SHARP

Version 1.0
Produced in April. 2000

**Sharp Programmable Controller**

NEW **Satellite** JW50H/70H/100H

**Module name**

# Ethernet module JW-51CM

## User's Manual

Thank you for purchasing the Ethernet module (JW-51CM) for the SHARP programmable controller JW50H/70H/100H.
Read this manual thoroughly to completely familiarize yourself with the operation.
Keep this manual for future reference. We are confident that this manual will be helpful whenever you encounter a problem.
Make sure to read the following manuals for JW-51CM and JW50H/70H/100H together with this manual.

**JW-51CM** — **User's manual (this manual)**

JW50H/70H/100H control module — User's manual - Hardware version
Programming manual

**Note**

- This manual is written with the utmost care.Should you have any questions or inquires, please feel free to contact one of our dealers, or our service department.
- 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 durung 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.

# Ethernet module JW-51CM

■ User's Manual

# Table of contents

# Chapter 1: Outline

The JW-51CM Ethernet module (or just “this module”) is an interface module used to connect the JW50H/70H/100H programmable controller (or “PC”) to an *Ethernet network. Installing this module in the JW50H/70H/100H will allow you to exchange data between host computers on Ethernet networks and LANs.

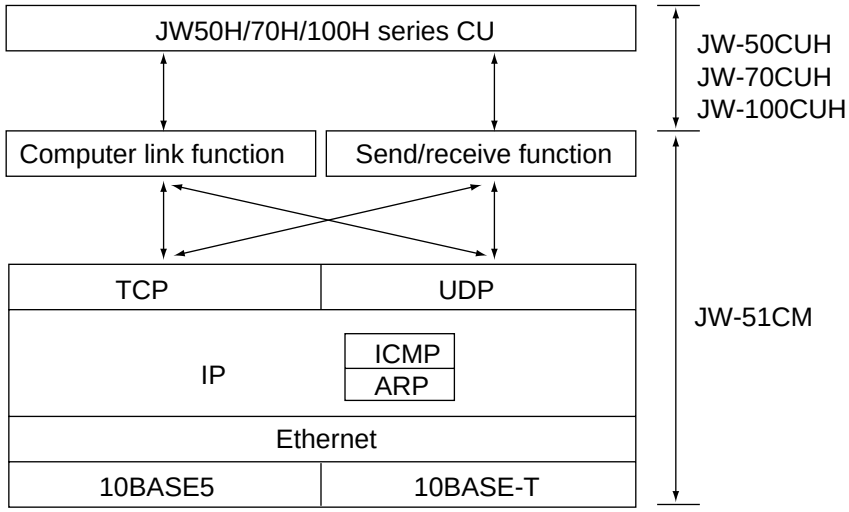
* Ethernet is a trademark of the Xerox Corporation.



**(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 PC computer link functions. It allows the host computer to access PCs.
③ Data communication is possible between host computers in an Ethernet network and PCs in a satellite network spanning two hierarchic layers.
④ This module supports the 10BASE5 and 10BASE-T interface. (Use either of the two.)
⑤ The JW-51CM is equipped with eight individual ports. Each port can make a separate connection.
⑥ Communication between PCs is possible by using the send/receive functions.
⑦ Using the subnet mask routing function, the JW-51CM can communicate with a large network system using a router.

**(2) Software system**




- **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, the JW-51CM will not retransmit the data, as is the case in the TCP mode.
- **IP (Internet Protocol)**
  In this method, the JW-51CM 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**
  The JW-51CM can handle the frame format of Ethernet version 2.

# Chapter 2: Handling Precautions



**(1) Installation**

- Do not install or store the JW-51CM in the following conditions.
  ① Direct sunlight
  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
- Prior to installing or detaching the JW50H/70H/100H, make sure to turn OFF the power supply to the PCs.
- All screws must be tightened firmly.
- 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.
- Mount the transceivers on electrically insulated objects, such as a wooden mounting block.

**(2) Wiring**

- Separate the data transmission cables from power cables (less than 60 cm).
- Do not run cables near any noise generating source.
- Terminating resistances are required for both ends of the coaxial cable. Make sure to install the specified terminating resistances.
- Use the 10BASE-T cable with a shield when installing a 10BASE-T system.
- Use an isolation shield transformer for a power supply to the hub.
- We recommend keeping the transceiver cable to 2 m or less.

**(3) Treatment**

- 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 the JW-51CM to avoid a possible hazard. Otherwise, it may be a cause of machine trouble.
- When a trouble or abnormal condition such as overheat, fume, or smoke is met, stop the operation immediately, and call your dealer or our service department.

**(4) Static electricity**

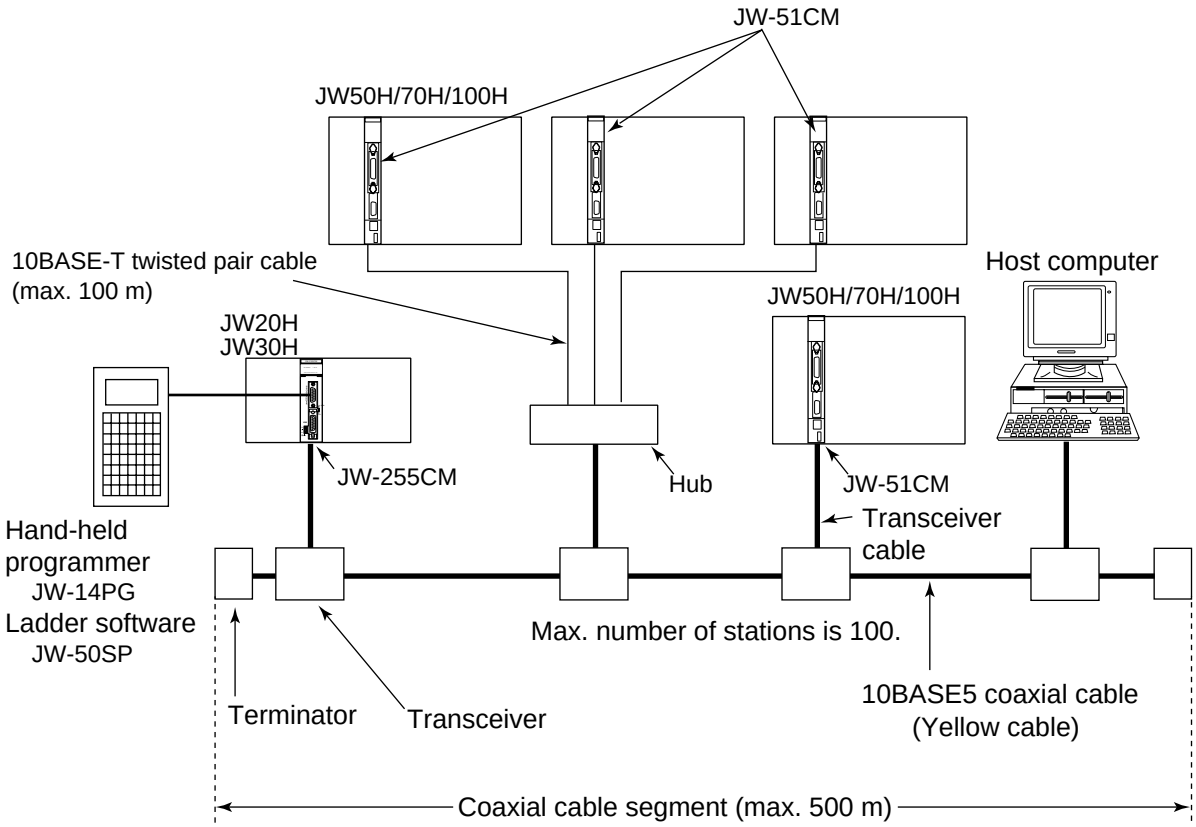
- In extremely dry circumstances, the human body may have excessive static current. This excessive static current may damage parts in the JW-51CM's PC board. Therefore, prior to accessing the JW-51CM, touch your hand to a grounded piece of metal to discharge the static current in your body.

**(5) Cleaning**

- Use a clean, dry cloth when cleaning the JW-51CM. Do not use volatile chemicals such as thinner or alcohol as it may result in deformation and color fading.

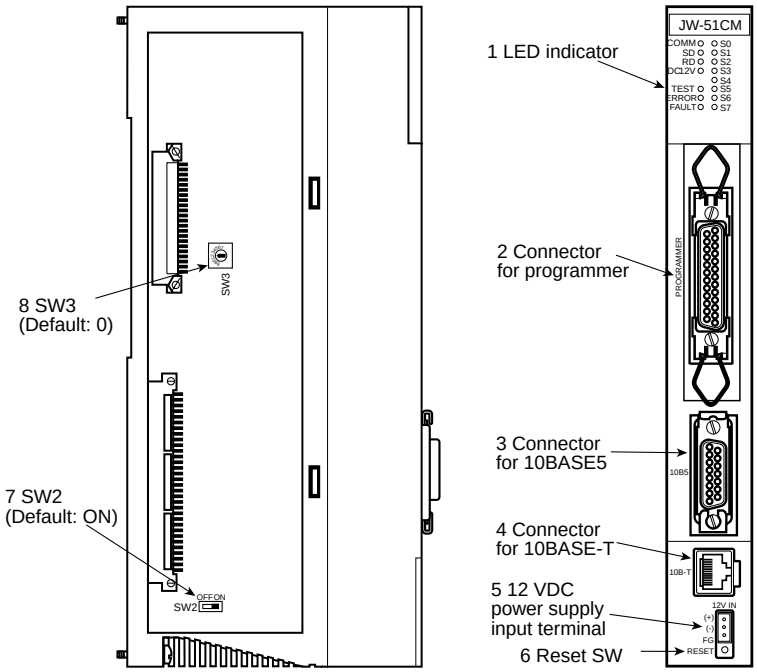
# Chapter 3: System Configuration

[Connection example]




Note: Coaxial cable, transceiver, transceiver cable, 10BASE-T twisted pair cable, and terminater, etc. should be prepared by user.

# Chapter 4: Name and Function of Each Part




| | Name | | Function |
|---|---|---|---|
| ① | Display panel | | Indicates this module s operation status by turning the LED ON and OFF. |
| | | COMM | Lights while operating. Is OFF when operation is stopped. |
| | | SD | Blinks when the JW-51CM is transmitting data. |
| | | RD | Blinks when the JW-51CM is receiving data. |
| | | 12 VDC | Lights when the JW-51CM is receiving 12 VDC power.(Only when using 10BASE5.) |
| | | TEST | Lights when the JW-51CM is in the test mode. |
| | | ERROR | Lights when a parameter setting error occurs. |
| | | FAULT | Lights when an error occurs in this module. |
| | | S0 to S7 | Display connection status monitor flag. |
| 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 | 10BASE-T cable connector | | Connects 10BASE-T twisted pair cable. |
| ⑤ | 12 VDC power supply input terminal | | When using 10BASE5, 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. Also use 0.5 A or more power with 12 VDC –5%. |
| ⑥ | Reset switch | | This switch is only for use by our service personnel. The user should never press this switch. |
| ⑦ | SW2 | ON | The cable shield attached to 10BASE-T and 10BASE5 connectors are connected to the FG (base) of the JW-51CM. |
| | | OFF | The cable shield attached to 10BASE-T and 10BASE5 connectors are not connected to the FG (base) of the JW-51CM.<br>- Separately connection the FG line on the 12VDC connector to the ground. |
| ⑧ | SW3 | | Always set to 0. |

Note: Only a 10BASE5 or 10BASE-T system can be used for communication. (Use of both types at the same time is not allowed.)

# Chapter 5: Installation/Wiring

## 5-1 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.

### [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.
- 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.

# 5-2 Installation

## [1] Installation of cable for option module

Install the optional cable on the basic rack panel that installed JW-51CM. The optional cables and corresponding basic rack panels available are as follows.

**· Cable type for option module**

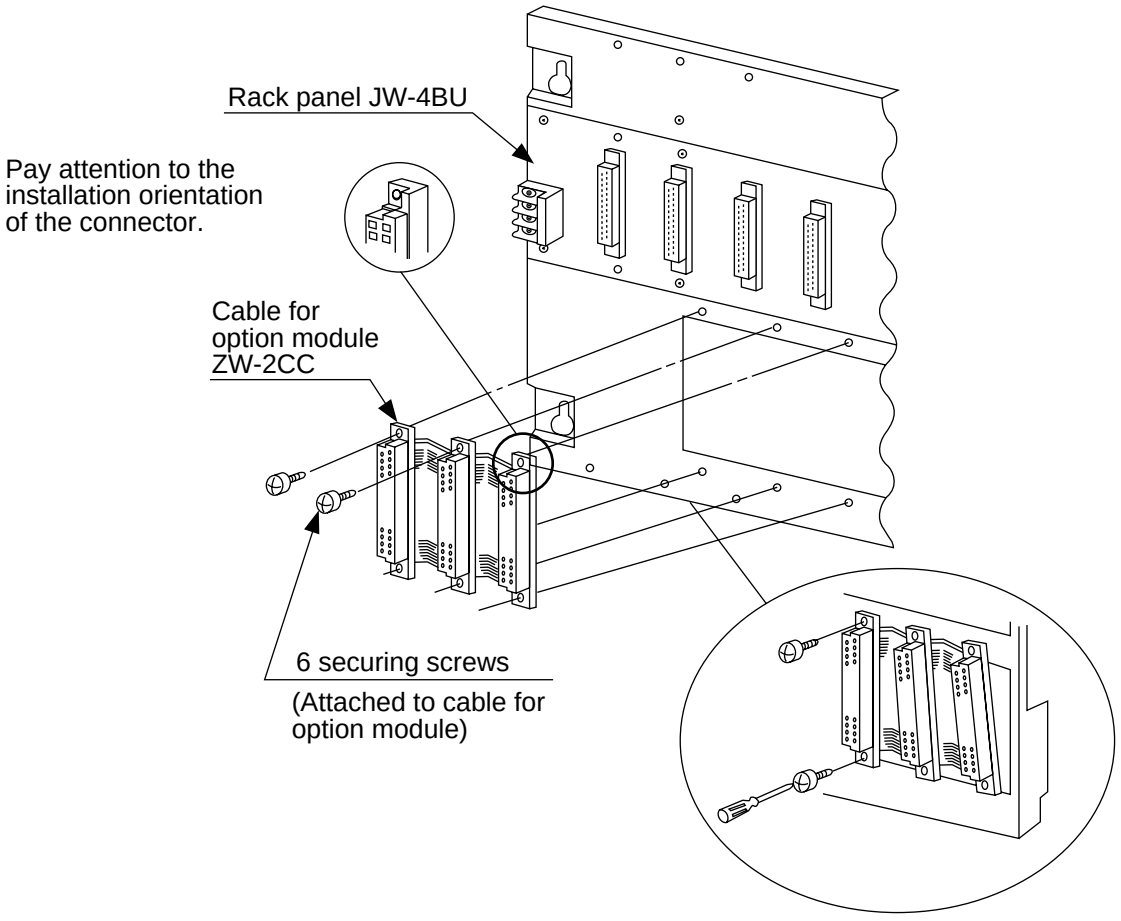
| Cable for option module | Maximum number of JW-51CM that can be installed |
|---|---|
| ZW-2CC | 2 |
| ZW-4CC | 4 |
| ZW-6CC | 6 |

**· Basic rack panel type**

| Model name of the rack panel on which optional cable is installed | Cable for option module (○ : Can be installed × : Cannot be installed) | | |
|---|---|---|---|
| | ZW-2CC | ZW-4CC | ZW-6CC |
| JW-4BU | ○ | × | × |
| JW-6BU | ○ | ○ | × |
| JW-8BU | ○ | ○ | ○ |
| JW-13BU | ○ | ○ | ○ |

5

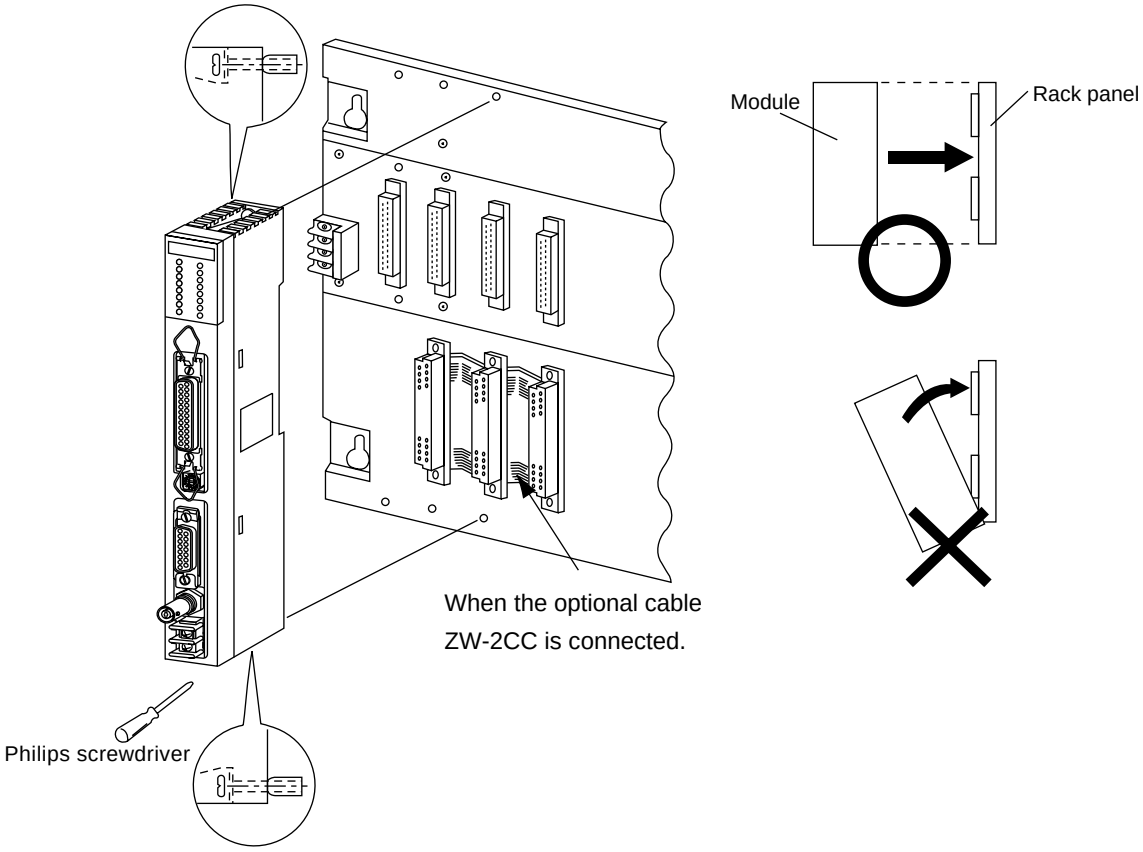
[Example] In case that install a rack panel JW-4BU to ZW-2CC



## [2] Installation of JW-51CM

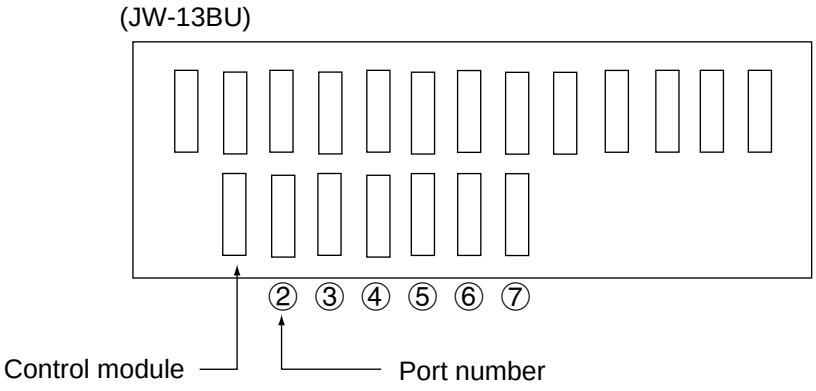
Attach the rack panel using the two attachment screws.
Before installation or removal, make sure to shut OFF the power supply to the PC.

