SHARP®

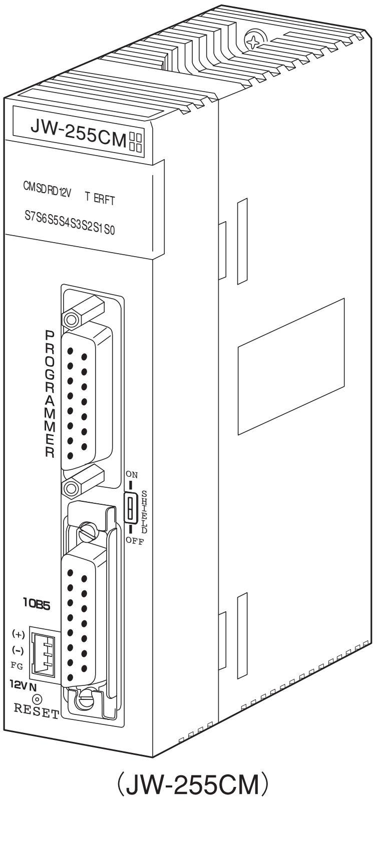
Version 2.0
Produced in May. 2004

Programmable Controller

Model name

Ethernet module JW-255CM
JW-25TCM
Ethernet board Z-339J

User's Manual



（JW-255CM）

Thank you for purchasing the Ethernet module (board) for the Sharp programmable controller.

Ethernet module	JW-255CM	(Mounting PLC) JW20H/30H
	JW-25TCM	
Ethernet board	Z-339J	J-board Z300/500

Read this manual thoroughly to completely familiarize yourself with the operation.
Make sure to read the following manuals for JW20H/30H and J-board together with this manual.

JW20H control module
- User's manual - Hardware version
- Programming manual - Ladder instruction version
- Programming manual - Step flow version

JW30H control module
- User's manual - Hardware version
- Programming manual - Ladder instruction version

J-board Z300 series
CPU board — Z-311J/312J User's manual - Hardware version
(JW20H Programming manual Ladder instruction version)

J-board Z500 series
CPU board
- Z-511J User's manual - Hardware version
- Z-511J User's manual - Hardware version

(JW30H Programming manual - Ladder instruction version)

Notes

- Though this manual is produced with the almost care, if you have any questions and inquiries, please feel free to contact our dealers.
- No part of this manual may be reproduced in any form without permission of SHARP corporation.
- The contents of this manual are subject to change without prior notice.

* Ethernet is a trademark of the XEROX Corporation.

Safety Precautions

Read this manual and attached documents carefully before installation, operation, maintenance and checking in order to use the machine correctly. Understand all of the machine knowledge, safety information, and cautions before starting to use. In this instruction manual, safety precautions are ranked into "danger" and "caution" as follows.

◈ Danger	: Wrong handling may possibly lead to death or heavy injury.
⚠ Caution	: Wrong handling may possibly lead to medium or light injury.

Even in the case of ⚠ Caution , a serious result may be experienced depending on the circumstances. Anyway, important points are mentioned. Be sure to observe them strictly.

The picture signs of prohibit and compel are explained below.

⊘	: It means don’ts. For example, prohibition of disassembly is indicated as (⊘).
❗	: It means a must. For example, obligation of grounding is indicated as (⏚).

1) Installation

⚠ Caution

- Use in the environments specified in the user's manual.
  Electric shock, fire or malfunction may be caused when used in the environments of high temperature, high humidity, dusty or corrosive atmosphere, vibration or impact.
- Install according to the user's manual.
  Wrong installation may cause drop, breakdown, or malfunction.
- Never admit wire chips or foreign matters.
  Or fire, breakdown or malfunction may be caused.

2) Wiring

⏚ Compel

- Be sure to ground for programmable controller.
  Unless grounded, electric shock or malfunction may be caused.

⚠ Caution

- Connect the rated power source.
  Connection of a wrong power source may cause a fire.
- Wiring should be done by qualified electrician.
  Wrong wiring may lead to fire, breakdown or electric shock.

3) **Use**

Danger

- Don’t touch the terminal while the power is being supplied or you may have an electric shock.
- Assemble the emergency stop circuit and interlock circuit outside of the programmable controller. Otherwise breakdown or accident damage of the machine may be caused by the trouble of the programmable controller.

Caution

- Change of program during operation, or "Run" or "Stop" during operation should be done with particular care by confirming safety. Misoperation may lead to damage or accident of the machine.
- Turn on the power source in the specified sequence. Turning ON with wrong sequence may lead to machine breakdown or accident.

4) **Maintenance**

Prohibit

- Don’t disassemble or modify the modules.
  Or fire, breakdown or malfunction may be caused.

Caution

- Turn OFF the power source before detaching or attaching the module.
  Or electric shock, malfunction or breakdown may be caused.

Chapter 1. Overview

Chapter 2. Precautions for Use

Chapter 3. System Configuration

Chapter 4. Name and Function of Each Part

Chapter 5. Installation Method

Chapter 6. Connection/Wiring

Chapter 7. Overview of Function

Chapter 8. Computer Link Function

Chapter 9. SEND/RECEIVE Functions

Chapter 10. Routing Function

Chapter 11. Errors and Correction

Chapter 12. Network Parameter

Chapter 13. Sample Program

Chapter 14. Specifications

Alphabetical Index

Table of Index

Safety Precautions

Chapter 1. Overview 1-1

Chapter 2. Precautions for Use 2-1

Chapter 3. System Configuration 3-1

Chapter 4. Name and Function of Each Part .... 4-1 to 4-4

[1] JW255CM 4-1
[2] JW25TCM 4-2
[3] Z-339J 4-3

Chapter 5. Installation Method .... 5-1 to 5-6

5-1 Installation of JW-255CM/25TCM 5-1
5-2 Installation of Z-399J 5-2
[1] Maximum number of boards to install 5-3
[2] Address allocation of I/O relays 5-4

Chapter 6. Connection/Wiring .... 6-1 to 6-7

6-1 Installing an Ethernet cable 6-1
[1] Equipment layout 6-1
[2] Wiring 6-1
6-2 Connection method 6-2
[1] Installation of JW-255CM 6-2
[2] Installation of JW-25TCM 6-4
[3] Installation of Z-339J 6-5
[4] Installation of 10BASE-T 6-7

Chapter 7. Overview of Function .... 7-1 to 7-7

7-1 Computer link function 7-1
7-2 SEND/RECEIVE function 7-2
7-3 Network parameter settings 7-3

Chapter 8. Computer Link Function .... 8-1 to 8-58

8-1 Basic format of computer link commands 8-1
[1] Communication format 8-1
[2] Memory address expression format 8-2
[3] Execution condition 8-3
[4] Table of commands 8-4
8-2 Descriptions of each command 8-5
8-3 Standard buffers 8-24
[1] How to specify a standard buffer 8-24
[2] Parameter setting 8-26
[3] Standard buffer information storage area 8-27
[4] Error processing when accessing standard buffers 8-27
[5] Description of commands used with standard buffers 8-28
8-4 Ring buffer 8-32
[1] How to use the ring buffer 8-32
[2] Operation of the ring buffer 8-35
[3] Parameter setting 8-39
[4] Ring buffer information storage area (in data memory) 8-40

[5] Error processing when accessing ring buffers 8-40
[6] Description of commands used with ring buffers 8-41
[7] An example using the ring buffer 8-49
8-5 Computer link error code table 8-54
8-6 Command execution completion information 8-55
[1] Setting the parameters 8-55
[2] Command execution completion information 8-55
8-7 Time interval required for communication 8-56
8-8 Two-layer communication with satellite net 8-57

Chapter 9. SEND/RECEIVE Functions .... 9-1 to 9-11
9-1 Instruction system 9-1
[1] Source/destination address and channel 9-1
[2] SEND/RECEIVE instructions operation 9-3
[3] Error recovery 9-7
[4] Other notes 9-7
9-2 Data memory starting system 9-8
[1] System 9-8
[2] Parameter setting 9-8
[3] Communication information storage area 9-10
[4] Other notes 9-10
[5] Program example for data memory starting system 9-11

Chapter 10. Routing Function .... 10-1 to 10-3
[1] Create a default router 10-1
[2] Create a customized routing table 10-2

Chapter 11. Errors and Correction .... 11-1 to 11-4
11-1 Connection status monitor 11-1
11-2 Settings for the retransmission timeout time 11-2
11-3 Settings for Keepalive 11-2
11-4 Settings for restart timer 11-3
11-5 Troubleshooting 11-4

Chapter 12. Network Parameter .... 12-1 to 12-11
12-1 Table of parameter 12-1
12-2 Setting procedure of parameters 12-7
[1] Setting procedures using the JW-14PG 12-8
[2] Setting method using the JW-100SP 12-10
[3] Setting procedures using the JW-52SP/92SP 12-11

Chapter 13. Sample Program .... 13-1 to 13-10

Chapter 14. Specifications .... 14-1 to 14-2
14-1 JW-255CM/25TCM 14-1
[1] General specifications 14-1
[2] Communication specifications 14-2
[3] Outside dimensions 14-3
14-2 Z-339J 14-4
[1] General specifications 14-4
[2] Communication specifications 14-5
[3] Outside dimensions 14-5

Alphabetical Index .... I-1 to I-3

Chapter 1. Overview

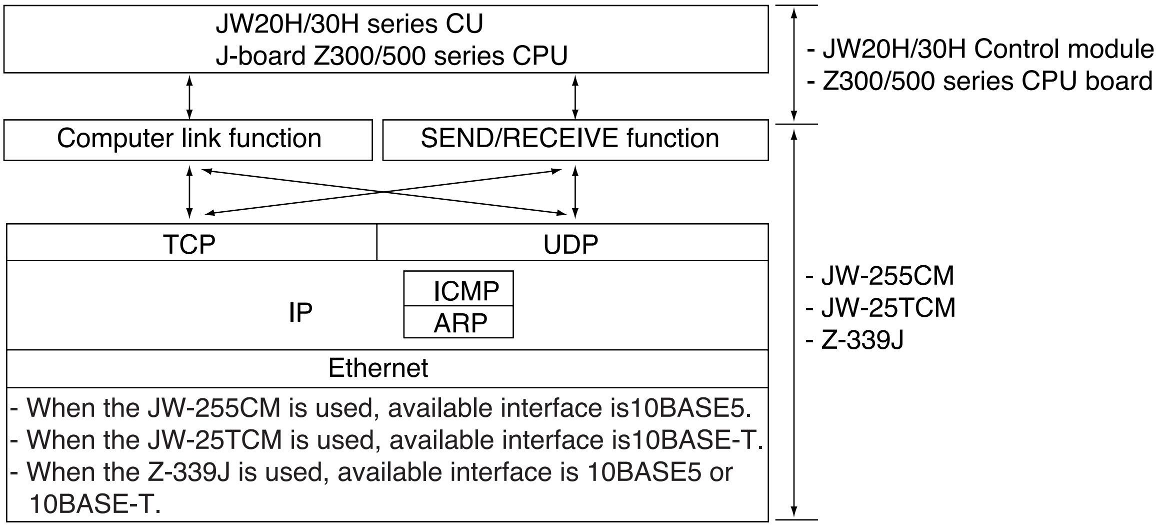
Ethernet modules and board (JW-255CM/25TCM and Z-339J) are interface modules (board) to exchange data with host computers on an Ethernet network and LAN. The Ethernet modules (JW-255CM/25TCM) shall be connected to a JW20H/30H programmable controller (PLC) for data exchange on a network. The Ethernet board (Z-339J) shall be mounted on the J-board Z300/500 to communicate with the Ethernet.
The JW-255CM/25TCM are option modules for the JW20H/30H. The Z-339J is a communication board for the J-board Z300/500.

* Ethernet is a trademark of the XEROX Corporation.

(1) Features

1) Both TCP/IP and UDP/IP protocols are available.
2) This module uses the same command format as used by Sharp's PLC computer link functions. It allows the host computer to access PLCs.
3) Data communication is possible between host computers in an Ethernet network and PLCs in a satellite network spanning two hierarchic layers.
4) As interfaces, the JW-255CM supports 10BASE5, the JW-25TCM supports 10BASE-T, and Z-339J supports 10BASE5/10BASE-T.
5) This module is equipped with eight individual ports. Each port can make a separate connection.
6) Communication between PLCs is possible by using the SEND/RECEIVE functions.
7) Using the subnet mask routing function, this module can communicate with a large network system using a router.

(2) Software system



- When the JW-255CM is used, available interface is10BASE5.
- When the JW-25TCM is used, available interface is10BASE-T.
- When the Z-339J is used, available interface is 10BASE5 or 10BASE-T.

- TCP (Transmission Control Protocol)
TCP is a method used for communication after establishing a connection with a target node. It offers a highly reliable communication environment, such as with control orders and automatic retransmission if an error occurs.

- UDP (User Datagram Protocol)
UDP is a method to communicate without first establishing a connection with a target node. It transmits data by assigning a target name to each transmission. If the data is not received by the target node, this module will not retransmit the data, as is the case in the TCP mode.

- IP (Internet Protocol)
In this method, this module communicates with the target node in units called datagrams.

- ICMP (Internet Control Message Protocol)
ICMP is a protocol used to assist IP operations.

- ARP (Address Resolution Protocol)
This protocol obtains MAC addresses (Ethernet physical address) derived from the connected nodes IP addresses.

- Ethernet
This module can handle the frame format of Ethernet version 2.

Chapter 2. Precautions for Use

When using JW-255CM/25TCM (simply called "this module" from here on), Z339J (simply called "this board" from here on), observe the following precautions.

(1) Installation

- Do not install or store the JW-255CM/25TCM and Z-339J in the following conditions.
  1) Locations proximate to a heat generating object.
  2) Ambient temperature exceeding the range of 0 to 55 °C (Storage temperature : -20 to 70 °C)
  3) The relative humidity exceeding the range of 35 to 90%.
  4) Sudden temperature changes which may cause condensation.
  5) Corrosive or inflammable gas.
  6) Vibration or hard jolts
- The minimum distance between transceivers is specified in the regulations. (2.5 m when the 10BASE5 is used.) When connecting devices, be sure to maintain these minimum distances. Cables used for 10BASE5 systems have marks every 2.5 m. Position each transceiver directly on one of these marks.
- Prior to installing or detaching the JW20H/30H, make sure to turn OFF the power supply to the PLCs.
- Before connecting this board, make sure to shut off the power to the J-board.
- All screws must be tightened firmly.
- Mount the transceivers on electrically insulated objects, such as a wooden mounting block.

(2) Wiring

- Separate the data transmission cables, high tension cables from power cables.
- Do not run cables near any noise generating source.
- For 10BASE-T twisted pair cables, use cables conform to category 5 with shield.

(3) Treatment

● **JW-255CM/25TCM**

- For ventilation, holes are provided in the cabinet to prevent a temperature rise. Do not block the ventilation holes. Good ventilation is necessary.
- Never allow a liquid such as water and chemical solution and a metallic object like a copper wire inside this module to avoid a possible hazard. Otherwise, it may be a cause of machine trouble.
- In an extremely dry area, large amounts of static electricity may be generated in a person. Before touching the programmer, discharge any static electricity by first touching a grounded metallic object.

● **Z-339J**

The J-board is a board structure and electronic parts are exposed. Therefore, be careful for the followings for handling.

1) Before touching the board, discharge static electricity in your body.
2) Do not touch the board with dirty hands such as stained by oil.
3) When putting the board alone, do not put it on a conductive object such as metal plate. (After the board is assembled to the CPU board and if it is put on a conductive object, the battery terminals of the CPU board may be shorted circuit and the backup memory will be broken.)
4) Do not load excessive force on the switches, connectors, and terminal blocks.

(4) Allocation of relay numbers

The PLC (JW20H/30H) on which this module is installed will reserve 16 points as input/output relay numbers for this module. However, since these 16 points are not used on this module, this becomes a dummy area. (As for the J-339J board, refere page 5-4 to 5-6.)

(5) Grounding

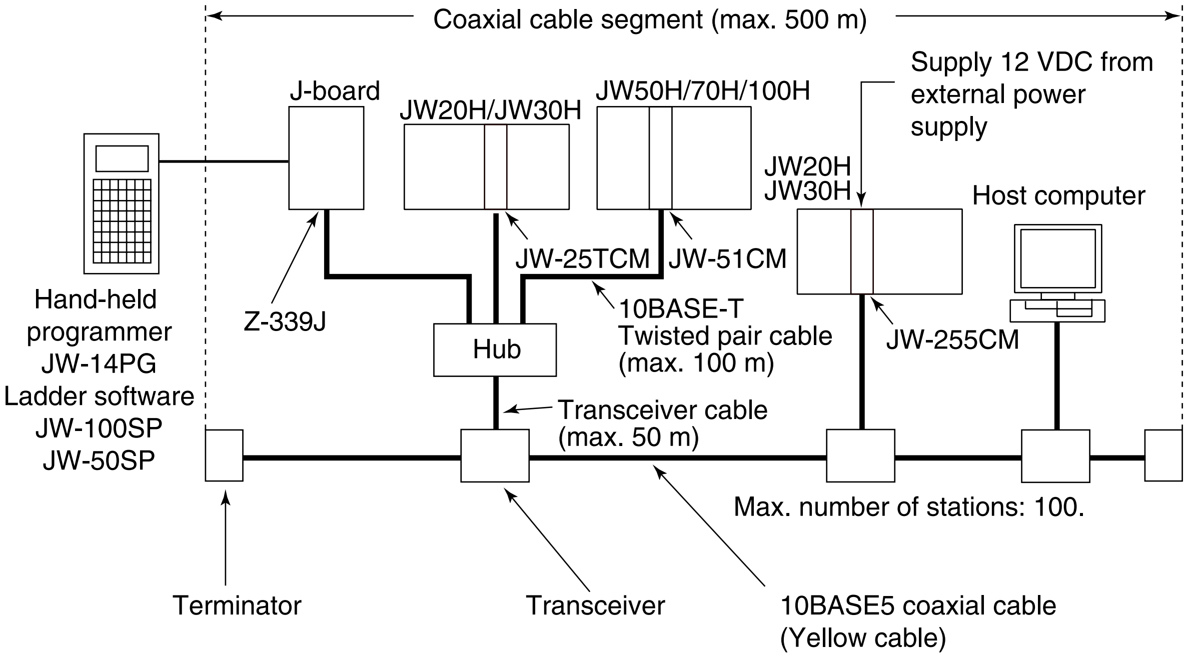
- Provide an independent, class D ground for the FG terminal on the J-board (terminal block on the CPU board). Do not connect it in common to a general purpose ground.
- The hexagonal stud on the J-board (there is one on each board) is for connecting the ground (FG). Tighten it securely.

(6) Cleaning

Use a clean, dry cloth when cleaning this module. Do not use volatile chemicals such as thinner or alcohol as it may result in deformation and color fading.

Chapter 3. System Configuration

● **Example of connection**



Note: 10BASE5 coaxial cable, 10BASE-T twisted pair cable, transceiver, transceiver cable, and terminator, etc. should be prepared by user.

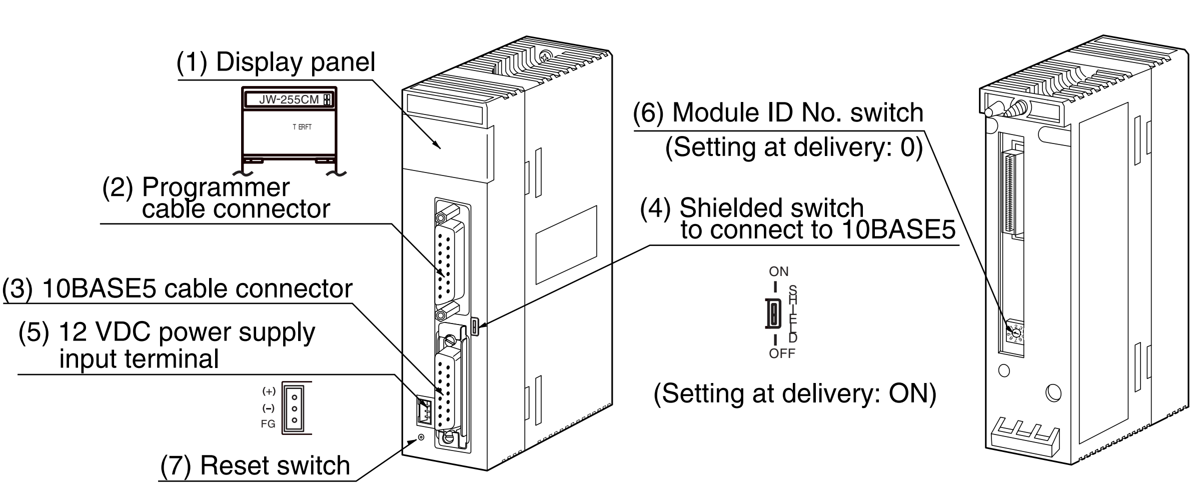
● **Sharp's Ethernet applicable models**

Model name	PLC on which to mount	Connection to network	Reference manual
JW-255CM	JW20H, JW30H	10BASE5	This manual
JW-25TCM		10BASE-T	
Z-339J	J-board (Z300/500)	Either of 10BASE5 or 10BASE-T	
JW-50CM	JW50H, JW70H, JW100H	Either of 10BASE5 or 10BASE-2	JW-50CM user's manual
JW-51CM		Either of 10BASE5 or 10BASE-T	JW-51CM user's manual

Chapter 4. Name and Function of Each Part

[1] JW-255CM

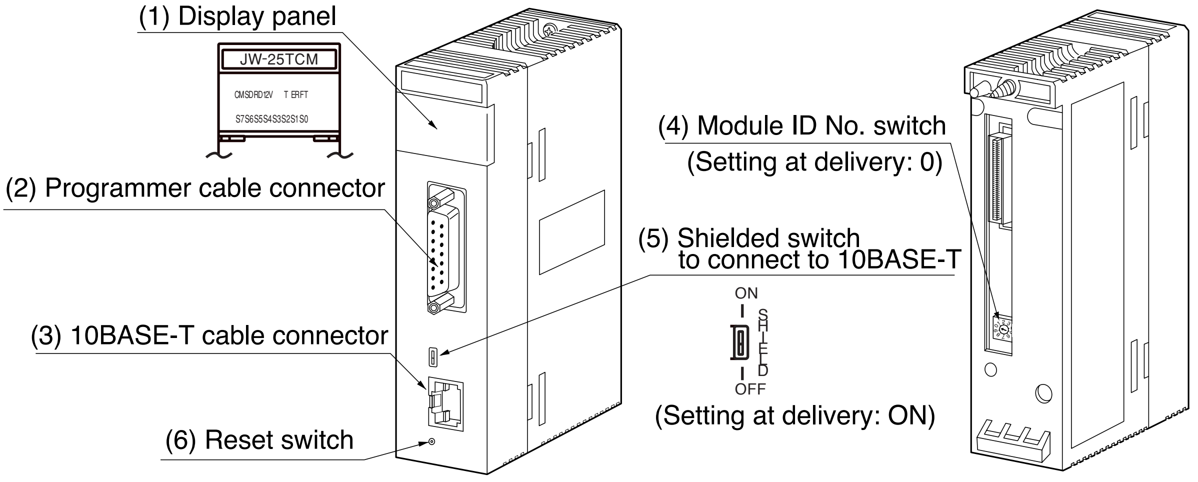
● Front side ● Rear side


(1) Display panel
JW-255CM
T ER FT
(2) Programmer cable connector
(3) 10BASE5 cable connector
(5) 12 VDC power supply input terminal
(+)
(−)
FG
(7) Reset switch
(6) Module ID No. switch
(Setting at delivery: 0)
(4) Shielded switch to connect to 10BASE5
ON
SHIELD
OFF
(Setting at delivery: ON)

	Name		Function
(1)	Display panel		Indicates this module’s operation status by turning the LED ON and OFF.
		CM	Lights when the JW-255CM starts normally.
		SD	Lights when the JW-255CM is transmitting data.
		RD	Lights when the JW-255CM is receiving data.
		12V	Lights when the JW-255CM is receiving 12 VDC power.
		T	Lights when the JW-255CM is in the test mode. (This is never lit in normal operation.)
		ER	Lights when a parameter setting error occurs.
		FT	Lights when an error occurs in this module.
		S0 to S7	Lights when a connection is established.
(2)	Programmer cable connector		Plug in the cable assembly connector in order to connect the JW-14PG programmer to this module. The JW-14PG is used to set this module’s parameters.
(3)	10BASE5 cable connector		Connect the 10BASE5 transceiver cable here. After connecting the cable, make sure to slide the lock securely to the "lock" position.
(4)	Shielded switch to connect to 10BASE5	ON	Turn ON this switch when the transceiver cable shield is connected to FG (rack) on this module.
		OFF	Turn OFF this switch when the transceiver cable shield is not connected to the rack. - Ground the FG wire from the 12 VDC connector separately.
(5)	12 VDC power supply input terminal		The DC input terminal used to supply the power to the transceiver. Use a connecting cable (accessory) and supply power from a commercial power supply. 12 VDC ± 5 %. 0.5 A or more is required.
(6)	Module ID No. switch		Set the module ID to a number within the range of 0 to 6. - Do not use a number that is already assigned to another option module.
(7)	Reset switch		Resets the hardware in this module. This switch is for our service staff only. Do not press this switch.

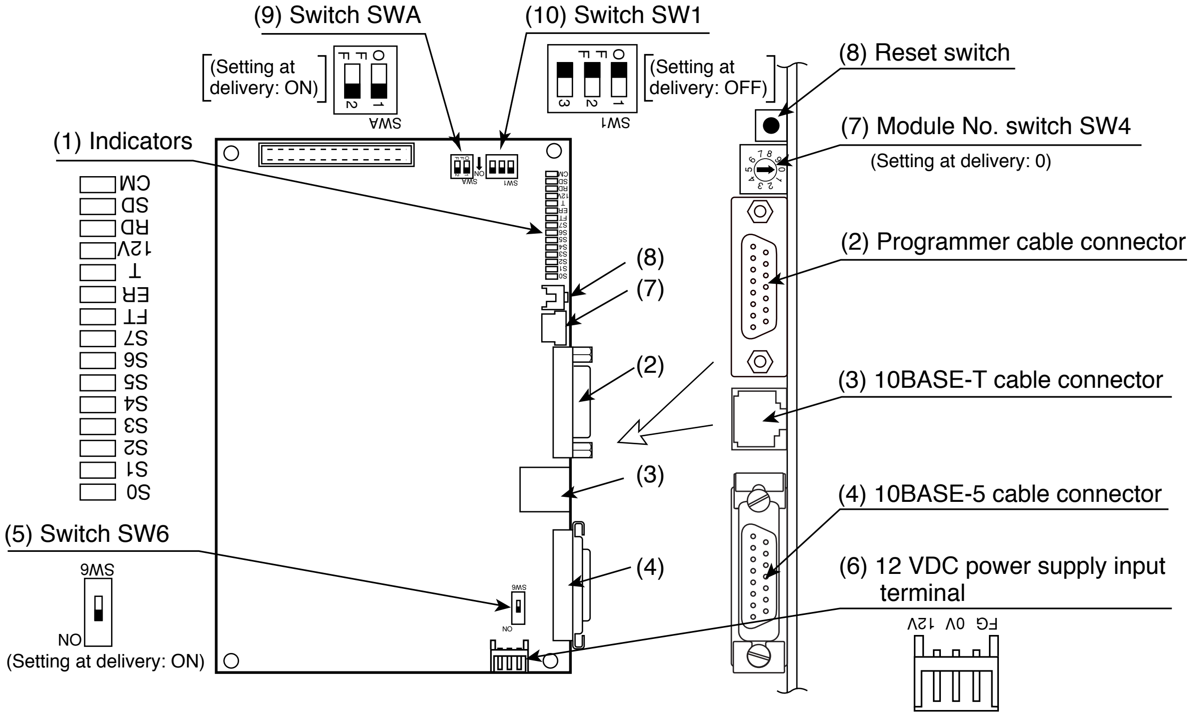
[2] JW-25TCM

● Front side ● Rear side



	Name		Function
(1)	Display panel		Indicates this module’s operation status by turning the LED ON and OFF.
		CM	Lights when the JW-25TCM starts normally.
		SD	Lights when the JW-25TCM is transmitting data.
		RD	Lights when the JW-25TCM is receiving data.
		12V	This LED is not used with this module.
		T	Lights when the JW-25TCM is in the test mode. (This is never lit in normal operation.)
		ER	Lights when a parameter setting error occurs.
		FT	Lights when an error occurs in this module.
		S0 to S7	Lights when a connection is established.
(2)	Programmer cable connector		Plug in the cable assembly connector in order to connect the JW-14PG programmer to this module. The JW-14PG is used to set this module’s parameters.
(3)	10BASE-T cable connector		Connect a twisted pair cable for 10BAS-T.
(4)	Module ID No. switch		Set the module ID to a number within the range of 0 to 6. - Do not use a number that is already assigned to another option module.
(5)	Shielded switch to connect to 10BASE5	ON	Turn ON this switch when the transceiver cable shield is connected to FG (rack) on this module.
		OFF	Turn OFF this switch when the transceiver cable shield is not connected to the rack. - Ground the FG wire from the 12 VDC connector separately.
(6)	Reset switch		Resets the hardware in this module. This switch is for our service staff only. Do not press this switch.

[3] Z-339J


(9) Switch SWA
(10) Switch SW1
(8) Reset switch
(Setting at delivery: ON)
OFF
1
2
SWA
(Setting at delivery: OFF)
1
2
3
SW1
(7) Module No. switch SW4
(Setting at delivery: 0)
(1) Indicators
CM
SD
RD
12V
T
ER
FT
S7
S6
S5
S4
S3
S2
S1
S0
(2) Programmer cable connector
(8)
(7)
(2)
(3) 10BASE-T cable connector
(3)
(4) 10BASE-5 cable connector
(5) Switch SW6
SW6
ON
(Setting at delivery: ON)
(4)
(6) 12 VDC power supply input terminal
FG 0V 12V

	Name		Function
(1)	Indicator		Indicates this board's operation status by turning the LED ON and OFF.
		CM	Lights when the Z-339J starts normally.
		SD	Lights when the Z-339J is transmitting data.
		RD	Lights when the Z-339J is receiving data.
		12V	Lights when the Z-339J is receiving 12 VDC power.
		T	Lights when the Z-339J is in the test mode. (This is never lit in normal operation.)
		ER	Lights when a parameter setting error occurs.
		FT	Lights when an error occurs in this board.
		S0 to S7	Lights when a connection is established.
(2)	Programmer cable connector		Plug in the cable assembly connector in order to connect the JW-14PG programmer to this board. The JW-14PG is used to set this board's parameters.
(3)	10BASE-T cable connector		Connect the 10BASE-T Twist cable here.
(4)	10BASE5 cable connector		Connect the 10BASE5 transceiver cable here. After connecting the cable, make sure to slide the lock securely to the "lock" position.
(5)	Switch SW6	ON	Turn ON this switch when the transceiver cable shield is connected to FG (rack) on this module.
		OFF	Turn OFF this switch when the transceiver cable shield is not connected to the rack. - Ground the FG wire from the 12 VDC connector separately.
(6)	12 VDC power supply input terminal		The DC input terminal used to supply the power to the transceiver. Use a connecting cable (accessory) and supply power from a commercial power supply. 12 VDC ± 5 %. 0.5 A or more is required.
(7)	Module ID No. switch SW4		Set the module ID to a number within the range of 0 to 6. - Do not use a number that is already assigned to another option module.
(8)	Reset switch		Resets the hardware in this module. This switch is for our service staff only. Do not press this switch.
(9)	Switch SWA		Specify the number of communication boards installed, including this board.
(10)	Switch SW1		No need setting. Use all the switches to default status (OFF).

Note: Either of 10BASE5 or 10BASE-T can be used for communication. (Shared use of the both is not possible.)
The board automatically switches between 10BASE5 and 10BASE-T by whether or not 12 VDC (item (6) above) is supplied.(When 12 VDC is supplied, the board automatically switches to 10BASE5.)

12 VDC Supply	10BASE5	10BASE-T
Provided	○	×
None	×	○

Chapter 5. Installation Method

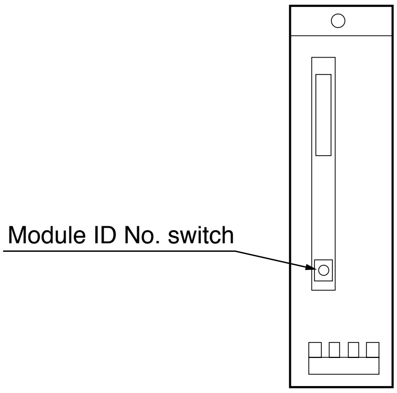
5-1. Installation of JW-255CM/25TCM

This paragraph describes how to attach this module to the JW20H/30H basic rack panel.

1) Turn OFF the power supply to the JW20H/30H.

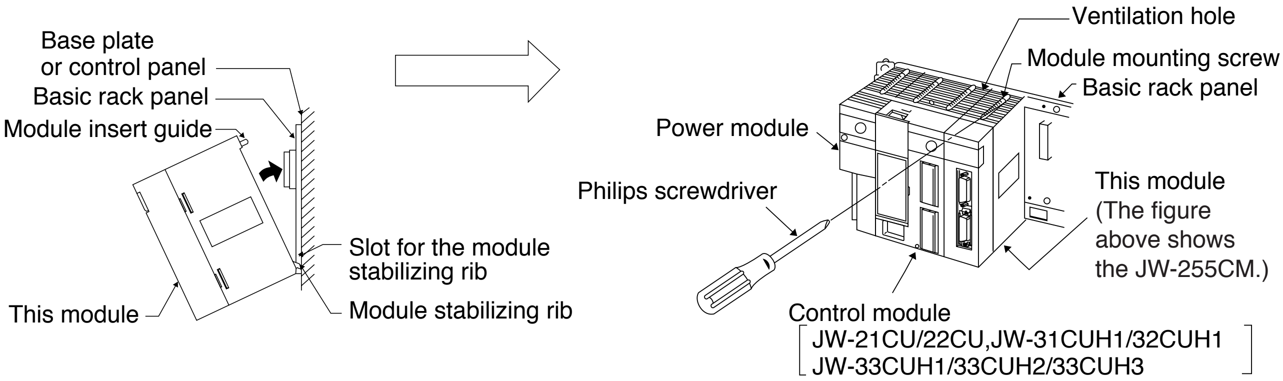
2) Set the module ID No. switch on the rear side of this module.

[Rear side of this module]



3) Put this module stabilizing rib into the slot on the JW20H/30H basic rack panel and push the module into the panel. Secure the module mounting screw on the top of this module using a Philips screwdriver.

[Installation example]



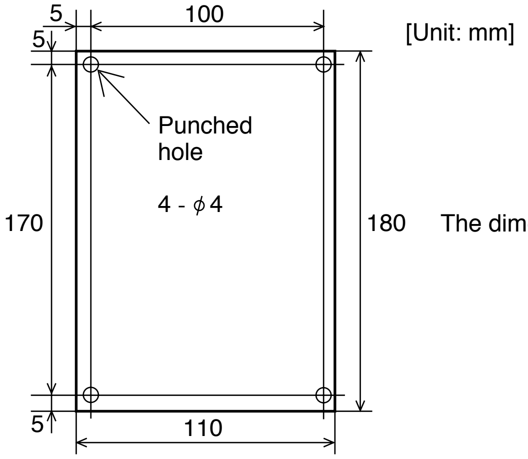
Remarks

- This module cannot be installed in the expansion rack panel.
- More than one optional module can be installed in the same control module (JW20H/30H basic rack panel). Do not use a number that is already assigned to another option module (or to another JW-255CM/25TCM).
- Secure the module mounting screw tightly. If the screw becomes loose, it may cause a malfunction.

5-2. Installation of Z-339J

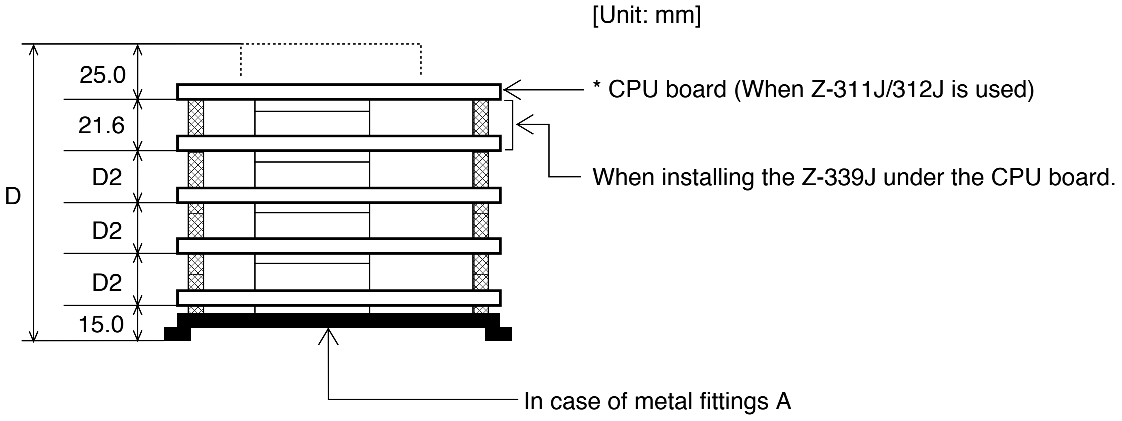
Below shows substrate dimensions and assembled dimensions of the Z-339J.

■ Board dimensions



The dimensions left do not include metal fittings.

■ Assembled dimensions


[Unit: mm]
25.0
21.6
D2
D
D2
D2
15.0
* CPU board (When Z-311J/312J is used)
When installing the Z-339J under the CPU board.
In case of metal fittings A

* The CPU board can be installed only at the top of them.

- For details about assembling and installation dimensions, see the manuals below.
  J-board Z-311J/312J User's manual - Hardware version
  J-board Z-511J User's manual - Hardware version
  J-board Z-512J User's manual - Hardware version
  Dimensions "D" and "D2" correspond to "D" and "D2" on the "board size" section in the manuals above.
- Make sure to secure conductivity of the metal fittings and installation section.

Below describes the maximum number of J-boards to install on the Z-339J (shown below) and address allocation of I/O relays.

Address allocation of I/O relays ─┬ When installed on the Z-311J/312J, Z-512J => Next page
└ When installed on the Z-511J => Refer to page 5-5

[1] Maximum number of boards to install

The Z-339J is a kind of communication boards for the J-board, and number of maximum boards to install on the J-board is limited by the total number of communication boards including other communication boards.

