information. Set the value that satisfies the following conditions.

- Condition 1: The IP address class is any of A, B, and C.
- Condition 2: The subnet address of the default router is the same as that of the E71 on the own station.
- Condition 3: The host address bits are not all "0" or all "1".

Point

If the corresponding subnet address does not exist in the router information when the connection is opened or data communications are performed, communications are performed via the default router.

(3) Router information

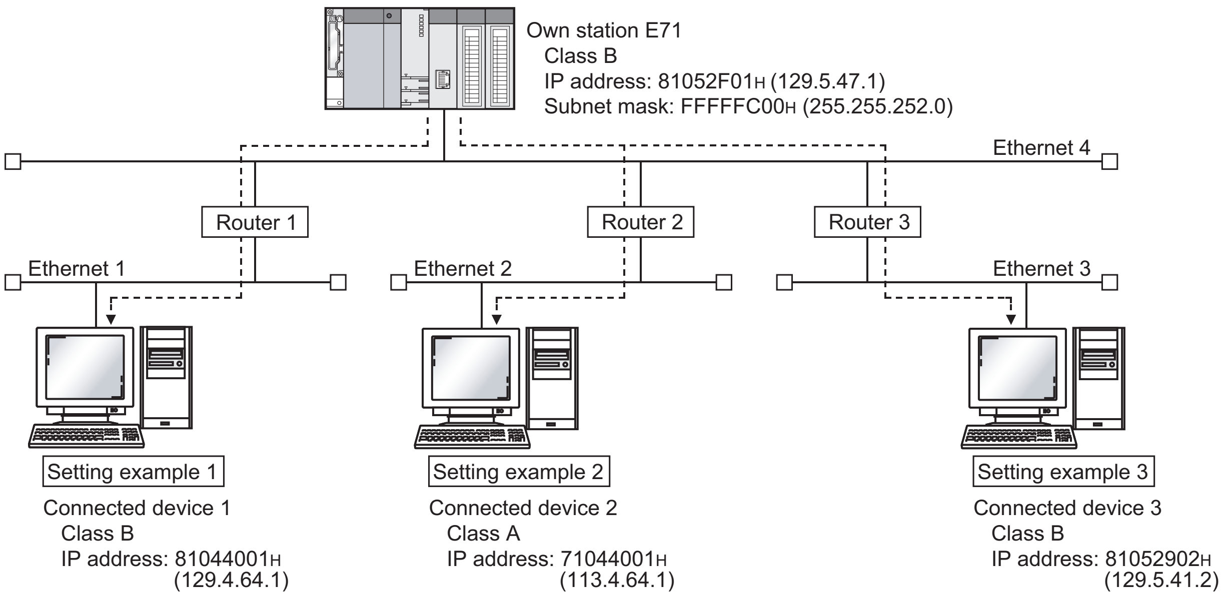
(a) Subnet address

Set the network address*1 or subnet address*2 of the connected device when the E71 communicates with the connected devices on other Ethernet networks via a router other than the default router. Set the value that satisfies the following conditions.

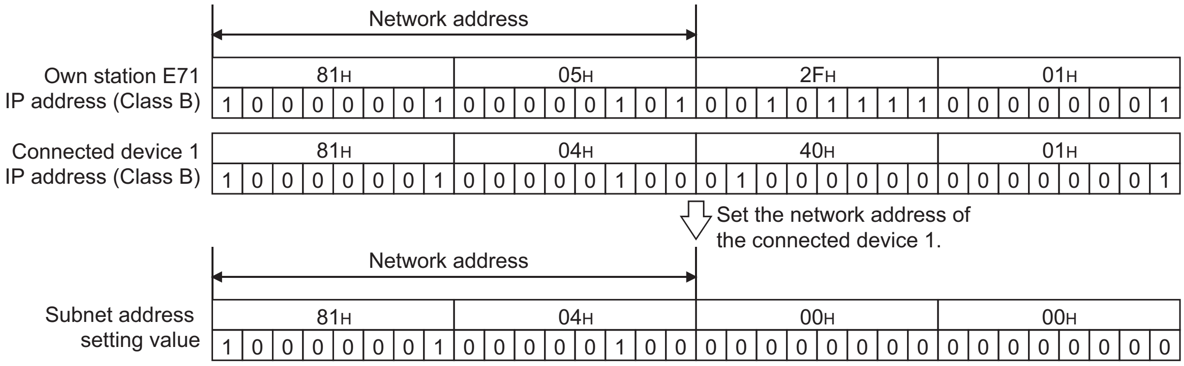
- Condition 1: The IP address class is any of A, B, and C.
- Condition 2: The host address bits are all "0".

*1 If the class (network address) of the E71 on the own station differs from that of the connected device, set the network address of the connected device.

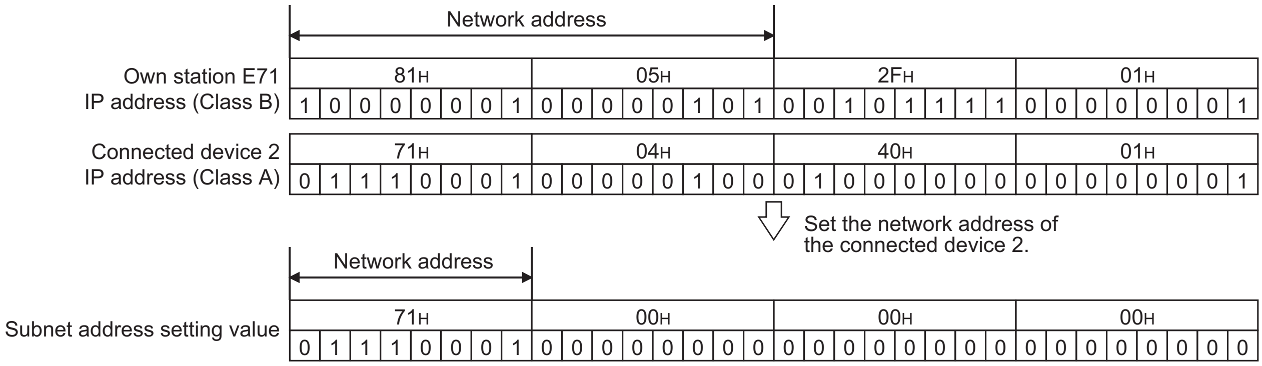
*2 If the class (network address) of the E71 on the own station is the same as that of the connected device, set the subnet address of the connected device.



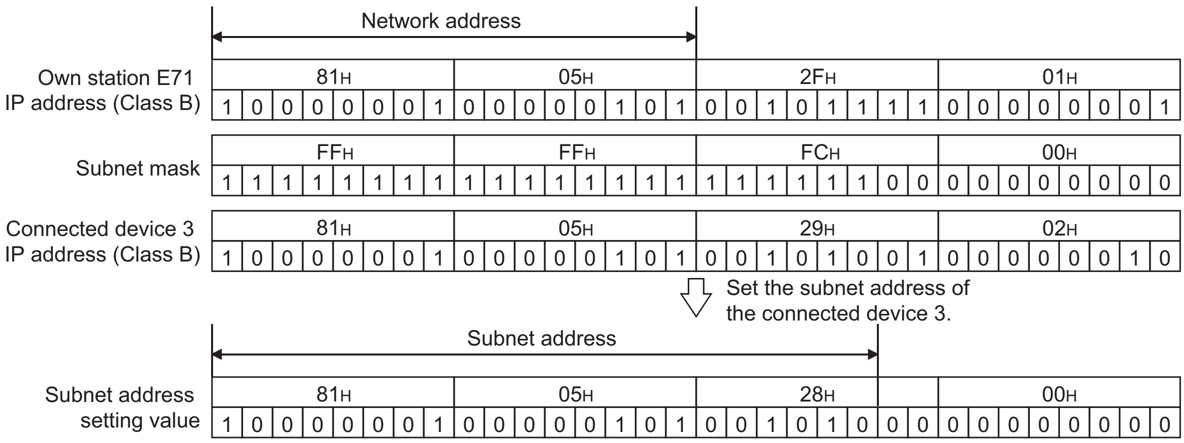
Ex. When the network addresses differ between the E71 on the own station and the connected device



Ex. When the classes differ between the E71 on the own station and the connected device



Ex. When the network address of the E71 on the own station is the same as that of the connected device



(b) Router IP address

Set the IP addresses of the routers when the E71 communicates with the connected devices on other Ethernet networks via a router other than the default router. Set the value that satisfies the following conditions.

- Condition 1: The IP address class is any of A, B, and C.
- Condition 2: The subnet address of the router is the same as that of the E71 on the own station.
- Condition 3: The host address bits are not all "0" or all "1".

Point

- When the E71 communicates with the connected devices via a router in Passive open status, communications can be performed without using the router relay function.
- The router relay function is not needed in a system that uses the Proxy router.

14.2 Communications Using an Auto-open UDP Port

This section describes communications using the auto-open UDP port.

14.2.1 Application

The auto-open UDP port is a UDP/IP port that automatically opens and closes at the following timing. Using this port makes the E71 ready for communications upon completion of an initial process, thus enabling communications without using any programs regardless of the open status of connection numbers 1 to 16.

(1) Open and close timing

The port automatically opens according to the registered parameter settings upon completion of the initial process of the E71. In addition, it automatically closes when the E71-mounted station is powered off or reset.

Point

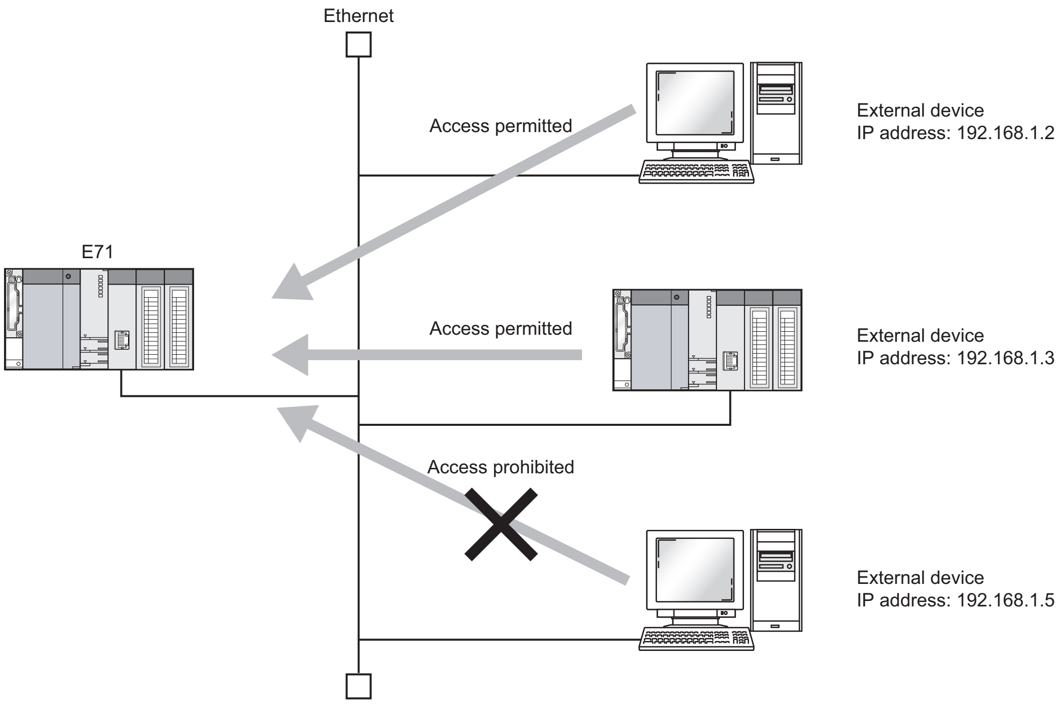
- The E71 enables communications using an auto-open UDP port after the initial process is normally completed, and the E71 on the own station waits for communication requests to itself (auto open).
- The E71 accepts requests from anywhere as long as they are addressed to the E71 itself.
- When a communication request is accepted from the connected device, the corresponding port number is occupied until the process is completed. Even if the next communication request is accepted during this time, the communication process will be waited.
- To change the auto-open UDP port number, a reinitialization process is required. (Page 354, Appendix 4)

14.3 IP Filter Function

This section describes the IP filter function.
This function is available only in the E71 with the serial number (first five digits) of "18072" or later.

14.3.1 Application

This function identifies the IP address of the access source, and prevents unauthorized access performed by IP address specification.
The IP address of the connected device to be allowed or denied access is set in the buffer memory, and access from connected devices is restricted.
Use of this function is recommended when using the E71 in an environment connected to a LAN line.



Remark

When a network other than Ethernet is used as the communication path, access cannot be restricted.

Point

The IP filter function is one method of preventing unauthorized access (such as program or data destruction) from external devices. However, this function does not guarantee prevention of all unauthorized access. To have the programmable controller system fully secured against unauthorized access from the external devices, take additional measures. Mitsubishi Electric Corporation does not hold any responsibility for any system problems caused by unauthorized access.
The following are examples of measures against unauthorized access.

- Install a firewall.
- Install a personal computer as a relay station and control the relay of communication data using an application program.
- Install an external device for which the access rights can be controlled as a relay station (Contact the network provider or equipment dealer for details on the external devices for which access rights can be controlled.)

14.3.2 Setting method

This section describes the setting method of IP filter.

(1) Setting procedure

The IP filter settings become enabled after the reinitialization process.

The following is a procedure for IP filter settings.

1. **Confirm the normal completion of the initial process. (Initial normal completion signal (X19): ON)**
2. **Terminate all data communications with the connected devices and perform a close process on all connections.**
3. **Write the settings in the buffer memory areas, IP filter settings (address: 5700_H to 5721_H).**
4. **Perform reinitialization process. (Page 358, Appendix 4.2)**
5. **Confirm the normal completion of the reinitialization process with the following buffer memory areas.**

- Reinitialization specification (address: $1F_H$ (b15)): 0_H
- Initial error code (address: 69_H): 0_H

Point

- When a connected device accesses the E71 before the completion of reinitialization process, the access cannot be denied because the IP filter is not yet enabled.
- The values stored in IP filter monitoring area (address: 5722_H to 5725_H) are cleared when the reinitialization process completes.
- Even if the connection is established as set with the open setting of E71 or the program, access from the connected device is either allowed or denied following IP address settings. Therefore, if the IP address set in the open setting of E71 is set to be denied with the IP filter settings, the IP filter function is enabled and communication with the connected device is denied.
- When the UINI instruction is executed with the IP filter settings written in the buffer memory (address: 5700_H to 5721_H), IP filter settings are applied as in the case of reinitialization process by the buffer memory. When using the UINI instruction, check the setting values in IP filter settings (address: 5700_H to 5721_H) before executing the instruction.

(2) Buffer memory areas used

The following buffer memory areas are used for IP filter settings.

Buffer memory name		Address Decimal (Hexadecimal)	Description
Communication condition setting (Ethernet Operation Setting) area		31 ($1F_H$)	Ethernet operation setting upon reinitialization process is set. Communication data code setting (b1) • 0_H: Binary Code • 1_H: ASCII Code TCP existence confirmation setting (b4) • 0_H: Use the Ping • 1_H: Use the KeepAlive Send frame setting (b5) • 0_H: Ethernet • 1_H: IEEE 802.3 Setting of write enable/disable at RUN time (b6) • 0_H: Disable • 1_H: Enable Initial timing setting (b8) • 0_H: Do not wait for OPEN (Communications impossible at STOP time) • 1_H: Always wait for OPEN (Communications possible at STOP time) Reinitialization specification (b15) • 0_H: Reinitialization process complete (reset by the system) • 1_H: Reinitialization process request (set by the user)
Initial error code		105 (69_H)	The processing results of initial process and reinitialization process are stored. • 0_H: Normal completion • Other than 0_H: Abnormal end (error code)
Use of IP filter settings		22272 (5700_H)	Set when using the IP filter function. • 0_H: Do not use (default) • 1_H: Use
IP filter function type setting		22273 (5701_H)	Select whether to allow or deny the access from the IP addresses specified in IP address setting 1 to 8. • 0_H: Allow (default) • 1_H: Deny
IP address setting 1	Start IP address	22274 to 22275 (5702_H to 5703_H)	Set the IP address to be allowed or denied. When specifying the IP addresses by range, set the start IP address of the range. 22274 (5702_H): Third octet, fourth octet 22275 (5703_H): First octet, second octet • 00000000_H: No setting (default) • 00000001_H to $DFFFFFFE_H$ (0.0.0.1 to 223.255.255.254)
	End IP address	22276 to 22277 (5704_H to 5705_H)	Set the end IP address of a range to be allowed or denied. When not specifying by range, set 00000000_H. 22276 (5704_H): Third octet, fourth octet 22277 (5705_H): First octet, second octet • 00000000_H: No setting/No range specification (default) • 00000001_H to $DFFFFFFE_H$ (0.0.0.1 to 223.255.255.254)
IP address setting 2 to 8		22278 to 22305 (5706_H to 5721_H)	The configuration is the same as IP address setting 1.

Buffer memory name	Address Decimal (Hexadecimal)	Description
Number of access denied by the IP filter function	22306 to 22307 (5722_H to 5723_H)	The number of times an access is denied by the IP filter function. • 0_H: None (default) • 1_H to $FFFFFFFF_H$ (1 to 4294967295): Number of access denied (When this value exceeds 4294967295, it returns to "1" and starts counting again.)
IP address denied by the IP filter function	22308 to 22309 (5724_H to 5725_H)	The latest IP address denied by the IP filter function is stored. 22308 (5724_H): Third octet, fourth octet 22309 (5725_H): First octet, second octet • 0_H: None (default) • Other than 0_H: IP address denied by the IP filter function

Remark

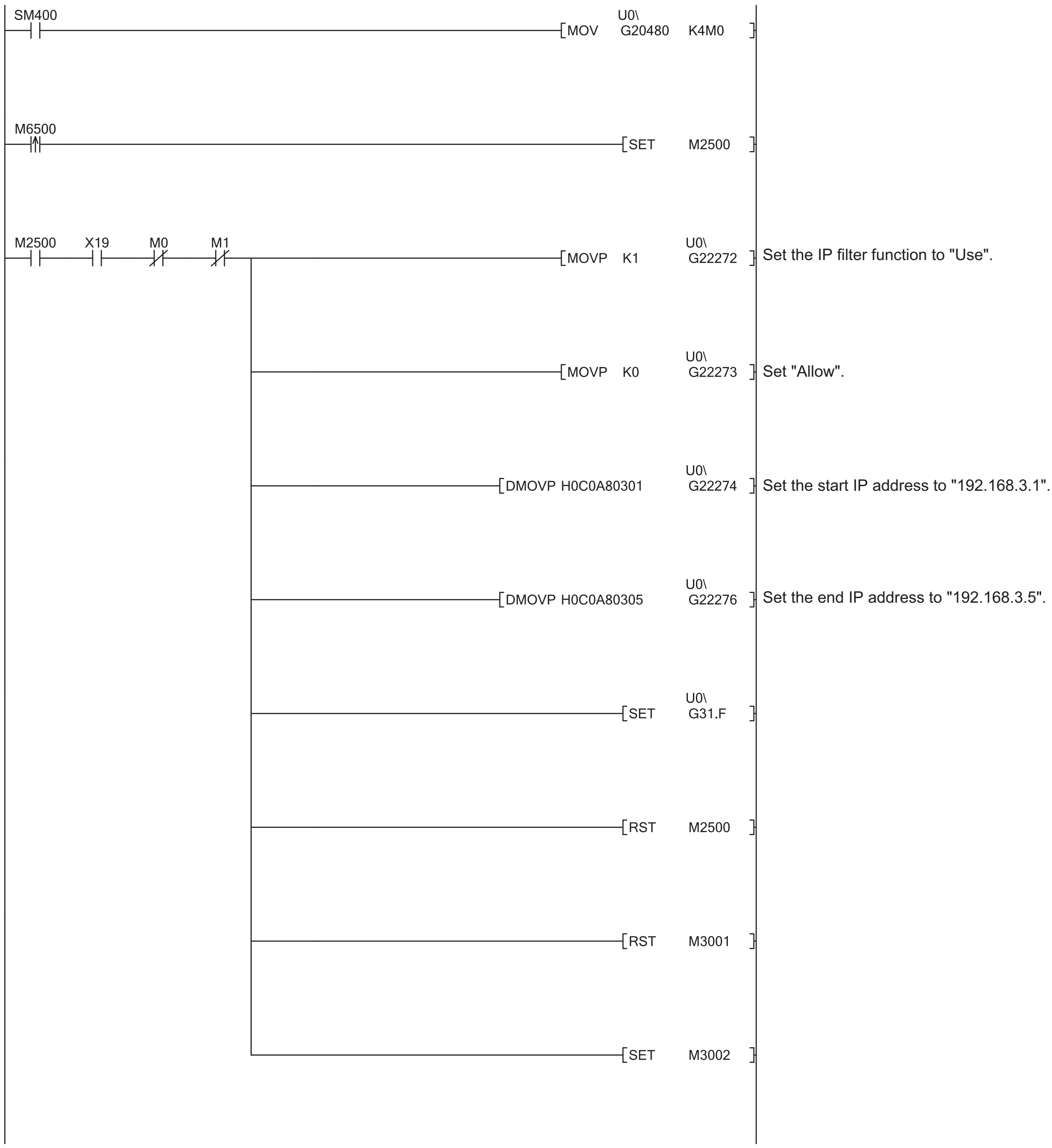
For example, when the IP address 192.168.3.40 is set to be allowed or denied as the start IP address in IP address setting 1, the stored values in the buffer memory will be as follows.

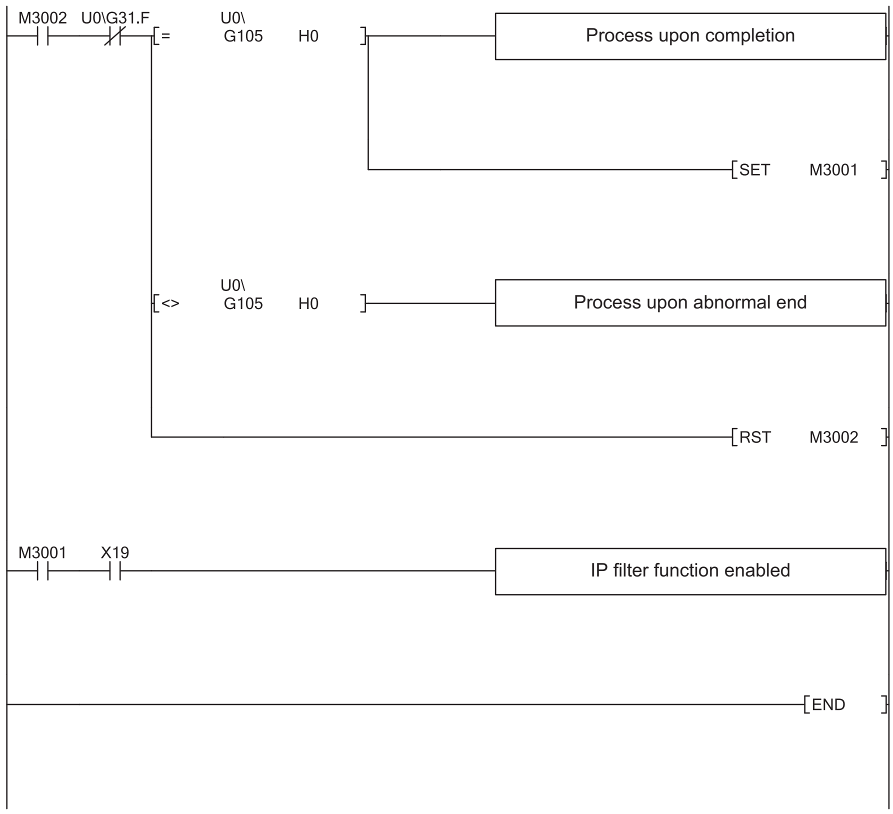
- 22274 (5702_H): 0328_H
- 22275 (5703_H): $C0A8_H$

14.3.3 Program example

The following is a program example that allows access only from the IP addresses 192.168.3.1 to 192.168.3.5. (When the I/O signals of the E71 are X/Y00 to X/Y1F)

(1) Sample program





Remark

This is a sample program for communications using connection numbers 1 and 2. When using another connection number, specify the corresponding signals and bits.

14.3.4 Precautions

- If there is a proxy server in the LAN line, deny access from the IP address of the proxy server. If it is allowed, it will not be possible to prevent access from personal computers that access the proxy server.
- Powering off and on or resetting the CPU module will clear the IP filter settings. When the CPU module is powered off and on or reset, set the values in the buffer memory (address: 5700_H to 5721_H) again and perform the reinitialization process again.

14.4 Remote Password

The CPU module can be protected by a remote password.

Point

A remote password is one method of preventing unauthorized access (such as program or data destruction) from external devices. However, this function does not guarantee prevention of all unauthorized access. To have the programmable controller system fully secured against unauthorized access from the external devices, take additional measures. Mitsubishi Electric Corporation does not hold any responsibility for any system problems caused by unauthorized access. The following are examples of measures against unauthorized access.

- Install a firewall.
- Install a personal computer as a relay station and control the relay of communication data using an application program.
- Install an external device for which the access rights can be controlled as a relay station (Contact the network provider or equipment dealer for details on the external devices for which access rights can be controlled.)

14.4.1 Application

This function allows or prohibits access from connected devices to the CPU module through the following modules. This prevents remote unauthorized access to the CPU module.

- E71
- Built-in Ethernet port QCPU
- C24

14.4.2 Remote password setting processes (unlock and lock processes)

This section describes the processes that enable/disable access from the connected devices to the programmable controller.

(1) Access permission process (unlock process)

- To access the specified CPU module, the connected device performs a remote password unlock process for the remote password-protected E71 on the station in direct connection (own station). The following are unlock process methods.
  - Dedicated command of the MC protocol (remote password (unlock): 1630)
  - When the file transfer (FTP server) function is in use: Dedicated FTP command (password-unlock)
  - Programming tool: Input of the remote password in the window
  - When the Web function is in use: Input of a remote password in the window displayed on the Web browser
- When the unlock process is not performed, the remote password-set E71 that has accepted a communication request checks an entered remote password and disables access to the specified station.
- All data received prior to the unlock process is treated as an error.

(2) Access process

- Access to the specified station is enabled when the remote password unlock process is completed normally.
- Specify the station to be accessed and access the station.

(3) Access prohibition process (lock process)

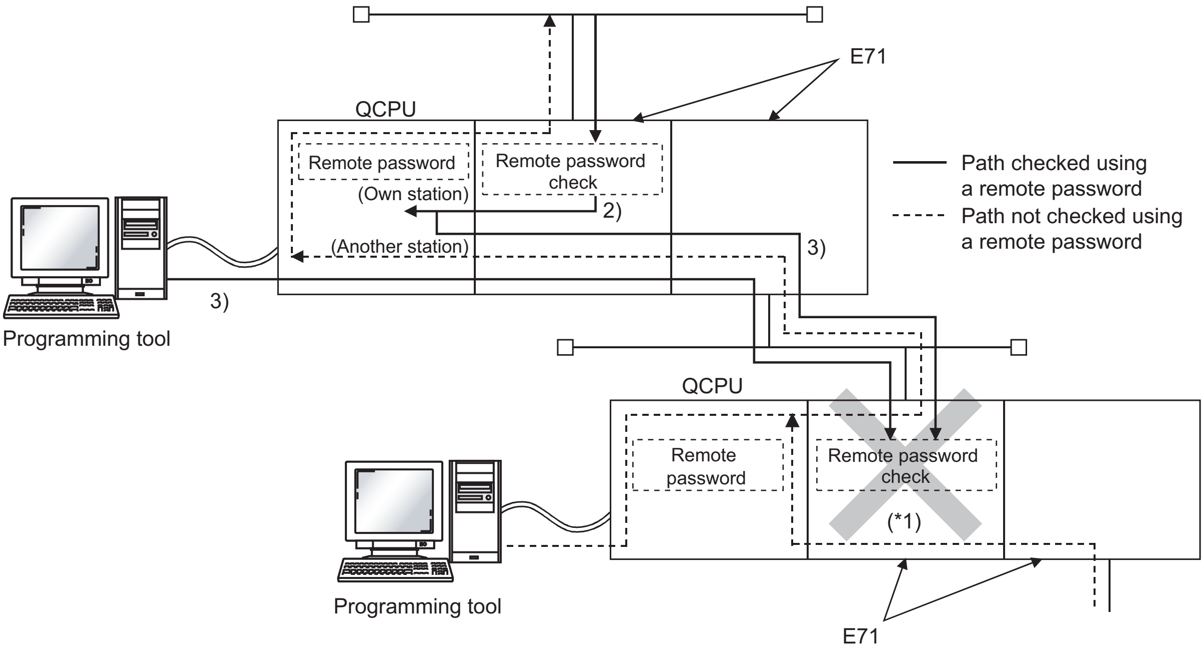
- To terminate the access to the specified station, the connected device performs the remote password lock process to disable subsequent access. The following are lock process methods.
  - Dedicated command of the MC protocol (remote password (lock): 1631)
  - When the file transfer (FTP server) function is in use: Dedicated FTP command (password-lock)
  - Programming tool: Automatically executes the lock process.
  - When the Web function is in use: Automatically performs a lock process when the Web browser is closed.

14.4.3 Remote password check procedure

This section describes the procedure of the remote password check performed by the E71.

(1) Communications in which an entered remote password is checked

- When the following parameters have been set for the E71 mounted on the QCPU station, the E71 checks a remote password for communication requests listed below.
  - When a remote password has been set in the CPU module
  - When the connection where the E71 communicates with a connected device has been set as the target for the remote password check
- The E71 checks an entered remote password for a communication request addressed to the own station/another station received from a connected device.
- The E71 performs a send process upon the following send requests without checking an entered remote password.
  - Send request from the CPU module on the own station (such as communications using a fixed buffer)
  - Communication request from the connected device (including the programming tool connected to the CPU module on the own station) to send data to another station upon request from the CPU module



*1 A communication request from the connected device cannot be accepted because a remote password check has been set. If the remote password check has not been set, the communication request is accepted, and the E71 can communicate with the connected devices.

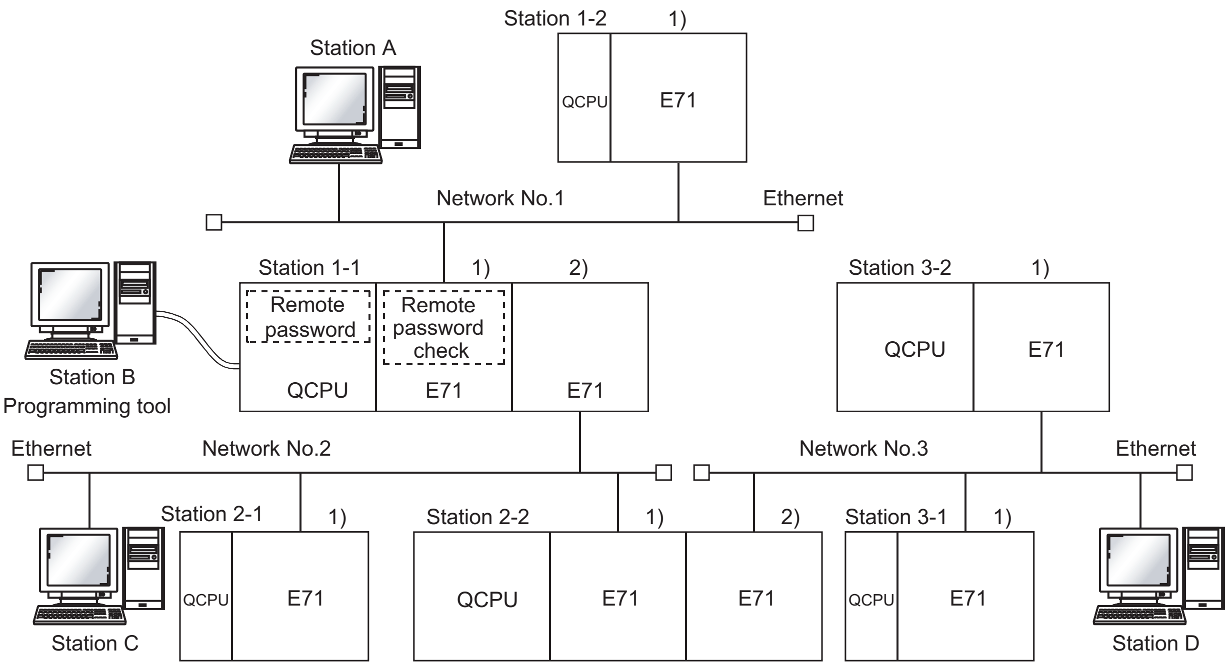
(2) Selecting a connection for which the remote password check is performed

A connection for which an entered remote password is checked can be selected using a parameter. (Page 210, Section 14.4.6)

(3) Stations that can be accessed when the remote password check is performed

When the CPU module is protected with a remote password, the stations accessible by the connected device and the QCPU stations that can perform the remote password unlock/lock processes are limited to those in the same network. The following shows an example of accessible stations.

Ex. When a remote password has been set in the QCPU on the station 1-1, and the remote password check has been set in 1) on the station 1-1



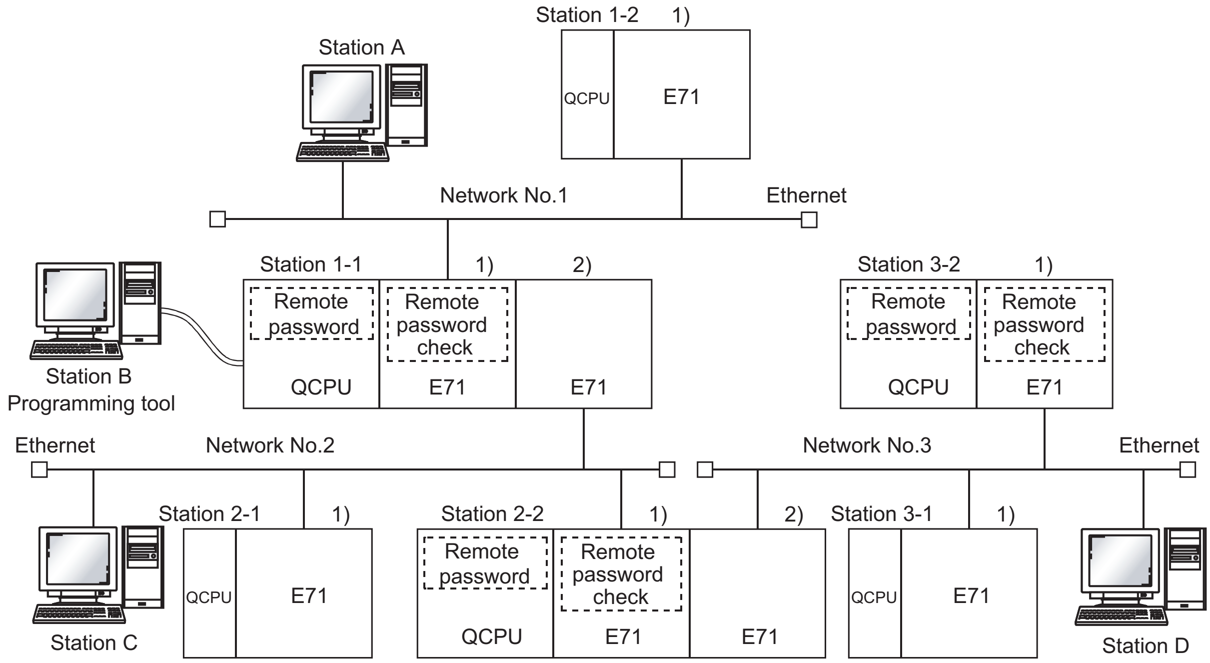
*1 The station A is the only connected device that can perform an unlock/lock process for 1) of the station 1-1.

●: Stations that can be accessed from the connected device after the remote password unlock process,
○: Stations that can be accessed from the connected device without performing the remote password unlock process

Connected device*2 (request source)	Target programmable controller station (request destination)					
	Station 1-1 QCPU	Station 1-2 QCPU	Station 2-1 QCPU	Station 2-2 QCPU	Station 3-1 QCPU	Station 3-2 QCPU
Station A	●	○	●	●	●	●
Station B	○	○	○	○	○	○
Station C	○	○	○	○	○	○
Station D	○	○	○	○	○	○

*2 The station A can access the stations marked with ● after the remote password unlock process is completed for 1) of the station 1-1. It can access the stations marked with ○ when the communication lines are open. Stations B, C, and D can access the stations marked with ○ when the communication lines are open.

Ex. When a remote password and the remote password check have been set in multiple QCPU stations



*1 The following are the connected devices that can perform unlock/lock processes.
- 1) of the station 1-1: By the station A only
- 1) of the station 2-2: By the station C only
- 1) of the station 3-2: By the station D only

●: Stations that can be accessed from the connected device after the remote password unlock process,
○: Stations that can be accessed from the connected device without performing the remote password unlock process,
×: Stations that cannot be accessed from the connected device

Connected device*2 (request source)	Target programmable controller station (request destination)					
	Station 1-1 QCPU	Station 1-2 QCPU	Station 2-1 QCPU	Station 2-2 QCPU	Station 3-1 QCPU	Station 3-2 QCPU
Station A	●	○	●	×	×	×
Station B	○	○	○	×	×	×
Station C	○	○	○	●	●	×
Station D	○	○	○	○	○	●

*2 The station A can access the stations marked with ● after the remote password unlock process is completed for 1) of the station 1-1, and can access the stations marked with ○ when the communication lines are open. The station B can access the stations marked with ○ when the communication lines are open. The station C can access the stations marked with ● after the remote password unlock process is completed for 1) of the station 2-2, and can access the stations marked with ○ when the communication lines are open. The station D can access the stations marked with ● after the remote password unlock process is completed for 1) of the station 3-2, and can access the stations marked with ○ when the communication lines are open.

Point

To disable access from the connected devices to other stations through CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 relay communication functions, select "MELSOFT Application Transmission port (UDP/IP), Dedicated Instruction, CC-Link IE, NET10(H) Relay Transmission Port" in the remote password setting of the relay station or a station to be accessed.

14.4.4 Comparison of functions according to the remote password check status (enabled/disabled)

The following table lists the behavior of the functions according to the remote password check status (enabled/disabled).

Function		Remote password check setting: Disabled	Remote password check setting: Enabled
Connection with MELSOFT products and a GOT		After an initial process is completed, communications are enabled by establishing a connection in the programming tool.	After the remote password is entered, communications are enabled. The remote password lock process is automatically performed when the project is closed.
MC protocol communications	User open port	After an open process is completed, communications are enabled.	After an open process is completed, communications are enabled from the time that the unlock command is received until the lock command is received.
MC protocol communications	Auto-open UDP port	After an initial process is completed, communications are enabled.	After an initial process is completed, communications are enabled from the time that the unlock command is received until the lock command is received.
Communications using SLMP	User open port	After an open process is completed, communications are enabled.	Use MC protocol to communicate with the CPU module where a remote password is set because SLMP does not have unlock and lock commands.
Communications using SLMP	Auto-open UDP port	After an initial process is completed, communications are enabled.	Use MC protocol to communicate with the CPU module where a remote password is set because SLMP does not have unlock and lock commands.
Communications using the predefined protocol		After an open process is completed, communications are enabled.	After an open process is completed, communications are enabled.[*1]
Communications using the predefined protocol support function		After an initial process is completed, communications are enabled by establishing a connection in GX Works2.	After the remote password is entered, communications are enabled. The remote password lock process is automatically performed when the protocol setting data are closed.
Communications using a fixed buffer	Procedure exists	After an open process is completed, communications are enabled.	After an open process is completed, communications are enabled from the time that the unlock command is received until the lock command is received.
Communications using a fixed buffer	No procedure	After an open process is completed, communications are enabled.	After an open process is completed, communications are enabled.[*1]
Communications using a random access buffer		After an open process is completed, communications are enabled.	After an open process is completed, communications are enabled from the time that the unlock command is received until the lock command is received.
E-mail function		After an initial process is completed, data sending and receiving are enabled.[*2]	After an initial process is completed, data sending and receiving are enabled.[*2]
Web function		After an initial process is completed, communications are enabled.	After the remote password is entered, communications are enabled. The remote password lock process is automatically performed when the Web browser is closed.
CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 relay communications		After an initial process is completed, communications are enabled.	After an initial process is completed, communications are enabled.[*3]
Communications using data link instructions		After an initial process is completed, communications are enabled.	After an initial process is completed, communications are enabled.[*3]

Function	Remote password check setting	
	Disabled	Enabled
File transfer (FTP server) function	Within Ethernet, communications are enabled with the connected device that has completed an open process.	After an open process is completed, communications are enabled from the time that the unlock command is received until the lock command is received.

*1 A dedicated connection is used for "No Procedure" communications using a fixed buffer or communications using the predefined protocol. Do not set the remote password for the connection with the connected device.

*2 Remote password check is not performed against the e-mail function.

*3 If a remote password has been set to the CPU module on the relay station or station to be accessed where the E71 is mounted, access to another station may not be performed.

14.4.5 Precautions

The precautions for using the remote password function of the E71 are provided below.

(1) Timing of activating a remote password

To enable the remote password setting, power off or reset the CPU module. After a remote password has been set, restart the CPU module. (CPU number 1 in a multiple CPU system).

(2) Target connections

Set a remote password check only for the connections used for data communications with the connected devices that can perform unlock and lock processes.

Ex. In communications using a fixed buffer, do not set the remote password check for the connections through which the connected device receives data sent from the CPU module.

(3) Connections where "No Procedure" communications using a fixed buffer are performed

Do not set a remote password check for connections where "No Procedure" communications using a fixed buffer are performed. The remote password check is not performed for such connections.

(4) Access to the programmable controller on another station

When the connected device accesses the programmable controller on another station through the E71, it may fail to access the remote-password-protected CPU modules on the relay station or the station to be accessed.

(5) For UDP/IP communications

(a) Connected device for data communications

Do not perform data communications with unspecified connected device. Set the destination device.

(b) Alive check function

Use the alive check function of the E71.*1 Also, when terminating data communications, perform the remote password lock process. If the process is not performed, data communications from other devices are enabled until a timeout occurs by the alive check function of the E71. For this reason, when setting a connection with the connected device using a programming tool, set the parameters as follows.

- When configuring the initial setting, set the start interval timer value and the interval timer value for the alive check function as small as possible.
- When configuring the open setting, select "Confirm" under "Existence Confirmation".

*1 The alive check is automatically performed when the connection for data communications through the auto-open UDP port is set as the target for the remote password check.

(6) Communications from the programming tool in the Ethernet connection

For communications from a programming tool in the Ethernet connection, using TCP/IP communications is recommended.

(7) When the unlock process or lock process fails

If the remote password unlock/lock process fails, check the remote password of the CPU module then perform the unlock/lock process again.

(a) E71 operation in case of a process failure

If the number of process failures exceeds the notification accumulated count*1 set in the buffer memory area, the E71 performs the following operations.

- The E71 turns on the COM.ERR. LED.
- The E71 stores $C200_H$ in the error code and end code storage area in the buffer memory area, Error log area (address: $E3_H$ to 174_H).

*1 This is the number of times stored in Remote password mismatch notification accumulated count designation (address: 5070_H and 5071_H) by the CPU module (using an instruction such as the TO instruction) when the E71 starts up.

In the above case, identify the connection where the unlock/lock process has failed using the following buffer memory areas.

- Accumulated count of unlock process abnormal end (address of Connection No.1: 5073_H)
- Accumulated count of lock process abnormal end (address of Connection No.1: 5075_H)

(b) Corrective action against a process failure

Take the following actions if required.

- Close the connection with the connected device.
- Write "0" in the buffer memory areas, Accumulated count of unlock process abnormal end (address: 20595 (5073_H) and Accumulated count of lock process abnormal end (address: 20597 (5075_H)). If this action is not taken, the process (a) above will be performed each time a process failure occurs exceeding the notification accumulated count.
- If the number of unlock/lock process failures for the connection to the connected device is greater than the above accumulated notification count, the possible cause is unauthorized access from the connected device. Disable the connection using the buffer memory area, System port use prohibited designation area (address: 5008_H) (After this, the unlock process cannot be performed for the connection until "Use allowed" is set.)
- Inform the system manager that the number of unlock/lock process failures is greater than the accumulated notification count, and take corrective actions.

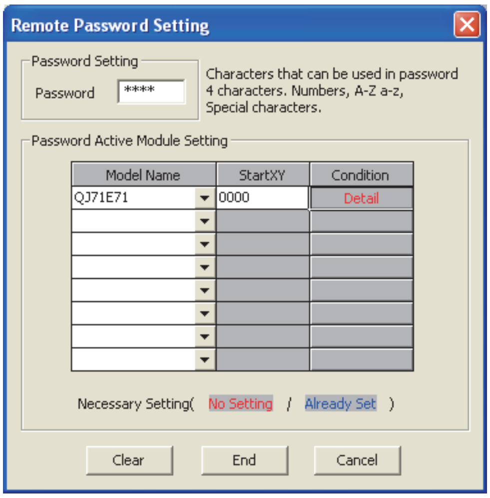
Point

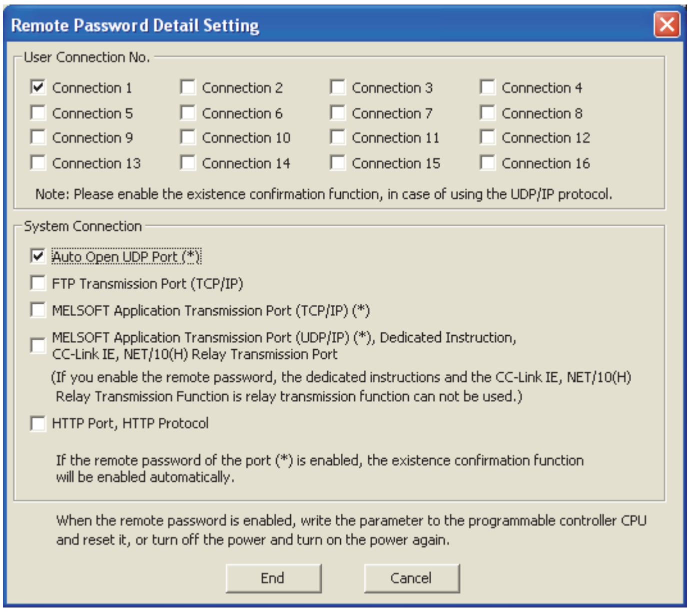
- For the method to turn off the COM.ERR. LED, refer to the method using the Ethernet diagnostics. (Page 340, Section 16.8)
- The accumulated counts stored in the following buffer memory areas can be cleared.
  - Accumulated count of unlock process abnormal end (address of Connection No.1: 5072_H)
  - Accumulated count of lock process abnormal end (address of Connection No.1: 5074_H)

14.4.6 Parameter settings

The remote password setting of the E71 is described.

Project window ➪ [Parameter] ➪ [Remote Password]





Item			Description	Setting range
Remote Password Setting	Password Setting		Enter a password set for the CPU module.	-
	Password Active Module Setting	Model Name	Select the module model that checks an entered remote password against the remote password set for the CPU module.	QJ71E71
		Start XY	Set the start address of the module that checks an entered remote password.	0000_H to $0FE0_H$
		Condition	Opens the "Remote Password Detail Setting" window.	-
Remote Password Detail Setting	User Connection No.	Connection 1 to Connection 16	Specify a connection where an entered remote password is checked.	-
	System Connection	Auto Open UDP Port		
		FTP Transmission Port (TCP/IP)		
		MELSOFT Application Transmission Port (TCP/IP)		
		MELSOFT Application Transmission Port (UDP/IP), Dedicated Instruction, CC-Link IE, NET/10(H) Relay Transmission Port		
		HTTP Port, HTTP Protocol		

Point

When determining a remote password, take care of the following.

- Avoid a simple character string (e.g. Character string consisting of alphanumeric characters only).
- Combine alphanumeric characters and special characters (e.g. "?", "!", "&", and "%").
- Avoid using a character string that represents the user's name or the date of birth.

14.5 Hub Connection Status Monitor Function

The current connection status and transmission speed of an E71 and a hub and the number of times that the E71 detected disconnection can be checked using the following buffer memory areas. For details on buffer memory areas, refer to the list of buffer memory addresses. (Page 38, Section 3.5.2)
This function is available only for the QJ71E71-100.

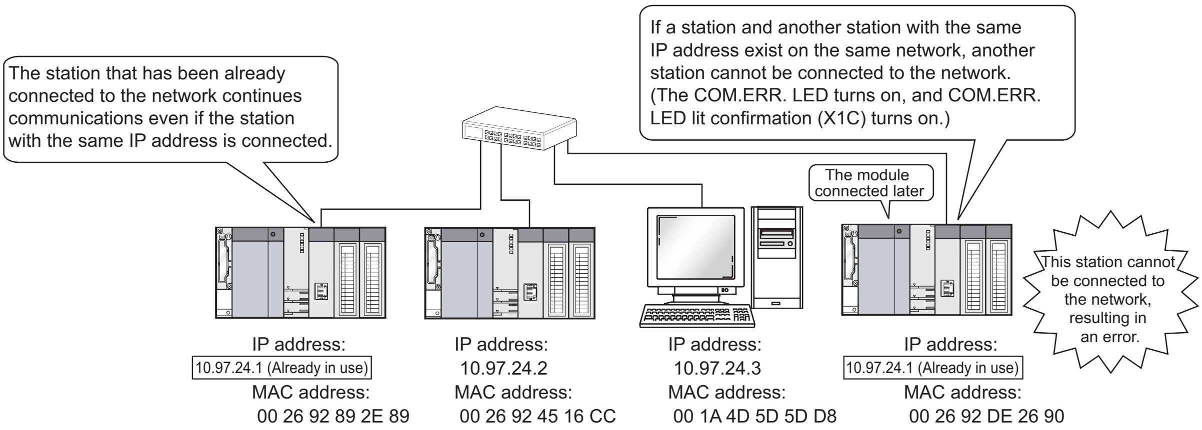
Buffer memory area	Description
Hub connection status area (address: $C9_H$)	Stores the current connection status and transmission speed of an E71 and a hub.
Disconnection detection count (address: 5203_H)	Stores the number of times that the E71 detected disconnection after an initial process. Disconnection is detected in any of the following cases. • Disconnection between the E71 and a hub • Cable removal from the hub side connector • Hub power-off • Cable removal from the E71 side connector

Point

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

14.6 IP Address in Use Detection Function

If different stations in the same network use the same IP address, the address in use can be detected. This prevents a network from stopping due to incorrect IP address.
This function is available only in the QJ71E71-100 with the serial number (first five digits) of "12062" or later. When the connected device with the same IP address does not support the IP address in use detection function, the error is not detected.



(1) Checking the IP address already in use

The following two methods are available:

- Checking buffer memory areas
- Checking using the Ethernet diagnostics

(a) Checking buffer memory areas

Both the station already connected to the network and the station with the same IP address (station connected to the network later), and the MAC addresses of the stations with the same IP address can be checked. The destination IP address can be checked only in the station with the same IP address (station connected to the network later).

Buffer memory area		Buffer memory address	Description
Destination IP address (stored in the station with the same IP address connected to the network later.)		234 (EA_H)	The third and fourth octets of the IP address
		235 (EB_H)	The first and second octets of the IP address
IP address status storage area	The MAC address of the station that has been already connected to the network (stored in the station with the same IP address connected to the network later).	21121 (5281_H)	The fifth and sixth octets of the MAC address
		21122 (5282_H)	The third and fourth octets of the MAC address
		21123 (5283_H)	The first and second octets of the MAC address
	The MAC address of the station with the same IP address connected to the network later (stored in the station that has been already connected to the network).	21124 (5284_H)	The fifth and sixth octets of the MAC address
		21125 (5285_H)	The third and fourth octets of the MAC address
		21126 (5286_H)	The first and second octets of the MAC address

Remark

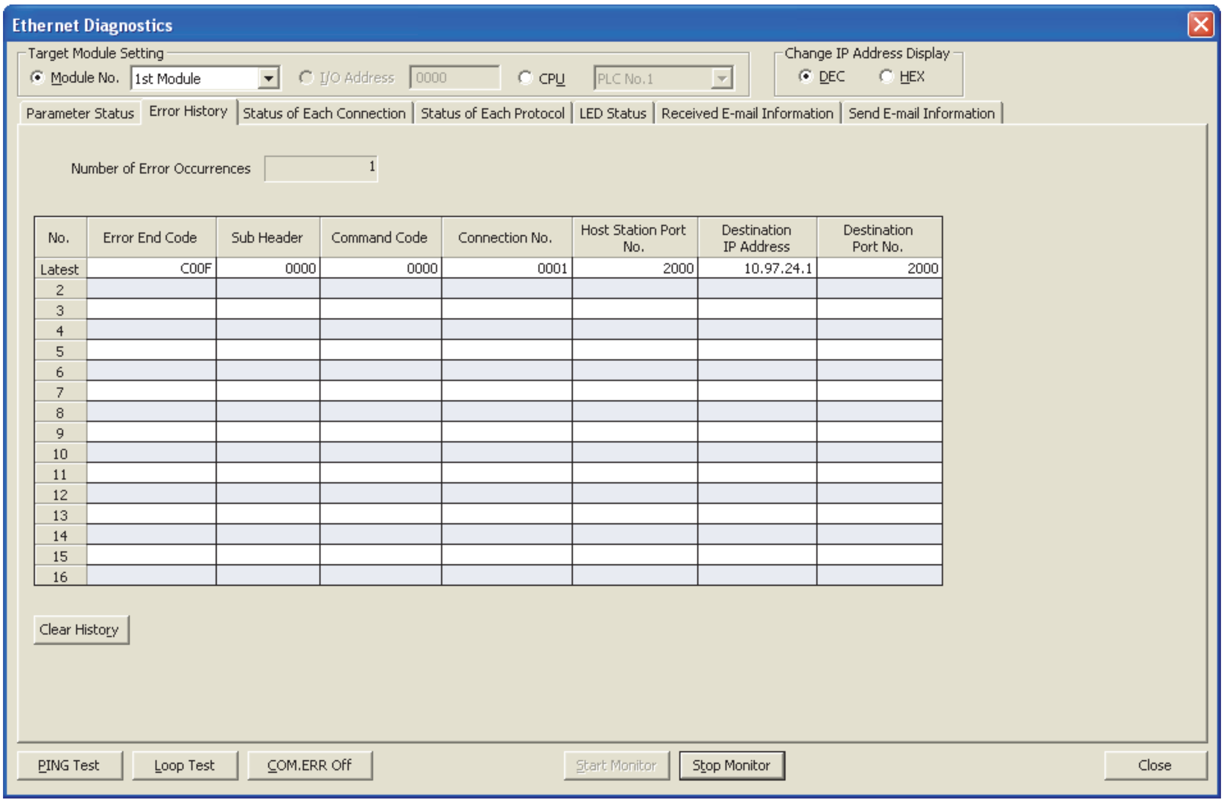
When the MAC address of the station that has been already connected to the network is 00.26.92.89.2E.89, the IP address already in use is 10.97.24.01, and the MAC address of the station connected to the network later is 00.26.92.DE.26.90, the value stored in each buffer memory area is as follows.

- 234 (EA_H): 1801_H (the third and fourth octets of the IP address)
- 235 (EB_H): $0A61_H$ (the first and second octets of the IP address)
- 21121 (5281_H): $2E89_H$ (the fifth and sixth octets of the MAC address)
- 21122 (5282_H): 9289_H (the third and fourth octets of the MAC address)
- 21123 (5283_H): 0026_H (the first and second octets of the MAC address)
- 21124 (5284_H): 2690_H (the fifth and sixth octets of the MAC address)
- 21125 (5285_H): $92DE_H$ (the third and fourth octets of the MAC address)
- 21126 (5286_H): 0026_H (the first and second octets of the MAC address)

(b) Checking using the Ethernet diagnostics

An error code ($C00F_H$) is displayed in the "Ethernet Diagnostics" window, and the IP address used in multiple stations is displayed in the "Destination IP Address" column. The destination IP address can be checked only in the station with the same IP address (station connected to the network later).

[Diagnostics] ⇨ [Ethernet Diagnostics...]



14.7 Redundant System Function

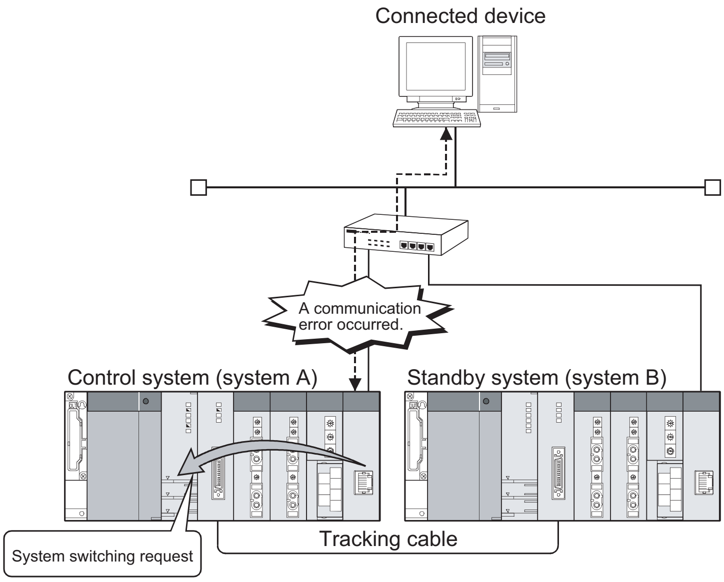
This section describes functions used when the E71 is mounted on the main base unit in a redundant system.

Remark

When the E71 is mounted on an extension base unit, skip this section.

14.7.1 System switching request to the control system CPU module

This function issues a system switching request to the control system CPU module if the E71 mounted with the control system CPU module in a redundant system detects a communication error or disconnection.



(1) Conditions under which a system switching request is issued

The E71 issues a system switching request when the following conditions are satisfied.

Condition		Description
Communication error detection	Alive check	After a connection is opened, the existence of the connected device cannot be confirmed.
	ULP timeout	An ACK response is not returned from the connected device within the TCP ULP timer time.
Disconnection detection		The cable connected to the E71 has been disconnected (the QJ71E71-100 only).

Point

In the following cases, system switching is not performed even if the E71 issues a system switching request.

- When the standby system has already been in error status (due to such as power-off, reset operation, and stop error)
- The network module redundant group settings have been configured for the E71 and either one is operating normally.

(2) Issuing a system switching request upon communication error

The E71 mounted with the control system CPU module monitors communications with the connected device on each connection. If the E71 detects a communication error, it issues a system switching request to the control system CPU module. The following table lists a communication error that triggers a system switching request.

Communication error	Description
Alive check	After a connection is opened, the existence of the connected device cannot be confirmed.
ULP timeout	An ACK response is not returned from the connected device within the TCP ULP timer time.

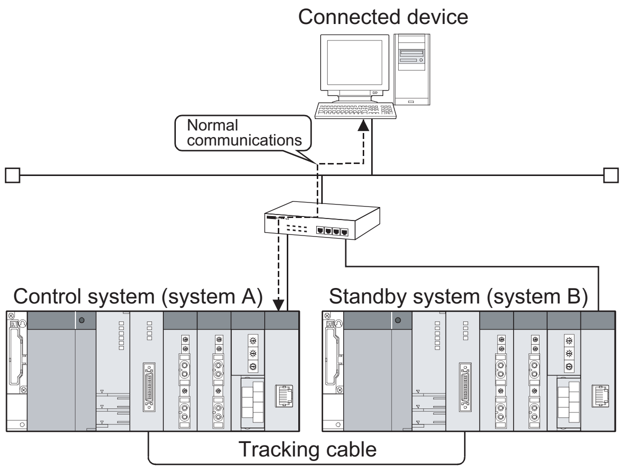
(a) Target connection

The E71 detects communication errors in the connections selected in the redundant settings. The following table lists connections that can be monitored.

Target connection	
User Connection No.	Connection 1 to Connection 16
System Connection	Auto Open UDP Port
	FTP Transmission Port (TCP/IP)
	MELSOFT Application Transmission Port (TCP/IP)
	MELSOFT Application Transmission Port (UDP/IP), Dedicated Instruction, CC-Link IE, NET/10(H) Relay Transmission Port
	HTTP Port, HTTP Protocol

(b) System switching request operation

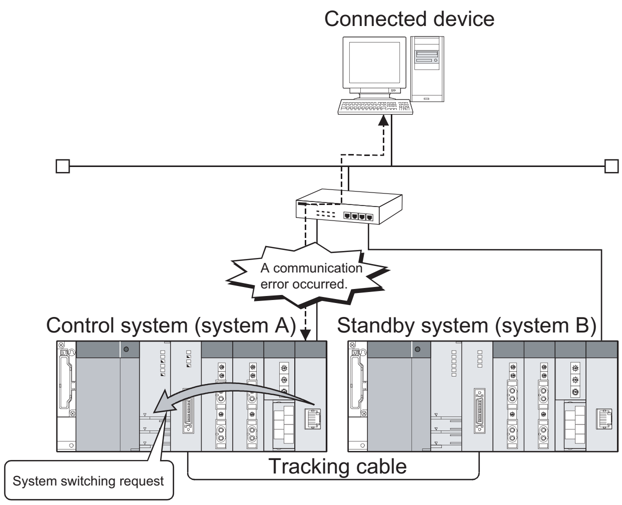
The E71 monitors communications with the connected device for an error using the alive check function and TCP ULP timer.



1. During normal communications

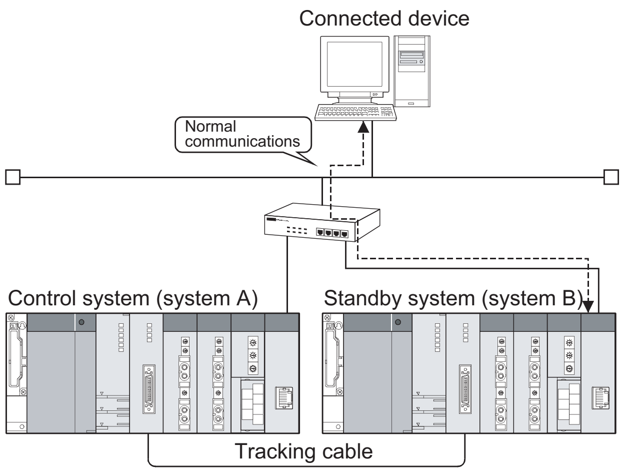
The system A is operating as a control system, and the system B as a standby system.

The connected device is communicating with the E71 mounted with the control system CPU module. (By connecting the connected device to the E71 mounted with the standby system CPU module in TCP/IP, an error in the standby system CPU module can be detected.)



2. At error detection[*1]

If a communication error has occurred between the connected device and the E71, and the E71 mounted with the control system CPU module detects an error, a system switching request is issued to the control system CPU module.[*2]



3. After system switching

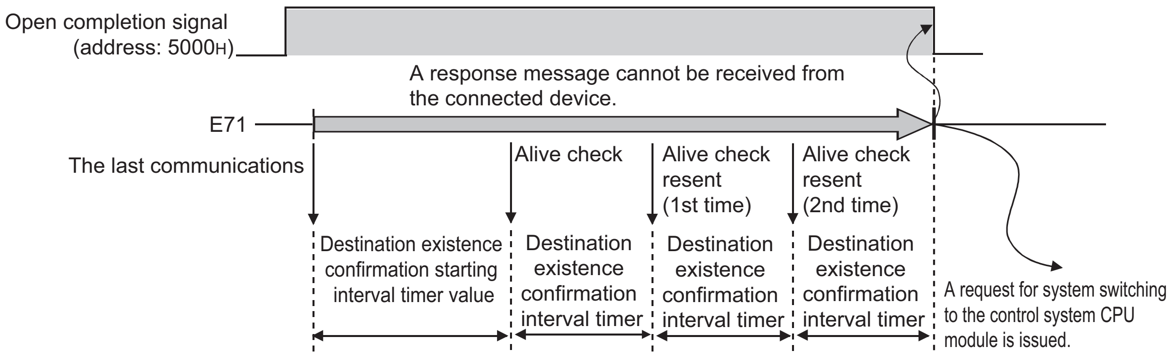
The system A operates as a standby system, and the system B as a control system.

The connected device changes the destination and communicates with the E71 mounted with the control system CPU module (system B).

*1 The following shows the timing of issuing the system switching request.

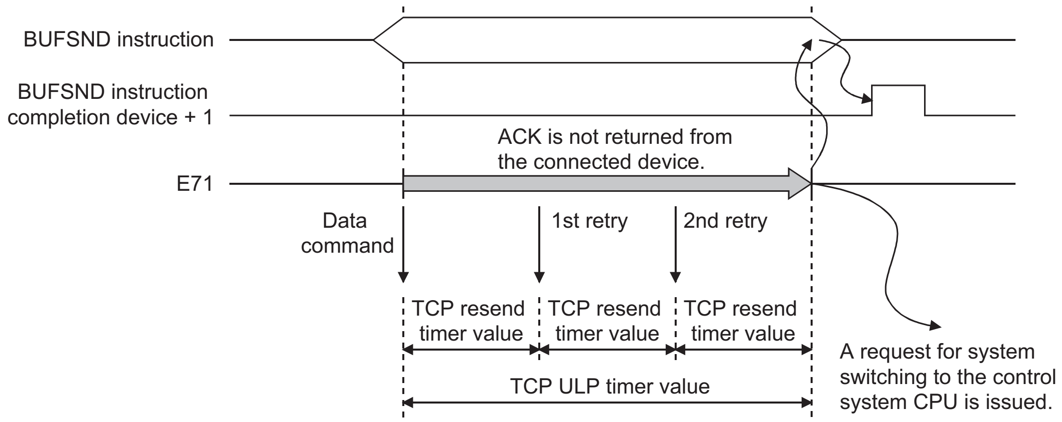
- System switching timing when an alive check is used

  When the E71 has not communicated with the connected device where a connection is open for a certain period of time, it performs an alive check. If the E71 cannot receive a response message from the connected device, it closes the corresponding connection and issues a system switching request to the control system CPU module. (The following figure shows an example where an alive check message is resent twice.)



- System switching timing when a ULP timeout is used

  If ACK is not returned from the connected device within the TCP ULP timer time when the connection is open in TCP/IP or data is sent, a sending error occurs and the E71 issues a system switching request to the control system CPU module. (The following figure shows an example where the number of retries is set twice.)



*2 Set whether to issue the system switching request in the redundant settings of the programming tool. (Page 221, Section 14.7.3)

14

14.7 Redundant System Function
14.7.1 System switching request to the control system CPU module

(3) Issuing a system switching request upon disconnection detection

The E71 mounted on the control system CPU module monitors the connection status of the cable connected to the E71, and on detection of disconnection, it issues a system switching request to the control system CPU module. Disconnection is detected in the following cases.

