HITACHI

S10mini

HARDWARE MANUAL

OPTION

FL.NET

SME-1-101 (C)

S10mini
HARDWARE MANUAL

OPTION
FL.NET

First Edition, September 2001, SME-1-101(A) (out of print)
Second Edition, July 2003, SME-1-101(B) (out of print)
Third Edition, October 2008, SME-1-101(C)

Printed in Japan.

BI-NR-HS<IC-NS> (FL-MW20, AI8.0)

This manual provides information for the following hardware product:

<Hardware product>
FL.NET (LQE000)

<Changes added to this manual>

Description of added changes	Page
Subsection 7.1.1, “Replacing or adding on the module” is newly added.	7-3

In addition to the above changes, all the unclear descriptions and typographical errors found are also corrected without prior notice.

Revision record

Revision No.	Revision Record (revision details and reason for revision)	Month, Year	Remarks
A	First Edition	September 2001	
C	Subsection 7.1.1, “Replacing or adding on the module” is newly added.	October 2008	

PREFACE

Thank you for purchasing the FL.NET module, which is an option for use with the S10mini.
This manual, named “HARDWARE MANUAL OPTION FL.NET,” describes how to use the FL.NET module. For proper use of the FL.NET module, it is requested that you thoroughly read this manual.

The S10mini products are available in two types: standard model and environmentally resistant model. The environmentally resistant model has thicker platings and coatings than those for the standard model.
The model number of the environmentally resistant model is marked by adding the suffix “-Z” to the model number of the standard model.

(Example) Standard model: LQE000
Environmentally resistant model: LQE000-Z

This manual is applicable to both the standard model and environmentally resistant models.
Although the descriptions contained in this manual are based on the standard model, follow the instructions set forth in this manual for proper use of the product even if you use the environmentally resistant model.

<Trademarks>

- Microsoft® Windows® operating system, Microsoft® Windows® 95 operating system, Microsoft® Windows® 98 operating system, Microsoft® Windows® 2000 operating system, Microsoft® Windows® XP operating system are registered trademarks of Microsoft Corporation in the United States and/or other countries.

<Note for storage capacity calculations>

- Memory capacities and requirements, file sizes and storage requirements, etc. must be calculated according to the formula 2^n. The following examples show the results of such calculations by 2^n (to the right of the equals signs).
  1 KB (kilobyte) = 1,024 bytes
  1 MB (megabyte) = 1,048,576 bytes
  1 GB (gigabyte) = 1,073,741,824 bytes
- As for disk capacities, they must be calculated using the formula 10^n. Listed below are the results of calculating the above example capacities using 10^n in place of 2^n.
  1 KB (kilobyte) = 1,000 bytes
  1 MB (megabyte) = $1{,}000^2$ bytes
  1 GB (gigabyte) = $1{,}000^3$ bytes

CONTENTS

FIGURES

TABLES

1 OVERVIEW

1.1 What Is FL-net

FL-net is an open FA (factory automation) network that is standardized by the Japan FA Open Systems Promotion Group (JOP) of the Manufacturing Science and Technology Center, which is an organization affiliated to the Ministry of International Trade and Industry.
As shown in Figure 1-1, this network can interconnect programmable controllers (PLCs), computer numerical control (CNC) devices, and various other factory automation controllers and personal computers of many different brands to exercise control and monitoring functions.

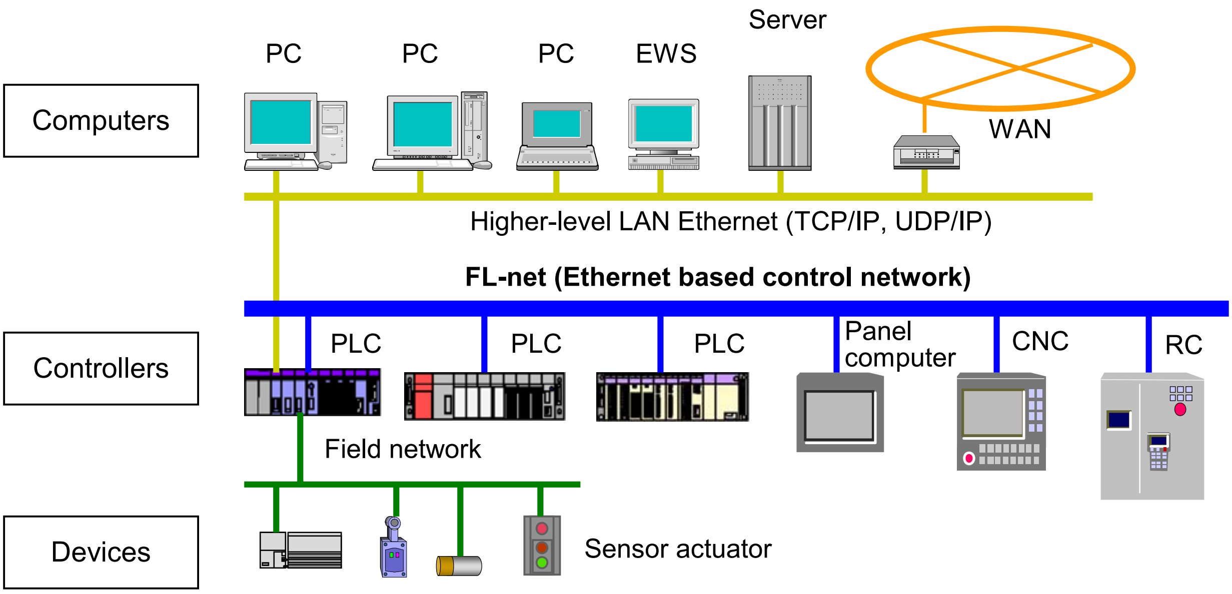


Figure 1-1 FA Control Network Configuration Example

1.2 FL-net Features

FL-net has the following features:

- An open system can be implemented.
- A multivendor environment can be established.
- Programmable controllers (PLCs), computer numerical control (CNC) devices, and various other factory automation controllers and personal computers of many different brands can be interconnected to exercise control and monitoring functions.

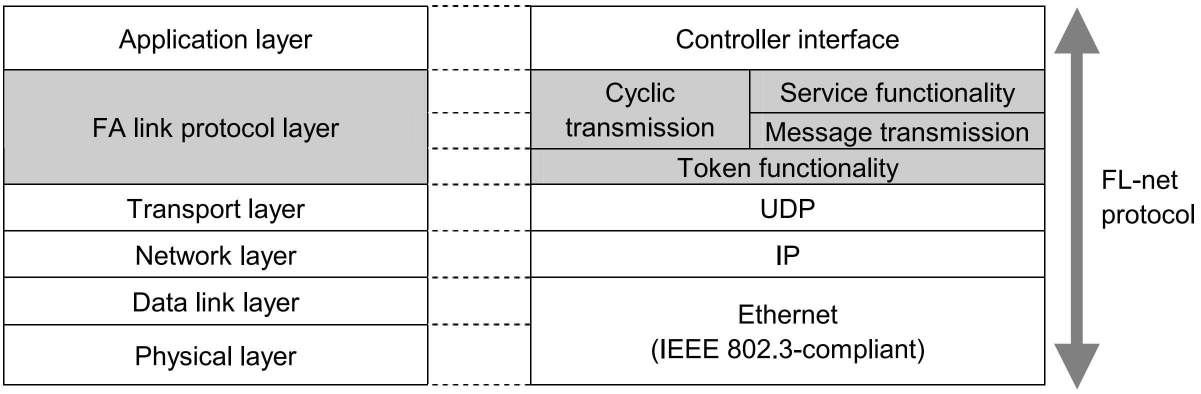


Figure 1-2 FL-net Protocol Basis Structure

<A widely accepted standard is complied with.>

FL-net is based on Ethernet, which is a standard for OA (office automation) equipment. Further, standard UDP/IP is used to achieve a high degree of communication efficiency. The use of Ethernet offers the following features:

- Low price
  Since common communication devices can be used as components, the price is decreased.
- Use of widespread network equipment
  Transceivers, hubs, cables, personal computer LAN cards, and various other widely accepted Ethernet devices can be used.
- Future speed increase
  In the future, you can expect that the speed of transmission will increase from 10 Mbps through 100 Mbps to 1 Gbps.
- Introduction of fiber-optic communication
  Optical repeaters and other widespread devices for Ethernet can be used to replace relevant portions with fiber-optic lines. The introduction of such fiber-optic lines will permit transmission over a distance of 500 m or longer, provide increased noise immunity, and protect against a lightning surge in outdoor wiring.

<The functions required for FA controllers are supported.>

Since the initial objective of FL-net is to meet user requests, it offers various features required for FA (factory automation).

- Large-scale network
  Up to 254 units of devices (nodes) can be connected.
- Selective use of two different communication features for various purposes
  Both the common memory feature and message communication feature are supported. The former permits all nodes to share the same data through cyclic communications. The latter allows necessary information to be exchanged only when it is needed.
- Large-capacity common memory
  The common memory has a capacity as large as 8k bits + 8k words.
- Fast response
  When FL-net-compliant modules are interconnected, the performance varies with the connected devices. However, when the S10mini FL.NET modules are interconnected, the response speed is as high as 64 ms/32 nodes (in 2k bit + 2k word mode).
- High reliability based on the use of a masterless system
  Since no master exists, individual nodes can participate and depart without affecting the communications maintained by the other nodes. As a result, any nodes can be freely turned ON/OFF and serviced.

2 GENERAL PRECAUTIONS

2.1 Safety Precautions

SAFETY PRECAUTIONS

Be sure to read this manual and all other attached documents carefully before installing, operating inspecting or conducting maintenance on this unit. Always use this unit properly. Be sure to carefully read the information about the device, the safety information and precautions before using this unit. Be sure that the person(s) responsible for maintenance receives and understands this manual completely.

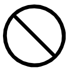
This manual divides the safety precautions into DANGERs and CAUTIONs.

DANGER : Failure to observe these warnings may result in death or serious injury.

CAUTION : Failure to observe these cautions may result in injury or property damage.

Failure to observe any **CAUTION** may lead to serious consequences.

All of these DANGERs and CAUTIONs provide very important precautions and should always be observed.
Additional safety symbols representing a prohibition or a requirement are as follows:

 : Prohibition. For example, “Do not disassemble” is represented by:

 : Requirement. For example, if a ground is required, the following will be shown:

2.2 General Usage Precautions

During the use of the FL.NET module, you should particularly observe the following precautions:

(1) Installation

CAUTION
● Use this product under the environmental conditions specified in the catalogs and manual. Utilizing this product in a hot, damp, or dusty atmosphere or in an atmosphere of corrosive gas, vibration or impact may lead to a malfunction, shock hazard or fire. ● Install this product according to the procedure outline in the manual. Imperfect installation may lead to a part drop, failure of malfunction. ● Do not put any wire chip or other foreign matter into this product. This may cause a malfunction, failure or fire.

(2) Wiring

REQUIREMENT

Be sure to ground this product with FG.
Failure to ground this product may lead to a malfunction or shock hazard.

CAUTION

- Connect this product to a power supply with the same ratings.
  Connecting this product to a power supply exceeding its voltage rating may lead to a fire.
- Wiring must be conducted by a qualified technician.
  Miswiring may lead to failure, shock hazard or fire.

CAUTION

When installing this module or making its wiring connections, be sure to comply with the instructions set forth in the manual. Also, be sure to furnish wiring personnel with the information about installation and wiring. If any improper installation or wiring procedure is performed, the module may malfunction due to decreased noise immunity.

(3) Handling precautions

DANGER

- Do not touch any terminal while this product is live, as this may lead to a shock hazard.
- If an emergency stop circuit, interlock circuit, or similar circuit is to be formulated, it must be positioned external to this module. If you do not observe this precaution, equipment damage or accident may occur when this module becomes defective.

CAUTION

- Make sure that everything is safe before changing programs, running or stopping this product while on the fly or producing forced output. Mishandling may lead to product breakdown or an accident.
- Turn on the product according to the correct power – on procedure. Mishandling may lead to product breakdown or an accident.
- While the power is ON, do not touch a 10BASE-5 connector. Failure to observe this precaution may cause the system to malfunction due to static electricity.

(4) Maintenance

PROHIBITION

Do not disassemble or remodel this product, as this may lead to a malfunction, failure or fire.

CAUTION

- Power off this product before attaching or detaching any module or unit as this may lead to a malfunction, failure of shock hazard.
- When replacing the fuse, use only the specified one. The use of a different fuse may cause a risk of fire and equipment malfunction.

2.3 Warranty and Servicing

WARRANTY AND SERVICING

Unless a special warranty contract has been arranged, the following warranty is applicable to this product.

1. Warranty period and scope
   Warranty period
   The warranty period for this product is for one year after the product has been delivered to the specified delivery site.

   Scope
   If a malfunction should occur during the above warranty period while using this product under normal product specification conditions as described in this manual, please deliver the malfunctioning part of the product to the dealer or Hitachi Engineering & Services Co., Ltd. The malfunctioning part will be replaced or repaired free of charge. If the malfunctioning is shipped, however, the shipment charge and packaging expenses must be paid for by the customer.

   This warranty is not applicable if any of the following are true.

   - The malfunction was caused by handling or use of the product in a manner not specified in the product specifications.
   - The malfunction was caused by a unit other than that which was delivered.
   - The malfunction was caused by modifications or repairs made by a vendor other than the vendor that delivered the unit.
   - The malfunction was caused by a relay or other consumable which has passed the end of its service life.
   - The malfunction was caused by a disaster, natural or otherwise, for which the vendor is not responsible.

   The warranty mentioned here means the warranty for the individual product that is delivered. Therefore, we cannot be held responsible for any losses or lost profits that result from the operation of this product or from malfunctions of this product. This warranty is valid only in Japan and is not transferable.

2. Range of services
   The price of the delivered product does not include on-site servicing fees by engineers.
   Extra fees will be charged for the following:

   - Instruction for installation and adjustments, and witnessing trial operations.
   - Inspections, maintenance and adjustments.
   - Technical instruction, technical training and training schools.
   - Examinations and repairs after the warranty period is concluded.
   - Even if the warranty is valid, examination of malfunctions that are caused by reasons outside the above warranty scope.

3 FL.NET MODULE

3.1 System Configuration

The FL.NET is Hitachi's module that is compliant with the FL-net protocol.
Figure 3-1 shows a typical system configuration for the use of FL-net communications.

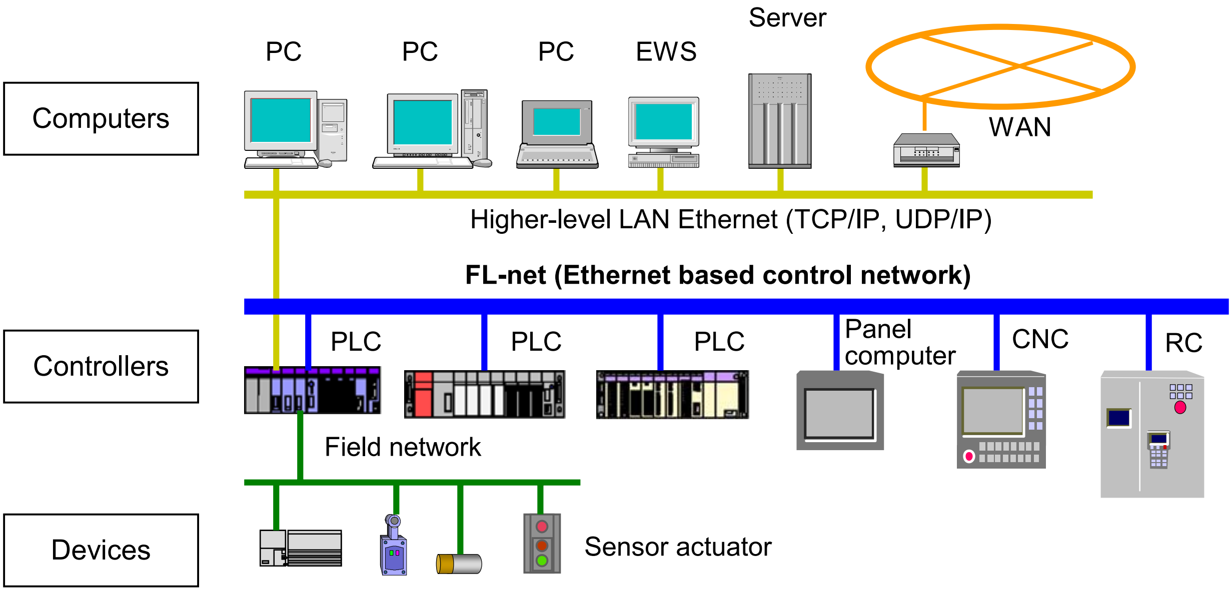


Figure 3-1 FA Control Network Configuration Example

Table 3-1 General Specifications

Item	Specifications
Operating ambient temperature	0 to 55°C
Storage ambient temperature	-20 to 75°C
Operating atmosphere	Dust: 0.1 mg/m^3 maximum; free from corrosive gas
Operating ambient humidity	30 to 90% RH (non-condensing)
Storage ambient humidity	10 to 90% RH (non-condensing)
Vibration resistance	JIS C 0040-compliant
Shock resistance	JIS C 0041-compliant

3.1.1 Functionality and performance specifications

Table 3-2 Functionality and Performance Specifications

Item	Specifications	Remarks
Type	LQE000	
Maximum number of mountable modules	2 modules per CPU unit (mounted contiguously beginning with the left end)	
Transmission speed	10 Mbps	
Electrical interface	IEEE 802.3-compliant (CSMA/CD-compliant)	
Transmission protocol	UDP/IP, FA link protocol	
Number of connectable units	Up to 254 units per network	
Connector	AUI connector (10BASE-5) RJ45 connector (10BASE-T)	
External AUI supply power (terminal block)	12 VDC, 500 mA or less (M3 screw terminal block)	Required for 10BASE-5 use only
Module outside dimensions	34 (W) × 130 (H) × 100.2 (D) mm	
Weight	240 g	
Transfer word count	Cyclic: 8.5k words maximum. (node) Message: 1,024 bytes maximum. (node)	
Transfer distance	10BASE-5: 2.5 km maximum. (inter-node) 10BASE-T: 1.5 km maximum. (inter-node)	
Cable length	Transceiver cable: 50 m maximum. (10BASE-5) Twisted-pair cable: 100 m maximum. (10BASE-T)	

3.1.2 Support tool specifications

For the use of the FL.NET module, you must set the node numbers, common memory, and various other items with the setup tool named “FL.NET For Windows®.”
For the detailed specifications for the setup tool, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139).”

3.1.3 Link data specifications

(1) Common memory area
The FL.NET module allows you to set up a common memory area for each node.
For the common memory area setup procedure, see “6.4.1 Link parameter setup procedure.”
For the procedure for reserving a common memory area in the CPU’s memory, see “6.4.2 CPU memory allocation procedure.”

Table 3-3 Registers Available for Common Memory Area Allocation in CPU Memory

Available register/address	Remarks
XW000 to XWFF0	External input
YW000 to YWFF0	External output
JW000 to JWFF0	Transfer register
QW000 to QWFF0	Receive register
GW000 to GWFF0	Global link register
RW000 to RWFF0	Internal register
EW400 to EWFF0	Event register
MW000 to MWFF0	Internal register
DW000 to DWFFF	Function data register
FW000 to FWBFF	Function work register
/100000 to /4FFFFE	Extension memory

(2) Virtual address space and physical memory

Table 3-4 Virtual Address Space and Physical Memory

<table>
<tr><th>Item</th><th colspan="2">Description</th></tr>
<tr><td>Area name</td><td colspan="2">S10mini memory address (0x000000 to 0xFFFFFF)</td></tr>
<tr><td>Access unit</td><td colspan="2">WORD</td></tr>
<tr><td>Area size</td><td colspan="2">16777216 bytes</td></tr>
<tr><td>Access attribute</td><td colspan="2">Read/Write
Conditions: A write operation cannot be performed on some ROM areas and OS areas.</td></tr>
<tr><td rowspan="5">Correspondence to virtual address word block (*)</td><td>Vendor-unique notation</td><td>Virtual address</td></tr>
<tr><td>0x000000, 0x000001</td><td>0x00000000</td></tr>
<tr><td>0x000002, 0x000003</td><td>0x00000001</td></tr>
<tr><td>~ ~</td><td>~</td></tr>
<tr><td>0xFFFFFE, 0xFFFFFF</td><td>0x007FFFFF</td></tr>
<tr><td>Data arrangement</td><td colspan="2">A 2-byte area corresponds to one word of a word block.
MSB [Address n] [Address n + 1] LSB</td></tr>
<tr><td>Other access method</td><td colspan="2">Nothing in particular</td></tr>
</table>

The value “0x000012AB” is a hexadecimal number (000012AB hex).
(*) To support the correspondence to virtual addresses, a byte block is not supported.

Table 3-5 shows the relationships among virtual address spaces, S10mini memory addresses, and registers.

Table 3-5 Virtual Address Spaces and Symbol Names

Register name	Symbol name	S10mini memory address	Virtual address space
External input	X000 to XFFF	0x0A0000 to 0x0A1FFE	0x00050000 to 0x00050FFF
External output	Y000 to YFFF	0x0A4000 to 0x0A5FFE	0x00052000 to 0x00052FFF
Internal register	R000 to RFFF	0x0AC000 to 0x0ADFFE	0x00056000 to 0x00056FFF
Global register	G000 to GFFF	0x0A8000 to 0x0A9FFE	0x00054000 to 0x00054FFF
On-delay timer	T000 to T1FF	0x0B3000 to 0x0B33FE	0x00059800 to 0x000599FF
One-shot timer	U000 to U0FF	0x0B5000 to 0x0B51FE	0x0005A800 to 0x0005A8FF
Counter	C000 to C0FF	0x0B7000 to 0x0B71FE	0x0005B800 to 0x0005B8FF
Keep	K000 to KFFF	0x0B0000 to 0x0B1FFE	0x00058000 to 0x00058FFF
System register	S000 to SBFF	0x0BE800 to 0x0BFFFE	0x0005F400 to 0x0005FFFF
Z-register	Z000 to Z3FF	0x0BE000 to 0x0BE7FE	0x0005F000 to 0x0005F3FF
Internal register	M000 to MFFF	0x0AE000 to 0x0AFFFE	0x00057000 to 0x00057FFF
Transfer register	J000 to JFFF	0x0A2000 to 0x0A3FFE	0x00051000 to 0x00051FFF
Receive register	Q000 to QFFF	0x0A6000 to 0x0A7FFE	0x00053000 to 0x00053FFF
External input (word)	XW000 to XWFF0	0x0E0000 to 0x0E01FE	0x00070000 to 0x000700FF
External output (word)	YW000 to YWFF0	0x0E0400 to 0x0E05FE	0x00070200 to 0x000702FF
Internal register (word)	RW000 to RWFF0	0x0E0C00 to 0x0E0DFE	0x00070600 to 0x000706FF
Global register (word)	GW000 to GWFF0	0x0E0800 to 0x0E09FE	0x00070400 to 0x000704FF
On-delay timer (word)	TW000 to TW1F0	0x0E1300 to 0x0E133E	0x00070980 to 0x0007099F
One-shot timer (word)	UW000 to UW0F0	0x0E1500 to 0x0E151E	0x00070A80 to 0x00070A8F
Counter (word)	CW000 to CW0F0	0x0E1700 to 0x0E171E	0x00070B80 to 0x00070B8F
Keep (word)	KW000 to KWFF0	0x0E1000 to 0x0E11FE	0x00070800 to 0x000708FF
System register (word)	SW000 to SWBF0	0x0E1E80 to 0x0E1FFE	0x00070F40 to 0x00070FFF
Z-register (word)	ZW000 to ZW3F0	0x0E1E00 to 0x0E1E7E	0x00070F00 to 0x00070F3F
Internal register (word)	MW000 to MWFF0	0x0E0E00 to 0x0E0FFE	0x00070700 to 0x000707FF
Transfer register (word)	JW000 to JWFF0	0x0E0200 to 0x0E03FE	0x00070100 to 0x000701FF
Receive register (word)	QW000 to QWFF0	0x0E0600 to 0x0E07FE	0x00070300 to 0x000703FF
Function data register	DW000 to DWFFF	0x061000 to 0x062FFE	0x00030800 to 0x000317FF
Function work register	FW000 to FWBFF	0x0E2000 to 0x0E37FE	0x00071000 to 0x00071BFF
Extension memory address	–	0x100000 to 0x4FFFFE	0x00080000 to 0x0027FFFF

(3) Error memory area
The FL.NET module offers tables for recording error information.

Table 3-6 Error Memory Area for Link Data

Item	Specifications
Error freeze information table	Stores information about abnormal operations performed in the module.
Error message data table	Stores the error messages generated by the other nodes.

For detailed information about the tables, see “8 TROUBLESHOOTING.”

(4) Status memory area
The status memory area for the FL.NET module can be referenced with the setup tool named “FL.NET For Windows®” (for details, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)”).

3.1.4 Link parameter setup area

The link parameters for the FL.NET module can be referenced with the setup tool named “FL.NET For Windows®” (for details, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)”).

3.1.5 Profile system parameter setup area

The profile system parameters for the FL.NET module can be referenced with the setup tool named “FL.NET For Windows®” (for details, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)”).
For the FL.NET module, the profile system parameters for the vendor name, manufacturer model, and protocol version are fixed data. The FL.NET module automatically changes the FA link status and self-node status in accordance with the communication status and other conditions. The user can merely perform node name setting with the setup tool.

Table 3-7 Profile System Parameter Setup Area (Entry Examples)

Item	Length	Data	Description
Vendor name	10 bytes	“HITACHI”	Vendor name.
Maker form	10 bytes	“S10mini”	Maker form/device name.
Node name (equipment name)	10 bytes	————	User-defined node name.
Protocol version	1 byte	————	Fixed at 0x80.
FA link status	1 byte	————	Entering/leaving, etc.
Self-node status	1 byte	————	Node number duplication detection, etc.

3.2 FL.NET Module Component Names and Functions

3.2.1 External views

Figure 3-2 presents external views of the FL.NET module.

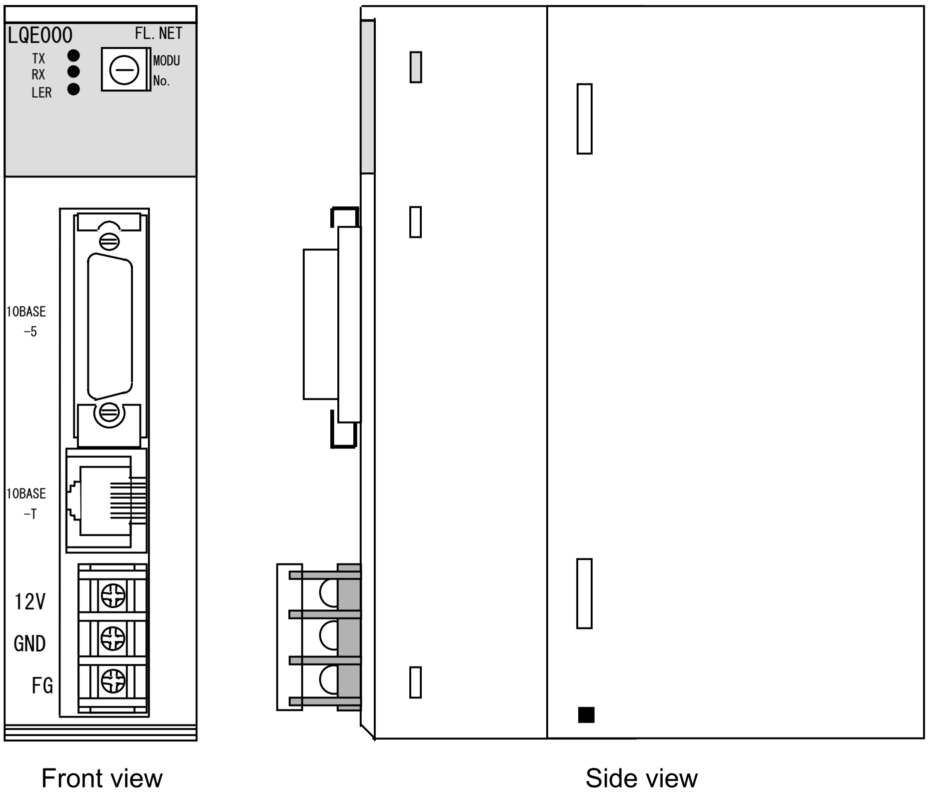


Figure 3-2 FL.NET Module External Views

3.2.2 Component names and functions

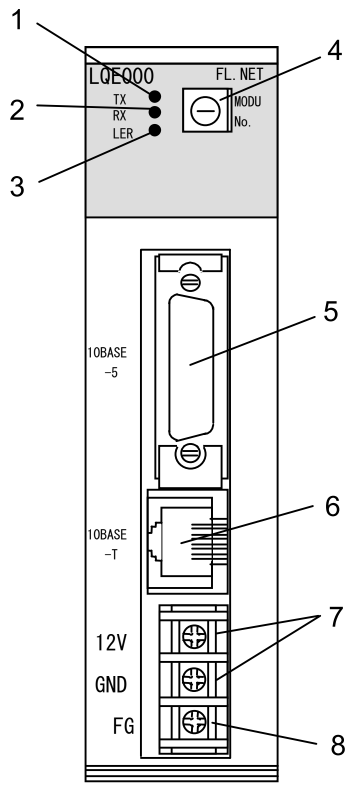


Figure 3-3 FL.NET Module Component Names

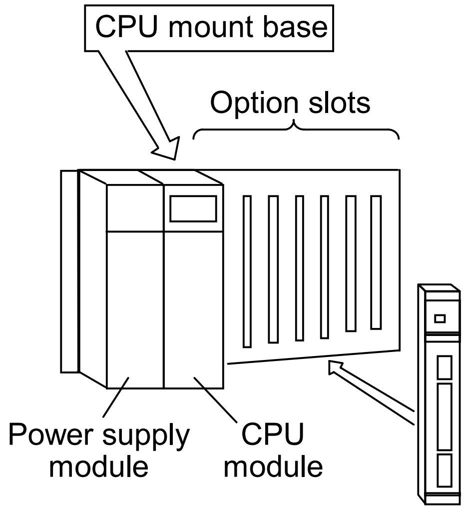
Table 3-8 FL.NET Module Component Names and Functions

No.	Name	Function
1	TX LED (Green)	This LED indicates while data is being transferred.
2	RX LED (Green)	This LED indicates while data flows along a transmission path (a carrier is detected).
3	LER LED (Red)	This LED indicates when a fatal fault occurs, unconnected FA link, or FA link communication failure.
4	Module number selector switch	This switch performs main module/submodule setup or selects a communication port type.
5	10BASE-5 interface connector	Connect this connector to the S10mini or other controller.
6	10BASE-T interface connector	Connect this connector to the S10mini or other controller.
7	Power input terminals	Connect these terminals to the power supply (12 VDC) for the transceiver that is to be connected to the 10BASE-5 interface connector.
8	Frame ground	Connect this ground to the shielded wire in the transceiver cable.

4 FL.NET MODULE INSTALLATION

4.1 Inserting the Module into Slot

Insert the FL-net module into an option slot in the CPU mount base.



<Hardware configuration>
CPU mount base: HSC-1020, HSC-1040, HSC-1080
PS slot: The power supply module (LQV000, LQV020, or LQV100) is inserted.
CPU slot: The CPU module (LQP000, LQP010, LQP011, or LQP120) is inserted.

LQP000: Model S
LQP010: Model H
LQP011: Model F
LQP120: Model D

Slots 0 through 7: A CPU option module or I/O is inserted.

Figure 4-1 Module Installation (Example)

- Install the FL.NET module immediately next to the CPU module. Ensure that no I/O module is mounted between them.
- When installing only one FL.NET module, be sure that its setting made is “main module.”

4.2 Module Mounting Method

Mount the module on the CPU mount base and screw it down. Ensure that the module is squarely mounted on the mount base (If the module is obliquely mounted as shown in bad examples, connector breakage may occur.)

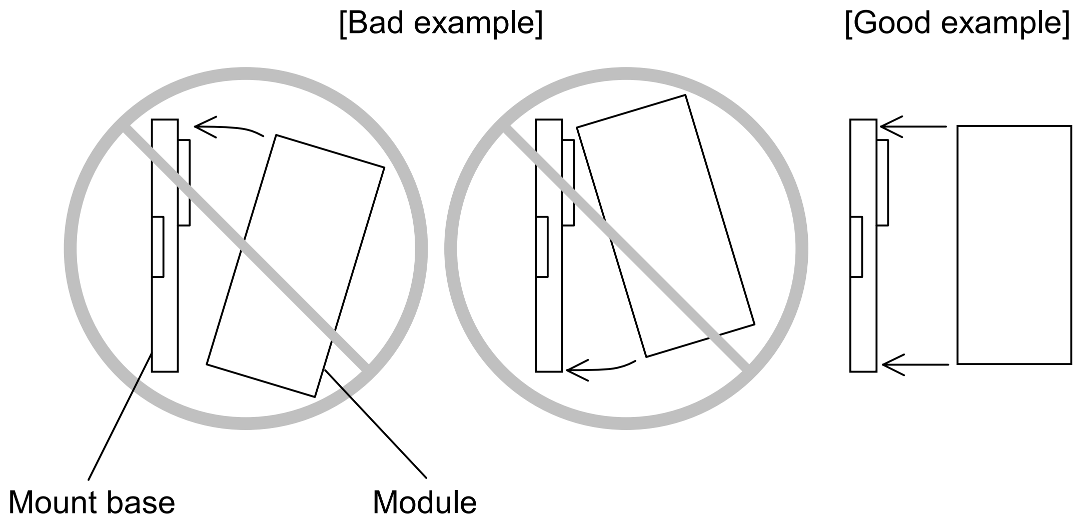


Figure 4-2 Module Installation Method (Example)

CAUTION
If the mount base is positioned overhead due to the employed enclosure structure, use a stepladder or the like and mount the module squarely. If you mount the module obliquely, the connector may become damaged.

4.3 Setting the Module Switch

When using the FL-net module, it is necessary to perform main module/submodule setup and communication port setup.
Setup can be completed by pointing the arrow mark on the module number selector switch toward the desired module number.

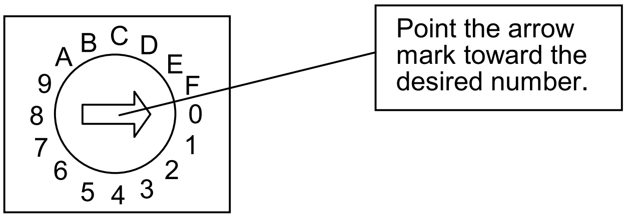


Figure 4-3 Setting the Module Switch

Table 4-1 shows the meanings of the module number selector switch settings.

Table 4-1 Details of Module Number Selector Switch

MODU No.		Meaning of setting
Main module	Submodule	
0	1	10BASE-5 communication.
2	3	10BASE-T communication.
4	5	These module number settings are invalid. When these settings are selected, the system indicates a module switch setting error and does not perform communication or other operations.
6	7	
8	9	
A	B	
C	D	
E	F	

When setting a module number, ensure that the power switch is OFF. Setting a module number with the power ON may cause the system to malfunction.

5 FL.NET MODULE WIRING PROCEDURES

5.1 Connecting the Communication Cable

(1) Connecting the 10BASE-5 transceiver cable

When the module number selector switch is set to 0 or 1, the module uses the 10BASE-5 interface connector to establish communication with the other modules.

Ensure that the 10BASE-5 cable is connected as shown in Figure 5-1.

For communications with 10BASE-5, it is also necessary to supply power from a 12-VDC external power source. For external power source connection, see "5.3 Power Supply Wiring."

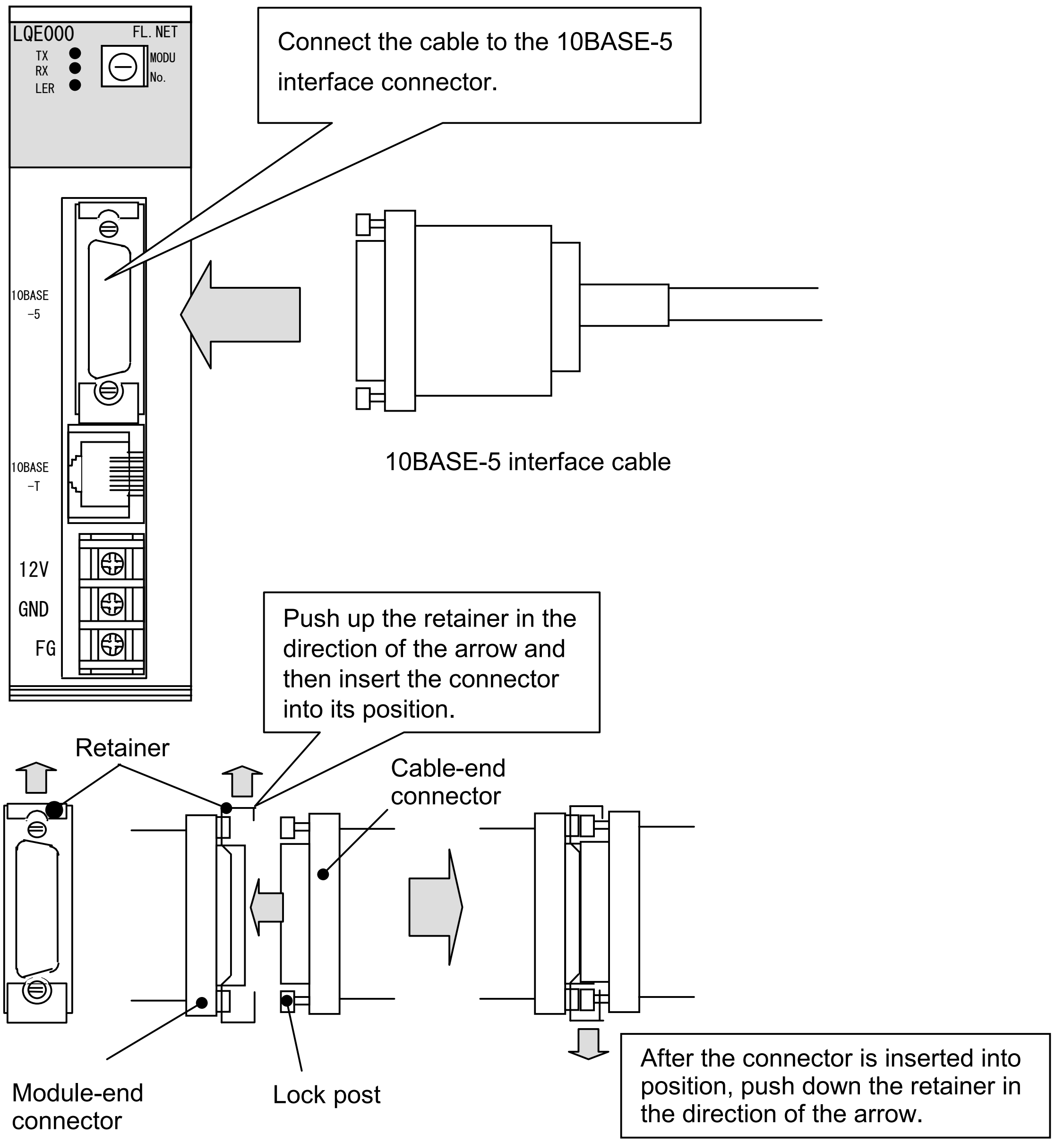


Figure 5-1 Connecting the 10BASE-5 Transceiver Cable to the Module

- When the 10BASE-5 connector is connected, make sure that the lock post is locked by the retainer.　If the lock post is not properly locked, a malfunction may be caused by a poor contact or open circuit.
- Do not touch the 10BASE-5 connector while the power is ON.　Failure to observe this precaution may cause the system to malfunction due to static electricity.

(2) Connecting the 10BASE-T cable

When the module number selector switch is set to 2 or 3, the module uses the 10BASE-T interface connector to establish communication with the other modules.

Ensure that the 10BASE-T cable is connected as shown in Figure 5-2.

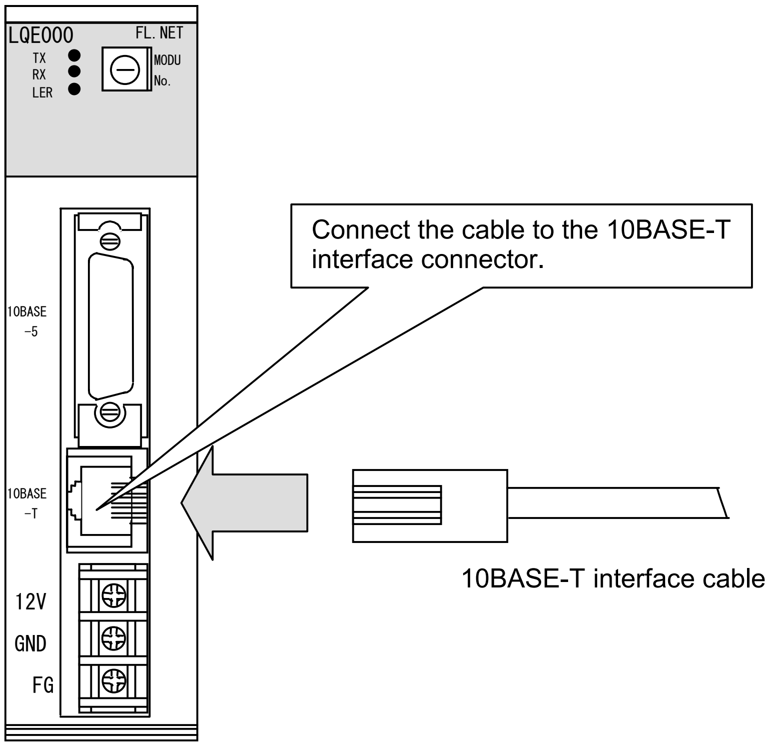


Figure 5-2　Connecting the 10BASE-T Cable to the Module

5.2 Applicable Communication Cables

(1) 10BASE-5 transceiver cable

Table 5-1 Communication Cable (10BASE-5 Transceiver Cable) Applicable to the Module

Product name	Model number	Manufacturer
Transceiver cable	HBN-TC-100	Hitachi Cable, Ltd

(2) 10BASE-T cable

Table 5-2 Communication Cable (10BASE-T Twisted-pair Cable) Applicable to the Module

Product name	Model number	Manufacturer
Twisted-pair cable	HUTP-CAT5 4P	Hitachi Cable, Ltd

5.3 Power Supply Wiring

For the use of 10BASE-5, it is necessary to supply power from an external power source as indicated in Figure 5-3.

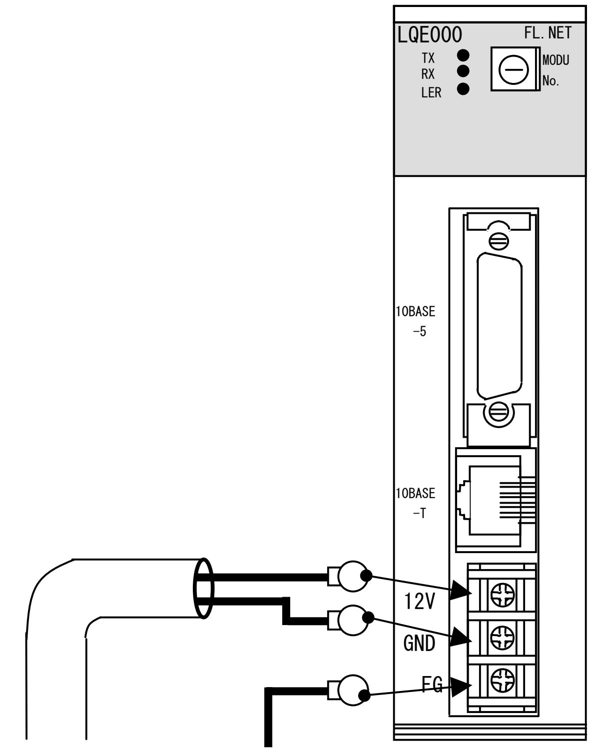


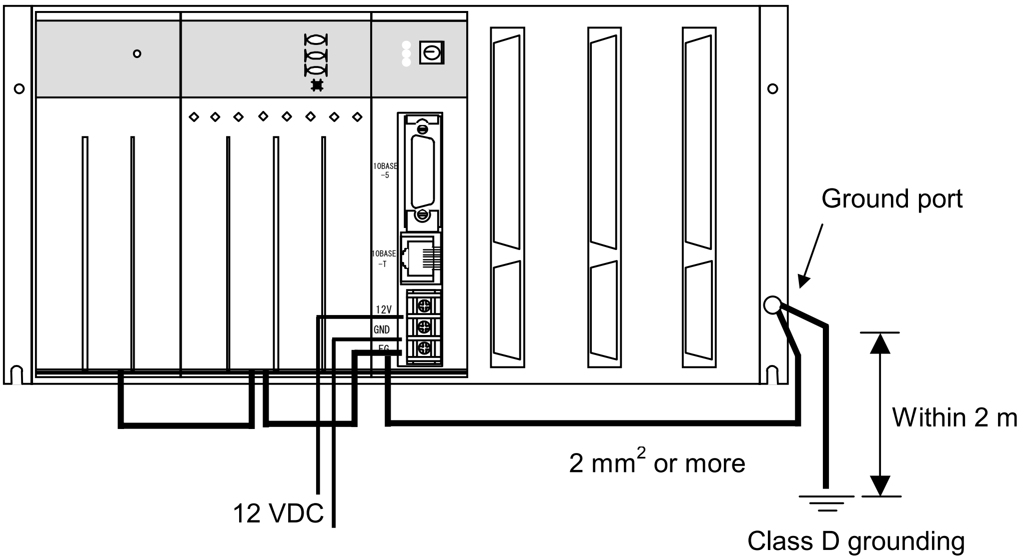
Figure 5-3 Module Power Supply Wiring

> It is recommended that you use the following 12-VDC external power source.
> Power source: Model HK-25A-12 (manufacturer: Densei-Lambda K.K.)

5.4 Ground Wiring

When using 10BASE-5, make the ground wiring connection as shown in Figure 5-4.

• Typical ground wiring for 10BASE-5



• Ground wiring for 10BASE-T (Do not make a ground wiring connection to the FG on the FL.NET module.)

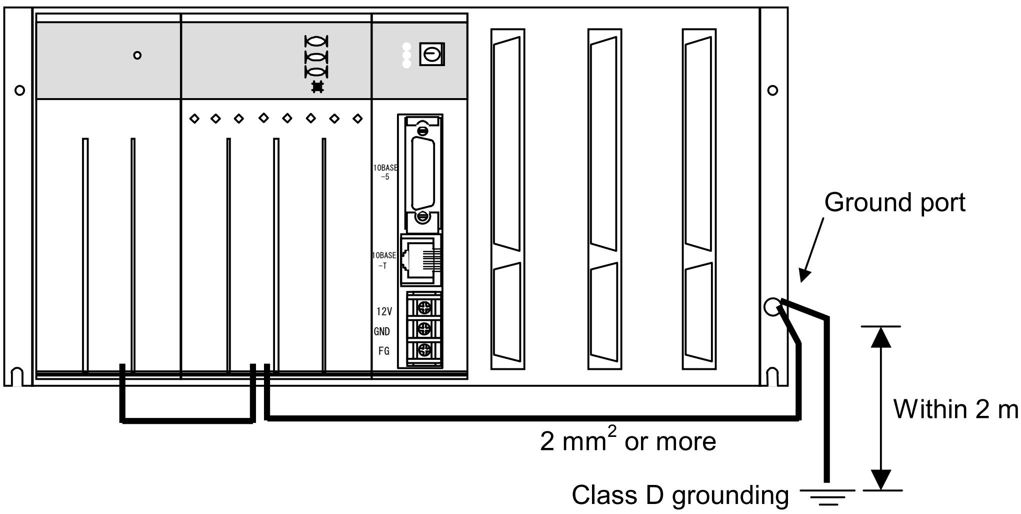


Figure 5-4 Unit Ground Wiring

Class D grounding is defined in the Technical Standard for Electrical Facilities of Japan. This standard states that the grounding resistance must be 100 ohms or less for equipment operating on 300 VAC or less, and 500 ohms or less for devices that shut down automatically within 0.5 seconds when shorting occurs in low tension lines.

REQUIREMENT

- Ground the FG (frame ground) terminal as follows: Connect the FG terminal on each module provided with external terminals to the grounding terminal on the mount base. Make sure that the line used for grounding is at most 2 m long.
  Perform Class D grounding for the grounding terminal on the mount base.
- Use ground lines whose size is 2 mm^2 or more.
- Do not touch the 10BASE-5 connector during power-on. Otherwise, the system may malfunction due to static electricity, etc.

6 USER GUIDE

6.1 Ethernet

6.1.1 10BASE-5 system

In the basic configuration, nodes are connected to a coaxial cable having a maximum length of 500 m as shown in Figure 6-1. Nodes are connected to the coaxial cable via transceiver cables (AUI cables) and transceivers. Two types of transceivers are available: single-port and multiport transceivers. Single-port transceivers permit the connection of only one transceiver cable (AUI cable). Multiport transceivers permit the connection of two or more transceiver cables.
This basic configuration is referred to as a segment, which consists of up to 100 nodes.

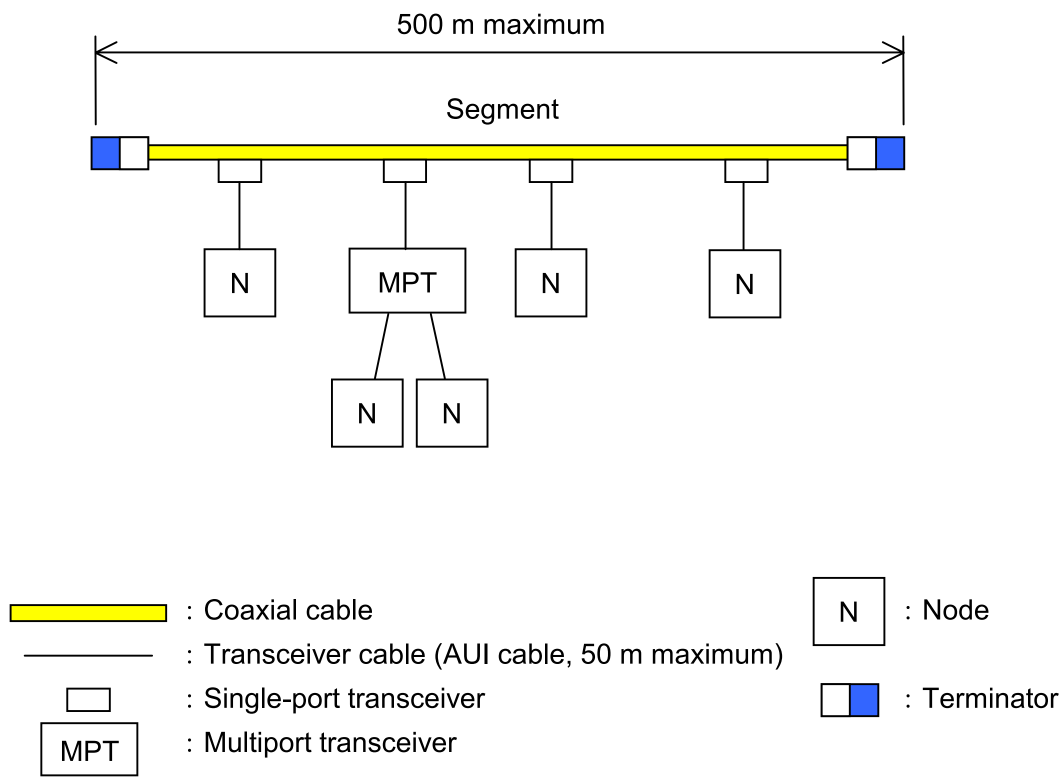


Figure 6-1 10BASE-5 System Basic Connection Method
(No Repeater Used; Maximum Distance: 500 m)

If the inter-node distance is longer than 500 m, you can increase the number of segments by making branches with repeaters as shown in Figure 6-2. In an example system shown in Figure 6-3, the maximum inter-node distance is not longer than 1500 m. Ensure that the number of repeaters does not exceed 2 in any path between arbitrarily selected two nodes.

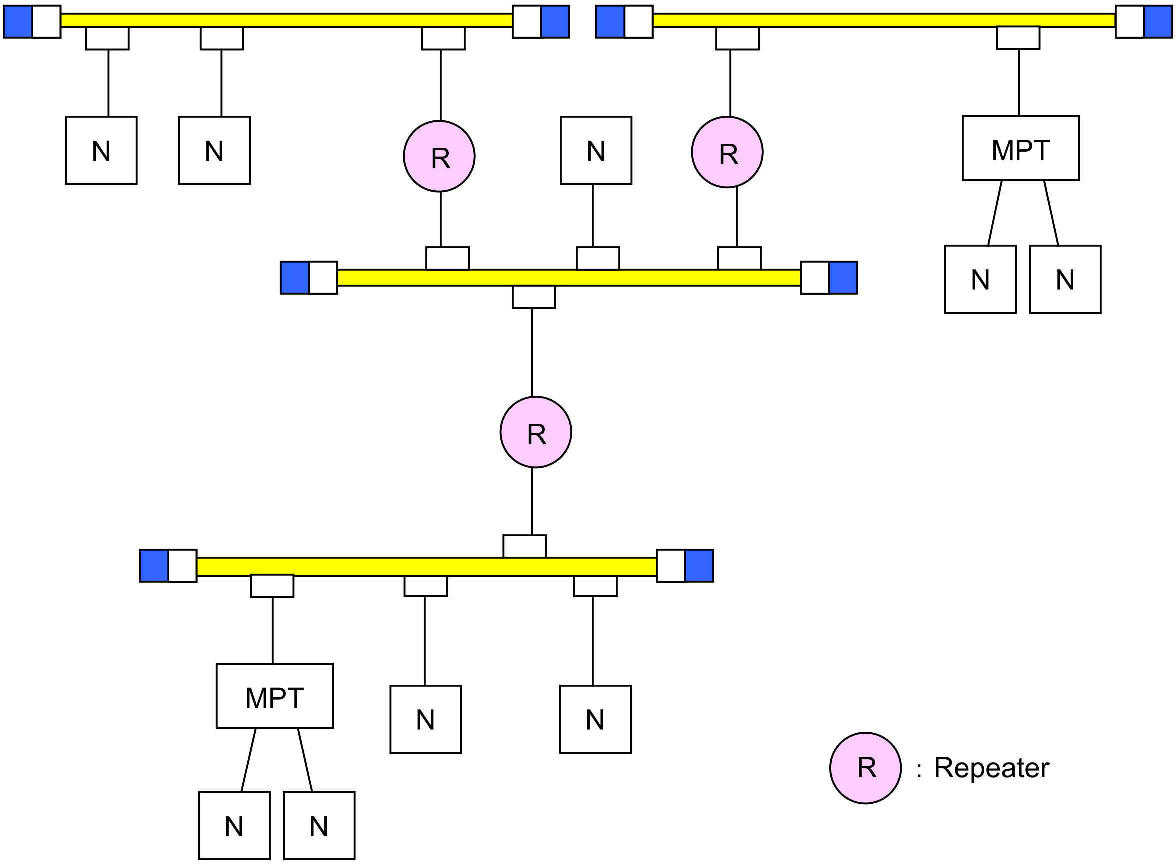


Figure 6-2 10BASE-5 System Basic Connection Method
(Repeaters Used; Maximum Distance: 1500 m)

CAUTION

- Connect each repeater to the coaxial cable via a transceiver cable and transceiver.
- A repeater can be mounted on any transceiver within the same coaxial segment.
- Transceivers must be installed at spacing intervals of an integer multiple of 2.5 m.

In the example shown in Figure 6-3, the maximum inter-node distance is 2500 m. A link cable (the maximum length is 500 m when the cable is coaxial) having repeaters on both ends is used to increase the transmission distance. A segment formed in this manner is called a link segment. No node is to be connected to a link segment. However, the portion enclosed by dotted lines, including the repeaters on both ends, can be counted as one repeater unit. Therefore, you can raise the limit on the total number of repeaters between nodes.

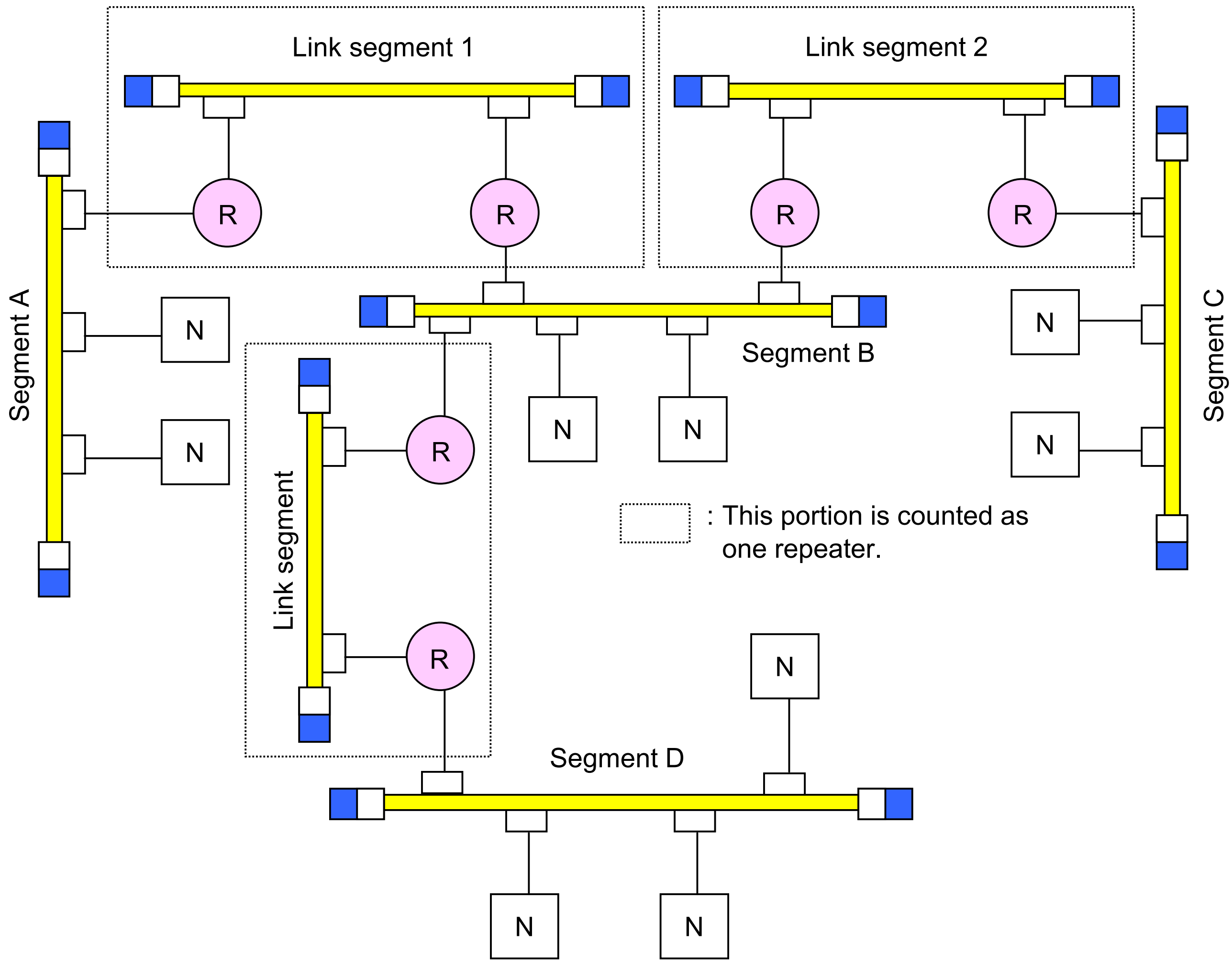


Figure 6-3 10BASE-5 System Basic Connection Method
(Repeaters Used; Maximum Distance: 2500 m)

- The maximum link segment length is 500 m.
- Do not connect any node to a link segment.
- When a link segment is used, the portion enclosed by dotted lines in the above figure, including the repeaters on both ends, is counted as one repeater.
- Ensure that the number of repeaters between nodes does not exceed 2.
- Only one segment can be connected to two or more repeaters.

Table 6-1 shows the system configuration parameters.

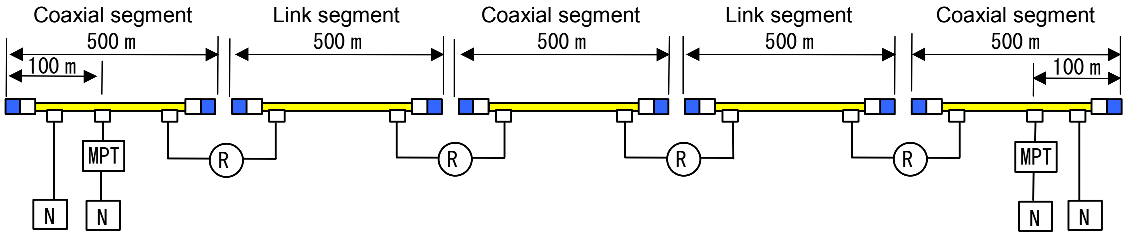
Table 6-1 General Specifications for Ethernet System Configuration

Item	Specification
Maximum segment length	500 m
Maximum number of transceivers in segment	100 units
Maximum distance between stations	2500 m or less (excluding transceiver cable)
Maximum number of system nodes	254 units
Maximum length of transceiver cable (AUI cable)	50 m
Cable length between transceiver and repeater	2 m or less (recommended value)
Maximum number of repeaters in an inter-node path	2 units (however, a link segment is counted as one repeater unit, including the repeaters on both ends).