[Example] Install on rack panel JW-4BU




This module can be installed in any one of the optional slots.
Be careful not to bend the connector pins on the module by applying too much force to them.

Optional slots have each port numbers. When an error occurs, the JW-51CM stores the port number corresponding to the error occurred module into system memory #050 in the PC.
This is applied only error code 53: Optional error.



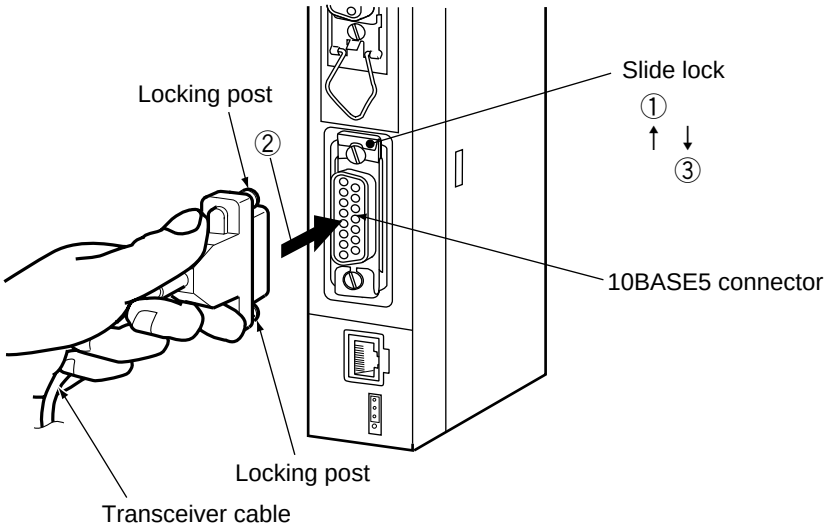
# 5-3 Connection method

This paragraph describes how to connect the JW-51CM to a 10BASE5 or 10BASE-T system.
Only a 10BASE5 or 10BASE-T system can be used for communication. (Use of both types at the same time is not allowed.)

## [1] When connecting to a 10BASE5

Connect the transceiver cable and power supply to the JW-51CM

**(1) Connecting the transceiver cable**



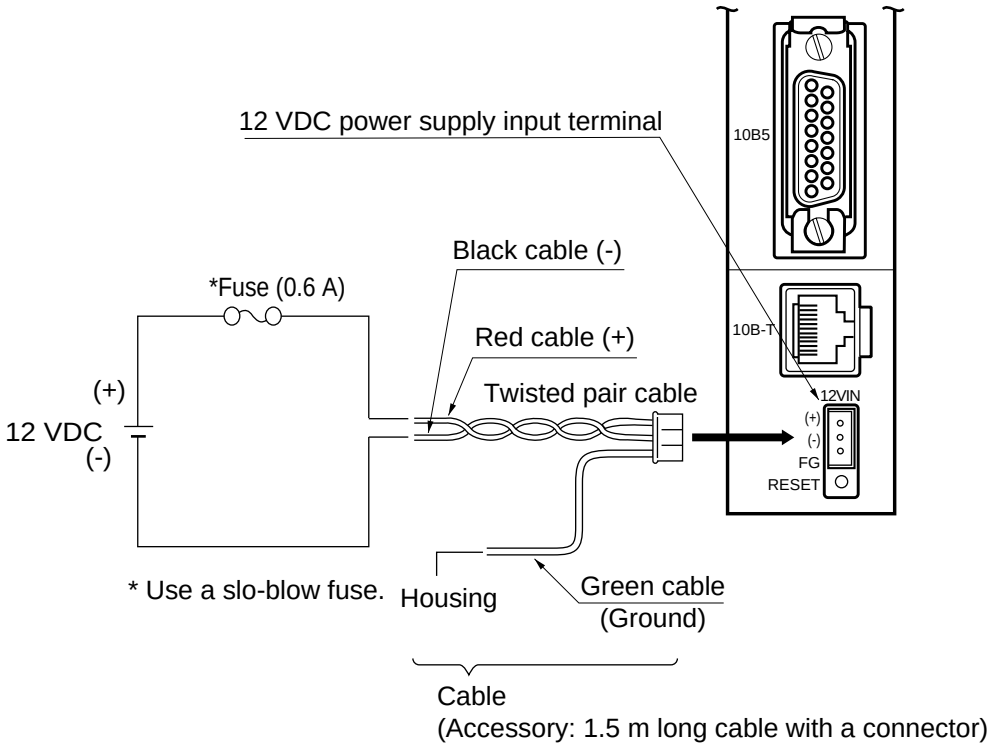

① Slide the lock on the 10BASE5 connector (on the JW-51CM) up.
② Insert the connector so that the two locking posts on the cable connector match the holes on the slide lock.
③ Slide the lock down to lock the cable connector.

**(2) Wiring the power source**

When a 10BASE5 is used, 12 VDC power should be supplied to the transceiver.
Supply power to the 12 VDC power terminals using a commercial constant voltage power supply unit.

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

■ Recommended crimping terminal

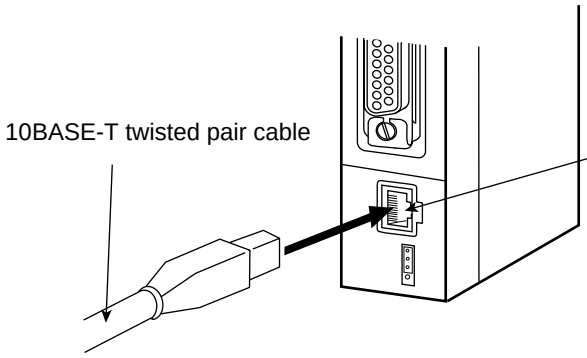



**Remarks**

· Use a power supply that is dedicated for use by the JW-51CM.
· Do not reverse the positive and negative connections to the power terminals. Reversing the polarity may damage the JW-51CM.

### [2] When connecting to a 10BASE-T

Insert the twisted pair cable T connector into the 10BASE-T connector on the JW-51CM.

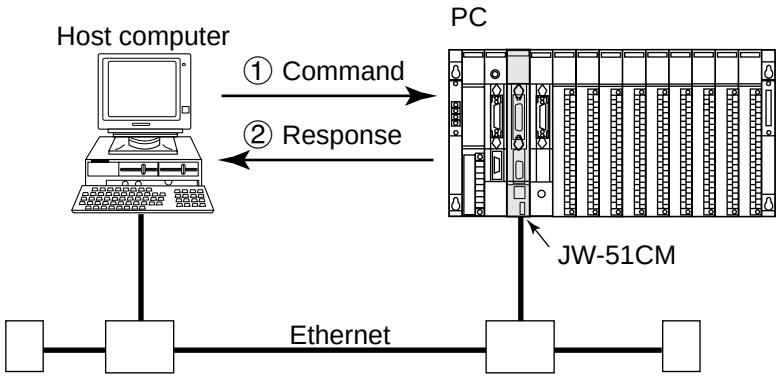



Note: Do not connect the cable to the 12 VDC power input terminal. Otherwise, you cannot communicate with the 10BASE-T.

# Chapter 6: Outline of Function

## 6-1 Computer link function

The  data can be read or written to a connected programmable controller with commands from the host computer.




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.”

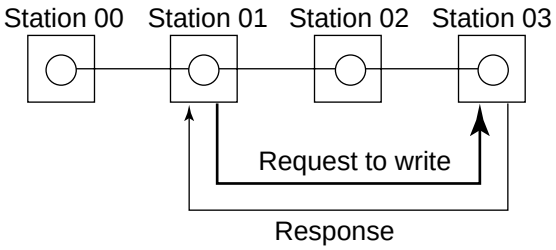
The command contains three types: read, write, and control commands.

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

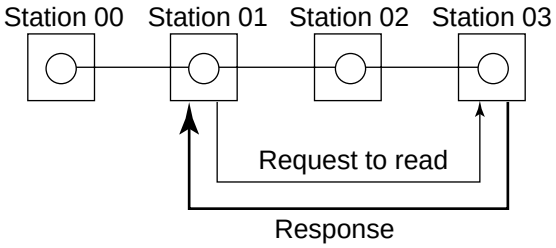
# 6-2 Send/receive function

The send/receive function allows the JW-51CM 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 JW50H/JW70H/JW100H.

**(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. | $6008_{(H)}$ |

# 6-3 Network parameter settings

The following items are set for use as network parameters in the EEPROM.
These parameters are read when the JW-51CM 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, the JW-51CM 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**

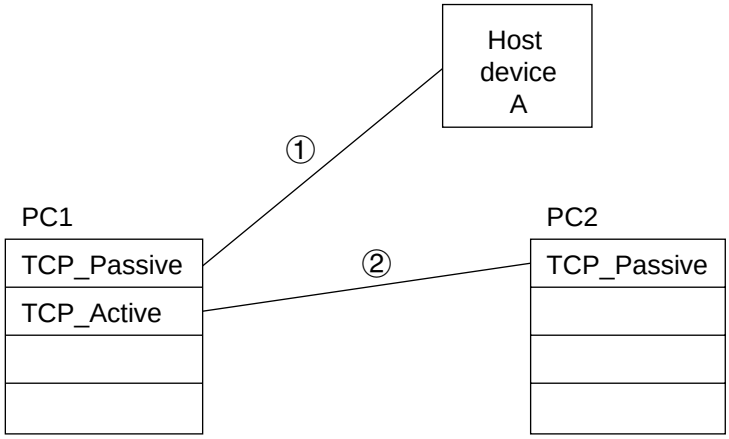
The TCP_Active mode is used to open connections to other stations. This mode can be used with a command triggering station using the send/receive functions. By using this method, the connection to another station can also be broken. While a connection is open, the port cannot communicate with other stations.

**(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-51CM.

<table>
<tr><th>Parameter address</th><th>Details</th></tr>
<tr><td>0000<br>0001<br>0002<br>0003</td><td>IP addresses inside the JW-51CM (0003 is used by the host.)<br>For the details about IP addresses, see the next page.</td></tr>
<tr><td>0004<br>0005<br>0006<br>0007</td><td>Subnet mask: See page 6.6</td></tr>
<tr><td>0100 to 0103</td><td>Settings for connection 0: See page 6.7<br><table><tr><td>0100</td><td>Open method 00(H): TCP_Passive<br>80(H): TCP_Active, 01(H): UDP</td></tr><tr><td>0101</td><td>00</td></tr><tr><td>0102</td><td rowspan="2">JW-51CM port number<br>(0102 as low, 0103 as high)</td></tr><tr><td>0103</td></tr></table></td></tr>
<tr><td>0104 to 0107</td><td>Settings for connection 1<br>(The setting details are the same as for connection 0.)</td></tr>
<tr><td>0110 to 0113</td><td>Settings for connection 2<br>(The setting details are the same as for connection 0.)</td></tr>
<tr><td>0114 to 0117</td><td>Settings for connection 3<br>(The setting details are the same as for connection 0.)</td></tr>
<tr><td>0120 to 0123</td><td>Settings for connection 4<br>(The setting details are the same as for connection 0.)</td></tr>
<tr><td>0124 to 0127</td><td>Settings for connection 5<br>(The setting details are the same as for connection 0.)</td></tr>
<tr><td>0130 to 0133</td><td>Settings for connection 6<br>(The setting details are the same as for connection 0.)</td></tr>
<tr><td>0134 to 0137</td><td>Settings for connection 7<br>(The setting details are the same as for connection 0.)</td></tr>
</table>

Set the communication start/halt conditions in the parameter shown below.

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

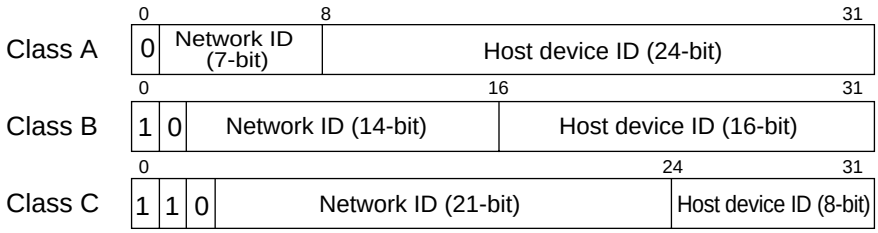
For the details about other parameters, see Chapters 7, 8, and 11.

## ■ 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, the JW-51CM 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.

e.g.: 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.

Nodes located in sub-nets are given different IDs for communcation. To communicate with each other, a router is required. => See page 9-3.

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 and 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 ports." The assignment of ports 1 to 1000 have already been determined.
With the JW-51CM, 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 ports.

## ■ Socket and connection

In the TCP and UDP connection open methods, the IP addresses and port Nos 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.

# Chpater 7: Computer Link Function

## 7-1 Basic format of computer link commands

### [1] Communication format

A message from the host computer to the JW-51CM is referred to as a “command.” A response from the JW-51CM 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 |
|---|---|---|---|---|

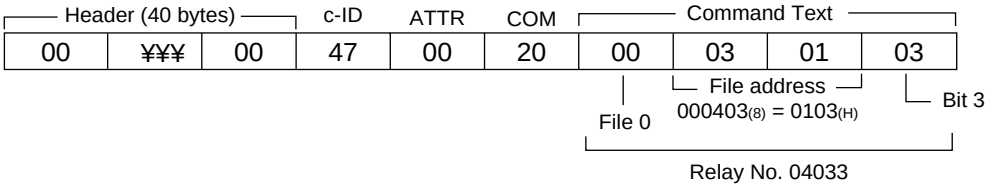
**■ Response**

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

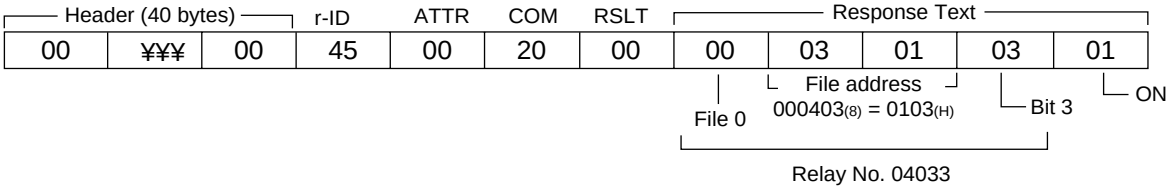
Header : Normally, all 40 bytes are $00_{(H)}$.
If you want to communicate with a satellite net using a JW-51CM to interface between layers of hierarchical communication, you have to use an extension header.
(See “7-8 Two-layer communication with satellite net”)
c-ID : $47_{(H)}$
r-ID : $45_{(H)}$
ATTR : $00_{(H)}$
COM : Command code (See page 7·3)
RSLT : Command execution result
Normaly terminated with $00_{(H)}$
If any byte other than $00_{(H)}$ is found, an error code will be output (See “7-5” Computer link error code table”).
If an error code is output, there is no response text.
Command Text : Command details (See “7-2 Descriptions of each command”)
Response Text : Response details (See “7-2 Descriptions of each command”)

[Example] When you want to monitor the ON/OFF status of relay 04033. (See page 7·6)

**■ Command**




**■ Response**




**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. For the UDP, the total number of bytes from the header to the command text must be less than 1024 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 "7-2 Descriptions of each command.")

PSEG : Program segment 8, 9 (corresponds to the file number.)
PADR : Program address $0000_{(H)}$ to $7DFF_{(H)}$
The program address is to be designated 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 0 to 7(corresponds to the file number.)
DADR : Data memory address For SEG 0 : $0000_{(H)}$ to $1FFF_{(H)}$
For SEG 1 to 7 : $0000_{(H)}$ to $FFFF_{(H)}$
(corresponds to the file number.)
BLOC : Bit location on the data memory 0 to 7
The register (file register) is to be designated using DSEG and DADR.
[Example] Register 09000 : DSEG = $00_{(H)}$, DADR = $0800_{(H)}$
030000 of the file 1 : DSEG = $01_{(H)}$, DADR = $3000_{(H)}$

The relay address is to be designated using DSEG, DADR, and BLOC.
The destination is made by the 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 000725 (]0725))
TADA : Timer/counter number $0000_{(H)}$ to $03FF_{(H)}$ ($0000$ to $1777_{(8)}$)
SADR : System memory address $0000_{(H)}$ to $047F_{(H)}$ ($0000$ to $2177_{(8)}$)

7

## [3] Execution condition

### (1) Write enable mode

Each command will be executed or 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, the JW-51CM 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) PC operation status

Some commands can be executed when the PC halts operation (writing programs: Command code $14_{(H)}$ etc.). Other commands can be executed whether the PC is halted or is running (reading programs: Command code $04_{(H)}$ etc.)

## [4] Table of commands

| Commad code | Contents | See page |
|---|---|---|
| 04(H) | Reading program | 7·15 |
| 14(H) | Write program | 7·16 |
| 20(H) | Monitoring relay | 7·6 |
| 23(H) | The current value monitor of the timers/counters | 7·9 |
| 24(H) | Monitoring register | 7·10 |
| 28(H) | Read from a standard buffer | 7·27 |
| 29(H) | Read a ring buffer | 7·40 |
| 30(H) | Set/reset relay | 7·7 |
| 32(H) | Set/reset timer/counter | 7·8 |
| 34(H) | Write in register | 7·11 |
| 35(H) | Write same data to register | 7·12 |
| 38(H) | Write to a standard buffer | 7·28 |
| 39(H) | Write to a ring buffer | 7·42 |
| 44(H) | Read out the system memory | 7·13 |
| 54(H) | Write to the system memory | 7·14 |
| 68(H) | Read information about a standard buffer | 7·29 |
| 69(H) | Read information about a ring buffer | 7·44 |
| 78(H) | Write information about a standard buffer | 7·30 |
| 79(H) | Write information about a ring buffer | 7·46 |
| A2(H) | Read date | 7·17 |
| A3(H) | Read time | 7·19 |
| B2(H) | Set date | 7·18 |
| B3(H) | Set time | 7·20 |
| E8(H) | Monitor PC operation status | 7·21 |
| E9(H) | Read out write enable mode | 7·4 |
| F8(H) | Halt and release halting of PC | 7·22 |
| F9(H) | Selecting the write enable mode | 7·5 |

# 7-2 Descriptions of each command

This section describes the “COM” settings and the items thereafter of the communication formats (page 7·1).
Commands for the standard buffer are described on pages 7·27 to 7·30. Commands for the ring buffer are described on pages 7·40 to 7·47.

## Read out write enable mode (COM=$E9_{(H)}$)

[Format]

■ Commad

| 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
· PC operation status : Stopping, operating

7

[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 |
|---|---|

■ Responce

| 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
· PC 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 |
|---|---|---|---|---|---|---|

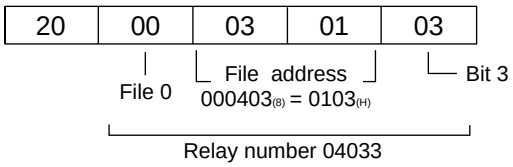
COM = $20_{(H)}$
DSED = Segment ($00_{(H)}$ to $07_{(H)}$)
$DADR_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)}$, if DSEG = $00_{(H)}$, $0000_{(H)}$ to $1FFF_{(H)}$)
BLOC = Bit position ($00_{(H)}$ to $07_{(H)}$)
DATA = Read data ($00_{(H)}$: OFF, $01_{(H)}$: ON)

[Function]
· Read the bit data (relay) shown in DSEG, DADR, and BLOC.