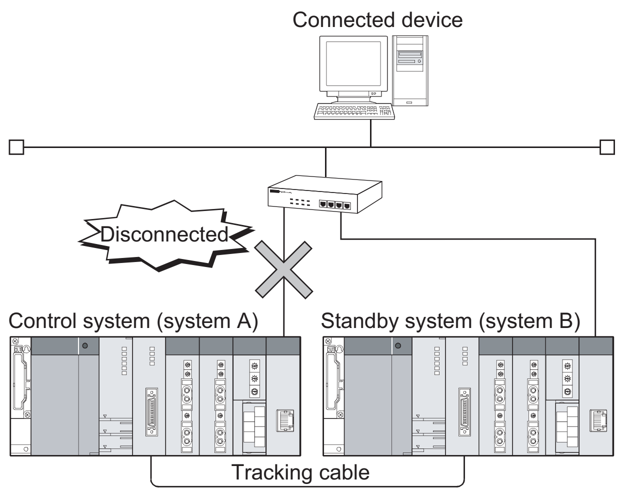
- Disconnection between the E71 and a hub
- Cable removal from the hub side connector
- Hub power-off
- Cable removal from the E71 side connector

Remark

This function is available for the QJ71E71-100 only.

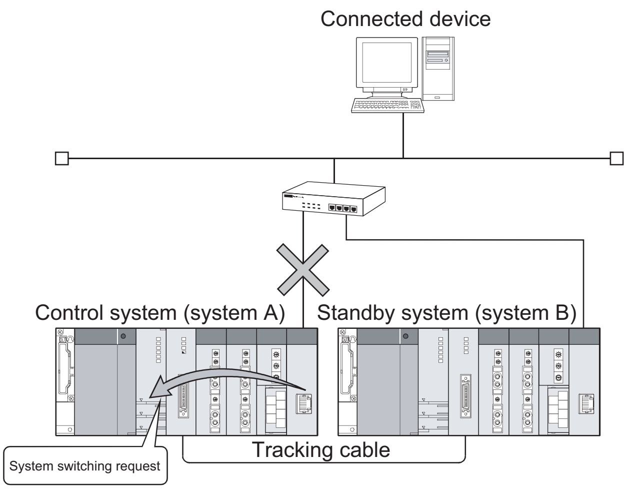
(a) System switching request operation

The E71 always monitors the connected cable for disconnection.



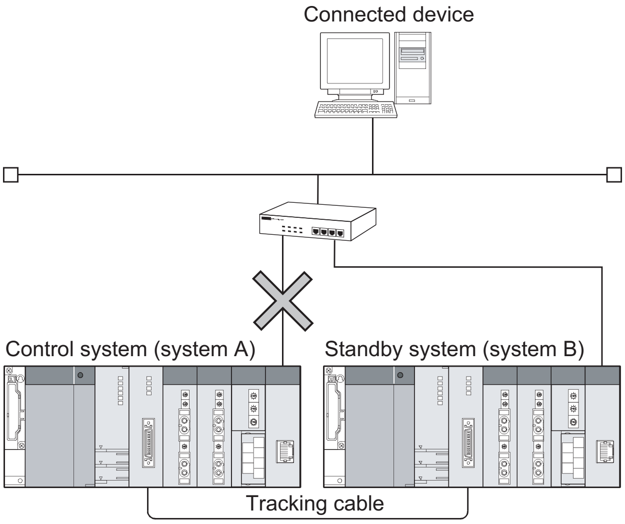
1. Disconnection monitoring

The E71 always monitors the connected cable for disconnection.*1 The monitoring result is stored to the buffer memory area, Hub connection status area (address: $C9_H$).



2. Upon disconnection detection*2

If the E71 mounted with the control system CPU module detects disconnection, it performs a disconnection status time check. If the disconnected status continues for the period of the disconnection detection monitoring time, the E71 issues a system switching request to the control system CPU module.



3. **After system switching**

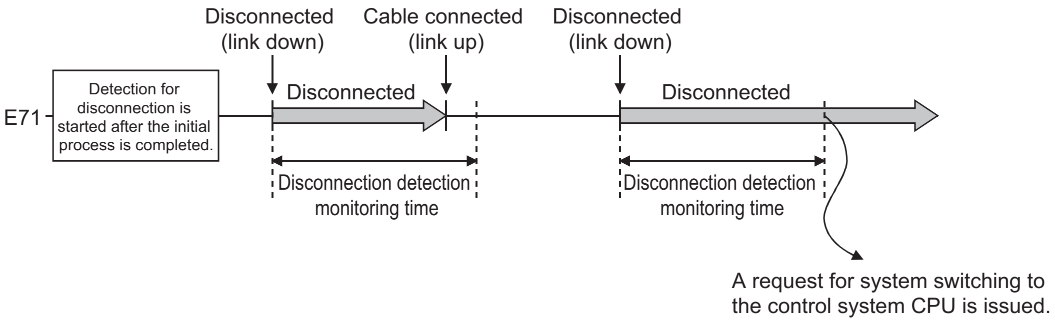
The system A operates as a standby system, and the system B as a control system.

*1 When the cable is not connected from the start, the E71 does not determine it as disconnection. (Disconnection is detected only when normal status turns to abnormal.)

*2 The following shows the timing of issuing the system switching request.

- System switching timing at disconnection detection

  The E71 starts disconnection detection after initial process completion. On detection of disconnection, the E71 performs a disconnection status time check, and when disconnected status continues for the period of the disconnection detection monitoring time, the E71 issues a system switching request to the control system CPU module. When the disconnected status returns to normal within the disconnection detection monitoring time, the E71 does not issue a system switching request.



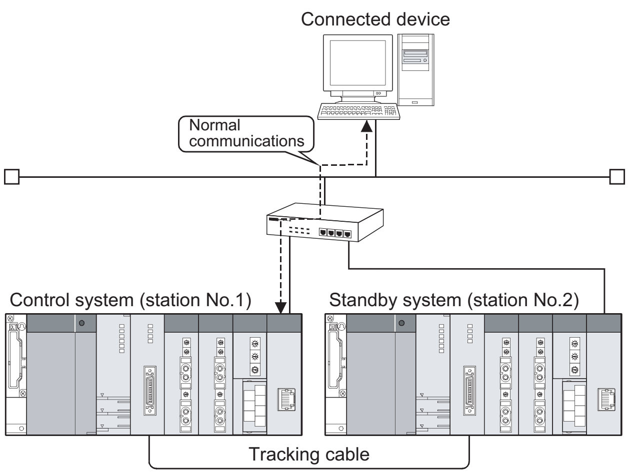
14.7.2 Communication path bypass function

When any of the following redundant system supported applications is used, the path where a communication error has occurred is automatically bypassed to continue communications if an error occurs in communications with the E71. The communication path needs not to be changed manually.

- Application that operates on the OPS (except the MELSOFT products such as GX Developer)
- GX Works2
- GX Developer
- PX Developer monitor tool

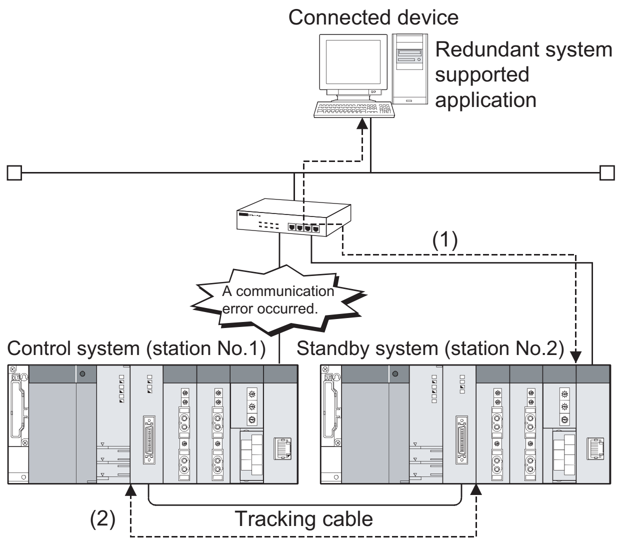
(1) Operation example of a redundant system supported application at a communication error

The following provides an example of redundant system supported application operation if an error occurs during communications with the control system CPU module.



1. During normal communications

The station number 1 is operating as a control system, and the station number 2 as a standby system. The redundant system supported application is communicating with the control system CPU module.



(1) The connected device accesses the control system CPU module around the path.
(2) Communications with the control system CPU module is continued via a tracking cable.

2. Upon a communication error

The station number 1 is operating as a control system, and the station number 2 as a standby system. (Without system switching)

[Operation of the redundant system supported application]
Because a communication error has occurred between the redundant system supported application and the E71 mounted with the control system CPU module, the redundant system supported application automatically changes the communication path and communicates with the control system CPU module through the standby system CPU module.

[Operation of the standby system CPU module]
Because the received data is addressed to the control system CPU module, data is relayed to the control system CPU module through the tracking cable.

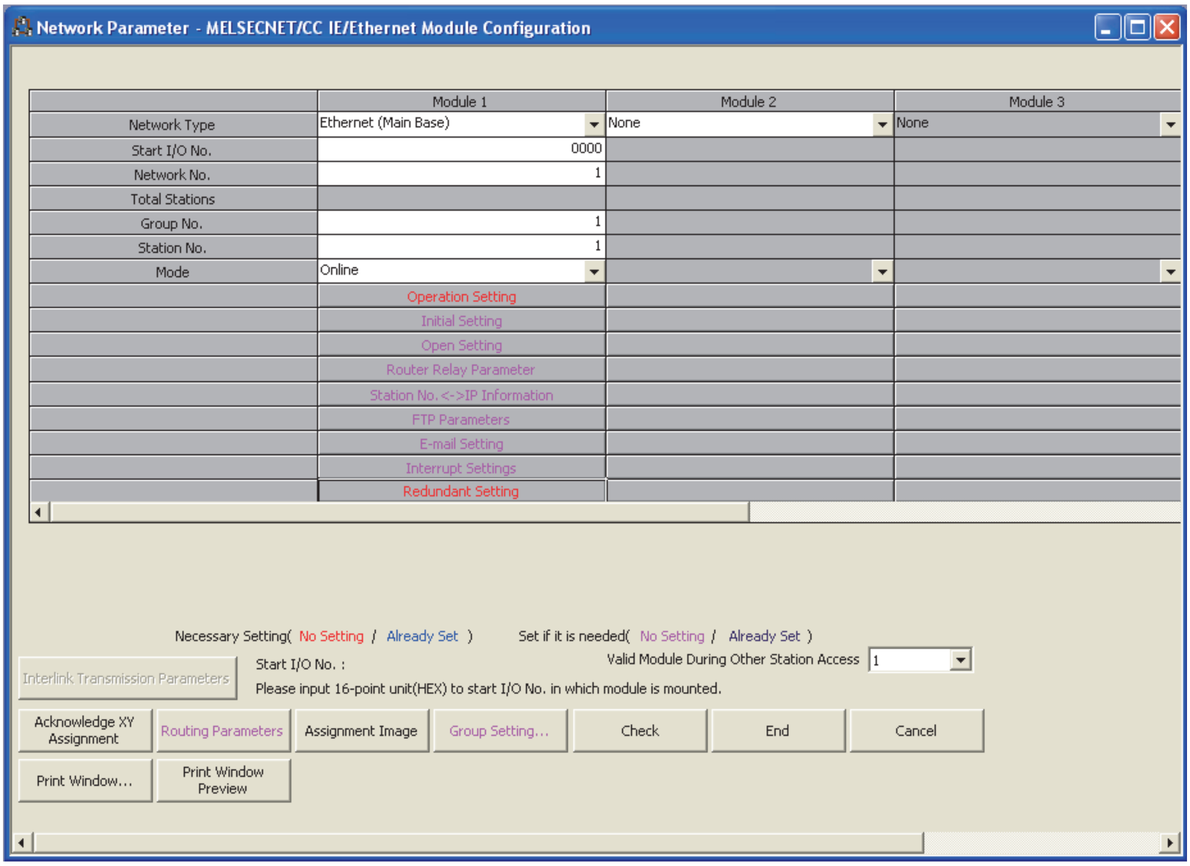
14.7.3 Parameter settings

This section describes the parameter settings for using the E71 in a redundant system. The following settings are required:

- Basic setting
- Ethernet operation setting
- Open setting (if necessary) (☞ Chapter for each function)
- Redundant settings

(1) Basic setting

Only the settings for a redundant system that differ from those for a single CPU system are listed.



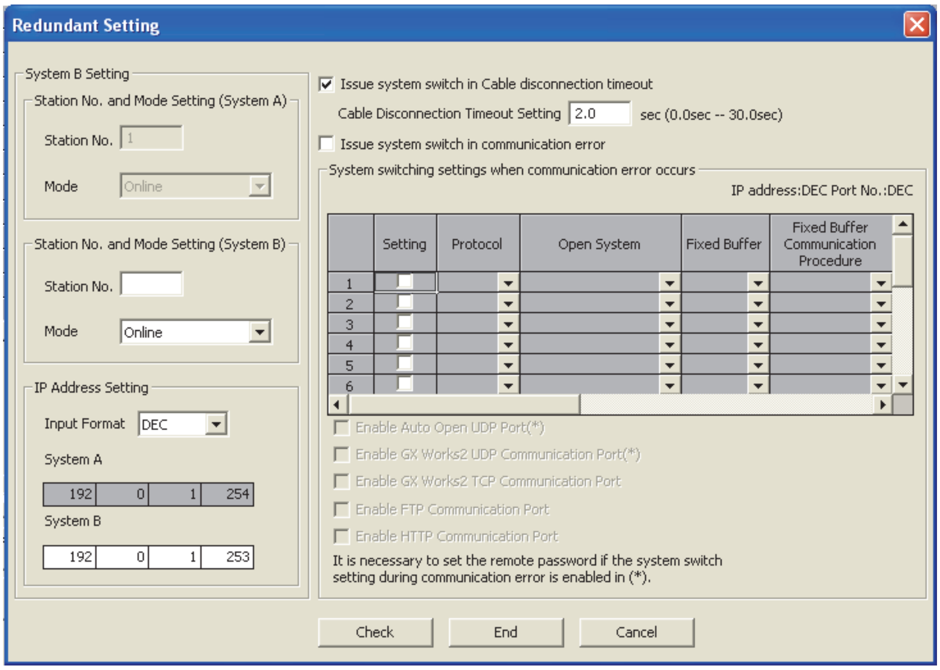
Item	Description	Setting range
Network Type	Select "Ethernet (Main Base)".	-
Station No.	Set the station number of the E71 in the system A. Set the station number of the E71 in the system B in the Redundant Setting.	1 to 64
Mode	Select the operation mode of the E71 in the system A. Set the operation mode of the E71 in the system B in the Redundant Setting.	• Online • Offline • Self-Loopback Test • H/W Test

(2) Ethernet operation setting

The IP address configured in the Ethernet operation setting becomes the IP address of the E71 in the system A. Set the IP address of the E71 in the system B in the redundant settings.

(3) Redundant settings

Configure the settings of the system B and issue of a system switching request.



Item			Description	Setting range
System B Setting	Station Number and Mode Setting (System A)	Station No.	Displays the settings configured in the basic setting.	-
		Mode		
	Station Number and Mode Setting (System B)	Station No.	Set the station number and mode of the system B. Set a different station number from that of the system A. Set the same mode as the system A.	Same as system A
		Mode		
	IP Address Setting	Input Format	Select an input format.	Same as system A
		System A	Displays the settings configured in the Ethernet operation setting.	-
		System B	Set the IP address of the system B. Set a different IP address from that of the system A.	Same as system A
Issue system switch in Cable disconnection timeout			Select whether a system switching request is issued upon detection of disconnection.	Selected/not selected
	Cable Disconnection Timeout Setting		Set the time from when disconnection is detected until the system switching request is issued to the control system CPU module.	0.0s to 30.0s
Issue system switch in communication error			Select whether a system switching request is issued upon a communication error.	Selected/not selected
	System switching settings when communication error occurs		Set the target connection where a communication error will cause a system switching request to be issued.	Target connection

*1 The settings of the auto-open UDP port and MELSOFT application transmission port (UDP) are enabled when the following conditions are satisfied. When the following conditions are not satisfied, the system switching request is not issued even if a communication error occurs in the target connection.
- The remote password setting is enabled.
- The remote password is unlocked.

*2 Do not set the connection with any of the following settings as the target connection. If set, the system may be consecutively switched in the redundant system if a communication error occurs due to cable disconnection or power-off of the connected device.
- "Initial timing" in the Ethernet operation setting: "Always wait for OPEN (Communication possible at STOP time)"
- "Protocol" in the open setting: "UDP"
- "Existence confirmation" in the open setting: "Confirm"

*3 It is recommended that the connection set to "MELSOFT connection" under "Open system" of the open setting is not set as the target connection. Doing so may set all the MELSOFT products connected to the network as the target connections, therefore; the connected device (target MELSOFT product) cannot be specified.

Point

- When using a redundant system in the backup mode, set the mode of the system B same as that of the system A. If the mode differs between them, an error will occur in the Redundant CPU.
- Configure "Cable disconnection timeout setting" from several seconds to several ten seconds. If the value is shorter than that, a system switching request may occur due to such as noise.

14.7.4 Data communications in a redundant system

This section describes data communications using the E71 mounted on the main base unit in a redundant system. Processes other than those described in this section are the same as those in a single CPU system.

(1) Initial process

(a) Reinitialization process

When performing a reinitialization process, do not change the settings, such as the own station IP address and Ethernet operation setting. Doing so may cause a communication error.

- When using the UINI instruction
  Execute the instruction after specifying "0_H" for Specification of changed item ((S1)+2) of the control data.
- When directly writing to the buffer memory
  Write "1" to the bit 15 of Communication condition setting (Ethernet Operation Setting) area (address: $1F_H$) without changing the value stored in the buffer memory area.

(b) Initial process using I/O signals

Because the output signal (Y) turns off in the standby system CPU module, an initial process cannot be performed using I/O signals. To perform an initial process, set the network parameters using a programming tool.

(2) Open/close processes

(a) Communications in TCP/IP

Place the E71 in open wait status (Passive open) and perform an open/close process from the connected device. If an Active open process is performed from the E71, the close process is performed from the E71 after data communications, but if the system is switched before execution of the close process, the process cannot be performed.

(b) When using a user connection to communicate with the standby system (for MC protocol communications or communications using a random access buffer)

- Ethernet operation setting
  Set "Always wait for OPEN (Communication possible at STOP time)" under "Initial timing".
- Open setting
  In TCP/IP communications, set "Unpassive or Fullpassive" under "Open system".

Point

Setting connections for communications with the system A and communications with the system B is recommended. This facilitates communications with the other system if a communication error has occurred in the own system or when the system is switched. A maximum of 16 user connections can be registered in the E71.

(c) Open/close processes using I/O signals

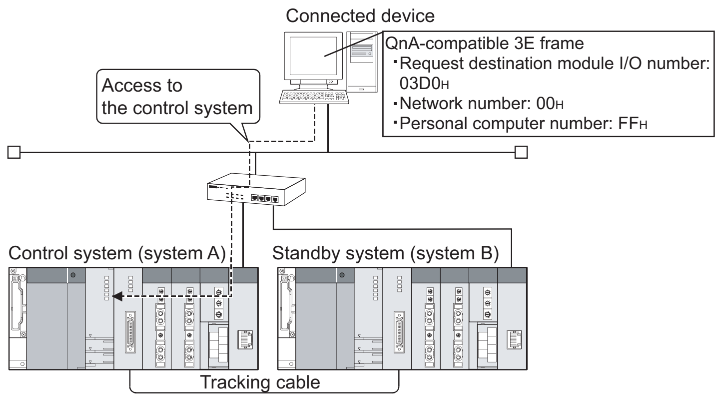
Because the output signal (Y) turns off in the standby system CPU module, open/close processes cannot be performed using I/O signals. To perform the processes, set "Always wait for OPEN (Communication possible at STOP time)" under "Initial timing" in the Ethernet operation setting or use dedicated instructions (OPEN/CLOSE instructions).

(3) Using MC protocol communications

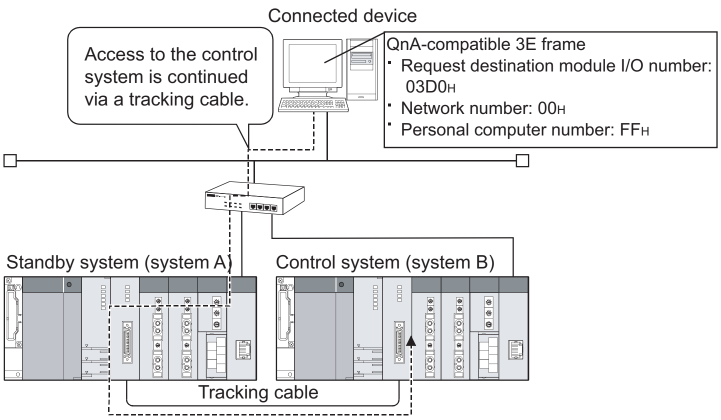
The QnA-compatible 3E frame or 4E frame can be used to access the control system/standby system or system A/system B.

(a) Operation performed for access to the control system/standby system or the system A/system B

Ex. When the system is switched (example of access to the control system CPU module)

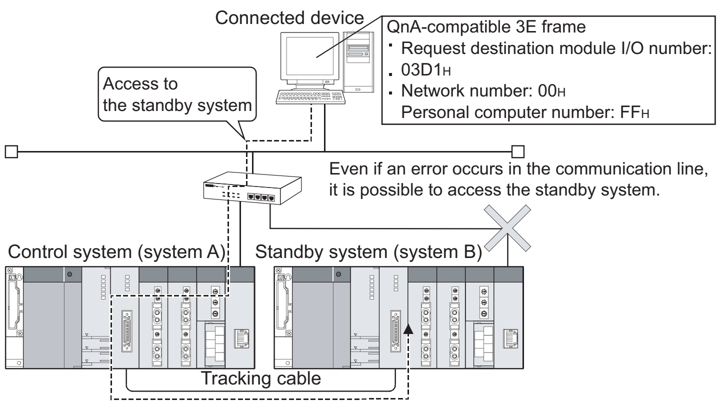


1. **Connect the connected device to the E71 mounted with the control system CPU module to access the control system CPU module.**



2. **When the system is switched, the connected device automatically continues access to the control system through the tracking cable. However, if the communication line with the connection destination is faulty or the standby system is powered off, for example, the destination must be changed on the connected device.**

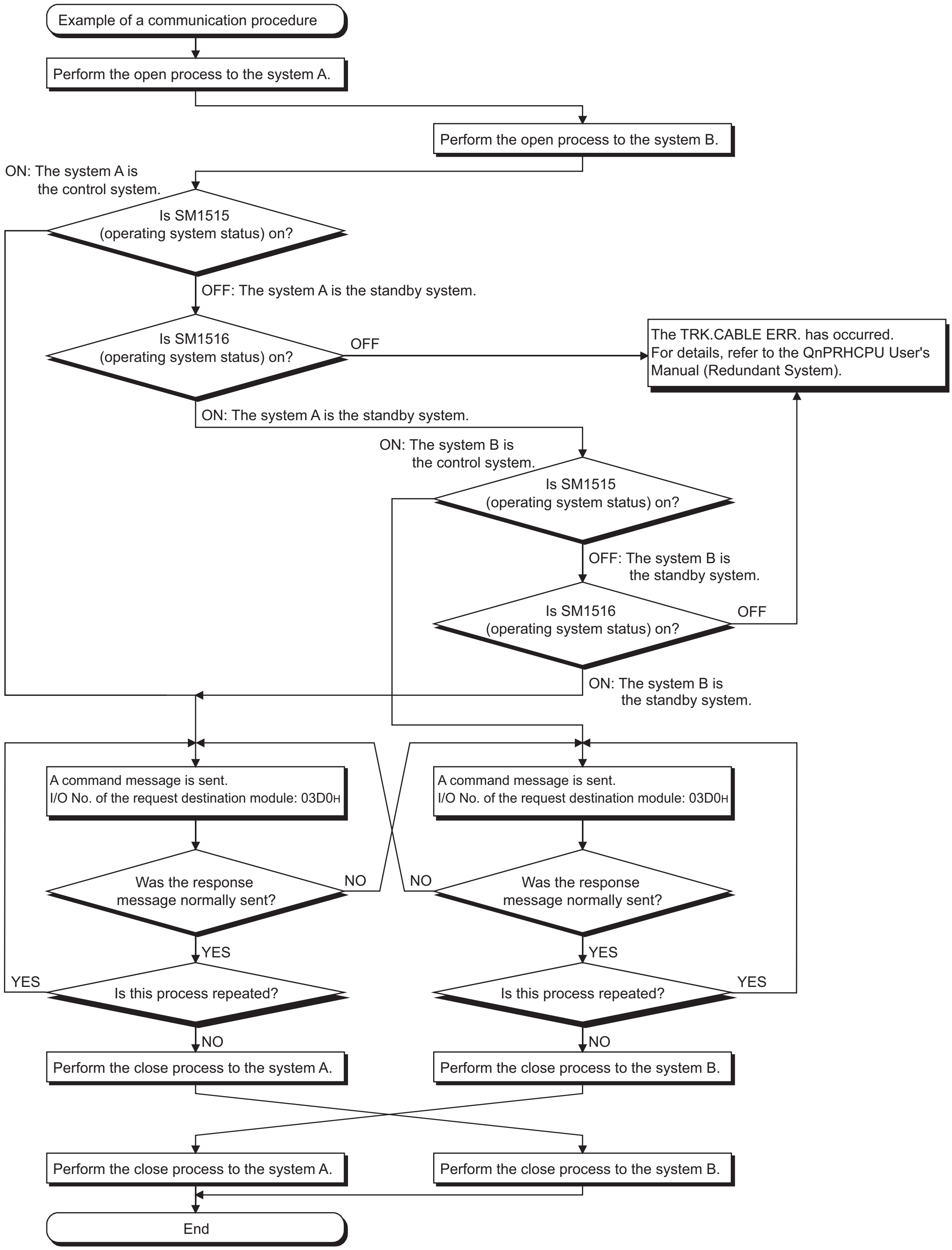
Ex. For access to the system that is not the connection destination (example of connection to the control system and access to the standby system CPU module)



3. **To access the standby system CPU module, the connected device connects to the E71 mounted with the control system CPU module through the tracking cable. This enables access even if the communication line between the connected device and standby system becomes faulty.**

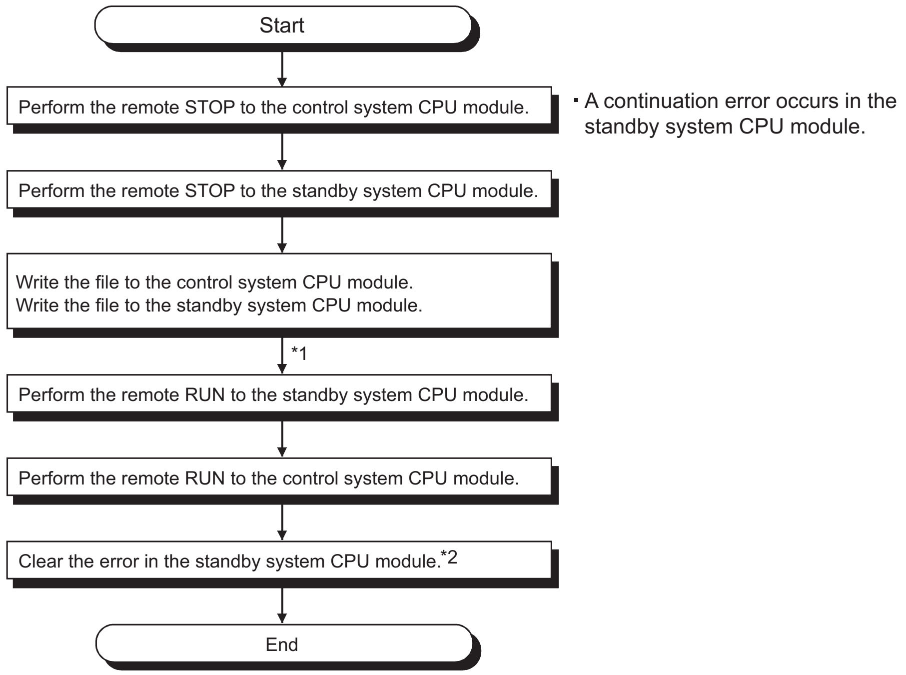
(b) Communication procedure example for access to the control system CPU module in a redundant system

The following shows a communication procedure example. Place the E71 in open wait status and perform open/close processes from the connected device.



(c) Precautions for writing a parameter file or program file

- Write the same file to the system A and system B. If different files are written to the systems, or a file is written to only one system, an error will result.
- Write the file while the switch on the CPU module is set to STOP.
- Write the file in the following procedure.



*1 After writing the parameter file, reset the both CPU modules.
*2 When the operating status of the control system CPU module is switched from STOP to RUN, check the error status of the standby system CPU module. If an error has occurred, set the error code (6010_H) to SD50 then turn on SM50 to clear the error.

(4) Communicating using a fixed buffer

(a) Receive process in a standby system

When data is sent to the E71 mounted with the standby system CPU module, the data received by the E71 is discarded, and the receive process is not performed. (Fixed buffer reception status signal does not turn on.)

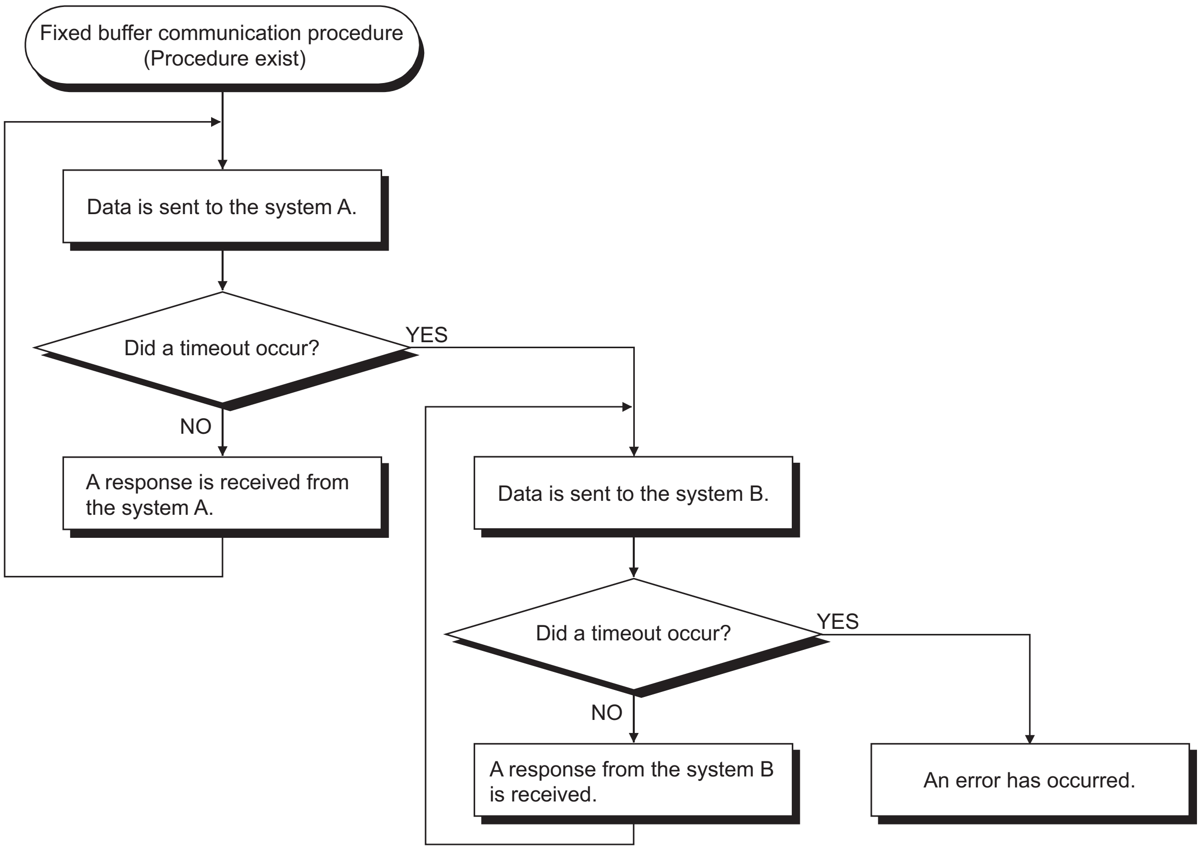
(b) Receive process in an interrupt program

When the control system is switched to the standby system by system switching before execution of an interrupt program, the interrupt factor is held. When the system is switched again and the standby system is switched to the control system, the interrupt program is executed by the held interrupt factor. (The interrupt factor is not transferred to the other system.)

(c) When data is sent from the connected device

- When using the "Procedure exist" method
  If a response timeout to the E71 occurs, change the connection destination to the other system and send the data.
- When using the "No procedure" method
  Send the same data to both the control system and standby system.

The following shows an example of the sending procedure for sending data to a redundant system using the "Procedure exist" method.



Point

Take care of the following when performing a resend process at system switching.

- When performing communications while synchronizing sending and receiving, system switching may occur during the communications, and the system may be switched with the sending and receiving not synchronized. To perform system switching, restart communications after initializing the synchronization for safety.
- After executing the WRITE instruction, execute it again because it is difficult to determine that the instruction has been completed. However, note that the same instruction may be executed twice.

(5) Communicating using a random access buffer

The buffer memory of the E71 is not tracked. Therefore, when writing data to the random access buffer, write the same data to the control system and standby system.

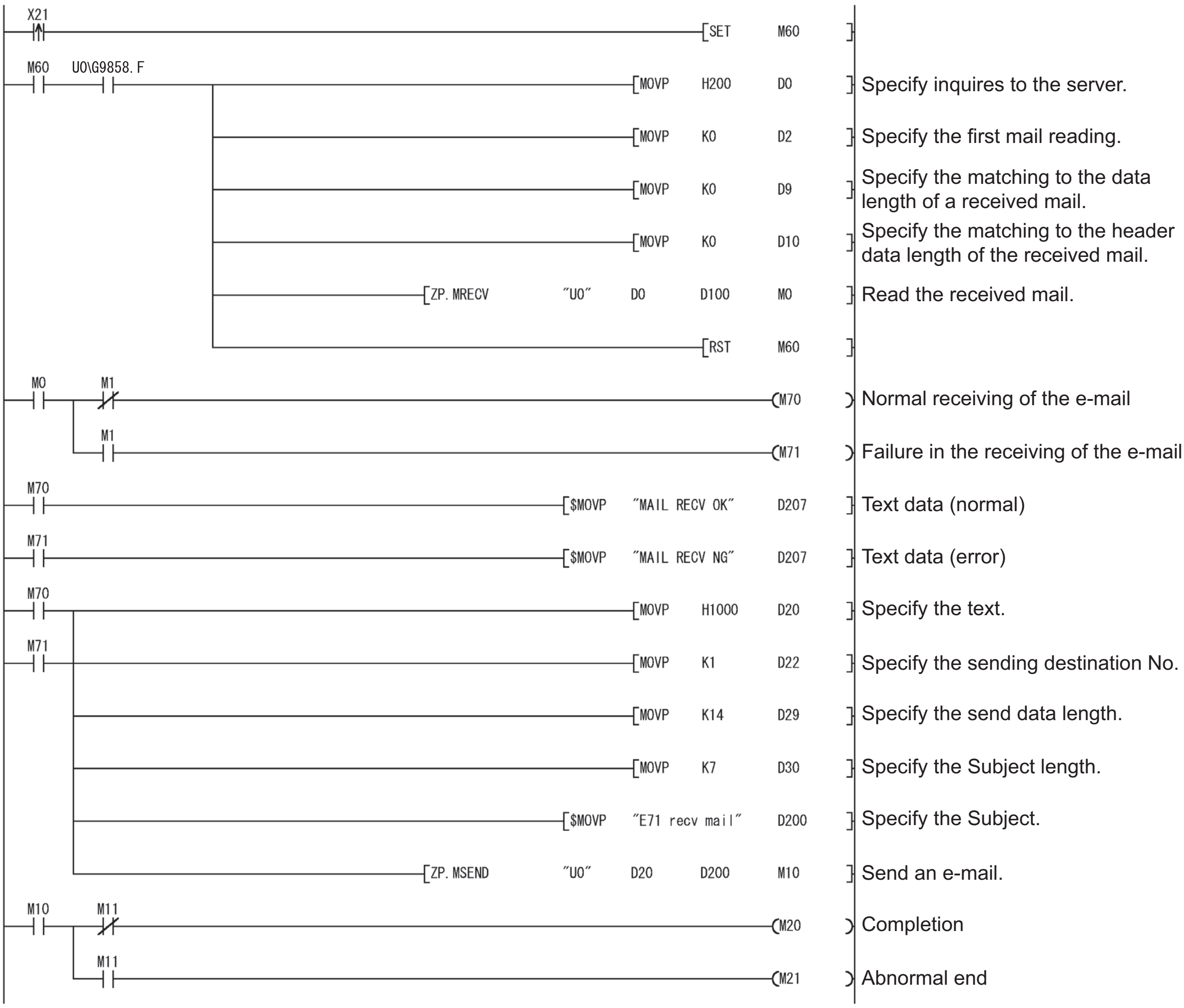
(6) Using the e-mail function

(a) Receiving e-mails

- After the E71 receives an e-mail, send a response mail to the mail sending source so that the mail sending source may recognize the arrival of the e-mail. If the receiving cannot be recognized, send the e-mail again.
- After the MRECV instruction is executed, the read e-mail is deleted from the mail server. Therefore, when system switching occurs during execution of the MRECV instruction, the mail may not be received by the new control system CPU module after system switching even if the MRECV instruction is re-executed. (The e-mail has been deleted from the mail server.)

(b) E-mail receiving program

In the following program, turning on X21 allows the E71 mounted on the slot where the I/O signals of X/Y00 to X/Y1F are assigned to receive an e-mail by execution of the MRECV instruction and send a response mail to the sending source by execution of the MSEND instruction.



(c) Using the notification function

Because a notification mail may be sent from both the control system and standby system in either of the following conditions, perform the process that discards the same e-mail that is received later on the receiving side personal computer.*1

- The CPU operating status has been set as the notification condition.
- The device data set as the notification condition is being tracked.

*1 By setting the following SM devices as notification conditions, the system mode in the redundant system can be identified with a notification mail. The devices can be used as conditions for identifying the system having the same e-mail.
- SM1511 (System A identification flag)
- M1512 (System B identification flag)
- SM1515, SM1516 (operating system status)

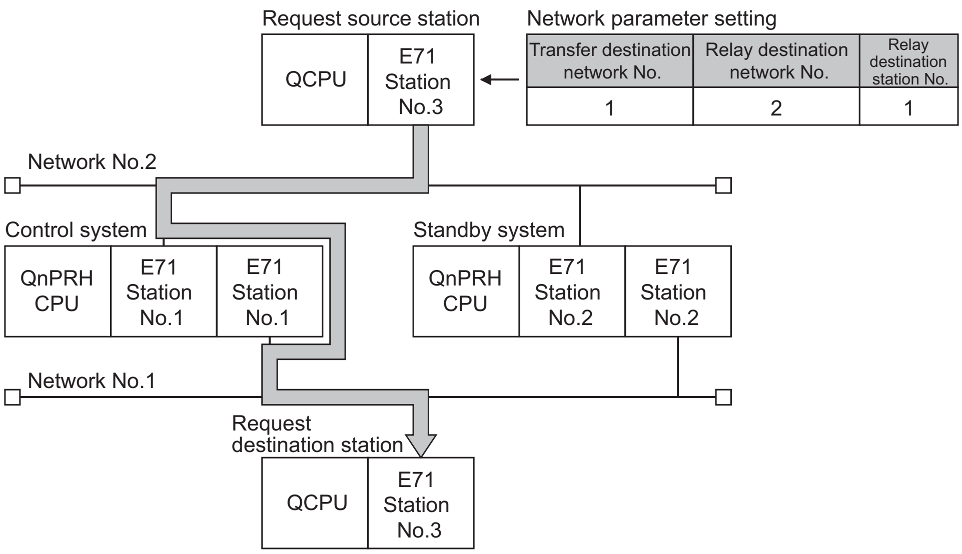
(7) Communicating over CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10

(a) When a redundant system is in the network

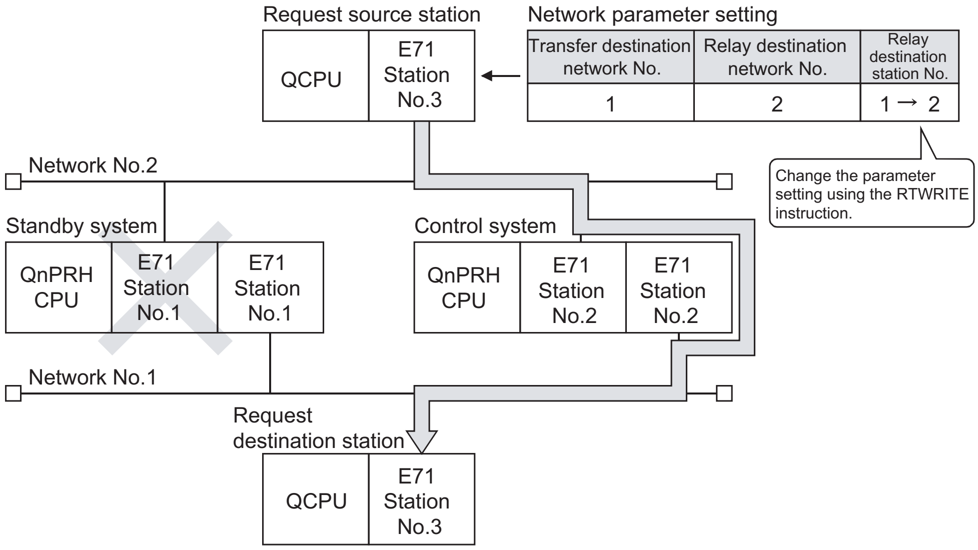
When performing access through a redundant system, change the routing parameter settings on the request source station or relay station using the RTWRITE instruction at system switching.

(b) When performing access through a redundant system

To access another station through Ethernet using the CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 relay communications, set the routing parameters on the request source station or relay station. To perform access through a redundant system, set the station that becomes a control system as a routing station.



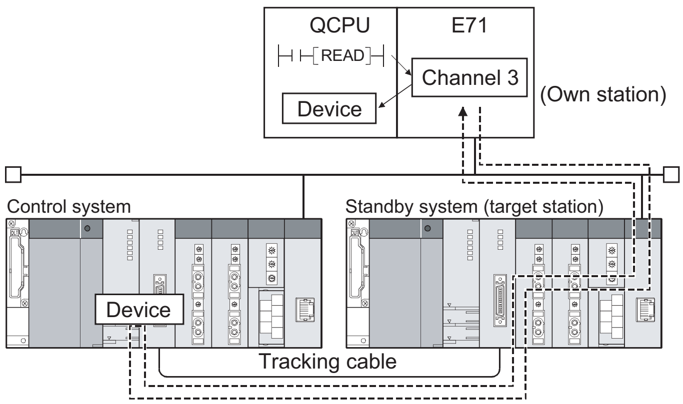
If system switching occurs, change the routing parameter settings using the RTWRITE instruction so that the request source station or relay station accesses the destination through the station of the new control system after system switching.



(8) When the CPU module accesses a programmable controller on another station using data link instructions

(a) Redundant system supported data link instructions

- By specifying the destination in Target station CPU type of the control data in any of the following data link instructions, access can be performed to the control system/standby system or system A/system B.
    - READ/SREAD instructions
    - WRITE/SWRITE instructions
    - REQ instruction
- Operation performed for access to the control system/standby system or the system A/system B (example of executing the READ instruction)
  When the target station is the standby system, because the received command is addressed to the control system CPU module (Target station CPU type: $3D0_H$), data is relayed to the control system CPU module through the tracking cable to read the data in the control system CPU module.



14

14.7 Redundant System Function
14.7.4 Data communications in a redundant system

(b) Process at abnormal end

When a data link instruction is executed for the specified control system CPU module or standby system CPU module in a redundant system, the data link instruction may result in an error if system switching occurs in the target station (error codes: 4244_H, 4248_H). If a data link instruction results in an error with either of the aforementioned error codes, execute the data link instruction again.

(c) SEND instruction

- When the target station is in a redundant system, the communication request source station must identify that the target station is the control system to execute the SEND instruction. When the target station is the standby system, the target station storage channel cannot be used since the RECV instruction is not executed at the target station after data is sent by the SEND instruction. (Channel being used)
- When a redundant system exists on the network where a broadcast is performed, the storage channel cannot be used since the RECV instruction is not executed for the standby system. (Channel being used)

(d) RECV instruction and interrupt program (RECVS instruction)

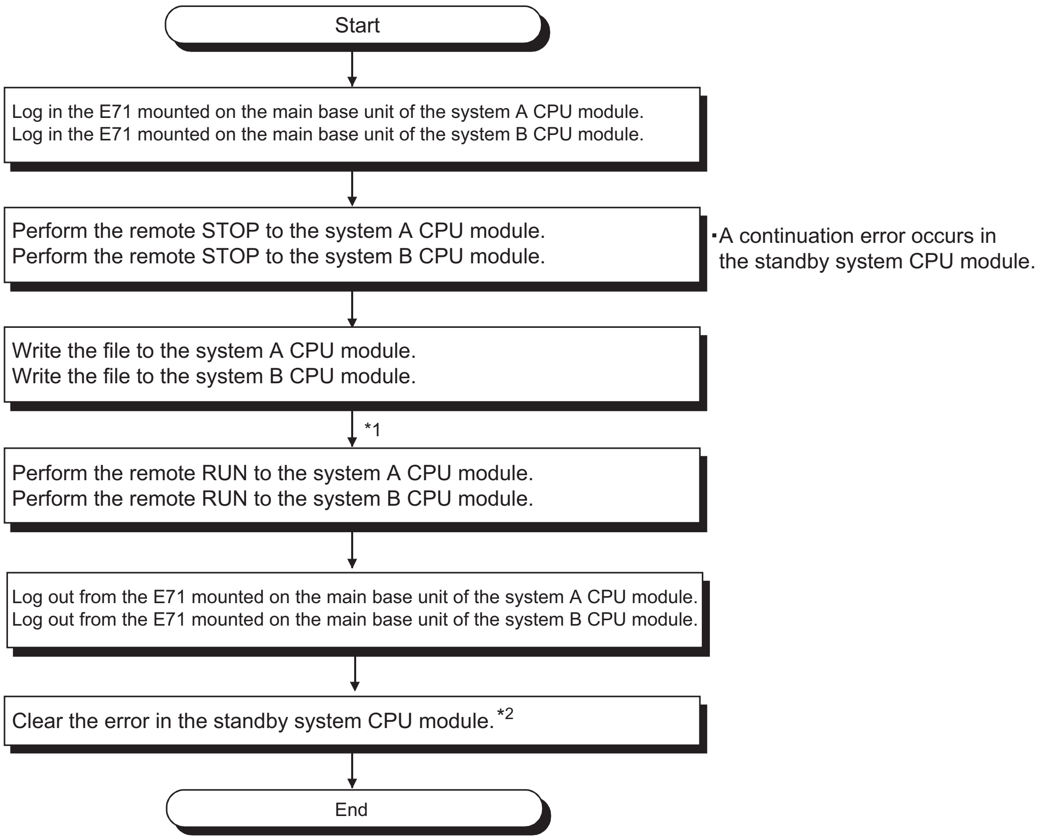
When the SEND instruction is executed for the redundant system, the process of the RECV instruction or interrupt program (RECVS instruction) changes depending on the following conditions.

- When system switching occurs between execution of the SEND instruction for the control system and execution of the RECV instruction or interrupt program
  When the control system is switched to the standby system before execution of the RECV instruction or interrupt program, it holds the buffer memory area data, RECV instruction execution request (address: CD_H) and the interrupt factor (interrupt pointer) of the interrupt program. When system switching occurs again and the standby system is switched to the control system, the RECV instruction and interrupt program are executed using the data held in the buffer memory area, RECV instruction execution request and the interrupt factor of the interrupt program.
- When the SEND instruction is executed for the standby system
  When the standby system receives data from the sending station, it holds data in the buffer memory area, RECV instruction execution request (address: CD_H) and the interrupt factor (interrupt pointer) of the interrupt program. When system switching occurs and the standby system is switched to the control system, the RECV instruction and interrupt program are executed using the data held in the buffer memory area, RECV instruction execution request area and the interrupt factor of the interrupt program.

(9) Using the file transfer (FTP server) function

(a) Precautions for writing a parameter file or program file

- Write the same file to the system A and system B. If different files are written to the systems, or a file is written to only one system, an error will result.
- Write the file while the switch on the CPU module is set to STOP.
- Write the file in the following procedure.



*1 After writing the parameter file, reset the both CPU modules.

*2 When the operating status of the control system CPU module is switched from STOP to RUN, check the error status of the standby system CPU module. If an error has occurred, set the error code (6010_H) to SD50 then turn on SM50 to clear the error.

(10)Using dedicated instructions

If system switching occurs during execution of the dedicated instruction, the dedicated instruction may not be completed. Execute the dedicated instruction again from the new control system CPU module after system switching.

14.8 Alive Check Function

When the E71 has not communicated with the connected device for a certain period of time while the connection is open, this function checks whether the connected device is alive by sending an alive check message to the connected device and waiting for the response.

(1) Parameter setting

To enable the alive check function, select "Confirm" under "Existence Confirmation" in the open setting.
(☞Page 87, Section 7.1.4)

In TCP/IP communications, the following methods can be selected in the Ethernet operation setting. (☞Page 85, Section 7.1.3)

(2) Alive check method

The following alive check methods are available.

(a) Check using the PING command

This method is used for a connection opened using TCP/IP or UDP/IP. The E71 performs an alive check by sending the PING command (ICMP echo request/response function) to the connected device with which communications have not been performed for a certain period of time and waiting to see whether the response is received.*1

*1 The E71 automatically sends 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 communications with the connected device is closed.)

(b) Check using the KeepAlive function

This method is used for a connection opened using TCP/IP. The E71 performs an alive check by sending an alive check ACK message to the connected device with which communications have not been performed for a certain period of time and waiting to see whether the response is received.*2

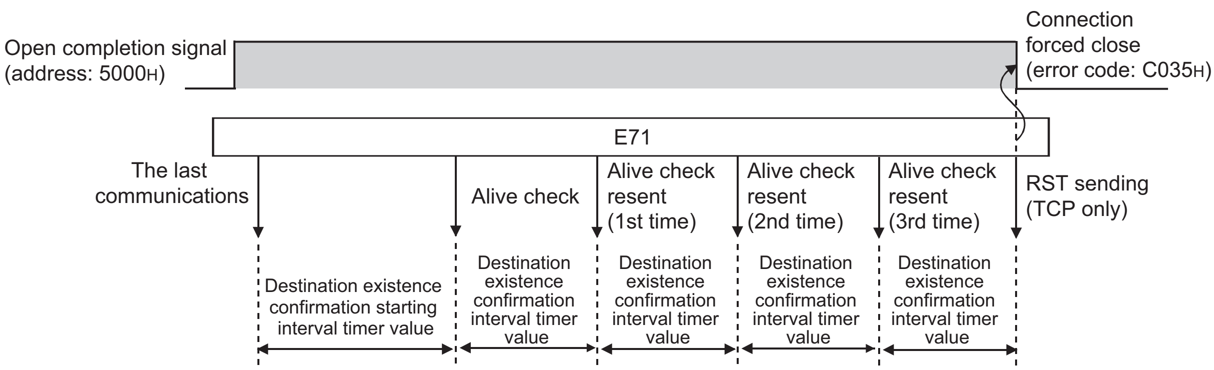
*2 The connection may be disconnected if the connected device does not support the TCP KeepAlive function (response to a KeepAlive ACK message).

(3) If a response message cannot be received from the connected device

If a response message cannot be received from the connected device (or if an error has been detected) using the alive check function, the following processes are performed.

- The corresponding connection will be forcibly closed. (The line is disconnected.) Open the connection using a user program again.
- Open completion signal is turned off, and the error code ($C035_H$) is stored in the open error code storage area.

Ex. Assuming that the values*1 are set so that the number of retries may be three, the E71 performs an alive check at the timing shown in the figure below (an example of alive check using the PING command).



*1 The values of the number of retries and timers can be changed in the initial setting. (Page 354, Appendix 4.1)

CHAPTER 15 DEDICATED INSTRUCTIONS

Dedicated instructions facilitate programming for using intelligent function modules. This chapter describes dedicated instructions that can be used in the E71.

15.1 List of Dedicated Instructions

(1) Dedicated instructions for using basic functions

The following table lists dedicated instructions for using the functions explained in this manual.

Instruction	Description	Reference
OPEN	Establishes (opens) a connection with the connected device to perform data communications.	Page 241, Section 15.5
CLOSE	Disconnects (closes) a connection with the connected device performing data communications.	Page 245, Section 15.6
ECPRTCL	Executes the protocols registered in the flash ROM of the E71 using the Predefined Protocol Support Function of GX Works2.	Page 248, Section 15.7
BUFSND	Sends data to the connected device through communications using a fixed buffer.	Page 259, Section 15.8
BUFRCV	Reads data received from the connected device through communications using a fixed buffer (used in the main program).	Page 263, Section 15.9
BUFRCVS	Reads data received from the connected device through communications using a fixed buffer (used in an interrupt program).	Page 267, Section 15.10
ERRCLR	Turns off LEDs of the E71 or clears error information stored in the buffer memory.	Page 270, Section 15.11
ERRRD	Reads error information stored in the buffer memory of the E71.	Page 273, Section 15.12
UINI	Re-initializes the E71.	Page 277, Section 15.13

(2) Dedicated instructions for using special functions

The following table lists dedicated instructions for using special functions. For details on the dedicated instructions, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Application)

Function	Instruction	Description
E-mail function	MRECV	Reads the received e-mails.
	MSEND	Sends e-mails.
Communications using data link instructions	READ	Reads word device data from other stations.
	RECV	Reads data received from other stations. (Used in the main program.)
	RECVS	Reads data received from other stations. (Used in an interrupt program.)
	REQ	Performs remote RUN/STOP to CPU modules on other stations.
		Reads/writes the clock data of other stations.
	SEND	Sends data to other stations.
	SREAD	Reads word device data from other stations (with a completion device).
	SWRITE	Writes word device data to other stations (with a completion device).
	WRITE	Writes word device data to other stations.
	ZNRD	Reads word device data from other stations (ACPU).
	ZNWR	Writes word device data to other stations (ACPU).

15.2 Parameter Settings for Using Dedicated Instructions

When using dedicated instructions, set the parameters of each function.

15.2.1 When using data link instructions

When using data link instructions, set the following parameters.

- Station No. <-> IP information setting
- Routing parameters setting

For details on the settings, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Application)

15.3 Precautions for Dedicated Instructions

This section describes precautions for using dedicated instructions.

(1) When changing data specified by dedicated instructions

Do not change any data (e.g. control data) until execution of the dedicated instruction is completed.

(2) When the dedicated instruction is not completed

Check that "Online" has been set under "Mode" in the network parameter window. Dedicated instructions cannot be executed during offline.

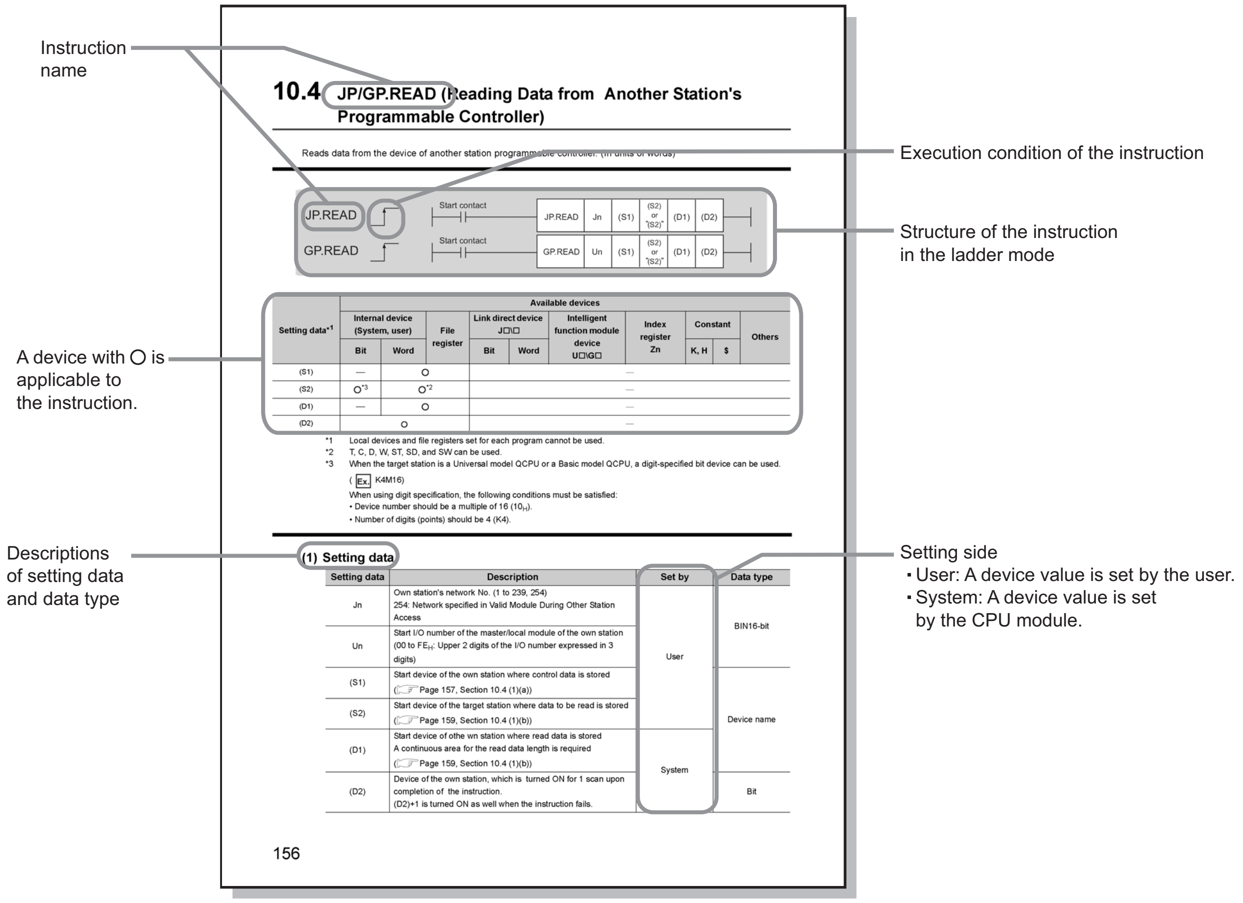
(3) Securing service process time

When accessing a programmable controller on another station using a data link instruction during the Ethernet diagnostics, execution of the instruction may be delayed. Take the following measures to execute the data link instruction after the Ethernet diagnostics. (For safety CPUs, the following measures cannot be applied.)

- Execute the COM instruction.
- Reserve time for a communication process for 2 to 3 ms.
    - For a Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU, set the service processing time in Time reserved for communication processing (SD315).
    - For a Universal model QCPU, configure "Service Processing Setting" in the "PLC System" tab of the PLC parameter window.

15.4 Organization of the Dedicated Instruction Sections

The following illustration is for explanation purpose only, and should not be referred to as an actual documentation.



(1) Instruction execution conditions

Instructions can be executed under the following conditions.

Any time	During on	On the rising edge	During off	On the falling edge
No symbol				

(2) Available devices

The following devices can be used.

Internal device		File register	Constant	
Bit*1	Word		K, H	$
X, Y, M, L, F, V, B	T, ST, C, D, W	R, ZR	K□, H□	$□

*1 For bit data, a bit-specified word device can be used. For example, Word device.Bit No. is used for the bit specification of a word device. (The bit number is specified in hexadecimal.)

For example, the bit 10 of D0 is specified by D0.A. However, bit specification cannot be used for timer (T), retentive timer (ST), and counter (C).

For details on each device, refer to the following.

User's manual for the CPU module used (function explanation, program fundamentals)

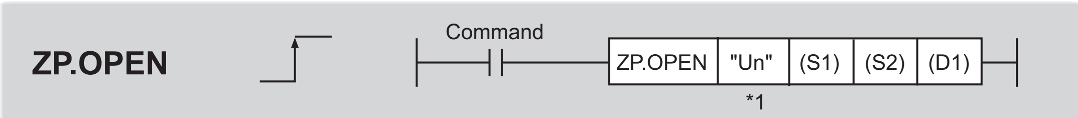
(3) Data type

The following data types can be used.

Data type	Description
Bit	Bit data or the start number of bit data
BIN 16-bit	16-bit binary data or the start number of word device
BIN 32-bit	32-bit binary data or the start number of double-word device
BCD 4-digit	Four-digit binary-coded decimal data
BCD 8-digit	Eight-digit binary-coded decimal data
Real number	Floating-point data
Character string	Character string data
Device name	Device name data

15.5 ZP.OPEN

This instruction establishes (opens) a connection with the connected device to perform data communications.



Setting data[*2]	Available device									
	Internal device (system, user)		File register	Link direct device J□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	-	○	○	-				○	-	-
(S2)	-	○	○	-				-	-	-
(D1)	○	○	○	-				-	-	-

*1 If the own station is a Basic model QCPU (function version B or later), Universal model QCPU, or safety CPU, " " (double quotation) of the first argument can be omitted.

*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Connection number (1 to 16)[*1]		BIN 16-bit
(S2)	Start number of the device that stores control data	User, system	Device name
(D1)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

*1 For safety CPUs, only connection numbers 1 to 8 can be specified. If the specified value is out of range, "OPERATION ERROR" (error code: 4101) occurs.

(2) Control data

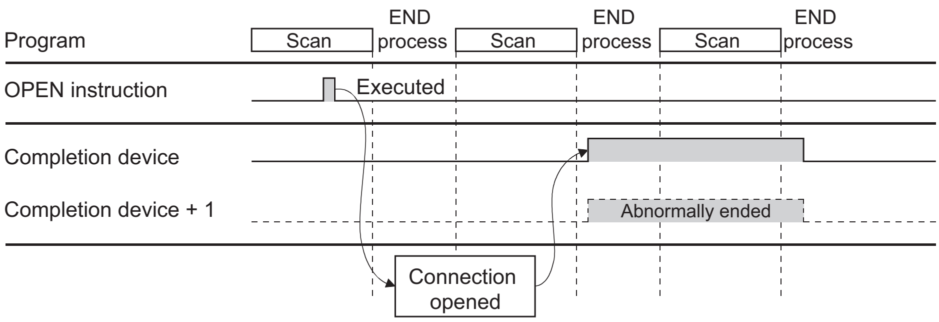
Device	Item	Setting data	Setting range	Set by
(S2)+0	Execution type/completion type	To open a connection, specify whether to use the parameter settings of a programming tool or to use the settings stored in the following control data starting from (S2)+2. 0000_H: Use the parameter settings of the programming tool. 8000_H: Use the settings of control data starting from (S2)+2.	0000_H, 8000_H	User
(S2)+1	Completion status	Stores the status at completion. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System
(S2)+2	Usage setting area	Specify the application of a connection. • Usage of fixed buffer (b0) 0: For sending or fixed buffer communications are not performed 1: For receiving • Destination existence confirmation (b1) 0: No confirm 1: Confirm • Pairing open (b7) 0: Disable 1: Enable • Communication method (protocol) (b8) 0: TCP/IP 1: UDP/IP • Fixed buffer communication procedure (b10, b9) 00: Procedure exists 01: No procedure 10: Predefined protocol • Open system (b15, b14) 00: Active open or UDP/IP 10: Unpassive open 11: Fullpassive open	As described in the left	User
(S2)+3	Own station port No.	Specify the port number of the own station.	401_H to 1387_H, $138B_H$ to $FFFE_H$	User
(S2)+4 (S2)+5	Destination IP address	Specify the IP address of the connected device.	1_H to $FFFFFFFF_H$ ($FFFFFFFF_H$: broadcast)	User
(S2)+6	Destination port No.	Specify the port number of the connected device.	1_H to $FFFF_H$ *1 ($FFFF_H$: broadcast)	User
(S2)+7 (S2)+8 (S2)+9	Destination MAC address	Specify the MAC address of the connected device.	000000000000_H to $FFFFFFFFFFFF_H$	User

*1 The range of 1_H to 400_H is available only for the QJ71E71-100 with the serial number (first five digits) of "15042" or later.

(3) Functions

- This instruction performs the open process for a connection specified by (S1) for the module specified by Un. The setting value used for the open process is selected by (S2)+0.
- Completion of the OPEN instruction can be checked with Completion device (D1)+0 and (D1)+1.
  - Completion device (D1)+0: Turns on at the END process of the scan where the OPEN instruction is completed and turns off at the next END process.
  - Completion device (D1)+1: Turns on and off depending on the completion status of the OPEN instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the OPEN instruction is completed and turns off at the next END process.

[Operation while the OPEN instruction is being executed]



- The ZP.OPEN instruction is executed on the rising edge (OFF → ON) of the open command.