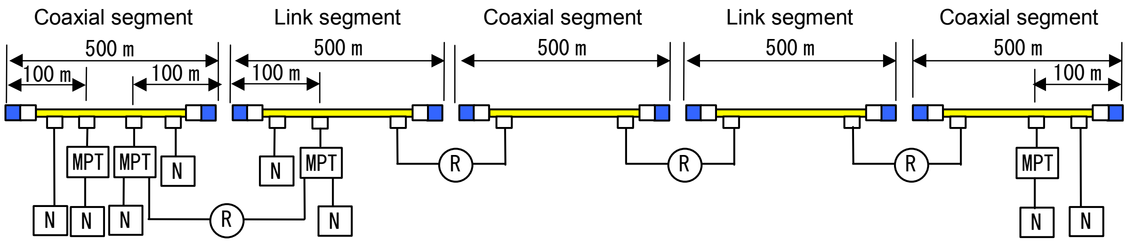
- Limitations on multiport transceiver installation location
- When a multiport transceiver is installed in the farthest coaxial cable segment in a system whose maximum coaxial cable length is 2500 m (5 segments), the multiport transceiver installation location is limited due to an increase in the data delay time. The maximum permissible coaxial cable length for an inter-node path decreases by 100 m whenever the number of multiport transceivers contained in the path increases by one. Therefore, the coaxial cable length L[m] between nodes is limited as indicated below:

  $L[m] \leq 2{,}500[m] - 100 \times N[m]$

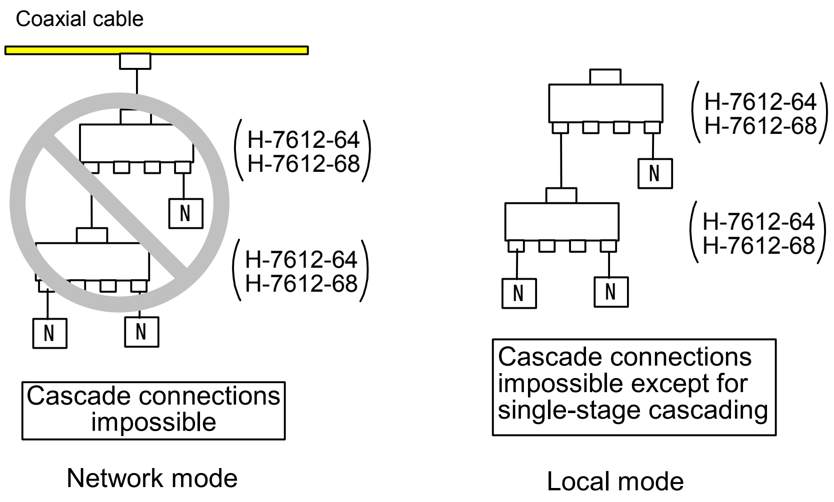
  N: Number of multiport transceivers contained in the path
- In a system consisting of a 2500 m coaxial cable, the multiport transceiver must be positioned at least 100 m inside of the farthest coaxial cable terminator (so as to reduce the inter-node distance).



- In situations where inter-segment repeaters are connected with multiport transceivers, any additionally installed multiport transceiver must be positioned so as to reduce the distance between the farthest nodes by 100 m.



- When the H-7612-64/68 multiport transceiver is used in network mode, cascade connections cannot be made due to the limitations on transmission characteristics.



6.1.2 10BASE-T system

As shown in Figure 6-4, you can connect two or more nodes to a hub that is connected to a transceiver via a transceiver cable (AUI cable).
For connecting nodes to a hub, use a twisted-pair cable (10BASE-T).

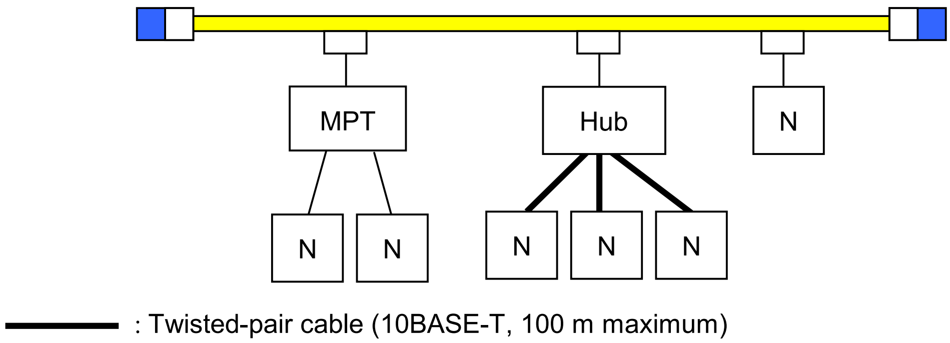


Figure 6-4 10BASE-T System Basic Connection Method 1

When the inter-node distance is short, you can connect nodes to a hub via twisted-pair cables as shown in Figure 6-5. These node connections can be made without a coaxial cable or transceiver.

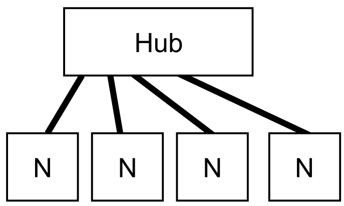
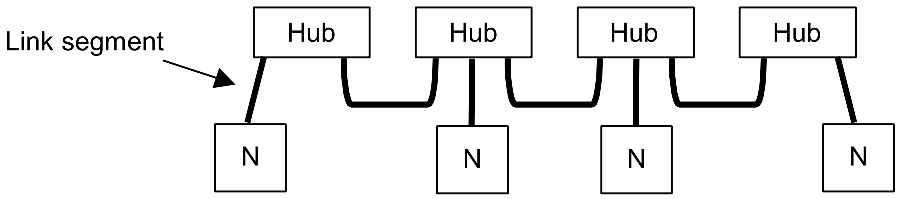


Figure 6-5 10BASE-T System Basic Connection Method 2

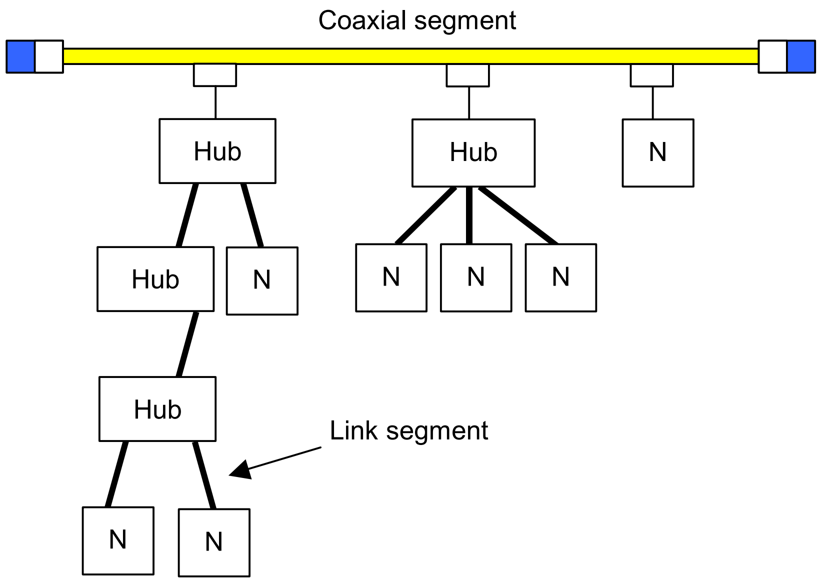
Limitations on hub use

- Hub cascading

  When using hubs (multiport repeaters) in cascade, ensure that no inter-node path contains more than four hubs or more than five link segments.



When using hubs that are connected to a coaxial, ensure that no inter-node path contains more than four hubs or more than five segments (or three coaxial segments).



<IP address>

UDP/IP uses a 32-bit logical address called an IP address. The IP address consists of a network address and host address. Class C is generally used in the field of FA (factory automation).

Class C	1	1	0	X	Network address (20-bit)	Host address (8-bit)

Figure 6-6 Ethernet IP Address Classification

The IP address consists of 8-bit strings of decimal numbers joined by periods (.). In Class C, it is indicated as shown in the following example:

11000000	00000001	00000000	00000011
192 .	001 .	000 .	003

Network address — Host number

For the FL-net, the default value is 192.168.250.N, where N is a node number between 1 and 254.

Figure 6-7 Ethernet Class C IP Address Example

6.2 FL-net

6.2.1 FL-net overview

(1) FL-net concept

The FL-net is an Ethernet-based factory-automation control network. It has a cyclic transmission function and message transmission function.

The basic ideas incorporated in the FL-net are as follows:

- Ethernet is used as the media (physical level, data link) of communication between FA (factory automation) controllers.
- UDP/IP prevalent throughout Ethernet is used to implement a basic means of data transmission.
- Transmission over a predetermined period of time is assured by managing/controlling communication media access of networked nodes (for collision avoidance) while making use of the above-mentioned basic means of data transmission.

The FL-net is designed for use in an FA (factory automation) control network that provides data exchanges between programmable controllers (PLCs), robot controllers (RCs), computer numerical control (CNC) devices, and various other production system controllers and control personal computers.

Figure 6-8 depicts the role of the FL-net.

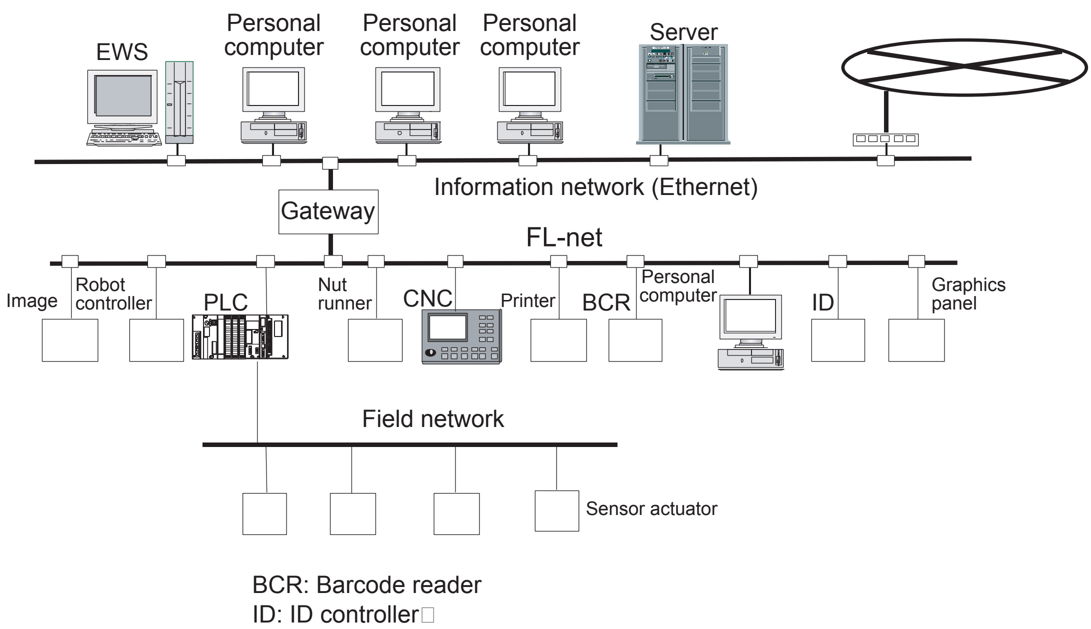


Figure 6-8 FL-net Concept

(2) FL-net protocol

The FL-net consists of six protocol layers as shown in Figure 6-9.

The transport layer and network layer use UDP/IP, whereas the data link layer and physical layer use Ethernet.

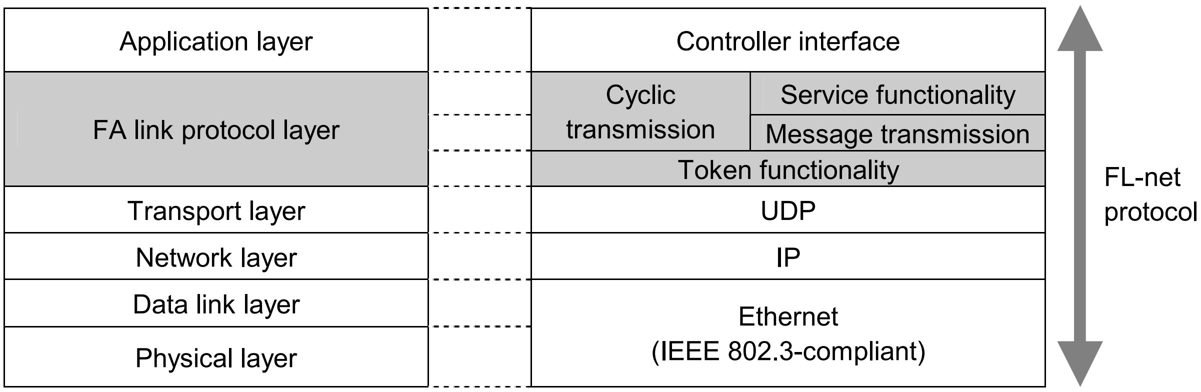


Figure 6-9 FA Link Protocol

(3) FL-net transmission system features

The features of the FL-net FA link protocol layer are outlined below:

- Masterless, token-based transmission management is exercised to avoid collisions.
- A token can be circulated at fixed intervals to define the refresh cycle time.
- A predetermined token is transmitted together with cyclic data.
- At startup, a token is transmitted from a node having the lowest node number.
- If a token is not transmitted for a predetermined period of time, the next node transmits a node.
- Thanks to the use of a masterless, token-based system, network inoperativeness can be avoided even when some nodes become faulty.
- Operating mode (RUN/STOP), hardware error (ALARM), and other information management tables are available for referencing the operating status of the other nodes.

(4) FL-net IP address

The IP address is an address that is used in IP(Internet Protocol)-based transmission to indicate a specific node (station). It is therefore necessary to set and manage IP addresses so as to avoid address duplication.

For each FL-net node, an IP address needs to be set. It is recommended that you use Class C IP addresses in accordance with the FL-net protocol.

The IP address default value for the FL-net is 192.168.250.***, where the *** portion is a node number. (The default value is recommended by the FL-net protocol.)

Network address	Host number (node number)
192.168.250.	n (n: 1 to 254)

Figure 6-10 FL-net IP Address

6.2.2 Connection capacity and node numbers

Up to 254 units can be connected. The available node numbers are from 1 to 254.

① Node number: 1 to 249 — For normal FL-net devices.

② Node number: 250 to 254 — For FL-net maintenance.

③ Node number: 255 — Used within the FL-net (used for global address broadcasting) and not available to the user.

④ Node number: 0 — Used within the FL-net and not available to the user.

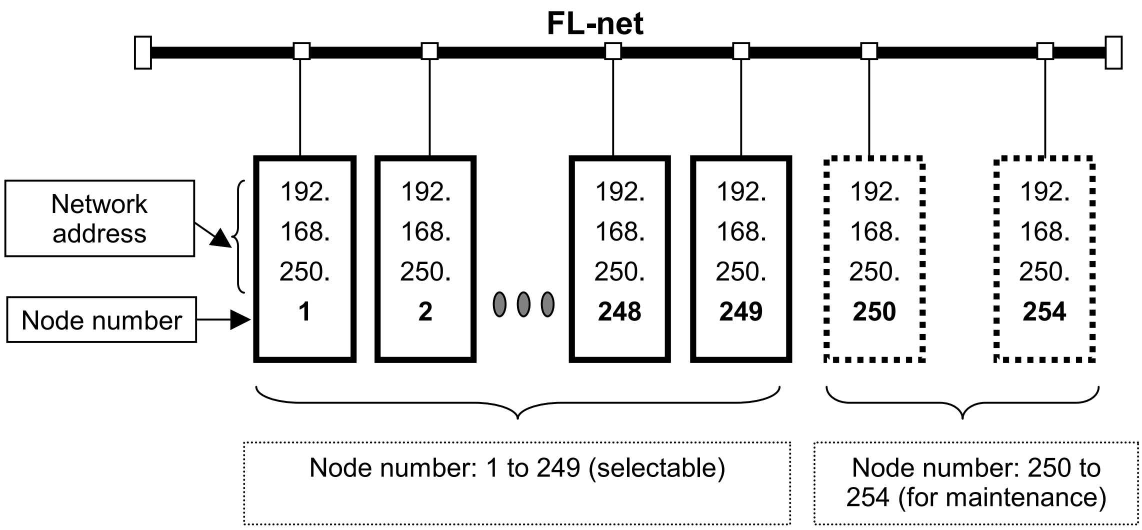


Figure 6-11 FL-net Connection Capacity and Node Numbers

6.2.3 Supported data communications

Data communications supported by the FL-net are cyclic transmission and message transmission.

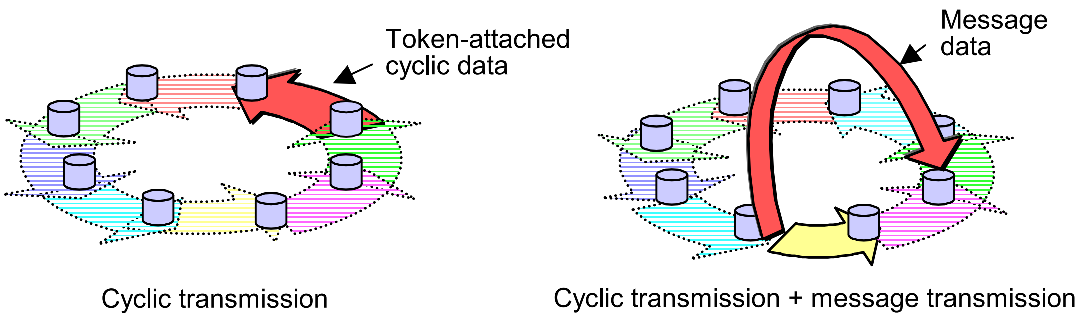


Figure 6-12 Data Communication Types Supported by FL-net

(1) Cyclic transmission

Cyclic transmission provides periodic transmission of data. All nodes can share data via the common memory (shared memory).

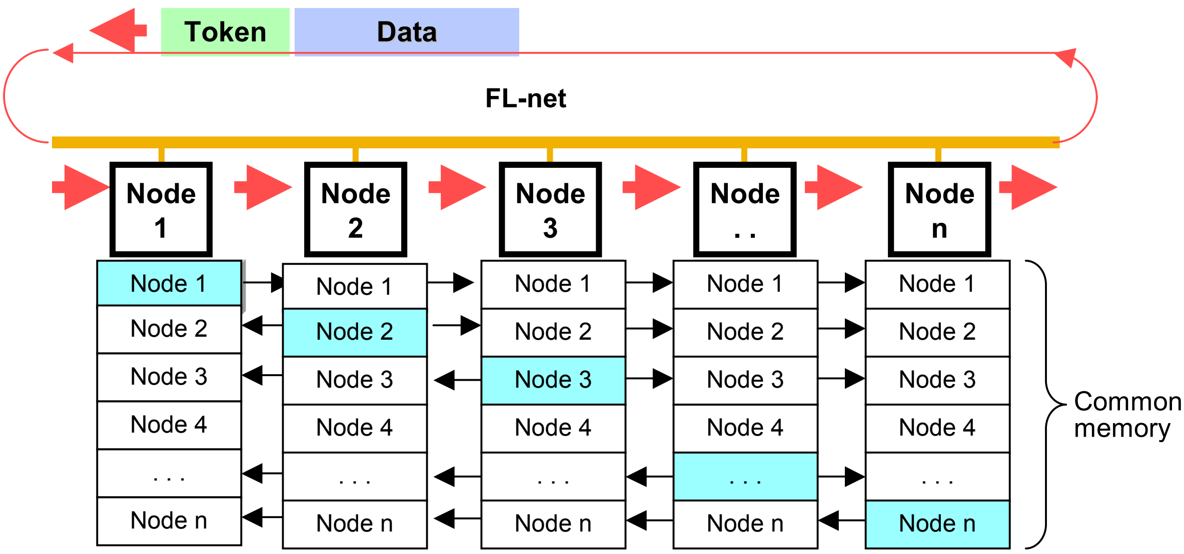


Figure 6-13 Typical Common Memory and Cyclic Transmission

(2) Message transmission

Message transmission provides nonperiodic transmission of data. Under normal conditions, communication is transmitted to a specific node upon request.

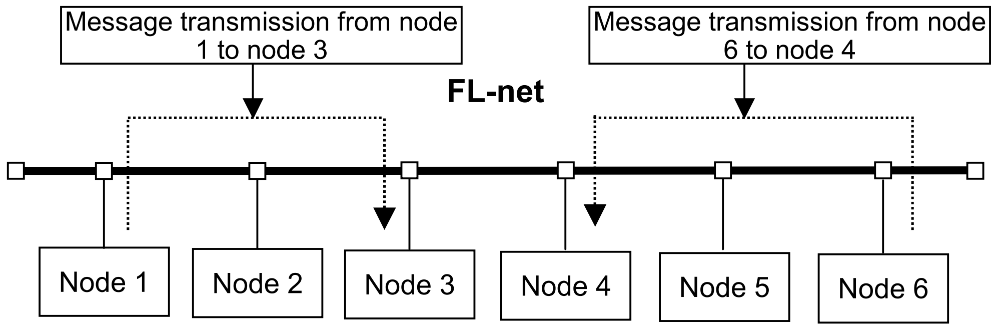


Figure 6-14 Typical Message Transmission

6.2.4 Amount of data transmission

(1) Cyclic transmission

The entire network has 8.5k words (= 8k bits + 8k words). The maximum permissible amount of transmission is 8.5k words per node. Note that each word consists of two bytes.

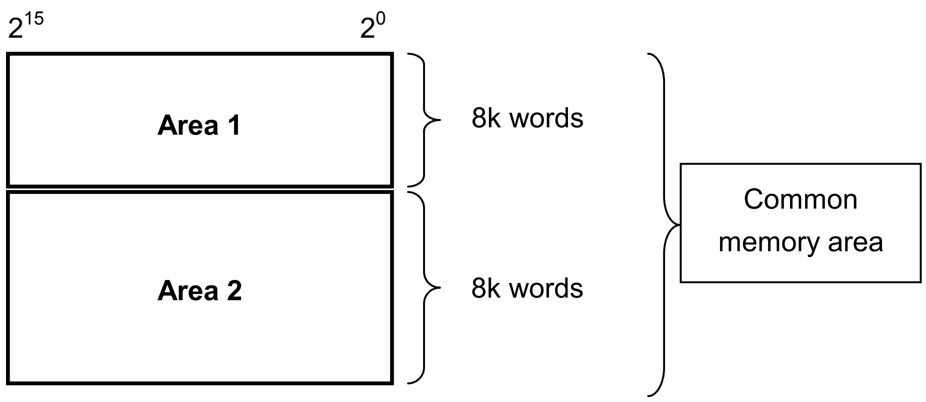


Figure 6-15 Data Amount of Cyclic Transmission

(2) Message transmission

The maximum permissible amount of one message frame is 1,024 bytes (excluding the header).

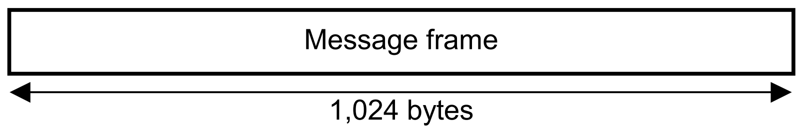


Figure 6-16 Data Amount of Message Transmission

6.2.5 Transfer cycle and monitoring

In cyclic data communication, the common memory is refreshed at virtually fixed intervals. Message transmission is controlled to prevent a single message communication from causing the time limit on the common memory refresh cycle to be exceeded.
Each node constantly monitors message communication frames flowing within the network during the time interval between the reception of a token addressed to itself and the reception of the next token addressed to itself. When no message communication frame flows within the network during such one cycle, 120 percent of the duration of one such cycle is used as the refresh cycle time limit.
Thanks to the above monitoring process, the refresh cycle time limit is dynamically determined by the number of participating nodes within the network.
Example: When only cyclic data communication is maintained by five nodes (no message communication is effected by any nodes)

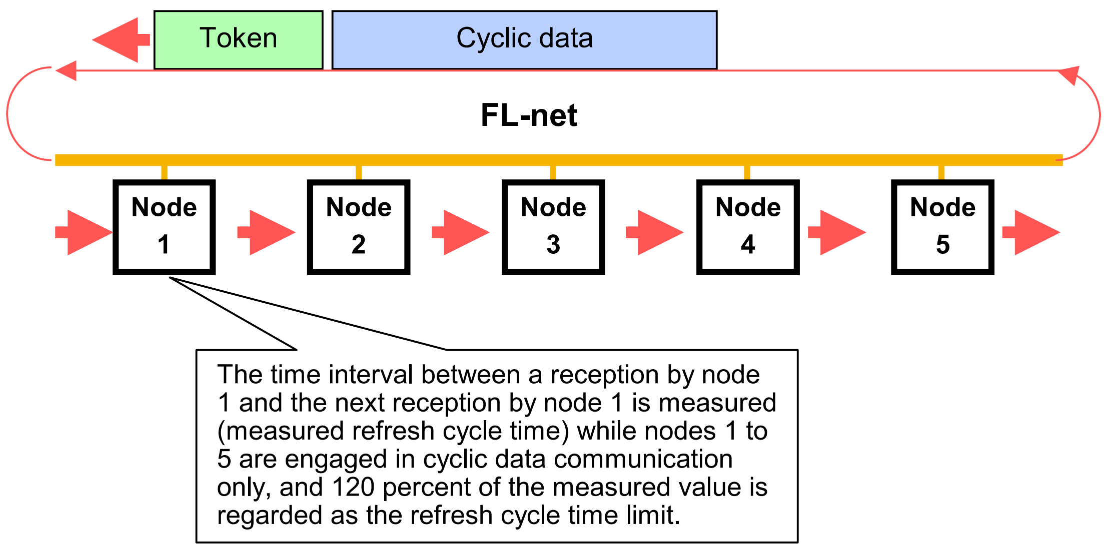


Figure 6-17 Typical Refresh Cycle Time Limit

For the calculation procedure, see “6.4.7 FL.NET module communication time.”

6.2.6 Data area and memory

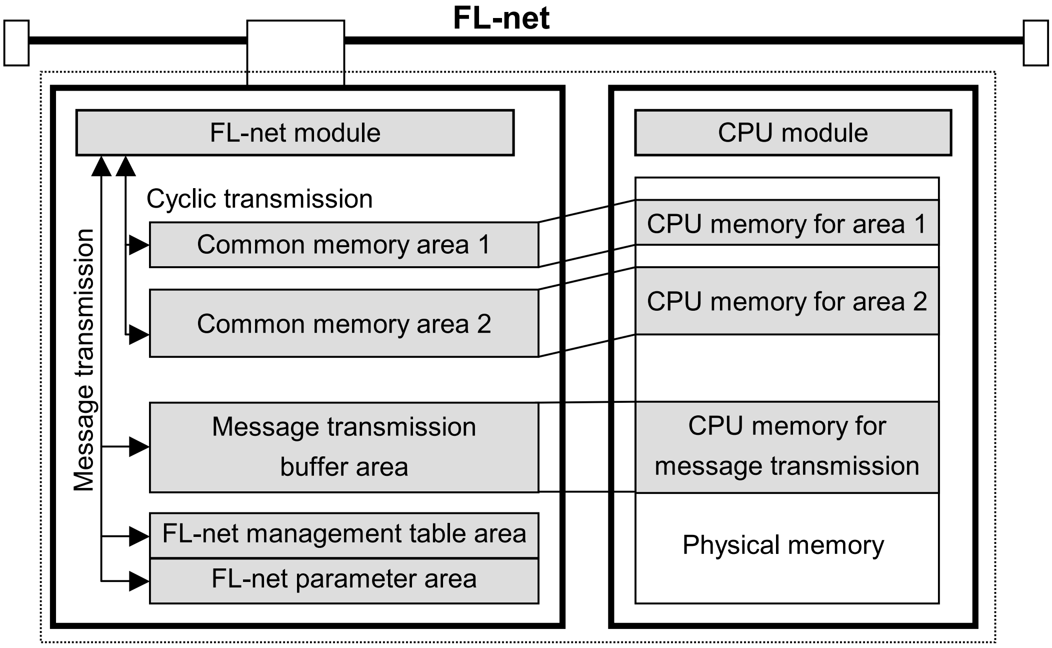


Figure 6-18 Data Area and Memory

6.2.7 Communication management tables

Node status management is provided by means of the self-node management table, participating node management table, and network management table.

(1) Self-node management table
The self-node management table is used to manage the self-node setup.

Table 6-2 Self-node Management Table

Name	Length	Description/data range
Node number	1 byte	1 to 254
Area 1 data starting address for common memory	2 bytes	Word address (0 to 0x1FF)
Area 1 data size for common memory	2 bytes	Size (0 to 0x1FF)
Area 2 data starting address for common memory	2 bytes	Word address (0 to 0x1FFF)
Area 2 data size for common memory	2 bytes	Size (0 to 0x1FFF)
Higher layer status	2 bytes	RUN/STOP/ALARM/WARNING/NORMAL
Token surveillance timeout	1 byte	Variable in 1-s units
Minimum frame interval	1 byte	Variable in 100μs units
Vendor name	10 bytes	Name of the vendor
Maker form	10 bytes	Maker form/device name
Node name (equipment name)	10 bytes	User-defined node name
Protocol version	1 byte	Fixed at 0x80
FA link status	1 byte	Entering/leaving, etc.
Self-node status	1 byte	Node number duplication detection, etc.

(2) Participating node management table

The participating node management table is used to manage the information about nodes that have participated in the network.

Table 6-3 Participating Node Management Table

Name	Length	Description/data range
Node number	1 byte	1 to 254
Higher layer status	2 bytes	RUN/STOP/ALARM/WARNING/NORMAL
Area 1 data starting address for common memory	2 bytes	Word address (0 to 0x1FF)
Area 1 data size for common memory	2 bytes	Size (0 to 0x1FF)
Area 2 data starting address for common memory	2 bytes	Word address (0 to 0x1FFF)
Area 2 data size for common memory	2 bytes	Size (0 to 0x1FFF)
Refresh cycle time	2 bytes	Variable in 1-s units
Token surveillance timeout	1 byte	Variable in 1-s units
Minimum frame interval	1 byte	Variable in 100μs units
FA link status	1 byte	Entering/leaving information, etc.

(3) Network management table

The network management table manages the information common in the network.

Table 6-4 Network Management Table

Name	Length	Description/data range
Token maintenance node number	1 byte	Node that currently holds the token
Minimum frame interval	1 byte	Variable in 100μs units
Refresh cycle time	2 bytes	Variable in 1-s units
Refresh cycle measurement time (current)	2 bytes	Variable in 1-s units
Refresh cycle measurement time (maximum)	2 bytes	Variable in 1-s units
Refresh cycle measurement time (minimum)	2 bytes	Variable in 1-s units

6.2.8 Cyclic transmission and area

(1) Cyclic transmission overview

The cyclic transmission function uses the common memory to periodically exchange data.

- Transmits data when the node holds the token.
- Recognizes nodes conducting no cyclic transmission as far as they participate in the network.
- Ensures that all the cyclic data to be transmitted are transmitted when the token is held.

Token: Only one token basically exists within the network. If two or more tokens should exist within the network, the one with the lowest destination node number takes precedence with the others discarded.

Token frame: A token frame (frame that contains the token) consists of a token destination node number and token transmission node number.
A node becomes a token holder when it agrees with the destination node number for the token in a token frame it receives.

Token sequence: The order of token rotation is determined by a node number. Token rotation occurs in the ascending order of nodes registered in the participating node management table. A node having the maximum node number passes the token to a node having the minimum node number.

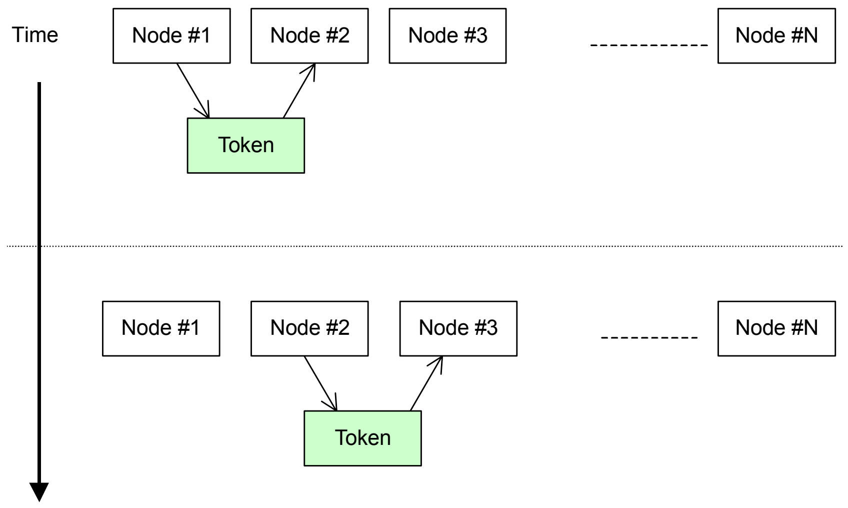


Figure 6-19 Token Rotation and Cyclic Transmission 1

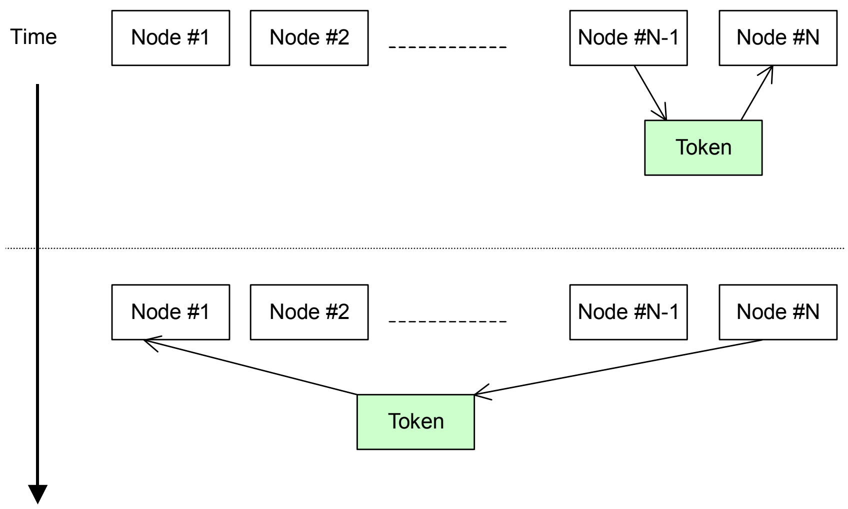


Figure 6-20 Token Rotation and Cyclic Transmission 2

(2) Common memory

The ideas of the common memory are described below:

- Furnishes the nodes conducting cyclic transmission with functions that can be handled as the functions of the common memory.
- Two areas (Area 1 and 2) can be allocated for one node.
- Two or more frames are used for data transmission when the 1,024-byte transmission size limit per frame is exceeded by an area transmitted by one node.
- In the reception of data divided into two or more frames, the common memory does not refresh until all the frames coming from a node are completely received.
- The common memory size furnished by the communication section of one node is fixed at 8.5k words (= 8k bits + 8k words).
- As the common memory's transmission area for one node, Area 1 and 2 can be both set up as desired as far as the maximum permissible size is not exceeded.
- The same data can be shared by the entire system when individual nodes broadcast data at fixed intervals. Nodes on the FL-net exchange data by using their own transmission areas that do not overlap with those of the other nodes. In a common memory operation, the transmission area allocated for one node is the reception area for another node.

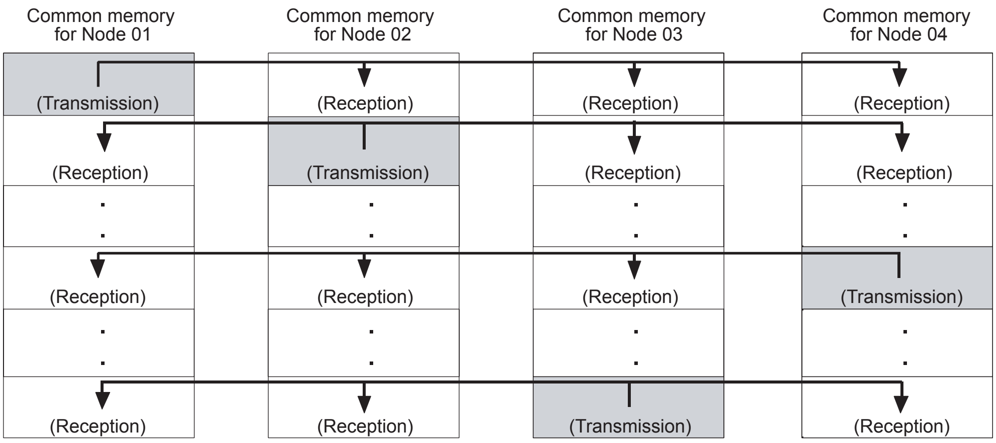


Figure 6-21 Cyclic Transmission Common Memory Area Example 1

It is also possible to use only a reception area of the common memory.

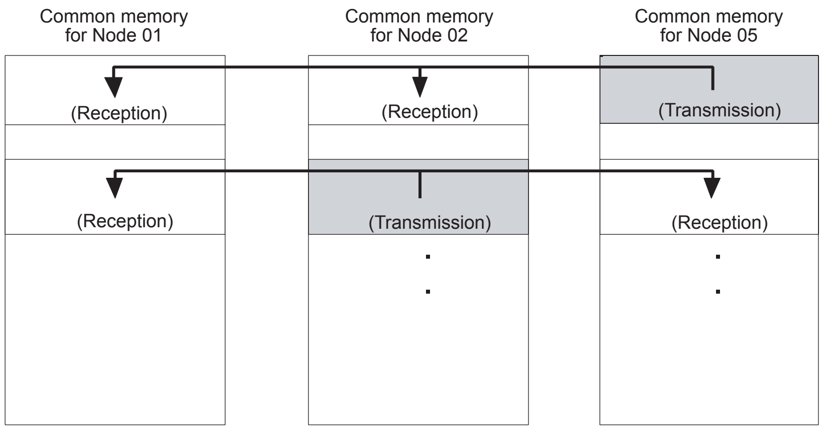


Figure 6-22 Cyclic Transmission Common Memory Area Example 2

(3) Area 1 and 2

One node can allocate two data areas (Area 1 and 2) in the common memory. Transmission area setup is performed in accordance with the area starting address and size.

For area access, a word address is used. Area 1 consists of 0.5k words and Area 2, 8k words.

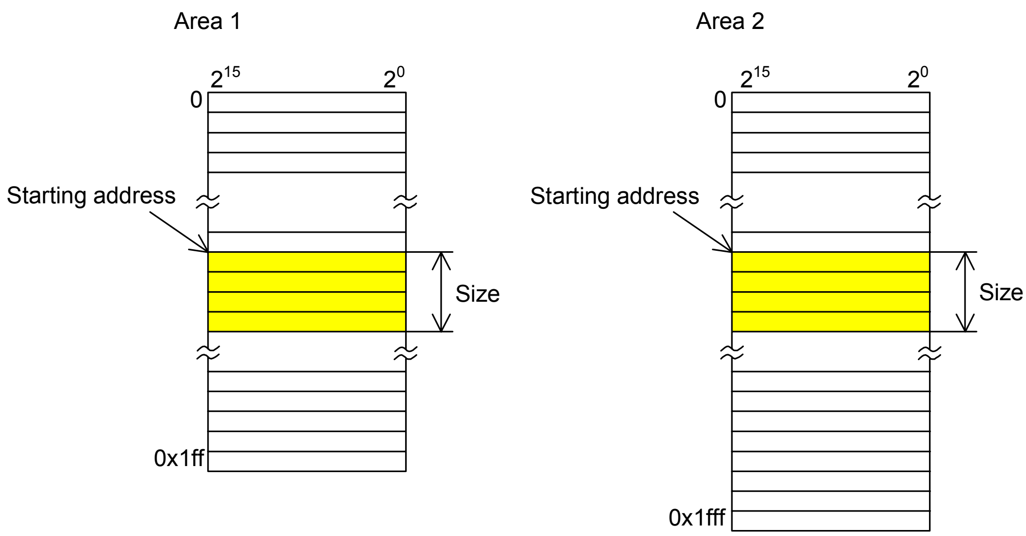


Figure 6-23 Common Memory Area 1 and 2

(4) Data synchronicity assurance

In cyclic transmission, the data to be transmitted is divided into frames depending its size. The following procedures are used to assure common memory (FL.NET internal memory) synchronicity on an individual node basis.

- Transmission timing

  When a data transmission is requested by a higher layer, the cyclic data of the self-node is copied to a buffer to prepare for transmission and then sequentially transmitted. If the size of the data retained by a transmitting node is larger than the maximum permissible size limit for the transmission of one frame, the buffered data is divided into two or more frames for transmission.

- Reception sequence refresh timing

  When a receiving node completely receives cyclic data from another node, it refreshes the associated area while maintaining the synchronicity with a higher layer.
  Even when cyclic data is transmitted after being divided into two or more frames, the associated area refreshes when all the frames sent from a node are completely received. If all such transmitted frames are not received, the entire data received from the transmitting node is discarded.

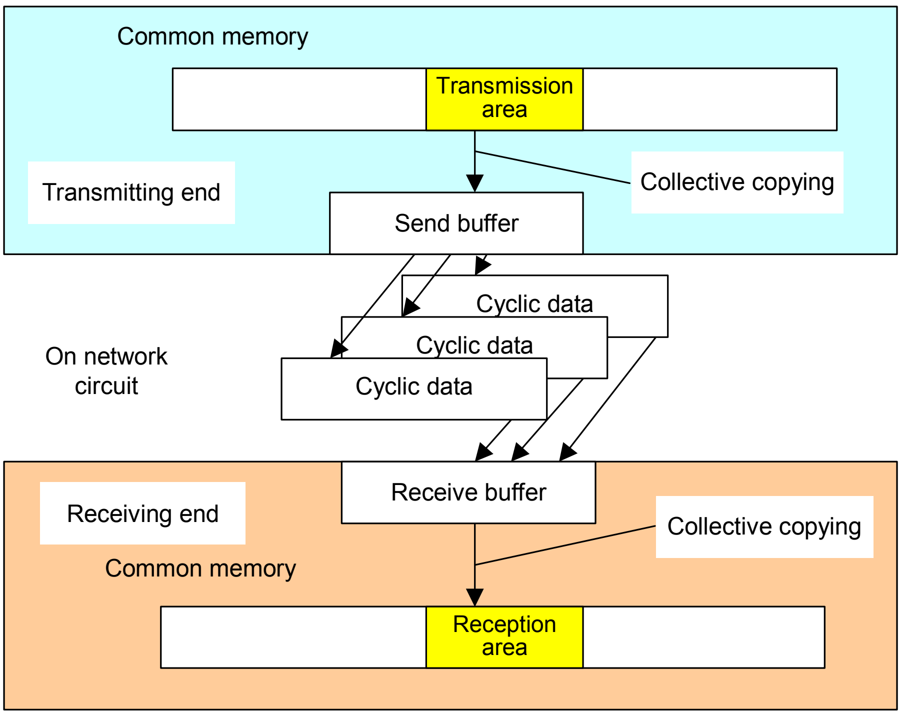


Figure 6-24 Data Synchronicity Assurance

6.2.9 Message transmission

(1) Message transmission overview

The message transmission function supports asynchronous data exchanges that are made between nodes.

The fundamentals of the message transmission function are described below:

- When a node receives the token, it can transmit up to one frame of data before cyclic frame transmission.
- The maximum amount of data that can be transmitted in one session is 1,024 bytes.
- A special algorithm is employed to ensure that the refresh cycle time limit for cyclic transmission is not exceeded.
- One-to-one transmission for sending data to a preselected remote node and one-to-n transmission for sending data to all nodes are both achievable.
- In one-to-one message transmission, the delivery verification feature is available to check whether transmitted data is properly received by a recipient.

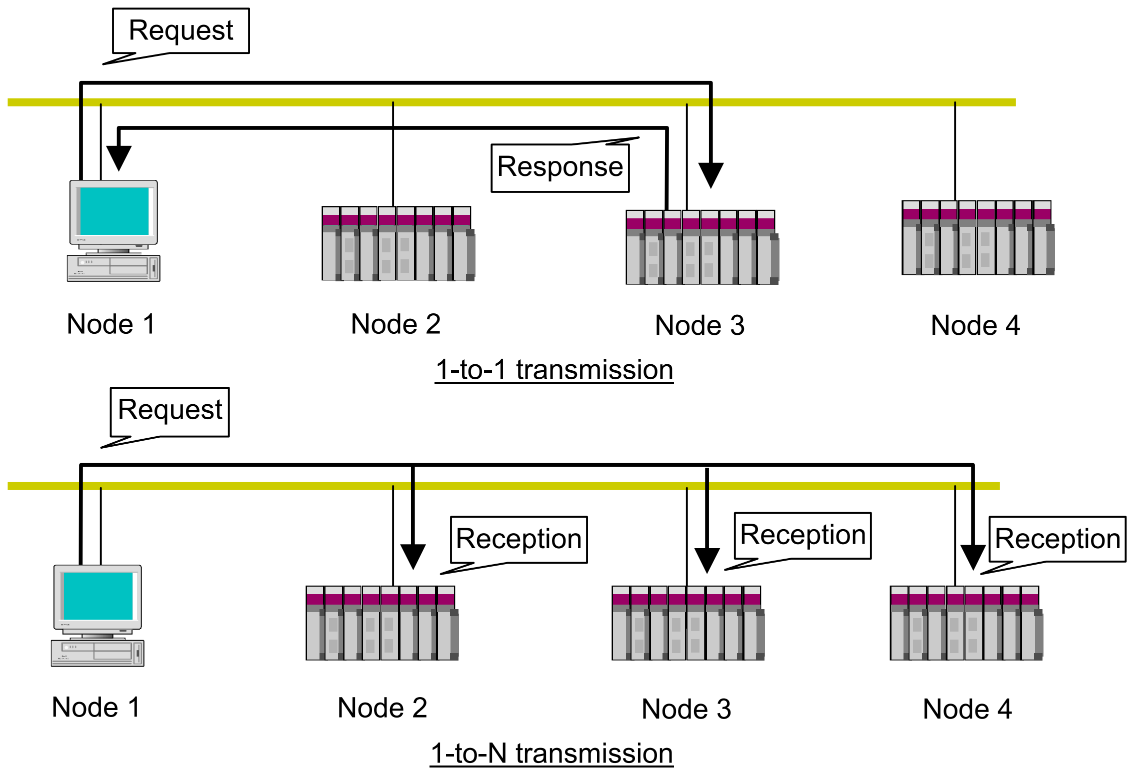


Figure 6-25 Message Transmission Overview

(2) List of supported messages

Table 6-5 lists the types of messages that are supported by the S10mini FL.NET (LQE000).

Table 6-5 List of Supported Message Transmissions

Message	Request	Response
Byte block read	ns	ns
Byte block write	ns	ns
Word block read	√	√
Word block write	√	√
Network parameter read	√	√
Network parameter write	√	√
Run/stop directive	√	ns
Profile read	ns	√
Communication log data read	√	√
Communication log data clear	√	√
Message return	√	√
Transparent message	√	√

√: Supported ns: Not supported

(3) Message function description

● Byte block read

This message function reads a remote node's virtual address space (32-bit address space) from the network byte by byte (in 1-address 8-bit units). This function is not supported by the S10mini FL.NET.

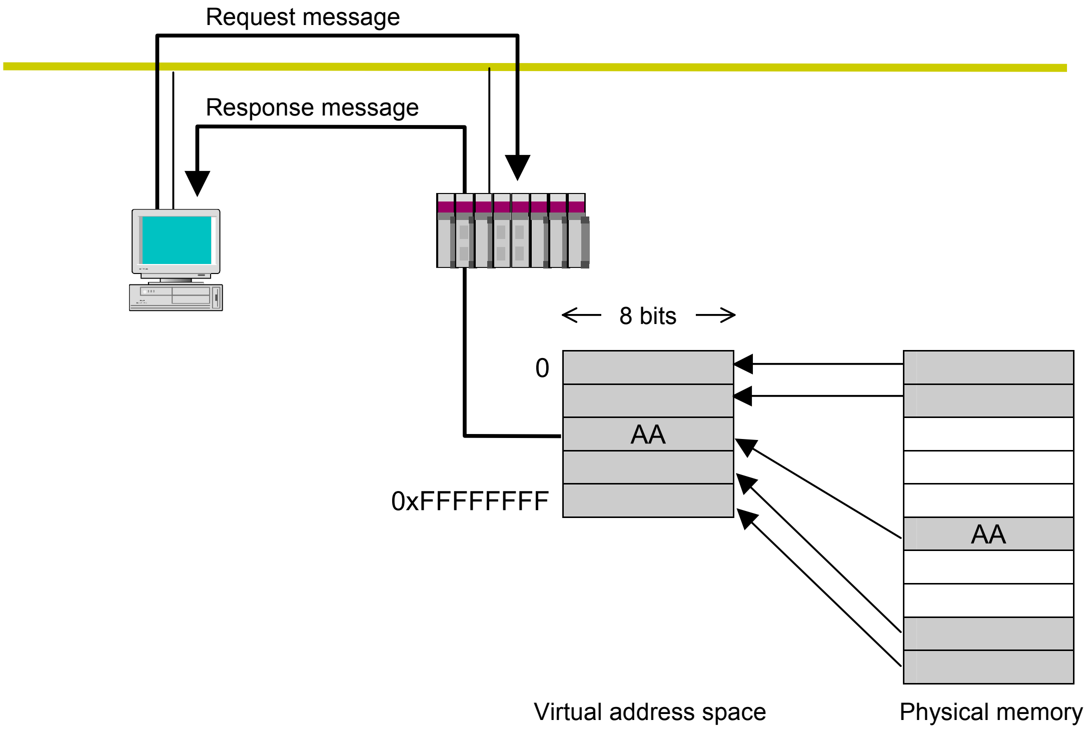


Figure 6-26 Byte Block Read

- Byte block write

  This message function writes into a remote node's virtual address space (32-bit address space) from the network byte by byte (in 1-address 8-bit units). This function is not supported by the S10mini FL.NET.

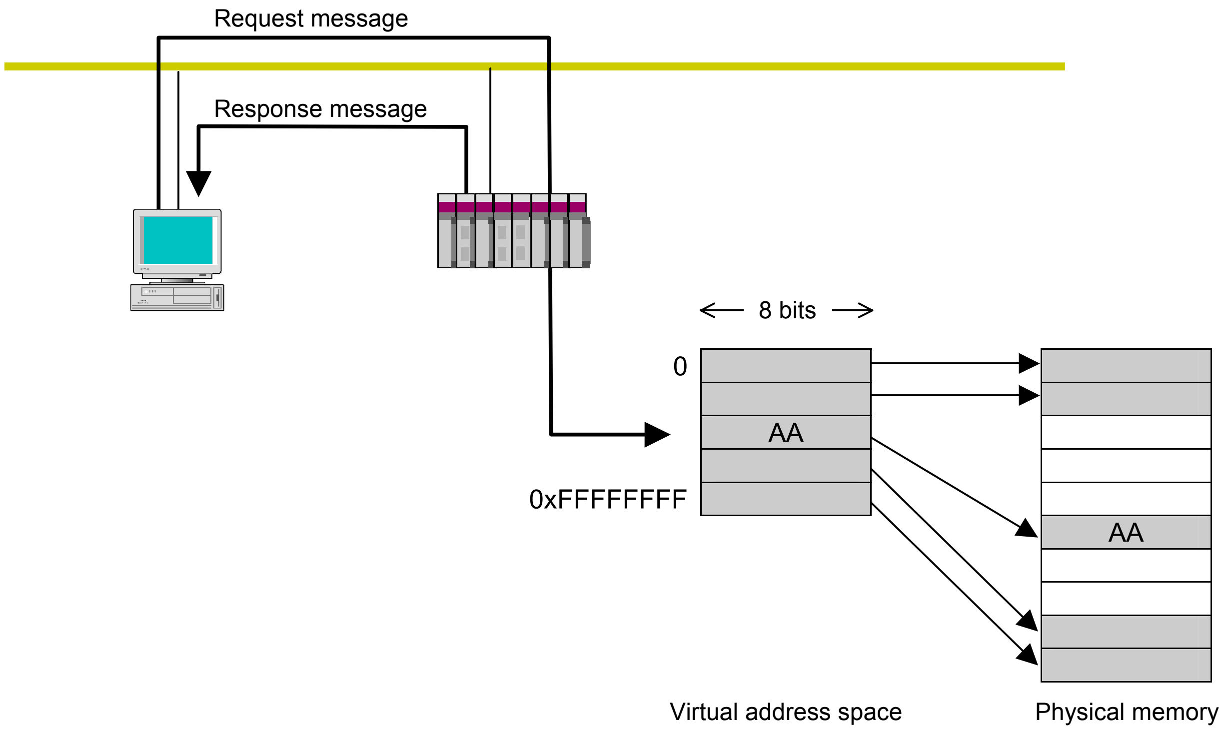


Figure 6-27 Byte Block Write

- Word block read

  This message function reads a remote node's virtual address space (32-bit address space) from the network word by word (in 1-address 16-bit units). For use instructions, see "6.4 Using the FL.NET."

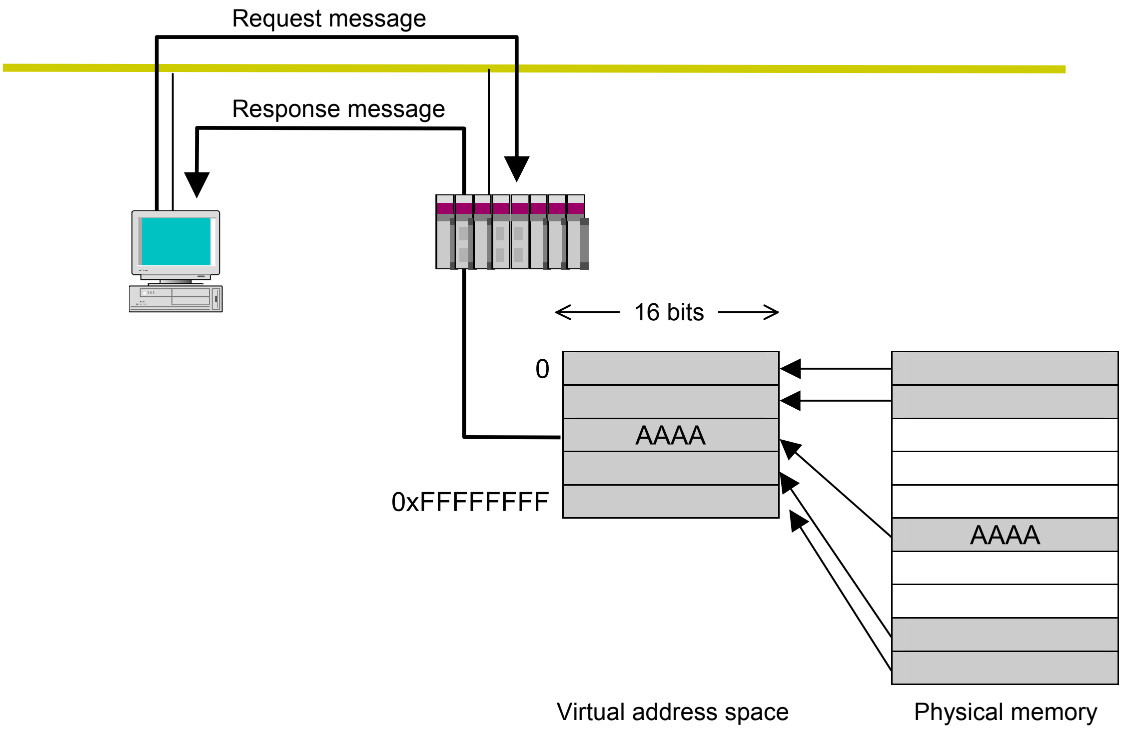


Figure 6-28 Word Block Read

- Word Block Write

  This message function writes into a remote node's virtual address space (32-bit address space) from the network word by word (in 1-address 16-bit units). For use instructions, see "6.4 Using the FL.NET."

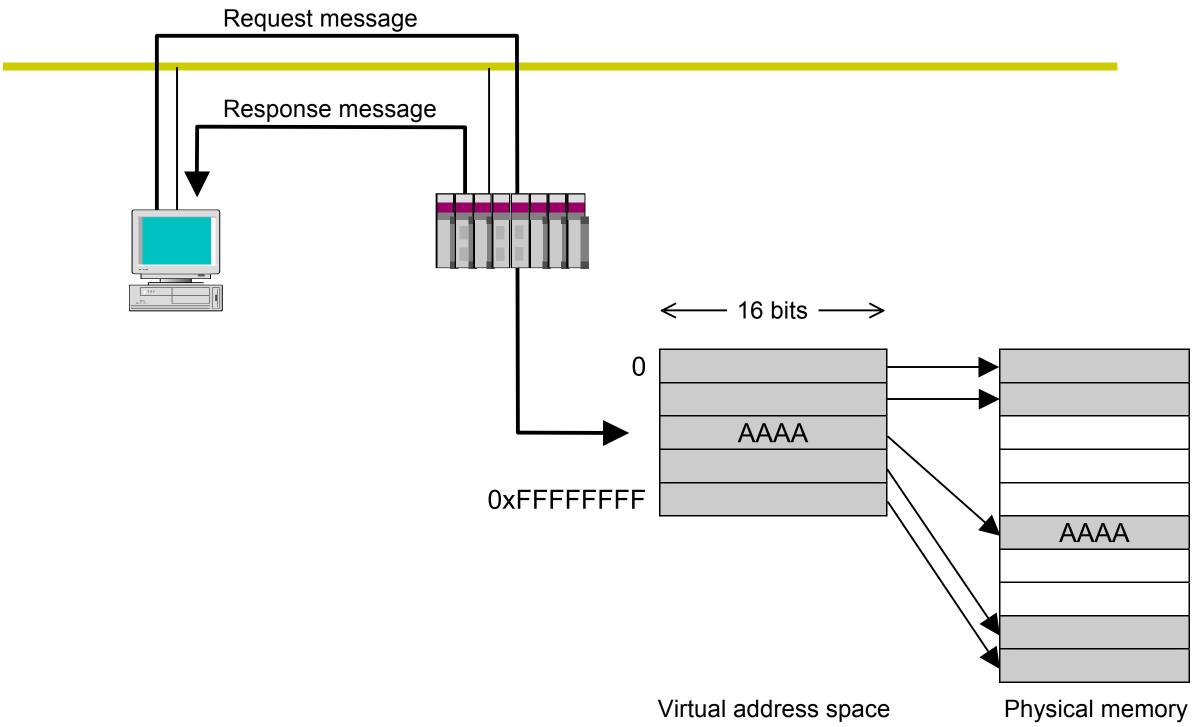


Figure 6-29 Word Block Write

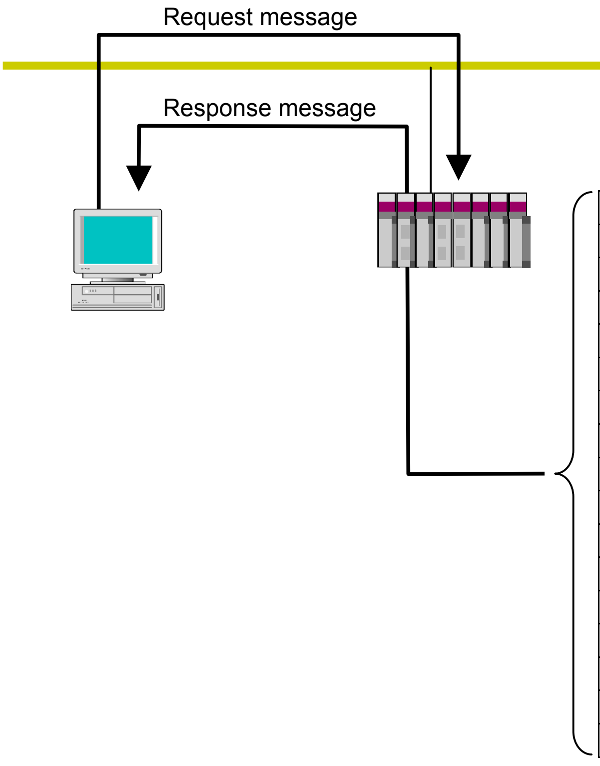
- Network parameter read

This function reads a remote node's network parameter information from the network.

The table below lists the items of information that this function reads. For use instructions, see "6.4 Using the FL.NET."

Table 6-6 Network Parameter Information

Node number
Vendor name
Maker form
Node name (equipment name)
Common memory address and size
Token surveillance timeout
Refresh cycle time
Measured refresh cycle time (measured value)
Minimum frame interval
Higher layer status
FL-net status
Protocol version



Node number
Maker form
Maker form
Node name (equipment name)
Starting address of area 1
Size of area 1
Starting address of area 2
Size of area 2
Token surveillance timeout
Minimum permissible frame interval
Higher layer status
FL-net status
Protocol version
Refresh cycle time (RCT) setting
Refresh measurement cycle time (current)
Refresh measurement cycle time (maximum)
Refresh measurement cycle time (minimum)

Network parameters

Figure 6-30 Network Parameter Read

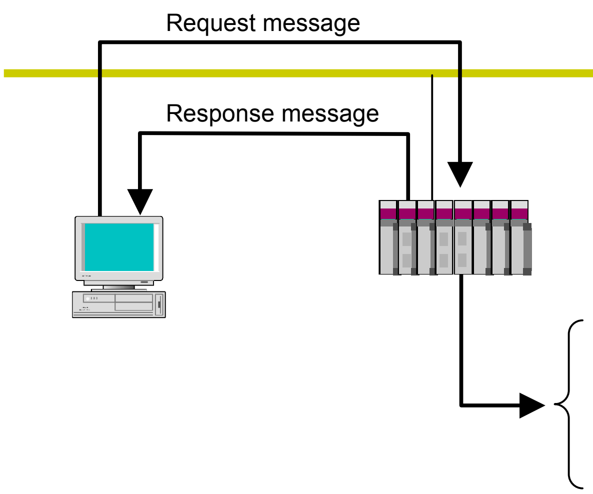
- Network parameter write

  This function edits a remote node’s network parameter information from the network. The following items of information can be edited. For use instructions, see “6.4 Using the FL.NET.”

  - Node name (equipment name)
  - Common memory area address and size

    If the starting address of area 1 or 2 is outside the common memory, an error occurs. However, the node name is not checked.

    If the common memory address or size is changed, the remote node departs from the network and then participates in it again. If only the node name is changed, the remote node does not depart from the network.



Node number
Vendor name
Maker form
Node name (equipment name)
Starting address of area 1
Size of area 1
Starting address of area 2
Size of area 2
Token surveillance timeout
Minimum frame interval
Higher layer status
FL-net status
Protocol version
Refresh cycle time (RCT) setting
Refresh measurement cycle time (current)
Refresh measurement cycle time (maximum)
Refresh measurement cycle time (minimum)

Network parameters

Figure 6-31 Network Parameter Write

- Run/stop directive

  This function allows the network to remotely run or stop the operation of a device connected to the FL-net. Only the associated request is supported by the S10mini FL.NET. For use instructions, see “6.4 Using the FL.NET.”