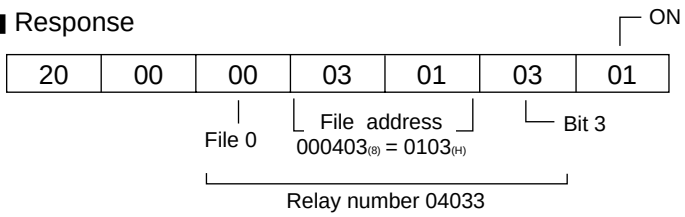
[Execution condition]
· Write enable mode : Mode 0, mode 1 and mode 2
· PC operation status : Stopping, operating

[Example]
· Monitor the ON/OFF status of relay number 04033.

■ Command




■ Response



## 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 |
|---|---|---|---|---|---|

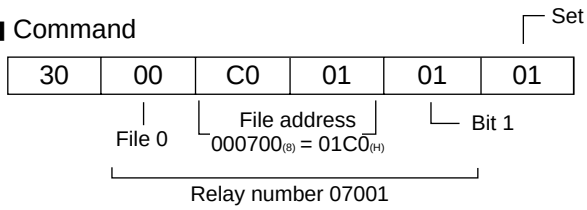
COM = $30_{(H)}$
DSED = Segment ($00_{(H)}$ to $07_{(H)}$)
$DADR_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)}$, if DSEG = $00_{(H)}$, $0000_{(H)}$ to $1FFF_{(H)}$)
BLOC = Bit position ($00_{(H)}$ to $07_{(H)}$)
DATA = Set/reset data ($00_{(H)}$: reset, $01_{(H)}$: set)

[Function]
· Set/reset the relays shown in DSEG, DADR, and BLOC.

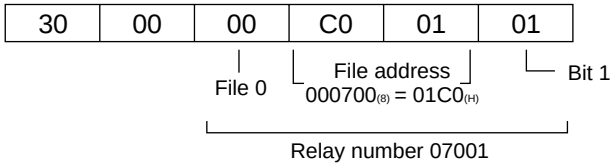
[Execution condition]
· Write enable mode : Mode 1 and mode 2
· PC operation status : Stopping, operating

[Example]
· Set relay number 07001.

■ Command




■ Response



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

[Format]

■ Command

| COM | $TADR_L$ | $TADR_H$ | DATA |
|---|---|---|---|

■ Responce

| COM | RSLT | $TADR_L$ | $TADR_H$ |
|---|---|---|---|

COM = $32_{(H)}$
$TADR_{L, H}$ = Timer·counter number ($0000_{(H)}$ to $03FF_{(H)}$)
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
· PC operation status : Stopping, operating

[Example]
· Set TMR0002.

■ Command

| 32 | 02 | 00 | 01 |
|---|---|---|---|

02: Timer and counter number 0002
01: Set

■ Responce

| 32 | 00 | 02 | 00 |
|---|---|---|---|

02: Timer and counter number 0002

7

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

[Format]

■ Commad

| 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 $03FF_{(H)}$)
$L_{L, H}$ = 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 shown below :

| | | | |
|---|---|---|---|
| $00_{(H)}$ | 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
· PC 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 $07_{(H)}$)
$DADR_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)}$, if DSEG = $00_{(H)}$, $0000_{(H)}$ to $1FFF_{(H)}$)
$L_{L, H}$ = 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
· PC 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 = $34_{(H)}$
PSEG = Segment ($00_{(H)}$ to $07_{(H)}$)
$PADR_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)}$, if DSEG = $00_{(H)}$, $0000_{(H)}$ to $1FFF_{(H)}$)
$L_{L, H}$ = 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
· PC 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

7

## 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 = $35_{(H)}$
PSEG = Segment ($00_{(H)}$ to $07_{(H)}$)
$PADR_{L, H}$ = Byte address ($0000_{(H)}$ to $FFFF_{(H)}$, if DSEG = $00_{(H)}$, $0000_{(H)}$ to $1FFF_{(H)}$)
$L_{L,H}$ = 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
· PC 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 $047F_{(H)}$)
$L_{L,H}$ = Data length (number of bytes)
$DATA_{1\ to\ N}$ = 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
· PC operation status : Stopping, operating

[Example]
· Read data of system memory #204 to 207.

■ Command

| 44 | 08 | 84 | 00 | 04 | 00 |
|---|---|---|---|---|---|

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

■ Response

| 44 | 00 | 08 | 84 | 00 | 04 | 00 | 80 | 01 | 08 | 00 |
|---|---|---|---|---|---|---|---|---|---|---|

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

7

## 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
SEG = Segment ($08_{(H)}$)
$SADR_{L, H}$ = System memory address ($0000_{(H)}$ to $047F_{(H)}$)
$L_{L, H}$ = Data length (number of bytes)
$DATA_{L\ to\ N}$ = 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
· PC 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 ($08_{(H)}$, $09_{(H)}$)
$PADR_{L,H}$ = Program address ($0000_{(H)}$ to $7DFF_{(H)}$)
$L_{L,H}$ = Data length (number of words)
$DATA_{1\ to\ N}$ = 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
· PC 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 ($08_{(H)}$, $09_{(H)}$)
$PADR_{L, H}$ = Program address ($0000_{(H)}$ to $7DFF_{(H)}$)
$L_{L, H}$ = 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
· PC 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 Westerrn 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
· PC operation status : Stopping, operating

[Example]
· Read date data.

■ Command

| A2 |
|---|

■ Response

| A2 | 00 | 97 | 12 | 17 | 03 |
|---|---|---|---|---|---|
| | | '97 | December | 17 | Wednesday |

## Set date (COM = $B2_{(H)}$)

[Format]

■ Command

| COM | Y | M | M | 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)

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

[Function]
· Set date data.

[Example]
· Set data to Friday, January 23, 1998.

■ Command

| B2 | 98 | 01 | 23 | 05 |
|---|---|---|---|---|
| | '98 | January | 23 | Friday |

■ Response

| B2 | 00 |
|---|---|

7

## 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
· PC 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
$08_{(H)}$: 30 sec. correction

[Function]
· Write time data

[Execution condition]
· Write enable mode : Mode 1 and mode 2
· PC 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 PC operation status (COM = $E8_{(H)}$)

[Format]

■ Command

| COM | MODE |
|---|---|

■ 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 instruciton from this module.

[Function]
· Monitor PC run/stop status.

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

[Example]
· Monitor PC operation status.

■ Command

| E8 |
|---|

■ Response

| E8 | 00 | 00 |
|---|---|---|

└— Operating

## Halt and release halting of PC(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 PC operation.

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

[Example]
· Halt PC operation

■ Command

| F8 | 01 |
|---|---|

└ Stopping

■ Response

| F8 | 00 | 01 |
|---|---|---|

# 7-3 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 PC 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 PC memory.

■ **Commands for standard buffers**

| Command code | Details | Reference page |
|---|---|---|
| $28_{(H)}$ | Read from a standard buffer | 7·27 |
| $38_{(H)}$ | Write to a standard buffer | 7·28 |
| $68_{(H)}$ | Read information about a standard buffer | 7·29 |
| $78_{(H)}$ | Write information about a standard buffer | 7·30 |

## [1] How to specify a standard buffer

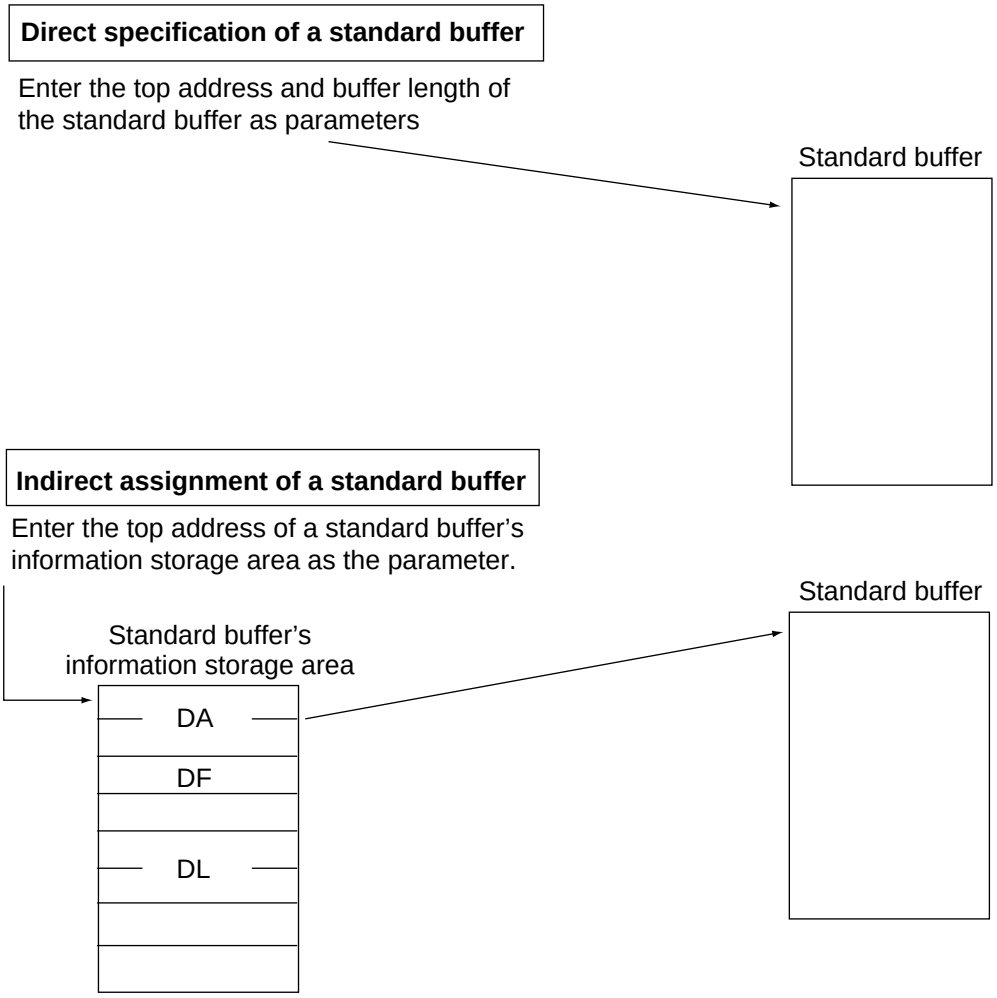
Establish a standard buffer in data memory. The buffer size can be specified (up to 64 k-bytes), in units of one byte. A maximum of 32 buffers can be referenced. Their buffer numbers, 00 to 1F, identifies these buffers.
The following area in data memory can be allocated to standard buffers.

| File number | File address |
|---|---|
| file 0 | 000000 to $017777_{(8)}$ |
| file 1 to 7 | 000000 to $177777_{(8)}$ |

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.

**a) Direct specification**
A method used to specify the top address, file number, and buffer length directly as JW-51CM parameters.

**b) 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.






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

| File number | File address |
|---|---|
| file 0 | 000000 to $017777_{(8)}$ |
| file 1 to 7 | 000000 to $177777_{(8)}$ |

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, the JW-51CM 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>Details</th></tr>
<tr><td>1000 to 1007</td><td>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<br>00(H): Deactivate the standard buffer<br>80(H): Direct assignment of the standard buffer<br>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="39">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<br>+1 | Top file address of the standard buffer (DA) |
| +2 | File number of the standard buffer (DF) |
| +3 | Not used |
| +4<br>+5 | Standard buffer length (DL)<br>Entering $0000_{(8)}$ creates buffer 64 k-bytes long. |
| +6<br>+7 | Not used |

## [4] Error processing when accessing standard buffers

The JW-51CM 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, the JW-51CM will indicate a parameter error (the ERROR lamp will light) when writing the parameters to the EEPROM.

If this happens, the JW-51CM 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, the JW-51CM 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 the JW-51CM 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 | TAG | $IP_H$ | $L_L$ | $L_H$ |
|---|---|---|---|---|---|---|

■ 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.
$DATA_{1 to N}$ = 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, the JW-51CM 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
· PC 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 7·24), 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, 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 | $IP_L$ | $IP_H$ | $L_L$ | $L_H$ | $DATA_1$ | ··· | $DATA_N$ |
|---|---|---|---|---|---|---|---|---|---|

■ Response

| COM | RSLT | DB | TAG | $IP_L$ | $IP_H$ | $L_L$ | $L_H$ |
|---|---|---|---|---|---|---|---|

COM = $38_{(H)}$
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 writing the data)
$L_{L, H}$ = Data length (the number of bytes to write). Any value up to 1024 bytes.
$DATA_{1 to N}$ = 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, the JW-50CM 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
· PC 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 7·24), 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, 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 | 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
$IADR_{L, H}$ = 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
$BADR_{L, H}$ = Top address of the standard buffer
$LB_{L, H}$ = Buffer size (number of bytes). $0000_{(H)}$ corresponds to 64 K bytes
Whether or not these settings are correct, the JW-51CM 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
· PC 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, the module will return error $0F_{(H)}$ (timeout while accessing memory).

**[Exapmle]**

· 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
$BADR_{L, H}$ = Top address of the standard buffer
$LB_{L, H}$ = 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.

**[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, the module will return error $0F_{(H)}$ (timeout while accessing memory).

**[Execution conditions]**

· Read enable mode : Mode 1 and mode 2
· PC operation status : Stopping, operating

**[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 |
|---|---|---|

# 7-4 Ring buffer

A ring buffer is used to transmit data between the host computer and PC in a single direction. An assigned area inside the PC 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 | 7·40 |
| $39_{(H)}$ | Write to ring buffer | 7·42 |
| $69_{(H)}$ | Read the ring buffer information | 7·44 |
| $79_{(H)}$ | Write the ring buffer information | 7·46 |

## [1] How to use the ring buffer

Create the ring buffer in the PC'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.

| File number | File address |
|---|---|
| file 0 | $000000$ to $017777_{(8)}$ |
| file 1 to 7 | $000000$ to $177777_{(8)}$ |

**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 (8) |
|---|---|
| 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.

| File number | File address |
|---|---|
| file 0 | $000000$ to $017777_{(8)}$ |
| file 1 to 7 | $000000$ to $177777_{(8)}$ |

**5) Direction (DIR)**
Assign a data transmission direction
$01_{(H)}$: Reading data from a control module to the JW-51CM
Use the ring buffer read command
$81_{(H)}$: Write data from the JW-51CM 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.

**a) Direct assignment**

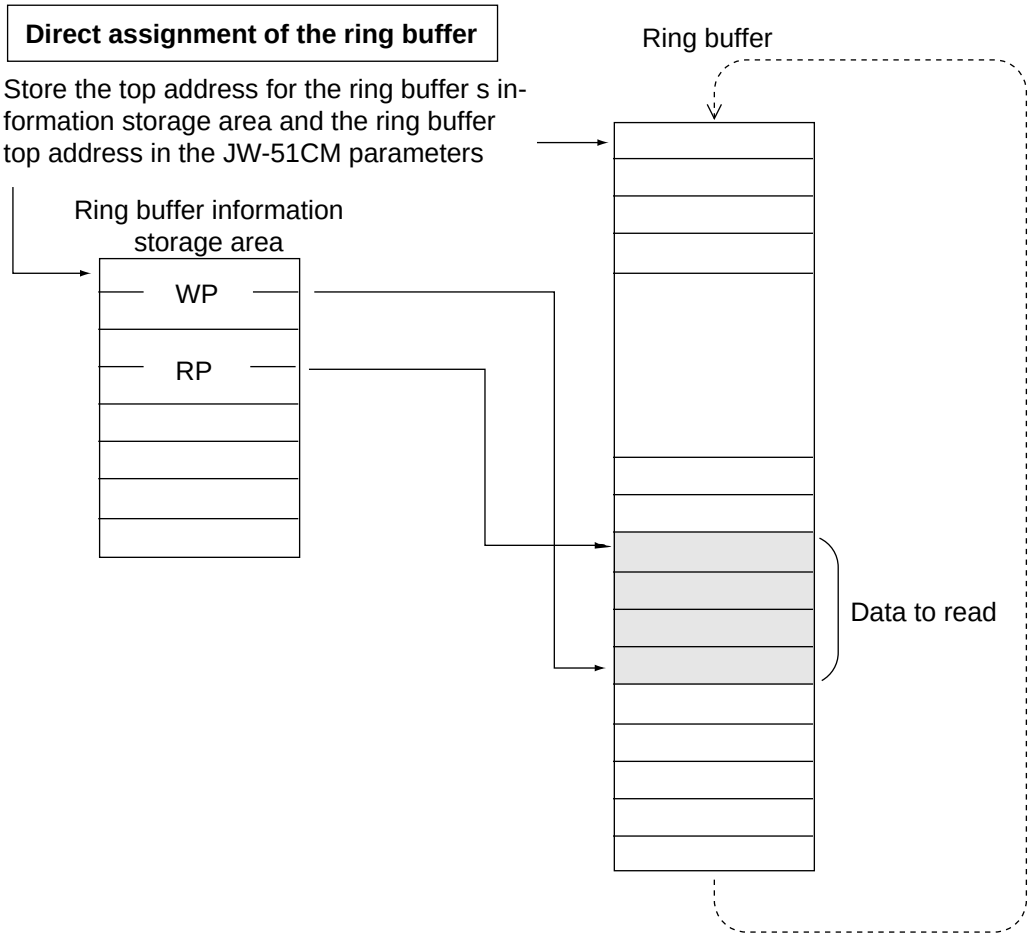
A method to assign the buffer top file address, file number, buffer length, and data transmission direction directly in the JW-51CM parameters.

**b) Indirect assignment**

A method to assign the buffer top file address, file number, buffer length, and data transmission direction in the ring buffer information storage area.

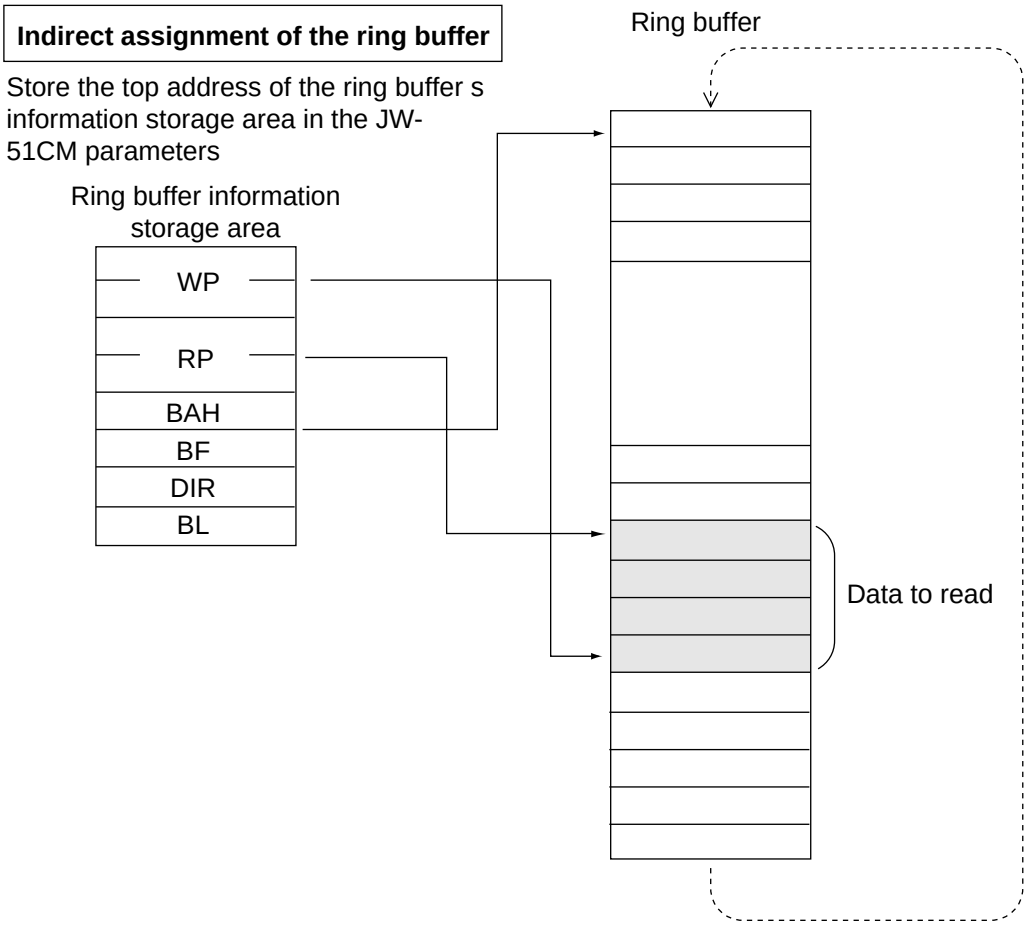
**Direct assignment of the ring buffer**

Store the top address for the ring buffer s information storage area and the ring buffer top address in the JW-51CM parameters




**Indirect assignment of the ring buffer**

Store the top address of the ring buffer s information storage area in the JW-51CM parameters



## [2] Operation of the ring buffer

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 PC to the host computer

① Processing the data with the ladder program
If there is data to transmit, the JW-51CM 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 (see page 7-48 to 7-50).