Point

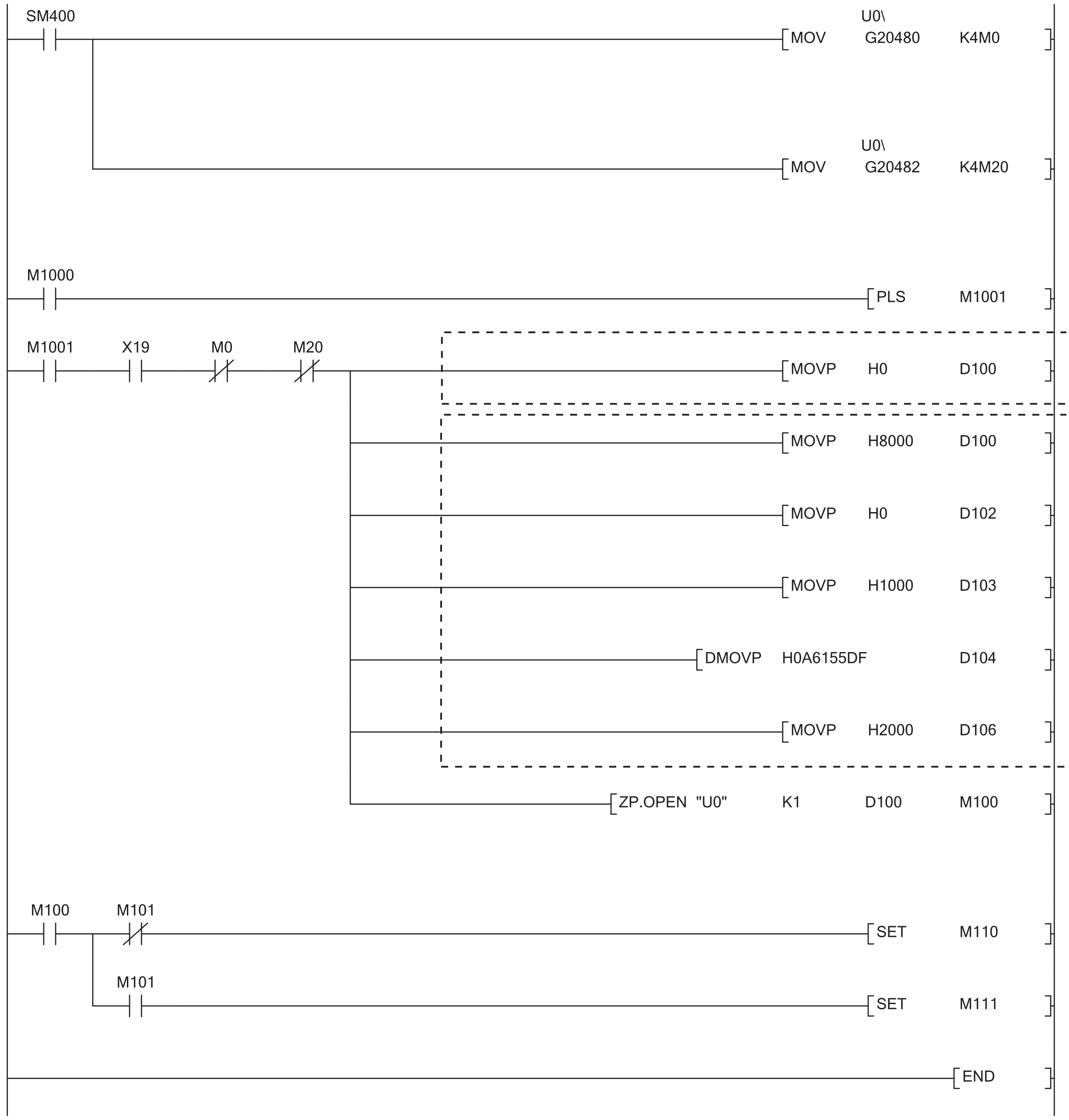
Do not perform an open process using I/O signals and using the dedicated instruction together for the same connection. Doing so will result in malfunction.

(4) Errors

- If a dedicated instruction ends with an error, Completion device (D1)+1 turns on, and the error code is stored in Completion status (S2)+1.

(5) Program example

The following shows a sample program that performs an Active open process on the connection number 1 for TCP/IP communications (when the I/O signals of the E71 are X/Y00 to X/Y1F).



*1 Necessary when the open setting of the programming tool is used. (In this case, the program marked with *2 is unnecessary.)

*2 Necessary when the open setting of the program is used. (In this case, the program marked with *1 is unnecessary.)

Point

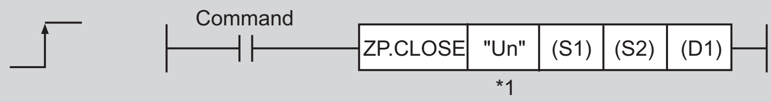
For safety CPUs, data stored in the buffer memory of the intelligent function module cannot be used. The program must be modified using the I/O signals of the intelligent function module corresponding to the buffer memory.
For use with safety CPUs, refer to the following.

QSCPU User's Manual (Function Explanation, Program Fundamentals)

15.6 ZP.CLOSE

This instruction disconnects (closes) a connection with the connected device performing data communications.

ZP.CLOSE



Setting data[*2]	Available device									
	Internal device (system, user)		File register	Link direct device J□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	-	○	○	-				○	-	-
(S2)	-	○	○	-				-	-	-
(D1)	○	○	○	-				-	-	-

*1 If the own station is a Basic model QCPU (function version B or later), Universal model QCPU, or safety CPU, " " (double quotation) of the first argument can be omitted.

*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Connection number (1 to 16)[*1]	User	BIN 16-bit
(S2)	Start number of the device that stores control data	System	Device name
(D1)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

*1 For safety CPUs, only connection numbers 1 to 8 can be specified. If the specified value is out of range, "OPERATION ERROR" (error code: 4101) occurs.

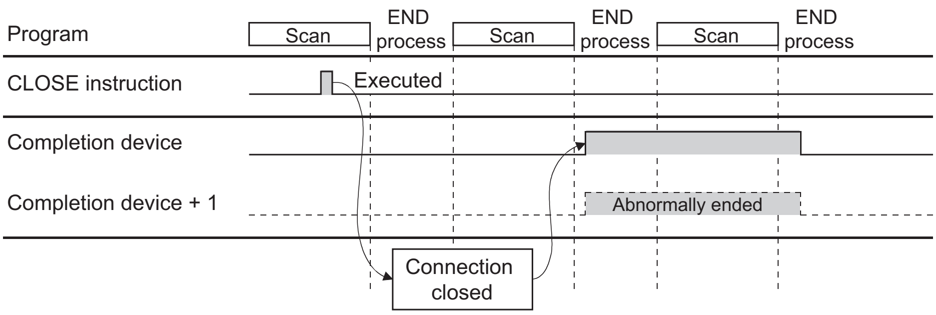
(2) Control data

Device	Item	Setting data	Setting range	Set by
(S2)+0	System area	-	-	-
(S2)+1	Completion status	Stores the status at completion. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System

(3) Functions

- This instruction performs the close process for a connection specified by (S1) for the module specified by Un (connection closed).
- Completion of the CLOSE instruction can be checked with Completion device (D1)+0 and (D1)+1.
  - Completion device (D1)+0: Turns on at the END process of the scan where the CLOSE instruction is completed and turns off at the next END process.
  - Completion device (D1)+1: Turns on and off depending on the completion status of the CLOSE instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the CLOSE instruction is completed and turns off at the next END process.

[Operation while the CLOSE instruction is being executed]



- The ZP.CLOSE instruction is executed on the rising edge (OFF → ON) of the close command.

Point

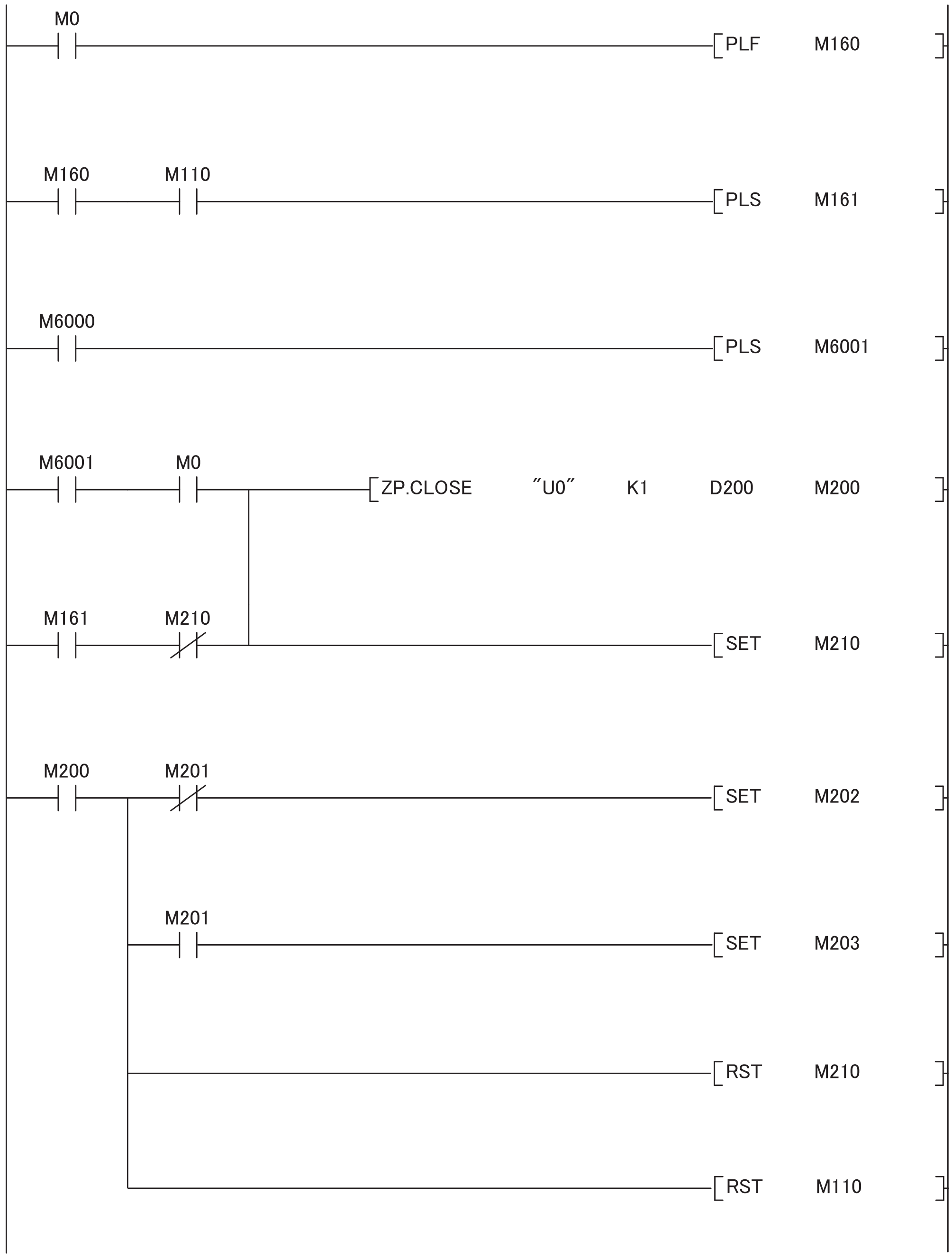
Do not perform a close process using I/O signals and using the dedicated instruction together for the same connection. Doing so will result in malfunction.

(4) Errors

- If a dedicated instruction ends with an error, Completion device (D1)+1 turns on, and the error code is stored in Completion status (S2)+1.

(5) Program example

The following shows a sample program that closes the connection number 1 (when the I/O signals of the E71 are X/Y00 to X/Y1F).



Point

For safety CPUs, data stored in the buffer memory of the intelligent function module cannot be used. The program must be modified using the I/O signals of the intelligent function module corresponding to the buffer memory.
For use with safety CPUs, refer to the following.
QSCPU User's Manual (Function Explanation, Program Fundamentals)

15.7 GP.ECPRTCL

This instruction executes the protocols registered in the flash ROM of the E71 using the predefined protocol support function of GX Works2.

GP.ECPRTCL

Setting data*1	Available device									
	Internal device (system, user)		File register	Link direct deviceJ□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
n1	○	○	○	-				○	-	-
n2	○	○	○	-				○	-	-
(S)	○	○	○	-				-	-	-
(D)	○	○	○	-				-	-	-

*1 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	BIN 16-bit
n1	Connection number (1 to 16)	User	BIN 16-bit Device name
n2	Number of consecutive protocol executions (1 to 8)	User	BIN 16-bit Device name
(S)	Start number of the device that stores control data	User/system	Device name
(D)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D)+1 also turns on if the instruction ends abnormally.	System	Bit

(2) Control data

Device	Item	Setting data	Setting range	Set by
(S)+0	Execution count result	Stores the number of protocols which are executed by using the ECPRTCL instruction. Protocols with errors are included in the count. When settings of the setting data or control data contain an error, "0" is stored.	0, 1 to 8	System
(S)+1	Completion status	Stores the status at completion. When multiple protocols are executed, the status of the protocol executed at last is stored. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System
(S)+2	Execution protocol number specification 1	Set the first protocol number to be executed.	1 to 128	User
(S)+3	Execution protocol number specification 2	Set the second protocol number to be executed.	0, 1 to 128	User
(S)+4	Execution protocol number specification 3	Set the third protocol number to be executed.	0, 1 to 128	User
(S)+5	Execution protocol number specification 4	Set the fourth protocol number to be executed.	0, 1 to 128	User
(S)+6	Execution protocol number specification 5	Set the fifth protocol number to be executed.	0, 1 to 128	User
(S)+7	Execution protocol number specification 6	Set the sixth protocol number to be executed.	0, 1 to 128	User
(S)+8	Execution protocol number specification 7	Set the seventh protocol number to be executed.	0, 1 to 128	User
(S)+9	Execution protocol number specification 8	Set the eighth protocol number to be executed.	0, 1 to 128	User
(S)+10	Matched receive packet No.1	When the communication type of the first protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the first protocol executed, "0" is stored.	0, 1 to 16	System
(S)+11	Matched receive packet No.2	When the communication type of the second protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the second protocol executed, "0" is stored. When the number of protocols which are executed is less than 2, "0" is stored.	0, 1 to 16	System

Device	Item	Setting data	Setting range	Set by
(S)+12	Matched receive packet No.3	When the communication type of the third protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the third protocol executed, "0" is stored. When the number of protocols which are executed is less than 3, "0" is stored.	0, 1 to 16	System
(S)+13	Matched receive packet No.4	When the communication type of the fourth protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the fourth protocol executed, "0" is stored. When the number of protocols which are executed is less than 4, "0" is stored.	0, 1 to 16	System
(S)+14	Matched receive packet No.5	When the communication type of the fifth protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the fifth protocol executed, "0" is stored. When the number of protocols which are executed is less than 5, "0" is stored.	0, 1 to 16	System
(S)+15	Matched receive packet No.6	When the communication type of the sixth protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the sixth protocol executed, "0" is stored. When the number of protocols which are executed is less than 6, "0" is stored.	0, 1 to 16	System
(S)+16	Matched receive packet No.7	When the communication type of the seventh protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the seventh protocol executed, "0" is stored. When the number of protocols which are executed is less than 7, "0" is stored.	0, 1 to 16	System
(S)+17	Matched receive packet No.8	When the communication type of the eighth protocol executed includes receiving, the matched receive packet number is stored. When the communication type is "Send Only", "0" is stored. If an error occurs to the eighth protocol executed, "0" is stored. When the number of protocols which are executed is less than 8, "0" is stored.	0, 1 to 16	System

(3) Functions

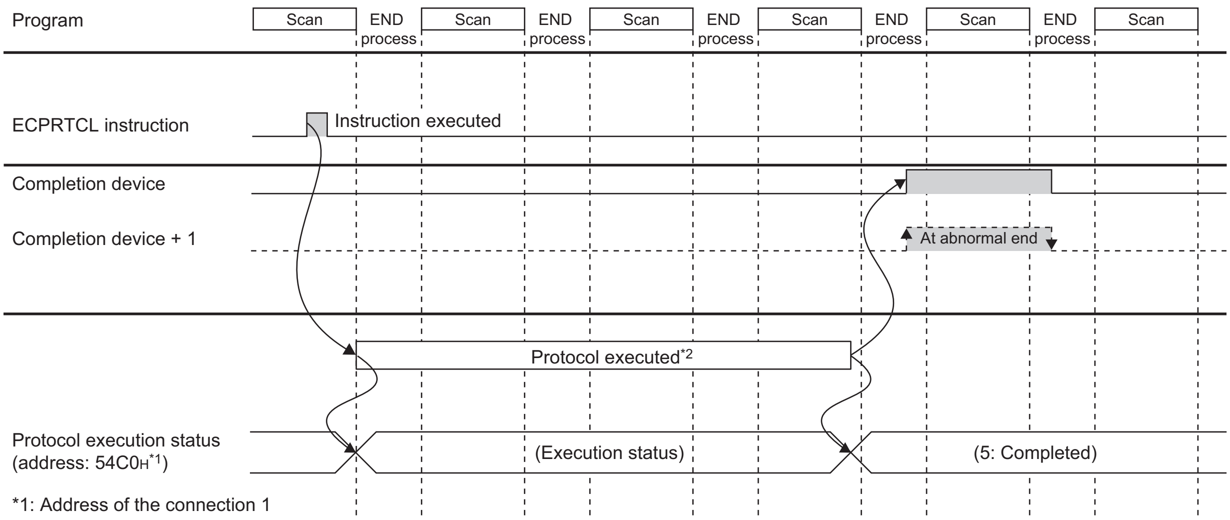
- The module specified by Un executes the protocol setting data written to the flash ROM.
  The protocol is executed according to the control data of the device specified by (S) and the following devices.
  The connection specified by n1 is used.
- Protocols are executed consecutively for the number of times specified by n2 (up to eight protocols) by one instruction.
- When two or more ECPRTCL instructions are executed simultaneously for the same connection, the following instruction will be ignored until the first instruction is completed.
- The number of protocols is stored in (S)+0.
- Protocol execution status can be checked in the predefined protocol support function execution status check area (address: $54C0_H$ to $55FF_H$).
- The communication type of the protocol executed and fixed buffer setting of the connection No. specified by n1 must be matched.
  The following table lists the combination of the communication type and the connection No. that can be specified by n1.

Communication type	Connection No. that can be specified by n1
When the communication type of all protocols executed (up to eight protocols) is "Send Only"	• A connection where "Send" is selected in the open setting (no paring open setting) • A connection where paring open is set*1*2
When the communication type of all protocols executed (up to eight protocols) is "Receive Only"	• A connection where "Receive" is selected in the open setting (no paring open setting) • A connection where paring open is set*1*2
When the communication type of any protocol executed (up to eight protocols) is "Send&Receive"	A connection where paring open is set*1*3
When the communication types of protocols executed (up to eight protocols) are "Send Only" and "Receive Only"	

*1 To specify a connection where paring open is set, either of pairing-opened connections can be specified.
*2 When a pairing-opened connection is specified, an extra connection is used.
*3 When the protocols executed include the communication type of "Send&Receive" or "Send Only" and "Receive Only", pairing open setting is required.

- Completion of the ECPRTCL instruction can be checked with Completion device (D) and (D1)+1.
  - Completion device (D)+0: Turns on at the END process of the scan where the ECPRTCL instruction is completed and turns off at the next END process.
  - Completion device (D)+1: Turns on and off depending on the completion status of the ECPRTCL instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the ECPRTCL instruction is completed and turns off at the next END process.

[Operation while the ECPRTCL instruction is being executed]



*2: Protocols are consecutively executed by the number of protocols specified in n2 of the setting data in the order specified with the control data (up to eight protocols) with one ECPRTCL instruction.

Point

- When multiple protocols are executed and an error has occurred in one protocol, the following protocols are not executed and the dedicated instruction ends abnormally.
- For the connection where the ECPRTCL instruction can be executed, "Predefined protocol" is set in Fixed Buffer Communication. Thus, the following communications are not available.
  - MC protocol communications
  - Communications using a fixed buffer (procedure exists)
  - Communications using a fixed buffer (no procedure)
  - Communications using a random access buffer
- File transfer (FTP server) function, e-mail function, Web function, or MELSOFT connection, which uses a dedicated connection, can be executed while the ECPRTCL instruction is being executed.
- When protocols which include Non-conversion Variable are executed and the data length of the variable for one packet exceeds 1920 bytes, the value of the CPU module device may not be acquired at one scan.
  Do not change the value of the CPU module device specified in Non-conversion Variable until the ECPRTCL instruction is completed.
  Buffer memory is not affected by the sequence scan of the CPU module. Assigning them in a variable enables more high-speed process than assigning the CPU module device.
- If the protocol setting data is written while the ECPRTCL instruction is being executed, protocol execution is canceled upon the writing completion. Protocol setting data writing during the ECPRTCL instruction execution (error code: $C430_H$) occurs and the ECPRTCL instruction ends abnormally.
- When the open setting for the specified connection and the communication type of the protocol executed does not match, Connection number setting error (error code: $C407_H$) occurs and the ECPRTCL instruction ends abnormally.
- When the waiting time is set to "0" (wait infinitely) in the Protocol Detailed Setting, the dedicated instruction is not completed until the data specified in the protocol setting is received.

(4) Errors

Completion device (D)+1 turns on and the error code is stored in Completion status (S)+1 in the following cases.

- When the setting value of the control data is abnormal
- When an error is detected in the protocol setting data registered in GX Works2
- When an error has occurred in protocols (the following protocols will not be executed.)
- When a value other than "Predefined protocol" is set in Fixed Buffer Communication for connection No. specified by n1

(5) Program example

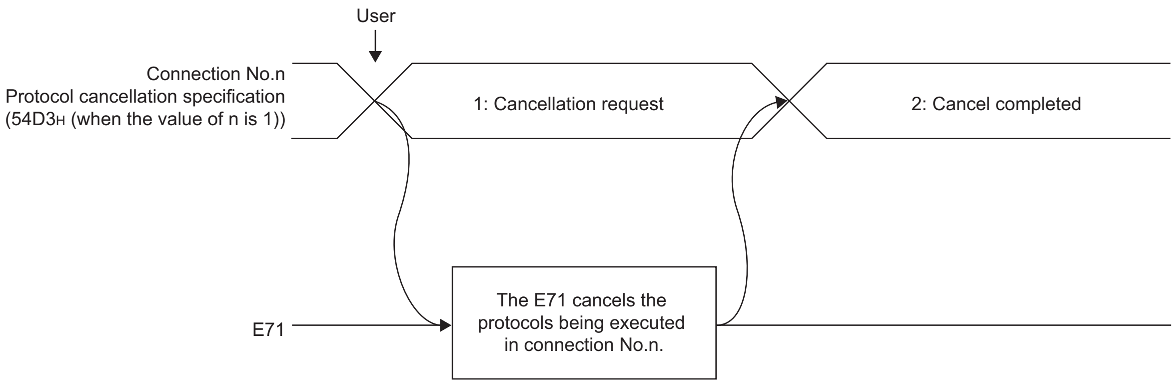
For the program example, refer to Page 128, Section 11.5.

(6) Canceling protocol execution

A protocol can be canceled during its execution.
This function is used to terminate the protocol execution forcibly when a communication error occurs with the connected device.

(a) Execution method for cancel request

Set the cancel request to Protocol cancellation specification (address: $54D3_H$ (Connection No.1)).



(b) Operations after execution of cancel request

[Operations of the ECPRTCL instruction]

- The ECPRTCL instruction ends abnormally and Protocol cancel request error (error code: $C404_H$) is stored in Completion status (S)+1.
- When cancel request is executed to the nth protocol while multiple protocols are being executed consecutively, the E71 terminates the nth protocol forcibly and the following protocols are not executed. The following table lists the control data that are stored when protocols are canceled.

Device	Item	Stored value
(S)+0	Execution count result	The number of protocols executed including canceled protocols (When cancel request is executed to the second protocol, 2 is stored.)
(S)+1	Completion status	Protocol cancel request error (error code: $C404_H$)
(S+10) to (S+17)	Matched receive packet No.1 to 8	Matched receive packet number of the executed protocol

[Operations of the E71]

- If the cancel request is executed when any protocol is not being executed, the E71 completes the cancel without any process.

Point

- If any data is received from the connected device after the protocol whose communication type includes receiving is canceled, the data will be discarded.
- The E71 checks whether any cancel request exists at periodic processing. For this reason, it may take some time to process the cancellation after the cancel request is executed.
- When Protocol cancellation specification (address: $54D3_H$ (Connection No.1)) is "1: Cancellation request", the next ECPRTCL instruction cannot be executed.

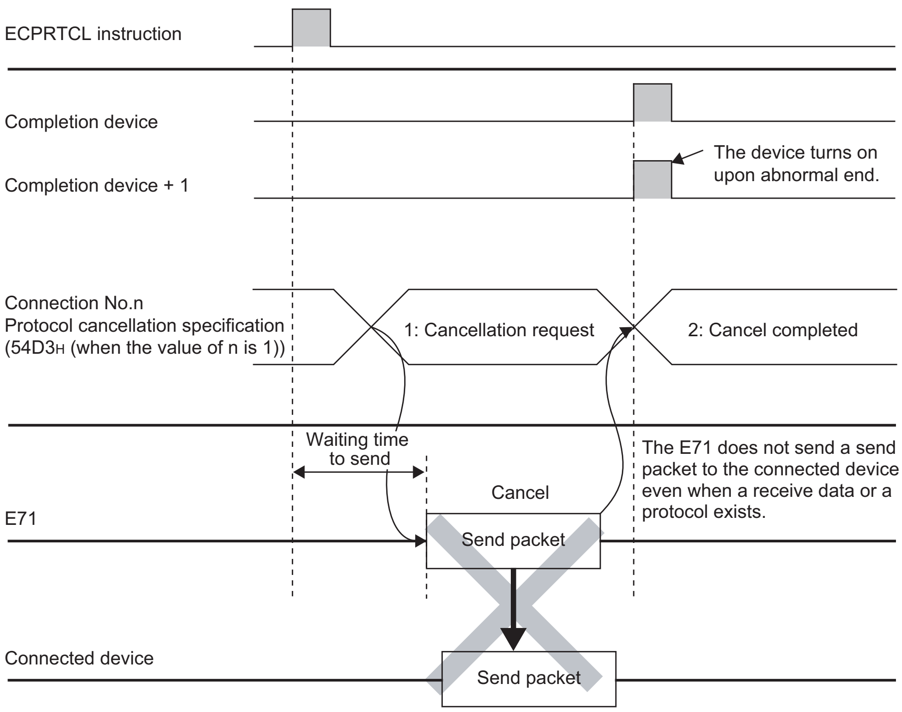
(c) Timing chart

The E71 performs the following process according to the timing of cancel request.

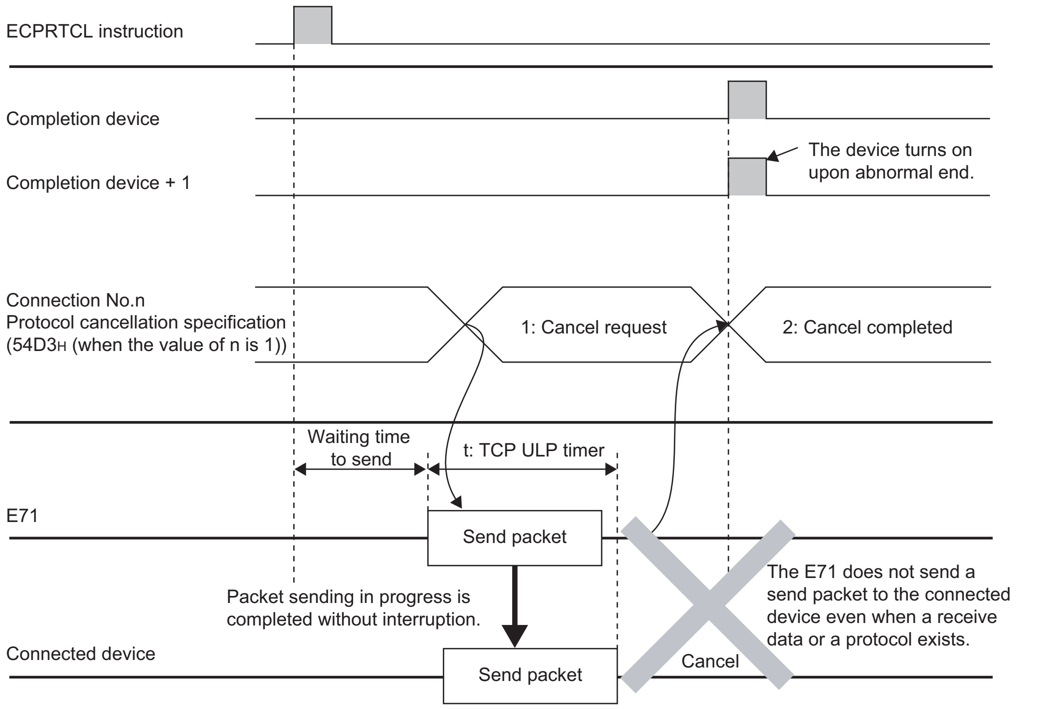
Protocol execution status (address: $54C0_H$)	E71 operations at cancel request
0: Unexecuted	No process
1: Waiting for transmission	The E71 cancels the sending and terminates the dedicated instruction forcibly.
2: Sending	The E71 terminates the dedicated instruction forcibly at sending completion.
3: Waiting for data reception	The E71 cancels the receiving and terminates the dedicated instruction forcibly.
4: Receiving	The E71 terminates the dedicated instruction forcibly at receiving completion.
5: Completed	The E71 terminates the dedicated instruction when protocols are executed consecutively.

The followings describe the operations of protocol cancellation according to the timing.

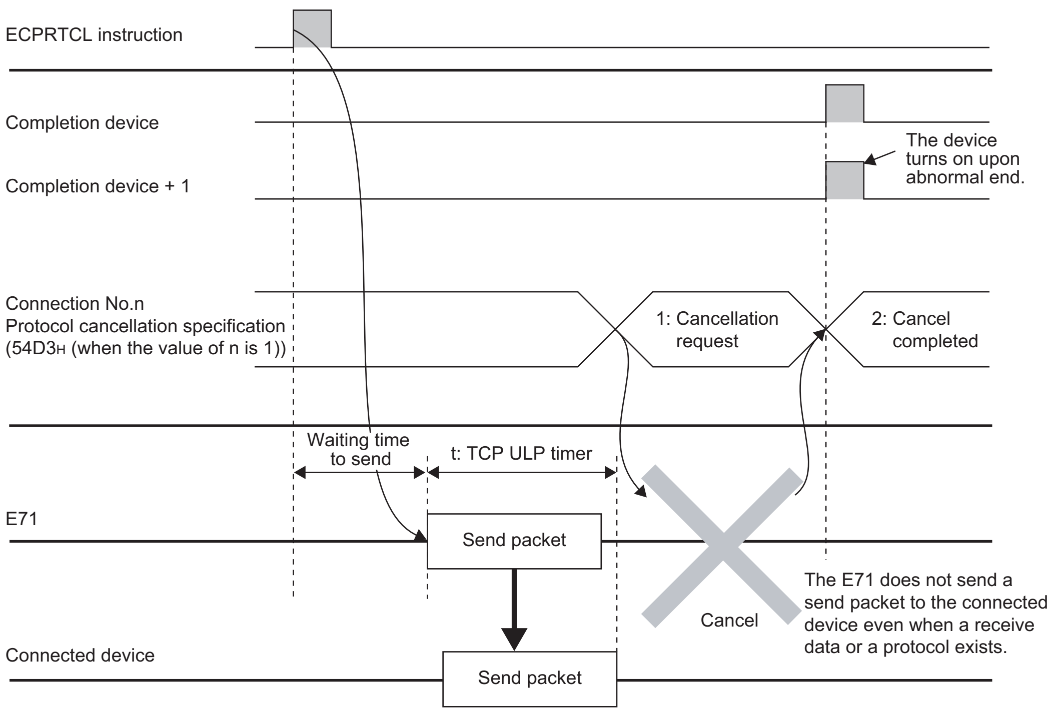
- When the cancel request is performed before sending (when "1" (Waiting for transmission) is set in Protocol execution status (address: $54C0_H$))



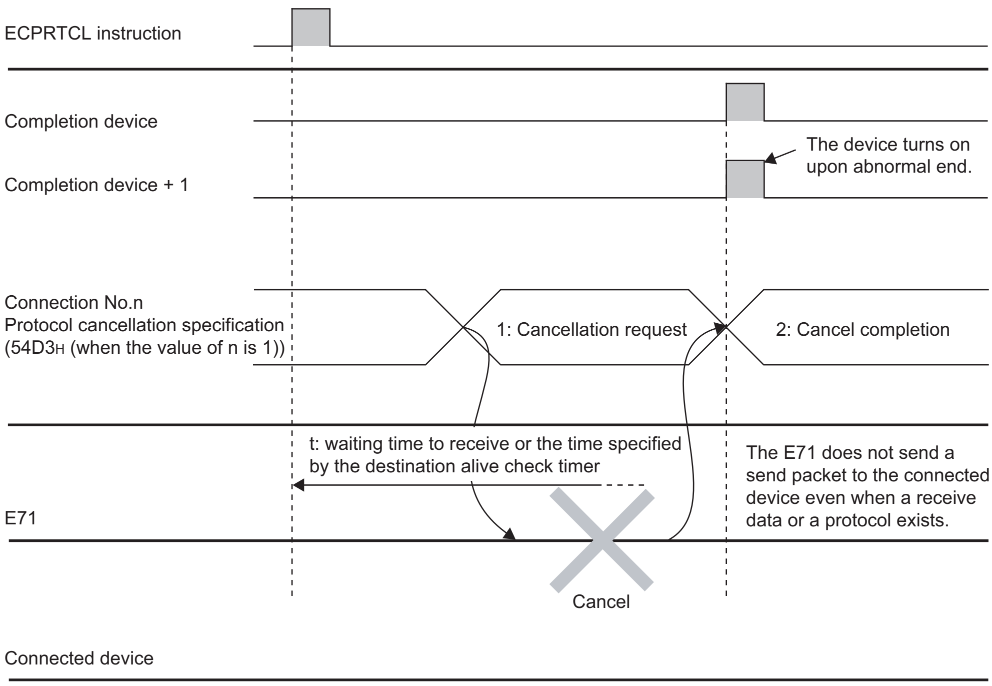
- When the cancel request is performed before sending completion (when "2" (Sending) is set in Protocol execution status (address: $54C0_H$) and the sending has not been completed)



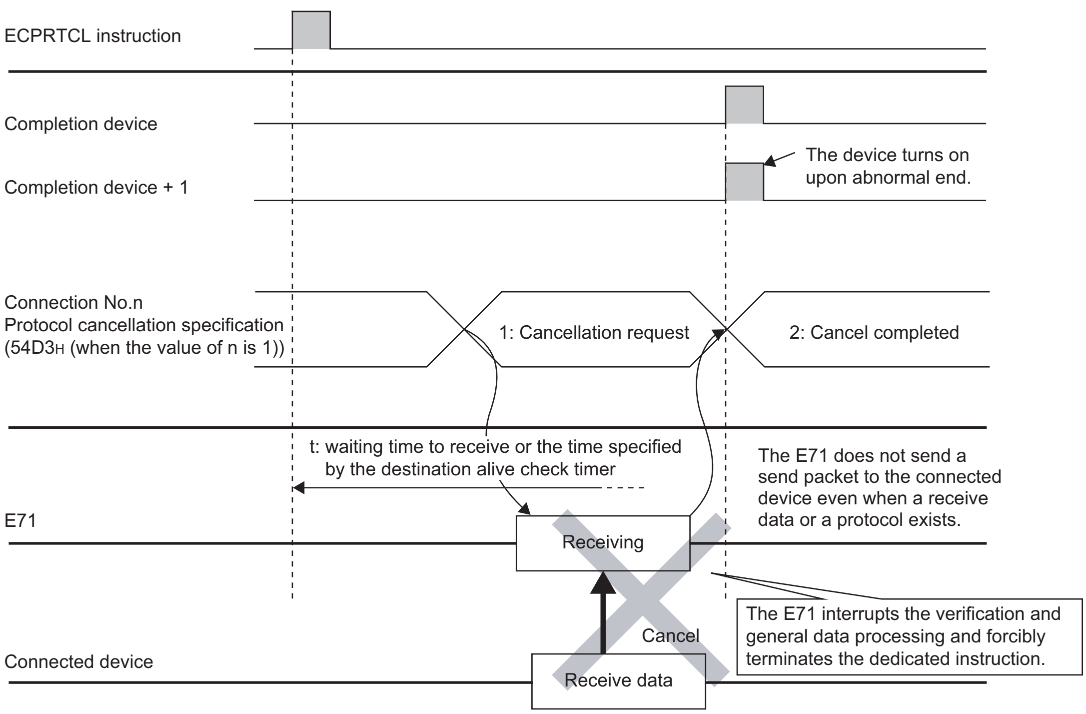
- When the cancel request is performed at sending completion (when "2" (Sending) is set in Protocol execution status (address: $54C0_H$) and the sending has been completed)



• When the cancel request is performed before receiving (when "3" (Waiting for data reception) is set in Protocol execution status (address: $54C0_H$))

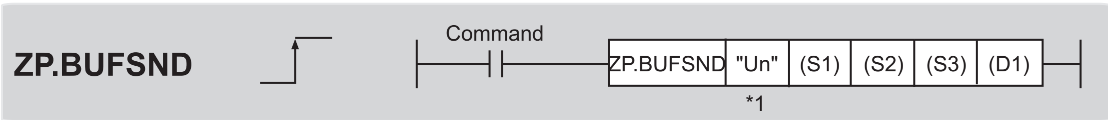


• When the cancel request is performed during receiving (when "4" (Receiving) is set in Protocol execution status (address: $54C0_H$))



15.8 ZP.BUFSND

This instruction sends data to the connected device through communications using a fixed buffer.



<table>
<tr><th rowspan="3">Setting data*2</th><th colspan="10">Available device</th></tr>
<tr><th colspan="2">Internal device (system, user)</th><th rowspan="2">File register</th><th colspan="2">Link direct device J□\□</th><th rowspan="2">Intelligent function module device U□\G□</th><th rowspan="2">Index register Zn</th><th colspan="2">Constant</th><th rowspan="2">Others</th></tr>
<tr><th>Bit</th><th>Word</th><th>Bit</th><th>Word</th><th>K, H</th><th>$</th></tr>
<tr><td>(S1)</td><td>-</td><td>○</td><td>○</td><td colspan="4">-</td><td>○</td><td>-</td><td>-</td></tr>
<tr><td>(S2)</td><td>-</td><td>○</td><td>○</td><td colspan="4">-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>(S3)</td><td>-</td><td>○</td><td>○</td><td colspan="4">-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>(D1)</td><td>○</td><td>○</td><td>○</td><td colspan="4">-</td><td>-</td><td>-</td><td>-</td></tr>
</table>

*1 If the own station is a Basic model QCPU (function version B or later), Universal model QCPU, or safety CPU, " Page 241, Section 15.5" (double quotation) of the first argument can be omitted.
*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Connection number (1 to 16)*1	User	BIN 16-bit
(S2)	Start number of the device that stores control data	System	Device name
(S3)	Start number of the device that stores send data	User	Device name
(D1)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

*1 For safety CPUs, only connection numbers 1 to 8 can be specified. If the specified value is out of range, "OPERATION ERROR" (error code: 4101) occurs.

(2) Control data

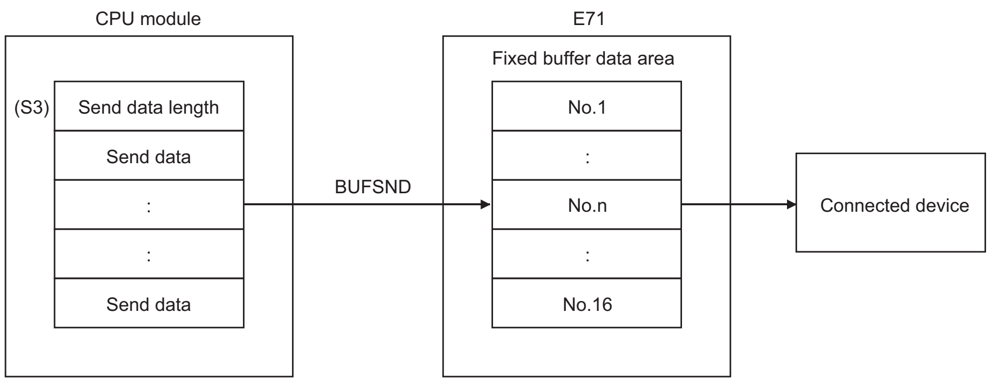
Device	Item	Setting data	Setting range	Set by
(S2)+0	System area	-	-	-
(S2)+1	Completion status	Stores the status at completion. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System

(3) Send data

Device	Item	Setting data	Setting range	Set by
(S3)+0	Send data length	Specify send data length. (Depending on the procedure of fixed buffer communications, the data length is the number of words or the number of bytes.)	-	User
		Procedure exists (communications using a binary code): Number of words	1 to 1017	
		Procedure exists (communications using an ASCII code): Number of words	1 to 508	
		No procedure (communications using a binary code): Number of bytes	1 to 2046	
(S3)+1 to (S3)+n	Send data	Specify the send data.	-	User

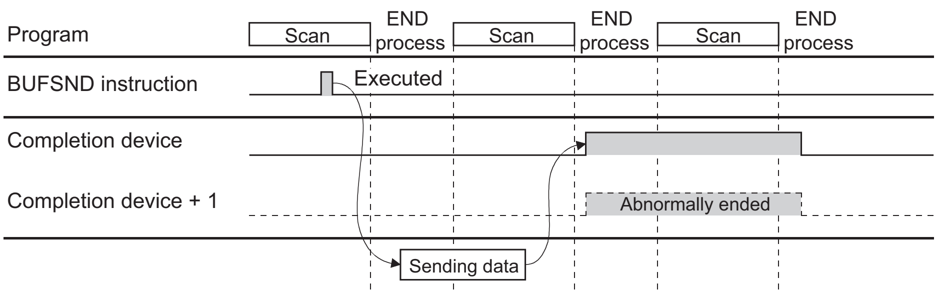
(4) Functions

- This instruction sends the data specified by (S3) to the connected device of the connection specified by (S1) for the module specified by Un.



- Completion of the BUFSND instruction can be checked with Completion device (D1)+0 and (D1)+1.
  - Completion device (D1)+0: Turns on at the END process of the scan where the BUFSND instruction is completed and turns off at the next END process.
  - Completion device (D1)+1: Turns on and off depending on the completion status of the BUFSND instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the BUFSND instruction is completed and turns off at the next END process.

[Operation while the BUFSND instruction is being executed]



- The ZP.BUFSND instruction is executed on the rising edge (OFF → ON) of the sending command.

Point

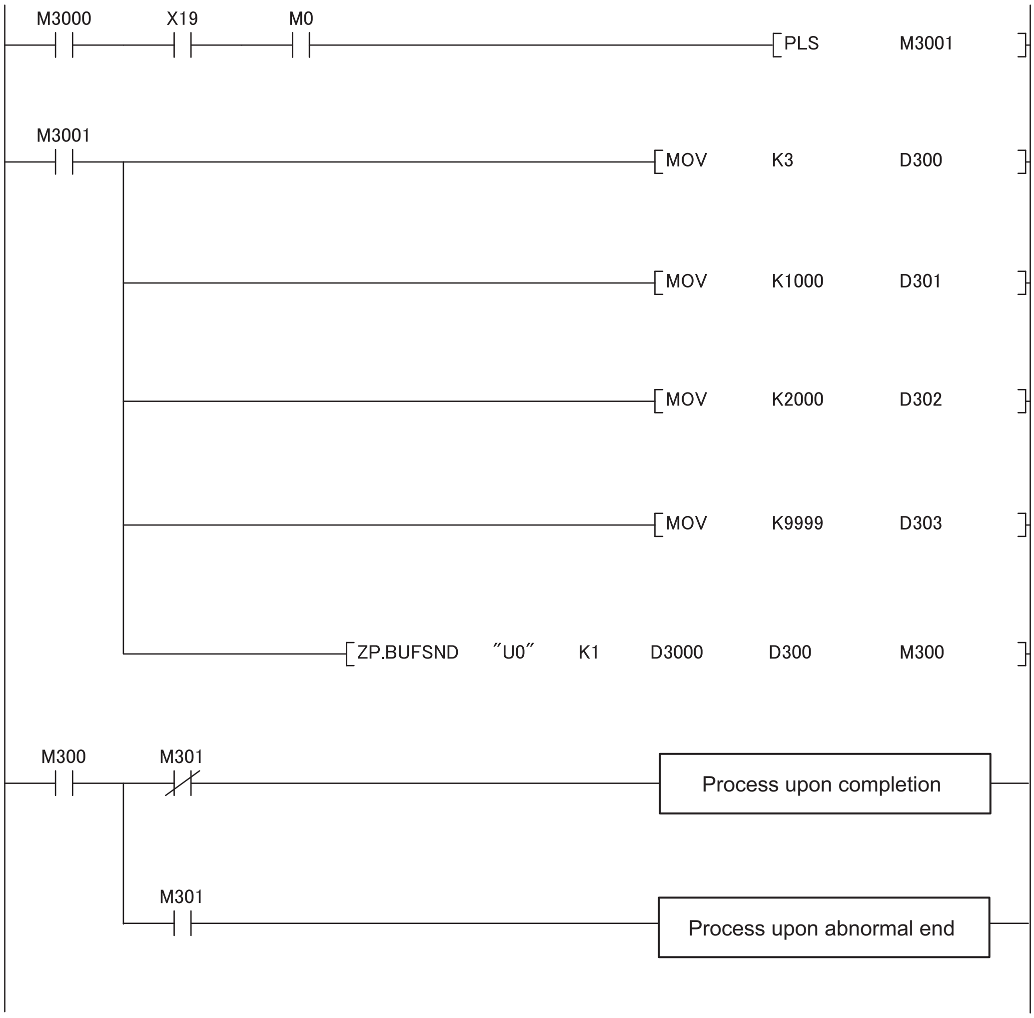
Do not perform a send process using I/O signals and using the dedicated instruction together for the same connection. Doing so will result in malfunction.

(5) Errors

- If a dedicated instruction ends with an error, Completion device (D1)+1 turns on, and the error code is stored in Completion status (S1)+1.

(6) Program example

The following shows a sample program that sends data stored on the fixed buffer of the connection number 1 (when the I/O signals of the E71 are X/Y00 to X/Y1F).



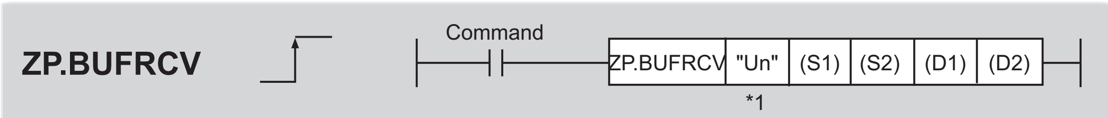
Point

For safety CPUs, data stored in the buffer memory of the intelligent function module cannot be used. The program must be modified using the I/O signals of the intelligent function module corresponding to the buffer memory.
For use with safety CPUs, refer to the following.

QSCPU User's Manual (Function Explanation, Program Fundamentals)

15.9 ZP.BUFRCV

This instruction reads data received from the connected device through communications using a fixed buffer (used in the main program).



Setting data*2	Available device									
	Internal device (system, user)		File register	Link direct device J□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	-	○	○	-				○	-	-
(S2)	-	○	○	-				-	-	-
(D1)	-	○	○	-				-	-	-
(D2)	○	○	○	-				-	-	-

*1 If the own station is a Basic model QCPU (function version B or later), Universal model QCPU, or safety CPU, " " (double quotation) of the first argument can be omitted.

*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE$_H$: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Connection number (1 to 16)*1	User	BIN 16-bit
(S2)	Start number of the device that stores control data	System	Device name
(D1)	Start number of the device that stores receive data	System	Device name
(D2)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

*1 For safety CPUs, only connection numbers 1 to 8 can be specified. If the specified value is out of range, "OPERATION ERROR" (error code: 4101) occurs.

(2) Control data

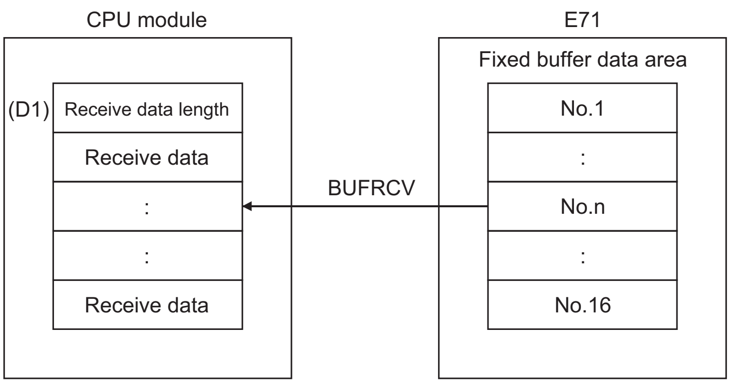
Device	Item	Setting data	Setting range	Set by
(S2)+0	System area	-	-	-
(S2)+1	Completion status	Stores the status at completion. 0000$_H$: Normal completion Values other than 0000$_H$ (error code): Abnormal end	-	System

(3) Receive data

Device	Item	Setting data	Setting range	Set by
(D1)+0	Receive data length	Stores the data length of the data read from the fixed buffer data area. (Depending on the procedure of fixed buffer communications, the data length is the number of words or the number of bytes.)	-	System
		Procedure exists (communications using a binary code): Number of words	1 to 1017	
		Procedure exists (communications using an ASCII code): Number of words	1 to 508	
		No procedure (communications using a binary code): Number of bytes	1 to 2046	
(D1)+1 to (D2)+n	Receive data	Stores the data read from the fixed buffer data area starting from the smallest address.	-	System

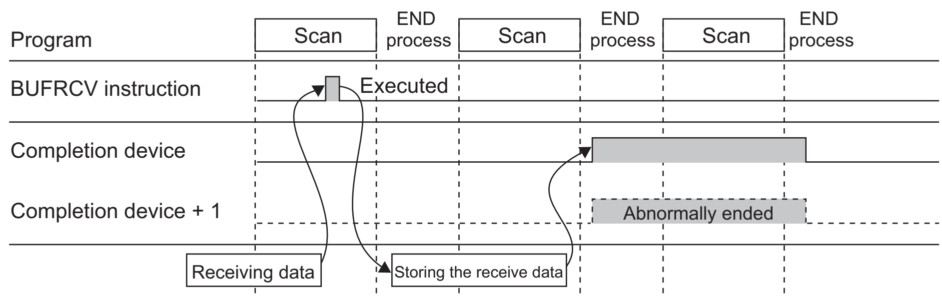
(4) Functions

- This instruction reads the receive data (for fixed buffer communications) of the connection specified by (S1) for the module specified by Un.



- Completion of the BUFRCV instruction can be checked with Completion device (D2)+0 and (D2)+1.
  - Completion device (D2)+0: Turns on at the END process of the scan where the BUFRCV instruction is completed and turns off at the next END process.
  - Completion device (D2)+1: Turns on and off depending on the completion status of the BUFRCV instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the BUFRCV instruction is completed and turns off at the next END process.

[Operation while the BUFRCV instruction is being executed]



- The ZP.BUFRCV instruction is executed on the rising edge (OFF → ON) of the read command (bit for a connection with the connected device set in Fixed buffer reception status signal (address: 5005_H)).

Point

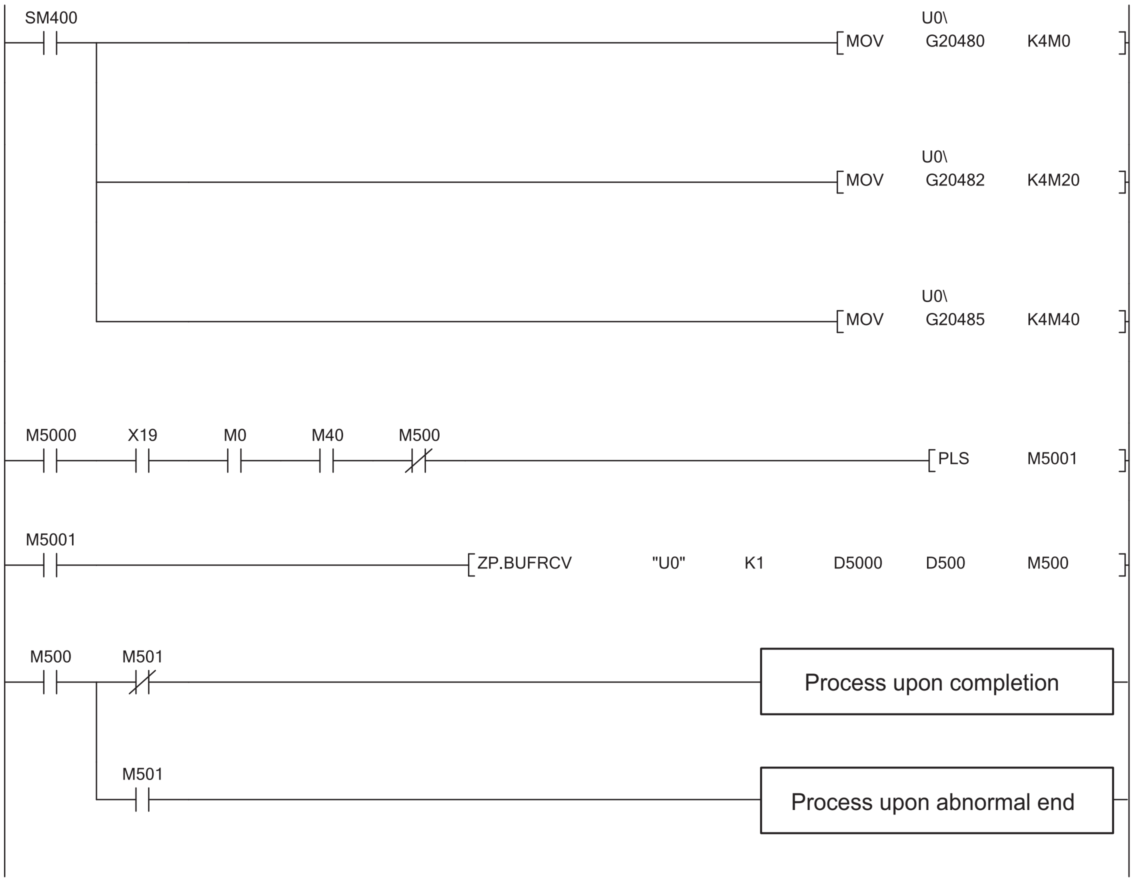
- Do not perform a receive process using I/O signals and using the dedicated instruction together for the same connection. Doing so will result in malfunction.
- When receive data is read from the same connection, the BUFRCVS instruction (for an interrupt program) cannot be used together.

(5) Errors

- If a dedicated instruction ends with an error, Completion device (D2)+1 turns on, and the error code is stored in Completion status (S2)+1.

(6) Program example

The following shows a sample program that reads the receive data from the fixed buffer of connection number 1 (when the I/O signals of the E71 are X/Y00 to X/Y1F).

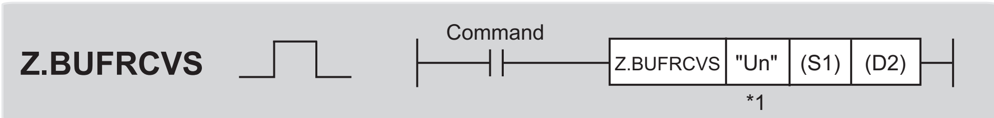


Point

For safety CPUs, data stored in the buffer memory of the intelligent function module cannot be used. The program must be modified using the I/O signals of the intelligent function module corresponding to the buffer memory.
For use with safety CPUs, refer to the following.
QSCPU User's Manual (Function Explanation, Program Fundamentals)

15.10 Z.BUFRCVS

This instruction reads data received from the connected device through communications using a fixed buffer (used in an interrupt program).



Setting data*2	Available device									
	Internal device (system, user)		File register	Link direct device J□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	-	○	○	-				○	-	-
(D1)	-	○	○	-				-	-	-

*1 If the own station is a Basic model QCPU (function version B or later) or Universal model QCPU, " " (double quotation) of the first argument can be omitted.

*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Connection number (1 to 16)		BIN 16-bit
(D1)	Start number of the device that stores receive data	System	Device name

(2) Receive data

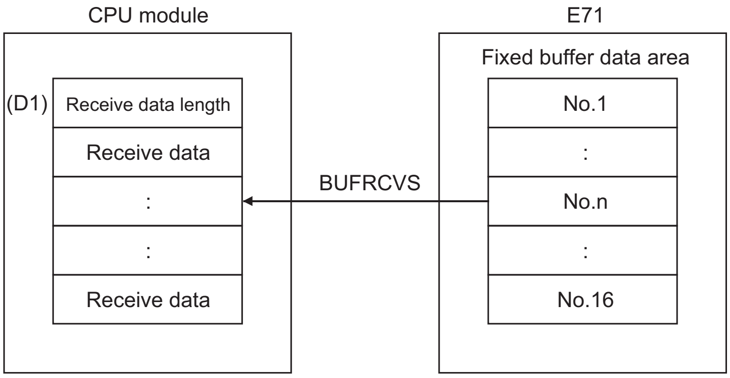
Device	Item	Setting data		Setting range	Set by
(D1)+0	Receive data length	Stores the data length of the data read from the fixed buffer data area. (Depending on the procedure of fixed buffer communications, the data length is the number of words or the number of bytes.)		-	System
			Procedure exists (communications using a binary code): Number of words	1 to 1017	
			Procedure exists (communications using an ASCII code): Number of words	1 to 508	
			No procedure (communications using a binary code): Number of bytes	1 to 2046	
(D1)+1 to (D1)+n	Receive data	Stores the data read from the fixed buffer data area starting from the smallest address.		-	System

15

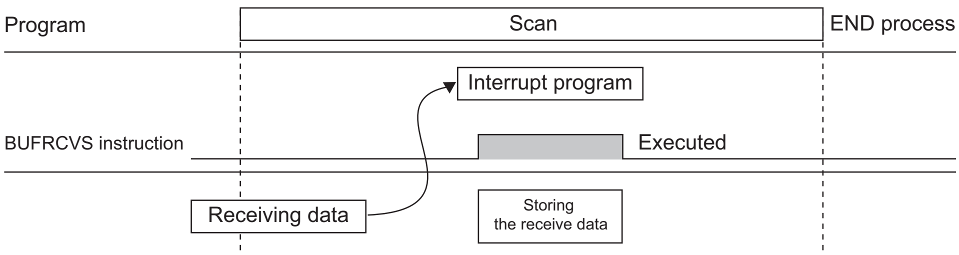
15.10 Z.BUFRCVS

(3) Functions

- This instruction reads the receive data (for fixed buffer communications) of the connection specified by (S1) for the module specified by Un.



[Operation while the BUFRCVS instruction is being executed]



- The Z.BUFRCVS instruction is executed in an interrupt program, and the process is completed within one scan.

Point

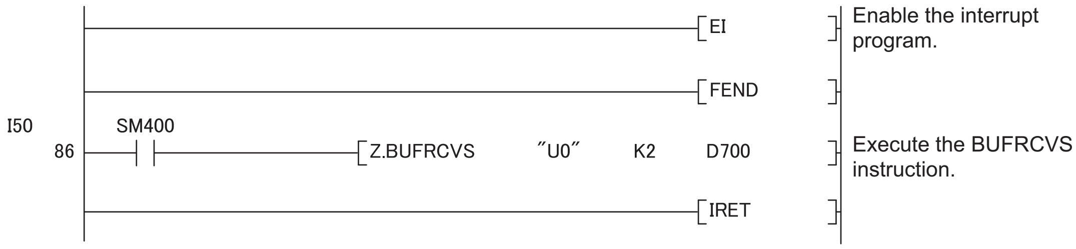
- To read receive data using an interrupt program, configure both the interrupt settings and interrupt pointer setting parameters of the programming tool. (Page 150, Section 12.5.1)
- When receive data is read from the same connection, the BUFRCV instruction (for the main program) cannot be used together.

(4) Errors

- If a dedicated instruction ends with an error, Diagnostic errors (SM0) turns on, and the error code is stored in SD0.

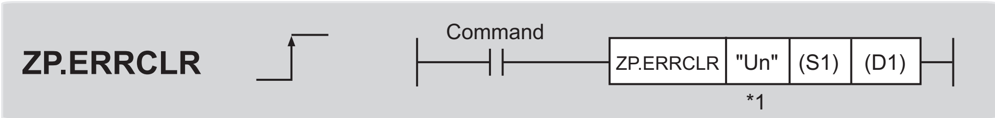
(5) Program example

The following shows a sample program that reads the receive data from the fixed buffer of connection number 2 (when the I/O signals of the E71 are X/Y00 to X/Y1F).



15.11 ZP.ERRCLR

This instruction turns off LEDs of the E71 or clears error information stored in the buffer memory.



Setting data*2	Available device									
	Internal device (system, user)		File register	Link direct device J□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	-	○	○	-				-	-	-
(D1)	○	○	○	-				-	-	-

*1 If the own station is a Basic model QCPU (function version B or later) or Universal model QCPU, " " (double quotation) of the first argument can be omitted.

*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Start number of the device that stores control data	User, system	Device name
(D1)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

(2) Control data

Device	Item	Setting data	Setting range	Set by
(S1)+0	System area	-	-	-
(S1)+1	Completion status	Stores the status at completion. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System
(S1)+2	Clear target specification	Specify error information to be cleared. 0000_H: Initial error code 0001_H to 0010_H: Open error code for a connection with the connected device 0100_H: Error log block area 0101_H: Communication status - status for each protocol 0102_H: Communication status - e-mail receiving status 0103_H: Communication status - e-mail sending status $FFFF_H$: Clears all of the above	As described in the left	User
(S1)+3	Clear function specification	Specify the function to be cleared. 0000_H: COM.ERR LED off, error code clear $FFFF_H$: Error log clear	0000_H, $FFFF_H$	User
(S1)+4 to (S1)+7	System area	-	-	-

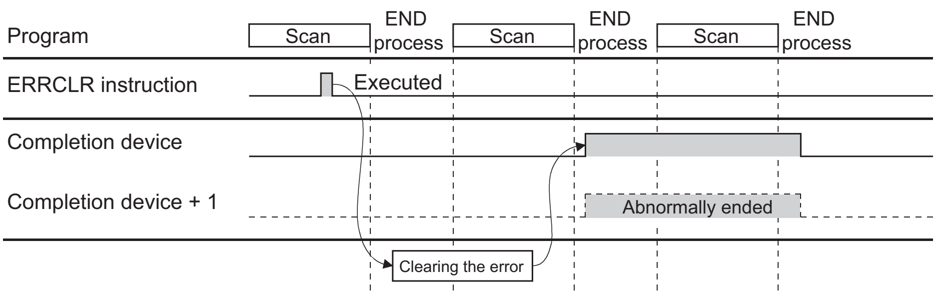
(3) Functions

- This instruction turns off the COM.ERR. LED and clears error information listed below for the module specified by Un.

Item		Target specification (S1)+2	Function specification (S1)+3	Error information to be cleared (buffer memory)
Initial error		0000_H	0000_H	• Initial error code (address: 69_H) • COM.ERR. LED off
Open error		0001_H to 0010_H	0000_H	• Open error code for a connection with the connected device (address: $7C_H$, 86_H...)
Error log		0100_H	$FFFF_H$	Error log (address: $E3_H$ to 174_H)
Communication status	Status for each protocol	0101_H	$FFFF_H$	Clears communication status. (address: 178_H to $1FF_H$)
	E-mail receiving status	0102_H	$FFFF_H$	E-mail receiving (address: 5871_H to $5B38_H$)
	E-mail sending status	0103_H	$FFFF_H$	E-mail sending (address: $5B39_H$ to $5CA0_H$)
All		$FFFF_H$	$FFFF_H$	Clears all of the above.

- Completion of the ERRCLR instruction can be checked with Completion device (D1)+0 and (D1)+1.
  - Completion device (D1)+0: Turns on at the END process of the scan where the ERRCLR instruction is completed and turns off at the next END process.
  - Completion device (D1)+1: Turns on and off depending on the completion status of the ERRCLR instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the ERRCLR instruction is completed and turns off at the next END process.

[Operation while the ERRCLR instruction is being executed]



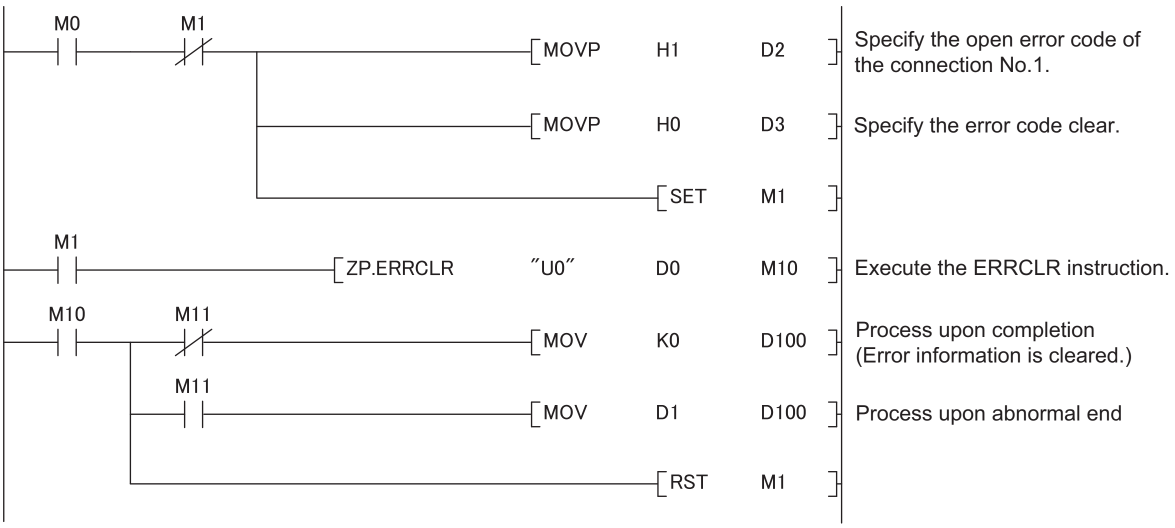
- The ZP.ERRCLR instruction is executed on the rising edge (OFF → ON) of the clear command.

(4) Errors

- If a dedicated instruction ends with an error, Completion device (D1)+1 turns on, and the error code is stored in Completion status (S1)+1.

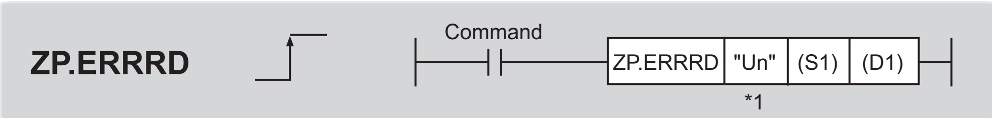
(5) Program example

The following shows a sample program that clears the open error code of the connection number 1 (when the I/O signals of the E71 are X/Y00 to X/Y1F).



15.12 ZP.ERRRD

This instruction reads error information stored in the buffer memory of the E71.



Setting data*2	Available device									
	Internal device (system, user)		File register	Link direct device J□\□		Intelligent function module device U□\G□	Index register Zn	Constant		Others
	Bit	Word		Bit	Word			K, H	$	
(S1)	-	○	○	-				-	-	-
(D1)	○	○	○	-				-	-	-

*1 If the own station is a Basic model QCPU (function version B or later) or Universal model QCPU, " " (double quotation) of the first argument can be omitted.