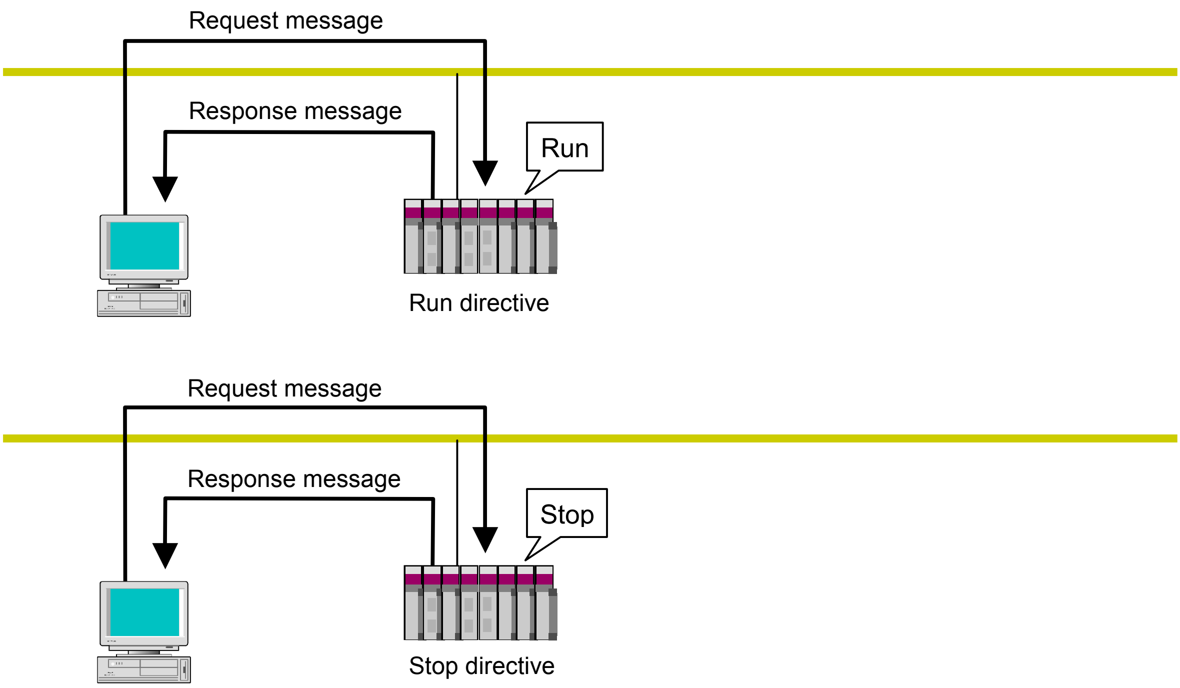


Figure 6-32 Run/Stop Directive

- Profile read

  This function reads from the network the device profile system parameters that represent the information about a remote node. Only the associated response is supported by the S10mini FL.NET.

  The system parameters represent the following items of information:

  • Common parameters (essential)

  • Device-specific parameters (optional)

  For the details of device profile system parameters, see “9 APPENDIXES.”

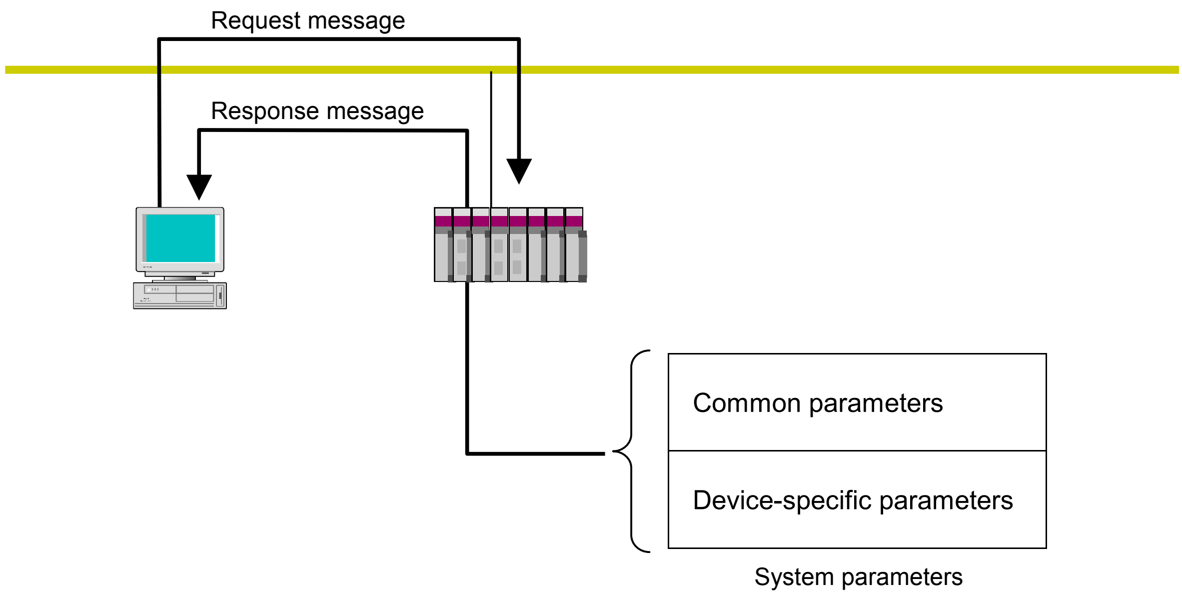


Figure 6-33 Profile Read

- Communication log data read

  This function reads a remote node’s log information from the network.　For use instructions, see “6.4　Using the FL.NET.”

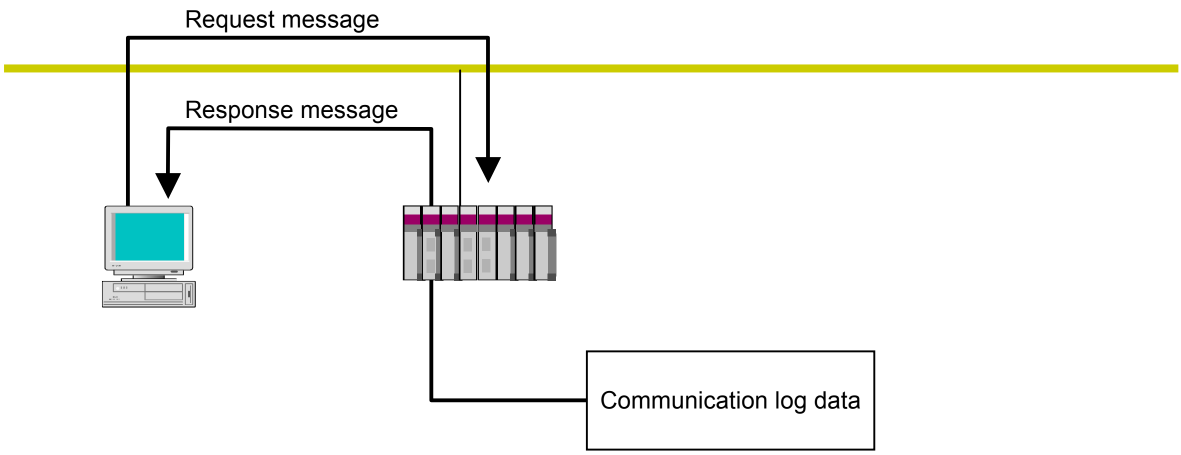


Figure 6-34　Communication Log Data Read

- Communication log data clear

  This function clears a remote node’s log information from the network.　For use instructions, see “6.4　Using the FL.NET.”

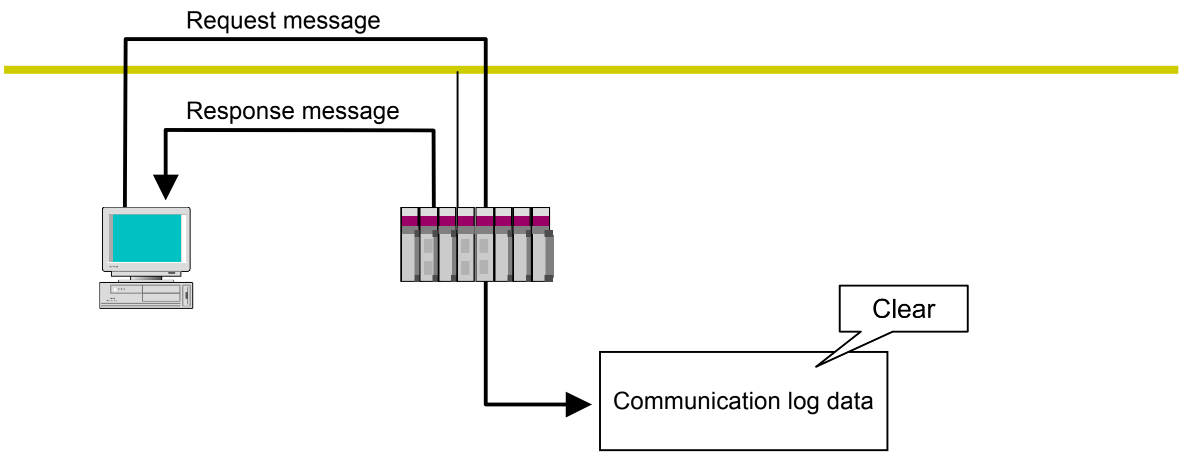


Figure 6-35　Communication Log Data Clear

- Message return

  This function returns a received message.

  A message return automatically takes place within the FL.NET.

  For use instructions, see “6.4 Using the FL.NET.”

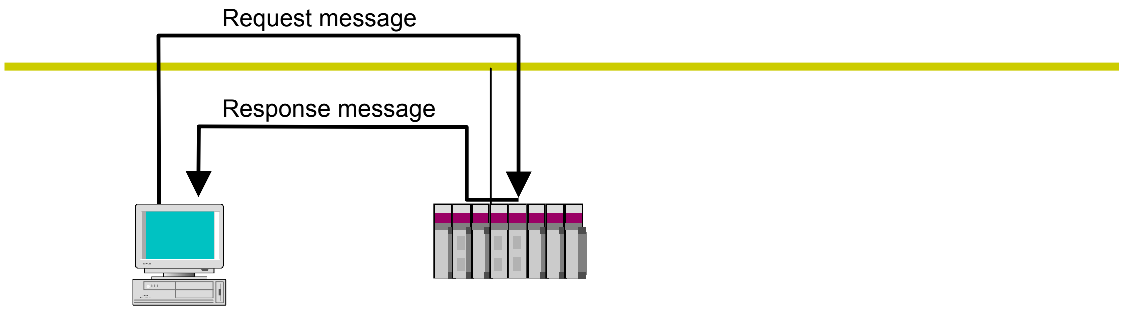


Figure 6-36 Message Return Function

- Transparent message transmission

  This function offers a transparent service to an FL-net higher layer.

  This service notifies an FL-net higher layer of a received message. Upon receipt of such a notification, the FL-net higher layer sends the same notification to the user interface level. When the notification is sent to the user interface level, it is necessary to create and return an associated response. For use instructions, see “6.4 Using the FL.NET.”

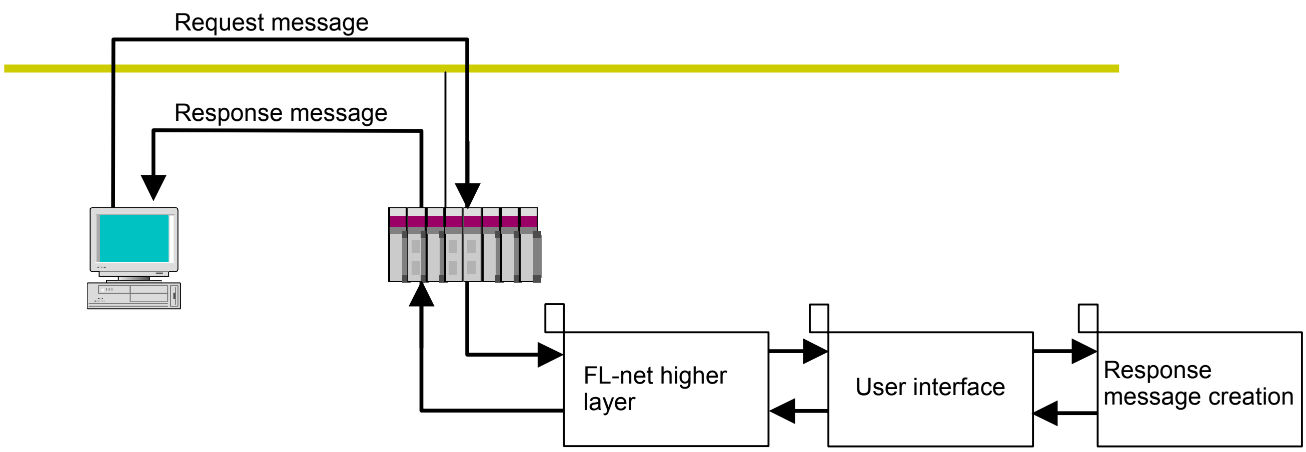
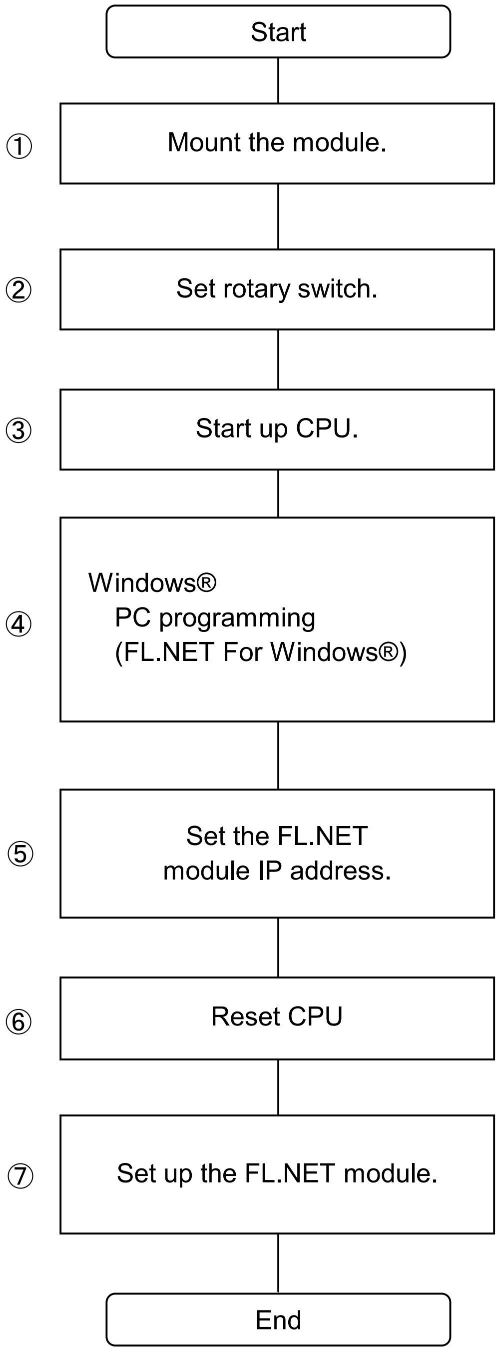


Figure 6-37 Transparent Message Transmission

6.3 FL.NET Setup Procedures

6.3.1 Startup procedure

The startup procedure for the module is described below:



① Turn OFF the CPU unit and then install the FL.NET module.

② Set the MODU No. selector switch on the FL.NET module, as indicated below:

MODU No.		Description
Main	Sub	
0	1	10BASE-5 communication
2	3	10BASE-T communication

③ Turn ON the CPU unit.

④ Connect the CPU unit to the Windows® personal computer via an RS-232C interface cable or Ethernet. Start the setup tool named “FL.NET For Windows®” (refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)”).

⑤ Set the IP address and subnet mask for the FL.NET module. (For IP address and subnet mask setup procedures, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)” and see “(1) IP address/subnet setup precautions,” on the next page.)

⑥ Reset the CPU unit.

⑦ Set up the FL.NET module.

(1) IP address/subnet setup precautions

For the FL.NET module, you can freely set the IP address and subnet mask. However, you must comply with the FL-net protocol. It is therefore recommended that you use Class C for IP address setup and set the network address to 192.168.250. Also, be sure that the subnet mask setting is 255.255.255.0.

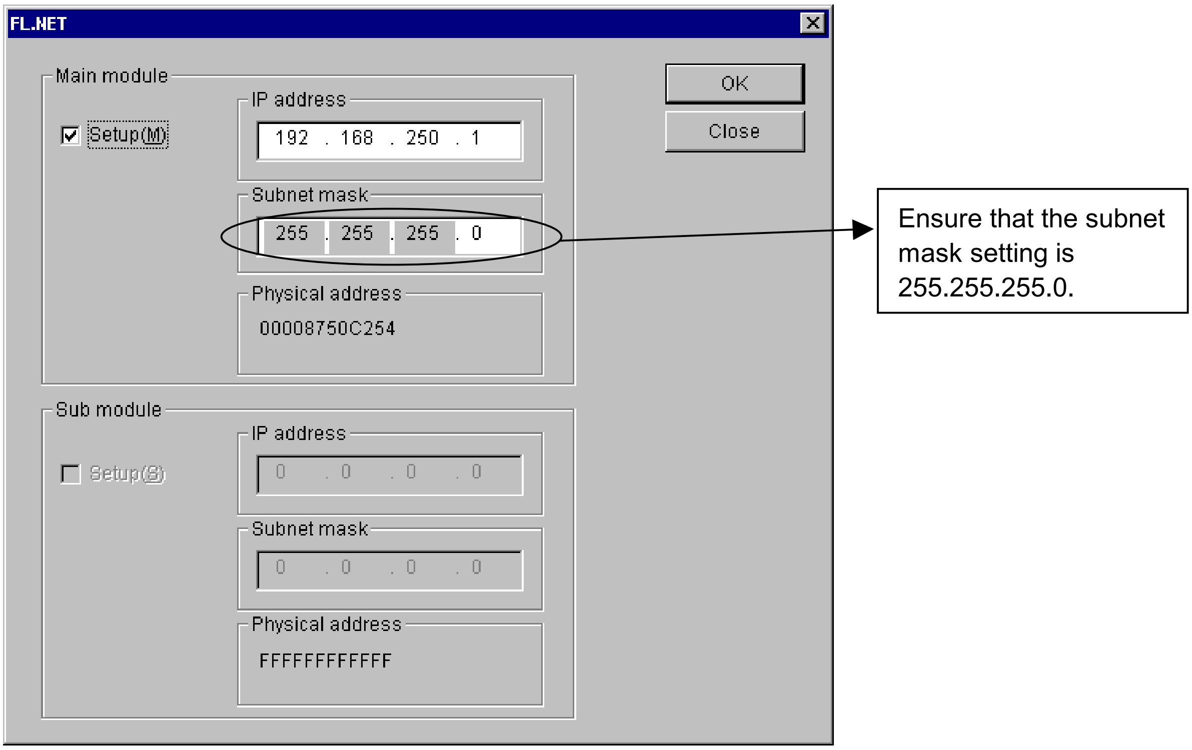


Figure 6-38 IP Address Setup Window

Note 1: If all the IP address entries are set to 0 or 255, an input error occurs.

Note 2: If all the host number entries are set to /0 or /F, the FL.NET module performs the same process as in cases where IP address setup is not completed.

Note 3: If the FL.NET module is not installed, the IP address setup window of the setup tool named “FL.NET For Windows®” indicates the physical address /FFFFFFFFFFFF. If the FL.NET module is installed but IP address setup is not completed, the window shows the physical address /000000000000.

Note 4: When referencing the physical address, install the FL.NET module and set the IP address. The IP address and subnet mask can be referenced even if the FL.NET module is not installed.

Note 5: If the IP address is not set or its setting is cleared, for instance, by clearing the memory at the time of OS loading, all the processes come to a stop except IP address setup, with the LER LED on the FL.NET module glowing and the CPU unit indicator showing the following error message:
When the main module IP address is not set: "FLNMIPNG"
When the submodule IP address is not set: "FLNSIPNG"

Note 6: Before performing self-node setup and remote node reception setup, be sure to set the IP address for the FL.NET module. If you perform self-node setup and remote node reception setup while leaving FL.NET module IP address setup incomplete, a timeout error occurs so that the setup process does not end normally.

6.3.2 Module installation and switch setup

For the procedures for module installation and switch setup, see "4 FL.NET MODULE INSTALLATION."

6.3.3 Tool connection procedure

For the tool connection procedure, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

6.3.4 Tool startup procedure

For the tool startup procedure, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

6.4 Using the FL.NET

6.4.1 Link parameter setup procedure

The link parameters are used to set up the FL.NET module common memory area for each node. Use the setup tool named “FL.NET For Windows®” to perform link parameter setup. The self-node setup window is shown below.

The “Current value” section shows the current settings for the FL.NET module.

To change the settings, enter desired values in the “Rewriting value” section and click [OK] button. The newly entered link parameter settings are then saved in the FL.NET module.

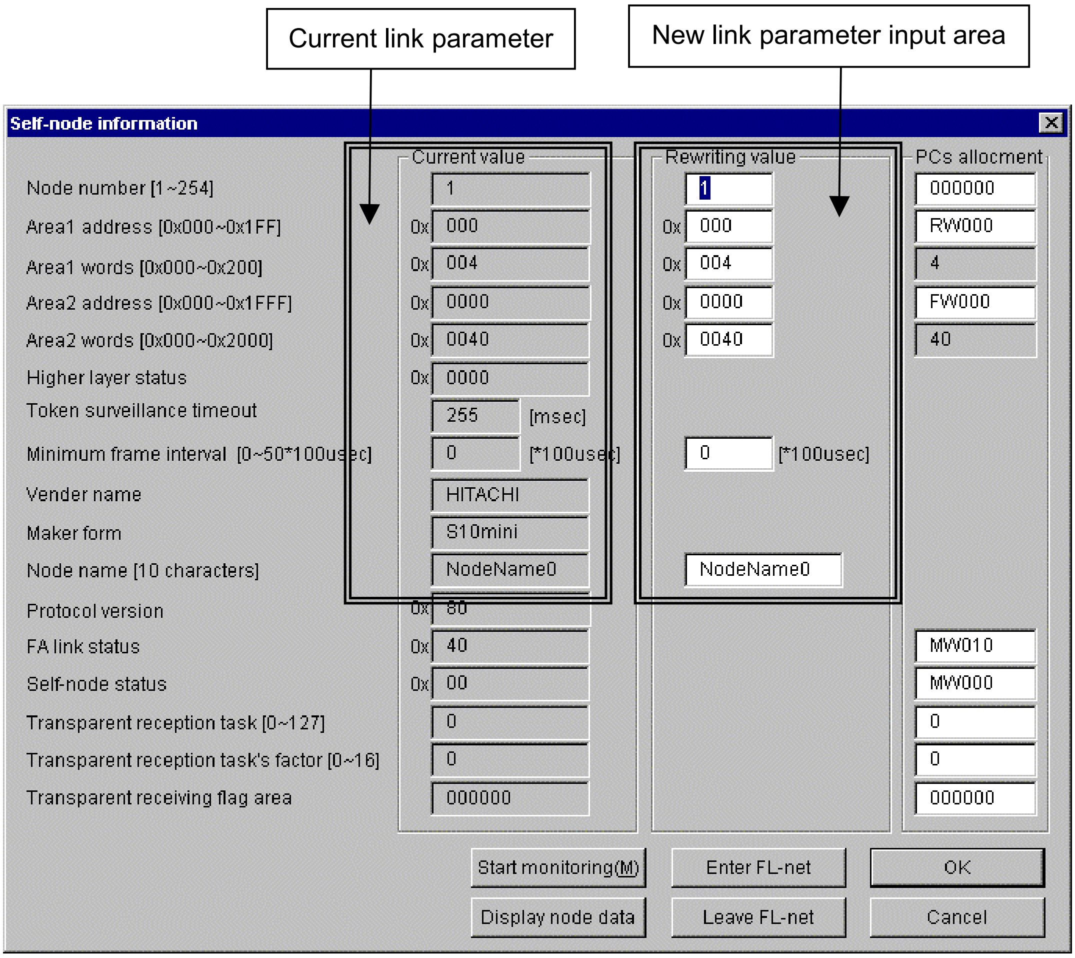


Figure 6-39 Link Parameter Setup Window

The table below shows the acceptable input value range for each setting item.

Input item	Description	Acceptable setting range
Node number	This number is used to identify the nodes for FL-net communication purposes.	1 to 254
Token surveillance timeout	This value represents the period of time (ms) during which the token addressed to the self-node can be held.	Fixed at 255 (cannot be edited by the user)
Minimum frame interval	This value (variable in 100μs units) represents the minimum time intervals at which frames can be consecutively transmitted or the minimum time interval between the instant at which the token addressed to the self-node is received and the instant at which frame transmission starts.	0 to 50
Area 1 address	This value specifies the starting address for the area 1 transmission area. Enter a hexadecimal number.	0 to 0x1FF (511)
Area 1 words	This value specifies the size of the area 1 transmission area. Enter a hexadecimal number.	0 to 0x200 (512)
Area 2 address	This value specifies the starting address for the area 2 transmission area. Enter a hexadecimal number.	0 to 0x1FFF (8191)
Area 2 words	This value specifies the size of the area 2 transmission area. Enter a hexadecimal number.	0 to 0x2000 (8192)
Node name	This entry represents the self-node name.	String of no more than 10 single-byte alphanumeric characters

For detailed procedure descriptions, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

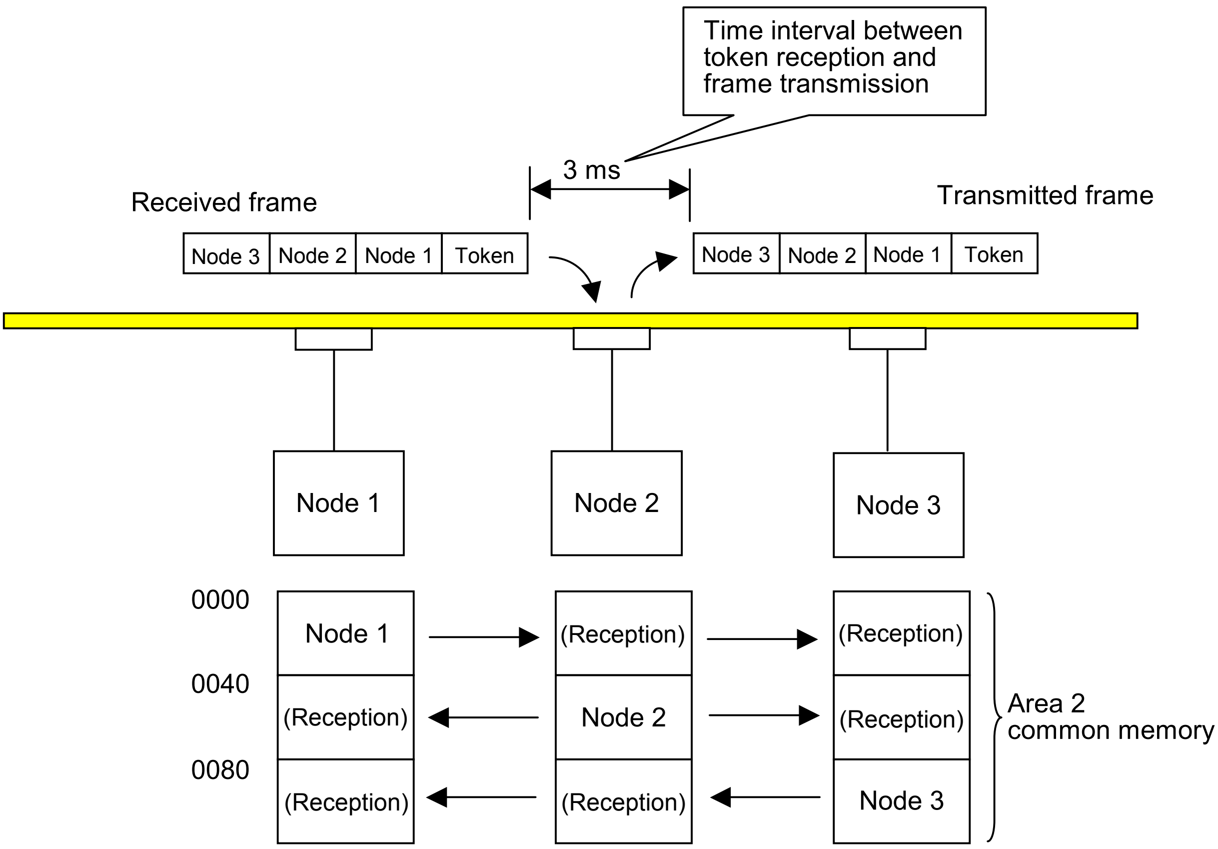
Note 1: If the minimum frame interval is set to "0", there will be no interval between frame transmissions.
Note 2: When no common memory transmission is to be initiated from the self-node, set both the starting address and size to "0".
Note 3: Ensure that the minimum frame interval setting is not smaller than 20 (2 ms). The use of the smaller settings may result in failure to establish connections to the other nodes.

Example: When the following link parameter settings are employed, the FL-net module operates as indicated below:

Node number : 1
Area 1 address : 0x000
Area 1 words : 0x000
Area 2 address : 0x0000
Area 2 words : 0x0040
Minimum frame interval : 30
Node name : CPU01

Node number : 2
Area 1 address : 0x000
Area 1 words : 0x000
Area 2 address : 0x0040
Area 2 words : 0x0040
Minimum frame interval : 30
Node name : CPU02

Node number : 3
Area 1 address : 0x000
Area 1 words : 0x000
Area 2 address : 0x0080
Area 2 words : 0x0040
Minimum frame interval : 30
Node name : CPU03



Data is exchanged using the area 2 common memory area as shown above.

6.4.2 CPU memory allocation procedure

The FL.NET module allocates the common memory area in the CPU memory. The data received by the FL.NET module is transferred to the resulting CPU memory.

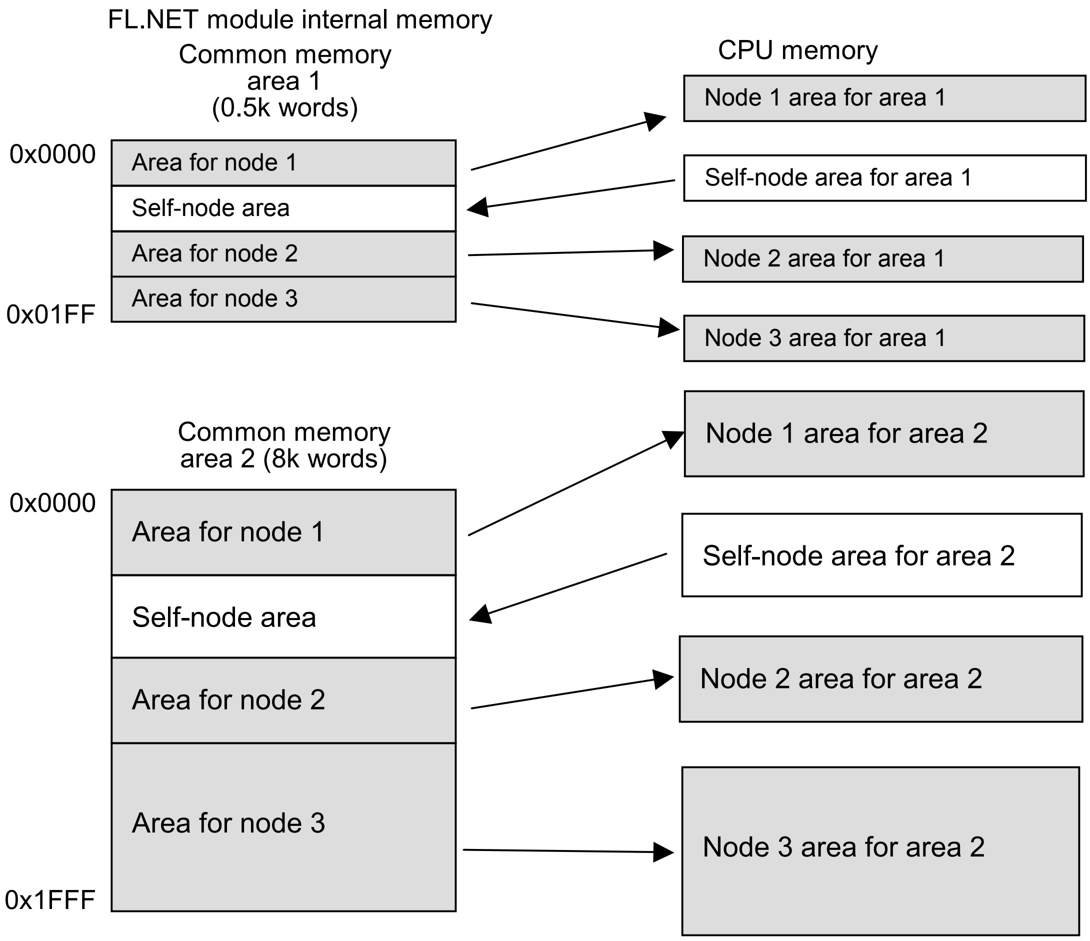


Figure 6-40 Common Memory Data Transfer Image of FL.NET Module

Setup for common memory area allocation in the CPU memory is to be performed with the setup tool named “FL.NET For Windows®.”

For the details of the procedures to be performed from various setup windows, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139).”

Table 6-7 shows the addresses that are available for common memory area allocation in the CPU memory.

Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory

Selectable register/address	Remarks
XW000 to XWFF0	External input
YW000 to YWFF0	External output
JW000 to JWFF0	Transfer register
QW000 to QWFF0	Receive register
GW000 to GWFF0	Global link register
RW000 to RWFF0	Internal register
EW400 to EWFF0	Event register
MW000 to MWFF0	Internal register
DW000 to DWFFF	Function data register
FW000 to FWBFF	Function work register
/100000 to /4FFFFE	Extension memory

(1) Allocating the self-node CPU memory
With the setup tool named "FL.NET For Windows®," open the self-node setup window.
In the "PCs allocation" section, enter the addresses for the PI/O or extension memory to be allocated as the self-node CPU memory area. After completion of input, click [OK] button.
The entered settings are then saved in the FL.NET module.

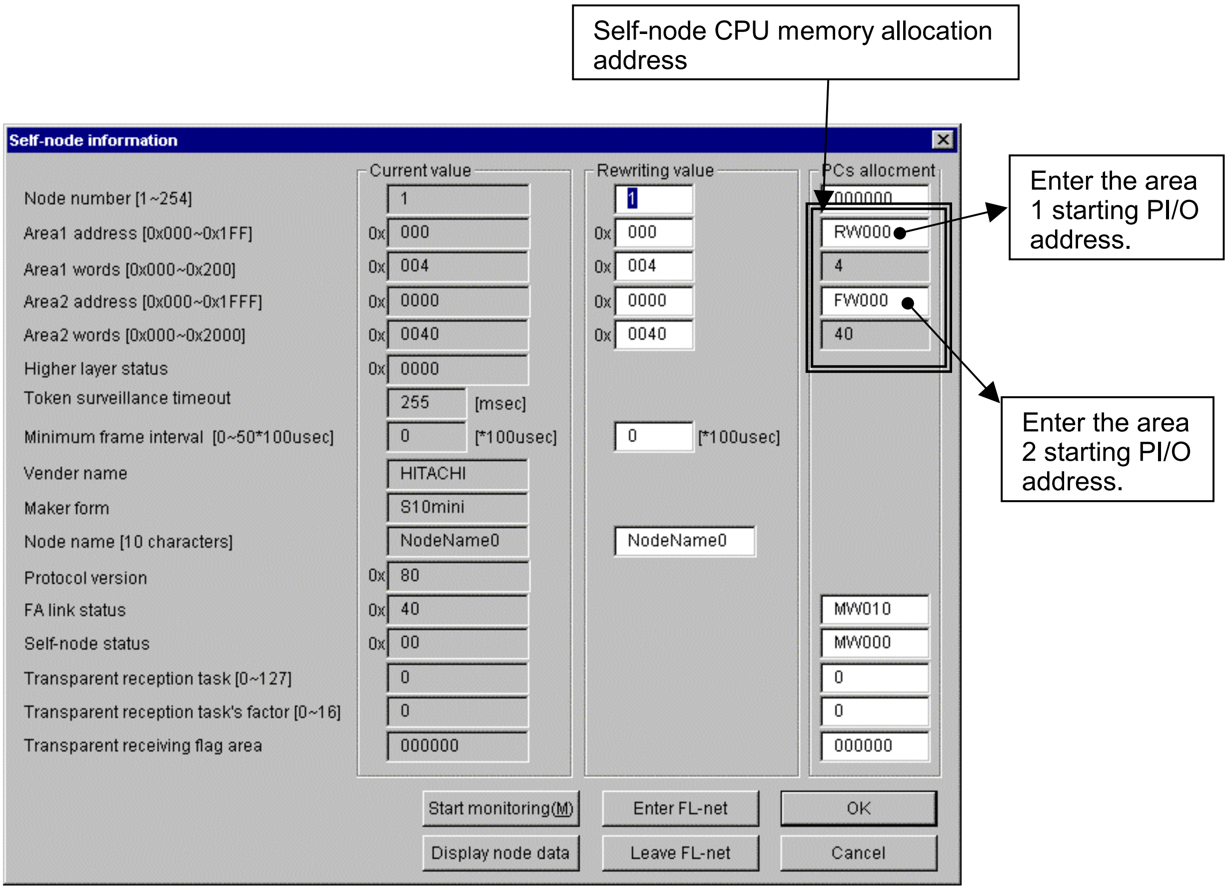


Figure 6-41 Self-node CPU Memory Allocation Window

Input item	Description	Acceptable selection range
Area 1 address (PCs allocation)	Set the starting address for the allocation of area 1 common memory and CPU memory.	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."
Area 2 address (PCs allocation)	Set the starting address for the allocation of area 2 common memory and CPU memory.	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."

Note 1: The area 1 word count (area 2 word count) for PCs allocation is the same as the link parameter area 1 word count (area 2 word count).

Note 2: The allocation address setting for the self-node CPU memory must not be a duplicate of any one of those for the other nodes.

Note 3: CPU memory allocation cannot be performed over more than one PI/O range.

Example: An unacceptable setup example is given below:

When the word count (size) setting for area 1 is 3 words, "RWFF0" cannot be specified for CPU memory allocation (because the RW area limit would be exceeded).

Note 4: If the link parameter area 1 word count (area 2 word count) is set to "0", the area 1 address (area 2 address) for PCs allocation cannot be set.

(2) Setting the CPU memory area for the other nodes

Perform reception setup for the other nodes with the setup tool named “FL.NET For Windows®.”

The data received from various nodes are transferred to the area allocated in the CPU memory. If the CPU memory area word count differs from the common memory size setting (link parameter area word count), the FL.NET module transfers data from the common memory. The amount of this data transfer is determined from the number of words specified by the CPU memory area word count setting. To acquire the entire data received by the common memory, ensure that the size setting made is equal to the common memory setting (link parameter setting) for the other nodes.

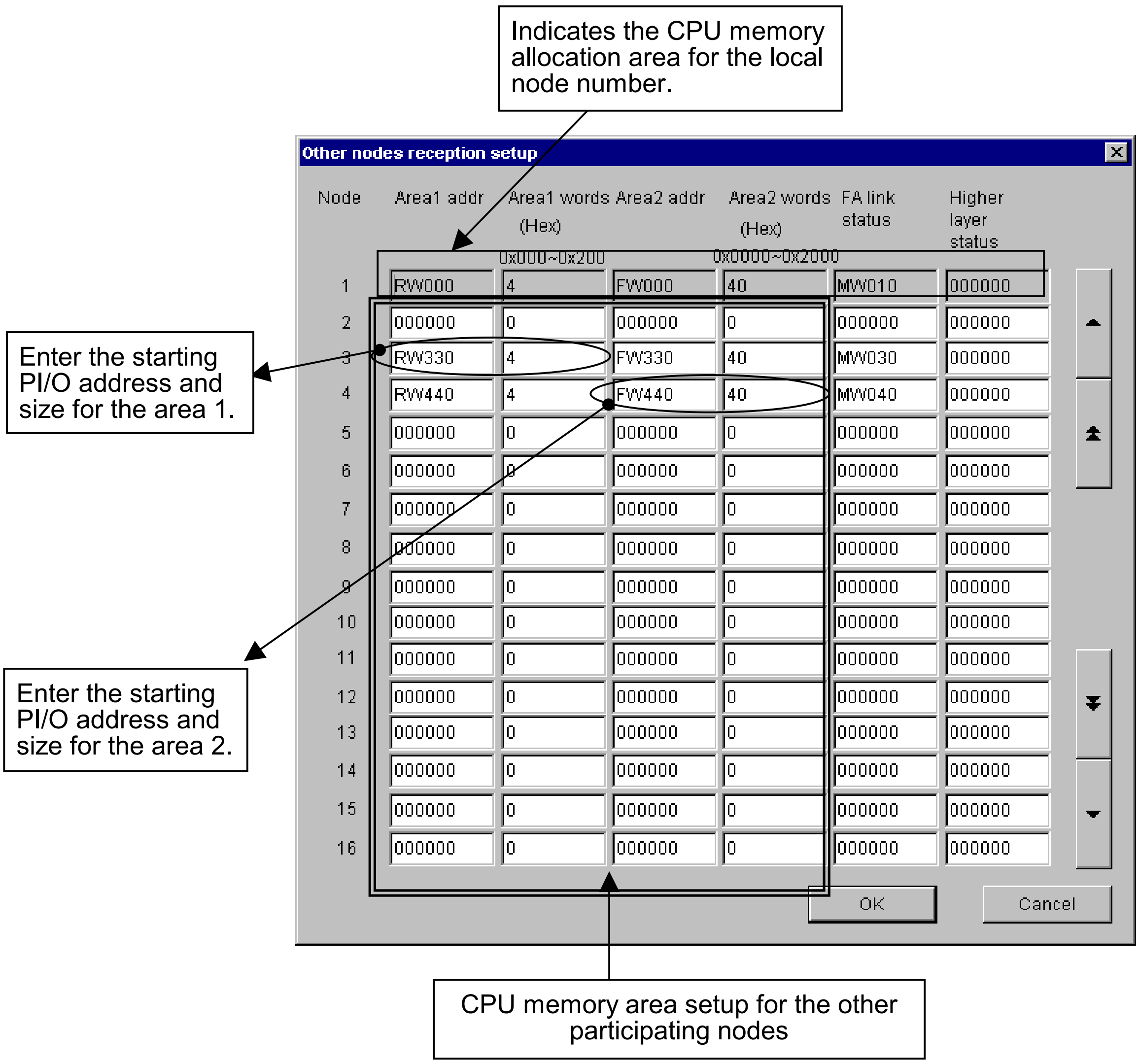


Figure 6-42 Common Memory Area Setup Window for Other Participating Nodes

The table below shows the acceptable input ranges for various input items.

Input item	Description	Acceptable selection range
Area 1 address	Specify the starting address of the area 1 reception area for the specified node (having the node number indicated at the left end).	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."
Area 1 word count	Specify the size of the area 1 reception area for the specified node (having the node number indicated at the left end). Enter a hexadecimal number.	0 to 0x200 (512)
Area 2 address	Specify the starting address of the area 2 reception area for the specified node (having the node number indicated at the left end).	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."
Area 2 word count	Specify the size of the area 2 reception area for the specified node (having the node number indicated at the left end). Enter a hexadecimal number.	0 to 0x2000 (8192)

Note 1: The area selected for reception setup for an other (remote) node must not be a duplicate of the area allocated in the self-node CPU memory area.

Note 2: CPU memory allocation cannot be performed over more than one PI/O range.

Example: An unacceptable setup example is given below:

When the word count (size) setting for area 1 is 3, "RWFF0" cannot be specified as the area 1 address (because the RW area limit would be exceeded).

- If the CPU memory area word count differs from the common memory size setting (link parameter area word count), the FL.NET module transfers data from the common memory. The amount of this data transfer is determined from the number of words specified by the CPU memory area word count setting.
- If any node is left without being set, the FL.NET module cannot transfer to the S10mini the data received from that node into the common memory.

(3) Referencing the self-node status and FA link status

The self-node status/FA link status of the FL.NET module, the FA link status of each node, and a higher layer status flag can be referenced with the setup tool named "FL.NET For Windows®."

When various items of status information on the S10mini need to be referenced, the status of participating nodes and the like can be grasped by setting the transfer area for each flag.

To set up the self-node status/FA link status transfer area for the self-node, use the self-node setup window.

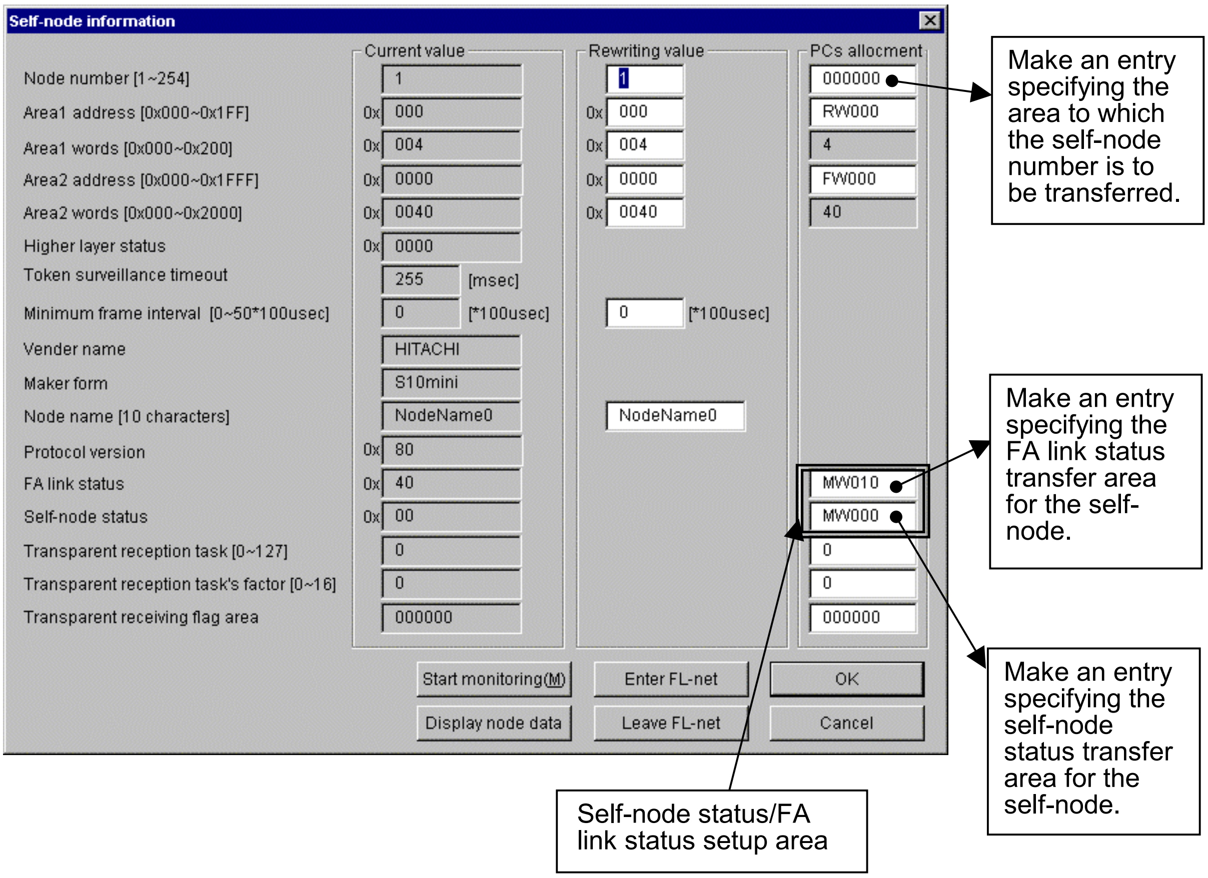


Figure 6-43 Setup Window for Setting Various Status Flag Areas for the Self-node

For the on-screen information (bit allocation) concerning the self-node status and FA link status, see "6.4.6 Using the management tables."

The table below shows the acceptable input ranges for various input items.

Input item	Description	Acceptable input range
Node number (PCs allocation)	Specify the area to which the self-node number is to be transferred (the area size is 1 word).	See “Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory.”
FA link status (PCs allocation)	Specify the area to which the FA link status of the self-node is to be transferred (the area size is 1 word).	See “Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory.”
Self-node status (PCs allocation)	Specify the area to which the self-node status is to be transferred (the area size is 1 word).	See “Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory.”

Note 1: The register/extension memory address range applicable to the self-node status area and FA link status area is the same as for the CPU memory allocation area.

Note 2: The self-node status/FA link status area setting must not be a duplicate of the CPU memory allocation area setting for any of the other participating nodes or the self-node.

- When the self-node status area is set up, the self-node status flag is transferred as 1-word data.
- When the FA link status area is set up, the FA link status flag for the self-node is transferred to the low-order byte of the specified area.
- If the area for self-node status or FA link status information is not specified, the FL.NET module cannot transfer the status information to the S10mini.

To set up the FA link status/higher layer flag area, use the setup window for the other participating nodes.

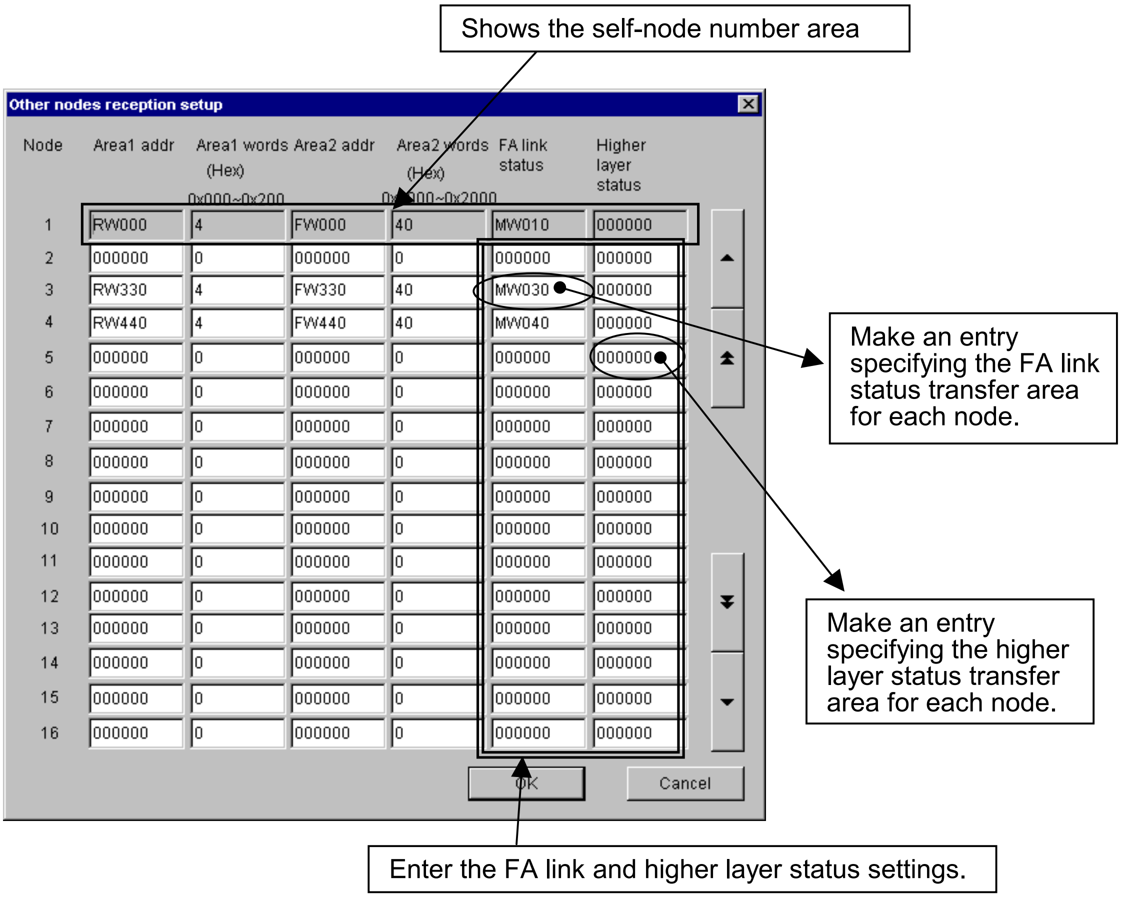


Figure 6-44 Setup Window for Setting the Status Flag Areas for the Other Participating Nodes

The table below shows the acceptable input ranges for the input items.

Input item	Description	Acceptable input range
FA link status	Specify the area to which the FA link status of the node having the node number indicated at the left end is to be transferred (the area size is 1 word).	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."
Higher layer status	Specify the area to which the higher layer status of the node having the node number indicated at the left end is to be transferred (the area size is 1 word).	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."

For the on-screen information (bit allocation) concerning the FA link status and higher layer status, see "6.4.6 Using the management tables."

- The register/extension memory address range applicable to the higher layer status area and FA link status area is the same as for the CPU memory allocation area.
- When the higher layer status area is set up, the contents of the higher layer status flag are transferred as 1-word data.
- When the FA link status area is set up, the contents of the FA link status flag for the node are transferred to the low-order byte of the specified area.
- The higher layer status/FA link status area setting must not be a duplicate of the CPU memory allocation area setting for a remote node (reception setting) or the self-node or of the setting for any other area.
- If any node is omitted from the node settings given, the FL.NET module cannot transfer the higher layer status or FA link status of that node to the S10mini.

6.4.3 Using bit data

The FL.NET module subjects the common memory area 1 data to bit conversion at the time of transmission/reception.

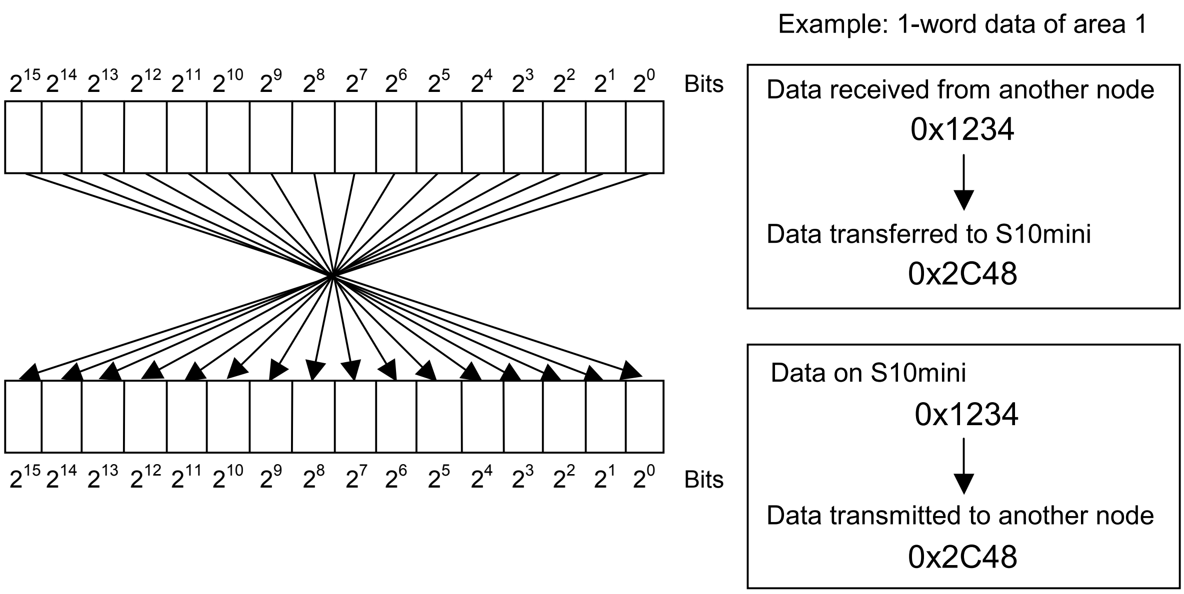


Figure 6-45 Bit Conversion of Area 1 Data

6.4.4 Using word data

The FL.NET module does not convert the common memory area 2 data at the time of transmission/reception. It transfers the data to the network without changing its arrangement on the S10mini, so the data received from the other nodes is transferred as is.

6.4.5 Using message communications

(1) Message communication (server side)
When the FL.NET module receives a message request from another node, it processes the message within itself. The user does not have to pay attention to it.
However, when the FL.NET module receives a transparent message, it notifies the user of the transparent message reception via the self-node status flag and transparent reception flag. For details, see "(4) Transparent message reception."
When a transparent message is received, use a C mode program or mathematical/logical function to fetch it from the FL.NET module.
If the FL.NET module is filled with transparent messages, it may not be able to process any more request messages.

(2) Message communication (client side)
When requesting a message for another node (or receiving a transparent message), it is necessary to issue a request to the FL.NET module from a C mode program or mathematical/logical function.
The FL.NET module offers a C mode handler and mathematical/logical function as a means of issuing a message request.
The procedures for using a C mode program or ladder program to issue a message request are explained below:

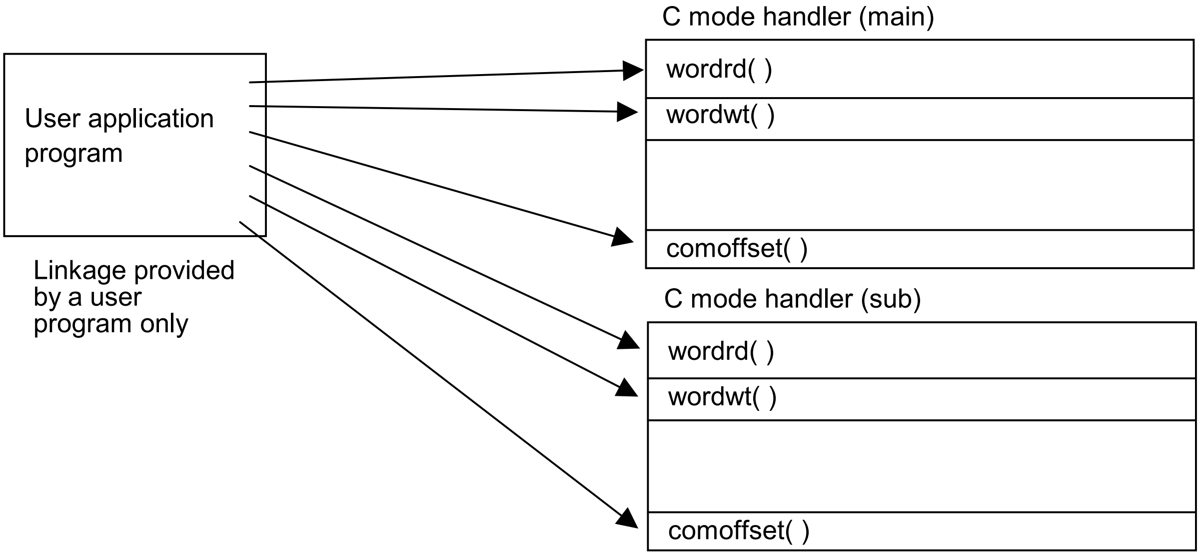
Note: For message communication, an environment for permitting a user program to reference the self-node status flag is required.
Set up a self-node status flag area from the self-node setup window of the setup tool named "FL.NET For Windows®." For detailed procedure descriptions, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

(3) C mode handler or mathematical/logical function

(a) Message transmission request by C mode handler

A C mode handler is called as a C function. It issues a message request to the FL.NET module to exchange data in place of a user program. Various C mode handlers are available for use with all message types.

A C mode handler must be called by specifying its entry address. A user program cannot be created (linked) in such a manner that a C mode handler is contained in it.



It should also be pointed out in this connection that the C mode handler for the FL.NET module merely issues a request to the FL.NET module and then terminates its whole process immediately.

When the return code fed from the C mode handler is -1 (0xFFFFFFFF), it means that a parameter abnormality exists or that the handler is busy processing another message. Check the data at an error code storage address specified at C mode handler startup. When the return code is 0, the C mode handler indicates that a request has been processed normally.

After receipt of a request, the FL.NET module turns ON (sets) the “user request processing in progress” bit in the self-node status flag, and turns it OFF (resets it) at the end of the processing.

To verify the end of message processing after issuance, check the self-node status flag. The status indicates the end of the processing.

After a request is issued by the C mode handler, do not make another message request until the “user request processing in progress” bit is reset. If any message request is issued before the bit is reset, it is not processed because another message is being processed (the “message processing already in progress” error code is set at an error code storage address). After the “user request processing in progress” bit is reset, enter the status prevailing after the end of the processing at an error code storage address specified at C mode handler startup.
To check whether the requested processing is ended normally, note the code that is set at the error code storage address.

Table 6-8 C Mode Handler List

Name	Subroutine call address		Functionality description
	Main	Sub	
wordrd( )	/D74112	/DF4112	Issues a word block read request.
wordwt( )	/D74118	/DF4118	Issues a word block write request.
parard( )	/D7411E	/DF411E	Issues a network parameter read request.
parawt( )	/D74124	/DF4124	Issues a network parameter write request.
reqstop( )	/D7412A	/DF412A	Issues a stop request.
reqrun( )	/D74130	/DF4130	Issues a run request.
logrd( )	/D7413C	/DF413C	Issues a log communication read request.
logclr( )	/D74142	/DF4142	Issues a log communication clear request.
mesret( )	/D74148	/DF4148	Issues a message return request.
reqmacro( )	/D74160	/DF4160	Issues a specified-task control request (Hitachi’s unique transparent type of support).
toukaread( )	/D74178	/DF4178	Issues a transparent message reception request (Hitachi’s unique transparent type of support).
toukasend( )	/D7417E	/DF417E	Issues a transparent message transmission request (Hitachi’s unique transparent type of support).
comoffset( )	/D74184	/DF4184	Issues a common memory offset feature request.

Note: For bit allocation for the self-node status flag, see “6.4.6 Using the management tables.”

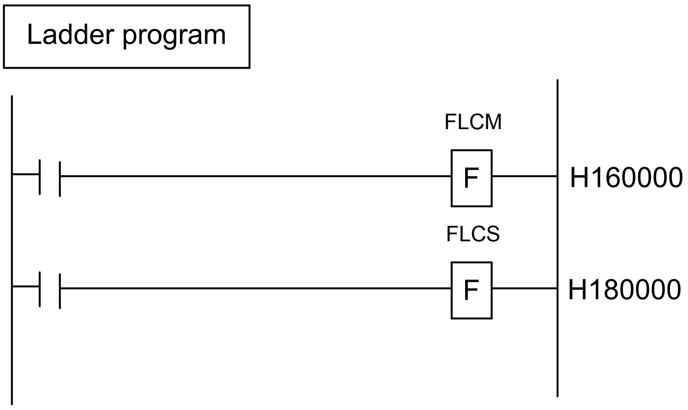
Do not mix to use C mode handler and mathematical/logical function.
Please be sure to unify either C mode handler or mathematical/logical function at FL.NET module unit.

(b) Message transmission request by mathematical/logical function

A mathematical/logical function is called from a ladder program to issue a message request to the FL.NET module for data exchange purposes. The mathematical/logical function for the main module/submodule is available. The parameter specified at mathematical/ logical function startup dictates the message processing to be requested.

When a ladder program uses the mathematical/logical function, it is necessary to register the main module/submodule mathematical/logical function beforehand with “FL.NET For Windows®.”

For the procedure for registering the FL.NET module mathematical/logical function, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139).”



As is the case with the C mode handler, the mathematical/logical function merely issues a request to the FL.NET module and then terminates the processing.

After receipt of a request, the FL.NET module sets the “user request processing in progress” bit in the self-node status flag, and resets it at the end of the processing.

To verify the end of the message processing after issuance, check the self-node status flag. The status indicates the end of the processing.

After a request is issued by the mathematical/logical function, do not make another message request until the “user request processing in progress” bit is resets. If any message request is issued before the bit is reset, it is not processed because another message is being processed (the “message processing already in progress” error code is set at an error code storage address).

After the “user request processing in progress” bit is reset, enter the status prevailing after the end of the processing at an error code storage address specified at mathematical/logical function startup.