J-board	CPU board	Actual number of Z-339J boards to install (including other communication boards)
Z300 series	Z-311J	Maximum 2 boards - If the total of 5 V current consumption of boards installed exceeds 800 mA, the number of boards shall be restricted.
	Z-312J	
Z500 series	Z-511J	Maximum 2 boards - A 5 V power supply to install externally shall have power capacity to cover total of each boards 5 V consumption current (maximum 2.7 A).
	Z-512J	Maximum 2 boards - If the total of 5 V current consumption of boards installed exceeds 1.8 A, the number of boards shall be restricted.

● Types of communication boards

Model name	Specifications
Z-331J *	Data link or computer link, satellite I/O link master station.
Z-332J	Data link or computer link.
Z-333J	Satellite I/O link master station
Z-334J	ME-NET board (with branch line extension function)
Z-335J	Satellite net board
Z-336J	FL-net board (Ver. 1)
Z-336J2	FL-net board (Ver. 2)
Z-337J	Device net board
Z-338J	Device net master board (with 32 points I/O)
Z-339J	Ethernet board

* Production by each order

[2] Address allocation of I/O relays

This section describes I/O relay addresses that are allocated to the Z-339J.

(1) When the Z-339J is installed on the Z-311J/312J, and Z-512J

The maximum number of Z-339J boards to install is two (including other communication boards). Below shows switch setting on the Z-339J (SWA switch to set the number communication boards installed) and allocation of I/O relays

1) When one communication board (Z-339J) is used.

Set the SWA switch to the number of communication board on the Z-339J as follows:
SWA switch setting on Z-339J

SWA switch setting on Z-339J		1 2 ON ON	
Allocation of I/O relays on Z-339J		**I/O relay address**	**Installed address**
	Z-339J (option) *1	]0000	R = 0, S = 0
		]0001	
	Dummy (vacant)	]0002	R = 0, S = 1
		]0003	
	Dummy (vacant)	]0004	R = 0, S = 2
		]0005	
	Dummy (vacant)	]0006	R = 0, S = 3
		]0007	

*1: Allocated as option. However it will be dummy area that is not used.

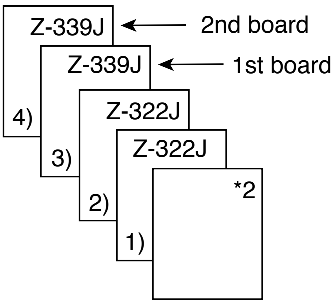
2) When using two communication with boards

Allocation of I/O relays varies whether it is installed on the 1st board or 2nd board.

		When using Z-339J as 1st board		**When using Z-339J on 2nd board**	
SWA switch setting on Z-339J		1 2 ON ON		1 2 OFF ON	
Allocation of I/O relays on Z-339J		**I/O relay address**	**Installed address**	**I/O relay address**	**Installed address**
	Z-339J (option) *1	]0000	R = 0, S = 0	]0010	R = 0, S = 4
		]0001		]0011	
	Dummy (vacant)	]0002	R = 0, S = 1	]0012	R = 0, S = 5
		]0003		]0013	
	Dummy (vacant)	]0004	R = 0, S = 2	]0014	R = 0, S = 6
		]0005		]0015	
	Dummy (vacant)	]0006	R = 0, S = 3	]0016	R = 0, S = 7
		]0007		]0017	

■ Allocation example

Switch setting and I/O allocation example when using two boards of Z-339J.



*2: Z-311J, Z-312J, or Z-512J.

▮ ON
▯ OFF

Installed board	SW1 (Rack No.)	SWA (SW2)	I/O relay address	Installed address
1)	1 2 3 ▮ ▯ ▯	SW2 1 2 ▮ ▮	]0020,]0021	R = 1, S = 0
			]0022,]0023	R = 1, S = 1
			]0024,]0025	R = 1, S = 2
			]0026,]0027	R = 1, S = 3
2)	1 2 3 ▯ ▮ ▯	SW2 1 2 ▮ ▮	]0030,]0031	R = 2, S = 0
			]0032,]0033	R = 2, S = 1
			]0034,]0035	R = 2, S = 2
			]0036,]0037	R = 2, S = 3
3)	1 2 3 ▯ ▯ ▯	SWA 1 2 ▮ ▮	]0000,]0001	R = 0, S = 0
			]0002,]0003	R = 0, S = 1
			]0004,]0005	R = 0, S = 2
			]0006,]0007	R = 0, S = 3
4)	1 2 3 ▯ ▯ ▯	SWA 1 2 ▯ ▮	]0010,]0011	R = 0, S = 4
			]0012,]0013	R = 0, S = 5
			]0014,]0015	R = 0, S = 6
			]0016,]0017	R = 0, S = 7

(2) When installing the Z-339J on the Z-511J

The maximum number of boards to install the Z-339J is two (including other communication boards). Below shows switch setting on the Z-511J and Z-339J, as well as I/O relay allocation of the Z-339J.

1) When one communication board (Z-339J) is used

■ Switch setting

SW1 and SWA switches on the Z-511J and SWA switch for the number of communication boards on the Z-339J shall be set as follows:

- Z-511J

Switch SW1			Switch SWA	
1	2	3	1	2
OFF	OFF	OFF	ON	ON

- Z-339J

Switch SWA	
1	2
OFF	ON

■ I/O relay allocation

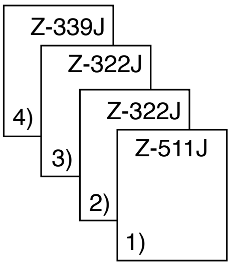
Allocate I/O relay address of the Z-339J as follows.

Allocation detail	I/O relay address	Installed address
Z-339J (option) *1	]0010	R = 0, S = 4
	]0011	
Dummy (vacant)	]0012	R = 0, S = 5
	]0013	
Dummy (vacant)	]0014	R = 0, S = 6
	]0015	
Dummy (vacant)	]0016	R = 0, S = 7
	]0017	

*1: Allocated as option. However it will be dummy area that is not used.

■ Allocation example

Below shows switch setting and I/O relay allocation example, when one communication board (Z-339J) is used.



■ ON
□ OFF

Installed board	SW1 (Rack No.)	SWA (SW2)	I/O relay address	Installed address
1)	1 2 3 □ □ □	SWA 1 2 ■ ■	]0000,]0001	R = 0, S = 0
			]0002,]0003	R = 0, S = 1
			]0004,]0005	R = 0, S = 2
			]0006,]0007	R = 0, S = 3
2)	1 2 3 ■ □ □	SW2 1 2 ■ ■	]0020,]0021	R = 1, S = 0
			]0022,]0023	R = 1, S = 1
			]0024,]0025	R = 1, S = 2
			]0026,]0027	R = 1, S = 3
3)	1 2 3 □ ■ □	SW2 1 2 ■ ■	]0030,]0031	R = 2, S = 0
			]0032,]0033	R = 2, S = 1
			]0034,]0035	R = 2, S = 2
			]0036,]0037	R = 2, S = 3
4)	1 2 3 □ □ □	SW2 1 2 □ ■	]0010,]0011	R = 0, S = 4
			]0012,]0013	R = 0, S = 5
			]0014,]0015	R = 0, S = 6
			]0016,]0017	R = 0, S = 7

2) When two communication boards are used

■ Switch setting

Set SW1 and SWA switches on the Z-511J and SWA switch for the number of communication boards on the Z-339J as follows:

- Z-511J

Switch SW1			Switch SWA	
1	2	3	1	2
ON	OFF	OFF	ON	ON

- Z-339J

Used on the 1st board		Used on the 2nd board	
Switch SWA		Switch SWA	
1	2	1	2
ON	ON	OFF	ON

5

■ I/O relay allocation

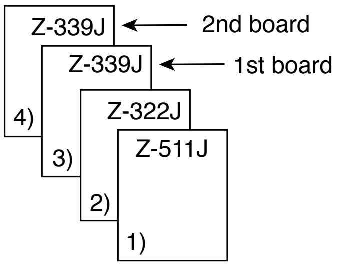
Allocate I/O relay address of the Z-339J as follows.

Allocation detail	When using Z-336J as 1st board		When using Z-339J on 2nd board	
	I/O relay address	Installed address	I/O relay address	Installed address
Z-339J (option) *	]0000	R = 0, S = 0	]0010	R = 0, S = 4
	]0001		]0011	
Dummy (vacant)	]0002	R = 0, S = 1	]0012	R = 0, S = 5
	]0003		]0013	
Dummy (vacant)	]0004	R = 0, S = 2	]0014	R = 0, S = 6
	]0005		]0015	
Dummy (vacant)	]0006	R = 0, S = 3	]0016	R = 0, S = 7
	]0007		]0017	

* Though allocated as an option, it will be dummy area that is not used.

■ Allocation example

Below shows switch setting and I/O relay allocation example, when two communication boards (Z-339J) are used.



▮ON
▯OFF

Installed board	SW1 (Rack No.)	SWA (SW2)	I/O relay address	Installed address
1)	1 2 3 ▮ ▯ ▯	SWA 1 2 ▮ ▮	]0020,]0021	R = 1, S = 0
			]0022,]0023	R = 1, S = 1
			]0024,]0025	R = 1, S = 2
			]0026,]0027	R = 1, S = 3
2)	1 2 3 ▯ ▮ ▯	SWA 1 2 ▮ ▮	]0030,]0031	R = 2, S = 0
			]0032,]0033	R = 2, S = 1
			]0034,]0035	R = 2, S = 2
			]0036,]0037	R = 2, S = 3
3)	1 2 3 ▯ ▯ ▯	SWA 1 2 ▮ ▮	]0000,]0001	R = 0, S = 0
			]0002,]0003	R = 0, S = 1
			]0004,]0005	R = 0, S = 2
			]0006,]0007	R = 0, S = 3
4)	1 2 3 ▯ ▯ ▯	SWA 1 2 ▯ ▮	]0010,]0011	R = 0, S = 4
			]0012,]0013	R = 0, S = 5
			]0014,]0015	R = 0, S = 6
			]0016,]0017	R = 0, S = 7

Chapter 6. Connection/Wiring

6-1 Installing an Ethernet cable

Workers who will install or hook up an Ethernet cable must have special training and knowledge, such as the safety procedures and standards required by this technology (JIS X5252).
We recommend that you contact a specialist for perform any installation or hook up.
(The Sharp Document Systems Co., Ltd. provides Ethernet wiring works and dealing network products made by Allied Systems Co., Ltd.)

[1] Equipment layout

- The minimum distance between nodes is specified in the regulations. (2.5 m when the 10BASE5 is used.)
  Cables used for 10BASE5 systems have marks every 2.5 m. Position each transceiver directly on one of these marks.
- Mount the transceivers on electrically insulated objects, such as a wooden mounting block.

[2] Wiring

- Separate the data transmission cables from power cables (60 cm or longer).
- Do not run cables near any noise generating source.
- Both ends of the coaxial cable must be terminated with a termination resistance. Make sure to install termination resistance on each end.

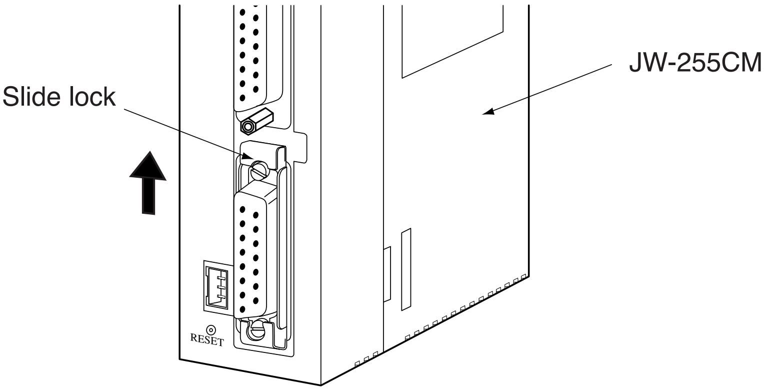
6-2 Connection method

[1] Installation of JW-255CM

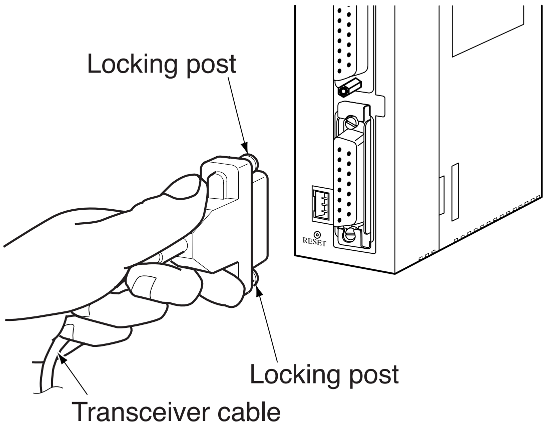
This paragraph describes how to connect the JW-255CM to a 10BASE5 system.

(1) Connecting transceiver cable

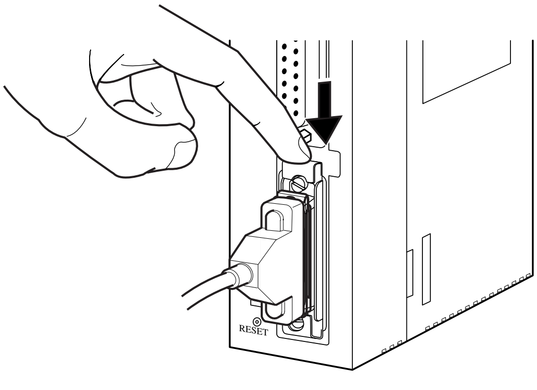
1) Slide the lock on the 10BASE5 connector (on the JW-255CM) up.



2) Insert the connector so that the two locking posts on the cable connector match the holes on the slide lock.



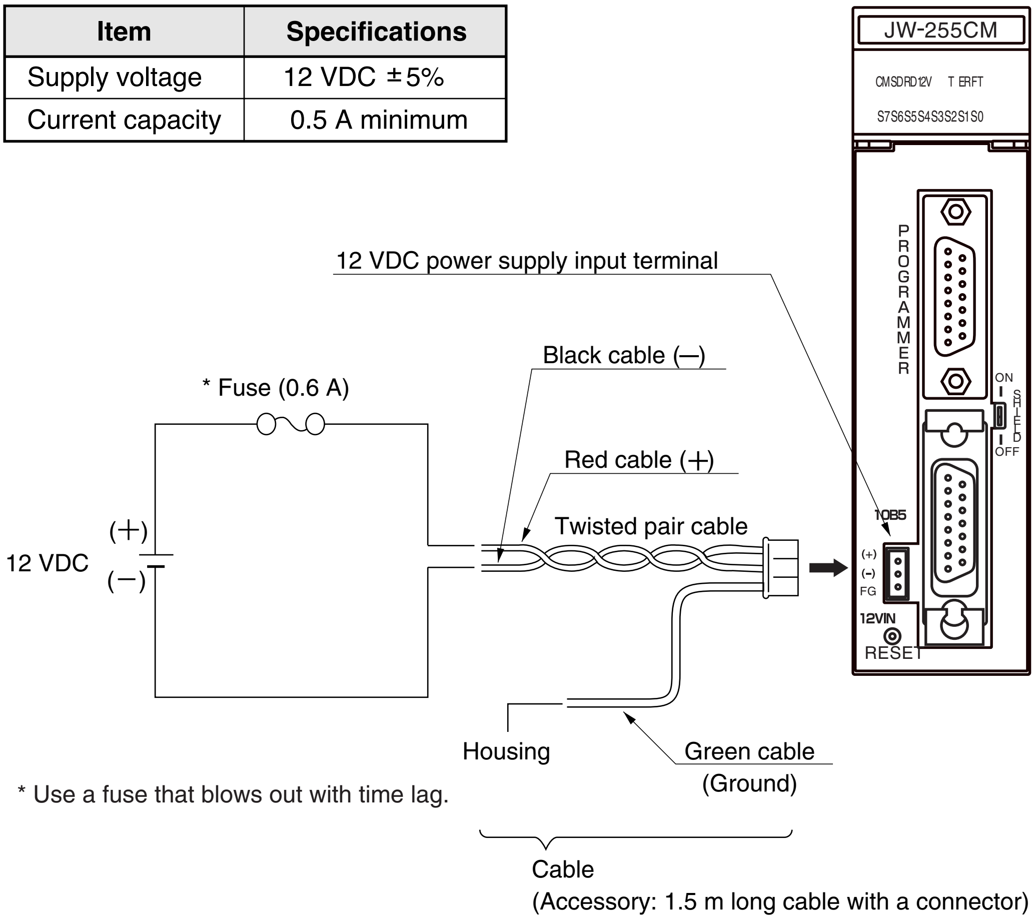
3) Slide the lock down to lock the cable connector.



(2) Wiring power source

When a 10BASE5 is used, 12 VDC power should be supplied to the transceiver.
Connect the supplied power cable with a connector to the 12 VDC power supply input terminal on the JW-255CM. Supply power to the terminal using a commercial constant voltage power supply (12 VDC).

Item	Specifications
Supply voltage	12 VDC ±5%
Current capacity	0.5 A minimum



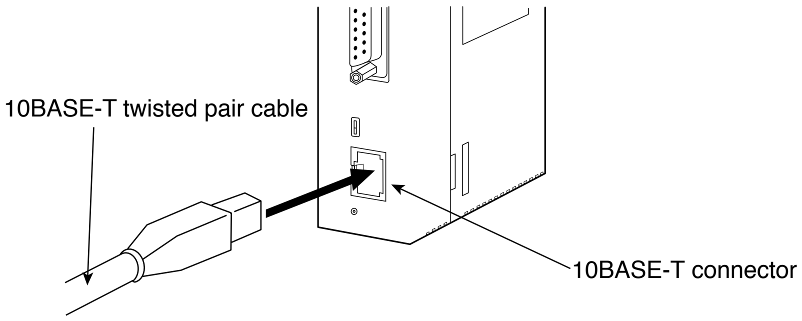
* Use a fuse that blows out with time lag.

Remarks

- Use a power supply that is dedicated for use by the JW-255CM.
- Do not reverse the positive and negative connections to the power terminals. Reversing the polarity may damage the JW-255CM.

[2] Installation of JW-25TCM

Connect a connector for a 10BASE-T twisted pair cable to the 10BASE-T connector on the JW-25TCM.

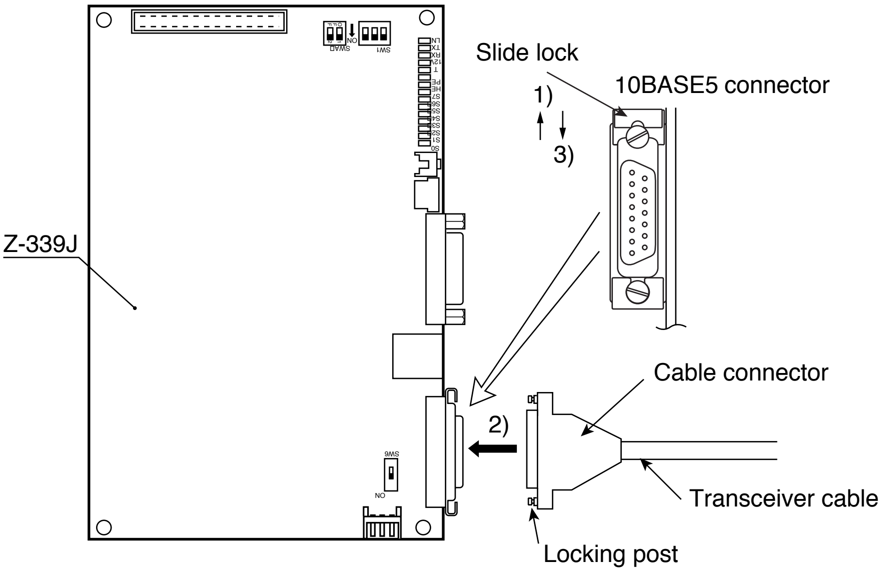


[3] Installation of Z-339J

(1) Connecting 10BASE5

This paragraph describes how to connect the Z-339J to a 10BASE5 system.

● Connnecting the transceiver cable



1) Slide the lock for 10BASE5 connector (on the Z-339J) up.
2) Insert the connector so that the two locking posts on the cable connector match the holes on the slide lock.
3) Slide the lock down to lock the cable connector.

Remarks

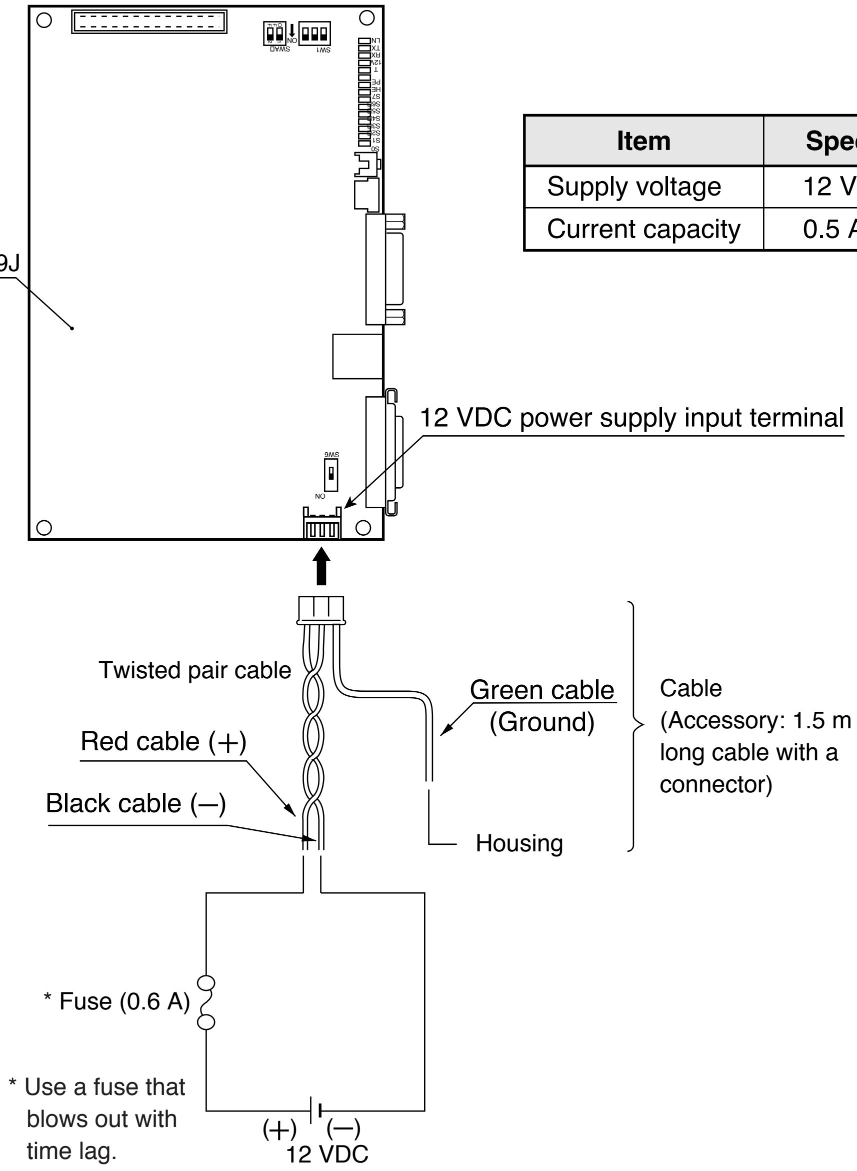
- Either of 10BASE5 or 10BASE-T can be used for communication. (Shared use of the both is not possible.)
  The board automatically switches between 10BASE5/10BASE-T by whether or not 12 VDC (next page) is supplied.

12 VDC supply	10BASE5	10BASE-T
Provided	○	×
None	×	○

● Wiring power source

When a 10BASE5 is used, 12 VDC power should be supplied to the transceiver.
Connect the supplied power cable with a connector to the 12 VDC power supply input terminal on the Z-339J. Supply power to the terminal using a commercial constant voltage power supply (12 VDC).

Item	Specifications
Supply voltage	12 VDC ±5%
Current capacity	0.5 A minimum


Z-339J
12 VDC power supply input terminal
Twisted pair cable
Green cable
(Ground)
Red cable (+)
Black cable (–)
Cable
(Accessory: 1.5 m long cable with a connector)
Housing
* Fuse (0.6 A)
* Use a fuse that blows out with time lag.
(+) (–)
12 VDC

6

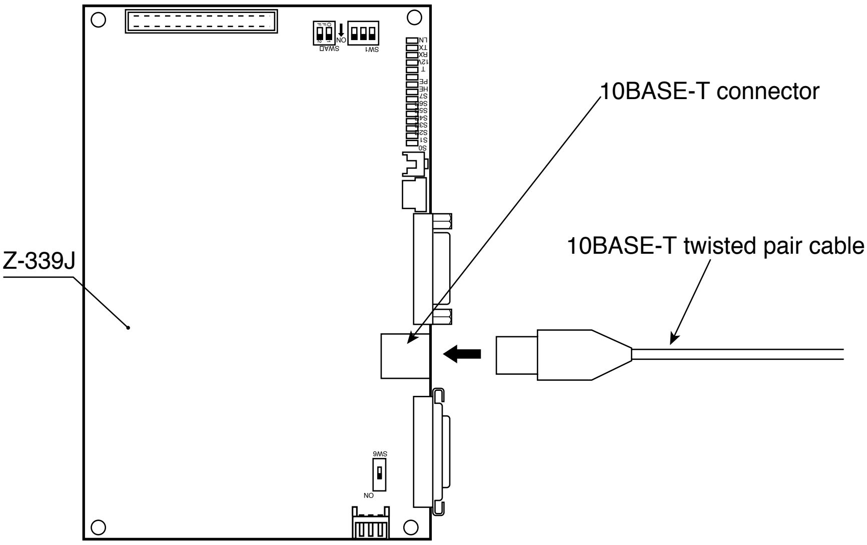
Remarks

- Use a power supply that is dedicated for use by the Z-339J.
- Do not reverse the positive and negative connections to the power terminals. Reversing the polarity may damage the Z-339J.
- The board automatically switches between 10BASE5/10BASE-T by whether or not 12 VDC is supplied.

12 VDC supply	10BASE5	10BASE-T
Provided	○	×
None	×	○

[4] Installation of 10BASE-T

Connect a connector for a 10BASE-T twisted pair cable to the 10BASE-T connector on the Z-339-J.



Remarks

- Either of 10BASE5 or 10BASE-T can be used for communication. (Shared use of the both is not possible.)
  The board automatically switches between 10BASE5/10BASE-T by whether or not 12 VDC (previous page) is supplied.

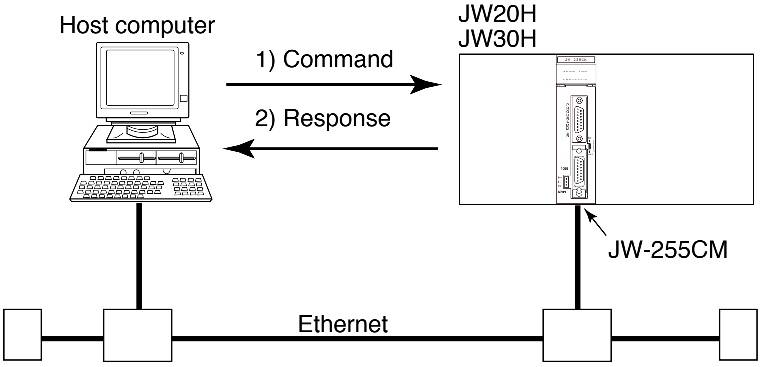
12 VDC supply	10BASE5	10BASE-T
Provided	○	×
None	×	○

Chapter 7. Overview of Function

7-1 Computer link function

The JW20H/30H, and J-board's data can be read or written to a connected programmable controller with commands from the host computer.

● **Example of connection**



1) The host computer instructs station number/communication contents/memory address/data etc. of the communicating station as a "command."
2) The "command" receiving station processes this data and returns the result as "response."

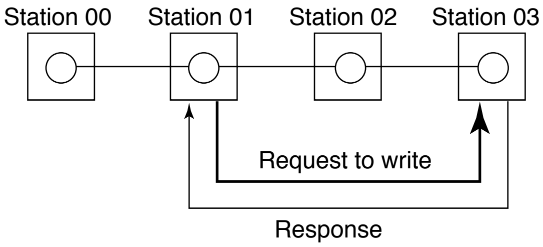
The command contains three types: read, write, and control commands.

Type	Function
Read command	Monitor relay Monitor timer/counter current value Monitor the register Read program memory Read system memory Read date Read time Read out the standard buffer Read out the ring buffer
Write command	Set/reset relay Set/reset timer or counter Write to register Write same data to register Write program Write to system memory Set date Set time Write to the standard buffer Write to the ring buffer
Control command	Monitor PLC operation status PLC stop/release stop operation Set write enable mode Monitor write enable mode Read out the standard buffer data Write the standard buffer data Read out the ring buffer data Write the ring buffer data

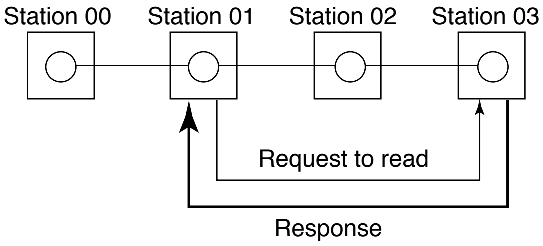
7-2 SEND/RECEIVE function

The SEND/RECEIVE function allows the JW-255CM/25TCM, and Z-339J to send data to other stations and receive data from other stations.

[An example of the SEND function]



[An example of the RECEIVE function]



The SEND/RECEIVE functions can use either the data instruction system or data memory starting system.

(1) Instruction system
The instruction system uses the application instructions F-202 (OPCH), F-204 (SEND), and F-205 (RCV), available with the JW20H/30H, and J-board.

(2) Data memory starting system
The data memory starting system places the target station No., the number of transmission bytes etc. in the data memory (communication information storage area).

Item	Instruction system	Data memory starting system
Number of channels	4	1
Number of data bytes	256 bytes max. in one instruction	1024 bytes max.
Port used	Starting from channel 0, 6000(H), 6001(H), 6002(H), and 6003(H) in order.	6008(H)

7-3 Network parameter settings

The following items are set for use as network parameters in the EEPROM.
These parameters are read when the JW-255CM/25TCM, and Z-339J (simply called "this module" from here on) starts up, and they control the details of each operation.
1) IP address, subnet mask
2)Method for opening each type of connection (TCP_Passive/TCP_Active/UDP) and port No. to use.
3)Address settings for the send/receive functions
4)Settings related to the specified buffer command
5)Settings related to the ring buffer command
6)Settings for routing
7)Settings related to the connection status flag
8)Settings related to the completion information of the computer link command

After the power is turned ON, this module will open each channel according to the details stored in the EEPROM. The method for opening a channel varies with the parameter settings, as shown below.

(1) TCP_Passive

The port which is opened after selecting the TCP_Passive mode waits for a connection from the other station.
This mode can be used in communication target stations with a computer link function or when the SEND/RECEIVE function is selected.
Connections opened in the TCP_Passive mode cannot be disconnected by the module using that mode. The station opened in the TCP_Passive cannot open or disconnect any connection. However, it can start instructions of the SEND/RECEIVE function. The port which is under opening the connection cannot communicate with other stations.

(2) TCP_Active

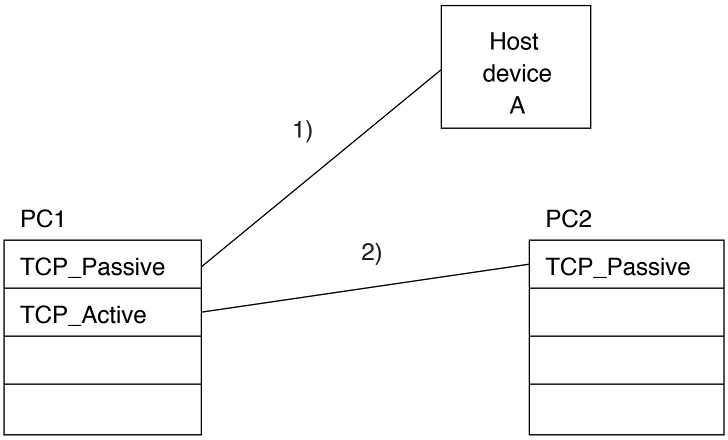
The TCP_Active mode is used to open connections to other stations. This mode can be used with a command triggering station using the SEND/RECEIVE functions. By using this method, the connection to another station can also be broken. While a connection is open, the port cannot communicate with other stations.

(3) UDP

The UDP mode is a mode not to open any connection. It can be selected by the computer link or with SEND/RECEIVE functions. The UDP is less reliable than TCP, since it does not allow confirmation of the data receipt (checking to see the data was received by the target station) at the data transmission protocol stage.

[Example]

Set the open method used for the communication between PC1, PC2, and the host device A.
1) Host device A communicates with PC1 using the TCP over the computer link.
2) PC1 communicates with PC2 using the send command (TCP_Passive).



Set the IP address and open method for each connection at the parameter addresses shown below. The following settings are essential when using the JW-255CM/25TCM, and Z-339J.

Parameter address	Details
0000	IP addresses inside the JW-255CM/25TCM, and Z-399J (0003 is used by the host.) For the details about IP addresses, refer to next page.
0001	
0002	
0003	
0004	Subnet mask => Refer to page 7-6
0005	
0006	
0007	
0100 to 0103	Settings for connection 0 => Refer to page 7-7 0100: Open method 00(H): TCP_Passive, 80(H): TCP_Active, 01(H): UDP 0101: 00 0102, 0103: JW-255CM port number (0102 as low, 0103 as high)
0104 to 0107	Settings for connection 1 (The setting details are the same as for connection 0.)
0110 to 0113	Settings for connection 2 (The setting details are the same as for connection 0.)
0114 to 0117	Settings for connection 3 (The setting details are the same as for connection 0.)
0120 to 0123	Settings for connection 4 (The setting details are the same as for connection 0.)
0124 to 0127	Settings for connection 5 (The setting details are the same as for connection 0.)
0130 to 0133	Settings for connection 6 (The setting details are the same as for connection 0.)
0134 to 0137	Settings for connection 7 (The setting details are the same as for connection 0.)

To start/stop communication, set to parameter address 3777.

Parameter address	Details
3777	Communication start switch 00(H): Halts communication 01(H): Checks the parameter, checks the BCC, and starts operation 08(H): Initializes the parameters 80(H): Checks the parameters, creates a BCC, writes it to EEPROM, and halts operation 81(H): Checks the parameter, creates a BCC, writes it to EEPROM, and starts operation (If the operation is resumed, this parameter will change to 01(H).)

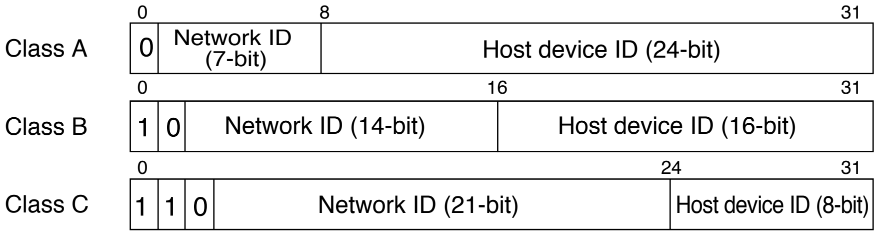
For the details about other parameter addresses, refer to Chapters 8, 9, and 12.

■ TCP and UDP

TCP is a method used for communication after establishing a connection with a target node. It offers a highly reliable communication environment, with control orders and automatic retransmission if an error occurs.
The TCP can be though as similar to the way a telephone work, due to its characteristics. (If you call someone, you can only to speak to that party until you hang up.)
UDP is a method to used for communication without needing to first establish a connection with a target node. It transmits data by assigning a target name to each transmission. If the data is not received by the target node, this module will not retransmit the data, as is the case in the TCP mode.
The UDP can be compared to writing a letter, due to its characteristics. (You send a letter after writing the address of a single recipient on the envelop.)

■ IP addresses

IP addresses are used to distinguish devices, which are communicating on a single Ethernet network. They are 32 bits long.
The IP address consists of the net ID, indicating the network device No., and the host ID, indicating the node No. inside the network. They are three classes of IP address, according to the number of ID bits used.



The numbers of network devices and hosts that can be identified, depend on the class of IP address used.

Class	Number of devices in the network	Number of host devices
Class A	Small scale	More than 65536
Class B	Middle scale	256 to 65535
Class C	Large scale	Less than 255

The 32 bits data in the address are divided into 8 bit groups, expressed in decimal notation, and linked together using periods.

[Example]

The following is a class C IP address: 192.9.200.2
11000000 00001001 11001000 00000010

Use the same net ID for devices in the same network. Specify an IP address that is different from the addresses for all other devices.
Enter the IP address in the parameter addresses (0000 to 0003) in the module.
In case of the example shown above, store the IP address in the parameter addresses as follows.

Parameter address	Set value(D)
0000	192
0001	9
0002	200
0003	2

■ Subnet mask

IP addresses are expressed using two types of identifiers (an IP address (see the NOTE) and a subnet mask address). The subnet mask indicates the length of the network address (network ID) contained in the bits of the IP address. With a subnet mask, the IP addresses in the each class can be used to divide a conceptual network into multiple physical networks (subnets). The subnet mask addresses should be allocated sequentially, starting with the upper most bit.

NOTE: The IP address described here refers to an IP address without a subnet mask.

● Subnet mask examples

The example shown below describes a subnet mask set to 255.255.255.0 with a class B IP address of 172.20.100.52.

When to indicate 170.20.100.52 in binary notation

IP address : 10101100 00010100 01100100 00110100
(Underlined bits are the class B network ID.)
Subnet mask : 11111111 11111111 11111111 00000000
10101100 00010100 01100100 00110100
(Underlined bits indicate a network ID that is extended with a subnet mask.)

When setting the ID using the above subnet mask

Network ID : 10101100 00010100 01100100 00000000 (172.20.100.0)
Host ID : 10101100 00010100 01100100 00000001 (172.20.100.1)
to to
10101100 00010100 01100100 11111110 (172.20.100.254)
(All underlined bits are for a network ID that is set using a subnet mask.)
Broadcast address : 10101100 00010100 01100100 11111111 (172.20.100.255)
(All underlined bits are for a network ID that is set using a subnet mask.)

· A broadcast address is used to transmit packets to all hosts connected to the same network.

Networks that are divided into subnets (one group of nodes is set using the subnet mask above and another group of nodes is set using a different subnet mask) are identified by different network IDs, even if they are in the same class (class B in the example above). A router is required to let the networks communicate with each other. => Refer to page 10-3.