*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Start number of the device that stores control data	User, system	Device name
(D1)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

(2) Control data

Device	Item	Setting data	Setting range	Set by
(S1)+0	System area	-	-	-
(S1)+1	Completion status	Stores the status at completion. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System
(S1)+2	Read information specification	Specify error information to be read. 0000_H: Initial error code 0001_H to 0010_H: Open error code for a connection with the connected device	0000_H, 0001_H to 0010_H	User
(S1)+3	Read target information specification	Specify whether to read the latest error information. 0000_H: Reads the latest error information.	0000_H	User
(S1)+4	Error information	Stores the error information that has been read. 0000_H: No error Values other than 0000_H (error code): Abnormal end	-	System
(S1)+5 to (S1)+7	System area	-	-	-

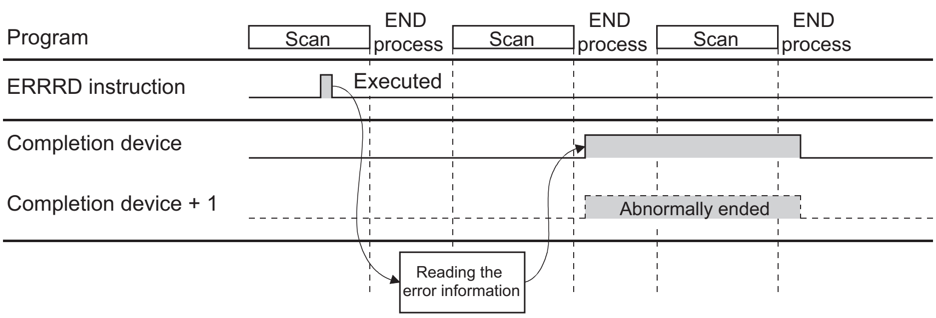
(3) Functions

- This instruction reads the error information of the module specified by Un.

Item	Target specification (S1)+2	Function specification (S1)+3	Error information to be read (buffer memory)
Initial error	0000_H	0000_H	Initial error code (address: 69_H)
Open error	0001_H to 0010_H	0000_H	Open error code for a connection with the connected device (address: $7C_H$, 86_H…)

- Completion of the ERRRD instruction can be checked with Completion device (D1)+0 and (D1)+1.
  - Completion device (D1)+0: Turns on at the END process of the scan where the ERRRD instruction is completed and turns off at the next END process.
  - Completion device (D1)+1: Turns on and off depending on the completion status of the ERRRD instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the ERRRD instruction is completed and turns off at the next END process.

[Operation while the ERRRD instruction is being executed]



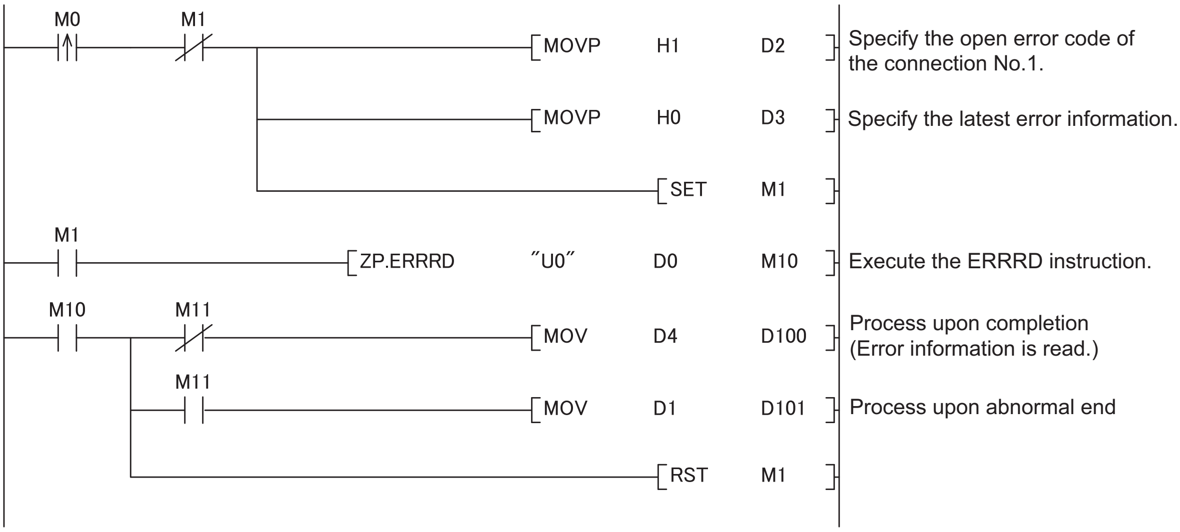
- The ZP.ERRRD instruction is executed on the rising edge (OFF → ON) of the read command.

(4) Errors

- If a dedicated instruction ends with an error, Completion device (D1)+1 turns on, and the error code is stored in Completion status (S1)+1.

(5) Program example

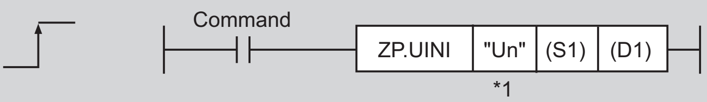
The following shows a sample program that reads the open error code of the connection number 1 (when the I/O signals of the E71 are X/Y00 to X/Y1F).



15.13 ZP.UINI

This instruction changes the setting, such as the Ethernet operation setting, and re-initializes the E71.

ZP.UINI



<table>
<tr><th rowspan="3">Setting data*2</th><th colspan="10">Available device</th></tr>
<tr><th colspan="2">Internal device (system, user)</th><th rowspan="2">File register</th><th colspan="2">Link direct device J□\□</th><th rowspan="2">Intelligent function module device U□\G□</th><th rowspan="2">Index register Zn</th><th colspan="2">Constant</th><th rowspan="2">Others</th></tr>
<tr><th>Bit</th><th>Word</th><th>Bit</th><th>Word</th><th>K, H</th><th>$</th></tr>
<tr><td>(S1)</td><td>-</td><td>○</td><td>○</td><td colspan="4">-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>(D1)</td><td>○</td><td>○</td><td>○</td><td colspan="4">-</td><td>-</td><td>-</td><td>-</td></tr>
</table>

*1 If the own station is a Basic model QCPU (function version B or later) or Universal model QCPU, " " (double quotation) of the first argument can be omitted.
*2 The file registers set for each local device and program cannot be used.

(1) Setting data

Setting data	Description	Set by	Data type
"Un"/Un	Start I/O number of the E71 (00 to FE_H: The first two digits of the three-digit I/O number)	User	Character string/BIN 16-bit
(S1)	Start number of the device that stores control data	User, system	Device name
(D1)	The start number of the bit device in the own station that turns on for one scan upon completion of the instruction. (D1)+1 also turns on if the instruction ends abnormally.	System	Bit

(2) Control data

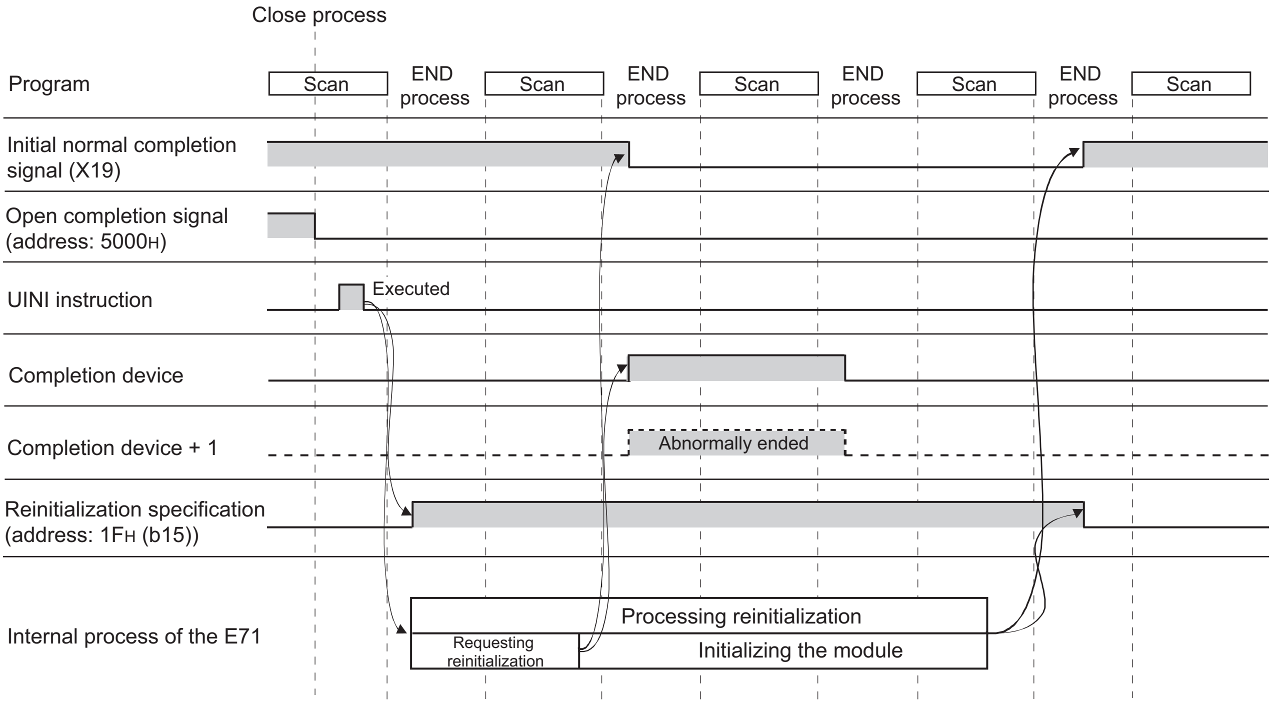
Device	Item	Setting data	Setting range	Set by
(S1)+0	System area	-	-	-
(S1)+1	Completion status	Stores the status at completion. 0000_H: Normal completion Values other than 0000_H (error code): Abnormal end	-	System
(S1)+2	Specification of changed item	Specify 0000_H to update the address information on the connected devices retained by the E71. Specify the target for setting change among the own station IP address, Ethernet operation setting, transmission speed, and communication mode. Do not specify the transmission speed and communication mode together with the own station IP address and Ethernet operation setting. If doing so, only the specification of the own station IP address and Ethernet operation setting is updated. • Specification of own station IP address change (b0): Specify whether to change the own station IP address. (Set the address in (S1)+3 and (S1)+4.) 0: Do not change 1: Change • Specification of the Ethernet operation setting change (b1): Specify whether to change the Ethernet operation setting. (Set the parameters in (S1)+5.) 0: Do not change 1: Change • Specification of transmission speed and communication mode changes (b12 to b15): Specify the transmission speed and the communicate mode. 0: Do not change 1: Auto negotiation 2: 100Mbps/full-duplex 3: 100Mbps/half-duplex 4: 10Mbps/full-duplex 5: 10Mbps/half-duplex	0000_H to 5000_H	User
(S1)+3 (S1)+4	Own station IP address	Specify the IP address of the own station.	00000001_H to $FFFFFFFE_H$	User
(S1)+5	Ethernet operation setting	Specify the Ethernet operation setting. • Communication data code setting (b1) 0: Binary Code 1: ASCII Code • TCP existence confirmation setting (b4) 0: Use the Ping 1: Use the KeepAlive • Send frame setting (b5) 0: Ethernet 1: IEEE 802.3 • Setting of write enable/disable at RUN time (b6) 0: Disable 1: Enable • Initial timing setting (b8) 0: Do not wait for OPEN (Communications impossible at STOP time) 1: Always wait for OPEN (Communications possible at STOP time)	As described in the left	User

*1 The E71 enables data exchange to be resumed by clearing the address information of the connected device retained in the E71 and performing a reinitialization process. (Initial normal completion signal (X19) turns on.)

(3) Functions

- This instruction re-initializes the module specified by Un.
- Completion of the UINI instruction can be checked with Completion device (D1)+0 and (D1)+1.
  - Completion device (D1)+0: Turns on at the END process of the scan where the UINI instruction is completed and turns off at the next END process.
  - Completion device (D1)+1: Turns on and off depending on the completion status of the UINI instruction.
    - Normal completion: Stays off and does not change.
    - Abnormal end: Turns on at the END process of the scan where the UINI instruction is completed and turns off at the next END process.

[Operation while the UINI instruction is being executed]



- The ZP.UINI instruction is executed on the rising edge (OFF → ON) of the reinitialization command.

Point

To re-initialize the E71, note the following.

- Terminate all data communications with connected devices, check that all connections are closed, then re-initialize the E71.
- Do not perform a reinitialization process using both direct writing to the buffer memory and the UINI instruction. Also, do not request another reinitialization process while reinitialization is being performed.
- When the IP address of the E71 has been changed, reset that of the connected device as well. (If the connected device retains the MAC address of the destination device, the communications may not be continued after the IP address of the E71 has been changed.)
- In a redundant system, do not change parameters such as IP address and Ethernet operation setting using the UINI instruction. If changed, normal communications cannot be performed. Change the parameters using a programming tool.

(4) Errors

- If a dedicated instruction ends with an error, Completion device (D1)+1 turns on, and the error code is stored in Completion status (S1)+1.

(5) Program example

The following pages show sample programs that change the following setting items using the UINI instruction.

- Ethernet operation setting
- Transmission speed and communication mode

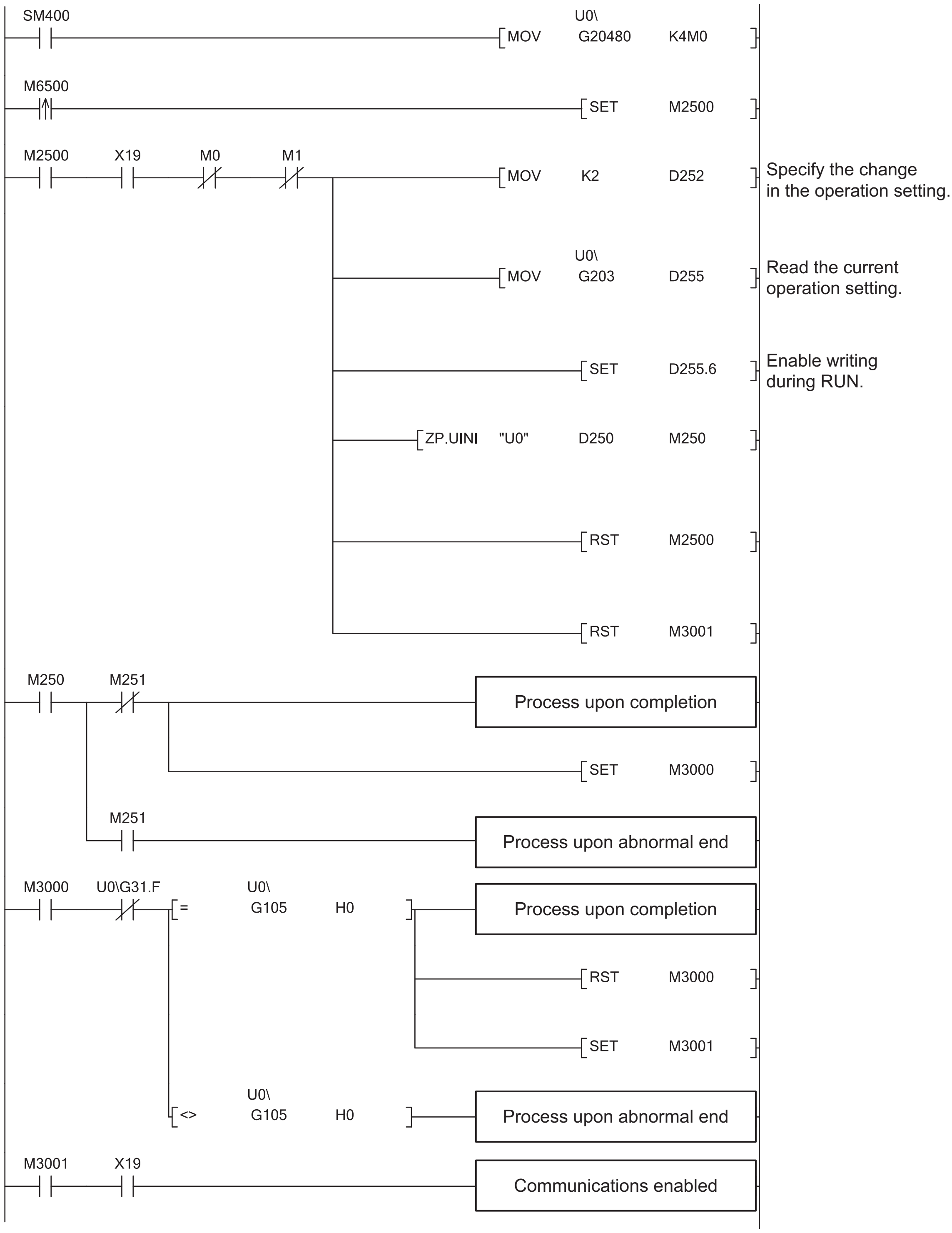
Remark

The following are sample programs for communications using connection numbers 1 and 2. When using other connections, specify the corresponding signals and bits for each connection.

The UINI instruction is also used in the program for a reinitialization process. When performing a reinitialization process using the UINI instruction, refer to the program for a reinitialization process. (Page 358, Appendix 4.2)

(a) Changing the Ethernet operation setting

The following shows a sample program that changes the Ethernet operation setting (enables writing during RUN) (when the I/O signals of the E71 are X/Y00 to X/Y1F).



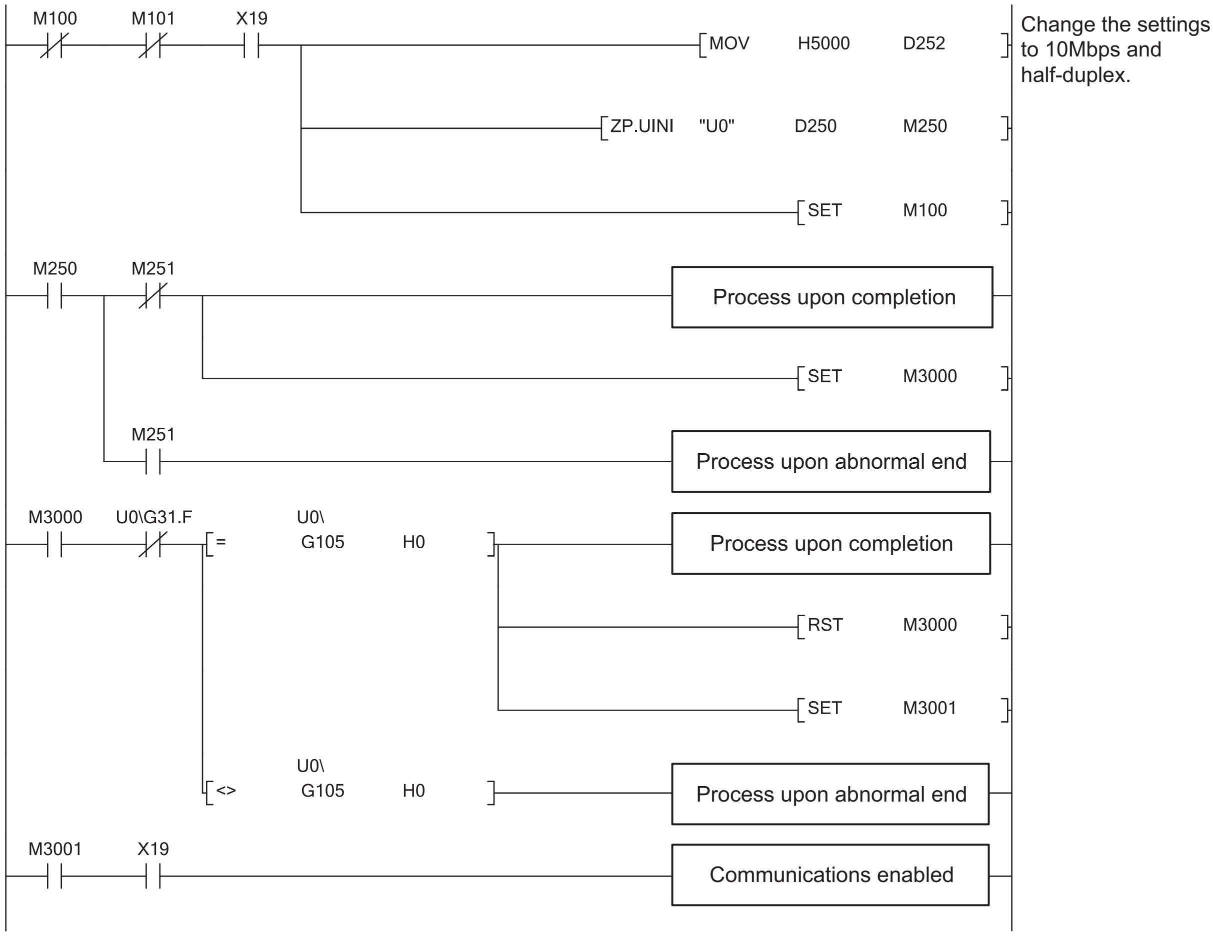
Point

For safety CPUs, data stored in the buffer memory of the intelligent function module cannot be used. The program must be modified using the I/O signals of the intelligent function module corresponding to the buffer memory. For use with safety CPUs, refer to the following.

QSCPU User's Manual (Function Explanation, Program Fundamentals)

(b) Changing the transmission speed and communication mode

The following shows a sample program that changes the transmission speed to 10Mbps and the communication mode to half-duplex (when the I/O signals of the E71 are X/Y00 to X/Y1F).



CHAPTER 16 TROUBLESHOOTING

This chapter describes how to identify causes of problems that occur when an E71 is used and take corrective action.

16.1 Before Troubleshooting

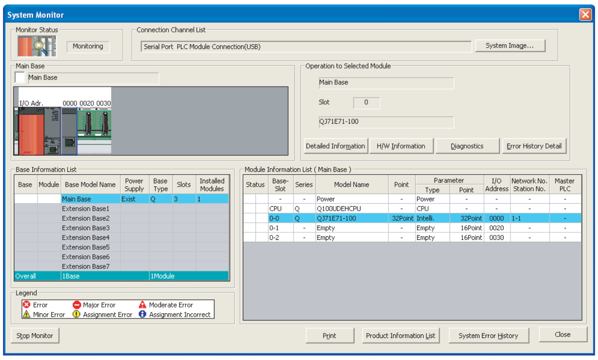
Check that the POWER LED of the power supply module and the MODE LED of the CPU module are on. If either or both are off, troubleshoot the CPU module.

User's Manual (hardware design, maintenance and inspection) for the CPU module used

16.2 Troubleshooting Procedure

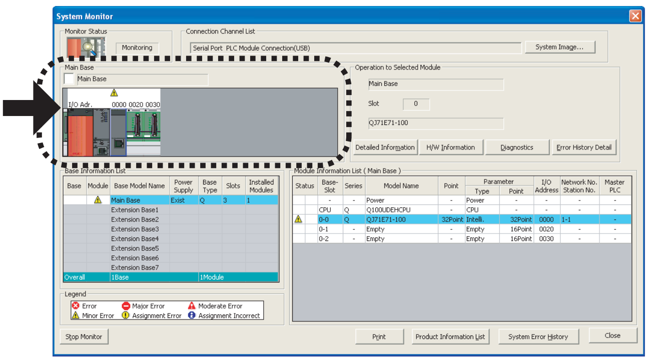
How to identify the cause of a problem and take corrective action is explained below. Use a programming tool to identify the cause and take corrective action.

(1) Procedure

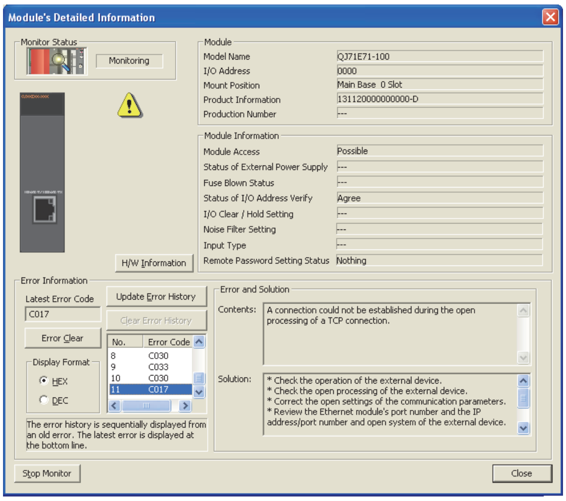


1. **Connect the CPU module to a programming tool, and open the "System Monitor" window.**

[Diagnostics] ⇨ [System Monitor]



2. **When an error in the E71 is reported, select the E71 and click the Detailed Information button.**
When an error in a module other than the E71 is reported, refer to the manual for that module.



3. **The "Module's Detailed Information" window is displayed. Clicking the** Update Error History **button displays the error details and corrective action.**

4. **If the error details cannot be obtained in the step above, perform the troubleshooting below.**

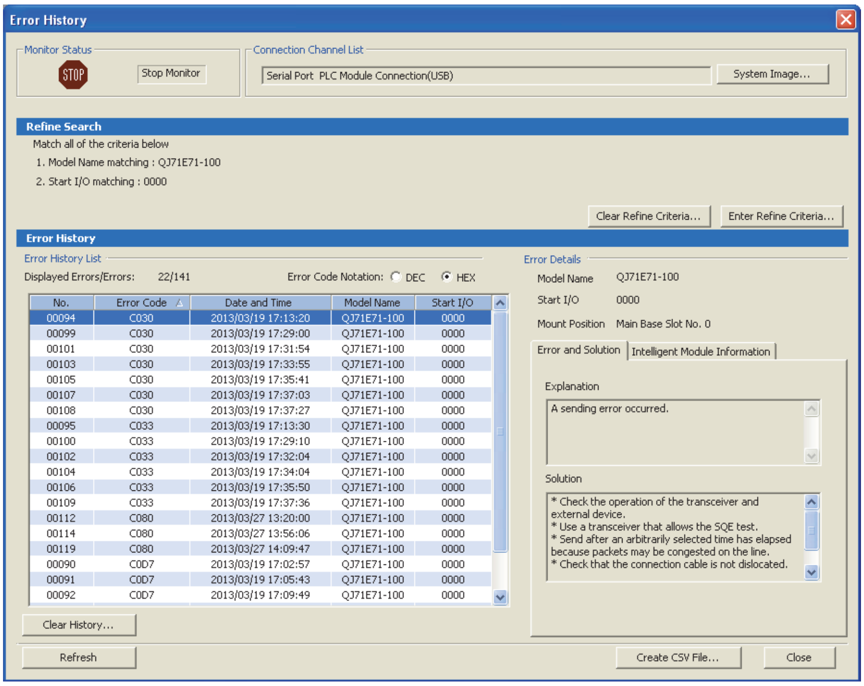
- Checking the LEDs
  (☞Page 286, Section 16.4)
- Troubleshooting by symptom
  (☞Page 288, Section 16.5)

16.3 Checking with the Module Error Collection Function

With the module error collection function, the errors that have occurred in the E71 are saved into the CPU module. This allows error details to be held even if the power is turned off or the CPU module is reset. Check the errors collected by the CPU module on the "Error History" window.

This function is available only in the QJ71E71-100 with the serial number (first five digits) of "15042" or later.

[Diagnostics] ⇨ [System Monitor...] ⇨ Error History Detail button



16.4 Checking the LEDs

The following describes how to troubleshoot the problem by checking the LEDs.

Point

The on/off status of the INIT. LED, OPEN LED, ERR. LED, and COM.ERR. LED is stored in Area for module status (address: $C8_H$) of the buffer memory. (Page 38, Section 3.5.2)

16.4.1 If the RUN LED turns off

The following table lists the items to be checked and the actions to be taken if the RUN LED turns off after the E71 is powered on.

Check item	Action
Is it a watchdog timer error?	Reset the CPU module and check that the RUN LED turns on. If the RUN LED still does not turn on, the E71 may be faulty. Please consult your local Mitsubishi representative.
Is the E71 mounted properly?	Securely mount the E71 on the base unit.
Is the power capacity of the power supply module sufficient?	Check if the power supply module has enough power capacity.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.4.2 If the ERR. LED or COM.ERR. LED turns on

Connect the CPU module connected to the E71 with the ERR. LED or COM.ERR. LED on to a programming tool to identify the cause. (Page 283, Section 16.2)
If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

The COM.ERR. LED does not turn off even after the error cause is removed. To turn it off, refer to "How to Turn Off the COM.ERR. LED". (Page 340, Section 16.8)

16.4.3 If the SD LED does not flash when data is sent

The following table lists the items to be checked and the actions to be taken if the SD LED does not flash when data is sent.

Check item	Action
Is the ERR. LED or COM.ERR. LED on?	Remove the cause that turned on the ERR. LED or COM.ERR. LED.
Are the cables properly connected?	Check the cable connection. In addition, perform a line test to check if there is a problem with the cable connection and the Ethernet line. (Page 364, Appendix 5)
Is the program correct?	Correct the send program of the E71.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.4.4 If data cannot be received with the RD LED off

The following table lists the items to be checked and the actions to be taken when an E71 cannot receive data with the RD LED off.

Check item	Action
Is the ERR. LED or COM.ERR. LED on?	Remove the cause that turned on the ERR. LED or COM.ERR. LED.
Are the cables properly connected?	Check the cable connection. In addition, perform a line test to check if there is a problem with the cable connection and the Ethernet line. (Page 364, Appendix 5)
Are the parameter settings correct?	Correct the IP address, router setting, and subnet mask setting of the own station.
Is the program correct?	Correct the send program of the connected device.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5 Troubleshooting by Symptom

The following describes how to troubleshoot for each phenomenon. When an error occurs in the E71, identify the cause with the programming tool. (Page 283, Section 16.2)

16.5.1 Communications cannot be performed with the connected device.

The following table lists how to troubleshoot the problem when the E71 cannot communicate with the connected device.

Check item	Action
Is the RUN LED of the E71 on?	Reset the CPU module. If resetting the CPU module does not turn on the RUN LED on the E71, the hardware of the E71 may be faulty. Replace the E71 and restart all the connected devices that were performing communications with the own station.*1
Does the connected device conform to the Ethernet standard?	Replace the device with one conforming to the Ethernet standard. (Page 63, CHAPTER 5)
Is the cable securely connected?	• Securely lock the cable. • Check the wiring. (Page 77, Section 6.2)
Is the network parameter mode set to "Online"?	Set the network parameter mode to "Online". (Page 84, Section 7.1.2)
Does the communication data code (ASCII or binary) setting match with that of the connected device?	If the data code settings do not match, data cannot be sent or received because the connected device cannot normally decode commands.*2 Set the same data code as that of the connected device in "Communication Data Code" of the Ethernet operation setting. (Page 85, Section 7.1.3)
Has the initial process been completed normally? (Is Initial normal completion signal (X19) on?)	• If Initial normal completion signal (X19) is off, perform the initial process. (Page 364, Appendix 5) • If the initial setting of the programming tool is same as that of the initial process program, delete the initial process program.
Was a device on the line (such as an E71, connected device, hub, and router) replaced by the one with the same IP address?	Restart all the devices on the line.*1
When the E71 is connected to 10BASE2, are the LG and FG terminals of the power supply module grounded?	Power off the E71-mounted station and ground the LG and FG terminals of the power supply module. (User's manual (hardware design, maintenance and inspection) for the CPU module used)*3 After grounding the terminals, start the E71 to perform communications with the connected device again.
Is the access from the connected device set to be denied by the IP filter function?	If the access from the connected device is denied by the IP filter function, correct the values in IP filter settings (address: 5700_H to 5721_H) and perform IP filter setting by the reinitialization process again. (Page 195, Section 14.3)

*1 Devices on an Ethernet network contain the IP to Mac address mapping, called an "ARP cache". If a device on a line is replaced by another device with the same IP address, the MAC address contained in the "ARP cache" and the MAC address of the replaced device do not match; therefore, communications may not be performed normally. The "ARP cache" is updated when the device is reset or after a certain period of time. The time varies depending on the devices.

*2 If the communication data code setting is different between the E71 and the connected device, error codes that are not found in the error code list may be returned to the connected device. When receiving data with different data codes, the E71 cannot decode commands normally. The E71 returns an error response according to the communication data code setting. (Page 156, Section 12.6.2 (1) (e), Remark)

*3 If the LG and FG terminals of the power supply module of the E71-mounted station are not connected, the communication line may be closed (disconnected) due to noise; therefore, communications with the connected device may not be performed.

If the action above does not solve the problem, check for an error in each communication process and take corrective action. (Page 290, Section 16.5.4 to Page 299, Section 16.5.16)

16.5.2 The E71 frequently fails to receive a message sent from the connected device.

The following table lists how to troubleshoot the problem when the E71 frequently fails to receive a message sent from the connected device.

Check item	Action
Are there many detected errors in Simultaneous transmission error detection count (address: $18E_H$ and $18F_H$)?	There may be a heavy load on the Ethernet line due to data sending and receiving between connected devices. • Taking action such as separating networks or decreasing the number of data sending reduces the load on the Ethernet line. • Consult the network administrator and reduce the load on the Ethernet line.
Has the error code $C0C7_H$ been stored in Error code/end code (address: $E5_H$) in the error log block?	
Has 1_H been stored in Receive buffer full detection signal (address: 5240_H)?*1	• Taking action such as separating networks or decreasing the number of data sending reduces the load on the Ethernet line. • When communications using a fixed buffer is used, check whether the BUFRCV instruction is executed. (Page 144, Section 12.4.1) • To receive data at shorter intervals than the scan time of the CPU module using a fixed buffer, add the normally closed contact for the completion device of the BUFRCV instruction to the execution conditions of the BUFRCV instruction. (Page 173, Section 12.9.3 (2) (b))
Has the number in Received TCP packet count (address: $1B8_H$ and $1B9_H$) increased?	If the packets cannot be received even though the number of received TCP packets has increased, set the value of TCP Maximum Segment Transmission setting area (address: $1E_H$) to 8000_H. Then perform the initial process again. (Page 354, Appendix 4)

*1 Available only for the QJ71E71-100.
Availability depends on the QJ71E71-100 version. (Page 351, Appendix 3)

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.3 A dedicated instruction is not completed.

The following table lists how to troubleshoot the problem when a dedicated instruction is not completed.

Check item	Action
Is the network parameter mode set to "Online"?	Set the network parameter mode to "Online". (Page 84, Section 7.1.2) (If a dedicated instruction is executed in the "Offline" mode, no error occurs but the instruction is not completed.)

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.4 MC protocol communications cannot be performed.

The following table lists how to troubleshoot the problem when the E71 cannot perform MC protocol communications.

Check item	Action
Has the connection with the connected device been opened? (Check the corresponding bit of 5000_H in the buffer memory.)*1	• Open the connection with the connected device. • Check if the open/close process for I/O signals and the OPEN/CLOSE instructions are not used together for the same connection. If so, correct the program.
Did the connected device send a command?	Send a command to the E71.
Was a response returned to the device that had sent the command?	• Check if the correct IP address was specified in the command. If not, correct the IP address and send the command again. • Check if the communication protocol (TCP/IP or UDP/IP) matches with that of the connected device. If not, correct it.
Does the communication protocol (TCP/IP or UDP/IP) match with that of the connected device?	Use the same communication protocol (TCP/IP or UDP/IP) as that of the connected device.
Does the communication data code (ASCII or binary) setting match with that of the connected device?	If the data code settings do not match, data cannot be sent or received because the connected device cannot normally decode commands.*2 Use the same setting of "Communication Data Code" in the Ethernet operation setting as that of the connected device. (Page 85, Section 7.1.3)
Is the end code of the response 0?	Check the end and error codes to correct the error.
Is the correct command format used for the command type, device, address, and others?	Correct the command format.
Is the write operation during RUN enabled?	Check the checkbox next to "Enable Write at RUN time" in the Ethernet operation setting.
Has an open or initial error occurred in the error log area?	Check and correct the error.

*1 If the connection of only the connected device is closed due to cable disconnection, personal computer restart, or other reasons, reopen the connection using the same port used before the error occurred. The E71 does not close a connection if it receives an Active open request again from the connected device with a different IP address or a port number.

*2 If the communication data code setting is different between the E71 and the connected device, error codes that are not found in the error code list may be returned to the connected device. When receiving data with different data codes, the E71 cannot decode commands normally. The E71 returns an error response according to the communication data code setting. (Page 156, Section 12.6.2 (1) (e), Remarks)

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.5 Communications using SLMP cannot be performed.

If communications cannot be performed with a connected device using SLMP, refer to the troubleshooting in the following and take corrective action.

SLMP Reference Manual

16.5.6 Communications using the predefined protocol cannot be performed.

The following table lists how to troubleshoot the problem when the E71 cannot perform communications using the predefined protocol.

Check item	Action
Has the connection with the connected device been opened? (Check the corresponding bit of 5000_H in the buffer memory.)	• Open the connection with the connected device. • Check if the open/close processes for I/O signals and the OPEN/CLOSE instructions are not used together for the same connection. If so, correct the program. • Read Open error code (address: $7C_H$) in the communication status storage area. Then check the error details and correct the error.
Is the IP address setting of the connected device correct?	Check and correct the IP address of the connected device.
Does the Protocol in the open setting match with that of the connected device (TCP/IP or UDP/IP)?	Check and correct the settings if they are unmatched.
Is "Predefined protocol" set to Fixed Buffer Communication in the open setting?	• Set "Predefined protocol" to Fixed Buffer Communication in the open setting. • Configure the setting so that the fixed buffer setting (Send or Receive) in the open setting matches the communication type of the protocol (Send Only, Receive Only, or Send&Receive)*1.
Is Predefined protocol ready (X1D) on?	Write the protocol setting data to the E71.
Was the ECPRTCL instruction executed?	Correct the execution conditions of the ECPRTCL instruction.
Was the ECPRTCL instruction completed normally?	Check the error code in the ECPRTCL instruction completion status area and correct the error.
Is the execution protocol number set to the control data of the ECPRTCL instruction?	Set the execution protocol number to the control data of the ECPRTCL instruction.
Is the communication status of the connected device normal?	Correct the error in the connected device.
Has an open or initial error occurred in the error log area?	Check and correct the error.

*1 For details on the settings, refer to Page 251, Section 15.7 (3).

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.7 The protocol setting data cannot be read or written.

The following table lists how to troubleshoot the problem when the protocol setting data cannot be read or written. Check that GX Works2 and the CPU module are connectable on the Transfer Setup of GX Works2 before troubleshooting the problem.

(1) To read the protocol setting data

Check item	Action
Are the values in Protocol setting data error information (address: 5324_H to 5327_H) and Number of registered protocols (address: 5328_H) set to 0?	Write the protocol setting data to the E71 because they have not been written.

(2) To write the protocol setting data

Check item	Action
Did the ERR.LED turn on after writing the protocol setting data?	Check the values in Protocol setting data error information (address: 5324_H to 5327_H) and correct the error.

Point

When an error is detected in the edited protocol, check that the conditions for element placement are satisfied in the Packet Setting. Especially the following conditions must be satisfied.

- When a Non-conversion Variable (Variable length) or Non-verified Reception (Variable number of characters) is placed behind a Length and is not included in the Length calculating range, place Static Data immediate after the Non-conversion Variable or Non-verified Reception.
- When multiple Length elements are set in one packet, configure the setting so that each Length calculating range does not overlap the others.

16.5.8 Data cannot be sent with communications using a fixed buffer.

The following table lists how to troubleshoot the problem when an E71 cannot send data with communications using a fixed buffer.

Check item	Action
Has the connection with the connected device been opened? (Check the corresponding bit of 5000_H in the buffer memory.)	• Open the connection with the connected device. • Check if the open/close processes for I/O signals and the OPEN/CLOSE instructions are not used together for the same connection. If so, correct the program. • Read Open error code (address: $7C_H$) in the communication status storage area. Then check the error details and correct the error.
Is the IP address setting of the connected device correct?	Check and correct the IP address of the connected device.
Does the communication protocol (TCP/IP or UDP/IP) match with that of the connected device?	Use the same communication protocol (TCP/IP or UDP/IP) as that of the connected device.
Was the BUFSND instruction executed?	• Correct the execution conditions of the BUFSND instruction. • Check if the send/receive process for I/O signals and the BUFSND/BUFRCV instructions are used together for the same connection. If so, correct the program.
Was the BUFSND instruction completed normally?	Check the error code in the BUFSND instruction completion status area and correct the error.
Has the data length of the send data been set in the control data?	Write the data length.
Is the communication status of the connected device normal?	Correct the error in the connected device.
Has an open or initial error occurred in the error log area?	Check and correct the error.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.9 Data cannot be received with communications using a fixed buffer.

The following table lists how to troubleshoot the problem when an E71 cannot receive data with communications using a fixed buffer.

Check item	Action
Does the receive data length match with the amount of data actually received?	If the amount of the actually received data is smaller than the receive data length, receive the remaining data. If it is larger than the receive data length, receive the excess data as the next request message or discard it.
Has the connection with the connected device been opened? (Check the corresponding bit of 5000_H in the buffer memory.)	• Open the connection with the connected device. • Check if the open/close processes for I/O signals and the OPEN/CLOSE instructions are not used together for the same connection. If so, correct the program. • Read Open error code (address: $7C_H$) in the communication status storage area. Then check the error details and correct the error.
Is the IP address setting of the connected device correct?	Check and correct the IP address of the connected device.
Does the communication protocol (TCP/IP or UDP/IP) match with that of the connected device?	Use the same communication protocol (TCP/IP or UDP/IP) as that of the connected device.
Is the reception completion signal of the corresponding fixed buffer on? (Check the corresponding bit of 5005_H in the buffer memory.)	Data was not sent from the connected device. Check the connected device on the sending side and correct the error.
Was the BUFRCV instruction executed?	• Correct the execution conditions of the BUFRCV instruction. • Check if the send/receive processes for I/O signals and the BUFSND/BUFRCV instructions are not used together for the same connection. If so, correct the program.
Was the BUFRCV instruction completed normally?	Check the error code in the BUFSND instruction completion status area and correct the error.
Was the BUFRCVS instruction executed?	• Correct the interrupt setting of the programming tool. • Check if the send/receive processes for I/O signals and the BUFSND/BUFRCV instructions are not used for the same connection. If so, correct the program.
Is the control data of the BUFRCVS instruction correct?	Correct the control data.
Has an open or initial error occurred in the error log area?	Check and correct the error.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.10 Communications using a random access buffer cannot be performed.

The following table lists how to troubleshoot the problem when an E71 cannot perform communications using a random access buffer.

Check item	Action
Has the connection with the connected device been opened? (Check the corresponding bit of 5000_H in the buffer memory.)	• Open the connection with the connected device. • Check if the open/close process for I/O signals and the OPEN/CLOSE instructions are used together for the same connection. If so, correct the program.
Did the connected device send a command?	Send a command to the E71.
Was a response returned to the device that had sent the command?	• Check if the correct IP address was specified in the command. If not, correct the IP address and send the command again. • Check if the communication protocol (TCP/IP or UDP/IP) matches with that of the connected device. If not, correct it.
Is the end code of the response 0?	Check the end and error codes to correct the error.
Is the correct buffer memory address specified for the command?	Correct the buffer memory address and send the command again.
Has data been set in the specified address of the random access buffer?	Write the data.
Has the write data been set in the connected device?	Set the data.
Has an open or initial error occurred in the error log area?	Check and correct the error.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.11 The access cannot be allowed/denied correctly by the IP filter.

The following table lists how to troubleshoot the problem when the access cannot be allowed/denied correctly by the IP filter.

Check item	Action
Is the setting value in IP filter function type setting (address: 5701_H) correct?	Correct the value in IP filter function type setting (address: 5701_H) and perform IP filter setting by the reinitialization process again.
Are the setting values in IP address setting 1 to 8 (address: 5702_H to 5721_H) correct?	Correct the values in IP address setting 1 to 8 (address: 5702_H to 5721_H) and perform IP filter setting by the reinitialization process again.
Is the IP address of the proxy server set to be allowed?	Set the IP address of the proxy server to be denied.

16.5.12 An e-mail cannot be sent.

The following table lists how to troubleshoot the problem when an E71 cannot send an e-mail.

(1) When sending an e-mail with a program

Check item	Action
Was the MSEND instruction executed?	Correct the execution conditions of the MSEND instruction.
Was the MSEND instruction completed normally?	Check the error code in the MSEND instruction completion status area and the e-mail sending error log to correct the error.
Is the error response received from the mail server using the MRECV instruction?	Check with the system administrator that the following settings are correct: • DNS setting • E-mail setting

If the action above does not solve the problem, check the following.

- Is the e-mail parameter setting of the E71 correct?
- Is the mail server system down?
- Is there any error in the communication path?

(2) When sending an e-mail with the programmable controller CPU monitoring function

Check item	Action
Are the monitoring conditions satisfied?	Check the notification setting with the programming tool.
Was the first sending performed using the programmable controller CPU monitoring function?	Check if there is any device outside the range of the device settings among the condition devices and correct the condition device.

Were the second and subsequent sendings performed using the programmable controller CPU monitoring function?	• Correct the following so that the monitoring condition disable time becomes longer than the value set in "PLC Inquiry Interval".
	  • PLC Inquiry Interval
	  • Condition Device
	• Correct the following so that the sending time of the SMTP server and the E71 becomes shorter than the value set in "PLC Inquiry Interval".
	  • Reduce the load on the SMTP server.
	  • When the SMTP server and the E71 are connected via routers or other devices, connect the E71 to the same main line as the SMTP server.

If the action above does not solve the problem, check the following.

- Is the e-mail parameter setting of the E71 correct?
- Is the mail server system down?
- Is there any error in the communication path?

16.5.13 An e-mail cannot be received.

The following table lists how to troubleshoot the problem when an E71 cannot receive an e-mail.

Check item	Action
Was the MRECV instruction executed?	Correct the execution conditions of the MSEND instruction.
Was the MRECV instruction completed normally?	Check the error code in the MRECV instruction completion status area and correct the error.
Does the receive data exist in the device?	• Shorten the inquiry interval. • Check if there is any error in the communication path. • Check the number of e-mails remaining on the mail server.
Is the receive data value incorrect? (Are the characters in the subject garbled?)	• Check and correct the e-mail parameter settings. • Check if the mail server system is down. • Check if there is any error in the communication path.
Is a device other than the E71 sending the data?	If EUC or SHIFT-JIS is used, perform the following: • Send an e-mail from a personal computer to the own station. • Check the header information of the received e-mail sent by the own station. For Outlook Express, select "Properties" → "Details". In the header information, check that the setting is "Content-Type:text-plain:charset=iso-2022jp". If charset of header information is "EUC-jp" or "SHIFT-JIS", correct it to "iso-2022jp".
Is the E71 sending the data?	Because the E71 does not decode received data, it cannot receive an e-mail normally. Try to send an e-mail from a device other than the E71.
Did the sending device send the e-mail in the ASCII code or convert the ASCII code further into the ASCII code?	Check the following. • Convert the e-mail attachment into the binary code and send it. • Check if the sent data (e-mail attachment) is not in the ASCII code. • When the destination is a personal computer, use another mailer. If the received e-mails differ depending on the mailer used, check the mailer settings. • Check if the encoding and decoding methods of the mail server differs from those of Ethernet.

16.5.14 Communications using data link instructions cannot be performed.

The following table lists how to troubleshoot the problem when an E71 cannot perform communications using data link instructions.

Check item	Action
Is the communication status of the connected device normal?	Correct the error of the connected device.
Is the number of resends set upon every instruction execution?	Modify the program to set the number of resends upon every instruction execution.
Does an error occur even after the arrival monitoring time is increased?	Check and correct the arrival monitoring time.
Does an error occur even after the setting value in "TCP Resend Timer" in the initial setting is increased?	Check and correct the setting value in "TCP Resend Timer".
Are the routing parameter settings correct?	Check the routing parameters and correct the error.
Is the Station No. <->IP information setting configured?	Set the "Station No.<->IP Information" setting in the network parameter window. (MELSEC-Q/L Ethernet Interface Module User's Manual (Application))

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.5.15 Communications cannot be performed during OPS connection in a redundant system.

The following table lists how to troubleshoot the problem when an E71 cannot perform communications during OPS connection in a redundant system.

Check item	Action
Is "OPS connection" set under "Open system" of the open setting?	Set "OPS connection" under "Open system" of the open setting.
Is the own station port number of the E71 correct?	Correct the own station port number of the E71.
Is the IP address of the OPS correct?	Correct the IP address of the OPS.

Point

The MELSOFT connection and MC protocol cannot be used together for a connection set for the OPS connection. In addition, a port set for the OPS connection can be connected using EZSocket only.

If the action above does not solve the problem, there may be other causes or the connected device may be in failure.

16.5.16 Systems cannot be switched in a redundant system.

The following table lists how to troubleshoot the problem when systems cannot be switched in a redundant system.

Check item	Action
Has the E71 issued a system switching request to the control system CPU module?	• Use SD1590 of the control system CPU module to check if the E71 has issued a system switching request. • Check Error log area (address: $E3_H$ to 174_H) of the E71 connected to the control system CPU module. Then correct the error.
Has "Issue system switch in communication error" been selected in "Redundant settings"?	Select "Issue system switch in communication error" in the "Redundant settings".
Has the correct number been selected in "System switching settings when communication error occurs" in "Redundant settings"?	Select the correct number in "System switching settings when communication error occurs" in the "Redundant settings".
Do the settings of the programming tool match those of the buffer memory?	After writing the parameters to the CPU module, power off and on the CPU module or reset the system.
Has disconnection been detected?	Check if the communication target is faulty.
Has "Issue system switch in Cable disconnection timeout" been selected in "Redundant settings"?	Select "Issue system switch in Cable disconnection timeout" in "Redundant settings".
Is the disconnection detection monitoring time setting correct?	Check and correct the disconnection detection monitoring time setting.
Is the setting value in "TCP ULP Timer" in the initial setting correct?	Check and correct the setting value in "TCP ULP Timer".
Has "Confirm" been selected in "Existence Confirmation" in the open setting?	Select "Confirm".
Is the setting value in "Dest. Confirmation Start Interval" in the initial setting correct?	Check and correct the setting value in "Dest. Confirmation Start Interval".
Is the setting value in "Dest. Confirmation Interval" in the initial setting correct?	Check and correct the setting value in "Dest. Confirmation Interval".
Is "Broadcast Communications" displayed in "Transmission target device IP address" in the open setting?	Set the IP address of the communication target station in "Transmission target device IP address".
Is the connection open?	Open the connection.
Has the group setting been configured?	Check the connected CPU module.
Is the power supply module of the standby system on?	Power on the power supply module of the standby system.
Is the RESET/L.CLR switch of the standby system CPU module set to the central position (reset clear)?	Set the RESET/L.CLR switch to the central position (reset clear).
Is the tracking cable connected properly?	Connect the tracking cable properly.
Is the standby system CPU module operating normally?	Check and correct the error of the standby system CPU module.
Is the connected CPU module the control system?	Set the connected CPU module as the control system.
Is there any cause, such as a standby system CPU module stop error, which disables system switching even when the E71 issues a system switching request?	Remove the cause of the system switching failure. QnPRHCPU User's Manual (Redundant System)
Is the CPU module in the backup mode?	Change it from the separate mode to the backup mode.

If the action above does not solve the problem, perform the following tests on the E71 to check for hardware error.

- Hardware test (Page 375, Appendix 6.2)
- Self-loopback test (Page 374, Appendix 6.1)

16.6 Error Code List

This section describes the error codes (abnormal codes) of the errors that may occur in each process for performing communications between the E71 and a connected device as well as errors caused by process requests issued from the CPU module of the own station, the error details, and the actions to be taken.
The following table lists the types of errors that may occur.

Error type	Description	Buffer memory area that stores the error code
Errors occurring in the initial process	• Setting value error • Initial process error	Page 301, Section 16.6 (1)
Errors occurring in the open process	• Setting value error • Open process error	Page 301, Section 16.6 (2)
Errors occurring in communications using a fixed buffer (sending) with the connected device	• Specified data error • Sending error	Page 301, Section 16.6 (3), Page 301, Section 16.6 (4)
Errors occurring in communications using a fixed buffer with the connected device	• Specified data error • Communication error (excluding errors occurring in communications using a fixed buffer (sending) with the connected device)	Page 301, Section 16.6 (4)
Errors returned to the connected device when communicating with it	• Errors returned in communications using a fixed buffer (end code) • Errors returned in communications using a random access buffer (end code)	-
	Errors returned in communications using the MC protocol	End code or error code for each command
Errors occurring while communicating with the destination (including the causes in the "Description" column), and whose error codes are stored in the error log area	• Specified data error • Errors where the source cannot be found • Errors that have occurred in MC protocol communications • Errors that have occurred in communications using a random access buffer	Page 302, Section 16.6 (5)
Errors occurring in data communications using the predefined protocol	• Setting data error • Receiving error	Page 303, Section 16.6 (6) Page 305, Section 16.6 (7)
Errors (response commands) occurring in communications with the destination using the file transfer (FTP server) function	• Specified data error • Communication error	Refer to the QnUCPU User's Manual (Communication via Built-in Ethernet Port).
Errors occurring in communications using the Web function	Communication error	Page 306, Section 16.6 (8)
Errors occurring when an e-mail is received	• Setting data error • Receiving error	Page 307, Section 16.6 (9) (a)
Errors occurring when an e-mail is sent	• Setting data error • Sending error	Page 309, Section 16.6 (9) (b)
Errors occurring in communications using data link instructions from the CPU module of the own station	• Specified data error • Communication error	Not stored (stored in the complete status area of the instructions)
Errors occurring in communications using the dedicated instructions from the CPU module of the own station	• Specified data error • Communication data error	Not stored (stored in the complete status area of the instructions)

(1) Initial error code (address: 69_H)

The error codes generated when the initial process is executed are stored. Error codes are stored in binary when Initial abnormal end signal (X1A) is on. The error codes are cleared when Initial normal completion signal (X19) turns on, but can be also cleared by the following processes:

- Resetting or powering off the CPU module
- Writing 0 to the initial error code storage area using a program

(2) Open error code (address: $7C_H$ to $C1_H$, 5824_H to 5869_H)

The result of the open process for connection with the connected device is stored in binary.

- 0: Normal completion
- Values other than 0: Abnormal end (Open abnormal detection signal (X18): ON)

The error codes are cleared by the following operations:

- Reopening the connection that caused an open error
- Powering off and on or resetting the CPU module

(3) Fixed buffer sending error code (address: $7D_H$ to $C2_H$, 5825_H to $586A_H$)

The error codes generated when an error has occurred in data sending to the destination during communications using a fixed buffer in the connection with the connected device are stored. A sending error code is cleared when the next data sending is completed normally.

(4) Connection end code (address: $7E_H$ to $C3_H$, 5826_H to $586B_H$)

The codes returned in the responses from the destination during communications using a fixed buffer in the connection with the connected device are stored in binary. Process the end codes in the responses according to the destination.

(5) Error log area (address: E0$_H$ to 1FF$_H$)

The following table lists the details of the error log area. The buffer memory addresses of the error log block are those of the error log block 1. For the addresses of the error log block 2 and later, refer to the list of buffer memory addresses. (Page 38, Section 3.5.2)

Buffer memory name		Address	Description
Number of errors		E3$_H$	The number of errors registered in the error log block area is stored. When the number of errors reaches 65536, the count stops at FFFF$_H$(65535).[*1]
Error log write pointer		E4$_H$	The number of the error log block where the latest error log is registered is stored. • 0: No error (no registration of error log) • 1 or more: The number of the error log block where the latest error log is registered. If the pointer value is 16, it means that the latest error log is registered in the error log block 16 area. When the number of errors reaches 17, the error log is then registered in the error log block 1 area again.[*2]
Error log block	Error code/end code	E5$_H$	An error code indicating error details is stored. The subheader code of the error message is stored in bits 0 to 7 of the corresponding area. (Bits 8 to 15 store 0.) For errors below the TCP/IP or UDP/IP level, 0 is stored.
	Command code	E7$_H$	The command code of the error message or the values of the lower bytes of the request type and subrequest type of the data link instructions are stored. b15 to b0: Command code or b15 to b8: Subrequest type, b7 to b0: Request type 0 is stored in the following cases: • For messages not containing a command code • For errors below the TCP/IP or UDP/IP level (because their commands are unknown)
	Connection No.	E8$_H$	The number of the error connection is stored in bits 0 to 7 of the corresponding area. (Bits 8 to 15 store 0.) For errors below the TCP/IP or UDP/IP level, 0 is stored.
	Own station port No.	E9$_H$	The port number of the error own station is stored. For errors below the TCP/IP or UDP/IP level, 0 is stored.
	Destination IP address	EA$_H$ and EB$_H$	The IP address of the error connected device is stored. 0 is stored in the following cases: • For errors below the IP level • When an error response was sent through the CPU module
	Destination Port No.	EC$_H$	The IP address of the error connected device is stored. For errors below the TCP/IP or UDP/IP level, 0 is stored.
Status for each protocol		178$_H$ to 1FF$_H$	The number of occurrences of the corresponding status of each protocol is stored. When the count value by the E71 exceeds two words, the count stops at FFFFFFFF$_H$ (4294967295).

*1 The error information is continued to be stored into the following areas even if the count of the errors is stopped:
- Error log write pointer storage area
- Error log block

*2 An error log block area consists of 16 error log blocks that have the same data order.

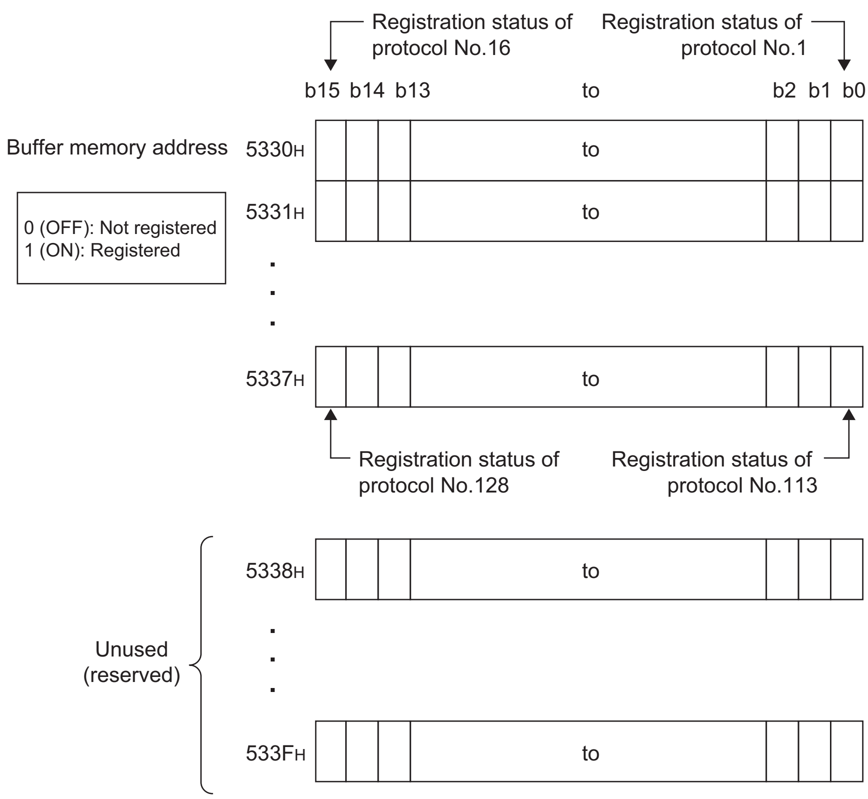
Point

Values stored in the buffer memory are cleared when an E71-mounted station is powered on or reset. (They are not cleared during the initial process.) Although this area does not normally need to be read, read it when necessary for maintenance or other purposes.

(6) Protocol setting data check area (address: 5320_H to $533F_H$)

The following table lists the details of the protocol setting data check area.

Buffer memory name		Address	Description
Protocol setting data error information	Protocol No.	5324_H	When an error is detected in the protocol setting data, the protocol No. where an error has been detected is stored. The protocols are checked from the lowest protocol No. and the first protocol No. where an error has been detected is stored. • 0: No errors • 1 to 128: Protocol No. • 65535: Unidentified*1
	Setting type	5325_H	When an error is detected in the Packet Setting or Element Setting, 0 is stored. When an error is detected in the Protocol Detailed Setting, 1 is stored. (Valid when the Protocol No. value is 1 to 128) • 0: Packet setting or element setting • 1: Protocol detailed setting • 65535: Unidentified*1
	Packet No.	5326_H	When an error is detected in the protocol setting data, the packet No. where an error has been detected is stored. The send packet is checked first, then receive packet (expected packet) is checked from the lowest packet No. and the first packet No. where an error has been detected is stored. (Valid when the Setting type value is 0) • 0: Send packet • 1 to 16: Receive packet • 65535: Unidentified*1
	Element No.	5327_H	When an error is detected in the protocol setting data, the element No. where an error has been detected is stored. The elements are checked from the lowest element No. and the first element No. where an error has been detected is stored. (Valid when the Setting type value is 0) • 1 to 32: Element No. • 65535: Unidentified*1
Number of registered protocols		5328_H	Number of registered protocols in the protocol setting data is stored. When the check results include any error, 0 is stored. • 0: No registration • 1 to 128: Number of registrations

Buffer memory name	Address	Description
Protocol registration	5330_H to $533F_H$	The information about whether protocol setting data has been registered is shown by on/off status of the corresponding bit. When the check results include any error, 0 is stored in all bits. 

*1 When the setting value gets Unidentified (65535), the following causes are possible:
- When settings undetectable with the E71 used are written
- When the protocol setting data is broken (hardware failure)

(7) Predefined protocol support function execution status check area (address: 54C0$_H$ to 55FF$_H$)

The following table lists the details of the predefined protocol support function execution status check area.
The buffer memory addresses are those of Connection No.1. For the addresses of the Connection No.2 and later, refer to the list of buffer memory addresses. (Page 38, Section 3.5.2)

Buffer memory name	Address	Description
Protocol execution status	54C0$_H$	The status of protocols being executed in Connection No.1 is stored. 0: Unexecuted 1: Waiting for transmission 2: Sending 3: Waiting for data reception 4: Receiving 5: Completed
Received data verification result (receive packet No.1)	54C2$_H$	Verification result of receive packet No.1 is stored. (Page 305, Section 16.6 (7) (a)) • b0 to b7: Element No. where the verification result did not match • b8 to b15: The cause of mismatch (verification result code)
Received data verification result (receive packet No.2 to 16)	54C3$_H$ to 54D1$_H$	The bit configuration is the same as receive packet No.1.
Number of protocol executions	54D2$_H$	The number of protocol executions in Connection No.1 is stored. • 0: No protocol execution • 1 to 65535: Number of executions (When the number exceeds 65535, the value remains 65535.)
Protocol cancellation specification	54D3$_H$	Cancels the protocol executed in Connection No.1. • 0: No cancellation specification • 1: Cancellation request (set by user) • 2: Cancel completed (set by the system)

(a) Received data verification result

The following information is stored in Received data verification result.

- Element No. where the verification result did not match (b0 to b7)

Stored value	Description
0	Verification matched
1 to 32	Element No. where the verification result did not match
FF$_H$	Verification not performed

(b) The cause of mismatch (verification result code) (b8 to b15)

Stored value	Description	Cause
00_H	Normal	-
01_H	Insufficient receive data	The total packet size of receive data is smaller than that set in protocol data.
10_H	Data not matched	The receive data do not match the value set in protocol data.
11_H	ASCII-binary conversion error	When "ASCII Hexadecimal" is set in Code Type, data not in ASCII code are received.
12_H	Data length error	The received Length value exceeded 2046 bytes.
30_H	Data length size error	The Length value received from the connected device does not match the actual length.
FF_H	Verification not performed	-

(8) HTTP status storage area (address: 5101_H to 5177_H)

The following table lists the HTTP status storage areas. The buffer memory addresses of the error log block are those of the error log block 1. For the addresses of the error log block 2 and later, refer to the list of buffer memory addresses. (Page 38, Section 3.5.2)

<table>
<tr><th colspan="2">Buffer memory name</th><th>Address</th><th>Description</th></tr>
<tr><td colspan="2">Error log pointer</td><td>5101_H</td><td>The number of the error log block where the latest error log is registered is stored.
• 0: No error (no registration of error log)
• 1 or more: The number of the error log block where the latest error log is registered.

If the pointer value is 16, it means that the latest error log is registered in the error log block 16 area. When the number of errors reaches 17, the error log is then registered in the error log block 1 area again.*1</td></tr>
<tr><td colspan="2">Log counter</td><td>5101_H to 5106_H</td><td>The number of times the HTTP response code was returned from the E71 to the Web browser is stored.</td></tr>
<tr><td rowspan="3">Error log block</td><td>HTTP response code</td><td>5108_H</td><td>The HTTP response code in case of an error is stored.</td></tr>
<tr><td>Destination IP address</td><td>5109_H to $510A_H$</td><td>The IP address of the server in case of an error is stored.</td></tr>
<tr><td>Error time</td><td>$510B_H$ to $510E_H$</td><td>The time when the error occurred is stored in the BCD code.
<table><tr><th>b15 to b8</th><th>b7 to b0</th></tr><tr><td>Month (01_H to 12_H)</td><td>Year (00_H to 99_H): Last two digits of the year</td></tr><tr><td>Hour (00_H to 23_H)</td><td>Day (01_H to 31_H)</td></tr><tr><td>Second (00_H to 59_H)</td><td>Minute (00_H to 59_H)</td></tr><tr><td>Year (00_H to 99_H): First two digits of the year</td><td>Day of the week (0 to 6)</td></tr></table></td></tr>
</table>

*1 An error log block area consists of 16 error log blocks that have the same data order.

(9) E-mail status storage area (address: 5870_H to $5FFF_H$)

When the stored count exceeds $FFFF_H$, the count starts from 0_H again. The following table lists the details of the e-mail status storage area. The buffer memory addresses of the error log block are those of the error log block 1. For the addresses of the error log block 2 and later, refer to the list of buffer memory addresses.
(☞Page 38, Section 3.5.2)

(a) Receive area

Buffer memory name	Address	Description
Number of mails remaining on the server	5870_H	The number of remaining e-mails returned when the E71 inquired the receive mail server is stored. • 0: No received mail in 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.
Dedicated instruction normal completion count	5871_H	The cumulative number of normal completions of the dedicated instruction (MRECV) is stored. • 0: The MRECV instruction has not been executed or no executions have resulted in normal completion. • 1 or more: The cumulative number of normal completions of the MRECV instruction
Dedicated instruction abnormal completion count	5872_H	The cumulative number of abnormal ends of the dedicated instruction (MRECV) is stored. • 0: The MRECV instruction has not been executed or no executions have resulted in abnormal end. • 1 or more: The cumulative number of abnormal ends of the MRECV instruction
Normal receiving count	5873_H	The cumulative number of received e-mails forwarded by the E71 to the mail buffer data area is stored. • 0: No mail has been transferred. • 1 or more: The number of normal mail transfer completions
Attached file receiving count	5874_H	The cumulative number of e-mails with an attachment received by the E71 is stored. • 0: No e-mail with an attachment has been received. • 1 or more: The number of normal reception of e-mails with an attachment
Server inquiry count	5875_H	The cumulative number of inquiries to the receive mail server according to the parameter setting is stored. • 0: No inquiry has been made to the server. • 1 or more: The cumulative number of inquiries to the server
Server communication error count	5876_H	The cumulative number of communication errors returned in response to inquiries to the receive mail server is stored. • 0: No communication error between servers or no inquiry • 1 or more: The cumulative number of communication errors
Error log write count	5877_H	The cumulative number of registrations in the receiving error log block area is stored. • 0: No error or no inquiry to the server • 1 or more: The cumulative number of writes to the error log block area
Receiving error log write pointer	5878_H	The number of the error log block where the latest receiving error log is registered is stored. • 0: No error (no registration of error log) • 1 or more: The number of the error log block where the latest error log is registered. If the pointer value is 16, it means that the latest error log is registered in the receiving error log block 16 area. When the number of errors reaches 17, the error log is then registered in the error log block 1 area again.[*1]

Buffer memory name		Address	Description
Error log block	Error code	5879_H	An error code indicating the error details is stored.
	Command code	$587A_H$	The system command code of the error message is stored.
	From	$587B_H$	The first eight words of the sender's e-mail address of the error e-mail in the communications with the mail server are stored in the ASCII code. (Example) If the sender's e-mail address is "use@from.add.sample.co.jp", "use@from.add.sam" is stored in the ASCII code.
	Date	5883_H	The date and time when the e-mail was received are stored in the BCD code. b15 to b8 | b7 to b0 Month (01_H to 12_H) | Year (00_H to 99_H): Last two digits of the year b15 to b8 | b7 to b0 Hour (00_H to 23_H) | Day (01_H to 31_H) b15 to b8 | b7 to b0 Second (00_H to 59_H) | Minute (00_H to 59_H) b15 to b8 | b7 to b0 Year (00_H to 99_H): First two digits of the year | Day of the week (0 to 6)
	Subject	5887_H	The first 30 words of the e-mail subject are stored. A subject is not stored successfully if it contains characters other than alphanumeric and ASCII code characters.