To check whether the requested processing is ended normally, note the code that is set at the error code storage address.

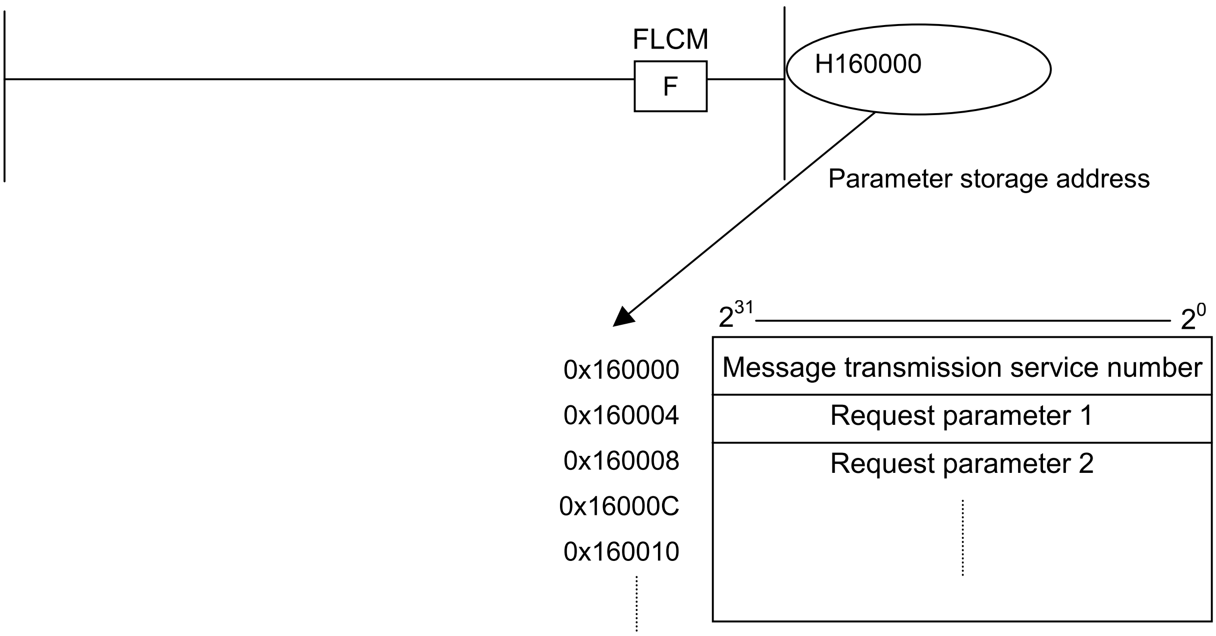
Table 6-9 Mathematical/Logical Function List

Name		Mathematical/logical function registration address		Functionality description
Main	Sub	Main	Sub	
FLCM	FLCS	/D74100	/DF4100	Issues various message transmission requests.

<Process request to FL.NET module by mathematical/logical function>

Create a ladder program with the mathematical/logical function for the FL.NET module. Write a request parameter at an address specified for the mathematical/logical function and then execute the mathematical/logical function.

When making a message transmission request from the mathematical/logical function, you must use a message transmission service number to specify the type of the message to request. For details on various parameters, see "(5) Parameters for various message requests."



Whenever specifying the parameters for use with a mathematical/logical function, use a long word length.

Also, selectively use an appropriate mathematical/logical function depending on the employed module setup (When issuing a request to a main module, use the main module mathematical/logical function FLCM. To issue a request to a submodule, use the submodule mathematical/logical function FLCS.)

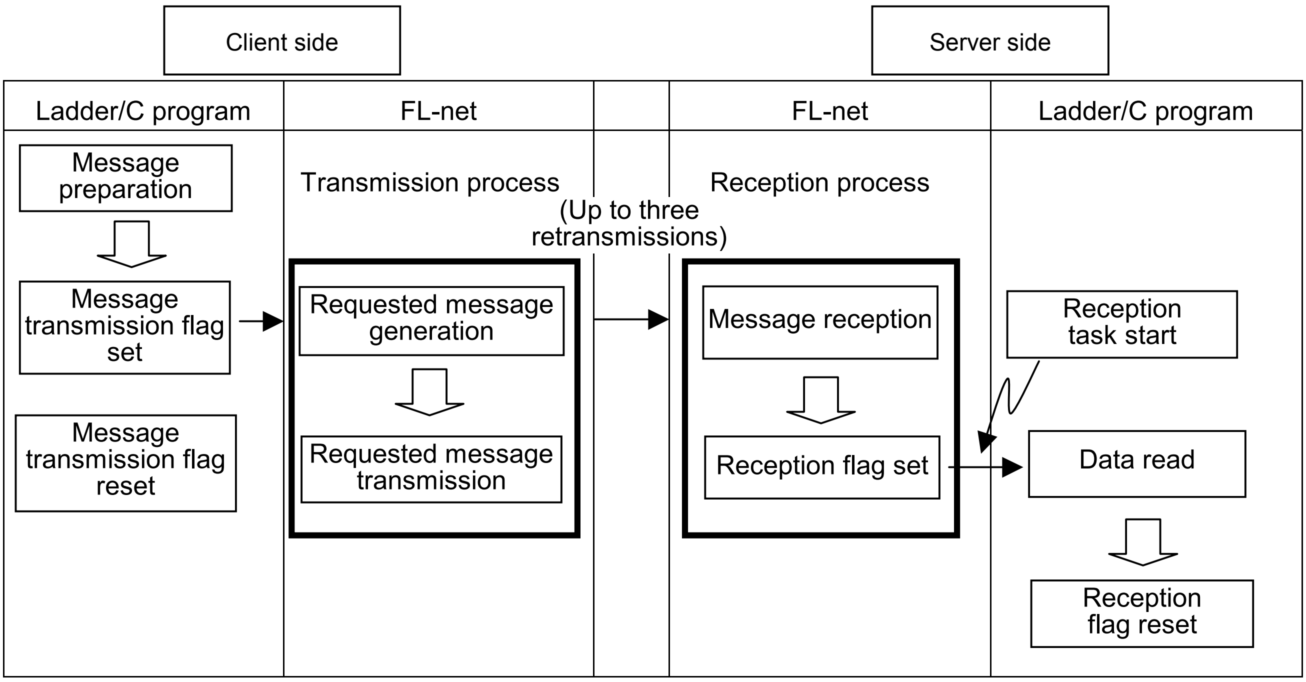
Note 1: For bit allocation for the self-node status flag, see "6.4.6 Using the management tables."

Note 2: When making a request from a mathematical/logical function, you must use the setup tool named "FL.NET For Windows®" to register an FL.NET mathematical/logical function on the S10mini. For details on the setup procedure, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

When issuing a request to the FL.NET module, do not simultaneously use the C mode handler and mathematical/logical function.

(4) Transparent message reception

The figure below outlines the operation performed for transparent messages.



The FL.NET module has special functions concerning the reception of transparent messages.

<Transparent reception task>

When a transparent message is received in situations where a transparent reception task is registered in the FL.NET module, task queuing takes place with the start factor (transparent reception task's factor) that is set for the task number of the registered task.
If the registered task is not released, however, it does not start.

<Transparent receiving flag>

When a transparent receiving flag is set for the FL.NET module, a 16-word area beginning with the associated selected address is used as a transparent-type reception flag area.
When the FL.NET module receives a transparent message, the transparent receiving flag area indicates the node from which the message has been transmitted.
A total of eight cases of transparent message reception buffer are readied.
When a transparent message is received, it is first stored in the reception buffer. When it is stored, its transmission source node number is checked, and the bit corresponding to the transmission source node number for the transparent receiving flag area is set.
After the transparent message is fetched by the user, a check is conducted to determine whether messages having the same transmission source node number as for the message delivered to the user remain within the transparent message reception buffer.
When there are no more messages having the same transmission source node number, the bit corresponding to the transmission source node number for the transparent receiving flag area is reset.
However, when a received transparent message is for a TCD that is related to a function (preselected task control and preselected subroutine control) uniquely supported by the FL.NET module, it is not handled as a transparent message.
A transparent receiving task and transparent receiving flag can be set up with the setup tool named "FL.NET For Windows®." For details on the procedure, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

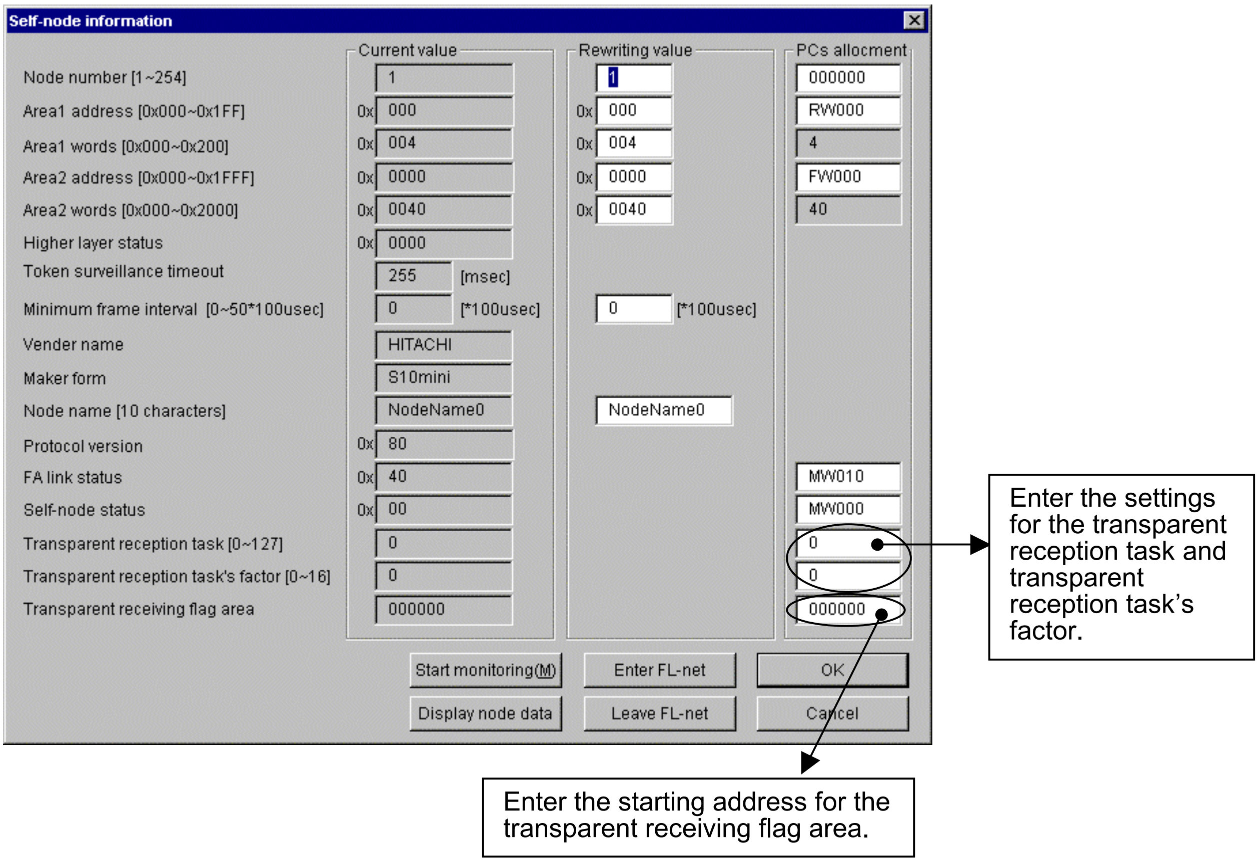


Figure 6-46 Transparent Receiving Flag Area and Transparent Reception Task Setup Window

The table below shows the acceptable input ranges for the setup items.

Input item	Description	Acceptable setting range
Transparent reception task	Task number of the task that is to start upon receipt of a transparent message.	The acceptable setting ranges from 0 to 127. However, you enter a setting between 2 and 100.
Transparent reception task's factor	Factor for starting a transparent reception task.	0 to 16
Transparent receiving flag area	This area reports a transmission source node number when a transparent message is received (the area size is 16 words).	See "Table 6-7 Address Ranges for Common Memory Area Allocation in CPU Memory."

Note: The symbols and extension memory addresses selectable for a transparent receiving flag area are the same as for a CPU memory area. Also, note that a transparent receiving flag area setting must not be a duplicate of any of those for the other set areas.

Table 6-10 shows the bit allocation for the transparent receiving flag area.

Table 6-10 Transparent Receiving Flag Allocation

Address	Bit No.															
	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
0x0000		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0002	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0004	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
0x0006	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
0x0008	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
0x000A	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
0x000C	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
0x000E	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
0x0010	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
0x0012	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
0x0014	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
0x0016	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
0x0018	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
0x001A	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
0x001C	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
0x001E	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	

The numerical values in the above table represent transmission source node numbers for transparent messages. Note that a blank bit location is not used.

(Indication example)

The transparent receiving flag area is set to MW000.

When the self-node receives a transparent message from node No. 17, the MW011 (the node 17 bit) is turned ON (set).

PI/O	Bit data
MW000	0x0000
MW010	0x4000
:	:
MW0F0	0x0000

(5) Parameters for various message requests
The rest of this section describes the parameters that the C mode handler and mathematical/logical function use to issue various message requests.

Word block read request: wordrd( )

[Linking procedure]

C language	
Main	Sub

<pre>Struct wordr_p {
              long node ;
     unsigned short *Erradr ;
     unsigned short *Setadr ;
              long Readadr ;
              long Readsz ;
} ;
            ~
long (*wordrd)( ) ;
long rtn ;
struct wordr_p *padr ;
            ~
wordrd = (long(*)( ))0xD74112 ;
            ~
rtn = (*wordrd)(padr) ;</pre>
	<pre>Struct wordr_p {
	              long node ;
	     unsigned short *Erradr ;
	     unsigned short *Setadr ;
	              long Readadr ;
	              long Readsz ;
	} ;
	            ~
	long (*wordrd)( ) ;
	long rtn ;
	struct wordr_p *padr ;
	            ~
	wordrd = (long(*)( ))0xDF4112 ;
	            ~
	rtn = (*wordrd)(padr) ;</pre>

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (3)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Read data storage address
+0x0010	Virtual address
+0x0014	Virtual size

[Parameters]

```
padr                                    : Input parameter storage starting address
Struct wordr_p {
            long node ;                 : Remote node number (1 to 254)
    unsigned short *Erradr ;            : Error code storage address
    unsigned short *Setadr ;            : Read data storage address
            long Readadr ;              : Virtual address
            long Readsz ;               : Virtual size (1 to 512 words)
} ;
```

Parameter	Input range
Message transmission service number	3 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Read data storage address	
Virtual address	0 to 0xFFFFFFFF
Virtual size	1 to 512 (variable in 1-word units)

[Function]

- A word block read request using the specified virtual address and virtual size will be issued to a remote node. When a response code is received, the data is transferred to a read data storage address.
- The virtual address varies from one FL.NET module to another. For the virtual address specifications for the FL.NET module, see "Table 3-4 Virtual Address Space and Physical Memory."

> The server feature of the FL.NET module returns an abnormal response if, on reception of a word block read request, the virtual size is found exceeding 512 words . The error code used in such an event is 0xFFFFFFFF.

Word block write request: wordwt( )

[Linking procedure]

<table>
<tr><th colspan="2">C language</th></tr>
<tr><th>Main</th><th>Sub</th></tr>
<tr><td>

```
Struct wordw_p {
            long node ;
    unsigned short *Erradr ;
    unsigned short *Setadr ;
            long writeadr ;
            long writesz ;
}
          ≀
long  (*wordwt)( ) ;
long rtn ;
struct wordw_p *padr ;
          ≀
wordwt = (long(*)( ))0xD74118 ;
          ≀
rtn = (*wordwt)(padr) ;
```

</td><td>

```
Struct wordw_p {
            long node ;
    unsigned short *Erradr ;
    unsigned short *Setadr ;
            long writeadr ;
            long writesz ;
}
          ≀
long  (*wordwt)( ) ;
long rtn ;
struct wordw_p *padr ;
          ≀
wordwt = (long(*)( ))0xDF4118 ;
          ≀
rtn = (*wordwt)(padr) ;
```

</td></tr>
</table>

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (4)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Write data storage address
+0x0010	Virtual address
+0x0014	Virtual size

[Parameters]

```
padr                                    : Input parameter storage starting address
Struct wordrw_p {
            long node ;                 : Remote node number (1 to 254)
    unsigned short *Erradr ;            : Error code storage address
    unsigned short *Setadr ;            : Write data storage address
            long Writeadr ;             : Virtual address
            long Writesz ;              : Virtual size (1 to 512 words)
} ;
```

Parameter	Input range
Message transmission service number	4 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Read data storage address	
Virtual address	0 to 0xFFFFFFFF
Virtual size	1 to 512 (variable in 1-word units)

[Function]

- A word block write request using the specified virtual address, virtual size, and data at the write data storage address will be issued to a remote node.
- The virtual address varies from one FL.NET module to another. For the virtual address specifications for the FL.NET module, see “Table 3-4 Virtual Address Space and Physical Memory.”

> The server feature of the FL.NET module returns an abnormal response if, on reception of a word block write request, the virtual size is found exceeding 512 words or conflicting with the data size . The error code used in such an event is 0xFFFFFFFF.

Network parameter read request: parard( )

[Linking procedure]

C language	
Main	Sub
Struct parar_p { long node ; unsigned short *Erradr ; unsigned char *Setadr ; } ; ∫ long (*parard)( ) ; long rtn ; struct parar_p *padr ; ∫ parard = (long(*)( ))0xD7411E ; ∫ rtn = (*parard)(padr) ;	Struct parar_p { long node ; unsigned short *Erradr ; unsigned char *Setadr ; } ; ∫ long (*parard)( ) ; long rtn ; struct parar_p *padr ; ∫ parard = (long(*)( ))0xDF411E ; ∫ rtn = (*parard)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (5)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Read parameter data storage address

[Parameters]

```
padr                              : Input parameter storage starting address
Struct parar_p {
            long node ;           : Remote node number (1 to 254)
    unsigned short *Erradr ;      : Error code storage address
    unsigned char *Setadr ;       : Read parameter data storage address
} ;
```

Parameter	Input range
Message transmission service number	5 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Read parameter data storage address	

[Function]

A network parameter read request will be issued to a remote node.

When a response code is received, the network parameter information about the specified node transfers to the read parameter data storage address.

The data transfers to the read parameter data storage address in the format shown below.

The node number does not transfer to the S10mini.

Read parameter data storage address → 0x0000

Address	2^{15}	2^{0}
0x0000	Node name (equipment name)	
0x000A	Vendor name	
0x0014	Maker format	
0x001E	Area 1 starting address	
0x0020	Area 1 size	
0x0022	Area 2 starting address	
0x0024	Area 2 size	
0x0026	Reserve	Token surveillance timeout
0x0028	Reserve	Minimum frame interval
0x002A	Reserve	Link status
0x002C	Reserve	Protocol version
0x002E	Higher layer status	
0x0030	Refresh cycle time (RCT) setting	
0x0032	Refresh cycle measurement value (current)	
0x0034	Refresh cycle measurement value (maximum)	
0x0036	Refresh cycle measurement value (minimum)	

Network parameter write request: parawt( )

[Linking procedure]

C language	
Main	Sub
Struct paraw_p { long node ; unsigned short *Erradr ; unsigned char *Dataadr ; } ; ∫ long (*parawt)( ) ; long rtn ; struct paraw_p *padr ; ∫ parawt = (long(*)( ))0xD74124 ; ∫ rtn = (*parawt)(padr) ;	Struct paraw_p { long node ; unsigned short *Erradr ; unsigned char *Dataadr ; } ; ∫ long (*parawt)( ) ; long rtn ; struct paraw_p *padr ; ∫ parawt = (long(*)( ))0xDF4124 ; ∫ rtn = (*parawt)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (6)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Write parameter data storage address

[Parameters]

```
padr                                  : Input parameter storage starting address
Struct paraw_p {
               long node ;            : Remote node number (1 to 254)
     unsigned short *Erradr ;         : Error code storage address
     unsigned  char *Dataadr ;        : Write parameter data storage address
} ;
```

Parameter	Input range
Message transmission service number	6 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Write data storage address	

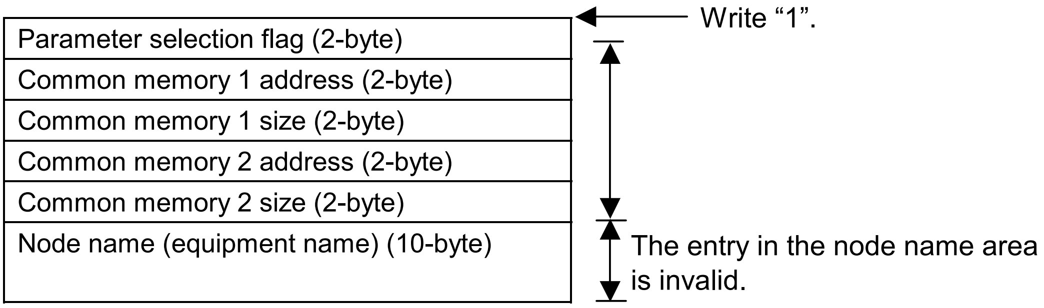
[Function]

A network parameter write request using the parameters at the write parameter data storage address will be issued to a remote node.

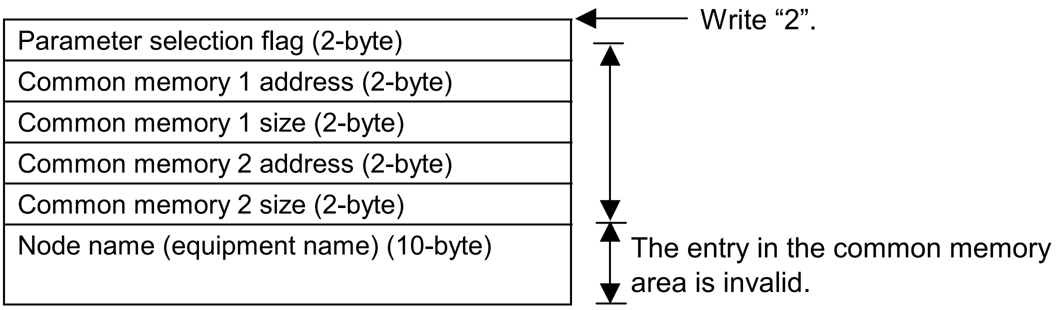
[Write parameter data structure] (contiguous areas)

Create the data as shown below in the area specified by the write parameter data storage address.

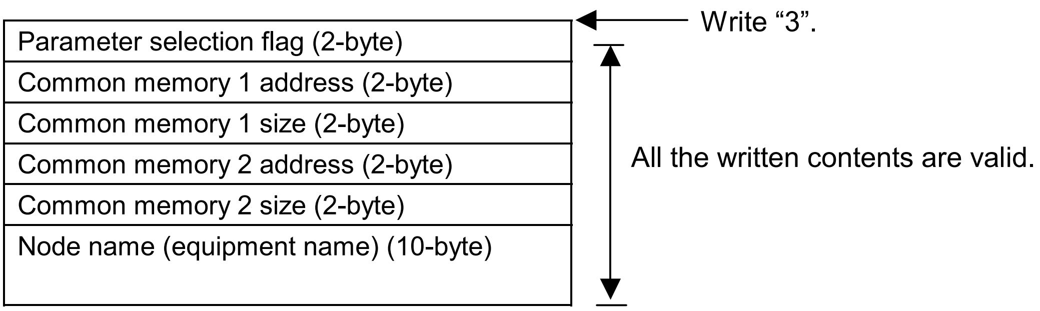
1. When rewriting the common memory address and common memory size only



2. When rewriting the node name (equipment name) only



3. When rewriting the common memory address, common memory size, and node name (equipment name)



- When rewriting the network parameters, exercise due care because common memory duplication may be caused. If such duplication occurs on the S10mini, you have to set up the FL.NET module again.
- The server feature of the FL.NET module returns an abnormal response if, upon reception of a network parameter write request, the parameter selection flag is found to be not between 1 and 3. The error code used in such an event is 0xFFFFFFFF.

Stop request: reqstop( )

[Linking procedure]

C language

Main	Sub
Struct reqs_p { long node ; unsigned short *Erradr ; } ; ⌇ long (*ReqStop)( ) ; long rtn ; struct reqs_p *padr ; ⌇ ReqStop = (long(*)( ))0xD7412A ; ⌇ rtn = (*ReqStop)(padr) ;	Struct reqs_p { long node ; unsigned short *Erradr ; } ; ⌇ long (*ReqStop)( ) ; long rtn ; struct reqs_p *padr ; ⌇ ReqStop = (long(*)( ))0xDF412A ; ⌇ rtn = (*ReqStop)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (7)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address

[Parameters]

```
padr                                  : Input parameter storage starting address
Struct reqs_p {
              long node ;             : Remote node number (1 to 254)
     unsigned short *Erradr ;         : Error code storage address
} ;
```

Parameter	Input range
Message transmission service number	7 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.

[Function]

A stop request will be issued to a remote node.

> The FL.NET module does not support the server feature for a stop directive. If a stop request is issued to the FL.NET module, it returns a response to indicate that it does not support the stop directive.

Run request: reqrun( )

[Linking procedure]

<table>
<tr><th colspan="2">C language</th></tr>
<tr><th>Main</th><th>Sub</th></tr>
<tr><td><pre>Struct reqr_p {
               long node ;
     unsigned short *Erradr ;
} ;
          ⸾
long (*ReqRun)( ) ;
long rtn ;
struct reqr_p *padr ;
          ⸾
ReqRun = (long(*)( ))0xD74130 ;
          ⸾
rtn = (*ReqRun)(padr) ;</pre></td><td><pre>Struct reqr_p {
               long node ;
     unsigned short *Erradr ;
} ;
          ⸾
long (*ReqRun)( ) ;
long rtn ;
struct reqr_p *padr ;
          ⸾
ReqRun = (long(*)( ))0xDF4130 ;
          ⸾
rtn = (*ReqRun)(padr) ;</pre></td></tr>
</table>

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (8)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address

[Parameters]

```
padr                                  : Input parameter storage starting address
Struct reqs_p {
               long node ;            : Remote node number (1 to 254)
     unsigned short *Erradr ;         : Error code storage address
} ;
```

Parameter	Input range
Message transmission service number	8 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.

[Function]

A run request will be issued to a remote node.

> The FL.NET module does not support the server feature for a run directive. If a run request is issued to the FL.NET module, it returns a response to indicate that it does not support the run directive.

Profile read

The FL.NET module does not support the client feature of profile read for another node.
However, it responds to a request from another node.
The FL.NET module's response to a profile read request from another node is indicated below:

Identifier	Length
0x30	0x816A

Identifier	Length
0x30	0x68

Identifier	Length	Description
0x13	0x06	"COMVER"
0x02	0x01	"0x01"
0x13	0x02	"ID"
0x13	0x07	"SYSPARA"
0x13	0x03	"REV"
0x02	0x01	"0x01"
0x13	0x07	"REVDATA"
0x30	0x0A	
0x02	0x02	"0x07CF (1999) "
0x02	0x01	"0x0B(11) "
0x02	0x01	"0x0F(15) "
0x13	0x0A	"DVCATEGORY"
0x13	0x03	"PLC"
0x13	0x06	"VENDOR"
0x13	0x07	"HITACHI"
0x13	0x07	"DVMODEL"
0x13	0x06	"LQE000"

Communication log data read request: logrd( )

[Linking procedure]

C language	
Main	Sub
Struct logr_p { long node ; unsigned short *Erradr ; unsigned char *logadr ; } ; ⁝ long (*Logrd)( ) ; long rtn ; struct logr_p *padr ; ⁝ Logrd = (long(*)( ))0xD7413C ; ⁝ rtn = (*Logrd)(padr) ;	Struct logr_p { long node ; unsigned short *Erradr ; unsigned char *logadr ; } ; ⁝ long (*Logrd)( ); long rtn ; struct logr_p *padr ; ⁝ Logrd = (long(*)( ))0xDF413C ; ⁝ rtn = (*Logrd)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (10)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Log data storage address

[Parameters]

```
padr                                    : Input parameter storage starting address
Struct logr_p {
            long node ;                 : Remote node number (1 to 254)
    unsigned short *Erradr ;            : Error code storage address
    unsigned char  *logadr ;            : Log data storage address
} ;
```

Parameter	Input range
Message transmission service number	10 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Log data storage address	

[Function]

- A communication log data read request will be issued to a remote node. When a response code returns, the communication log data within the response code transfers to the log data storage address.
- The communication log data to be transferred consists of 512 bytes. Each data set consists of 4-byte data. For details on communication log data, see "6.4.8 Using the communication log."

Communication log data clear request: logclr( )

[Linking procedure]

C language	
Main	Sub
Struct logclr_p { long node ; unsigned short *Erradr ; } ; ⁓ long (*Logclr)( ) ; long rtn ; struct logclr_p *padr ; ⁓ Logclr = (long(*)( )) 0xD74142 ; ⁓ rtn = (*Logclr)(padr) ;	Struct logclr_p { long node ; unsigned short *Erradr ; } ; ⁓ long (*Logclr)( ) ; long rtn ; struct logclr_p *padr ; ⁓ Logclr = (long(*)( ))0xDF4142 ; ⁓ rtn = (*Logclr)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (11)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address

[Parameters]

```
padr                                  : Input parameter storage starting address
Struct logclr_p {
               long node ;            : Remote node number (1 to 255)
     unsigned short *Erradr ;         : Error code storage address
} ;
```

Parameter	Input range
Message transmission service number	11 (only when a mathematical/logical function is used)
Remote node number	1 to 255
Error code storage address	Specify a real address (even-numbered address) on the S10mini.

[Function]

- A communication log data clear request will be issued to a remote node.
- When the remote node number is set to 255, a 1-to-N transmission occurs.

Message return request: mesret( )

[Linking procedure]

<table>
<tr><th colspan="2">C language</th></tr>
<tr><th>Main</th><th>Sub</th></tr>
<tr><td><pre>Struct mesreq_p {
               long node ;
     unsigned short *Erradr ;
     unsigned char *SendData ;
               long Sendsiz ;
     unsigned char *Recvadr ;
} ;
          ⁝
long (*Mesret)( ) ;
long rtn ;
struct mesreq_p *padr ;
          ⁝
Mesret = (long(*)( ))0xD74148 ;
          ⁝
rtn = (*Mesret)(padr) ;</pre></td><td><pre>Struct mesreq_p {
               long node ;
     unsigned short *Erradr ;
     unsigned char *SendData ;
               long Sendsiz ;
     unsigned char *Recvadr ;
} ;
          ⁝
long (*Mesret)( ) ;
long rtn ;
struct mesreq_p *padr ;
          ⁝
Mesret = (long(*)( ))0xDF4148 ;
          ⁝
rtn = (*Mesret)(padr) ;</pre></td></tr>
</table>

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (12)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Message data storage starting address
+0x0010	Message data size
+0x0014	Return message storage address

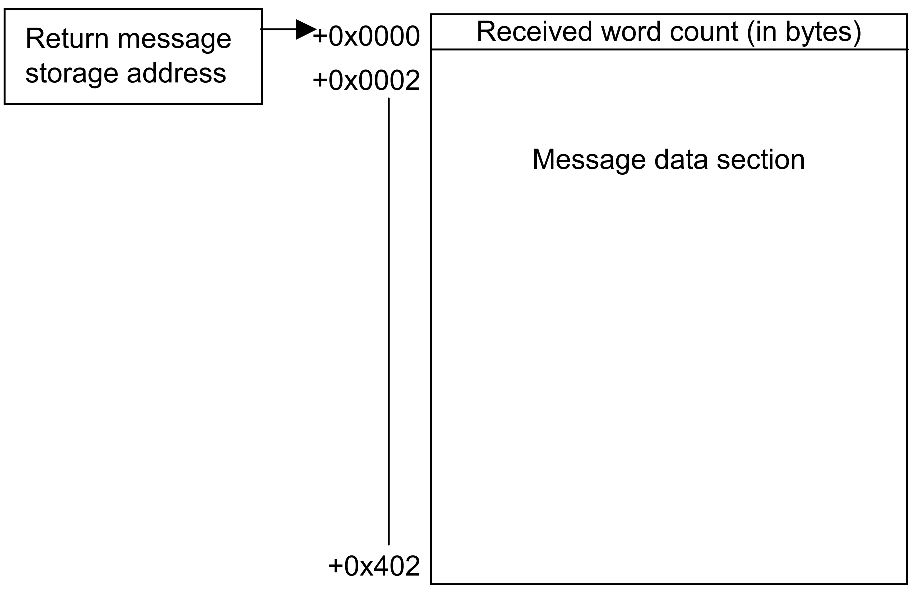
[Parameters]

```
padr                                    : Input parameter storage starting address
Struct mesreq_p {
               long node ;              : Remote node number (1 to 254)
     unsigned short *Erradr ;           : Error code storage address
     unsigned char  *SendData ;         : Message data storage starting address
               long Sendsiz ;           : Message data size (1 to 1,024)
     unsigned char  *Recvadr ;          : Return message storage address
} ;
```

Parameter	Input range
Message transmission service number	12 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Message data storage starting address	
Message data size	1 to 1,024
Return message storage address	Specify a real address (even-numbered address) on the S10mini.

[Function]

- A message return request using the data at the message data storage starting address will be issued to a remote node. When a response message returns, the message data within the response code transfers to the return message storage address.
- The data transfers to the return message storage address in the following format:



Specified task control request (Hitachi's unique transparent type of support): reqmacro( )

[Operation performed]

Exercises control (abort/release/queue) over a user-specified task number.

[Linking procedure]

<table>
<tr><th colspan="2">C language</th></tr>
<tr><th>Main</th><th>Sub</th></tr>
<tr><td><pre>Struct Reqmacro_p {
               long node
     unsigned short *Erradr ;
      unsigned long *Retadr ;
               long ParaCnt ;
      unsigned long Para[3] ;
} ;
          ⸾
long (*ReqMacro)( ) ;
long rtn ;
struct Reqmacro_p *padr ;
          ⸾
ReqMacro = (long(*)( ))0xD74160 ;
          ⸾
rtn = (*ReqMacro)(padr) ;</pre></td><td><pre>Struct Reqmacro_p {
               long node
     unsigned short *Erradr ;
      unsigned long *Retadr ;
               long ParaCnt ;
      unsigned long Para[3] ;
} ;
          ⸾
long (*ReqMacro)( ) ;
long rtn ;
struct Reqmacro_p *padr ;
          ⸾
ReqMacro = (long(*)( ))0xDF4160 ;
          ⸾
rtn = (*ReqMacro)(padr) ;</pre></td></tr>
</table>

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (13)
+0x0004	Remote node number (1 to 254)
+0x0008	Error code storage address
+0x000C	CPMS macro execution result storage address
+0x0010	CPMS macro parameter count
+0x0014	CPMS macro parameter 1
+0x0018	CPMS macro parameter 2
+0x001C	CPMS macro parameter 3

[Parameters]

```
padr                                  : Input parameter storage starting address
Struct Reqmacro_p {
          long node ;                 : Remote node number (1 to 254)
   unsigned short *Erradr ;           : Error code storage address
   unsigned long *Retdadr ;           : CPMS macro execution result storage address
          long ParaCnt ;              : CPMS macro parameter count
   unsigned long Para[3] ;            : CPMS macro parameter
} ;
```

Parameter	Input range
Message transmission service number	13 (only when a mathematical/logical function is used)
Remote node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini
CPMS macro execution result storage address	
CPMS macro parameter count	See [Function]
CPMS macro parameter	

[Function]

A task abort, release, or queue execution request can be issued to a remote node.

When a response message is received, the task control execution result (log value) is written at the CPMS macro execution result storage address.

Specify the parameters as indicated below:

<To execute a task abort>

CPMS macro parameter count = 2

CPMS macro parameter [0] = 1

CPMS macro parameter [1] = task number (1 to 128)

<To execute a task release>

CPMS macro parameter count = 2

CPMS macro parameter [0] = 2

CPMS macro parameter [1] = task number (1 to 128)

<To execute a task queue>

CPMS macro parameter count = 3

CPMS macro parameter [0] = 3

CPMS macro parameter [1] = task number (1 to 128)

CPMS macro parameter [2] = task number (0 to 16)

Exercise due care when using the above transparent message request. If the task number is erroneously specified, the CPU of the remote node (S10mini) may become inoperative.

Transparent message reception request (Hitachi's unique transparent type of support): toukaread( )

[Linking procedure]

C language	
Main	Sub
Struct ToukaRead_p { long node ; unsigned short *Erradr ; unsigned char *dataadr ; unsigned long datasiz ; } ; ⟆ long (*ToukaRead)( ) ; long rtn ; struct ToukaRead_p *padr ; ⟆ ToukaRead = (long(*)( ))0XD74178 ; ⟆ rtn = (*ToukaRead)(padr) ;	Struct ToukaRead_p { long node ; unsigned short *Erradr ; unsigned char *dataadr ; unsigned long datasiz ; } ; ⟆ long (*ToukaRead)( ) ; long rtn ; struct ToukaRead_p *padr ; ⟆ ToukaRead = (long(*)( ))0XDF4178 ; ⟆ rtn = (*ToukaRead)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (17)
+0x0004	Reception destination node number (0 to 254)
+0x0008	Error code storage address
+0x000C	Reception data storage address
+0x0010	Data word count

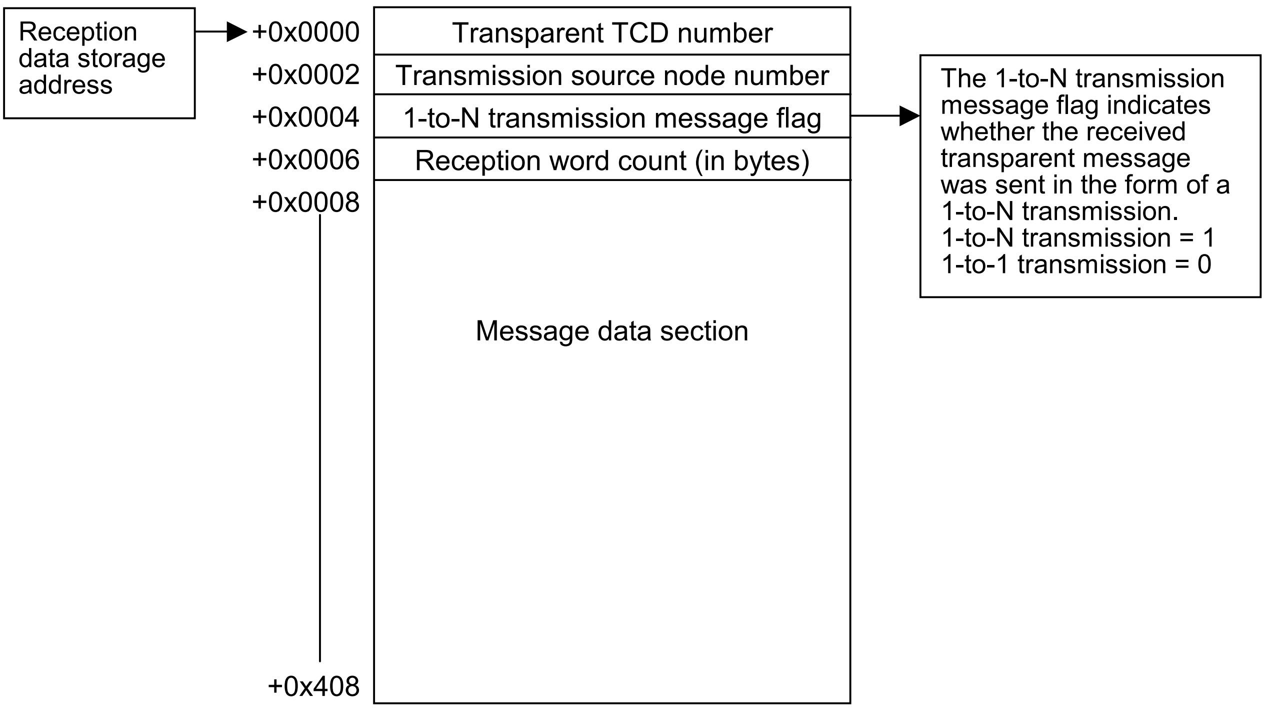
[Parameters]

```
padr                              : Input parameter storage starting address
Struct ToukaRead_p {
               long node ;        : Reception destination node number (0 to 254)
     unsigned short *Erradr ;     : Error code storage address
      unsigned char *dataadr ;    : Reception data storage address
      unsigned long datasiz ;     : Data word count (0 to 1,024)
} ;
```

Parameter	Input range
Message transmission service number	17 (only when a mathematical/logical function is used)
Reception destination node number	0 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Reception data storage address	
Data word count	0 to 1,024 (variable in 1-byte units)

[Function]

- Reception word count will be transferred to the area which is specified received transparent message by reception data storage address.
- If there is no transparent message from a specified reception destination node number, a parameter error occurs.
- When the specified reception destination node number is 0, the first-retrieved data within a transparent message queue transfers to the reception data storage address without regard to the transmission source node number (this transfer does not take place in order of reception). If no transparent message is received, a parameter error occurs.
- When the user-specified data word count is lower than the actually received word count, only as many words of message data as specified by the user will be transferred.
- A transparent message transfers to the reception data storage address in the following format:



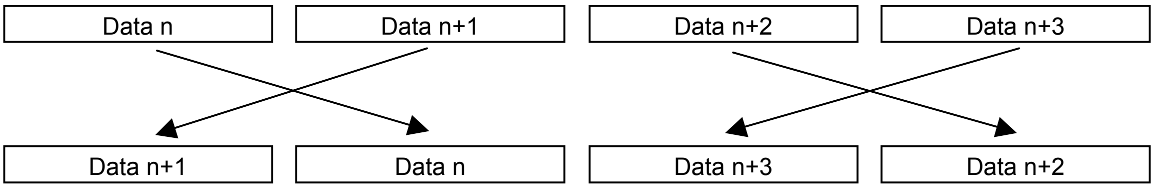
Note: When the user-specified data word count is expressed by an odd number of bytes, the 1-byte data “0x00” is added to the finally-transferred received data.

- FL.NET module’s unique feature

  When the received transparent message TCD number is between 0 and 999, the reception message data is handled as word data in little-endian format. However, if the reception message data word count is expressed by an odd number of bytes, the data does not convert normally into little-endian format. If the specified TCD number is other than the above, the received data is transferred as is.

Data conversion effected when a transparent message having a TCD number between 0 and 999 is received

Example: When 4-byte data is received



Byte-swapped on an individual word basis and the received.

Transparent message transmission request (Hitachi's unique transparent type of support): toukasend( )

[Linking procedure]

C language	
Main	Sub
Struct ToukaSend_p { long node ; unsigned short *Erradr ; unsigned char *dataadr ; unsigned long datasiz ; unsigned long TcdNo ; } ; ⁓ long (*ToukaSend)( ) ; long rtn ; struct ToukaSend_p *padr ; ⁓ ToukaSend = (long(*)( ))0XD7417E ; ⁓ rtn = (*ToukaSend)(padr) ;	Struct ToukaSend_p { long node ; unsigned short *Erradr ; unsigned char *dataadr ; unsigned long datasiz ; unsigned long TcdNo ; } ; ⁓ long (*ToukaSend)( ) ; long rtn ; struct ToukaSend_p *padr ; ⁓ ToukaSend = (long(*)( ))0xDF417E ; ⁓ rtn = (*ToukaSend)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (18)
+0x0004	Transmission destination node number (1 to 255)
+0x0008	Error code storage address
+0x000C	Transmission data storage address
+0x0010	Data word count (0 to 1,024)
+0x0014	Transmission message TCD (0 to 59999)

[Parameters]

```
padr                                : Input parameter storage starting address
Struct ToukaSend_p {
              long node ;           : Transmission destination node number (1 to 255)
     unsigned short *Erradr ;       : Error code storage address
     unsigned  char *dataadr ;      : Transmission data storage address
     unsigned  long datasiz ;       : Data word count (0 to 1,024)
     unsigned  long TcdNo ;         : Transmission message TCD (0 to 59999)
} ;
```

<table>
<tr><th>Parameter</th><th>Input range</th></tr>
<tr><td>Message transmission service number</td><td>18 (only when a mathematical/logical function is used)</td></tr>
<tr><td>Transmission destination node number</td><td>1 to 255</td></tr>
<tr><td>Error code storage address</td><td rowspan="2">Specify a real address (even-numbered address) on the S10mini.</td></tr>
<tr><td>Transmission data storage address</td></tr>
<tr><td>Data word count</td><td>0 to 1,024 (specified in bytes)</td></tr>
<tr><td>Transmission message TCD</td><td>0 to 59999 (excluding 10000, 10001, 10200, and 10201)</td></tr>
</table>

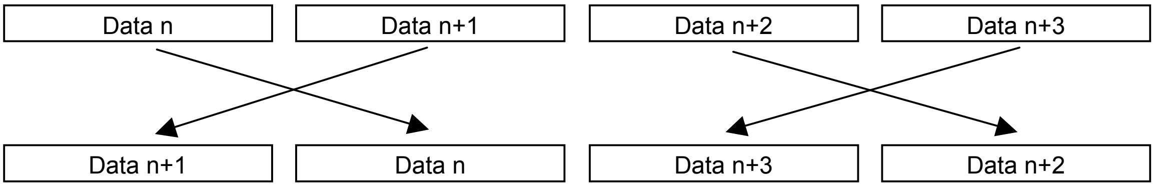
[Function]

- As many data words as specified in bytes will be transferred from the area specified by a transmission data storage address to the remote node having a specified node number as a transparent message having the TCD number specified by a transmission message TCD.
- When the transmission destination node number is set to 255, a 1-to-N transmission takes place.
- When the data word count is set to 0, a transmission takes place with no data added.
- For a transmission message TCD, you cannot specify the TCD numbers (10000, 10001, 10200, and 10201) that are used by the transparent message-unique support feature (specified task control/specified subroutine control).

- FL.NET module-unique feature
  When a TCD number between 0 and 999 is specified, the transmission data is sent as data in little-endian format. If the transmission word count is expressed by an odd number of bytes, the transmission data is converted into little-endian format with the 1-byte data “0x00” added. If the specified TCD number is other than the above, the transmission data is sent as is.

Data conversion effected when the specified TCD number is between 0 and 999

Example: When 4-byte data is received

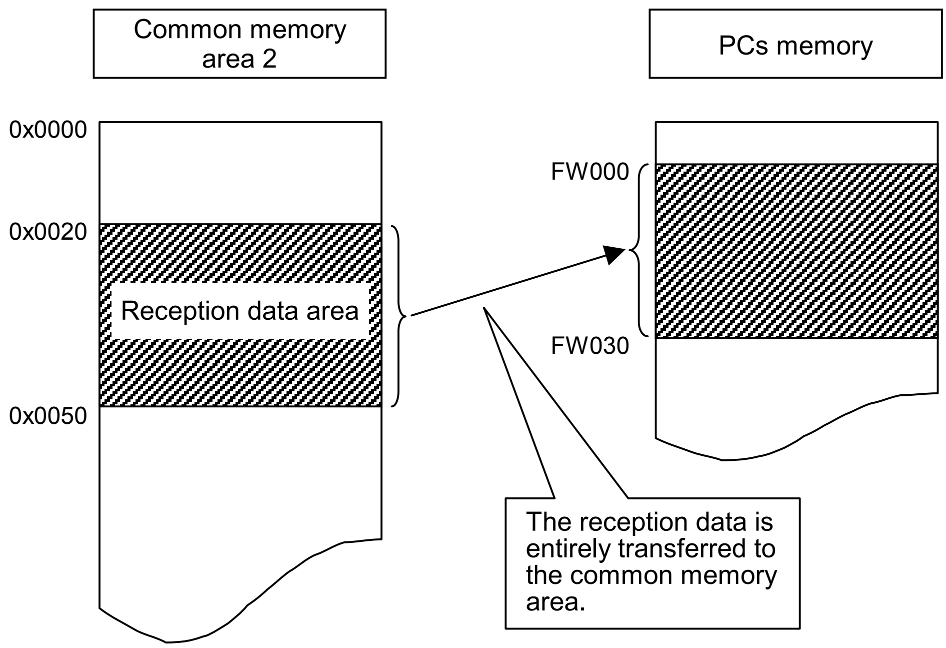


Byte-swapped on an individual word basis and then transmitted.

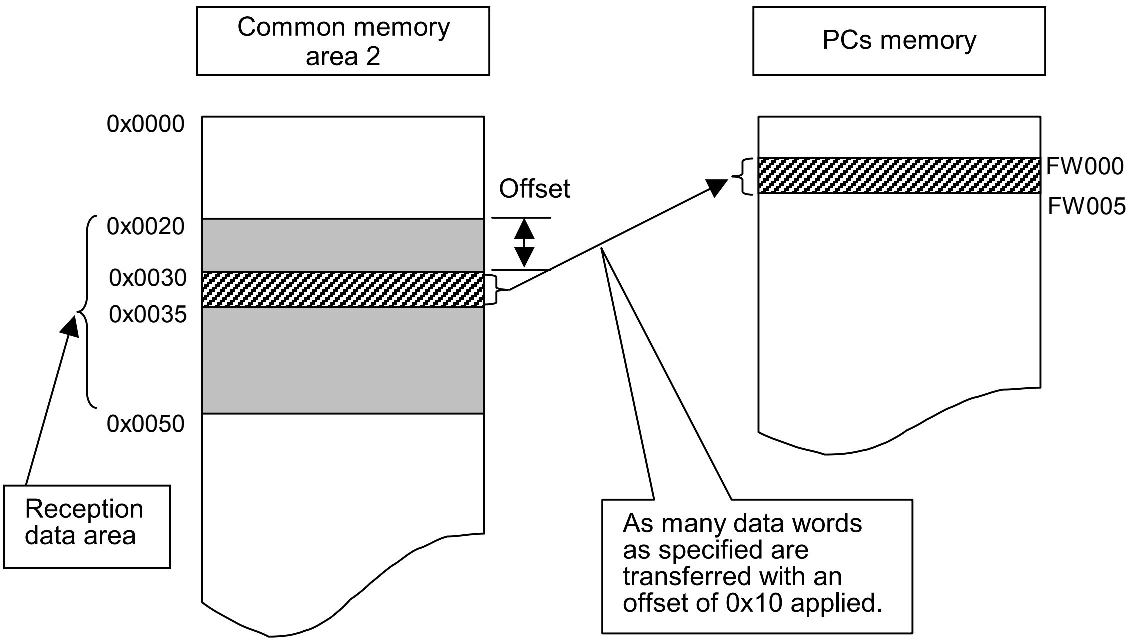
Common memory offset feature: comoffset( )

[Operation performed]
When this handler is executed, the starting address for data transfer is set at a position that is shifted by a specified offset amount from the beginning of a specified node's common memory area.
For each node, the offset size can be variously set for common memory area 1 and 2.
This handler is effective when only part of the received data is to be used.
The examples given below indicate the difference between a normal operation (no offset used) and an operation performed with the offset specified.

Example: When a normal operation is performed (with no offset specified)
Common memory area 2 address: 0x0020
Common memory area 2 word count: 0x0030
PCs memory address: FW000
PCs memory word count: 30



When an offset is specified
Common memory area 2 address: 0x0020
Common memory area 2 word count: 0x0030
PCs memory address: FW000
PCs memory word count: 5
Offset size: 0x10



[Linking procedure]

C language	
Main	Sub
Struct CommonOffset_p { long node ; unsigned short *Erradr ; unsigned long com1offset ; unsigned long com2offset ; } ; ⁝ long (*comoffset)( ) ; long rtn ; struct CommonOffset_p *padr ; ⁝ comoffset= (long(*)( ))0xD74184 ; ⁝ rtn = (*comoffset)(padr) ;	Struct CommonOffset_p { long node ; unsigned short *Erradr ; unsigned long com1offset ; unsigned long com2offset ; } ; ⁝ long (*comoffset)( ) ; long rtn ; struct CommonOffset_p *padr ; ⁝ comoffset= (long(*)( ))0xDF4184 ; ⁝ rtn = (*comoffset)(padr) ;

[Parameters for mathematical/logical function]

+0x0000	Message transmission service number (19)
+0x0004	Specified node number (1 to 254)
+0x0008	Error code storage address
+0x000C	Common memory area 1 offset size
+0x0010	Common memory area 2 offset size

[Parameters]

```
padr                                 : Input parameter storage starting address
Struct CommonOffset_p {
        long node ;                  : Specified node number (1 to 254)
    unsigned short *Erradr ;         : Error code storage address
    unsigned long com1offset ;       : Common memory area 1 offset size
    unsigned long com2offset ;       : Common memory area 2 offset size
} ;
```

Parameter	Input range
Message transmission service number	19 (only when a mathematical/logical function is used)
Specified node number	1 to 254
Error code storage address	Specify a real address (even-numbered address) on the S10mini.
Common memory area 1 offset size	0 to 0x1FF (variable in 1-word units)
Common memory area 2 offset size	0 to 0x1FFF (variable in 1-word units)

[Function]

- When the node having a specified node number is to be subjected to common memory transfer, an offset size will be reported to the FL.NET module.
- If the specified node number coincides with the self-node number, a parameter error occurs.
- If the maximum size (0x200) of common memory area 1 is exceeded by the offset size specified for common memory area 1, a parameter error occurs.
- If the maximum size (0x2000) of common memory area 2 is exceeded by the offset size specified for common memory area 2, a parameter error occurs.
- If you specify the offset size during message processing, a “message processing already in progress” error occurs.

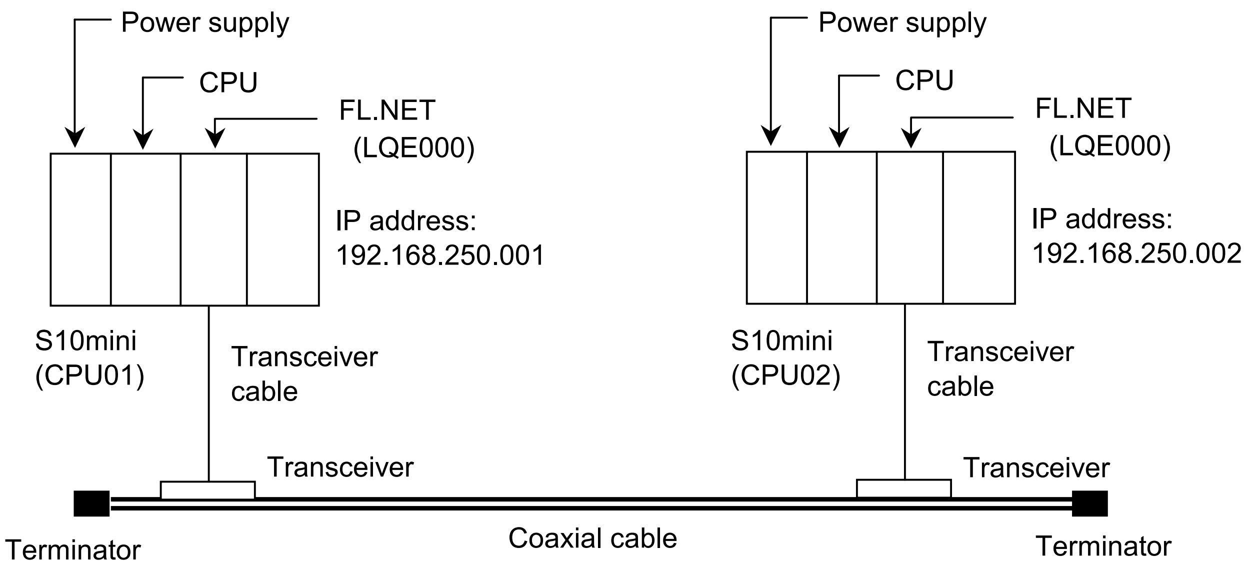
- When an offset size is specified for a specific node number, it is stored in memory until the FL.NET module is turned OFF or reset. Note, however, that the user cannot reference the offset setting within the FL.NET module after completion of offset setup. The offset setting must be managed by the user.
- The above feature is designed for use in cases where the available memory space of the S10mini is insufficient. It is recommended that you do not use the feature when the memory space of the S10mini is adequate.
- When the address limit of common memory area 1 or 2 is exceeded by the application of an offset, the resulting excess portion of address data will not transfer to the S10mini memory. Exercise care not to exceed the limits of the common memory areas.

(6) Program examples

Program examples are shown below. These examples permit two units of the FL.NET module to transmit/receive transparent messages.

(a) Transmission/reception by C mode handler

[System configuration]



Ensure that the FL.NET module MODU No. selector switches for CPU01 and CPU02 are set to 0.

System component list

Product name	Model	Quantity	Remarks
Power supply	LQV000	2	
CPU	LQP010	2	
FL.NET	LQE000	2	
Transceiver cable	HBN-TC-100	2	Manufacturer: Hitachi Cable, Ltd.
Transceiver	HLT-200TB	2	Manufacturer: Hitachi Cable, Ltd.
Coaxial cable	HBN-CX-100	1	Manufacturer: Hitachi Cable, Ltd.
Terminator	HBN-T-NJ	2	Manufacturer: Hitachi Cable, Ltd.
12-V power supply	HK-25A-12	2	Manufacturer: Densei-Lambda K.K.

[FL.NET module settings]

The FL.NET module settings for CPU01 and CPU02 are shown below:

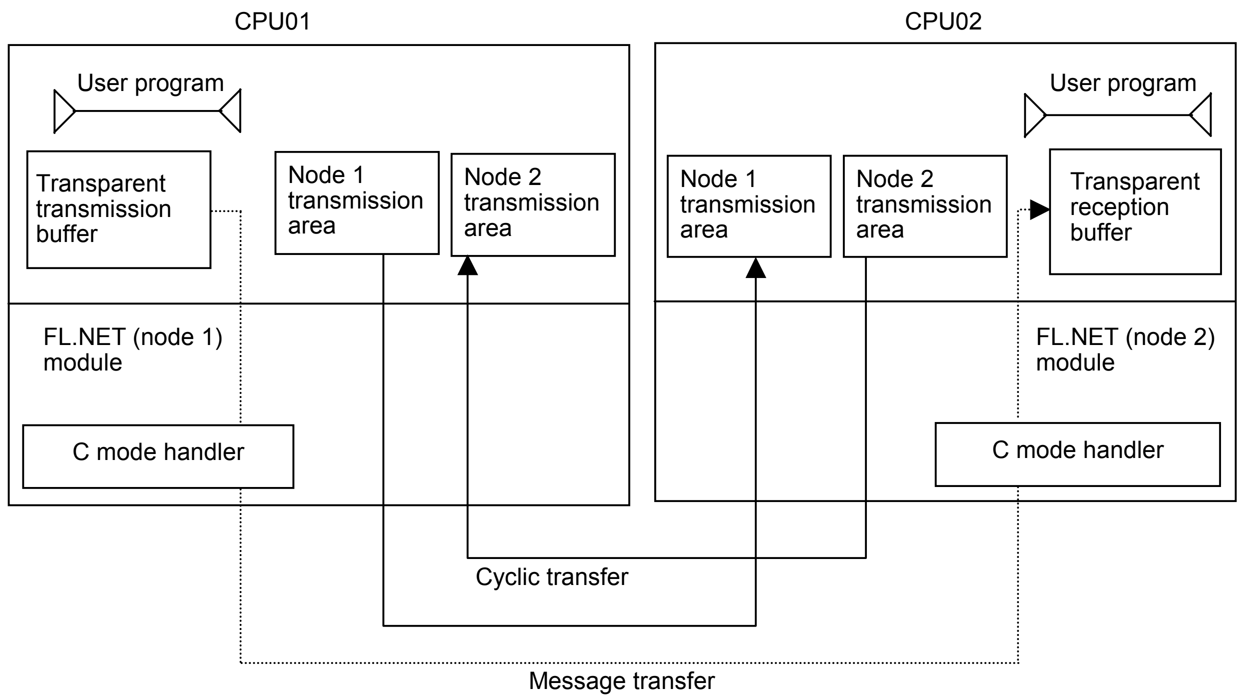
Self-node settings for the FL.NET modules

Setup item	CPU01 setting	CPU02 setting
Node number	1	2
Area 1 address (setting)	0x000	0x004
Area 1 words	0x004	0x004
Area 1 address (PCs allocation)	RW000	RW040
Area 2 address (setting)	0x0000	0x0040
Area 2 words	0x0040	0x0040
Area 2 address (PCs allocation)	DW000	DW040
Self-node status (PCs allocation)	RW080	RW080
Transparent reception task	―――――	―――――
Transparent reception task's factor	―――――	―――――
Transparent receiving flag	―――――	RW100

Remote node settings for FL.NET modules

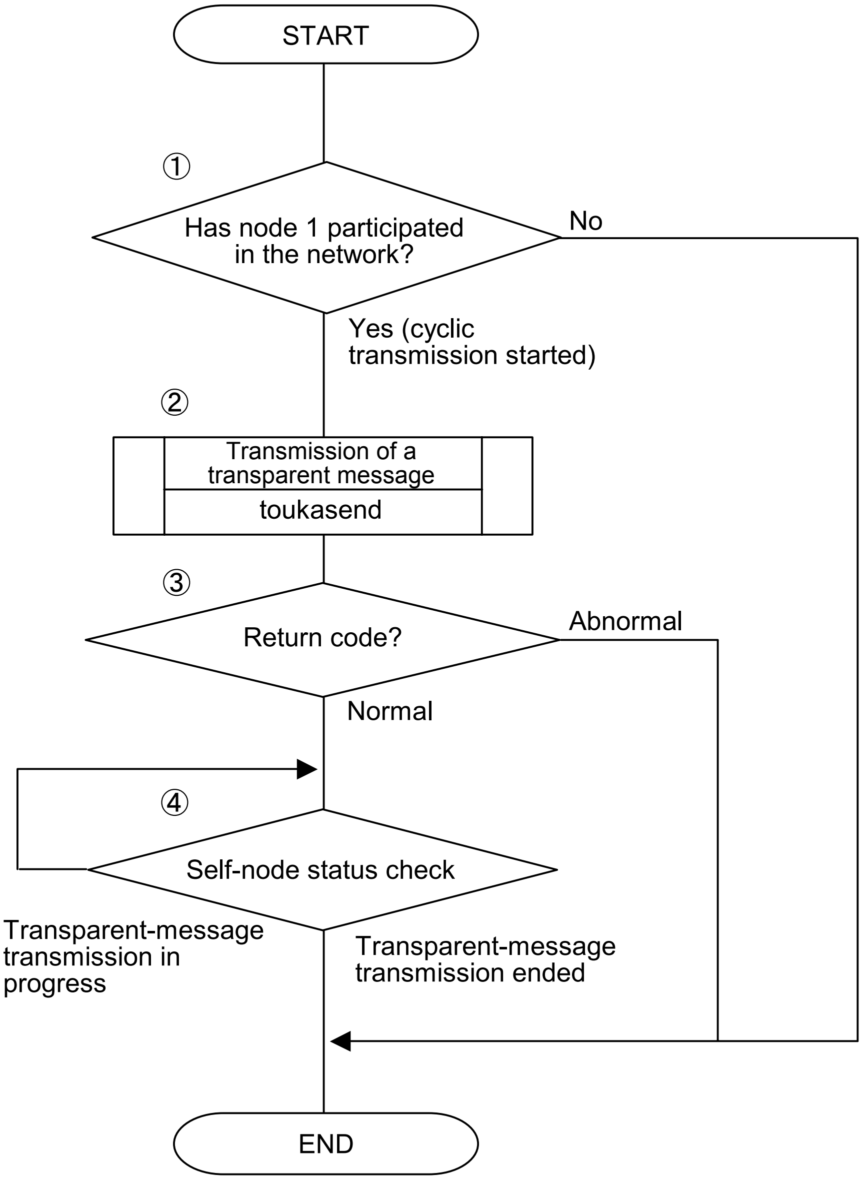
<table>
<tr><th rowspan="2">Setup item</th><th colspan="2">CPU01 setting</th><th colspan="2">CPU02 setting</th></tr>
<tr><th>Node 1</th><th>Node 2</th><th>Node 1</th><th>Node 2</th></tr>
<tr><td>Area 1 address</td><td rowspan="6">No setting is required because node 1 is the self-node.</td><td>RW040</td><td>RW000</td><td rowspan="6">No setting is required because node 2 is the self-node.</td></tr>
<tr><td>Area 1 words</td><td>0x004</td><td>0x004</td></tr>
<tr><td>Area 2 address</td><td>DW040</td><td>DW000</td></tr>
<tr><td>Area 2 words</td><td>0x0040</td><td>0x0040</td></tr>
<tr><td>FA link status</td><td>―――――</td><td>―――――</td></tr>
<tr><td>Higher layer status</td><td>―――――</td><td>―――――</td></tr>
</table>

[Program structure]
The program structure is shown below. With the module having the node number 1 (FL.NET module for CPU01) and the module having the node number 2 (FL.NET module for CPU02) interconnected, cyclic transfer operations are performed between the FL.NET modules. (The FL.NET modules perform cyclic transfer operations. The user does not have to pay attention to the operations.)
Subsequently, the node 1 module transmits a 1,024-byte transparent message (TCD number = 1000) to the node 2 module. The node 2 module receives the transparent message from node 1. This program example operates in this manner.
When running this user program, be sure to start it from CPU01.