Assign the subnet mask address by placing it in parameter addresses 0004 to 0007 in this module. In the case of the example shown above, the subnet mask bytes in the parameter are assigned as follows:

Parameter address	Set value(D)
0004	255
0005	255
0006	255
0007	0

If all of the parameter addresses from 0004 to 0007 are set to 0, it means “a subnet is not used.” This means that the specific subnet mask address assigned is equal to the bit length of the particular class of network ID.

For example, when the IP address in this module is set to 192.168.150.3 (class C) and all of the parameters for the subnet mask are set to 0, it will be equal to assigning a subnet mask of 255.255.255.0.

■ Port No.

The port No. is the logical communication doorway provided in a node. The port number can be between 1 to 65534 (a 16-bit long). No. 0 and 65535 have special meanings.
Together with the TCP and IP, the port No. is used to identify the applicable protocols. The applicable protocols corresponding to the port No. have already been determined. (For example, the file transmission FTP is assigned to 21, and the remote terminal telnet is assigned to 23.) These are called "Well-known port." The assignment of ports 1 to 1000 have already been determined.
With this module, the port No. can be set freely in the range 1 to 65534. However, we recommend assigning a port No. (upper value No.) that is not one of the well-known port.

■ Socket and connection

In the TCP and UDP connection open methods, the IP addresses and port numbers are used to specify the destination addresses and the senders. Normally, only one value is used for the node for an IP address. However, a parallel communication process with multiple ports is possible by opening multiple ports inside a node. Then, each port becomes a logical doorway to a communication circuit and is called a "socket" in the terminology used for TCP and UDP communications.
Sockets are broadly divided into two types: One type uses the TCP, and the other uses the UDP.
The TCP forms a virtual communication route by making a connection with the communication target. This is referred to as "establishing the connection." After the connection is established, the socket can only communicate with this target. After the communication is complete, the devices perform a disconnection procedure. The TCP offers highly reliable communications with special functions, such as automatic retransmission in case of a time-out. However, the TCP has a large overhead, since connection and disconnection procedures are required, and the module must wait for confirmation from the target each time data is transmitted.
The UDP does not use a connection process to find a communication target. The data is transmitted by specifying the target each time. The UDP does not retransmit the data if it is not received by the target. Therefore, the UDP also does not need to perform any connection or disconnection procedures. However, it offers less reliability than the TCP.

Chapter 8. Computer Link Function

8-1 Basic format of computer link commands

[1] Communication format

A message from the host computer to this module is referred to as a "command." A response from this module to the host computer is referred to as a "response."
The communication formats of the command and response are as follows:

■ Command

Header (40 bytes)	c-ID	ATTR	COM	Command Text

■ Response

Header (40 bytes)	r-ID	ATTR	COM	RSLT	Response Text

Header : Normally, all 40 bytes are $00_{(H)}$.
If you want to communicate with a satellite net using this module to interface between layers of hierarchical communication, you have to use an extension header.
(See "8-8 Two-layer communication with satellite net")

c-ID : $47_{(H)}$

r-ID : $45_{(H)}$

ATTR : $00_{(H)}$

COM : Command code => Refer to page 8-4.

RSLT : Command execution result
Normally terminated with $00_{(H)}$
If any byte other than $00_{(H)}$ is found, an error code will be output => See "8-5 Computer link error code table."
If an error code is output, there is no response text.

Command Text : Command details => See "8-2 Descriptions of each command."

Response Text : Response details => See "8-2 Descriptions of each command."

[Example] When you want to monitor the ON/OFF status of relay 04033. => Refer to page 8-7.

■ Command

Header (40 bytes)			c-ID	ATTR	COM	Command Text			
00	···	00	47	00	20	00	03	01	03

File 0 (00) — File address $000403_{(8)} = 0103_{(H)}$ (03 01) — Bit 3 (03)

Relay No. 04033

■ Response

Header (40 bytes)			r-ID	ATTR	COM	RSLT	Response Text				
00	···	00	45	00	20	00	00	03	01	03	01

File 0 (00) — File address $000403_{(8)} = 0103_{(H)}$ (03 01) — Bit 3 (03) — ON (01)

Relay No. 04033

Remarks

The maximum data length for read/write operations is 1024 bytes. In case of two-layer communication with the satellite net, however, the maximum length is 256 bytes.

[2] Memory address expression format

The format expressing memory address contained in the command (command text/response text) is as shown below. => For more details, refer to "8-2 Descriptions of each command."

PSEG : Program segment (corresponds to the file number.)

	JW20H	JW30H *1	J-board (Z300/Z500)
PSEG	$08_{(H)}$	$08_{(H)}$, $09_{(H)}$	$08_{(H)}$

*1 The memory capacity varies, depending on the control module type. (Shown above are maximum values.)

PADR : Program address

	JW20H *2	JW30H *3	J-board (Z300) *4	J-board (Z500) *4
PADR	0000 to $1DFF_{(H)}$	0000 to $7DFF_{(H)}$	0000 to $1DFF_{(H)}$	0000 to $7DFF_{(H)}$

*2 The memory capacity varies, depending on the memory module type.
*3 The memory capacity varies, depending on the control module type.
*4 The memory capacity varies, depending on the CPU board type.
(Shown above are maximum values.)

The program address is specified using PSEG and PADR.

Address 000000 to $076777_{(8)}$: PSEG = 8, PADR is the address expressed in hexadecimal notation.
Address 100000 to $176777_{(8)}$: PSEG = 9, PADR is the value in hexadecimal notation obtained by subtracting $100000_{(8)}$ from the address.

[Example] Address $043256_{(8)}$: PSEG = $08_{(H)}$, PADR= $46AE_{(H)}$
Address $153762_{(8)}$: PSEG= $09_{(H)}$, PADR = $57F2_{(H)}$

DSEG : Data memory segment (corresponds to the file number.)

	JW20H	JW30H *5	J-board (Z300)	J-board (Z500)
DSEG	$00_{(H)}$	00 to $03_{(H)}$, 10 to $2C_{(H)}$	$00_{(H)}$	$00_{(H)}$ to $02_{(H)}$

*5 The memory capacity varies, depending on the control module type. (Shown above are maximum values.)

DADR : Data memory address (corresponds to the file number.)

	JW20H J-board (Z300)	JW30H *6		J-board (Z500) *7	
DADR	0000 to $1FFF_{(H)}$	(Valid DSEG values) $00_{(H)}$	0000 to $3BFF_{(H)}$	(Valid DSEG values) $00_{(H)}$	0000 to $3BFF_{(H)}$
		$01_{(H)}$	0000 to $3FFF_{(H)}$	$01_{(H)}$	0000 to $3FFF_{(H)}$
		02 to $03_{(H)}$	0000 to $FFFF_{(H)}$	$02_{(H)}$	0000 to $FFFF_{(H)}$
		10 to $2C_{(H)}$	0000 to $FFFF_{(H)}$		

*6 The memory capacity varies, depending on the control module type.
*7 The memory capacity varies, depending on the CPU board type.
(Shown above are maximum values.)

BLOC : Bit location in the data memory
The register (file register) is specified using DSEG and DADR.
[Example] Register 09000 : DSEG = 00(H), DADR = 0800(H)
030000 in file 1 : DSEG = 01(H), DADR = 3000(H)

The relay address is specified using DSEG, DADR, and BLOC. The destination address is composed of a combination of the file address and the bit location.
[Example] Relay 07252: DSEG = 00(H), DADR = 01D5(H), BLOC = 02(H)
(bit 2 of the file address 000725 (]0725))

TADR : Timer/counter number
The timer/counter number is specified using TADR (in hexadecimal notation).

	JW20H	JW30H	J-board (Z300)	J-board (Z500)
TADR	0000 to 01FF(H)	0000 to 03FF(H)	0000 to 01FF(H)	0000 to 03FF(H)

SADR : System memory address
The system memory address is specified using SADR (in hexadecimal notation).
The SEG is contained in the command. Always set SEG to 08(H).

	JW20H	JW30H	J-board (Z300)	J-board (Z500)
SADR	0000 to 00FF(H)	0000 to 047F(H)	0000 to 00FF(H)	0000 to 047F(H)

[3] Execution condition

(1) Write enable mode

Each command will be executed depending on the current status of the write enable mode.

Write enable mode	Details
Mode 0	Writing to all of memory is prohibited
Mode 1	Writing is only enabled to data memory
Mode 2	Writing is enabled to all of memory

When the power is first applied, this module is in "mode 0." Therefore, if you want to write data from the host computer, change to "mode 1" or "mode 2" using the setting command (command code F9(H)). The current status can be read using the reading command (command code E9(H)) for the write enable command.

(2) PLC operation status

Some commands can be executed when the PLC halts operation (writing programs: command code 14(H) etc.). Other commands can be executed whether the PLC is halted or is running (reading programs: command code 04(H) etc.)

[4] Table of commands

Command code	Contents	Reference page
04(H)	Reading program	8-16
14(H)	Write program	8-17
20(H)	Monitoring relay	8-7
23(H)	The current value monitor of the timers/counters	8-10
24(H)	Monitoring register	8-11
28(H)	Read from a standard buffer	8-28
29(H)	Read a ring buffer	8-41
30(H)	Set/reset relay	8-8
32(H)	Set/reset timer/counter	8-9
34(H)	Write in register	8-12
35(H)	Write same data to register	8-13
38(H)	Write to a standard buffer	8-29
39(H)	Write to a ring buffer	8-43
44(H)	Read out the system memory	8-14
54(H)	Write to the system memory	8-15
68(H)	Read information about a standard buffer	8-30
69(H)	Read information about a ring buffer	8-45
78(H)	Write information about a standard buffer	8-31
79(H)	Write information about a ring buffer	8-47
A2(H)	Read date	8-18
A3(H)	Read time	8-20
B2(H)	Set date	8-19
B3(H)	Set time	8-21
E8(H)	Monitor PLC operation status	8-22
E9(H)	Read out write enable mode	8-5
F8(H)	Halt and release halting of PLC	8-23
F9(H)	Selecting the write enable mode	8-6

8-2 Descriptions of each command

This section describes the "COM" settings and the items thereafter of the communication formats.
=> Refer to page 8-1.
Commands for the standard buffer are described on pages 8-28 to 8-31. Commands for the ring buffer are described on pages 8-41 to 8-48.

Read out write enable mode (COM=E9$_{(H)}$)

[Format]

■ **Command**

COM

■ **Response**

COM	RSLT	WMOD

COM = E9$_{(H)}$
WMOD = 00$_{(H)}$: Mode 0 (All memory write-disabled)
01$_{(H)}$: Mode 1 (Only the data memory write-enabled)
02$_{(H)}$: Mode 2 (All memory write-enabled)

[Function]
- Reads the status of the write-enable mode.

[Execution condition]
- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]
- Reads the status of the write-enable mode.

■ **Command**

E9

■ **Response**

E9	00	02

Mode 2 (All memory write-enabled)

Selecting the write enable mode (COM = F9(H))

[Format]

■ **Command**

COM	WMOD

■ **Response**

COM	RSLT

COM = F9(H)
WMOD = 00(H) : Mode 0 (All memory write-disabled)
01(H) : Mode 1 (Only the data memory write-enabled)
02(H) : Mode 2 (All memory write-enabled)

[Function]

- Selecting the write enable mode.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Set the write enable mode to mode 2 (Writing is enable to all of memory).

■ **Command**

F9□	02

└── Mode 2 (All memory write-enabled)

■ **Response**

F9□	00

Monitoring relay (COM = $20_{(H)}$)

[Format]

■ Command

COM	DSEG	DADR$_L$	DADR$_H$	BLOC

■ Response

COM	RSLT	DSEG	DADR$_L$	DADR$_H$	BLOC	DATA

COM = $20_{(H)}$
DSED = Segment ($00_{(H)}$ to $03_{(H)}$, $10_{(H)}$ to $2C_{(H)}$) => Refer to page 8-2.
DADR$_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)})$) => Refer to page 8-2.
BLOC = Bit position ($00_{(H)}$ to $07_{(H)}$)
DATA = Read data ($00_{(H)}$: OFF, $01_{(H)}$: ON)

[Function]

- Read the bit data (relay) shown in DSEG, DADR, and BLOC.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Monitor the ON/OFF status of relay number 04033.

■ Command

20	00	03	01	03

File 0

File address $000403_{(8)} = 0103_{(H)}$

Bit 3

Relay number 04033

■ Response

ON

20	00	00	03	01	03	01

File 0

File address $000403_{(8)} = 0103_{(H)}$

Bit 3

Relay number 04033

Set/reset relay (COM = $30_{(H)}$)

[Format]

■ **Command**

COM	DSEG	DADR$_L$	DADR$_H$	BLOC	DATA

■ **Response**

COM	RSLT	DSEG	DADR$_L$	DADR$_H$	BLOC

COM = $30_{(H)}$
DSED = Segment ($00_{(H)}$ to $03_{(H)}$, $10_{(H)}$ to $2C_{(H)}$) => Refer to page 8-2.
DADR$_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)}$) => Refer to page 8-2.
BLOC = Bit position ($00_{(H)}$ to $07_{(H)}$)
DATA = Set/reset data ($00_{(H)}$: reset, $01_{(H)}$: set)

[Function]

- Set/reset the relays shown in DSEG, DADR, and BLOC.

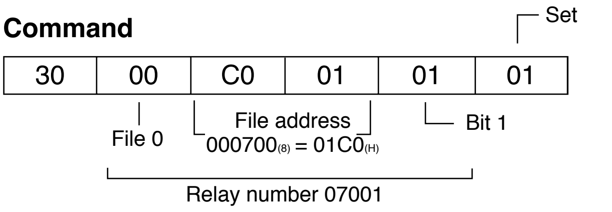
[Execution condition]

- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

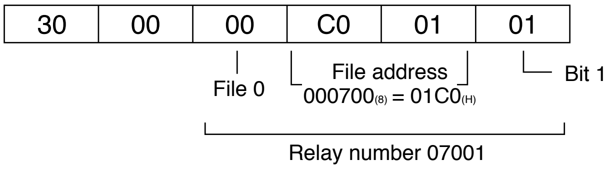
- Set relay number 07001.

■ **Command**



30	00	C0	01	01	01

■ **Response**



30	00	00	C0	01	01

Set/reset timer/counter (COM = 32$_{(H)}$)

[Format]

■ **Command**

COM	TADR$_L$	TADR$_H$	DATA

■ **Response**

COM	RSLT	TADR$_L$	TADR$_H$

COM = 32$_{(H)}$
TADR$_{L, H}$ = Timer/counter number (0000$_{(H)}$ to 03FF$_{(H)}$) => Refer to page 8-3.
DATA = Set/reset data (00$_{(H)}$: reset, 01$_{(H)}$: set)

[Function]

- Set/reset the timer/counter displayed on TADR.

[Execution condition]

- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Set TMR0002.

■ **Command**

32	02	00	01

Timer and counter number 0002 Set

■ **Response**

32	00	02	00

Timer and counter number 0002

The current value monitor of the timers/counters (COM = 23$_{(H)}$)

[Format]

■ Command

COM	TADR$_L$	TADR$_H$	L$_L$	L$_H$

■ Response

COM	RSLT	TADR$_L$	TADR$_H$	L$_L$	L$_H$	DATA$_1$	...

	DATA$_N$	ATTR$_1$	...	ATTR$_N$

COM = 23$_{(H)}$
TADR$_{L, H}$ = Timer and counter number (0000$_{(H)}$ to 03FF$_{(H)}$) *Refer to page 8-3.
L$_{L, H}$ = Number of data to read
DATA$_{1 \text{ to } N}$ = The current value data (read current value field of the timer and the counter)
ATTR$_{1 \text{ to } N}$ = The attribute data of the timer and the counter

[Function]

- Reads the current values and the attributes of the timers/counters identified by the starting number TADR and the number of data L.
- Up to 256 timers/counters can be read at a time.
- The current value data is read from the timer/counter's current range (b0000 to xxxxx).
- The attributes are as follows.

00$_{(H)}$	Not in use	0A$_{(H)}$	UTMR(BCD)
01$_{(H)}$	MD	0B$_{(H)}$	UTMR(BIN)
02$_{(H)}$	CNT	0C$_{(H)}$	DCNT(BCD)
04$_{(H)}$	TMR	0D$_{(H)}$	DCNT(BIN)
08$_{(H)}$	DTMR(BCD)	0E$_{(H)}$	UCNT(BCD)
09$_{(H)}$	DTMR(BIN)	0F$_{(H)}$	UCNT(BIN)

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Reads the current values of TMR0000 and TMR0001.

■ Command

23	00	00	02	00
	Top number of the timer and the counter		Number of data	

■ Response

23	00	00	00	02	00	34	92	78	D6	08	0A
		Top number of the timer and the counter		Number of data		The current value of TMR0000 1234		The current value of TMR0001 5678		DTMR (BCD)	UTMR (BCD)

Monitoring register (COM = 24$_{(H)}$)

[Format]

■ Command

COM	DSEG	DADR$_L$	DADR$_H$	L$_L$	L$_H$

■ Response

COM	RSLT	DSEG	DADR$_L$	DADR$_H$	L$_L$	L$_H$	DATA$_1$	......	DATA$_N$

COM = 24$_{(H)}$
DSEG = Segment (00$_{(H)}$ to 03$_{(H)}$,10$_{(H)}$ to 2C$_{(H)}$) => Refer to page 8-2.
DADR$_{L, H}$ = Byte address (0000$_{(H)}$ to FFFF$_{(H))}$) => Refer to page 8-2.
L$_{L, H}$ = Data length (Number of bytes)
DATA$_{1 \text{ to } N}$ = Read data

[Function]
- Read the register data with the length shown by L, starting from DSEG, DADR.
- Up to 1024 bytes can be read at a time.

[Execution condition]
- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]
- Read 4 bytes data from register 09000 to 09003.

■ Command

24	00	00	08	04	00

File number 0
File address 0800$_{(H)}$ = 004000$_{(8)}$
Data length
Top register number 09000

■ Response

24	00	00	00	08	04	00	00	4F	32	01

File number 0
File address 0800$_{(H)}$ = 004000$_{(8)}$
Data length
Value at 09000
Value at 09001
Value at 09002
Value at 09003
Top register number 09000

Write in register (COM = 34$_{(H)}$)

[Format]

■ Command

COM	DSEG	DADR$_L$	DADR$_H$	L$_L$	L$_H$	DATA$_1$	......	DATA$_N$

■ Response

COM	RSLT	DSEG	DADR$_L$	DADR$_H$	L$_L$	L$_H$

COM = 34$_{(H)}$
DSEG = Segment (00$_{(H)}$ to 03$_{(H)}$,10$_{(H)}$ to 2C$_{(H)}$) => Refer to page 8-2.
DADR$_{L, H}$ = Byte address (0000$_{(H)}$ to FFFF$_{(H)}$) => Refer to page 8-2.
L$_{L, H}$ = Data length (number of bytes)
DATA$_{1 \text{ to } N}$= Write data

[Function]

- Write the register data with the length shown by L, starting from DSEG, DADR.
- Up to 1024 bytes can be write at a time.

[Execution condition]

- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Write 00$_{(H)}$, 4F$_{(H)}$, 32$_{(H)}$, and 01$_{(H)}$ to registers 09000 to 09003.

■ Command

34	00	00	08	04	00	00	4F	32	01
	File number 0	File address 0800$_{(H)}$ = 004000$_{(8)}$		Data length		Value at 09000	Value at 09001	Value at 09002	Value at 09003
	Top register number 09000								

■ Response

34	00	00	00	08	04	00
		File number 0	File address 0800$_{(H)}$ = 004000$_{(8)}$		Data length	
		Top register number 09000				

Write same data to register (COM = 35$_{(H)}$)

[Format]

■ **Command**

COM	DSEG	DADR$_L$	DADR$_H$	L$_L$	L$_H$	DATA

■ **Response**

COM	RSLT	DSEG	DADR$_L$	DADR$_H$	L$_L$	L$_H$

COM = 35$_{(H)}$
DSEG = Segment (00$_{(H)}$ to 03$_{(H)}$,10$_{(H)}$ to 2C$_{(H)}$) => Refer to page 8-2.
DADR$_{L, H}$ = Byte address (0000$_{(H)}$ to FFFF$_{(H)}$) => Refer to page 8-2.
L$_{L,H}$ = Data length (number of bytes)
DATA = Write data

[Function]
- Write the same data with the length shown by L, starting from DSEG, DADR.

[Execution condition]
- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]
- Write 4F$_{(H)}$ to register 19000 to 19003 (4 bytes).

■ **Command**

35	00	00	0A	04	00	4F

File number 0
File address
0A00$_{(H)}$ = 005000$_{(8)}$
Data length
Data
Top register
number 19000

■ **Response**

35	00	00	00	0A	04	00

File number 0
File address
0A00$_{(H)}$ = 005000$_{(8)}$
Data length
Top register
number 19000

Read out the system memory (COM = 44$_{(H)}$)

[Format]

■ Command

COM	SEG	SADR$_L$	SADR$_H$	L$_L$	L$_H$

■ Response

COM	RSLT	SEG	SADR$_L$	SADR$_H$	L$_L$	L$_H$	DATA$_1$	......	DATA$_N$

COM = 44$_{(H)}$
SEG = Segment (08$_{(H)}$)
SADR$_{L,H}$ = System memory address (0000$_{(H)}$ to 047F$_{(H)}$) => Refer to page 8-3.
L$_{L,H}$ = Data length (number of bytes)
DATA$_{1 \text{ to } N}$ = Read data

[Function]

- Read the system memory data with the length shown by L, starting from SEG, SADR.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Read data of system memory #204 to #207.

■ Command

44	08	84	00	04	00
		System memory address 0084$_{(H)}$ = 000204$_{(8)}$		Data length	

■ Response

44	00	08	84	00	04	00	80	01	08	00
			System memory address 0084$_{(H)}$ = 000204$_{(8)}$		Data length		Value at #204	Value at #205	Value at #206	Value at #207

Write to the system memory (COM = $54_{(H)}$)

[Format]

■ **Command**

COM	SEG	SADR_L	SADR_H	L_L	L_H	DATA_1	......	DATA_N

■ **Response**

COM	RSLT	SEG	SADR_L	SADR_H	L_L	L_H

COM	= $54_{(H)}$
SEG	= Segment ($08_{(H)}$)
$\text{SADR}_\text{L, H}$	= System memory address ($0000_{(H)}$ to $047F_{(H)}$) => Refer to page 8-3.
$\text{L}_\text{L, H}$	= Data length (number of bytes)
$\text{DATA}_\text{1 to N}$	= Write data

[Function]

- Write the system memory data with the length shown by L, starting from SEG, SADR.

[Execution condition]

- Write enable mode: Mode 2
- PLC operation status: Stopping

[Example]

- Set $81_{(H)}$, $00_{(H)}$, $00_{(H)}$, and $04_{(H)}$ to system memory #204 to #207.

■ **Command**

54	08	84	00	04	00	81	00	00	04
		System memory address $0084_{(H)}=000204_{(8)}$		Data length		Value at #204	Value at #205	Value at #206	Value at #207

■ **Response**

54	00	08	84	00	04	00
			System memory address $0084_{(H)}=000204_{(8)}$		Data length	

Reading program (COM = 04$_{(H)}$)

[Format]

■ **Command**

COM	PSEG	PADR$_L$	PADR$_H$	L$_L$	L$_H$

■ **Response**

COM	RSLT	PSEG	PADR$_L$	PADR$_H$	L$_L$	L$_H$	DATA$_1$	......	DATA$_N$

COM = 04$_{(H)}$
PSEG = Program segment (08$_{(H)}$, 09$_{(H)}$) => Refer to page 8-2.
PADR$_{L,H}$ = Program address (0000$_{(H)}$ to 7DFF$_{(H)}$) => Refer to page 8-2.
L$_{L,H}$ = Data length (number of words)
DATA$_{1 \text{ to } N}$ = Read data (2 bytes = one step)

[Function]

- Read a program with a length (number of words) shown by L, from address PSEG, PADR.
- Up to 512 words can be read at a time.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Read the contents of the program address 000000 to 000002 (file number 8)

■ **Command**

04	08	00	00	03	00
		Top program address		Data length	

■ **Response**

04	00	08	00	00	03	00	00	80	00	91
			Top program address		Data length		Address 000000 contents		Address 000001 contents	

08	B8
Address 000002 contents	

Note: Inquiries concerning the bit configuration of programs cannot be accepted.

Write program (COM = 14$_{(H)}$)

[Format]

■ **Command**

COM	PSEG	PADR$_L$	PADR$_H$	L$_L$	L$_H$	DATA$_1$	.....	DATA$_N$

■ **Response**

COM	RSLT	PSEG	PADR$_L$	PADR$_H$	L$_L$	L$_H$

COM = 14$_{(H)}$
PSEG = Program segment (08$_{(H)}$, 09$_{(H)}$) => Refer to page 8-2.
PADR$_{L, H}$ = Program address (0000$_{(H)}$ to 7DFF$_{(H)}$) => Refer to page 8-2.
L$_{L, H}$ = Data length (number of words)
DATA$_{1 \text{ to } N}$ = Write data (2 bytes = one step)

[Function]

- Write a program with a length (number of words) shown by L, from address PSEG, PADR.
- Up to 512 words can be write at a time.

[Execution condition]

- Write enable mode: Mode 2
- PLC operation status: Stopping

[Example]

- Write the contents below in program address 000000 to 000002 (file number 8).

■ **Command**

14	08	00	00	03	00	00	80	00	91	08	B8
		Top program address		Data length		Address 000000 contents		Address 000001 contents		Address 000002 contents	

■ **Response**

14	00	08	00	00	03	00
			Top program address		Data length	

Note: Inquiries concerning the bit configuration of programs cannot be accepted.

Read date (COM = $A2_{(H)}$)

[Format]

■ **Command**

COM

■ **Response**

COM	RSLT	Y	M	D	DW

COM = $A2_{(H)}$
Y = Year (express lower two digits of Western year, $00_{(H)}$ to $99_{(H)}$)
M = Month ($01_{(H)}$ to $12_{(H)}$)
D = Date ($01_{(H)}$ to $31_{(H)}$)
DW = Day of week ($00_{(H)}$: Sunday, $01_{(H)}$: Monday, $02_{(H)}$: Tuesday, $03_{(H)}$: Wednesday, $04_{(H)}$: Thursday, $05_{(H)}$: Friday, $06_{(H)}$: Saturday)

[Function]
- Read date data.

[Execution condition]
- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]
- Read date data.

■ **Command**

A2

■ **Response**

A2	00	03	12	17	03
		’03	December	17	Wednesday

Set date (COM = B2(H))

[Format]

■ **Command**

COM	Y	M	D	DW

■ **Response**

COM	RSLT

COM	= B2(H)
Y	= Year (express lower two digits of Western year in BCD. 00(H) to 99(H))
M	= Month (01(H) to 12(H))
D	= Date (01(H) to 31(H))
DW	= Day of week (00(H): Sunday, 01(H): Monday, 02(H): Tuesday, 03(H): Wednesday, 04(H): Thursday, 05(H): Friday, 06(H): Saturday)

[Function]
- Set date data.

[Execution condition]
- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]
- Set data to Friday, January 23, 2003

■ **Command**

B2	03	01	23	05
	'03	January	23	Friday

■ **Response**

B2	00

Read time (COM = A3(H))

[Format]

■ **Command**

COM

■ **Response**

COM	RSLT	H	M	S

COM	= A3(H)	
H	= Hour	(00(H) to 23(H): BCD)
M	= Minute	(00(H) to 59(H): BCD)
S	= Second	(00(H) to 59(H): BCD)

[Function]

- Read time data.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Read time data.

■ **Command**

A3

■ **Response**

A3	00	21	12	37

21 o'clock 12 minutes 37 seconds

Set time (COM = B3$_{(H)}$)

[Format]

■ **Command**

COM	H	M	S	CTRL

■ **Response**

COM	ACK

COM	=B3$_{(H)}$	
H	= Hour	(00$_{(H)}$ to 23$_{(H)}$: BCD)
M	= Minute	(00$_{(H)}$ to 59$_{(H)}$: BCD)
S	= Second	(00$_{(H)}$ to 59$_{(H)}$: BCD)
CTRL	= Control data	00$_{(H)}$: Run clock 01$_{(H)}$: Stop clock

[Function]
- Write time data

[Execution condition]
- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]
- Set time data to 18 o'clock, 10 minutes, and 20 seconds.

■ **Command**

B3	18	10	20	00
	18 o'clock	10 minutes	20 seconds	Run clock

■ **Response**

B3	00

Monitor PLC operation status (COM = E8$_{(H)}$)

[Format]

■ **Command**

COM

■ **Response**

COM	RSLT	MODE

COM = E8$_{(H)}$
MODE = 00$_{(H)}$: Operating
01$_{(H)}$: Stopped operation by an instruction from other module.
02$_{(H)}$: Stopped operation by an instruction from this module.

[Function]

- Monitor PLC run/stop status.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Monitor PLC operation status.

■ **Command**

E8

■ **Response**

E8	00	00

└─ Operating

Halt and release halting of PLC (COM = F8$_{(H)}$)

[Format]

■ **Command**

COM	MODE

■ **Response**

COM	RSLT	MODE

COM = F8$_{(H)}$
MODE = 00$_{(H)}$: Release halt
01$_{(H)}$: Halt

[Function]

- Halt/release halting of PLC operation.

[Execution condition]

- Write enable mode: Mode 0, mode 1 and mode 2
- PLC operation status: Stopping, operating

[Example]

- Halt PLC operation

■ **Command**

F8	01

Stopping

■ **Response**

F8	00	01

8-3 Standard buffers

To access a file register using normal computer commands (command code $24_{(H)}$, $34_{(H)}$, etc.), a file register address must be assigned.

Otherwise, you can use commands for a standard buffer. In this case, you have to set up a buffer in the PLC data memory, and assign it a number. Then, you call the buffer by number to select it, not its address. The second method makes it possible to create an application without knowing the actual address in the PLC memory.

■ Commands for standard buffers

Command code	Details	Reference page
$28_{(H)}$	Read from a standard buffer	8-28
$38_{(H)}$	Write to a standard buffer	8-29
$68_{(H)}$	Read information about a standard buffer	8-30
$78_{(H)}$	Write information about a standard buffer	8-31

[1] How to specify a standard buffer

Establish a standard buffer in data memory. The buffer size can be specified (up to 64 k-bytes), in units of one byte. A maximum of 32 buffers can be referenced. Their buffer numbers, 00 to 1F, identifies these buffers.

The following area in data memory can be allocated to standard buffers.

JW20H, J-board (Z300)		JW30H *1		J-board (Z500) *2	
File number	File addresses	File number	File addresses	File number	File addresses
File 0	000000 to $017777_{(8)}$	File 0	000000 to $035777_{(8)}$	File 0	000000 to $035777_{(8)}$
		File 1	000000 to $037777_{(8)}$	File 1	000000 to $037777_{(8)}$
		File 2 to 3, 10 to $2C_{(H)}$	000000 to $177777_{(8)}$	File 2	000000 to $177777_{(8)}$

*1 The JW30H memory capacity varies, depending on the control module type.
*2 The Z500 memory capacity varies, depending on the CPU board type.
(Shown above are maximum values.)

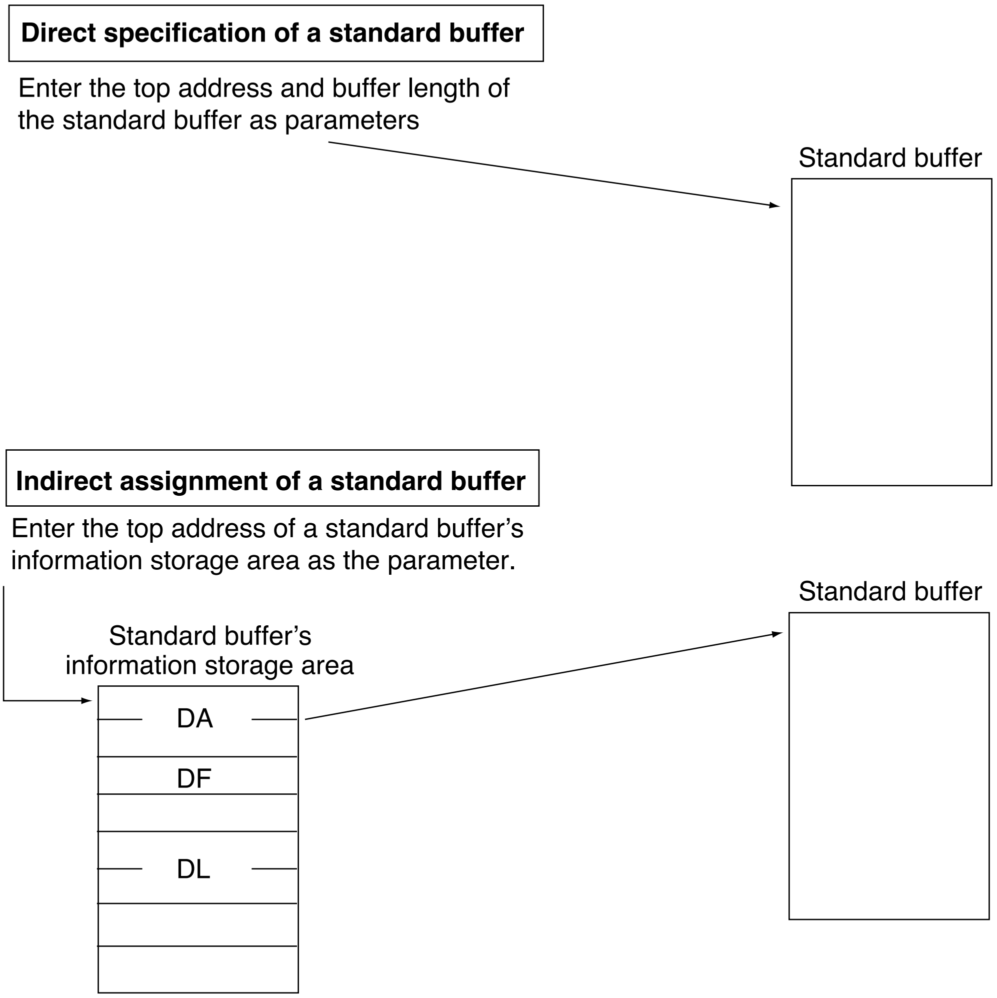
To specify a standard buffer area, specify the top file address DA, the file number DF, and the buffer length DL. Both direct and indirect methods of creation can be used.

1) Direct specification

A method used to specify the top address, file number, and buffer length directly as this module parameters.

2) Indirect specification

Enter the top address, and file number for a standard buffer's information storage area. Then enter the top file address, file number, and buffer length, into that information storage area.


Direct specification of a standard buffer
Enter the top address and buffer length of the standard buffer as parameters
Standard buffer
Indirect assignment of a standard buffer
Enter the top address of a standard buffer's information storage area as the parameter.
Standard buffer's information storage area
DA
DF
DL
Standard buffer

The following data memory area can be used as a standard buffer's information storage area.

JW20H, J-board (Z300)		JW30H *1		J-board (Z500) *2	
File number	File addresses	File number	File addresses	File number	File addresses
File 0	000000 to 017777$_{(8)}$	File 0	000000 to 035777$_{(8)}$	File 0	000000 to 035777$_{(8)}$
		File 1	000000 to 037777$_{(8)}$	File 1	000000 to 037777$_{(8)}$
		File 2 to 3, 10 to 2C$_{(H)}$	000000 to 177777$_{(8)}$	File 2	000000 to 177777$_{(8)}$

*1 The JW30H memory capacity varies, depending on the control module type.
*2 The Z500 memory capacity varies, depending on the CPU board type.
} (Shown above are maximum values.)

In order to access a standard buffer, use the read and write commands (command code 28$_{(H)}$ and 38$_{(H)}$). To use them, assign a buffer number, an offset, and the number of bytes to access. The offset is the displacement of the address from the top. If you assign 0 for the offset, this module will access the top of the buffer.

In order to access information about the buffer itself, use the read and write commands (command code 68$_{(H)}$ and 78$_{(H)}$) to get at the standard buffer information. Using these commands, the top address, file number, and buffer length can be read out by supplying the buffer number. The indirect assignment method can be used to change the information they contain.

[2] Parameter setting

Use parameters 1000 to 1377 to access the standard buffer.

<table>
<tr><th>Parameter address</th><th colspan="2">Details</th></tr>
<tr><td>1000 to 1007</td><td colspan="2">Information concerning standard buffer 00
<table>
<tr><th></th><th>When direct assignment (1007 = 80$_{(H)}$) is used</th><th>When indirect assignment (1007 = C0$_{(H)}$) is used</th></tr>
<tr><td>1000</td><td rowspan="2">Top file address of the standard buffer</td><td rowspan="2">Top file address of the standard buffer information storage area</td></tr>
<tr><td>1001</td></tr>
<tr><td>1002</td><td>File number of the standard buffer</td><td>File number of the standard buffer information storage area</td></tr>
<tr><td>1003</td><td>Not used</td><td>Not used</td></tr>
<tr><td>1004</td><td rowspan="2">Length of the standard buffer (64 K-bytes when 0000$_{(H)}$)</td><td rowspan="2">Not used</td></tr>
<tr><td>1005</td></tr>
<tr><td>1006</td><td>Not used</td><td>Not used</td></tr>
<tr><td>1007</td><td colspan="2">Selection of the standard buffer
00$_{(H)}$: Deactivate the standard buffer
80$_{(H)}$: Direct assignment of the standard buffer
C0$_{(H)}$: Indirect assignment of the standard buffer</td></tr>
</table>
</td></tr>
<tr><td>1010 to 1017</td><td>Information concerning standard buffer 01</td><td rowspan="31">Set the same as the information about standard buffer 00</td></tr>
<tr><td>1020 to 1027</td><td>Information concerning standard buffer 02</td></tr>
<tr><td>1030 to 1037</td><td>Information concerning standard buffer 03</td></tr>
<tr><td>1040 to 1047</td><td>Information concerning standard buffer 04</td></tr>
<tr><td>1050 to 1057</td><td>Information concerning standard buffer 05</td></tr>
<tr><td>1060 to 1067</td><td>Information concerning standard buffer 06</td></tr>
<tr><td>1070 to 1077</td><td>Information concerning standard buffer 07</td></tr>
<tr><td>1100 to 1107</td><td>Information concerning standard buffer 08</td></tr>
<tr><td>1110 to 1117</td><td>Information concerning standard buffer 09</td></tr>
<tr><td>1120 to 1127</td><td>Information concerning standard buffer 0A</td></tr>
<tr><td>1130 to 1137</td><td>Information concerning standard buffer 0B</td></tr>
<tr><td>1140 to 1147</td><td>Information concerning standard buffer 0C</td></tr>
<tr><td>1150 to 1157</td><td>Information concerning standard buffer 0D</td></tr>
<tr><td>1160 to 1167</td><td>Information concerning standard buffer 0E</td></tr>
<tr><td>1170 to 1177</td><td>Information concerning standard buffer 0F</td></tr>
<tr><td>1200 to 1207</td><td>Information concerning standard buffer 10</td></tr>
<tr><td>1210 to 1217</td><td>Information concerning standard buffer 11</td></tr>
<tr><td>1220 to 1227</td><td>Information concerning standard buffer 12</td></tr>
<tr><td>1230 to 1237</td><td>Information concerning standard buffer 13</td></tr>
<tr><td>1240 to 1247</td><td>Information concerning standard buffer 14</td></tr>
<tr><td>1250 to 1257</td><td>Information concerning standard buffer 15</td></tr>
<tr><td>1260 to 1267</td><td>Information concerning standard buffer 16</td></tr>
<tr><td>1270 to 1277</td><td>Information concerning standard buffer 17</td></tr>
<tr><td>1300 to 1307</td><td>Information concerning standard buffer 18</td></tr>
<tr><td>1310 to 1317</td><td>Information concerning standard buffer 19</td></tr>
<tr><td>1320 to 1327</td><td>Information concerning standard buffer 1A</td></tr>
<tr><td>1330 to 1337</td><td>Information concerning standard buffer 1B</td></tr>
<tr><td>1340 to 1347</td><td>Information concerning standard buffer 1C</td></tr>
<tr><td>1350 to 1357</td><td>Information concerning standard buffer 1D</td></tr>
<tr><td>1360 to 1367</td><td>Information concerning standard buffer 1E</td></tr>
<tr><td>1370 to 1377</td><td>Information concerning standard buffer 1F</td></tr>
</table>

[3] Standard buffer information storage area

When the indirect assignment method is used, set the top file address, file number, and buffer length for the standard buffer in the standard buffer's information storage area.