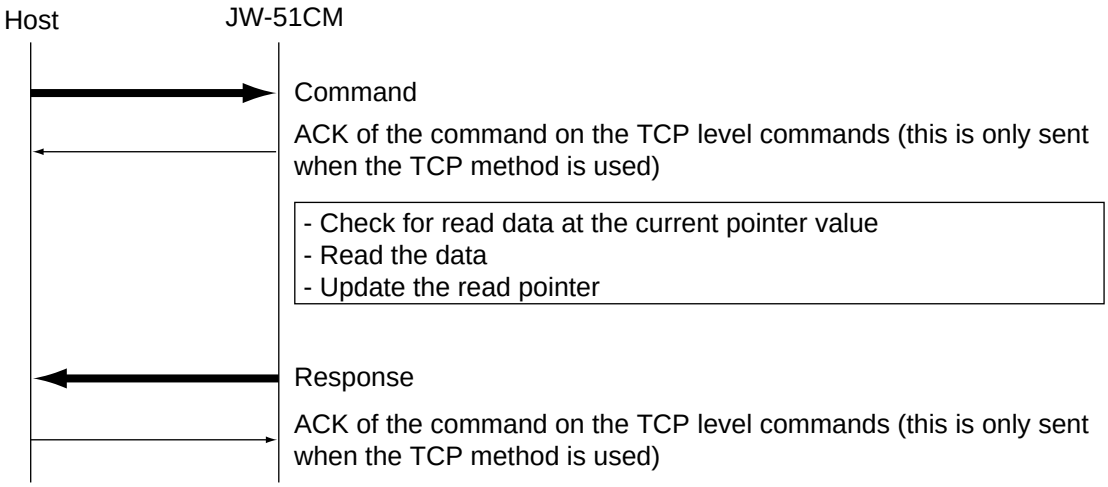
② Processing by the JW-51CM
If the RP matches with the WP, the JW-51CM assumes that there is no data to read. If the RP does not match with the WP data will be read from the area between the RP and the WP-1.
To read the data, use the ring buffer read command (command code $29_{(H)}$). Set the ring buffer number and number of data bytes to read using this command. When the JW-51CM 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.

**a) Non-confirmation type**
Update the pointer at the same time the data is read. Then, the JW-51CM will send its response.

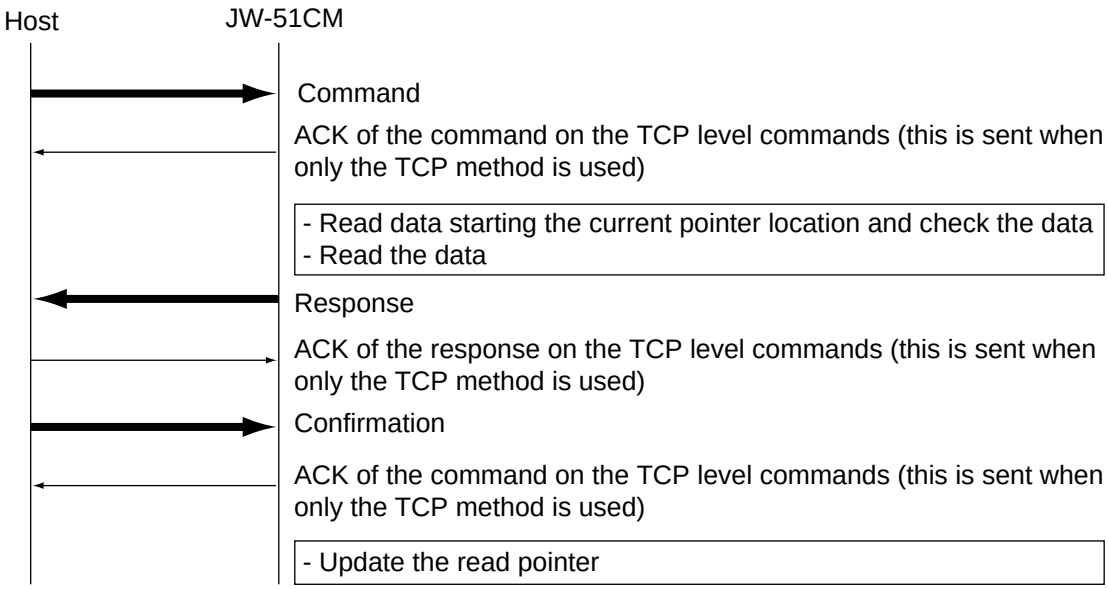
**b) Confirmation type**
The pointer is not updated when the data is read. The JW-51CM waits for confirmation from the host after sending its response. When the JW-51CM receives the confirmation, it will update the pointer. Therefore, the host computer has to send a confirmation after receiving a response.

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 (see page 7·36).

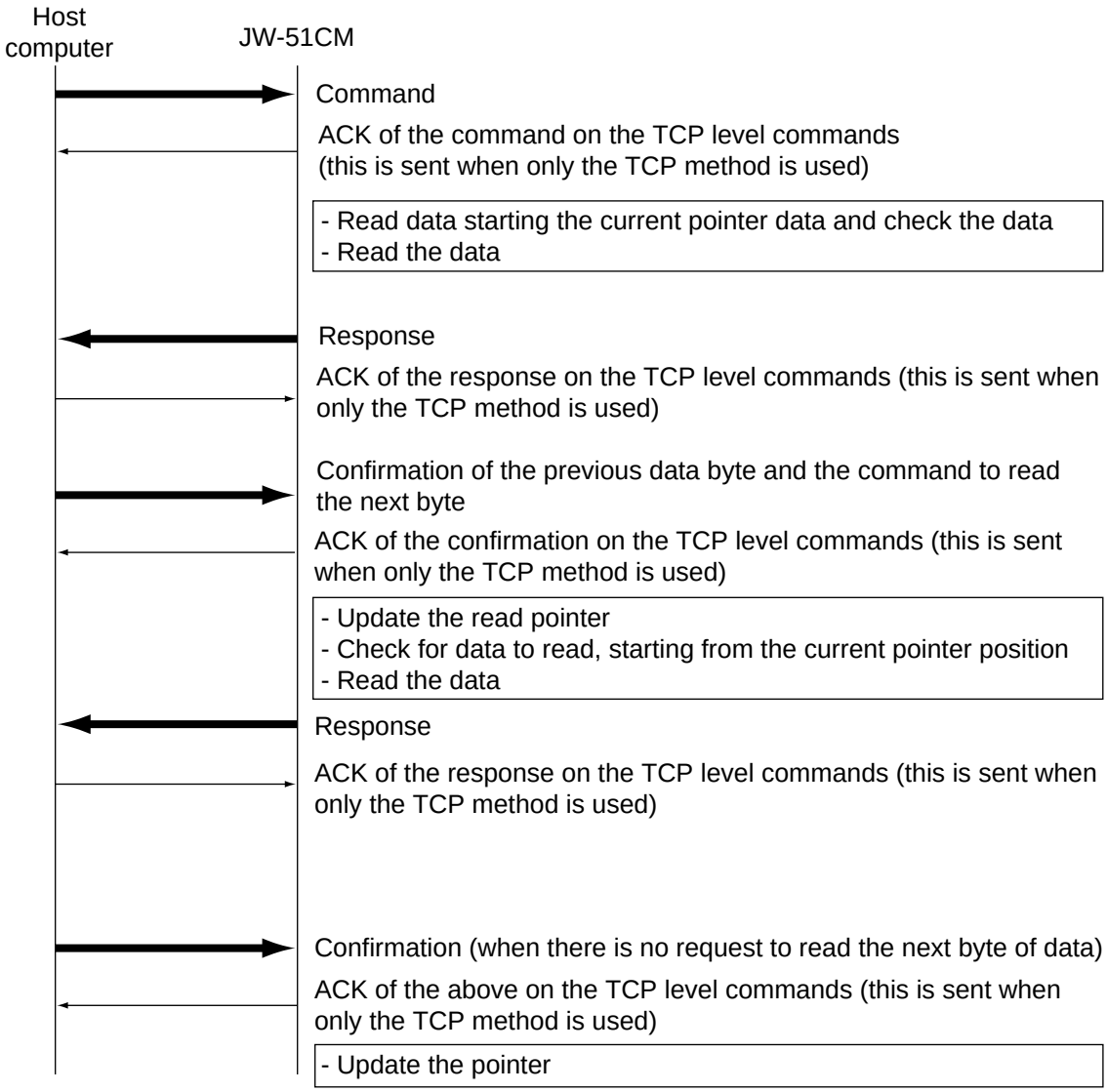
**Non-confirmation type**




**Confirmation type**



**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)**




The non-confirmation type completes the communication process in one application scan cycle. However, if the host computer goes down when the JW-51CM returns a response after processing the 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 JW-51CM 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 JW-51CM 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 JW-51CM will read the number of data bytes available. However, it can only read a maximum of 1024 bytes at one time.

The JW-51CM 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)
a) When there is no data to read, the number of data bytes available will be 0, and the JW-51CM will not continue trying to read.
b) When the data available to read is less than 1024 bytes, the JW-51CM will read all of the data available and then stop reading.
c) When the amount of data to read exceeds 1024 bytes, the JW-51CM 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
d) When there is no data to read, the number of data bytes available will be 0, and the JW-51CM will not continue trying to read.
e) When the number of data bytes available is less than the number of data bytes specified in the command, the JW-51CM will read out all of the data in the buffer, regardless of the number of bytes specified, and then it will stop reading.
f) When the number of data bytes specified in the command is smaller than the number of data bytes available to read, the JW-51CM will read the specified number of data bytes, and then get ready to read the next string data.

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

① Processing by the JW-51CM
The JW-51CM 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 JW-51CM 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 JW-51CM 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 JW-51CM 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 (see page 7·51 to 52).

### [3] Parameter setting

Use parameters 1400 to 1577 to create the ring buffer.

<table>
<tr><th>Parameter address</th><th>Details</th></tr>
<tr><td>1400 to 1407</td><td>
Information concerning ring buffer 00
<table>
<tr><td>1400<br>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="3">Reads data from control module to JW-51CM</td></tr>
<tr><td>81</td><td colspan="3">Writes data from JW-51CM to control module</td></tr>
<tr><td rowspan="6">1404</td><td colspan="4">Ring buffer top address (upper bytes of the file address)<br>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 colspan="2"></td></tr>
</table>
<table>
<tr><td>1407</td><td>Selection of the ring buffer<br>00(H): Disable the ring buffer<br>80(H): Direct assignment of the ring buffer<br>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="16">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

The JW-51CM 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, the JW-51CM will indicate a parameter error (the ERROR lamp will light) when writing the parameters into the EEPROM.
If this happens, the JW-51CM 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, the JW-51CM 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 the JW-51CM 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 the JW-51CM 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 $1F_{(H)}$)
FUN = Selected function
$01_{(H)}$: 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 JW-51CM 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)
$LR_{L, H}$ = 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.
$DATA_{1 to N}$ = 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 JW-51CM 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 |
| | $1 \leqq LP \leqq 1024$ | $00_{(H)}$ | LR = LP |
| | $LP > 1024$ | $01_{(H)}$ | LR = 1024 |
| $1 \leqq LC \leqq 1024$ | LP = 0 | $00_{(H)}$ | LR = 0 |
| | $LP \leqq LC$ | $00_{(H)}$ | LR = LP |
| | $LC < LP$ | $01_{(H)}$ | LR = LC |

**[Execution conditions]**
· Write enable mode : Mode 0, mode 1, and mode 2
· PC 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 7·32), 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, 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: 03 00 — Data length: 04 00

| 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 $1F_{(H)}$)
FUN = Selected function
$01_{(H)}$: 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.
$DATA_{1 to N}$ = 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)
$LR_{R, H}$ = 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.

7

**[Execution conditions]**

· Write enable mode : Mode 1 and mode 2
· PC 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 7·32), 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, 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 control module to JW-51CM)
$81_{(H)}$: Data flow direction (from JW-51CM to control module)
$LB_{L, H}$ = Buffer size (number of bytes). $0000_{(H)}$ corresponds to 64 K-bytes.

Regardless of whether or not these settings are correct, the JW-51CM 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
· PC 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 |
|---|---|---|---|---|---|---|---|---|---|---|

Register 09000 — Register 29000

Direct assignment

| 03 | 00 | 06 | 00 | 81 | 00 | 04 |
|---|---|---|---|---|---|---|

WP — RP — 1 K bytes

Writing data

## 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 control module to JW-51CM)
  $81_{(H)}$: Data flow direction (from JW-51CM to control module)
$LB_{L, H}$ = Buffer size (number of bytes). $0000_{(H)}$ corresponds to 64 K-bytes.

7

**[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
· PC 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, 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 |
|---|---|---|---|---|---|---|---|---|---|---|

File 3, from $0000_{(H)}$ — WP — RP

Indirect assignment

| 80 | 00 | 80 |
|---|---|---|

32 K- bytes

Reading

■ Response

| 79 | 00 | 04 |
|---|---|---|

## [7] An example using the ring buffer

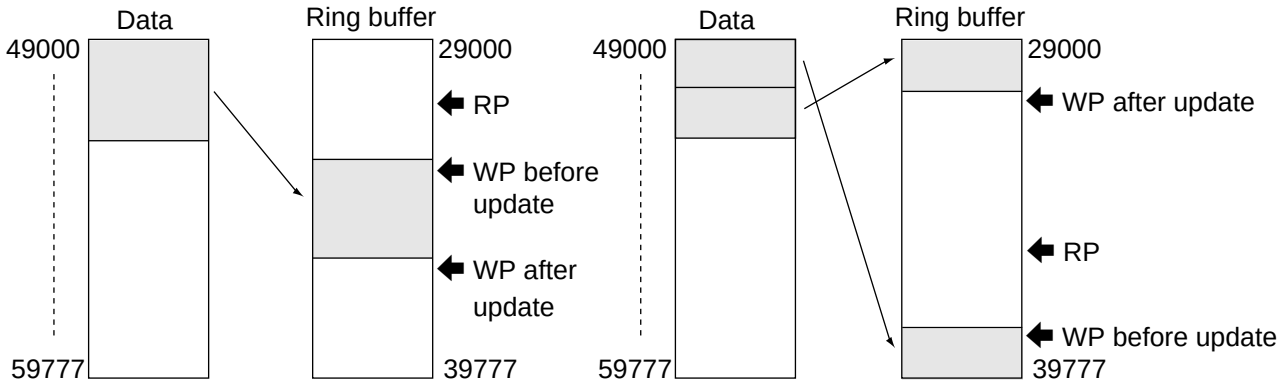
### (1) Data flow direction (from PC to the host computer)

· 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 JW-51CM.
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 JW-51CM updates the WP.

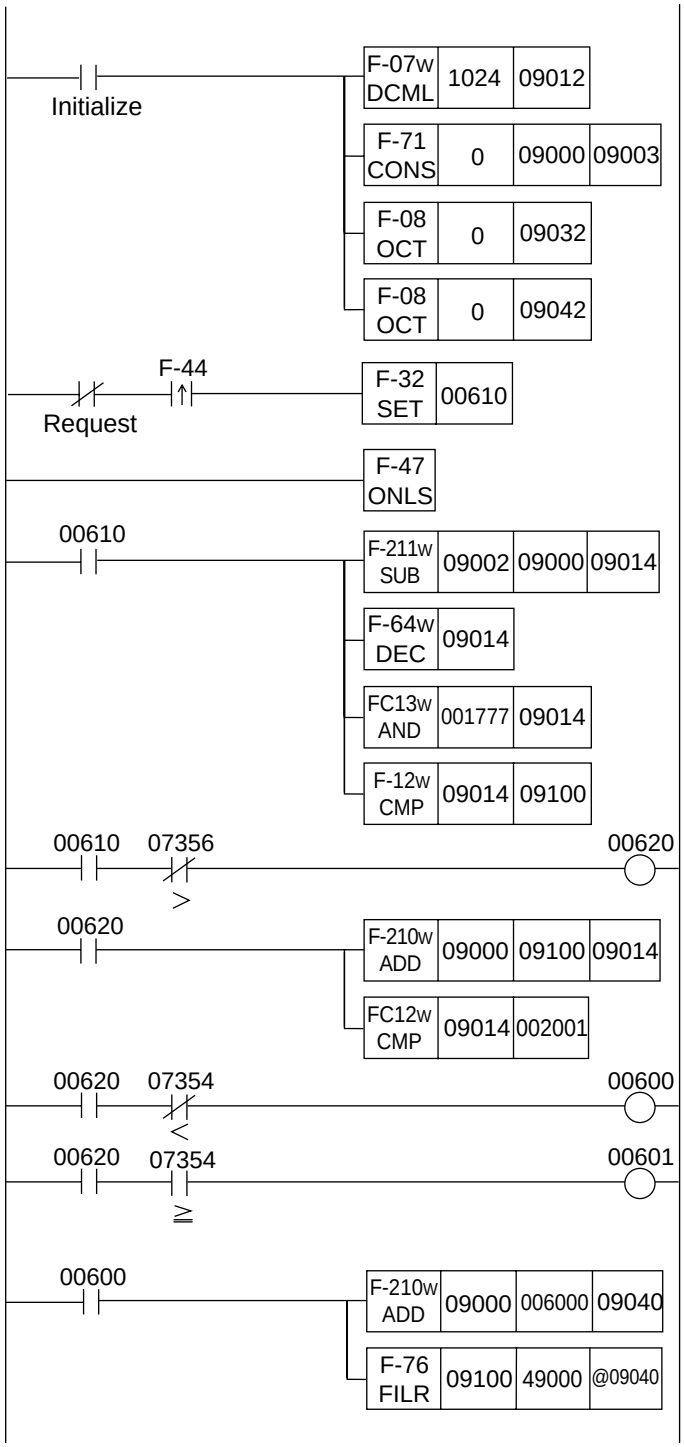
There are two ways to transmit data, as shown below:




(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 |




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.