*1 A receiving error log block area consists of 16 error log blocks that have the same data order.

(b) Send area

Buffer memory name		Address	Description
Dedicated instruction normal completion count		$5B39_H$	The cumulative number of normal completions of the dedicated instruction (MSEND) is stored. • 0: The MSEND instruction has not been executed or no executions have resulted in normal completion. • 1 or more: The cumulative number of normal completions of the MSEND instruction
Dedicated instruction abnormal completion count		$5B3A_H$	The cumulative number of abnormal ends of the dedicated instruction (MSEND) is stored. • 0: The MSEND instruction has not been executed or no executions have resulted in abnormal end. • 1 or more: The cumulative number of abnormal ends of the MSEND instruction
Number of mails normally completed		$5B3B_H$	The cumulative number of sent e-mails forwarded by the E71 to the send mail server is stored. • 0: No mail has been sent. • 1 or more: The number of normal completions of mail sending
Attached file sending count		$5B3C_H$	The cumulative number of e-mails with an attachment sent by the E71 is stored. • 0: No e-mail with an attachment has been sent. • 1 or more: The number of normal sending of e-mails with an attachment
Sending to the server count		$5B3D_H$	The cumulative number of e-mails sent by the E71 to the send mail server is stored. • 0: No e-mail has been sent to the server. • 1 or more: The cumulative number of sending e-mails to the server
Number of mails abnormally completed		$5B3E_H$	The cumulative number of communication errors returned in response to send requests to the send mail server is stored. • 0: No communication error between servers or no e-mail sent • 1 or more: The cumulative number of communication errors
Error log write count		$5B3F_H$	The cumulative number of registrations in the send error log block area is stored. • 0: No error or no inquiry to the server • 1 or more: The cumulative number of writes to the error log block area
Sending error log write pointer		$5B40_H$	The number of the error log block where the latest send error log is registered is stored. • 0: No error (no registration of sending error log) • 1 or more: The number of the error log block where the latest sending error log is registered. If the pointer value is 8, it means that the latest error log is registered in the receiving error log block 8 area. When the number of receiving errors reaches nine, the error log is then registered in the error log block 1 area again.*1
Error log block	Error code	$5B41_H$	An error code indicating the error details is stored.
	Command code	$5B42_H$	The system command code of the error message is stored.
	To	$5B43_H$	The first eight words of the receiver's e-mail address of the error e-mail in the communications with the mail server are stored in the ASCII code. (Example) If the sender's e-mail address is "use@from.add.sample.co.jp", "use@from.add.sam" is stored in the ASCII code.
	Date	$5B4B_H$	The date and time when the e-mail was sent are stored in the BCD code (similar to 5883_H of (a)).
	Subject	$5B4F_H$	The first 15 words of the e-mail subject are stored. A subject is not stored successfully if it contains characters other than alphanumeric and ASCII code characters.

*1 A sending error log block area consists of 16 error log blocks that have the same data order.

16.6.1 End codes returned to a connected device during data communications

The following table lists the error codes stored in the end code appended to a response during data communications.

○: Stored in the end code appended to a response

Error code	Error description	Action	Data communication type: MC protocol communications	Data communication type: Communications using a fixed buffer	Data communication type: Communications using a random access buffer
00_H	• Normal completion	• When each communication is normally completed, the error code 00_H is stored.	○	○	○
02_H	• The device range destination of devices to be read/written from/to is incorrect.	• Check and correct the specified start device and the number of points.	○		
50_H	• Codes for the command/response type of a subheader are not within the specifications. • MC protocol communications: 00_H to $3C_H$ • Communications using a fixed buffer: 60_H • Communications using a random access buffer: 61_H, 62_H • In communications using a fixed buffer, if the data length setting is smaller than the actual data amount, the remaining data is processed as the second data. In this case, a subheader undefined command type error may occur.	• Check and correct the command/response type set for the connected device. (Because the E71 automatically adds the command/response type, the user setting is not required.) • Check and correct the data length.	○	○	○
51_H	• In communications using a random access buffer, the start address specified by a connected device has been set outside the range from 0 to 6143.	• Check and correct the specified start address.			○
52_H	• In communications using a random access buffer, the start address and the number of data words (depending on the setting when reading data) specified by a connected device exceeds the range from 0 to 6143. • Data by the number of words specified (text) cannot be sent in one frame. (The data length value or the amount of the text sent/received are not in the allowable range.)	• Check and correct the start address and the number of data words. • Correct the number of read/write points.		○	○
54_H	• When "ASCII Code" has been selected in the communication data code setting in the Ethernet operation setting, ASCII code data which cannot be converted into binary code data has been sent from the connected device.	• Check and correct the data sent from the connected device.	○	○	○
55_H	• When "Enable Online Change" was disabled (or not selected) in the Ethernet operation setting, the connected device requested a data write while the CPU module was running. • While the CPU module was running, the connected device requested for writing a parameter, program, or microcomputer program.	• Enable (select) "Enable Online Change" in the Ethernet operation setting and write data. (However, a parameter, program, or microcomputer program cannot be written while the CPU module is running.) • Set the switch on the CPU module to STOP and write the data.	○		

Error code	Error description	Action	Data communication type		
			MC protocol communications	Communications using a fixed buffer	Communications using a random access buffer
56_H	• The connected device specified a wrong device.	• Correct the device specification.	○		
57_H	• The number of points for a command specified by a connected device exceeds the maximum number of processing points (number of processes that can be executed per communication) for each process. • Addresses from the start address (start device number and start step number) to the specified number of points exceed the largest addresses (device number and step number) for each process.	• Correct the specified points or the start address (device number and step number).	○		
	• The byte length of a command is not within the range defined by the specifications. • When data is written, the set number of write data points is different from the specified number of points.	• Check the data length of the command and set the data again.	○		
	• A monitor request was issued even though monitoring data is not registered.	• Register the monitoring data.	○		
	• For data read/write in a microcomputer program, an address after the last address that can be set in the parameter setting has been specified.	• Data cannot be read from/written to an area with an address after the last address. Correct the specified address.	○		
	• In the block number specification of the extension file register, a block number exceeding the range of the corresponding memory cassette size has been specified.	• Correct the block number.	○		
58_H	• The start address (start device number and start step number) of a command specified by a connected device has been set outside the range that can be specified. • For data read/write in a microcomputer program or file register, values exceeding the CPU module parameter setting range has been specified.	• Correct the value to the one within the range that can be specified for each process.	○		
	• A block number specified for an extension file register does not exist.	• Correct the block number.	○		
	• A file register cannot be specified.	• Correct the device specification.	○		
	• A word device has been specified for the command for bit devices. • The start number of bit devices has been specified using a value other than a multiple of 16 in the command for word devices.	• Correct the command or specified device.	○		
59_H	• An extension file register cannot be specified.	• Correct the device specification.	○		
$5B_H$	• The CPU module and the E71 cannot communicate. • The CPU module cannot process a request from a connected device.	• Check the error code appended to the end code and correct the error.	○		
60_H	• The communication time between the CPU module and the E71 exceeded the CPU monitoring timer value.	• Increase the CPU monitoring timer value.	○		

Error code	Error description	Action	Data communication type		
			MC protocol communications	Communications using a fixed buffer	Communications using a random access buffer
63_H	• In communications using a fixed buffer, the remote password of the port for the destination E71 is in the lock status.	• After unlocking the remote password using the MC protocol, perform communications using a fixed buffer. • Exclude ports used for communications using a fixed buffer from the targets of the remote password check.		○	○
$A0_H$ to $FFFF_H$	• The error details and how to troubleshoot the problem are the same as those for the error code stored in the buffer memory. (Page 314, Section 16.6.3)				

16.6.2 Abnormal codes returned during communications using an A-compatible 1E frame

The following table lists the error codes stored in the abnormal code appended to a response in the A-compatible 1E frame in MC protocol communications. (An abnormal code is appended only when an end code is $5B_H$.)

Response format	Subheader	End code	Abnormal code	00_H

If an abnormal code is stored, "$5B_H$" is stored here.

Error code	Error	Error description	Action
10_H	PC number error	The PC number specified with a command is not FF of the own station or not the station number set with the MELSECNET link parameters.	Change the PC number to FF_H of the own station or not the station number set using the link parameter to perform communications again.
11_H	Mode error	After the E71 receives a request from the connected device normally, communications cannot be normally performed between the E71 and the CPU module due to some causes such as noise.	Perform communications again. If the error occurs again, check for noise and other causes and/or replace the E71.
12_H	Intelligent function module designation error	The specified intelligent function module number has not been assigned to an intelligent function module that has the buffer memory and where communications can be performed. (For example, the specified module is an I/O module or the specified slot is empty.)	Change the specified data contents in the control procedure or change the mounting position of the intelligent function module. Then perform communications again.
18_H	Remote error	The remote RUN/STOP cannot be executed. (The remote STOP/PAUSE has been already executed from another module.)	Check if the remote STOP/PAUSE has been already executed from another module. Then cancel the operation to perform communications again.
$1F_H$	Device error	Invalid device specification	• Correct the specified device. • Do not access a device which does not exist.
20_H	Link error	The CPU module of the request destination has been disconnected from the data link.	Check if the CPU module of the station number set for the PC number has been disconnected. Then remove the cause of the disconnection to perform communications again.
21_H	Intelligent function module bus error	Memory access to the intelligent function module cannot be performed due to the following causes: • The control bus to the intelligent function module is faulty. • The intelligent function module is in failure.	There is a hardware problem with the CPU module, base unit, intelligent function module, or the E71. Please consult your local Mitsubishi representative.

16.6.3 Error codes stored in the buffer memory

The following table lists the error codes stored in each buffer memory area when an error occurs. For the buffer memory areas where these error codes are stored, refer to the descriptions of the corresponding buffer memory area. (Page 300, Section 16.6)

Error code	Error description	Action
00_H	The error details and how to troubleshoot the problem are the same as those for the end code returned to the connected device in data communications. (Page 310, Section 16.6.1)	
02_H		
0050_H		
0051_H		
0052_H		
0054_H		
0055_H		
0056_H		
0057_H		
0058_H		
0059_H		
$005B_H$		
0060_H		
0063_H		
$00A0_H$	This request cannot be specified in connection with the connected device.	• Correct the request details. • Correct the open setting.
$00A1_H$	Contents of the request cannot be analyzed because the text length or request data length is too short.	• Check and correct the text length or request data length of the Qn header. Then send data to the Ethernet module again.
$00A2_H$	This request cannot be processed.	• Correct the request details and command.
$3E8_H$ to $4FFF_H$	(Errors detected by a programmable controller CPU)	• Refer to the troubleshooting section of the user's manual (hardware design, maintenance and inspection) for the CPU module used, and take corrective action.
7000_H to $7FFF_H$	(Errors detected by a module such as a serial communication module)	• Refer to a manual such as a serial communication module user's manual, and take corrective action.
$B000_H$ to $BFFF_H$	(Errors detected by a CC-Link module)	• Refer to the CC-Link System Master/Local Module User's Manual, and take corrective action.
$C001_H$	• The IP address setting value of the E71 for the initial process is incorrect. • The setting value of the subnet mask field for the router relay function is incorrect.	• Correct the IP address. Set the class to A/B/C. • Correct the subnet mask.
$C002_H$	Some of the various timer setting values for the initial process are outside the allowable range.	• Check and correct each timer setting value for the initial process.
$C003_H$	The setting value of the auto-open UDP port number for the initial process is outside the allowable range.	• Check and correct the auto-open UDP port number.
$C004_H$	The setting value of the subnet mask field is incorrect.	• Correct the subnet mask and perform the initial process again.

Error code	Error description	Action
C005H	• The setting value of the default router IP address for the router relay function is incorrect. • The network address (network address after the subnet mask) of the default router IP address is different from that of the IP address of the Ethernet module in the own station.	• Correct the default router IP address and perform the initial process again. • Set the same network address as that of the IP address of the Ethernet module in the own station.
C006H	The setting value of the subnet address for the router relay function is incorrect.	• Correct the subnet address and perform the initial process again.
C007H	• The setting value of the router IP address for the router relay function is incorrect. • The network address (network address after the subnet mask) of the router IP address is different from that of the IP address of the Ethernet module in the own station.	• Correct the router IP address and perform the initial process again. • Change the IP address of the Ethernet module in the own station to the same address as the network address.
C008H	The setting value in IP filter function type setting (address: 5701H) is not correct.	Correct the value in IP filter function type setting (address: 5701H) and perform IP filter setting by the reinitialization process again.
C009H	• A value out of range is set in IP address setting 1 to 8 (address: 5702H to 5721H). • In IP address setting 1 to 8 (address: 5702H to 5721H), the value of start IP address is greater than the end IP address.	Correct the values in IP address setting 1 to 8 (address: 5702H to 5721H) and perform IP filter setting by the reinitialization process again.
C00EH	The send/receive process has been executed while the IP address in use detection function is being executed.	Do not execute the send/receive process while executing the IP address in use detection function.
C00FH	One IP address is used for two or more modules.	• Set different IP addresses. The MAC addresses of the Ethernet modules in the stations with the same IP address can be checked in IP address status storage area (5281H to 5286H).
C010H	The setting value of the Ethernet module port number for the open process is incorrect.	• Correct the port number.
C011H	The setting value of the port number of the connected device for the open process is incorrect.	• Correct the port number.
C012H	The port number set is used in a connection already opened in TCP/IP.	• Correct the port numbers of the Ethernet module and the connected device.
C013H	The port number used in a connection already opened is set for the open process in UDP/IP.	• Correct the port number of the Ethernet module.
C014H	The initial process or open process of the Ethernet module is not completed.	• Perform the initial process or open process.
C015H	The setting value of the IP address of the connected device for the open process is incorrect.	• Correct the IP address. Set the class to A/B/C.
C016H	The open process of the connection (or the next connection) specified for pairing open has been already completed.	• Check that none of the connections targeted for pairing open has been opened. • Correct the combination of modules set for pairing open.
C017H	A connection could not be established in the open process of the TCP connection.	• Check the operation of the connected device. • Check the open process of the connected device. • Correct the open setting of the communication parameters. • Correct the port number of the Ethernet module, the IP address and port number of the connected device, and the open setting. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.

Error code	Error description	Action
C018$_H$	The setting value of the IP address of the connected device is incorrect.	• Correct the IP address.
C020$_H$	The data length exceeds the allowable range.	• Correct the data length. • When the amount of data to be sent exceeds the limit, divide the data into smaller chunks to send it.
C021$_H$	An abnormal end response was received for communications using a fixed buffer.	• Read the response end code from the connection end code or error log area and perform the required action.
C022$_H$	• A response could not be received within the response monitoring timer value. • The connection with the connected device was closed while waiting for a response.	• Check the operation of the connected device. • Correct the response monitoring timer value. • Check the open status of the connection with the connected device.
C023$_H$	• The connection with the connected device has not been opened. • The connection with the connected device has been closed.	• Open the connection with the connected device.
C024$_H$	• When "Predefined protocol" is set in "Fixed Buffer Communication" setting, communications using a fixed buffer or a random access buffer are executed. • When "Procedure Exist" or "No Procedure" is set in "Fixed Buffer Communication" setting, a predefined protocol is executed.	• Correct the connection number of BUFSND/BUFRCV/BUFRCVS/ECPRTCL instructions. • Correct the "Fixed Buffer Communication" setting (Procedure Exist/No Procedure/Predefined protocol) of the corresponding connection.
C025$_H$	There is an error in the usage setting area when starting the open process by the OPEN instruction or input/output signals.	• When starting the open by using the OPEN instruction, correct the usage setting area of the control data. • When starting the open by input/output signals, correct the connection usage setting area of the buffer memory.
C026$_H$	An error has occurred when reading/writing/verifying the predefined protocol setting data.	• Check that connection cable with the engineering tool is not disconnected and read/write/verify the predefined protocol setting data again. • Do not write data simultaneously when writing protocol setting data from multiple engineering tools.
C030$_H$	A sending error has occurred.	• Check the operation of the transceiver and connected device. * Use a transceiver which can perform the SQE test. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. • Perform the self-diagnostics test to check for an error in the Ethernet module.
C031$_H$	A sending error has occurred.	• Check the operation of the transceiver and connected device. * Use a transceiver which can perform the SQE test. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. • Perform the self-diagnostics test to check for an error in the Ethernet module.

Error code	Error description	Action
C032$_H$	A TCP ULP timeout error has occurred in the TCP/IP communication. (The connected device does not send an ACK response.)	• Check the operation of the connected device. • Correct the TCP ULP timeout value and perform the initial process again. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
C033$_H$	There is no connected device with the IP address that has been set.	• Correct the IP address of the connected device and the Ethernet address. • Use the default value (FFFFFFFFFFFF$_H$) if the connected device has the ARP function. If it does not have the function, set the MAC address. • Check the operation of the connected device. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
C034$_H$	On the ARP table, there is no connected device with the IP address that has been set.	• Decrease the number of communication destinations.
C035$_H$	The existence of the connected device could not be checked within the response monitoring timer value.	• Check the operation of the connected device. • Correct the settings of the alive check. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
C036$_H$	Because a cable is not connected or disconnected, the send process cannot be executed.	• Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. • Perform the loopback test to check for an error in the line. • Perform the self-diagnostics test to check for an error in the Ethernet module.
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 the message divided at the TCP/IP level could not be received within the response monitoring timer value.	• Correct the data length of the communication data. • Because there may be congestion of packets on the line, correct the settings of the initial process. • Send the same data from the connected device again.
C041$_H$	There is an error in the checksum of the receive data when TCP is used.	• Correct the checksum value sent from the connected device and send the correct value. • Check the situation of the line, such as noise, the distance between the line and power cable, and the grounding of each device.
C042$_H$	There is an error in the checksum of the receive data when UDP is used.	• Correct the checksum value sent from the connected device and send the correct value. • Check the situation of the line, such as noise, the distance between the line and power cable, and the grounding of each device.
C043$_H$	The checksum in the header of the IP packet received is incorrect.	• Correct the checksum value sent from the connected device and send the correct value. • Check the situation of the line, such as noise, the distance between the line and power cable, and the grounding of each device.

Error code	Error description	Action
$C044_H$	An error packet of ICMP was received.	• Check the operation of the connected device. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
$C045_H$	An error packet of ICMP was received.	• Check the operation of the connected device. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
$C046_H$	An error packet of ICMP was received.	• Check the operation of the connected device. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
$C047_H$	An error packet of ICMP was received.	• Check the operation of the connected device. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
$C048_H$	An error packet of ICMP was received.	• Check the operation of the connected device. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor.
$C049_H$	An error packet of ICMP was received.	• Check the operation of the connected device. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. • If an alive check timeout occurs, correct the IP assembly timer value of the connected device.
$C04A_H$	An error packet of ICMP was received. (An IP assembly timeout error has occurred in the connected device.)	• Check the operation of the connected device. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. • If an alive check timeout occurs, correct the IP assembly timer value of the connected device.
$C04B_H$	An IP assembly timeout error has occurred. (The remaining part of the divided data could not be received and a timeout has occurred.)	• Check the operation of the connected device. • Because there may be congestion of packets on the line, send data after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. • Correct the IP assembly timer value and perform the initial process again.
$C04C_H$	Because there is no space in the internal buffer, such as the IP header buffer, data cannot be sent.	• Send the same data again and check the receipt of a response.
$C04D_H$	• In communications using an auto-open UDP port or communications using a fixed buffer in the "No Procedure" control method, there is an error in the data length specified in the application data of the message received by the Ethernet module. • Not all the receive data can be stored.	• Correct the data length. • Correct the text size so that the text data size is within the receive buffer memory size.

Error code	Error description	Action
$C050_H$	When "ASCII Code" has been selected in the communication data code setting in the Ethernet operation setting, ASCII code data which cannot be converted into binary code data has been sent from the connected device.	• Select "Binary Code" in the Ethernet operation setting. Then restart the Ethernet module to perform communications again. • Check and correct the data sent from the connected device and send it again.
$C051_H$	The number of read/write points is outside the allowable range.	• Correct the number of read/write points and send the data to the Ethernet module again.
$C052_H$	The number of read/write points is outside the allowable range.	• Correct the number of read/write points and send the data to the Ethernet module again.
$C053_H$	The number of read/write points is outside the allowable range.	• Correct the number of read/write points and send the data to the Ethernet module again.
$C054_H$	The number of read/write points is outside the allowable range.	• Correct the number of read/write points and send the data to the Ethernet module again.
$C055_H$	The number of file data read/write points is outside the allowable range.	• Correct the number of read/write points (or the number of bytes) and send the data to the Ethernet module again.
$C056_H$	• The read/write request exceeds the largest address. • The address is 0.	• Correct the start address or the number of read/write points and send the data to the Ethernet module again. (Make sure that the request does not exceed the largest address.)
$C057_H$	The request data length does not match with the number of data in the character (a part of text).	• Check and correct the text or requested data length of the Qn header. Then send the data to the Ethernet module again.
$C058_H$	The request data length after the ASCII/binary conversion does not match with the number of data in the character (a part of text).	• Check and correct the text or requested data length of the Qn header. Then send the data to the Ethernet module again.
$C059_H$	Incorrect designation of the command and subcommand	• Correct the request details.
$C05A_H$	The Ethernet module cannot read/write data from/to the specified device.	• Check the device for read/write.
$C05B_H$	The Ethernet module cannot read/write data from/to the specified device.	• Check the device for read/write.
$C05C_H$	Incorrect request (For example, a request for data read/write in bit units has been issued to a word device.)	• Correct the request details and send it to the Ethernet module again. (For example, change the subcommand.)
$C05D_H$	The monitor registration is not performed.	• Register the monitoring data and perform monitoring.
$C05E_H$	The communication time between the Ethernet module and the programmable controller CPU exceeded the CPU monitoring timer value.	• Increase the CPU monitoring timer value. • Check that the programmable controller CPU is operating normally. • Correct the network number or PC number. • If the destination is a station with a different network number, correct the routing parameter setting. • If the destination is a station with a different network number, check if the network number is not in use.
$C05F_H$	This request cannot be executed to the target programmable controller.	• Correct the network number or PC number. • Correct the read/write request details.
$C060_H$	Incorrect request (For example, incorrect data was specified for bit devices.)	• Correct the request details and send it to the Ethernet module again. (For example, correct the data.)
$C061_H$	The request data length does not match with the number of data in the character (a part of text).	• Check and correct the text or requested data length of the Qn header. Then send the data to the Ethernet module again.
$C062_H$	When writing was prohibited while the CPU was running, data has been written to the remote I/O station using the MC protocol (with a QnA-compatible 3E frame or 4E frame).	• To write data to the remote I/O station using the MC protocol (with a QnA-compatible 3E frame or 4E frame), select "Enable Write at RUN time" in the Ethernet operation setting.

Error code	Error description	Action
$C070_H$	The device memory cannot be extended for the target station.	• Read/write data without the device memory set for extension. • Device memory can be extended only in an Ethernet module-connected station and a Q/QnACPU via CC-Link IE Controller Network, MELSECNET/H, or MELSECNET/10.
$C071_H$	The number of device points for data read/write set for modules other than a Q/QnACPU is too large.	• Correct the number of device points for data read/write and send the data to the Ethernet module again.
$C072_H$	Incorrect request (For example, a request for data read/write in bit units has been issued to a word device.)	• Check whether the data can be requested to the target programmable controller CPU. • Correct the request details and send it to the Ethernet module again. (For example, change the subcommand.)
$C073_H$	The Ethernet module cannot issue this request to the target programmable controller CPU. (For example, the number of double word access points cannot be specified for modules other than a Q/QnACPU.)	• Correct the request details.
$C074_H$	This request cannot be executed to the target programmable controller CPU.	• Correct the network number or PC number. • Correct the read/write request details.
$C080_H$	The destination IP address could not be obtained during CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications or communications using data link instructions.	• Set the Station No. <-> IP information in the Ethernet module. • Change the conversion method for the CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 relay communication parameters.
$C081_H$	The termination process for the Ethernet module is in process, and whether communications using data link instructions have reached cannot be checked.	• Finish all the communications to perform the termination process of the Ethernet module.

$C082_H$	The communication process was abnormally ended in the following communications.
	• Communications with a programming tool (UDP/IP)
	• CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications
		• Check if the relay station/connected station is operating normally. (If the communication is alive, no action for this error is required.)
		• Check if there is an error with the cable connection between the own station and the connected station.
		• If the line is heavily loaded, reduce the load.
		• Increase the number of resends for the data link instructions.
		• Increase the time reserved for the communication process.
		  • For a Basic model QCPU, High Performance model QCPU, Process CPU, or Redundant CPU, use Time reserved for communication process (SD315).
		  • For a Universal model QCPU, configure the service processing setting in "PLC System" under "PLC Parameter".

$C083_H$	The communication process was abnormally ended in communications using data link instructions.	• Check if the relay station/connected station is operating normally. (If the communication is alive, no action for this error is required.)
		• Check if there is an error with the cable connection between the own station and the connected station.
		• If the line is heavily loaded, reduce the load.
		• Increase the number of resends for the data link instructions.
		• Increase the time reserved for the communication process.
		  • For a Basic model QCPU, High Performance model QCPU, Process CPU, or Redundant CPU, use Time reserved for communication process (SD315).
		  • For a Universal model QCPU, configure the service processing setting in "PLC System" under "PLC Parameter".

Error code	Error description	Action

$C084_H$	The communication process was abnormally ended in communications using data link instructions.	• Check if the own station/relay station/connected station is running normally.
		• Check if there is a problem with the cable connection between the own station and the connected station.
		• Increase the TCP resend timer value.
		• Increase the time reserved for the communication process.
		  • For a Basic model QCPU, High Performance model QCPU, Process CPU, or Redundant CPU, use Time reserved for communication process (SD315).
		  • For a Universal model QCPU, configure the service processing setting in "PLC System" under "PLC Parameter".

$C085_H$	The own station's channel specified by another station in communications using data link instructions is currently in use.	• Execute the request from another station again.
$C086_H$	A message exceeding the receive message size was received.	• Correct the send message size of the request source.
$C087_H$	There is an error in the IP address set in the Station No. <-> IP information setting for CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications.	• In the Station No.<->IP information setting, set the IP addresses for the CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 relay communication target devices.
$C0B2_H$	There is no insufficient space in the receive buffer of the relay station PC number for MELSOFT connection or data link instructions, or the communication request destination station. (Receive buffer full error)	• Increase the request interval. • Decrease the number of request nodes. • Wait for a response to the previous request before sending the next request. • Correct the timeout value.
$C0B3_H$	A request that cannot be processed was issued from the programmable controller CPU.	• Correct the request details. • Correct the network number or PC number.
$C0B5_H$	Data that cannot be processed by the programmable controller CPU or Ethernet module was specified.	• Correct the request details. • Cancel the current request.
$C0B6_H$	The channel number is outside the allowable range.	• Specify the channel number from 1 to 8.
$C0B7_H$	A channel number currently in use was specified.	• Change the channel number. • Perform communications again after the current communications are completed.
$C0B8_H$	• The network number or PC number is outside the allowable range. • A response from the programmable controller CPU is faulty.	• Correct the network number or PC number. • Check the programmable controller CPU operation.
$C0B9_H$	The open process of the TCP connection has not been completed.	• Perform the open process. • Check the operation of the connected device. • After sending a close request (FIN) from the connected device to the E71, perform the open process again and wait for 500ms or more.
$C0BA_H$	Because the close process is in process using the CLOSE instruction, a send request cannot be accepted.	• Perform the open process and send the request.
$C0BB_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.

Error code	Error description	Action
$C0BC_H$	The specified communication line is closed.	• Open the communication line. • Correct the target connection number.
$C0BD_H$	Requests have been accepted successively and cannot be sent.	• Check if requests have been sent successively without waiting for a response.
$C0BE_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0BF_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0C0_H$	The open process of the UDP connection has not been completed.	• Perform the open process. • Check the operation of the connected device.
$C0C1_H$	The transmission interval of UDP is too short.	• Check if a send request is being repeatedly issued. • Increase the send interval.
$C0C2_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0C3_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0C4_H$	The UINI instruction has been executed during communications.	• Execute the UINI instruction after closing all connections.
$C0C5_H$	• A send request was issued to the connected device with the class/network address different from those of the own station when the router relay function is not used. • The setting of the router relay parameter is incorrect.	• Enable the router relay function and perform the initial process. • Set the correct data in the router relay parameter. • Correct the IP address of the connected device and perform the open process. • Check if the network address is correct. • After changing the network address, perform the initial process again.

Error code	Error description	Action
C0C6H	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0C7H	An Ethernet module system error has occurred.	• Take action such as separating networks or decreasing the number of data sending to reduce the load on the Ethernet line. • Consult the network administrator and reduce the load on the Ethernet line. • Perform the self-diagnostics test to check for an error in the Ethernet module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0C8H	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0C9H	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0CAH	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0CBH	Another send request was issued before the send process has not been completed.	• Wait for completion of transmission before sending the next send request.
C0CCH	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.

Error code	Error description	Action
$C0CF_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0D0_H$	Incorrect data length was specified.	• Correct the specified data length.
$C0D1_H$	The number of resends is incorrect.	• Correct the number of resends.
$C0D2_H$	The arrival monitoring time is incorrect.	• Correct the arrival monitoring time.
$C0D3_H$	The number of relay stations in CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 exceeded the allowable range.	• Check the specified value for the communication destination. • Correct the settings in the Station No. <-> IP information for the stations between the own station and the communication destination.
$C0D4_H$	The number of relay stations in CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, and MELSECNET/10 exceeded the allowable range.	• Check the specified value for the communication destination. • Correct the settings in the Station No. <-> IP information for the stations between the own station and the communication destination.
$C0D5_H$	The number of retries is incorrect.	• Correct the number of retries.
$C0D6_H$	The network number or station number is incorrect.	• Correct the settings for the communication destination. • Correct the specified value for the communication destination.
$C0D7_H$	Data were sent without the initial process completed.	• Set the parameters with the programming tool and write them to the CPU module. Then perform communications with the connected device. • Wait for normal completion of the initial process before performing communications with the connected device.
$C0D8_H$	The number of blocks exceeded the range.	• Correct the specified value for the number of blocks.
$C0D9_H$	The subcommand value is incorrect.	• Correct the specified value for the subcommand.
$C0DA_H$	A response to the PING test could not be received within the time of the communication time check.	• Correct the IP address and host name of the Ethernet module where the PING test is executed. • Enable the Ethernet module where the PING test is executed to perform communications. (Complete the initial process.)
$C0DB_H$	The IP address and host name of the Ethernet module for the PING test are incorrect.	• Correct the IP address and host name of the Ethernet module to perform the PING test.
$C0DC_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.

Error code	Error description	Action
C0DD$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0DE$_H$	Data could not be received within the specified arrival monitoring time.	• Correct the specified arrival monitoring time. • Correct the specified channel number. • Check the status of the send source station and relay station.
C0DF$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C0E0$_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
C0E1$_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
C0E2$_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
C0E3$_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.

Error code	Error description	Action
$C0E4_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0E5_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0E6_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0E7_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0E8_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0E9_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.

Error code	Error description	Action
$C0EA_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0EB_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0EC_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0ED_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0EE_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0EF_H$	An error was detected in the programmable controller CPU.	• Check that the CPU module and each intelligent function module are properly mounted on the base unit. • Check that the programmable controller CPU is not in the reset status. • Check if an error exists in the programmable controller CPU. If an error exists, take the action required to correct it. • Replace the power supply module, programmable controller CPU, or intelligent function module.
$C0F0_H$	An Ethernet module RAM error was detected in the hardware test (H/W test).	• Perform the hardware test again. If an error occurs again, the Ethernet module may have a hardware error. Please consult your local Mitsubishi representative.
$C0F1_H$	An Ethernet module ROM error was detected in the hardware test (H/W test).	• Perform the hardware test again. If an error occurs again, the Ethernet module may have a hardware error. Please consult your local Mitsubishi representative.

Error code	Error description	Action
$C0F3_H$	A system error (major error) was detected in the CPU module.	• Remove the error cause of the CPU module in the own station.
$C0F4_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0F5_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0F6_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C0F7_H$	An error has occurred in the self-loopback test.	• Because there may be congestion of packets on the line, send data from the connected device after a certain period of time. • Check if the cable is disconnected. • Check if there is a problem with the connection to the transceiver or terminating resistor. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C100_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C101_H$	A response could not be received from the DNS client.	• Check the address of the DNS server. • Check if communications with the DNS server can be performed using the PING command. • Check if the IP addresses of the own station and the DNS server belong to the same class. (If they belong to different classes, check the router setting.)

Error code	Error description	Action
$C102_H$	A response from the SMTP layer could not be received.	• Check if the SMTP server name has been registered in the DNS. • Delete the SMTP server name and change the setting to the IP address setting. Then check the operation. • Check if communications with the SMTP server can be performed using the PING command.
$C103_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. • If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C104_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C105_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C106_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C110_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C111_H$	A response could not be received from the DNS client.	• Check the cables and hubs. • Check if communications with the DNS server can be performed using the PING command.

Error code	Error description	Action
$C112_H$	A response from the POP3 layer could not be received.	• Check if the POP3 server name has been registered in the DNS. • Delete the POP3 server name and change the setting to the IP address setting. Then check the operation. • Check if communications with the POP3 server can be performed using the PING command.
$C113_H$	An e-mail without an attachment was received. (This error occurs when an attachment is not read normally.)	• Specify the attachment on the sending side. • Check the program on the sending side. • If data sending using the MSEND instruction previously failed with the sending source of the mail server, check the destination of the MSEND instruction. • Check if the e-mail specifications, such as the encoding/decoding style and file format, on the sending side are the same as those for the Ethernet module. • Receive an e-mail with an unknown destination and unknown destination server from the SMTP server.
$C114_H$	An e-mail that was received has an attachment with an invalid name.	• Check if the file extension of the attachment is "bin" or "asc" on the sending side. • Check that the e-mail is not compressed or encrypted. • Check the destination of the MSEND instruction. • Receive an e-mail with an unknown destination and unknown destination server from the SMTP server.
$C115_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C116_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C117_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.

Error code	Error description	Action
$C118_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C119_H$	There is no received mail.	• Read Mail information storage area (address: 2682_H) of the buffer memory and read received e-mails remaining on the server.
$C11A_H$	The received mail failed to be converted.	• Check that the e-mail is not compressed or encrypted. • Check if the e-mail specifications, such as the encoding/decoding style and file format, on the sending side are the same as those for the Ethernet module. • Check if the sending side has not divided the file.
$C11B_H$	An e-mail was sent and an error mail was received from the mail server of the destination.	• Receive an e-mail with an unknown destination and unknown destination server from the SMTP server. (The received mail is stored in the mail buffer.) • Check if the portion before "@" is correct in the mail address setting in the parameter settings. • Check if the portion before "@" has been registered on the destination mail server.
$C11D_H$	The attachment size exceeded the allowable size.	• Check if the attachment size is 6K words or less. • Check that the sending side has not divided the attachment.
$C120_H$	The SMTP server could not be opened.	• Check that the port number of the SMTP server is 25. • Check if communications with the SMTP server can be performed using the PING command.
$C121_H$	Communications cannot be performed with the SMTP server. (Error response)	• Check if the SMTP server is busy.
$C122_H$	Communications cannot be performed with the SMTP server. (Abort)	• Check if the SMTP server is busy.
$C123_H$	Communications cannot be performed with the SMTP server. (Reset response)	• Check if the SMTP server is busy.
$C124_H$	A response from the SMTP server timed out.	• Check for an error in the SMTP server. • Check if the network is heavily loaded.
$C125_H$	Forcefully disconnected from the SMTP server	• Check for an error in the SMTP server. • Check if the network is heavily loaded.
$C126_H$	The SMTP server could not be closed.	• Check for an error in the SMTP server. • Check if the network is heavily loaded.
$C127_H$	Closing the SMTP server gave an error response.	• Check for an error in the SMTP server. • Check if the network is heavily loaded.
$C130_H$	The communication channel is closed because the service is not available.	• Check the status of the SMTP server.
$C131_H$	The SMTP server is in process, and an error response was received.	• Check that a user name not registered in the server has not been specified. • Send the e-mail again after a certain period of time.
$C132_H$	The SMTP server is in process, and an error response was received. (Local error)	• Check the status of the SMTP server.
$C133_H$	The SMTP server is in process, and an error response was received. (Insufficient memory area)	• Check the status of the SMTP server.

Error code	Error description	Action
C134$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C135$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C136$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C137$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C138$_H$	The SMTP server is in process, and an error response was received. (Mailbox not found)	• Check if the mail address of the Ethernet module has been correctly set.
C139$_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C13A$_H$	The SMTP server is in process, and an error response was received. (Exceeded the allocation of the memory area)	• Check the status of the SMTP server.
C13B$_H$	The SMTP server is in process, and an error response was received. (Illegal mailbox name)	• Check if the mail address of the Ethernet module has been correctly set.

Error code	Error description	Action
$C13C_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C140_H$	The POP3 server could not be opened.	• Check that the port number of the POP3 server is 110. (The port number of the Ethernet module is fixed to 110.) • Check if communications with the POP3 server can be performed using the PING command.
$C141_H$	Communications cannot be performed with the POP3 server. (Error response)	• Check if the POP3 server is busy.
$C142_H$	Communications cannot be performed with the POP3 server. (Abort)	• Check if the POP3 server is busy.
$C143_H$	Communications cannot be performed with the POP3 server. (Reset response)	• Check if the POP3 server is busy.
$C144_H$	A response from the POP3 server could not be received.	• Check for an error in the POP3 server. • Check if the network is heavily loaded.
$C145_H$	Forcefully disconnected from the POP3 server	• Check for an error in the POP3 server.
$C146_H$	The POP3 server could not be closed.	• Check for an error in the POP3 server. • Check if the network is heavily loaded.
$C147_H$	Closing the POP3 server gave an error response.	• Check for an error in the POP3 server.
$C150_H$	POP3 server verification error	• Check the status of the POP3 server.
$C151_H$	The Ethernet module's mail address (e-mail setting parameter) is different from the account name in the mailbox on the server side.	• Check the account name of the mailbox on the server side and correct the account of the mailbox set in the Ethernet module.
$C152_H$	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 of the Ethernet module.
$C153_H$	An error has occurred in acquisition of the received mail list. (The list of the mail that arrived at the POP3 server failed to be acquired.)	• Reset the server inquiry time to the default value and restart the CPU module of the own station.
$C154_H$	An error has occurred upon receiving a mail. (An e-mail cannot be read from the POP3 server.)	• Check that the e-mail is not compressed or encrypted. • Check if the e-mail specifications, such as the encoding/decoding style and file format, on the sending side are the same as those for the Ethernet module.
$C160_H$	A response was received from the DNS server after timeout.	• Check if the network is heavily loaded. • Check the status of the DNS server.
$C161_H$	A response from the DNS server could not be received.	• Check if the network is heavily loaded. • Check the status of the DNS server.
$C162_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.

Error code	Error description	Action
$C163_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C171_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C172_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C173_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C174_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C175_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C176_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C177_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C178_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C179_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.

Error code	Error description	Action
$C17A_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C17B_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C17C_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C17D_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C17E_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C17F_H$	An error was returned from the DNS server.	• Check if the IP address of the DNS server is correct. • Check if the mail server names (SMTP server names and POP server names) are correct. • Check with the network administrator if the DNS function of the server specified in the DNS setting is properly operating.
$C180_H$	The device specified in "Condition Device" in "News Setting" is outside the range set in the [Device] tab of the PLC parameter window.	• Correct the device settings in the PLC parameter window and specify the condition devices within the range of the device settings.
$C1A0_H$	An illegal request was made.	Try again. If the same error occurs, the Ethernet module may have a hardware error. Please consult your local Mitsubishi representative.
$C1A2_H$	A response to the request could not be received.	• Check and correct the response waiting time.
$C1A4_H$	A request or subrequest is incorrect.	• Correct the request and subrequest.
$C1A5_H$	The specified target station or clear target is incorrect.	• Correct the specified target station or clear target.
$C1A6_H$	The specified connection number is incorrect.	• Specify the connection number within the range of 1 to 16. • Do not select the connection number 8 or 16 when using the pairing open method.
$C1A7_H$	The specified network number is incorrect.	• Correct the specified network number.
$C1A8_H$	The specified station number is incorrect.	• Correct the specified station number.
$C1A9_H$	The specified device number is incorrect.	• Correct the specified device number.
$C1AA_H$	The specified device name is incorrect.	• Correct the specified device name.
$C1AC_H$	The specified number of resends is incorrect.	• Correct the number of resends.
$C1AD_H$	The specified data length is incorrect.	• Correct the specified data length.

Error code	Error description	Action
$C1AE_H$	The send/receive data length and header length are incorrect.	• Correct the specified send/receive data length and header length to values within the allowable ranges. • The send/receive data length needs to be equal to or longer than the header length.
$C1AF_H$	The specified port number is incorrect.	• Correct the specified port number.
$C1B0_H$	The open process of the specified connection has been already completed.	• Perform the close process before the open process.
$C1B1_H$	The open process of the specified connection has not been completed.	• Perform the open process.
$C1B2_H$	The OPEN or CLOSE instruction is being executed in the specified connection.	• Try again after the OPEN or CLOSE instruction is completed.
$C1B3_H$	Another send or receive instruction is being executed in the specified channel.	• Change the channel number. • Try again after the send or receive instruction is completed.
$C1B4_H$	The specified arrival monitoring time is incorrect.	• Set the arrival monitoring time to a value within the allowable range.
$C1B5_H$	Data could not be received within the specified arrival monitoring time.	• Correct the specified arrival monitoring time.
$C1B6_H$	The specified send mail address number is incorrect.	• Correct the specified send mail address number. • Correct the send mail address setting in the parameter setting.
$C1B7_H$	A reading operation was executed while no receiving e-mail was stored in the mail buffer data area.	• If there is a mail received in the mail information, execute the MRECV instruction.
$C1B8_H$	The RECV instruction was executed for the channel that had not received data.	• Correct the execution condition of the RECV instruction. • Correct the channel number.
$C1B9_H$	The OPEN instruction cannot be executed for the specified connection.	• Correct the connection number.
$C1BA_H$	The dedicated instruction was executed with the initialization not completed.	• Execute the dedicated instruction after the initial process is completed.
$C1BB_H$	The target station CPU type is incorrect.	• Correct the specified target station CPU type.
$C200_H$	The remote password is incorrect.	• Correct the remote password, and unlock/lock the remote password again.
$C201_H$	The remote password status of the port used for communications is in the lock status.	• After unlocking the remote password, perform communications.
$C202_H$	When another station was accessed, the remote password could not be unlocked.	• When accessing another station, do not set the remote password on the relay station or access station or do not execute the remote password check on them.
$C203_H$	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
$C204_H$	The device is different from the one requesting the remote password unlock.	• Request the lock process of the remote password from the connected device that requested the unlock process of the remote password.
$C205_H$	When another station was accessed, the remote password could not be unlocked.	• When accessing another station, do not set the remote password on the relay station or access station or do not execute the remote password check on them.

Error code	Error description	Action
C206H	System error (The OS detected an error.)	• Follow the procedure below. (1) Check if each module is properly mounted on the base unit. (2) Check if the operating environment of the Ethernet module conforms to the general specifications of the CPU module. (3) Check if the power capacity is sufficient. (4) Check if the hardware is normal according to the manual for each module. If the actions above do not solve the problem, please consult your local Mitsubishi representative.
C207H	The file name has too many characters.	• Name the file with 255 characters or less.
C300H	A response could not be received within the response monitoring timer value.	• Check the operation of the connected device. • Correct the response monitoring timer value.
C400H	Protocol not-ready error	• Check that Predefined protocol ready (X1D) has turned on before executing the ECPRTCL instruction. • Execute the ECPRTCL instruction after rewriting the protocol setting data to the Ethernet module. • If an error still occurs even after rewriting, replace the module.
C401H	Protocol unregistered error	• Correct the specified protocol number and execute the protocol again. • Register the corresponding protocol to the specified protocol number.
C402H	Protocol setting data error	• Correct the protocol setting data and register it again.
C403H	Simultaneous dedicated instructions execution error	• Do not execute dedicated instructions which do not support simultaneous execution. • Correct the specified connection number and execute the dedicated instruction again.
C404H	Protocol cancel request error	• Check the canceled protocol in the control data of the ECPRTCL instruction (execution count result) and eliminate the cause of the cancellation.
C405H	Protocol number setting error	• Correct the specified protocol number and execute the protocol again.
C406H	Continuous protocol execution count setting error	• Correct the number of protocols to be executed continuously and execute the protocol again.
C407H	Connection number setting error	• Correct the specified connection number and execute the protocol again. • Correct the connection setting of the specified connection number and execute the protocol again.
C410H	Receive wait timeout error	• Check if the cable is disconnected. • Correct the connection setting of the specified connection number and execute the protocol again. • Check that there is no error in the connected device. • Check that the sending from the connected device is not interrupted. • Check that there is no data lost due to a receive error. • Check that there is no error in the data (packet) sent by the connected device.
C411H	Packet size error	• Check the data sent by the connected device. • To send data that exceeds 2046 bytes from the connected device, divide the data into several parts and send them separately.
C417H	Data length setting error, data quantity setting error	• Check the maximum allowable data length and specify the maximum length or less in the data length storage area. • Check the maximum allowable data quantity, and specify the maximum quantity or less in the data quantity storage area.

Error code	Error description	Action
$C420_H$	Flash ROM write error	• Write the data again. • If an error still occurs even after rewriting, replace the module.
$C421_H$	Flash ROM write count over limit error	• Replace the module because the number of writes exceeded the limit.
$C430_H$	Protocol setting data writing during the ECPRTCL instruction execution	• If the ECPRTCL instruction is under execution, cancel the instruction and write the protocol setting data.
$C431_H$	Connection close during the ECPRTCL instruction execution	• Check the operation of the connected device. • Check the connection open status with the connected device. • Open the connection with the connected device again and execute the instruction.
$D000_H$ to $DFFF_H$	(Errors detected by CC-Link IE Field Network)	• Refer to the manual for each module.
$E000_H$ to $EFFF_H$	(Errors detected by CC-Link IE Controller Network)	• Refer to the manual for each module.
$F000_H$ to $FFFF_H$	(Errors detected by the MELSECNET/H and MELSECNET/10 network system)	• Refer to the MELSECNET/H or MELSECNET/10 network system reference manual, and take corrective action.

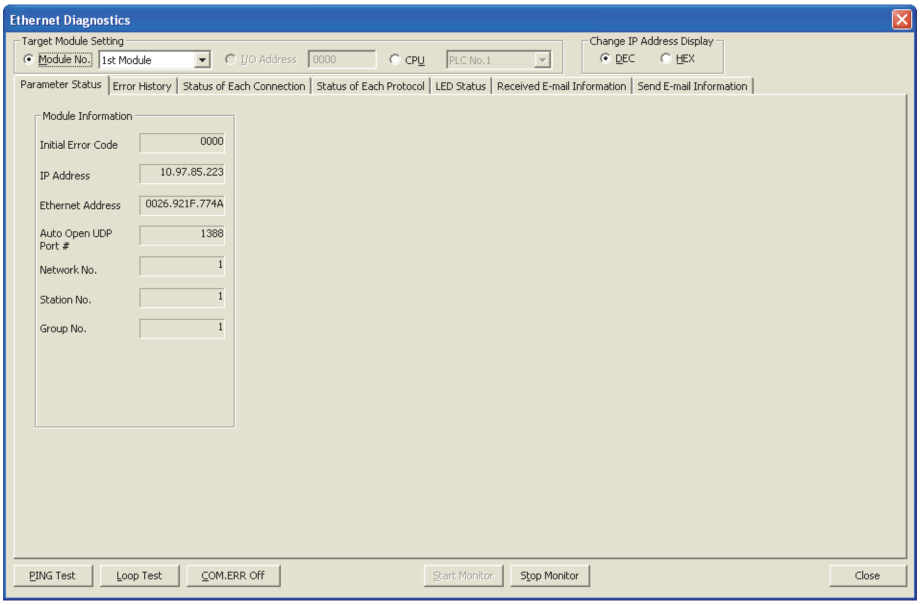
Remark

The error codes listed in the table include those returned to the connected device. Each buffer memory may also store the error numbers of the response messages returned from the connected device. If an error code other than those listed is stored, check the manual for the connected device and the response message from the connected device.

16.7 Ethernet Diagnostics

Items such as the module status, parameter settings, communication status, and error history of an E71 can be checked. Perform the diagnostics in the "Ethernet Diagnostics" window of the programming tool.

[Diagnostics] ⇨ [Ethernet Diagnostics...]



Item	Description
Target Module Setting (Module No.)	Specify the E71 to be monitored. The number of other network modules is not included.
Parameter Status	Module information (such as the IP address and basic settings) is displayed.
Error History	The number of errors and their details (such as the error code and information about the connection where the error has occurred) is displayed.
Status of Each Connection	The status of each connection (such as error codes and the open setting) is displayed.
Status of Each Protocol	Information such as the total number of data sending/receiving times for each protocol is displayed.
LED Status	The operation mode and the LED status of the E71 are displayed.
Received E-mail Information	Received mail information and the error log are displayed.
Send E-mail Information	Sent mail information and the error log are displayed.
PING Test button	The "PING Test" window is displayed.
Loop Test button	The "Loopback Test" window is displayed.
COM.ERR Off button	The COM.ERR. LED is turned off. (Page 340, Section 16.8)

Remark

For each window and detailed explanations, refer to the following.

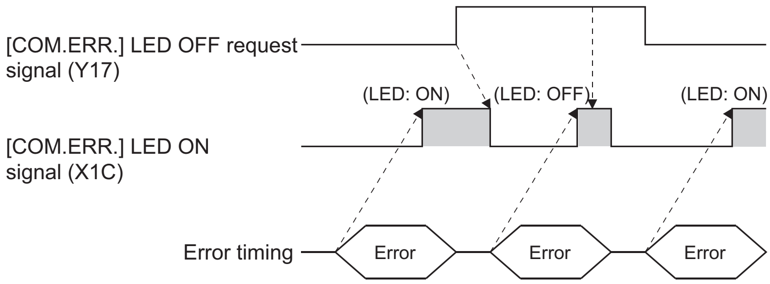
Operating manual for the programming tool used

16.8 How to Turn Off the COM.ERR. LED

The COM.ERR. LED turns on when an error occurs during communications with the connected device. The COM.ERR. LED does not turn off even after the error cause is removed. Turn it off using the following methods.

(1) Using an I/O signal

Turn off the COM.ERR. LED by turning on COM.ERR. LED OFF request signal (Y17). An off request continues to be issued while COM.ERR. LED OFF request signal (Y17) is on. However, this method does not clear the error information in the error log area in the buffer memory.



(2) Using a dedicated instruction

Turn off the COM.ERR. LED using the ERRCLR instruction. (Page 270, Section 15.11) This method can also clear the error information (initial error code, open error code) or the error log.

(3) In the "Ethernet Diagnostics" window

Turn off the COM.ERR. LED by clicking the COM.ERR Off button in the "Ethernet Diagnostics" window. However, this method does not clear the error information in the error log area in the buffer memory. For the window, refer to "Ethernet Diagnostics". (Page 339, Section 16.7)

(4) Using the MC protocol

Turn off the COM.ERR. LED using the dedicated command for the MC protocol (LED off, Error code initialization: 1617). For details, refer to the following.

MELSEC Communication Protocol Reference Manual

A

APPENDICES

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, and system configuration. Use the values obtained from the expressions as a guideline for the processing time when communications are performed in only one connection.

(1) Minimum processing time of communications using a fixed buffer (communications between the E71s)

(a) Procedure exists

Tfs = St + Ke + (Kdf × Df) + Sr

- Tfs : The time from the start of sending to the completion of sending (unit: ms)
- St : Sending station scan time
- Ke, Kdf : Constant (Refer to the table below.)
- Df : Number of send data words
- Sr : Receiving station scan time

Item	QJ71E71-100				QJ71E71-B5, QJ71E71-B2			
	TCP/IP communications		UDP/IP communications		TCP/IP communications		UDP/IP communications	
	Ke	Kdf	Ke	Kdf	Ke	Kdf	Ke	Kdf
Communications using a binary code	12	0.0065	10	0.0069	25	0.020	20	0.019
Communications using an ASCII code	12	0.030	10	0.029	25	0.068	21	0.068

(b) No procedures

Tfs = St + Ke + (Kdf × Df)

- Tfs : The time from the start of sending to the completion of sending (unit: ms)
- St : Sending station scan time
- Ke, Kdf : Constant (Refer to the table below.)
- Df : Number of send data bytes

Item	QJ71E71-100				QJ71E71-B5, QJ71E71-B2			
	TCP/IP communications		UDP/IP communications		TCP/IP communications		UDP/IP communications	
	Ke	Kdf	Ke	Kdf	Ke	Kdf	Ke	Kdf
Communications using a binary code	7	0.0018	4	0.0014	16	0.0057	9	0.0025

[Calculation example]
The time between the QJ71E71-B5 sending 1017-word data of a binary code and it completing the sending when the QJ71E71-B5 communicates with another QJ71E71-B5 in TCP/IP by the "Procedure Exist" method (unit: ms)
Assume that the scan time on the sending side is 10ms, and the scan time on the receiving side is 8ms.
63.34 (ms) ≒ 10 + 25 + (0.020 × 1017) + 8

(2) Minimum processing time of communications using a random access buffer

Trs = Kr + (Kdr × Df) + ACK processing time of a connected device (added only for TCP/IP communications)

• Trs	:	The time between the E71 receiving request data from a personal computer and it completing the process (unit: ms)
• Kr, Kdr	:	Constant (Refer to the table below.)
• Df	:	Number of request data words
• ACK processing time of the connected device	:	The time between a random access buffer reading/writing completed and the connected device returning ACK

Item		QJ71E71-100				QJ71E71-B5, QJ71E71-B2			
		TCP/IP communications		UDP/IP communications		TCP/IP communications		UDP/IP communications	
		Kr	Kdr	Kr	Kdr	Kr	Kdr	Kr	Kdr
Reading	Communications using a binary code	3.1	0.004	2.1	0.005	9.4	0.008	6.6	0.008
	Communications using an ASCII code	3.1	0.016	2.2	0.016	9.1	0.030	6.5	0.030
Writing	Communications using a binary code	3.1	0.006	2.1	0.005	9.5	0.014	6.6	0.012
	Communications using an ASCII code	3.2	0.017	2.2	0.015	9.6	0.042	6.7	0.036

[Calculation example 1]

The time between the QJ71E71-B5 receiving request data from a personal computer and it completing reading when the QJ71E71-B5 communicates with the personal computer in TCP/IP and reads 508-word data of a binary code from a random access buffer (unit: ms)

13.46 + ACK processing time of the connected device (ms) ≒ 9.4 + (0.008 × 508) + ACK processing time of the connected device

[Calculation example 2]

The time between the QJ71E71-B5 receiving request data from a personal computer and it completing writing when the QJ71E71-B5 communicates with the personal computer in TCP/IP and writes 508-word data of a binary code to a random access buffer (unit: ms)

16.61 + ACK processing time of the connected device (ms) ≒ 9.5 + (0.014 × 508) + ACK processing time of the connected device

(3) Minimum processing time of MC protocol communications (batch reading and batch writing)

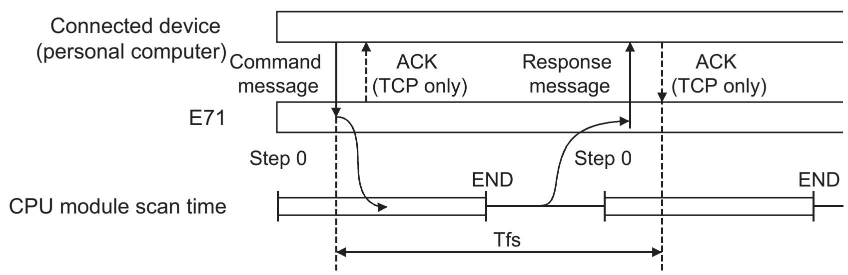
Tfs = Ke + (Kdt × Df) + Scr × number of scans required for the process + ACK processing time of the connected device

- Tfs : The time between the E71 receiving request data from a personal computer and it completing the process (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 : CPU module processing time
  (a) When the target station is the QCPU
  - Own station access: E71-mounted station scan time
  - Another station access over MELSECNET/10: Transmission delay time + E71-mounted station scan time

  (b) When the target station is in a redundant system and data is sent through a tracking cable*2
  - Own station access: Control system CPU module scan time + tracking transfer time
  - Another station access over MELSECNET/10: Transmission delay time + control system CPU module scan time + tracking transfer time

Item		QJ71E71-100				QJ71E71-B5, QJ71E71-B2			
		TCP/IP communications		UDP/IP communications		TCP/IP communications		UDP/IP communications	
		Ke	Kdt	Ke	Kdt	Ke	Kdt	Ke	Kdt
Batch reading	Communications using a binary code	14	0.009	13	0.008	21	0.012	19	0.011
	Communications using an ASCII code	18	0.015	13	0.017	23	0.020	18	0.020
Batch writing	Communications using a binary code	14	0.009	13	0.008	21	0.020	19	0.013
	Communications using an ASCII code	16	0.027	14	0.027	22	0.037	20	0.033

*1 The time from when the E71 receives the request data from the personal computer until it completes the process is shown below.



*2 When data is sent through the tracking cable of a redundant system, add the tracking transfer time. For data transfer time of when the tracking cable is used, refer to the following.

QnPRHCPU User's Manual (Redundant System)

[Calculation example 1]
The time between the QJ71E71-B5 receiving request data from a personal computer and it completing reading when the QJ71E71-B5 communicates with the personal computer in TCP/IP and reads 100-point data of an ASCII code from the data register (D) in the own station (unit: ms)
Assume that the scan time of the QJ71E71-B5-mounted station is 10ms, the command data length is 21 words, and the response data length is 211 words.
37.64 + ACK processing time of the connected device (ms) ≒ 23 + (0.020 × (21 + 211)) + 10 × 1 + ACK processing time of the connected device

[Calculation example 2]
The time between the QJ71E71-B5 receiving request data from a personal computer and it completing writing when the QJ71E71-B5 communicates with the personal computer in TCP/IP and writes 100-point data of an ASCII code to the data register (D) in the own station (unit: ms)
Assume that the scan time of the QJ71E71-B5-mounted station is 10ms, the command data length is 221 words, the response data length is 11 words, and writing during RUN is enabled.
40.58 (ms) ≒ 22 + (0.037 × (221 + 11)) + 10 × 1

A

(4) Processing time of the dedicated instructions

The following table lists approximate operation processing time of each dedicated instruction. The operation processing time differs depending on the system configuration and the scan time on the sending/receiving stations.

(a) QJ71E71-100

<table>
<tr><th rowspan="3">Instruction name</th><th colspan="2" rowspan="2">Number of access points</th><th colspan="8">Processing time (unit: ms)</th><th rowspan="3">Instruction execution condition</th></tr>
<tr><th colspan="2">Basic model QCPU</th><th colspan="2">Q02CPU</th><th colspan="2">QnHCPU, Process CPU, Redundant CPU</th><th colspan="2">Universal model QCPU</th></tr>
<tr><th>Condition 1)</th><th>Condition 2)</th><th>For 1)</th><th>For 2)</th><th>For 1)</th><th>For 2)</th><th>For 1)</th><th>For 2)</th><th>For 1)</th><th>For 2)</th></tr>
<tr><td>OPEN</td><td colspan="2" rowspan="2">1 port</td><td colspan="2">4.2</td><td colspan="2">3.8</td><td colspan="2">3.0</td><td colspan="2">2.9</td><td>A UDP/IP transmission port is open.</td></tr>
<tr><td>CLOSE</td><td colspan="2">4.2</td><td colspan="2">3.3</td><td colspan="2">3.2</td><td colspan="2">3.2</td><td>A UDP/IP transmission port is closed.</td></tr>
<tr><td>BUFSND</td><td rowspan="3">1 word</td><td rowspan="3">1017 words</td><td>14.0</td><td>23.5</td><td>12.8</td><td>19.2</td><td>11.5</td><td>18.1</td><td>8.2</td><td>15.7</td><td rowspan="3">TCP/IP communications, binary code communications, and fixed buffer communications (procedure exists)</td></tr>
<tr><td>BUFRCV</td><td>2.2</td><td>5.8</td><td>1.3</td><td>1.8</td><td>0.9</td><td>1.4</td><td>0.7</td><td>1.1</td></tr>
<tr><td>BUFRCVS</td><td>0.8</td><td>2.9</td><td>0.5</td><td>0.9</td><td>0.3</td><td>0.7</td><td>0.2</td><td>0.6</td></tr>
<tr><td>ERRCLR</td><td colspan="2">All error information is cleared.</td><td colspan="2">3.4</td><td colspan="2">2.2</td><td colspan="2">2.0</td><td colspan="2">1.7</td><td>-</td></tr>
<tr><td>ERRRD</td><td colspan="2">An initial error code is read.</td><td colspan="2">2.4</td><td colspan="2">1.2</td><td colspan="2">0.8</td><td colspan="2">0.7</td><td>-</td></tr>
<tr><td rowspan="2">SEND</td><td rowspan="12">1 word</td><td>960 words</td><td rowspan="2">11.5</td><td>16.4</td><td rowspan="2">7.9</td><td>15.7</td><td rowspan="2">7.5</td><td>15.4</td><td rowspan="2">7.5</td><td>15.4</td><td rowspan="12">Communications between E71-mounted stations</td></tr>
<tr><td>480 words</td><td>16.8</td><td>11.2</td><td>10.8</td><td>10.8</td></tr>
<tr><td rowspan="2">RECV</td><td>960 words</td><td rowspan="2">1.8</td><td>6.8</td><td rowspan="2">2.1</td><td>4.3</td><td rowspan="2">2.0</td><td>3.8</td><td rowspan="2">2.0</td><td>3.8</td></tr>
<tr><td>480 words</td><td>4.3</td><td>3.2</td><td>2.9</td><td>2.9</td></tr>
<tr><td rowspan="2">RECVS</td><td>960 words</td><td rowspan="2">0.8</td><td>1.5</td><td rowspan="2">0.6</td><td>0.9</td><td rowspan="2">0.3</td><td>0.7</td><td rowspan="2">0.2</td><td>0.6</td></tr>
<tr><td>480 words</td><td>1.2</td><td>0.8</td><td>0.5</td><td>0.4</td></tr>
<tr><td rowspan="2">READ, SREAD</td><td>960 words</td><td rowspan="2">14.7</td><td>24.3</td><td rowspan="2">17.2</td><td>28.8</td><td rowspan="2">17.1[*1]</td><td>28.2[*1]</td><td rowspan="2">9.9</td><td>19.2</td></tr>
<tr><td>480 words</td><td>20.9</td><td>22.7</td><td>21.7[*1]</td><td>13.9</td></tr>
<tr><td rowspan="2">WRITE, SWRITE</td><td>960 words</td><td rowspan="2">14.5</td><td>24.4</td><td rowspan="2">17.3</td><td>28.8</td><td rowspan="2">17.0[*1]</td><td>28.4[*1]</td><td rowspan="2">9.7</td><td>18.9</td></tr>
<tr><td>480 words</td><td>19.8</td><td>23.0</td><td>22.2[*1]</td><td>13.7</td></tr>
<tr><td>ZNRD</td><td rowspan="2">230 words</td><td>12.1</td><td>14.1</td><td>14.4</td><td>17.1</td><td>13.8</td><td>16.6</td><td>9.8</td><td>11.8</td></tr>
<tr><td>ZNWR</td><td>12.0</td><td>14.8</td><td>14.2</td><td>17.5</td><td>13.9</td><td>16.4</td><td>9.8</td><td>12.0</td></tr>
<tr><td>UINI</td><td colspan="2">-</td><td colspan="2">26.9</td><td colspan="2">26.7</td><td colspan="2">26.7</td><td colspan="2">26.7</td><td>Time between the UINI instruction accepted and a reinitialization process completed</td></tr>
<tr><td>ECPRTCL</td><td colspan="10">*2</td><td>-</td></tr>
</table>

*1 When data is sent through the tracking cable of a redundant system, add the tracking transfer time. For data transfer time of when the tracking cable is used, refer to the following.

QnPRHCPU User's Manual (Redundant System)

*2 The processing time of the ECPRTCL instruction differs depending on a connected device or a protocol.