+0	Top file address of the standard buffer (DA)
+1	
+2	File number of the standard buffer (DF)
+3	Not used
+4	Standard buffer length (DL) Entering 0000$_{(8)}$ creates buffer 64 k-bytes long.
+5	
+6	Not used
+7	

[4] Error processing when accessing standard buffers

The JW-255CM/25TCM, and Z-339J performs the following operation when an error occurs when dealing with a standard buffer.

(1) When setting parameters

If any of the parameters is not set correctly, this module will indicate a parameter error (the ERROR lamp will light) when writing the parameters to the EEPROM.

If this happens, this module will not write the parameters into the EEPROM and will keep the start switch value address 3777 unchanged 81$_{(H)}$.

(2) When communicating

If an error occurs during communication, this module will return a response with an error code attached as follows:

Error code (Hexadecimal)	Details	Meaning
01	Format error	The standard buffer number is not correct (other than 0 to F). The specified number of data bytes to read (write) exceeded the maximum 1024 bytes allowed.
48	Undefined standard buffer	The standard buffer corresponding to the buffer number could not be found.
49	Incorrectly identified a standard buffer	The area for the corresponding buffer has not been assigned correctly. This was determined when this module received a command using the indirect assignment method.
4A	Incorrect data length	The number of bytes to read or write exceeded the buffer length of the specified buffer.

[5] Description of commands used with standard buffers

Read from a standard buffer (COM = 28$_{(H)}$)

[Format]

■ **Command**

COM	DB	TAG	IP_L	IP_H	L_L	L_H

■ **Response**

COM	RSLT	DB	TAG	IP_L	IP_H	L_L	L_H	$DATA_1$	···	$DATA_N$

COM = 28$_{(H)}$
DB = Standard buffer number (00$_{(H)}$ to 1F$_{(H)}$)
TAG = 01$_{(H)}$
$IP_{L,\,H}$ = Offset address (Enter an offset value from the top of the buffer for the place to start reading data to read)
$L_{L,\,H}$ = Data length (the number of bytes to read). Any value up to 1024 bytes.
$DATA_{1 \text{ to } N}$ = Read data

[Function]

- Read L bytes of data starting from offset address IP in the standard buffer DB. If 00$_{(H)}$ is entered for IP, this module will read from the top of the standard buffer.
- Up to 1024 bytes can be read at a time.

[Execution condition]

- Write enable mode: Mode 0, mode 1, and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error 01$_{(H)}$ (format error).
  1. The DB or TAG value is not correct.
  2. The IP or L exceeds 1024.
  3. The command length is not correct.
- If the ring buffer is not defined, the module will return error 48$_{(H)}$ (undefined standard buffer).
- While the module receives a command by indirect assignment, if the target standard buffer area is not correctly assigned (except for the storage area shown in page 8-24), the module will return error 49$_{(H)}$ (incorrectly identified standard buffer).
- If the final data address to read (the data length is L bytes from the IP address) exceeds the last address in the standard buffer area, the module will return error 4A$_{(H)}$ (data length incorrect).
- If a timeout occurs while the module is accessing data using the control module (CPU board), the module will return error 0F$_{(H)}$ (timeout while accessing memory).

[Example]

- Read 4 bytes of data starting from address 0000$_{(H)}$ in standard buffer 01.

■ **Command**

28	01	01	00	00	04	00

■ **Response**

28	00	01	01	00	00	04	00	11	22	44	88
								0000	0001	0002	0003

Write to a standard buffer (COM = 38(H))

[Format]

■ **Command**

COM	DB	TAG	IPL	IPH	LL	LH	DATA1	···	DATAN

■ **Response**

COM	RSLT	DB	TAG	IPL	IPH	LL	LH

COM	= 38(H)
DB	= Standard buffer number (00(H) to 1F(H))
TAG	= 01(H)
IPL, H	= Offset address (Enter an offset value from the top of the buffer for the place to start writing the data)
LL, H	= Data length (the number of bytes to write). Any value up to 1024 bytes.
DATA1 to N	= Write data

[Function]

- Write L byte of data starting from the offset location specified in IP, the standard buffer named in DB. If 00(H) is entered for IP, this module will start reading from the top of the buffer.
- Up to 1024 bytes can be read at a time.

[Execution conditions]

- Write enable mode: Mode 0, mode 1, and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error 01(H) (format error).
  1. The DB or TAG value is not correct.
  2. The IP or L exceeds 1024.
  3. The command length is not correct.
- If the standard buffer is not defined, the module will return error 48(H) (undefined standard buffer).
- While the module receives a command by indirect assignment, if the target standard buffer area is not correctly assigned (except for the storage area shown in page 8-24), the module will return error 49(H) (incorrectly identified standard buffer).
- If the final data address to write (the data length is L bytes from the IP address) exceeds the last address in the standard buffer area, the module will return error 4A(H) (data length incorrect).
- If the write enable mode is set to 0, the module will return error 10(H) (mis-matched write enable mode).
- When the module detects a verification error in the written data, it will return error 07(H) (write command verification error).
- If a timeout occurs while the module is accessing data using the control module (CPU board), the module will return error 0F(H) (timeout while accessing memory)

[Example]

- Write 12(H), 34(H), 56(H) and 78(H) to 4 bytes address starting from address 0000(H) in standard buffer 02.

■ **Command**

38	02	01	00	00	04	00	12	34	56	78

■ **Response**

38	00	02	01	00	00	04	00

Read information about a standard buffer (COM = $68_{(H)}$)

[Format]

■ **Command**

COM	DB

■ **Response**

COM	RSLT	DB	TAG	DINF	ISEG	IADR_L	IADR_H	BSEG	BADR_L	BADR_H

LB_L	LB_H

COM = $68_{(H)}$
DB = Standard buffer number ($00_{(H)}$ to $1F_{(H)}$)
TAG = $01_{(H)}$
DINF = Setting for the standard buffer
$00_{(H)}$: Not defined
$01_{(H)}$: Direct assignment
$02_{(H)}$: Indirect assignment
$81_{(H)}$: Indicates invalid setting for direct assignment
$82_{(H)}$: Indicates invalid setting for indirect assignment
If the standard buffer has not been defined, $00_{(H)}$ will returned for the parameters below.
ISEG = File number of the standard buffer's information storage area
$\text{IADR}_{L,\,H}$ = Top address of the standard buffer's information storage area
When the direct assignment method is used, $0000_{(H)}$ is stored in ISEG and IADR.
BSEG = File number of the standard buffer
$\text{BADR}_{L,\,H}$ = Top address of the standard buffer
$\text{LB}_{L,\,H}$ = Buffer size (number of bytes). $0000_{(H)}$ corresponds to 64 K bytes
Whether or not these settings are correct, this module will read them. If they are incorrect, DINF will be $81_{(H)}$ (direct assignment) or $82_{(H)}$ (indirect assignment).

[Function]

- Read the information about the standard buffer named in DB.

[Execution conditions]

- Read enable mode: Mode 0, mode 1, and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error $01_{(H)}$ (format error).
  1. The RB or TAG value is not correct.
  2. The command length is not correct.
- If a timeout occurs while the module is accessing data using the control module (CPU board), the module will return error $0F_{(H)}$ (timeout while accessing memory).

[Example]

- Read information from standard buffer 02.

■ **Command**

68	02

■ **Response**

68	00	02	01	02	01	00	00	02	00	00

File 1, from $0000_{(H)}$ File 2, from $0000_{(H)}$

Indirect assignment

00	01

256 bytes

Write information about a standard buffer (COM = 78$_{(H)}$)

[Format]

■ **Command**

COM	DB	TAG	DINF	BSEG	BADR$_L$	BADR$_H$	LB$_L$	LB$_H$

■ **Response**

COM	RSLT	DB

COM = 78$_{(H)}$
DB = Standard buffer number (00$_{(H)}$ to 1F$_{(H)}$)
TAG = 01$_{(H)}$
DINF = Select the assignment method for the buffer being defined
02$_{(H)}$: Indirect assignment
BSEG = File number of the standard buffer segment
BADR$_{L, H}$ = Top address of the standard buffer segment
LB$_{L, H}$ = Buffer size (number of bytes). 0000$_{(H)}$ corresponds to 64 K-bytes

[Function]

- Write the information about the standard buffer named in DB.
- The information details consist of the buffer file number, buffer top address, and buffer size.
- The information about a direct assignment buffer cannot be changed. Changing between direct and indirect methods of accessing a given buffer is not possible.

[Execution conditions]

- Read enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error 01$_{(H)}$ (format error).
  1. The DB, TAG, DINF, BSEG, BADR, or LB value is not correct.
  2. The command length is not correct.
- If the buffer is not defined, or the buffer assignment method is different (for example, DINF=02$_{(H)}$ is assigned to the direct buffer), the module will return error 48$_{(H)}$ (undefined standard buffer).
- If the write enable mode is set to 0, the module will return error 10$_{(H)}$ (mis-matched write enable mode).
- When the module detects a verification error in the written data, it will return error 07$_{(H)}$ (write command verification error).
- If a timeout occurs while the module is accessing data using the control module (CPU board), the module will return error 0F$_{(H)}$ (timeout while accessing memory).

[Example]

- Set the beginning of standard buffer 03 to address starting from 0000$_{(H)}$ in file 2 and make it 256 bytes long.

■ **Command**

78	03	01	02	02	00	00	00	01
				File 2, from 0000$_{(H)}$			256 bytes	

■ **Response**

78	00	03

8-4 Ring buffer

A ring buffer is used to transmit data between the host computer and PLC in a single direction. An assigned area inside the PLC is used as a ring buffer, and data is written into and then read from this area.

■ **Commands for the ring buffer**

Command code	Details	Reference page
29$_{(H)}$	Read from ring buffer	8-41
39$_{(H)}$	Write to ring buffer	8-43
69$_{(H)}$	Read the ring buffer information	8-45
79$_{(H)}$	Write the ring buffer information	8-47

[1] How to use the ring buffer

Create the ring buffer in the PLC's data memory (as a register or a file register).
The buffer size can be set to 256, 512, 1 K, 2 K, 4 K, 8 K, 16 K, 32 K, or 64 K bytes. A maximum of 16 ring buffers can be created. These ring buffers are identified by a ring buffer number (0 to F).
To create a ring buffer, enter the top address (BAH), the ring buffer file number (BF), and the buffer length (BL). In order to access data, a write pointer (WP), and a read pointer (RP) are used. Both the write and read pointer are kept in the ring buffer's information storage area of data memory. Set the top address for the ring buffer's information storage area as a parameter. The data memory used for the ring buffer's information storage areas is as follows.

JW20H, J-board (Z300)		JW30H *1		J-board (Z500) *2	
File number	File addresses	File number	File addresses	File number	File addresses
File 0	000000 to 017777$_{(8)}$	File 0	000000 to 035777$_{(8)}$	File 0	000000 to 035777$_{(8)}$
		File 1	000000 to 037777$_{(8)}$	File 1	000000 to 037777$_{(8)}$
		File 2 to 3, 10 to 2C$_{(H)}$	000000 to 177777$_{(8)}$	File 2	000000 to 177777$_{(8)}$

*1 The JW30H memory capacity varies, depending on the control module type.
*2 The Z500 memory capacity varies, depending on the CPU board type.
(Shown above are maximum values.)

(1) Write pointer (WP)
Address to write the next byte of data to (offset from buffer top, which is 0)

(2) Read pointer (RP)
Address to read the next byte of data from (offset from the buffer top, which is 0)

(3) Buffer address (BAH)
Upper bytes of the ring buffer top address (file address). Buffers can be created in units of 1 K bytes. Therefore, the possible buffer sizes and address are as follows.

Set value (Hexadecimal)	Actual file address (Octal)
00	000000
04	002000
08	004000
0C	006000
:	:
F8	174000
FC	176000

(4) Buffer file number (BF)

Assign a file number to the ring buffer.
The data memory area used for the ring buffers is as follows.

JW20H, J-board (Z300)		JW30H *1		J-board (Z500) *2	
File number	**File addresses**	**File number**	**File addresses**	**File number**	**File addresses**
File 0	000000 to 017777$_{(8)}$	File 0	000000 to 035777$_{(8)}$	File 0	000000 to 035777$_{(8)}$
		File 1	000000 to 037777$_{(8)}$	File 1	000000 to 037777$_{(8)}$
		File 2 to 3, 10 to 2C$_{(H)}$	000000 to 177777$_{(8)}$	File 2	000000 to 177777$_{(8)}$

*1 The JW30H memory capacity varies, depending on the control module type.
*2 The Z500 memory capacity varies, depending on the CPU board type.
(Shown above are maximum values.)

(5) Direction (DIR)

Assign a data transmission direction

01$_{(H)}$: Reading data from a control module to this module
Use the ring buffer read command

81$_{(H)}$: Write data from this module to a control module
Use the ring buffer write command

(6) Buffer length (BL)

Set the size of the ring buffer

Set value (Hexadecimal)	Buffer length
00	64 K-bytes
01	256 bytes
02	512 bytes
04	1 K-bytes
08	2 K-bytes
10	4 K-bytes
20	8 K-bytes
40	16 K-bytes
80	32 K-bytes

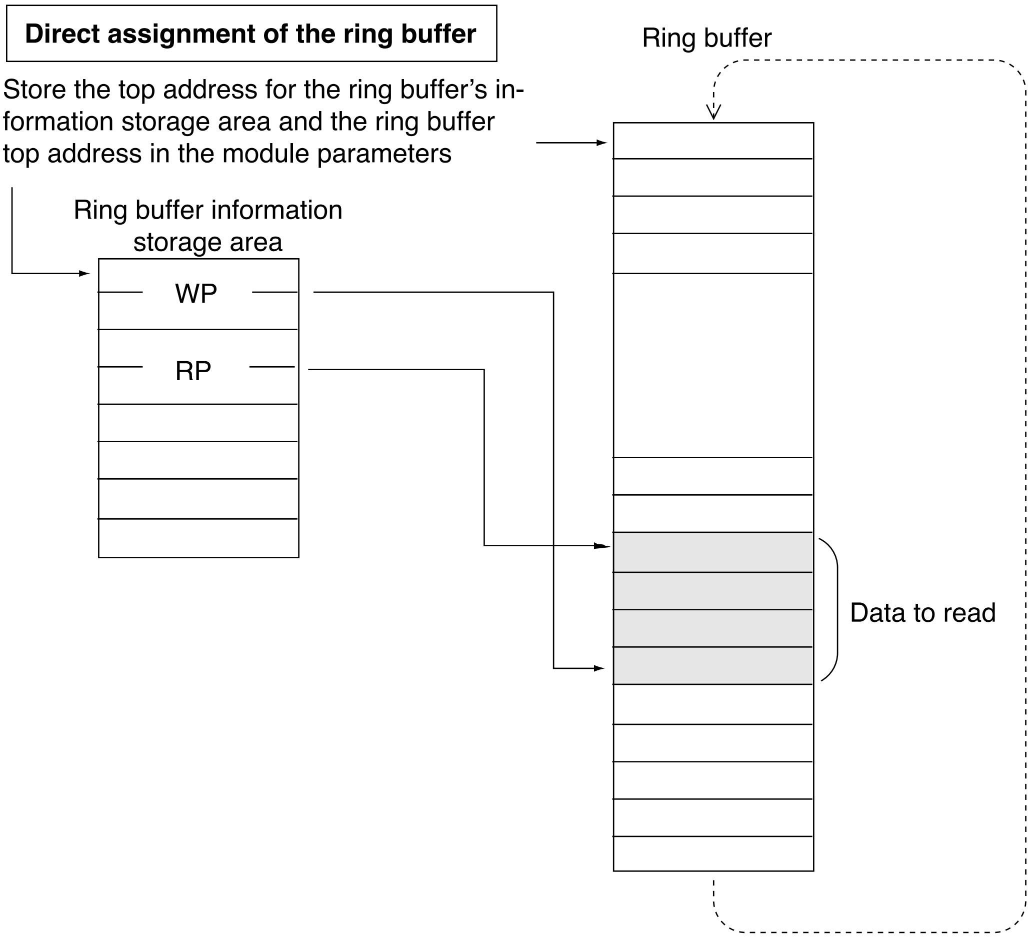
To assign the top address, buffer file number, and buffer length for the ring buffer, either direct or indirect assignment methods can be used.

1) Direct assignment

A method to assign the buffer top file address, file number and number of bytes directly in the module parameters.

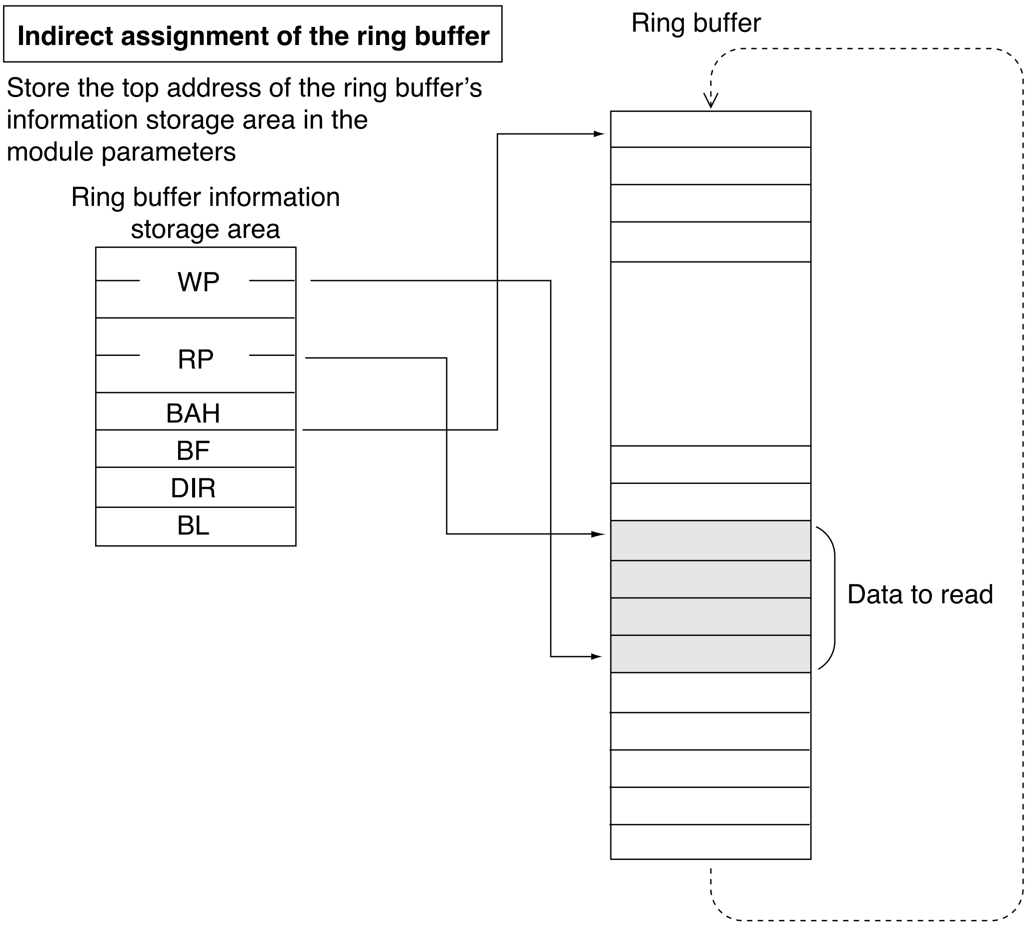
2) Indirect assignment

A method to assign the buffer top file address, file number, buffer length and number of bytes in the ring buffer information storage area.



Direct assignment of the ring buffer

Store the top address for the ring buffer's information storage area and the ring buffer top address in the module parameters



Indirect assignment of the ring buffer

Store the top address of the ring buffer's information storage area in the module parameters

[2] Operation of the ring buffer

Use the ring buffer in order to transfer date in one direction between a host computer and the PLC. Both the read pointer (RP) and write pointer (WP) start out pointing at the buffer top. To initialize these settings, use a ladder program.

(1) Data transmission procedure when the read direction is from the PLC to the host computer

1) Processing the data with the ladder program

If there is data to transmit, the module will write the data starting at the current WP position. It increases the WP by the number of bytes of data it writes into the buffer. If the WP reaches the end of the buffer, it will be pointed back to the top (0). Be careful not to advance the WP if it will be pointing at the same position as the RP. => Refer to page 8-49 to 51.

2) Processing by the module

If the RP matches with the WP, the module assumes that there is no data to read. If the RP does not match with the WP data will be read from the area between the RP and the WP-1.
To read the data, use the ring buffer read command (command code $29_{(H)}$). Set the ring buffer number and number of data bytes to read using this command. When the module receives the command, it will read data starting from the RP. Then, it advances the RP by the number of data bytes to read. Pointer updating can be done ether of two ways as described below.

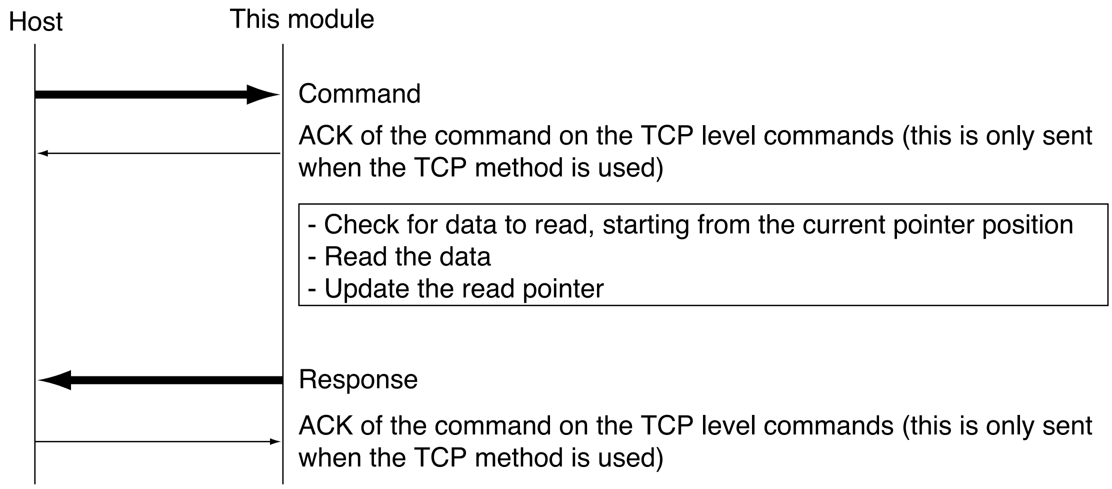
1) Non-confirmation type

Update the pointer at the same time the data is read. Then, the module will send its response.

2) Confirmation type

The pointer is not updated when the data is read. The module waits for confirmation from the host after sending its response. When the module receives the confirmation, it will update the pointer. Therefore, the host computer has to send a confirmation after receiving a response.
To choose whether to use the non-confirmation or confirmation method of communication, include the method in the command. To read data continuously using the confirmation method, you can send a data read request command for the next byte of data together with the confirmation for the current byte of data. => Refer to page 8-37.

Non-confirmation type



Host

This module

Command

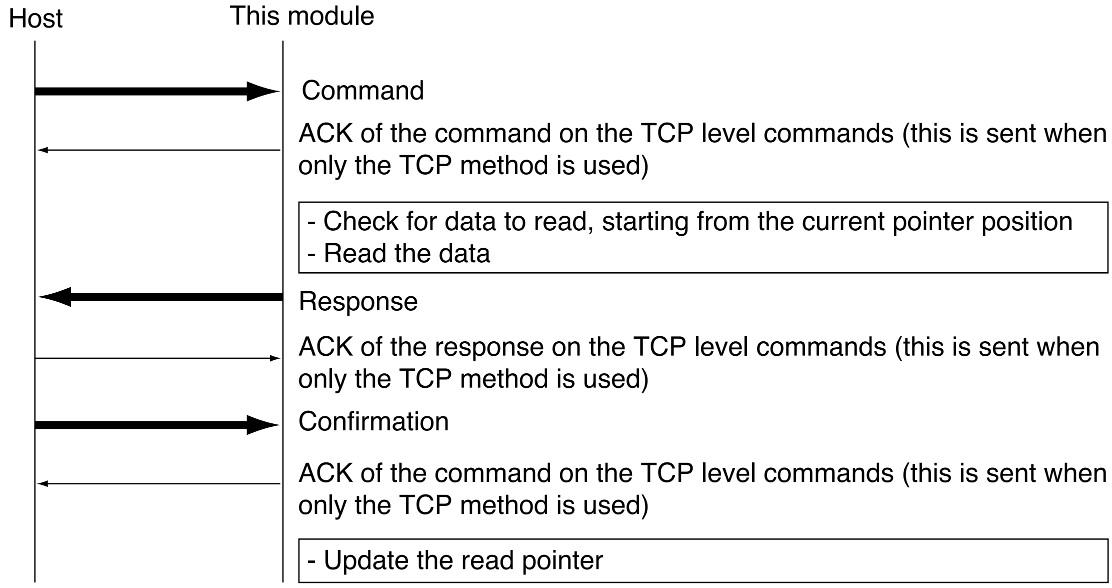
ACK of the command on the TCP level commands (this is only sent when the TCP method is used)

- Check for data to read, starting from the current pointer position
- Read the data
- Update the read pointer

Response

ACK of the command on the TCP level commands (this is only sent when the TCP method is used)

Confirmation type



Host

This module

Command

ACK of the command on the TCP level commands (this is sent when only the TCP method is used)

- Check for data to read, starting from the current pointer position
- Read the data

Response

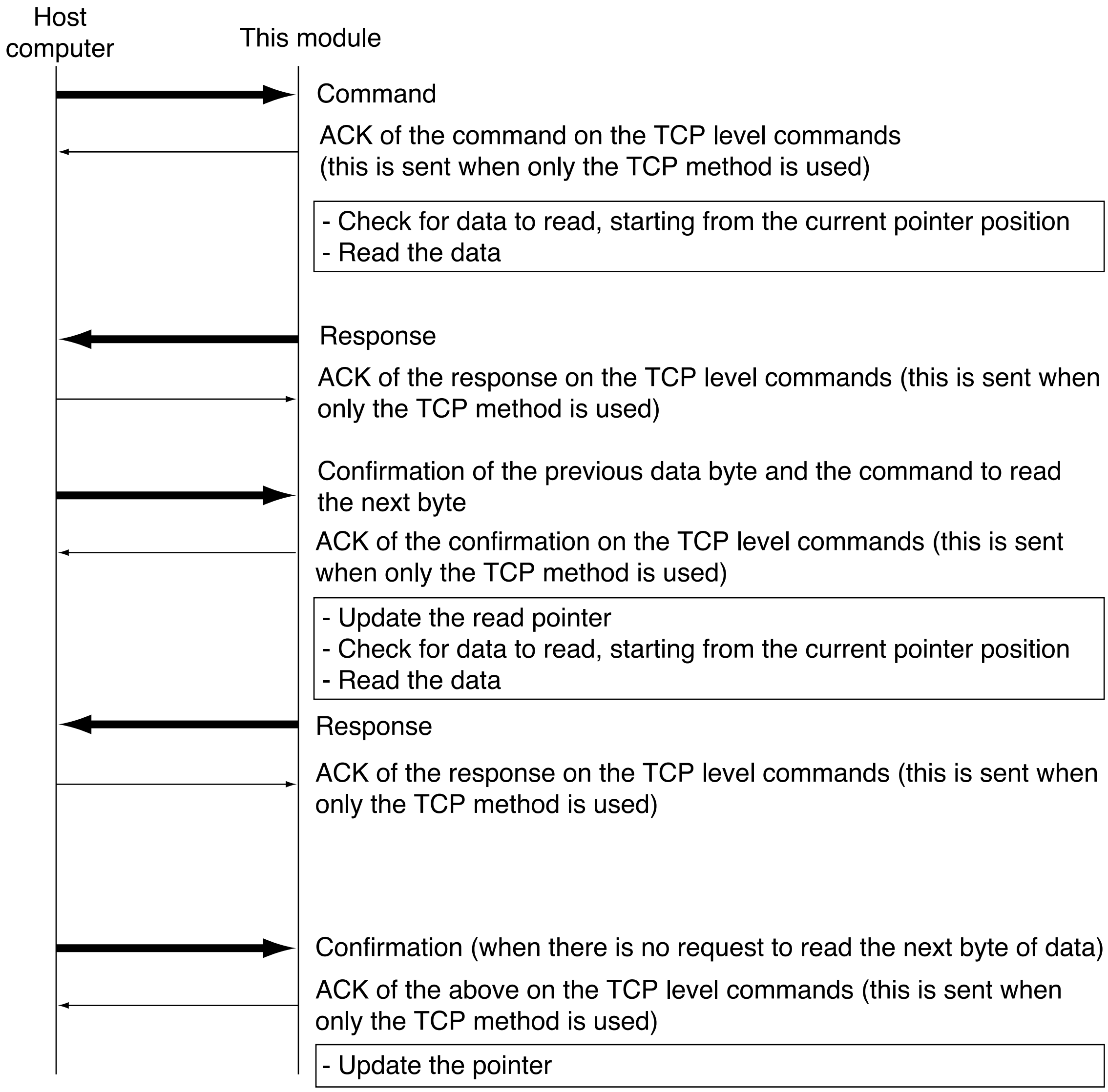
ACK of the response on the TCP level commands (this is sent when only the TCP method is used)

Confirmation

ACK of the command on the TCP level commands (this is sent when only the TCP method is used)

- Update the read pointer

Confirmation type (when you want to attach a read command for the next byte of data to the confirmation for the current byte of data)


Host computer
This module
Command
ACK of the command on the TCP level commands (this is sent when only the TCP method is used)
- Check for data to read, starting from the current pointer position
- Read the data
Response
ACK of the response on the TCP level commands (this is sent when only the TCP method is used)
Confirmation of the previous data byte and the command to read the next byte
ACK of the confirmation on the TCP level commands (this is sent when only the TCP method is used)
- Update the read pointer
- Check for data to read, starting from the current pointer position
- Read the data
Response
ACK of the response on the TCP level commands (this is sent when only the TCP method is used)
Confirmation (when there is no request to read the next byte of data)
ACK of the above on the TCP level commands (this is sent when only the TCP method is used)
- Update the pointer

The non-confirmation type completes the communication process in one scan. However, if the host computer goes down while the module is returning a response after processing the received command, the read pointer will be updated. But, the host computer will not receive the response since it is down. Then, when the host computer resends this command after coming back up, the module will have already updated the read pointer. Therefore, the data read at the time the host computer system goes down will actually be lost.
The confirmation type only updates the pointer after a read when the module receives the confirmation of its response to the host computer. Therefore, if the host computer fails to receive the response due to the system going down, the pointer will not be updated. This decreases the possibility that data will be lost during a crash and recovery. However, the confirmation type needs 1.5 times the number of communication cycles, compared with the non-confirmation type.

When the number of data bytes to read is set to 0 in the command, the module will read the number of data bytes available. However, it can only read a maximum of 1024 bytes at one time.

The JW-255CM/25TCM, and Z-339J will return the number of bytes actually read, the “continuation information” (whether there is still data left to read in the buffer or not), and the data to be read as a response. The relationship between the number of data bytes to read, specified in the command, and the number of data bytes actually read is shown below.

LC: Number of data bytes to read, specified in the command
LP: Number of data bytes available to read
LR: Number of data bytes actually read

Number of data bytes to read, specified in the command (LC)			Number of data bytes available to read (LP)	Continuous information (MORE)	Number of data bytes actually read (LR)
(1)	LC = 0	a	LP = 0	Do not continue	LR = 0
		b	1 ≦ LP ≦1024	Do not continue	LR = LP
		c	LP ＞1024	Continue	LR = 1024
(2)	1≦LC≦1024	d	LP = 0	Do not continue	LR = 0
		e	LP≦LC	Do not continue	LR = LP
		f	LC＜LP	Continue	LR = LC

1) When the number of data bytes to read is not specified in the command (set to 0)

1) When there is no data to read, the number of data bytes available will be 0, and the module will not continue trying to read.
2) When the data available to read is less than 1024 bytes, the module will read all of the data available and then stop reading.
3) When the amount of data to read exceeds 1024 bytes, the module will read the first 1024 bytes of data and get ready to read another string of data.

2) When the number of data bytes to read is specified in the command

4) When there is no data to read, the number of data bytes available will be 0, and the module will not continue trying to read.
5) When the number of data bytes available is less than the number of data bytes specified in the command, the module will read out all of the data in the buffer, regardless of the number of bytes specified, and then it will stop reading.
6) When the number of data bytes specified in the command is smaller than the number of data bytes available to read, the module will read the specified number of data bytes, and then get ready to read the next string data.

8

(2) Data transmission procedure when the writing direction is from the host computer to the PLC

1) Processing by the module

The module uses the ring buffer write command to write data in to the buffer (command code $39_{(H)}$). Specify a ring buffer number, the number of data bytes being written, and the data to write. When the module receives this command, it will writes the data starting at the current WP position, and advances the WP by the number of data bytes it transmitted. When the WP reaches the end of the buffer, it will be set to point back to the top (0). If the WP matches the RP (or goes past the RP) by illegally advancing the WP, the module will enter a buffer full error condition.

The non-confirmation and confirmation communication types are also available when writing data.

2) Processing by the ladder program

When the RP matches the WP, the module assumes that there is no data to read. If the RP does not match with the WP, the area from the RP to a position one less than the WP will be seen as data to read. Temporarily move the data to read to some other type of memory (not data memory) and then advance the RP by the number of data bytes available to read => Refer to page 8-52 to 53.

[3] Parameter setting

Use parameters 1400 to 1577 to create the ring buffer.

<table>
<tr><th>Parameter address</th><th colspan="2">Details</th></tr>
<tr><td>1400 to 1407</td><td colspan="2">Information concerning ring buffer 00
<table>
<tr><td>1400
1401</td><td>Top file address of the ring buffer's information storage area</td></tr>
<tr><td>1402</td><td>File number of the ring buffer's information storage area</td></tr>
</table>
Enter values for 1403 to 1406 (1407 = 80(H)) when direct assignment is used.
<table>
<tr><td rowspan="4">1403</td><td colspan="4">Ring buffer data transmission direction</td></tr>
<tr><td>Set value(H)</td><td colspan="3">Details</td></tr>
<tr><td>01</td><td colspan="2">Reads data from CU to the module</td><td rowspan="2">CU =
control module
(CPU board)</td></tr>
<tr><td>81</td><td colspan="2">Writes data from the module to CU</td></tr>
<tr><td rowspan="6">1404</td><td colspan="4">Ring buffer top address (upper bytes of the file address)
Settable in units of 1 K bytes</td></tr>
<tr><td>Set value(H)</td><td>File address(8)</td><td>Set value(H)</td><td>File address(8)</td></tr>
<tr><td>00</td><td>000000</td><td>:</td><td>:</td></tr>
<tr><td>04</td><td>002000</td><td>F4</td><td>172000</td></tr>
<tr><td>08</td><td>004000</td><td>F8</td><td>174000</td></tr>
<tr><td>0C</td><td>006000</td><td>FC</td><td>176000</td></tr>
<tr><td>1405</td><td colspan="4">Ring buffer file number</td></tr>
<tr><td rowspan="7">1406</td><td colspan="4">Upper bytes of the ring buffer length</td></tr>
<tr><td>Set value(H)</td><td>Buffer length</td><td>Set value(H)</td><td>Buffer length</td></tr>
<tr><td>00</td><td>64 K bytes</td><td>10</td><td>4 K bytes</td></tr>
<tr><td>01</td><td>256 bytes</td><td>20</td><td>8 K bytes</td></tr>
<tr><td>02</td><td>512 bytes</td><td>40</td><td>16 K bytes</td></tr>
<tr><td>04</td><td>1 K bytes</td><td>80</td><td>32 K bytes</td></tr>
<tr><td>08</td><td>2 K bytes</td><td></td><td></td></tr>
</table>
<table>
<tr><td>1407</td><td>Selection of the ring buffer
00(H): Disable the ring buffer
80(H): Direct assignment of the ring buffer
C0(H): Indirect assignment of the ring buffer</td></tr>
</table>
</td></tr>
<tr><td>1410 to 1417</td><td>Information concerning ring buffer 01</td><td rowspan="15">Set the same as the information for ring buffer 00</td></tr>
<tr><td>1420 to 1427</td><td>Information concerning ring buffer 02</td></tr>
<tr><td>1430 to 1437</td><td>Information concerning ring buffer 03</td></tr>
<tr><td>1440 to 1447</td><td>Information concerning ring buffer 04</td></tr>
<tr><td>1450 to 1457</td><td>Information concerning ring buffer 05</td></tr>
<tr><td>1460 to 1467</td><td>Information concerning ring buffer 06</td></tr>
<tr><td>1470 to 1477</td><td>Information concerning ring buffer 07</td></tr>
<tr><td>1500 to 1507</td><td>Information concerning ring buffer 08</td></tr>
<tr><td>1510 to 1517</td><td>Information concerning ring buffer 09</td></tr>
<tr><td>1520 to 1527</td><td>Information concerning ring buffer 0A</td></tr>
<tr><td>1530 to 1537</td><td>Information concerning ring buffer 0B</td></tr>
<tr><td>1540 to 1547</td><td>Information concerning ring buffer 0C</td></tr>
<tr><td>1550 to 1557</td><td>Information concerning ring buffer 0D</td></tr>
<tr><td>1560 to 1567</td><td>Information concerning ring buffer 0E</td></tr>
<tr><td>1570 to 1577</td><td>Information concerning ring buffer 0F</td></tr>
</table>

[4] Ring buffer information storage area (in data memory)

+0	Write pointer (WP)	
+1		
+2	Read pointer (RP)	
+3		
+4	Upper bytes of the buffer top address (BAH)	Only when the indirect assignment is used
+5	Buffer file number (BF)	
+6	Data transmission direction (DIR)	
+7	Upper bytes of the buffer length (BL)	

The write pointer and read pointer are used to access the ring buffer. They are incremental addresses, using the top address of the ring buffer location 0. Upper bytes of the buffer top address (BAH), buffer file number (BF), data transmission direction (DIR), and upper bytes of buffer length (BL) are only assigned when indirect assignment is used. The setting details are the same as for the setting values used as parameters when direct assignment is used.