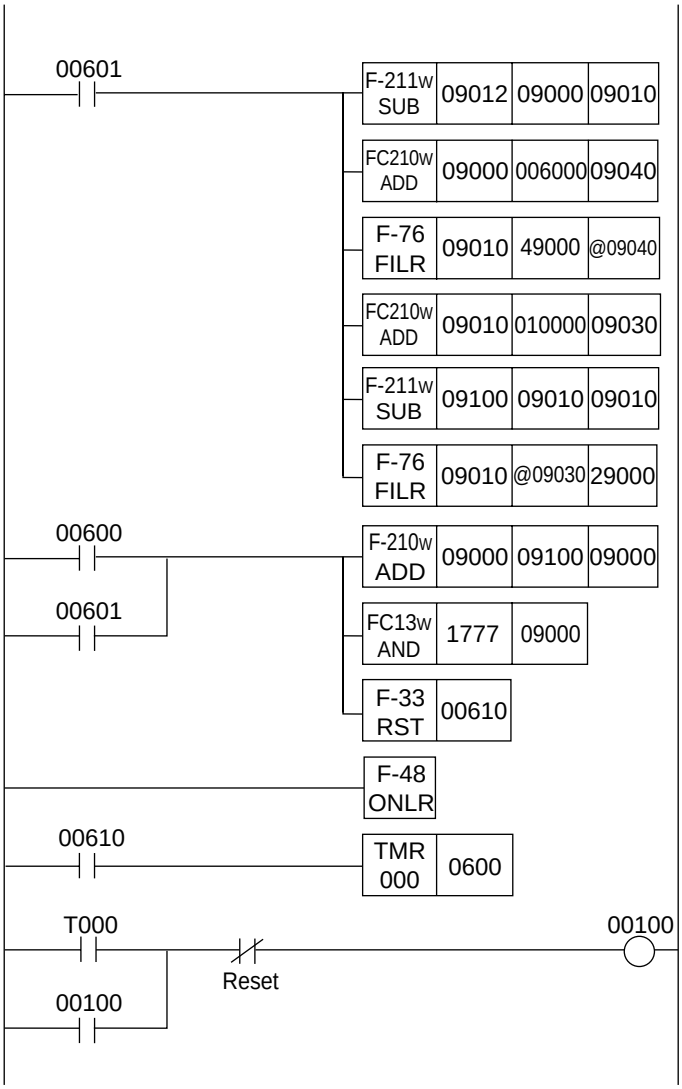
(a) The normal case

(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




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-51CM 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 PC)**

· 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 | |

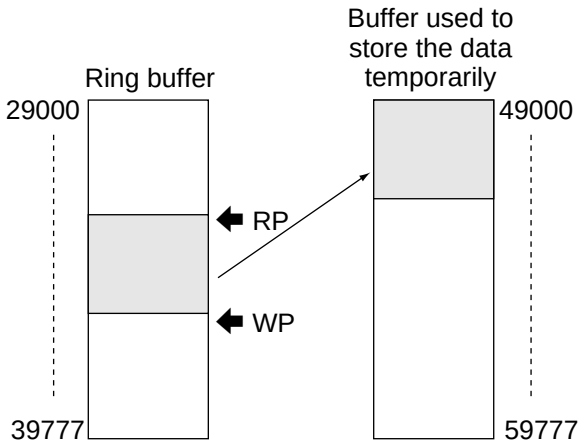
■ Processing ladder program

① Initialize the pointers
The read and write pointers should be initialized (set to 0) using a ladder program when starting the JW-51CM.

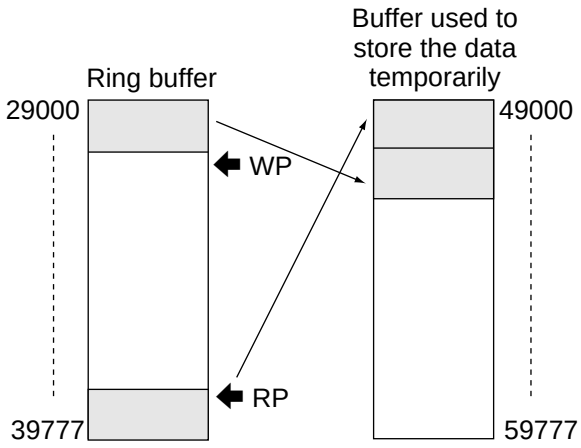
② Picking up the write data
When the JW-51CM receives a ring buffer write command from the host computer, it will pick up the data that has been received (temporarily stored in other memory). To find out whether it has been received or not, check whether the WP and RP are in the same position. To determine the number of bytes received, compare WP and RP values.

③ Update the RP
After temporarily storing the data, the JW-51CM updates the RP.

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

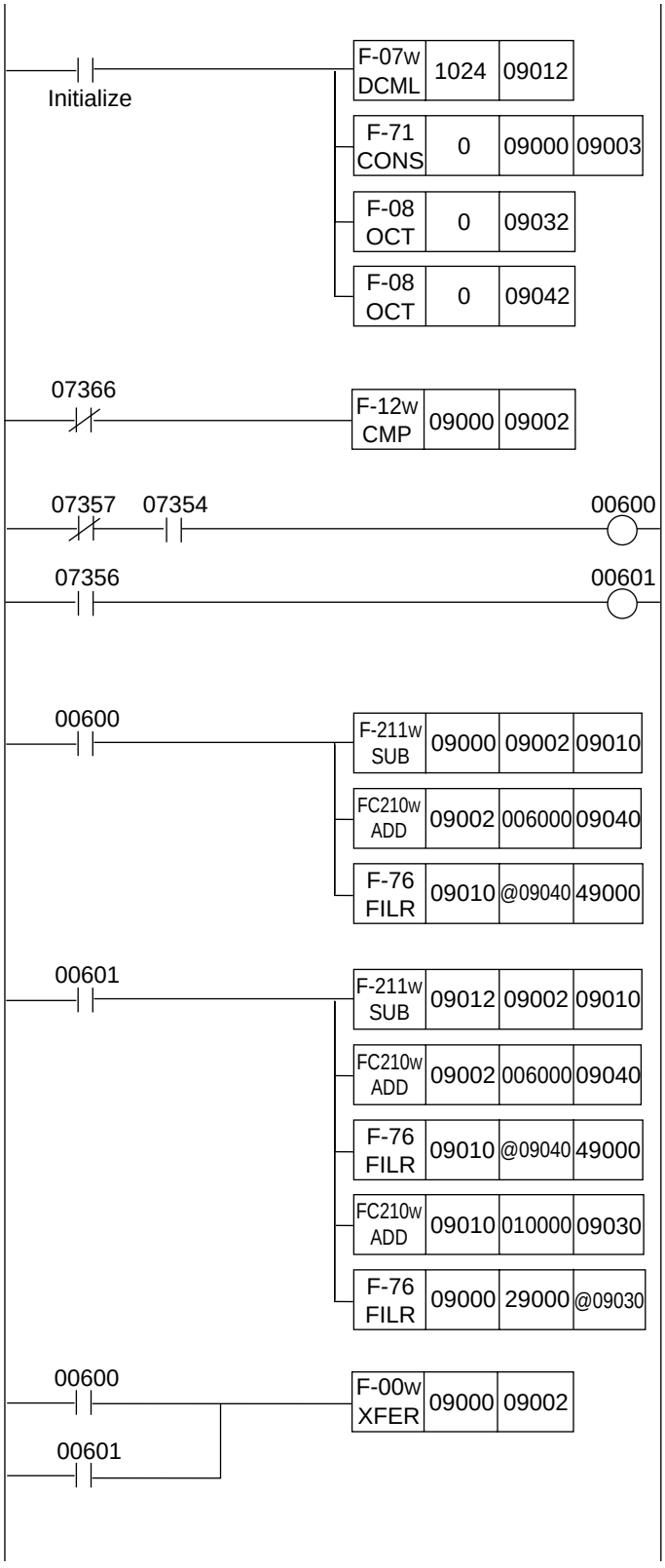



(a) Normal case




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

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




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.

# 7-5 Computer link error code table

| RSLT (Hexadeci-mal) | Details |
|---|---|
| 00 | Normally end |
| 01 | Format error |
| 06 | PC does not stop operation |
| 07 | Verify error of write command. |
| 0F | Time out while accessing memory. |
| 13 | Tried to set/reset TMR/CNT while PC 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. |

# 7-6 Command execution completion information

When the JW-51CM completes execution of a computer link command, it writes the completion details in the PC's data memory.
This function can be selected by setting a parameter.

## [1] Setting the parameters

<table>
<tr><th>Parameter address</th><th>Details</th></tr>
<tr><td>3660<br>to<br>3667</td><td>Set the command execution completion information storage area
<table>
<tr><td>3660</td><td rowspan="2">Top file address of the command execution completion information</td></tr>
<tr><td>3661</td></tr>
<tr><td>3662</td><td>File number of the command execution completion information</td></tr>
<tr><td>3663</td><td>Not used</td></tr>
<tr><td>3664</td><td rowspan="2">Set the number of command execution completion information bytes<br>Assign at least 16 bytes. 64 bytes is the maximum size.</td></tr>
<tr><td>3665</td></tr>
<tr><td>3666</td><td>Not used</td></tr>
<tr><td>3667</td><td>This information will become effective when 80(H) is entered</td></tr>
</table></td></tr>
</table>

## [2] Command execution completion information

The command execution completion information has the following format.

<table>
<tr><td>+00</td><td rowspan="4">Target station IP address</td></tr>
<tr><td>+01</td></tr>
<tr><td>+02</td></tr>
<tr><td>+03</td></tr>
<tr><td>+04</td><td rowspan="2">Target station port number</td></tr>
<tr><td>+05</td></tr>
<tr><td>+06</td><td>Connection number of the source station</td></tr>
<tr><td>+07</td><td>00(H)</td></tr>
<tr><td>+10</td><td rowspan="3">Execution result</td></tr>
<tr><td>+12</td></tr>
<tr><td>+13</td></tr>
<tr><td>+14</td><td rowspan="5">Copy to the receive command (except the header)</td></tr>
<tr><td>+15</td></tr>
<tr><td>+16</td></tr>
<tr><td>:</td></tr>
<tr><td>+n</td></tr>
</table>

This area is written after the JW-51CM executes the command. To clear the contents, use the ladder program.
The following area in the data memory can be used for this format.

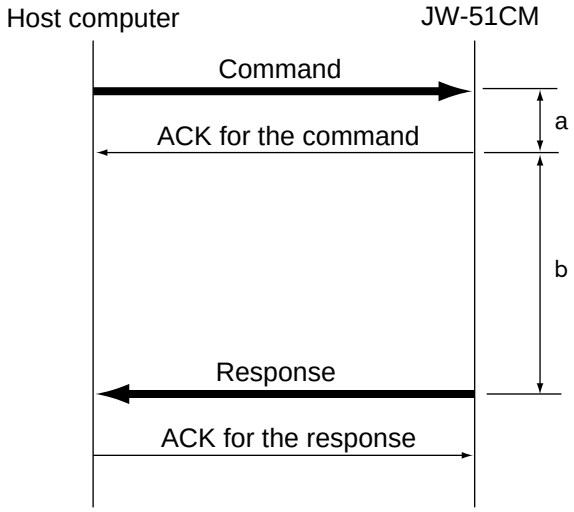
| File number | File address |
|---|---|
| file 0 | 000000 to 017777(8) |
| file 1 to 7 | 000000 to 177777(8) |

# 7-7 Time interval required for communication

The time interval after the JW-51CM receives the command until its sends the response varies with the PC 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. PC scan time: 8 ms. Data: 1024 bytes

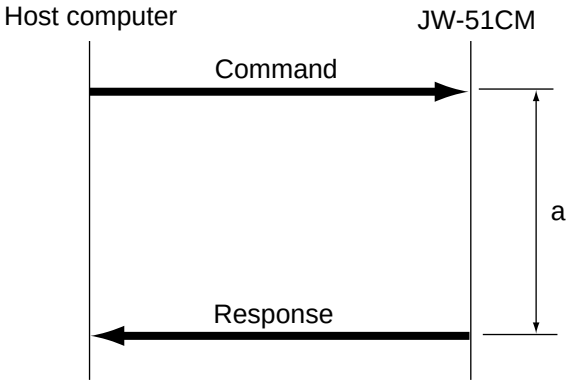
**(1) When the TCP method is used**




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

b: Time after returning an ACK until a response is sent
40 ms on average This time includes the time that the JW-51CM must wait to access the control module. (Maximum PC scan time.)

**(2) When the UDP method is used**

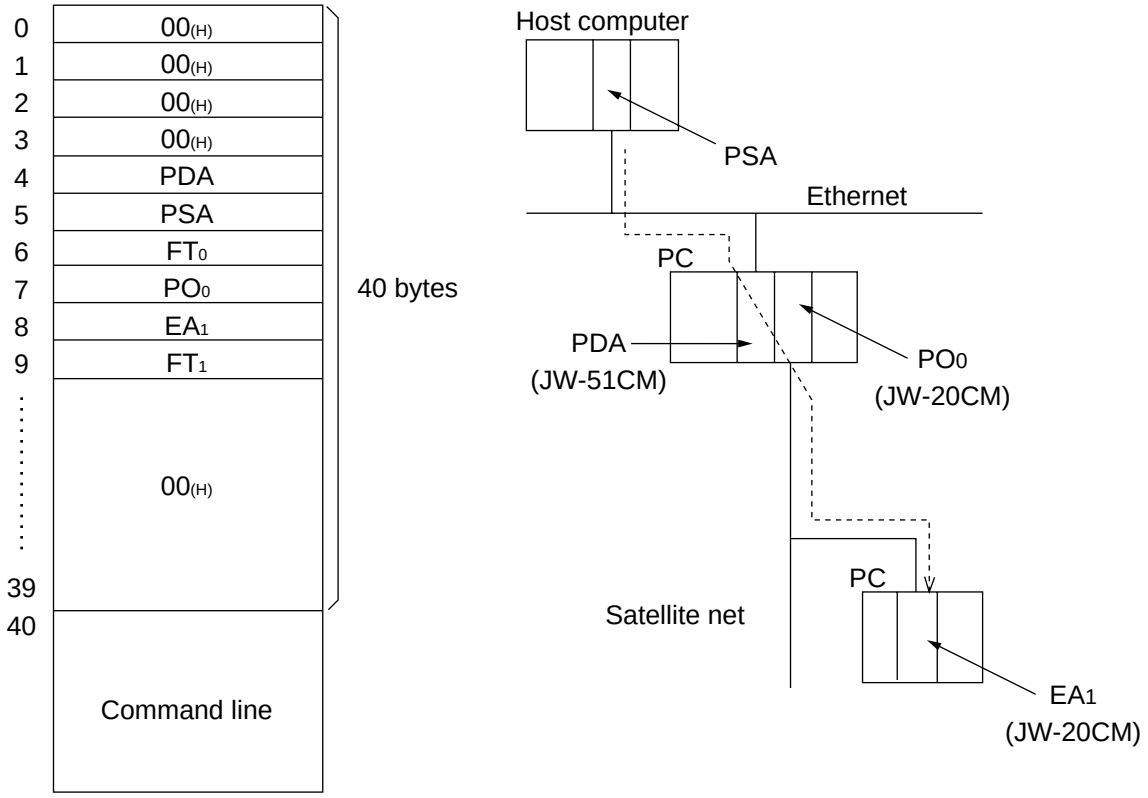



a: Time after receiving the command until a response is sent
40 ms on average This time includes the time that the JW-51CM must wait to access the control module. (Maximum PC scan time.)

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 PC scan time gets longer.

# 7-8 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 (see page 7·1) as an extension header.




7

· 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 JW-51CM that connects with the satellite net. This may be any value within the range from $01_{(H)}$ to $40_{(H)}$ that can be descriminated 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
Assign a slot number where the JW-20CM satellite network module is installed on the junction station PC. The slots are numbered 2, 3, 4 and so on up to 7, starting from the slot after the control module (when the ZW-6CC basic rack panel 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 $40_{(H)}$.

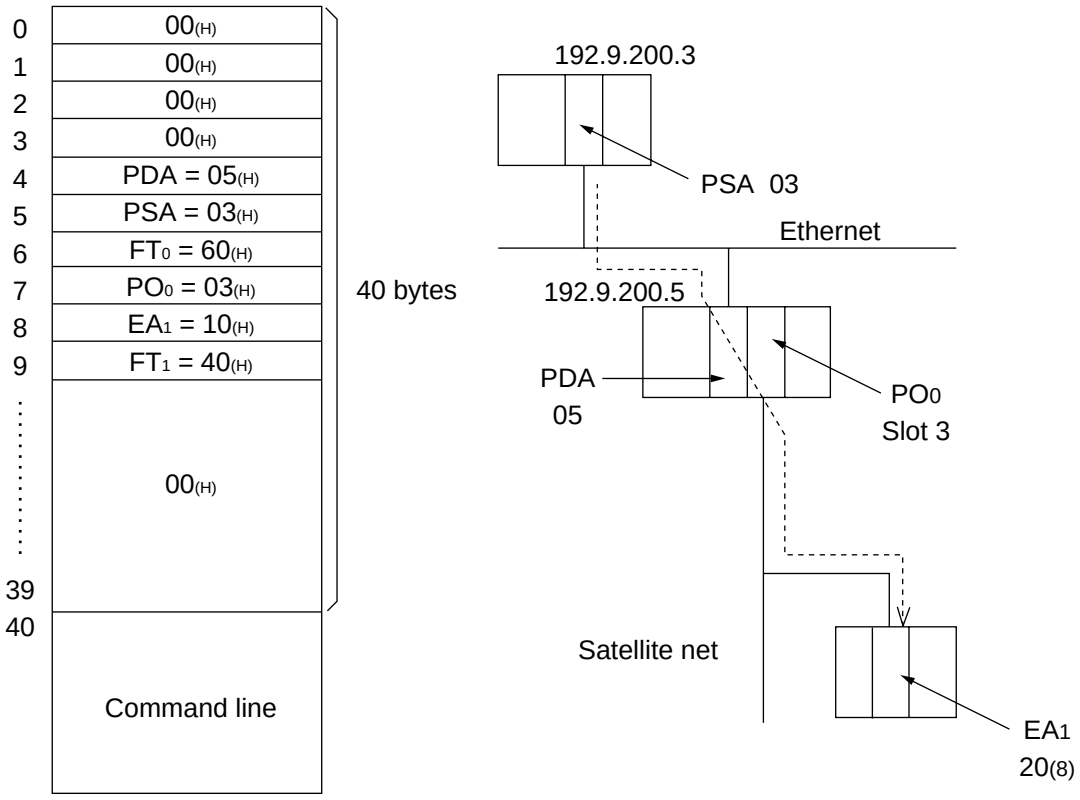
(f) $FT_1$ : Frame type 1
Set the $40_{(H)}$.

(g) Command line : Command/response line
c-ID/r-ID and after of communication format (page 7·1)

**Note**

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.



# Chapter 8: Send/Receive Functions

The send/receive functions transmit data from this module to other stations or receive data from other stations. The send/receive functions have two types : instruction system and data memory starting system.

## 8-1 Instruction system

### [1] Source/destination address and channel

The send/receive instructions set the rack, slot, and channel addresses of the mounted 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 JW-51CM 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 JW-51CM 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.

a) 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 $40_{(H)}$.

b) 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>Details</th></tr>
<tr><td>0400</td><td>Assign a station number corresponding to the table (for SEND/RECEIVE)<br>Assignment of the relationship between the station number used with the send/receive function and the actual address.<br>If automatic is selected, the JW-51CM treats the target station numbers as node numbers on the IP addresses, and sets the target port to 6010(H).<br>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.)<br>01(H): Automatic<br>02(H): Individual</td></tr>
<tr><td>0401 to 0407</td><td>Reserved area</td></tr>
<tr><td>0410<br>to<br>0417</td><td>Station number correspondence table 1.<br>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)<br>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<br>(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.<br>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.<br>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.<br>This is effective only when parameter 0400 is 02(H).</td></tr>
<tr><td>0770 to 0777</td><td>Station number corresponding table 37.<br>This is effective only when parameter 0400 is 02(H).</td></tr>
</table>

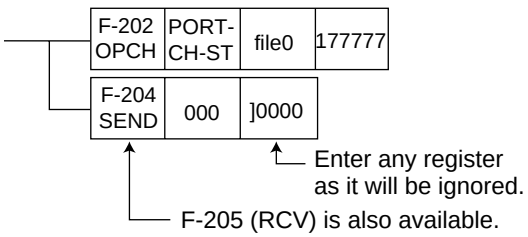
**(3) TCP connection and disconnection**

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.

a) Establishing the connection

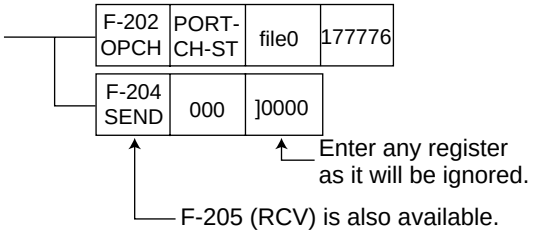
Executing the send/receive functions with the memory address of the target station file number = 0, file address = $177777_{(8)}$, and the number of bytes transmitted = 0 will start the connection establishing operations with the target station.