Item \ CPU		CPU01	CPU02
Functionality		Transparent message transmission	Transparent message reception
Transmission buffer	Address	0x1E6000	—
	Number of bytes	1,024	—
Reception buffer	Address	—	0x1E6000
	Number of bytes	—	1,024
C mode handler starting address	toukaread( )	0xD74178	0xD74178
	toukasend( )	0xD7417E	0xD7417E

[CPU01-side program flowchart]



① The self-node status transfer area is checked. When the contents of the checked area are 0x0080, the system concludes that the self-node has participated in the network.

② A request for the transmission of a transparent message is issued to the node 2 module.

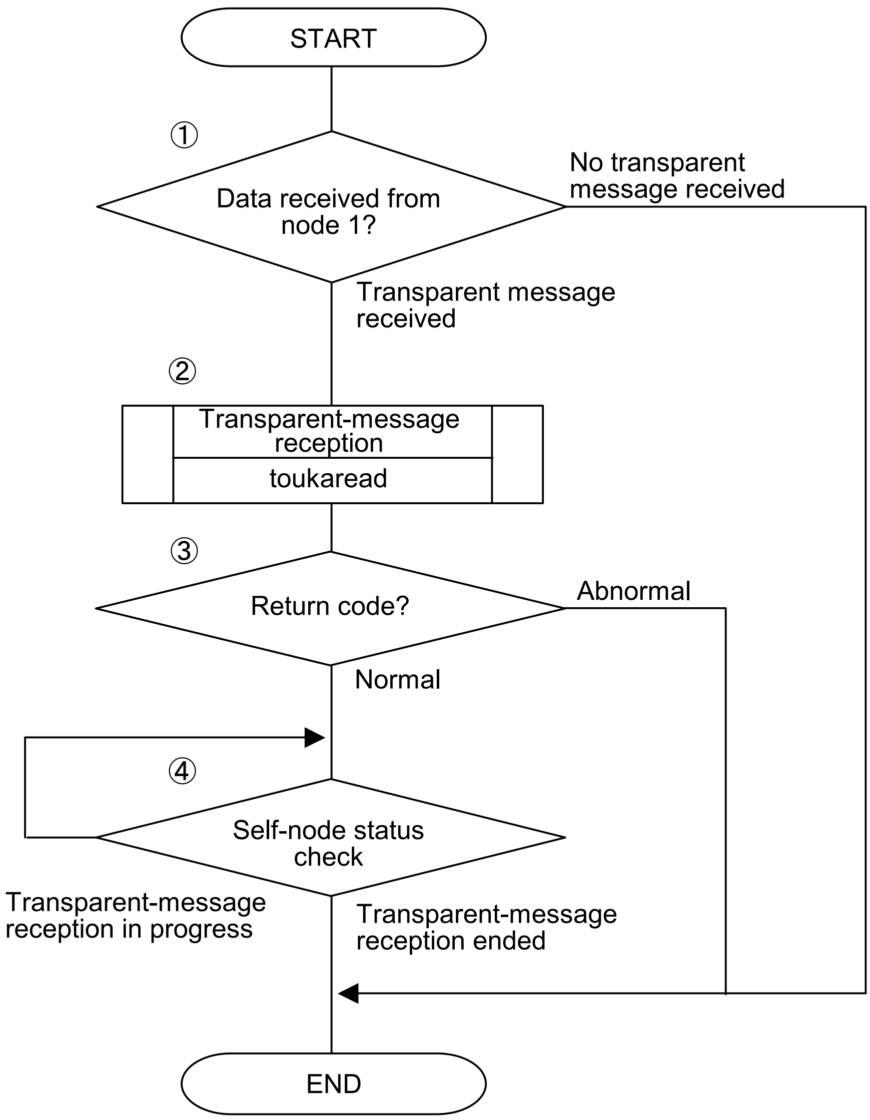
③ The C mode handler checks a return code to determine whether the request is accepted normally. (The request is accepted normally if the return code is 0 or not accepted normally if the return code is -1.)

④ The self-node status transfer area is checked to wait until bit 2^{15} (0x8000) turns OFF. (Bit 2^{15} of the self-node status flag is the “user request processing in progress” bit.)

[CPU01-side C program example]

```
#define TOUKA_SEND      0xD7417EL     /* toukasend() starting address (main) */
#define SBUFADR         0x1E6000L     /* Transmisson buffer address */
#define PARADDR         0x1E5000L     /* Input parameter storage starting address */
#define RW080           0x0E0C10L     /* Self-node status transfer area (RW080) */
#define RW090           0x0E0C12L     /* Transparent-message transmission error code area */

struct ToukaSend_p {
      long              node;         /* Transmission destination node number */
      unsigned short    *Erradr;      /* Error code storage address */
      unsigned char     *dataadr;     /* Transmission data storage starting address */
      unsigned long     datasiz;      /* Transmission data byte count */
      unsigned long     TcdNo;        /* Transparent-message TCD number */
};

/****************************/
/* task2:Transmission(CPU01)    */
/****************************/

main()
{
      register long     ( *toukasend )();
      long              rtn;
      struct            ToukaSend_p      *send;
      unsigned short    *nodeflg;

      nodeflg    = ( unsigned short *)RW080;
      toukasend = ( long(*)( ))TOUKA_SEND;
      send       = ( struct ToukaSend_p *)PARADDR;
      if( !( *nodeflg & 0x0080)){                       /* Self-node status check */
         return;
      }

      send->node     = 0x00000002;                      /* Transmission destination node number */
      send->Erradr   = (unsigned short*)RW090;          /* Error code storage address */
      send->dataadr = (unsigned char*)SBUFADR;          /* Transmission data storage starting address */
      send->datasiz  = 1024;                            /* Transmission data byte count */
      send->TcdNo      = 1000;                          /* Transparent-message TCD number */
      rtn = ( toukasend )( send );                      /* Transparent-message transmission */

      if( rtn != 0){                                    /* Return code check */
         return;
      }

      while( 1 ){
            if( !( *nodeflg & 0x8000)){                 /* Wait for message processing termination */
               break;
            }
      }
}
```

[CPU02 side program flowchart]



① The transparent-type reception flag area is checked. When the contents of the checked area are 0x4000 and bit 2^{14} within the self-node status transfer area is ON (set), the system concludes that there is a transparent message from node 1. (Starting address bit 2^{14} of the transparent-type reception flag is the node 1 reception bit.)

② A request for the reception of a transparent message is issued.

③ The C mode handler checks the return code to determine whether the request is accepted normally. (The request is accepted normally if the return code is 0 or not accepted normally if the return code is -1.)

④ The self-node status transfer area is checked to wait until bit 2^{15} (0x8000) is turned OFF (reset). (Bit 2^{15} of the self-node status flag is the “user request processing in progress” bit.)

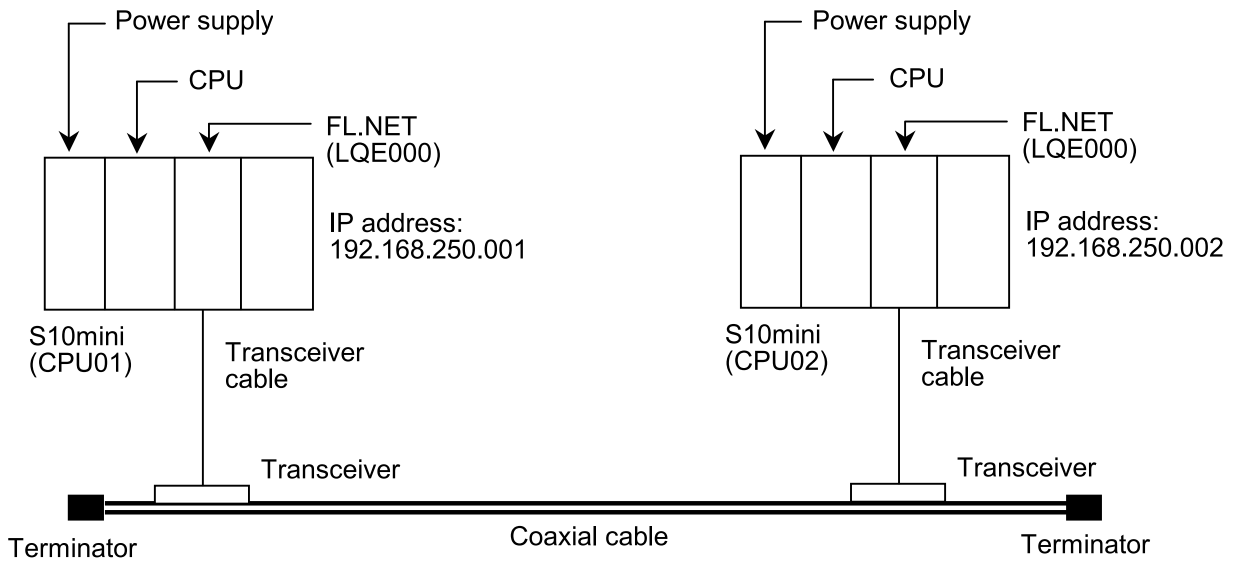
[CPU02-side C program example]

```
#define TOUKA_READ    0xD74178L     /* toukaread() starting address (main) */
#define RBUFADR       0x1E6000L     /* Reception buffer address */
#define PARADDR       0x1E5000L     /* Input parameter storage starting address */
#define RW080         0x0E0C10L     /* Self-node status transfer area (RW080) */
#define RW090         0x0E0C12L     /* Transparent-message transmission error code area */
#define RW100         0x0E0C20L     /* Transparent receiving flag area */

struct ToukaRead_p {
      long             node;        /* Transmission destination node number */
      unsigned short   *Erradr;     /* Error code storage address */
      unsigned char    *dataadr;    /* Reception data storage starting address */
      unsigned long    datasiz;     /* Received data byte count */
};

/*************************/
/* task3:Reception(CPU02)    */
/*************************/

main()
{
      register long    ( *toukaread )();
      long             rtn;
      struct           ToukaRead_p       *read;
      unsigned short   *nodeflg, *recvarea;

      recvarea   = ( unsigned short *)RW100;            /* Transparent receiving flag area */
      nodeflg    = ( unsigned short *)RW080;
      toukaread  = ( long(*)( ))TOUKA_READ;
      read       = ( struct ToukaRead_p *)PARADDR;
      if( ( *recvarea != 0x4000) ||                     /* Transparent receiving flag area and */
         !( *nodeflg & 0x4000)){                        /* Self-node status check */
          return;
      }

      read->node    =   0x00000001;                     /* Transmission source node number */
      read->Erradr  = ( unsigned short *)RW090;         /* Error code storage address */
      read->dataadr = ( unsigned char *)RBUFADR;        /* Reception data storage starting address */
      read->datasiz = 1024;                             /* Received data byte count */
      rtn = ( toukaread )( read );                      /* Transparent-message reception */

      if( rtn != 0){                                    /* Return code check */
          return;
      }

      while( 1 ){
            if( !( *nodeflg & 0x8000)){                 /* Wait for message processing termination */
                break;
            }
      }
}
```

Note: The user program examples presented here are created to facilitate understanding. In a practical program, it is necessary to check an error code that is written to an error code storage address after a “user request processing in progress” bit is turned OFF (reset) subsequently to message issuance.

(b) Transmission/reception by mathematical/logical function

[System configuration]



Ensure that the FL.NET module MODU No. selector switches for CPU01 and CPU02 are set to 0.

System component list

Product name	Model	Quantity	Remarks
Power supply	LQV000	2	
CPU	LQP010	2	
FL.NET	LQE000	2	
Transceiver cable	HBN-TC-100	2	Manufacturer: Hitachi Cable, Ltd.
Transceiver	HLT-200TB	2	Manufacturer: Hitachi Cable, Ltd.
Coaxial cable	HBN-CX-100	1	Manufacturer: Hitachi Cable, Ltd.
Terminator	HBN-T-NJ	2	Manufacturer: Hitachi Cable, Ltd.
12-V power supply	HK-25A-12	2	Manufacturer: Densei-Lambda K.K.

[FL.NET module settings]
The FL.NET module settings for CPU01 and CPU02 are shown below:

Self-node settings for the FL.NET modules

Setup item	CPU01 setting	CPU02 setting
Node number	1	2
Area 1 address (set value)	0x000	0x004
Area 1 words	0x004	0x004
Area 1 address (PCs allocation)	RW000	RW040
Area 2 address (set value)	0x0000	0x0040
Area 2 words	0x0040	0x0040
Area 2 address (PCs allocation)	DW000	DW040
Self-node status (PCs allocation)	RW080	RW080
Transparent reception task	————	————
Transparent reception task's factor	————	————
Transparent receiving flag	————	RW100

Remote node settings for FL.NET modules

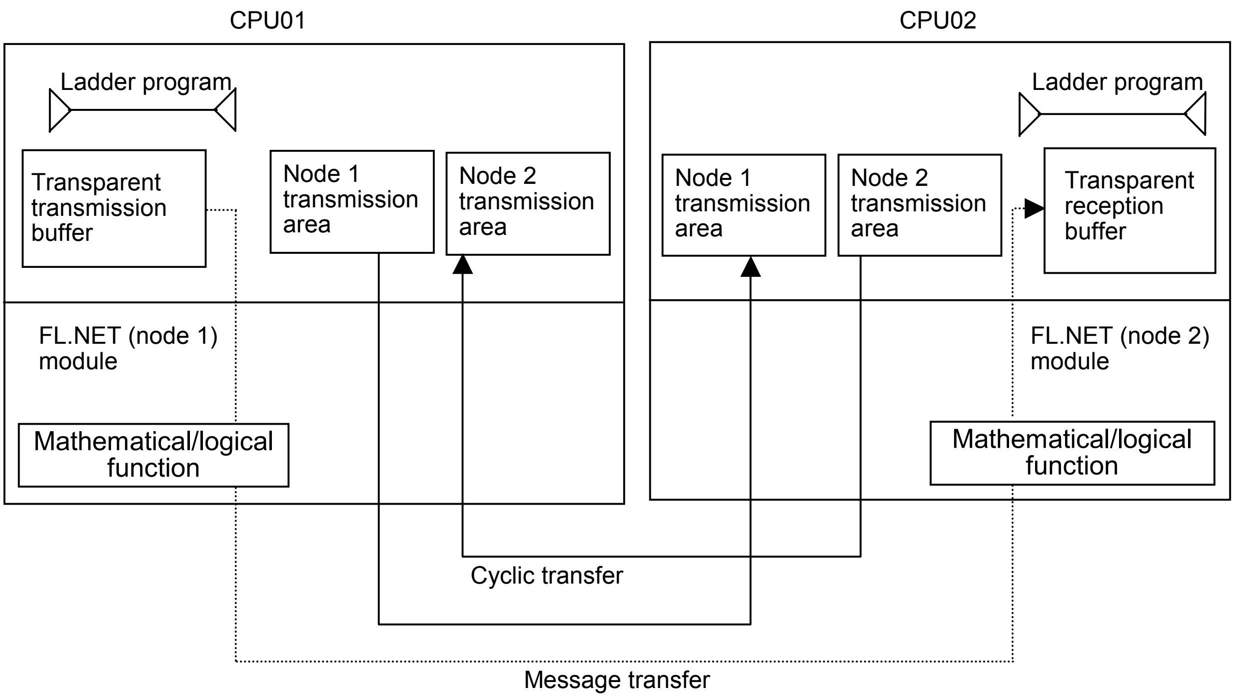
<table>
<tr><th rowspan="2">Setup item</th><th colspan="2">CPU01 setting</th><th colspan="2">CPU02 setting</th></tr>
<tr><th>Node 1</th><th>Node 2</th><th>Node 1</th><th>Node 2</th></tr>
<tr><td>Area 1 address</td><td rowspan="6">No setting is required because node 1 is the self-node.</td><td>RW040</td><td>RW000</td><td rowspan="6">No setting is required because node 2 is the self-node.</td></tr>
<tr><td>Area 1 words</td><td>0x004</td><td>0x004</td></tr>
<tr><td>Area 2 address</td><td>DW040</td><td>DW000</td></tr>
<tr><td>Area 2 words</td><td>0x0040</td><td>0x0040</td></tr>
<tr><td>FA link status</td><td>————</td><td>————</td></tr>
<tr><td>Higher layer status</td><td>————</td><td>————</td></tr>
</table>

[Program structure]

The program structure is shown below. With the module having the node number 1 (FL.NET module for CPU01) and the module having the node number 2 (FL.NET module for CPU02) interconnected, cyclic transfer operations are performed between the FL.NET modules. (The FL.NET modules perform cyclic transfer operations. The user does not have to pay attention to the operations.)

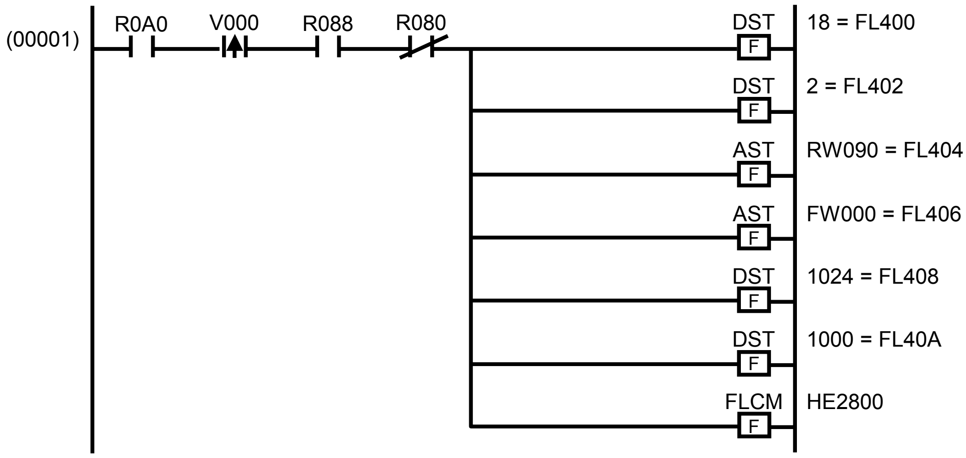
When R0A0 startup is detected in CPU01 side ladder program subsequently, the node 1 module transmits a 1,024-byte transparent message (TCD number = 1000) to the node 2 module. The node 2 module receives the transparent message from node 1. This program example is a ladder program, which runs in this manner.

When using this ladder program, be sure to place the S10mini in the RUN state.



Item \ CPU		CPU01	CPU02
Functionality		Transparent message transmission	Transparent message reception
Transmission buffer	Address	FW000 to FW1FF	—
	Number of bytes	1,024 bytes	—
Reception buffer	Address	—	FW000 to FW203
	Number of bytes	—	1,024 bytes + 4 words
Mathematical/logical function	FLCM	0xD74100	0xD74100

[CPU01 side ladder program]



When the R0A0 turns ON, the program runs and checks the network participation status bit (R088) and user-requested-processing-in-progress bit (R080) in the self-node status flag (message processing can be performed when the network participation status bit is set with the user-requested-processing-in-progress bit reset.)
The parameters for transparent-message transmission are specified by mathematical/logical functions.

- DST 18 = FL400
  The message transmission service number is set to 18. A transparent-message transmission request is specified.

- DST 2 = FL402
  The remote station (CPU02) is specified.

- AST RW090 = FL404
  The error code storage address (RW090 real address) is specified.

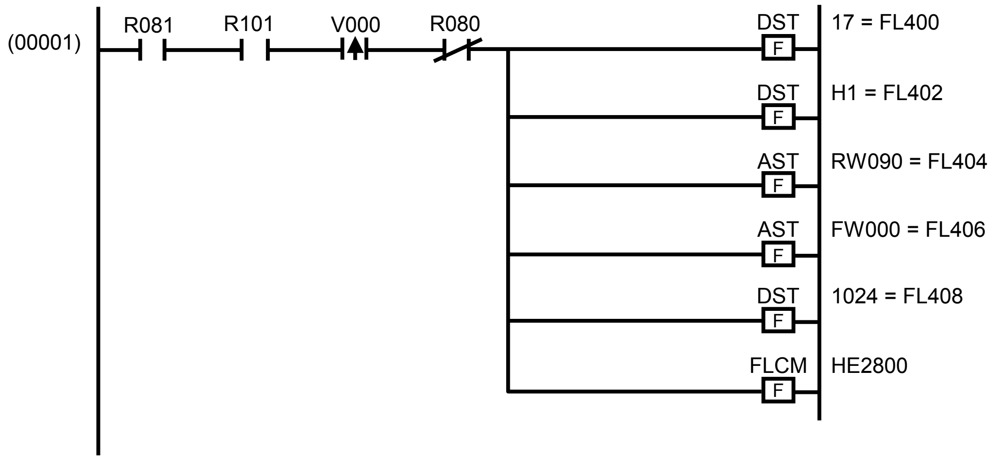
- AST FW000 = FL406
  The transmission buffer address (FW000 real address) is specified.

- DST 1024 = FL408
  The transmission word count is specified.

- DST 1000 = FL40A
  The TCD number (1000) is specified for the transparent message to be transmitted.

- FLCM HE2800
  The parameter storage address (HE2800 = FW400 real address) is specified for the mathematical/logical function (FLCM). A message request (transparent-message transmission) is issued to the FL.NET module.

[CPU02-side ladder program]



The program runs when the transparent-message reception bit (R081) in the self-node status flag is set and the node 1 reception bit (R101) in the transparent-type reception flag area is also set. The program then checks the user-requested-processing-in-progress bit (R080) in the self-node status flag (message processing can be performed when the user-requested-processing-in-progress bit is reset.)
The parameters for transparent-message transmission are specified by mathematical/logical functions.

- DST 17 = FL400
  The message transmission service number is set to 17. A transparent-message reception request is specified.

- DST 1 = FL402
  A transparent message from the remote station (CPU01) is specified.

- AST RW090 = FL404
  The error code storage address (RW090 real address) is specified.

- AST FW000 = FL406
  The reception buffer address (FW000 real address) is specified.

- DST 1024 = FL408
  The reception word count is specified.

- FLCM HE2800
  The parameter storage address (HE2800 = FW400 real address) is specified for the mathematical/logical function (FLCM). A message request (transparent-message transmission) is issued to the FL.NET module.

Note: The ladder program examples presented here are created to facilitate understanding. In a practical program, it is necessary to check an error code that is written at an error code storage address after a “user request processing in progress” bit is reset subsequently to message issuance.

6.4.6 Using the management tables

The FL.NET module uses various management tables to manage the status of communications with remote nodes.
You can determine the status of communications with the remote nodes by referencing these management tables.
Note, however, that the setup tool named "FL.NET For Windows®" must be used to reference the management tables within the FL.NET module.
For details on the procedures to be performed from various windows, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

(1) Referencing the self-node management table
To view the self-node management table, use the setup tool "FL.NET For Windows®" to display the information about participating nodes. When a list of the numbers of the nodes that have participated in the network appears on the display, select the on-screen icon for the self-node.

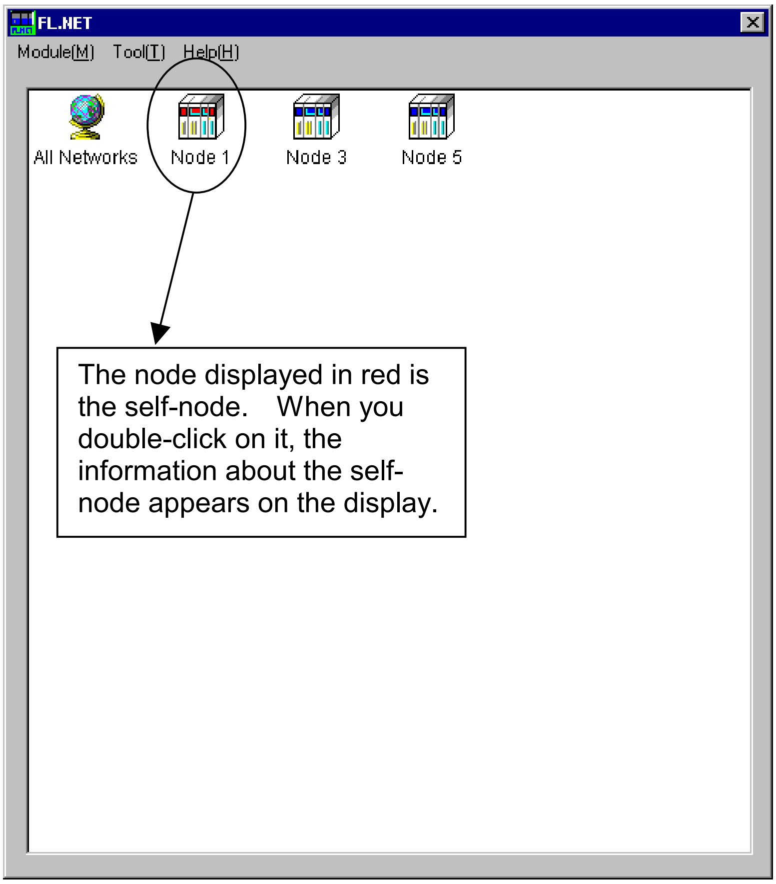


Figure 6-47 Participating Node Number List Window 1

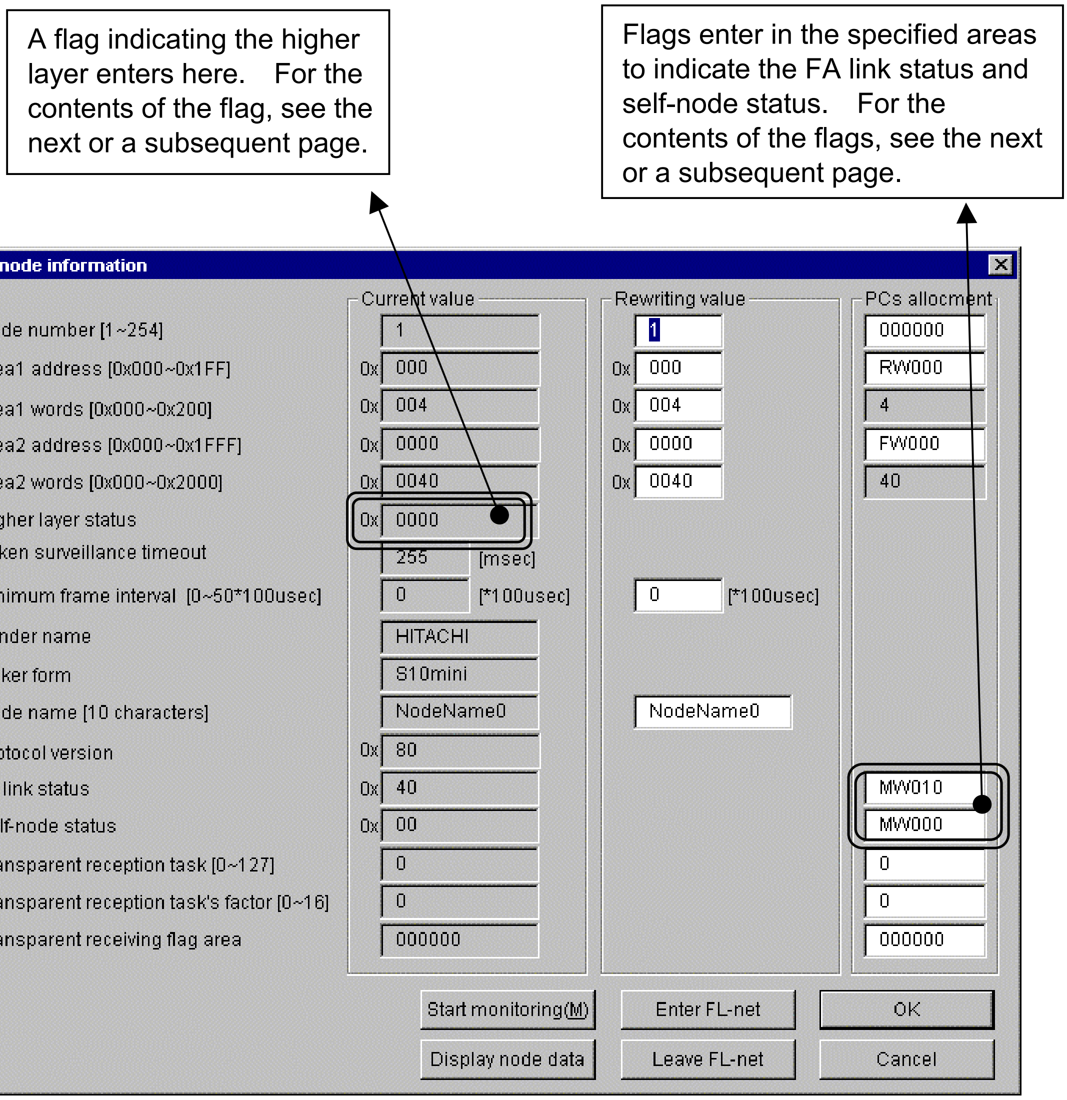
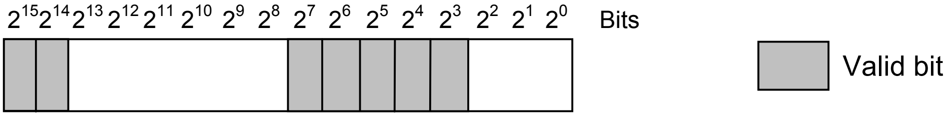


Figure 6-48 Self-node Information (Self-node Management Table) Window

[Contents of self-node status flag]

In the FL.NET module, the bit allocation for the self-node status flag is as indicated below:



Token hold timeout
- =0: A token hold timeout has not occurred.
- =1: A token hold timeout has occurred.

FL.NET module setup error
- =0: The self-node has not participated in the network.
- =1: The self-node has participated in the network.

Frame wait status
- =0: The network participation process is in progress or network participation has already done.
- =1: Although a request for participation has been transmitted three or more times, the existence of a remote node cannot be confirmed. The participation process is being continuously performed.

Node number duplication
- =0: There are no duplicate node numbers.
- =1: There is a node having the same number as the self-node.

Network participation status
- =0: The self-node has not participated in the network.
- =1: The self-node has participated in the network.

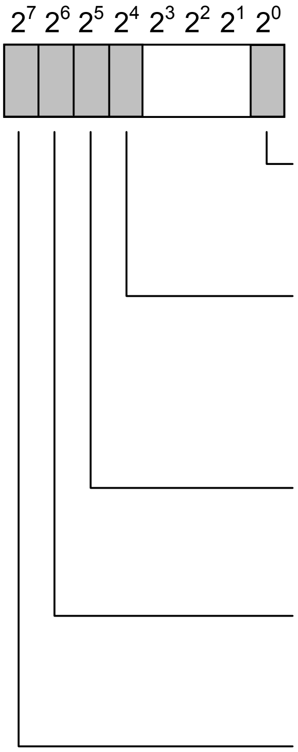
Transparent-message reception
- =0: There is no transparent message from a remote node.
- =1: There is a transparent message from a remote node.

User request processing in progress
- =0: A request from the user is being awaited.
- =1: A request from the user is being processed.

Note: The above self-node status represents the contents that are transferred to the S10mini. The self-node information dialog box of the setup tool "FL.NET For Windows®" allows you to reference the low-order byte (bits 2^7 through 2^0) only.

[Contents of FA link status flag (FA link status)]

The bit allocation for the FA link status flag is as indicated below:



Valid bit

Participation status bit
- =0: The self-node has not participated in the network.
- =1: The self-node has participated in the network.

Common memory duplication bit
- =0: There is no duplicate common memory setting.
- =1: There is a duplicate common memory setting for a remote node. All the common memory settings for the self-node will be changed to 0 to participate in the network.

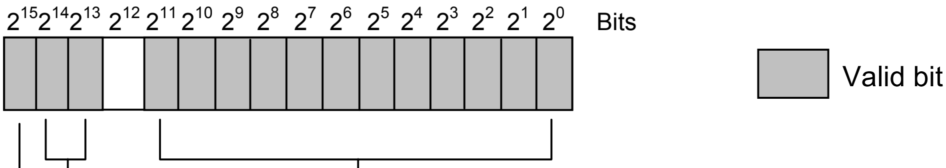
Common memory setup completion bit
- =0: Common memory setup is not complete (an error exists).
- =1: Common memory setup is completed (no error exists).

Common memory data validity bit
- =0: The common memory data is invalid.
- =1: The common memory data is valid.

Higher layer bit
- =0: The higher layer is normal.
- =1: The higher layer is abnormal. For details, refer to the higher layer status flag.

[Contents of higher-layer status flag]

The bit allocation for the higher-layer status flag is as indicated below:



Higher-layer error code (12-bit)
The code for an error existing in the higher layer is displayed.
The following table shows the error codes for the FL.NET module.

Error code	Description
0x000	The higher layer (S10mini) is normal.
0x0FF	The CPU unit in the higher layer (S10mini) is inoperative.

Higher-layer error status bits (2 bits)
These bits describe a higher-layer error when it occurs.

<table>
<tr><th>Bit 2^{14}</th><th>Bit 2^{13}</th><th>Description</th></tr>
<tr><td>OFF</td><td>OFF</td><td>The higher layer is normal.</td></tr>
<tr><td>OFF</td><td>ON</td><td>An error exists in the higher layer but permits the continuation of operation (ALARM state).
The cyclic data and message data are guaranteed.</td></tr>
<tr><td>ON</td><td>OFF</td><td rowspan="2">An error exists in the higher layer and does not permit the continuation of operation (WARNING state).
The cyclic data and message data are not guaranteed.</td></tr>
<tr><td>ON</td><td>ON</td></tr>
</table>

When a higher-layer error occurs, the FL.NET module turns ON (sets) bits 2^{14} and 2^{13}.

Higher-layer operation status bit
=0: The higher layer (S10mini) is halted (in the STOP state).
=1: The higher layer (S10mini) is running (in the RUN state).

(2) Referencing the participating node management table
When you display information about participating nodes with the setup tool “FL.NET For Windows®,” you can view the “Other node information” dialog box, which lists the node numbers of the nodes that have participated in the network and the participating node management table contents concerning the participating node numbers.
Note, however, that the setup tool “FL.NET For Windows®” merely allows you to view the contents of the participating node management table concerning the nodes that have participated in the network.

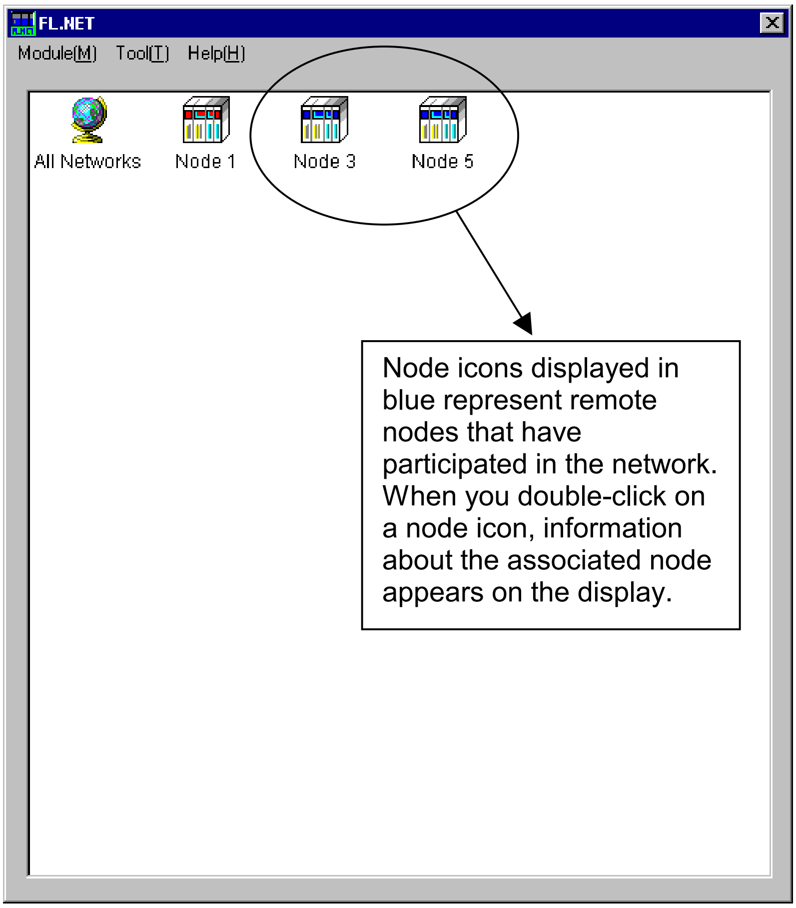


Figure 6-49 Participating Node Number List Window 2

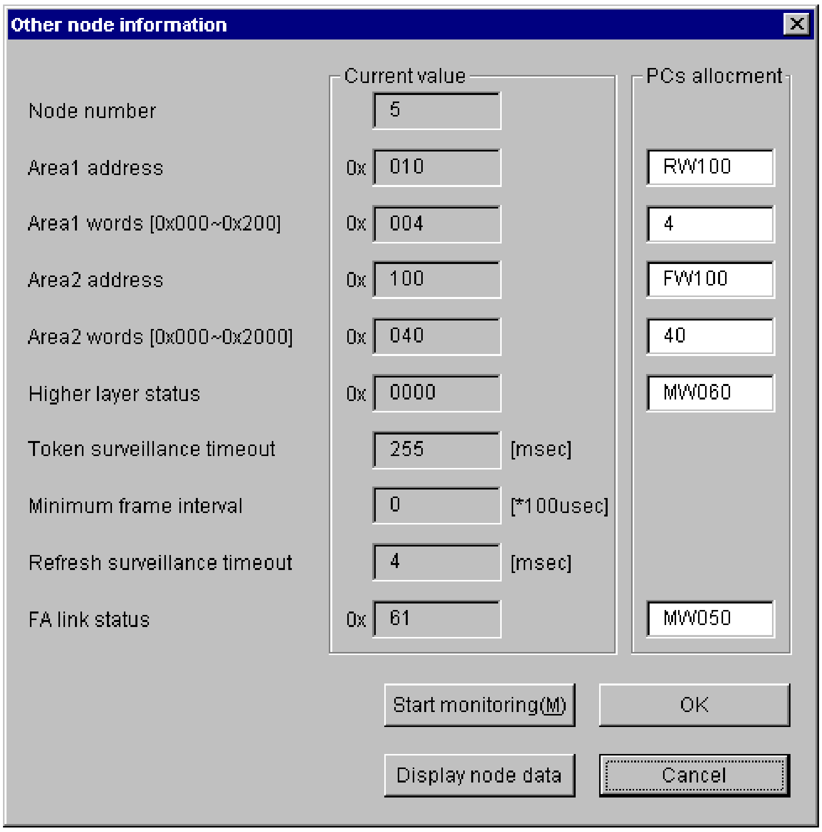


Figure 6-50 Participating Other Node Link Information Window

For the bit allocation for the higher layer status and FA link status about individual nodes, refer to the respective node manuals.

(3) Referencing the network management table
The network status window of the setup tool “FL.NET For Windows®” allows you to view the contents of the network management table.

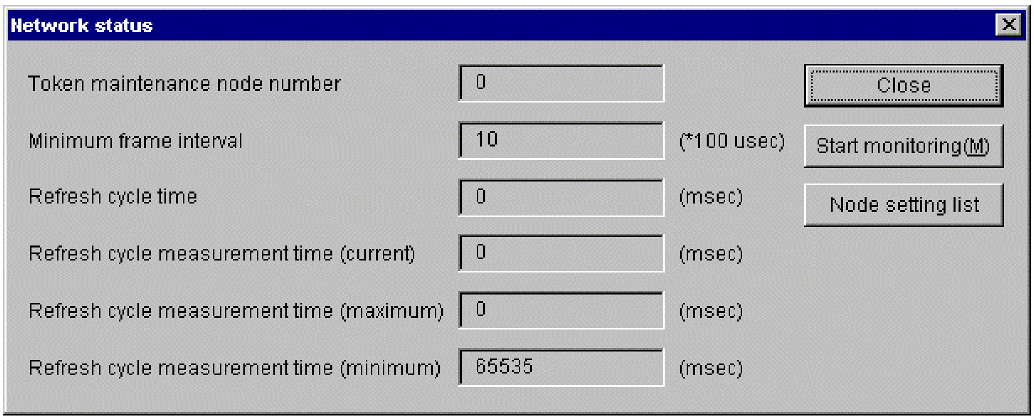


Figure 6-51 Network Status (Network Management Table) Window

The contents of the above window are explained below:

On-screen item	Description
Token maintenance node number	This is the node number of the node that currently holds the token.
Minimum frame interval	This value denotes the minimum frame interval that is allowed on the network. It represents the maximum value of the nodes that compose the network.
Refresh cycle time	This value is 1.2 times the time required for a token circulation through the network.
Refresh cycle measurement time (current)	These values are the current value, maximum value, or minimum value of the measured time required for a token circulation through the network.
Refresh cycle measurement time (maximum)	
Refresh cycle measurement time (minimum)	

6.4.7 FL.NET module communication time

The FL.NET communication performance characteristics of the S10mini are stated below. Note that the performance varies with the remote station to communicate with and the word count. The communication performance characteristics stated below prevail under the following conditions:

Conditions
- The S10mini FL.NET units are interconnected.
- The connected FL.NET module employs the same communication word count C (C1 + C2) set.

The meanings of the symbols used in calculation formulas are explained below:

C: Communication word count of area 1 and 2 (words)
C1: Area 1 communication word count (words)
C2: Area 2 communication word count (words)
T: Remainder obtained when the communication word count is divided by 512.
If the word count is a factor of 512, T = 512 (T = 1 to 512 words).
RCT: Refresh cycle (ms)
SCT: Sequence cycle (ms)
MS: Transmitted message word count (bytes)
MR: Received message word count (bytes)

- Token hold time (ms)
  When the token hold time (ms) > minimum permissible frame interval (ms)
  (If the minimum permissible frame interval is greater than the token hold time, the former is used as the token hold time value.)
  When $C \leq 512$: $1.51 + 0.00371*C$
  When $C > 512$: $1.51 + 0.01164*C1 + 0.01006*C2 - 0.01055*T$

- Measured refresh cycle time (ms)
  Sum of the token hold time values of all the nodes. When three units (nodes 1 through 3) are interconnected, this value = node 1 token hold time + node 2 token hold time + node 3 token hold time. (For token hold time calculation, see the explanation of the preceding item.)

- Cyclic transfer throughput (ms)
  $2 \times RCT + 2 \times SCT$ (worst case for ladder).

- Word block read (ms)
  Time interval between mathematical/logical function (subroutine) start and data reception.
  When $C \leq 512$: $3.1 + 0.0177 \times C1 + 0.0169 \times C2 + 2RTC$ + (total token hold time required between the self-node and transmission destination node)
  When $C > 512$: $3.1 + 0.0217 \times C1 + 0.0201 \times C2 - 0.0032 \times T + 2RTC$ + (total token hold time required between the self-node and transmission destination node)

- Word block write (ms)
  Time interval between mathematical/logical function (subroutine) start and ACK reception from a remote node.
  When $C \leq 512$: $4.32 + 0.0106 \times C1 + 0.0118 \times C2 + 2RTC$ + (total token hold time required between the self-node and transmission destination node)
  When $C > 512$: $4.32 + 0.0138 \times C1 + 0.0148 \times C2 - 0.0032 \times T + 2RTC$ + (total token hold time required between the self-node and transmission destination node)

- Transparent message (ms)
  When the message creation time (ms) = 0.02 + (0.0007 × MS) > minimum permissible frame interval (ms)
  (If the minimum permissible frame interval is greater than the message creation time, the former is used as the message creation time value.)
    1-to-1 transmission request process time
    Measured refresh cycle time × 2 + self-node token hold time + message creation time + 1.272 + (0.00096 × MS) + (0.0013 × MR)
    1-to-N transmission request process time
    Measured refresh cycle time + message creation time + 0.742 + (0.00096 × MS)
    (For the calculations of measured refresh cycle time and token hold time, see the explanations of the associated preceding items.)

6.4.8 Using the communication log

With the setup tool “FL.NET For Windows®,” you can view the RAS information (communication log) maintained in the FL.NET module.

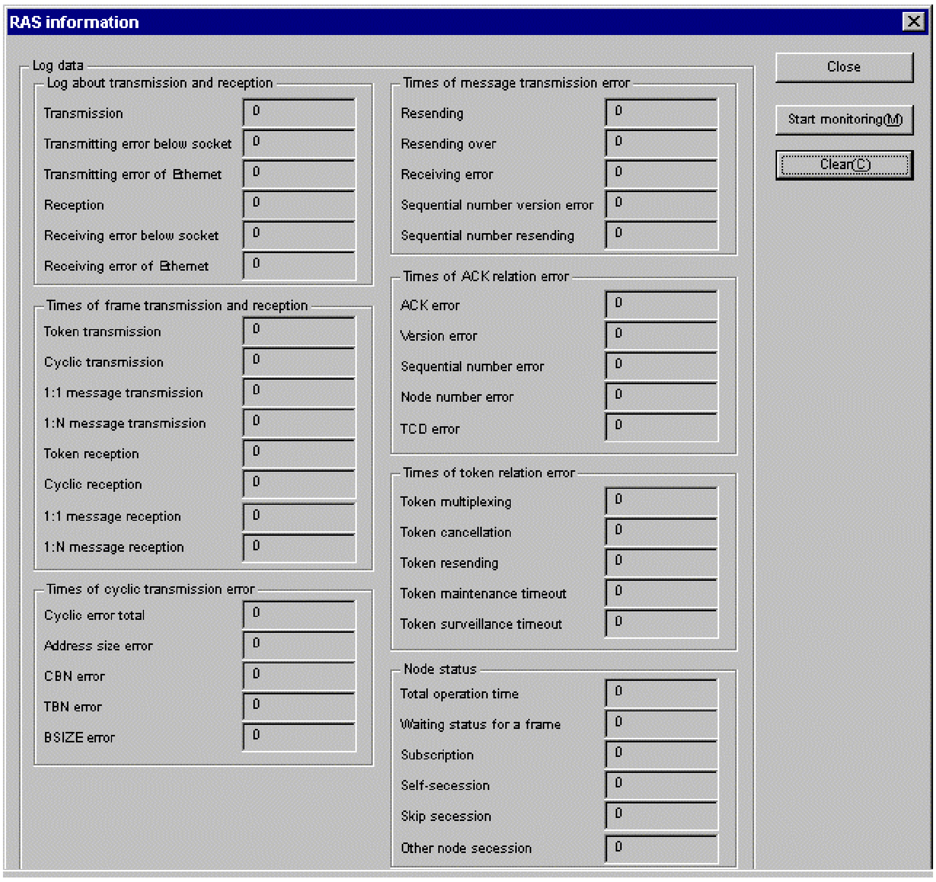


Figure 6-52 RAS Information Dialog Box

The table below describes the on-screen log information (RAS information).

Item		On-screen information description
Log about transmission and reception	Transmission	Number of frames whose transmissions were requested
	Transmitting error below socket	Number of transmission errors in the socket section
	Transmitting error of Ethernet	Unused
	Reception	Number of frames received from the socket section
	Receiving error below socket	Number of reception errors and abnormal frame receptions in the socket section
	Receiving error of Ethernet	Unused
Times of frame transmission and reception	Token transmission	Number of token transmissions
	Cyclic transmission	Number of cyclic frame transmissions without a token
	1-to-1 message transmission	Number of 1-to-1 message transmissions
	1-to-N message transmission	Number of 1-to-N message transmissions
	Token reception	Number of token receptions
	Cyclic reception	Number of cyclic frame receptions without a token
	1-to-1 message reception	Number of 1-to-1 message receptions
	1-to-N message reception	Number of 1-to-N message receptions
Times of cyclic transmission error	Cyclic error total	Number of errors in cyclic transmission reception
	Address size error	Unused
	CBN error	Number of errors concerning frame arrangement
	TBN error	Number of errors concerning frame segment count
	BSIZE error	Number of errors concerning frame size
Times of message transmission error	Resending	Number of message transmission retries made
	Resending over	Number of message transmission retries exceeded errors
	Receiving error	Number of message transmission reception errors
	Sequential number version error	Number of sequence number/version error recognitions in message receptions
	Sequential number resending	Number of message transmission retry recognitions in message receptions
Times of ACK relation error	ACK error	Number of ACK errors
	Version error	Number of ACK sequence number/version mismatches
	Sequential number error	Number of ACK sequence number mismatches
	Node number error	Unused
	TCD error	Unused
Times of token error	Token multiplexing	Number of recognitions of multiplexed tokens
	Token cancellation	Number of token cancellations
	Token sending	Number of token reissues
	Token maintenance timeout	Number of token hold timeout occurrences
	Token surveillance timeout	Number of token monitoring timeout occurrences
Node status	Total operating time	Unused
	Waiting status for a frame	Number of frame waits due to the lack of remote nodes on the network
	Subscription	Number of subscriptions to the network
	Self secession	Number of occurrences of three consecutive token hold timeouts or departures due to the lack of remote nodes on the network
	Skip secession	Number of departures due to the skip of a token addressed to the self-node

7 MAINTENANCE

7.1 Maintenance Programs

To use the S10mini in an optimum condition, check the items listed below. Make this check at routine inspection or periodic inspection (twice or more per year).

- Module appearance
  Check that no fissure or crack exists in the module case. If the case has such a damage, there is a possibility that the internal circuit may also be damaged, resulting in a system malfunction.

- Indicator's ON status and indication
  From the indicator status, check that no special fault exists.

- Looseness of mounting screws and terminal base screws
  Check that the mounting screws and terminal base screws of the module are not loose. If any of these screws is found to be loose, tighten it. Such a loose screw may result in a system malfunction or a burn-out due to overheating.

- Module replacement
  Hot swapping of modules will lead to hardware or software damage. Be sure to replace a module in a power OFF state.

- Cable sheath condition
  Check that the cable sheath is not abnormal. A peeled sheath may cause a system malfunction or electric shock, or may result in a burn-out due to short circuit.

- Dust sticking condition
  Check if dust and dirt collects on the module. If dust collects on the module, remove it with a vacuum cleaner. Dust on the module may short the internal circuit, resulting in a burn-out.

- Power supply voltage
  Check that neither the internal power supply of the module nor the external power supply to it is out of the specified range. If the power supply voltage deviates from the rating, a system malfunction may result.

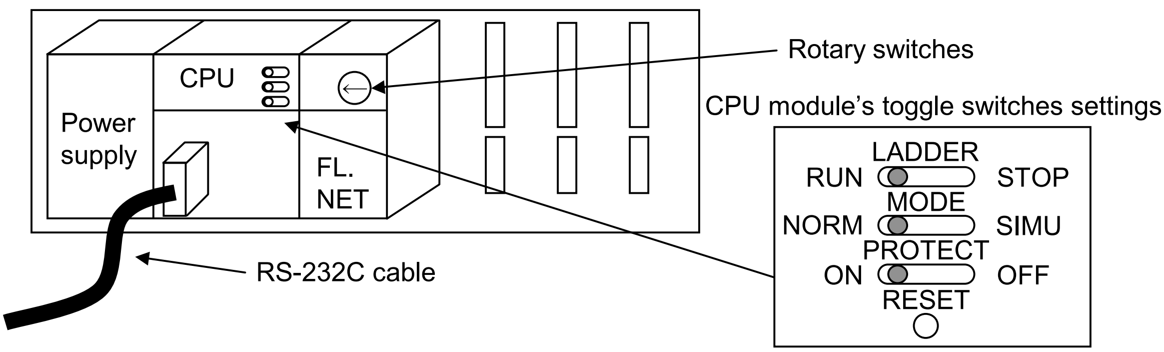
CAUTION

Static electricity may damage the module. Before starting the work, discharge all electrostatic charge from your body.

7.1.1 Replacing or adding on the module

- What you should get in preparation
  ① Personal computer (with Hitachi's S10 FL.NET System installed in it)
  ② RS-232C cable (or 10BASE-T cable if the communication module used is an ET.NET module)
  ③ New or add-on FL.NET module (LQE000)
  ④ Copies of the parameter values for the module to be replaced. (These copies are prepared for use in cases where the parameters are not accessible for some reason.)
  ⑤ The above-mentioned ET.NET module is an optional module and, if it is mounted in place, may be selected as the type of communication module to be used. For more information, refer to Section 2.1, "Names and Functions of Each Part," and Section 3.2, "Mounting the Module," in the USER'S MANUAL OPTION ET.NET (LQE520) (manual number SVE-1-103).

- Replacement procedure
  ① Write down, on a piece of paper, the current settings of the rotary switches that are, as shown below, accessible at the front side of the FL.NET module to be replaced.
  ② Write down also the current settings of three switches, labeled LADDER (toggle switch), MODE (toggle switch), and PROTECT (toggle switch), respectively, that are, as shown below, accessible at the front side of the CPU module.



  ③ Connect the personal computer and the CPU module together with the RS-232C cable.
  ④ Start Hitachi's S10 FL.NET System. Then, make a hand-written record of the currently used IP address and save the set values of all the existing parameters. (If the existing parameters are not accessible for some reason, use the copies of their set values [item ④] that were obtained in preparation.)
  ⑤ Set the CPU module's LADDER switch in STOP position and turn off the power supply of the controller unit.

⑥ Remove the connecting cables from the FL.NET module to be replaced.
⑦ Replace the existing FL.NET module with the new one and set the new FL.NET module's rotary switches in the same way as you wrote down in Step ①.
⑧ Turn on the power supply of the controller unit. Then, enter the same IP address as you recorded in Step ④ and send out the parameter values you saved in Step ④, all by using the S10 FL.NET System.
⑨ Check that all the set parameter values are identical to those that were saved in Step ④.
⑩ Turn off the power supply of the controller unit.
⑪ Remove the RS-232C cable from both the personal computer and CPU module, which were connected together in Step ③.
⑫ Connect to the new FL.NET module the connecting cables that you removed in Step ⑥.
⑬ Set the CPU module's LADDER and T/M switches in the same way as you wrote down in Step ②.
⑭ Turn on the power supply of the controller unit and check that the new FL.NET module is running normally.

- Add-on procedure

① Write down, on a piece of paper, the current settings of three switches, labeled LADDER (toggle switch), MODE (toggle switch), and PROTECT (toggle switch), respectively, that are accessible at the front side of the CPU module, the one that is installed in the controller unit in which you are adding on a FL.NET module.
② Ensure that your application system has been shut down. Then, set the CPU module's LADDER switch in STOP position and turn off the power supply of the controller unit.
③ Mount the add-on FL.NET module in place according to the instructions given under "4.1 Mounting the Module."
④ Set the add-on FL.NET module's rotary switches in such a way that a new module No. setting, which must be a sub-module No. setting, will not duplicate with the current rotary switch settings of the existing main FL.NET module.
⑤ Connect the personal computer and the CPU module together with the RS-232C cable. Then, turn on the power supply of the controller unit and set parameters for the add-on FL.NET module by using the S10 FL.NET System.
⑥ Turn off the power supply of the controller unit and connect the connecting cables to the add-on FL.NET module.
⑦ Set the CPU module's LADDER, MODE, and PROTECT switches in the same way as you wrote down in Step ①.
⑧ Remove the RS-232C cable from both the personal computer and CPU module, which were connected together in Step ⑤.
⑨ Turn on the power supply of the controller unit and check that the add-on FL.NET module is running normally.

8 TROUBLESHOOTING

8.1 Trouble Detection and Solution

When you wonder if a failure may have occurred, check the following item.

- Check that the module is properly installed.
- Check that the module switches are properly set.
- Check that network IP address setup is properly completed.
- Check that the common memory area is properly set.
- Check that the module connector connections are tight.
- Check that the communication cables are properly connected.
- Check that the 10BASE-5 coaxial cable terminal resistor is properly connected.
- Check that the 10BASE-5 coaxial cable is properly grounded.
- Check that a 10BASE-T cross cable is not used.
- Check that the employed 10BASE-T cable complies with the Category 5 specifications.
- Check that the Ethernet hub and repeater are turned ON.

8.2 Network Problems and Repairing

(1) Network problems and remedies (concerning communication failures)

Table 8-1 Network Problems and Repairing (Concerning Communication Failures)

Problem	Inspection item	Inspection procedure	Remedy
Communication cannot be established.	Power supply	Check whether the device main power lamp is illuminated.	Check for a faulty power supply, disconnected power cable, and improper voltage.
		Check whether the power lamp on the AUI power unit is illuminated.	Check for a faulty power supply, disconnected power cable, and improper voltage.
		Check whether the AUI power supply unit outputs the specified voltage (12 V).	Check for a faulty power supply, disconnected power cable, and improper voltage.
		Check whether the hub power lamp is illuminated.	Check for a faulty power supply, disconnected power cable, and improper voltage.
		Check whether the AUI power cable is properly connected to the device.	Check for a faulty power supply, disconnected power cable, and improper voltage.
	Communication cable connection to transceiver	Check that the transceiver mount is secured.	Perform the installation procedure again as directed in "9.6 FL-net Network Installation Procedures."
		With the installed-transceiver checker, check for abnormalities.	Make adjustments until normality is restored. If the abnormality persists, change the mounting location.
		Check whether the transceiver is properly insulated.	Perform the installation procedure again as directed in "9.6 FL-net Network Installation Procedures."
		Check that the transceiver is properly installed on the marker location of the communication cable.	Review the mounting location in accordance with "9.6 FL-net Network Installation Procedures."
	Transceiver cable connection to transceiver	Check that the transceiver cable mount is secured.	Ensure that the cable is connected as directed in "9.6 FL-net Network Installation Procedures." Retighten the connection as needed.
		With the installed-transceiver checker, check for abnormalities.	Check the installed transceiver in accordance with the checker instruction manual.
		Check that the transceiver is properly locked.	Properly lock the transceiver in accordance with "9.6 FL-net Network Installation Procedures."
		Check that the transceiver LED is illuminated normally.	Check for a faulty power supply, disconnected power cable, and improper voltage.
	Transceiver cable connection to device	Check that the transceiver cable connection is secured.	Ensure that the cable is connected as directed in "9.6 FL-net Network Installation Procedures." Retighten the connection as needed.
		Check that the TX (transmission) and RX (reception) LEDs on the device glow normally.	Identify the abnormality as directed in "8 TROUBLESHOOTING."
		Check that the media selector switch (SQE, etc.) is properly set.	Review the setting in accordance with "9.6 FL-net Network Installation Procedures."

(2) Network problems and remedies (concerning unstable communications)

Table 8-2 Network Problems and Repairing (Concerning Unstable Communications)

Problem	Inspection item	Inspection procedure	Remedy
Communi-cation cannot be established or is unstable.	Transmission path	Check whether the coaxial cable external conductor is grounded at one point.	Establish the proper ground connection as directed in "9.6 FL-net Network Installation Procedures."
		Check that the AUI cable's shielded wire is properly grounded.	Ground the shielded wire in accordance with manufacturer instructions.
		Check that all the stations properly respond to the Ping command.	Check the power supplies and cables of stations that fail to respond.
		Check whether the collision lamp frequently glows.	Check that the cables and connectors are not in poor contact. With an analyzer, identify abnormalities.
		Check that no more than four repeaters are connected in cascade.	Review the configuration in accordance with "9.6 FL-net Network Installation Procedures."
		Check that each segment is within the specified length limit.	Establish the proper ground connection as directed in "9.6 FL-net Network Installation Procedures."
		Check that two terminal resistors are installed on both ends.	Establish the proper ground connection as directed in "9.6 FL-net Network Installation Procedures."
		Check that the number of connected devices inside each segment is within the specified limits.	Establish the proper ground connection as directed in "9.6 FL-net Network Installation Procedures."
		Check that no more than three segments are connected to the device.	Establish the proper ground connection as directed in "9.6 FL-net Network Installation Procedures."
		Check that the repeater power is ON.	Check for a faulty power supply, disconnected power cable, and improper voltage.
	Communica-ting-station device setup	Check that the network IP address is properly set.	With the support tool and analyzer, ensure that the IP address setting is correct.
		Check that the device station number is properly set.	With the support tool and analyzer, ensure that the station number setting is correct.
		Check that the device parameters are properly set.	With the support tool, ensure that the device parameter settings are correct.
		Check whether the CD (carrier detection) lamp glows steadily or intermittently.	Ensure that the communication cable, AUI power supply, and other items are normal.
		Check whether the TX (transmission) lamp glows steadily or intermittently.	Ensure that the device settings are correct.
		Check whether the LK (link) lamp glows steadily.	Ensure that the device parameter settings are correct.

(3) Checking the IP address with the personal computer's "ping feature"

The connection and IP address setting for the target FL-net device can be checked without using the FL-net network analyzer or any other special tool. Such a check can also be conducted with the "Ping feature" of a general-purpose Windows® 95 personal computer or the like. The table below outlines the IP address checkout procedure that is to be performed with the "Ping feature."