[5] Error processing when accessing ring buffers

This module performs the following operation when an error occurs while handling a ring buffer.

(1) When setting parameters

If any of the parameters is not set correctly, this module will indicate a parameter error (the ERROR lamp will light) when writing the parameters into the EEPROM.

If this happens, this module will not write the parameters into the EERPOM and will keep the start switch value at address 3777 unchanged $81_{(H)}$.

(2) When communicating

If an error occurs during communication, this module will return a response with an error code attached as follows:

Error code (Hexadecimal)	Details	Meaning
01	Format error	The specified buffer number was not correct (other than 0 to F). The specified data length to read (write) exceeded the maximum 1024 bytes allowed.
40	Undefined ring buffer	The ring buffer corresponding to the buffer number could not be found.
41	Incorrectly identified a ring buffer	The area for the corresponding buffer has not been assigned correctly. This was determined when this module received a command using the indirect assignment method.
42	Incorrect pointer	The current read and/or writes pointer values are outside of the buffer range. This was determined when this module received a command.
43	No free area	The data you want to write will exceed the free space in the buffer when the data is actually written. (When the write pointer is advanced the actual number of bytes to be written, it will equal or exceed the read pointer position.)
44	Buffer overflow	The data you want to write is larger than the buffer can hold.

[6] Description of commands used with ring buffers

Read a ring buffer (COM = 29$_{(H)}$)

[Format]

■ **Command**

COM	RB	FUN	TAG	LC_L	LC_H

■ **Response**

COM	RSLT	RB	TAG	RP_L	RP_H	LR_L	LR_H	MORE

$DATA_1$	···	$DATA_N$

COM = 29$_{(H)}$
RB = Ring buffer number (00$_{(H)}$ to 0F$_{(H)}$)
FUN = Function switch D_0 = 1: With read instruction
D_1 = 1: Confirmation of the previous response
D_7 = 1: Confirmation type; 0: Non-confirmation type
With the settings above, values for FUN are shown as below:
01$_{(H)}$: Reading using the non-confirmation type
81$_{(H)}$: Reading using the confirmation type
82$_{(H)}$: Confirmation of the response
83$_{(H)}$: Confirmation of the response and value of next byte read
TAG = 01$_{(H)}$
$LC_{L, H}$ = Data length (the number of bytes to read)
0000$_{(H)}$ to 0400$_{(H)}$. If 0000$_{(H)}$ is entered, the module will read the number of data bytes currently stored in the buffer.
(Max. 1024 bytes)
$RP_{L, H}$ = Read pointer for the data to read (offset from the buffer top address)
$L_{L, H}$ = Data length (the number of bytes to read).
MORE = Continuation information
00$_{(H)}$: There is no more data to read.
01$_{(H)}$: There is data not yet read.
$DATA_{1 \text{ to } N}$ = Read data. The data length is given in LR.

[Function]

- Read LC bytes of data starting from the ring buffer specified by RB. If 00$_{(H)}$ is entered for LC, the module will read all of the unread data in the ring buffer up to 1024 bytes.
- The response will contain data length, LR, which expresses number of bytes actually read, and the continuation information MORE, which tells whether or not there is more data to read after executing the command.
- After executing this command, the read pointer in the ring buffer is advanced by LR. The timing for advancing the pointer depends on whether the non-confirmation and confirmation type was used.
  - Non-confirmation type: Pointer advanced after reading the data (before returning the response)
  - Confirmation type: Pointer advanced after receiving the confirmation of the response from the host computer.

- The number of data bytes actually read will vary considerably, based the number of data bytes left to read in the ring buffer and the number of bytes requested in the read command. The relationship of these factors is shown below.

LC: Number of data bytes to read, specified in the command
LP: Number of data bytes available to read
LR: Number of data bytes actually read

Number of data bytes to read, specified in the command (LC)	Number of data bytes available to read (LP)	Continuation information (MORE)	Number of data bytes actually read (LR)
LC = 0	LP = 0	00(H)	LR = 0
	1 ≦ LP≦1024	00(H)	LR = LP
	LP ＞1024	01(H)	LR = 1024
1□≦ LC ≦1024	LP = 0	00(H)	LR = 0
	LP≦LC	00(H)	LR = LP
	LC＜LP	01(H)	LR = LC

[Execution conditions]

- Write enable mode: Mode 0, mode 1, and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error 01(H) (format error).
  1. The RB, FUN, or TAG value is not correct.
  2. The LC exceeds 1024.
  3. The command length is not correct.
- If the ring buffer is not defined, the module will return error 40(H) (undefined ring buffer).
- When the module receives a command by indirect assignment, if the target ring buffer area is not correctly defined (except for the storage area shown in page 8-32), it will return error 41(H) (the area for the corresponding ring buffer has not been assigned correctly).
- When the module is receiving a command, if the current read and/or write pointer values are out of the buffer range, it will return error 42(H) (the current read and/or write pointer values are outside of the buffer range).
- If a timeout occurs while the module is accessing data using the control module (CPU board), the module will return error 0F(H) (timeout while accessing memory).

[Example]

- Read 4 bytes of data in ring buffer 01 using the confirmation type.

■ **Command**

29	01	81	01	04	00

■ **Response**

29	00	01	01	03	00	04	00	00
				RP		Data length		

12	34	56	78
0003	0004	0005	0006

Write to a ring buffer (COM = 39(H))

[Format]

■ **Command**

COM	RB	FUN	TAG	LC_L	LC_H	$DATA_1$	···	$DATA_N$

■ **Response**

COM	RSLT	RB	TAG	WP_L	WP_H	LR_L	LR_H	LE_L	LE_H

COM = 39(H)
RB = Ring buffer number (00(H) to 0F(H))
FUN = Function switch D_0 = 1: With write instruction
D_1 = 1: Confirmation of the previous response
D_7 = 1: Confirmation type; 0: Non-confirmation type
With the settings above, values for FUN are shown as below:
01(H): Write using the non-confirmation type
81(H): Write using the confirmation type
82(H): Confirmation of the response
83(H): Confirmation of the response and value of next byte write
TAG = 01(H)
$LC_{L, H}$ = Data length (the number of bytes to write). Up to 1024 bytes.
$DATA_{1 \text{ to } N}$ = Write data. The data length is given in LC.
$WP_{L, H}$ = Write pointer for the next location to write (offset from the ring buffer top address)
$L_{R, H}$ = Data length (the number of bytes to write).
$LE_{L, H}$ = Number of free bytes in the ring buffer after writing the data

[Function]

- Write LC bytes of data in the ring buffer specified in RB.
- Up to 1024 bytes of data can be written at one time using this function.

[Execution conditions]

- Write enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error 01(H) (format error).
  1. The RB, FUN, or TAG value is not correct.
  2. The LC exceeds 1024.
  3. The command length is not correct.
- If the ring buffer is not defined, the module will return error 40(H) (undefined ring buffer).
- When the module receives a command by indirect assignment, if the target ring buffer area is not correctly defined (except for the storage area shown in page 8-32), it will return error 41(H) (the area for the corresponding buffer has not been assigned correctly).
- When the module is receiving a command, if the current read and/or write pointer values are out of the buffer range, it will return error 42(H) (the current read and/or write pointer values are outside of the buffer range).
- When the data to be written exceeds the vacant area in the buffer, the module will return error 43(H) (no area to write).

- When the data to be written exceeds the buffer area, the module will return error $44_{(H)}$ (buffer overflow).
- If the write enable mode is set to 0, the module will return error $10_{(H)}$ (mis-matched write enable mode).
- When the module detects a verification error in the written data, it will return error $07_{(H)}$ (write command verification error).
- If a timeout occurs while the module is accessing data using the control module (CPU board), the module will return error $0F_{(H)}$ (timeout while accessing memory).

[Example]

- Write 4 bytes of data $01_{(H)}$, $02_{(H)}$, $03_{(H)}$, and $04_{(H)}$ in ring buffer 01 using the confirmation type.

■ Command

39	01	81	01	04	00	01	02	03	04

■ Response

39	00	01	01	10	00	04	00	20	00
				WP		Data length		Free area: 32 bytes	

Read information about a ring buffer (COM = 69$_{(H)}$)

[Format]

■ **Command**

COM	RB

■ **Response**

COM	RSLT	RB	TAG	DINF	ISEG	IADR$_L$	IADR$_H$	BSEG	BADR$_L$	BADR$_H$

WP$_L$	WP$_H$	RP$_L$	RP$_H$	DIR	LB$_L$	LB$_H$

COM = 69$_{(H)}$
RB = Ring buffer number (00$_{(H)}$ to 0F$_{(H)}$)
TAG = 01$_{(H)}$
DINF = Setting for the ring buffer
00$_{(H)}$: Not defined
01$_{(H)}$: Direct assignment
02$_{(H)}$: Indirect assignment
81$_{(H)}$: Indicates invalid setting for direct assignment
82$_{(H)}$: Indicates invalid setting for indirect assignment

If the ring buffer has not been defined, 00$_{(H)}$ will be returned for the parameters below.

ISEG = File number of the ring buffer's information storage area
IADR$_{L, H}$ = Top address of the ring buffer's information storage area
BSEG = File number of the ring buffer
BADR$_{L, H}$ = Top address of the ring buffer
WP$_{L, H}$ = Write pointer (offset from the top of the buffer)
RP$_{L, H}$ = Read pointer (offset from the top of the buffer)
DIR = Data transmission direction
80$_{(H)}$: Data flow direction (from CU to the module)
81$_{(H)}$: Data flow direction (from the module to CU)
[CU = Control module (CPU board)]
LB$_{L, H}$ = Buffer size (number of bytes). 0000$_{(H)}$ corresponds to 64 K-bytes.

Regardless of whether or not these settings are correct, the module will read them. If they are incorrect, DINF will be 81$_{(H)}$ (direct assignment) or 82$_{(H)}$ (indirect assignment).

[Function]

- Read the information about the ring buffer named in RB.

[Execution conditions]

- Read enable mode: Mode 0, mode 1, and mode 2
- PLC operation status: Stopping, operating

[Error handling]

- If the format does not match the format shown below, the module will return error 01$_{(H)}$ (format error).
  1. The RB value is not correct.
  2. The command length is not correct.
- If a timeout occurs while the module is accessing data using the control module, the module will return error 0F$_{(H)}$ (timeout while accessing memory).

[Example]

- Read information from ring buffer 01.

■ **Command**

69	01

■ **Response**

69	00	01	01	01	00	00	08	00	00	0C
				Direct assignment	Register 09000			Register 29000		

03	00	06	00	81	00	04
WP		RP		Writing data	1 K bytes	

Write information about a ring buffer (COM = 79$_{(H)}$)

[Format]

■ **Command**

COM	RB	TAG	DINF	BSEG	BADR$_L$	BADR$_H$	WP$_L$	WP$_H$	RP$_L$	RP$_H$

DIR	LB$_L$	LB$_H$

■ **Response**

COM	RSLT	RB

COM = 79$_{(H)}$
RB = Ring buffer number (00$_{(H)}$ to 0F$_{(H)}$)
TAG = 01$_{(H)}$
DINF = Select the assignment method for the buffer being defined
01$_{(H)}$: Direct assignment (When direct assignment is selected, the contents of BSEG and BADR will be ignored)
02$_{(H)}$: Indirect assignment
BSEG = File number of the ring buffer
BADR$_{L, H}$ = Top address of the ring buffer
RP$_{L, H}$ = Read pointer (offset from the top of the buffer)
WP$_{L, H}$ = Write pointer (offset from the top of the buffer)
DIR = Data transmission direction
80$_{(H)}$: Data flow direction (from CU to the module)
81$_{(H)}$: Data flow direction (from the module to CU)
[CU = Control module (CPU board)]
LB$_{L, H}$ = Buffer size (number of bytes). 0000$_{(H)}$ corresponds to 64 K-bytes.

[Function]

- Write the information about the ring buffer that is named in RB.
- The information details consist of the buffer file number, buffer top address, read pointer, write pointer, data transmission direction, and buffer size.
- When direct assignment is used for the ring buffer, this write command (79$_{(H)}$) cannot be used to change the buffer file number, buffer top address, data transmission direction, or buffer size, since they are set using parameters. (The values corresponding to these items in the command will be ignored.) Selection between direct and indirect assignment is also set using a parameter, and cannot be changed by this command.

[Execution conditions]

- Read enable mode: Mode 1 and mode 2
- PLC operation status: Stopping, operating

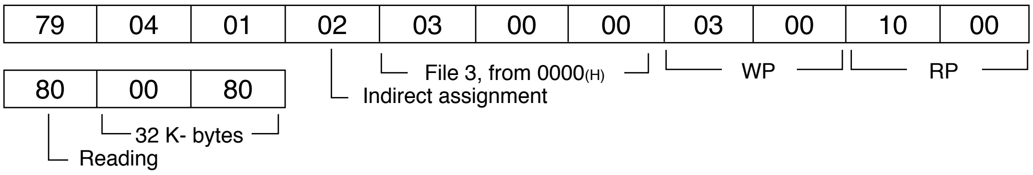
[Error handling]

- If the format does not match the format shown below, the module will return error $01_{(H)}$ (format error).
  1. The RB, TAG, DINF, BSEG, BADR, WP, RP, DIR, or LB value is not correct.
  2. The command length is not correct.
- If the buffer is not defined, or the buffer assignment method is different (for example, DINF=$02_{(H)}$ is assigned to the direct buffer), the module will return error $48_{(H)}$ (undefined standard buffer).
- If the write enable mode is set to 0, the module will return error $10_{(H)}$ (mis-matched write enable mode).
- When the module detects a verification error in the written data, it returns error $07_{(H)}$ (write command verification error).
- If a timeout occurs while the module is accessing data using the control module(CPU board), the module will return error $0F_{(H)}$ (timeout while accessing memory).

[Example]

- Set ring buffer 04 in file 3 to $0000_{(H)}$ and make it 32 K-bytes long. Set the write pointer to $0003_{(H)}$ and the read pointer to $0010_{(H)}$.

■ **Command**



79	04	01	02	03	00	00	03	00	10	00

80	00	80

■ **Response**

79	00	04

[7] An example using the ring buffer

(1) Data flow direction (from PLC to the host computer)

- Direct assignment
- Ring buffer 01
  Size and address: Occupy 1 K-bytes starting at 29000
- Ring buffer information storage area: Starts at 09000

■ **Parameter settings**

Parameter address	Set value	Details	
1410, 1411	$004000_{(8)}$	File 004000	09000
1412	00	File 0	
1413	$01_{(H)}$	Data flow direction	
1414	$0C_{(H)}$	File 006000	29000
1415	00	File 0	
1416	04	1 K-bytes	
1417	$80_{(H)}$	Direct assignment	

■ **Processing ladder program**

(1) Initialize the pointers

The read and write pointers should be initialized (set to 0) using a ladder program when starting the module.

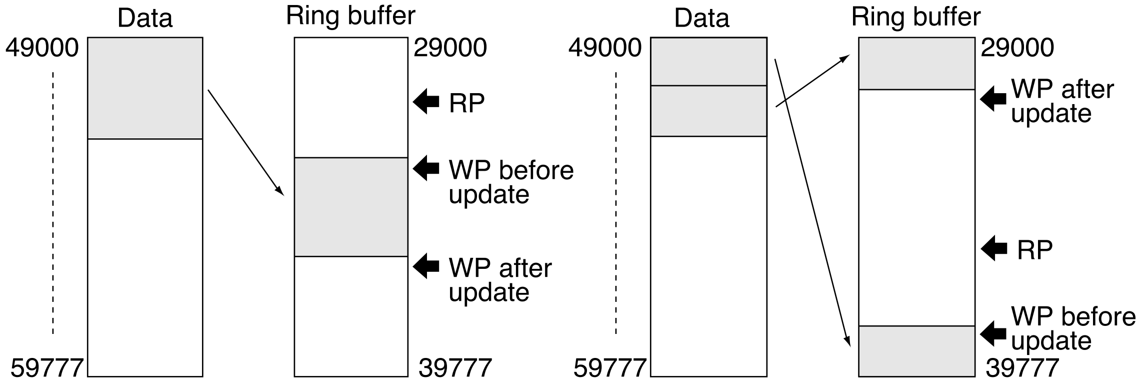
(2) Write data to a ring buffer

If there is free space in the ring buffer, write data in it.
In this example, we assume that the data is written starting at 49000, and set the number of bytes to write in locations 09100 and 09101.

(3) Update the WP

After transmitting the data, the module updates the WP.

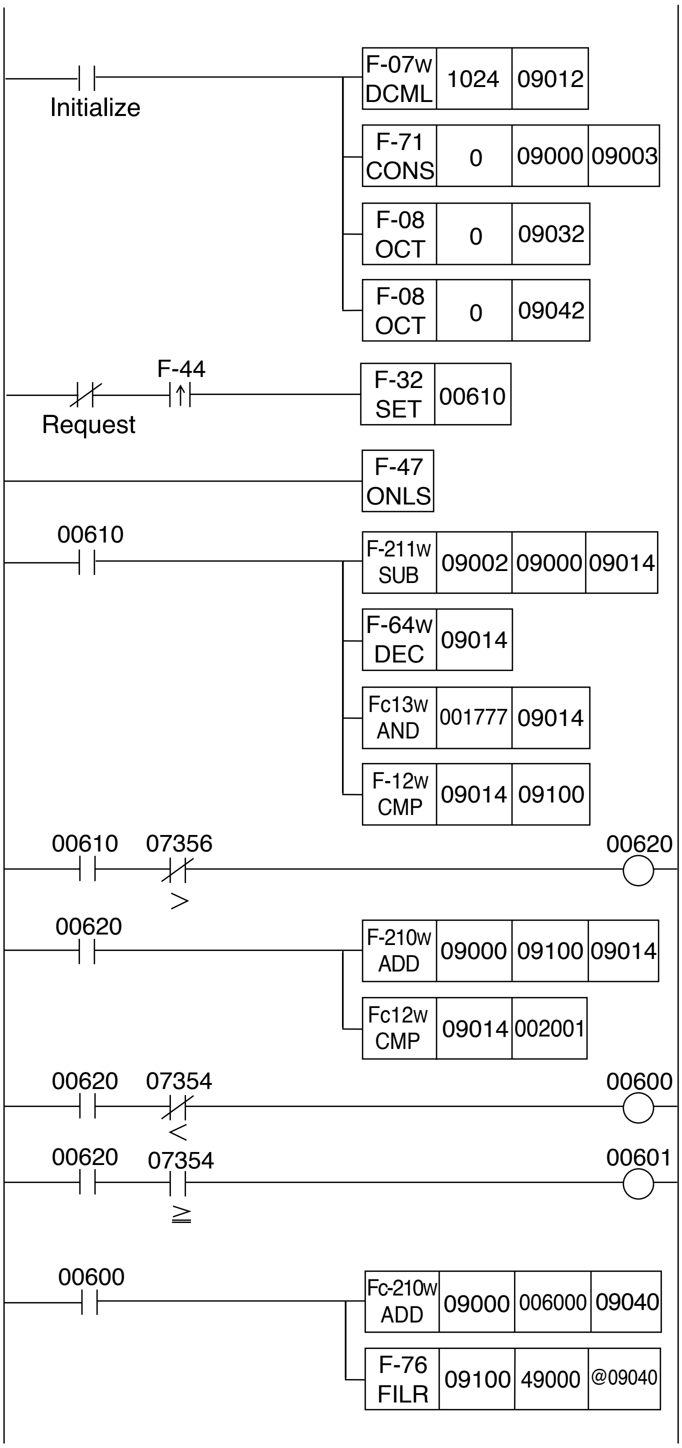
There are two ways to transmit data, as shown below:



(a) Normal case

(b) Write to the end of the buffer and return to the top

09000, 1	WP
09002, 3	RP
09010, 11	Work register for number of data bytes transmitted
09012, 13	Constant (1024)
09014, 15	For checking the buffer size
09030 to 32	Data buffer pointer
09040 to 42	Ring buffer pointer


Initialize
F-07w DCML | 1024 | 09012
F-71 CONS | 0 | 09000 | 09003
F-08 OCT | 0 | 09032
F-08 OCT | 0 | 09042
F-44
Request
F-32 SET | 00610
F-47 ONLS
00610
F-211w SUB | 09002 | 09000 | 09014
F-64w DEC | 09014
Fc13w AND | 001777 | 09014
F-12w CMP | 09014 | 09100
00610
07356
>
00620
00620
F-210w ADD | 09000 | 09100 | 09014
Fc12w CMP | 09014 | 002001
00620
07354
<
00600
00620
07354
≥
00601
00600
Fc-210w ADD | 09000 | 006000 | 09040
F-76 FILR | 09100 | 49000 | @09040

Initial setting

- Set the constant (1 K-bytes).
- Initialize RP and WP.
- Initialize the file number for the data buffer pointer.
- Initialize the file number for the ring buffer pointer.

00610 turns ON to make a request to send.

Calculate the free area in the buffer.

- RP—WP
- Subtract one, in order to keep the RP ahead of the WP.
- Mask 1 K-byte area.

Compare the number of bytes of data to be transmitted with the free space in the buffer.

Turn ON when there is enough free space in the buffer.

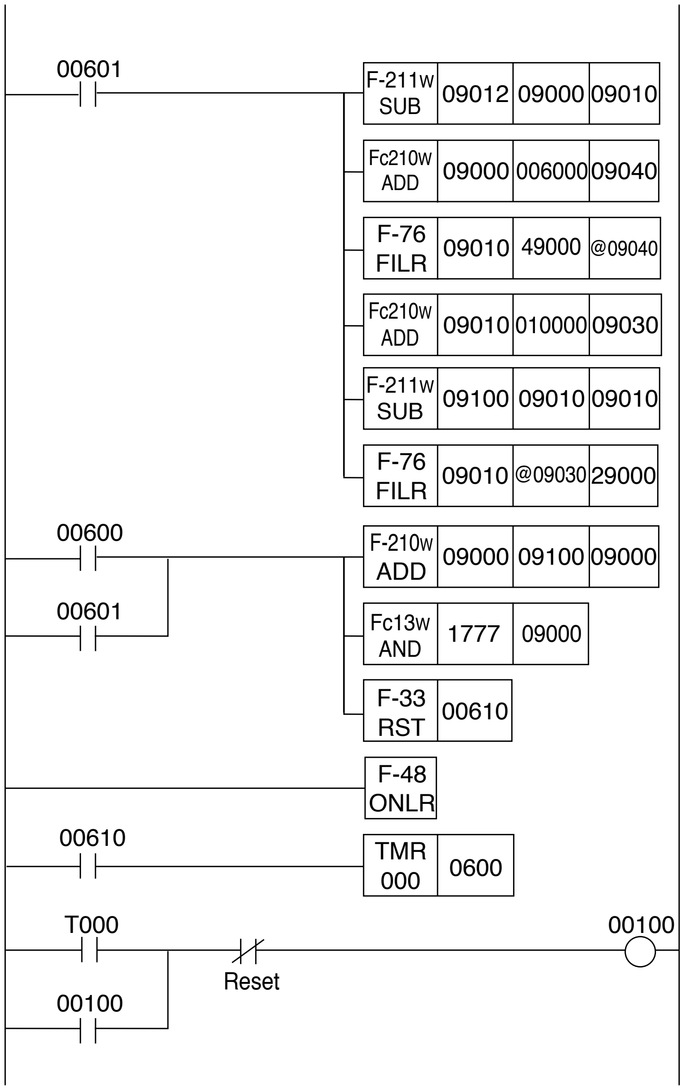
Check the instruction type for whether to return to the top or not when RP reaches the end of the buffer.

(a) The normal case

(b) Write to the end of the buffer and return to the top.

Data transmission for case (a)

- Calculate the ring buffer pointer (absolute address) using the WP.
- Data transmission


00601
F-211w SUB | 09012 | 09000 | 09010
Fc210w ADD | 09000 | 006000 | 09040
F-76 FILR | 09010 | 49000 | @09040
Fc210w ADD | 09010 | 010000 | 09030
F-211w SUB | 09100 | 09010 | 09010
F-76 FILR | 09010 | @09030 | 29000
00600
00601
F-210w ADD | 09000 | 09100 | 09000
Fc13w AND | 1777 | 09000
F-33 RST | 00610
F-48 ONLR
00610
TMR 000 | 0600
T000
00100
Reset
00100

Data transmission for case (b)

- Calculate the number of bytes from the current WP to the end of the ring buffer.
- Calculate ring buffer pointer (absolute address) using the WP.
- Data transmission
- Enter the data buffer pointer of the last half (file 0, 010000 = 49000).
- Calculate the number of bytes remaining.
- Transfer the balance of the data

Update the WP

End of data transmission

If the data transmission terminates less than one minute after the request to send, the JW-255CM will indicates that an error has occurred.

(This will also happen when there is no free space in the ring buffer and the host computer does not pick up the data with the ring buffer read command.)

It is recommended that the host computer be instructed to periodically issue a ring buffer read command and pick up any data waiting in the buffer.

(2) Data flow direction (from host computer to the PLC)

- Direct assignment
- Ring buffer 01
  Size and address: Occupy 1 K bytes starting at 29000
- Ring buffer information storage area: Starts at 09000

■ **Parameter settings**

Parameter address	Set value	Details	
1410, 1411	004000(8)	File 004000	09000
1412	00	File 0	
1413	81(H)	Data flow direction	
1414	0C(H)	File 006000	29000
1415	00	File 0	
1416	04	1 K-bytes	
1417	80(H)	Direct assignment	

■ **Processing ladder program**

1) Initialize the pointers

The read and write pointers should be initialized (set to 0) using a ladder program when starting the module.

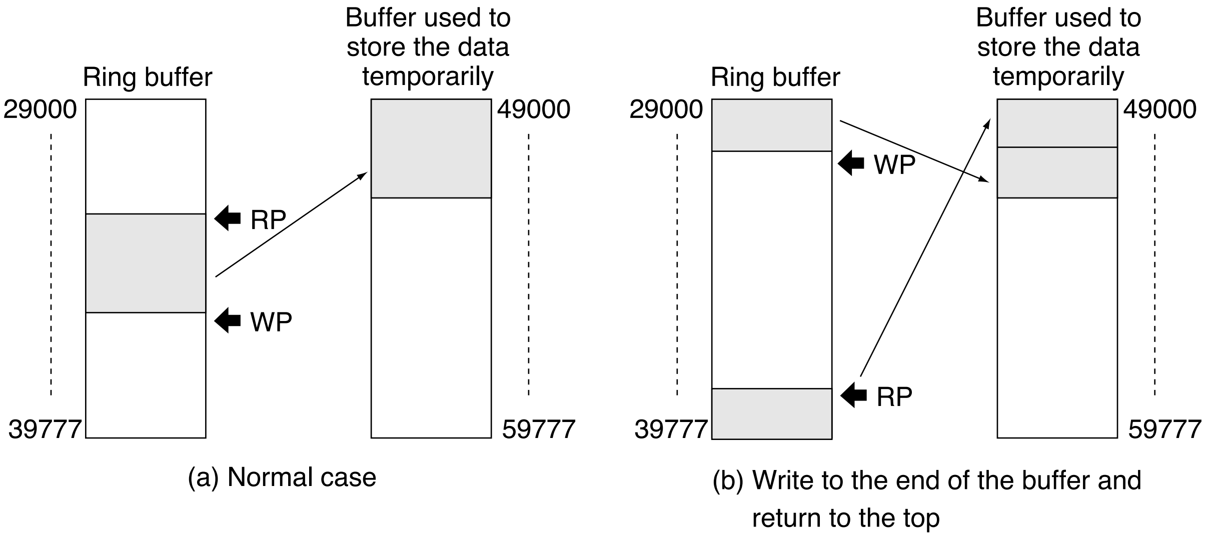
2) Picking up the write data

When the module receives a ring buffer write command from the host computer, it will pick up the data that has been received (temporarily stored in other memory). To find out whether it has been received or not, check whether the WP and RP are in the same position. To determine the number of bytes received, compare WP and RP values.

3) Update the RP

After temporarily storing the data, the module updates the RP.

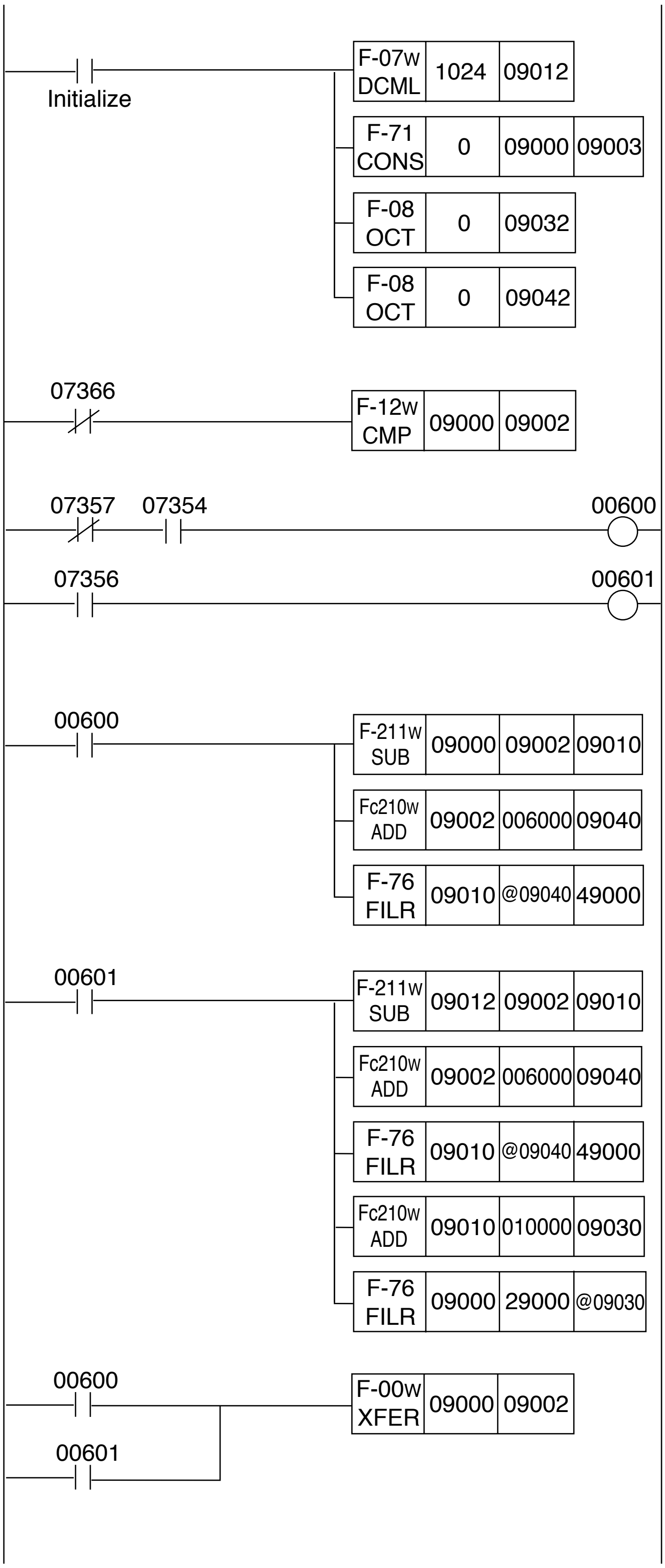
There are two ways to transmit data, as shown below:


Ring buffer
Buffer used to store the data temporarily
29000
39777
49000
59777
RP
WP
Ring buffer
Buffer used to store the data temporarily
29000
39777
49000
59777
WP
RP

(a) Normal case

(b) Write to the end of the buffer and return to the top

09000, 1	WP
09002, 3	RP
09010, 11	Number of data bytes transmitted
09012, 13	Constant (1024)
09030 to 32	Temporary storage buffer pointer
09040 to 42	Ring buffer pointer


Initialize
F-07w DCML 1024 09012
F-71 CONS 0 09000 09003
F-08 OCT 0 09032
F-08 OCT 0 09042
07366
F-12w CMP 09000 09002
07357
07354
00600
07356
00601
00600
F-211w SUB 09000 09002 09010
Fc210w ADD 09002 006000 09040
F-76 FILR 09010 @09040 49000
00601
F-211w SUB 09012 09002 09010
Fc210w ADD 09002 006000 09040
F-76 FILR 09010 @09040 49000
Fc210w ADD 09010 010000 09030
F-76 FILR 09000 29000 @09030
00600
00601
F-00w XFER 09000 09002

Initial setting

- Set the constant (1 K-bytes).
- Initialize RP and WP.
- Initialize the file number of the temporary data storage buffer pointer.
- Initialize the file number of the ring buffer pointer.

Check for fresh data

- Whether the WP has been updated.

(a) Normal update method (WP > RP).

(b) Write to the buffer end and return to the top (WP < RP).

Method (a)

- Calculate the number of data bytes written.
- Calculate the top data address from the RP.
- Transmit the data to the temporary storage buffer.

Method (b)

- Calculate the number of data bytes written. (From RP to the end of the ring buffer)
- Calculate the data top address from the RP.
- Transmit the data to the temporarily storage buffer.
- Update the temporary storage buffer pointer. (File 0, 010000 = 49000)
- Transfer the balance of the data (from top of the ring buffer to the WP) to the temporary storage buffer.

Update the RP.

8-5 Computer link error code table

RSLT (Hexadeci-mal)	Details
00	Normally end
01	Format error
06	PLC does not stop operation
07	Verify error of write command.
0F	Time out while accessing memory.
13	Tried to set/reset TMR/CNT while PLC stops operation.
10	Miss match write enable mode.
40	Undefined ring buffer
41	The area for the corresponding buffer has not been assigned correctly.
42	The current read and/or writes pointer values are outside of the buffer range.
43	The data you want to write will exceed the free space in the buffer when the data is actually written.
44	The data you want to write is larger than the buffer can hold.
48	Undefined standard buffer
49	Incorrectly identified a standard buffer
4A	The number of bytes to read or write exceeded the buffer length of the standard buffer.

8-6 Command execution completion information

When the module completes execution of a computer link command, it writes the completion details in the PLC's data memory.
This function can be selected by setting a parameter.

[1] Setting the parameters

Parameter address	Details
3660 to 3667	Set the command execution completion information storage area

3660	Top file address of the command execution completion information
3661	
3662	File number of the command execution completion information
3663	Not used
3664	Set the number of command execution completion information bytes Assign at least 16 bytes. 64 bytes is the maximum size.
3665	
3666	Not used
3667	This information will become effective when $80_{(H)}$ is entered

[2] Command execution completion information

The command execution completion information has the following format.

+00	Target station IP address
+01	
+02	
+03	
+04	Target station port number
+05	
+06	Connection number of the source station
+07	$00_{(H)}$
+10	Execution result
+12	
+13	
+14	Copy to the receive command (except the header)
+15	
+16	
:	
+n	

This area is written after the module executes the command. To clear the contents, use the ladder program.
The following area in the data memory can be used for this format.

JW20H, J-board (Z300)		JW30H *1		J-board (Z500) *2	
File number	File addresses	File number	File addresses	File number	File addresses
File 0	000000 to $017777_{(8)}$	File 0	000000 to $035777_{(8)}$	File 0	000000 to $035777_{(8)}$
		File 1	000000 to $037777_{(8)}$	File 1	000000 to $037777_{(8)}$
		File 2 to 3, 10 to $2C_{(H)}$	000000 to $177777_{(8)}$	File 2	000000 to $177777_{(8)}$

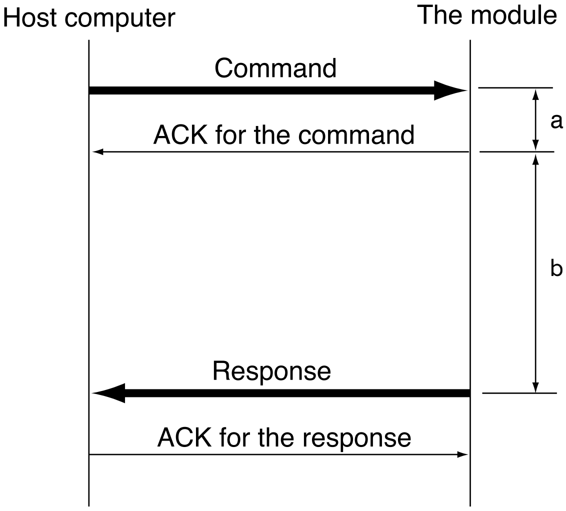
*1 The JW30H memory capacity varies, depending on the control module type.
*2 The Z500 memory capacity varies, depending on the CPU board type.
(Shown above are maximum values.)

8-7 Time interval required for communication

The time interval after the module receives the command until its sends the response varies with the PLC scan time, the number of nodes connected, communication data volume, etc. Shown below is a rough estimate of the time interval.

Assumption: Number of nodes connected: 1. PLC scan time: 8 ms. Data: 1024 bytes

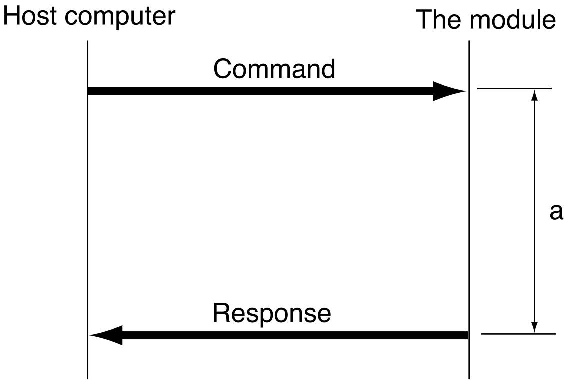
(1) When the TCP method is used



a: Time after receiving the command until returning the TCP level ACK for the command.
10ms on average

b: Time after returning an ACK until a response is sent
40ms on average This time includes the time that the module must wait to access the control module (CPU board). (Maximum PLC scan time.)