The connection takes approximately two seconds to complete.




b) 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.




c) Data transmission

Any address and number of bytes transmitted other than shown at a) and b) will start the actual send/receive operations.

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

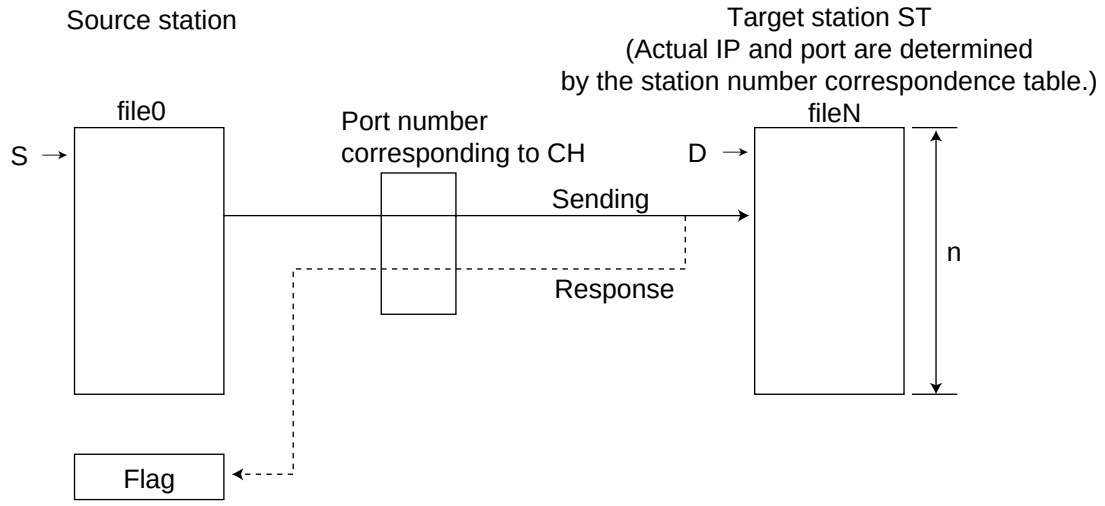
## [2] SEND/RECEIVE instructions operation

### (1) Send

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

| F-202 OPCH | PORT-CH-ST | fileN | D |
|---|---|---|---|
| F-204 SEND | n | S | |

PORT : The port (slot) where the JW-51CM is installed.
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 page 8·2)
file N : Data area of the target station PC (file number)
ADRS : Data area top file address of target station PC
n : Number of data bytes transmitted (000 to $377_{(8)}$, 256 bytes at 000)
S : Data area top register of source station data



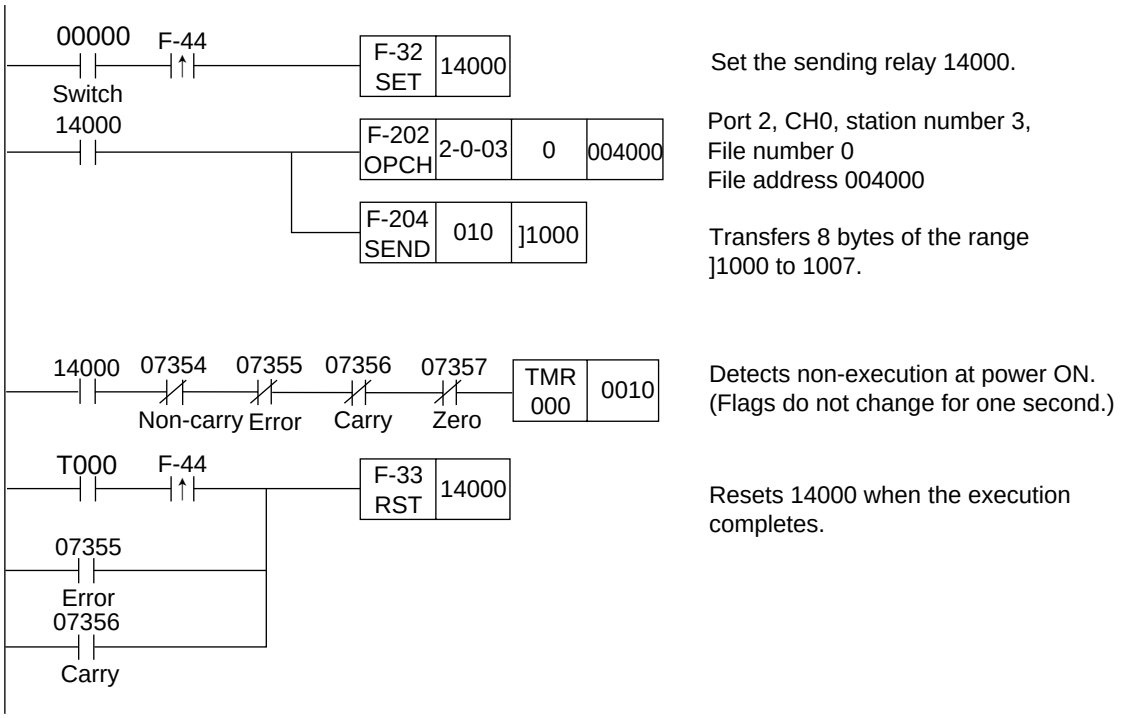

Flag status during and after the operation

| | Zero 07357 | Carry 07356 | Error 07356 | Non-carry 07356 | Explanation |
|---|---|---|---|---|---|
| No response from port | 0 | 0 | 1 | 0 | The JW-51CM is not installed in the slot. |
| 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 JW-51CM of source station : 2
Channel used : 0




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 $6000_{(H)}$.

8

**Note**

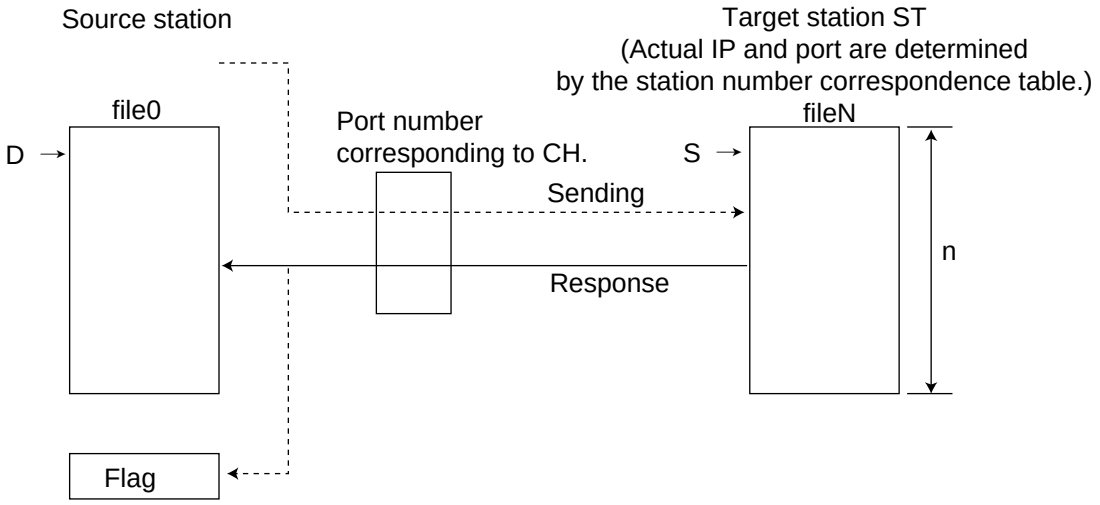
- The entry condition of F-202 or 204 instruction needs to be kept ON until the execution of the instruction completes (or until any error occurs or the carry flags turns ON). If the entry condition turns OFF while the instruction is being run, the instruction will end in an incomplete condition. Once 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 PC, 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).

| F-202 OPCH | PORT-CH-ST | fileN | ADRS |
|---|---|---|---|
| F-205 RCV | n | D | |

PORT : Port (or slot) where the JW-51CM is installed
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 page 8·2)
file N : Data area of the target station PC (file number)
ADRS : Data area top file address of target station PC
n : Number of data bytes transmitted (000 to $377_{(8)}$, 256 bytes at 000)
D : Data area top register of source station data



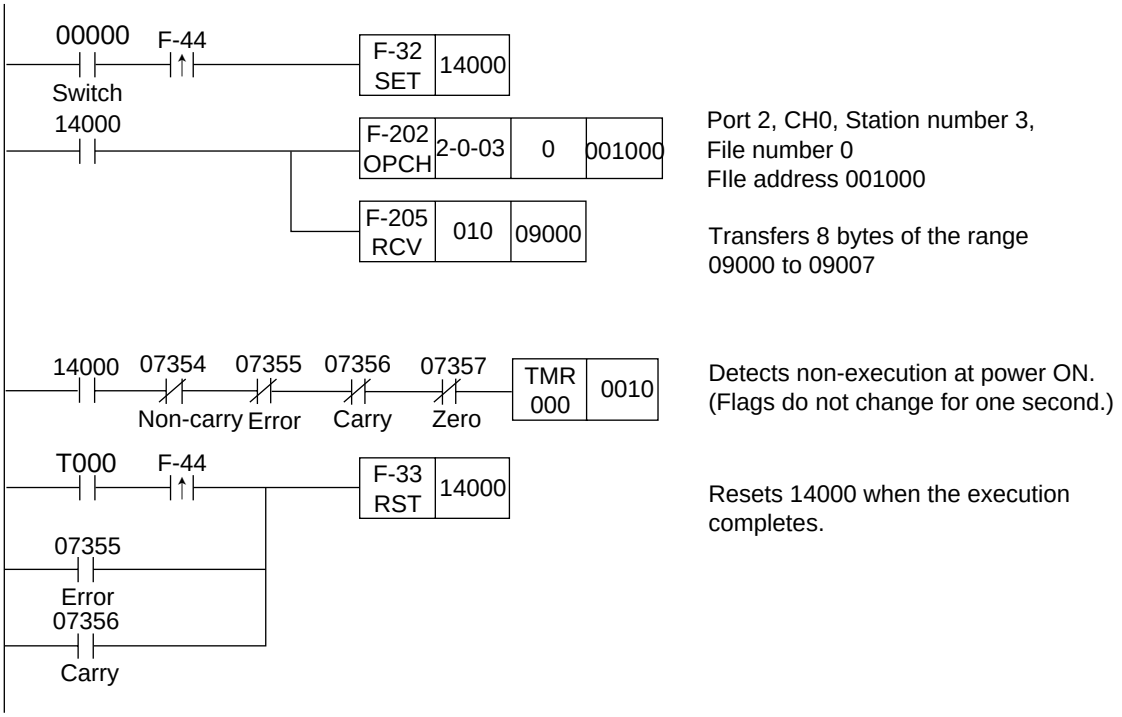

Flag status during and after the operation

| | Zero 07357 | Carry 07356 | Error 07356 | Non-carry 07356 | Explanation |
|---|---|---|---|---|---|
| No response from port | 0 | 0 | 1 | 0 | The JW-51CM is not installed in the slot. |
| 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 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 JW-51CM of source station : 2
Channel used : 0




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 $6000_{(H)}$.

8

**Note**

- 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 PC, 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 100 ms 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<br>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<br>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<br>(The contents is the same as CH0.) |
| 0026 to 0027 | Setting value of monitor timer for the send/receive functions CH1 UDP<br>(The contents is the same as CH0.) |
| 0030 to 0031 | Setting value of monitor timer for the send/receive functions CH2 TCP<br>(The contents is the same as CH0.) |
| 0032 to 0033 | Setting value of monitor timer for the send/receive functions CH2 UDP<br>(The contents is the same as CH0.) |
| 0034 to 0035 | Setting value of monitor timer for the send/receive functions CH3 TCP<br>(The contents is the same as CH0.) |
| 0036 to 0037 | Setting value of monitor timer for the send/receive functions CH3 UDP<br>(The contents is the same as CH0.) |

## [4] Other notes

① When the JW-51CM 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 10 seconds. So, allow 10 seconds 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.

# 8-2 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.

| File No. | File address |
|---|---|
| file 0 | 000000 to 017777(8) |
| file1 to 7 | 000000 to 177777(8) |

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

| Parameter address | Details |
|---|---|
| 3770 to 3773 | Top address of the send/receive function communication information storage area<br>3770, 3771: Top file address in the communication information storage area<br>3772: File number for the communication information storage area<br>3773: This information is effective when terminated by $80_{(H)}$. |

■ Parameters for "automatic/individual" registration

| Parameter address | Details | |
|---|---|---|
| 0400 | Assign a station number corresponding to the table (for SEND/RECEIVE)<br>Assignment of the relationship between the station number used with the send/receive function and the actual address.<br>If automatic is selected, the JW-51CM treats the target station numbers as node numbers on the IP addresses, and sets the target port to $6010_{(H)}$.<br>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.)<br>$01_{(H)}$: Automatic<br>$02_{(H)}$: Individual | |
| 0401 to 0407 | Reserved area | |
| 0410 to 0417 | Station number correspondence table 1.<br>This is effective only when parameter 0400 is $02_{(H)}$.<br>0410: Setting $00_{(H)}$: Not set (the information below will be ignored), $01_{(H)}$: Set<br>0411: Target station number<br>0412, 0413: Target station port number (assign a part as one word of data (two bytes) in octal notation)<br>0414 to 0417: Target station IP address (0417 is host ID) | |
| 0420 to 0427 | Station number corresponding table 2.<br>This is effective only when parameter 0400 is $02_{(H)}$. | Set the same as the station numbers in correspondence table 1. |
| 0430 to 0437 | Station number corresponding table 3.<br>This is effective only when parameter 0400 is $02_{(H)}$. | |
| ⋮ | ⋮ | |
| 0760 to 0767 | Station number corresponding table 36.<br>This is effective only when parameter 0400 is $02_{(H)}$. | |
| 0770 to 0777 | Station number corresponding table 37.<br>This is effective only when parameter 0400 is $02_{(H)}$. | |

## [3] Communication information storage area

| | | |
|---|---|---|
| +00 | FLAGS | Operation flag (same as ]0735 of F-204 and F-205)<br>$00_{(H)}$: When not in use<br>$90_{(H)}$: When communicating.<br>Until execution of the instruction is complete.<br>$40_{(H)}$: Normal end.<br>$60_{(H)}$: Abnormal end (communication time-out)<br>$E0_{(H)}$: Abnormal end (error response) |
| +01 | TIMER | Time-out time (unit: 100 ms)<br>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.<br>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).<br>n(L) is the lower byte.<br>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.

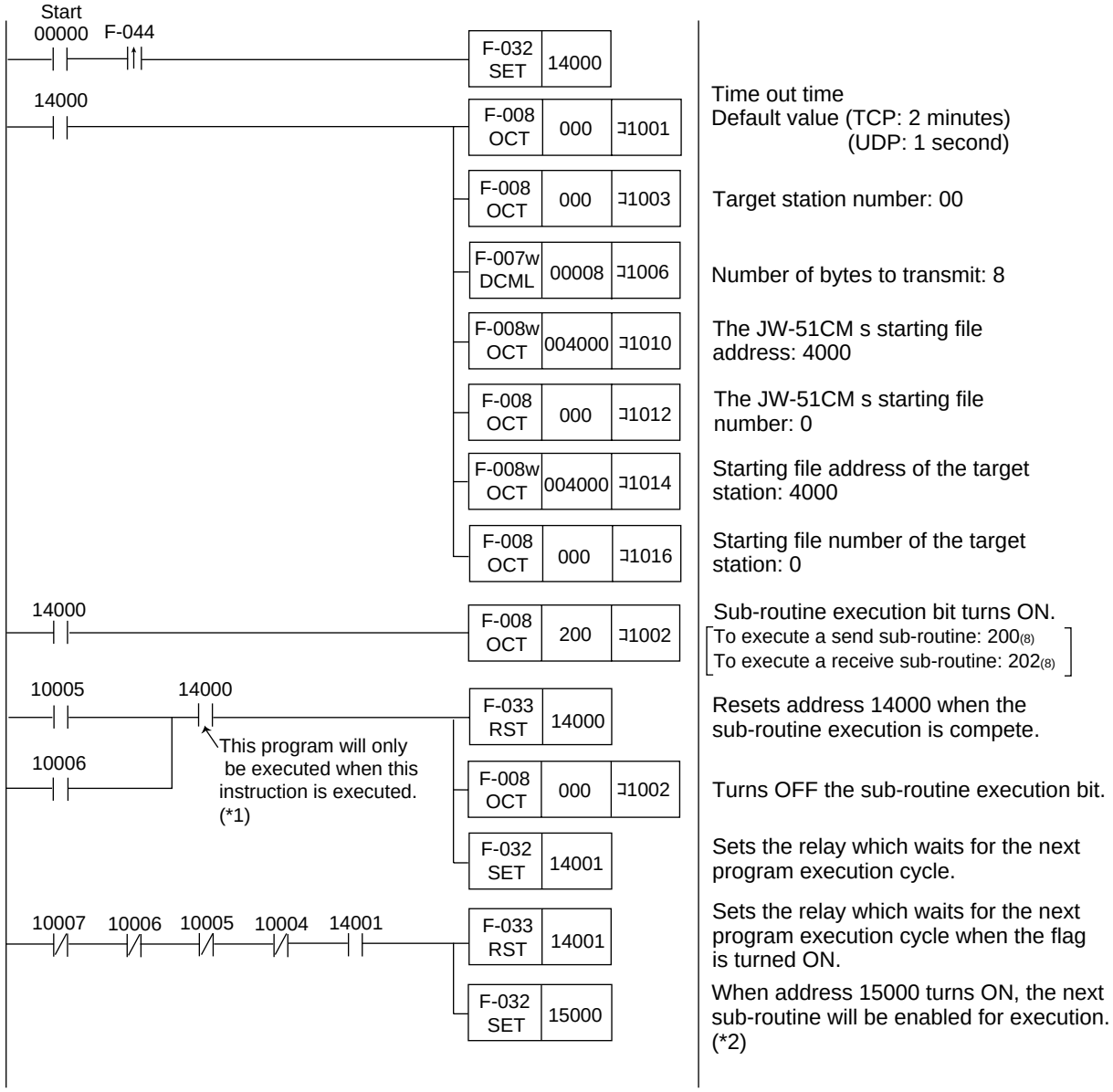
· Connection: SEG_B = 0, ADR_B = $FFFF_{(H)}$, n = 0
· Disconnection: SEG_B = 0, ADR_B = $FFFE_{(H)}$, n = 0

## [4] Other notes

① When the JW-51CM 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 10 seconds. So, allow 10 seconds 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




*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, the JW51CM 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 9: 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 (see page 9·3).

· 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.