The IP connection can be checked with the "Ping" command. The procedure is outlined below.

① For Windows 95, select [Start] button - [Programs] - [MS-DOS Prompt] to display [MS-DOS Prompt].

② Enter the "Ping" command to conduct a basic communication test for the purpose of checking the communication between the link unit and personal computer. To make a Ping command entry, type in "Ping [IP-address]" or "Ping [host-name]."
Example (IP address): Ping 192.168.250. 13
When the target FL-net device is properly set, the following message is displayed.

```
Pinging 192.168.250. 13 with 32 bytes of data
   Reply from 192.168.250. 13: bytes=32 time=2ms TTL=32
   Reply from 192.168.250. 13: bytes=32 time=1ms TTL=32
   Reply from 192.168.250. 13: bytes=32 time=1ms TTL=32
   Reply from JEMA 192.168.250. 13 : bytes=32 time=1ms TTL=32
C:\WINDOWS>
```

③ If an NG result is obtained (the connection is not verified), the following timeout indication is displayed.

```
Pinging 192.168.250. 13 with 32 bytes of data:
   Request timed out.
   Request timed out.
   Request timed out.
   Request timed out.
C:\WINDOWS>
```

8.3 Precautions for FL-net Use

For the FL-net transmission path requirements, see the aforementioned section or IEEE 802.3 standard. In addition to such requirements, you must observe the FL-net-specific precautions.

- Ensure that no other Ethernet communication data flows along the FL-net communication cable.
- Do not connect the FL-net to a router.
- The use of a switching hub for the FL-net does not produce any beneficial effect.
- The use of infrared, radio, or like medium may substantially decrease the real-time capability of communications.
- When you use a personal computer, the real-time capability of communications may substantially change depending on the personal computer capacity and the employed OS and applications.
- Use a specified IP address only. The same network address must be used (the standard network address is 192.168.250). For the node number (station number) for the IP address, the following input range is recommended. When a node number is set, it is not checked for duplication until it is actually used for communication. If a duplicate node number is used for communication, a node number duplication error occurs. Exercise care not to set a duplicate node number.

Network address	Node number
192.168.250.	1 to 249

- Make a proper ground wire connection. Ensure that the employed ground wire has a sufficient thickness.
- Ensure that the FL.NET is positioned at an adequate distance from a noise source. Also, avoid installing the FL-net together with a mains power line or the like.
- When cyclic data communication and message data communication are simultaneously effected, the real-time capability may deteriorate depending on the data amount and the like.
- Cyclic data communication areas (common memory areas) need not contiguously be allocated.
- When the transceiver is equipped with an SQE switch, set it up in accordance with its instruction manual.
- The regular communication capability depends on the processing capacities of the connected devices. The entire communication is effected with the communication processing speeds of all the networked devices adjusted for the slowest device's communication processing capability (minimum permissible frame interval). Therefore, the connection or addition of a single device may significantly deteriorate the real-time capability of the entire system.
- Although the header section of message data communications is in big-endian format, the data section is in little-endian format. Note, however, that the system parameters in the data section for a profile read are in big-endian format (the use of the big-endian format transmits the MSB first).

8.4 Error Indications and Repairing

(1) Tool's error indications
For error indications given by the tool, refer to "SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139)."

(2) CPU module error indications
The CPU module indicator readouts vary depending on whether they are related to the main module or submodule as shown in Table 8-3.

Table 8-3 CPU Module Indicator Readouts

Module setting	Readout	Description	Remedy
Main	FL-M @. @	The FL.NET module (main) has participated normally in the network.	This readout does not indicate any error.
	FLNM□□□□	A board abnormality is detected in the FL.NET module (main).	See the subsequent instructions concerning the hardware errors or FA protocol errors.
Sub	FL-S @. @	The FL.NET module (sub) has participated normally in the network.	This readout does not indicate any error.
	FLNS□□□□	A board abnormality is detected in the FL.NET module (sub).	See the subsequent instructions concerning the hardware errors or FA protocol errors.

• The "@.@" portion represents the version and revision of the FL.NET module.
• The "□□□□" portion represents the on-screen indication of a hardware error or FA protocol error.

[FA protocol errors]
When an error stated in the FA protocol is detected during an FL.NET module operation, the CPU module indicator shows a message as explained in Table 8-4.
The FL.NET module may stop running in compliance with the FA protocol, depending on the contents of the displayed message.

Table 8-4 Details of CPU Module Indications Given upon FA Protocol Error Occurrence

On-screen message	Error description	Remedy	Remarks
LER	Network participation not completed.	The FL.NET module has not participated in the network. Follow applicable troubleshooting instructions.	Processing is being performed to ensure that the module participates in the network.
PER	FL.NET module setting error.	The FL.NET module is improperly set. Check the settings in the module with the FL.NET setup tool. If any abnormal setting is found, correct it. If the error recurs after the setting is corrected, it is conceivable that the FL.NET module may be defective. Replace the module.	
ADBL	Duplicate common memory settings.	Common memory settings for the self-node are the same as for another node. Compare the self-node common memory settings with those of the other nodes, and make necessary corrections to ensure there are no duplicate node settings.	The common memory settings for the self-node are all 0. With “FL.NET For Windows®,” set up the self-node again.
NDBL	Duplicate node numbers.	A node existing in the network uses the same node number as the FL.NET module. Confirm the node number settings for the other nodes and ensure that there are no duplicate node number settings.	Network participation is halted in compliance with the FA link protocol. Turn the power OFF and then back ON or make a participation request with “FL.NET For Windows®” and then participate in the network again.
TABT	Token hold timeout.	The preselected token hold time limit has been exceeded more than three times repeatedly. It is conceivable that the FL.NET module may be defective. Replace the module.	Network participation is halted in compliance with the FA link protocol.

[Hardware errors]

When the FL.NET module detects a hardware error, the CPU module indicator shows an error message in accordance with Table 8-5. The FL.NET module also illuminates the error LED and collects error freeze information. The FL.NET module operation comes to a stop.

Table 8-5 Details of CPU Module Indications Given upon Hardware Error Occurrence

On-screen message	Error description	Remedy
BUS	Bus error	It is conceivable that the FL.NET module may be defective. Replace the module.
ADDR	Address error	
ILLG	Invalid instruction	
ZERO	Division by zero	
PRIV	Privilege violation	
FMAT	Format error	
SINT	Spurious interrupt	
EXCP	Unused exception	
PTY	Parity error	
MDSW	Module No. setting switch error	Check the module No. setting switch.
ROM1	ROM1 sum error	It is conceivable that the FL.NET module may be defective. Replace the module.
RAM1	RAM1 compare error	
RAM2	RAM2 compare error	
ROM3	ROM3 sum error	
IPNG	IP address not registered	Register the IP address.
MAC	MAC address not registered	It is conceivable that the FL.NET module may be defective. Replace the module.
PRG	Microprogram error	

Note: If an FA protocol error and hardware error simultaneously occur, only a hardware error indication is given.

When the FL.NET module detects a hardware error, it illuminates the LER lamp and saves error freeze information. The FL.NET module operation comes to a stop.

Table 8-6 Details of Error Freeze Information Table

Main module	Submodule	2^{31} ——— 2^{16}	2^{15} ——— 2^{0}
/D40400	/DC0400	Error code	———
/D40404	/DC0404	———	
/D40410	/DC0410	D0 register	
/D40414	/DC0414	D1 register	
/D40418	/DC0418	D2 register	
/D4041C	/DC041C	D3 register	
/D40420	/DC0420	D4 register	
/D40424	/DC0424	D5 register	
/D40428	/DC0428	D6 register	
/D4042C	/DC042C	D7 register	
/D40430	/DC0430	A0 register	
/D40434	/DC0434	A1 register	
/D40438	/DC0438	A2 register	
/D4043C	/DC043C	A3 register	
/D40440	/DC0440	A4 register	
/D40444	/DC0444	A5 register	
/D40448	/DC0448	A6 register	
/D4044C	/DC044C	A7 register	
/D40450	/DC0450	Stack frames (4 words, 6 words, bus error)	
/D404FC	/DC04FC		

Code	Error description
0010H	Bus error
0011H	Address error
0012H	Invalid instruction
0013H	Division by zero
0014H	Privilege violation
0016H	Format error
0017H	Spurious interrupt
0018H	Unused exception (CHK, TRAPV, L1010, etc.)
0019H	Parity error
001AH	Power failure notice
0100H	Module No. setting switch error
0102H	ROM1 sum error
0103H	RAM1 compare error
0105H	RAM2 compare error
010BH	ROM3 sum error
0113H	IP address not registered
0114H	MAC address error

Note: Stack frame detailed descriptions are given on the next page.

Figure 8-1 shows the details of the stack frames in the error freeze information table.

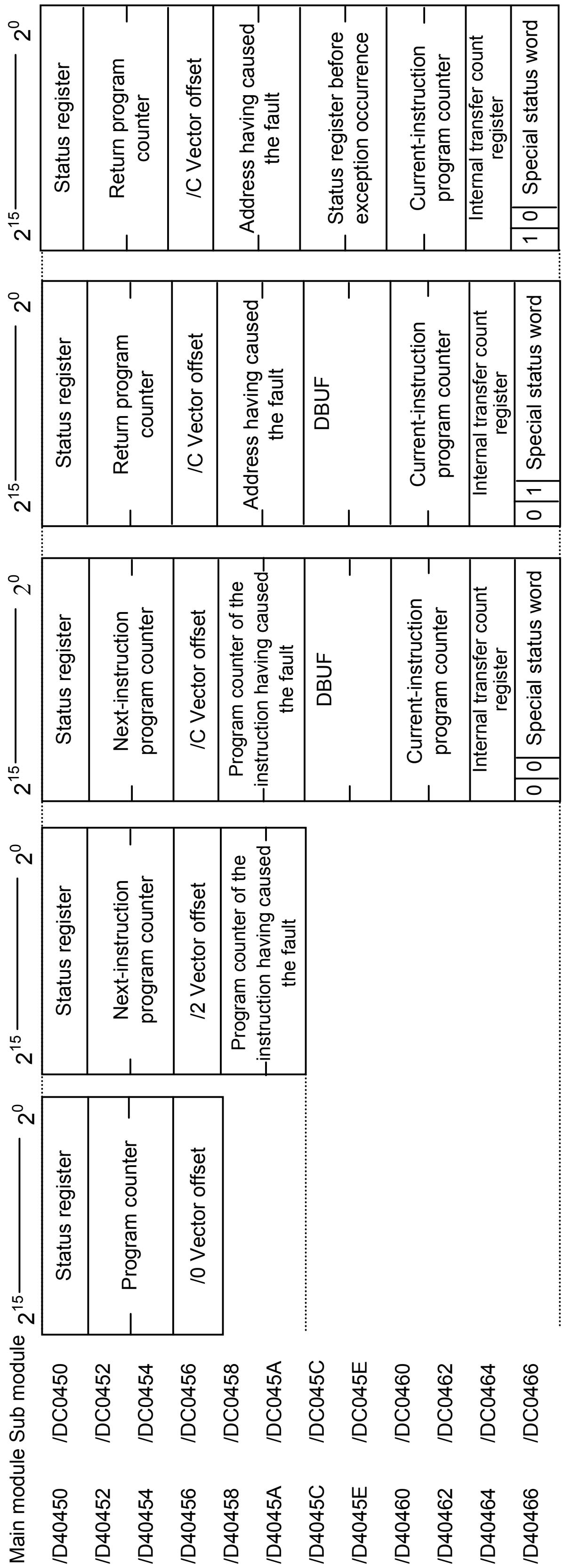


Figure 8-1 Details of Stack Frames in Error Freeze Information Table

[Code table of errors that can be detected by C mode handler and mathematical/logical function]
Table 8-7 shows error codes and remedies concerning the errors that may occur when the C mode handler or mathematical/logical function issues a request to the FL.NET module.

Table 8-7 List of Detectable Codes

(1/2)

Error code	Description	Cause	Remedy
0x0000	Normal end of message handling	――――――	――――――
0x0001	Message response error	An abnormal response message is received from a specified node number.	The contents of the abnormal response message are stored in the error message storage table. Refer to the instruction manual for the specified node and check the status of the specified node.
0x0002	Message support not provided	The specified node does not support the user-requested message feature.	Do not issue any unsupported message to the node.
0xFE00	Parameter error	A user-specified parameter is abnormal. If a transparent message reception request has been issued, the associated transparent message is not received.	Check the parameters used when the request was issued. Do not initiate a transparent message reception until message receptions are verified.
0xFE01	Self-node not connected	The FL.NET module has not participated in the network.	Issue a request after the FL.NET module has participated in the network.
0xFE02	Specified node not connected	No node having a user-specified node number has participated in the network.	Specify the node number of a node that has participated in the network.
0xFE03	Message processing already in progress	The newly issued request cannot be accepted because the previously issued request is being processed.	Issue the new request again after the processing of the previously issued request terminates.
0xFE04	Message ACK response not received	An ACK response is not received from the node having a specified node number.	It is conceivable that the module may be defective. Replace the module.
0xFE06	No data received	No response to a message request was received within 30 seconds after message request issuance to a specified node number.	It is conceivable that the module may be defective. Replace the module.

(2/2)

Error code	Description	Cause	Remedy
0xFE08	ACK reception sequence number error	An ACK response returned from a specified node number reported a sequence number error.	It is conceivable that the module may be defective. Replace the module.
0xFE09	ACK reception sequence number version error	An ACK response returned from a specified node number reported a sequence number version error.	
0xFE12	Message queue full	The message queue for a specified node number is full. The specified node number cannot receive a request.	Reissue the request after a while or decrease the number of requests to the specified node number.
0xFE13	Initialization error	Message processing initialization is not completed for a specified node number.	Reissue the request after a while.
0xFE16	Message size error	A specified node number reported that an abnormal message size was requested by the self-node.	It is conceivable that the module may be defective. Replace the module.
0xF0XX or 0xFFXX	Driver abnormal	An abnormality was detected by a driver when a user-requested message was transmitted.	

[Error message data table]
When a response message for a message request from the self-node is an abnormal response message, the message data is stored in the error message data table within the FL.NET module. The specification for the error message table is given below:

Main module	Submodule	2^{15} … 2^{0}
0xD41380	0xDC1380	Total number of error messages
0xD41382	0xDC1382	Transmission source node number
0xD41384	0xDC1384	Received TCD
0xD41386	0xDC1386	Error message word count (in bytes)
0xD41388 – 0xD41788	0xDC1388 – 0xDC1788	Error code data section

Item	Detail
Total number of error messages	Total number of abnormal response messages received after power ON.
Transmission source node number	Node number of the source of transmission of an abnormal response message
Received TCD	TCD number of an abnormal response message
Error message word count	Data section size (error code size) of an abnormal response message. It is displayed in bytes.
Error code data section	This area is used to store the data section (error code) of an abnormal response message. It can hold up to 1,024 bytes of data.

Note: If an abnormal response message is received when another abnormal response message is already stored in the error message table, the total error message count is incremented (by one) and the contents of the error message are overwritten by those of the newly received one.

(3) Module error indications

When an error occurs in the FL.NET module, the LER LED on the module turns ON.

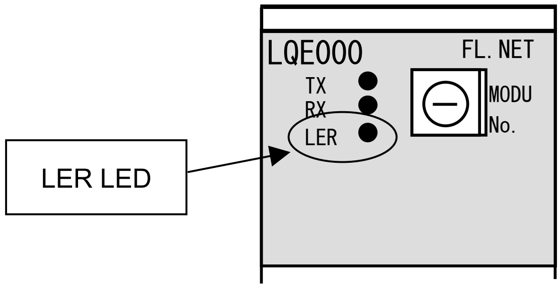


Figure 8-2 Module Error Lamp

Note that the LER LED turns ON even when the module has not participated in the network. Therefore, the LER LED indication is not adequate for judging whether the module is abnormal.

When the LER LED turns ON, note the CPU module indicator on the S10mini because it furnishes relevant detailed information.

For details on indicator readouts, see “(2) CPU module error indications.”

(4) Communication driver error alarm

The FL.NET module does not indicate an error alarm that is issued by a driver.

(5) Viewing the communication log data

You can view the communication log maintained in the module with the setup tool named “FL.NET For Windows®.”

For detailed operating procedure, refer to “SOFTWARE MANUAL OPTION FL.NET For Windows® (manual number SAE-3-139).”

9 APPENDIXES

9.1 System Configuration Guide

9.1.1 Ethernet overview

Ethernet is a LAN (Local Area Network) standard for communications among personal computers, printers, and other devices. It defines communication data formats, cables, connectors, and communication-related items. The Ethernet standard is established by the Ethernet Working Group (IEEE 802.3). So far, the 10BASE-5, 10BASE-2, 10BASE-T, and other system requirements are stipulated. At present, the requirements for 1000BASE-T and other new systems are being studied. Figure 9-1 shows the standardization scheme that the IEEE 802.3 has followed.

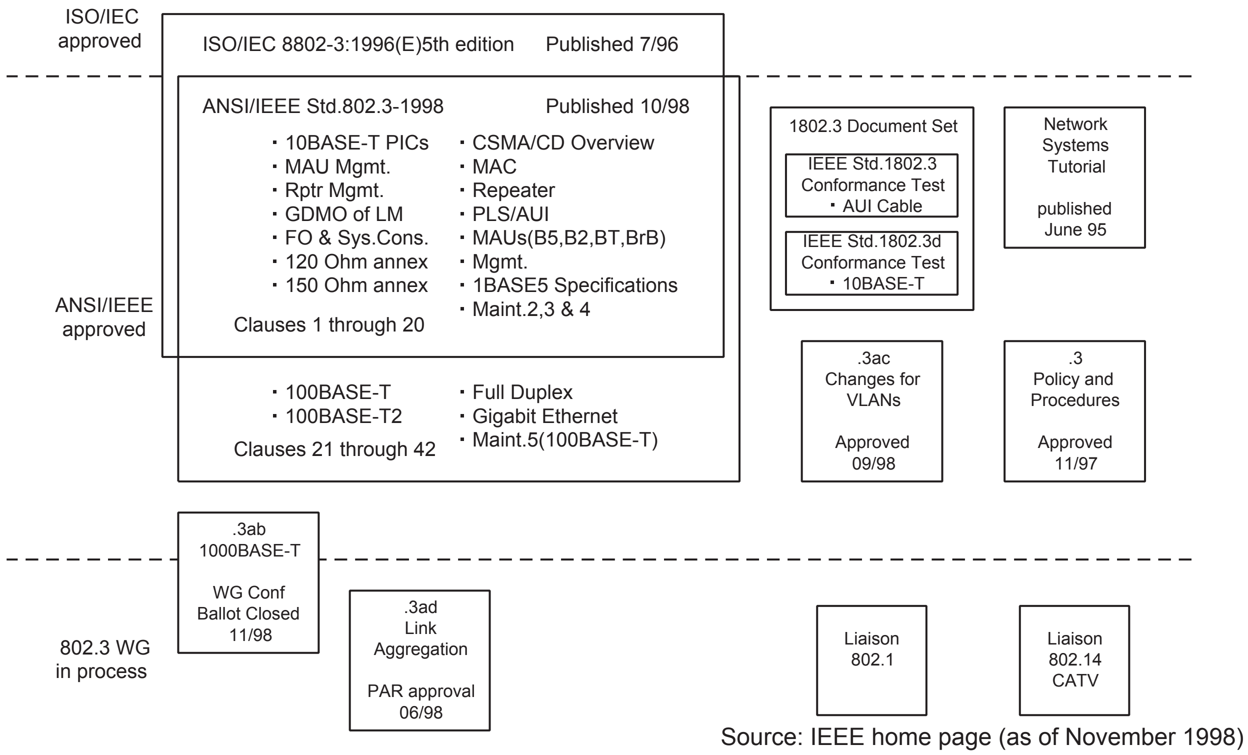


Figure 9-1 Scheme of Standardization by IEEE 802.3

9.1.2 10BASE-5 specifications

10BASE-5 is an Ethernet connection method that uses a coaxial cable about 10 mm in thickness (also referred to as a thick cable or yellow cable). The number “10” of the designation “10BASE-5” represents a transmission speed of 10 Mbps. The “BASE” portion indicates the use of a baseband transmission system. The suffix “5” indicates that the trunk transmission distance is 500 m. For the connection to a personal computer or like device, a transceiver is attached to a coaxial cable and connected to the personal computer via a transceiver cable (also referred to as an AUI cable).
The 10BASE-5 connection method uses a thick cable that cannot easily be used to establish a network connection. Therefore, it is not frequently used for office networks. However, it is commonly used for a trunk network connection because of its long transmission distance.
Figure 9-2 shows a typical 10BASE-5 Ethernet configuration.

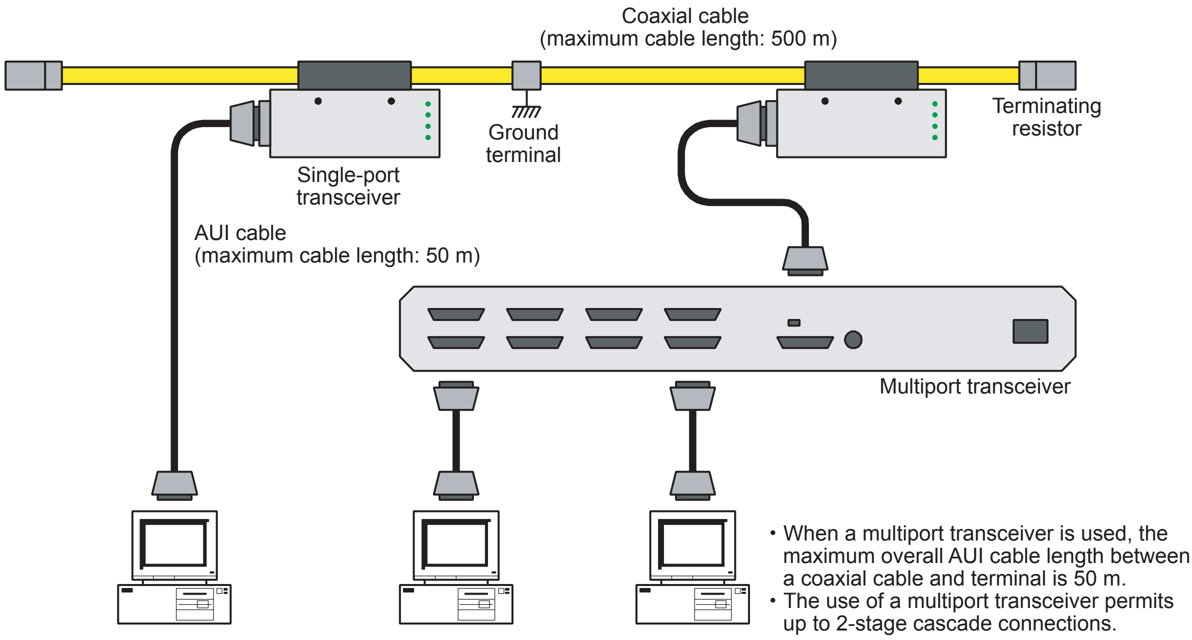


Figure 9-2 10BASE-5 Ethernet Configuration Example

9.1.3 10BASE-T specifications

10BASE-T is an Ethernet connection method that uses a twisted-pair cable. The number “10” of the designation “10BASE-T” represents an Ethernet transmission speed of 10 Mbps. The “BASE” portion indicates the use of a baseband transmission system. The suffix “T” indicates that a twisted-pair cable is used as a transmission medium. In a 10BASE-T network, personal computers and other devices need to be interconnected in a star topology via a hub. They cannot be directly connected to each other. (Although the use of a special cable called “cross cable” allows devices to be interconnected on a one-to-one basis, it is not commonly applied.) The maximum permissible cable length between a hub and devices is 100 m.

The 10BASE-T connection method uses a thin cable that can be routed with ease. It also permits various devices to be connected to and disconnected from a network on an individual basis. Therefore, it is frequently used for office networks.

Figure 9-3 shows a typical 10BASE-T Ethernet configuration.

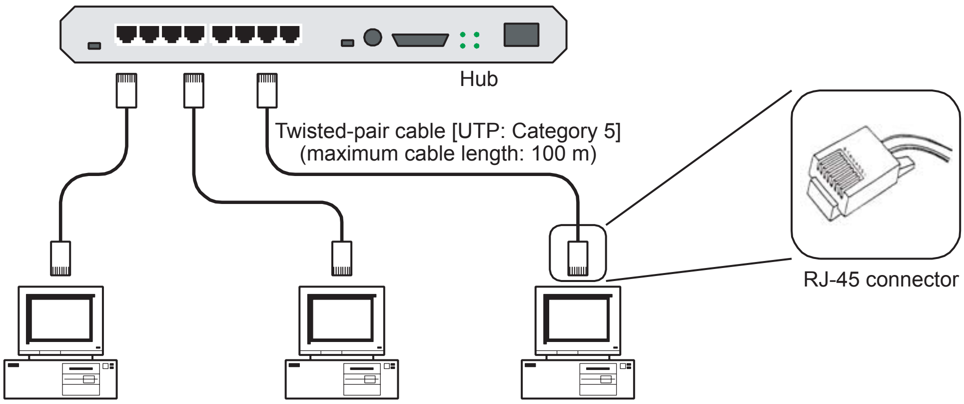


Figure 9-3 10BASE-T Ethernet Configuration Example

9.1.4 Other Ethernet specifications

(1) 10BASE-2

10BASE-2 is an Ethernet connection method that uses a coaxial cable about 5 mm in thickness (also referred to as a thin cable). The number "10" of the designation "10BASE-2" represents a transmission speed of 10 Mbps. The "BASE" portion indicates the use of a baseband transmission system. The suffix "2" indicates that the trunk transmission distance is 185 m (≒ 200 m). For the connection to a personal computer or like device, a T-shaped branch connector is attached to the BNC connector on a device, and coaxial cables are connected to both ends of the branch connector. Figure 9-4 shows a typical 10BASE-2 Ethernet configuration.

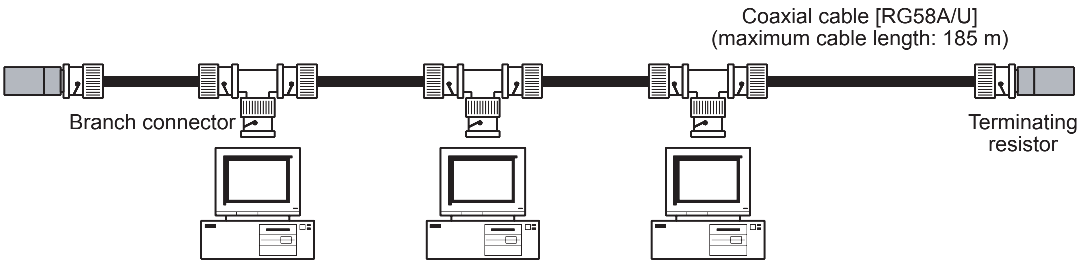


Figure 9-4 10BASE-2 Ethernet Configuration Example

(2) Optical Ethernet

Optical Ethernet uses a fiber-optic cable as the transmission medium. It is used in systems where 500 m or longer distance transmission and increased noise immunity are called for. Optical Ethernet connection methods defined by the IEEE 802.3 standard are 10BASE-FP, 10BASE-FB, 10BASE-FL, 100BASE-FX, 1000BASE-LX, and 1000BASE-SX. Figure 9-5 shows a typical optical Ethernet configuration.

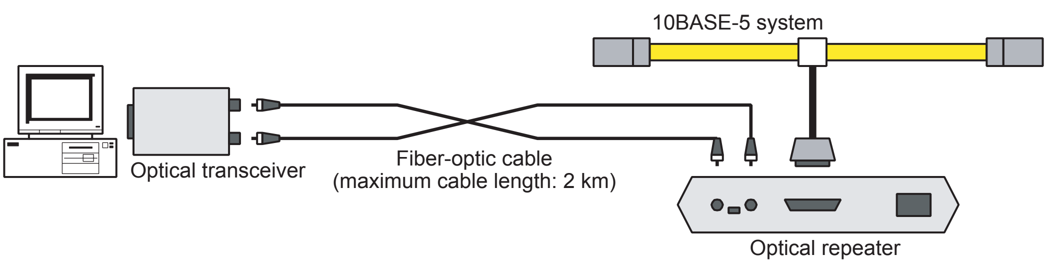


Figure 9-5 Optical Ethernet Configuration Example

(3) Wireless Ethernet

A wireless LAN uses radio waves or infrared rays as the transmission medium. It is used, for instance, to connect a mobile device to a LAN. Wireless LAN is being standardized by the IEEE Wireless LAN Working Group (IEEE 802.11). For the interconnection between a wireless LAN and an Ethernet network, the use of a bridge is required because they differ in MAC layer protocol.

Figure 9-6 shows a typical wireless Ethernet configuration.

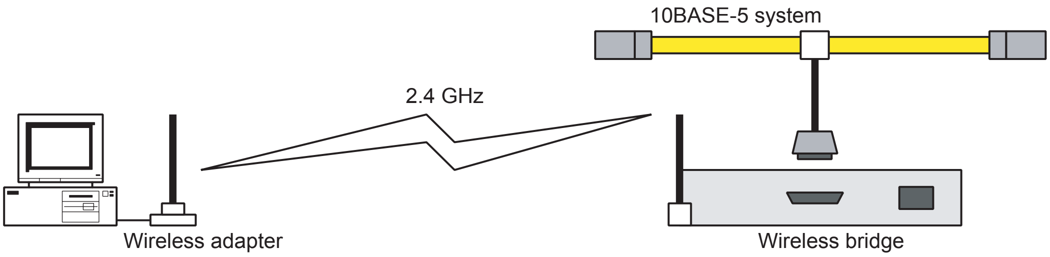
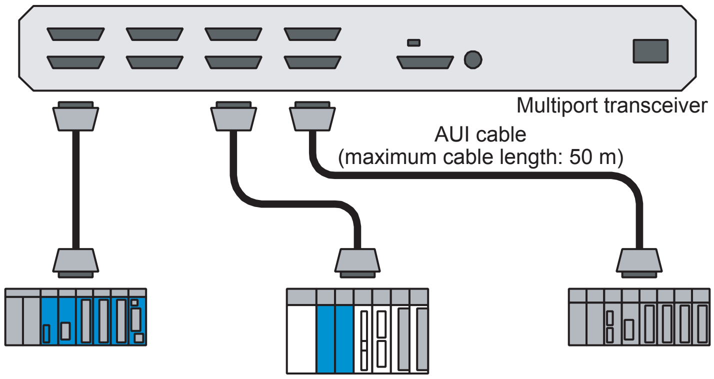


Figure 9-6 Wireless Ethernet Configuration Example

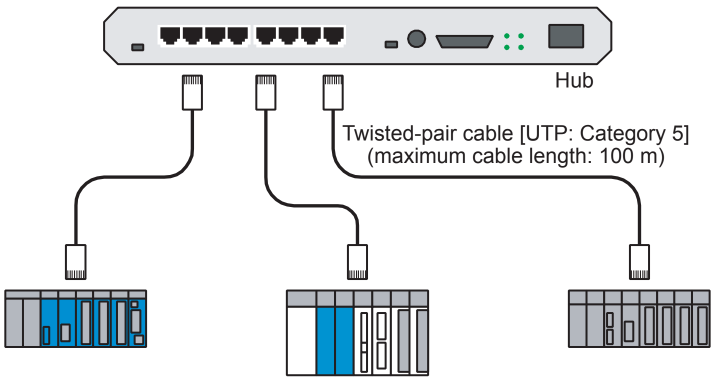
9.2 System Configuration Examples

9.2.1 Small-scale configuration

A network system consisting of several units of devices can be configured with one multiport transceiver or hub.



(a) Use of a multiport transceiver



(b) Use of a hub

Figure 9-7 Small-scale Configuration Example

9.2.2 Basic configuration

A network system consisting of dozens of units of devices can be configured by connecting several multiport transceivers or hubs to one coaxial cable.

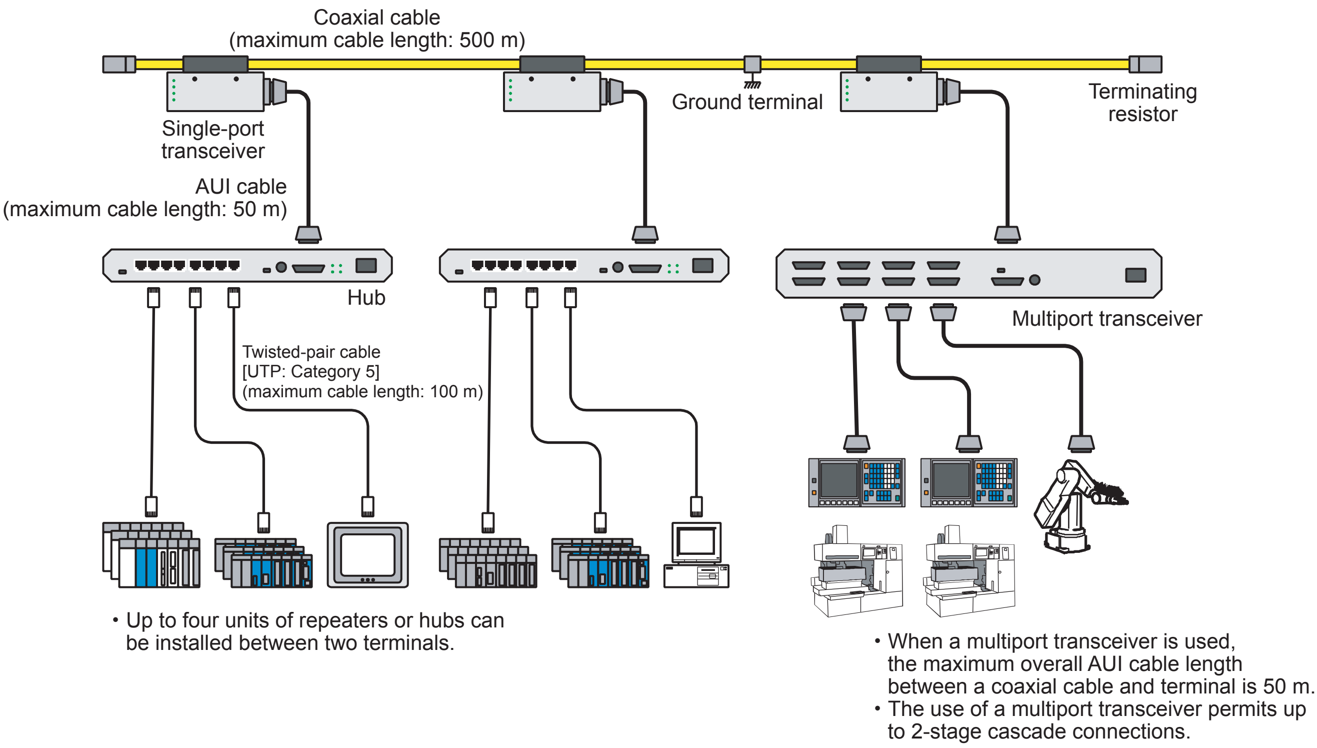


Figure 9-8 Basic Configuration Example

9.2.3 Large-scale configuration

A network system consisting of several hundred units of devices can be configured by connecting 10BASE-5 network segments with repeaters.

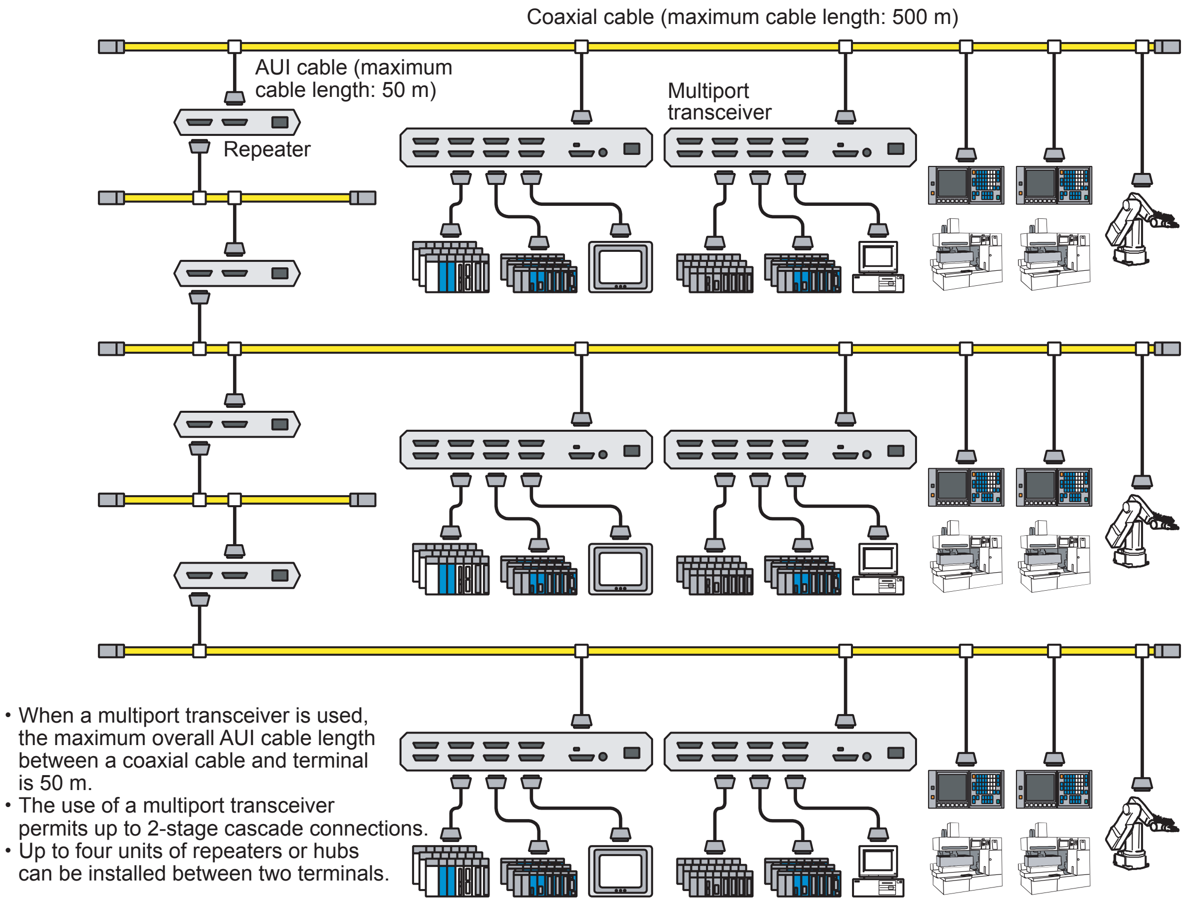


Figure 9-9 Large-scale Configuration Example

9.2.4 Long-distance distributed configuration

If the distance between network segments in a large-scale network system exceeds the 10BASE-5 transmission distance limit (500 m), you can connect network segments with optical repeaters to establish a network system whose inter-repeater distance is 2 km.

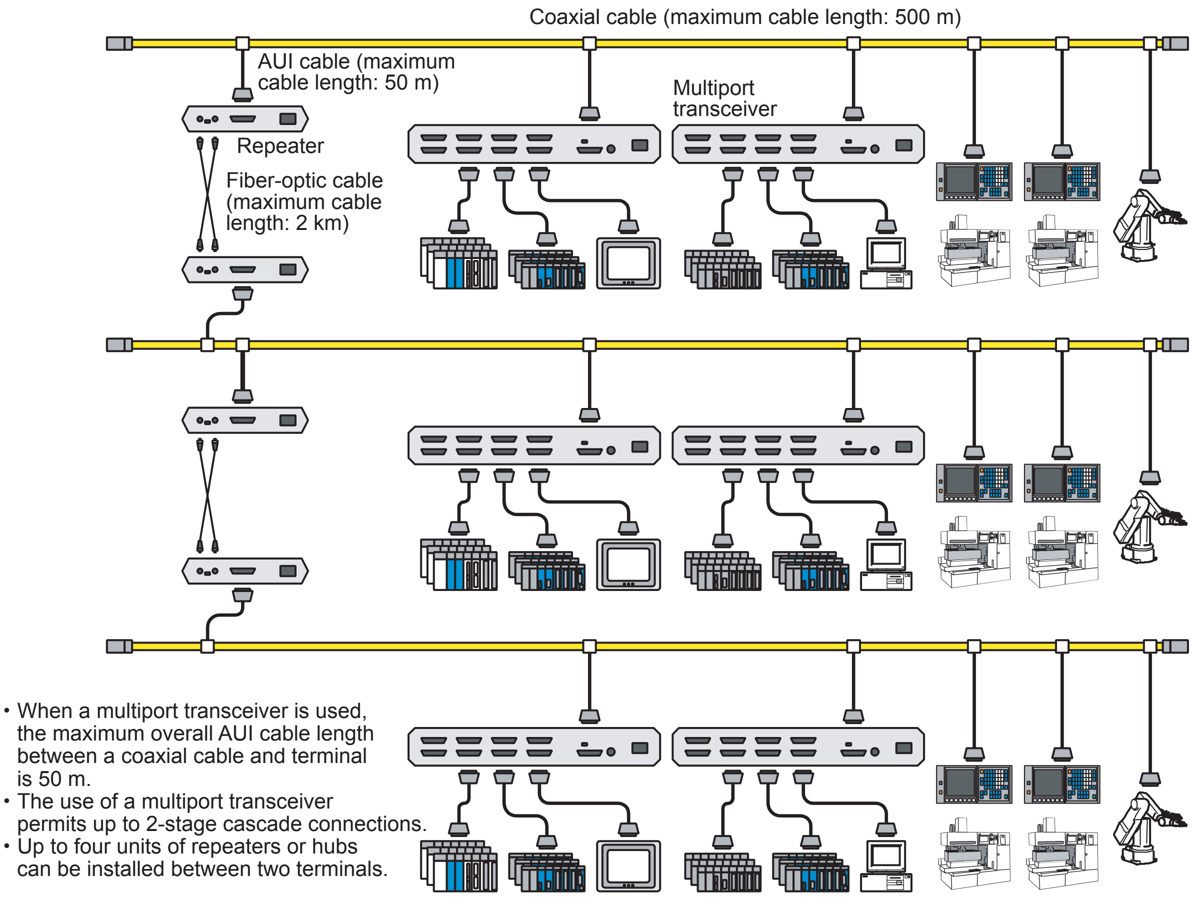


Figure 9-10 Long-distance Distributed Configuration Example

9.2.5 Locally concentrated configuration

When dozens of units of devices are locally concentrated, a network system can be configured with stackable hubs.

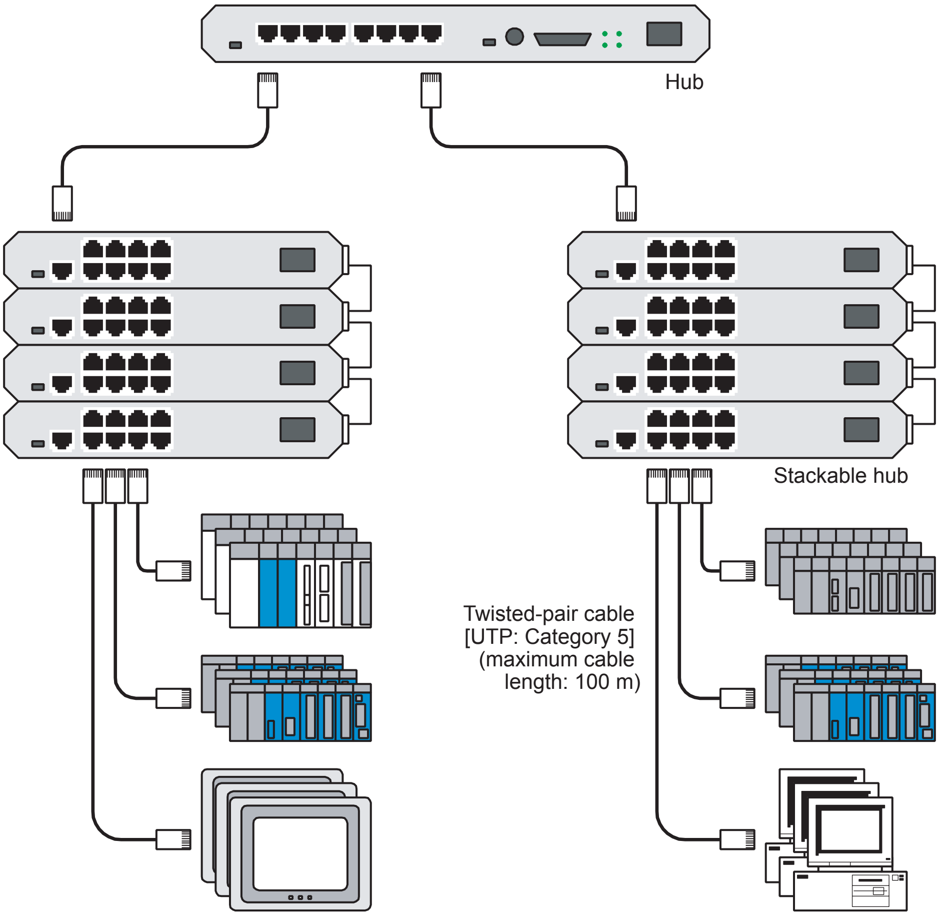


Figure 9-11 Locally Concentrated Configuration Example

9.2.6 Locally and widely distributed configuration

When a certain controller in a network system having a basic configuration is positioned at a distance or there is a high-voltage power supply or noise source near a network, a long-haul network having an excellent noise immunity can be configured by dividing the network into two segments and interconnecting them with an optical repeater.

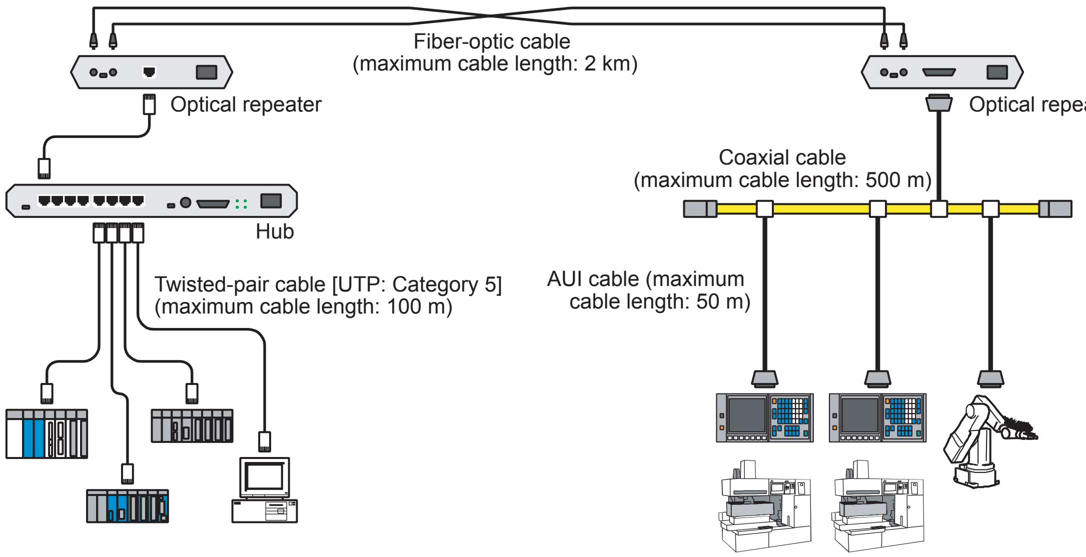


Figure 9-12 Locally and Widely Distributed Configuration Example

9.2.7 Concept of FL-net system

The objective of the FL-net is to establish real-time communication among production system controllers such as programmable controllers, robot controllers, and computer numerical control devices.

FL-net establishes a broadcast-based token passing mechanism on the UDP/IP protocol of Ethernet and uses it to effect cyclic communication and message communication.

9.2.8 Differences between general-purpose Ethernet and FL-net

- Since the FL-net is a network for FA (factory automation) fields, it cannot use all the general-purpose Ethernet devices. Some devices are inapplicable to the FL-net due to their noise immunity or environment resistance.
- The FL-net is required to offer a specified degree of real-time communication response capability for control purposes. Therefore, it can be connected to FL-net-compliant controllers and other control devices only.
- The FL-net employs a cyclic communication method that exercises the broadcast features of 10BASE-5/10BASE-T based UDP/IP communication. Therefore, the following limitations are imposed due to the current standard.
  - Currently supported devices are limited to 10 Mbps Ethernet LAN devices.
  - Connections to the other general-purpose Ethernet networks cannot be established.
  - The TCP/IP communication features are not supported.
  - The use of a switching hub produces no beneficial effect.
  - The inability to function may result if a router or the like is used.

9.3 Network System Definitions

9.3.1 Standard compliance of communication protocol

A communication protocol is a set of rules (communication regulations) to enable a system to exchange information with another system through a communications link. The communication protocols adopted by the FL-net comply with the following standards:

Table 9-1 FL-net Communication Protocols

FL-net communication protocol	Standard complied with
FL-net	FA Link Protocol Specification (issued by the Control Network Expert Committee of MSTC FA Open Systems Promotion Group)
UDP	RFC768
IP ICMP, etc.	RFC791,792,919,922,950
ARP, etc.	RFC826,894
–	IEEE802.3

9.3.2 Communication protocol hierarchical structure

A communication protocol is modeled in hierarchical form to express and standardize a communication process by systematically dividing it into various levels. The FL-net consists of the following six protocol layers.

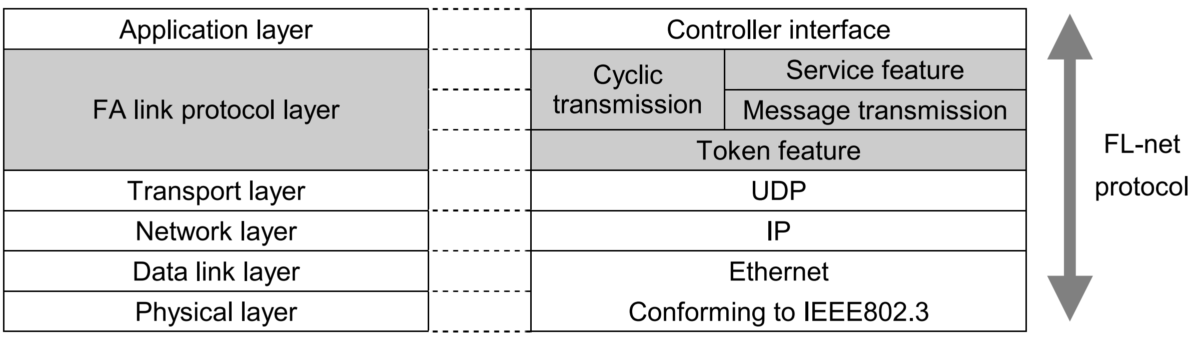


Figure 9-13 FA Link Protocol Definition

9.3.3 FL-net physical layer

For a transmission speed of 10 Mbps, the Ethernet physical layer offers five different transmission methods: 10BASE-5, 10BASE-2, 10BASE-T, 10BASE-F, and 10BROAD36 (this last one is not commonly used). Note that 100 Mbps Ethernet also exists.
The FL-net adopts 10BASE-5 (recommended), 10BASE-2, and 10BASE-T.

9.3.4 IP address

An address named "IP address (INET address)" is used to differentiate a specified communication device from many other communication devices connected to Ethernet. Therefore, all the communication devices connected to Ethernet must have unique IP addresses.
The IP address is divided into two sections. One section shows a network address to which a communication device is connected. The other section indicates a host address for the communication device. Three different network classes (A, B, and C) are used depending on the network size (Classes D and E are additionally used for special purposes).

Table 9-2 IP Address Classification

	First 1-octet value	Network address section	Host address section
Class A	0 to 127	xxx.xxx.xxx.xxx	xxx.xxx.xxx.xxx
Class B	128 to 191	xxx.xxx.xxx.xxx	xxx.xxx.xxx.xxx
Class C	192 to 223	xxx.xxx.xxx.xxx	xxx.xxx.xxx.xxx

Shaded "xxx" portions represent the associated address sections.

The IP addresses of communication devices connected to a network have the same network address section and a unique host address section that is not duplicated within the network.
The FL-net uses a Class C IP address. It is recommended that you set the network address to "192.168.250.N" (N is a node number between 1 and 254). It is also recommended that the low-order host address coincide with an FL-net protocol node number.

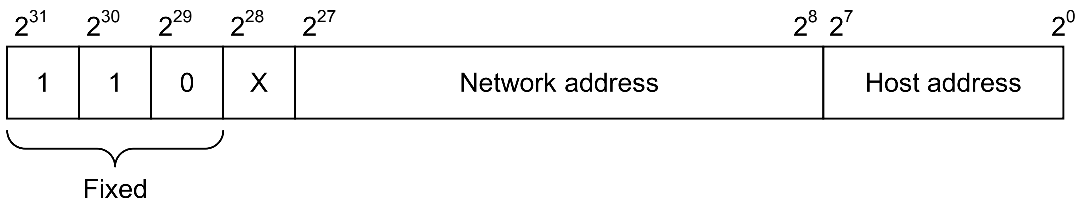


Figure 9-14 FL-net IP Address

9.3.5 Subnet mask

In compliance with the FL-net protocol, the subnet mask is fixed at 255.255.255.0. The FL-net user must always use a subnet mask setting of 255.255.255.0. This value has the same network address section and host address section as for Class C.

9.3.6 TCP/IP and UDP/IP communication protocols

TCP, UDP, and IP are important protocols that Ethernet uses.
IP corresponds to the communication protocol's network layer and controls the flow of communications data.
TCP and UDP correspond to the transport layer. They both use IP as a network layer but greatly differ in the contents of services.
TCP offers highly reliable services without causing a higher layer to be aware of data segments.
UDP, on the other hand, functions to transmit a datagram (one pack or unit of information) from IP to a higher layer, but does not guarantee the delivery of data to a transmission destination. It lets a higher layer perform data acknowledgment, retransmission, and other processes.
UDP is not as reliable as TCP. However, it offers communication services with a minimum of overhead.
The FL-net uses UDP. The reason is that TCP's complicated data verification/retransmission procedure is redundant for the FL-net. The FL-net provides an increased data exchange speed by skipping such a complicated procedure and having the FL-net protocol layer on a higher level perform token-based transmission right management, multiple-frame division/synthesis, and other relevant processes.

9.3.7 Port numbers

For the FL-net, the following port numbers are predetermined to ensure that services are implemented by the FL-net protocol layer, which is positioned higher than the transport layer. Note, however, that the FL-net user does not have to set these port numbers with parameters or the like.

Table 9-3 FL-net Port Numbers

Name	Port number
Cyclic transmission port number	55000 (fixed)
Message communication port number	55001 (fixed)
Enter request frame port number	55002 (fixed)
Transmission port number	55003 (fixed)

9.3.8 FL-net data format

(1) FL-net data format overview
The data transmitted/received by the FL-net is capsuled in each communication protocol layer as indicated below:

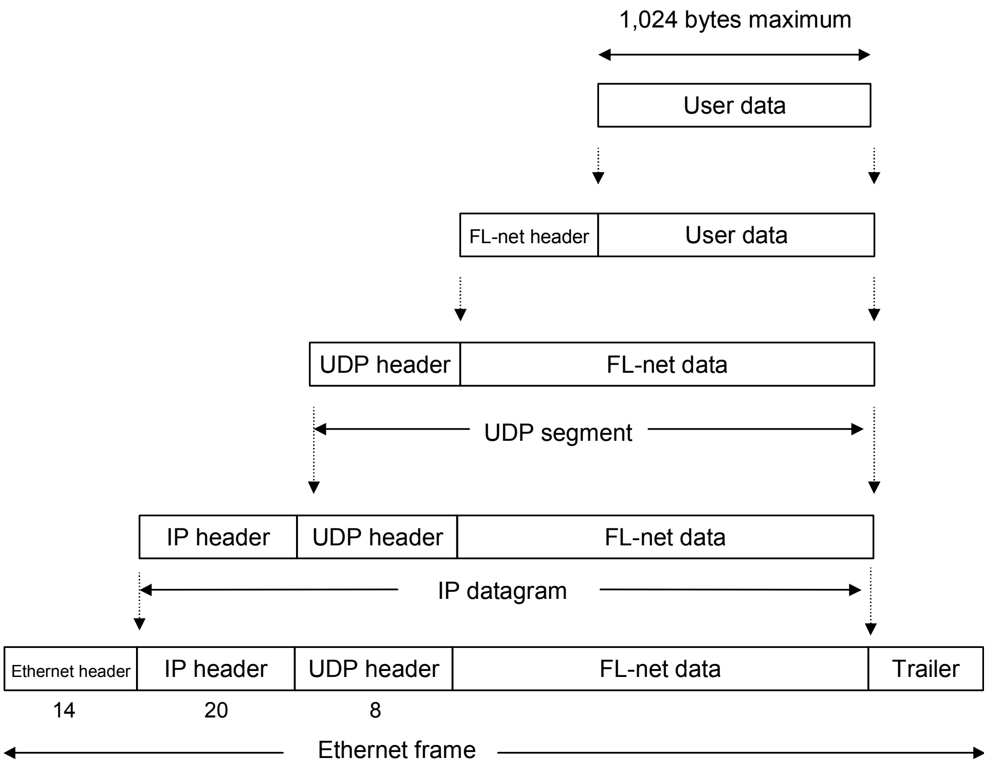


Figure 9-15 FL-net Data Format Overview

FL-net data (one-frame) observable over a communications line is indicated in the example below. In this example, 128-byte cyclic data is transferred.

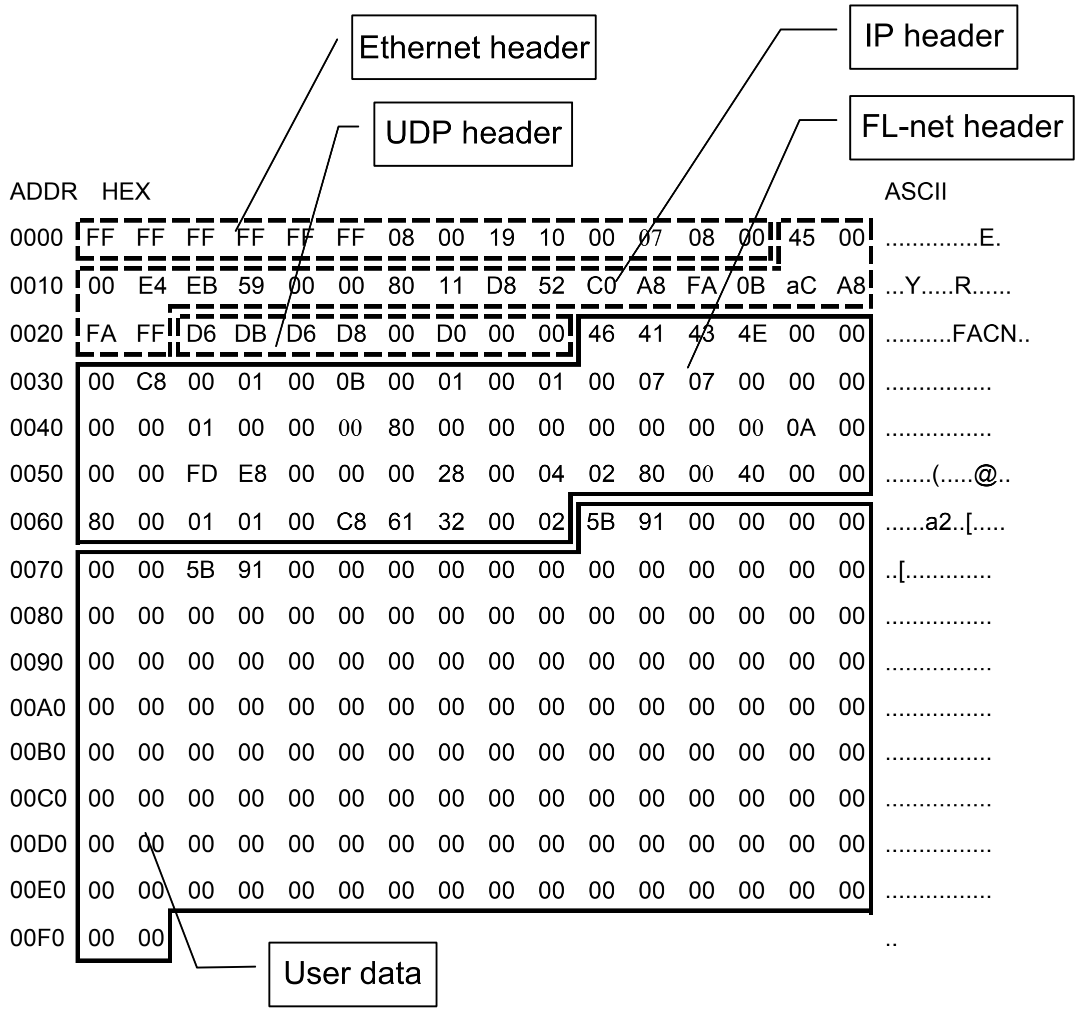


Figure 9-16 FL-net Data (One-frame) Example

(2) FL-net header format

The FL-net header consists of 64 or 96 bytes.

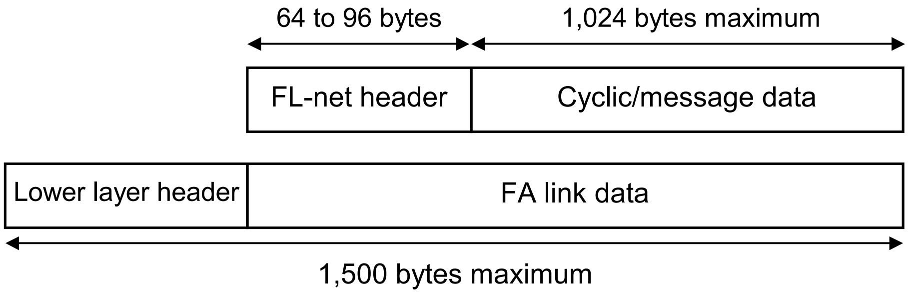


Figure 9-17 FL-net Header

The FL-net header is attached to the beginning of every frame related to the FL-net protocol.

9.3.9 FL-net transaction codes

The FL-net offers the following message transmission services:

- Word block data read
- Word block data write
- Network parameter read
- Network parameter write
- Stop directive (support for request only)
- Run directive (support for request only)
- Profile read (support for response only)
- Log data read
- Log data clear
- Message return
- Transparent type

The header of each message contains a request transaction code or response transaction code that provides message frame identification.