(2) When the UDP method is used



a: Time after receiving the command until a response is sent
40ms on average This time includes the time that the module must wait to access the control module (CPU board). (Maximum PLC scan time.)

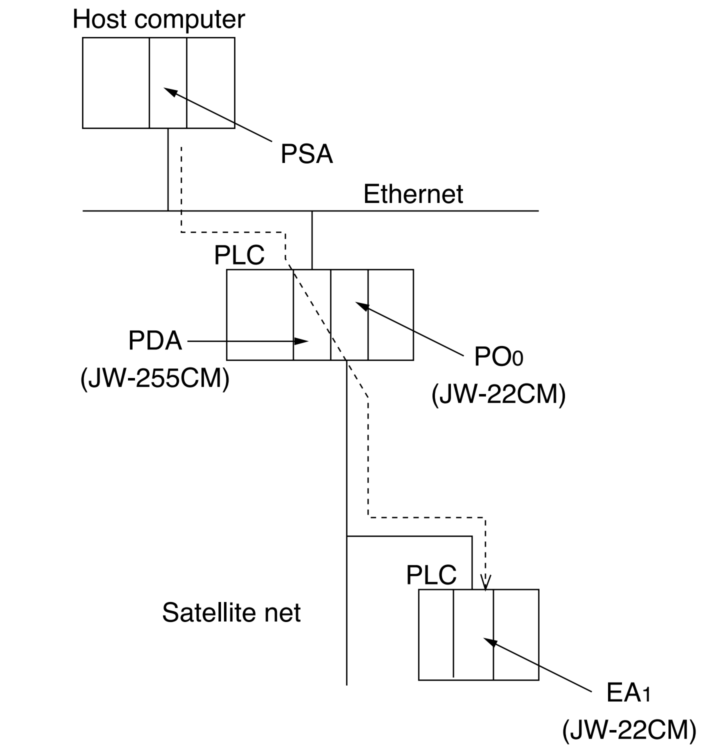
Note: The time intervals above are for reference only. They may vary with each change in conditions. Generally, the communication time will be longer in the conditions below:
1) When the number of ports used is increased.
2) When the PLC scan time gets longer.

8-8 Two-layer communication with satellite net

In order to communicate with the satellite net on a different layer, use the following information in the communication format header (refer to page 8-1) as an extension header.

0	$00_{(H)}$	40 bytes
1	$00_{(H)}$	
2	$00_{(H)}$	
3	$00_{(H)}$	
4	PDA	
5	PSA	
6	FT_0	
7	PO_0	
8	EA_1	
9	FT_1	
⋮	$00_{(H)}$	
39		
40	Command line	

● Example of connection



- When making a two-layer communication with a satellite net, the frame needs to contain the information including the source, transit stations, and destination, and slot number (i.e. designating the communication path). The satellite net uses eight bits to represent a station number. For that reason, when designating a module on the Ethernet, a station number for the satellite net needs to be designated. The address is referred to as a pseudo station number.

(a) PDA: Pseudo destination address
Designate the station number of the module that connects with the satellite net. This may be any value within the range from $01_{(H)}$ to $40_{(H)}$ that can be discriminated from other equipment on the Ethernet.

(b) PSA: Pseudo source address
Designate the station number for the equipment sending the command. This may be any value within the range from $01_{(H)}$ to $40_{(H)}$ that can be discriminated from other equipment on the Ethernet.
With respect to the response, the pseudo target station number that is given by the command will be set.

(c) FT_0: Frame type 0
Designate $60_{(H)}$.

(d) PO_0: Transit slot number
Designate the slot number on the transit station PLC that the satellite/net work module JW-22CM is installed on. When the transit station is JW50H/70H/100H, this number will be the slot No. on the transit station. This number is 2, 3 ... up to 7 from the next position of the control module (in case ZW-6CC is used.)

(e) EA_1: End target station address
Designate the end target station address $01_{(H)}$ to $40_{(H)}$ on the satellite net. If the data link master station is the destination, set the $40_{(H)}$.

(f) FT_1: Frame type 1
Set the $40_{(H)}$.

(g) Command line: Command/response line
c-ID/r-ID and after of communication format (refer to page 8-1)

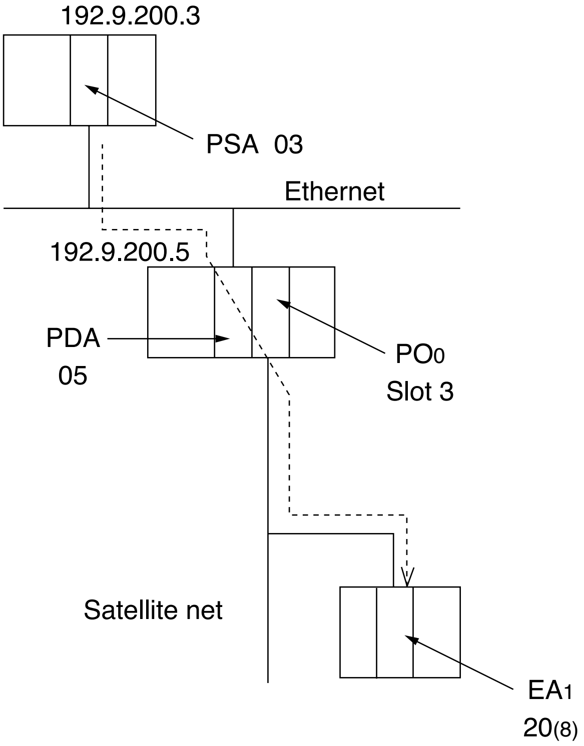
Remarks

The two-layer communication is possible only with the computer link to the module on the satellite net from the host computer on the Ethernet via transit stations. Communication in the reverse direction, or the computer link from the host computer on the satellite net to this module, is not possible.

[Example] In the following example as shown in the diagram, the expansion header needs to be as follows.

0	00$_{(H)}$
1	00$_{(H)}$
2	00$_{(H)}$
3	00$_{(H)}$
4	PDA = 05$_{(H)}$
5	PSA = 03$_{(H)}$
6	FT$_0$ = 60$_{(H)}$
7	PO$_0$ = 03$_{(H)}$
8	EA$_1$ = 10$_{(H)}$
9	FT$_1$ = 40$_{(H)}$
⋮ 39	00$_{(H)}$
40	Command line

40 bytes



Chapter 9. SEND/RECEIVE Functions

The SEND/RECEIVE functions transmit data from this module to other stations or receive data from other stations. There are two types of SEND/RECEIVE functions: an instruction system and a start from data memory system.

9-1 Instruction system

This instruction method will start the SEND/RECEIVE function using an exclusive instruction, and can be used only when the JW-255CM/25TCM is installed on the JW30H, and the Z-339J is installed on the J-board Z500 series models. (The instruction system is not available when the module is installed on the JW20H and J-board Z300 series.)

[1] Source/destination address and channel

The SEND/RECEIVE instructions set the rack, slot, and channel addresses of the installed module, target station number, and the data memory addresses on the target station of the implemented module by the F-202(OPCH) instruction, and the data memory address on the module and the number of bytes to be transmitted by the F-204 (send) or F-205 (receive) instructions.

Among the above items, the channel number and the target station number are expressed in the address system of the satellite net. The module corresponds these with the following.

(1) Channel number

Channel numbers CH0 to CH3 correspond to the following port address.

No. of channels	Port address
CH0	6000(H)
CH1	6001(H)
CH2	6002(H)
CH3	6003(H)

Those stations, using the send/receive instructions, need to open the connection using the above port numbers. Use TCP_Active or UDP for the connection used for the SEND/RECEIVE instructions. The method for opening the target station is as shown below.

Instruction starting station	Communication target station
TCP_Active	TCP_Passive
UDP	UDP

Any value may be used for the port number of the target station.

(2) Target station number

The correspondence of the target station number includes two types: automatic correspondence and individual registration.

1) Automatic correspondence

This method assumes the station number of the send/receive instructions as the node number of the target station IP address, and fixes the connected port to 6010(H). When the target station number is set to 000, the node number of the IP address becomes 40(H).

2) Individual registration

This method registers the parameters of the correspondence of the station number designated by F-202 to the IP address port number of the actual target station. Up to 31 records of correspondence may be set.

The choice between the automatic correspondence and the individual registration is set by a parameter.

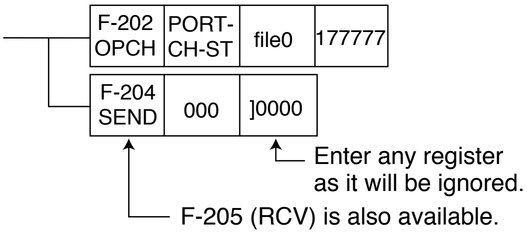
■ Parameters for "automatic/individual" registration

<table>
<tr><th>Parameter address</th><th colspan="2">Details</th></tr>
<tr><td>0400</td><td colspan="2">Assign a station number corresponding to the table (for SEND/RECEIVE)
Assignment of the relationship between the station number used with the SEND/RECEIVE function and the actual address.
If "automatic" is selected, the module treats the target station numbers as node numbers on the IP addresses, and sets the target port to 6010$_{(H)}$.
If "individual" is selected, a maximum of 31 types of relationship can be registered concerning the relationship between the station number used with the command and the target address. (Parameters 0410 to 0777 can be used with this selection.)
01$_{(H)}$: Automatic
02$_{(H)}$: Individual</td></tr>
<tr><td>0401 to 0407</td><td colspan="2">Reserved area</td></tr>
<tr><td>0410
to
0417</td><td colspan="2">Station number correspondence table 1.
This is effective only when parameter 0400 is 02$_{(H)}$.
<table>
<tr><td>0410</td><td>Setting 00$_{(H)}$: Not set (the information below will be ignored)
01$_{(H)}$: Set</td></tr>
<tr><td>0411</td><td>Target station number</td></tr>
<tr><td>0412</td><td rowspan="2">Target station port number
(assign a part as one word of data (two bytes) in octal notation)</td></tr>
<tr><td>0413</td></tr>
<tr><td>0414</td><td rowspan="4">Target station IP address (0417 is host ID)</td></tr>
<tr><td>0415</td></tr>
<tr><td>0416</td></tr>
<tr><td>0417</td></tr>
</table></td></tr>
<tr><td>0420 to 0427</td><td>Station number corresponding table 2.
This is effective only when parameter 0400 is 02$_{(H)}$.</td><td rowspan="5">Set the same as the station numbers in correspondence table 1.</td></tr>
<tr><td>0430 to 0437</td><td>Station number corresponding table 3.
This is effective only when parameter 0400 is 02$_{(H)}$.</td></tr>
<tr><td>:</td><td>:</td></tr>
<tr><td>0760 to 0767</td><td>Station number corresponding table 36.
This is effective only when parameter 0400 is 02$_{(H)}$.</td></tr>
<tr><td>0770 to 0777</td><td>Station number corresponding table 37.
This is effective only when parameter 0400 is 02$_{(H)}$.</td></tr>
</table>

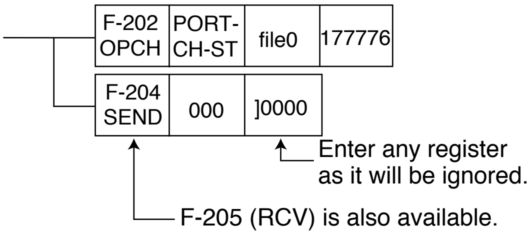
(3) TCP connection and disconnection

The SEND/RECEIVE instructions have nothing to do with the concept of establishing/breaking the connection. When establishing or breaking the connection is required when using the SEND/RECEIVE functions at the port that has been opened in TCP_Active, this operation corresponds to the subsequent SEND/RECEIVE instructions.

1) **Establishing the connection**
Executing the SEND/RECEIVE functions with the memory address of the target station file number = 0, file address = 177777$_{(8)}$, and the number of bytes transmitted = 0 will start the connection establishing operations with the target station. The connection takes approximately two seconds to complete.


F-202
OPCH
PORT-
CH-ST
file0
177777
F-204
SEND
000
]0000
Enter any register as it will be ignored.
F-205 (RCV) is also available.

2) **Disconnection**
Executing the SEND/RECEIVE functions with the memory address of the target station file number = 0, file address = 177776$_{(8)}$, and the number of bytes transmitted = 0 will start the disconnecting operations from the target station.


F-202
OPCH
PORT-
CH-ST
file0
177776
F-204
SEND
000
]0000
Enter any register as it will be ignored.
F-205 (RCV) is also available.

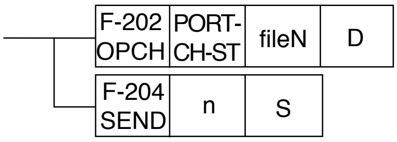
3) **Data transmission**
Any address and number of bytes transmitted other than shown at 1) and 2) will start the actual SEND/RECEIVE operations.

No connection and/or disconnection is required when the opening is made in UDP.

[2] SEND/RECEIVE instructions operation

(1) SEND

This function operates by the combination of F-202 (OPCH) and F-204 (SEND).



PORT: Module No. switch of this module (0 to 6)

CH: Channel number used (0 to 3)
Channels CH0 to CH3 correspond to 6000(H) to 6003(H).

ST: Target station number (00 to 77(8))
Actual target IP address and port number are determined by the station number correspondence table.
(See the previous page.)

file N: Data area of the target station PLC (File number)

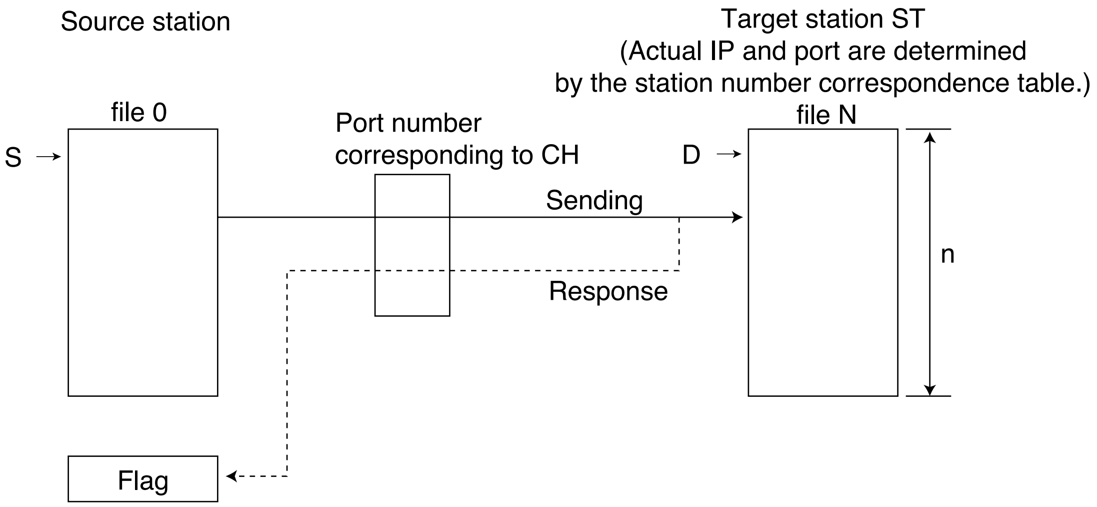
ADRS: Data area top file address of target station PLC

n: Number of data bytes transmitted
(000 to 377(8), 256 bytes at 000)

S: Data area top register of source station data

- Available setting ranges for S

]0000 to]1577
]2000 to]7577
b0000 to b1777
b2000 to b3777
09000 to 99777
E0000 to E7777
File 1 000000 to 037777
@]0000 to @]1574
@]2000 to @]7574
@b0000 to @b1774
@b2000 to @b3774
@09000 to @99774
@E0000 to @E7774
File 1 @000000 to @037774



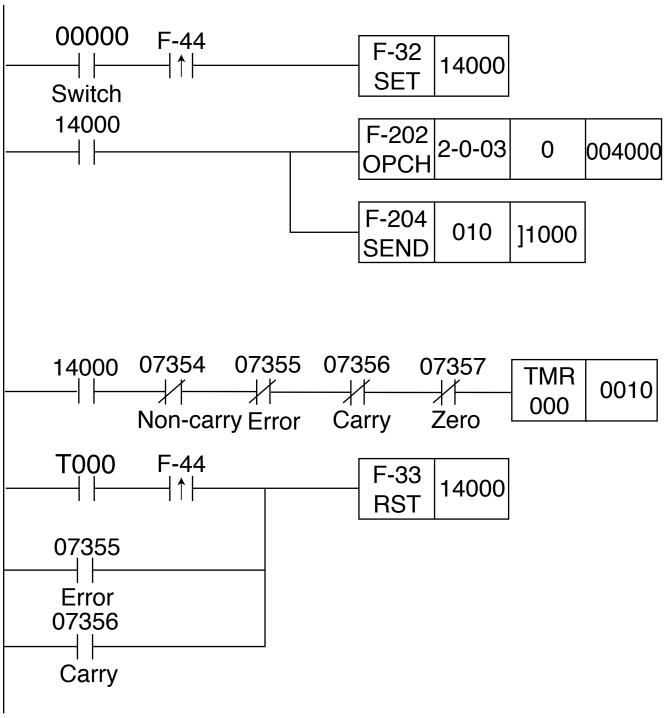
Flag status during and after the operation

	Zero 07357	Carry 07356	Error 07355	Non-carry 07354	Explanation
No response from port	0	0	1	0	The setting values for PORT and the module No. switch in this module are different.
Communication jam	0	0	0	1	This condition may instantaneously occur while other send instruction is being executed. However, as soon as the condition becomes clear for the execution of an instruction, the status will turn to "Communicating."
Communicating	1	0	0	1	The communicating is being run. Once the communication completes, the status will change to either "normal end" or "abnormal end."
Normal end	0	1	0	0	When the SEND instruction has successfully completed.
Abnormal end (time out)	0	1	1	0	When there is no response from the other end.
Abnormal end (error)	1	1	1	0	Some of the potential causes : No connection existent. A communication attempt has been made without establishing any connection in TCP. A connection-establishing attempt has been made when a connection is already established in TCP.

■ Sample of program

When transferring 8 bytes of data from source station register]1000 to the register 09000 of the target station number 03 :

Port number installed on this module of source station : 2
Channel used : 0



Set the sending relay 14000.

Port 2, CH0, station number 3,
File number 0
File address 004000

Transfers 8 bytes of the range
]1000 to 1007.

Detects non-execution at power ON.
(Flags do not change for one second.)

Resets 14000 when the execution completes.

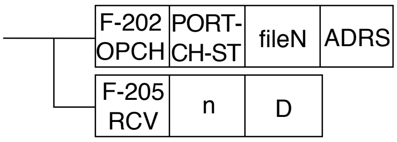
In this sample program, the SEND function is executed to transfer data to the target station corresponding to 3 on the station number correspondence table. The number of the ports used on source station is 6000(H).

Remarks

- The entry condition of F-202 or 204 instruction needs to be kept ON until the execution of the instruction completes (or until any error occurs or the carry flags turns ON). If the entry condition turns OFF while the instruction is being run, the instruction will end in an incomplete condition. Once this condition occurs, a "communication jam" occurs when an instruction execution is attempted the next time, and the instruction will not run properly. To restore the condition, power OFF the PLC, and turn it ON again.
- If the entry condition turns OFF, due to an instantaneous power failure, turn the entry condition to "latched relay" as a remedy. If, however, any power loss occurs while an instruction is being run using "latched relay," turning the power ON again will cause F-202 or F-204 instructions process being run to disappear, and the entry condition will stay ON. Therefore, the start of the entry may not be detected. Since, in this case, all flags will be turned OFF, detect the continuation of the OFF condition of all flags using the timer, and then reset the entry condition before running the next instruction.

(2) RECEIVE

This function is executed by the combination of F-202 (OPCH) and F-205 (RCV).



PORT: Module No. switch of this module (0 to 6)

CH: Channel number used (0 to 3)
Channels CH0 to CH3 correspond to 6000(H) to 6003(H).

ST: Target station number (00 to 77(8))
Actual target IP address and port number are determined by the station number correspondence table.
(Refer to page 9-2.)

file N: Data area of the target station PLC (File number)

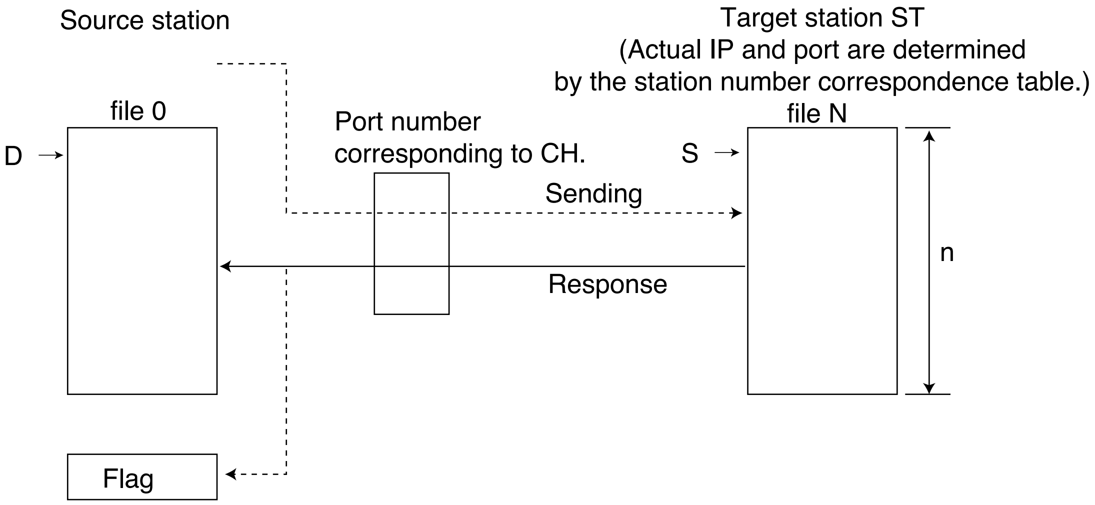
ADRS: Data area top file address of target station PLC

n: Number of data bytes transmitted
(000 to 377(8), 256 bytes at 000)

S: Data area top register of source station data

- Available setting ranges for D

]0000 to]1577
]2000 to]7577
b0000 to b1777
b2000 to b3777
09000 to 99777
E0000 to E7777
File 1 000000 to 037777
@]0000 to @]1574
@]2000 to @]7574
@b0000 to @b1774
@b2000 to @b3774
@09000 to @99774
@E0000 to @E7774
File 1 @000000 to @037774



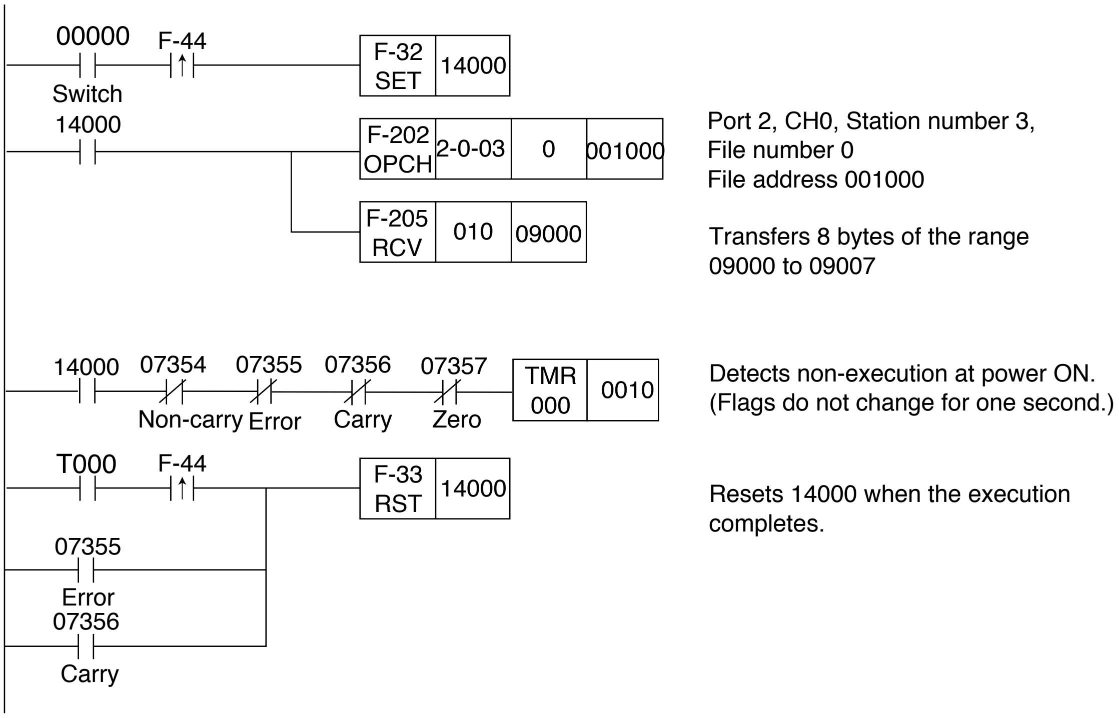
Flag status during and after the operation

	Zero 07357	Carry 07356	Error 07355	Non-carry 07354	Explanation
No response from port	0	0	1	0	The setting values for PORT and the module No. switch in this module are different.
Communication jam	0	0	0	1	This condition may instantaneously occur while other RECEIVE instruction is being executed. However, as soon as the condition becomes clear for the execution of an instruction, the status will turn to "Communicating."
Communicating	1	0	0	1	The communicating is being run. Once the communication completes, the status will change to either "normal end" or "abnormal end."
Normal end	0	1	0	0	When the RECEIVE instruction has successfully completed.
Abnormal end (time out)	0	1	1	0	When there is no response from the other end.
Abnormal end (error)	1	1	1	0	Some of the potential causes: No connection existent. A communication attempt has been made without establishing any connection in TCP. A connection-establishing attempt has been made when a connection is already established in TCP.

■ Sample of program

When transferring 8 bytes of data from the register]1000 of target station 03 to the register 09000 of the source station.

Port number installed on this module of source station : 2
Channel used : 0



Port 2, CH0, Station number 3,
File number 0
File address 001000

Transfers 8 bytes of the range
09000 to 09007

Detects non-execution at power ON.
(Flags do not change for one second.)

Resets 14000 when the execution completes.

In this sample program, the RECEIVE function is executed to transfer data to the target station corresponding to 3 on the station number correspondence table. The number of the ports used on source station is 6000(H).

Remarks

- The entry condition of F-202 or 205 instruction needs to be kept ON until the execution of the instruction completes (or until any error occurs or the carry flags turns ON). If the entry condition turns OFF while the instruction is being run, the instruction will end in an incomplete condition. Once this condition occurs, a "communication jam" occurs when an instruction execution is attempted the next time, and the instruction will not run properly. To restore the condition, power OFF the PLC, and turn it ON again.
- If the entry condition turns OFF, due to an instantaneous power failure, turn the entry condition to "latched relay" as a remedy. If, however, any power loss occurs while an instruction is being run using "latched relay," turning the power ON again will cause F-202 or F-205 instructions process being run to disappear, and the entry condition will stay ON. Therefore, the start of the entry may not be detected. Since, in this case, all flags will be turned OFF, detect the continuation of the OFF condition of all flags using the timer, and then reset the entry condition before running the next instruction.

[3] Error recovery

SEND/RECEIVE functions involve a watchdog timer at the application level. It has the following default value.

- When using TCP
  Default value = 2 minutes (a long time is used in consideration of the retry at the lower layer.)
- When using UDP
  Default value = 1 second
  Default value may be modified in 100ms unit for each of the channels through the parameters.

■ Parameter for setting the monitor timer

Parameter address	Contents
0020 to 0021	Setting value of monitor timer for the SEND/RECEIVE functions CH0 TCP Sets the application level monitor timer when using CH0 of the SEND/RECEIVE functions in TCP. A unit is 100ms. Setting of 0000(H) represents the default value of 2 minutes.
0022 to 0023	Setting value of monitor timer for the SEND/RECEIVE functions CH0 UDP Sets the application level monitor timer when using CH0 of the SEND/RECEIVE functions in UDP. A unit is 100ms. Setting of 0000(H) represents the default value of 1 second.
0024 to 0025	Setting value of monitor timer for the SEND/RECEIVE functions CH1 TCP (The contents is the same as CH0.)
0026 to 0027	Setting value of monitor timer for the SEND/RECEIVE functions CH1 UDP (The contents is the same as CH0.)
0030 to 0031	Setting value of monitor timer for the SEND/RECEIVE functions CH2 TCP (The contents is the same as CH0.)
0032 to 0033	Setting value of monitor timer for the SEND/RECEIVE functions CH2 UDP (The contents is the same as CH0.)
0034 to 0035	Setting value of monitor timer for the SEND/RECEIVE functions CH3 TCP (The contents is the same as CH0.)
0036 to 0037	Setting value of monitor timer for the SEND/RECEIVE functions CH3 UDP (The contents is the same as CH0.)

[4] Other notes

1) When the this module is connected to an Ethernet site with "TCP_Active," a space is left before making the next connection, in order to guarantee a disconnection at the end of the communication. This time interval is referred to as "2MSL."
   In this module, 2MSL is set at 4 minutes. So, allow 4 minutes or more before reconnecting the channel after the disconnection.
2) With the connection that is established in TCP, if the communication is disabled due to the loss of power or any other causes on the station on the other end, a time out will occur, and the connection will be automatically break at the same time. Therefore, to communicate with the same station the next time, the connection needs to be made again.

9-2 Data memory starting system

[1] System

This system is used to store information related to the SEND/RECEIVE function at a specific data memory location (SEND/RECEIVE communication information storage area) using a ladder program, and provides processing instructions. The following area in the data memory can be used for this storage.

JW20H, J-board (Z300)		JW30H *1		J-board (Z500) *2	
File number	**File addresses**	**File number**	**File addresses**	**File number**	**File addresses**
File 0	000000 to 017777$_{(8)}$	File 0	000000 to 035777$_{(8)}$	File 0	000000 to 035777$_{(8)}$
		File 1	000000 to 037777$_{(8)}$	File 1	000000 to 037777$_{(8)}$
		File 2 to 3, 10 to 2C$_{(H)}$	000000 to 177777$_{(8)}$	File 2	000000 to 177777$_{(8)}$

*1 The JW30H memory capacity varies, depending on the control module type.
*2 The Z500 memory capacity varies, depending on the CPU board type.
} (Shown above are maximum values.)

The SEND/RECEIVE function in the data memory starting system can be operated independently from the command systems. A maximum of 1 K-bytes of data can be sent and received.
Use port 6008$_{(H)}$ for the SEND/RECEIVE function in the data memory starting system.

[2] Parameter setting

■ **Parameters for setting the communication information storage area**

<table>
<tr><th>Parameter address</th><th>Details</th></tr>
<tr><td>3770
to
3773</td><td>Top address of the SEND/RECEIVE function communication information storage area
<table>
<tr><td>3770</td><td rowspan="2">Top file address in the communication information storage area</td></tr>
<tr><td>3771</td></tr>
<tr><td>3772</td><td>File number for the communication information storage area</td></tr>
<tr><td>3773</td><td>This information is effective when terminated by 80$_{(H)}$.</td></tr>
</table>
</td></tr>
</table>

■ Parameters for “automatic/individual” registration

<table>
<tr><th>Parameter address</th><th colspan="2">Details</th></tr>
<tr><td>0400</td><td colspan="2">Assign a station number corresponding to the table (for SEND/RECEIVE)
Assignment of the relationship between the station number used with the SEND/RECEIVE function and the actual address.
If "automatic" is selected, this module treats the target station numbers as node numbers on the IP addresses, and sets the target port to 6010$_{(H)}$.
If "individual" is selected, a maximum of 31 types of relationship can be registered concerning the relationship between the station number used with the command and the target address. (Parameters 0410 to 0777 can be used with this selection.)
01$_{(H)}$: Automatic
02$_{(H)}$: Individual</td></tr>
<tr><td>0401 to 0407</td><td colspan="2">Reserved area</td></tr>
<tr><td>0410
to
0417</td><td colspan="2">Station number correspondence table 1.
This is effective only when parameter 0400 is 02$_{(H)}$.
<table>
<tr><td>0410</td><td>Setting 00$_{(H)}$: Not set (the information below will be ignored)
01$_{(H)}$: Set</td></tr>
<tr><td>0411</td><td>Target station number</td></tr>
<tr><td>0412</td><td rowspan="2">Target station port number
(assign a part as one word of data (two bytes) in decimal notation)</td></tr>
<tr><td>0413</td></tr>
<tr><td>0414</td><td rowspan="4">Target station IP address (0417 is host ID)</td></tr>
<tr><td>0415</td></tr>
<tr><td>0416</td></tr>
<tr><td>0417</td></tr>
</table>
</td></tr>
<tr><td>0420 to 0427</td><td>Station number corresponding table 2.
This is effective only when parameter 0400 is 02$_{(H)}$.</td><td rowspan="5">Set the same as the station numbers in correspondence table 1.</td></tr>
<tr><td>0430 to 0437</td><td>Station number corresponding table 3.
This is effective only when parameter 0400 is 02$_{(H)}$.</td></tr>
<tr><td>:</td><td>:</td></tr>
<tr><td>0760 to 0767</td><td>Station number corresponding table 36.
This is effective only when parameter 0400 is 02$_{(H)}$.</td></tr>
<tr><td>0770 to 0777</td><td>Station number corresponding table 37.
This is effective only when parameter 0400 is 02$_{(H)}$.</td></tr>
</table>

[3] Communication information storage area

+00	FLAGS	Operation flag (same as]0735 of F-204 and F-205) 00$_{(H)}$: When not in use 90$_{(H)}$: When communicating. Until execution of the instruction is complete. 40$_{(H)}$: Normal end. 60$_{(H)}$: Abnormal end (communication time-out) E0$_{(H)}$: Abnormal end (error response)
+01	TIMER	Time-out time (unit: 100ms) When 00$_{(H)}$ is entered, the time-out time will be the default value (TCP: 2 minutes, UDP: 1 second)
+02	G/TYPE	G (D7): Start instruction. Turns ON when the communication is started. TYPE (D6 to D0), 00$_{(H)}$: SEND, 02$_{(H):}$ RECEIVE
+03	ST1	Target station number. 00 to 77$_{(8)}$
+04		Not used
+05		Not used
+06	n(L)	Number of bytes to be transmitted. 0 to 400$_{(H)}$ (0 to 1024). n(L) is the lower byte. A value of 0 should be used only when connecting or disconnecting. [Note]
+07	n(H)	
+10	ADR_A(L)	File address for source station
+11	ADR_A(H)	
+12	SEG_A	File number for source station
+13		Not used
+14	ADR_B(L)	File address for target station
+15	ADR_B(H)	
+16	SEG_B	Target station file number
+17		Not used

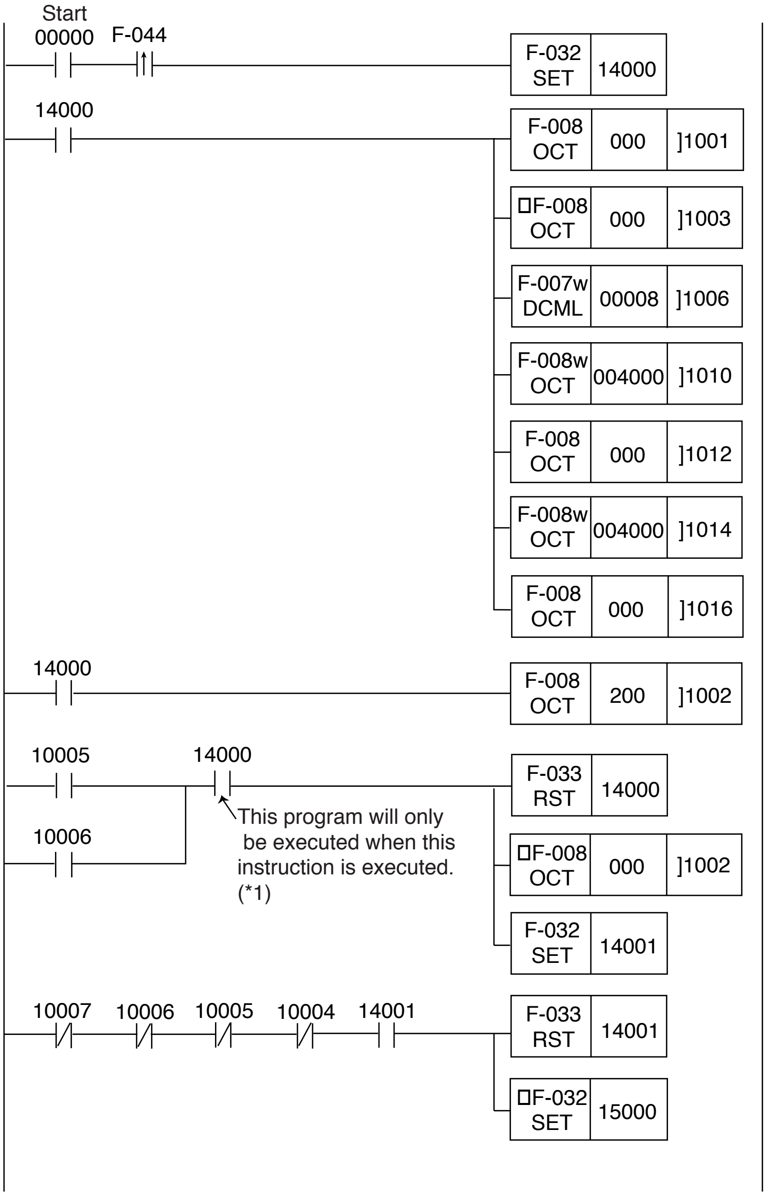
Note: Flag area data is transmitted from this module to the control module.
Other areas are set by the control module.

When you want to use TCP as the communication protocol, the appropriate connection/disconnection operations are required. In this case, assign the same address as used for the instruction system.
- Connection: SEG_B = 0, ADR_B = FFFF$_{(H)}$, n = 0
- Disconnection: SEG_B = 0, ADR_B = FFFE$_{(H)}$, n = 0

[4] Other notes

1) When this module is connected to an Ethernet site with "TCP_Active," a space is left before making the next connection, in order to guarantee a disconnection at the end of the communication. This time interval is referred to as "2MSL."
In this module, 2MSL is set at 4 minutes. So, allow 4 minutes or more before reconnecting the channel after the disconnection.
2) With the connection that is established in TCP, if the communication is disabled due to the loss of power or any other causes on the station on the other end, a time out will occur, and the connection will be automatically break at the same time. Therefore, to communicate with the same station the next time, the connection needs to be made again.