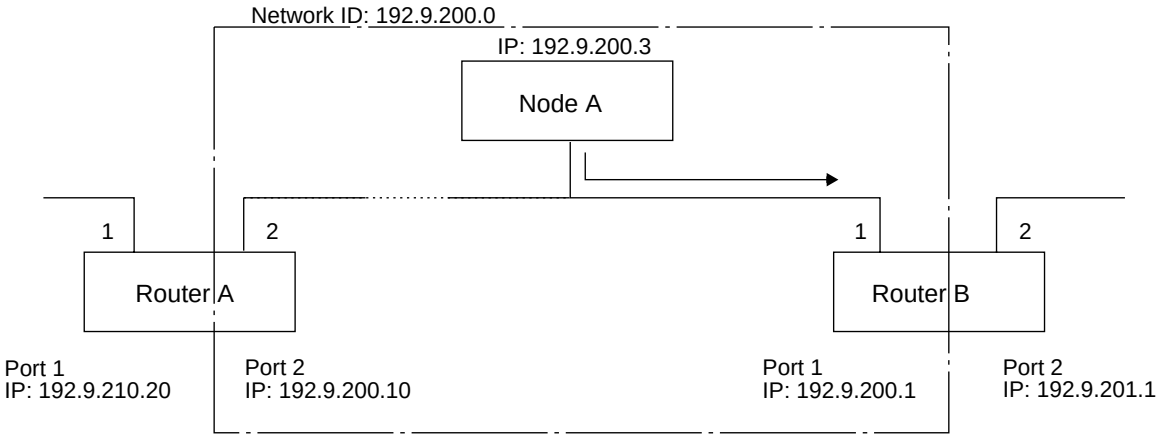
## [1] Create a default router

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<br>$00_{(H)}$: Without a default setting (the information below will be invalid);<br>$01_{(H)}$: 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 |

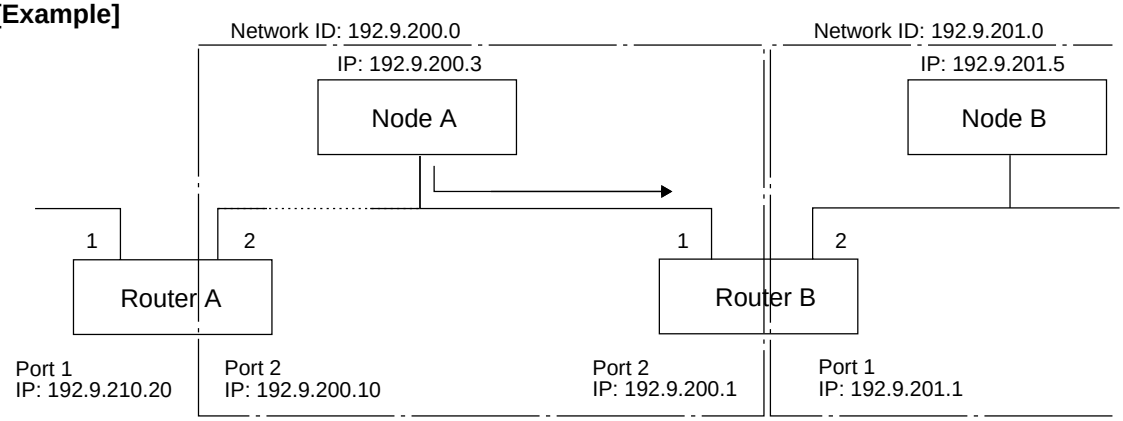
## [2] Create a customized routing table

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="8">1600 to 1607</td><td colspan="2">Routing table 0</td></tr>
<tr><td>1600</td><td>With or without customized routing table settings<br>00(H): Without customized settings<br>(the information below will be ignored)<br>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>
<tr><td>1610 to 1617</td><td>Routing table 1</td><td rowspan="8">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.

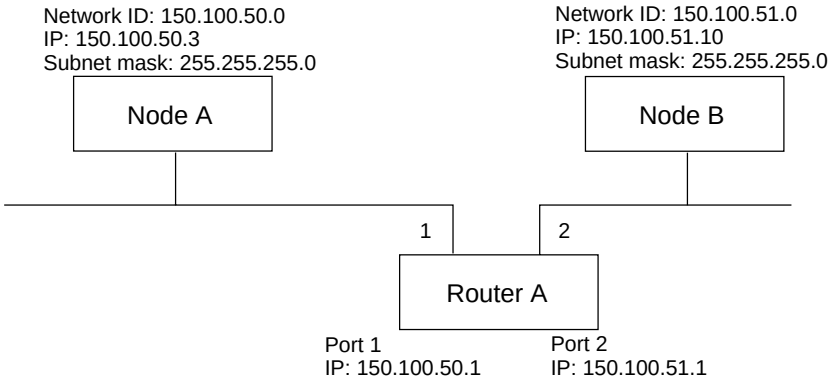
• To assign addresses in routing table 0

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

**Different network ID**

· Net ID refers to a network address. If a sub-net mask is used, the bit length will be assigned to the sub-net mask, not to each class. When the net ID assigned in this bit length is different, and if you want to communicate using this ID, a router is required. (IP address, subnet mask => See page 6-5 to 6-6.)

**[Example]**




Both node A and node B have the same net ID (16 bits width) as class B. However, the sub-net mask assigns net IDs as 24 bits. Therefore, their net IDs are different. A router is required to communicate between node A and node B.

# Chapter 10: Errors and Correction

## 10-1 Connection status monitor

The JW-51CM 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<br>$00_{(H)}$: Do not output<br>$80_{(H)}$: Output |

## 10-2 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 elaped, 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. |

## 10-3 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.

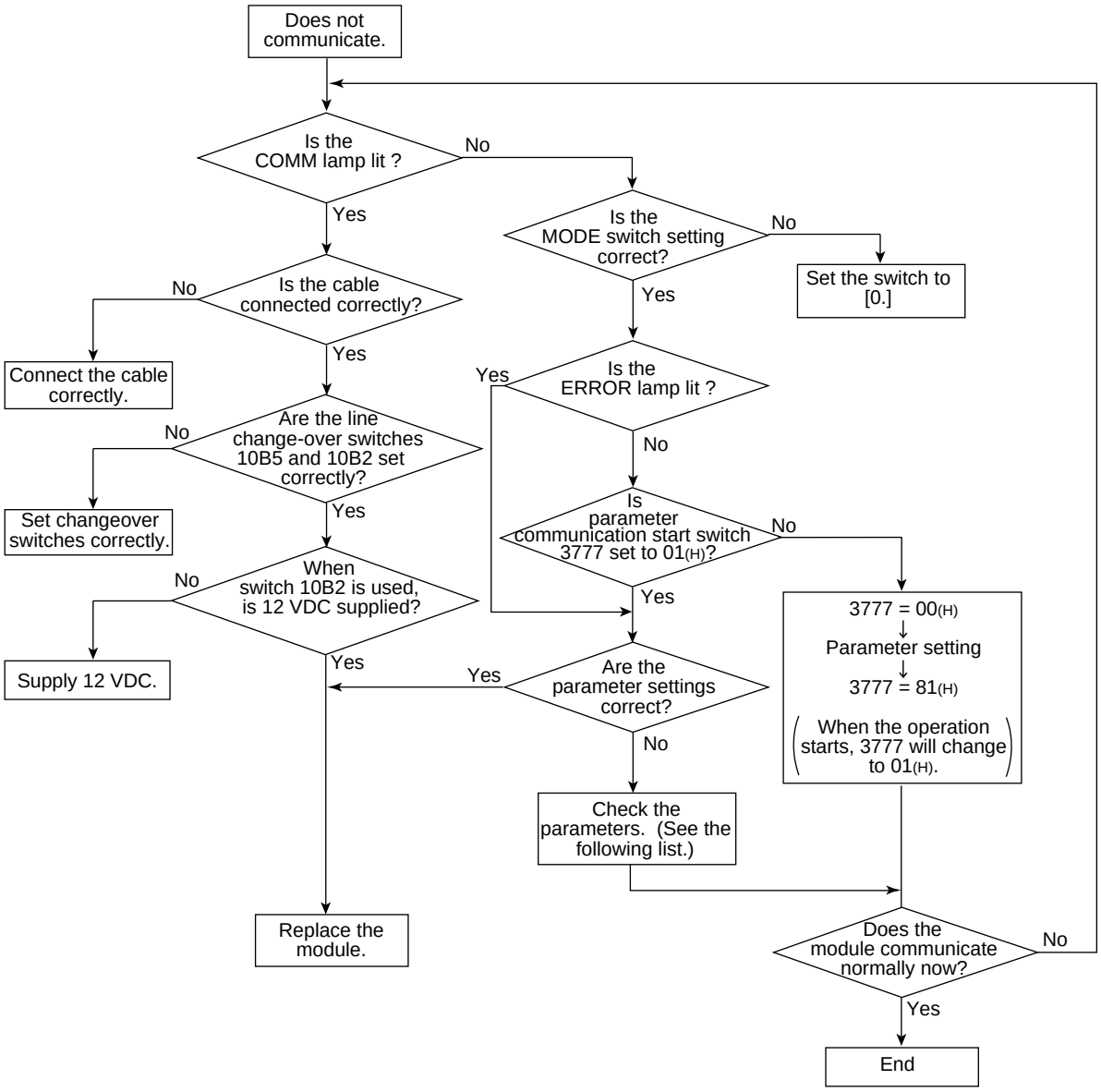
■ Keepalive timeout time parameter



| Parameter address | Details |
|---|---|
| 3714 to 3717 | Keepalive timeout time<br>Unit: ms (E.g.: If 100 is entered, the timeout time will be 100 ms.)<br>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. |

# 10-4 Troubleshooting

When the JW-51CM does not communicate normally, answer the questions in the following flow chart to identify the problem.




■ 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 should be specified in the instruction initiation station.)

# Chapter11: Network Parameter

## 11-1 Table of parameter

The network parameters are set on an EEPROM inside the 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)

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 A | B | C | D | E | Reference page |
|---|---|---|---|---|---|---|---|
| 0000<br>0001<br>0002<br>0003 | IP address of this module (0003 is on the host ID side) | ◎ | ◎ | ◎ | ◎ | ◎ | 6·4 |
| 0004 to 0007 | Subnet mask (when all of the bits are 0, the subnet mask is not used.) | ◎ | ◎ | ◎ | ◎ | ◎ | 6·6 |
| 0010 to 0017 | Reserved area | | | | | | — |
| 0020 to 0021 | Value set for the send/receive function monitor timer CH0 TCP<br>Set the monitor timer value using an application, when CH0 of the send/receive function is used for TCP. (Binary value) Unit: 100 ms.<br>If $000_{(H)}$ is entered, this parameter will be set to the default value (2 minutes). | △ | | | | | 8·7 |
| 0022 to 0023 | Value set for the send/receive function monitor timer CH0 UDP<br>Set the monitor timer value using an application, when CH0 of the send/receive function is used for UDP. (Binary value) Unit: 100 ms.<br>If $000_{(H)}$ is entered, this parameter will be set to the default value (1 minutes). | △ | | | | | |
| 0024 to 0025 | Value set for the send/receive function monitor timer CH1 TCP<br>(the details are the same as for CH0). | △ | | | | | |
| 0026 to 0027 | Value set for the send/receive function monitor timer CH1 UDP<br>(the details are the same as for CH0). | △ | | | | | |
| 0030 to 0031 | Value set for the send/receive function monitor timer CH2 TCP<br>(the details are the same as for CH0). | △ | | | | | |
| 0032 to 0033 | Value set for the send/receive function monitor timer CH2 UDP<br>(the details are the same as for CH0). | △ | | | | | |
| 0034 to 0035 | Value set for the send/receive function monitor timer CH3 TCP<br>(the details are the same as for CH0). | △ | | | | | |
| 0036 to 0037 | Value set for the send/receive function monitor timer CH3 UDP<br>(the details are the same as for CH0). | △ | | | | | |
| 0040 to 0077 | Reserved area | | | | | | — |
| 0100 to 0103 | Settings for connection 0<br>0100: Open method $00_{(H)}$: TCP_Passive, $80_{(H)}$: TCP_Active, $01_{(H)}$: UDP<br>0101: $00_{(H)}$<br>0102, 0103: Port number of the source station (0102 as low bit, 0103 as high bit) | ○ | ○ | ○ | ○ | ○ | 6·4 |

11

| Parameter address | Details | Function A | B | C | D | E | F | Reference page |
|---|---|---|---|---|---|---|---|---|
| 0104 to 0107 | Settings for connection 1 (the details are the same as for connection 0) | | | | | | | |
| 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) | ○ | ○ | ○ | ○ | ○ | ○ | 6·4 |
| 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)<br>Assignment of the relationship between the station number used with the send/receive function and the actual address. If “automatic” is selected, the JW-51CM treats the target station numbers as node numbers on the IP addresses, and sets the target port to $6010_{(H)}$.<br>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.)<br>$01_{(H)}$: Automatic<br>$02_{(H)}$: Individual | ◎ | ◎ | | | | | |
| 0401 to 0407 | Reserved area | | | | | | | |
| 0410 to 0417 | Station number correspondence table 1.<br>This is effective only when parameter 0400 is $02_{(H)}$.<br>0410: Setting $00_{(H)}$: Not set (the information below will be ignored) / $01_{(H)}$: Set<br>0411: Target station number<br>0412, 0413: Target station port number (assign as one word of data (two bytes) in decimal notation)<br>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 $02_{(H)}$. | | | | | | | 8·2 |
| 0460 to 0467 | Station number correspondence table 6. This is effective only when parameter 0400 is $02_{(H)}$. | | | | | | | 8·8 |
| 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 $02_{(H)}$. | | | | | | | |

<table>
<tr><th rowspan="2">Parameter address</th><th rowspan="2">Details</th><th colspan="5">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></tr>
<tr><td>1000 to 1007</td><td>The information concerning standard buffer 00
<table>
<tr><th></th><th>Direction assignment (1007 = 80(H))</th><th>Indirect assignment (1007 = C0(H))</th></tr>
<tr><td>1000</td><td rowspan="2">Top file address for the standard buffer storage area</td><td rowspan="2">Top file address for the standard buffer information</td></tr>
<tr><td>1001</td></tr>
<tr><td>1002</td><td>Standard buffer’s file number</td><td>File number for the standard buffer’s 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 (0000(H) as 64 K-bytes)</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<br>00(H): Invalid standard buffer<br>80(H): Direct assignment of the standard buffer<br>C0(H): Indirect assignment of the standard buffer</td></tr>
</table></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1010 to 1017</td><td>The information concerning standard buffer 01</td><td rowspan="32">Set the same way as the information concerning standard buffer 00</td><td rowspan="32"></td><td rowspan="32"></td><td rowspan="32">○</td><td rowspan="32"></td><td rowspan="32">7·25</td></tr>
<tr><td>1020 to 1027</td><td>The information concerning standard buffer 02</td></tr>
<tr><td>1030 to 1037</td><td>The information concerning standard buffer 03</td></tr>
<tr><td>1040 to 1047</td><td>The information concerning standard buffer 04</td></tr>
<tr><td>1050 to1057</td><td>The information concerning standard buffer 05</td></tr>
<tr><td>1060 to 1067</td><td>The information concerning standard buffer 06</td></tr>
<tr><td>1070 to 1077</td><td>The information concerning standard buffer 07</td></tr>
<tr><td>1100 to 1107</td><td>The information concerning standard buffer 08</td></tr>
<tr><td>1110 to 1117</td><td>The information concerning standard buffer 09</td></tr>
<tr><td>1120 to 1127</td><td>The information concerning standard buffer 0A</td></tr>
<tr><td>1130 to 1137</td><td>The information concerning standard buffer 0B</td></tr>
<tr><td>1140 to 1147</td><td>The information concerning standard buffer 0C</td></tr>
<tr><td>1150 to 1157</td><td>The information concerning standard buffer 0D</td></tr>
<tr><td>1160 to 1167</td><td>The information concerning standard buffer 0E</td></tr>
<tr><td>1170 to 1177</td><td>The information concerning standard buffer 0F</td></tr>
<tr><td>1200 to 1207</td><td>The information concerning standard buffer 10</td></tr>
<tr><td>1210 to 1217</td><td>The information concerning standard buffer 11</td></tr>
<tr><td>1220 to 1227</td><td>The information concerning standard buffer 12</td></tr>
<tr><td>1230 to 1237</td><td>The information concerning standard buffer 13</td></tr>
<tr><td>1240 to 1247</td><td>The information concerning standard buffer 14</td></tr>
<tr><td>1250 to 1257</td><td>The information concerning standard buffer 15</td></tr>
<tr><td>1260 to 1267</td><td>The information concerning standard buffer 16</td></tr>
<tr><td>1270 to 1277</td><td>The information concerning standard buffer 17</td></tr>
<tr><td>1300 to 1307</td><td>The information concerning standard buffer 18</td></tr>
<tr><td>1310 to 1317</td><td>The information concerning standard buffer 19</td></tr>
<tr><td>1320 to 1327</td><td>The information concerning standard buffer 1A</td></tr>
<tr><td>1330 to 1337</td><td>The information concerning standard buffer 1B</td></tr>
<tr><td>1340 to 1347</td><td>The information concerning standard buffer 1C</td></tr>
<tr><td>1350 to 1357</td><td>The information concerning standard buffer 1D</td></tr>
<tr><td>1360 to 1367</td><td>The information concerning standard buffer 1E</td></tr>
<tr><td>1370 to 1377</td><td>The information concerning standard buffer 1F</td></tr>
</table>

<table>
<tr><th>Parameter address</th><th>Details</th><th colspan="5">Function</th><th>Reference page</th></tr>
<tr><th></th><th></th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th></th></tr>
<tr><td>1400 to 1407</td><td>
Information concerning ring buffer 00
<table>
<tr><td>1400<br>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="4">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</td><td colspan="3">Reading data from the control module into the JW-51CM</td></tr>
<tr><td>81</td><td colspan="3">Writing data from the JW-51CM into the control module</td></tr>
<tr><td rowspan="6">1404</td><td colspan="4">Ring buffer’s top address (file address upper byte)<br>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="8">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<br>00(H): Invalid ring buffer<br>80(H): Direct assignment of ring buffer<br>C0(H): Indirect assignment of ring buffer</td></tr>
</table>
</td><td></td><td></td><td></td><td>○</td><td></td><td>7·38</td></tr>
<tr><td>1410 to 1417</td><td>The information concerning ring buffer 01</td><td colspan="6" rowspan="12">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><td colspan="6" rowspan="3">Set the same way as the information concerning ring buffer 00</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">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>Routing table 0<br><table><tr><td>1600</td><td>With or without customized routing table settings<br>00(H): Without customized settings (the information below will be ignored)<br>01(H): With customized settings</td></tr><tr><td>1601<br>1602<br>1603</td><td>Target network ID</td></tr><tr><td>1604<br>1605<br>1606<br>1607</td><td>IP address in the router corresponding to the network ID<br>(1607 is always the host ID location)</td></tr></table></td><td></td><td></td><td></td><td></td><td></td><td>○</td><td>9·2</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 rowspan="7"></td><td rowspan="7"></td><td rowspan="7"></td><td rowspan="7"></td><td rowspan="7"></td><td rowspan="7"></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>
<tr><td>1700 to 1707</td><td><table><tr><td>1700</td><td>With and without the default router settings<br>00(H): Without a default setting (the information below will be invalid)<br>01(H): With the default settings</td></tr><tr><td>1701 to 1703</td><td>Not used</td></tr><tr><td>1704<br>1705<br>1706<br>1707</td><td>Default router IP addresses (address 1707 is the host ID)</td></tr></table></td><td></td><td></td><td></td><td></td><td></td><td>○</td><td>9·1</td></tr>
<tr><td>1710 to 3657</td><td>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>Setting command execution completion information storage area<br><table><tr><td>3660<br>3661</td><td>Top file address occupied by the command execution completion information</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<br>3665</td><td>Size of the command execution completion information (number of bytes)<br>At least 16 bytes should be assigned. Maximum 64 bytes.</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>7·54</td></tr>
<tr><td>3670 to 3677</td><td>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>Minimum value for the retransmission timeout time. Unit: ms.<br>The default setting (0 ms) is assigned when 0 is entered in this parameter.</td><td></td><td></td><td></td><td></td><td></td><td rowspan="4"></td><td rowspan="4">10·2</td></tr>
<tr><td>3704 to 3707</td><td>Maximum value for the retransmission timeout time. Unit: ms. The default setting (240000 ms) is assigned when 0 is entered in this parameter.</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3710 to 3713</td><td>Initial value for the retransmission timeout time. Unit: ms.<br>The default setting (3000 ms) is assigned when 0 is entered in this parameter.</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3714 to 3717</td><td>Keepalive timeout time<br>Unit: ms (E.g.: If 100 is entered, the timeout time will be 100 ms.)<br>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></tr>
<tr><td>3720 to 3763</td><td>Reserved area</td><td></td><td></td><td></td><td></td><td></td><td></td><td>—</td></tr>
</table>