(b) QJ71E71-B5, QJ71E71-B2

Instruction name	Number of access points		Processing time (unit: ms)								Instruction execution condition
			Basic model QCPU		Q02CPU		QnHCPU, Process CPU, Redundant CPU		Universal model QCPU		
	Condition 1)	Condition 2)	For 1)	For 2)	For 1)	For 2)	For 1)	For 2)	For 1)	For 2)	
OPEN	1 port		5.2		4.3		3.3		3.0		A UDP/IP transmission port is open.
CLOSE	1 port		6.0		4.5		4.5		2.9		A UDP/IP transmission port is closed.
BUFSND	1 word	1017 words	28.2	50.0	27.6	45.3	24.5	45.0	12.9	23.7	TCP/IP communications, binary code communications, and fixed-buffer communications (procedure exists)
BUFRCV	1 word	1017 words	2.3	5.8	1.9	2.4	1.2	1.6	0.7	1.1	
BUFRCVS	1 word	1017 words	0.8	2.9	0.5	0.9	0.3	0.7	0.2	0.6	
ERRCLR	All error information is cleared.		3.4		2.7		2.2		1.8		-
ERRRD	An initial error code is read.		2.5		1.7		1.1		0.7		-
SEND	1 word	960 words	22.8	38.8	21.4	39.4	20.3	37.9	11.0	22.3	Communications between E71-mounted stations
		480 words		30.8		30.4		29.1		16.1	
RECV		960 words	5.3	11.1	5.3	7.9	5.2	7.4	2.4	4.4	
		480 words		8.2		6.6		6.3		3.3	
RECVS		960 words	0.8	1.6	0.6	1.0	0.3	0.7	0.2	0.6	
		480 words		1.2		0.8		0.5		0.4	
READ, SREAD		960 words	27.7	50.1	30.1	52.1	27.7*1	52.3*1	12.5	25.9	
		480 words		38.9		41.1		40.0*1		18.3	
WRITE, SWRITE		960 words	28.2	47.6	30.0	53.6	29.4*1	52.4*1	12.9	25.4	
		480 words		37.9		41.8		40.9*1		18.2	
ZNRD		230 words	27.8	33.2	29.0	34.3	29.0	34.7	12.6	15.3	
ZNWR		230 words	27.6	33.4	29.7	36.4	29.4	35.2	12.9	15.6	
UINI	-		26.8		26.7		26.7		26.7		Time between the UINI instruction accepted and a reinitialization process completed

*1 When data is sent through the tracking cable of a redundant system, add the tracking transfer time. For data transfer time of when the tracking cable is used, refer to the following.

QnPRHCPU User's Manual (Redundant System)

(5) System switching time of a redundant system

The following shows the system switching time required when the E71 mounted with the control system CPU module in a redundant system issues a system switching request to the control system CPU module at communication error or disconnection detection. The system switching time is the time from when a communication error or disconnection is detected until the control system CPU module is switched to the standby system CPU module.

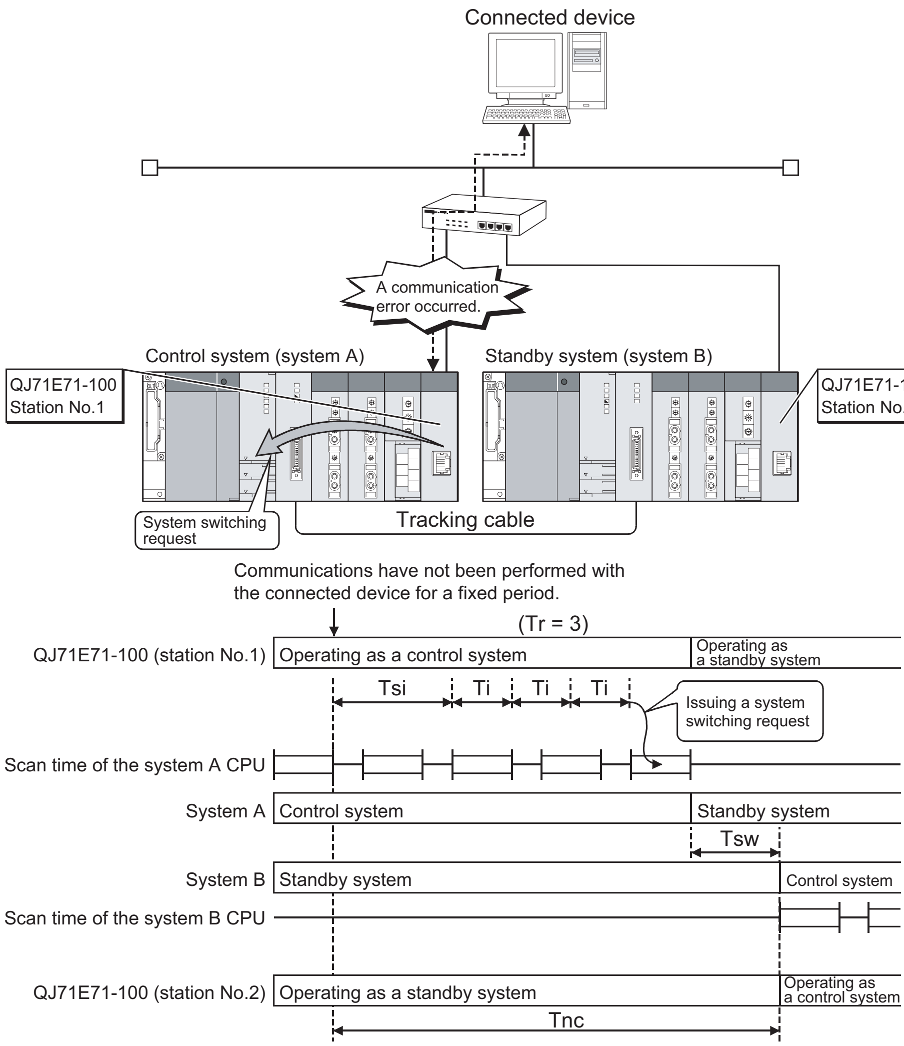
(a) When a communication error is detected

- For alive check errors

  Tnc = Tsi + Ti × Tr + St + Tsw

  - Tnc : System switching time
  - Tsi : Destination existence confirmation starting interval timer value
  - Ti : Timer value of the alive check
  - Tr : Destination existence confirmation resending time
  - St : One scan time
  - Tsw : CPU system switching time*1

The following shows the timing for system switching operation if an alive check error occurs.



*1 For details on CPU system switching time, refer to the following.

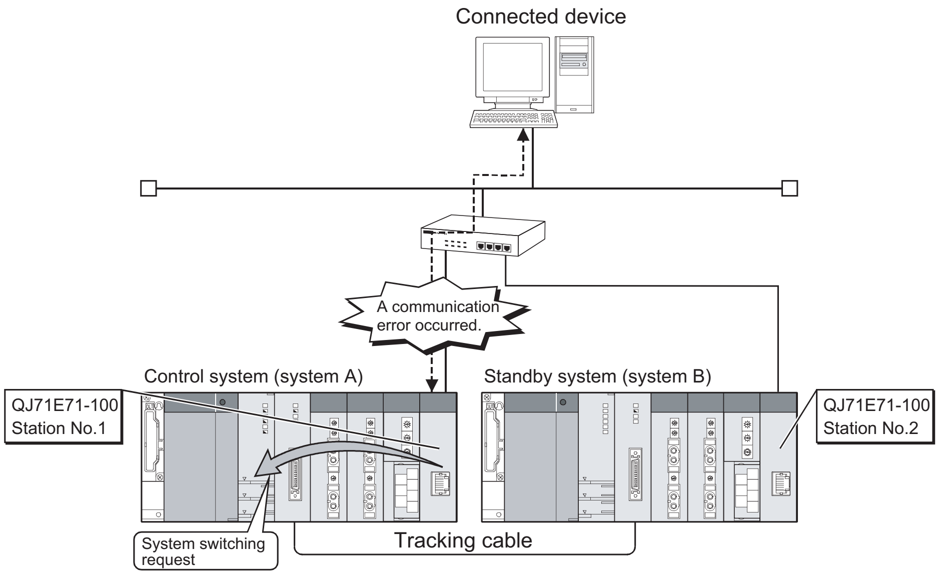
QnPRHCPU User's Manual (Redundant System)

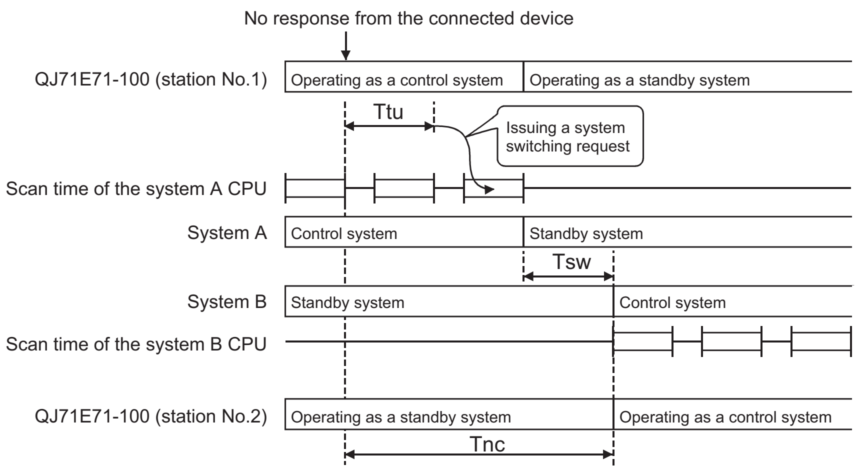
- When a ULP timeout occurs

  Tnc = Ttu + St + Tsw

  - Tnc : System switching time
  - Ttu : TCP ULP timer value
  - St : One scan time
  - Tsw : CPU system switching time*1

The following shows the system switching operation timing if a ULP timeout occurs.





*1 For details on CPU system switching time, refer to the following.

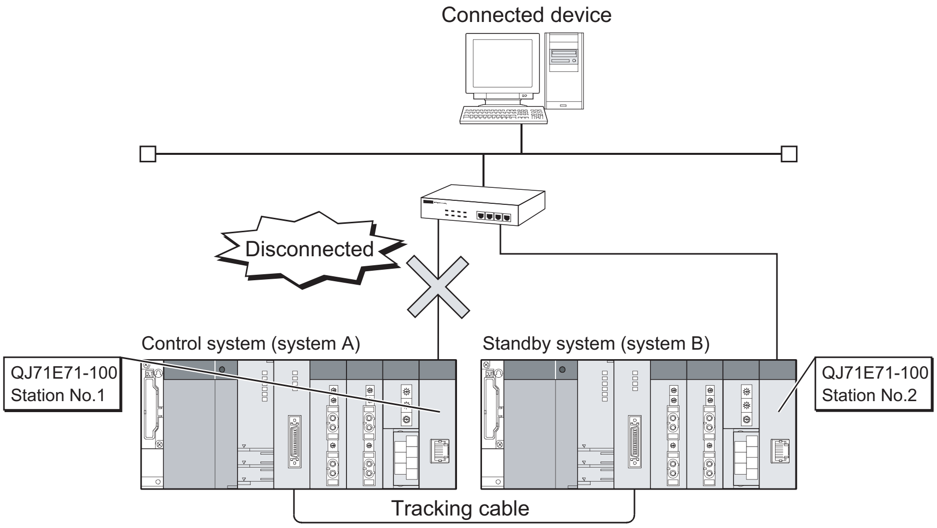
QnPRHCPU User's Manual (Redundant System)

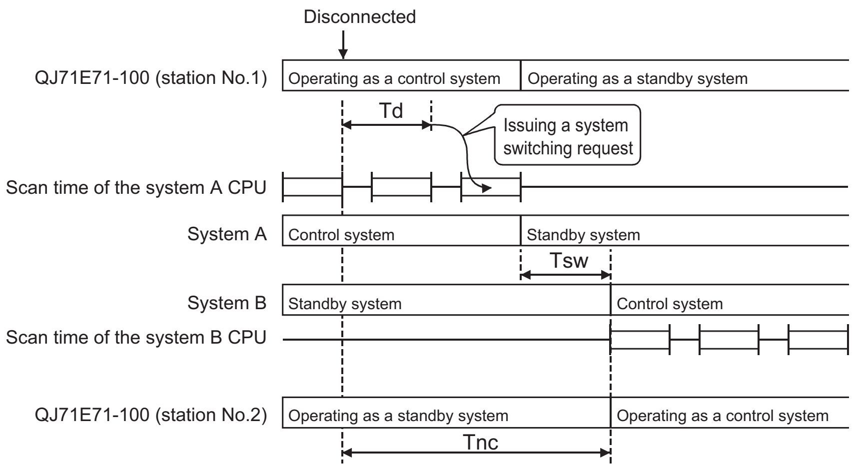
(b) When disconnection is detected

Tnc = Td + St + Tsw

- Tnc : System switching time
- Td : Disconnection detection monitoring time
- St : One scan time
- Tsw : CPU system switching time[*1]

The following shows the system switching operation timing if disconnection is detected.





*1 For details on CPU system switching time, refer to the following.
QnPRHCPU User's Manual (Redundant System)

Appendix 2 Port Numbers Used for the E71

The following table lists port numbers reserved for an E71 system.

Port number	Application
1388_H (5000)	Auto-open UDP port (Default value)*1
1389_H (5001)	MELSOFT application transmission port (UDP)
$138A_H$ (5002)	MELSOFT application transmission port (TCP)

*1 The port number can be changed.

Ex. The port number can be changed by overwriting the buffer memory address, Automatic open UDP port number (address: 14_H).

Appendix 3 New and Improved Functions

The following table lists the new and improved functions in the E71 and supported versions of modules and programming tool.

[How to see the table]

- Numbers in quotation marks in the table represent the first five digits of the serial number of each module.

○: Available (No version restriction), ×: Not available

Function		Version of E71			Version of CPU module	Version of GX Developer	Version of GX Works2	Reference
		QJ71E71-100	QJ71E71-B5	QJ71E71-B2				
Support for IEEE 802.3 frame		○	○	Function version B or later of "03102"	○	Version 7 or later	○	Page 85, Section 7.1.3
Reinitialization process of the E71	Reinitialization process with the program	Function version B or later of "03102"	○	Function version B or later of "03061"	○	○	○	Page 358, Appendix 4.2
	Reinitialization process with the UINI instruction			Function version B or later of "03102"				Page 277, Section 15.13
	Changing the transmission speed and the communication mode with the UINI instruction	Function version D or later of "11012"	×	×				Page 277, Section 15.13
	TCP Maximum Segment transmission	Function version B or later of "05051"	○	Function version B or later of "05051"				Page 358, Appendix 4.2
Re-open process of the E71[*1]		Function version B or later of "05051"	○	Function version B or later of "05051"	○	○	○	Page 91, Section 7.2.3
Alive check function	Check using KeepAlive	Function version B or later of "05051"	○	Function version B or later of "05051"	○	Version 8.05F or later	○	Page 85, Section 7.1.3
Connection up to 17 MELSOFT products in TCP/IP communications		○	○	Function version B or later of "02122"	○	Version 6.05F or later	○	Page 100, CHAPTER 8
Easy connection with MELSOFT products	Easy access to other stations	Function version B or later of "05051"	○	Function version B or later of "05051"	○	○	○	Page 100, CHAPTER 8
	Access to the station with the same station number							
Ethernet diagnostics	Monitoring of various E71 status	○	○	Function version B or later	Function version A or later of "02092"	Version 6 or later	○	Page 283, CHAPTER 16
	PING test/loopback test via Ethernet board							Page 364, Appendix 5
	PING test through a CPU module					Version 7 or later	○	Page 367, Appendix 5.1 (2)

Function		Version of E71			Version of CPU module	Version of GX Developer	Version of GX Works2	Reference
		QJ71E71-100	QJ71E71-B5	QJ71E71-B2				
Specification of station numbers 65 to 120 for data link instructions (for access to CC-Link IE Controller Network and CC-Link IE Field Network)		Function version D or later of "09042"	×	×	Universal model QCPU with function version B or later of "09042"	○	○	MELSEC-Q/L Ethernet Interface Module User's Manual (Application)
Target station CPU type specification for data link instructions		Function version D or later	Function version D or later	Function version D or later	○	○	○	
Expansion of data length for data link instructions (from 480 to 960 words)		Function version D or later of "07082"	Function version D or later of "07082"	Function version D or later of "07082"	○	○	○	
E-mail function	Sending an attachment as a CSV file	○	○	Function version B or later	○	Version 6 or later	○	MELSEC-Q/L Ethernet Interface Module User's Manual (Application)
	Sending the main text	○	○	Function version B or later of "03102"		Version 7 or later	○	
	Support for encoding/ decoding	Function version B or later of "03102"	○	Function version B or later of "03102"		○	○	
	Sending character strings as the main text in an e-mail using the programmable controller CPU monitoring function	Function version D or later of "07082"	Function version D or later of "07082"	Function version D or later of "07082"		Version 8.27D or later	○	
Support of a multiple CPU system by the file transfer (FTP server) function		○	○	Function version B or later of "03102"	Function version B or later	○	○	
Support of 4E frame data in MC protocol communications		Function version D or later of "07082"	Function version D or later of "07082"	Function version D or later of "07082"	○	○	○	
Access to the link direct device of LW10000 or later in MC protocol communications (4E frame and QnA-compatible 3E frame only)		Function version D or later of "09042"	×	×	Universal model QCPU with function version B or later of "09042"	○	○	
Access to the extended data register of D65536 or later and the extended link register of W10000 or later in MC protocol communications (4E frame and QnA-compatible 3E frame only)		Function version D or later of "09042"	×	×	Universal model QCPU with function version B or later of "09042"	○	○	
Web function		○	○	Function version B or later of "05051"	○	○	○	
Remote password check		○	○	Function version B or later	Function Version A or later of "02092"	Version 6 or later	○	Page 201, Section 14.4
Hub connection status monitor function		Function version D or later	×	×	○	○	○	Page 211, Section 14.5

<table>
<tr><th rowspan="2">Function</th><th colspan="3">Version of E71</th><th rowspan="2">Version of CPU module</th><th rowspan="2">Version of GX Developer</th><th rowspan="2">Version of GX Works2</th><th rowspan="2">Reference</th></tr>
<tr><th>QJ71E71-100</th><th>QJ71E71-B5</th><th>QJ71E71-B2</th></tr>
<tr><td>IP address in use detection function</td><td>Function version D or later of "12062"</td><td>×</td><td>×</td><td>○</td><td>○</td><td>○</td><td>Page 212, Section 14.6</td></tr>
<tr><td>Setting the arrival monitoring time of the dedicated instruction in 100ms increments (the READ, SREAD, WRITE, and SWRITE instructions only)</td><td>Function version D or later of "12062"</td><td>×</td><td>×</td><td>○</td><td>○</td><td>○</td><td>MELSEC-Q/L Ethernet Interface Module User's Manual (Application)</td></tr>
<tr><td>Communications using SLMP</td><td>Function version D or later of "15042"</td><td>×</td><td>×</td><td>○</td><td>×</td><td>○</td><td>Page 107, CHAPTER 10</td></tr>
<tr><td>Data communications using the predefined protocol</td><td>Function version D or later of "15042"</td><td>×</td><td>×</td><td>○</td><td>×</td><td rowspan="3">Version 1.492N or later</td><td>Page 112, CHAPTER 11</td></tr>
<tr><td>Module error collection function</td><td>Function version D or later of "15042"</td><td>×</td><td>×</td><td>Universal model QCPU with function version B or later of "11043"</td><td>×</td><td>Page 285, Section 16.3</td></tr>
<tr><td>Setting range extension of the communication destination port number (Port number 1_H to 400_H can be set.)</td><td>Function version D or later of "15042"</td><td>×</td><td>×</td><td>○</td><td>×</td><td>Page 87, Section 7.1.4</td></tr>
<tr><td>Receive buffer full detection signal (address: 5240_H)</td><td>Function version D or later of "17032"</td><td>×</td><td>×</td><td>○</td><td>○</td><td>○</td><td>Page 38, Section 3.5.2</td></tr>
<tr><td>IP filter function</td><td>Function version D or later of "18072"</td><td>Function version D or later of "18072"</td><td>Function version D or later of "18072"</td><td>○</td><td>○</td><td>○</td><td>Page 195, Section 14.3</td></tr>
</table>

*1 The operations of the E71 has been changed for the case where an Active open request from the connected device is received again during open completion status in TCP/IP.

Appendix 4 Initial Process

The initial process is for enabling data communications with connected devices by setting the minimum number of parameters for the E71. For the Q Series, a program for the initial process is not required because the initial process is automatically performed.

(1) Checking the initial process result

The results of the initial process can be checked using the LEDs and I/O signals.

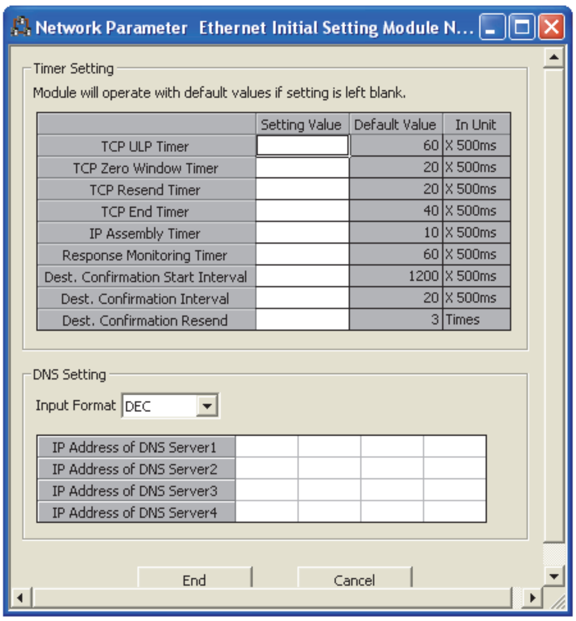
Initial process	INIT. LED	Initial normal completion signal (X19)	Initial abnormal end signal (X1A)
At normal completion	ON	ON	OFF
At abnormal end	OFF	OFF	ON

If the initial process is not normally completed, correct the set values of parameters listed in Chapter 7 and those in the initial process setting parameter described in this chapter. After correcting the parameters, write them to the CPU module again, power off and on the CPU module or reset the CPU module.

Appendix 4.1 Setting the initial process

Configure the setting in the initial setting window. Each setting time is calculated by "setting value × 500ms".

Project window ➪ [Parameter] ➪ [Network Parameter] ➪ [Ethernet/CC IE/MELSECNET] ➪ Select "Ethernet" under "Network Type". ➪ Initial Setting



Item		Description	Setting range
Timer Setting	TCP ULP Timer	Set Time to Live during packet sending in TCP/IP.	2 to 32767
	TCP Zero Window Timer	Window indicates the receive buffer on the recipient. If the receive buffer on the recipient has no free space (window size = 0), data sending is waited until the recipient has free receive buffer space. At this time, the sender sends packets for checking the sending window to the recipient according to the TCP zero window timer value to check the receiving status.	2 to 32767
	TCP Resend Timer	Set the time to resend ACK if it is not returned when a connection is opened or data is sent in TCP/IP. This timer is also used for Time to Live for the ARP function. (ARP is resent by the time of "TCP resend timer value/2" if no response to an ARP request is returned.) This timer is also used as the minimum set time for arrival monitoring time for data link instructions.	2 to 32767
	TCP End Timer	When closing the TCP/IP connection by the own station, set the monitoring time for waiting for a FIN from the connected device after the own station sends a FIN and an ACK is returned from the connected device. If a FIN is not received from the connected device within the time specified by the TCP end timer value, a RST is sent to the connected device and the connection is forcibly closed.	2 to 32767
	IP Assembly Timer	The communication data may be divided at the IP level before being sent due to the buffer limitation of the sending station or the receiving station. Set the waiting time for the divided data in such a case.	1 to 32766
	Response Monitoring Timer	Set the following waiting time. • Time for waiting a response after a command is sent • For receiving divided messages, time until the last message is received after the first message is received	2 to 32767
	Dest. Confirmation Start Interval	Set the time until starting the alive check after communications with the connected device of the connection that has been opened where the alive check target exists is lost.	1 to 32767
	Dest. Confirmation Interval	For the connection that is opened where the alive check target exists, set the interval to perform the alive check again when a response from the connected device as the alive check target is not received.	1 to 32767
	Dest. Confirmation Resend	For the connection that has been opened where the alive check target exists, set the number of times that the alive check is performed again when a response from the connected device as the alive check target is not received.	1 to 32767
DNS Setting*1	Input Format	Select an input format.	DEC/HEX
	IP Address of DNS Server1	Set the IP address of each DNS server.	-
	IP Address of DNS Server2		
	IP Address of DNS Server3		
	IP Address of DNS Server4		

*1 The DNS setting is configured for using the e-mail function. For details, refer to the following.

MELSEC-Q/L Ethernet Interface Module User's Manual (Application)

(1) Precautions for settings

- Specify the setting value of each timer on the E71 side so that the following formula is met.

$$\left(\text{Response monitoring timer value}\right) \geqq \left(\text{TCP ULP timer value}\right) \geqq \left(\text{TCP end timer value}\right) \geqq \left(\text{TCP resend timer value}\right) > \left(\text{IP assembly timer value}\right)$$

$$\left(\text{TCP resend timer value}\right) = \left(\text{TCP zero window timer value}\right)$$

When connecting Mitsubishi products to the line, configure the same settings for both modules.

- Specify the setting value of each timer on the connected device side so that the following formula is met. The frequency of a communication error, such as a transmission timeout, may be higher if the timer values do not meet the formula.

$$\left(\text{TCP ULP timer value on the connected device}\right) > \left(\text{TCP resend timer value on the E71}\right)$$

$$\left(\text{Monitoring timer value of the application software on the connected device}\right) > \left\{\left(\text{TCP ULP timer value on the E71}\right) \times n^{*1}\right\}$$

*1 "n" is the number of TCP segment transmission and is calculated by the following formula.

$$n = \text{A value that } \left(\frac{\text{Message size sent by the E71}}{\text{Maximum Segment Size}}\right) \text{ is rounded up to the nearest integer}$$

Ex. The number of TCP segment transmission when communications are performed on the same line
The Maximum Segment Size is 1460 bytes on the same line (without a router) and the number of TCP Segment transmission is as follows.

- n = 1 when the size of the message sent by the E71 is 1460 bytes or less
- n = 2 when the size of the message sent by the E71 is greater than 1460 bytes

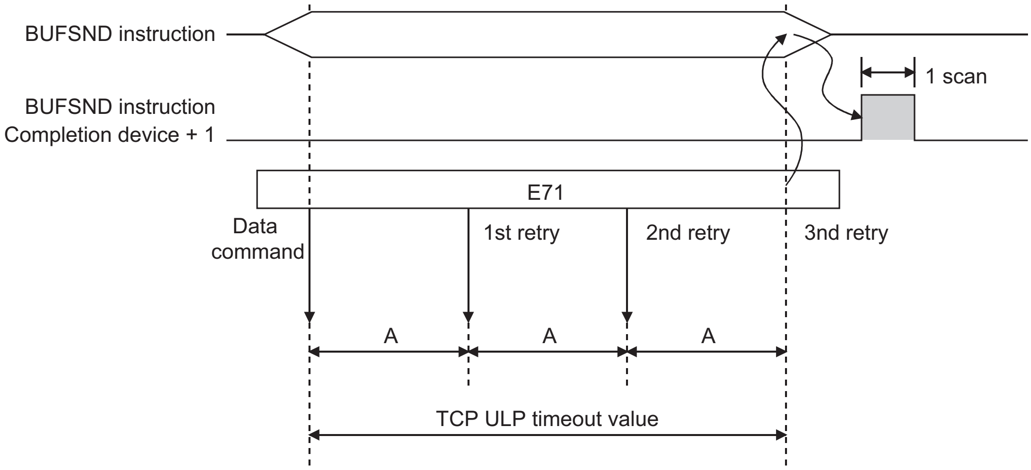
Ex. The number of TCP segment transmission when communications are performed on a different line
The Maximum Segment Size is at least 536 bytes on another line (via a dialup router or other communication device) and the number of TCP Segment transmission is as follows.

- n = 1 when the size of the message sent by the E71 is 536 bytes or less
- n = 2 when the size of the message sent by the E71 is greater than 536 bytes and no more than 1072 bytes
- n = 3 when the size of the message sent by the E71 is greater than 1072 bytes and no more than 1608 bytes

- When a communication failure occurs due to a problem such as noise, change the value so that the number of retries may increase. The number of retries is obtained using the following formula. (When using the default values, 3 = (60 ÷ 20))

Number of retries = TCP ULP timer value ÷ TCP resend timer value

Ex. When the number of retries is three and data sending fails, a data sending error occurs at the timing shown in the figure below (in communications using the fixed buffer).



A: TCP resend timer value
(The time at which data are sent when ACK is not returned after data transmission)

- When not performing the above retry process (set 0 times), configure the setting so that the following formula is met.

(TCP ULP timer value) = (TCP end timer value) = (TCP resend timer value)

(Set the same value for the timer values.)

Appendix 4.2 Reinitialization process

The reinitialization process sets the E71 to the status after startup without restarting the programmable controller (or resetting the CPU module). Perform the reinitialization process using a program.

(1) Purpose of a reinitialization process

The purpose of a reinitialization process is described.

(a) To update address information of a connected device held by the E71

To prevent other devices from illegally accessing the programmable controller using the IP address of the connected device with which communications are normally performed, the E71 holds the IP address of the connected device and the corresponding MAC address.*1 If a module or the board of the connected device is replaced due to a failure, the address information of the connected device held by the E71 must be cleared by the reinitialization process.

*1 MAC address is unique to a device. No devices share the same MAC address.

(b) To change the IP address of the E71 on the own station

When a system is changed, by only changing the own station IP address set by the programming tool, communications with the connected device can be resumed.

(c) To change the Ethernet operation setting

By changing the communication conditions in the Ethernet operation setting set using the programming tool, communications with the connected device can be resumed.

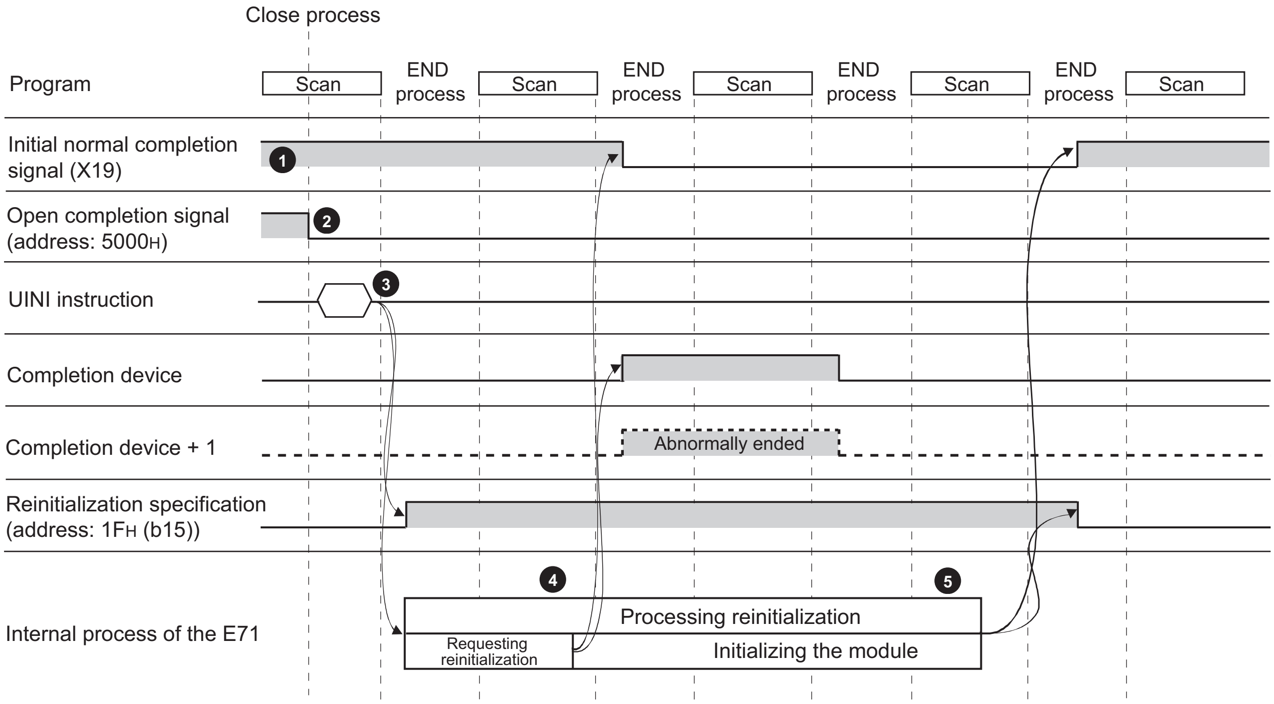
(d) To change the transmission speed and communication mode

By changing the transmission speed and communication mode, communications with the connected device can be resumed.

(2) Reinitialization processing program

Perform the reinitialization process using a program. The following shows the process timing and sample programs.

(a) Process timing



1. Check the normal completion of the initial process. (Initial normal completion signal (X19): ON)
2. Terminate all data communications with the connected devices and perform a close process on all connections.
3. Execute the UINI instruction. Specify the parameters (such as the IP address of the own station and Ethernet operation setting) using the control data of the dedicated instruction and initialize the E71.
4. When the E71 reinitialization request is completed, Initial normal completion signal (X19) turns off.
5. When the reinitialization processes are all completed, Reinitialization specification (address: $1F_H$ (b15) turns "0" and Initial normal completion signal (X19) turns on. If the reinitialization process fails, an error code is stored in the initial error code storage area.

(b) Sample program

The following two methods are available:

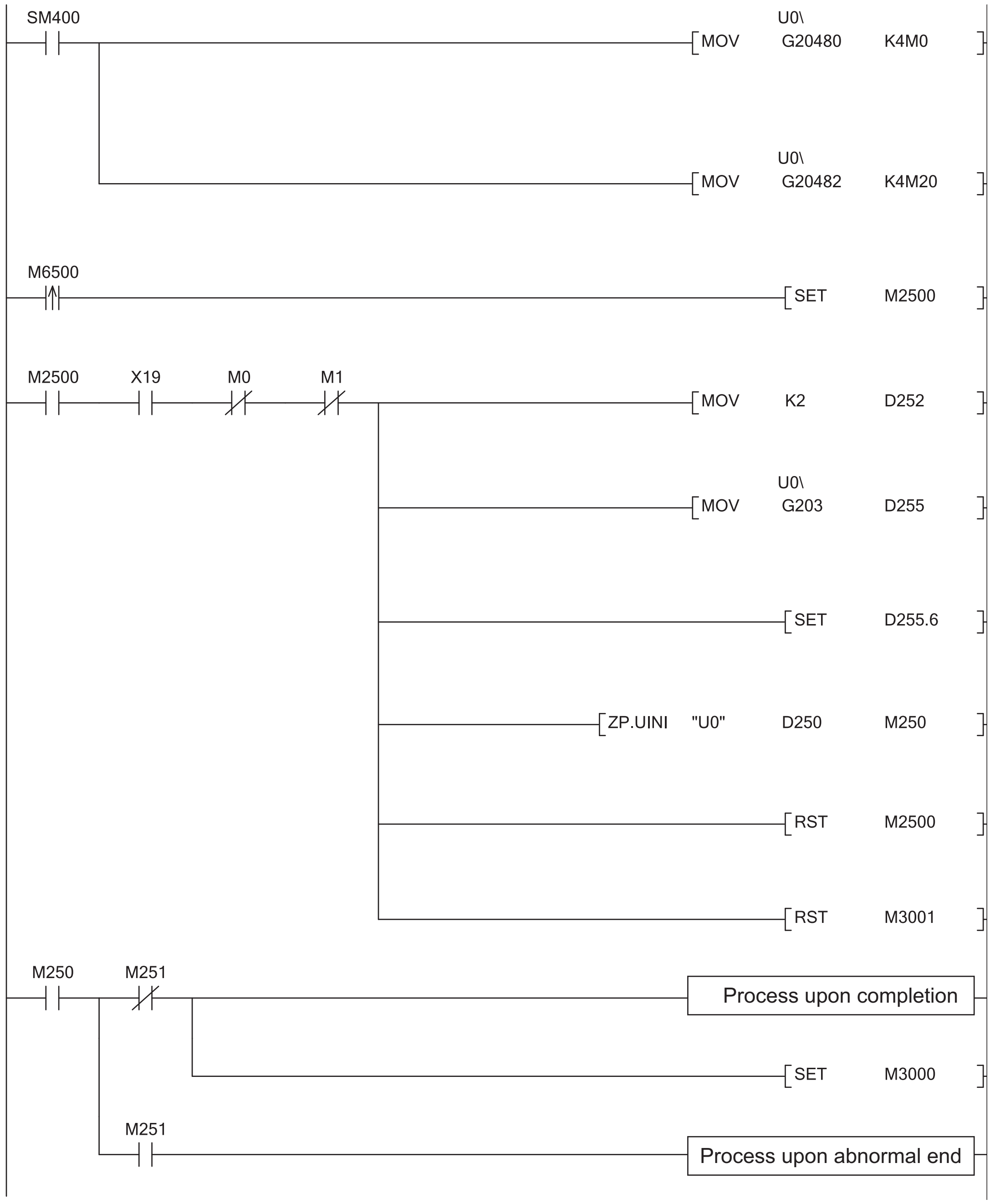
- Using the dedicated instruction (UINI instruction)
- Writing directly to the buffer memory

Point

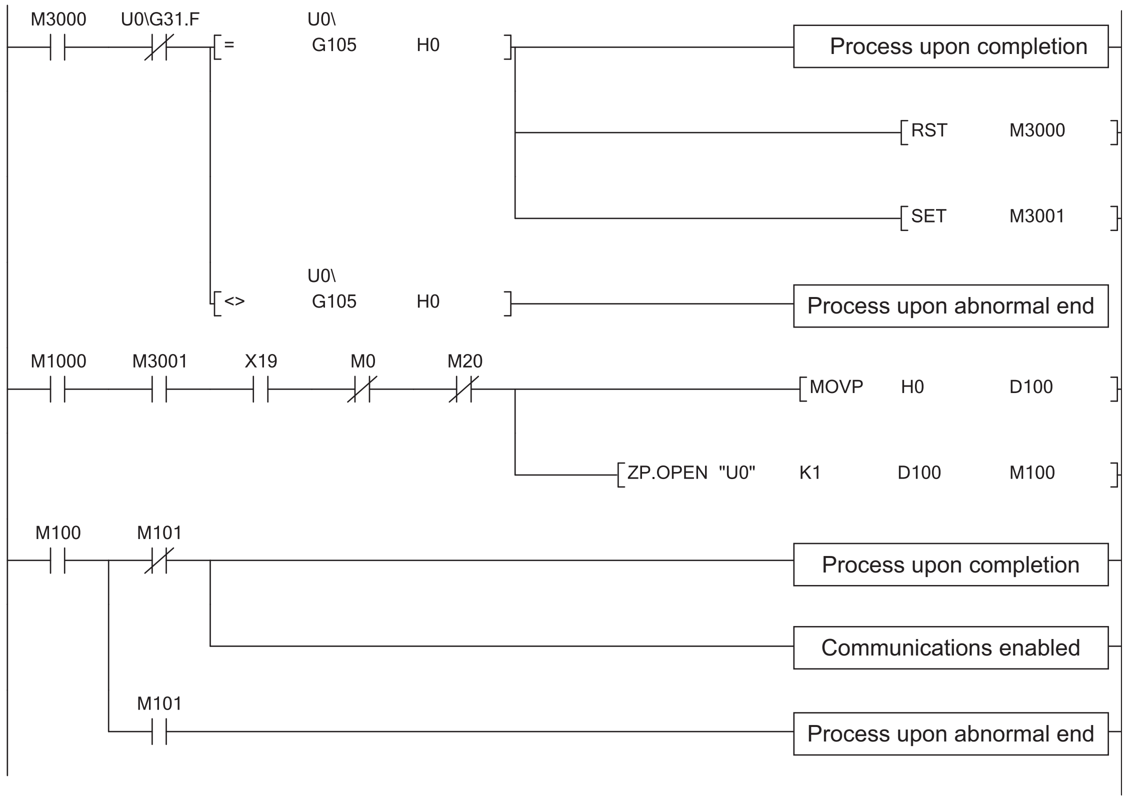
Do not use the method using the dedicated instruction (UINI instruction) and the method by writing directly to the buffer memory together.

(c) Using the dedicated instruction (UINI instruction)

The following shows an example where the reinitialization process is performed using the UINI instruction and the OPEN instruction is executed after the process is completed. (When the I/O signals of the E71 are X/Y00 to X/Y1F)

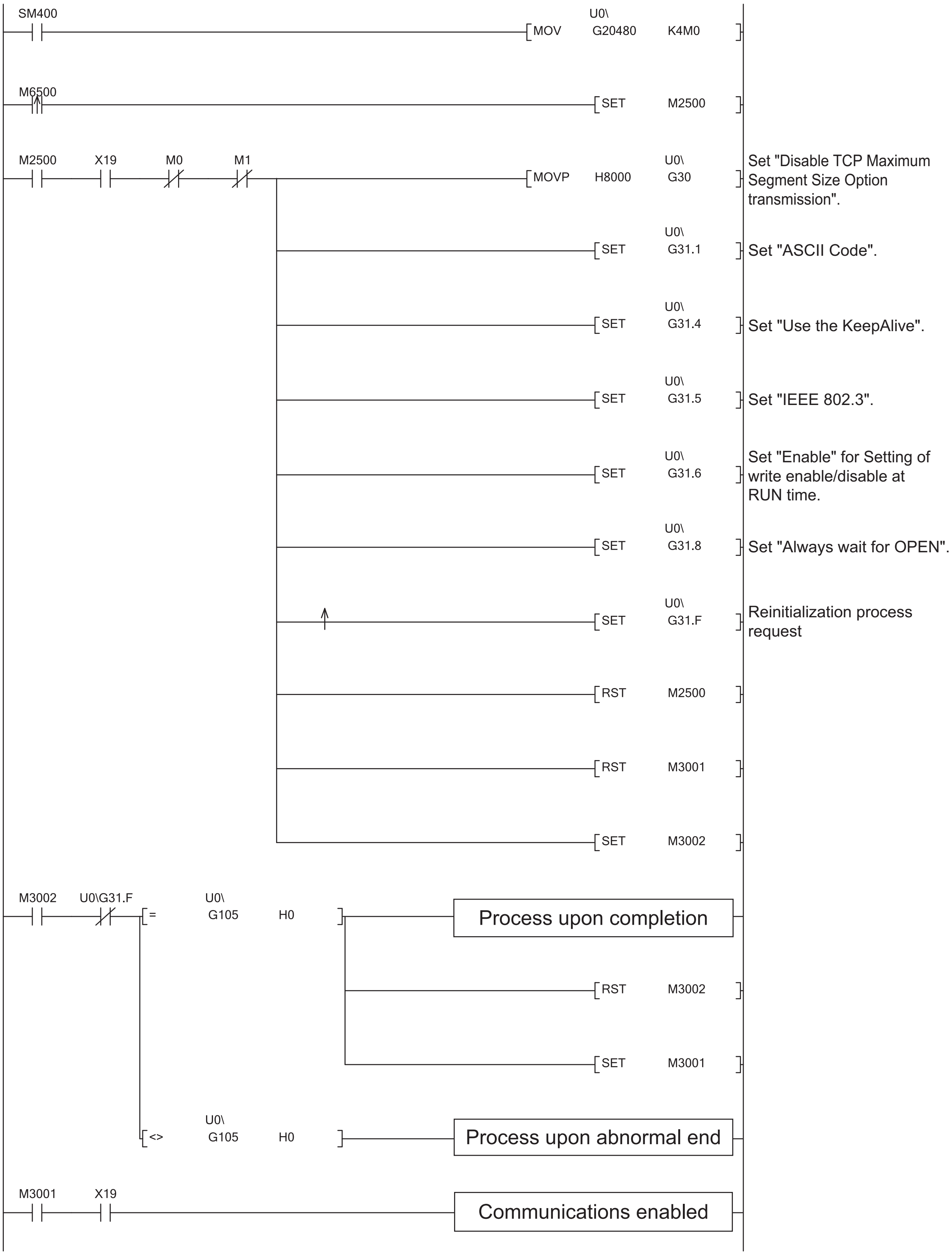


(To the next page)


M3000
U0\G31.F
= U0\G105 H0
Process upon completion
RST M3000
SET M3001
<> U0\G105 H0
Process upon abnormal end
M1000
M3001
X19
M0
M20
MOVP H0 D100
ZP.OPEN "U0" K1 D100 M100
M100
M101
Process upon completion
Communications enabled
M101
Process upon abnormal end

(d) Writing directly to the buffer memory

The following shows an example where the stored values in the TCP Maximum Segment Transmission setting area and Communication condition setting (Ethernet Operation Setting) area is changed (when the I/O signals of the E71 are X/Y00 to X/Y1F). Create a program with a contact (such as a flag for user) indicating the status of the open completion signal storage area.



Remark

This is a sample program for communications using connection numbers 1 and 2. When using another connection number, specify the corresponding signals and bits.

(3) Precautions

(a) MELSOFT products supporting TCP Maximum Segment transmission

To set "Enable TCP Maximum Segment Size Option transmission" in the reinitialization process, use the following MELSOFT products together.

MELSOFT product	Supported version
GX Works2	Version 1.15R or later
GX Developer	Version 8.07H or later
MX Component	Version 3.03D or later
MX Links	Version 3.08J or later

To perform communications over Ethernet using a MELSOFT product other than those listed above, select "Disable TCP Maximum Segment Size Option transmission" or use UDP/IP communications. When "Enable TCP Maximum Segment Size Option transmission" is selected, the program may not be normally read/written.

(b) Buffer memory area settings

The following parameters operate according to the settings of the corresponding buffer memory areas configured when a reinitialization process is performed. Therefore, do not change the settings of the buffer memory areas.

- Timer setting of the initial setting
- Open setting
- Router relay parameter setting
- Station number <-> IP information setting
- FTP parameters setting

(c) Reinitialization process request

Do not request another reinitialization process while a reinitialization process is in progress.

Appendix 5 Line Status Check

The E71 line status, parameter settings, and progress of the initial process can be checked. The following two methods can be used to check the line status.

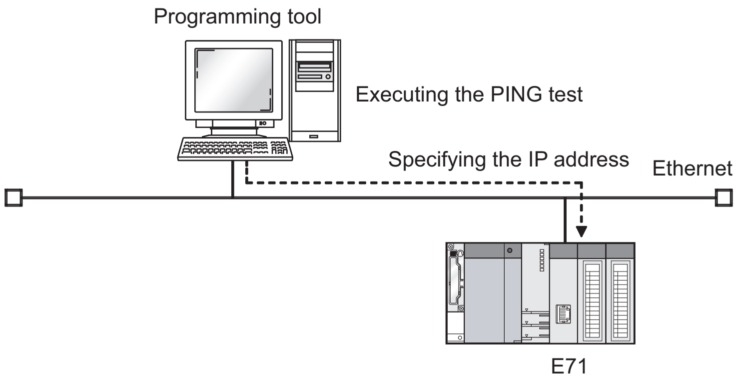
- PING test
- Loopback test

Appendix 5.1 PING test

This section describes the PING test.

(1) Method using direct connection to the E71

This test is to check the existence of an E71[*1] that has completed the initial process on the same Ethernet network using the programming tool or to check the existence of a connected device, such as a personal computer, having a specified IP address.



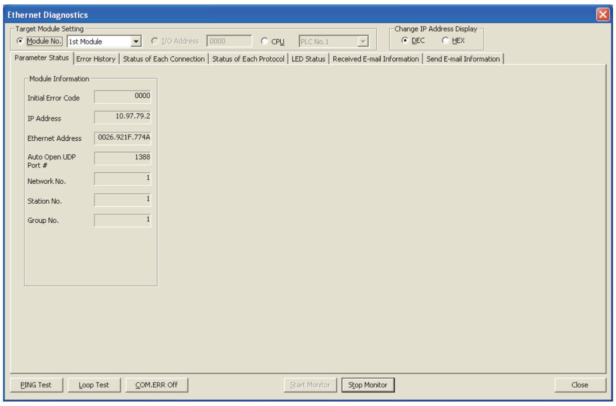
*1 This also applies to QnA/A series modules. However, for an AJ71E71, AJ71E71-B2, and AJ71E71-B5, software versions S and later are required.

(a) Applications

The line status between the programming tool (personal computer) and the E71 can be checked.

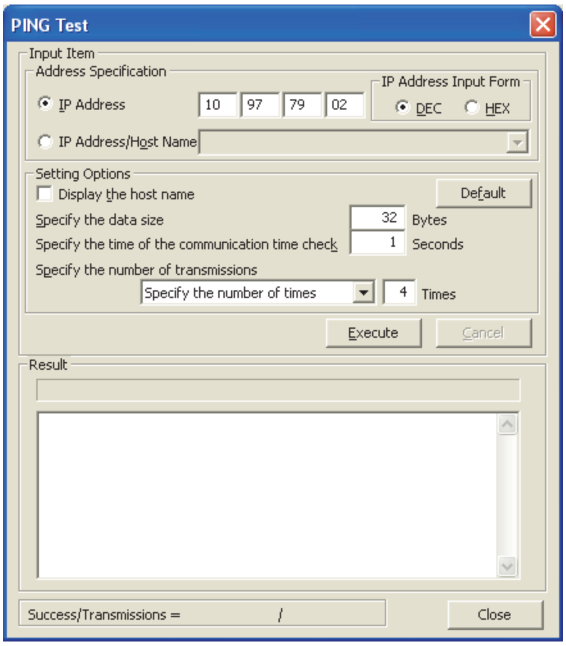
(b) Operating procedure

Use the following procedure.

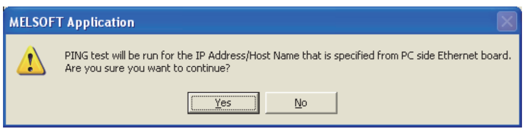


1. **Click the** PING Test **button in the "Ethernet Diagnostics" window.**

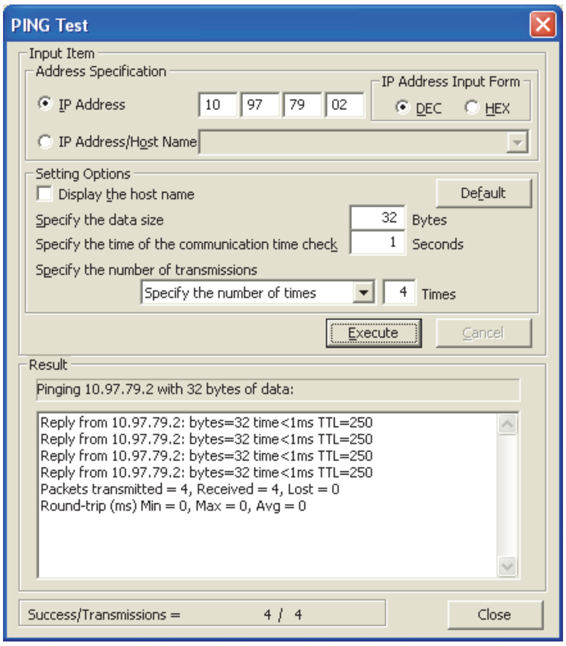
[Diagnostics] ➪ [Ethernet Diagnostics] ➪ "Target Module Setting" under "Module No."



2. **Configure the settings and click the** Execute **button.**



3. **Click the** Yes **button.**



4. **The test results are displayed.**

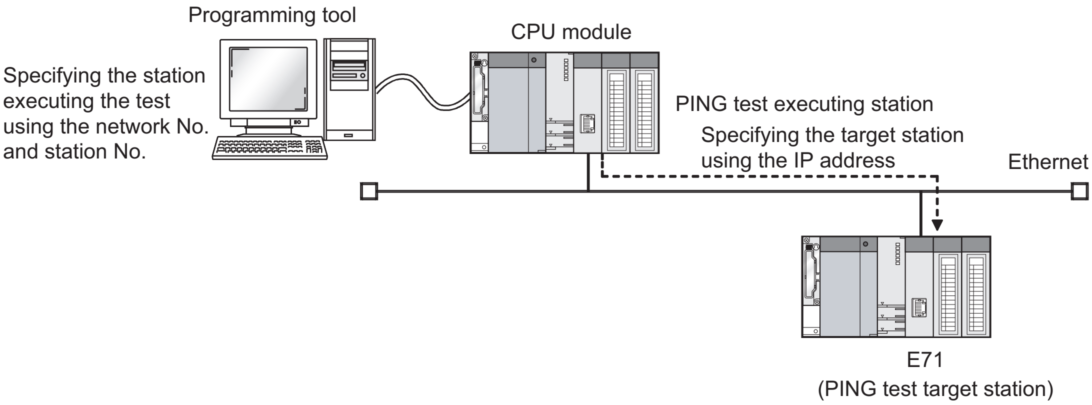
(c) Action for abnormal end

If the test fails, check the following and perform the test again.

- Whether the E71 is properly mounted on the base unit
- Connection to the Ethernet network
- Parameter settings written in the CPU module
- Operating status of the CPU module (whether or not an error has occurred)
- IP addresses set in the programming tool and the PING test target station
- Whether the connected device has been reset after the E71 was replaced

(2) Method using direct connection to the CPU module

This test is to check the existence of an E71[*1, *2] that has completed the initial process on the same Ethernet network where a station connected to the programming tool exists or to check the existence of a connected device, such as a personal computer, having a specified IP address, by directly connecting the programming tool and the CPU module.



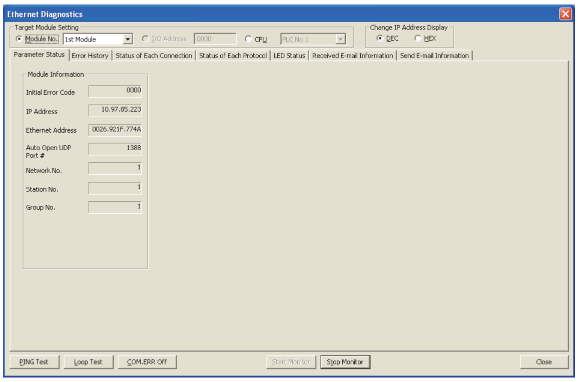
*1 This also applies to QnA/A series modules. However, for an AJ71E71, AJ71E71-B2, and AJ71E71-B5, software versions S and later are required.
*2 The PING test cannot be executed on the own station.

(a) Applications

By specifying the execution station and the target station, the PING test can be executed from a remote device.

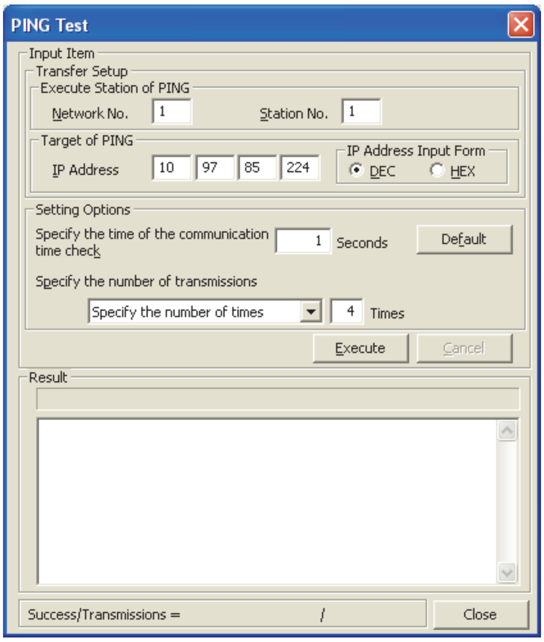
(b) Operating procedure

Use the following procedure.

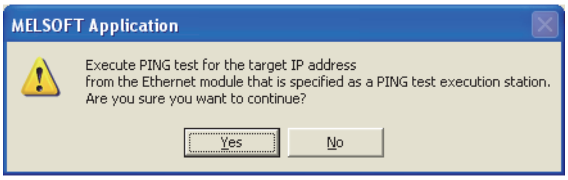


1. **Click the** PING Test **button in the "Ethernet Diagnostics" window.**

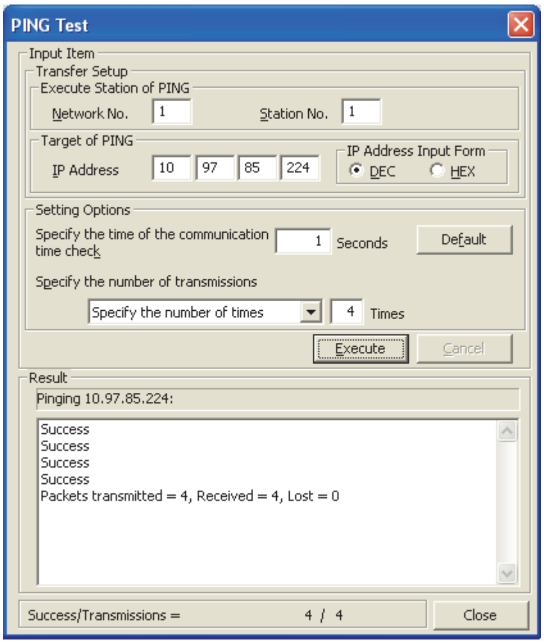
[Diagnostics] ➪ [Ethernet Diagnostics] ➪ "Target Module Setting" under "Module No."



2. **Configure the settings and click the Execute button.**



3. **Click the Yes button.**



4. **The test results are displayed.**

(c) Action for abnormal end

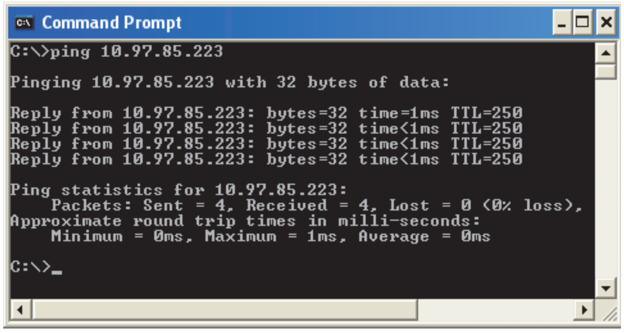
If the test fails, take the same action as that for the method using direct connection to the E71. (Page 366, Appendix 5.1 (1) (c))

(3) Method using the PING command

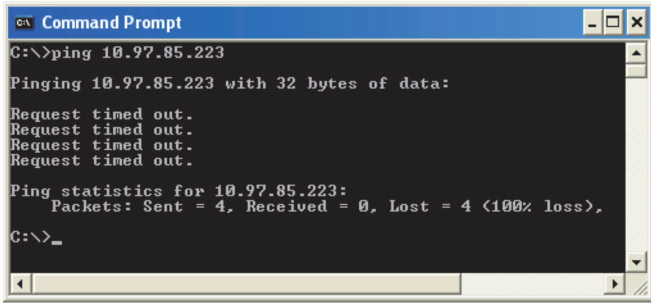
The PING test can be also executed using the PING command. The following shows an example of checking whether the initial process has been completed by issuing a PING command to the E71 of the own station from a connected device in the same Ethernet network.

[Specification method] ping IP Address
[Sample program] IP Address of an E71(10.97.85.223)

(Window upon normal completion)



(Window upon abnormal end)



(a) Action for abnormal end

If the test fails, check the following and perform the test again.

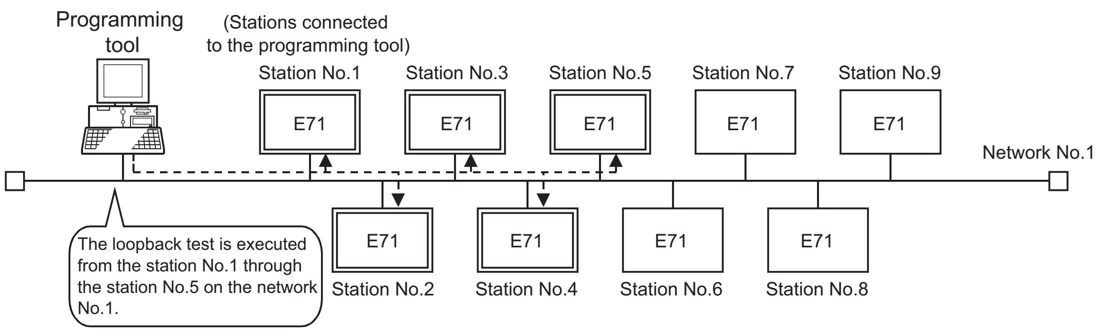
- Whether the E71 is properly mounted on the base unit
- Connection to the Ethernet network
- Parameter settings written in the CPU module
- Operating status of the CPU module (whether or not an error has occurred)
- IP address of the destination E71 specified using the PING command

Appendix 5.2 Loopback test

This section describes the loopback test.

(1) Method using the programming tool

This test is to check whether the initial process for each module has been completed. The test is executed on the network of stations connected to the programming tool, and loopback test messages are sent sequentially to each E71*1 on the network and in the station number range specified for the loopback test request designation.



*1 Because the E71 of function version A and QnA/A series modules do not have a function for responding to this request, the test results cannot be checked.

(a) Applications

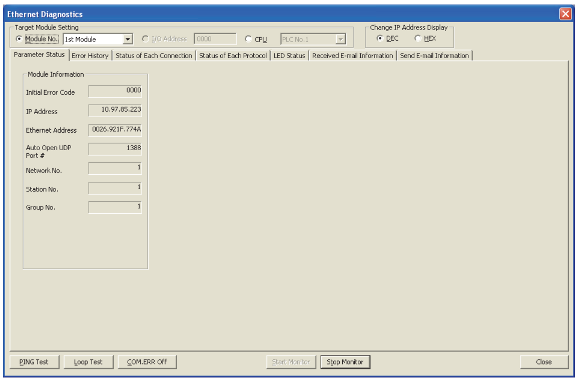
The test can be executed only by specifying the network number for the test and target station number range.

Point

For the E71-mounted station subjected to the loopback test, do not specify the MELSOFT application transmission port (UDP/IP) as a valid port for the password check. The loopback test cannot be executed.

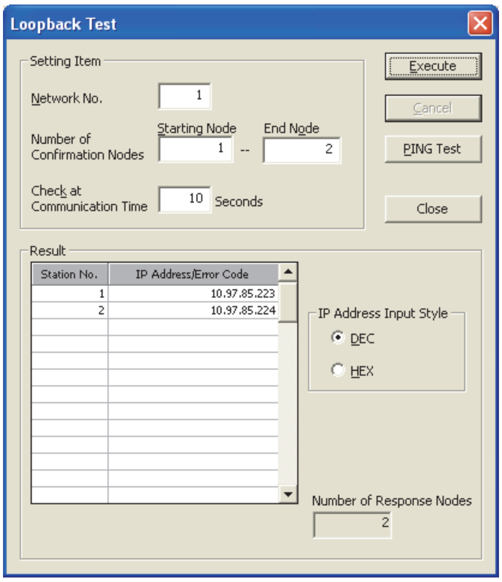
(b) Operating procedure

Use the following procedure.



1. **Click the** Loop Test **button in the "Ethernet Diagnostics" window.**

[Diagnostics] ➪ [Ethernet Diagnostics] ➪ "Target Module Setting" under "Module No."



2. **Configure the settings and click the** Execute **button to display the test results.**

Point

If the same IP address or station number is assigned to multiple stations, only the results of the station that responds first are displayed.

(c) Action for abnormal end

If the loopback test fails, "No response" or an error code is displayed.

Displayed test result	Status of the target E71	Cause	Action
IP address	Initial processing normal completion status (INIT. LED turns on.)	Normal completion	-
No response	No error	The initial process for the target E71 has not been completed normally.	Correct the following parameter settings. • Basic setting • Ethernet operation setting • Initial setting
		There is an error, such as cable disconnection and line disconnection, in the line connection to the target E71.	• Check the cable. • Check the transceiver.
		The IP address of the target E71 is incorrect. (The class or subnet address differs from that of the E71.)	Correct the set values of the Ethernet operation settings.
		The same IP address has been set to multiple target E71 modules.	
		The same network number or station number has been set to multiple target E71 modules.	Perform the PING test on the "No response" module. When the test is completed normally, correct the set values in the basic settings.
	No error/Error exists	The Ethernet line is heavily loaded (including a case where an error equivalent to the error codes $C030_H$ and $C031_H$ is occurring).	When the Ethernet line is not heavily loaded, perform the test again.
	Error exists	The routing parameter is not set (an error equivalent to the error code $C080_H$ is occurring).	Correct the set value of the routing parameter.
Error code	No error	The MELSOFT application transmission port (UDP/IP) for the target E71 is locked with the remote password.	Disable the remote password setting and write the parameter to the CPU module.
		The target E71 is a function version A module.	Check the module name and function version.
	No error/Error exists	The Ethernet line is heavily loaded (including a case where an error equivalent to the error codes $C030_H$ and $C031_H$ is occurring).	When the Ethernet line is not heavily loaded, perform the test again.

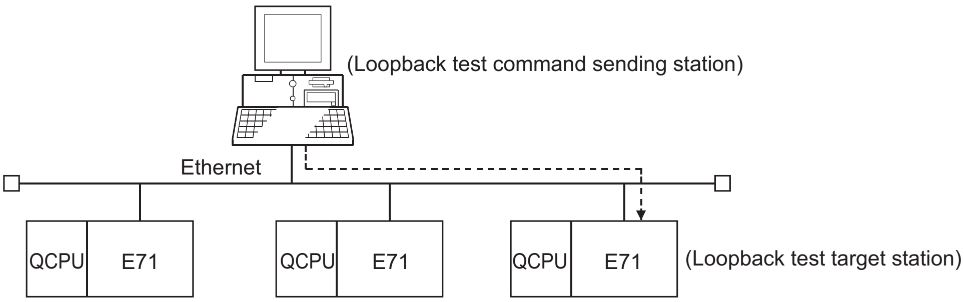
Point

For an E71 where the loopback test failed, check the error and take corrective action. Then restart the E71-mounted station. Restarting the E71 executes the initial process. To check whether the initial process has been completed in the E71, perform the PING test. The PING test can be also performed on the "Loopback Test" window.

(2) Method using the MC protocol

The loopback test can be also executed using MC protocol communications. Note, however, that only the E71 in the own station can be used. The test cannot be executed on E71 modules in other stations over the network. For details, refer to the following.

MELSEC Communication Protocol Reference Manual



(a) Execution method

Execute the test using the MC protocol dedicated command (Loopback test: 0619).

Point

When the user port on the E71 side is used to perform MC protocol communications, the line connection process is necessary. Execute the open process for the connection used for the E71 side.

Appendix 6 Self-Diagnostic Tests

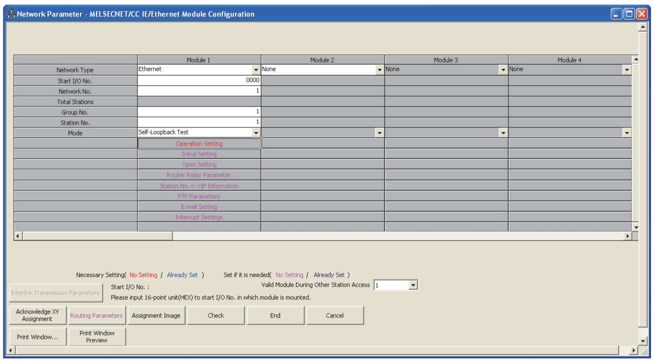
This section describes the self-diagnostic tests to check data communications and hardware of an E71.

Appendix 6.1 Self-loopback test

Hardware including the E71 communication circuit is checked. Send the test message destined to the own station of the E71 to the line. Then check whether the same message can be received over the network.

(1) Operating procedure

Use the following procedure.