[5] Program example for data memory starting system



Time out time
Default value (TCP: 2 minutes)
(UDP: 1 second)

Target station number: 00

Number of bytes to transmit: 8

The module's starting file address: 4000

The module's starting file number: 0

Starting file address of the target station: 4000

Starting file number of the target station: 0

Sub-routine execution bit turns ON.
[To execute a SEND sub-routine: 200(8)
To execute a RECEIVE sub-routine: 202(8)]

Resets address 14000 when the sub-routine execution is compete.

Turns OFF the sub-routine execution bit.

Sets the relay which waits for the next program execution cycle.

Sets the relay which waits for the next program execution cycle when the flag is turned ON.

When address 15000 turns ON, the next sub-routine will be enabled for execution. (*2)

*1 When multiple SEND/RECEIVE sub-routines are contained in the program for data memory starting system, each sub-routine uses a common flag address. As a result, while one instruction is executing, a flag for a function that is not executing will be influenced by the instruction currently executing. To prevent this, you must create a step that can determine whether or not the next cycle should be executed.

*2 With the data memory starting system, this module needs to confirm whether the execution bit is turned OFF and all flags are 0 (after address 15000 has turned ON). Then, it can be allowed to enter the next SEND/RECEIVE sub-routine program.

Chapter 10. Routing Function

This module can be used for communication via a router (routing function). The routing function can provide communication between host computers that use different net IDs (refer to page 10-3).

- The routing function needs to create a routing table (route control table) in the parameters for this module. There are two methods to create routing table: "Create a default router" and "Create a customized routing table."

Remarks

- Only the computer link functions can be communicated over a router. The SEND/RECEIVE functions cannot be communicated over a router.

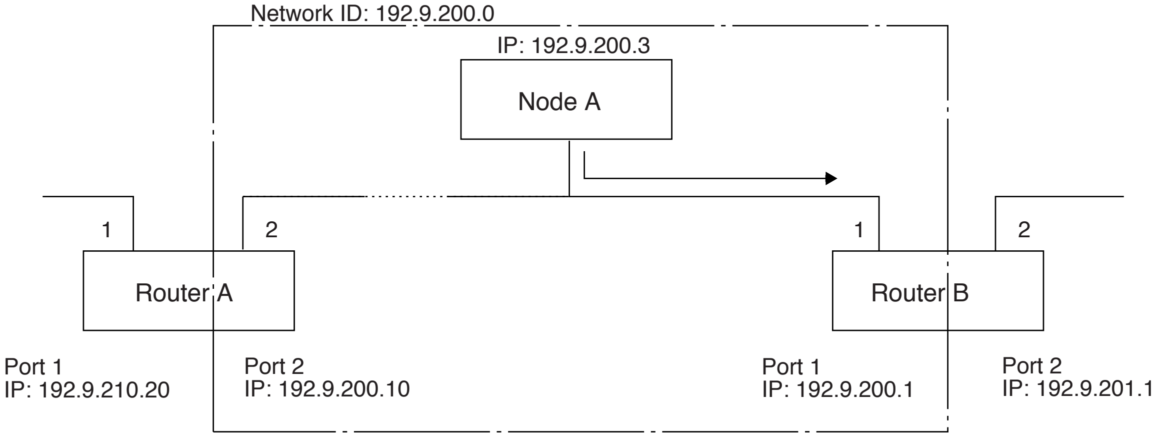
[1] Create a default router

Assign a default router IP address in the parameter locations (addresses 1700 and 1704 to 1707). When the module tries to communicate with IP addresses other than addresses on its own network, the default router table will be used.

■ Default setting of the router parameter

Parameter addresses	Details
1700	With and without the default router settings 00(H): Without a default setting (the information below will be invalid) 01(H): With the default settings
1704	Default router IP addresses (address 1707 is the host ID)
1705	
1706	
1707	

[Example]



When the default router IP address of 192.9.200.1 is assigned in node A, all packets that are intended to go outside the 192.9.200.0 network (such as 192.9.201.0 or 150.24.58.0) will be sent to router B first (port 1: IP address is 192.9.200.1).

Parameter address	Settings (decimal)
1700	1
1704	192
1705	9
1706	200
1707	1

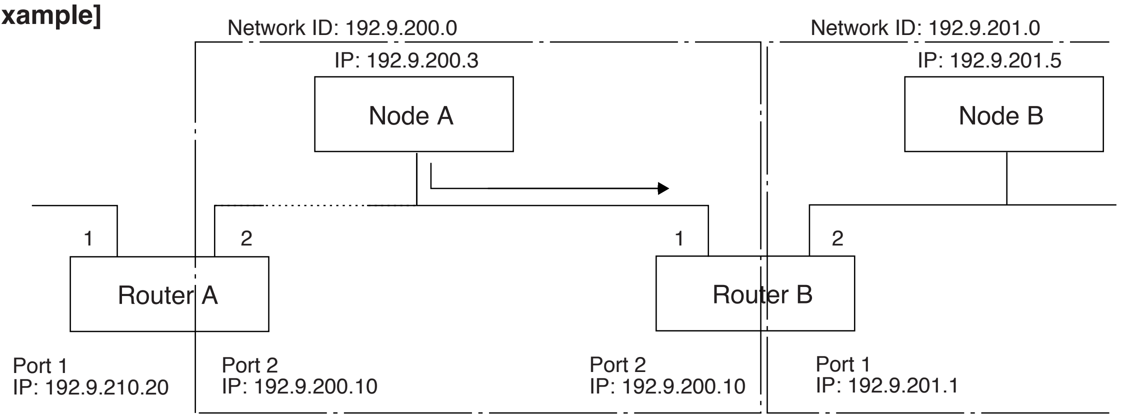
[2] Create a customized routing table

You assign router IP addresses corresponding to the desired network ID in the router parameters (addresses 1600 to 1677). Up to eight network IDs can be defined. The module cannot communicate with any IP address that has a network ID which is not defined in these settings.

■ Parameters for creating the routing table

<table>
<tr><th>Parameter addresses</th><th colspan="2">Details</th></tr>
<tr><td rowspan="4">1600 to 1607</td><td colspan="2">Routing table 0</td></tr>
<tr><td>1600</td><td>With or without customized routing table settings
$00_{(H)}$: Without customized settings
(the information below will be ignored)
$01_{(H)}$: With customized settings</td></tr>
<tr><td>1601
1602
1603</td><td>Target network ID</td></tr>
<tr><td>1604
1605
1606
1607</td><td>IP address in the router corresponding to the network ID (1607 is always the host ID location)</td></tr>
<tr><td>1610 to 1617</td><td>Routing table 1</td><td rowspan="7">Setup each table the same way as routing table 0.</td></tr>
<tr><td>1620 to 1627</td><td>Routing table 2</td></tr>
<tr><td>1630 to 1637</td><td>Routing table 3</td></tr>
<tr><td>1640 to 1647</td><td>Routing table 4</td></tr>
<tr><td>1650 to 1657</td><td>Routing table 5</td></tr>
<tr><td>1660 to 1667</td><td>Routing table 6</td></tr>
<tr><td>1670 to 1677</td><td>Routing table 7</td></tr>
</table>

[Example]



When 192.9.201.0 is used for the target network ID and 192.9.200.1 for the router IP address in node A, all of the packets that are intended for 192.9.201.0 will be sent to router B (192.9.200.1) first.

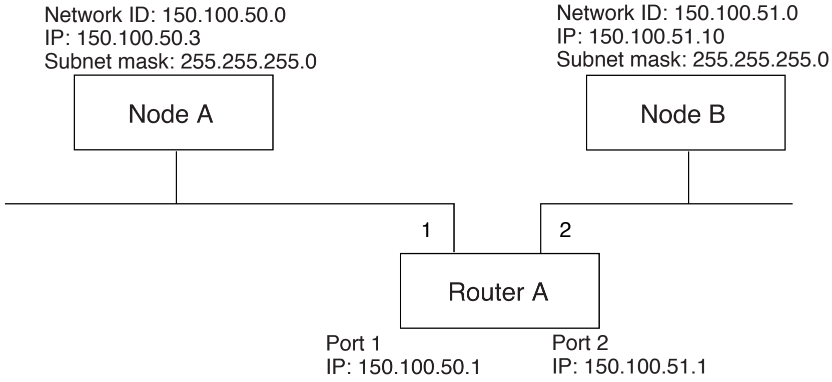
● To assign addresses in routing table 0

Parameter address	Settings (decimal)
1600	1
1601	192
1602	9
1603	201
1604	192
1605	9
1606	200
1607	1

Different network ID

- The net ID means network address. When the sub net mask is used, bit length specified by the sub net mask is applied, not bit lengths specified by each class.
  To communicate between nodes that are different in length specified by this bit length, a router is necessary.
  (IP addresses and subnet masks => Refer to pages 7-5 to 6.)

[Example]



Node A and node B have the same net ID (16 bit lengths) as class B. However, the sub net mask specifies net ID as 24 bits. Therefore, net ID format is different between node A and B, and a router is required for communication between nodes A and B.

Chapter 11. Errors and Correction

11-1 Connection status monitor

This module can set connection status monitor flags in the data memory that indicate which connections are currently effective. This function is enabled by setting the following parameters.
- Enable/disable setting the connection status monitor flags in the data memory
- Assign the connection status monitor flag memory address

If enabled, one byte at the assigned address will contain the connection status monitor flags.

7	6	5	4	3	2	1	0
CN7	CN6	CN5	CN4	CN3	CN2	CN1	CN0

CN 0 to CN 7: Bits that indicate the status of each connection

The bit value varies with the status, as shown below.

1) When the TCP is selected for connection

When the connection is established, the bit will be 1 (ON), and when disconnected, the bit will be 0 (OFF).

2) When the UDP is selected for connection

Since the UDP does not allow for connection or disconnection, when the power is supplied, the bit will be 1 (ON).

If the connection status monitor flags are enabled, the same information will be displayed on LEDs S0 to S7 of the front panel.

● **Connection status monitor flag parameters and settings**

Parameter address	Details
3764	File address of the connection status monitor flag
3765	
3766	File number of the connection status monitor flag
3767	Flag output enabled/disabled 00(H): Do not output 80(H): Output

11-2 Settings for the retransmission timeout time

This module can adjust the maximum, minimum, and initial values used for retransmission timeout time (RTO). However, you should use the default values except for in special situations. If you want to change these values away from their default settings, read and thoroughly understand the details described below and the document RFC793.
When you want to transmit a command from the module, if it does not receive a response after the retransmission timeout time has elapsed, the module will retransmit the command. Although the retransmission timeout time will vary, depending on the interval between transmitting a command and receiving a response (see the Note), the transmission timeout time starts with a specified initial value and will not exceed the maximum or minimum values.

Note: For details about how to calculate the RTO, see the document RFC793. RFC793 (Request For Comment) is a group of documents that describe the details of standardization on the Internet. The RFC is specified by the international association, the Internet Architecture Board (IAB), that sets the standard protocols used on the Internet. The RFC793 documents describe TCP protocols.

● Parameters for setting the retransmission timeout time

Parameter address	Details
3700 to 3703	Minimum value for the retransmission timeout time. Unit: ms. The default setting (0 ms) is assigned when 0 is entered in this parameter.
3704 to 3707	Maximum value for the retransmission timeout time. Unit: ms. The default setting (240,000 ms) is assigned when 0 is entered in this parameter.
3710 to 3713	Initial value for the retransmission timeout time. Unit: ms. The default setting (3,000 ms) is assigned when 0 is entered in this parameter.

11-3 Settings for Keepalive

This module can use a Keepalive function in a TCP connection. The Keepalive is a function that detects a lack of activity when communicating with a host and disconnects its own node's connection to the target node. By using the Keepalive function, this module sends a packet to check the status of the host every time the Keepalive times out. If the module receives a response, it knows that the target node is still operating, and will continue monitoring the node. If the module does not receive a response, it assumes that the target node has stopped operation, and disconnects the connection to the target node.
The parameter set value is a time until sending Keepalive packet. It is not a time until closing the connection.
Time until closing connection varies with circuit condition. Details are as follows.
- Minimum: 4 minutes
- Maximum: 8 minutes + Keepalive timeout time

● Keepalive timeout time parameter

Parameter address	Details
3714 to 3717	Keepalive timeout time Unit: ms (E.g.: If 100 is entered, the timeout time will be 100 ms.) When 0 is entered, the timeout time will be 7,200,000 ms. The default value of the Keepalive timeout time is FFFFFFFF(H). When this value is used, the Keepalive timeout is enabled.

11-4 Setting restart timer

This module has a function that when an application communication (computer link, Send/Rcv) is halted for a certain period, it closes a socket in the module and restart operation.

When this function is enabled, and if the application communication is halted longer than the period specified by the parameter, this module restarts operation. When more than one connections are established, and if all of the connections are halted for the specified period, the module restarts operation. (If one connection is maintained, the module does not restart operation.)

Parameter address	Details
3762	Restart timer setting time Unit: 10 seconds. (Ex.: When "5" is entered, it will be 50 seconds.) When "0" is entered, the restart timer will be invalid. This function can set up to 60 (600 seconds).

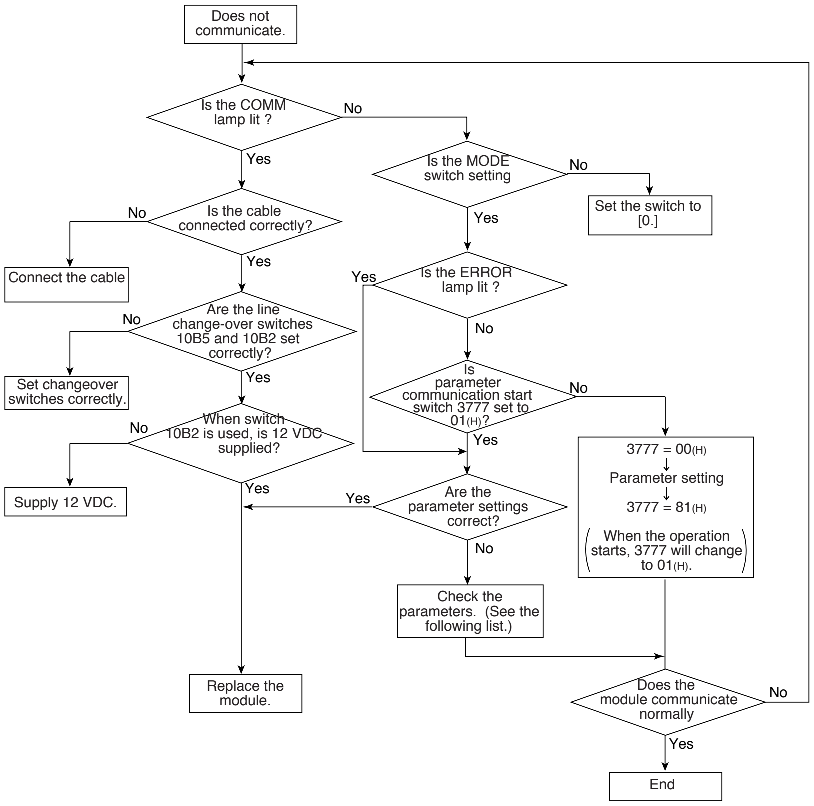
● **Difference between "Keepalive" and "restart timer"**

The Keepalive and restart timer are functions to close ports. However, there are following differences.

	Keepalive	Restart timer
Objective connection	Operate for each port	All ports as package
Close condition	When there is no response to Keepalive packet for each port. (This function does not close ports unless connected.)	When application communication is stopped on all ports. (regardless of existence/non existence of target station)
Operation	Close only the objective ports.	Restart the module.
Meaning of set value	Keepalive packet sending interval. (Not interval until close.)	Time until restart.

11-5 Troubleshooting

When the module does not communicate normally, answer the questions in the following flow chart to identify the problem.


Does not communicate.
Is the COMM lamp lit ?
No
Yes
Is the MODE switch setting
No
Set the switch to [0.]
Yes
Is the cable connected correctly?
No
Connect the cable
Yes
Is the ERROR lamp lit ?
Yes
No
Are the line change-over switches 10B5 and 10B2 set correctly?
No
Set changeover switches correctly.
Yes
Is parameter communication start switch 3777 set to 01(H)?
No
Yes
When switch 10B2 is used, is 12 VDC supplied?
No
Supply 12 VDC.
Yes
3777 = 00(H)
↓
Parameter setting
↓
3777 = 81(H)
(When the operation starts, 3777 will change to 01(H).)
Are the parameter settings correct?
Yes
No
Check the parameters. (See the following list.)
Replace the module.
Does the module communicate normally
No
Yes
End

■ Parameter check (important items to check)

(1) When the computer link function is used

1) When the TCP is used (connection from the host computer)

- Is the IP address correct?
- Is the port No. correct?
- Is the TCP_Passive mode specified?

2) When the UDP is used

- Is the IP address correct?
- Is the port No. correct?
- Is the UDP protocol specified?

(2) When the SEND/RECEIVE function is used

● Instruction initiation station

- Is the IP address correct?
- Is the port No. correct? (6000(H) to 6003(H), or 6008(H)?)
- Is the TCP_Active mode specified?
  (If so, the TCP_Passive mode should be specified in the target station.)
- Is the TCP_Passive mode specified?
  (If so, the TCP_Active mode should be specified in the target station.)
- Is the UDP mode specified?
  (If so, the UDP mode should be specified in the target station.)

● Target station

- Is the IP address correct?
- Is the port No. correct?
- Is the TCP_Passive mode specified?
  (If so, the TCP_Active mode should be specified in the instruction initiation station.)
- Is the TCP_Active mode specified?
  (If so, the TCP_Passive mode should be specified in the instruction initiation station.)
- Is the UDP mode specified?
  (If so, the UDP mode should be specified in the instruction initiation station.)

Chapter 12. Network Parameter

12-1 Table of parameter

The network parameters are set on an EEPROM inside this module.

The following table classifies the parameters by the level of necessity:

A: Parameters necessary even when not using the SEND/RECEIVE functions
B: Parameters necessary when using the SEND/RECEIVE functions
C: Parameters necessary when using the computer link function (standard buffer)
D: Parameters necessary when using the computer link function (ring buffer)
E: Parameters necessary when using the computer link function (expect C and D)
F: Parameters needed for using routing function.

The necessity of making various settings is indicated for each setting using one of the symbols below:

◎: Must always be set
○: When a parameter appears in more than one location, you must set its value in at least one position
△: Set as required
Blank: No need to set

Do not enter a value other than $00_{(H)}$ in the reserved area.
The value when shipped and when initialized (3777 = $08_{(H)}$) is $00_{(H)}$.

Parameter address	Details	Function						Refer-ence page
		A	B	C	D	E	F	
0000 0001 0002 0003	IP address of this module (0003 is on the host ID side)	◎	◎	◎	◎	◎	◎	7-4
0004 to 0007	Subnet mask (when all of the bits are 0, the subnet mask is not used.)	◎	◎	◎	◎	◎	◎	7-6
0010 to 0017	Reserved area							—
0020 to 0021	Value set for the SEND/RECEIVE function monitor timer CH0 TCP Set the monitor timer value using an application, when CH0 of the SEND/RECEIVE function is used for TCP. (Binary value) Unit: 100 ms. If $0000_{(H)}$ is entered, this parameter will be set to the default value (2 minutes).	△						9-7
0022 to 0023	Value set for the SEND/RECEIVE function monitor timer CH0 UDP Set the monitor timer value using an application, when CH0 of the SEND/RECEIVE function is used for UDP. (Binary value) Unit: 100 ms. If $0000_{(H)}$ is entered, this parameter will be set to the default value (1 second).	△						
0024 to 0025	Value set for the SEND/RECEIVE function monitor timer CH1 TCP (the details are the same as for CH0).	△						
0026 to 0027	Value set for the SEND/RECEIVE function monitor timer CH1 UDP (the details are the same as for CH0).	△						
0030 to 0031	Value set for the SEND/RECEIVE function monitor timer CH2 TCP (the details are the same as for CH0).	△						
0032 to 0033	Value set for the SEND/RECEIVE function monitor timer CH2 UDP (the details are the same as for CH0).	△						
0034 to 0035	Value set for the SEND/RECEIVE function monitor timer CH3 TCP (the details are the same as for CH0).	△						
0036 to 0037	Value set for the SEND/RECEIVE function monitor timer CH3 UDP (the details are the same as for CH0).	△						
0040 to 0077	Reserved area							—
0100 to 0103	Settings for connection 0 0100: Open method $00_{(H)}$: TCP_Passive, $80_{(H)}$: TCP_Active, $01_{(H)}$: UDP 0101: $00_{(H)}$ 0102, 0103: Port number of the source station; 0102 as low bit, 0103 as high bit	○	○	○	○	○	○	7-4

Parameter address	Details	Function						Refer-ence page
		A	B	C	D	E	F	
0104 to 0107	Settings for connection 1 (the details are the same as for connection 0)	○	○	○	○	○	○	7-4
0110 to 0113	Settings for connection 2 (the details are the same as for connection 0)							
0114 to 0117	Settings for connection 3 (the details are the same as for connection 0)							
0120 to 0123	Settings for connection 4 (the details are the same as for connection 0)							
0124 to 0127	Settings for connection 5 (the details are the same as for connection 0)							
0130 to 0133	Settings for connection 6 (the details are the same as for connection 0)							
0134 to 0137	Settings for connection 7 (the details are the same as for connection 0)							
0140 to 0377	Reserved area							—
0400	Assign a station number corresponding to the table (for SEND/RECEIVE) Assignment of the relationship between the station number used with the SEND/RECEIVE function and the actual address. If “automatic” is selected, the module treats the target station numbers as node numbers on the IP addresses, and sets the target port to 6010(H). If "individual" is selected, a maximum of 31 types of relationship can be registered concerning the relationship between the station number used with the command and the target address. (Parameters 0410 to 0777 can be used with this selection.) 01(H): Automatic 02(H): Individual	◎	◎					9-2 9-8
0401 to 0407	Reserved area							
0410 to 0417	Station number correspondence table 1. This is effective only when parameter 0400 is 02(H). 0410 Setting 00(H): Not set (the information below will be ignored) 01(H): Set 0411 Target station number 0412, 0413 Target station port number (assign as one word of data (two bytes) in decimal notation) 0414, 0415, 0416, 0417 Target station IP address (0417 is on the host ID side)	○	○					
0420 to 0427	Station number correspondence table 2. This is effective only when parameter 0400 is 02(H).							
0430 to 0437	Station number correspondence table 3. This is effective only when parameter 0400 is 02(H).							
0440 to 0447	Station number correspondence table 4. This is effective only when parameter 0400 is 02(H).							
0450 to 0457	Station number correspondence table 5. This is effective only when parameter 0400 is 02(H).							
0460 to 0467	Station number correspondence table 6. This is effective only when parameter 0400 is 02(H).							
0470 to 0477	Station number correspondence table 7. This is effective only when parameter 0400 is 02(H).							
0500 to 0507	Station number correspondence table 10. This is effective only when parameter 0400 is 02(H).							
0510 to 0517	Station number correspondence table 11. This is effective only when parameter 0400 is 02(H).							
0520 to 0527	Station number correspondence table 12. This is effective only when parameter 0400 is 02(H).							
0530 to 0537	Station number correspondence table 13. This is effective only when parameter 0400 is 02(H).							
0540 to 0547	Station number correspondence table 14. This is effective only when parameter 0400 is 02(H).							
0550 to 0557	Station number correspondence table 15. This is effective only when parameter 0400 is 02(H).							
0560 to 0567	Station number correspondence table 16. This is effective only when parameter 0400 is 02(H).							
0570 to 0577	Station number correspondence table 17. This is effective only when parameter 0400 is 02(H).							
0600 to 0607	Station number correspondence table 20. This is effective only when parameter 0400 is 02(H).							
0610 to 0617	Station number correspondence table 21. This is effective only when parameter 0400 is 02(H).							
0620 to 0627	Station number correspondence table 22. This is effective only when parameter 0400 is 02(H).							
0630 to 0637	Station number correspondence table 23. This is effective only when parameter 0400 is 02(H).							
0640 to 0647	Station number correspondence table 24. This is effective only when parameter 0400 is 02(H).							
0650 to 0657	Station number correspondence table 25. This is effective only when parameter 0400 is 02(H).							
0660 to 0667	Station number correspondence table 26. This is effective only when parameter 0400 is 02(H).							
0670 to 0677	Station number correspondence table 27. This is effective only when parameter 0400 is 02(H).							
0700 to 0707	Station number correspondence table 30. This is effective only when parameter 0400 is 02(H).							
0710 to 0717	Station number correspondence table 31. This is effective only when parameter 0400 is 02(H).							
0720 to 0727	Station number correspondence table 32. This is effective only when parameter 0400 is 02(H).							
0730 to 0737	Station number correspondence table 33. This is effective only when parameter 0400 is 02(H).							
0740 to 0747	Station number correspondence table 34. This is effective only when parameter 0400 is 02(H).							
0750 to 0757	Station number correspondence table 35. This is effective only when parameter 0400 is 02(H).							
0760 to 0767	Station number correspondence table 36. This is effective only when parameter 0400 is 02(H).							
0770 to 0777	Station number correspondence table 37. This is effective only when parameter 0400 is 02(H).							

Parameter address	Details			Function						Refer-ence page
				A	B	C	D	E	F	
1000 to 1007	Information concerning standard buffer 00					○				8-26
		Direction assignment (1007 = 80$_{(H)}$)	**Indirect assignment (1007 = C0$_{(H)}$)**							
	1000 1001	Top file address for the standard buffer storage area	Top file address for the standard buffer information storage area							
	1002	Standard buffer's file number	File number for the standard buffer□s information storage area							
	1003	Not used	Not used							
	1004 1005	Length of the standard buffer (0000$_{(H)}$ as 64 K-bytes)	Not used							
	1006	Not used	Not used							
	1007	Selection of the standard buffer 00$_{(H)}$: Invalid standard buffer 80$_{(H)}$: Direct assignment of the standard buffer C0$_{(H)}$: Indirect assignment of the standard buffer								
1010 to 1017	The information concerning standard buffer 01	Set the same way as the information concerning standard buffer 00								
1020 to 1027	The information concerning standard buffer 02									
1030 to 1037	The information concerning standard buffer 03									
1040 to 1047	The information concerning standard buffer 04									
1050 to1057	The information concerning standard buffer 05									
1060 to 1067	The information concerning standard buffer 06									
1070 to 1077	The information concerning standard buffer 07									
1100 to 1107	The information concerning standard buffer 08									
1110 to 1117	The information concerning standard buffer 09									
1120 to 1127	The information concerning standard buffer 0A									
1130 to 1137	The information concerning standard buffer 0B									
1140 to 1147	The information concerning standard buffer 0C									
1150 to 1157	The information concerning standard buffer 0D									
1160 to 1167	The information concerning standard buffer 0E									
1170 to 1177	The information concerning standard buffer 0F									
1200 to 1207	The information concerning standard buffer 10									
1210 to 1217	The information concerning standard buffer 11									
1220 to 1227	The information concerning standard buffer 12									
1230 to 1237	The information concerning standard buffer 13									
1240 to 1247	The information concerning standard buffer 14									
1250 to 1257	The information concerning standard buffer 15									
1260 to 1267	The information concerning standard buffer 16									
1270 to 1277	The information concerning standard buffer 17									
1300 to 1307	The information concerning standard buffer 18									
1310 to 1317	The information concerning standard buffer 19									
1320 to 1327	The information concerning standard buffer 1A									
1330 to 1337	The information concerning standard buffer 1B									
1340 to 1347	The information concerning standard buffer 1C									
1350 to 1357	The information concerning standard buffer 1D									
1360 to 1367	The information concerning standard buffer 1E									
1370 to 1377	The information concerning standard buffer 1F									

<table>
<tr><th rowspan="2">Parameter address</th><th rowspan="2" colspan="2">Details</th><th colspan="6">Function</th><th rowspan="2">Refer-ence page</th></tr>
<tr><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th></tr>
<tr><td>1400 to 1407</td><td colspan="2">Information concerning ring buffer 00
<table>
<tr><td>1400
1401</td><td>Top file address for the ring buffer information storage area</td></tr>
<tr><td>1402</td><td>File number for the ring buffer information storage area</td></tr>
</table>
Enter a parameter address from 1403 to 1406 if you selected direct assignment (1407 = 80(H))
<table>
<tr><td rowspan="3">1403</td><td colspan="4">Ring buffer data direction</td></tr>
<tr><td>Set value (H)</td><td colspan="3">Details</td></tr>
<tr><td>01
81</td><td colspan="2">Reading data from the control module into this module
Writing data from this module into the control module</td><td>CU = Control module (CPU board)</td></tr>
<tr><td rowspan="6">1404</td><td colspan="4">Ring buffer's top address (file address upper byte)
Set in units of 1 K-bytes</td></tr>
<tr><td>Set value (H)</td><td>File address (8)</td><td>Set value (H)</td><td>File address (8)</td></tr>
<tr><td>00</td><td>000000</td><td>:</td><td>:</td></tr>
<tr><td>04</td><td>002000</td><td>F4</td><td>172000</td></tr>
<tr><td>08</td><td>004000</td><td>F8</td><td>174000</td></tr>
<tr><td>0C</td><td>006000</td><td>FC</td><td>176000</td></tr>
<tr><td>1405</td><td colspan="4">Ring buffer's file number</td></tr>
<tr><td rowspan="7">1406</td><td colspan="4">Upper byte of the ring buffer length</td></tr>
<tr><td>Set value (H)</td><td>Buffer length</td><td>Set value (H)</td><td>Buffer length</td></tr>
<tr><td>00</td><td>64 Kbytes</td><td>10</td><td>4 Kbytes</td></tr>
<tr><td>01</td><td>256 bytes</td><td>20</td><td>8 Kbytes</td></tr>
<tr><td>02</td><td>512 bytes</td><td>40</td><td>16 Kbytes</td></tr>
<tr><td>04</td><td>1 Kbytes</td><td>80</td><td>32 Kbytes</td></tr>
<tr><td>08</td><td>2 Kbytes</td><td colspan="2"></td></tr>
</table>
<table>
<tr><td>1407</td><td>Ring buffer setting
00(H): Invalid ring buffer
80(H): Direct assignment of ring buffer
C0(H): Indirect assignment of ring buffer</td></tr>
</table>
</td><td rowspan="16"></td><td rowspan="16"></td><td rowspan="16"></td><td rowspan="16">○</td><td rowspan="16"></td><td rowspan="16"></td><td rowspan="16">8-39</td></tr>
<tr><td>1410 to 1417</td><td>The information concerning ring buffer 01</td><td rowspan="15">Set the same way as the information concerning ring buffer 00</td></tr>
<tr><td>1420 to 1427</td><td>The information concerning ring buffer 02</td></tr>
<tr><td>1430 to 1437</td><td>The information concerning ring buffer 03</td></tr>
<tr><td>1440 to 1447</td><td>The information concerning ring buffer 04</td></tr>
<tr><td>1450 to 1457</td><td>The information concerning ring buffer 05</td></tr>
<tr><td>1460 to 1467</td><td>The information concerning ring buffer 06</td></tr>
<tr><td>1470 to 1477</td><td>The information concerning ring buffer 07</td></tr>
<tr><td>1500 to 1507</td><td>The information concerning ring buffer 08</td></tr>
<tr><td>1510 to 1517</td><td>The information concerning ring buffer 09</td></tr>
<tr><td>1520 to 1527</td><td>The information concerning ring buffer 0A</td></tr>
<tr><td>1530 to 1537</td><td>The information concerning ring buffer 0B</td></tr>
<tr><td>1540 to 1547</td><td>The information concerning ring buffer 0C</td></tr>
<tr><td>1550 to 1557</td><td>The information concerning ring buffer 0D</td></tr>
<tr><td>1560 to 1567</td><td>The information concerning ring buffer 0E</td></tr>
<tr><td>1570 to 1577</td><td>The information concerning ring buffer 0F</td></tr>
</table>

<table>
<tr><th rowspan="2">Parameter address</th><th rowspan="2" colspan="2">Details</th><th colspan="6">Function</th><th rowspan="2">Refer-ence page</th></tr>
<tr><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th></tr>
<tr><td>1600 to 1607</td><td colspan="2">Routing table 0<table>
<tr><td>1600</td><td>With or without customized routing table settings
00(H): Without customized settings (the information below will be ignored)
01(H): With customized settings</td></tr>
<tr><td>1601</td><td rowspan="3">Target network ID</td></tr>
<tr><td>1602</td></tr>
<tr><td>1603</td></tr>
<tr><td>1604</td><td rowspan="4">IP address in the router corresponding to the network ID (1607 is always the host ID location)</td></tr>
<tr><td>1605</td></tr>
<tr><td>1606</td></tr>
<tr><td>1607</td></tr>
</table></td><td></td><td></td><td></td><td></td><td></td><td>○</td><td>10-2</td></tr>
<tr><td>1610 to 1617</td><td>Routing table 1</td><td rowspan="7">Setup each table the same way as routing table 0.</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1620 to 1627</td><td>Routing table 2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1630 to 1637</td><td>Routing table 3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1640 to 1647</td><td>Routing table 4</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1650 to 1657</td><td>Routing table 5</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1660 to 1667</td><td>Routing table 6</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1670 to 1677</td><td>Routing table 7</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1700 to 1707</td><td colspan="2"><table>
<tr><td>1700</td><td>With and without the default router settings
00(H): Without a default setting (the information below will be invalid)
01(H): With the default settings</td></tr>
<tr><td>1701 to 1703</td><td>Not used</td></tr>
<tr><td>1704</td><td rowspan="4">Default router IP addresses (address 1707 is the host ID)</td></tr>
<tr><td>1705</td></tr>
<tr><td>1706</td></tr>
<tr><td>1707</td></tr>
</table></td><td></td><td></td><td></td><td></td><td></td><td>○</td><td>10-1</td></tr>
<tr><td>1710 to 3657</td><td colspan="2">Reserved area</td><td></td><td></td><td></td><td></td><td></td><td></td><td>—</td></tr>
<tr><td>3660 to 3667</td><td colspan="2">Setting command execution completion information storage area<table>
<tr><td>3660</td><td rowspan="2">Top file address occupied by the command execution completion information</td></tr>
<tr><td>3661</td></tr>
<tr><td>3662</td><td>File number occupied by the completion information for command execution</td></tr>
<tr><td>3663</td><td>Not used</td></tr>
<tr><td>3664</td><td rowspan="2">Size of the command execution completion information (number of bytes)
At least 16 bytes should be assigned. Maximum 64 bytes.</td></tr>
<tr><td>3665</td></tr>
<tr><td>3666</td><td>Not used</td></tr>
<tr><td>3667</td><td>This information is valid when 80(H) is entered for this parameter.</td></tr>
</table></td><td></td><td></td><td>△</td><td>△</td><td>△</td><td></td><td>8-55</td></tr>
<tr><td>3670 to 3677</td><td colspan="2">Reserved area</td><td></td><td></td><td></td><td></td><td></td><td></td><td>—</td></tr>
<tr><td>3700 to 3703</td><td colspan="2">Minimum value for the retransmission timeout time. Unit: ms.
The default setting (0ms) is assigned when 0 is entered in this parameter.</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td rowspan="4">11-2</td></tr>
<tr><td>3704 to 3707</td><td colspan="2">Maximum value for the retransmission timeout time. Unit: ms.
The default setting (240000ms) is assigned when 0 is entered in this parameter.</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td></tr>
<tr><td>3710 to 3713</td><td colspan="2">Initial value for the retransmission timeout time. Unit: ms.
The default setting (3000ms) is assigned when 0 is entered in this parameter.</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td></tr>
<tr><td>3714 to 3717</td><td colspan="2">Keepalive timeout time
Unit: ms (E.g.: If 100 is entered, the timeout time will be 100 ms.)
When 0 is entered, the timeout time will be 7200000 ms. The default value of the Keepalive timeout time is FFFFFFFF(H). When this value is used, the Keepalive timeout is enabled.</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td></tr>
<tr><td>3720 to 3761</td><td colspan="2">Reserved area</td><td></td><td></td><td></td><td></td><td></td><td></td><td>—</td></tr>
<tr><td>3762</td><td colspan="2">Restart timer setting time
Unit: 10 seconds. (Ex.: When "5" is entered, it will be 50 seconds.)
When "0" is entered, the restart timer will be invalid.
This function can set up to 60 (600 seconds).</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td>△</td><td>11-3</td></tr>
<tr><td>3763</td><td colspan="2">Reserved area</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

Parameter address	Details	Function						Refer-ence page
		A	B	C	D	E	F	
3764 to 3767	Setting connection status monitor flag 3764, 3765: File address of the connection status monitor flag 3766: File number of the connection status monitor flag 3767: Flag output 00(H): Do not output, 80(H): Output	△	△	△	△	△	△	11-1
3770 to 3773	Assigning the SEND/RECEIVE function storage area for communication information 3770, 3771: Top file address of the storage area for communication information 3772: File number of the storage area for communication information 3773: This information is valid when 80(H) is entered for this parameter		◎					9-8
3774 to 3775	Reserved area							—
3776	BCC (Block check code) Add 8 bits data to 0000 through 3775 and take the 2□s complement from this value. (This module automatically calculates and stores this value.)							—
3777	Communication start switch 00(H): Stop communication 01(H): Check the parameters, check the BCC, and start operation 08(H): Initialize the parameters (All parameter = 00(H)) 80(H): Check the parameters, create a BCC, write the EEPROM, and stop operation (After writing 80(H) to this address, if it changes to 00(H), the communication has ended normally.) 81(H): Check the parameters, create a BCC, write the EEPROM, and start operation (After writing 81(H) to this address, if it changes to 01(H), the communication has ended normally.)	◎	◎	◎	◎	◎	◎	7-4

Parameter address	Details
4000 to 4005	MAC address (read only) [A MAC address is specified for each machine and is 48 bits long. A unique address is allocated specifically to each machine and cannot be changed. Normally, this address is not used.]
10050	Module ID No. switch output Outputs the setting value (0 to 6) of the module ID No. switch of this module.