Table 9-4 List of Transaction Codes

Transaction code	Description
0 to 59999	Transparent message
60000 to 64999	Reserve
65000	Cyclic header (with token)
65001	Cyclic header (without token)
65002	Enter request frame header
65003	Byte block data read (request)
65004	Byte block data write (request)
65005	Word block data read (request)
65006	Word block data write (request)
65007	Network parameter read (request)
65008	Network parameter write (request)
65009	Stop directive (request)
65010	Run directive (request)
65011	Profile read (request)
65012	Trigger header
65013	Log read (request)
65014	Log clear (request)
65015	For message return testing (request)
65016 to 65202	Reserve (for future extension)
65203	Byte block data read (response)
65204	Byte block data write (response)
65205	Word block data read (response)
65206	Word block data write (response)
65207	Network parameter read (response)
65208	Network parameter write (response)
65209	Stop directive (response)
65210	Run directive (response)
65211	Profile read (response)
65212	Reserve
65213	Log data read (response)
65214	Log data clear (response)
65215	For message return testing (response)
65216 to 65399	Reserve (for future extension)
65400 to 65535	Reserve

9.4 FL-net Network Management

9.4.1 FL-net token management

(1) Token

Under normal conditions, a node can transmit data while it holds the token. A node without the token can transmit only two items: a token reissue request in the event of a token monitoring timeout and an enter request frame for situations where network enter is not completed.

- The FA net circulates one token among nodes.
- Upon receipt of the token, a node holds the right to transmit data to the network until it passes the token to the next node.
- The token circulates among all nodes that have entered in the FL-net.
- The token permits a transmission with cyclic data included.
- The token can be circulated with no data attached to it.
- The token is monitored by a timer in such a manner that it is automatically reissued if it stops circulating through the network for a predetermined period of time.
- If two or more tokens should exist within the network, a unification feature works so that only one token prevails.

(2) Flow of token

Basically, only one token exists within the network.

If two or more tokens should exist within the network, the one having the lowest destination node number takes precedence with the others discarded.

A frame having a token (token frame) has a token's destination node number and a token transmission node number.

When a node matches the token's destination node number contained in the received token frame, it becomes a token holder.

The order of token rotation is determined by node numbers.

Token rotation for nodes occurs in the ascending order of node registrations in the entering node management table.

A node having the highest node number passes the token to a node having the lowest node number.

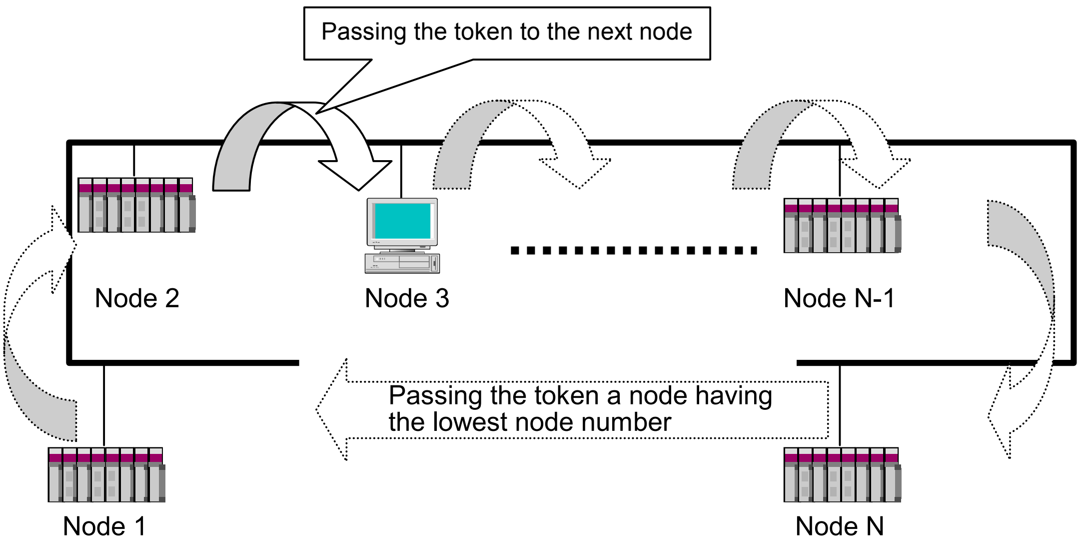


Figure 9-18 Token Flow

(3) Token and data

The following six different data patterns are used for token transmission:

Table 9-5 Token and Data

Pattern	Description
When no data is involved	Only the token is transmitted.
[Token]	
When only cyclic data is transmitted	The cyclic data is transmitted with the token attached to it.
[Token + cyclic data]	
When only cyclic data is transmitted after being divided into frames	Only the cyclic data is transmitted with the token attached to the last frame.
[Token + cyclic data] [Cyclic data]	
When only message data is transmitted	The token is transmitted subsequently to message data transmission.
[Token] [Message data]	
When cyclic data and message data is transmitted	After transmission of the message data, the cyclic data is transmitted with the token attached to it.
[Token + cyclic data] [Message data]	
When cyclic data and message data are transmitted with the cyclic data divided into frames	After transmission of the message data, only the cyclic data is transmitted with the token attached to the last frame.
[Token + cyclic data] [Cyclic data] [Message data]	

(4) Frame interval (minimum permissible frame interval)
The frame interval is the time interval between the instant at which the self-node receives the token from another node and the instant at which the self-node starts a frame transmission.
The minimum permissible frame interval is the minimum period of time for which a node must wait to initiate a frame transmission.
In FL-net, this minimum permissible frame interval is commonly used within the whole network.
The maximum value of the minimum permissible frame intervals set by entering nodes in the network is calculated and updated for all nodes upon each node enter/leave.

9.4.2 Entering in/Leaving from FL-net

(1) Entering in FL-net
When a node starts up, it monitors the line until an enter token detection timeout occurs. If it does not receive the token in this instance, it concludes that the network startup process is in progress, and then newly participates in the network. If it receives the token, on the other hand, it concludes that it is in a midway enter state, and then participates in the network on the spot.

(a) New enter
If a node does not receive the token within the enter token detection timeout period, the node prepares for trigger transmission and then start a transmission after an elapse of predetermined time (which is determined by multiplying the remainder of node number/8 by 4 ms). If trigger reception occurs before trigger transmission, the node does not transmit a trigger. For a period of enter request frame reception wait time (1200 ms) after trigger reception, all nodes wait to transmit an enter request frame while checking for a duplicate node number or address and updating the entering node management table. When the enter request frame transmission wait time (node number × 4 ms) elapses after trigger reception, the nodes transmit an enter request frame. If a node recognizes a duplicate address in accordance with an enter request frame of another node, the former node sets the common memory starting address and common memory size for area 1 and 2 to 0 and does not transmit cyclic data. When a node recognizes a duplicate address, it sets an address duplication flag and resets a common memory data validity notification flag. When an enter request frame reception wait timeout occurs, a entering node having the lowest node number transmits the token first in accordance with the entering node management table. When a node recognizes a duplicate node number, it starts no transmission/reception.

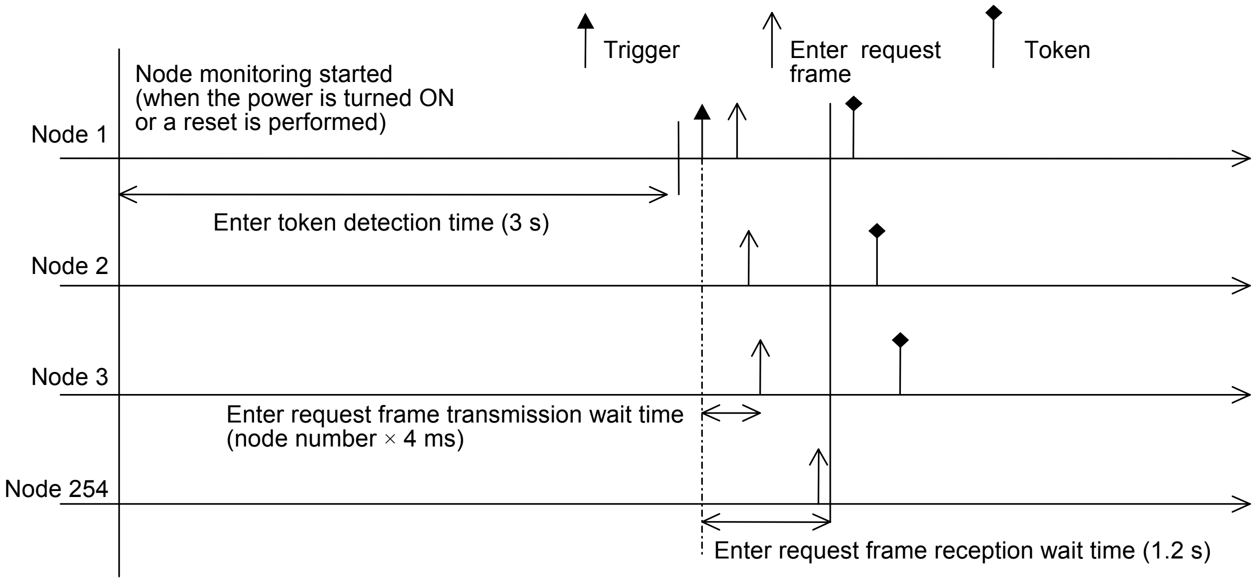


Figure 9-19 Startup Timing Diagram 1

(b) Midway enter

If a node receives the token within the enter token detection time, it concludes that a link is already established, and does not transmit an enter request frame until the token makes three rounds. A frame received during such an interval is used to check for node number, address, and other duplications and update the entering node management table. If an address duplication is detected in such an instance, the common memory starting address and common memory size of area 1 and 2 are set to 0 and cyclic data transmission does not take place. When a node recognizes an address duplication, it sets an address multiplexing flag and resets the common memory data validity notification flag. If the node number is normal, the node transmits an enter request frame after an elapse of the enter request frame transmission wait time. The enter request frame is sent no matter whether the token is held. When a node recognizes a node number duplication, it does not transmit an enter request frame and refrains from entering in the network.

(Remarks)

Enter token detection time:

Period of time for checking whether the network is operating.

Round:

Rounds are determined with respect to the time at which the token addressed to the lowest node number is received.

Enter request frame transmission wait time:

A enter request frame transmission takes place after an elapse of [self-node number × 4] ms in order to avoid a conflict with any other newly entering node.

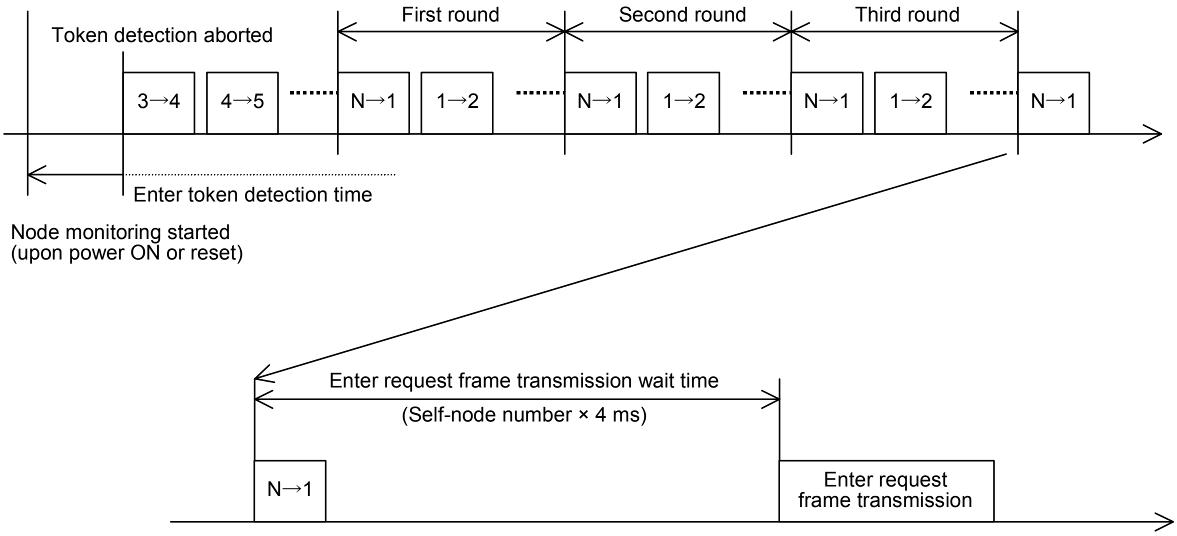


Figure 9-20 Startup Timing Diagram 2

(2) Leaving from FL-net

A node number check is conducted upon each token frame reception. If the token frame is not received from a certain node three successive times, that node is considered to have left from the FL-net (including a case where a token holder node does not transmit the token even after an elapse of the token monitoring time).

When a node is considered to have left from the network as explained above, the information about that node is deleted from the management tables.

9.4.3 Node status management

As outlined in Table 9-6, node status management is exercised with the self-node management table, entering node management table, and network management table.

Table 9-6 Node Status Management Table Overview

Name	Description
Self-node management table	Manages the self-node settings.
Entering node management table	Manages the information about nodes that have entered in the network.
Network management table	Manages the common information about the entire network.

9.4.4 FL-net self-node management table

(1) Basic features

This table manages the data about self-node settings as summarized below:

- Used to read enter request frames and network parameters.
- Management data is set by a higher FL-net layer at the time of node startup.
- The node name and the transmission area starting address and size of common memory can be set from the network.

(2) Management data

Table 9-7 Self-node Management Table

Item	Length	Description
Node number	1 byte	1 to 254
Area 1 data starting address for common memory	2 bytes	Word address (0 to 0x1FF)
Area 1 data size for common memory	2 bytes	Size (0 to 0x1FF)
Area 2 data starting address for common memory	2 bytes	Word address (0 to 0x1FFF)
Area 2 data size for common memory	2 bytes	Size (0 to 0x1FFF)
Higher layer status	2 bytes	RUN/STOP/ALARM/WARNING/NORMAL
Token surveillance timeout	1 byte	Variable in 1 ms units
Minimum frame interval	1 byte	Variable in 100μs units
Vendor name	10 bytes	Name of the vendor
Maker form	10 bytes	Maker form/device name
Node name (equipment name)	10 bytes	User-defined node name
Protocol version	1 byte	Fixed at 0x80
FA link status	1 byte	Entering/leaving, etc.
Self-node status	1 byte	Node number duplication detection, etc.

9.4.5 FL-net entering node management table

(1) Basic features

The status of nodes that have entered in the network is monitored with the management table that each node has. This table handles on an individual node basis the data about nodes that have entered in the network. The basic functions of this table are summarized below:

- A token frame is received at startup to update the entering node management table and network management table.
- Upon each token frame reception, each node updates the entering node management table.
- Upon receipt of a new node's enter request frame, the entering node management table is updated.
- When a node's token frame no-reception condition or timeout is detected three successive times, the node is deleted from the table.

(2) Management data

The token of each node is constantly monitored to create and manage the entering node management table.

Table 9-8 Entering Node Management Table

Item	Length	Description
Node number	1 byte	1 to 254
Higher layer status	2 bytes	RUN/STOP/ALARM/WARNING/NORMAL
Area 1 data starting address for common memory	2 bytes	Word address (0 to 0x1FF)
Area 1 data size for common memory	2 bytes	Size (0 to 0x1FF)
Area 2 data starting address for common memory	2 bytes	Word address (0 to 0x1FFF)
Area 2 data size for common memory	2 bytes	Size (0 to 0x1FFF)
Refresh cycle time	2 bytes	Variable in 1 ms units
Token surveillance timeout	1 byte	Variable in 1 ms units
Minimum frame interval	1 byte	Variable in 100μs units
FA link status	1 byte	Entering/leaving information, etc.

"0x1FFF" is a hexadecimal number (1FFF hex).
Remark: Contained in a received token frame.

9.4.6 FL-net status management

(1) Basic feature

Provides management of parameters related to network status.

(2) Management data

Table 9-9 Network Management Table

Item	Length	Description
Token maintenance node number	1 byte	Node that currently holds the token.
Minimum frame interval	1 byte	Variable in 100 μs units.
Refresh cycle time	2 bytes	Variable in 1 ms units.
Refresh cycle measurement time (current)	2 bytes	Variable in 1 ms units.
Refresh cycle measurement time (maximum)	2 bytes	Variable in 1 ms units.
Refresh cycle measurement time (minimum)	2 bytes	Variable in 1 ms units.

9.4.7 FL-net message sequence number management

(1) Basic feature

Manages the message transmission sequence number and sequence number/version number.

(2) Transmission management data

Table 9-10 Transmission Management Data for Message Sequence Number Management

Item	Length	Description
Sequence number/version number	4 bytes	Sequence number/version number for message transmission
Sequence number (1-to-N transmission)	4 bytes	0x1 to 0xFFFFFFFF
Sequence number (1-to-1 transmission)	4 bytes	0x1 to 0xFFFFFFFF

“0xFFFFFFFF” is a hexadecimal number (FFFFFFFF hex).

(3) Reception management data

Table 9-11 Reception Management Data for Message Sequence Number Management

Item	Length	Description
Sequence number/version number	4 bytes	0x1 to 0xFFFFFFFF
Sequence number (1-to-1 reception)	4 bytes	0x1 to 0xFFFFFFFF
Sequence number (1-to-N reception)	4 bytes	0x1 to 0xFFFFFFFF

"0xFFFFFFFF" is a hexadecimal number (FFFFFFFF hex).

9.5 Network Components

9.5.1 Ethernet component list

Figure 9-21 shows Ethernet components.

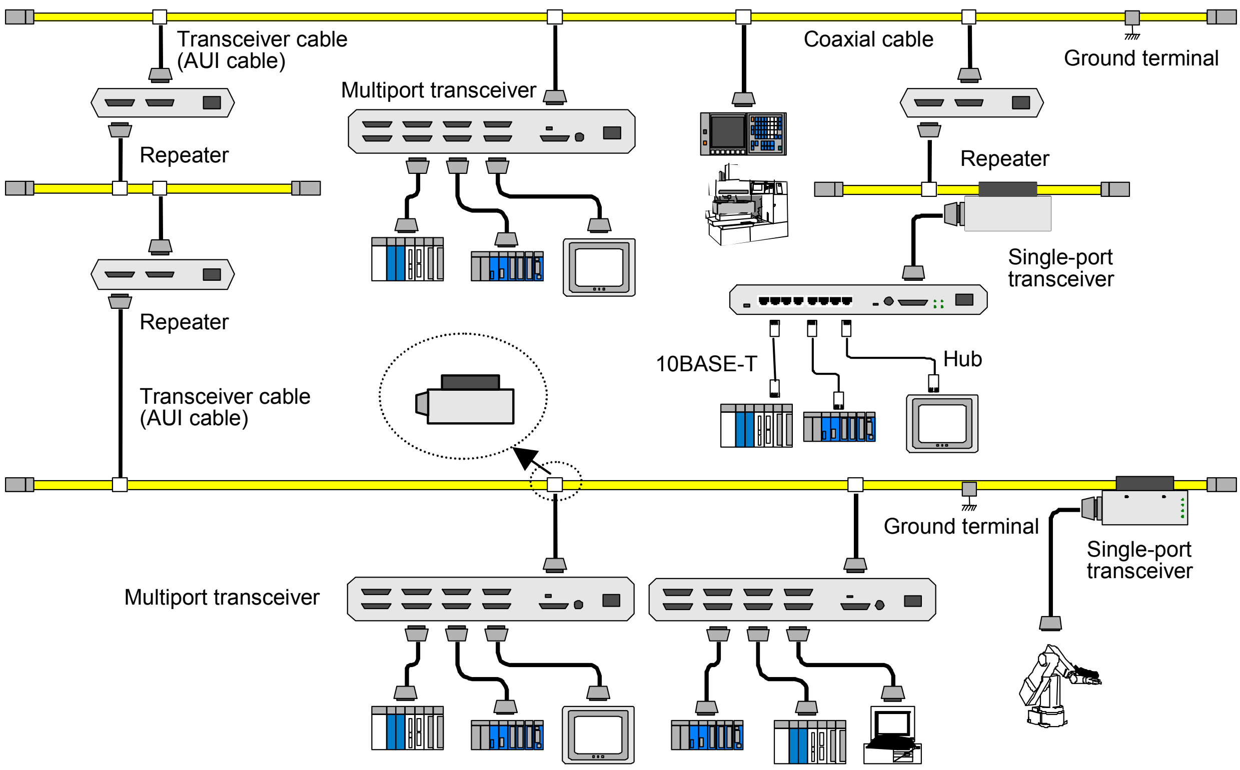


Figure 9-21 Ethernet Components

Table 9-12 Ethernet Component List

Product name	Manufacturer	Model number	Remarks
FL.NET module	Hitachi, Ltd.	LQE000	For S10mini
Transceiver	Hitachi Cable, Ltd.	HLT-200	Connector-type
		HLT-200TB HBN200TZ HBN200TD	Tap-type
Repeater	Hitachi Cable, Ltd.	HLR-200H	Coaxial cable transmission distance extension
Multiport transceiver	Hitachi, Ltd.	H-7612-64 H-7612-68	4-port/8-port (with a built-in AC power supply)
Coaxial cable	Hitachi Cable, Ltd.	HBN-CX-100	Specify the indoor cable length (500 m maximum).
Coaxial connector	Hitachi Cable, Ltd.	HBN-N-PC	Coaxial cable
Relay connector	Hitachi Cable, Ltd.	HBN-N-AJJ	Coaxial cable
Terminator	Hitachi Cable, Ltd.	HBN-T-NJ HBN-T-NP	J-type P-type
Ground terminal	Hitachi Cable, Ltd.	HBN-G-TM	Coaxial cable
Transceiver cable	Hitachi Cable, Ltd.	HBN-TC-100	D-sub 15-pin connector. Specify the cable length (50 maximum).
Twisted-pair cable	Hitachi Cable, Ltd.	HUTP-CAT5 4P	Specify the cable length and choose between the straight and cross cables (100 m maximum).
Multiport repeater (hub)	Allied Telesis K.K.	Centre COM 3624TRS	24-port
12 V power supply	Densei-Lambda K.K.	HK-25A-12	12 V for 10BASE-5

9.5.2 10BASE-5 components

(1) Transceiver

The transceiver converts a signal flowing along a coaxial cable (yellow cable) to a signal required for a node or vice versa.

Transceivers to be attached to a coaxial cable must be positioned at spacing intervals of an integer multiple of 2.5 m. Ensure that they are installed over the markings on a coaxial cable.

Before attaching transceivers to a coaxial cable, turn OFF the node and transceiver power supplies. If you make transceiver connections with the power applied, an electrical short may occur.

Ensure that the employed transceivers comply with the IEEE 802.3 standard.

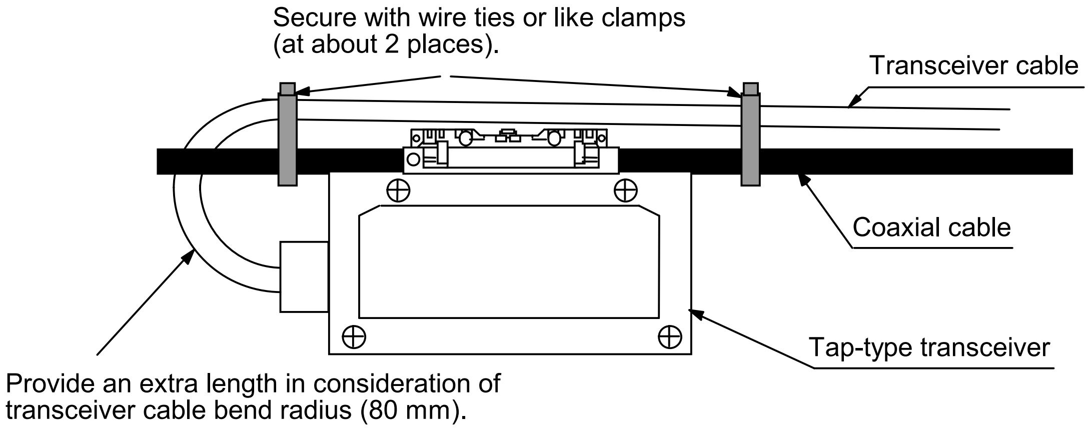


Figure 9-22 Transceiver Overview

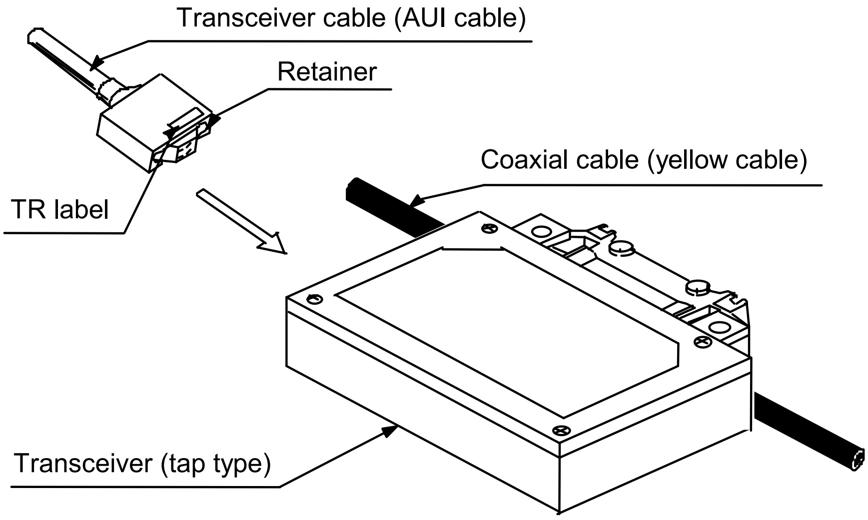


Figure 9-23 Transceiver and Transceiver Cable (AUI)

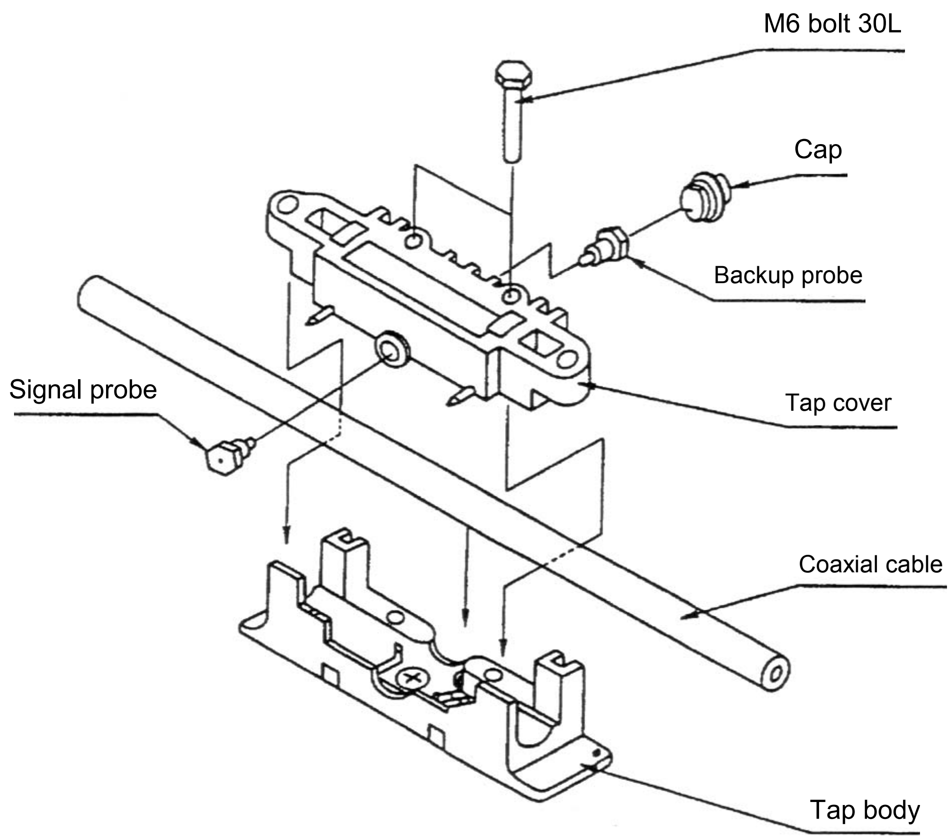


Figure 9-24 Tap and Coaxial Cable Installation

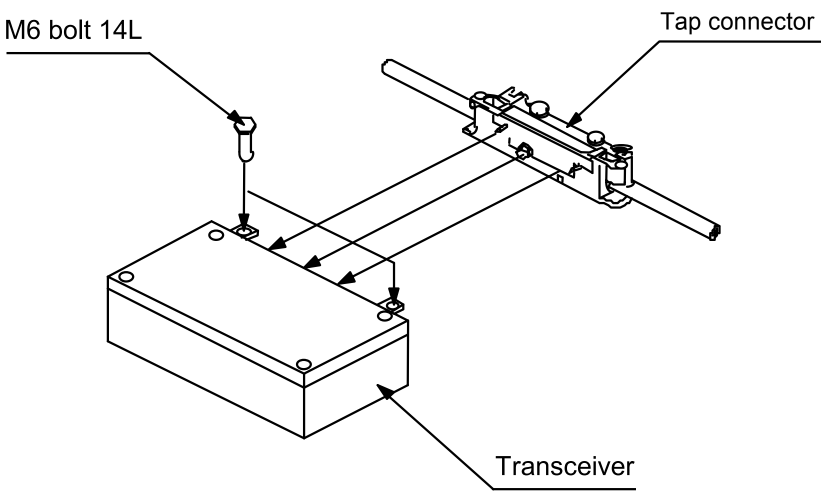


Figure 9-25 Tap and Transceiver Body Installation

- Transceiver (tap type)

  When connecting a tap-type transceiver, make a hole in a coaxial cable, insert a needle into the hole until it comes into contact with the inner conductor, and use a alligator-tooth claw to tear the insulation jacket and make a connection to the shield conductor. The use of special tools is required for transceiver connection.

  Transceiver power (12 VDC) is supplied from a node via a transceiver cable. Note that some nodes may require the use of a 12-VDC power supply for transceiver cable use. For details, refer to the node hardware manual.

  The SQE switch on the transceiver must generally be set as indicated below:

  When connected to a node: ON

  When connected to a repeater: OFF

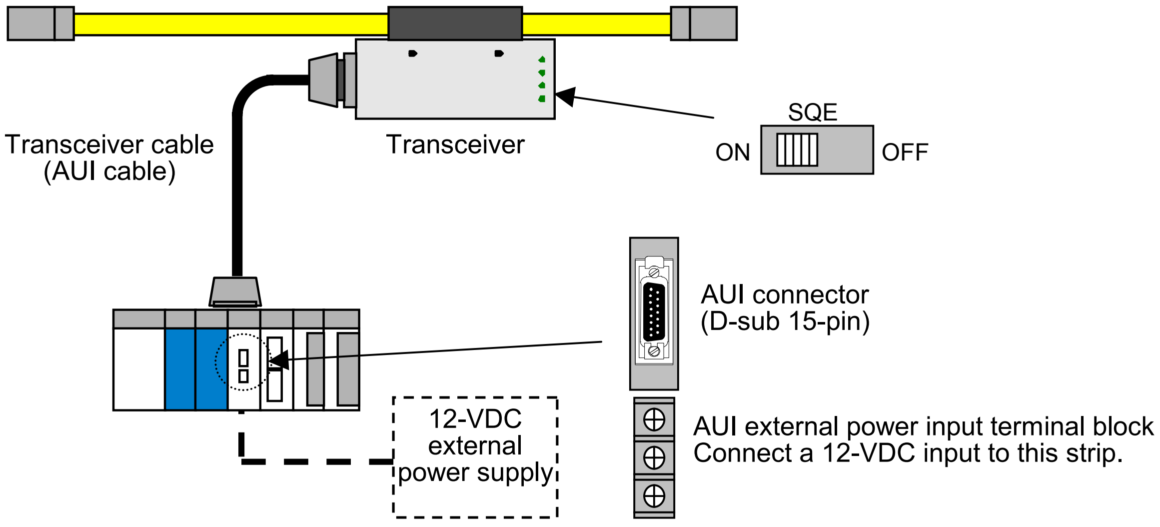


Figure 9-26 Ethernet Transceiver (Tap Type)

- Transceiver (connector type)

For connector-type transceiver connection, attach a connector to a coaxial cable and then connect the connector to the transceiver.

The connector-type transceiver can be connected without using special tools. They can easily be connected and disconnected.

Transceiver power is supplied from a node via a transceiver cable.

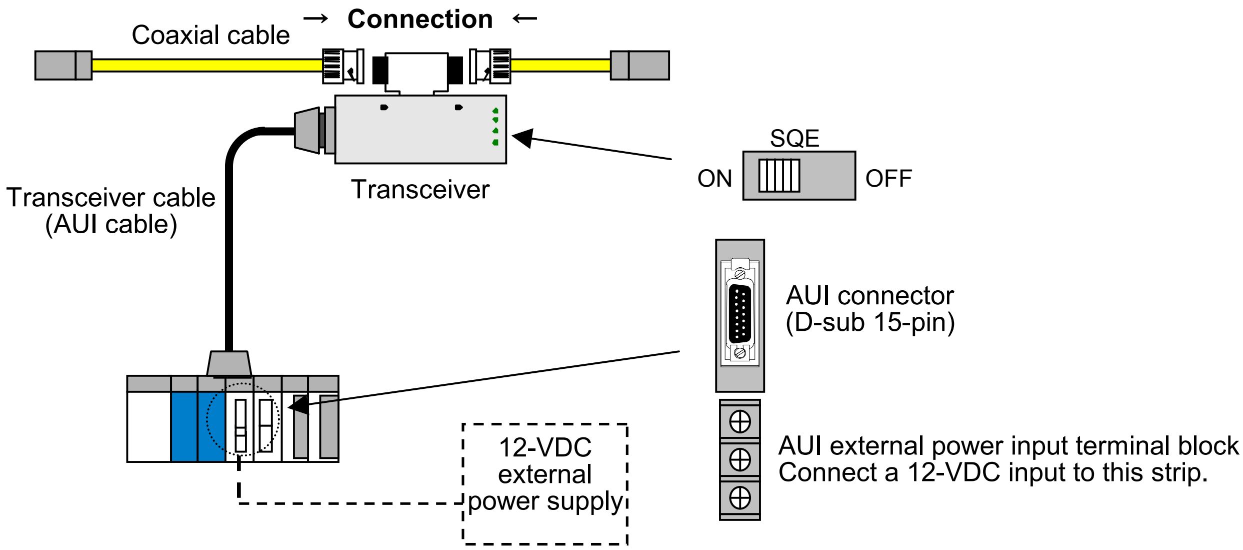
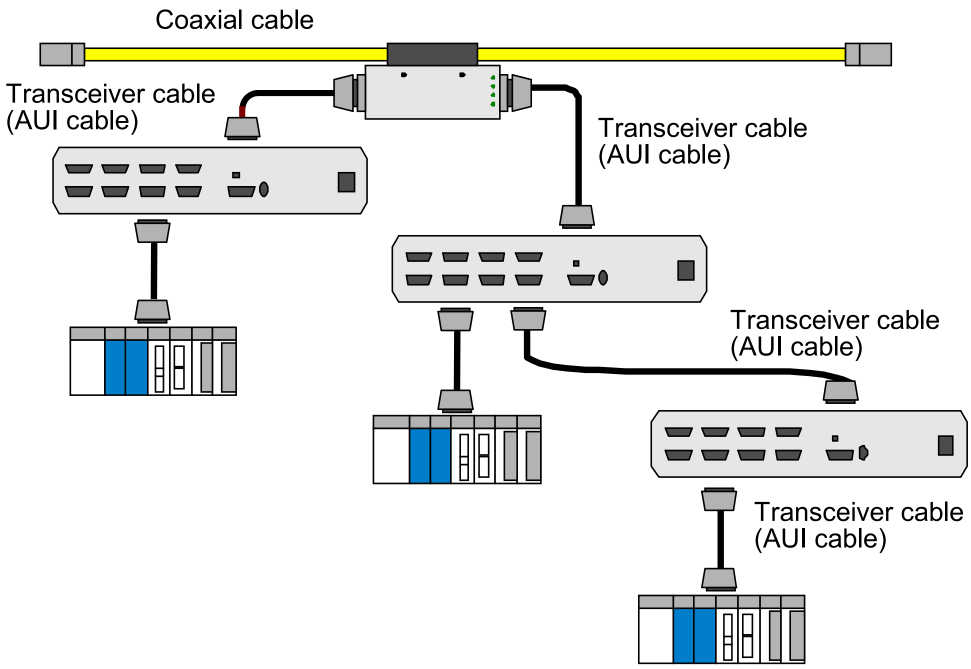


Figure 9-27 Ethernet Transceiver (Connector Type)

● Multiport transceiver

Unlike the tap-type and connector-type transceivers, which permit the connection of one terminal unit per transceiver, the multiport transceiver permits the connection of two or more terminal units. In general, 4- and 8-port transceivers are frequently used.



Transceiver power is supplied via a power cable.

Figure 9-28 Ethernet Multiport Transceiver

- Repeater

The repeater is a device that regenerates a transmission signal. It is used to interconnect differing media segments, extend the media segment distance, increase the number of terminal unit connections, or effect cable media conversion.

The repeater receives a signal from one of interconnected segments, subjects it to waveform shaping, amplifies it to a predetermined level, and transmits (repeats) it to all the segments that are connected to the repeater.

The maximum length of a transceiver cable connectable to the repeater is 50 m. However, it is recommended that you limit the transceiver cable length to 2 m in consideration of noise immunity and other factors. You should also pay attention to the SQE switch setting.

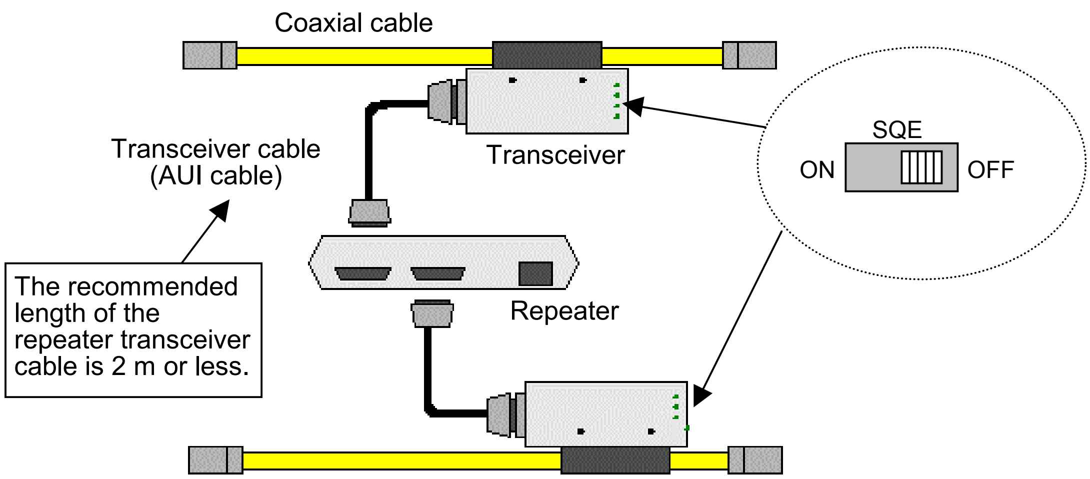


Figure 9-29 Ethernet Repeater

(2) Coaxial cable

The coaxial cable consists of an inner conductor and outer conductor. The outer conductor functions as a shield. The impedance of the coaxial cable used for Ethernet connections is 50 Ω. Typical coaxial cables are the RG58A/U for 10BASE-2 and the so-called yellow cable for 10BASE-5.

The maximum permissible cable length is 185 m for 10BASE-2 or 500 m for 10BASE-5.

When using a coaxial cable, be sure to establish a ground connection for noise control purposes (provide Class D single-point grounding).

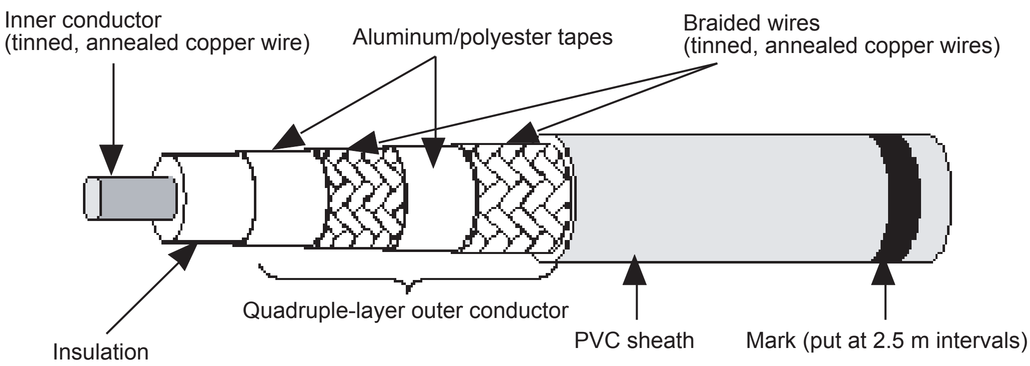


Figure 9-30 Ethernet Coaxial Cable

(3) Coaxial connector

The coaxial connector is commonly known as the N-type connector. It is used for connecting a coaxial cable to a terminator or connector-type transceiver.

Figure 9-31 Ethernet Coaxial Connector

(4) Junction connector

This connector provides a connection between coaxial cables to increase the overall coaxial cable length. In marked contrast to the repeater, which provides segment extension, the junction connector increases the overall cable length for a single segment.

If two or more junction connectors are used at one time, the coaxial cable's electrical resistance may change (it is recommended that you avoid simultaneously using two or more junction connectors).

Figure 9-32 Ethernet Junction Connector

(5) Terminator (terminating resistor)

When a bus-type wiring connection is made, the terminator is connected to both ends of a cable to prevent signal reflections. The terminator connections must be made at all times. If the terminator connections are not made, the network becomes inoperative due to signal reflections (collisions). Two types of terminators are available: J and P types. The J type is used when a tap-type transceiver is employed. The P type is used when a connector-type transceiver is employed. Ensure that the terminator is positioned at the marking on a coaxial cable.

Figure 9-33 Ethernet Terminator (Terminating Resistor)

(6) Coaxial cable ground terminal
This terminal is used to prevent a communication data error from being caused by noise contained in a coaxial cable. Ensure that the coaxial cable ground terminal is connected to a coaxial cable at one point. Provide Class D grounding.

Figure 9-34 Ethernet Coaxial Cable Ground Terminal

(7) Transceiver cable
The transceiver cable is used to connect a node to a transceiver. This cable is terminated on both ends with D-sub 15-pin AUI connectors. The maximum permissible transceiver cable length is 50 m. However, it is recommended that you use a 15 m or shorter cable at FA sites due to noise immunity considerations.

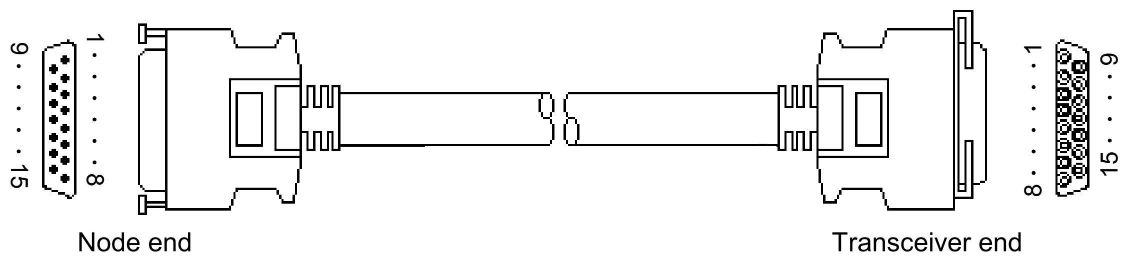


Figure 9-35 Ethernet Transceiver Cable

(8) 10BASE-5/10BASE-T converter

This converter is used to connect a cable having a 10BASE-5 interface to 10BASE-T.

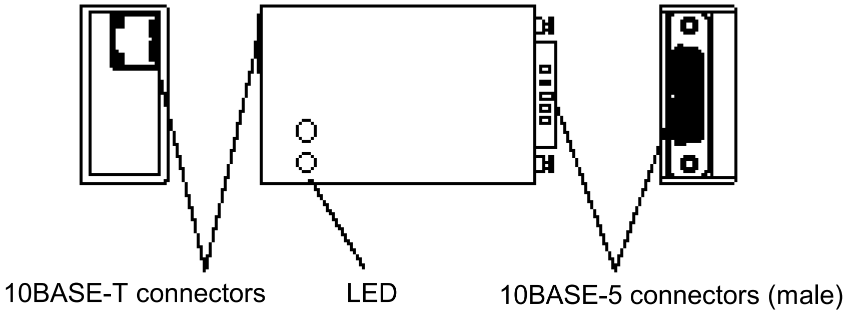


Figure 9-36 Ethernet 10BASE-5/10BASE-T Converter

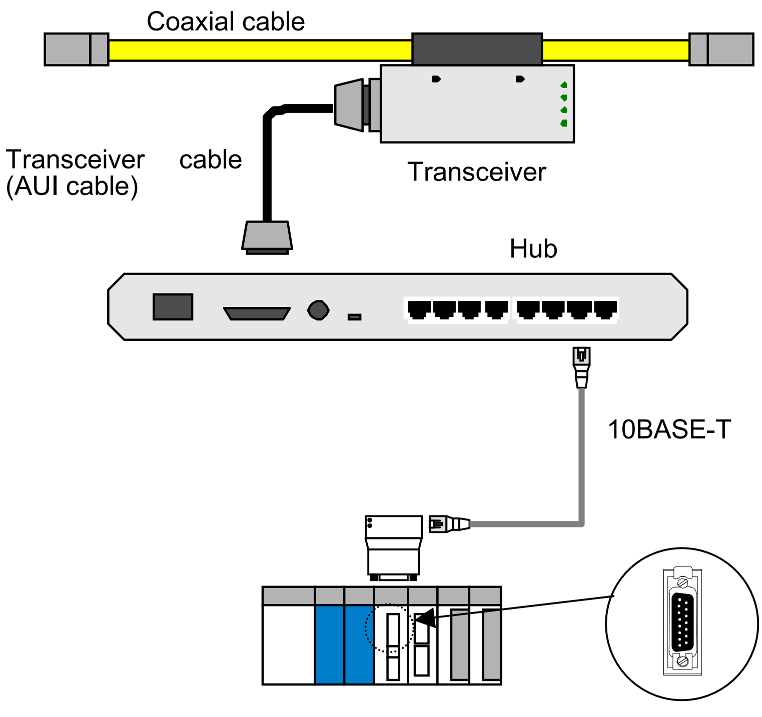


Figure 9-37 Ethernet 10BASE-5/10BASE-T Converter Installation

(9) Coaxial/optical media converter/repeater

The coaxial/optical media converter/repeater converts an electrical signal on a coaxial cable (10BASE-5/10BASE-2) to an optical signal. Typical examples are the FOIRL (Fiber-Optic Inter-Repeater Link) for inter-repeater connection and the 10BASE-FL for connection to a terminal. The coaxial/optical media converter/repeater is used, for instance, for noise immunity assurance or cable extension.

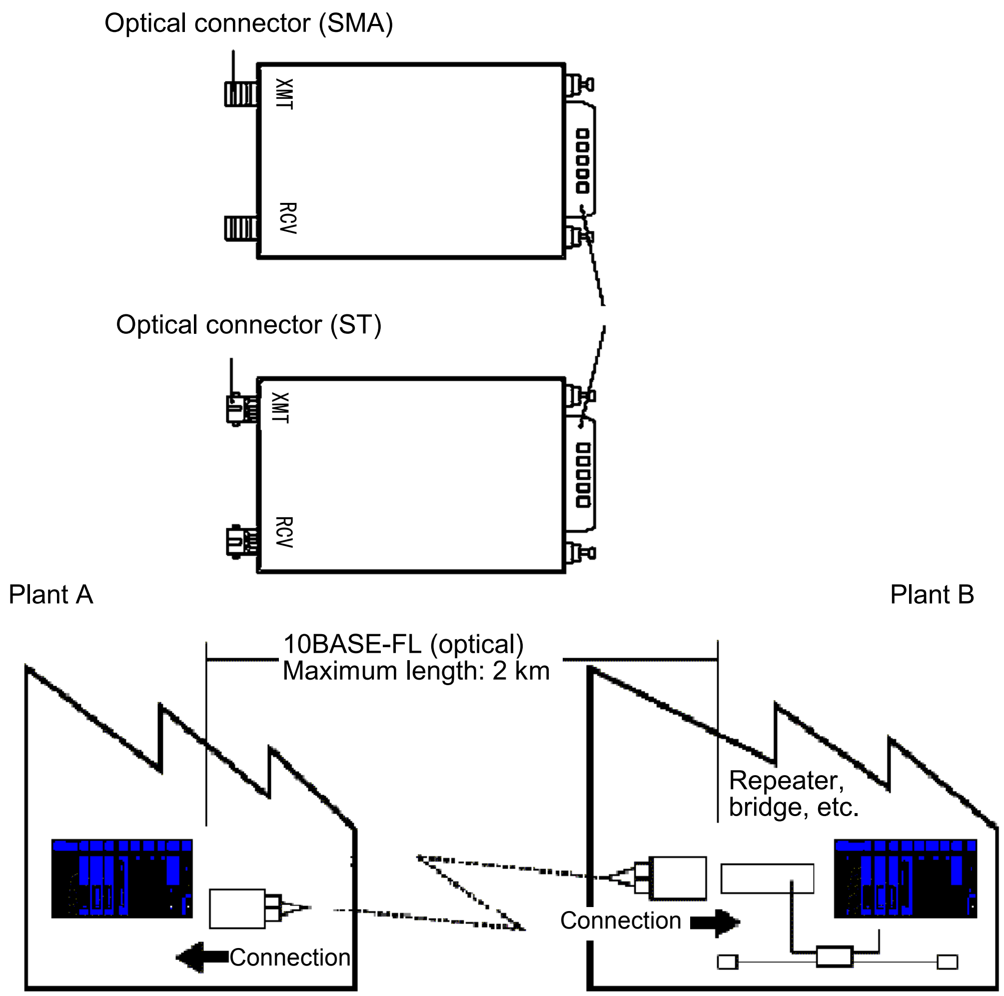


Figure 9-38 Ethernet Coaxial/Optical Media Converter/Repeater

9.5.3 10BASE-T components

(1) Hub

The hub is a line concentrator that has a repeater facility for accommodating twisted-pair cables for 10BASE-T.

The hub is available in various types including the one having a 10BASE-2 interface or an interface for cascading (making connections in cascade). Up to four hubs can be connected in cascade. You can also use a stackable hub, which consists of a number of hubs but allows you to handle it as a single hub.

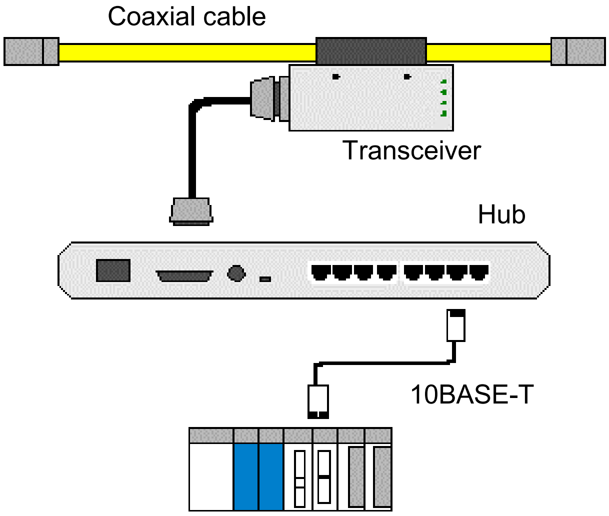


Figure 9-39 Ethernet Hub

(2) 10BASE-T cable

The 10BASE-T cable contains several twisted-pair cables or pairs of stranded copper wires and is covered by an external protective jacket. This cable is divided into the following types:

- Shielded STP cables and unshielded UTP cables
- Cross cables for making direct connections between nodes and straight cables for making connections via a hub

The maximum transmission speed provided by the 10BASE-T cable is 10 Mbps. The maximum permissible cable length is 100 m. The cable is to be terminated on both ends with 8-pin modular connectors that are stipulated in the ISO 8877 standard. Also, be sure to use Category 5 10BASE-T cables.

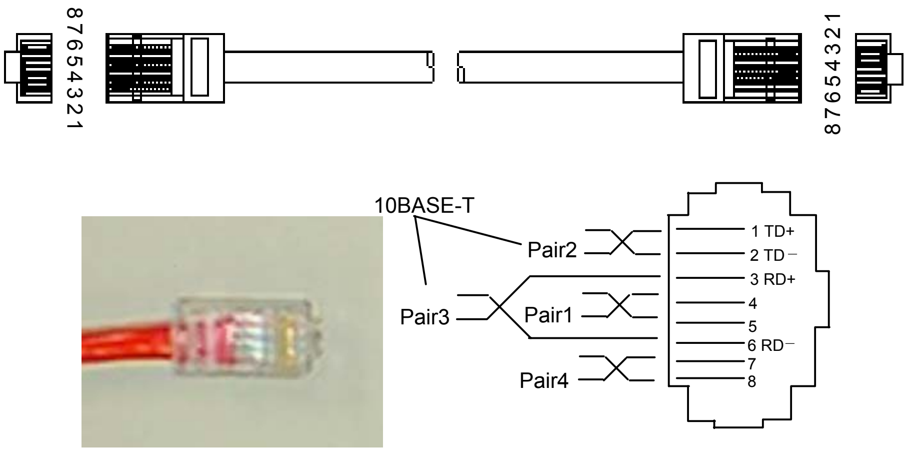


Figure 9-40 Ethernet 10BASE-T Cable

(3) 10BASE-T/optical media converter/repeater

The 10BASE-T/optical media converter/repeater converts an electrical signal on a 10BASE-T cable to an optical signal.

Typical examples are the FOIRL (Fiber-Optic Inter-Repeater Link) for inter-repeater connection and the 10BASE-FL for connection to a terminal. The 10BASE-T/optical media converter/repeater is used, for instance, for noise immunity assurance or cable extension.

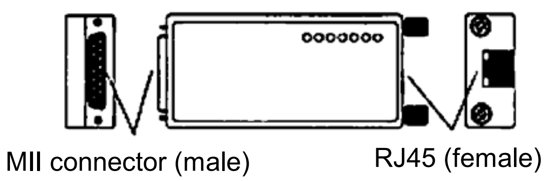


Figure 9-41 Ethernet 10BASE-T/Optical Media Converter/Repeater

9.6 FL-net Network Installation Procedures

9.6.1 10BASE-5 coaxial cable wiring

(1) Cable laying/wiring
The cable can be laid and wired by various methods depending on the installation site.
Typical methods are enumerated below:

- Wall-surface exposed cable wiring
- Free-access, floor-pit inner cable wiring
- Cable rack inner cable wiring
- Above-ceiling unfixed cable wiring

(2) Cable laying/wiring precautions
Observe the following cable laying/wiring precautions:

- Under normal conditions, the 10BASE-5 coaxial cable is to be laid and wired indoors.
- When fastening the cable to a wall surface or the like, clamp it at spacing intervals of about 1 m unless special conditions are imposed. Failure to observe this precaution will cause the cable to be stressed by its own weight. When clamping the cable in this manner, exercise care not to deform it.
- When fastening the cable to a cable rack or ceiling, clamp it at appropriate spacing intervals to avoid cable deformation.
- When laying the cable under the floor or near the floor surface, you should protect it because it can be easily deformed or damaged by pedestrians or furniture.
- In the interest of safety, the cable's outer conductor should be grounded.
- When establishing a ground connection, provide Class D or higher grounding at one point of a segment.
- To prevent exposed metal portions of the cable except a grounding point from coming into contact with the earth ground or extraneous metal surface, insulate N-type connectors, L-type connectors, linear sleeves, and terminators by covering them with the supplied boots or wrapping insulating tapes around them.
- Ensure that the cable is positioned at least 60 cm away from a power cable (100 VAC or higher voltage).

(3) Main specifications for coaxial cable installation

Table 9-13 shows the main specifications for coaxial cable installation.

Table 9-13 Coaxial Cable Specifications

Item	Specifications
Cable laying	100 mm or more in radius
Cable clamping	100 mm or more in radius
Tension	245 N maximum
Cable weight	188 kg (per length of 1 km)

(4) 10BASE-5 coaxial cable

For the cable recommended for use with the S10mini FL-net module, see “9.5.1 Ethernet component list.”

(5) Coaxial connector attachment

(a) Applicable connector

For the connector recommended for use with the S10mini FL-net module, see “9.5.1 Ethernet component list.”

(b) Installation procedure

The procedure for attaching the coaxial connector (N-PC) is shown below:

① Peeling off the PVC sheath

Figure 9-42 Peeling Off the Coaxial Cable Covering (PVC Sheath)

② Removing the aluminum tape

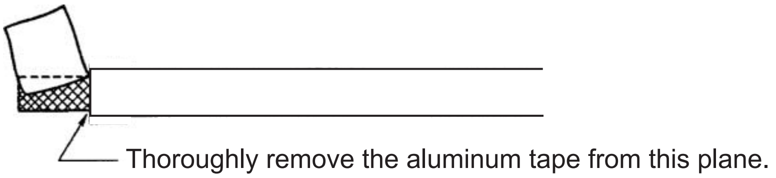


Figure 9-43 Coaxial Cable Aluminum Tape Removal 1

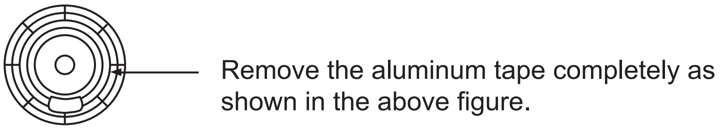


Figure 9-44 Coaxial Cable Aluminum Tape Removal 2

③ Peeling the insulator

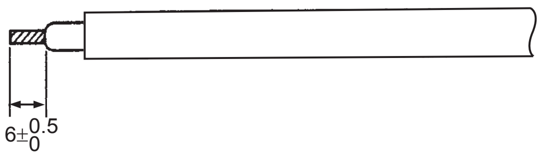


Figure 9-45 Peeling the Coaxial Cable Insulation

④ Parts setting and shield treatment

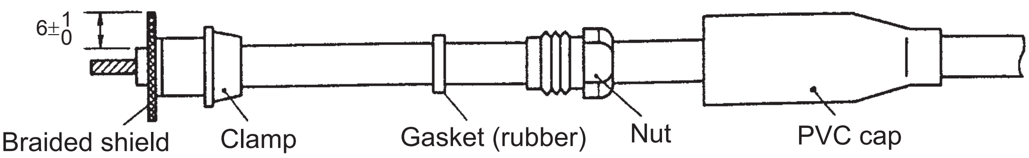


Figure 9-46 Coaxial Cable Parts Incorporation and Shield Treatment

⑤ Soldering of pin contact

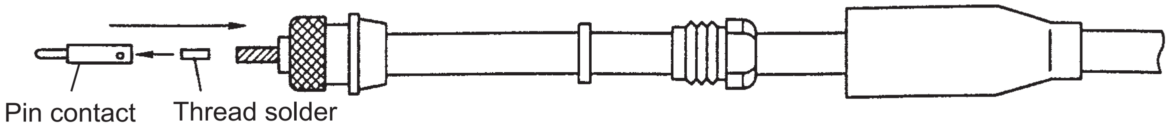


Figure 9-47 Coaxial Cable Shield Treatment and Pin Contact Soldering

⑥ Connector attachment

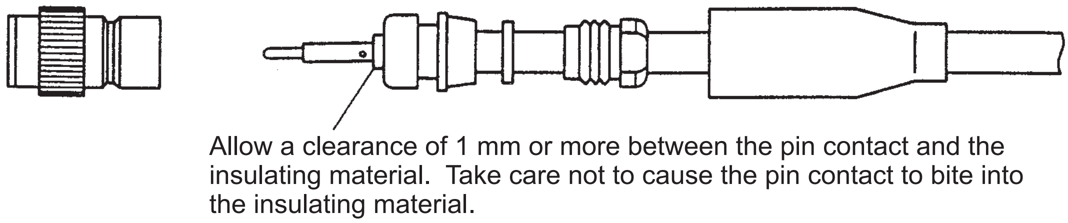


Figure 9-48 Coaxial Cable Connector Assembly

(6) Transceiver

(a) Transceiver (tap type) installation

The transceiver installation location and method vary with the installation site conditions. Typical installation locations are enumerated below:

- Wall surface
- Under the floor (in a free-access pit)
- Above the ceiling or on a cable rack
- On the side toward a station

When installing the transceiver, observe the following precautions:

- When equipped with mounting feet, the transceiver can be installed on a floor and secured with wood screws.
- When installing the transceiver above the ceiling or under the floor, you should position it so as to provide ease of maintenance.
- When installing two or more transceivers, space them at intervals of 2.5 m (mount the transceivers over the marks that are put at 2.5 m intervals on a cable).

(b) Applicable transceiver

For the transceiver recommended for use with the S10mini FL.NET module, see “9.5.1 Ethernet component list.”

(c) Installation procedure

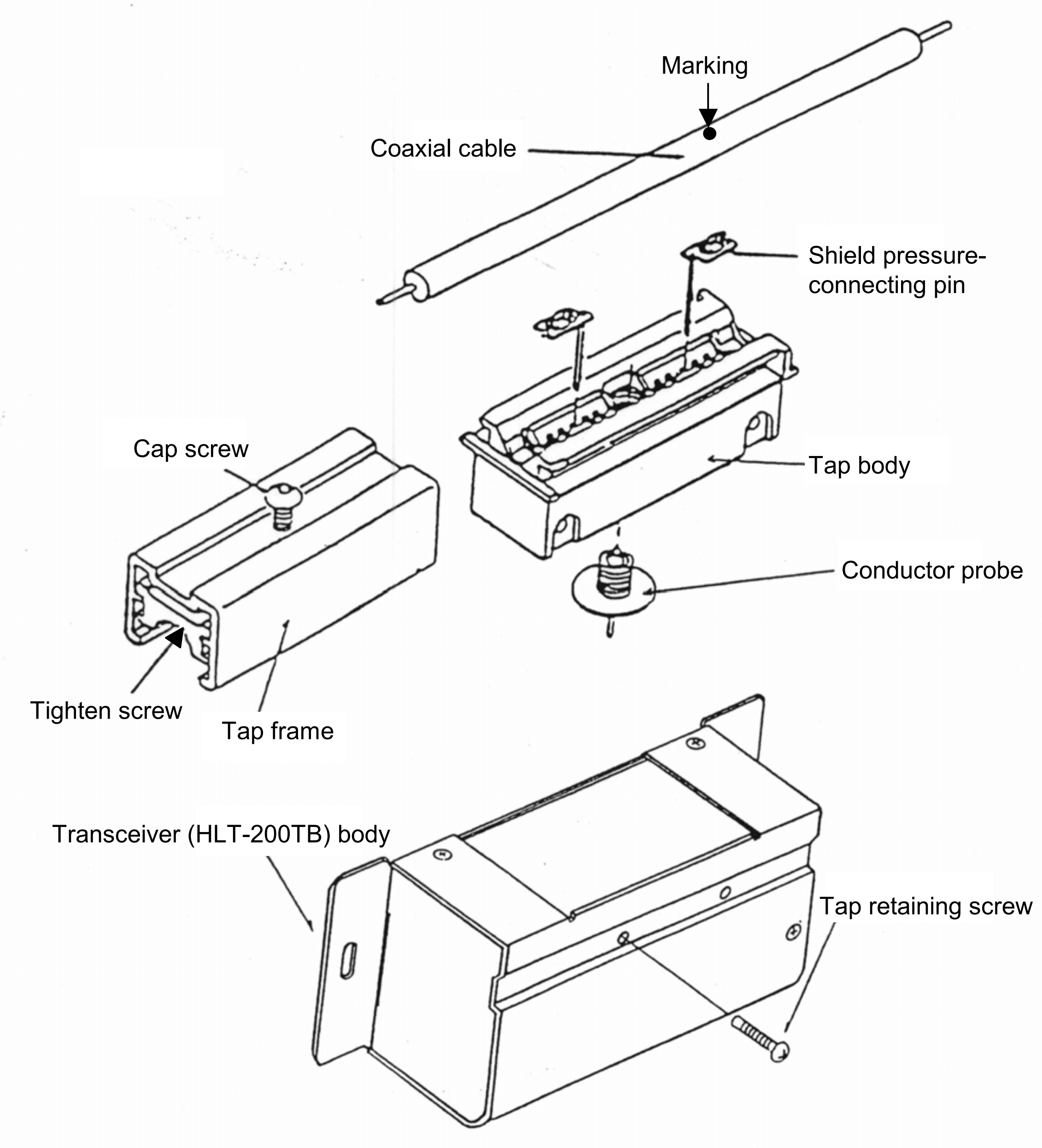


Figure 9-49 Transceiver Component Names

① Insert the shield pressure-connecting pins into the tap body.

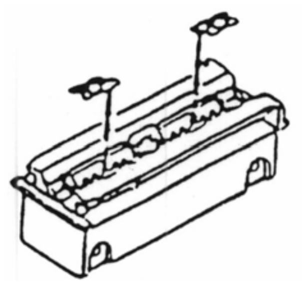


Figure 9-50 Transceiver Shield Pressure-connecting Pin into Tap Body

② Loosen the cap screw to the extent that it does not come off.

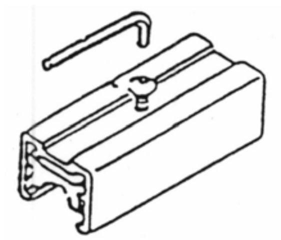


Figure 9-51 Loosing the Cap Screw of the Transceiver Tap Frame

③ Align the tap body with a 2.5 m intervals mark. Slide the frame into position and secure it by tightening the cap screw tighten the screw until the gap between the tap body upper surface and cap screw is reduced to about 1 mm.

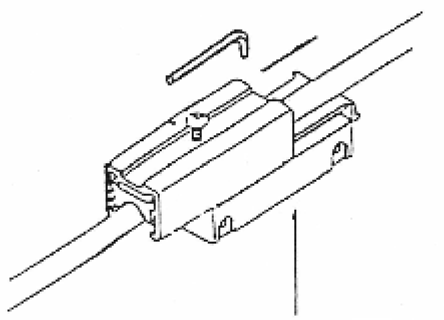


Figure 9-52 Installing the Transceiver Tap Frame and Tap Body

Insert the frame to ensure that the cable is positioned at the center of the shield pressure-connecting pin. If the cap screw is extremely tilted when tightened to a certain extent, loosen tighten the screw, center the cable, and tighten the screw.

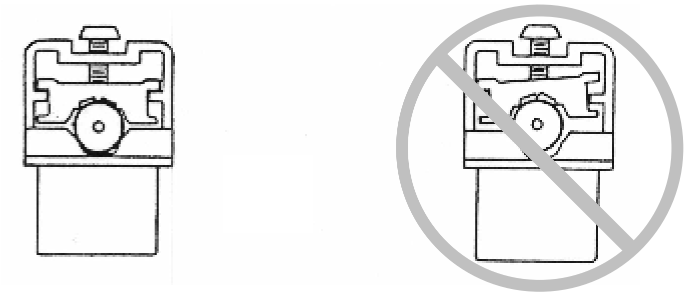


Figure 9-53 Inserting the Transceiver Tap Frame and Coaxial Cable

④ With a drill, make a conductor probe hole to the extent that a white insulation is visible. If the cap screw is heated, the aluminum tape may remain. Remove shield cuttings from the hole section.

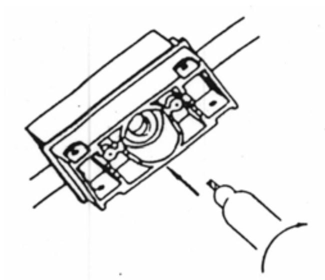


Figure 9-54 Drilling a Conductor Probe Hole in a Coaxial Cable

⑤ Tighten the conductor probe with the dedicated mounting wrench.

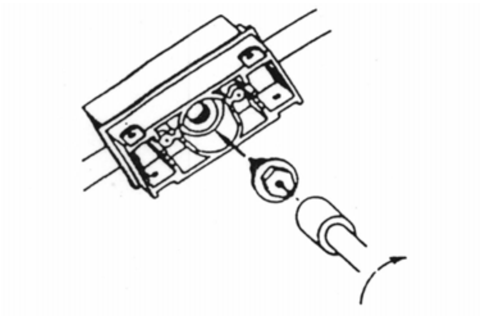


Figure 9-55 Mounting the Conductor Probe on a Coaxial Cable

Tap connection installation is now completed.

[Reference] A test method for making sure of a correct connection is described below.

- The shield pressure-connecting pins should be shorted.
- When terminators are attached to both ends of a coaxial cable, the resistance between the conductor probe and shield pressure-connecting pin should be 25Ω.

However, if the system is already running, do not run the above checks because the system may malfunction.

⑥ Insert the transceiver body into the tap connector. Accomplish centering so that the shield pressure-connecting pin is perpendicular to the conductor probe.

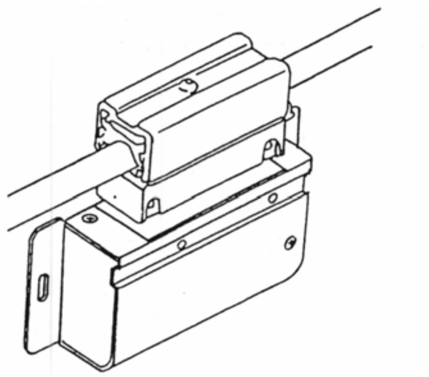


Figure 9-56 Inserting the Transceiver Body into the Tap Connector

⑦ If the shield pressure-connecting pin and conductor probe seem to be improperly positioned, pull them out after their insertion. If they are not accurately inserted, their bends are visible. In such a situation, accomplish centering again, insert the tap retaining screw into the hole in the top of the transceiver body, and tighten it.

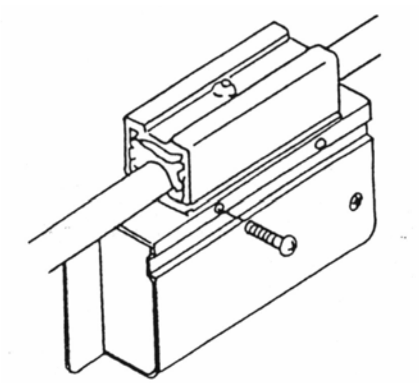


Figure 9-57 Securing the Transceiver Body and Tap

(d) Setting the SQE switch
Table 9-14 shows the SQE switch settings.

Table 9-14 SQE Switch Settings

Item	Setting
When connected to a node	ON
When connected to a repeater	OFF
When connected to an optical repeater	OFF
When connected to a router	ON
When connected to a multiport transceiver	OFF
When connected to a bridge	OFF
When connected to a hub (multiport repeater)	OFF

(7) Transceiver (connector type) installation
Same as explained under "(6)-(a) Transceiver (tap type) installation."

(a) Applicable transceiver
For the transceiver recommended for use with the S10mini FL.NET module, see "9.5.1 Ethernet component list."

(b) Installation procedure
① Finishing the coaxial cable
For the procedure for mounting the coaxial connector on a coaxial cable, see "(5) Installing the coaxial connector."

② Mount the transceiver body on the coaxial connector. Screw down the transceiver body to prevent it from being disconnected. Also, insulate the terminator by installing a rubber boot over it.

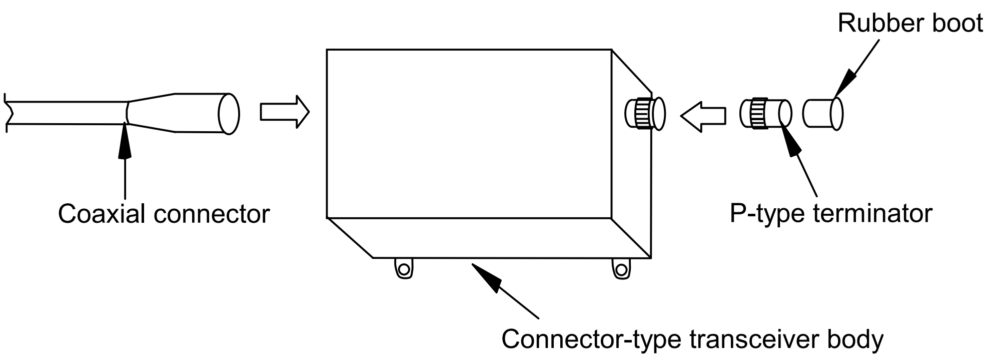


Figure 9-58 Installing the Connector-type Transceiver

(c) Setting the SQE switch
See "Table 9-14 SQE Switch Settings."

(8) Repeater installation

① Connecting the repeater

Before connecting the transceiver cable, be sure to turn OFF the repeater.

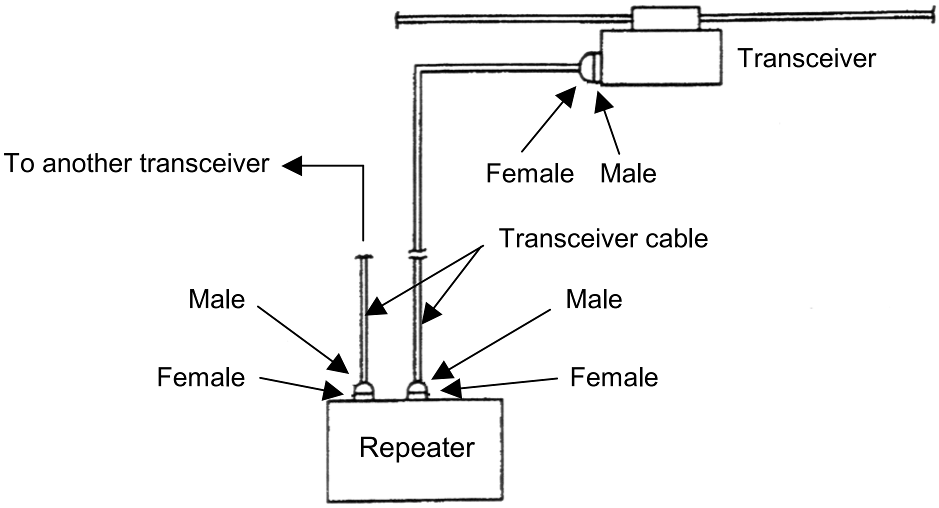


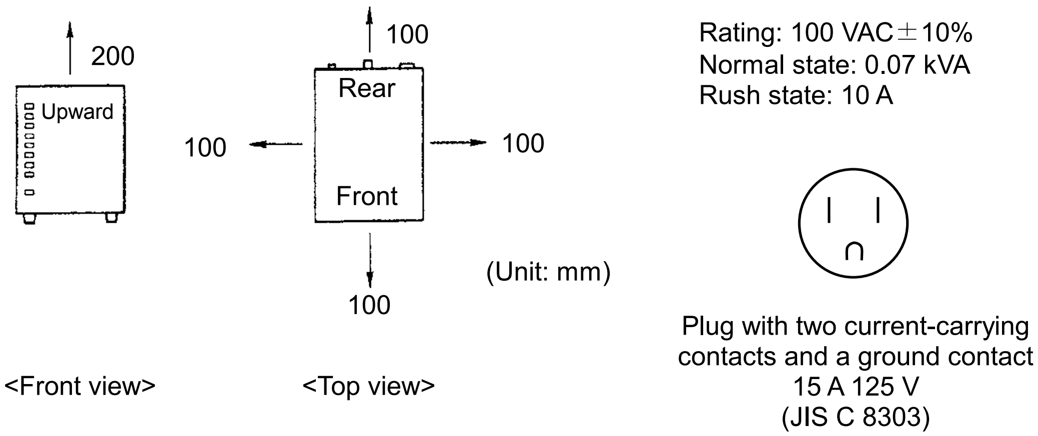
Figure 9-59 Connecting the Repeater

② Applicable repeater

For the repeater recommended for use with the S10mini FL.NET module, see “9.5.1 Ethernet component list.”

③ Installation location and space requirements

- When installing the repeater, select an appropriate site near a workstation (server) that provides an ease of maintenance (do not select a location above the ceiling or basement of an ordinary office room). Also, ensure that the front/rear, left/right, and upward clearance requirements indicated below are met. Note that the repeater requires an AC power source. Therefore, furnish a grounded outlet.



Note: The clearance from the front panel surface should be as great as possible.

Figure 9-60 Repeater Installation Space Requirements

- Do not use the repeater in a dusty location.
- The air intake and air discharge ports are provided in the bottom and top of the repeater, respectively. Do not cover such openings.
- For ease of maintenance, it is recommended that you install a telephone near the repeater installation site.
- Furnish the repeater with an independent power source that cannot possibly be turned OFF inadvertently. If the repeater is OFF, no communication can be established.

④ Setting the SQE switch

See "Table 9-14 SQE Switch Settings."

(9) Insulating the terminator and connector
The connectors (junction connector and L-type connector) must be insulated as indicated in Figures 9-61 and 9-62.

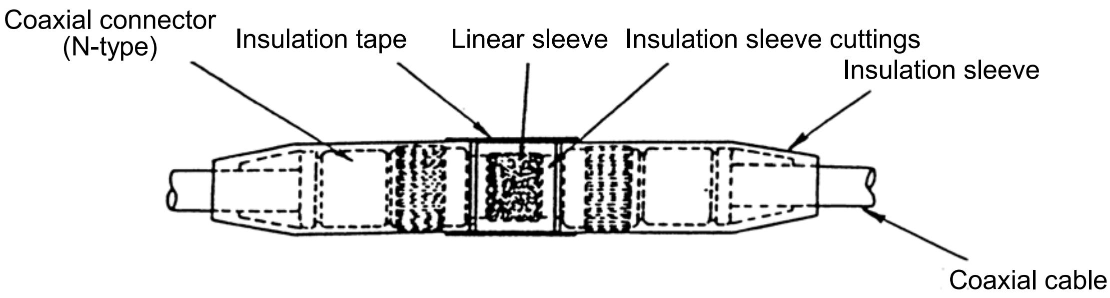


Figure 9-61 Junction Connector Insulation

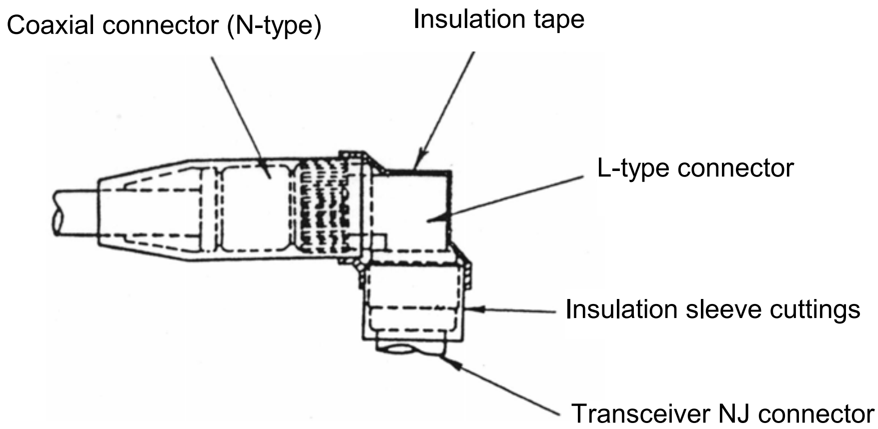


Figure 9-62 L-type Connector Insulation

The terminators (T-NP male and T-NJ female) must be insulated as indicated below:
- Install an insulation sleeve (black) (I-NPC) over the male T-NP.
- Install an insulation sleeve (black) (I-NJP) over the female T-NJ and tape it down.

(10) Applicable terminator (terminal resistor)
For the terminator recommended for use with the S10mini FL.NET module, see "9.5.1 Ethernet component list."

(11) Transceiver cable installation

Figure 9-63 shows a typical installation of the transceiver and transceiver cable.

<Wall-mounting example (1)>

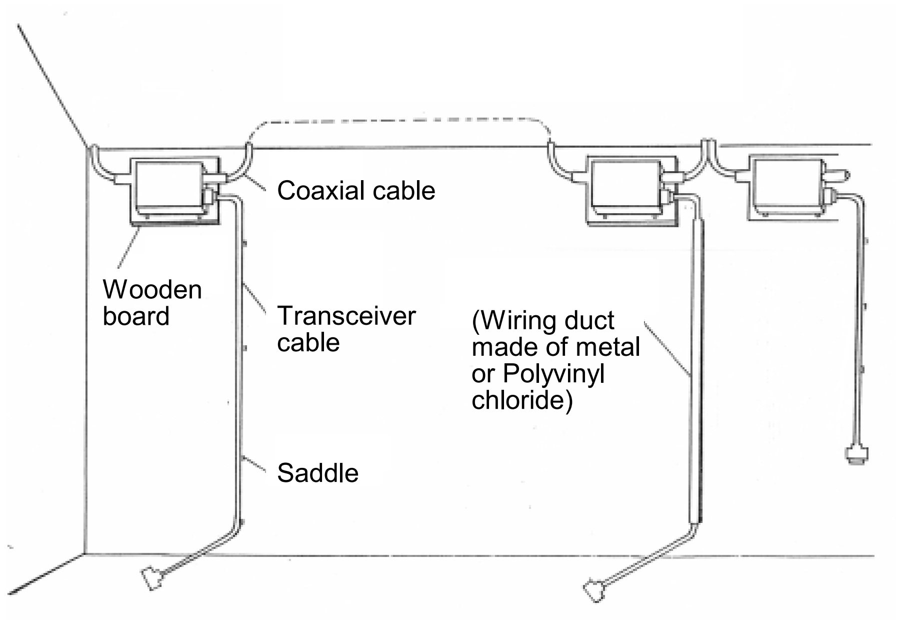
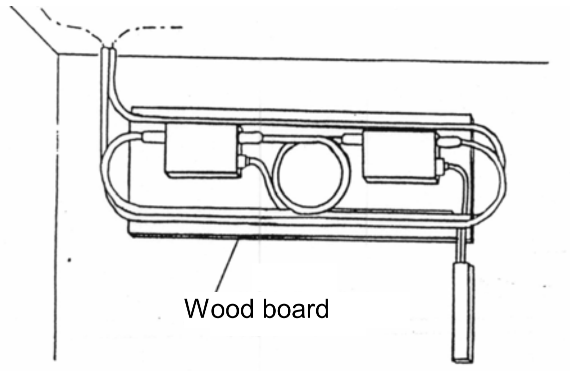
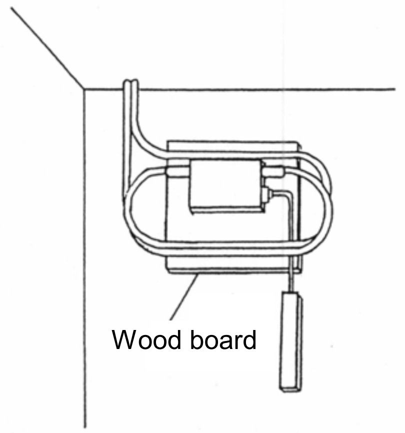


Figure 9-63 Transceiver and Transceiver Cable Wall-mounting Example 1

<Wall-mounting example (2)>



<Wall-mounting example (3)>



<Wall-mounting example (4)>

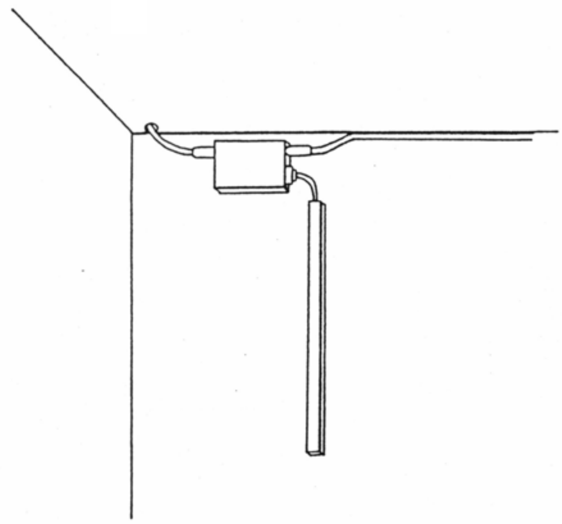
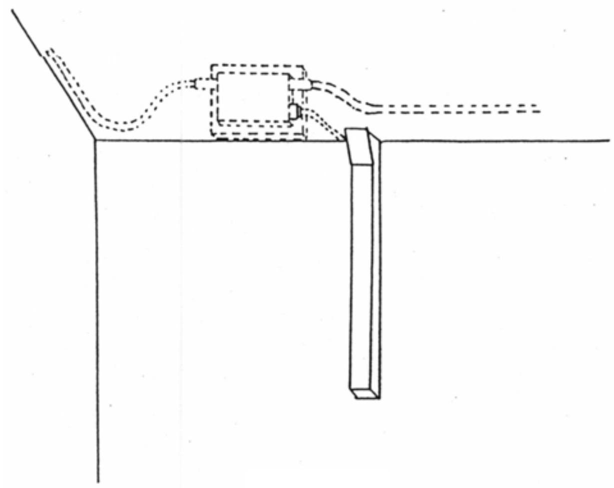


Figure 9-64 Transceiver and Transceiver Cable Wall-mounting Examples 2

<Above-the-ceiling mounting example>



<Under-the-floor mounting example>

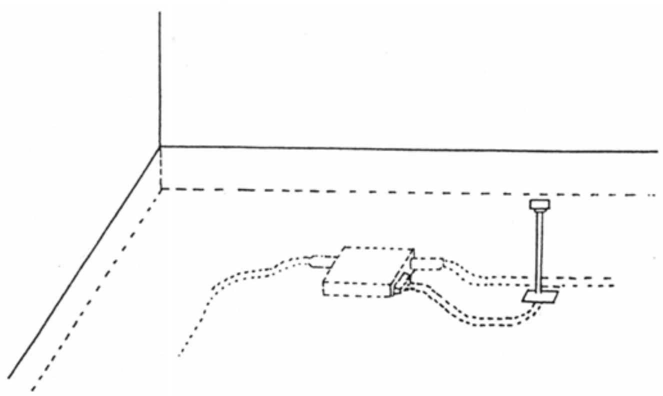


Figure 9-65 Transceiver and Transceiver Cable Above-the-ceiling Mounting and Under-the-floor Mounting Example

(a) Applicable transceiver cable

For the cable recommended for use with the S10mini FL.NET module, see “9.5.1 Ethernet component list.”

(12) Coaxial cable ground terminal installation

(a) Applicable ground terminal

For the ground terminal recommended for use with the S10mini FL.NET module, see “9.5.1 Ethernet component list.”

(b) Installation procedure

Installing the coaxial cable ground terminal

Figure 9-67 shows how to install the coaxial cable ground terminal. For the coaxial cable, provide single-point grounding (Class D or higher) with a ground terminal (G-TM). Ground the coaxial cable at an arbitrary point.

① Insert the insertion claw into the ground terminal body.

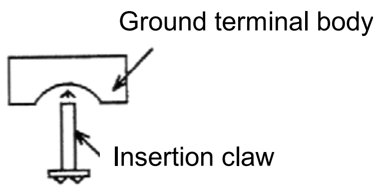


Figure 9-66 Mounting the Insertion Claw in the Ground Terminal Body

② Mount the ground terminal on a coaxial cable and alternately tighten the M4 screws. Attach a crimp terminal to either screw. Select only one location on a coaxial segment so as to provide ease of ground connection.

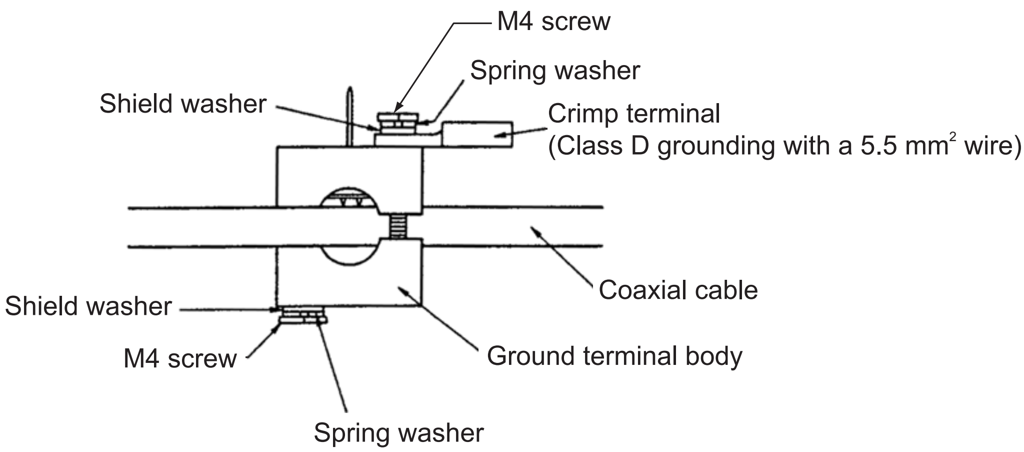


Figure 9-67 Installing the Ground Terminal

③ After tightening the M4 screw, cut off the excess length of the insertion claw.

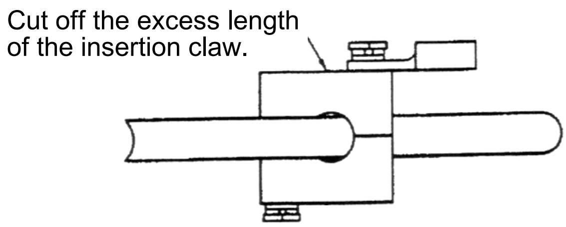


Figure 9-68 Cutting the Insertion Claw

④ Provide single-point grounding (Class D).

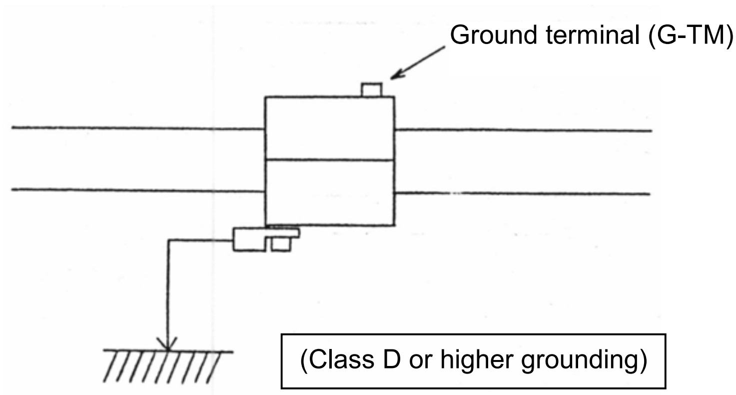


Figure 9-69 Mounting the Coaxial Cable Ground Terminal

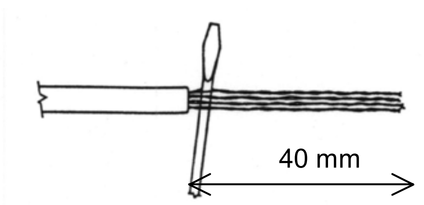
9.6.2 10BASE-T (UTP) cable

(1) Preparing a 10BASE-T (UTP) able

① Removing the covering (sheath) from the 10BASE-T (UTP) cable

Cut about 40 mm of sheath and then loosen the wire strands to rearrange them in proper order.

Under normal conditions, a normal cable (straight cable) should be used.



Arrangement

	T568B	T568A
	Normal	Cross
8	Brown	Brown
7	White/brown	White/brown
6	Green	Orange
5	White/blue	White/blue
4	Blue	Blue
3	White/green	White/orange
2	Orange	Green
1	White/orange	White/green

Figure 9-70 Removing the UTP Cable Sheath

② Cutting the 10BASE-T (UTP) cable signal wires

With nippers or the like, cut the cable in such a manner that about 14 mm of signal wires protrudes out of the remaining sheath.

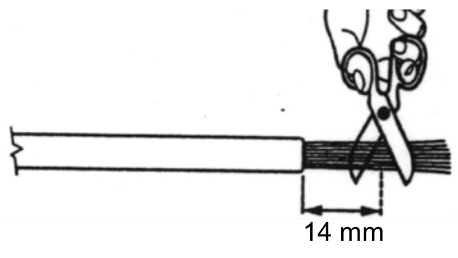


Figure 9-71 Cutting the UTP Cable Signal Wires

③ Inserting the UTP cable into the connector

Attach the cable to the connector while retaining the proper arrangement of wires. View the front, top, and bottom surfaces of the cable to check whether the cable is inserted all the way into the connector.

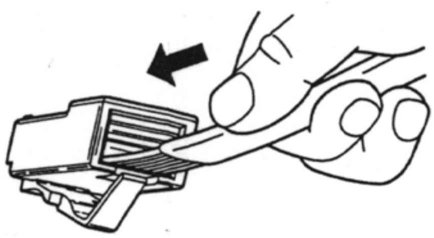


Figure 9-72 Inserting the UTP Cable into the Connector

④ Assembling the UTP cable connector

Verify that the cable is properly inserted into the connector, and then pressure-connect the cable to the connector with a dedicated tool. After completion of pressure-connection, be sure to verify the connection with a dedicated tester.

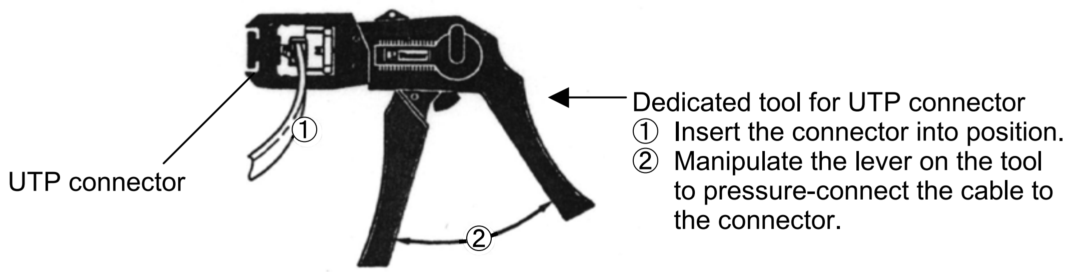


Figure 9-73 Assembling the UTP Cable Connector

⑤ Applicable cable

For the cable recommended for use with the S10mini FL.NET module, see “9.5.1 Ethernet component list.”

(2) Hub

(a) Applicable hub

For the hub recommended for use with the S10mini FL.NET module, see “9.5.1 Ethernet component list.”

9.7 Grounding the FL-net System

9.7.1 Overview of FL-net system grounding

Figures 9-74 and 9-75 show the methods of grounding the FL-net system's controller control panel with a steel frame of a building.

The control panel can be grounded with a building's steel frame only when the following conditions are met. If the conditions are not met, make a ground connection that is dedicated to the controller (provide Class D or higher grounding).

- Steel frames are welded together.
- The Class D grounding work requirements are met between the earth ground and steel frames.
- No high-voltage circuit current flows to the control panel's grounding point.
- The control panel's grounding point is positioned at least 15 m away from a high-voltage circuit panel.

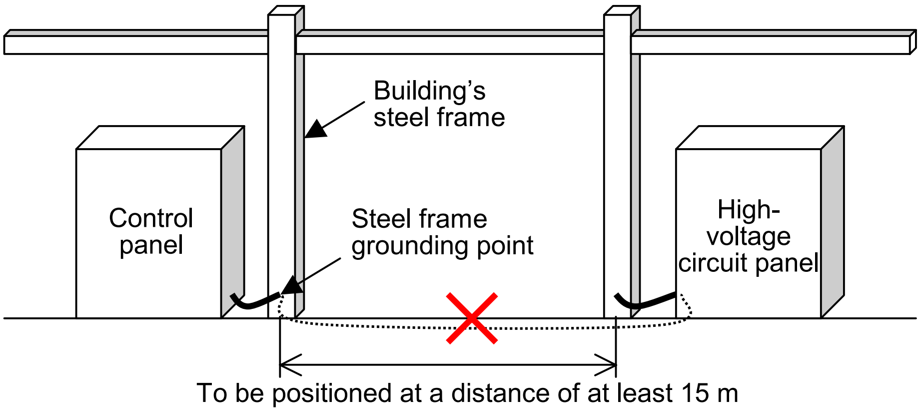


Figure 9-74 Controller Control Panel Grounding Method 1
(Steel Frame Grounding)

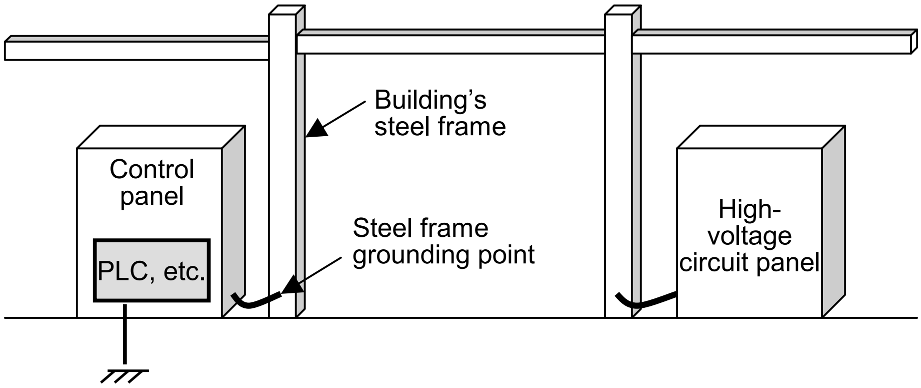


Figure 9-75 Controller Control Panel Grounding Method 2
(Controller's Dedicated Class D Grounding)

9.7.2 Power supply wiring and grounding

For the explanation of the FL-net system's power supply wiring and grounding, the example in Figure 9-76 indicates the power supply wiring and grounding of the distribution switchboard and controller panel.

When making power supply wiring and ground connections, observe the following instructions:

- With an insulating transformer with a static shield, provide insulation between the control power supply and controller power supply.
- Provide Class D grounding for the frames of the distribution switchboard and controller control panel.
- Do not connect the controller's FG (frame ground) terminal to the frame of the control panel. Provide the controller with dedicated Class D or higher grounding.
- Minimize the length of the controller input power supply wiring. Use twisted wiring for the controller input power supply.
- Connect the controller LG (line ground) terminal to the shield terminal on the insulating transformer and then to the panel's frame ground.

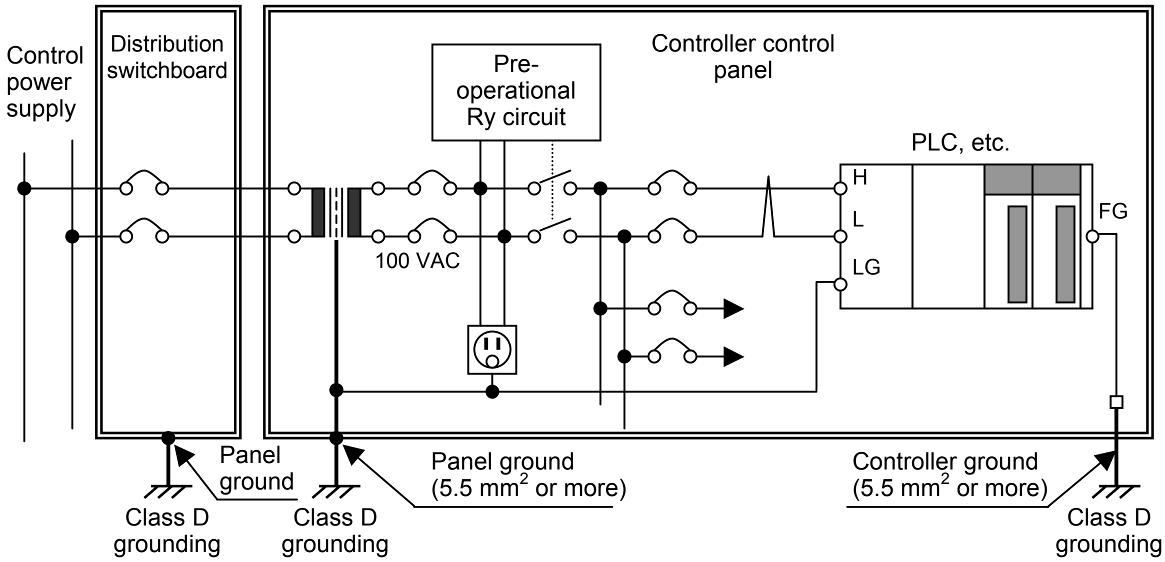


Figure 9-76 Typical FL-net System Power Supply Wiring and Grounding

9.7.3 Power supply wiring and grounding for FL-net system network devices

Figure 9-77 shows a typical power supply wiring/grounding scheme for the network devices of the FL-net system.

When making power supply wiring and ground connections, observe the following instructions:

- Connect the ground terminal on a coaxial cable to the Class D ground that is dedicated to the controller.
- Connect the frame ground of the 10BASE-T hub to the Class D ground that is dedicated to the controller. Also, ensure that the 10BASE-T hub receives power from the same insulating transformer with a static shield as for the controller power supply.
- Do not connect the controller's FG (frame ground) terminal to the frame of the control panel. Provide the controller with dedicated Class D or higher grounding.
- Connect the FG (frame ground) terminal on the FL.NET module to the FG (frame ground) terminal on the controller.
- Connect the shield ground of the transceiver (AUI) cable to the FG (frame ground) terminal on the FL.NET module.

- If the transceiver (AUI) needs power supply from a DC power source (12 VDC, etc.), furnish the network with a dedicated regulated power supply and connect its DC output to the specified terminal on the FL.NET module. The required 100 VAC input power must be supplied from the same insulating transformer with a static shield as for the controller.

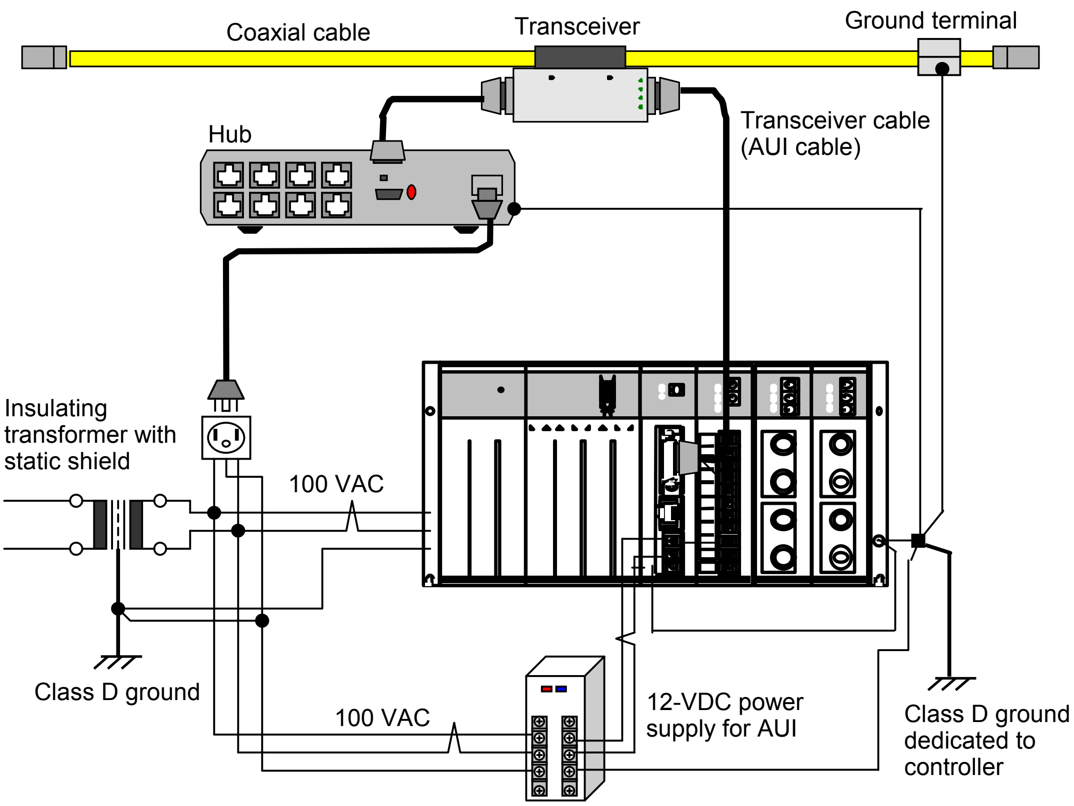


Figure 9-77 Typical Power Supply Wiring/Grounding Scheme for FL-net System Network Devices

9.7.4 Wiring duct/conduit wiring and grounding

The examples in Figures 9-78 and 9-79 indicate the wiring duct/conduit wiring and grounding schemes for the FL-net system. As regards wiring work, observe the following instructions:

- When using a wiring duct for wiring purposes, furnish a separator to separate the power and signal lines in accordance with their levels. Also, provide the wiring duct (cover and separator included) with Class D grounding.
- When making wiring connections through conduit use, furnish all the power and signal cable levels with a separate conduit that complies with the JIS C 8305 standard. Also, provide each conduit with Class D grounding.

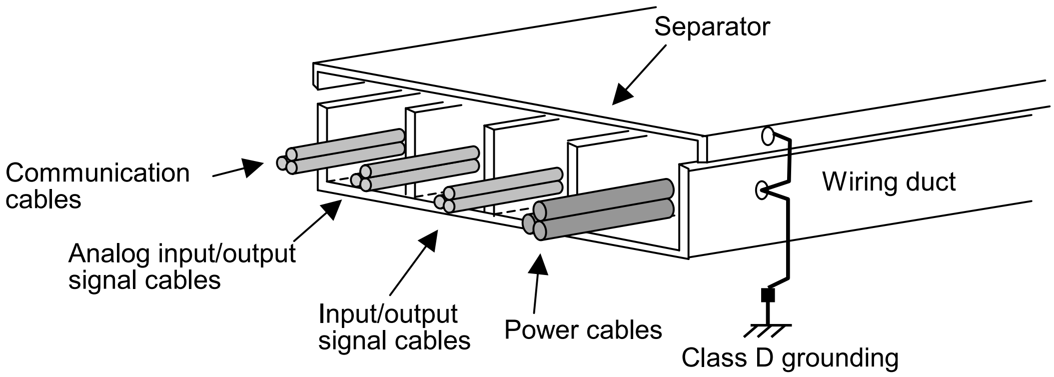


Figure 9-78 Typical Wiring with Wiring Duct

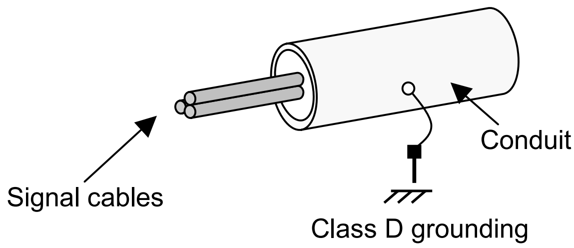


Figure 9-79 Typical Wiring with Conduit

9.8 FL-net Installation Work Check Sheet

Table 9-15 FL-net Installation Work Check Sheet

FL-net Installation Work Check Sheet

Communication line name: ____________ Area code: ____________ Inspection date: ____________

Inspector Company: ____________

Person in charge: ____________

	Check	Entry column	Setup switch column
Cable	Check that all the connectors are properly locked.		
	Check that each cable bend radius is not smaller than specified.		
	Check that the connectors are protected with a jacket or the like.		
	Check that the wires are properly marked with an identification number (line number).		
	Check that no cables are positioned beneath heavy items.		
	Check that no communication cables are bundled with a power cable or the like.		
	Check that the repeater AUI cable length is 2 m or less and that the transceiver AUI cable length is 50 m or less.		
	Check that the coaxial cable (10BASE-5) length is 500 m or less.		
	Check that the coaxial cables are properly grounded with a ground terminal.		
	Check that the coaxial cable shield and transceiver are insulated.		
	Check that the a terminal resistor is properly attached to a coaxial cable.		
	Check that the number of cascaded hubs and repeaters is within the specified limit.		
	Check that the employed twisted-pair cable is of a straight type.		
	Check that the employed twisted-pair cable belongs to Category 5 and does not exceed 100 m in length.		
Unit	Check that the GND terminals of the devices are properly grounded.		
	Check that each unit is properly fastened to the base.		
	Check that the base unit is properly fastened to the control panel.		
	Check that the AUI cable is properly locked.		
	Check that a door or other item does not apply any undue force to the AUI cable mount.		
Hub, etc.	Check that the RJ45 connector is properly connected.		
	Check that the AUI cable connector is locked.		
	Check that the line number labels are properly attached.		
	Check that the transceiver is properly installed at a marked position.		
	Check that the transceiver SQE switch is set in compliance with the device specifications.		
	Check that the hub is properly secured.		
	Check that the HUB/MAU selector switch is properly set for the hub.		
	Check that the specified supply voltage is applied to the hub.		

- Whenever a modification or change is made or a check is conducted, be sure to run the above checks and enter their results.
- In the result columns on the right, enter "○" to indicate an "OK" result or "×" to indicate an "NG" result. In a selector switch column (inner), enter a rotary switch number or DIP switch position (ON or OFF).

9.9 FL-net Profile

9.9.1 Device communication information classification

In the FL-net, the communications-related information about networked devices is classified into three types as shown in Figure 9-80.

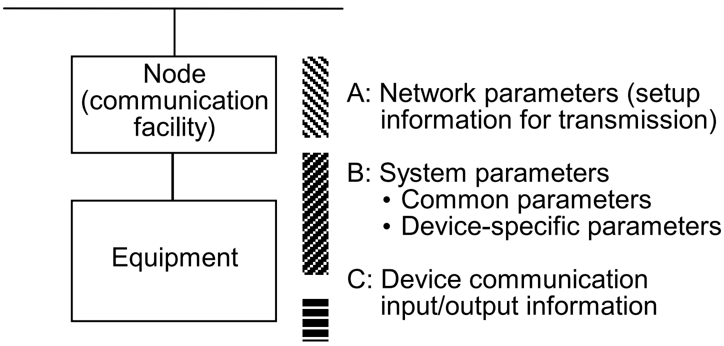


Figure 9-80 Device Communication Information Classification

- The network parameters (A) are essential to the information setup for transmission.
- The system parameters (B) are static parameters, which furnish management information for identifying the networked devices. These parameters are divided into common parameters and device-specific parameters.
- The device communication input/output information can be accessed from remote networked devices whenever it is needed for applications. This information includes dynamic information, which varies with application operations and device conditions.

9.9.2 Details of common parameters

Table 9-16 shows the details of common parameters.

Table 9-16 Details of Common Parameters

Parameter name	Name character string [PrintableString type] (Length), (String)	Data type [type]	Contents of parameter (Length), (Contents)
Device profile common specification version	6, “COMVER”	INTEGER	Example: 1, 1
System parameter identification string	2, “ID”	PrintableString	7, “SYSPARA”
System parameter revision number	3, “REV”	INTEGER	Example: 1, 0
System parameter revision date	7, “REVDATE”	[INTEGER], 2, (0001-9999), [INTEGER], 1, (01-12), [INTEGER], 1, (01-31)	Example: 2, 1998 1, 9 1, 30
Device category	10, “DVCATEGORY”	PrintableString	Example: 3, “PLC” (see Note)
Vendor name	6, “VENDOR”	PrintableString	Example: 4, “MSTC”
Product model	7, “DVMODEL”	PrintableString	Example: 3, “JOP”

Note: The contents of the device category parameter are indicated below:
“PC” or “PLC”: Programmable controller
“NC” or “CNC”: Computer numerical control device
“RC” or “ROBOT”: Robot controller
“COMPUTER”: Personal computer, panel computer, workstation, display, or other computer
“SP_*··*”: To be specified by the vendor as a vendor-specific entry
(*··*: single-byte alphanumeric characters)
“OTHER”: Other

As a syntactical rule for transfer, the entire system parameter set or common parameter set, the system parameter revision date, or the entire device-specific parameter set (optional) must be of a SEQUENCE structure type (device-specific parameter inner structuring is optional).

9.9.3 Details of device-specific parameters (when used)

Table 9-17 Details of Device-Specific Parameters

Parameter name	Name character string	Data type	Contents of parameter
Device-specific parameter identification string	2, “ID”	PrintableString	7, “DEVPARA”
The vendor freely defines the contents for each device.			

9.9.4 System parameter examples (PLC examples)

(1) PLC example of system parameter tabular document notation

Table 9-18 System Parameter Tabular Document Notation (PLC Examples)

Parameter name	Name character string [PrintableString type] (Length), (String)	Data type [type]	Contents of parameter (Length), (Contents)
SysPara			
Device profile common specification version	6, “COMVER”	INTEGER	1,1
System parameter identification string	2, “ID”	PrintableString	7, “SYSPARA”
System parameter revision number	3, “REV”	INTEGER	1, 0
System parameter revision date	7, “REVDATE”	[INTEGER], 2, (0001-9999), [INTEGER], 1, (01-12), [INTEGER], 1, (01-31)	2, 1998 1, 9 1, 30
Device category	10, “DVCATEGORY”	PrintableString	3, “PLC”
Vendor name	6, “VENDOR”	PrintableString	29, “MSTC-JOP Electric Corporation”
Product model	7, “DVMODEL”	PrintableString	5, “PLC-M”

Parameter name	Name character string	Data type	Contents of parameter
PlcmPara			
Device-specific parameter identification string	2, “ID”	PrintableString	7, “DEVPARA”
CPU1 name	8, “CPU1NAME”	PrintableString	9, “PMSP35-5N”
CPU2 name	8, “CPU2NAME”	PrintableString	9, “PMSP25-2N”
CPU3 name	8, “CPU3NAME”	PrintableString	9, “PMSP25-2N”
CPU4 name	8, “CPU4NAME”	PrintableString	9, “PMBP20-0N”
Module 105 name	9, “IO105NAME”	PrintableString	9, “PMWD64-4N”
Module 106 name	9, “IO106NAME”	PrintableString	9, “PMLD01-0N”
Module 107 name	9, “IO107NAME”	PrintableString	9, “PMLE01-5N”

(2) Abstract syntax

<Type definition>

```
PlcmRecord ::=SEQUENCE
                    {
                    syspara              SysparaType,
                    plcmpara             PlcmType
                    }
SysparaType::=SEQUENCE
                    {
                    nameCOMVER           NameType,
                    paraCOMVER           INTEGER,
                    nameID               NameType,
                    paraID               NameType,
                    nameREV              NameType,
                    paraREV              INTEGER,
                    nameREVDATE          NameType,
                    paraREVDATE          DateType,
                    nameDVCATEGORY       NameType,
                    paraDVCATEGORY       NameType,
                    nameVENDOR           NameType,
                    paraVENDOR           NameType,
                    nameDVMODEL          NameType,
                    paraDVMODEL          NameType
                    }
PlcmType::=         SEQUENCE
                    {
                    nameID               NameType,
                    paraID               NameType,
                    module               SEQUENCE OF ModInfo
                    DEFAULT { }
                    }
 NameType::=        PrintableString
 DateType::=        SEQUENCE
                    {
                    year                 INTEGER,
                    month                INTEGER,
                    day                  INTEGER
                    }
ModInfo::=          SEQUENCE
                    {
                    nameMODULE           NameType,
                    paraMODULE           NameType
                    }
```

<Value definition>

```
{
        syspara   {
                  nameCOMVER          "COMVER",
                  paraCOMVER          1,
                  nameID              "ID",
                  paraID              "SYSPARA",
                  nameREV             "REV",
                  paraREV             0,
                  nameREVDATE         "REVDATE",
                  paraREVDATE         {     year    1998,
                                      month   9,
                                      day     30      },
                  nameDVCATEGORY      "DVCATEGORY",
                  paraDVCATEGORY      "PLC",
                  nameVENDOR          "VENDOR",
                  paraVENDOR          "MSTC-JOP Electric Corporation",
                  nameDVMODEL         "DVMODEL",
                  paraDVMODEL         "PLC-M"
                  }
        plcmpara  {
                  nameID              "ID",
                  paraID              "DEVPARA",
                  module
                  {
                  {       nameMODULE  "CPU1NAME",
                  paraMODULE  "PMSP35-5N"   },
                          {   nameMODULE  "CPU2NAME",
                              paraMODULE  "PMSP25-2N"   },
                          {   nameMODULE  "CPU3NAME",
                              paraMODULE  "PMSP25-2N"   },
                          {   nameMODULE  "CPU4NAME",
                  paraMODULE  "PMBP20-0N"   },
                          {   nameMODULE  "IO105NAME",
                              paraMODULE  "PMWD64-4N"   },
                          {   nameMODULE  "IO106NAME",
                              paraMODULE  "PMLD01-0N"   },
                          {   nameMODULE  "IO107NAME",
                              paraMODULE  "PMLE01-5N"}
                      }
              }
}
```

(3) Transfer syntax data array (encoding)

Identifier		Length	Contents
$30		$820133	
$30		$7D	
	$13	$06	"COMVER"
	$02	$01	1
	$13	$02	"ID"
	$13	$07	"SYSPARA"
	$13	$03	"REV"
	$02	$01	0
	$13	$07	"REVDATE"
	$30	$0A	
	$02	$02	$07CE
	$02	$01	$09
	$02	$01	$1E
	$13	$0A	"DVCATEGORY"
	$13	$03	"PLC"
	$13	$06	"VENDOR"
	$13	$1D	"MSTC-JOP Electric Corporation"
	$13	$07	"DVMODEL"
	$13	$05	"PLC-M"
	$30	$81B1	
	$13	$02	"ID"
	$13	$07	"DEVPARA"
	$30	$15	
	$13	$08	"CPU1NAME"
	$13	$09	"PMSP35-5N"
	$30	$15	
	$13	$08	"CPU2NAME"
	$13	$09	"PMSP25-2N"
	$30	$15	
	$13	$08	"CPU3NAME"
	$13	$09	"PMSP25-2N"
	$30	$15	
	$13	$08	"CPU4NAME"
	$13	$09	"PMBP20-0N"
	$30	$16	
	$13	$09	"IO105NAME"
	$13	$09	"PMWD64-4N"
	$30	$16	
	$13	$09	"IO106NAME"
	$13	$09	"PMLD01-0N"
	$30	$16	
	$13	$09	"IO107NAME"
	$13	$09	"PMLE01-5N"

(4) Data arrangement on circuit

The sequence of data transmitted over a circuit is shown below.

Data transmission begins with address (0) in the relative address 00 column shown below.

Data is transmitted, byte by byte, in the order indicated by the horizontal arrow mark. Upon completion of the transmission of the data in the relative address 00 column, the data in the relative address 10 column begins to be transmitted. Transmission continues in the same manner in the order of relative address.

Relative address → ↓ Data (in hexadecimal notation)

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(A)	(B)	(C)	(D)	(E)	(F)
00	30	82	01	33	30	7D	13	06	“C”	“O”	“M”	“V”	“E”	“R”	02	01
10	01	13	02	“I”	“D”	13	07	“S”	“Y”	“S”	“P”	“A”	“R”	“A”	13	03
20	“R”	“E”	“V”	02	01	00	13	07	“R”	“E”	“V”	“D”	“A”	“T”	“E”	30
30	0A	02	02	07	CE	02	01	09	02	01	1E	13	0A	“D”	“V”	“C”
40	“A”	“T”	“E”	“G”	“O”	“R”	“Y”	13	03	“P”	“L”	“C”	13	06	“V”	“E”
50	“N”	“D”	“O”	“R”	13	1D	“M”	“S”	“T”	“C”	“-”	“J”	“O”	“P”	“ ”	“E”
60	“l”	“e”	“c”	“t”	“r”	“i”	“c”	“ “	“C”	“o”	“r”	“p”	“o”	“r”	“a”	“t”
70	“I”	“o”	“n”	13	07	“D”	“V”	“M”	“O”	“D”	“E”	“L”	13	05	“P”	“L”
80	“C”	“-”	“M”	30	81	B1	13	02	“I”	“D”	13	07	“D”	“E”	“V”	“P”
90	“A”	“R”	“A”	30	15	13	08	“C”	“P”	“U”	“1”	“N”	“A”	“M”	“E”	13
A0	09	“P”	“M”	“S”	“P”	“3”	“5”	“-”	“5”	“N”	30	15	13	08	“C”	“P”
B0	“U”	“2”	“N”	“A”	“M”	“E”	13	09	“P”	“M”	“S”	“P”	“2”	“5”	“-”	“2”
C0	“C”	“N”	30	15	13	08	“P”	“U”	“3”	“N”	“A”	“M”	“E”	13	09	“P”
D0	“M”	“S”	“P”	“2”	“5”	“-”	“2”	“N”	30	15	13	08	“C”	“P”	“U”	“4”
E0	“N”	“A”	“M”	“E”	13	09	“P”	“M”	“B”	“P”	“2”	“0”	“-”	“0”	“N”	30
F0	16	13	09	“I”	“O”	“1”	“0”	“5”	“N”	“A”	“M”	“E”	13	09	“P”	“M”
100	“W”	“D”	“6”	“4”	“-”	“4”	“N”	30	16	13	09	“I”	“O”	“1”	“0”	“6”
110	“-”	“0”	“N”	30	16	13	09	“I”	“O”	“1”	“0”	“7”	“N”	“A”	“M”	“E”
120	13	09	“N”	“A”	“M”	“E”	13	09	“P”	“M”	“L”	“D”	“0”	“1”	“P”	“M”
130	“L”	“E”	“0”	“1”	“-”	“5”	“N”									

9.9.5 System parameter examples (CNC examples)

Table 9-19 shows CNC system parameter examples.

Table 9-19 System Parameter Tabular Document Notation (CNC Examples)

Parameter name	Name character string [PrintableString type] (Length), (String)	Data type [type]	Contents of parameter (Length), (Contents)
SysPara			
Device profile common specification version	6, "COMVER"	INTEGER	1,1
System parameter identification string	2, "ID"	PrintableString	7, "SYSPARA"
System parameter revision number	3, "REV"	INTEGER	1, 0
System parameter revision date	7, "REVDATE"	[INTEGER], 2, (0001-9999), [INTEGER], 1, (01-12), [INTEGER], 1, (01-31)	2, 1998 1, 9 1, 30
Device category	10, "DVCATEGORY"	PrintableString	3, "CNC"
Vendor name	6, "VENDOR"	PrintableString	9, "MSTCJ LTD"
Product model	7, "DVMODEL"	PrintableString	16, "MSTCJ Series 16a"

Parameter name	Name character string	Data type	Contents of parameter
CncPara			
Device-specific parameter identification string	2, "ID"	PrintableString	7, "DEVPARA"
Model name	5, "MODEL"	PrintableString	8, "MS16a-MA"
Series	6, "SERIES"	PrintableString	4, "MSF1"
Revision	3, "REV"	INTEGER	1, 0
System	7, "System"	SEQUENCE	*
System information	7, "SysInfo"	SEQUENCE	*

* This parameter is a Constructed type and has the data listed below.

System			
Option configuration flag	5, "SFLAG"	BIT STRING	8, "00100101"
Control axis count	4, "AXES"	INTEGER	2, 4

SysInfo			
Input virtual address	2, "IN"	OCTET STRING	6, "000000"
Output virtual address	3, "OUT"	OCTET STRING	6, "040000"

(1) Abstract syntax

<Type definition>

```
CncRecord ::=       SEQUENCE
                    {
                    SysPara                 SysParaType,
                    CncPara                 CncParaType,
                    }
SysParaType::=      SEQUENCE
                    {
                    nameCOMVER              NameType,
                    paraCOMVER              INTEGER,
                    nameID                  NameType,
                    paraID                  NameType,
                    nameREV                 NameType,
                    paraREV                         INTEGER,
                    nameREVDATE             NameType,
                    paraREVDATE             DateType,
                    nameDVCATEGORY          NameType,
                    paraDVCATEGORY          NameType,
                    nameVENDOR              NameType,
                    paraVENDOR              NameType,
                    nameDVMODEL             NameType,
                    paraDVMODEL             NameType
                    }
CncParaType::=      SEQUENCE
                    {
                    nameID                  NameType,
                    paraID                  NameType,
                    nameMODEL               NameType,
                    paraMODEL               NameType,
                    nameSERIES              NameType,
                    paraSERIES              NameType,
                    nameREV                 NameType,
                    paraREV                         INTEGER,
                    nameSystem              NameType,
                    paraSystem              SystemType,
                    nameSysInfo             NameType,
                    paraSysInfo             SysInfoType
                    }
SystemType::=       SEQUENCE
                    {
                    nameINPUT               NameType,
                    paraINPUT               BIT STRING,
                    nameAXES                NameType,
                    paraAXES                INTEGER
                    }
SysInfoType::=      SEQUENCE
                    {
                    nameIN                  NameType,
                    paraIN                  OCTET STRING,
                    nameOUT                 NameType,
                    paraOUT                         OCTET STRING
                    }
NameType:=          PrintableString
DateType::=                 SEQUENCE
                    {
                    year                    INTEGER,
                    month                   INTEGER,
                    day                     INTEGER
                    }
```

<Value definition>

```
{
SysPara         {
                nameCOMVER          "COMVER",
                paraCOMVER          1,
                nameID              "ID",
                paraID              "SYSPARA",
                nameREV             "REV",
                paraREV             0,
                nameREVDATE         "REVDATE",
                paraREVDATE         {       year     1998,
                                            month    9,
                                            day      30              },
                nameDVCATEGORY      "DVCATEGORY",
                paraDVCATEGORY      "CNC",
                nameVENDOR          "VENDOR",
                paraVENDOR          "MSTCJ LD",
                nameDVMODEL         "DVMODEL",
                paraDVMODEL         "MSTCJ Series 16a"
                }

CncPara         {
                nameID              "ID",
                paraID              "DEVPARA",
                nameMODEL           "MODEL",
                paraMODEL           "MS16a-MA",
                nameSERIES          "SERIES",
                paraSERIES          "MSF1",
                nameREV             "REV",
                paraREV             0,
                nameSystem          "System",
                paraSystem          {       nameINPUT     "SFLAG",
                                            paraINPUT     '00100101'B,
                                            nameAXES      "AXES",
                                            paraAXES      4                 },
                nameSysInfo         "SysInfo",
                paraSysInfo         {       nameIN        "IN",
                                            paraIN        '000000000000'H
                                            nameOUT       "OUT",
                                            paraOUT       '000400000000'H   }
               }
}
```