1. **Connect the E71 to the line.**[*1]

2. **Set the switch on the CPU module to "STOP".**

3. **Select "Self-Loopback Test" under "Mode" and write the parameter to the CPU module.**

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ "Ethernet" under "Network Type"

4. **Resetting the CPU module starts the test. Test time is approximately five seconds. The RUN and OPEN LEDs turn on during the test.**

5. **Check the LED of the E71 after five seconds. When the test is running normally, the RUN LED turns on. If the test fails, the RUN and ERR. LEDs turn on.**

*1 For a QJ71E71-100, when the line is not connected, the self-loopback test is not performed and ends normally.

If an error is detected in the self-loopback test, the following causes are possible. The error information is stored in Error log area (address: $E5_H$) in the buffer memory.

- E71 hardware error
- Ethernet line error
- External power supply 12VDC error (only 10BASE5)

Point

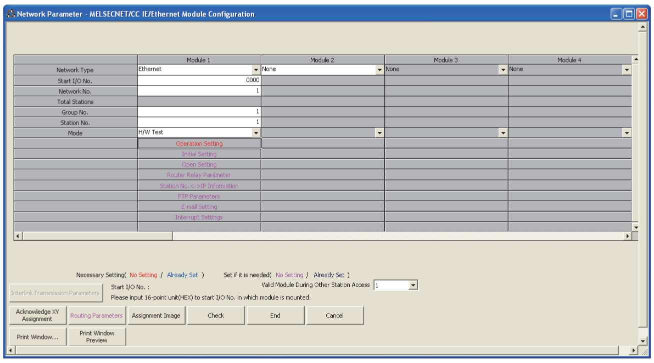
There is no hardware-related problem even when the self-loopback test is executed while the destination is online. Because a packet collision occurs when there are many packets on the line, the test may fail or may not be completed in five seconds. In this case, execute the test after stopping data communications between other devices.

Appendix 6.2 Hardware test (H/W Test)

The RAM and ROM of an E71 are checked.

(1) Operating procedure

Use the following procedure.



1. **Set the switch on the CPU module to "STOP".**

2. **Select "H/W Test" under "Mode" and write the parameter to the CPU module.**

Project window ⇨ [Parameter] ⇨ [Network Parameter] ⇨ [Ethernet/CC IE/MELSECNET] ⇨ "Ethernet" under "Network Type"

3. **Resetting the CPU module starts the test. Test time is approximately five seconds. The RUN and OPEN LEDs turn on during the test.**

4. **Check the LED of the E71 after five seconds. When the test is running normally, the RUN LED turns on. If the test fails, the RUN and ERR. LEDs turn on.**

If an error is detected in the hardware test (H/W test), the following cause is possible. The error information is stored in Error log area (address: $E5_H$) in the buffer memory.

- E71 RAM/ROM error

Point

If the test fails, perform it again. If it fails again, there may be a hardware problem with the E71. Please consult your local Mitsubishi representative.

Appendix 7 Differences from Ethernet Modules of Other Series

This section describes the differences between an E71 and Ethernet modules of other series.

Appendix 7.1 Comparison with a Built-in Ethernet port QCPU

For comparison of the specifications and functions between an E71 and a Built-in Ethernet port QCPU, refer to the following.

QnUCPU User's Manual (Communication via Built-in Ethernet Port)

Appendix 7.2 Comparison with QnA/A Series Modules

This section provides the functional comparison between an E71 and QnA/A series modules and precautions for using the same program. The QnA/A series modules are the following products.

Series	Model name	Product name
A series	AJ71E71	AJ71E71, A1SJ71E71-B2, A1SJ71E71-B5
	AJ71E71-S3	AJ71E71-S3, A1SJ71E71-B2-S3, A1SJ71E71-B5-S3
	AJ71E71N	AJ71E71N3-T, AJ71E71N-B5, AJ71E71N-B2, AJ71E71N-T, AJ71E71N-B5T, A1SJ71E71N3-T, A1SJ71E71N-B5, A1SJ71E71N-B2, A1SJ71E71N-T, A1SJ71E71-B5T
QnA series	QE71(N)	AJ71QE71, AJ71QE71-B5, A1SJ71QE71-B2, A1SJ71QE71-B5, AJ71QE71N3-T, AJ71QE71N-B5, AJ71QE71N-B2, AJ71QE71N-T, AJ71QE71N-B5T, A1SJ71QE71N3-T, A1SJ71QE71N-B5, A1SJ71QE71N-B2, A1SJ71QE71N-T, A1SJ71QE71N-B5T

(1) Functional comparison

The following table lists the functional comparison between an E71 and QnA/A series modules.

Function		AJ71E71	AJ71E71-S3, AJ71E71N	QE71(N)		E71
				9706 or earlier	9706B or later	
Initial process	Initial process with the program	○	○	○	○	○
	Initial process with parameter settings	×	×	×	○	○
Open process*1	Open process with the program	○	○	○	○	○
	Open process with parameter settings	×	×	×	×	○
Communications using a fixed buffer	Procedure Exist	○	○	○	○	○*2
	No Procedure	×	○	○	○	○*2
Communications using a random access buffer		○	○	○	○	○
MC protocol communications		○	○	○	○*3	○*3
Communications using data link instructions		×	×	×	○*4	○
Interrupt process (upon data receiving)	BUFRCVS instruction	×	×	×	×	○
	RECVS instruction	×	×	×	×	○
E-mail function	Send/receive with the program	×	×	×	×	○
	Send with the auto notification function	×	×	×	×	○
File transfer (FTP server) function		×	×	×	○	○
Web function		×	×	×	×	○
Broadcast communications		×	○	○	○	○
Communications with the switch on the CPU module set to STOP		×	○	×	○	○
Selection of the communication data code (ASCII/binary)		○	○	○	○	○
CC-Link IE Controller Network, CC-Link IE Field Network, MELSECNET/H, MELSECNET/10 relay communications		×	×	×	○*4	○
Router relay function		×	○	○	○	○
Connected device alive check	Ping	×	○	○	○	○
	KeepAlive	×	×	×	×	○
Pairing open		×	○	○	○	○
Unit of each timer setting value for data communications	500ms	×*5	○	○ (Fixed)	○ (Fixed)	○ (Fixed)
	2s	○ (Fixed)	○	×	×	×
Connection to MELSOFT products	TCP/IP	×	×	×	×	○
	UDP/IP	×	○	○	○	○
Installation of EEPROM		×	×	○	○	×*6
TCP Maximum Segment Size Option transmission		×	○*7	×	○*7	○*8
Hub connection status monitor function		×	×	×	×	○*9
IP address in use detection function		×	×	×	×	○*9
Communications using SLMP		×	×	×	×	○*9
Data communications using the predefined protocol		×	×	×	×	○*9
Module error collection function		×	×	×	×	○*9

*1 For the E71, the number of connections for the open process from the CPU module has increased to 16 connections.
*2 Compatible with the I/O signal and QE71(N) buffer memory
*3 Data of up to 960 words can be read from/written to an E71, and up to 480 words in a QE71(N).
*4 Availability depends on the CPU module and programming tool version.
*5 This applies to a module with the software version of Q or earlier.
*6 No EEPROM is installed. Items registered in the EEPROM of a QE71(N) are set using the parameter settings of the programming tool.
*7 This applies to a module with the software version of E or later.
*8 This function can be used in an E71 with a serial number (first five digits) of "05051" or later. If the E71 cannot communicate data when combined with a certain connected device, the setting can be changed in the buffer memory (address: $1E_H$). (Page 38, Section 3.5.2) After changing the setting, execute the reinitialization process.
*9 Only the QJ71E71-100 can be used.
Availability depends on the QJ71E71-100 and a programming tool version (Page 351, Appendix 3).

Point

The response performance of the E71 to connected devices is faster than that of the A/QnA series modules. When the E71 is used, the compatibility with the A/QnA series modules cannot be precisely maintained. If there is a problem due to the performance of the connected device, create a timing similar to that of the existing system using the CPU module constant scan setting.

(2) Precautions for using the same program

The QnA/A series modules used for data communications between a CPU module and the connected device on the Ethernet network can be replaced by an E71. This section describes precautions for using the same program used in QnA/A series modules.

(a) AJ71E71(-S3) and AJ71E71N

- Program of connected devices
The following communication function programs of connected devices for an AJ71E71(-S3) and AJ71E71N can be used for communications with an E71. However, because the response speed of the AJ71E71(-S3) and AJ71E71N is different from that of the E71, the program may not be used as is. Before using the same program, check the operation.

○: Communications allowed by using the program for an AJ71E71(-S3) and AJ71E71N on the connected device side,
-: Not allowed to use

Function	Connected device → E71	E71 → connected device	AJ71E71(-S3), AJ71E71N → E71	E71 → AJ71E71(-S3), AJ71E71N
Communications using a fixed buffer ("Procedure Exist")	○	○	○	○
Communications using a random access buffer	○	-	-	-
Data read/write in the CPU module*1	○	-	-	-

*1 Only A-compatible 1E frame commands can be used for data communications. To perform data communications using a command other than the A-compatible 1E frame commands, create a new program.

- Program for an AJ71E71(-S3) and AJ71E71N in the own station
Because the assignment of the buffer memory areas for an E71 is different from that for an AJ71E71(-S3) and AJ71E71N, a program for the AJ71E71(-S3) and AJ71E71N cannot be used for an E71. Create a new program by referring to chapters that explain each function.
- Program using an I/O signal
Programs cannot be used together with parameter settings on the programming tool.

(b) QE71(N)

Program of connected devices

Programs of the connected devices for a QE71(N) can be used for communications with the E71 except for the following programs.

- Program for file operation-related commands
- Program to access the data link system (A QCPU (Q Mode) cannot be connected to MELSECNET (II) or MELSECNET/B.)

However, because the response speed of an E71 is different from that of a QE71(N), the program may not be used as is. Before using the same program, check the operation.

Program for a QE71(N) in the own station

- When using a program for the initial process and the termination process, do not write the parameters of the E71 (network parameters) set on the programming tool to the QCPU. Note the following when not using the parameters of an E71 set on the programming tool.
    - Because a QE71(N) operates with all of its communication condition setting switches off, set the communication conditions using the reinitialization process.
    - When MELSOFT products (such as a programming tool) are directly connected to an E71, the MELSOFT products (such as a programming tool) cannot access a QCPU.
- To set the parameters of an E71 using the programming tool, delete the programs for the initial process and the termination process.
- Programs for the QE71(N) in the own station can be used for communications with an E71 except for the following programs.
    - Program to access the data link system
    - Program related to the EEPROM
    - Connection number 8 pairing open setting
    - Parameter setting program using the EPRSET instruction

  However, because the response speed of an E71 is different from that of a QE71(N), the program may not be used as is. Before using the same program, check the operation.

Program using an I/O signal

- Programs cannot be used together with parameter settings on the programming tool.

Point

- For an E71, set the operation mode and communication conditions using the programming tool. Settings cannot be configured using a switch unlike QnA/A series modules. (There is no switch for settings.)
- For an E71, after the Passive open process is executed, the open request cannot be canceled before open completion. After the open completion, execute the close process.

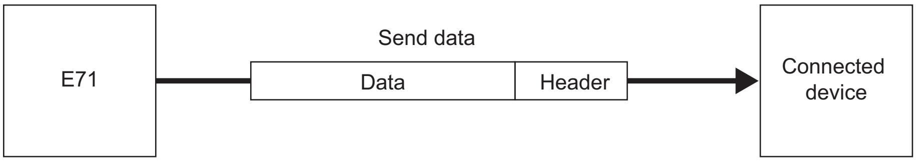
Appendix 8 Operation Image and Data Structure of Predefined Protocol

Appendix 8.1 Operation image of each communication type of protocol

In the predefined protocol function, communications with connected devices is performed through the communication type "Send Only", "Receive Only", or "Send&Receive".
This section describes the operation images of each communication type.

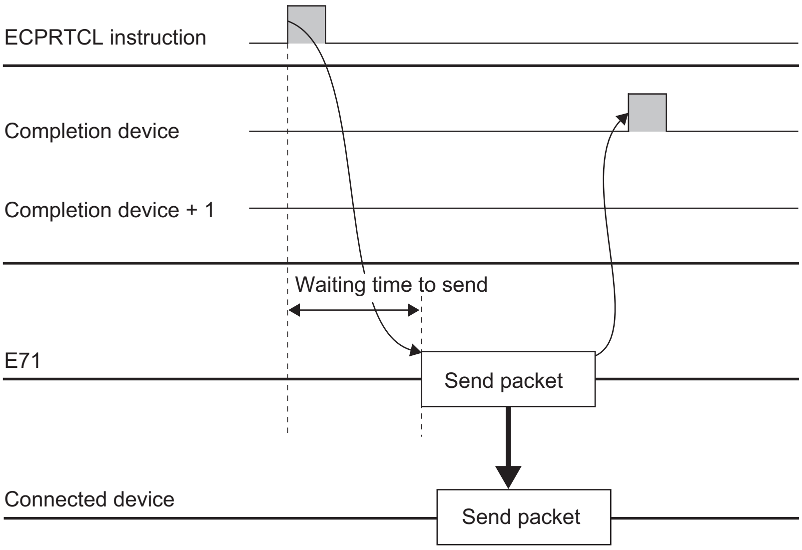
(1) When the communication type is "Send Only"

The specified packet is sent once.

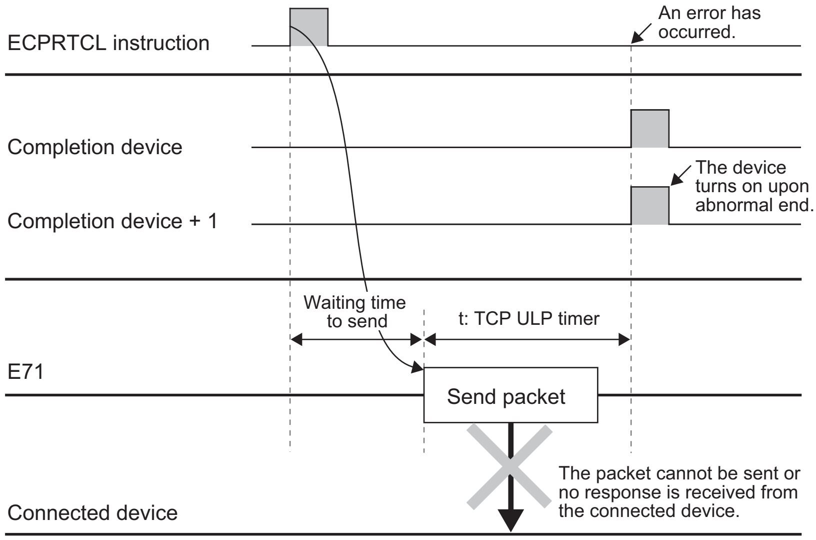


The operation image of "Send Only" is as follows.

(a) When the instruction ends normally



(b) When the instruction ends abnormally at TCP (timeout error)

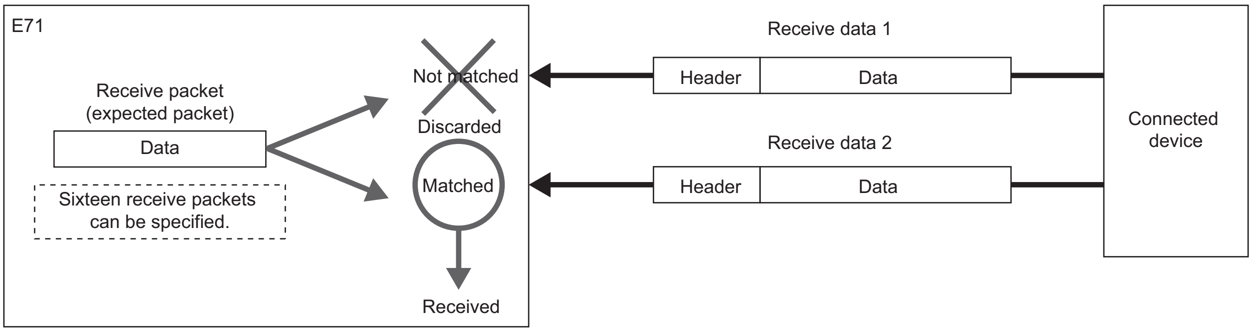


Point

For an error occurred when the communication ends abnormally, refer to the error code stored in the buffer memory.
(Page 314, Section 16.6.3)

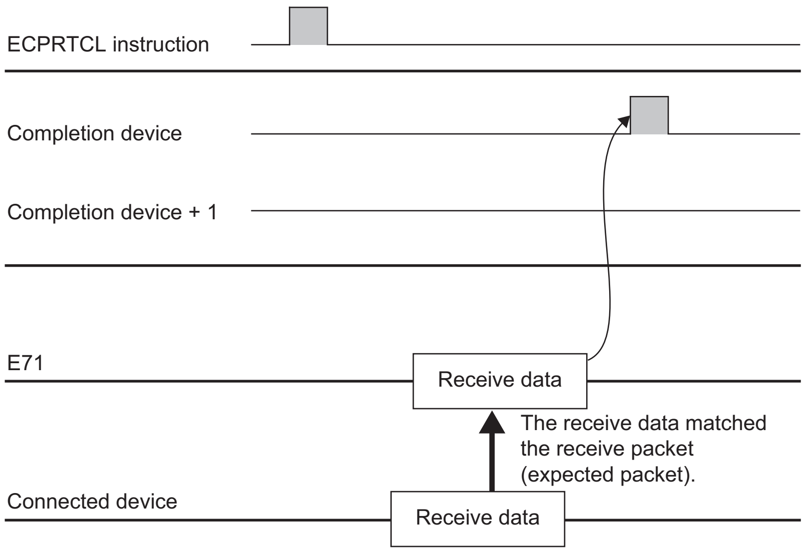
(2) When the communication type is "Receive Only"

The receive process completes when the data received from a connected device match the receive packet (expected packet). If they do not match, the data are discarded.

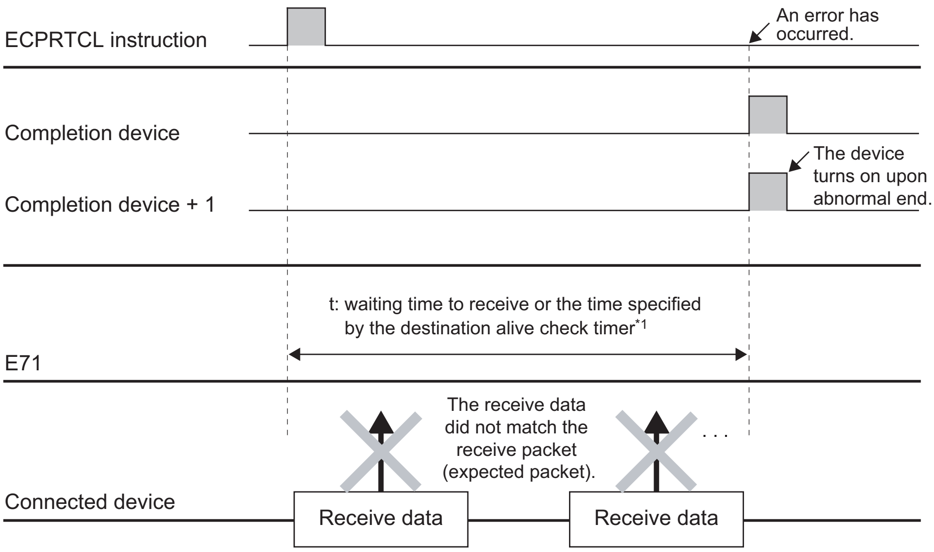


The operation image of "Receive Only" is as follows.

(a) When the instruction ends normally



(b) When the instruction ends abnormally (timeout error)



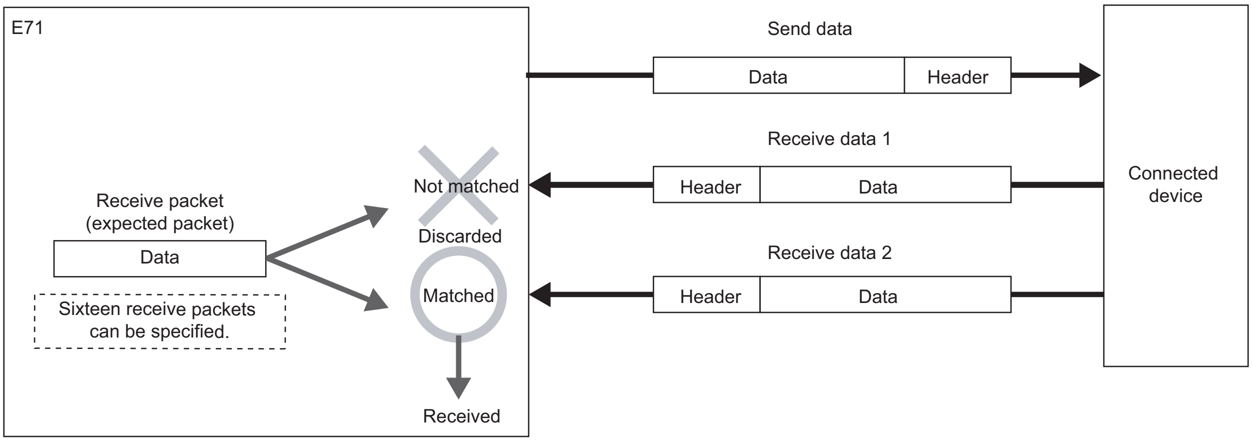
*1 When the receive packet (expected packet) is not matched to the data by the waiting time to receive or when the existence of a connected device cannot be checked within the time specified by the destination alive check timer (the time calculated based on the destination existence confirmation starting interval timer value, destination existence confirmation interval timer value, and Destination existence confirmation resending time), the instruction ends abnormally.

Point

- When variables are included in receive packet (expected packet) elements, variable data are not verified.
- Receive packets (expected packets) can be specified up to 16.
- When multiple receive packets (expected packets) are specified, the received data are verified with the registered receive packet (expected packet) in order of registration. When the receive packet (expected packet) is matched to the data, the receive process is completed. The following verification is not performed.
- The receive packet number which is matched in the verification is stored in the control data of the ECPRTCL instruction and the buffer memory.
- For the error occurred when the communication ends abnormally, refer to the error code stored in the buffer memory. (Page 314, Section 16.6.3)

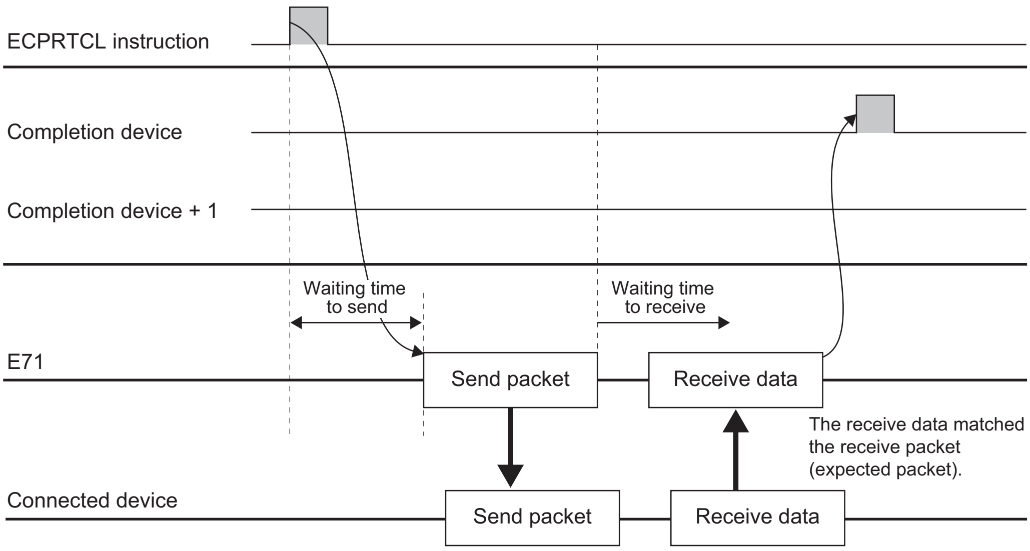
(3) When the communication type is "Send&Receive"

When the packet is sent once and the sending completes normally, the status of the E71 changes to the receive wait status. The process completes when the data received from a connected device match the receive packet (expected packet) and the receive process is performed.

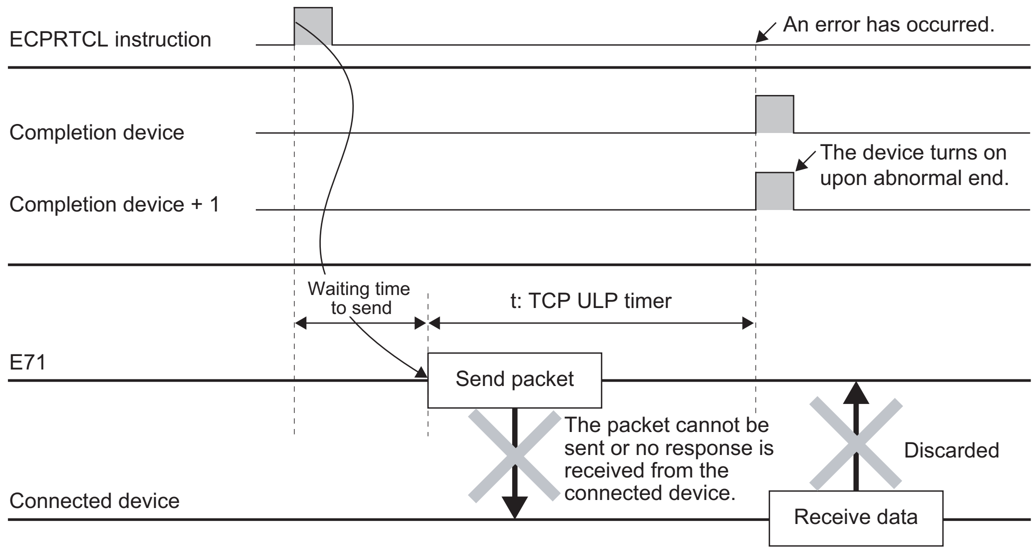


The operation image of "Send&Receive" is as follows.

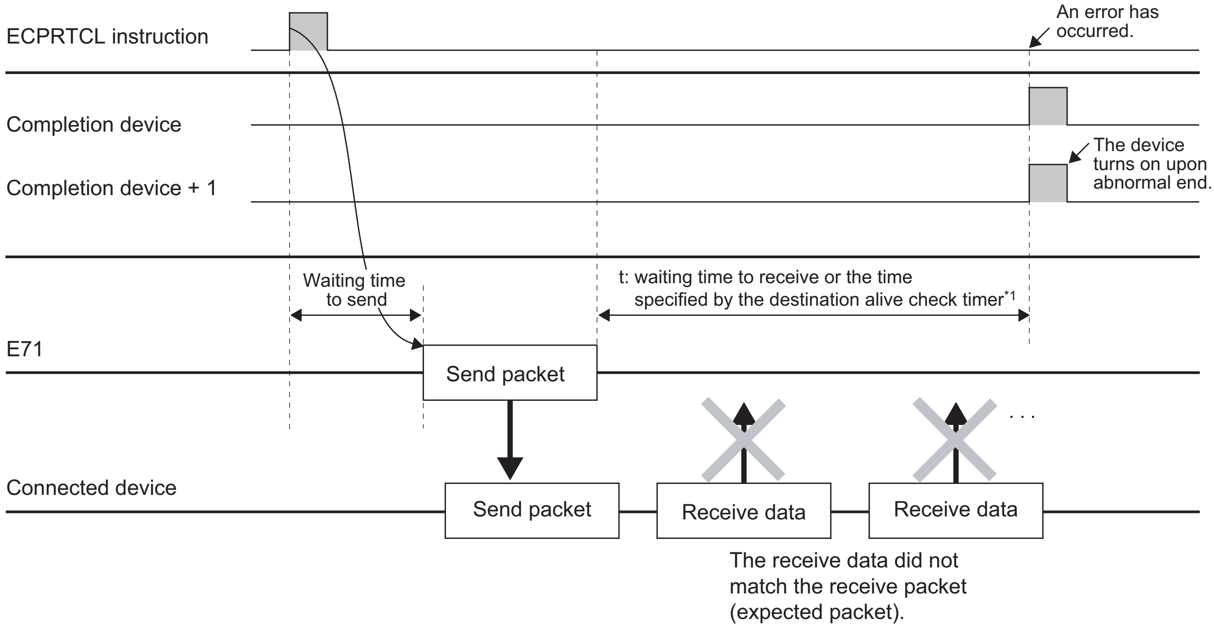
(a) When the instruction ends normally



(b) When the instruction ends abnormally (timeout error at sending)



(c) When the instruction ends abnormally (timeout error of waiting time to receive)



*1 When the receive packet (expected packet) is not matched to the data by the waiting time to receive or when the existence of a connected device cannot be checked within the time specified by the destination alive check timer (the time calculated based on the destination existence confirmation starting interval timer value, destination existence confirmation interval timer value, and Destination existence confirmation resending time), the instruction ends abnormally.

Point

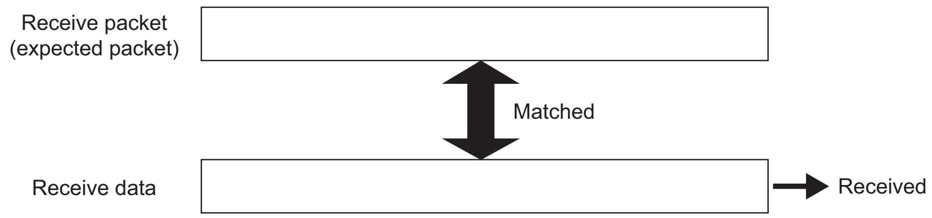
For the error occurred when the communication ends abnormally, refer to the error code stored in the buffer memory.
(☞ Page 314, Section 16.6.3)

Appendix 8.2 Verification operation of receive packet

This section describes the verification operation for a receive packet (expected packet) when the E71 communicates with a connected device with protocols whose communication type includes receiving.

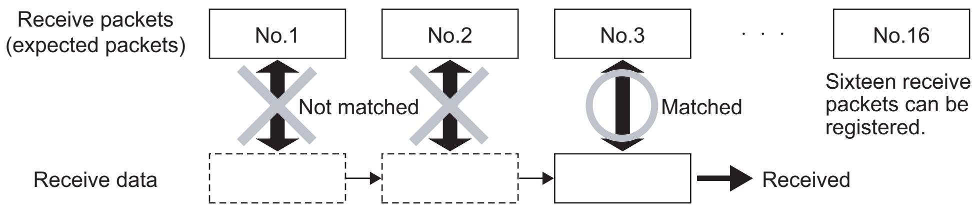
(1) When received data are matched to a receive packet (expected packet)

The receive process is completed when the receive data are compared with the receive packet (expected packet), and they are matched.



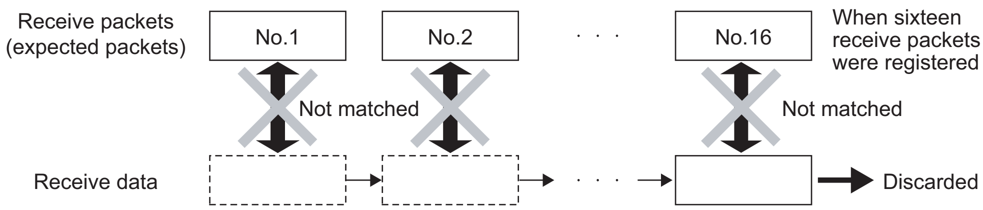
(2) When multiple receive packets (expected packets) are specified

The predefined protocol support function enables to register up to 16 receive packets (expected packets).
When the data are received, the registered receive packet (expected packet) is verified in order of registration.
When the receive packet (expected packet) is matched to the data, the receive process is completed.



(3) When received data are not matched to all receive packets (expected packets)

The received data are discarded.



Appendix 8.3 Data examples of packet elements

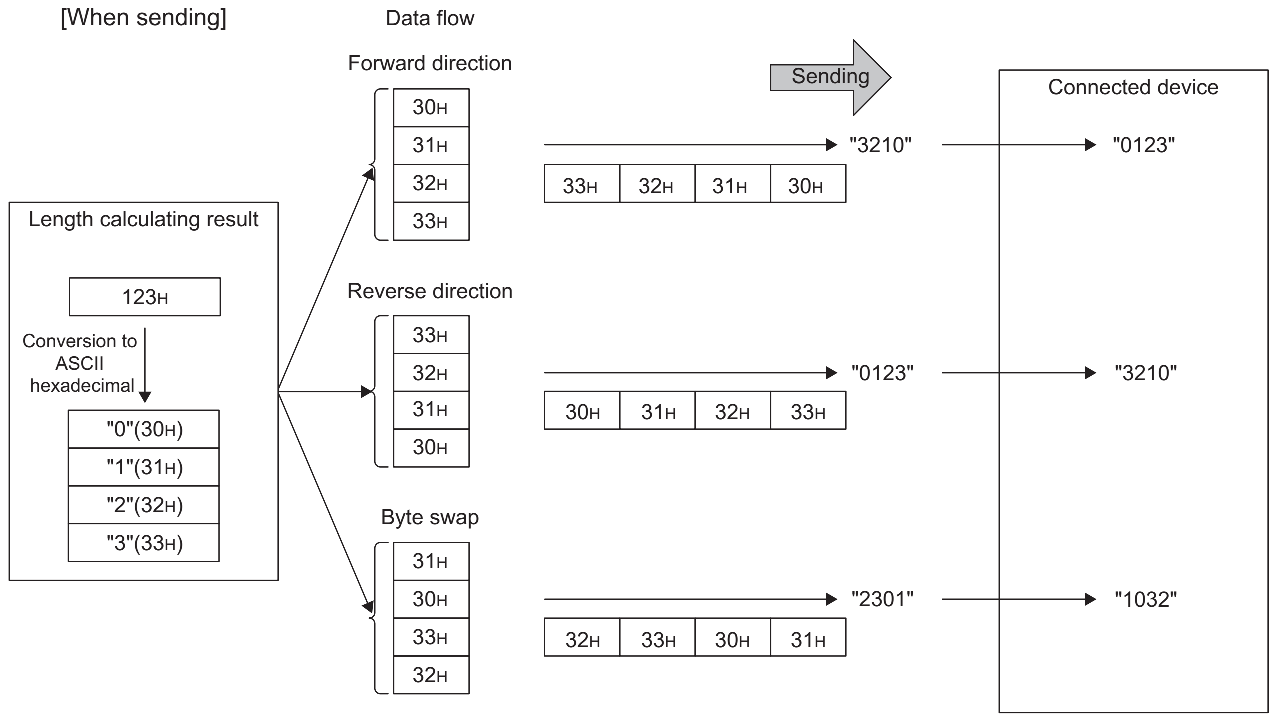
This section describes the procedures for the processing and practical data examples of each element placed in a packet.

(1) Length

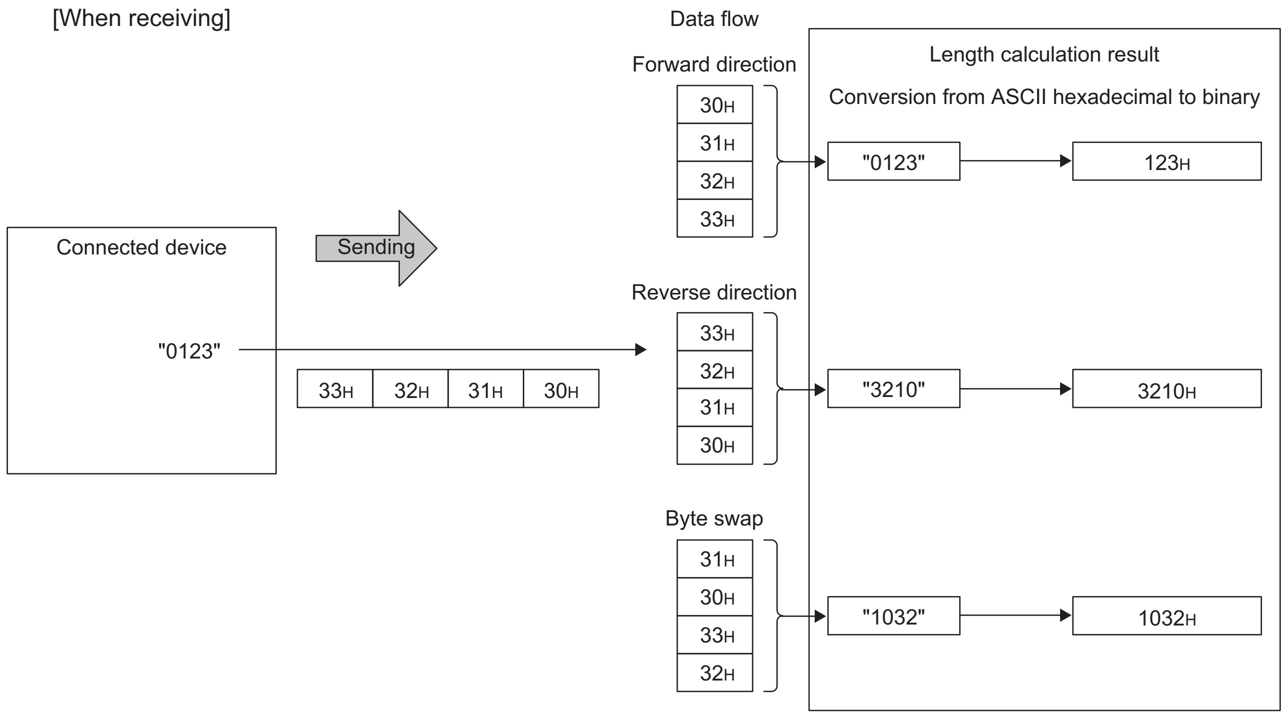
(a) Procedures

The E71 processes a Length element as follows.

[When sending]



[When receiving]



Appendix 8 Operation Image and Data Structure of Predefined Protocol
Appendix 8.3 Data examples of packet elements

(b) Data flow

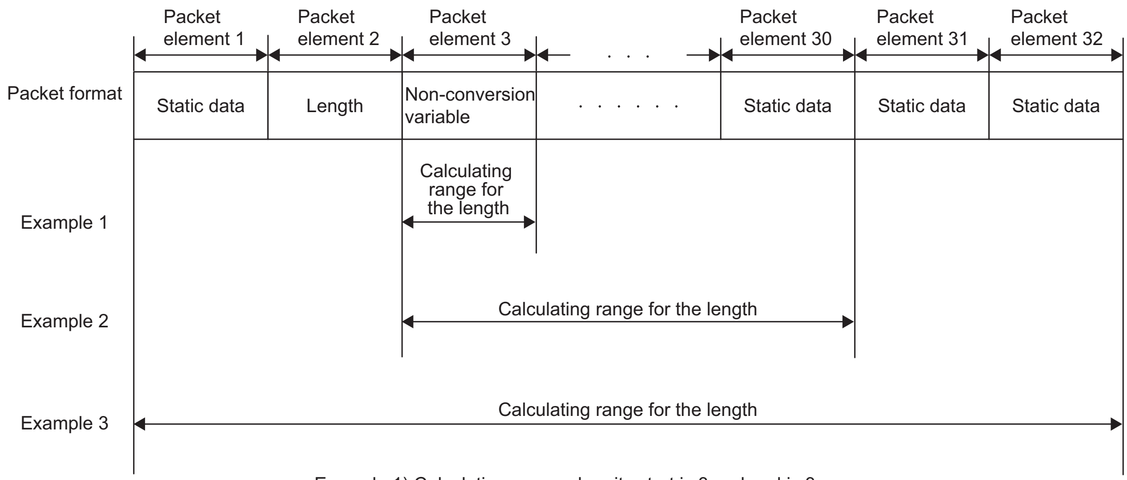
Data Flow is used to specify the order of send data and receive data.

"Forward Direction (Upper Byte→Lower Byte)", "Reverse Direction (Lower Byte→Upper Byte)", and "Byte Swap (by Word)" can be used as the Data Flow.

- Forward Direction and Reverse Direction: Available when the Data Length is 2 bytes or more.
- Byte Swap: Available only when the Data Length is 4 bytes.

(c) Calculating range for the length

The following shows the specification examples of calculating range for the length.



Example 1) Calculating range when its start is 3 and end is 3
Example 2) Calculating range when its start is 3 and end is 30
Example 3) Calculating range when its start is 1 and end is 32

(2) Non-conversion Variable

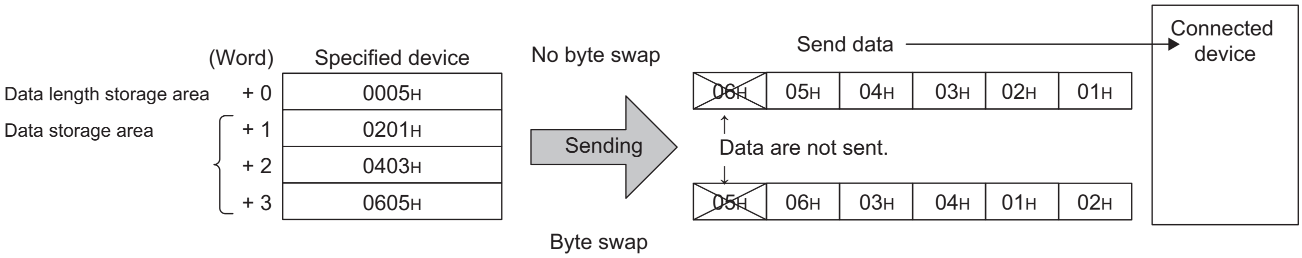
(a) Procedures

The E71 processes a Non-conversion Variable element as follows.

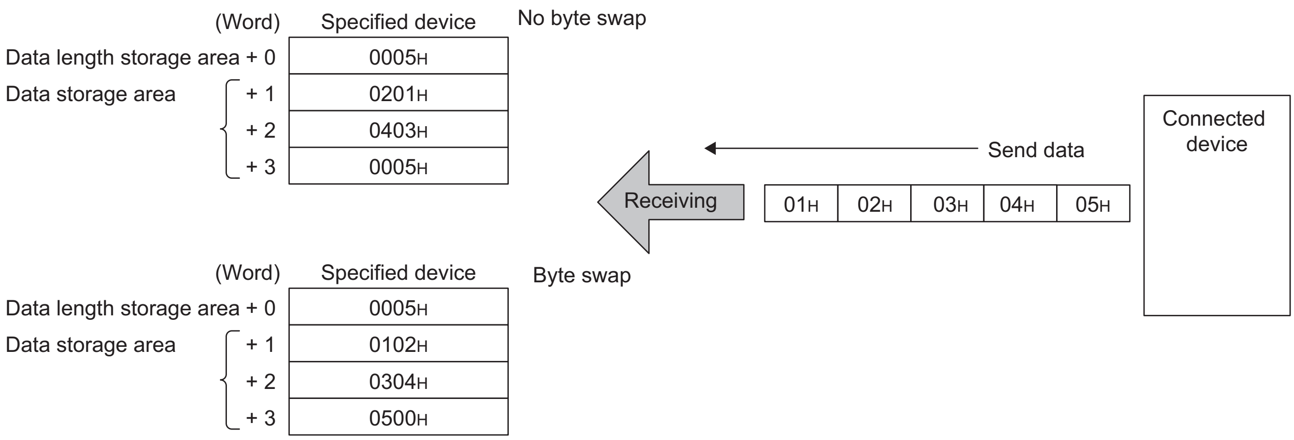
When the setting for "Unit of Stored Data" is "Lower Byte + Upper Byte"

- When the data length of a send packet is an odd number, the upper byte (lower byte for "Byte Swap") of the last device is not sent.
- When the data length of a receive packet is an odd number, the last data is stored with one byte of 00_H.

Ex. When sending data whose length is an odd number



Ex. When receiving data whose length is an odd number

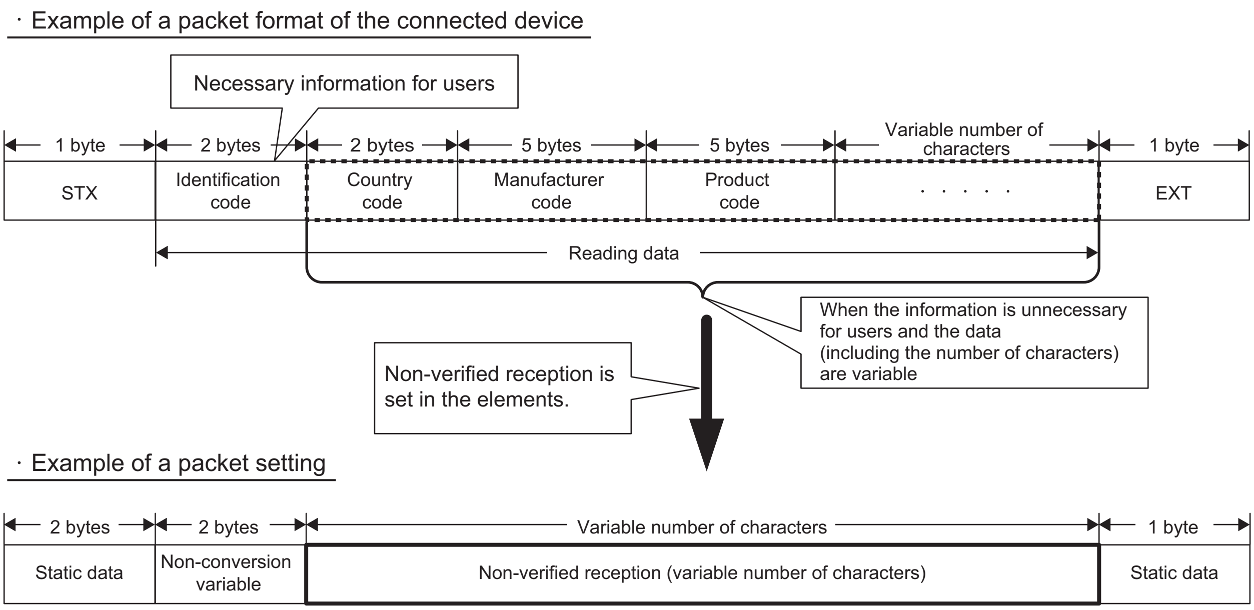


When the setting for "Unit of Stored Data" is "Lower Bytes Only"

- Twice the size of the data length is occupied.
  For the upper data, the E71 ignores the data at sending and adds 00_H to the data at receiving.

(3) Non-verified Reception

The following shows the usage example for a Non-verified Reception element.



Setting the packet format as above and setting the "Non-verified Reception" enable the followings.

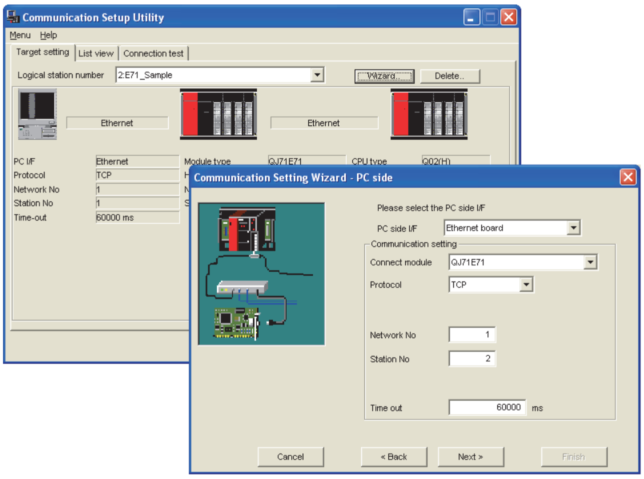
- Only necessary information can be stored in a device of the CPU module or the buffer memory.
- Only one protocol (packet) is required even when the data that may change in each communication are included in the receive packet.

Appendix 9 Usage example of MX Component

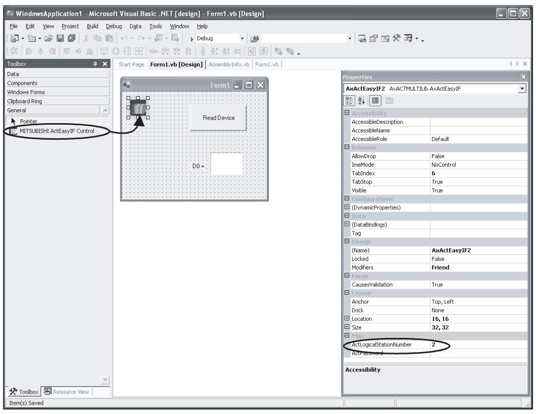
MX Component is an ActiveX control library that supports all the communication routes between a personal computer and programmable controller. Communications can be performed with a simple process without considering different predefined protocols of each communication. This appendix describes how to create a program and sample programs using MX Component.

Appendix 9.1 How to create a program

Visual Basic®.NET 2003 is used for the following procedure.



1. **Follow the wizard to configure communication settings from the personal computer to the programmable controller. (For some controllers, settings are configured only using the program, without using any wizard.) In the wizard, configure necessary settings such as a logical station number, connection module type, and connection destination programmable controller.**

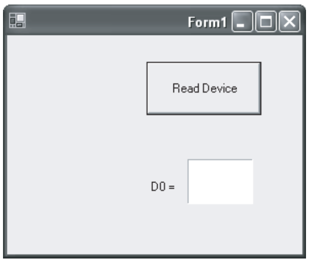


2. **Paste the ACT control icon onto the form. Then set the logical station number set in Step 1 to the control as its property.**

3. **Use a function to write the program that reads the device data.**

```
Private Sub Button1_Click(ByVal sender As System.Obj
    Dim rtn As Integer
    Dim iData As Integer
    rtn = AxActEasyIF1.Open()

    rtn = AxActEasyIF1.GetDevice("D0", iData)
    Label1.Text = iData

End Sub
```

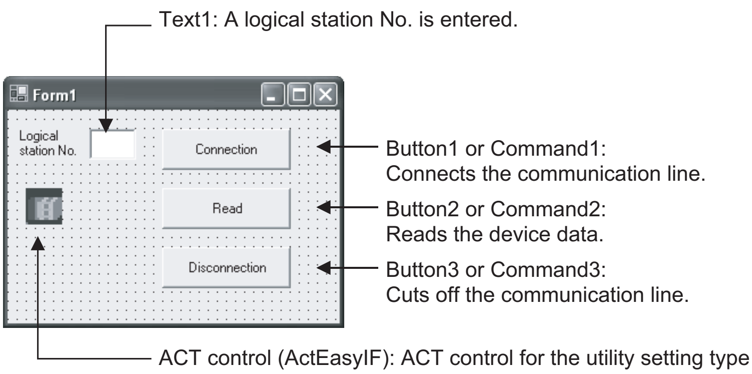
4. **Click the "Read Device" button.**



Appendix 9.2 Sample Program

The following sample program is used to read D0 to D4 (five points) of a target programmable controller using a logical station number.

(1) Window example (Form1)



(2) Program example

Program examples for development software below are described on the following pages.

- Visual Basic®.NET 2003
- Visual C++®.NET 2003

(a) Visual Basic®.NET 2003

```
Private Sub Command1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Command1.Click
' ****************************************
' Connection
' *****************************************
Dim rtn As Integer
        'Get LogicalstationNumber
        AxActEasyIF1.ActLogicalStationNumber = Val(Text1.Text)

        'Connection
        rtn = AxActEasyIF1.Open()
        If rtn = 0 Then
                MsgBox("The Connection was successful")
        Else
                MsgBox("Connection Error:" & Hex(rtn))
        End If
End Sub


Private Sub Command2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Command2.Click
' ****************************************
' Read
' *****************************************
Dim rtn As Integer
Dim idata(5) As Short
        'D0-D4 (5 points) are read
        rtn = AxActEasyIF1.ReadDeviceBlock2("D0", 5, idata(0))
        If rtn = 0 Then
                MsgBox("D0-D4 = " & idata(0) & "," & idata(1) & "," & idata(2) & "," & idata(3) &
                "," & idata(4))
        Else
                MsgBox(Read Error:" & Hex(rtn))
        End If
End Sub
```

(To the next page)

```
Private Sub Command3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Command3.Click
' *****************************************
' Disconnection
' ******************************************
Dim rtn As Integer
        'Disconnection
        rtn = AxActEasyIF1.Close()
        If rtn = 0 Then
                MsgBox("The disconnection was successful")
        Else
                MsgBox("Disconnection Error:"& Hex(rtn))
        End If
End Sub
```

(b) Visual C++®.NET 2003

```
//****************************************
//Connection
//****************************************
private: System::Void button1_Click(System::Object * sender, System::EventArgs * e)
{
        int iRet

        //Get LogicalStationNumber
        axActEasyIF1->ActLogicalStationNumber=Convert::ToInt32(textBox1->Text);

        //Connection
        iRet = axActEasyIF1->Open();
        if( iRet == 0 ){
                MessageBox::Show("The connection was successful" );
        } else {
                MessageBox::Show( String::Format( "Connection Error:0x{0:x8} [HEX]", __box
                (iRet) ) );
        }
}

//****************************************
//Read
//****************************************
private: System::Void button2_Click(System::Object * sender, System::EventArgs * e)
{
        int iRet;
        short sData[5];
        String* szMessage= "";
        String* lpszarrData[];
        int iNumber;
        String* szReadData

        //D0-D4 are read
        iRet = axActEasyIF1->ReadDeviceBlock2( "D0", 5, sData );
        if( iRet == 0 ){
                lpszarrData = new String * [ 5 ];
                lpszarrData[0] = "D0-D4 = ";
```

(To the next page)

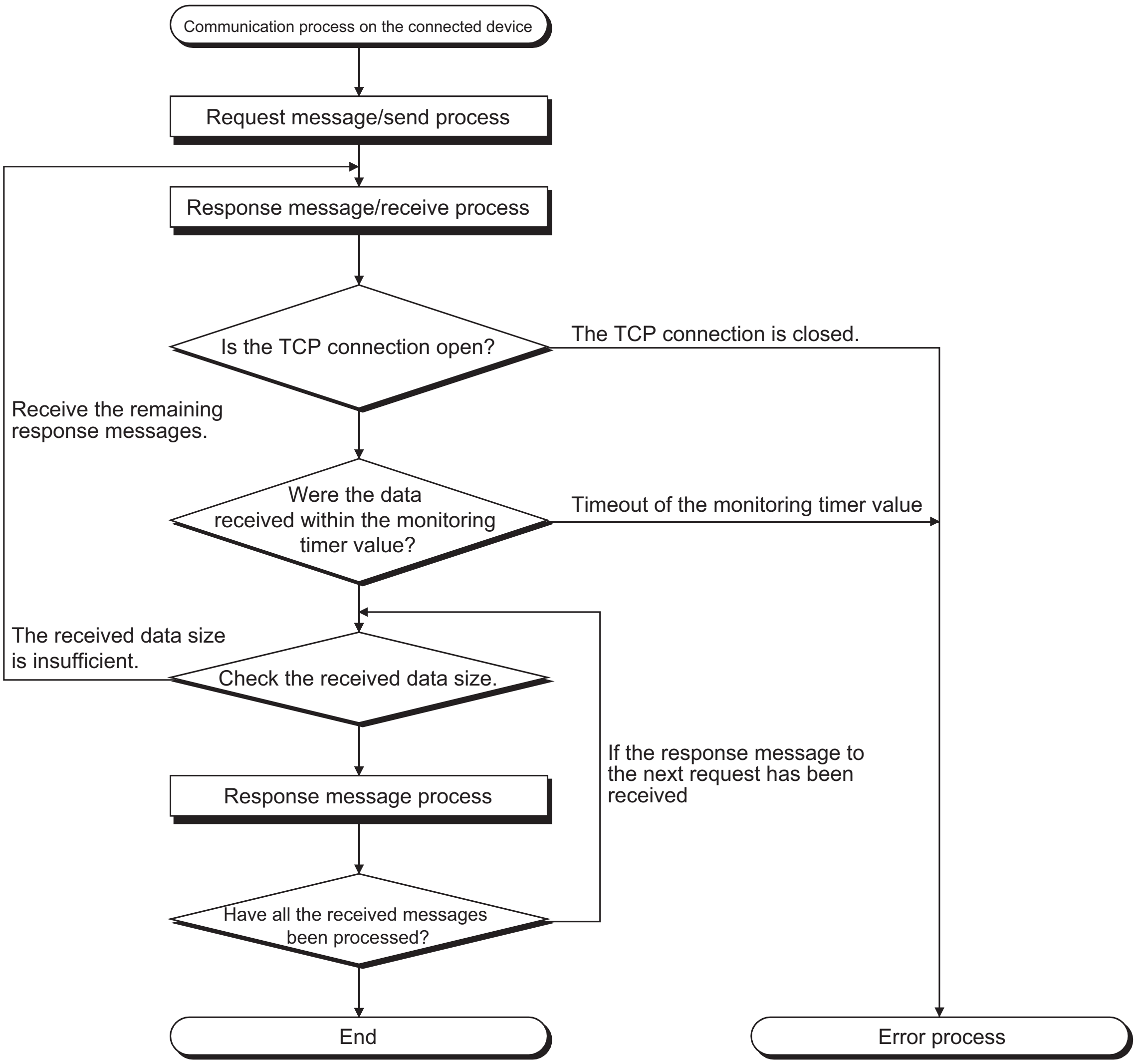
```
            // Result display data is stored.
            for( iNumber = 0 ; iNumber < 5 ; iNumber++ )
            {
                lpszarrData[ iNumber ] = sData[ iNumber ].ToString();
            }
            szReadData = String::Join(",",lpszarrData);
            MessageBox::Show(String::Format("D0-D4 = {0}",szReadData));
        } else {
            MessageBox::Show( String::Format( "Read Error:0x{0:x8} [HEX]", __box(iRet) ) );
        }
}

//****************************************
//Disconnection
//****************************************
private: System::Void button3_Click(System::Object * sender, System::EventArgs * e)
{
        int iRet;

        //Disconnection
        iRet = axActEasyIF1->Close();
        if( iRet == 0 ){
            MessageBox::Show( "The disconnection was successful" );
        } else {
            MessageBox::Show( String::Format( "Disconnection Error:0x{0:x8} [HEX]", __box
            (iRet) ) );
        }
}
```

Appendix 10 Sample Program on the Connected Device Side

The following shows a sample program for the connected device on the same Ethernet network as the E71. Each program example describes only the minimum programming that is required to perform a communication test. Change the IP address, port number and other values according to the system configuration used. Other processes such as a process performed upon an error can be added.

(1) Receive process of the connected device

A receive processing example on the connected device side is shown below.



Point

For Ethernet communications, the TCP socket functions are used inside the personal computer. However, these functions do not have any limit. Therefore, when the "send" function is executed once to transmit data, the receiving side needs to execute the "recv" function once or more to read the data ("send" and "recv" are not proportional to 1:1 execution). For this reason, the receive process above is required.

(2) When the receive process of the connected device is not supported

When the receive process of the connected device is not the one shown in (1), the following may occur when "Enable TCP Maximum Segment Size Option transmission" is set for communications.

- When batch reading is executed from the connected device using the MC protocol, data cannot be normally read.
- After replacing the E71 (which does not support the TCP Maximum Segment Size Option transmission function) with the alternative module supporting the function, data cannot be normally read.
- Even though the value in Received TCP packet count (Address: $1B8_H$ and $1B9_H$) in the buffer memory was changed, data cannot be received.

If these errors occur, select "Disable TCP Maximum Segment Size Option transmission".

Appendix 10.1 When Visual C++®.NET is used (single CPU system)

This section describes the program execution environment, data communication details, and a sample program for when Visual C++®.NET is used on the program of a connected device.

(1) Execution environment of the program example

(a) CPU module side

Item		Description
A model name of the QCPU in the E71-mounted station		Q25HCPU
E71 I/O signal		X/Y00 to X/Y1F
E71 IP address		C0.00.01.FD$_H$ (192.00.01.253)
E71 port number		2000$_H$
Programming tool setting	Ethernet operation setting	Refer to (3)(a) in this section.
	Open setting	Refer to (3)(b) in this section.

(b) Connected device side

Item	Description
Operation environment	Microsoft® Windows® XP Professional Operating System Ver.2002 Service Pack2
Ethernet interface board model name	WINSOCK compatible board
Library to link	WSOCK32.LIB
Software development environment	Visual C++®.NET 2003 manufactured by Microsoft® Corporation
MAC address	This setting is not required because the ARP function can be used.
IP address	Received upon an Active open
Port number	Received upon an Active open

(c) Communication protocol

TCP/IP is used.

(2) Outline of the program example

(a) Program on the CPU module side

Because parameters are set on the programming tool, no program is required.

(b) Program on the connected device side

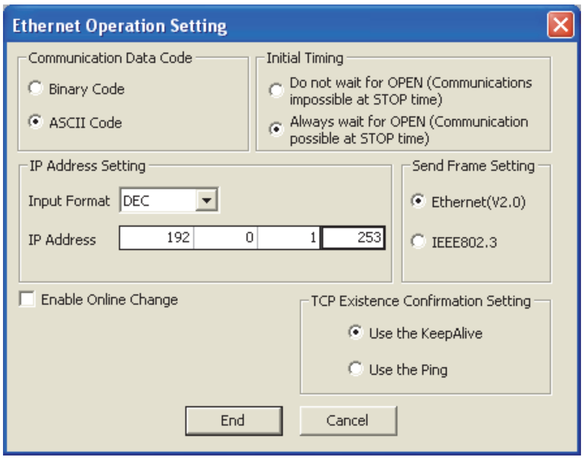
Read/write data from/to the CPU module using the library described above.

- Write in word units (for five points from D0 to D4)
- Read in word units (for five points from D0 to D4)

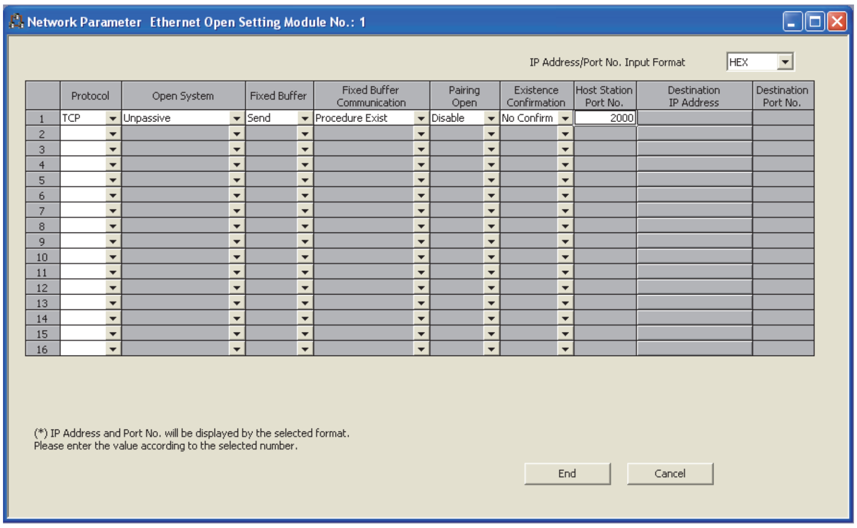
(3) Programming tool setting

Set the parameters using the programming tool as shown below.