12-2 Setting procedure of parameters

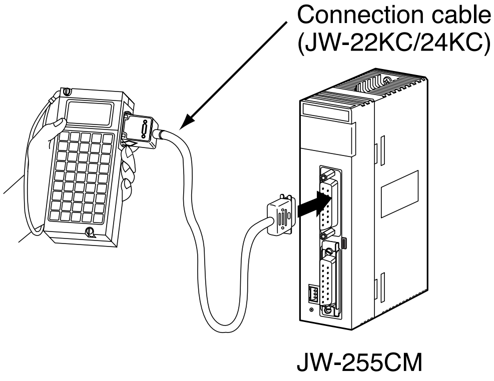
■ Example of settings

- IP address 192.9.200.3
- Connection 0: Use TCP/IP Passive, port number 3000
- Connection 1: Use TCP/IP Active, port number 24576 ($6000_{(H)}$)... For SEND/RECEIVE CH0
- SEND/RECEIVE station number correspondence : Individually set the corresponding relation of station number $13_{(8)}$ of the instruction to the IP address 192.9.200.4 and the port 3001.
- Set the connection status monitoring flag to]0740.

Parameter address	Set value (hex)	Setting details	Details
0000	C0	192	IP address
0001	09	9	
0002	C8	200	
0003	03	3	
:	00	—	—
0100	00	TCP_Passive in used	Connection 0 setting
0101	00		
0102	B8	Port number 3000	
0103	0B		
0104	80	TCP_Active in used	Connection 1 setting
0105	00		
0106	00	Port number 24576 ($6000_{(H)}$)	
0107	60		
:	00		
0400	02	Individual setting	Station No. correspondence table designation
:	00	—	—
0410	01	Setting	Station number correspondence table 1
0411	0D	Instruction station No.13	
0412	B9	Target station port No. 3001	
0413	0B		
0414	C0	Target station IP address 192	
0415	09	9	
0416	C8	200	
0417	04	4	
:	00	—	—
3764	E0	Address]0740 (File address 000740)	Connection status monitoring flag
3765	01		
3766	00	File 00	
3767	80	Flag output enabled	

[1] Setting procedures using the JW-14PG

The parameter setting procedures for the JW-14PG hand-held programmer are shown in the setting example on the previous page.



(1) Connect JW-14PG to the programmer interface connector of this module.

(2) Put the PLC into the program mode.

`クリア CLR` `*` `*` `PROG MODE` `SET 8`

(3) Set to the initial mode (parameter setting).

`クリア CLR` `*` `*` `シフト SHIFT` `INTL DISP` `SET 8` `B 1`

(4) Rewrite the start switch to 00 (stop communication).

`D 3` `7` `7` `7` `モニタ MNTR` `A 0` `書込 ENT`

Screen display of JW-14PG

```
   03775      HEX  00
   03776      HEX  00
I  PARAM.
>03777        HEX  00
```

(5) Set IP address.

`クリア CLR` `モニタ MNTR` `変換 CONV` `変換 CONV`

`B 1` `RESET 9` `C 2` `書込 ENT` `STEP (+)`

`RESET 9` `書込 ENT` `STEP (+)`

`C 2` `A 0` `A 0` `書込 ENT` `STEP (+)`

`D 3` `書込 ENT`

Decimal notation of the parameter value 0000

```
   00001      DCM  009
   00002      DCM  200
I  PARAM.
>00003        DCM  003
```

(6) Set parameters for the connection 0.

`クリア CLR` `B 1` `A 0` `A 0` `モニタ MNTR`

`A 0` `書込 ENT` `STEP (+)`

`A 0` `書込 ENT` `STEP (+)`

`FORCE LNGTH` `変換 CONV` `変換 CONV` `D 3` `A 0` `A 0` `A 0` `書込 ENT` `STEP (+)`

Hexadecimal notation of the parameter 0100

Parameter 0100 = 00

Parameter 0101 = 00

Write 3000 in decimal notation word onto the parameters 0102 and 0103.

```
   00076      D   00000
   00100      D   00000
I  PARAM.
>00102        D   03000
```

(7) Set parameters for the connection 1.

`FORCE LNGTH` `変換 CONV` `変換 CONV` `SET 8` `A 0` `書込 ENT` `STEP (+)`

`A 0` `書込 ENT` `STEP (+)`

`FORCE LNGTH` `6` `A 0` `A 0` `A 0` `書込 ENT`

Parameter 0104 = 80(H)

Parameter 0105 = 00(H)

Write 6000(H) in hexadecimal notation word onto the parameters 0106 and 0107.

```
   00102      H    0BB8
   00104      H    0080
I  PARAM.
>00106        H    6000
```

(8) Set the type of designating the SEND/RECEIVE station number correspondence table.

`クリア CLR` `E 4` `A 0` `A 0` `モニタ MNTR`

`C 2` `書込 ENT`

Hexadecimal notation of the parameter 0400

Parameter 0400 = 02(H)

```
   00376      HEX   00
   00377      HEX   00
I  PARAM.
>00400        HEX   02
```

(9) Set the station number correspondence table 1.

[クリア CLR] [E 4] [B 1] [A 0] [モニタ MNTR]

Hexadecimal notation of the parameter 0410

[B 1] [書込 ENT] [STEP (+)]

Parameter 0410 = 01$_{(H)}$

[変換 CONV] [B 1] [D 3] [書込 ENT] [STEP (+)]

Parameter 0411 = 13$_{(8)}$

```
  00406      D  00000
  00100      D  02817
I  PARAM.
>00412       D  03001
```

[FORCE LNGTH] [変換 CONV] [D 3] [A 0] [A 0] [B 1] [書込 ENT] [STEP (+)]

Write 3001$_{(H)}$ in decimal notation word onto the parameters 0412 and 0413

[FORCE LNGTH] [B 1] [RESET 9] [C 2] [書込 ENT] [STEP (+)]

Parameter 0414 = 192$_{(8)}$

[RESET 9] [書込 ENT] [STEP (+)]

Parameter 0415 = 9

[C 2] [A 0] [A 0] [書込 ENT] [STEP (+)]

Parameter 0416 = 200

[E 4] [書込 ENT]

Parameter 0417 = 4

```
  00415      DCM  009
  00416      DCM  200
I  PARAM.
>00417       DCM  004
```

(10) Set the connection status monitoring flag.

[クリア CLR] [D 3] [7] [6] [E 4] [モニタ MNTR] [FORCE LNGTH] [変換 CONV]

Octal notation word of the parameters 3764 and 3765.

[7] [E 4] [A 0] [書込 ENT] [STEP (+)]

Write 740 in octal notation.

[FORCE LNGTH] [変換 CONV] [変換 CONV] [変換 CONV] [A 0] [書込 ENT] [STEP (+)]

Parameter 3766 = 00

[SET 8] [A 0] [書込 ENT]

Parameter 3767 = 80$_{(H)}$

```
  03765      HEX  01
  03766      HEX  00
I  PARAM.
>03767       HEX  80
```

(11) Write onto the EEPROM and start.

[クリア CLR] [D 3] [7] [7] [7] [モニタ MNTR] [SET 8] [B 1] [書込 ENT]

```
  03775      HEX  00
  03776      HEX  EC
I  PARAM.
>03777       HEX  81
```

Writing 81 gets the data written on the EEPROM, and then starts operation. It will take about 5 seconds.
Once the operation starts, the LED for COMM will illuminate.

(12) Get the PLC in operating condition.

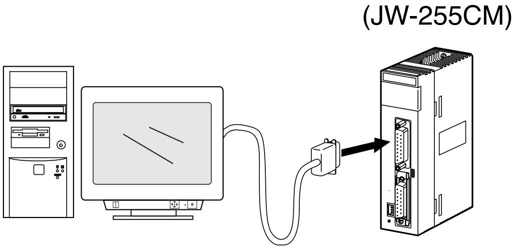
[クリア CLR] [∗] [∗] [MNTR MODE] [SET 8]

```

M    00000
>
```

[2] Setting procedures using the JW-100SP

This section describes rough procedures to set parameters, write in, read from, and verify data in the module using the ladder logic programming software JW-100SP (Windows version).
As for operation details, see the "JW-100SP instruction manual."



Connection cable (JW-22KC/24KC)
+
Communication adapter (JW-100SA)

(1) Parameter setting

1) Click the [Tools] and then the [Option Parameter] on the menu bar, and display the [option parameter tool] window.
2) On the [Option Parameter Tool] window, click the [File] -> the [New] on the menu bar, and display the [Create] dialog box.
3) On the [Create] dialog box, select the [EtherNet] and click the [OK] button to display the [Option Parameter Tool: EtherNet] window.
4) On the [Option Parameter Tool: EtherNet] window, enter values to parameter address.

(2) Write parameter to the module

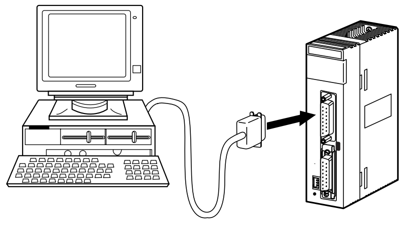
1) Connect a personal computer to the module.
2) On the [Option Parameter Tool: EtherNet] window, click the [PC] -> the [PC Transfer] -> the [Write], the parameters are written to the module.

- To read or verify parameters, click the [PC] -> the [PC Transfer] -> the [Verify].

[3] Setting procedures using the JW-52SP/92SP

(JW-255CM)

Described below is an the outline of how to set, write, and store the JW-255CM parameters, using the JW-52SP ladder software (for DOS/V personal computer and the JW-92SP (for PC-98 personal computer).
For the details about the operation, see the instruction manual for the ladder software.



Connection cable (JW-22KC/24KC)
+
Communication adapter
(supplied with JW-92SP)
Converter
(supplied with JW-52SP)

Select the PLC model whose parameters you want to set.
[Main menu] ➡ 1: Program edit ➡ 1: Model selection ➡ 2: Model name setting as follows ➡ 0: Executes

- JW-255CM/25TCM (PLC model)
  - When install on the JW20H --- JW21, JW22
  - When install on the JW30H --- JW31H/H1, JW32H/H1, JW33H/H1, JW33H2/H3
- Z-339J
  - When install on the J-board (Z300) --- JW22
  - When install on the J-board (Z500) --- JW32H/H1

(1) Parameter settings
[Main menu] ➡ 4: Tool transfer ➡ 8: Other parameters ➡ 1: Set parameters ➡ After each parameter is set, press F10 (Write) ➡ F7 (Quit)

(2) Writing the parameters to this module
Connect the personal computer to this module.
[Main menu] ➡ 7: PLC transfer ➡ 2: Write ➡ 6: Remote station PARAM. ➡ 7: PLC stop (Place the CU in the stop mode.) ➡ 2: Execution stop (Stop this module's operation.) ➡ 1: Parameter writing (Transmit the parameters to the module.) ➡ 5: Start: write the EEPROM (Write the transmitted parameters to the EEPROM in this module and start its operation.) ➡ 6: PLC run (Place the CU in the run mode.)

CU = Control module (CPU board)

Note: After the EEPROM writing operation process is executed, leave more than two seconds before starting the next process.

(3) Reading the parameters from this module
Connect the personal computer to the this module.
[Main menu] ➡ 7: PLC transfer ➡ 3: Read ➡ 6: Remote master station PARAM. ➡ 7: PLC stop (Place the CU in the stop mode.) ➡ 2: Execution stop (Stop the module operation.) ➡ 1: Read out PARAM (Transmit the parameters from this module.) ➡ 4: Operation start: read (Reading operation starts.) ➡ 6: PLC run (Place the CU in the run mode.)

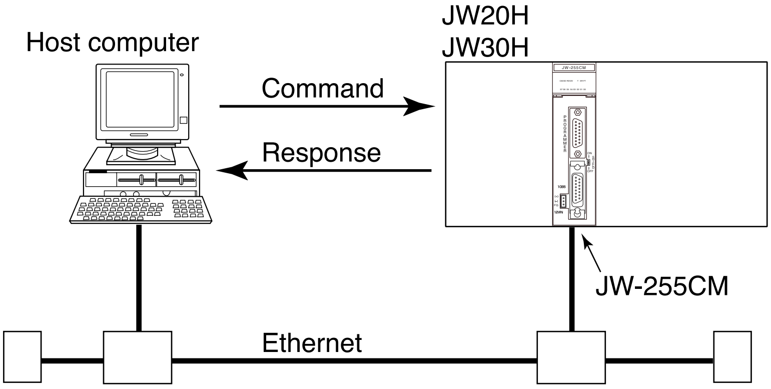
(4) Recording the parameters on a floppy disc
[Main menu] ➡ 6: FD transfer ➡ 1: Write ➡ 9: Master station PARAM. ➡ Yes ➡ F1 (Drive) ➡ Specify drive/directory ➡ Enter a file name ➡ Enter comment ➡ 0: Execute

(5) Restoring the parameters from a floppy disc
[Main menu] ➡ 6: FD transfer ➡ 2: Load ➡ F1 (Drive) ➡ Specify drive/directory ➡ Enter a file name using the space key ➡ 0: Execute

Chapter 13. Sample Program

This chapter describes an example of a program for the host computer (using the computer link function).

● **Example of connection**



Shown above is an example of communication using the TCP method of communication.
By entering a host name and port number for the JW-255CM/25TCM and Z-339J, the host computer can establish contact with this module.
Assume that the host side port number is 4000.
Once the connection is made, the host computer sends, two-byte read commands, starting from 09002, five times and then disconnects.
This example uses a WIN socket interface. However, be careful because this interface may have a different function name and argument when called by another processing group.

Description of the program steps

388 to 402 Normally, each node is controlled by name (host name), over the Ethernet. The network has a table of corresponding host names and IP addresses. These lines are used to get this module's IP address based on the host name and port number entered. The "gethostbyname" function is used to obtain the IP address from the host name. In order to use this function, the relationship between the host name and IP address of this module must first be registered on the host computer.

405 Establish a connection with this module.

  141 Creates a TCP socket. Use the "socket" function to create a socket.

  148 to 153 Both IP addresses and port numbers are stored as address structure. The host computer stores target station (this module) IP address and port number, as well as the port number of the host computer, into the address structure.

  155 to 159 Assigns 4000 the port number to be used by the computer. Use the "bind" function to make this assignment. (Refer to next page.)

  161 to 164 Proceeds to connect the host to the target station. Use the "connect" function.

412 Executes computer link communication.

  358 to 360 Sets a command. The command here is "read two bytes, starting from register 09002."

  365 Sends the command to this module. Use the "send" function.

  371 Receives the response from this module.

  331 Sets the time-out value to one second.

  333 Checks whether the host computer is receiving data. Use the "select" function to check.

  336 If the data was received, the host computer will store the data in the receive buffer using the "recv" function.
  The host computer will repeat this operation five times.

414 Disconnects the communication link.

  176 Disconnects the communication link using the "shutdown" function.

  177 Closes the socket using the "soclose" function.

Note: Setting the port number for the host side

When you want to establish a communication link from a host using the TCP method, the host needs to create a socket for the target to connect to.

There are two methods to set a port number, as follows:

a) Assign a port number to a socket using the “bind” function.

b) Instead of using the "bind" function, let the system make the assignment. (The port number will be different each time a connection is made.)

If you use assignment method b) and terminate the connection from the host side abnormally (such as shutting off the power without going through the normal quitting procedures), the host may not be able to reconnect the next time you start the computer. This is because this module maintains the connection, even if the host performs abnormal disconnection. In this condition, this module will refuse a request for another connection.

If you use assignment method a), this module can recognize that the another request for connection is being made by the same port number. Therefore, the module can reset itself. Then, it will possible to make a new connection. In order to prevent problems after an abnormal disconnection, we recommend that you use assignment method a).

```
/****************************************************************
 *                       A program example                       *
 * Connect to a target station using TCP method, and send two-byte read  *
 * commands five times, starting at register 09002. Then, disconnect.     *
 *                                                               *
 * This example does not have a complete set of error functions. If you  *
 * call it from some of other processing groups, the function name for the  *
 * socket interface may be different.                             *
 *                                                               *
 ****************************************************************/


#include <stdio.h>
#include <conio.h>
#include <ctype.h>
#include <time.h>
#include <stdlib.h>
#include <errno.h>
#include "netdb.h"
#include "sys\ib_types.h"
#include "sys\ib_time.h"
#include "sys\ib_errno.h"
#include "sys\socket.h"
#include "netinet\in.h"

#define NUMSOCKMAX 4
#define BUFLEN  1024
#define HEADLEN 40

char theader[HEADLEN] = {0,0,0,0,0,0,0,
                    0,0,0,
                    0,0,0, 0,0,0, 0,0,0, 0,0,0, 0,0,0,
                    0,0,0, 0,0,0, 0,0,0, 0,0,0, 0,0,0
                    };

struct SENDFRAME {
        char header[HEADLEN];
        char cl_command_frame[BUFLEN-HEADLEN];
};

struct RECEIVEFRAME {
        char header[HEADLEN];
        char cl_command_frame[BUFLEN-HEADLEN];
};


struct SBUF {
           char buf[BUFLEN];
};

struct RBUF {
           char buf[BUFLEN];
};

union SEND {
           struct SENDFRAME s_upper;
           struct SBUF s_socket;
} sendbuf;


union RECEIVE {
        struct RECEIVEFRAME r_upper;
        struct RBUF r_socket;
} receivebuf;

struct {
        int errno;
        char *errmsg;
} errlist [] = {
        0,              "No error",
        EIO,            "I/O error",
        ENOMEM,         "No memory",
        ENODEV,         "No such adaptor",
        EINVAL,         "Invalid command ar argument",
        EMFILE,         "Too many endpoints or connections",
        EMSGSIZE,       "Too large message",
        EOPNOTSUPP,     "Operation is not supported",
        EADDRINUSE,     "Address is already used",
        ENETDOWN,       "Network is down",
        EHOSTUNREACH,   "Destination is unreachable",
        ENETUNREACH,    "Network is unreachable",
        ECONNABORTED,   "Connection is aborted",
        ECONNRESET,     "Connection is reset",
        ESHUTDOWN,      "Connection shutdown",
        ETIMEDOUT,      "Operation timeout",
        ECONNREFUSED,   "Connection refused"
};




void so_perror(char *, int);
int comopen(unsigned long, int);
void comclose(int);
char a2b_1c(char);
int a2b(char *, char *);
int ascbin(char *, char *);
char b2a_1c(char);
void b2a(char, char *);
void bin2asc(char *, char *, int);
void set_command(char *, int);
int get_command_default(char *, char *);
void disp_response(char *, int);
void disp_command(char *);
int receive_response(int);
int communication(int);



/********************************************************
 *                  Error display routine                   *
 ********************************************************/


```

```
void so_perror(char *str, int err)
{
        int i;

        for(i = 0; i < 16; ++i)
                if(err == errlist[i].errno)
                        break;
        if(i < 16)
                printf("%s: %s            \n", str, errlist[i].errmsg);
        else
                printf("%s: unknown error\n");
}



/*******************************************************
 *                    Establish a connection                 *
 *******************************************************/


int comopen(unsigned long ip, int port)
{
struct sockaddr_in myaddr;
struct sockaddr_in youraddr;
int s;

        s = socket(PF_INET, SOCK_STREAM, 0); /* Create a socket (TCP) */

        if(s == -1) {
                so_perror("socket", errno);
                soclose(s);
                return(-1);
        }
        youraddr.sin_family = AF_INET;          /* Store the target station address in the address structure */
        youraddr.sin_port = port;
        youraddr.sin_addr.s_addr = ip;

        myaddr.sin_family = AF_INET;
        myaddr.sin_port = htons(4000);
                                      /* Use 4000 for the source station port number */
        if(bind(s, (struct sockaddr *)&myaddr, sizeof (myaddr)) < 0) {
                so_perror("bind", errno);
                soclose(s);
                return(-2);
        }
                                                /* Make the connect */
        if(connect (s, (struct sockaddr *) & youraddr, sizeof (youraddr)) < 0) {
                so_perror("connect", errno);
                soclose(s);
                return(-2);
        }
        return(s);
}


/*******************************************************
 *                    Close the connection                   *
```

```
  *******************************************************/

void comclose(int s)
{
        shutdown(s, 1);
        soclose(s);
}



/*******************************************************
 *   Convert one hexadecimal ASCII character into binary     *
 *******************************************************/

char a2b_1c(char data)
{
        return(isdigit(data) ? data - '0' :
                (isupper(data) ? data - 'A' + 10 : data - 'a' + 10));
}



/*******************************************************
 *   Convert two hexadecimal ASCII characters into binary    *
 *******************************************************/

int a2b(char *ascbuf, char *bindata)
{
        if(isxdigit(ascbuf[0]) && isxdigit(ascbuf[1])) {
                *bindata = a2b_1c(ascbuf[0]) * 16 + a2b_1c(ascbuf[1]);
                return(0);
        } else
                return(-1);
}


/*******************************************************
 *          Convert ASCII character strings into binary           *
 *******************************************************/

int asc2bin(char *ascbuf, char *binbuf)
{
int a, i, j;
        for(i = 0, j = 0; ascbuf[i] != 0; j++, i++) {
                a = a2b(&ascbuf[i], &binbuf[j]);
                if(a < 0)
                        return(a);
                i++;
        }
        return(j);
}


/*******************************************************
 * Convert 4 bits binary into hexadecimal ASCII character   *
 *******************************************************/

```

```
char b2a_1c(char data)
{
        return((data < 10) ? data + '0' : data + 'A' - 10);
}


/*********************************************************
 *   Convert 8 bits binary into 2 hexadecimal ASCII characters    *
 *********************************************************/

void b2a(char bindata, char *ascbuf)
{
char a;
        a = (bindata >> 4) & 0xf;
        ascbuf[0] = b2a_1c(a);
        a = bindata & 0xf;
        ascbuf[1] = b2a_1c(a);
}

/*********************************************************
 *        Convert binary code into an ASCII character string          *
 *********************************************************/

void bin2asc(char *binbuf, char *ascbuf, int len)
{
int i, j;

        for(j = 0, i = 0 ; i < len; i++) {
                b2a(binbuf[i], &ascbuf[j]);
                j += 2;
        }

}

/*********************************************************
 *                  Place a command in the send buffer                 *
 *********************************************************/

void set_command(char *cbuf, int len)
{
int i;
        for(i = 0; i < HEADLEN; i++)
                sendbuf.s_upper.header[i] = theader[i];

        for(i = 0; i < len; i++)
                sendbuf.s_upper.cl_command_frame[i] = cbuf[i];
}

/*********************************************************
 *                          Receive the command                          *
 *********************************************************/

int get_command_default(char *kbuf, char *cbuf)
{
char cntbuf[32];
char intbuf[32];

```

```
	return(asc2bin(kbuf, cbuf));
}


/********************************************************
 *                   Display response                   *
 ********************************************************/

void disp_response(char *buf, int len)
{
int i;
	bin2asc(receivebuf.r_upper.cl_command_frame, buf, len);
	buf[2*len] = 0;
	printf("response = ");
	puts(buf);

}


/********************************************************
 *                   Display command                    *
 ********************************************************/

void disp_command(char *buf)
{
	printf("command = ");
	puts(buf);
}


/********************************************************
 *                   Receive response                   *
 ********************************************************/

int receive_response(int s)
{
fd_set readfds;
struct timeval tout;
int rlen, n;
char cbuf[1024];
char dbuf[1024];


	FD_ZERO(&readfds);
	FD_SET(s, &readfds);
	tout.tv_sec = 1;		/* Set the time-out value to one second */

	n = select(32, &readfds, NULL, NULL, &tout);
	if(n > 0) {			/*If received, OK*/
		if(FD_ISSET(s, &readfds))
			rlen = recv(s, receivebuf.r_socket.buf, BUFLEN, 0);
	} else {
		so_perror("select", errno);
		return(-1);
	}
	rlen -= HEADLEN;
	disp_response(dbuf, rlen);
```

```
        return(0);
}


/*********************************************************
 *                    Communication process               *
 *********************************************************/

int communication(int s)
{
char kbuf[1024] = "4700240002080200";   /* Command */
char cbuf[1024];                /* Command (binary) */
int data_len, r;
unsigned int i;

        data_len = get_command_default(kbuf, cbuf);
        set_command(cbuf, data_len);
        data_len += HEADLEN;

        for (i = 0; i < 5; i++) {
                disp_command(kbuf);
                                /* Send to the target station */
                r = send(s, sendbuf.s_socket.buf, data_len, 0);
                if (r != data_len) {    /* If the data cannot be sent normally, “error” is returned. */
                        so_perror("send", errno);
                        return(-1);
                }
                                /* Receive the response */
                if (receive_response(s) < 0)
                        return(-1);
        }
}

/*********************************************************
 *                                                       *
 *                        Main routine                   *
 *                                                       *
 *********************************************************/

void main(int argc, char *argv[])
{
struct hostent *hp;     /* Define the name structure */
unsigned long ipaddr;   /* IP address */
int portno;             /* Port number */
int s;                  /* Socket identifier */

        if (argc < 2) {
                printf("CLTEST name port\n");
                printf("      name : Target station name\n");
                printf("      port : Target port number\n");
                exit(1);
        }
                                /* Get IP address from the name */
        hp = gethostbyname(argv[1]);
        if (hp == NULL) {
                printf("%s: Undefined host\n",argv[1]);
                exit(1);
```

```
    }
    ipaddr = *(unsigned long *)hp->h_addr;
    portno = htons(atoi(argv[2]));

                        /* Establish a connection */
    if (comopen(ipaddr, portno) < 0)
        exit(1);

    printf("Complete connection to target station = %s\n", argv[1]);
    printf("     Complete connection to port = %s\n", argv[2]);

                        /* Communication processing */
    communication(s);
                        /* Disconnect */
    comclose(s);

}
```

Chapter 14. Specifications

14-1 JW-225CM/25TCM

[1] General specifications

Item	Specifications	
	JW-255CM	JW-25TCM
PLC model	JW20H/30H (install on a basic rack panel)	
Storage temperature	-20 to +70˚C	
Ambient operation temperature	0 to +55˚C	
Ambient humidity	35 to 90 %RH (Non-condensing)	
Vibration proof	JIS C 0911 or equivalent. Vibration width: 0.15 mm (10 to 58 Hz), 9.8 m/s^2(58 to 150Hz) (Two hour each in X , Y, and Z direction)	
Shock proof	JIS C 0912 or equivalent 98 m/s^2 (three times each in X, Y, and Z direction)	
Internal current consumption (5 VDC)	370 mA	350 mA
External power supply capacity	12 VDC ±5% 0.5 A	None
Ethernet interface	10BASE5 AUI (D-sub,15-pin)	10BASE-T (RJ-45 connector)
Programmer interface	D-sub, 15-pin	D-sub, 15-pin
Weight	Approx. 240 g	Approx. 180 g
Accessories	Cable (1) , Instruction manual (1)	Instruction manual (1)

[2] Communication specifications

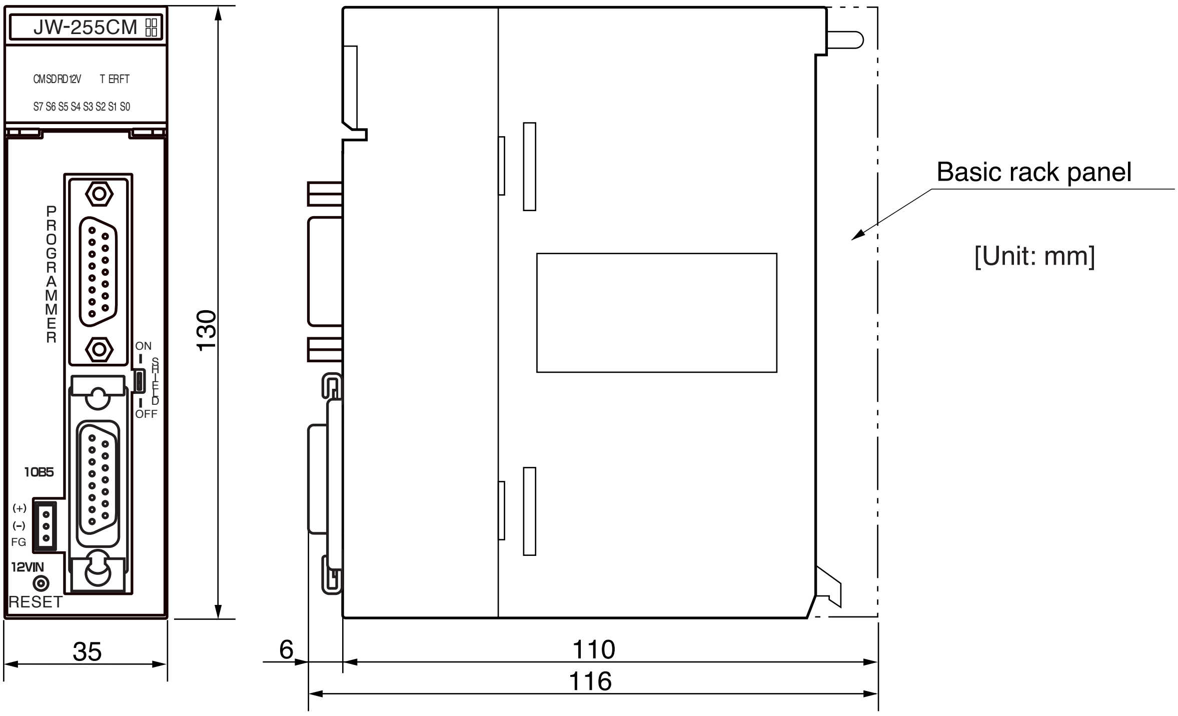
Item		Specifications	
		JW-255CM	JW-25TCM
Connection with network		10BASE5	10BASE-T
Transfer speed		10 Mbps	
Physical topology		Bus	Star
Transmission device		50 ohms coaxial cable	10BASE-T twist pair cable
Transmission method		Baseband	
Max. transfer length		500 m/segment 2.5 km/network *Note 1	100 m/segment 500 m/network *Note 2
Station interval		An integer multiplied value of 2.5 m (10BASE5)	
Max. No. of stations		100 sets/segment (10BASE5)	
Protocol structure	Application	Sharp computer link/original command	
	Transport	TCP/UDP	
	Network	IP (ARP)	
	Data link	Ethernet V2	
No. of connections		8	
Application		Computer link functtion, SEND/RECEIVE functions	

*Note 1: Max. transmission distance between stations while connecting multi-segments using repeaters.

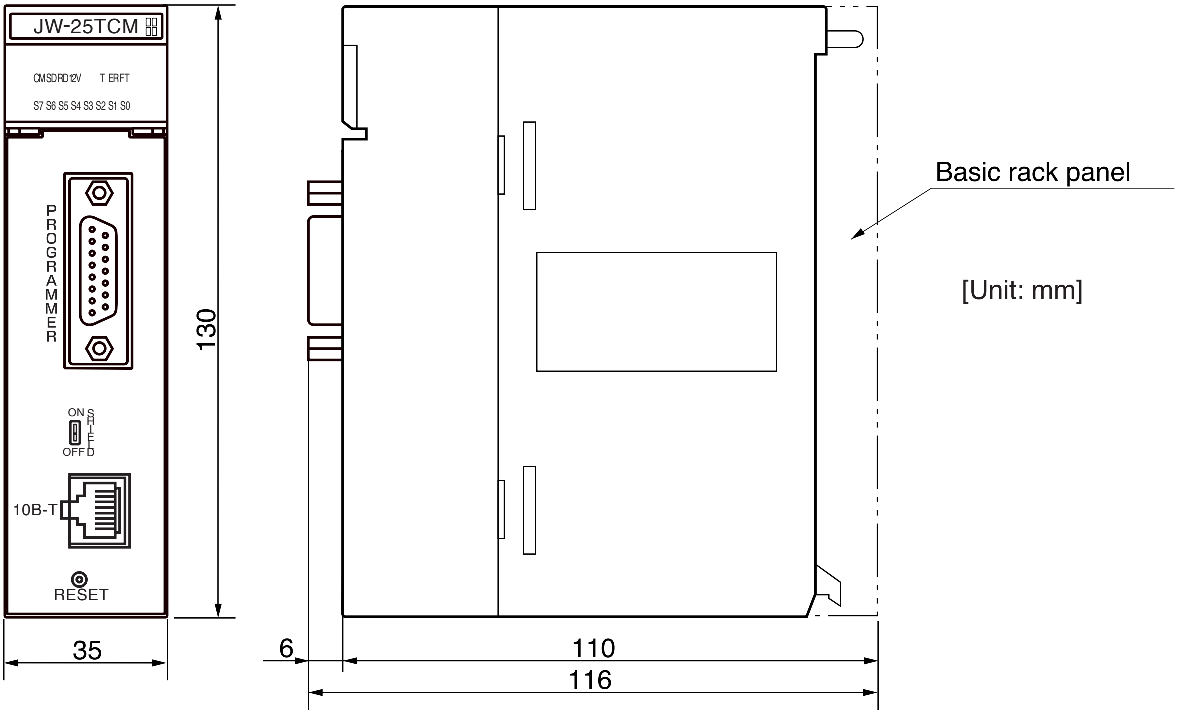
*Note 2: Max. transmission distance between stations while connecting multi-segments using hubs.

[3] Outside dimensions

■ JW-255CM



■ JW-25TCM



14-2 Z-339J

[1] General specifications

Item	Specifications
PLC model	J-board Z300/500 series
Storage temperature	-20 to +70˚C
Ambient operation temperature	0 to +55˚C
Ambient humidity	35 to 90 %RH (Non-condensing)
Vibration proof	JIS C 0911 or equivalent. Vibration width: 0.15 mm (10 to 58 Hz), 9.8 m/s^2 (58 to 150Hz) (Two hour each in X , Y, and Z direction)
Shock proof	JIS C 0912 or equivalent 98 m/s^2 (three times each in X, Y, and Z direction)
Internal current consumption (5 VDC)	380 mA
External power supply capacity	12 VDC ±5% 0.5 A (need only when 10BASE5 is used.)
Ethernet interface	10BASE5 AUI (D-sub,15-pin), 10BASE-T (RJ-45 connector)
Programmer interface	D-sub,15-pin
Number of boards mounted	Z300 series (Z-311J/312J) --- Max. 2 Z500 series (Z-511J/512J) --- Max. 2
Weight	Approx. 180 g
Accessories	Cable (1), board securing bosses (20 mm + 6 mm protruded part) (4) Screws (Semus M3 x 6 mm) (4), Instruction manual (1)

[2] Communication specifications

Item		Specifications
Connection with network		Either of 10BASE5 or 10BASE-T
Transfer speed		10 Mbps
Physical topology		Bus (10BASE5)/star (10BASE-T)
Transmission device		50 ohms yellow cable (10BASE5), Twisted pair cable (10BASE-T)
Transmission method		Baseband
Max. No. of transfer length		10BASE5: 500 m/segment 2.5 km/network *Note 1 10BASE-T: 100 m/segment 500 m/network *Note 2
Station interval		An integer multiplied value of 2.5 m (10BASE5)
Max. No. of stations		100 sets/segment (10BASE5)
Protocol structure	Application	Sharp computer link/original command
	Transport	TCP/UDP
	Network	IP (ARP)
	Data link	Ethernet V2
No. of connections		8
Application		Computer link function, SEND/RECEIVE functions

*Note 1: Max. transmission distance between stations while connecting multi-segments using repeaters.

*Note 2: Max. transmission distance between stations while connecting multi-segments using hubs.

[3] Outside dimensions

=> Refer to page 5-2.

Alphabetical Index

[A]
Address allocation of I/O relays 5-4
An example using the ring buffer 8-49
ARP 1-1

[B]
BLOC 8-3
Buffer address (BAH) 8-32
Buffer file number (BF) 8-33
Buffer length (BL) 8-33

[C]
Channel number 9-1
Class (A, B, and C) 7-5
Command execution completion information 8-55
Commands for standard buffers 8-24
Commands for the ring buffer 8-32, 8-41
Communication format 8-1
Communication information storage area 9-10
Communication specifications (JW-255CM/25TCM) 14-2
Communication specifications (Z-339J) 14-5
Computer link commands 8-1
Computer link error code table 8-54
Computer link function 7-1
Connecting the transceiver cable (JW-255CM) 6-2
Connecting the transceiver cable (Z-339J) 6-5
Connection status monitor flag parameters and settings 11-1
Connection status monitor 11-1
Connection 7-7
Create a customized routing table 10-2
Create a default router 10-1

[D]
DADR 8-2
Data memory starting system 7-2, 9-8
Default setting of the router parameter 10-1
Direction (DIR) 8-33
DSEG 8-2

[E]
Error processing when accessing ring buffers 8-40
Error processing when accessing standard buffers 8-27
Ethernet 1-1

Execution condition 8-3

[G]
General specifications (JW-255CM/25TCM) 14-1
General specifications (Z-339J) 14-4

[H]
Host ID 7-5
How to use the ring buffer 8-32

[I]
ICMP 1-1
Instruction system 7-2, 9-1
IP address 7-5
IP 1-1

[M]
Maximum number of board to install 5-3
Memory address expression format 8-2

[N]
Network ID 7-5
Network parameter settings 7-3
Network parameter 12-1

[O]
Operation of the ring buffer 8-35
Outside dimensions (JW-255CM/25TCM) 14-3
Outside dimensions (Z-339J) 5-2

[P]
PADR 8-2
Parameter for setting the monitor timer 9-7
Parameters for creating the routing table 10-2
Parameters for setting the communication information storage area 9-8
Parameters for setting the retransmission timeout time 11-2
Parameters for "automatic/individual" registration (Data memory starting system) 9-9
Parameters for "automatic/individual" registration (Instruction system) 9-2
PLC operation status 8-3
Port No. 7-7
PSEG 8-2

[R]
Read pointer (RP) 8-32
RECEIVE 9-5
Ring buffer information storage area 8-40

Routing function 10-1

[S]

SADR 8-3
Sample program 13-1
SEND 9-3
SEND/RECEIVE function 7-2, 9-1
SEND/RECEIVE instructions operation 9-3
Setting procedure of parameters 12-7
Setting procedure using the JW-100SP 12-10
Setting procedures using the JW-14PG 12-8
Setting procedures using the JW-52SP/92SP 12-11
Setting the parameters (Command execution completion information) 8-55
Setting the parameters (Data memory starting system) 9-8
Setting the parameters (Ring buffer) 8-39
Setting the parameters (Standard buffers) 8-26
Settings for Keepalive 11-2
Setting restart timer 11-3
Socket 7-7
Source/destination address and channel 9-1
Specifications 14-1
Standard buffer information storage area 8-27
Standard buffers 8-24
Subnet mask 7-6

[T]

Table of commands 8-4
Table of parameter 12-1
TADR 8-3
Target station number 9-1
TCP 1-1, 7-5, 8-56
TCP active 7-3
TCP connection and disconnection 9-2
TCP passive 7-3
Time interval required for communication 8-56
Troubleshooting 11-4
Two-layer communication with satellite net 8-57

[U]

UDP 1-1, 7-3, 7-5, 8-56

[W]

Wiring the power source (JW-255CM) 6-3
Writing the power source (Z-339J) 6-6
Write enable mode 8-3
Write pointer (WP) 8-32