11

| Parameter address | Details | Function A | B | C | D | E | F | Reference page |
|---|---|---|---|---|---|---|---|---|
| 3764 to 3767 | Setting connection status monitor flag<br>3764, 3765: File address of the connection status monitor flag<br>3766: File number of the connection status monitor flag<br>3767: Flag output<br>$00_{(H)}$: Do not output, $80_{(H)}$: Output | △ | △ | △ | △ | △ | △ | 10·1 |
| 3770 to 3773 | Assigning the send/receive function storage area for communication information<br>3770, 3771: Top file address of the storage area for communication information<br>3772: File number of the storage area for communication information<br>3773: This information is valid when $80_{(H)}$ is entered for this parameter | | | ◎ | | | | 8·8 |
| 3774 to 3775 | Reserved area | | | | | | | — |
| 3776 | BCC (Block check code)<br>Add 8 bits data to 0000 through 3775 and take the 2's complement from this value. (The JW-51CM automatically calculates and stores this value.) | | | | | | | — |
| 3777 | Communication start switch<br>$00_{(H)}$: Stop communication<br>$01_{(H)}$: Check the parameters, check the BCC, and start operation<br>$08_{(H)}$: Initialize the parameters (All parameter = $00_{(H)}$)<br>$80_{(H)}$: Check the parameters, create a BCC, write the EEPROM, and stop operation<br>(After writing $80_{(H)}$ to this address, if it changes to $00_{(H)}$, the communication has ended normally.)<br>$81_{(H)}$: Check the parameters, create a BCC, write the EEPROM, and stop operation<br>(After writing $81_{(H)}$ to this address, if it changes to $01_{(H)}$, the communication has ended normally.) | ◎ | ◎ | ◎ | ◎ | ◎ | ◎ | 6·4 |

| Parameter address | Details |
|---|---|
| 4000 to 4005 | MAC address (read only)<br>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<br>Outputs the setting value (0 to 6) of the module ID No. switch of this module. |

11

# 11-2 Setting procedure of parameters

■ Example of settings

· EIP 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

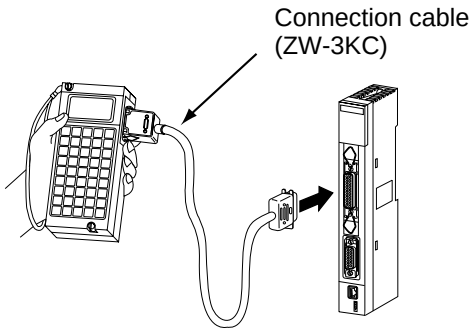
· Send/receive station number correspondence : Individually set the corresponding relation of station number $13_{(8)}$ of the instruction to the IP address 192.9.200.4 and the port 3001.

· Set the connection status monitoring flag to ]0740.

| Parameter address | Set value (hex) | Setting contents | Contents |
|---|---|---|---|
| 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 24567 | |
| 0107 | 60 | ($6000_{(H)}$) | |
| : | 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 | |
| 0413 | 0B | No. 3001 | |
| 0414 | C0 | Target station 192 | |
| 0415 | 09 | IP address 9 | |
| 0416 | C8 | 200 | |
| 0417 | 04 | 4 | |
| : | 00 | | |
| 3764 | E0 | Address ]0740 | Connection status monitoring flag |
| 3765 | 01 | (File address 000740) | |
| 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 PC 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.

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
```

11

(9) Set the station number correspondence table 1.

クリア CLR / E 4 / B 1 / A 0 / モニタ MNTR

B 1 / 書込 ENT / STEP (+)

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

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

FORCE LNGTH / 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 (+)

E 4 / 書込 ENT

Hexadecimal notation of the parameter 0410

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

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

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

Parameter 0414 = $192_{(10)}$

Parameter 0415 = 9

Parameter 0416 = 200

Parameter 0417 = 4

```
 00406  D    00000
 00410  D    02817
I PARAM.
>00412  D    03001
```

```
 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

7 / E 4 / A 0 / 書込 ENT / STEP (+)

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

SET 8 / A 0 / 書込 ENT

Octal notation word of the parameters 3764 and 3765.
Write 740 in octal notation.

Parameter 3766 = 00

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

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.

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

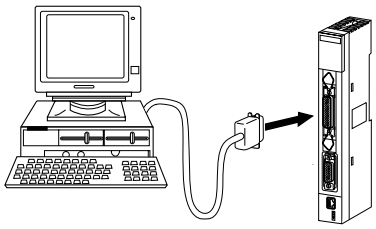
(12) Get the PC in operating condition.

クリア CLR / * / * / MNTR MODE / SET 8

```
M  00000
>
```

## [2] Setting procedures using the JW-50SP

Described below is an the outline of how to set, write, and store the JW-51CM parameters, using the JW-50SP ladder software (for IBM/PC).
For the details about the operation, see the instruction manual for the ladder software.



Connection cable (ZW-3KC)
+
Converter
(supplied with JW-50SP)

Select the PC model JW50H/JW70H/100H whose parameters you want to set.
[Main menu] ➡ 1: Program edit ➡ 1: Model selection ➡ 2: Enter model name (JW50H/JW70H/100H) ➡ 0: Executes

**(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 the JW-51CM**

Connect the personal computer to the JW-51CM.
[Main menu] ➡ 7: PC transfer ➡ 2: Write ➡ 7: Remote station PARAM. ➡ 7: PC stop (Place the CU in the stop mode.) ➡ 2: Execution stop (Stop the 51CM operation.) ➡ 1: Parameter writing (Transmit the parameters to the 51CM.) ➡ 5: Start: write the EEPROM (Write the transmitted parameters to the EEPROM in the 51CM and start its operation.) ➡ 6: PC run (Place the CU in the run mode.)

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

**(3) Reading the parameters from the JW-51CM**

Connect the personal computer to the JW-51CM.
[Main menu] ➡ 7: PC transfer ➡ 3: Read ➡ 7: Remote master station PARAM. ➡ 7: PC stop (Place the CU in the stop mode.) ➡ 2: Execution stop (Stop the 51CM operation.) ➡ 1: Read out PARAM (Transmit the parameters from the 51CM.) ➡ 4: Start: read (Reading operation starts.) ➡ 6: PC run (Place the CU in the run mode.)

**(4) Recording the parameters on a floppy disc**

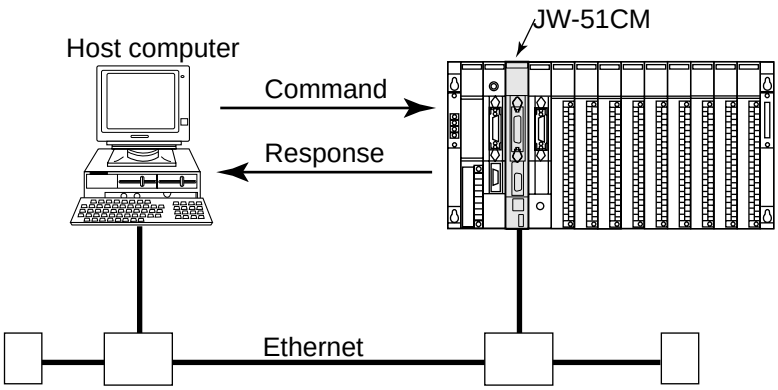
[Main menu] ➡ 6: FD transfer ➡ 1: Save ➡ 9: Master station PARAM. ➡ Yes ➡ F1 (Drive) ➡ Specify drive/directory ➡ Enter a file name ➡ Enter comment ➡ 0: Execute

**(5) Restoring the parameters from a floppy disc**

[Main menu] ➡ 6: FD transfer ➡ 2: Load ➡ 9: Remote master station parameter ➡ Execute ➡ F1 (Drive) ➡ Specify drive/directory ➡ Enter a file name using the space key ➡ 0: Execute

# Chapter 12: Sample Program

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




Shown above is an example of communication using the TCP method of communication.
By entering a host name and port number for the JW-51CM, the host computer can establish contact with the JW-51CM.
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 the JW-51CM'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 the JW-51CM must first be registered on the host computer.

405 Establish a connection with the JW-51CM.

- 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 (51CM) 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. (See the 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 the JW-51CM. Use the "send" function.
- 371 Receives the response from the JW-51CM.
- 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 the JW-51CM maintains the connection, even if the host performs abnormal disconnection. In this condition, the JW-51CM will refuse a request for another connection.
If you use assignment method a), the JW-51CM can recognize that the another request for connection is being made by the same port number. Therefore, the JW-51CM 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                      *
 *******************************************************/


```

```
115  void so_perror(char *str, int err)
116  {
117          int i;
118
119          for(i = 0; i < 16; ++i)
120                  if(err == errlist[i].errno)
121                          break;
122          if(i < 16)
123                  printf("%s: %s           \n", str, errlist[i].errmsg);
124          else
125                  printf("%s: unknown error\n");
126  }
127
128
129
130  /*********************************************************
131   *                        Establish a connection                        *
132   *********************************************************/
133
134
135  int comopen(unsigned long ip, int port)
136  {
137  struct sockaddr_in myaddr;
138  struct sockaddr_in youraddr;
139  int s;
140
141          s = socket(PF_INET, SOCK_STREAM, 0);    /* Create a socket (TCP) */
142
143          if(s == -1) {
144                  so_perror("socket", errno);
145                  soclose(s);
146                  return(-1);
147          }
148          youraddr.sin_family = AF_INET;                  /* Store the target station address in the address structure */
149          youraddr.sin_port = port;
150          youraddr.sin_addr.s_addr = ip;
151
152          myaddr.sin_family = AF_INET;
153          myaddr.sin_port = htons(4000);
154                                                  /* Use 4000 for the source station port number */
155              if(bind(s, (struct sockaddr *)&myaddr, sizeof (myaddr)) < 0) {
156                  so_perror("bind", errno);
157                  soclose(s);
158                  return(-2);
159          }
160                                                          /* Make the connect */
161          if(connect (s, (struct sockaddr *) & youraddr, sizeof (youraddr)) < 0) {
162                  so_perror("connect", errno);
163                  soclose(s);
164                  return(-2);
165          }
166          return(s);
167  }
168
169
170  /*********************************************************
171   *                        Close the connection                        *
```

```
172     ********************************************************/
173
174   void comclose(int s)
175   {
176           shutdown(s, 1);
177           soclose(s);
178   }
179
180
181
182   /*********************************************************
183    *    Convert one hexadecimal ASCII character into binary          *
184    *********************************************************/
185
186   char a2b_1c(char data)
187   {
188           return(isdigit(data) ? data - '0' :
189                   (isupper(data) ? data - 'A' + 10 : data - 'a' + 10));
190   }
191
192
193
194   /*********************************************************
195    *    Convert two hexadecimal ASCII characters into binary         *
196    *********************************************************/
197
198   int a2b(char *ascbuf, char *bindata)
199   {
200           if(isxdigit(ascbuf[0]) && isxdigit(ascbuf[1])) {
201                   *bindata = a2b_1c(ascbuf[0]) * 16 + a2b_1c(ascbuf[1]);
202                   return(0);
203           } else
204                   return(-1);
205   }
206
207
208   /*********************************************************
209    *           Convert ASCII character strings into binary               *
210    *********************************************************/
211
212   int asc2bin(char *ascbuf, char *binbuf)
213   {
214   int a, i, j;
215           for(i = 0, j = 0; ascbuf[i] != 0; j++, i++) {
216                   a = a2b(&ascbuf[i], &binbuf[j]);
217                   if(a < 0)
218                           return(a);
219                   i++;
220           }
221           return(j);
222   }
223
224
225   /*********************************************************
226    *   Convert 4 bits binary into hexadecimal ASCII character       *
227    *********************************************************/
228
```

```
229  char b2a_1c(char data)
230  {
231          return((data < 10) ? data + '0' : data + 'A' - 10);
232  }
233
234
235  /*********************************************************
236   *    Convert 8 bits binary into 2 hexadecimal ASCII characters    *
237   *********************************************************/
238
239  void b2a(char bindata, char *ascbuf)
240  {
241  char a;
242          a = (bindata >> 4) & 0xf;
243          ascbuf[0] = b2a_1c(a);
244          a = bindata & 0xf;
245          ascbuf[1] = b2a_1c(a);
246  }
247
248  /*********************************************************
249   *         Convert binary code into an ASCII character string          *
250   *********************************************************/
251
252  void bin2asc(char *binbuf, char *ascbuf, int len)
253  {
254  int i, j;
255
256          for(j = 0, i = 0 ; i < len; i++) {
257                  b2a(binbuf[i], &ascbuf[j]);
258                  j += 2;
259          }
260
261  }
262
263  /*********************************************************
264   *                  Place a command in the send buffer                  *
265   *********************************************************/
266
267  void set_command(char *cbuf, int len)
268  {
269  int i;
270          for(i = 0; i < HEADLEN; i++)
271                  sendbuf.s_upper.header[i] = theader[i];
272
273          for(i = 0; i < len; i++)
274                  sendbuf.s_upper.cl_command_frame[i] = cbuf[i];
275  }
276
277  /*********************************************************
277   *                           Receive the command                       *
279   *********************************************************/
280
281  int get_command_default(char *kbuf, char *cbuf)
282  {
283  char cntbuf[32];
284  char intbuf[32];
285
```

```
286         return(asc2bin(kbuf, cbuf));
287 }
288
289
290 /*********************************************************
291  *                          Display response                          *
292  *********************************************************/
293
294 void disp_response(char *buf, int len)
295 {
296 int i;
297         bin2asc(receivebuf.r_upper.cl_command_frame, buf, len);
298         buf[2*len] = 0;
299         printf("response = ");
300         puts(buf);
301
302 }
303
304
305 /*********************************************************
306  *                          Display command                          *
307  *********************************************************/
308
309 void disp_command(char *buf)
310 {
311         printf("command = ");
312         puts(buf);
313 }
314
315
316 /*********************************************************
317  *                          Receive response                          *
318  *********************************************************/
319
320 int receive_response(int s)
321 {
322 fd_set readfds;
323 struct timeval tout;
324 int rlen, n;
325 char cbuf[1024];
326 char dbuf[1024];
327
328
329         FD_ZERO(&readfds);
330         FD_SET(s, &readfds);
331         tout.tv_sec = 1;            /* Set the time-out value to one second */
332
333         n = select(32, &readfds, NULL, NULL, &tout);
334         if(n > 0) {                 /*If received, OK*/
335                 if(FD_ISSET(s, &readfds))
336                         rlen = recv(s, receivebuf.r_socket.buf, BUFLEN, 0);
337         } else {
338                 so_perror("select", errno);
339                 return(-1);
340         }
341         rlen -= HEADLEN;
342         disp_response(dbuf, rlen);
```

```
343             return(0);
344     }
345
346
347     /*********************************************************
348      *                          Communication process                          *
349      *********************************************************/
350
351     int communication(int s)
352     {
353     char kbuf[1024] = "4700240002080200";       /* Command */
354     char cbuf[1024];                    /* Command (binary) */
355     int data_len, r;
356     unsigned int i;
357
358             data_len = get_command_default(kbuf, cbuf);
359             set_command(cbuf, data_len);
360             data_len += HEADLEN;
361
362             for (i = 0; i < 5; i++) {
363                     disp_command(kbuf);
364                                     /* Send to the target station */
365             r = send(s, sendbuf.s_socket.buf, data_len, 0);
366             if (r != data_len) {  /* If the data cannot be sent normally, “error” is returned. */
367                             so_perror("send", errno);
368                             return(-1);
369                     }
370                                     /* Receive the response */
371                     if (receive_response(s) < 0)
372                             return(-1);
373             }
374     }
375
376     /*********************************************************
377      *                                                                       *
378      *                              Main routine                             *
379      *                                                                       *
380      *********************************************************/
381
382     void main(int argc, char *argv[])
383     {
384     struct hostent *hp;  /* Define the name structure */
385     unsigned long ipaddr;  /* IP address */
386     int portno;             /* Port number */
387     int s;                  /* Socket identifier */
388
389             if (argc < 2) {
390                     printf("CLTEST name port\n");
391                     printf("  name : Target station name\n");
392                     printf("  port : Target port number\n");
393                     exit(1);
394             }
395                                     /* Get IP address from the name */
396             hp = gethostbyname(argv[1]);
397             if (hp == NULL) {
398                     printf("%s: Undefined host\n",argv[1]);
399                     exit(1);
```

```
400             }
401             ipaddr = *(unsigned long *)hp->h_addr;
402             portno = htons(atoi(argv[2]));
403
404                                     /* Establish a connection */
405             if (comopen(ipaddr, portno) < 0)
406                     exit(1);
407
408             printf("Complete connection to target station =  %s\n", argv[1]);
409             printf("          Complete connection to port = %s\n", argv[2]);
410
411                                     /* Communication processing */
412             communication(s);
413                                     /* Disconnect */
414             comclose(s);
415
416     }
```

# Chapter 13: Specifications

## 13-1 General specifications

| Item | Specifications |
|---|---|
| PC to install | Install into optional slots of the JW50H/70H/100H (up to 6 modules) |
| 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 hours each in X , Y, and Z directions) |
| Shock proof | JIS C 0912 or equivalent 98 $m/s^2$ (three times each in X, Y, and Z directions) |
| Internal current consumption (5 VDC) | 400 mA |
| External power supply capacity | 12 VDC ±5%, 0.5 A (when only using the 10BASE5) |
| Weight | Approx. 380 g |
| Accessories | One cable, one instruction manual |

## 13-2 Communication specifications

| Item | | Specifications |
|---|---|---|
| Connection with network | | Either 10BASE5 or 10BASE-T. |
| Transfer speed | | 10M bits/s |
| Physical topology | | Bus (10BASE5)/Star (10BASE-T) |
| Transmission device | | 50 ohm yellow cable (10BASE5), Twisted pair cable (10BASE-T) |
| Transmission method | | Baseband |
| Max. No. of transfer length | | 10BASE5 500 m/segments, 2.5 km/network *Note 1<br>10BASE-T 100 m/segments, 500 m/network *Note 2 |
| Station interval | | 2.5 m or more |
| Max. No. of stations | | 100 sets/segments |
| Protocol structure | Application | Sharp computer link/original command |
| | Transport | TCP/UDP |
| | Network | IP |
| | Data link | Ethernet V2 |
| No. of connections | | 8 |
| Application | | Computer link functtion,send/recieve functions. |

*Note 1: The max. number of transfer length between stations connecting multi segments with the repeater.

*Note 2: The maximum data transfer distance between stations when multiple 10BASE-T segments are connected using a hub.

# 13-3 Outside dimensions