(a) Ethernet operation setting



(b) Open setting



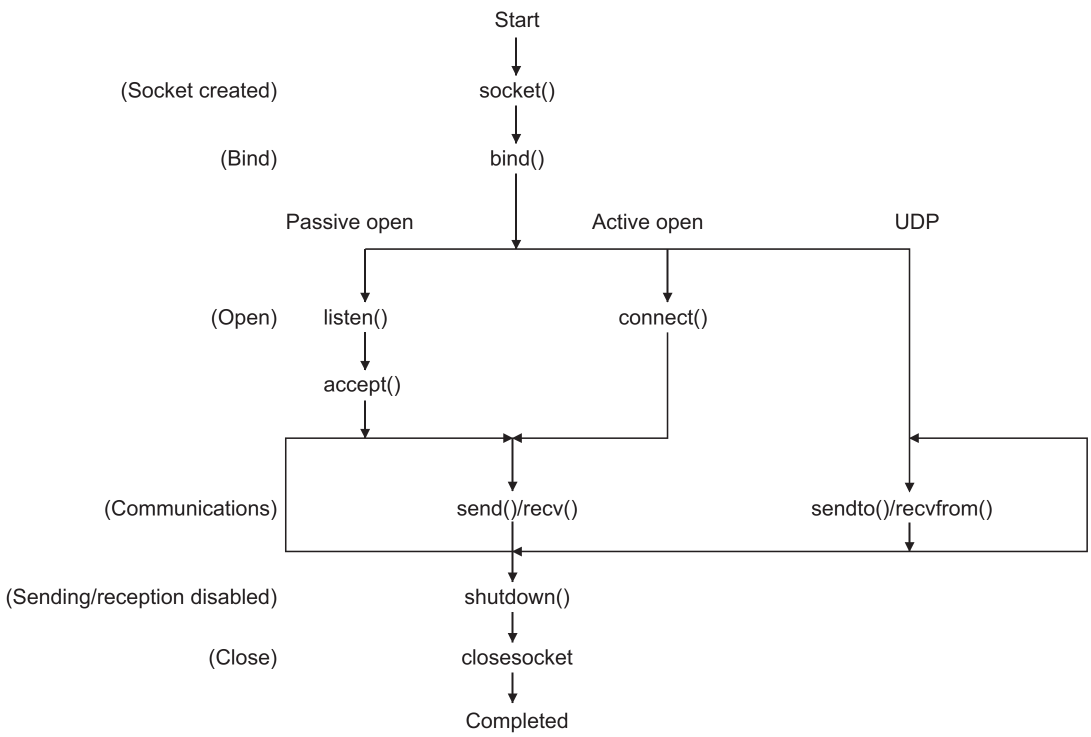
(4) Sample program on the connected device side

The program example of the connected device accessing the Q25HCPU in the E71-mounted station is described. When this program is executed, the contents of the following communication messages are displayed in order:

- Batch write command message in word units
- Batch write response message in word units
- Batch read command message in word units
- Batch read response message in word units

Remark

- The following is an outline of how to compile a program created using Microsoft® Corporation Visual C++®.NET.
  - Start Visual C++®.NET.
  - Create a project. From [File]→[New]→[Project], select ".NET" in "Project Types" and "Empty Project" in "Templates", and set the project name (e.g. AJSAMP) and location.
  - Create a source file. Open Solution Explorer. Then right-click Source Files and select [Add]→[Add New Item]. Set the file name (e.g. AJSAMP.cpp) and location, and create a program according to the program example.
  - From the project setting window, get WSOCK32.LIB linked. Open Solution Explorer. Then right-click the project name (AJSAMP) and select [Properties]→[Configuration Properties]→[Linker]→[Command Line]. Type WSOCK32.LIB in Additional Options and press the OK button.
  - On the Build menu, click Build Solution to create an execution file (AJSAMP.EXE).
  - Exit Visual C++®.NET.
  - Execute AJSAMP.EXE.
- The following is an outline of how to implement a socket routine call.



```
/**************************************************************************/
/**      Sample program (program name: AJSAMP.CPP)          **/
/**                                                         **/
/**      This program is a sample program to conduct a      **/
/**      connection test between the E71 and target device. **/
/**      This program accesses the data register (D) of     **/
/**      the PLC CPU installed together with the E71        **/
/**                                                         **/
/**      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_1  22  // Receiving size of response message in reply to device write (3E frame)
//#define RECV_ANS_2  24  // Receiving size of response message in reply to device read (1E frame)
#define RECV_ANS_2  42  // Receiving size of response message in reply to device read (3E frame)

typedef struct sck_inf{
      struct in_addr my_addr;
      unsigned short my_port;
      struct in_addr aj_addr;
      unsigned short aj_port;
}sck_inf;
```

(To the next page)

```
int nErrorStatus;                                      // Error information storage variable
int Dmykeyin;                                          // Dummy key input
int Closeflag;                                         // Connection completion flag
int 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 aj71e71;                        // E71 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.aj_addr.s_addr=inet_addr("192.0.1.253");        // E71 side IP address (C00001FDH)
    sc.aj_port=htons(0x2000);                          // E71 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("AJ_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);
    }
```

(To the next page)

```
        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);
        }

        aj71e71.sin_family=AF_INET;
        aj71e71.sin_port=sc.aj_port;
        aj71e71.sin_addr.s_addr=sc.aj_addr.s_addr;

        if(connect(socketno,(LPSOCKADDR)&aj71e71,sizeof(aj71e71))!=SOCK_OK){
                                                                 // Connection (Active open)
                Sockerror(ERROR_CONNECT);                           // Error handling
                return(SOCK_NG);
        }

        Closeflag=FLAG_ON;                                      // Connection completion flag ON

        // Set to non-blocking mode
        ulCmdArg = 1;
        ioctlsocket(socketno, FIONBIO, &ulCmdArg);              // Set to non-blocking mode

//      strcpy((char *)(s_buf), "03FF000A4420000000000500112233445566778899AA");
                                                        // D0 to D4 batch write request (1E frame)
        strcpy((char *)(s_buf),"500000FF03FF00002C000A14010000D
                                                        *0000000005112233445566778899AA");
                                                        // D0 to D4 batch write request (3E frame)

        length = strlen((char *)(s_buf));

        if(send(socketno, (char *)(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);
```

(To the next page)

```
// 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, (char*) (&r_buf[rbuf_idx]), (BUF_SIZE - rbuf_idx), 0);
                                                 // Response data receiving
    if(length == 0) { // Is connection cut off?
        Sockerror(ERROR_RECEIVE);               // Error handling
        return (SOCK_NG);
    }

    if(length == SOCKET_ERROR) {
        nErrorStatus = WSAGetLastError();
        if(nErrorStatus != WSAEWOULDBLOCK) {
            Sockerror(ERROR_RECEIVE);           // 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((char *)(s_buf), "01FF000A4420000000000500");    // D0 to D4 batch read request
                                                            // (1E frame)
strcpy((char *)(s_buf), "500000FF03FF000018000A04010000D*0000000005");
                                                            // D0 to D4 batch read request
                                                            // (3E frame)

length = strlen((char *)(s_buf));

if(send(socketno, (char *)(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);
```

(To the next page)

```
    // 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, (char*)(&r_buf[rbuf_idx]), (BUF_SIZE - rbuf_idx), 0);
                                                     // Response data receiving
        if(length == 0) {                            // Is connection cut off?
            Sockerror(ERROR_RECEIVE);                // Error handling
            return (SOCK_NG);
        }

    if(length == SOCKET_ERROR) {
        nErrorStatus = WSAGetLastError();
        if(nErrorStatus != WSAEWOULDBLOCK) {
            Sockerror(ERROR_RECEIVE);                // 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
```

(To the next page)

```
        printf("\nAJ_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_RECEIVE:
                        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 10.2 When Visual C++®.NET is used (redundant system)

This section describes the program execution environment, data communication details, and a sample program for when Visual C++®.NET is used on the program of the connected device.

(1) Execution environment of the program example

(a) CPU module side

Item		Description
A model name of the QCPU in the E71-mounted station		Q25PRHCPU
E71 I/O signal		X/Y00 to X/Y1F
E71 IP address	System A	C0.00.01.FC_H (192.00.01.252)
	System B	C0.00.01.FD_H (192.00.01.253)
E71 port number		2000_H
Programming tool setting	Ethernet operation setting	Refer to (3)(a) in this section.
	Open setting	Refer to (3)(b) in this section.
	Redundant setting	Refer to (3)(c) in this section.

(b) Connected device side

Item	Description
Operation environment	Microsoft® Windows® XP Professional Operating System Ver.2002 Service Pack2
Ethernet interface board model name	WINSOCK compatible board
Library	WSOCK32.LIB
Software development environment	Visual C++®.NET 2003 manufactured by Microsoft® Corporation
MAC address	This setting is not required because the ARP function can be used.
IP address	Received upon an Active open
Port number	Received upon an Active open

(c) Communication protocol

TCP/IP is used.

(2) Outline of the program example

(a) Program on the CPU module side

Because parameters are set on a programming tool, no program is required.

(b) Program on the connected device side

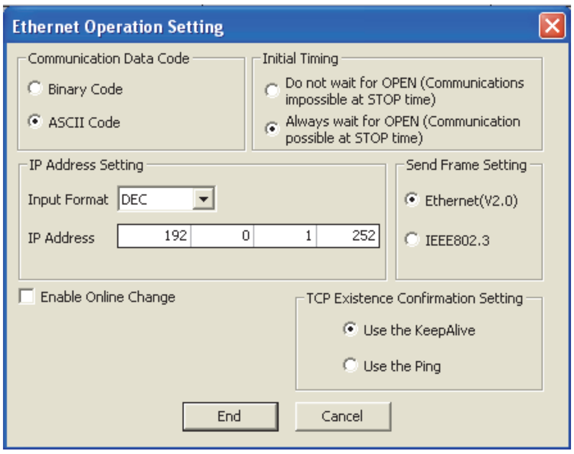
Write data to the data registers D0 to D4 (five points) of the redundant CPU (control system) in word units using the library described above. If data write from the system A side fails due to a communication error or other reasons at this time, data are written from the system B side to the data registers of the redundant CPU (control system).

A

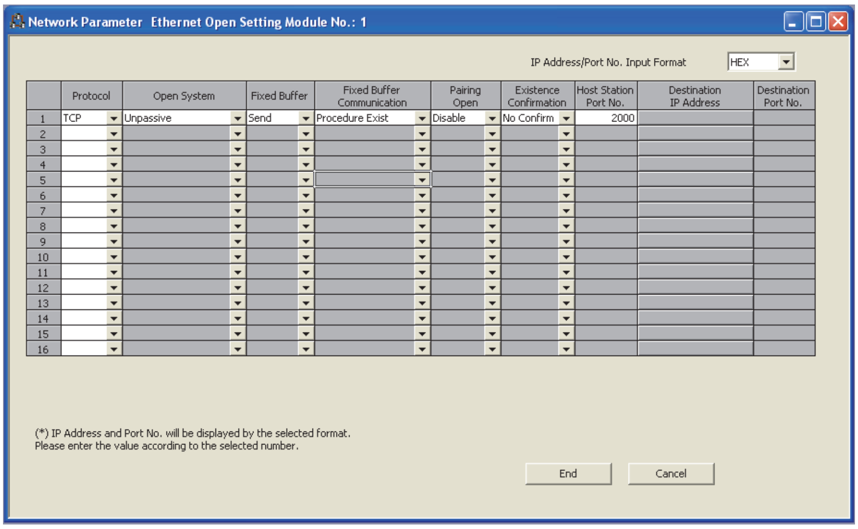
(3) Programming tool setting

Set the parameters using a programming tool as shown below.

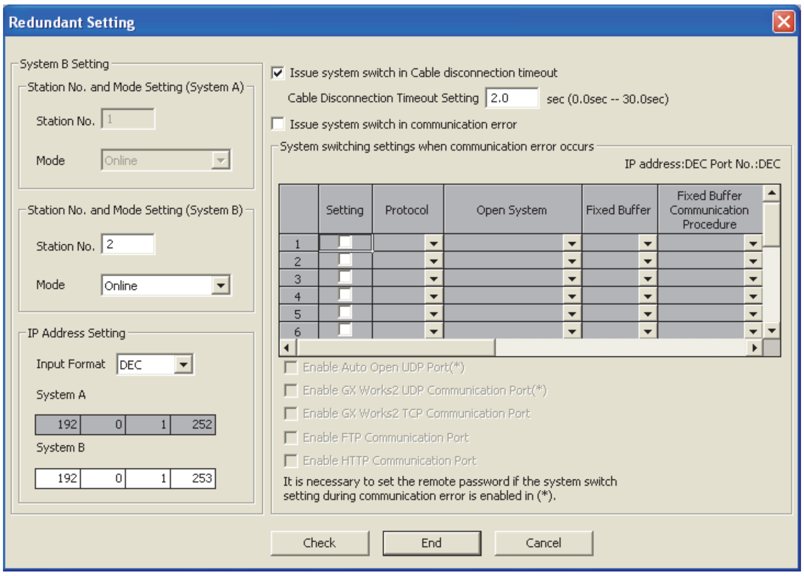
(a) Ethernet operation setting



(b) Open setting



(c) Redundant setting



(4) Sample program on the connected device

The program example of the connected device accessing the Q25PRHCPU in the E71-mounted station is described. When this program is executed, the contents of the following communication messages are displayed in order:

- Version of used Winsock
- Test starting message
- Batch write command message in word units
- Batch read response message in word units
- Test end message

Remark

The following is an outline of how to compile a program created using Microsoft® Corporation Visual C++®.NET.

- Start Visual C++®.NET.
- Create a project. From [File]→[New]→[Project], select ".NET" in "Project Types" and "Empty Project" in "Templates", and set the project name (e.g. QJSAMP) and location.
- Create a source file. Open Solution Explorer. Then right-click Source Files and select [Add]→[Add New Item]. Set the file name (e.g. QJSAMP.cpp) and location, and create a program according to the program example.
- From the project setting window, get WSOCK32.LIB linked. Open Solution Explorer. Then right-click the project name (QJSAMP) and select [Properties]→[Configuration Properties]→[Linker]→[Command Line]. Type WSOCK32.LIB in Additional Options and press the OK button.
- On the Build menu, click Build Solution to create an execution file (QJSAMP.EXE).
- Exit Visual C++®.NET.
- Execute QJSAMP.EXE.

```
/***************************************************************************/
/**   Sample program (program name: QJSAMP.CPP)                   **/
/**                                                                **/
/**   This program is a sample program for testing the            **/
/**   connection of the E71 and external device.                  **/
/**   This program accesses the data registers (D) of             **/
/**   the redundant CPU (control system) mounted with the E71. **/
/**                                                                **/
/**   Copyright(C) 2005 Mitsubishi Electric Corporation           **/
/**   All Rights Reserved                                          **/
/***************************************************************************/
  #include <stdio.h>
  #include <winsock.h>

      #define FLAG_OFF              0         // End flag OFF
      #define FLAG_ON               1         // End flag ON
      #define SOCK_OK               0         // Normal terminationP
      #define SOCK_NG              -1         // Unsuccessful termination
      #define BUF_SIZE           4096         // Receive buffer size

      #define ERROR_NO_ERROR        0         // No error
      #define ERROR_INITIAL         1         // Initial error
      #define ERROR_SOCKET          2         // Socket creation error
      #define ERROR_BIND            3         // Bind error
      #define ERROR_CONNECT         4         // Connect error
      #define ERROR_SEND            5         // Send error
      #define ERROR_SHUTDOWN        6         // Shutdown error
      #define ERROR_CLOSE           7         // Line close error

  // Definition for checking receive size
  #define RECV_ANS_1     22     // Response message receive size in reply to device write (3E frame)

  typedef struct sck_inf{
        struct in_addr my_addr;
        unsigned short my_port;
        struct in_addr qj_addr;
        unsigned short qj_port;
  } sck_inf;


  int nErrorStatus;                          // Error information storage variable
  int Dmykeyin;                              // Dummy key input
  int ShutdownflagA;                         // Shutdown flag (for System A connection)
  int ShutdownflagB;                         // Shutdown flag (for System B connection)
```

(To the next page)

```
int CloseflagA;            // Connection end flag (for System A connection)
int CloseflagB;            // Connection end flag (for System B connection)
int socketnoA;
int socketnoB;
int ConnectLastErrorA;     // Connect processing error information (for System A connection)
int ConnectLastErrorB;     // Connect processing error information (for System B connection)
int SendFlag; // Send completion flag

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_bufA[BUF_SIZE],r_bufB[BUF_SIZE];   // Receive buffer
    struct sck_inf scA,scB;
    struct sockaddr_in hostdataA,hostdataB;            // External device side data
    struct sockaddr_in qj71e71A,qj71e71B;              // E71 side data
    BOOL DataRecv(int, unsigned char *, int);          // Receive processing function
    void Sockerror(int, int);                          // Error handling function

    unsigned long ulCmdArgA,ulCmdArgB;                 // Non-blocking mode setting flag
    scA.my_addr.s_addr = scB.my_addr.s_addr = htonl(INADDR_ANY);
                                                       // External device side IP address
    scA.my_port = scB.my_port = htons(0);              // External device side port number
    scA.qj_addr.s_addr = inet_addr("192.0.1.252");
                                        // E71 side IP address (System A: C00001FCh)
    scB.qj_addr.s_addr = inet_addr("192.0.1.253");
                                        // E71 side IP address (System B: C00001FDh)
    scA.qj_port = scB.qj_port = htons(0x2000);         // E71 side port number
    ShutdownflagA = ShutdownflagB = FLAG_OFF;          // Shutdown flag OFF
    CloseflagA = CloseflagB = FLAG_OFF;                // Connection end flag OFF

    nErrorStatus = WSAStartup(wVersionRequested, &wsaData);   // Winsock initial processing

    ConnectLastErrorA = ERROR_NO_ERROR;
                      // Connect processing error information initialization (for System A)
    ConnectLastErrorB = ERROR_NO_ERROR;
                      // Connect processing error information initialization (for System B)
    if(nErrorStatus != SOCK_OK) {
        Sockerror(ERROR_INITIAL, ERROR_INITIAL);       // Error handling
        return (SOCK_NG);
    }

    printf("Winsock Version is %1d.%1d\n", HIBYTE(wsaData.wVersion),
                                              LOBYTE(wsaData.wVersion));
    printf("QJ_test Start\n");
```

(To the next page)

```
// System A connect processing
socketnoA = socket(AF_INET, SOCK_STREAM, 0);
                                    // TCP/IP socket (for System A connection) creation
if(socketnoA != INVALID_SOCKET) {
    hostdataA.sin_family = AF_INET;
    hostdataA.sin_port = scA.my_port;
    hostdataA.sin_addr.s_addr = scA.my_addr.s_addr;

    if(bind(socketnoA, (LPSOCKADDR)&hostdataA, sizeof(hostdataA)) == SOCK_OK) {
                                              // Bind (System A)
        qj71e71A.sin_family = AF_INET;
        qj71e71A.sin_port = scA.qj_port;
        qj71e71A.sin_addr.s_addr = scA.qj_addr.s_addr;

        if(connect(socketnoA, (LPSOCKADDR)&qj71e71A, sizeof(qj71e71A)) ==
                                                        SOCK_OK) {
                                              // Connect (Active open: System A)
            ShutdownflagA = FLAG_ON;          // Shutdown flag ON
            CloseflagA = FLAG_ON;             // Connection end flag ON
            // Set to non-blocking mode
            ulCmdArgA = 1;
            ioctlsocket(socketnoA, FIONBIO, &ulCmdArgA);
                                 // Set to non-blocking mode (for System A connection)
        } else {
            ConnectLastErrorA = ERROR_CONNECT; // Connection establishment failure
        }
    } else {
        ConnectLastErrorA = ERROR_BIND;       // Bind failure
    }
} else {
    ConnectLastErrorA = ERROR_SOCKET;         // Socket creation failure
}

// System B connect processing
socketnoB = socket(AF_INET, SOCK_STREAM, 0);
                                    // TCP/IP socket (for System B connection) creation
if(socketnoB != INVALID_SOCKET) {
    hostdataB.sin_family = AF_INET;
    hostdataB.sin_port = scB.my_port;
    hostdataB.sin_addr.s_addr = scB.my_addr.s_addr;

    if(bind(socketnoB, (LPSOCKADDR)&hostdataB, sizeof(hostdataB)) == SOCK_OK) {
                                              // Bind (System A)
        qj71e71B.sin_family = AF_INET;
        qj71e71B.sin_port = scB.qj_port;
        qj71e71B.sin_addr.s_addr = scB.qj_addr.s_addr;
```

(To the next page)

```
            if(connect(socketnoB,(LPSOCKADDR)&qj71e71B,sizeof(qj71e71B))== SOCK_OK) {
                                                  // Connect (Active open: System B)
                ShutdownflagB = FLAG_ON;            // Shutdown flag ON
                CloseflagB = FLAG_ON;               // Connection end flag ON
                // Set to non-blocking mode
                ulCmdArgB = 1;
                ioctlsocket(socketnoB, FIONBIO, &ulCmdArgB);
                                    // Set to non-blocking mode (for System B connection)
            } else {
                ConnectLastErrorB = ERROR_CONNECT; // Connection establishment failure
            }
        } else {
            ConnectLastErrorB = ERROR_BIND;          // Bind failure
        }
    } else {
        ConnectLastErrorB = ERROR_SOCKET;            // Socket creation failure
    }
    // Connect completion processing
        if( (CloseflagA == FLAG_OFF) && (CloseflagB == FLAG_OFF) ){
                                                  // When both systems are abnormal
            Sockerror(ConnectLastErrorA, ConnectLastErrorB);     // Error handling
            return (SOCK_NG);
        }

    strcpy((char*)(s_buf), "500000FF03D000002C000A14010000D
                                              *000000000511223344556677889 9AA");
                          // D0-D4 batch write request (3E frame, addressed to control system)
    length = strlen((char*)(s_buf));

    printf("Send starts. Press any key. \n");
    Dmykeyin = getchar();                         // Waiting for key input

    SendFlag = FLAG_OFF;                              // Send completion flag OFF
    // System A send processing
    if( CloseflagA == FLAG_ON && (SendFlag == FLAG_OFF) ){
        if(send(socketnoA, (char*)(s_buf), length, 0) != SOCKET_ERROR) {
                                                  // Data send (System A)
            printf("\n Send data (System A) \n%s\n", s_buf);   // Send data display (System A)
            SendFlag = FLAG_ON;                       // Send completion flag ON
                                                  // Receive processing
            if(DataRecv(socketnoA, r_bufA, RECV_ANS_1) == TRUE) {  // Data receive
                printf("\n Receive data (System A) \n%s\n", r_bufA);    // Receive data display
            } else {
                printf("Receive failure (System A) \n");
            }
        } else {
            printf("Send failure (System A) \n");
        }
    }
```
(To the next page)

```
    // System B send processing
    if( (CloseflagB == FLAG_ON) && (SendFlag == FLAG_OFF) ){
        if(send(socketnoB, (char*)(s_buf), length, 0) != SOCKET_ERROR) {
                                                        // Data send (System B)
            printf("\n Send data (System B) \n%s\n", s_buf);   // Send data display (System B)
            SendFlag = FLAG_ON;                              // Send completion flag ON
                                                        // Receive processing
            if(DataRecv(socketnoB, r_bufB, RECV_ANS_1) == TRUE) {  // Data receive
                printf("\n Receive data (System B) \n%s\n", r_bufB);   // Receive data display
            } else {
                printf("Receive failure (System B) \n");
            }
        }else{
            printf("Send failure (System B) \n");
        }
    }

    // Send completion processing
    if( SendFlag == FLAG_OFF ){
        Sockerror(ERROR_SEND, ERROR_SEND);                   // Error handling
        return (SOCK_NG);
    }
    if(CloseflagA == FLAG_ON) {
        ShutdownflagA = FLAG_OFF;                            // Shutdown flag OFF
        if(shutdown(socketnoA, 2) != SOCK_OK) { // Send/receive inhibit processing (System A)
            Sockerror(ERROR_SHUTDOWN, ERROR_NO_ERROR);       // Error handling
            return (SOCK_NG);
        }
    }
    if(CloseflagB == FLAG_ON) {
        ShutdownflagB = FLAG_OFF;                            // Shutdown flag OFF
        if(shutdown(socketnoB, 2) != SOCK_OK) { // Send/receive inhibit processing (System B)
            Sockerror(ERROR_NO_ERROR, ERROR_SHUTDOWN);       // Error handling
            return (SOCK_NG);
        }
    }
    CloseflagA = FLAG_OFF;                                   // Connection end flag OFF
    if(closesocket(socketnoA) != SOCK_OK) {                  // Close processing (System A)
        Sockerror(ERROR_CLOSE, ERROR_NO_ERROR);              // Error handling
        return (SOCK_NG);
    }
    CloseflagB = FLAG_OFF;                                   // Connection end flag OFF
    if(closesocket(socketnoB) != SOCK_OK) {                  // Close processing (System B)
        Sockerror(ERROR_NO_ERROR, ERROR_CLOSE);              // Error handling
        return (SOCK_NG);
    }
```

(To the next page)

```
    WSACleanup();                                        // Winsock.DLL release

    printf("\nQJ_test End. \n\nNormally completed. \n");
    printf("Program is closed. Press any key. \n");
    Dmykeyin = getchar();                                // Waiting for key input

    return (SOCK_OK);
}

BOOL DataRecv(int socketno, unsigned char *pR_buf, int size_max)  // Receive processing function
{
    int length;                                      // Communication data length
    int rbuf_idx;                                    // Receive data storage starting index
    int recv_size;                                   // Number of received data

    // Performs receive processing while simultaneously making size check
    rbuf_idx = 0;                          // Receive data storage starting index initialization
    recv_size = 0;                         // Initializes the number of received data
    while(1) {
        length = recv(socketno, ((char*)(pR_buf + rbuf_idx)), (BUF_SIZE - rbuf_idx), 0);
                                                     // Response data receive
        if(length == 0) {                            // Has connection been cut?
            return (FALSE);                          // Error handling
        }
        if(length == SOCKET_ERROR) {
            nErrorStatus = WSAGetLastError();
            if(nErrorStatus != WSAEWOULDBLOCK) {
                return (FALSE);                      // Error handling
            } else {
                continue;                            // Repeated until data are received
            }
        } else {
            rbuf_idx += length;                      // Updates receive data storage position
            recv_size += length;                     // Updates the number of received data
            if(recv_size >= size_max)                // Have all response messages received?
                break;                               // Stops repeating as data are received
        }
    }
    *(pR_buf + rbuf_idx) = '\0' ;                    // At the end of received data
                                                     // set NULL
    return (TRUE);                                   // Normal termination
}
```

(To the next page)

```
void Sockerror(int error_kind_A, int error_kind_B) // Error handling function
{
        if (error_kind_A == ERROR_INITIAL){
                printf("Initial processing is abnormal. \n");
        }
        else{
                nErrorStatus = WSAGetLastError();
                switch(error_kind_A){
                case ERROR_SOCKET:
                        printf("Socket could not be created. (System A)\n");
                        break;
                case ERROR_BIND:
                        printf("Bind could not be executed. (System A)\n");
                        break;
                case ERROR_CONNECT:
                        printf("Connection could not be established. (System A)\n");
                        break;
                case ERROR_SEND:
                        printf("Send could not be executed. \n");
                        break;
                case ERROR_SHUTDOWN:
                        printf("Shutdown could not be executed. (System A)\n");
                        break;
                case ERROR_CLOSE:
                        printf("Normal close could not be executed. (System A)\n");
                        break;
                }
                switch(error_kind_B){
                case ERROR_SOCKET:
                        printf("Socket could not be created. (System B)\n");
                        break;
                case ERROR_BIND:
                        printf("Bind could not be executed. (System B)\n");
                        break;
                case ERROR_CONNECT:
                        printf("Connection could not be established. (System B)\n");
                        break;
                case ERROR_SHUTDOWN:
                        printf("Shutdown could not be executed. (System B)\n");
                        break;
                case ERROR_CLOSE:
                        printf("Normal close could not be executed. (System B)\n");
                        break;
                }
        }
        printf("Error code is %d. \n", nErrorStatus);
```

(To the next page)

```
    if (ShutdownflagA == FLAG_ON){
        nErrorStatus = shutdown(socketnoA, 2);         // Shutdown processing (System A)
        ShutdownflagA = FLAG_OFF;                      // Shutdown flag OFF (System A)
    }
    if (ShutdownflagB == FLAG_ON){
        nErrorStatus = shutdown(socketnoB, 2);         // Shutdown processing (System B)
        ShutdownflagB = FLAG_OFF;                      // Shutdown flag OFF (System B)
    }
    if (CloseflagA == FLAG_ON){
        nErrorStatus = closesocket(socketnoA);         // Close processing (System A)
        CloseflagA = FLAG_OFF;                         // Connection end flag OFF (System A)
    }
    if (CloseflagB == FLAG_ON){
        nErrorStatus = closesocket(socketnoB);         // Close processing (System B)
        CloseflagB = FLAG_OFF;                         // Connection end flag OFF (System B)
    }

    printf("Program is closed. Press any key. \n");
    Dmykeyin = getchar();                              // Waiting for key input
    WSACleanup();                                      // Winsock.DLL release
    return;
}
```

Appendix 10.3 When Visual Basic®.NET is used

This section describes the program execution environment, data communication details, and a sample program for when Visual Basic®.NET is used on the program of the connected device.

(1) Execution environment of the program example

(a) CPU module side

Item		Description
A model name of the QCPU in the E71-mounted station		Q25HCPU
E71 I/O signal		X/Y00 to X/Y1F
E71 IP address		$C0.00.01.FD_H$ (192.00.01.253)
E71 port number		2000_H
Programming tool Setting	Ethernet operation setting	Refer to (3)(a) in this section.
	Open setting	Refer to (3)(b) in this section.

(b) Connected device side

Item	Description
Operation environment	Microsoft® Windows® XP Professional Operating System Ver.2002 Service Pack2
Ethernet interface board model name	WINSOCK compatible board
Software development environment	Visual Basic®.NET 2003 manufactured by Microsoft® Corporation
MAC address	This setting is not required because the ARP function can be used.
IP address	An arbitrary number is assigned.
Port number	An arbitrary number is assigned.

(c) Communication protocol

TCP/IP is used.

(2) Outline of the program example

(a) Program on the CPU module side

Because parameters are set on the programming tool, no program is required.

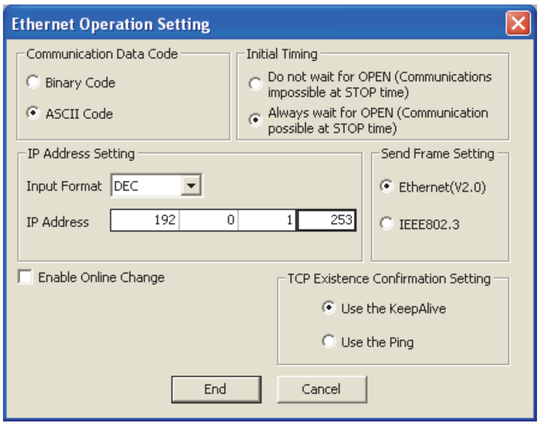
(b) Program on the connected device side

Data (D0 to D4) in the CPU module are read out.

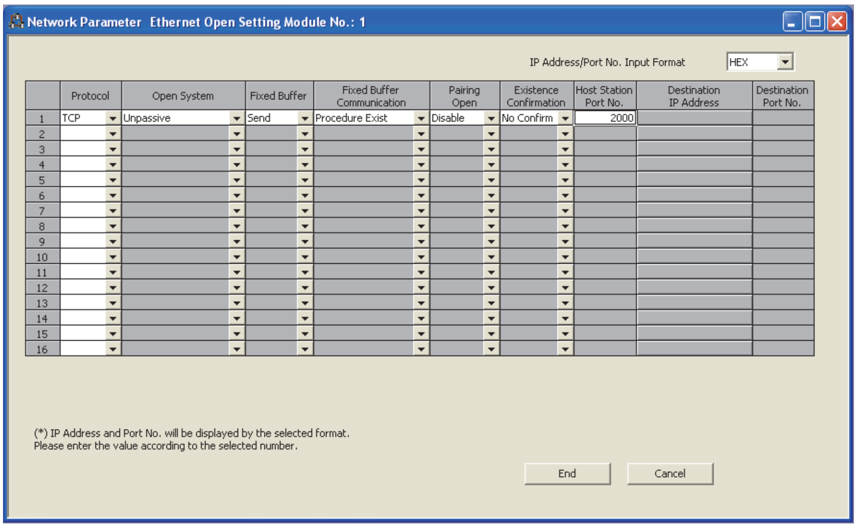
(3) Programming tool setting

Set the parameters using the programming tool as shown below.

(a) Ethernet operation setting



(b) Open setting



(4) Sample program on the connected device side

The program example of the connected device accessing the Q25HCPU in the E71-mounted station is described. In this program, data of D0 to D4 (five points) in the Q25HCPU in the E71-mounted station are read with an A-compatible 1E frame command (01: Batch read in word units). The following are basic operation procedures:

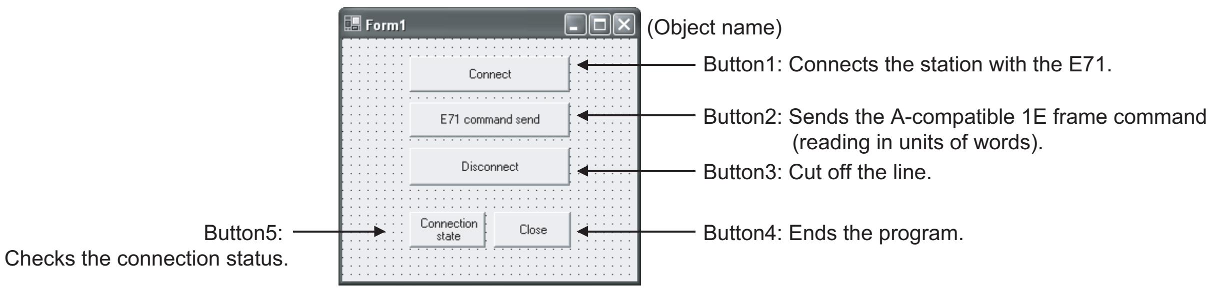
- Sending a command with the line disconnected (Connect the line. After completing the connection, the E71 sends a command.)
- Reconnecting the line with it connected (Disconnect the line. After the disconnection is completed, reconnect it.)

Remark

The following is an outline of how to compile a program created using Microsoft® Corporation Visual Basic®.NET.

- Start Visual Basic®.NET.
- Create a project. From [File]→[New]→[Project], select "Visual Basic Project" in "Project Types" and "Windows Application" in "Templates", and set the project name (e.g. AJSAMP) and location.
- Create a form and a program. Use the toolbox "Button" to create the sample window shown below (Form1.vb) and create a program referring to the sample program.
- On the Build menu, click Build Solution to create an execution file (AJSAMP.EXE).
- Exit the Visual Basic®.NET.
- Execute the AJSAMP.EXE.

[Window example (Form 1.vb)]



[Sample program (Form 1.vb)]

```
Option Strict Off
Option Explicit ON

Imports System
Imports System.Text
Imports System.Net

Friend Class Form1
	Inherits System. Windows. Forms. Form
#Region "Windows Form Designer generated code"
	Public Sub New()
	MyBase.New()
	If m_vb6FormDefInstance Is Nothing Then
		If m_InitializingDefInstance Then
			m_vb6FormDefInstance = Me
		Else
				Try
				'For the start-up form, the first instance created is the default instance.
				If System. Reflection. Assembly. GetExecutingAssembly. EntryPoint
					.DeclaringType Is Me.GetType Then
					m_vb6FormDefInstance = Me
				EndIf
			Catch
			End Try
		End If
	End If
	' This call is required by the Windows form designer.
	InitializeComponent()
End Sub
```

(To the next page)

```
'Form overrides dispose to clean up 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
    'Required by the Windows Form Designer.
    Private components As System.ComponentModel.IContainer
    Public WithEvents Command5 As System.Windows.Forms.Button
    Public WithEvents Command4 As System.Windows.Forms.Button
    Public WithEvents Command3 As System.Windows.Forms.Button
    Public WithEvents Command2 As System.Windows.Forms.Button
    Public WithEvents Command1 As System.Windows.Forms.Button
    Dim Ajsock As Sockets.Socket
    Private State As Boolean = False

        'NOTE: The following procedure is required by the Windows Form Designer.
        'It can be modified using the Windows Form Designer.
        'Do not modify it using the code editor.
    <System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()
        Me.Command5 = New System.Windows.Forms.Button
        Me.Command4 = New System.Windows.Forms.Button
        Me.Command3 = New System.Windows.Forms.Button
        Me.Command2 = New System.Windows.Forms.Button
        Me.Command1 = New System.Windows.Forms.Button
        Me.SuspendLayout()
        '
        'Command5
        '
        Me.Command5.BackColor = System.Drawing.SystemColors.Control
        Me.Command5.Cursor = System.Windows.Forms.Cursors.Default
        Me.Command5.ForeColor = System.Drawing.SystemColors.ControlText
        Me.Command5.Location = New System.Drawing.Point(64, 152)
        Me.Command5.Name = "Command5"
        Me.Command5.RightToLeft = System.Windows.Forms.RightToLeft.No
        Me.Command5.Size = New System.Drawing.Size(72, 32)
        Me.Command5.TabIndex = 4
        Me.Command5.Text = "Connection status"
        '
        'Command4
        '
```

(To the next page)

```
Me.Command4.BackColor = System.Drawing.SystemColors.Control
Me.Command4.Cursor = System.Windows.Forms.Cursors.Default
Me.Command4.ForeColor = System.Drawing.SystemColors.ControlText
Me.Command4.Location = New System.Drawing.Point(144, 152)
Me.Command4.Name = "Command4"
Me.Command4.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Command4.Size = New System.Drawing.Size(73, 32)
Me.Command4.TabIndex = 3
Me.Command4.Text = "Close"
'
'Command3
'
Me.Command3.BackColor = System.Drawing.SystemColors.Control
Me.Command3.Cursor = System.Windows.Forms.Cursors.Default
Me.Command3.ForeColor = System.Drawing.SystemColors.ControlText
Me.Command3.Location = New System.Drawing.Point(64, 96)
Me.Command3.Name = "Command3"
Me.Command3.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Command3.Size = New System.Drawing.Size(152, 33)
Me.Command3.TabIndex = 2
Me.Command3.Text = "disconnect"
'
'Command2
'
Me.Command2.BackColor = System.Drawing.SystemColors.Control
Me.Command2.Cursor = System.Windows.Forms.Cursors.Default
Me.Command2.ForeColor = System.Drawing.SystemColors.ControlText
Me.Command2.Location = New System.Drawing.Point(64, 56)
Me.Command2.Name = "Command2"
Me.Command2.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Command2.Size = New System.Drawing.Size(152, 31)
Me.Command2.TabIndex = 1
Me.Command2.Text = "Sending a E71 command"
'
'Command1
'
Me.Command1.BackColor = System.Drawing.SystemColors.Control
Me.Command1.Cursor = System.Windows.Forms.Cursors.Default
Me.Command1.ForeColor = System.Drawing.SystemColors.ControlText
Me.Command1.Location = New System.Drawing.Point(64, 16)
Me.Command1.Name = "Command1"
Me.Command1.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Command1.Size = New System.Drawing.Size(152, 31)
Me.Command1.TabIndex = 0
Me.Command1.Text = "connect"
```

(To the next page)

```
            '
            'Form1
            '
            Me.AutoScaleBaseSize = New System.Drawing.Size(5, 12)
            Me.BackColor = System.Drawing.SystemColors.Control
            Me.ClientSize = New System.Drawing.Size(280, 214)
            Me.Controls.Add(Me.Command5)
            Me.Controls.Add(Me.Command4)
            Me.Controls.Add(Me.Command3)
            Me.Controls.Add(Me.Command2)
            Me.Controls.Add(Me.Command1)
            Me.Cursor = System.Windows.Forms.Cursors.Default
            Me.Location = New System.Drawing.Point(329, 189)
            Me.Name = "Form1"
            Me.RightToLeft = System.Windows.Forms.RightToLeft.No
            Me.StartPosition = System.Windows.Forms.FormStartPosition.Manual
            Me.Text = "Form1"
            Me.ResumeLayout(False)
        End Sub
    #End Region

#Region "Upgrade Wizard support code"
    Private Shared m_vb6FormDefInstance As Form1
    Private Shared m_InitializingDefInstance As Boolean
    Public Shared Property DefInstance() As Form1
        Get
            If m_vb6FormDefInstance Is Nothing OrElse m_vb6FormDefInstance.IsDisposed Then
                m_InitializingDefInstance = True
                m_vb6FormDefInstance = New Form1()
                m_InitializingDefInstance = False
            End If
            DefInstance = m_vb6FormDefInstance
        End Get
        Set
            m_vb6FormDefInstance = Value
        End Set
    End Property
#End Region
```

(To the next page)

```
Private Sub Command1_Click(ByVal eventSender As System.Object, ByVal eventArgs
    As System.EventArgs) Handles Command1.Click
        'Connect to the Ethernet interfece module.
        Dim sock As New Sockets.Socket(Sockets.AddressFamily.InterNetwork, _
        Sockets.SocketType.Stream, Sockets.ProtocolType.Tcp)
        Ajsock = sock
        Dim ip As IPAddress = Dns.Resolve("192.0.1.253").AddressList(0)
        Dim ipend As IPEndPoint = New IPEndPoint(ip, "8192")

        Me.Ajsock.Connect(ipend)
        MsgBox("Connection Completed")
        State = Me.Ajsock.Connected()

    End Sub

    Private Sub Command2_Click(ByVal eventSender As System.Object, ByVal eventArgs
    As System.EventArgs) Handles Command2.Click
        Dim SData As Byte()
        Dim RData(256) As Byte

        'Rend D0 to D4 (5 points) with the A-compatible 1E frame command.
        SData = Encoding.ASCII.GetBytes("01FF000A4420000000000500")
        'Read D0 to D4 (5 points) with the QnA-compatible 3E frame command.
        'SData = Encoding.ASCII.GetBytes("500000FF03FF000018000A04010000D
                                                                    *0000000005")
        'Send the data.
        Me.Ajsock.Send(SData)
        MsgBox("Send completion", MsgBoxStyle.Information)

        'Read the response from the PLC CPU.
        Me.Ajsock.Receive(RData)
        MsgBox(Encoding.ASCII.GetString(RData), MsgBoxStyle.Information)

    End Sub
```

(To the next page)

```
    Private Sub Command3_Click(ByVal eventSender As System.Object, ByVal eventArgs
    As System.EventArgs) Handles Command3.Click
        'Close the TCP (UDP) connection socket (disconnect the line).
        Me.Ajsock.Shutdown(Net.Sockets.SocketShutdown.Both)
        Me.Ajsock.Close()
        MsgBox("The disconnection was successful", MsgBoxStyle.Information)
        State = Me.Ajsock.Connected()
    End Sub

    Private Sub Command4_Click(ByVal eventSender As System.Object, ByVal eventArgs
    As System.EventArgs) Handles Command4.Click
        'End the program.
        End

    End Sub

    Private Sub Command5_Click(ByVal eventSender As System.Object, ByVal eventArgs
    As System.EventArgs) Handles Command5.Click
        'Check the connection state.
        If State Then
            MsgBox("Connected")
        Else
            MsgBox("Closed")
        End If

    End Sub

End Class
```

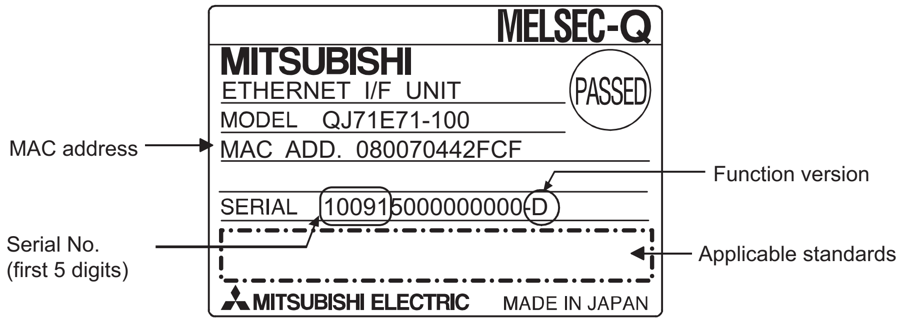
Appendix 11 Checking the Serial Number and Function Version

The serial number and function version of the E71 can be checked with the following.

- Rating plate
- Front of the module
- Programming tool system monitor

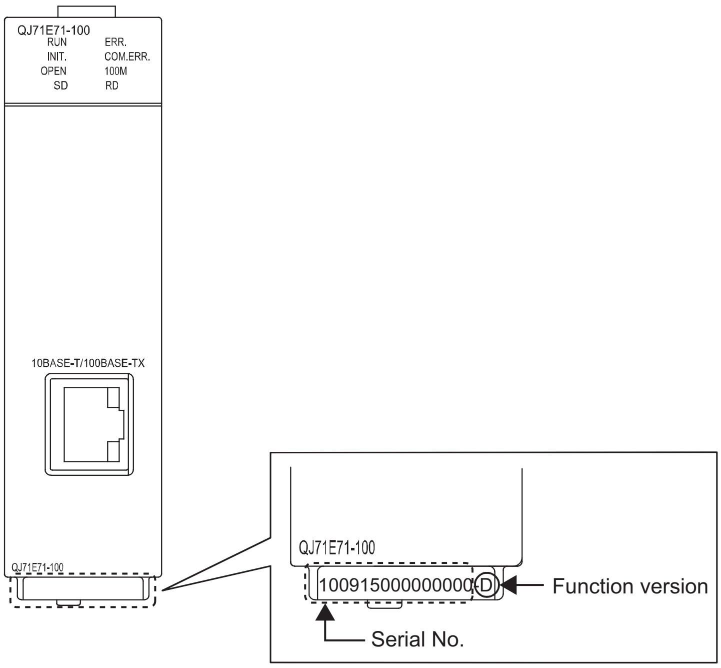
(1) Checking on the rating plate

The rating plate is located on the side of the module. It also shows the MAC address and certification marks of applicable standards.



(2) Checking on the front of the module

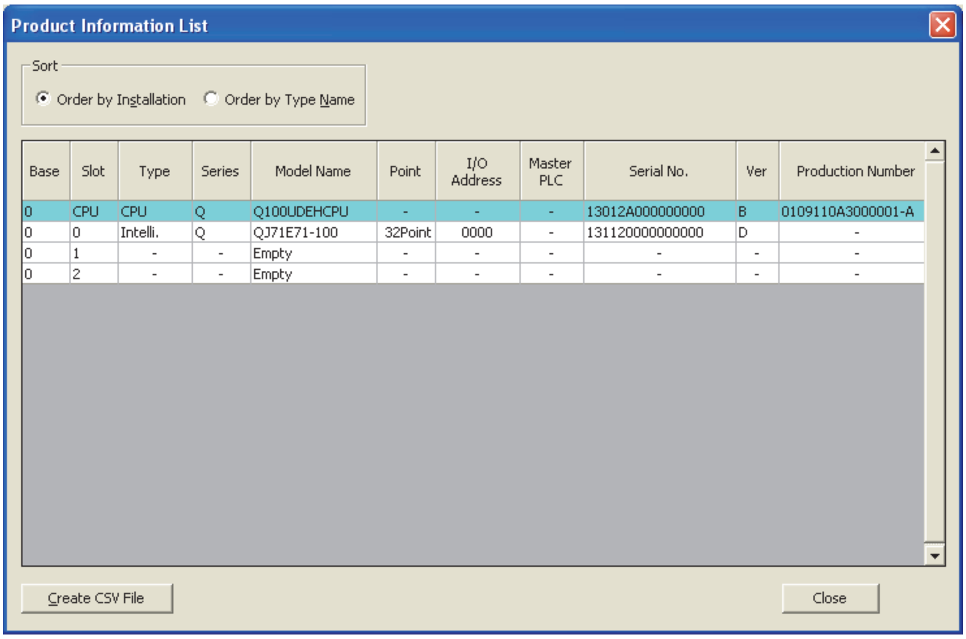
The serial number and function version on the rating plate are printed on the front (at the bottom) of the module.



(3) Checking on the system monitor (product information list window)

The serial number and function version can be checked on the system monitor of the programming tool. The "Serial No." column shows the serial numbers and the "Ver" column shows the function versions. Because an E71 does not support the production number, "-" is displayed on the "Production Number" column.

[Diagnostics] ⇨ [System Monitor] ⇨ Product Information List button



Point

The serial number displayed on the "Product Information List" window of the programming tool may differ from that on the rating plate or on the front of the module.

- The serial number on the rating plate or on the front of the module indicates the management information of the product.
- The serial number displayed on the "Product Information List" window is the functional information of the product. The functional information of the product is updated when a function is added.

Appendix 11.1 Compatible software versions

The following table lists the compatible software versions.

CPU module		Software version	
		GX Developer	GX Works2
Basic model QCPU	Q00(J)/Q01CPU	Version 7 or later*1	Refer to the GX Works2 Version 1 Operating Manual (Common).
High Performance model QCPU	Q02(H)/Q06H/Q12H/Q25HCPU	Version 4 or later*2	
Process CPU	Q02PH/Q06PHCPU	Version 8.68W or later	
	Q12PH/Q25PHCPU	Version 7.10L or later	
Redundant CPU	Q12PRH/Q25PRHCPU	Version 8.18U or later*3	
Universal model QCPU	Q00U(J)/Q01UCPU	Version 8.76E or later	
	Q02U/Q03UD/Q04UDH/ Q06UDHCPU	Version 8.48A or later	
	Q10UDH/Q20UDHCPU	Version 8.76E or later	
	Q13UDH/Q26UDHCPU	Version 8.62Q or later	
	Q03UDE/Q04UDEH/ Q06UDEH/ Q13UDEH/Q26UDEHCPU	Version 8.68W or later	
	Q10UDEH/Q20UDEHCPU	Version 8.76E or later	
	Q50UDEH/Q100UDEHCPU	Incompatible	
Safety CPU	QS001CPU	Version 8.65T or later	
CPU module other than the above		Incompatible	
MELSECNET/H remote I/O station		Version 6 or later	

*1 To use a module in a multiple CPU system, Version 8 or later is required.
*2 To use a module in a multiple CPU system, Version 6 or later is required.
*3 To mount a module on an extension base unit, Version 8.45X or later is required.

Remark

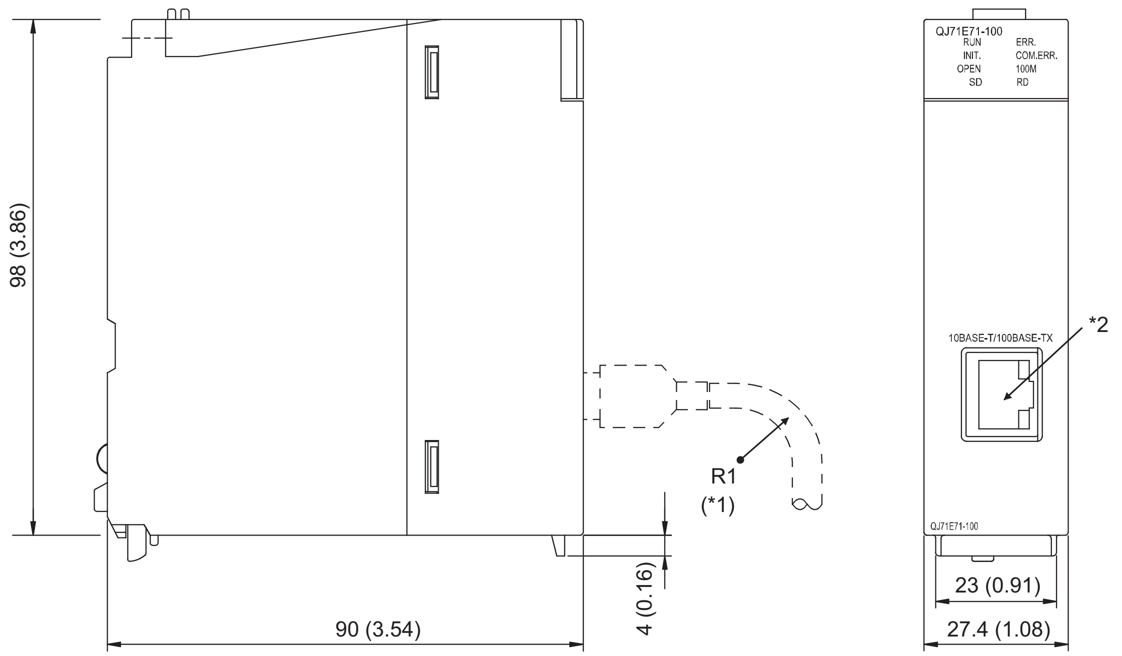
For the compatible versions of MX Component, refer to the following.

MX Component Version 3 Operating Manual

Appendix 12 External Dimension Diagram

This appendix provides the external dimension diagrams of an E71.

(1) QJ71E71-100

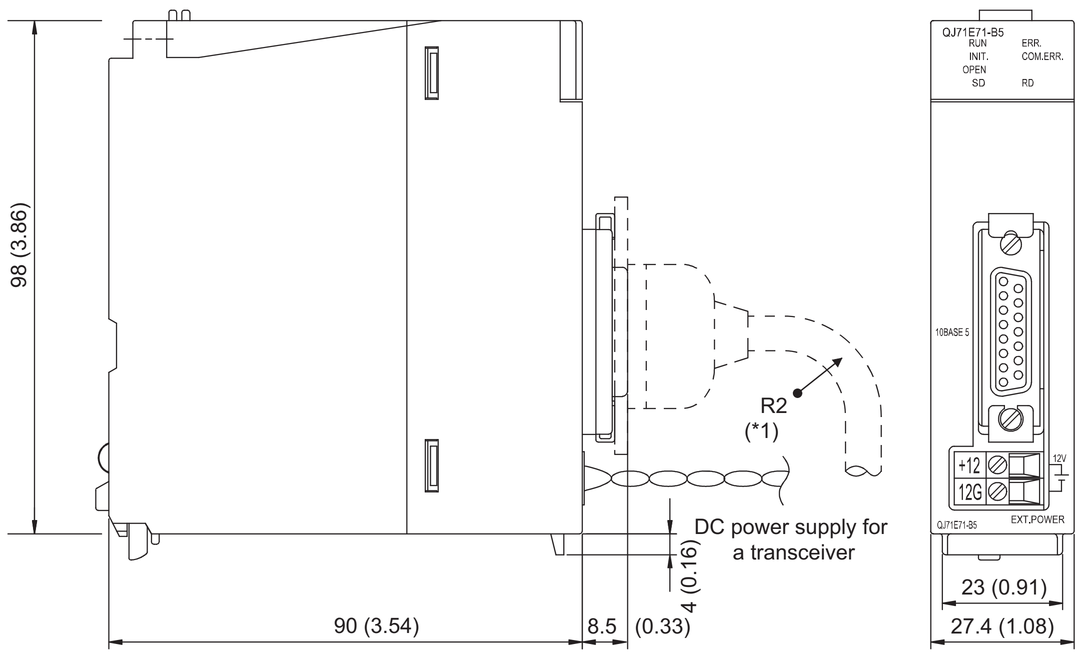


(Unit: mm (in.))

*1 When connecting a twisted pair cable, keep the bending radius near the connector (reference value: R1) above four times the outside diameter of the cable.

*2 The orientation of the connector is different (rotated) depending on the serial number.

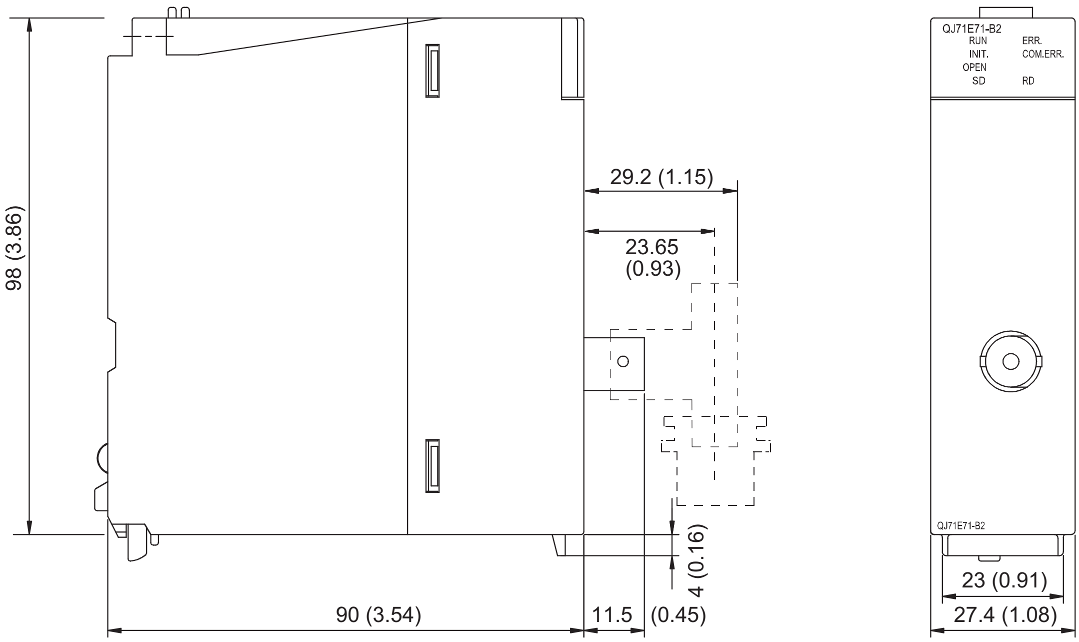
(2) QJ71E71-B5



(Unit: mm (in.))

*1 When connecting an AUI cable, keep the bending radius near the connector (reference value: R2) above four times the outside diameter of the cable.

(3) QJ71E71-B2



(Unit: mm (in.))

A

Appendix 13 ASCII Code List

The following is the ASCII code table.

			Higher 4 bits								
	Hexadecimal		0	1	2	3	4	5	6	7	8–F
		Binary	0000	0001	0010	0111	0100	0101	0110	0111	1000–1111
Lower 4 bits	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	

INDEX

0 to 9

A

B

C

D

E

F

G

H

I

T

U

W

Z

I

REVISIONS

*The manual number is given on the bottom left of the back cover.

Print Date	*Manual Number	Revision
Dec., 1999	SH(NA)-080009-A	First Edition
Oct., 2000	SH(NA)-080009-B	• Reflect the contents of the function version B. • Put Windows® base software products together from Mitsubishi Programmable Controller MELSEC series to Mitsubishi integrated FA software MELSOFT series. • Standardize the name from software package (GPP function) to product name (GX Developer). Correction Entire manual (change MELSECNET/10H to MELSECNET/H), SAFETY PRECAUTIONS, Contents, About Manuals, The Manual's Usage and Structure (Structure of this manual (2)), About the Generic Terms and Abbreviations, Section 1.1, 1.2, 1.3, 2.1, 2.2, 2.3, 2.5, 2.6, 3.1, 3.2, 3.5, 3.6, 3.7, 3.8(2), 4.1.1, 4.2, 4.3, 4.4.1, 4.5 (entire), 4.5.1 (Table), 4.6, 4.7, 4.8, 4.9 (entire), 5.2.1, 5.2.2, 5.3, 5.5, 5.6 (entire), 5.7.2, 5.8, Chapter 6 (entire), Section 7.1, 7.2, 7.3.1, 7.3.2, 7.4.2, 7.5.2, Chapter 8, Section 8.1, 8.2, 8.3.1, 8.3.2, 8.5.1, 8.6.2, 9.2.3, 10.2 to 10.8, Chapter 11, Section 11.1.1, 11.1.2(2), 11.2, 11.2.2, 11.2.4, 11.3 (5) (6), 11.3.1 to 11.3.3, 11.4, 11.4.4, Appendix 1 (entire), Appendix 2 (entire), Appendix 3, Appendix 8 (entire), Appendix 11 Addition Entire manual (add the explanation on MELSECNET/H remote I/O station), The Manual's Usage and Structure (2) (e), Section 5.4, 11.2, 11.2.1, 11.2.3, 11.3.1 (error code 63H), 11.3.3 (error code 0063_H, $C086_H$, $C087_H$, $C0DA_H$, $C0DB_H$, $C119_H$, $C200_H$ to $C205_H$)
Jun., 2001	SH(NA)-080009-C	• Added the description of the model QJ71E71-100 Ethernet interface module. Model Addition QJ71E71-100 Correction Compliance with the EMC and Low Voltage Directives, The Manual's Usage and Structure, About the Generic Terms and Abbreviations, Product Configuration, Section 1.2, 1.3, 1.4 (Figure), 2.1, 2.2, 2.3 (POINT), 2.4, 2.5, 2.7, Chapter 3 (entire), Section 4.2, 4.3, 4.4, 4.7, 4.8.1, 5.1 (Figure), 5.3 (1)(Figure), 5.4, 5.5, 5.6 (POINT), 5.6.1, 5.6.2, 5.8 (2) (3), 5.9.1 (1) (2), 5.9.3 (Figure), 5.9.5, 5.9.6 (3), 5.9.7, 6.2, 7.1, 7.2 (Figure), 7.3.1 (Figure), 7.3.2 (2) (Figure), 7.5.2, 8.2 (Figure), 8.3.1 (Figure), 8.3.2 (2) (Figure), 8.5.1, 8.6.2, 9.2.3 (2) (Figure), 10.1 (POINT), 10.6, 10.8, 11.1.1, 11.1.2, 11.2.3, 11.3 (7) (Figure), 11.3.3 (error codes $C113_H$, $C114_H$, and $C14E_H$), 11.4 (POINT), 11.4.6 (Figure), Appendix 1.1, Appendix 2.1, 2.2 (2), Appendix 8.3, Appendix 11 Addition Section 1.2 (5) (6), 2.2 (1), 2.6, 5.4.2, 11.3 (6), 11.3.2 (error code $1F_H$), 11.3.3 (error codes $C0F7_H$ and $C300_H$), Appendix 4, Appendix 7, Appendix 9, Appendix 11
Oct., 2001	SH(NA)-080009-D	Correction Section 1.2 (4) (7), 1.3, 1.4 (1), 2.2 (POINT), 2.5 (1), 2.7 (1), 3.1, 3.5, 3.8, 4.1.1 (2), 4.3, 4.4.1 (POINT), 4.7 (5), 5.2.2 (REMARKS), 5.2.3, 5.4.2, 5.5 (Table), 5.6.1, 5.6.2, 5.6.3, 10.1, Section 11.2.2 (2), 11.2.4, 11.3.3, Appendix 1.1, Appendix 8.2, Appendix 9 Addition Section 10.9

Print Date	*Manual Number	Revision
Mar., 2002	SH(NA)-080009-E	Correction About the Generic Terms and Abbreviations, Section 1.2 (7), 2.1(1), 2.7 (1), 3.7 (Table), 5.1 (Figure), 5.2.2 (REMARKS), 5.2.3, 7.5.1, 10.9, 11.3.1 (Table), 11.3.3(REMARKS), 11.4.2 (Figure), Appendix 1.1 (2) (Table), Appendix 2.2 (2)(a), Appendix 8 Addition Section 11.3.3 (error code $C1BA_H$), 11.4 2, Appendix 4 (4)
Apr., 2003	SH(NA)-080009-F	Model Addition QJ71E71-B5 Model Deletion QJ71E71 Correction SAFETY PRECAUTIONS, About the Generic Terms and Abbreviations, Section 1.2 (1) (5) (6), 1.3, 1.4 (1), 2.1, 2.2, 2.5, 2.6, 2.7, 3.1, 3.4, 3.8, 4.1, 4.3, 4.4, 4.5, 4.6, 5.2.2 (REMARKS)(5), 5.2.3 (REMARKS), 5.3, 5.4.2 (1)(c), 5.4.3, 5.5 (6), 5.6 (2) (POINT), 5.9.1, 5.9.6 (3) (6), 6.2, 10.8, 10.9, 11.1.1, 11.2.2(2), 11.3, 11.4.5 (Figure), Appendix 1.1, Appendix 2, Appendix 4, Appendix 7, Appendix 10, Appendix 11 Addition Section 11.2.1 (POINT), 11.3.3 (error code $C0B2_H$, $C0E0_H$ to $C0EF_H$, $C171_H$ to $C17F_H$), Appendix 9
Jul., 2003	SH(NA)-080009-G	Correction The Manual's Usage and Structure, Section 1.4 (1), 2.2 (1), 2.7, 3.8, 4.7, 5.2.2 (REMARKS), 5.5 (POINT), 5.6.1, 5.6.2, 7.5.2, 8.6.2, 11.4, Appendix 2, Appendix 8.1, Appendix 11, Appendix 12
Sep., 2003	SH(NA)-080009-H	Correction Section 1.4 (1), 3.8, 5.2.3, Appendix 2.1, Appendix 8
Jun., 2004	SH(NA)-080009-I	• Addition of the description of function version D Correction About the Generic Terms and Abbreviations, Section 1.1 (7), 1.3, Chapter 2 (entire), Section 3.6, 3.8, 4.5, 4.6, 5.3, 5.5, 5.6 (2) (POINT), 5.8 (POINT), 6.1.4, 10.9, Appendix 1, Appendix 4, Appendix 11, Appendix 12 Addition Section 1.1 (9), 5.10, 5.11, 11.3.3, 11.4.7, Appendix 8.2
Aug., 2005	SH(NA)-080009-J	Correction SAFETY PRECAUTIONS, Section 1.1 (5), 1.3, 2.2, 2.7, 5.2.3, 5.11.3, 10.9, 11.3.3 (error code $C062_H$, $C0B9_H$, $C0C0_H$, $C0C4_H$, $C0D7_H$), Appendix 1.1, Appendix 4 (4), Appendix 11 Addition Appendix 8.1, Appendix 8.2, Appendix 9
Jun., 2006	SH(NA)-080009-K	Correction Section 2.1, 3.8, 5.6.3, 5.7.2, 5.9.3, 7.1, 7.3.1, 7.5.2, 8.1, 8.3.1, 8.6.2, 10.2, 10.6, 10.7, 10.8 Addition Section 2.7, 3.4, 5.6, 7.3.1, 10.1

Print Date	*Manual Number	Revision
Jun., 2007	SH(NA)-080009-L	Correction About the Generic Terms and Abbreviations, Section 1.2 (9), 1.3, 2.1, 2.7, 3.6, 4.1.1, 4.5.2 (12), 4.6, 4.9.1, 5.6 (1), (2) Point, 5.11, 5.11.3, 6.1.4, 10.2 to 10.9, 11.3.3, 11.4.7, Appendix 1.1, Appendix 4 (4), Appendix 9.1 (4) Addition Section 2.5.1, 2.5.2
Oct., 2008	SH(NA)-080009-M	Correction SAFETY PRECAUTIONS, Compliance with the EMC and Low Voltage Directives, The Manual's Usage and Structure, About the Generic Terms and Abbreviations, Section 1.2 to 1.4, Chapter 2, Section 3.1 to 3.6, 3.8, 4.1.1, 4.3 4.4.2, 4.4.3, 4.5.2, 4.6 to 4.8, 5.1 to 5.5, 5.6.1, 5.6.2, 5.8, 5.9.1, 5.9.3, 5.9.5, 5.9.7, 5.11.3, 5.11.5, 6.1.1, 6.1.3, 6.1.4,6.2, 7.3.2, 7.5.2, 8.6.2, Chapter 10, Section 11.2.1, 11.3, 11.4, Appendix 1.1, Appendix 2.1, Appendix 2.2, Appendix 4, Appendix 7, Appendix 9.1, Appendix 12
Apr., 2009	SH(NA)-080009-N	Correction Section 1.3, 2.2, 2.7, 4.3, 5.2.3, 5.11.5, 8.5.1, 10.9, Appendix 1.1
Aug., 2010	SH(NA)-080009-O	Correction Addition of "CC-Link IE Field Network" to the whole manual SAFETY PRECAUTIONS, About the Generic Terms and Abbreviations, Section 1.2, 2.1, 2.5.2, 4.1.1, 4.9.1, 5.9.3, 5.11, 5.12.3, 11.3.3, 11.4
May, 2012	SH(NA)-080009-P	The entire manual has been revised due to the manual layout change.
Jun., 2013	SH(NA)-080009-Q	Correction SAFETY PRECAUTIONS, RELEVANT MANUALS, TERM, Chapter 1, Section 3.1, 3.2, 3.3.1, 3.3.2, 3.4, 3.5.2, 5.1, 7.1.1, 7.1.4, 12.4, 12.8, 12.9.3, 14.3.4, 14.6.2, 14.6.3, 14.6.4, 14.7, 15.1, 15.2.1, 15.5, 15.6, 15.8, 15.9, 15.10, 15.11, 15.12, 15.13, 16.4, 16.6, Appendix 1, Appendix 3, Appendix 4, Appendix 5, Appendix 7, Appendix 10, Appendix 11.1 Addition Chapter 10, 11, Section 15.7, 16.3, 16.5.5, 16.5.6, 16.5.7, Appendix 8
Apr., 2015	SH(NA)-080009-R	Correction RELEVANT MANUALS, Section 3.2, 3.5.2, 5.1.5, 5.2.1, 6.2.2, 7.1.4, Chapter 9, Section 10.5, 11.3, 12.4.2, 12.6.2, 12.9.3, 13.4.2, 14.1.2, 15.5, 15.7, 16.5.2, 16.8, Appendix 3, 5
Jul., 2016	SH(NA)-080009-S	Correction Chapter 1, Section 3.2, 3.3.1, 3.3.2, 3.5.2, 5.1.2, 5.1.4, 5.1.5, 5.2.1, 11.3, 15.5, 16.5.1, 16.6.3, Appendix 3 Addition Section 14.3, 16.5.11
Sep., 2018	SH(NA)-080009-T	Correction Section 3.3.2, 3.5.2, 14.3.1, Appendix 4.1
Feb., 2019	SH(NA)-080009-U	Correction Section 3.4

Japanese Manual Version SH-080004-AB

WARRANTY

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

(1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.

(2) Even within the gratis warranty term, repairs shall be charged for in the following cases.

1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
2. Failure caused by unapproved modifications, etc., to the product by the user.
3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

2. Onerous repair term after discontinuation of production

(1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.

(2) Product supply (including repair parts) is not available after production is discontinued.

3. Overseas service

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

4. Exclusion of loss in opportunity and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

(1) Damages caused by any cause found not to be the responsibility of Mitsubishi.

(2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.

(3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.

(4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

5. Changes in product specifications

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

SH-080009-U

SH(NA)-080009-U(1902)MEE
MODEL: QJ71E71-U-KI-E
MODEL CODE: 13JL88

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN