User's Manual

R-IN32M3 Series

User's Manual

(CC-Link IE Field Intelligent device station)

- R-IN32M3-CL

All information of mention is things at the time of this document publication, and Renesas Electronics may change the product or specifications that are listed in this document without a notice. Please confirm the latest information such as shown by website of Renesas

Document number:R18UZ0015EJ0500
Issue date : Jan 31, 2018

Renesas Electronics
www.renesas.com

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.
(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

Instructions for the use of product

In this section, the precautions are described for over whole of CMOS device.
Please refer to this manual about individual precaution.
When there is a mention unlike the text of this manual, a mention of the text takes first priority

1.Handling of Unused Pins

Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

-The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2.Processing at Power-on

The state of the product is undefined at the moment when power is supplied.

-The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3.Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

-The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4.Clock Signals

After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

-When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

How to use this manual

1. Purpose and target readers

This manual is intended for users who wish to understand the functions of “ CC-Link IE Field Nework of intelligent device station” for designing application of it.

It is assumed that the reader of this manual has general knowledge in the fields of electrical engineering, logic circuits, and microcontrollers.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

Related Documents

Literature may be preliminary versions. Note, however, that the following descriptions do not indicate "Preliminary". Some documents on cores were created when they were planned or still under development. So, they may be directed to specific customers. Last four digits of document number(described as ****) indicate version information of each document. Please download the latest document from our web site and refer to it.

The document related to CC-Link IE Field Network

Document name	Document number
R-IN32M3 Series Datasheet	R18DS0008EJ****
R-IN32M3-CL User’s Manual	R18UZ0005EJ****
R-IN32M3 series User’s Manual Peripheral function	R18UZ0007EJ****
R-IN32M3 Series Proguraming Manual (OS edition)	R18UZ0011EJ****
R-IN32M3 Series Proguraming Manual (Driver edition)	R18UZ0009EJ****
R-IN32M3 Series User’s Manual CC-Link IE Intelligent device station	This manual

2. Notation of Numbers and Symbols

Weight in data notation: Left is high-order column, right is low-order column

Active low notation:

xxxZ (capital letter Z after pin name or signal name)

or xxx_N (capital letter _N after pin name or signal name)

or xxnx (pin name or signal name contains small letter n)

Note:

explanation of (Note) in the text

Caution:

Item deserving extra attention

Remark:

Supplementary explanation to the text

Numeric notation:

Binary ··· xxxx , xxxxB or n'bxxxx (n bits)

Decimal ··· xxxx

Hexadecimal ··· xxxxH or n'hxxxx (n bits)

Prefixes representing powers of 2 (address space, memory capacity):

K (kilo)··· $2^{10} = 1024$

M (mega)··· $2^{20} = 1024^2$

G (giga)··· $2^{30} = 1024^3$

Data Type:

Double word ··· 32 bits

Word ··· 16 bits

Byte ··· 8 bits

Contents

Contents of Figures

Contents of Tables

1. OVERVIEW

This manual describes how to develop an intelligent device station using "Communication LSI R-IN32M3-CL for CC-Link IE Field Network intelligent device station".

The main information included in this manual is as follows:

- User program design
- R-IN32M3-CL driver specifications

1.1 R-IN32M3-CL Performance Specifications

The following table lists the R-IN32M3-CL performance specifications.

Table 1.1 Performance Specifications (Overview)

Item	Specifications	
Station type	Intelligent device station	
Station number	1 to 120	
Network number	1 to 239	
Communication speed	1 Gbps	
Network topology	Line, star, and ring (Coexistence of line topology and star topology is possible.)	
Connection cable	Ethernet cable that satisfies 1000BASE-T standards: Category 5e or higher (double shielded, STP), straight cable	
Maximum station-to-station distance	100 m	
Overall cable distance	Link topology: 12000m (when cables are connected to 1 master station and 120 slave stations) Star topology: Depends on the system configuration. Ring topology: 12100m (when cables are connected to 1 master station and 120 slave stations)	
Number of cascade connections	Up to 20	
Maximum number of link points per station	RX	2048 points (2048 bits), 256 bytes
	RY	2048 points (2048 bits), 256 bytes
	RWr	1024 points (1024 words), 2048 bytes
	RWw	1024 points (1024 words), 2048 bytes

1.2 Preparing for Development

This section describes the preparations and investigations to be made before development.

The following is an example of the user development process. The preparations and investigations to be made in each step are described in the sections that follow.

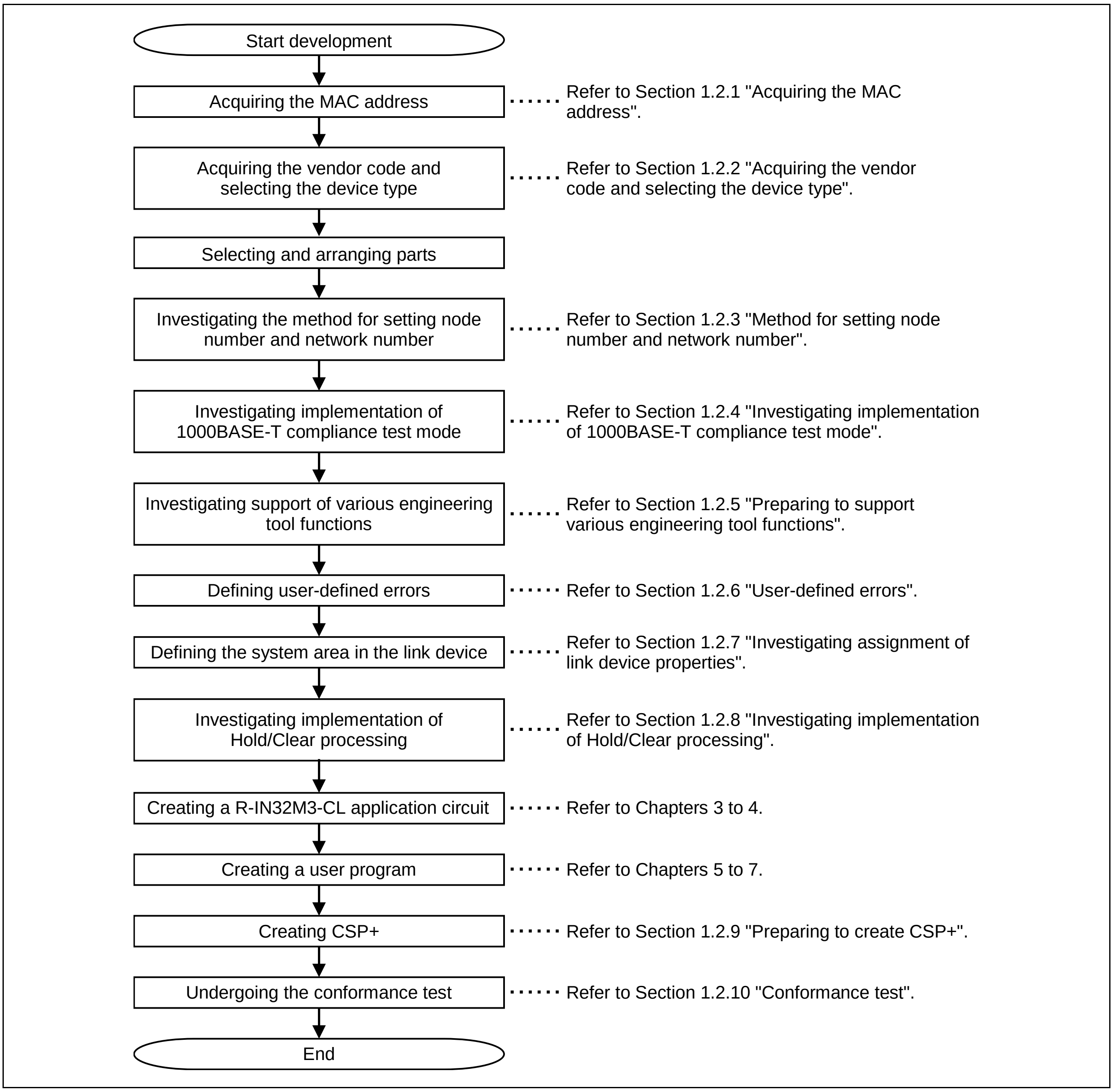

Start development
Acquiring the MAC address
Refer to Section 1.2.1 "Acquiring the MAC address".
Acquiring the vendor code and selecting the device type
Refer to Section 1.2.2 "Acquiring the vendor code and selecting the device type".
Selecting and arranging parts
Investigating the method for setting node number and network number
Refer to Section 1.2.3 "Method for setting node number and network number".
Investigating implementation of 1000BASE-T compliance test mode
Refer to Section 1.2.4 "Investigating implementation of 1000BASE-T compliance test mode".
Investigating support of various engineering tool functions
Refer to Section 1.2.5 "Preparing to support various engineering tool functions".
Defining user-defined errors
Refer to Section 1.2.6 "User-defined errors".
Defining the system area in the link device
Refer to Section 1.2.7 "Investigating assignment of link device properties".
Investigating implementation of Hold/Clear processing
Refer to Section 1.2.8 "Investigating implementation of Hold/Clear processing".
Creating a R-IN32M3-CL application circuit
Refer to Chapters 3 to 4.
Creating a user program
Refer to Chapters 5 to 7.
Creating CSP+
Refer to Section 1.2.9 "Preparing to create CSP+".
Undergoing the conformance test
Refer to Section 1.2.10 "Conformance test".
End

Figure 1.1 Development Process Example

1.2.1 Acquiring the MAC address

CC-Link IE Field Network devices are Ethernet (IEEE 802.3ab) compliant. Be sure to acquire a MAC address MA-L (MAC Address Block Large) unique to the device. To acquire a MAC address, contact the following authority (department) in the USA.

The IEEE Registration Authority
445 Hoes Lane
Piscataway, NJ 08854 USA
Phone: +1 (732) 465-6481
Fax: +1 (732) 562-1571
Web: http://standards.ieee.org/develop/regauth/oui/

1.2.2 Acquiring the vendor code and selecting the device type

CC-Link IE Field Network devices require registration of a vendor code and device type. The vendor code and device type are assigned and managed by the CC-Link Partner Association. If you have any questions, contact the CC-Link Partner Association.

Table 1.2 Vendor Code and Device Type

Item	Description
Vendor code (vendorCode)	ID number (fifth to eighth digits) issued when the vendor joined the CC-Link Partner Association.Note
Device type (deviceType)	Select the applicable device type from the "CC-Link IE Field Network Specification (Device Profile)". If an applicable device type does not exist, consult with the CC-Link Partner Association.

Note. If the ID number is 123-456-7890, the vendor code is 5678.

1.2.3 Method for setting node number and network number

To create a data link to the own station, a node number and network number need to be set in R-IN32M3-CL. Investigate method for setting the node number and network number in accordance with the specifications of the developed device in advance.

For example, the node number and network number can be set by using a hardware switch or by using the engineering tool of the developed product.
For either method, use "gerR_IN32_SetNodeAndNetworkNumber" (Refer to Section 6.4.1(3) "gerR_IN32_SetNodeAndNetworkNumber") of the R-IN32M3-CL driver interface functions in the user program "iUserInitialization" (Refer to Section 6.2.2 "Initialization processing").

The following describes examples of how to set the node number and network number.

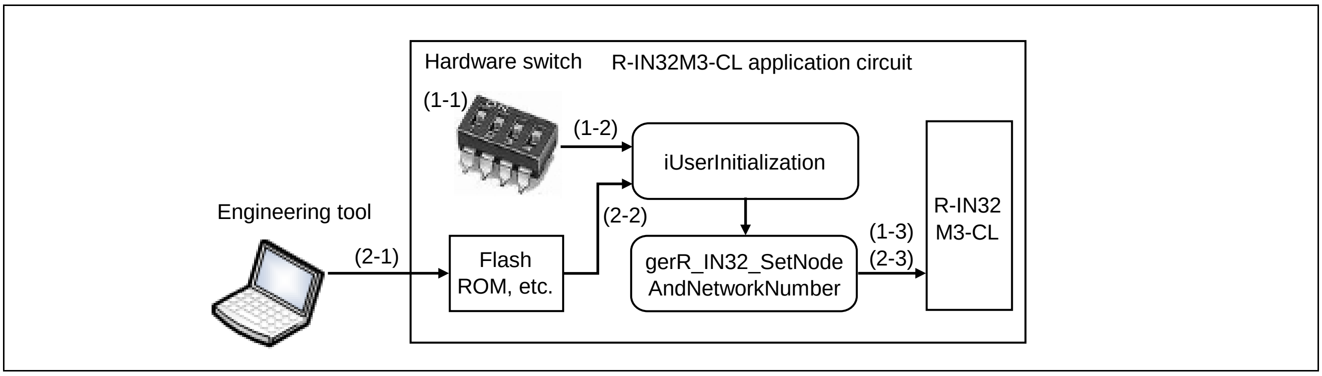


Figure 1.2 Image of Setting Node Number and Network Number

Table 1.3 Using a hardware switch (Example1)

Step	Description
1-1	Set the node number and network number using a hardware switch.
1-2	The user program "iUserInitialization" acquires the current values of the hardware switch, and sets the values in the arguments of the R-IN32M3-CL driver interface function "gerR_IN32_SetNodeAndNetworkNumber". The process for acquiring the current values of the hardware switch is not written in the user program "iUserInitialization". Add the process in accordance with user specifications.
1-3	The R-IN32M3-CL driver interface function "gerR_IN32_SetNodeAndNetworkNumber" sets the argument values in R-IN32M3-CL.

Select a hardware switch that corresponds to the value range of the node number and network number.

Table 1.4 Hardware Switch Range

Item	Value Range
Node number	01h to 78h (1 to 120)
Network number	01h to EFh (1 to 239)

Table 1.5 Using the engineering tool (Example 2)

Step	Description
2-1	Set the node number and network number data in Flash ROM or the like using the engineering tool.
2-2	The user program "iUserInitialization" acquires the data set in Flash ROM, and sets the data in the arguments of the R-IN32M3-CL driver interface function "gerR_IN32_SetNodeAndNetworkNumber". The process for acquiring the data set in Flash ROM is not written in the user program "iUserInitialization". Add the process in accordance with user specifications.
2-3	The R-IN32M3-CL driver interface function "gerR_IN32_SetNodeAndNetworkNumber" sets the argument values in R-IN32M3-CL.

1.2.4 Investigating implementation of 1000BASE-T compliance test mode

CC-Link IE Field Network compatible products are 1000BASE-T compliant, and require implementation of the 1000BASE-T compliance test based on IEEE 802.3ab specifications.

Caution. Verify the results of the 1000BASE-T compliance test using the CC-Link Partner Association conformance test.

The 1000BASE-T compliance test measures four test waveforms from the Ethernet ports as verification of transmission path waveforms.

Consider whether to implement functions and processing that switch the test waveform at desired timings during testing in the developed device.

For example, the test waveform can be switched by using a hardware switch or by using the engineering tool (peripheral device) of the developed device.

For either method, call "gerR_IN32_IEEETest" (Refer to Section 6.4.13(1) "gerR_IN32_IEEETest") of the R-IN32M3-CL driver interface functions from the user program "UserIEEETest" (Refer to Section 6.2.45 "Hardware test (IEEE 802.3ab compliance test)").

Table 1.6 Switching the 1000BASE-T compliance test mode (Example)

Step	Description
1	Implement a function that switches the mode to a "compliance test mode (offline mode)", which is other than the mode (online mode) used during normal operation, from an external source (such as a hardware switch) of the developed device.
2	Implement processing that can specify the arguments "MODE1" to "MODE4" of the R-IN32M3-CL driver interface function "gerR_IN32_IEEETest" from an external source (such as a hardware switch) of the developed device.

1.2.5 Preparing to support various engineering tool functions

The CC-Link IE Field Network diagnostics and parameter processing/command execution of slave stations can be performed using the engineering tool. Consider whether or not the specifications of the developed device (slave station) will support engineering tool functions in advance.

[CC-Link IE Field Network diagnostics]

The engineering tool graphically displays the status of CC-Link IE Field Network.

For details, refer to Section 2.6 "CC-Link IE Field Network Diagnostics".

[Parameter processing/command execution of slave stations]

By using the engineering tool, parameter setup and command execution can be performed on the developed device without programming.

For details, refer to Section 1.2.9(1) "Parameter processing/command execution of slave stations".

The above two functions are performed using transient transmission (SLMP frames).

The developed device (slave station) needs to respond to SLMP request frames from the master station.

Consider whether to implement SLMP frame request reception and response send processing (For details, refer to Section 6.2.18 "Transient1 receive data processing") in advance.

Table 1.7 Engineering Tool Functions

No.	Engineering Tool Function		Items Required in Developed Devices
1	CC-Link IE Field Network diagnostics		・SLMP frame request reception and response send processing
	a	Selected station communication status monitor	
	b	Communication test	
	c	Cable test	
2	Parameter processing/command execution of slave stations		・Describe CSP+ up to scope [3] in Figure 1.3 ・SLMP frame request reception and response send processing

1.2.6 Investigating error status and error code specifications

When an error occurs in a developed device, the error status is reported to other stations.
In addition, when a request frame is abnormally received during transient transmission, the error code is reported to the request source.
Investigate the specifications of error status and error code in advance in accordance with the specifications of the developed device.

For error status specifications, refer to this section, (1) "Investigating error status specifications".
For transient transmission error code specifications, refer to this section, (2) "Investigating specifications of error codes stored in return codes (RSTS) and end codes (End Code)".

[Error codes for errors other than transient transmission errors]

Error codes include those for R-IN32M3-CL hardware errors and for user application area[Note] errors.

Note. User-defined function areas (I/O areas, sensor areas, temperature control areas, etc.)

The error codes for R-IN32M3-CL hardware errors are detected by the R-IN32M3-CL driver but not reported to other stations. The error processing after detection is optional. (Refer to Section 6.6(1) "gR_IN32_CallbackFatalError ".)

The error codes for user application area errors are not reported to other stations. The definitions and implementation are optional.

(1) Investigating error status specifications

The R-IN32M3-CL driver stores its own station error status in the detailed application error status of the MyStatus frame, and reports the status to other stations. (Refer to Section 5.4 "MyStatus Overview".)

Define minor, moderate, and major error statuses in accordance with developed device specifications.
For a reference when defining error status, the error statuses of the programmable controller CPU module are classified as follows:

- Minor error : An error in which the CPU module continues operation, such as a battery error
- Moderate error : An error in which the CPU module stops operation, such as a WDT error
- Major error : An error in which the CPU module stops operation, such as a RAM error
  (Errors that are more serious than moderate errors that may involve hardware failure are considered as major errors.)

(2) Investigating specifications of error codes stored in return codes (RSTS) and end codes (End Code)

When a developed device abnormally receives Transient2 request frame, the error code is stored in the return code (RSTS) of the response frame and sent to the request source to report the error and cause.
Store error codes in the return code (RSTS) in accordance with the specifications of the developed device so that the end user can check the return code and take action using the user's manual of the developed device.
For details, refer to Section 5.3.4 "CC-Link compatible transient frame format".

When a developed device abnormally receives SLMP request frame, as is the case with the Transient2 request frame, store the error code in the end code (End Code) in accordance with the specifications of the developed device.
For details, refer to Section 5.3.5 "SLMP frame format".

1.2.7 Investigating assignment of link device properties

Link devices are classified into three property groups. Investigate the property groups to be assigned to the link devices of the developed device in advance.

(For details, refer to "CC-Link IE Field Network Specification (Device Profile)".)

(1) Direct input/output group

The direct input/output group uses link devices for general input/output and not for specific functions.

(For example, input/output of master/local modules, remote I/O modules, and others.)

(2) System input/output group

The system input/output group uses link devices for interlock between the master station and its own station, notification of own station status.

(For details, refer to Chapter 7 "LINK DEVICE SYSTEM AREA")

Table 1.8 Example of Link Device Property Definitions (System Area)

Link Device	Name	Link Device	Name
RX07	Warning status flag	-	-
RX08	Initial data processing request flag	RY08	Initial data processing complete flag
RX09	Initial data setting complete flag	RY09	Initial data setting request flag
RX0A	Error status flag	RY0A	Error reset request flag
RX0B	Remote ready	-	-

(3) Vendor input/output group

The vendor input/output group uses arbitrarily defined link devices.

Table 1.9 Example of Link Device Property Definitions (System Area)

Link Device	Name	Link Device	Name
RX10	Forward rotation status/stop status	RY10	Forward rotation command/stop command
RX11	Reverse rotation status/stop status	RY11	Reverse rotation command/stop command
RWr00	Output frequency status	RWw00	Output frequency setting

1.2.8 Investigating implementation of Hold/Clear processing

Hold/Clear processing is processing in which the developed device continues (Hold) or stops (Clear) output when the developed device controls external output and cyclic transmission has stopped for reasons such as a master station application stop/error, or data link disconnection.

Consider the following precautions, and investigate implementation of Hold/Clear processing as a fail-safe for when a master station stop/error or data link disconnection occurs.

For details on the Hold/Clear processing in the case of a master station application stop/error, refer to Section 6.2.9 "MyStatus from master station and cyclic receive processing".
The master station application status can be monitored by receiving MyStatus frames.
For details on the master station application information acquired by the MyStatus frame, refer to Section 5.4.2 "Receiving MyStatus".

For details on the Hold/Clear processing in the case of data link disconnection, refer to Section 6.2.12 "Communication status update processing".

Caution. Cyclic data received in a slave station (own station) is acquired by the R-IN32M3-CL driver (gerR_IN32_GetReceivedCyclicData).
Contents of acquired cyclic data differ depending on the operation/error status or data link status of the master station application.
For details on the acquired cyclic data, refer to Section 6.2.9 "MyStatus from master station and cyclic receive processing" and Section 6.2.12 "Communication status update processing".

1.2.9 Preparing to create CSP+

CSP+ is specifications for describing required information for starting, operating, and maintaining CC-Link Family compatible products.

Providing CSP+ to the end users of the developed device allows them to manage all stations of CC-Link IE Field Network using one engineering tool.

For CSP+ details, refer to "Control & Communication System Protocol Specification".

To create CSP+, use "CSP+ Creation Support Tool".

The following shows the scope in which CSP+ files are to be created for the intelligent device station.

The conformance test includes CSP+ verification. Be sure to create CSP+ of scope [1].

Consider which functions (creation scopes [2] and [3]) of the engineering tool are to be supported by the specifications of the developed device in advance.

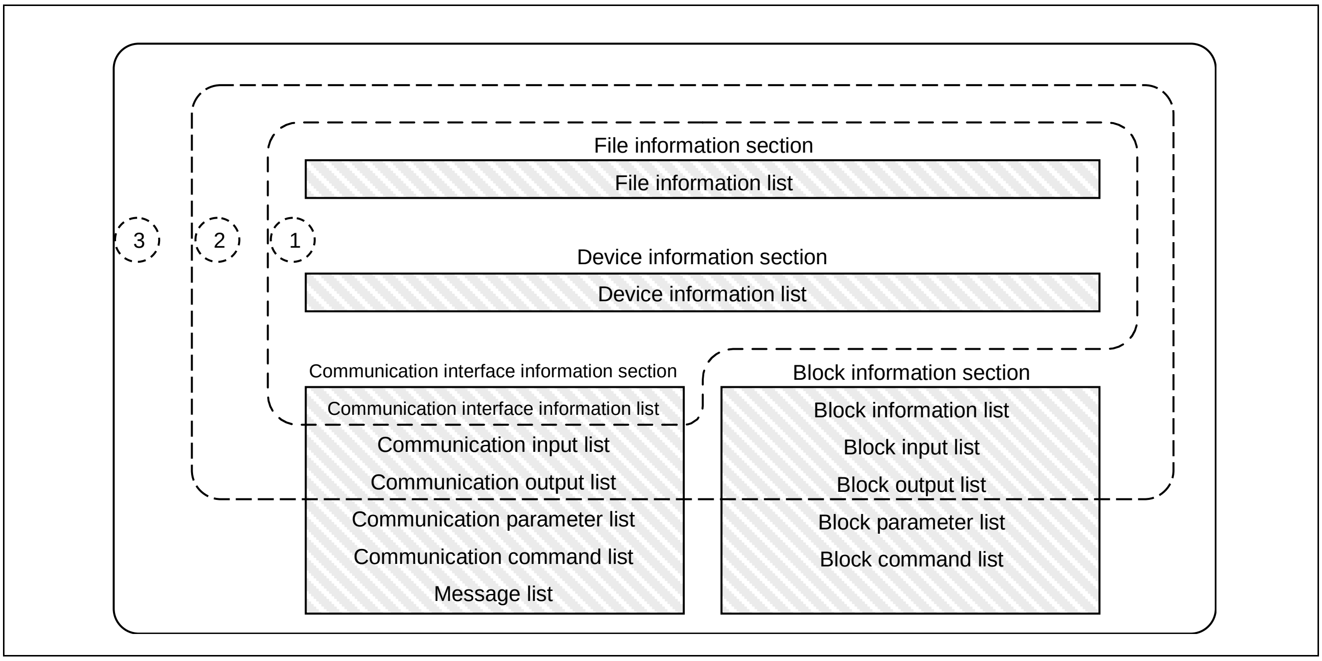

3
2
1
File information section
File information list
Device information section
Device information list
Communication interface information section
Communication interface information list
Communication input list
Communication output list
Communication parameter list
Communication command list
Message list
Block information section
Block information list
Block input list
Block output list
Block parameter list
Block command list

Figure 1.3 CSP+ File Section Configuration

Table 1.10 CSP+ File Section Configuration

Scope	Description	Necessity
[1]	Information required for verifying mandatory items in the CC-Link Partner Association conformance test [GX Works2, GX Works3] Developed devices are displayed in the CC IE Field Configuration window and the network configuration can be easily created.	Required
[2]	Information required for displaying slave station link device and master station device assignments	Optional
[3]	Information required for executing parameter processing/command execution of slave stations[Note] [GX Works2, GX Works3] The parameters of CC-Link IE Field Network compatible products can be easily set from the CC IE Field Configuration window.	Optional

Note. For details, refer to this section, (1) "Parameter processing/command execution of slave stations".

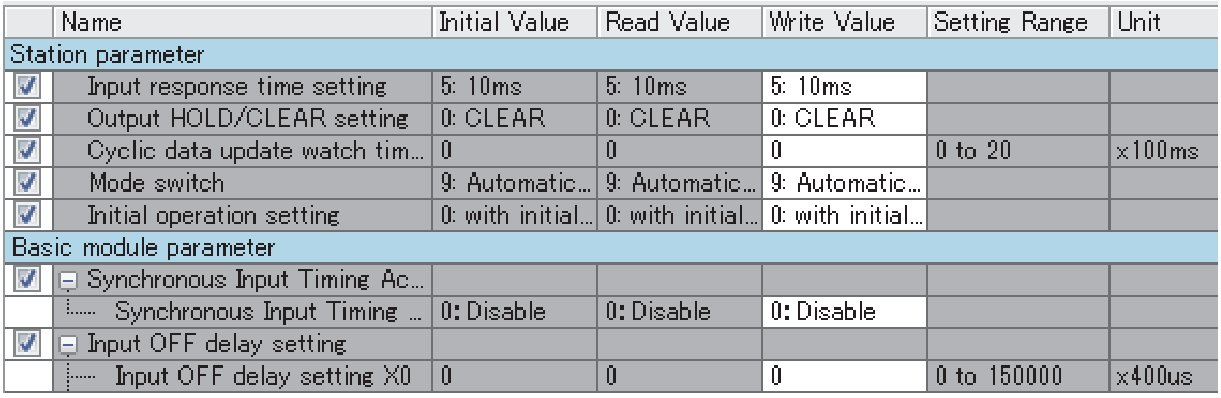
(1) Parameter processing/command execution of slave stations

Developed devices support parameter processing/command execution of slave stations, making it possible to reduce the programming required for parameter setup and command execution by the end user of the developed device.

Consider whether or not the specifications of the developed device will support parameter processing/command execution of slave stations by the engineering tool in advance.

[Example of parameter processing of slave station]

The following parameters can be set up on a Mitsubishi Electric remote I/O module (NZ2GF2B1-***) without programming.



	Name	Initial Value	Read Value	Write Value	Setting Range	Unit
Station parameter						
☑	Input response time setting	5: 10ms	5: 10ms	5: 10ms		
☑	Output HOLD/CLEAR setting	0: CLEAR	0: CLEAR	0: CLEAR		
☑	Cyclic data update watch tim...	0	0	0	0 to 20	x100ms
☑	Mode switch	9: Automatic...	9: Automatic...	9: Automatic...		
☑	Initial operation setting	0: with initial...	0: with initial...	0: with initial...		
Basic module parameter						
☑	⊟ Synchronous Input Timing Ac...					
	Synchronous Input Timing ...	0:Disable	0:Disable	0:Disable		
☑	⊟ Input OFF delay setting					
	Input OFF delay setting X0	0	0	0	0 to 150000	x400us

Figure 1.4 Example of Slave Station Parameter Processing by CC IE Field Configuration Window

Parameter processing/command execution of slave stations can be achieved by satisfying the following:

- Describe CSP+ up to scope [3] in Figure 1.3.
- Implement the SLMP frame send/receive processing described in CSP+ on the developed device.

1.2.10 Conformance test

The conformance test is a test implemented for each device in order to ensure high reliability in the communication of CC-Link IE Field Network compatible products. The test verifies that the product developed by a user satisfies the CC-Link IE Field Network communication specifications and is connectable to the network.

Acquire the conformance test specifications when preparing for development, and design the user product so that it satisfies the test requirement specifications.

A CC-Link IE Field Network compatible product that passes the conformance test can be included as a qualified product in the "CC-Link Partner Product Catalog" and other medium.

Remark. Some functions may not be supported depending on the development timing. When implementing the conformance test, contact the CC-Link Partner Association.

(1) Items required for the conformance test

Among the functions or processing described in this manual, the items described below are essential to implement the conformance test.

[Cyclic transmission function]

The cyclic transmission function is required throughout the conformance test.

Implement the processing whose "Implementation Required" is "Required" in Table 6.3.

[Transient transmission function]

The response to Transient1 detailed node information acquisition is necessary.

Implement the processing whose "Implementation Required" is "Required" in Table 6.4.

[1000BASE-T compliance test]

Transmission path waveforms must be verified based on IEEE 802.3 specifications.

Implement the processing described in Table 6.5.

[CSP+]

Create CSP+ of scope [1] in Figure 1.3.

2. R-IN32M3-CL FUNCTIONS

This chapter describes the functions supported by R-IN32M3-CL.

Table 2.1 R-IN32M3-CL Function List

Function	Overview
Bus access	Accesses 16/32-bit registers by an external 32-bit bus.
LED status display	・Displays status information (RUN, RD, SD, ERR, D LINK). ・Displays the port status (port 1 L ER, port 2 L ER). ・User LED x2.
Interrupt	・Outputs MPU interrupts. ・Inputs external WDTs. ・Outputs internal WDTs. ・Master watch timer
Reset	・Inputs power-on reset. ・Inputs system reset. ・Outputs PHY reset.
WDT	・Internal WDT ・External WDT
Bypass mode	Continues linkup even if an error that impacts communication occurs on the own station. Allows transmission of frames received by port 1 (port 2) using port 2 (port 1).
MyStatus/Cyclic send	Automatically creates a MyStatus/cyclic send frame via the R-IN32M3-CL driver by setting the address in which send data is stored in R-IN32M3-CL, and sends the frame.
MyStatus/Cyclic reception	Automatically writes the data of MyStatus/cyclic frame received from other stations to the specified storage location via the R-IN32M3-CL driver.
MDIO	Comprises an interface for PHY initialization and status monitoring. Accessible only if the MAC access is enabled.
MIB (statistical) information	Acquires the statistical information of 2 ports, including information on HEC error frame reception or DCS/FCS error frames reception.
Transient send	Sends a transient frame via the R-IN32M3-CL driver by setting the address in which send data is stored in R-IN32M3-CL.
Transient reception	Writes the data of the transient frame received from other stations to the specified storage location via the R-IN32M3-CL driver.
CC-Link IE Field Network diagnostics	The status of CC-Link IE Field Network can be checked using the engineering tool. Error locations, error causes, corrective actions, and event history can be checked using the engineering tool.

2.1 Communication Functions

R-IN32M3-CL supports the communication functions of cyclic transmission, transient transmission, and MyStatus.

Table 2.2 Communication Function List

Name	Description
Cyclic transmission	Cyclically sends/receives data with the master station. R-IN32M3-CL automatically performs to send/receive the data of the cyclic transmission. Link devices (RX, RY, RWw, RWr) are used for the data communication. The following shows the data size handled by the intelligent device station. RX : 2048 bits (2048 points), 256 bytes RY : 2048 bits (2048 points), 256 bytes RWw : 1024 words (1024 points), 2048 bytes RWr : 1024 words (1024 points), 2048 bytes
Transient transmission	Sends/receives data when there is a communication request from a user program or another station. The following shows the functions and data size handled by the intelligent device station. Client function: Supported Server function: Supported Data size: 2048 bytes (data area size of a transient frame)
MyStatus send/receive	R-IN32M3-CL sets own station information in the MyStatus frame and notify the master station of it. It also receives the MyStatus frame from the master station and monitors the status of the master station.

2.2 Status Display Function

R-IN32M3-CL can display the status of the own station and the status of the ports using LEDs.
For details of each LED, refer to Chapter 4 "STATUS DISPLAY FUNCTION".

2.3 Interrupts

R-IN32M3-CL supports four interrupt functions.

The interrupt functions include the "MPU interrupt function", "master watch timer function", "internal WDT function", and "external WDT function".
The internal WDT function and the external WDT function cannot be used simultaneously. Make sure to use them exclusively.

Table 2.3 R-IN32M3-CL Interrupt List

Name	Signal Name	Interrupt Type	Description
MPU interrupt function	INTL	Output	The MPU interrupt function is used by R-IN32M3-CL to output the interrupt signal INTL at "Low" when an event occurs in a case where "MPU interrupt function use"[Note] set by the R-IN32M3-CL driver interface function gerR_IN32_Initialize is R_IN32_TRUE. The R-IN32M3-CL driver uses the function gerR_IN32_GetEvent to acquire R-IN32M3-CL events, and thus the vendor does not need to be aware of the interrupt signal (INTL).
Master watch timer function	-	Internal	The master watch timer function generates an interrupt when the master station malfunctions. R-IN32M3-CL monitors whether or not the reception interval of the MyStatus frame sent by the master station is within the timeout time to detect master station errors. The function generates an interrupt when R-IN32M3-CL detects a master station operation error. R-IN32M3-CL automatically receives the timeout time from the master station and sets the time thus received. For the processing performed when a master watch timer interrupt occurs, refer to Section 6.2.8 "Event processing".
Internal WDT function	INTL	Output	The internal WDT function generates an interrupt (outputs the interrupt signal NMIL at "Low") when the user program operates abnormally. At this time, R-IN32M3-CL changes to bypass mode. R-IN32M3-CL monitors whether or not the WDT reset interval from the user program is within the WDT monitoring time set by initial processing to detect user program errors. The user program implements processing that resets WDT within the WDT monitoring time. For internal WDT function setup, refer to Section 6.2.2 "Initialization processing".
External WDT function	WDTIL	Input	The external WDT function monitors whether or not the user program is operating normally using an external WDT detection circuit. If you want to use the external WDT function, mount a WDT detection circuit that detects user program errors and connect the circuit to the R-IN32M3-CL WDTIL pin. Design the circuit so that the interrupt signal WDTIL is held at "Low" after the external WDT detection circuit detects an error. When a "Low" signal is input to the interrupt signal WDTIL, R-IN32M3-CL recognizes the user program error and changes the mode to bypass mode.

Note. For details of "MPU interrupt function use", refer to Section 6.4.1(2) "gerR_IN32_Initialize".

2.4 Bypass Mode

Bypass mode is a function that maintains a network connection (linkup), even when system reset or an error that affects communication, such as a WDT error or own station error, occurs in a line or ring topology, so that communication with downstream stations from the own station is not affected.

2.5 MIB Information

R-IN32M3-CL counts the number of frame receptions, the number of error frame receptions, and the like per port, and stores that information in MIB as information for managing the communication status.

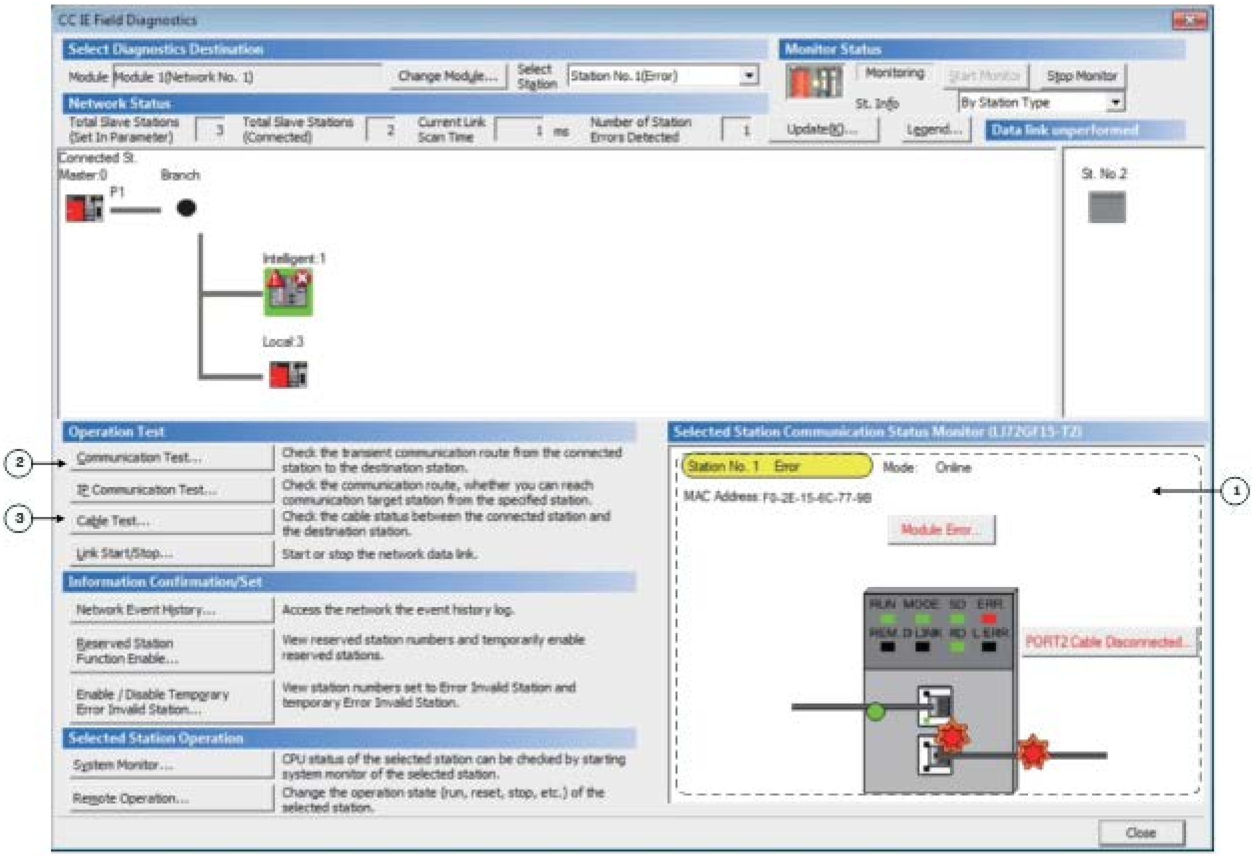
Vendors can use MIB information to identify the communication error status of port 1 and port 2 of the own station.

For MIB information details, refer to Sections (1), (2), and (3) of Section 6.2.14 "MIB information acquisition processing".

2.6 CC-Link IE Field Network Diagnostics

The CC-Link IE Field Network diagnostics graphically displays the status of CC-Link IE Field Network using the engineering tool. Error locations, error causes, corrective actions, and event history can be checked using the engineering tool. For function details, refer to the user's manual of the master/local module.

This function displays the developed device on the CC-Link IE Field Network diagnostics window by responding to SLMP frame requests from the master station. The function also allows you to execute various tests and operations.



Module: Module 1(Network No. 1)

Select Station: Station No. 1(Error)

Total Slave Stations (Set In Parameter): 3

Total Slave Stations (Connected): 2

Current Link Scan Time: 1 ms

Number of Station Errors Detected: 1

St. Info: By Station Type

Communication Test... — Check the transient communication route from the connected station to the destination station.

IP Communication Test... — Check the communication route, whether you can reach communication target station from the specified station.

Cable Test... — Check the cable status between the connected station and the destination station.

Link Start/Stop... — Start or stop the network data link.

Network Event History... — Access the network the event history log.

Reserved Station Function Enable... — View reserved station numbers and temporarily enable reserved stations.

Enable / Disable Temporary Error Invalid Station... — View station numbers set to Error Invalid Station and temporary Error Invalid Station.

System Monitor... — CPU status of the selected station can be checked by starting system monitor of the selected station.

Remote Operation... — Change the operation state (run, reset, stop, etc.) of the selected station.

Station No. 1 Error

Mode: Online

MAC Address: F0-2E-15-6C-77-9B

Figure 2.1 Diagnostic Window/Operation Locations (GX Works2)

Table 2.4 Diagnostic Window/Operation Locations and SLMP Requests

No.	Item	Description	SLMP Request Frame (Command)
1	Selected station communication status monitor	Displays the status of the selected station and error details.	Selected station communication status request (0x3119)
2	Communication test	Tests the communication path of transient transmission from the own station to the communication destination.	Communication test request (0x3040)
3	Cable test	Tests cable disconnection and no connection.	Cable test request (0x3050)

[SLMP request frame response]

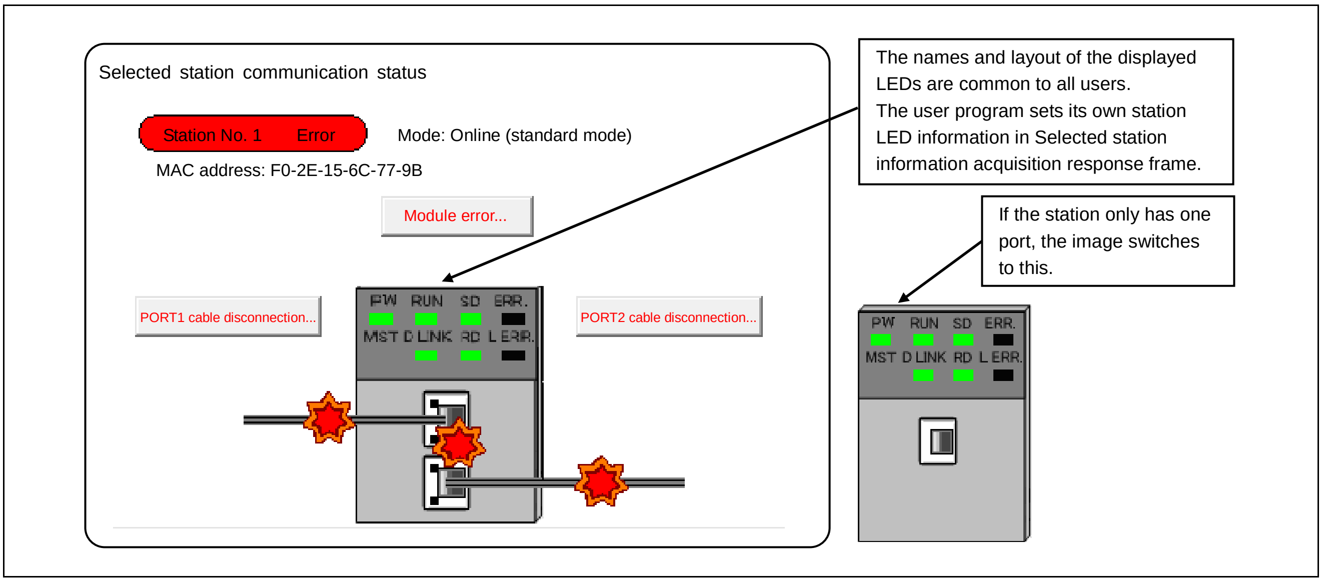
In the user program "UserHandleReceivedTransient1" (Section 6.2.18 "Transient1 receive data processing"), the applicable SLMP frame response processing (request frame receive processing) is performed.

The processing of the above No.1 to 3 is described in the sample code. Use the processing described.

(Implementation of the above No.1 to 3 is recommended.)

2.6.1 Selected station communication status monitor LEDs

The LED status of the own station can be displayed on the selected station communication status monitor by creating LED information in "UserHandleReceivedSelectInfoRequest" (Section 6.2.28 "Selected station information acquisition request frame receive processing") and issuing a response to the selected station communication status request.



Selected station communication status

Station No. 1 Error

Mode: Online (standard mode)

MAC address: F0-2E-15-6C-77-9B

The names and layout of the displayed LEDs are common to all users. The user program sets its own station LED information in Selected station information acquisition response frame.

If the station only has one port, the image switches to this.

Figure 2.2 Display Example of Selected Station Communication Status Monitor

[Example of LED use of selected station communication status monitor]

When the LED status of the developed device is not visible during end user troubleshooting, the LED status can be checked by using CC-Link IE Field Network diagnostics.

[Displayable LEDs]

The LED names[Note] and LED layout that can be displayed on the selected station communication status monitor are as shown in the figure above.

Note. PW, RUN, SD, ERR., MST (not used; grayed out), D LINK, RD, L ERR.

For details on creating LED information, refer to Section 6.4.11(9) "gulR_IN32_SetSelectInfo_Response".

3. Basic Design Precautions

3.1 Component selection

Select components taking into consideration the information provided in the table below.

Table 3.1 Component Selection Check Sheet

No.	Item	Description	Check
1	MPU selection	Did you select an MPU that satisfies the following specifications? (1)Data width: 16 bits or higher (2)Address width: 17 bits or higher (3)Endian: Little endian (4)Timing indicated in Chapter 4	
2	RJ-45 connector selection	Is the connector an 8-pin ANSI/TIA/EIA-568-B shielded connector?	
3	Pulse transformer selection	Did you select an IEEE 802.3 1000BASE-T compatible component?	
4	PHY selection	Did you select a component that satisfies the following specifications? (1)IEEE 802.3 1000BASE-T full duplex compatible component (2)Component having an auto negotiation function (3)Component having a GMII interface (4)Component having an auto MDI/MDIX negotiation function (5)Component capable of operating at an MDC clock frequency of 7.812 MHz	
5	125-MHz crystal oscillator selection	Did you select a component having a frequency deviation within ±50 ppm?	
6	2.097152-MHz crystal oscillator selection	Did you select a component having a frequency deviation with ±50 ppm?	
7	PHY clock crystal oscillator selection	Did you select a PHY clock crystal oscillator in accordance with the required specifications of the PHY used? Frequency of crystal oscillator Total jitter of crystal oscillator	

3.2 Circuit design

Design the peripheral circuits of R-IN32M3-CL taking into consideration the information provided in the table below.

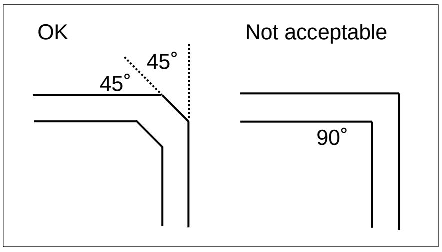
Table 3.2 Circuit Design Check Sheet

No.	Item	Description	Check
1	GMII wiring	Is a damping resistor installed for the GMII signal to supress overshooting/undershooting?	
2	PHY- RJ45 connctor connection	The signal lines between PHY and RJ45 connector mus be connected in + side and + side of each terminal, - side and – side of each terminal. Otherwise 1000BASE-T compliance test fails.	
3	Data signal	Are pull-up resistors installed for the data signals D15 to D00? (10-kΩ pull-up resistors are used in the circuit diagram examples.)	
4	PHY address	PHY address must be same as the port number of R-IN32M3-CL. PHY address 1 must be connected to MAC port 1. PHY address 2 must be connected to MAC port 2.	

3.3 Pattern design

Design the pattern wiring of the R-IN32M3-CL periphery taking into consideration the information provided in the table below.

Table 3.3 Pattern Design Check Sheet

No.	Item	Description	Check
1	2.097152-MHz crystal oscillator connected to R-IN32M3-CL	When connecting a 2.097152-MHz crystal oscillator to R-IN32M3-CL, place the oscillator near R-IN32M3-CL. Is the pattern length to the the CLK 2_097M pin shortest as possible? Is the pattern to the CLK 2_097M pin shielded by SG patterns?	
2	GMII wiring	Has the wiring layer and signal line thickness for the signal (GMII), which connects R-IN32M3-CL and PHY, been determined to achieve shortest pattern wiring and 50 Ω impedance?	
3	Signal pattern bending	When a pattern is bent, is it always bent at 45 degrees as shown below?  	
4	Power supply / GND pattern	Is the power supply / GND pattern wired using the thickest pattern possible?	

4. STATUS DISPLAY FUNCTION

4.1 Status Display by LEDs

A R-IN32M3-CL application circuit allows you to mount the own station status LEDs and the LEDs for indicating the port 1 status and port 2 status as shown in Table 4.1 "LED Status Display List". From the viewpoint of ease of use by the end user, mounting all LEDs is recommended.

For LED control, refer to Section 4.2 "Controlling the LEDs".

Table 4.1 LED Status Display List

Type	LED Name		Function
Own station status display	RUN		Indicates the operating status.
		On	Operating normally.
		Off	A hardware failure or a WDT error has occurred.
	RD		Indicates the reception status of data.
		On	Receiving data.
		Off	Data not received.
	SD		Indicates the sending status of data.
		On	Sending data.
		Off	Data not sent.
	D LINK		Indicates the status of the data link.
		On	Data link in operation (cyclic transmission in progress)
		Off	Data link not performed (disconnected)
		Blinking	Data link in operation (cyclic transmission stopped)
	ERR.		Indicates the R-IN32M3-CL error status.
		On	Error in own station
		Off	Normal operation
	L ERR.		Indicates the error status of the received data and the line. When this LED is on, you can check the port that detected the error using the L ER LED.
		On	Abnormal data received or loopback in progress
		Off	Normal data received or loopback not performed
	User LED 1, 2		Indicates a vendor-defined status.
Port 1 status display	LINK	On	Link up
		Off	Link down
	L ER	On	Abnormal data received or loopback in progress
		Off	Normal data received or loopback not performed
Port 2 status display	LINK	On	Link up
		Off	Link down
	L ER	On	Abnormal data received or loopback in progress
		Off	Normal data received or loopback not performed

4.1.1 User LED 1 and User LED 2

User LED 1 and User LED 2 are temporary names. Name the LEDs by the vendor. These LEDs indicate the vendor-defined status.

For example, the LEDs can indicate the following status. Use the examples as a reference for development.

・Online/offline status of the intelligent device station

・Testing/normal operating status when the hardware test, line test, and others are implemented on the intelligent device station

For User LED 1 and User LED 2 control, refer to Section 4.2.2 "Controlling User LED 1 and User LED 2".

4.2 Controlling the LEDs

4.2.1 LED control overview

There are two ways to control LEDs: control by hardware and control by the R-IN32M3-CL driver interface functions called from a user program.

For the LEDs controlled by hardware, R-IN32M3-CL, PHY, or the power supply check circuit controls the LEDs.

R-IN32M3-CL automatically controls the LEDs according to the status of the own station.

PHY automatically controls the LEDs when the link is up.

Control the LEDs by the power supply check circuit according to the status of the circuit mounted.

For the LEDs controlled by the R-IN32M3-CL driver interface functions, the LED on/off control functions control the LEDs. Refer to Section 6.4.7 "LED control".

Table 4.2 LED Control List

LED Name		R-IN32M3-CL Output Signal Name	Control Source	Output at Reset/Error		
				Power-on Reset	System Reset	Internal WDT[Note1]/ External WDT[Note1]/ Own Station Error[Note2]
Own station status display LEDs	PW	-	Power supply check circuit	-	-	-
	RUN	RUNLEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	Off
	RD	RDLEDL	R-IN32M3-CL	Off	-	-
	SD	SDLEDL	R-IN32M3-CL	Off	-	-
	ERR.	ERRLEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	On
	D LINK	DLINKLEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	Off
	User LED 1	USER1LEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	Off
	User LED 2	USER2LEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	Off
	L ERR.	-	Turns on according to the logical product of LERR1LEDL and LERR2LEDL[Note3] (Turns on based on L ER signal of each port)	-	-	-
Port 1 status display LEDs	LINK	-	PHY (Wire the LED so that it turns on when the PHY link is up.)	-	-	-
	L ER	LERR1LEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	Off
Port 2 status display LEDs	LINK	-	PHY (Wire the LED so that it turns on when the PHY link is up.)	-	-	-
	L ER	LERR2LEDL	R-IN32M3-CL driver interface functions, R-IN32M3-CL	Off	Off	Off

Note 1. For internal/external WDTs, refer to Section 2.3 "Interrupts".

2. This is an error that occurs for user program reasons. For details, refer to Section 6.2.6 "Own station error processing" and Section 6.4.5(2) "gerR_IN32_ForceStop".

3. For L ERR. LED control, refer to Section 4.2.3 "Controlling the L ERR. LED".

4.2.2 Controlling User LED 1 and User LED 2

R-IN32M3-CL provides two LEDs, User LED 1 and User LED 2, which can be used to define any functions.

The on/off status of User LED 1 and User LED 2 is controlled by executing the gerR_IN32_SetUSER1LED function and the gerR_IN32_SetUSER2LED function.

4.2.3 Controlling the L ERR. LED

For the L ERR. LED signal, set the external AND logic for LERR1LEDL and LERR2LEDL signals as shown in the figure below.

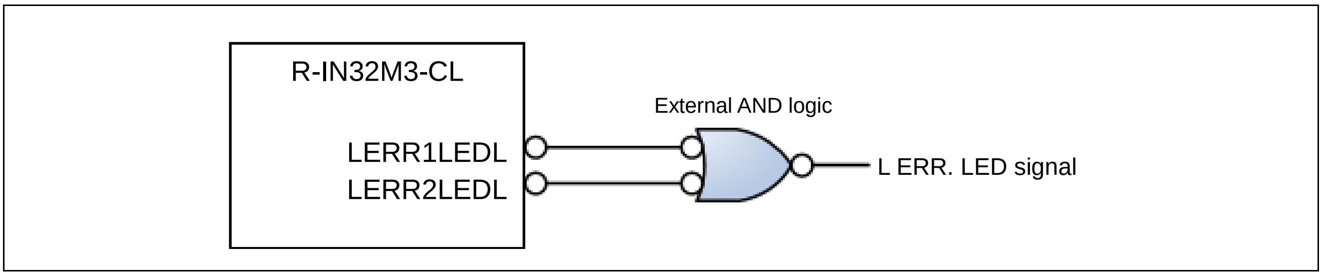


Figure 4.1 External AND Logic for Turning L ERR. On

4.3 Enabling/Disabling LEDs

LEDs in the table below can be enabled and disabled.

Determine the LED enable/disable specifications by the vendor as necessary, as shown in the example below.

Example: Disable the L ER LEDs of port 1 and port 2 in a link down state since the LED light sometimes stays ON when the link is down.

To disable the LED indicator, use the function gerR_IN32_DisableLED.

To enable the LED indicator, use the function gerR_IN32_EnableLED.

For the details of the gerR_IN32_DisableLED function and the gerR_IN32_EnableLED function, refer to Section 6.4.7 "LED control".

Table 4.3 LEDs that Can Be Enabled/Disabled

LED Name	Function
Own station status display LEDs	
RUN	Operation status display
ERR.	Error status display
D LINK	Data link status display
User LED 1	Vendor-defined status display
User LED 2	Vendor-defined status display
Port 1 status display LEDs	
L ER	Port 1 reception data error status display
Port 2 status display LEDs	
L ER	Port 2 reception data error status display

5. DATA COMMUNICATION METHOD OF CC-LINK IE FIELD NETW ORK

This chapter describes an overview of cyclic transmission, transient transmission, and MyStatus.

5.1 Cyclic Transmission Overview

The cyclic transmission is a communication method to periodically exchanges data using link devices.

The status of each link device (RY, RWw) of the master station is outputted to a slave station, and input from a slave station is stored in the link device (RX, RWr) of the master station.

By simply initiating the R-IN32M3-CL driver interface function, R-IN32M3-CL automatically reads/writes data from/to the link devices.

(Refer to Section 6.2.9 "MyStatus from master station and cyclic receive processing" and Section 6.2.11 "Cyclic send processing".)

・When data of the link devices (RX, RWr) is sent to the master station, set in R-IN32M3-CL the address in which the user program stores the send data. The R-IN32M3-CL driver automatically creates and sends a cyclic send frame.

・The R-IN32M3-CL driver automatically writes data in the received cyclic frame to the specified storage location when data of the link devices (RX, RWr) is received from the master station. The user program should read data from the storage location.

The following figure shows the flow of cyclic data.

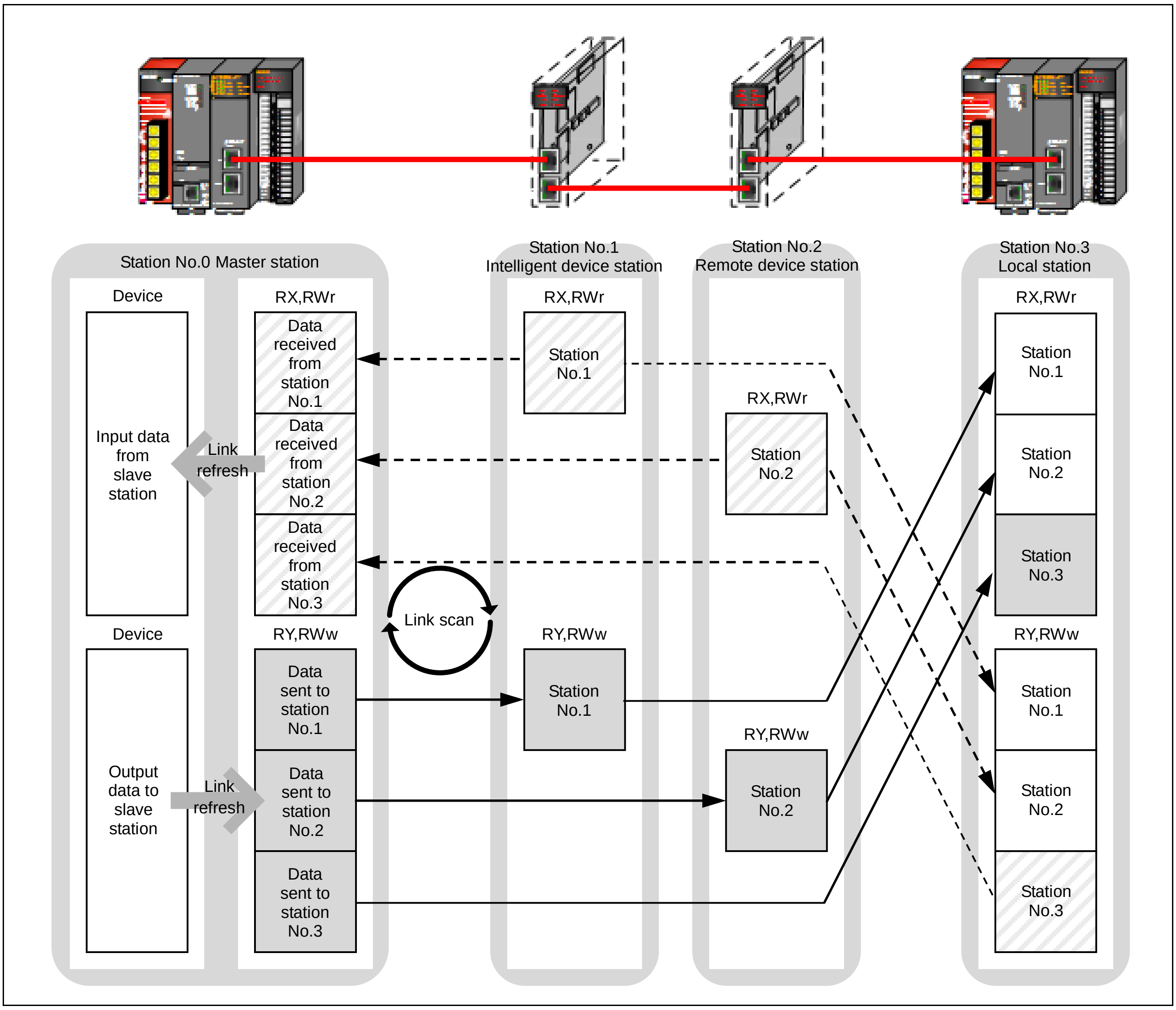


Figure 5.1 Flow of Cyclic Data

5.2 Transient Transmission Overview

Transient transmission communicates data when there is a communication request from another station or its own station. The function directly accesses the device/buffer memory of the other station and communicates the data.

Transient transmission achieves send/receive easier than cyclic transmission in the following cases:

- When reading and writing a large volume of data that exceeds the number of own/other station link device points
- When there is no send/receive area for general-purpose data (such as error history and parameter setting values) in the own/other station link device

The following shows the flow of transient data with a read instruction.

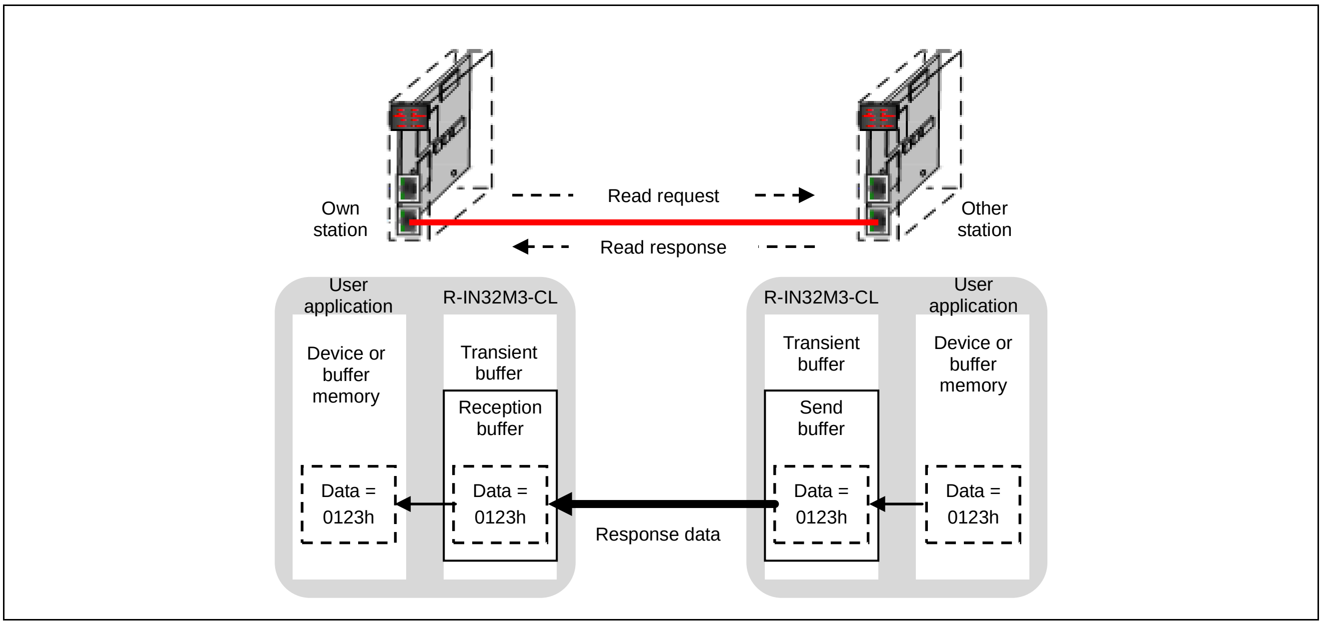


Figure 5.2 Flow of Transient Data

(1) Transient transmission client and server functions

Transient transmission includes a client function and server function.
The client function sends transient requests to nodes with a server function.
The server function sends transient responses to transient requests from nodes with a client function.

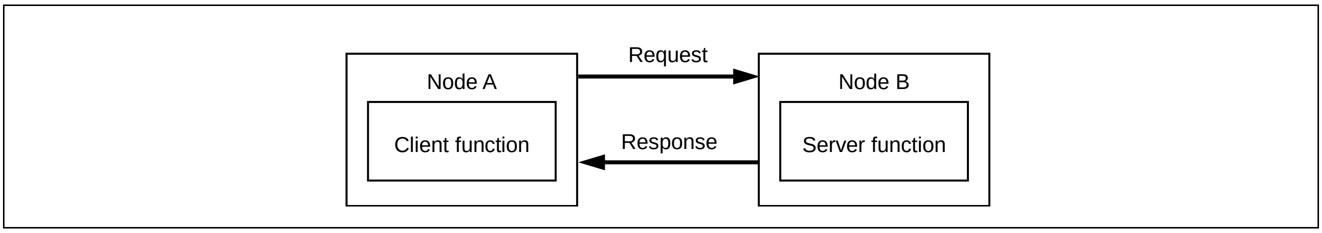


Figure 5.3 Transient Client/Server Function

(2) Transient frames of transient transmission

The following table lists the frames of transient transmission supported by the developed device, and indicates whether the send/receive processing for each frame needs to be implemented.

Table 5.1 Transient Frame List and Need for Implementation

No.	Frame Name[Note1]	Frame Type (FType)		Data Type (DataType)		Data Sub-Type		Implementation
1	CC-Link IE Field specific transient transmission	0x22	Transient1	0x07	CC-Link IE Field specific transient transmission	0x0002	System specific	Required
2	SLMP	0x22	Transient1	0x05	Network common	0x0002	SLMP	Optional
3	CC-Link compatible transient transmission	0x25	Transient2	0x04	CC-Link compatible transient transmission	-	-	Optional
4	TransientAck	0x23	TransientAck	Note2	Note2	Note2,3	Note2,3	Required

Note 1. In this manual, each frame is described using the above names.

2. TransientAck sends an acknowledgement response using the data type and data sub-type of the received frame.

3. For the TransientAck response to CC-Link compatible transient transmission, the data sub-type is set to the fixed value of 0x0000.

1. The frame for the CC-Link IE Field specific transient transmission is used by the master station to collect slave station information and manage the network.
2. The SLMP frame is used by extension functions (CC-Link IE Field Network diagnostics, parameter processing/command execution of slave stations, etc.) that use the engineering tool.
3. The CC-Link compatible transient transmission frame is mainly used in communication between vender products. The frames are compatible with CC-Link transient frames.
4. TransientAck is used to issue verification responses to the send source when Transient1 and Transient2 frames are received.

(3) Transient frames of transient transmission

The transient transmission commands that require implementation of the client and server functions differ according to the node type.

The following table indicates whether or not the implementation is required for each transient transmission command described in this manual.

Table 5.2 Necessity of Implementing Client and Server Functions of Transient Transmission Command

Frame Name	Command Type	Intelligent Device Station		[For Reference] Remote Device Station[Note1]	Remarks
		Client Function (Request)	Server Function (Response)	Server Function (Response)	
CC-Link IE Field specific transient transmission	Node information distribution	×	△[Note2]	×	-
	Statistical information acquisition	×	△	△	-
	Detailed node information acquisition	×	◎	◎	-
	Option information acquisition	×	○	○	Note3
CC-Link compatible transient transmission	Memory access information acquisition	△	△	△	Required when access codes are used
	RUN	△	△	△	-
	STOP	△	△	△	-
	Memory read	△	△	△	Equivalent to the master/local module dedicated instruction RIRD
	Memory write	△	△	△	Equivalent to the master/local module dedicated instruction RIWT
SLMP	Selected station information acquisition	×	○	○	Required for the CC-Link IE Field Network diagnostics[Note4]
	Communication test	×	○	○	
	Cable test	×	○	○	
	Memory read	△	△	△	-
	Memory write	△	△	△	-

Remark. ◎: Required, ○: Recommended, △: Optional, × : Not required

Note 1. The remote device station does not require the client function of the commands above.

2. A TransientAck and response are not required. Only processing for receiving the distributed MAC address data of other stations is required.

3. Option information acquisition is a command by which the master station confirms the presence or non-presence of slave station options. Option information is information indicating the support of extension functions of CC-Link IE Field Network, such as SLMP frame send/receive and CC-Link IE Field Network diagnostics.

4. Refer to Section 2.6 "CC-Link IE Field Network Diagnostics".

5.2.1 Transient1 request reception procedure

The following shows an image of the processing procedure in which the server sends Transient1 response frame in response to Transient1 request frame from the client. The following is an example of Statistical information acquisition, Detailed node information acquisition, SLMP memory read, and SLMP memory write.

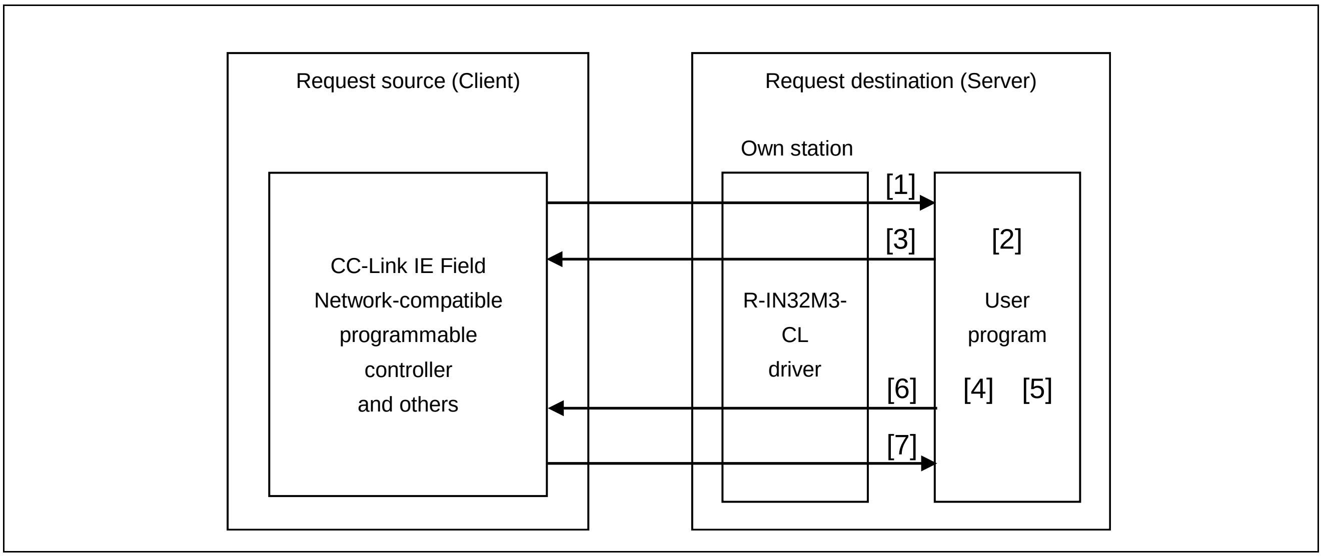

Request source (Client)
Request destination (Server)
Own station
CC-Link IE Field Network-compatible programmable controller and others
R-IN32M3-CL driver
User program
[1]
[2]
[3]
[4]
[5]
[6]
[7]

Figure 5.4 Transient1 Request Reception Procedure

Table 5.3 Transient1 Request Reception Procedure

No.	Processing	Reference
[1]	Receives Transient1 request frame.	Section 6.2.15
[2]	Creates TransientAck frame.	Section 6.2.34
[3]	Sends TransientAck frame.	Section 6.2.17
[4]	Analyzes the command of Transient1 request frame.	Section 6.2.18
[5]	Creates Transient1 response frame in accordance with the command.	CC-Link IE Field specific: Section 6.2.24 and 6.2.26 SLMP: Section 6.2.39 and 6.2.40
[6]	Sends Transient1 response frame.	Section 6.2.17
[7]	Receives TransientAck frame.	Section 6.2.15

5.2.2 Transient1 request sending procedure

The following shows an image of the processing procedure in which the client sends Transient1 request frame and receives Transient1 response frame from the server. The following is an example of SLMP memory read.

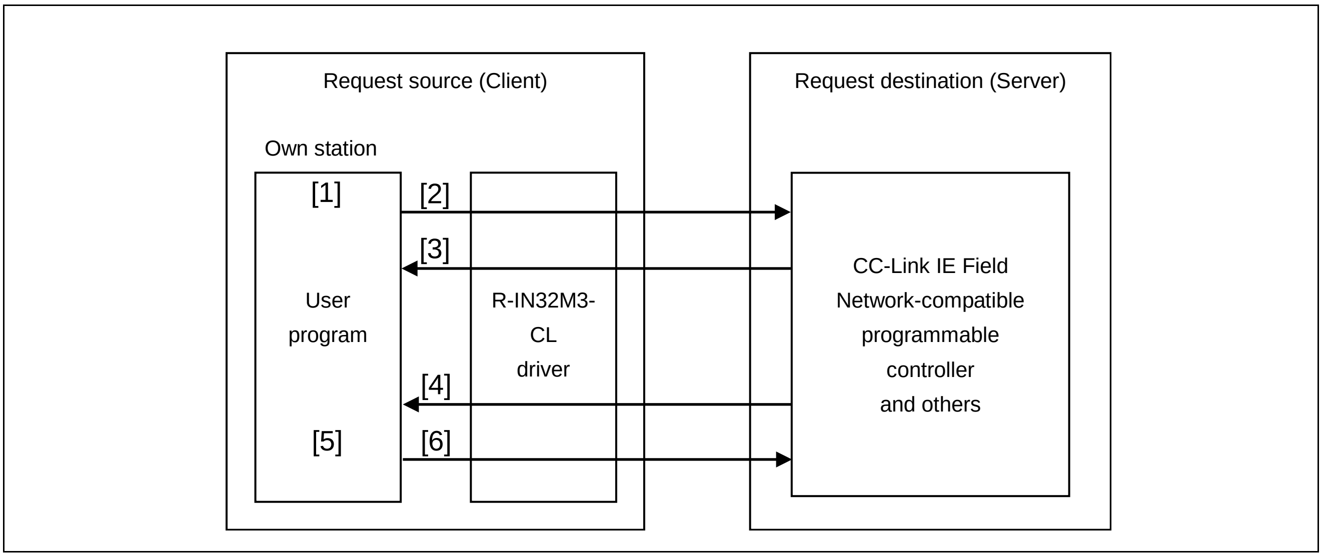

Request source (Client)
Own station
[1]
User program
[5]
[2]
[3]
[4]
[6]
R-IN32M3-CL driver
Request destination (Server)
CC-Link IE Field Network-compatible programmable controller and others

Figure 5.5 Transient1 Request Sending Procedure

Table 5.4 Transient1 Request Sending Procedure

No.	Processing	Reference
[1]	Creates Transient1 request frame.	Section 6.2.41
[2]	Sends TransientAck frame.	Section 6.2.17
[3]	Receives TransientAck frame.	Section 6.2.15
[4]	Receives Transient1 response frame.	Section 6.2.15
[5]	Creates TransientAck frame.	Section 6.2.34
[6]	Sends TransientAck frame.	Section 6.2.17

5.2.3 Transient2 request reception procedure

The following shows an image of the processing procedure in which the server sends Transient2 response frame in response to Transient2 request frame from the client.

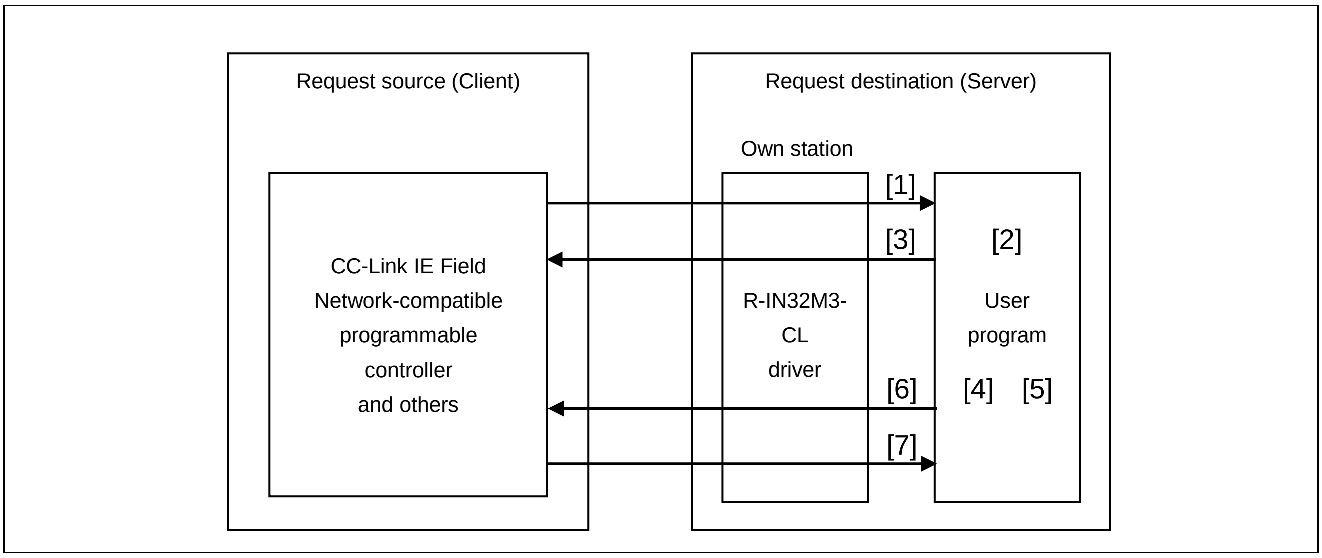

Request source (Client)
CC-Link IE Field Network-compatible programmable controller and others
Request destination (Server)
Own station
R-IN32M3-CL driver
User program
[1]
[2]
[3]
[4]
[5]
[6]
[7]

Figure 5.6 Transient2 Request Reception Procedure

Table 5.5 Transient2 Request Reception Procedure

No.	Processing	Reference
[1]	Receives Transient2 request frame.	Section 6.2.15
[2]	Creates TransientAck frame.	Section 6.2.34
[3]	Sends TransientAck frame.	Section 6.2.17
[4]	Analyzes the command of Transient2 request frame.	Section 6.2.31
[5]	Creates Transient2 response frame in accordance with the command.	Section 6.2.35
[6]	Sends Transient2 response frame.	Section 6.2.17
[7]	Receives TransientAck frame.	Section 6.2.15

5.2.4 Transient2 request sending procedure

The following shows an image of the processing procedure in which the client sends Transient2 request frame and receives Transient2 response frame from the server.

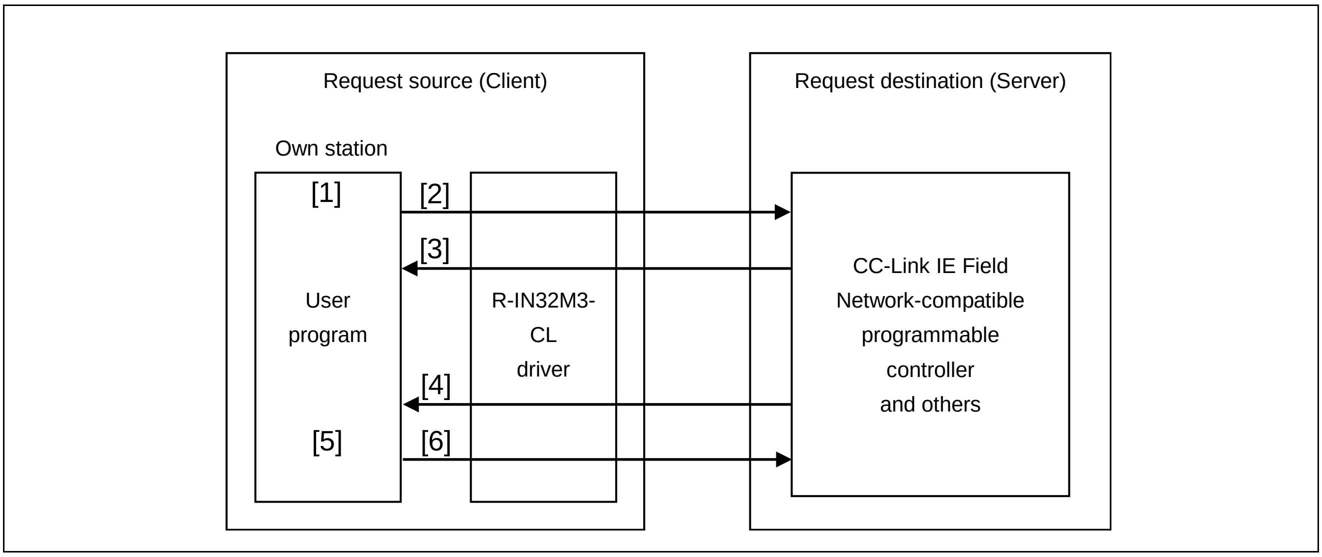

Request source (Client)
Own station
[1]
User program
[5]
[2]
[3]
[4]
[6]
R-IN32M3-CL driver
Request destination (Server)
CC-Link IE Field Network-compatible programmable controller and others

Figure 5.7 Transient2 Request Sending Procedure

Table 5.6 Transient2 Request Sending Procedure

No.	Processing	Reference
[1]	Creates Transient2 request frame.	Section 6.2.16
[2]	Sends TransientAck frame.	Section 6.2.17
[3]	Receives TransientAck frame.	Section 6.2.15
[4]	Receives Transient2 response frame.	Section 6.2.15
[5]	Creates TransientAck frame.	Section 6.2.34
[6]	Sends TransientAck frame	Section 6.2.17

5.3 Transient Transmission Frame Format Overview

The frames of the CC-Link IE Field Network are IEEE 802.3 Ethernet frame compatible. The Ethernet frame size is 64 to 1518 bytes starting from the MAC header to FCS.

This section describes the following transient frames that require vendors to set in user programs.

- Transient1 frame (CC-Link IE Field specific transient transmission and SLMP)
- Transient2 frame (CC-Link compatible transient transmission)
- TransientAck frame

5.3.1 Transient frame common format

The transient frame common format is a format used in common by Transient frames.

Table 5.7 Overview of Transient Frame Common Format

No.	Item	Size (Bytes)	Remarks
1	MAC header	14	-
	CC-Link IE header	14	
2	Transient data	1482	-
3	DCS	4	Data Check Sequence^Note
4	FCS	4	Frame Check Sequence^Note

Note. Automatically calculated and added by R-IN32M3-CL.

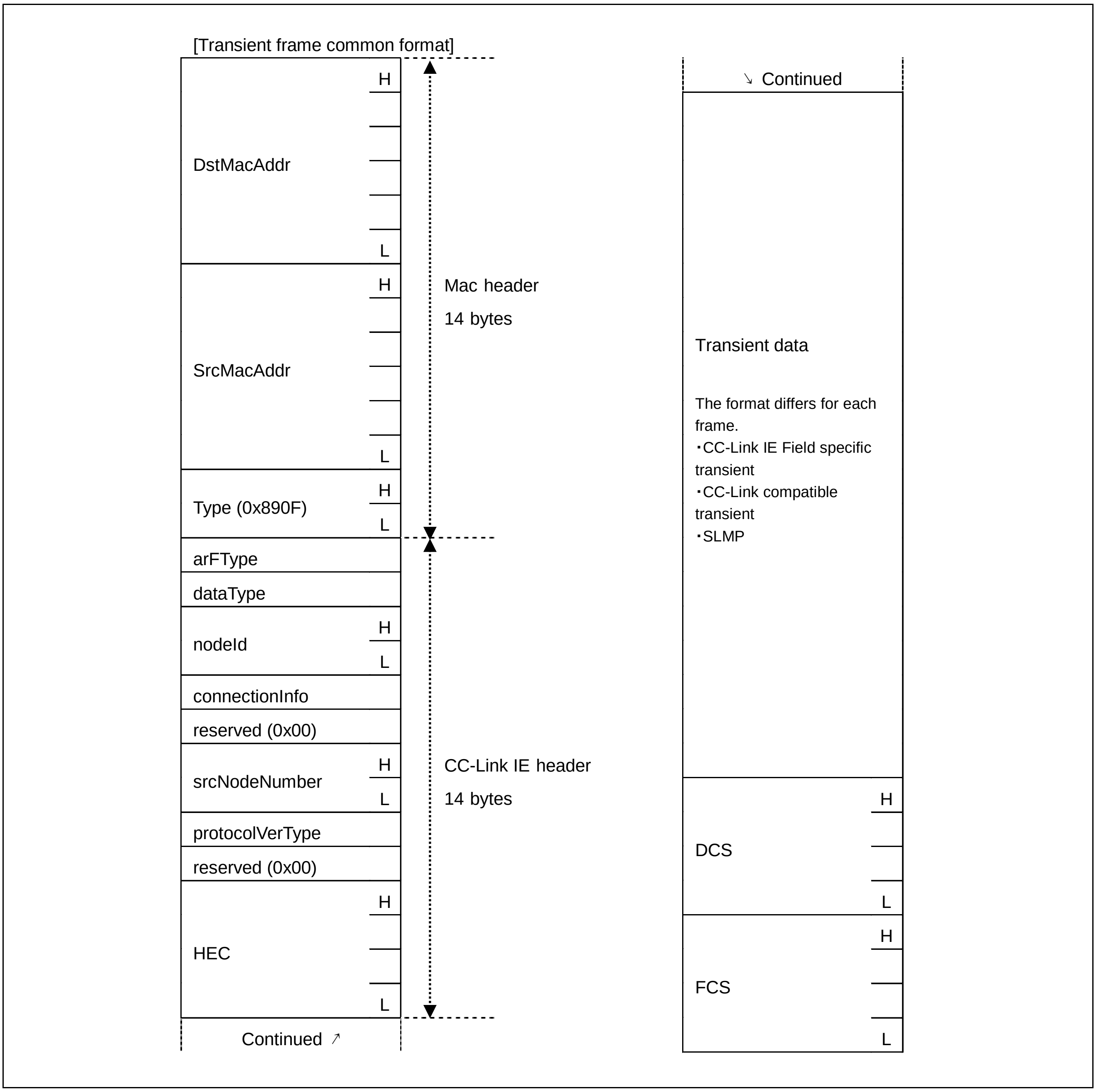


Figure 5.8 Transient Frame Common Format

Table 5.8 MAC Header Items

Item	Description	Value	Remarks
Dst/SrcMacAddr (first octet)	MAC address of send destination/source	Value managed by IEEE	0x01 when the MAC address is 01-23-45-67-89-AB. Set the I/G bit to 0b. When the I/G bit is set to 1b, the address becomes a multicast address and communication cannot be performed normally with the master station.
Dst/SrcMacAddr (second octet)	MAC address of send destination/source		0x23 when the MAC address is 01-23-45-67-89-AB.
Dst/SrcMacAddr (third octet)	MAC address of send destination/source		0x45 when the MAC address is 01-23-45-67-89-AB.
Dst/SrcMacAddr (fourth octet)	MAC address of send destination/source	Value managed by vendor	0x67 when the MAC address is 01-23-45-67-89-AB.
Dst/SrcMacAddr (fifth octet)	MAC address of send destination/source		0x89 when the MAC address is 01-23-45-67-89-AB.
Dst/SrcMacAddr (sixth octet)	MAC address of send destination/source		0xAB when the MAC address is 01-23-45-67-89-AB.
Type	Type	Fixed to 0x890F	Indicates that the frame is a CC-Link IE Field Network transmission frame.

Note. Set all items in this table using big endian.

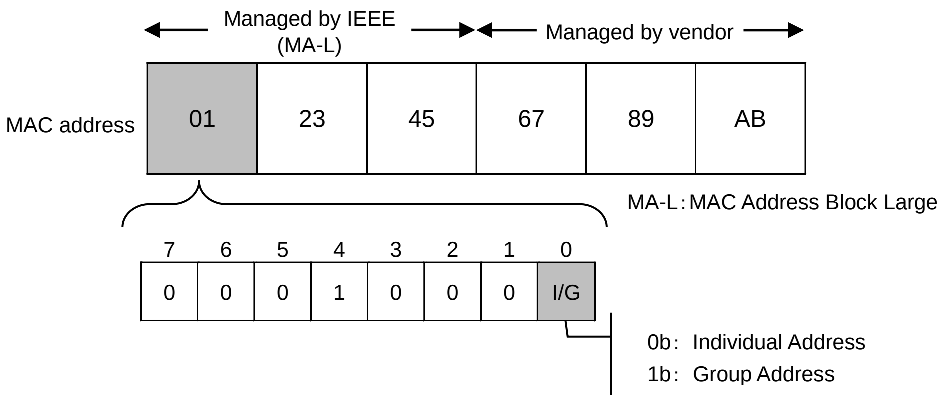


Figure 5.9 MAC Address I/G Bit

Table 5.9 CC-Link IE Header Items

Item	Description		Value	Remarks
arFType	Frame type		Refer to Table 5.1	-
dataType	Data type			-
nodeId	Node identifier		0x0000 to 0x00F0 (0 to 240)	Management information of each slave station connected to the master station (The number differs from a station number.) Acquired by function gusR_IN32_GetNodeID.Note1 Set using big endian.
connectionInfo	Transient identification information		0x01 to 0xFF (1 to 255)	Information for identifying the transient frame sent during one token hold. Acquired by function gerR_IN32_GetSendTransientBuffer.Note2
reserved	Reserved		Fixed to 0x00	-
srcNodeNumber	Own node number		0x0001 to 0x0078 (1 to 120)	Set using big endian.
protocolVerType	Bit7-4	Protocol version	Fixed to 0x0	-
	Bit3-0	Protocol type	Fixed to 0x1	0x01: CC-Link IE Field Network
HEC	Header Error Control		Automatically calculated by R-IN32M3-CL.	-

Note 1. Refer to Section 6.4.11(2) "gusR_IN32_GetNodeID".
2. Refer to Section 6.4.11(5) "gerR_IN32_GetSendTransientBuffer".

5.3.2 CC-Link IE Field specific transient frame format

The following table provides an overview of CC-Link IE Field specific transient frame format.

Table 5.10 Overview of CC-Link IE Field Specific Transient Frame Format

No.	Item		Size (Bytes)	Remarks
1	MAC header		14	Refer to Section 5.3.1
	CC-Link IE header		14	
2	Transient1 header		16	-
3	Transient1 data areaNote2	Extension header	20	-
		Data	0 to 1446	
4	DCS		4	Data Check SequenceNote1
5	FCS		4	Frame Check SequenceNote1

Note 1. Automatically calculated and added by R-IN32M3-CL.
2. When Transient1 data area is used as "CC-Link IE Field specific transient".
Refer to the Section 5.3.5 "SLMP frame format" when Transient1 data area is used as "SLMP".

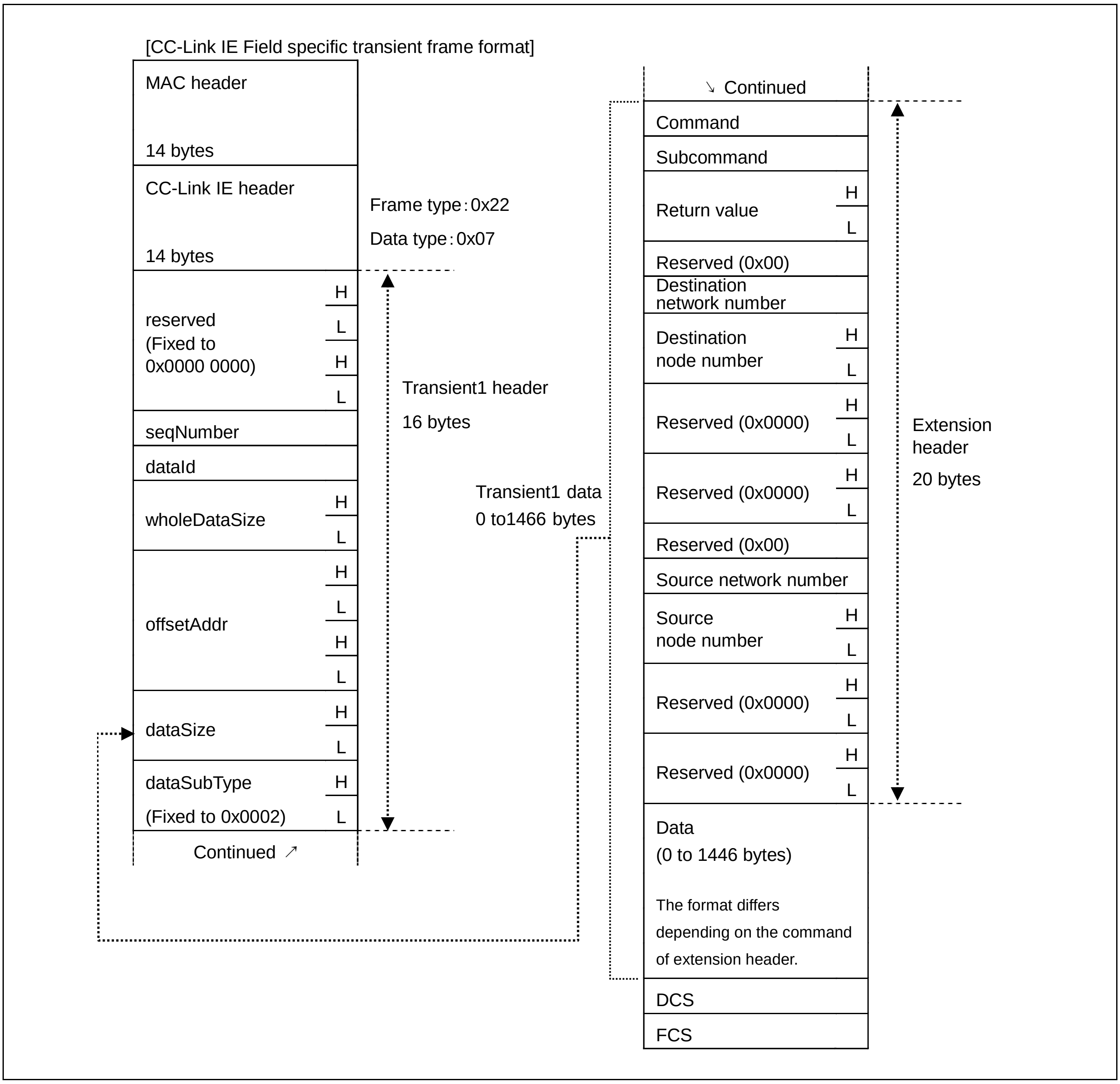


Figure 5.10 Overview of CC-Link IE Field Specific Transient Frame Format

(1) MAC header, CC-Link IE header

Refer to Section 5.3.1 "Transient frame common format".

(2) Transient1 header

The Transient1 header is added to Transient1 frame (CC-Link IE Field specific transient and SLMP).

Table 5.11 Transient1 Header Items

Item	Description		Value	Remarks
reserved	Reserved		Fixed to 0x00000000	-
seqNumber	Bit7	Final frame identification	0b: Divided frame 1b: Final divided frame	A number assigned when transient data is divided
	Bit6-0	Transient1 frame sequential number	0x00 to 0x7F	
dataId	Transient data identification number		0x00 to 0xFF	Set the same identification number for divided frames.
wholeDataSize	Size of entire Transient1 data area		0x0000 to 0x0800 (0 to 2048)	Entire transient data size before divided
offsetAddr	Offset address from the start of entire transient data		0x0000 0000 to 0x7FFF FFFF	When not divided: Fixed to 0 First frame when divided: Fixed to 0 For the second frame and later, the storage location within the entire transient data is indicated using an offset address from the start of the data. Note
dataSize	Size of transient data in the frame		0x0000 to 0x05BA (0 to 1466)	Transient data size after divided Note
dataSubType	Data sub-type		0x0002: System specific 0x0002: SLMP	Note

Note. Set using big endian.

The following example explains the relationship between the sequential number and identification number of Transient1 header.

Transient data No.1: Not divided
Transient data No.2: Divided into three
Transient data No.3: Divided into two

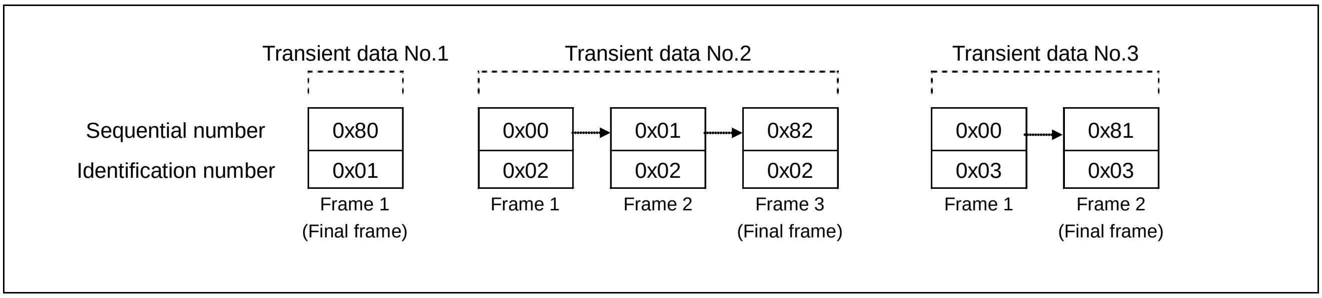


Figure 5.11 Transient1 Header: Relationship Between Sequential Number and Identification Number of Transient Data

(3) Transient1 data area

For CC-Link IE Field specific transient transmission, a frame consists of the extension header and data.

Table 5.12 Extension Header Items

Item	Description	Value		Remarks
Command	Command	Refer to Table 5.13		-
Subcommand	Subcommand			
Return value	Return value in response to request	During request	0x0000 (Fixed)	Note
		During response	0x0000 (Normal) 0x0001 to 0xFFFF (Abnormal)	
Reserved	Reserved	Fixed to 0		-
Destination network number	Destination network number	0: Broadcast 1 to 239: Destination network		-
Destination node number	Destination node number	1 to 120: Slave station 0x007D: Master station 0xFFFF: Broadcast		Note
Reserved	Reserved	Fixed to 0		-
Reserved	Reserved	Fixed to 0		-
Reserved	Reserved	Fixed to 0		-
Source network number	Network number of send source	1 to 239		-
Source node number	Node number of send source	1 to 120		Note
Reserved	Reserved	Fixed to 0		-
Reserved	Reserved	Fixed to 0		-

Note. Set using big endian.

Table 5.13 CC-Link IE Field Specific Transient Transmission Command List

Command	Subcommand	Command Type	Send Direction	Remarks
0x01	0x00	Node information distribution request	Master station → Slave station	Response not required
0x03	0x00	Statistical information acquisition request	Master station → Slave station	-
0x03	0x80	Statistical information acquisition response	Master station ← Slave station	
0x04	0x00	Detailed node information acquisition request	Master station → Slave station	
0x04	0x80	Detailed node information acquisition response	Master station ← Slave station	
0x0A	0x00	Option information acquisition request	Master station → Slave station	
0x0A	0x80	Option information acquisition response Note	Master station ← Slave station	

Note. Option information acquisition responses are created using interface functions. The frame format does not need to be considered.

(a) Node information distribution

Node information distribution frame distributes the destination MAC address, which is required when a slave station uses the client function, from the master station to the slave station.

TransientAck or Transient1 response frame (response to Node information distribution request) does not need to be sent.

If the number of pieces of distributed node information is 60 or more, the frame size exceeds 1518 bytes, which is the maximum size of the Ethernet frame. Therefore, the frame is divided into two frames. In this case, the Transient1 reception data needs to be reconstructed using the user program.

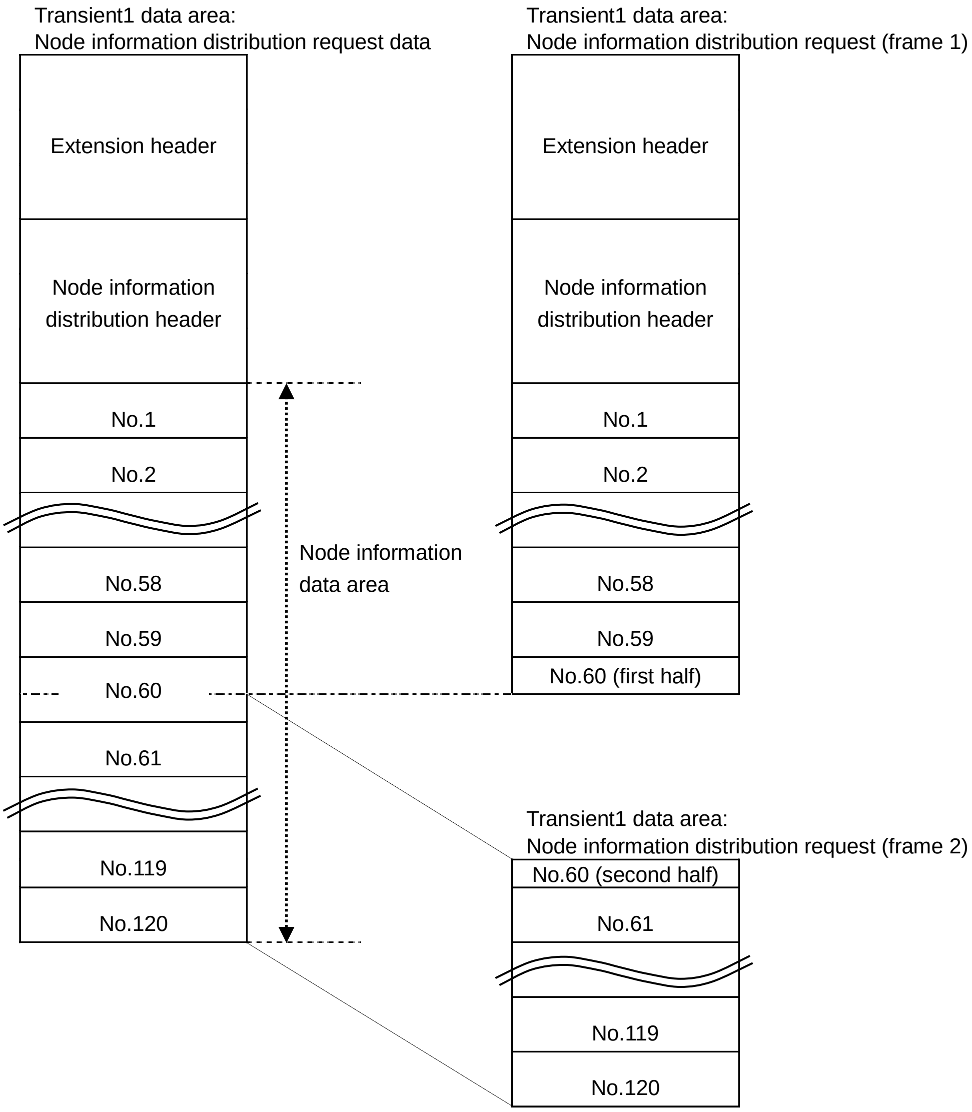

Transient1 data area:
Node information distribution request data
Extension header
Node information distribution header
No.1
No.2
No.58
No.59
No.60
No.61
No.119
No.120
Node information data area
Transient1 data area:
Node information distribution request (frame 1)
Extension header
Node information distribution header
No.1
No.2
No.58
No.59
No.60 (first half)
Transient1 data area:
Node information distribution request (frame 2)
No.60 (second half)
No.61
No.119
No.120

Figure 5.12 Transient1 Data Area: Divided Frames at Node Information Distribution Request

For the frame format, refer to Figure 5.13, Figure 5.14, and Figure 5.15 in accordance with the table below.

Table 5.14 Frame Format for Node Information Distribution

Number of Distributions	Reference
Less than 60	Figure 5.13 "Transient1 Data Area: Node Information Distribution Request"
60 or more	Figure 5.14 "Transient1 Data Area: Node Information Distribution Request (Frame 1)" Figure 5.15 "Transient1 Data Area: Node Information Distribution Request (Frame 2)"

For details on reconstructing the Transient1 reception data, refer to Section 6.2.19 "Transient1 receive data reconstruction start processing" and Section 6.2.20 "Transient1 receive data reconstruction processing".

The following shows the format of Node information distribution request frame involving less than 60 distributions.

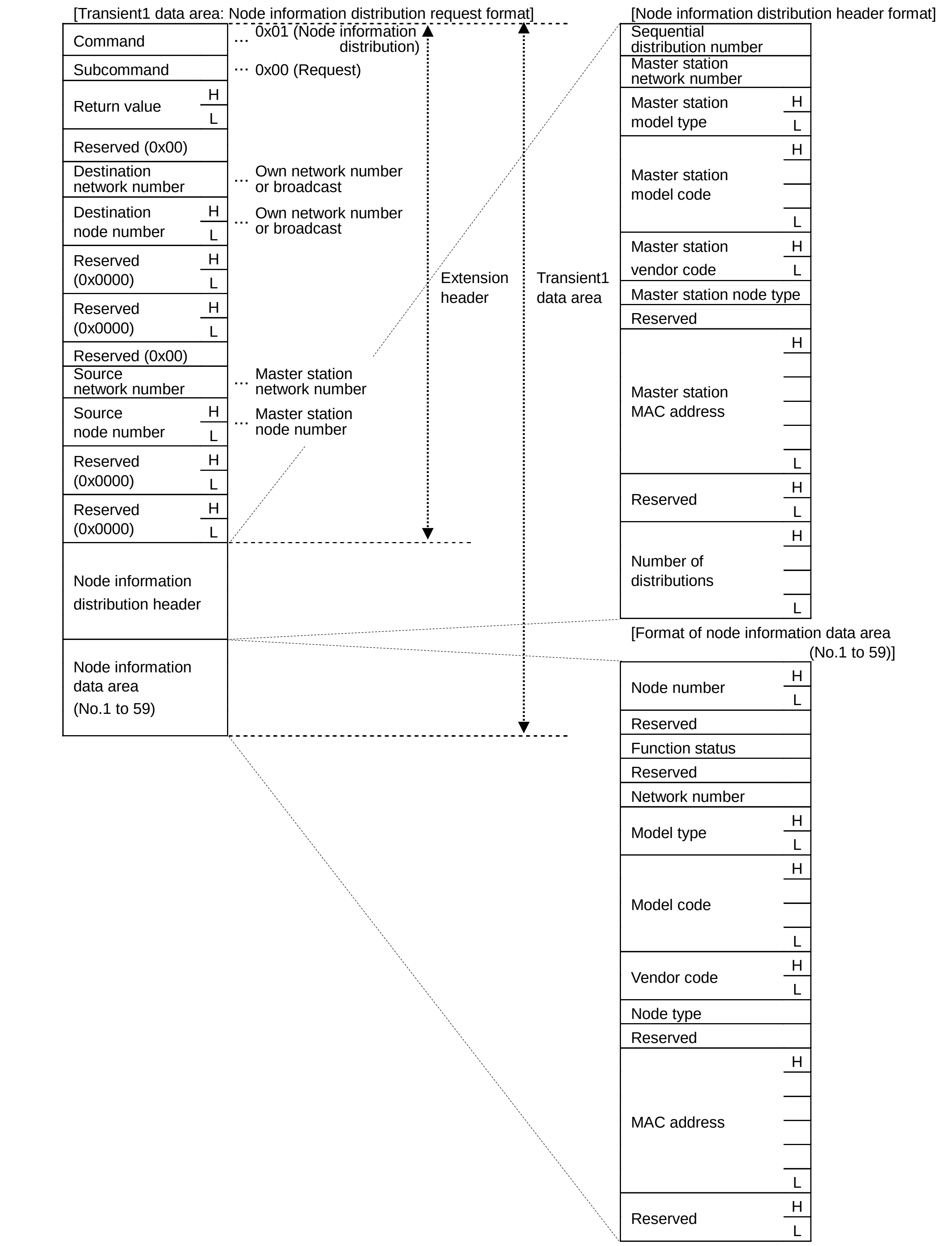

[Transient1 data area: Node information distribution request format]
Command
… 0x01 (Node information distribution)
Subcommand
… 0x00 (Request)
Return value
H
L
Reserved (0x00)
Destination network number
… Own network number or broadcast
Destination node number
… Own network number or broadcast
Reserved (0x0000)
Reserved (0x0000)
Reserved (0x00)
Source network number
… Master station network number
Source node number
… Master station node number
Reserved (0x0000)
Reserved (0x0000)
Node information distribution header
Node information data area (No.1 to 59)
Extension header
Transient1 data area
[Node information distribution header format]
Sequential distribution number
Master station network number
Master station model type
Master station model code
Master station vendor code
Master station node type
Reserved
Master station MAC address
Reserved
Number of distributions
[Format of node information data area (No.1 to 59)]
Node number
Reserved
Function status
Reserved
Network number
Model type
Model code
Vendor code
Node type
Reserved
MAC address
Reserved

Figure 5.13 Transient1 Data Area: Node Information Distribution Request

The following shows the format of Node information distribution request frame (frame 1) involving 60 or more distributions.

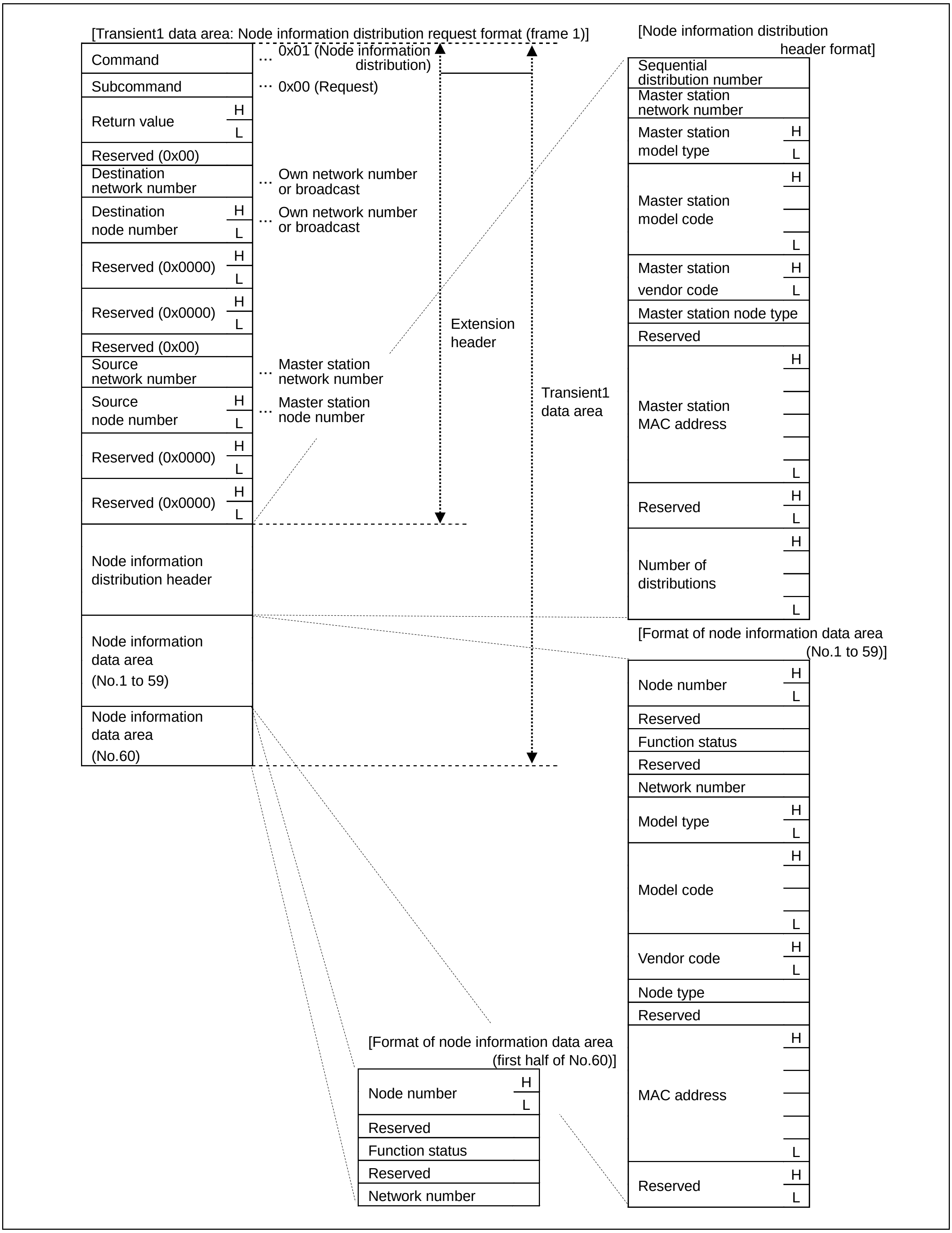

[Transient1 data area: Node information distribution request format (frame 1)]
Command
… 0x01 (Node information distribution)
Subcommand
… 0x00 (Request)
Return value
H
L
Reserved (0x00)
Destination network number
… Own network number or broadcast
Destination node number
… Own network number or broadcast
Reserved (0x0000)
Reserved (0x0000)
Reserved (0x00)
Source network number
… Master station network number
Source node number
… Master station node number
Reserved (0x0000)
Reserved (0x0000)
Extension header
Transient1 data area
Node information distribution header
Node information data area (No.1 to 59)
Node information data area (No.60)
[Node information distribution header format]
Sequential distribution number
Master station network number
Master station model type
Master station model code
Master station vendor code
Master station node type
Reserved
Master station MAC address
Reserved
Number of distributions
[Format of node information data area (No.1 to 59)]
Node number
Reserved
Function status
Reserved
Network number
Model type
Model code
Vendor code
Node type
Reserved
MAC address
Reserved
[Format of node information data area (first half of No.60)]
Node number
Reserved
Function status
Reserved
Network number

Figure 5.14 Transient1 Data Area: Node Information Distribution Request (Frame 1)

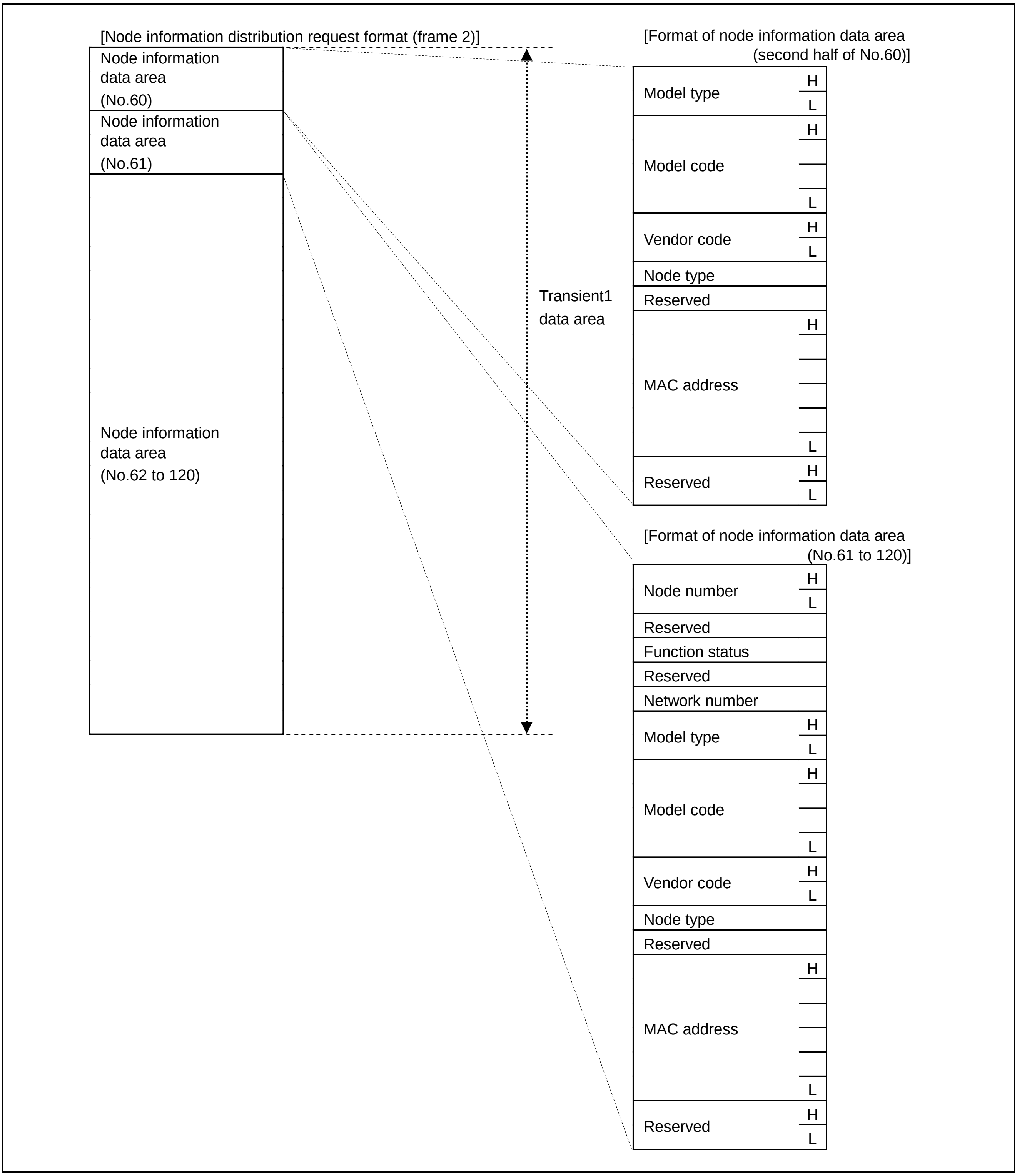

[Node information distribution request format (frame 2)]
Node information data area (No.60)
Node information data area (No.61)
Node information data area (No.62 to 120)
Transient1 data area
[Format of node information data area (second half of No.60)]
Model type
H
L
Model code
H
L
Vendor code
H
L
Node type
Reserved
MAC address
H
L
Reserved
H
L
[Format of node information data area (No.61 to 120)]
Node number
H
L
Reserved
Function status
Reserved
Network number
Model type
H
L
Model code
H
L
Vendor code
H
L
Node type
Reserved
MAC address
H
L
Reserved
H
L

Figure 5.15 Transient1 Data Area: Node Information Distribution Request (Frame 2)

Table 5.15 Node Information Distribution Header Items

Item	Description	Value	Remarks
Sequential distribution number	Sequential distribution number	1 to 7	When the sequential distribution numbers are the same, the node information is the same. Discard it.
Master station network number	Network number of master station	1 to 239	-
Master station model type	Model type of master station	0x0001 to 0xFFFF	Model type managed by CC-Link Partner Association
Master station model code	Model code of master station	0x00000000 to 0xFFFFFFFF	Model code of network that is unique within the same vendor code
Master station vendor code	Vendor code of master station	0x0000 to 0xFFFF	Vendor code managed by CC-Link Partner Association
Master station node type	Node type of master station	Fixed value: 0x30	-
Reserved	Reserved	Fixed value: 0x00	-
Master station MAC address	MAC address of master station	6-byte MAC address	-
Reserved	Reserved	Fixed value: 0x0000	-
Number of distributions	Number of distributions of node information	1 to 120	-

Table 5.16 Node Information Data Area Items

Item	Description	Value	Remarks
Node number	Node number of slave station	1 to 120	-
Reserved	Reserved	Fixed value: 0x00	-
Function status	Slave station transient reception function status	Provided: 0x01 Not provided: 0x00	-
Reserved	Reserved	Fixed value: 0x00	-
Network number	Network number of slave station	1 to 239	-
Model type	Model type of slave station	0x0001 to 0xFFFF	Model type managed by CC-Link Partner Association
Model code	Model code of slave station	0x00000000 to 0xFFFFFFFF	Model code of network that is unique within the same vendor code
Vendor code	Vendor code of slave station	0x0000 to 0xFFFF	Vendor code managed by CC-Link Partner Association
Node type	Node type of slave station	Refer to Table 5.17	-
Reserved	Reserved	Fixed value: 0x00	-
MAC address	MAC address of slave station	6-byte MAC address	-
Reserved	Reserved	Fixed value: 0x0000	-

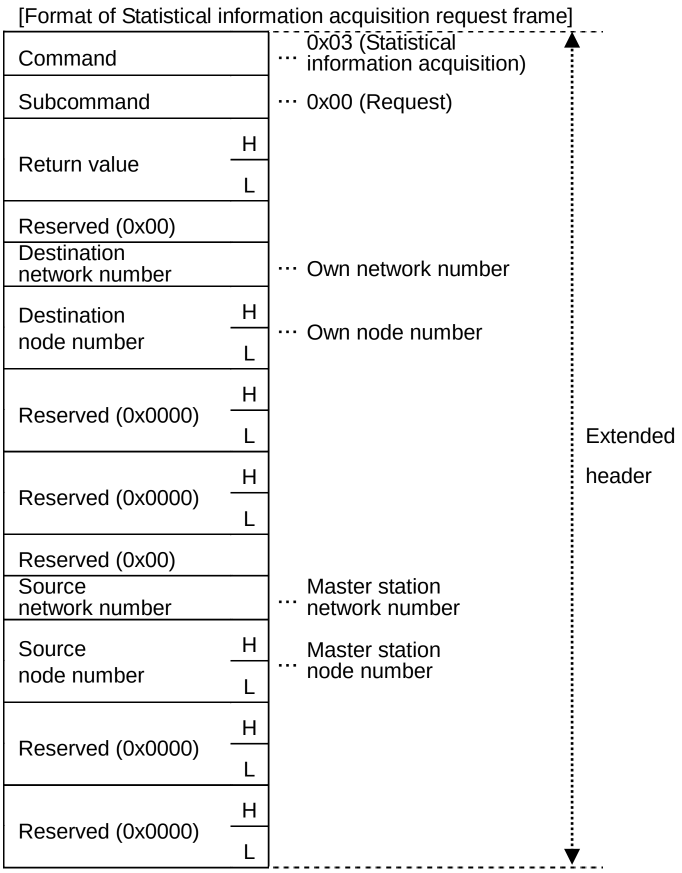
Table 5.17 Node Type List

Node Type	Description	Remarks
0x30	Master station	-
0x31	Reserved	-
0x32	Local station	-
0x33	Intelligent device station	-
0x34	Remote device station	-
0x35	Remote I/O station	-

(b) Statistical information acquisition

Statistical information acquisition is used for the master station to collect error information related to port 1 and port 2 of a slave station.

The following shows the format of Statistical information acquisition frame.



[Format of Statistical information acquisition request frame]

Command		··· 0x03 (Statistical information acquisition)
Subcommand		··· 0x00 (Request)
Return value	H L	
Reserved (0x00)		
Destination network number		··· Own network number
Destination node number	H L	··· Own node number
Reserved (0x0000)	H L	
Reserved (0x0000)	H L	
Reserved (0x00)		
Source network number		··· Master station network number
Source node number	H L	··· Master station node number
Reserved (0x0000)	H L	
Reserved (0x0000)	H L	

Figure 5.16 Transient1 Data Area: Statistical Information Acquisition Request

For details on each item in the figure above, refer to Table 5.12 "Extension Header Items".

The following figure shows the format of Statistical information acquisition response frame.

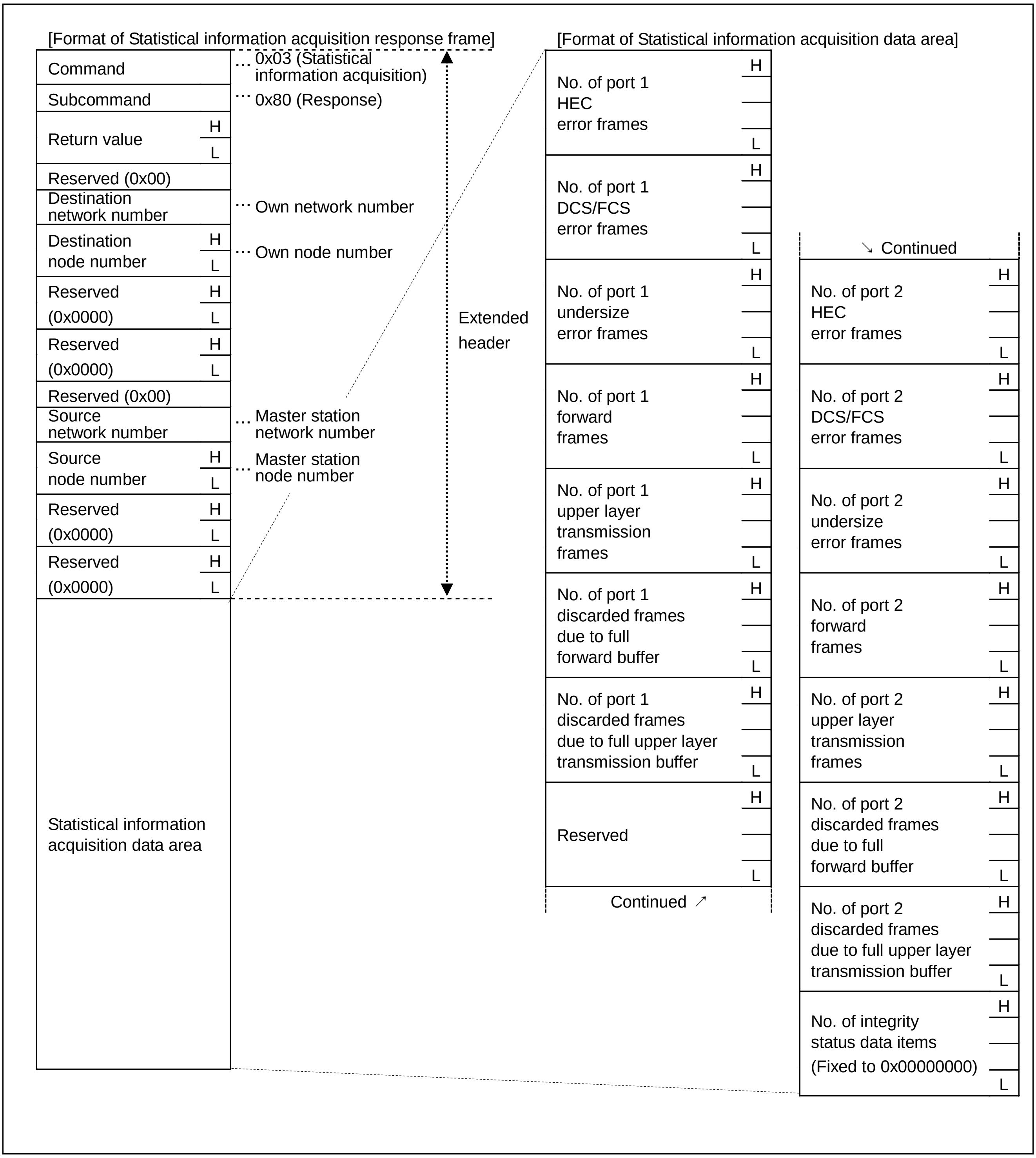

[Format of Statistical information acquisition response frame]
Command
··· 0x03 (Statistical information acquisition)
Subcommand
··· 0x80 (Response)
Return value H L
Reserved (0x00)
Destination network number
··· Own network number
Destination node number H L
··· Own node number
Reserved (0x0000) H L
Reserved (0x0000) H L
Reserved (0x00)
Source network number
··· Master station network number
Source node number H L
··· Master station node number
Reserved (0x0000) H L
Reserved (0x0000) H L
Extended header
Statistical information acquisition data area
[Format of Statistical information acquisition data area]
No. of port 1 HEC error frames H L
No. of port 1 DCS/FCS error frames H L
No. of port 1 undersize error frames H L
No. of port 1 forward frames H L
No. of port 1 upper layer transmission frames H L
No. of port 1 discarded frames due to full forward buffer H L
No. of port 1 discarded frames due to full upper layer transmission buffer H L
Reserved H L
Continued ↗
↘ Continued
No. of port 2 HEC error frames H L
No. of port 2 DCS/FCS error frames H L
No. of port 2 undersize error frames H L
No. of port 2 forward frames H L
No. of port 2 upper layer transmission frames H L
No. of port 2 discarded frames due to full forward buffer H L
No. of port 2 discarded frames due to full upper layer transmission buffer H L
No. of integrity status data items (Fixed to 0x00000000) H L

Figure 5.17 Transient1 Data Area: Statistical Information Acquisition Response

Each data area item of Statistical information acquisition response frame shown in the table below is acquired by the function gerR_IN32_GetMIB. Refer to Section 6.4.6(6) "gerR_IN32_GetMIB".

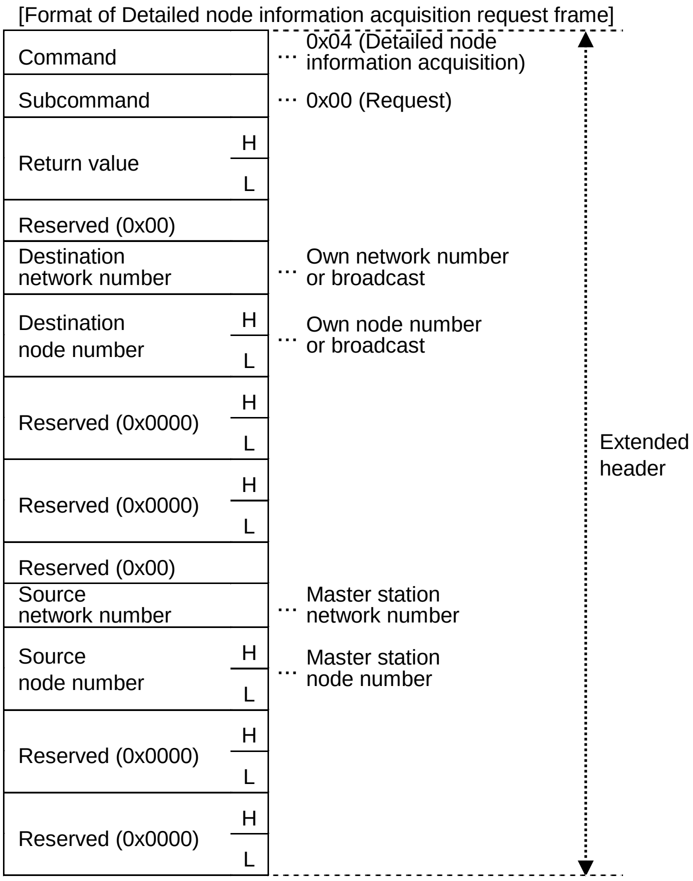
Table 5.18 Statistical Information Acquisition Response Data Items

Item	Description	Value	Remarks
No. of port 1 HEC error frames	No. of HEC error frames of port 1	0 to 4294967295	Counts the number of HEC errors in received frames.
No. of port 1 DCS/FCS error frames	No. of DCS/FCS error frames of port 1	0 to 4294967295	Counts the number of DCS/FCS errors in received frames.
No. of port 1 undersize error frames	No. of undersize error frames of port 1	0 to 4294967295	Counts the number of received error frames with a size less than 28 bytes.
No. of port 1 forwarded frames	No. of forwarded frames of port 1	0 to 4294967295	Counts the number of forwarded frames.
No. of port 1 upper layer transmission frames	No. of upper layer transmission frames of port 1	0 to 4294967295	Counts the number of frames transmitted to upper layers.
No. of port 1 discarded frames due to full forward buffer	No. of port 1 frames discarded due to full forward buffer	0 to 4294967295	Counts the number of frames discarded due to a full forward buffer.
No. of port 1 discarded frames due to full upper layer transmission buffer	No. of port 1 frames discarded due to full upper layer transmission buffer	0 to 4294967295	Counts the number of frames discarded due to a full upper layer transmission buffer.
Reserved	Reserved	Fixed value: 0x00000000	-
No. of port 2 HEC error frames	No. of HEC error frames of port 2	0 to 4294967295	Counts the number of HEC errors in received frames.
No. of port 2 DCS/FCS error frames	No. of DCS/FCS error frames of port 2	0 to 4294967295	Counts the number of DCS/FCS errors in received frames.
No. of port 2 undersize error frames	No. of undersize error frames of port 2	0 to 4294967295	Counts the number of received error frames with a size less than 28 bytes.
No. of port 2 forward frames	No. of forwarded frames of port 2	0 to 4294967295	Counts the number of forwarded frames.
No. of port 2 upper layer transmission frames	No. of upper layer transmission frames of port 2	0 to 4294967295	Counts the number of frames transmitted to upper layers.
No. of port 2 discarded frames due to full forward buffer	No. of port 2 frames discarded due to full forward buffer	0 to 4294967295	Counts the number of frames discarded due to a full forward buffer.
No. of port 2 discarded frames due to full upper layer transmission buffer	No. of port 2 frames discarded due to full upper layer transmission buffer	0 to 4294967295	Counts the number of frames discarded due to a full upper layer transmission buffer.
No. of integrity status data items	No. of integrity status data items	Fixed value: 0x00000000	-

(c) Detailed node information acquisition

Detailed node information acquisition is used by the master station to collect the detailed node information of a slave station.

The following figure shows the format of Detailed node information acquisition request frame.



[Format of Detailed node information acquisition request frame]

Field	Byte	Value
Command		… 0x04 (Detailed node information acquisition)
Subcommand		… 0x00 (Request)
Return value	H	
	L	
Reserved (0x00)		
Destination network number		… Own network number or broadcast
Destination node number	H	… Own node number or broadcast
	L	
Reserved (0x0000)	H	
	L	
Reserved (0x0000)	H	
	L	
Reserved (0x00)		
Source network number		… Master station network number
Source node number	H	… Master station node number
	L	
Reserved (0x0000)	H	
	L	
Reserved (0x0000)	H	
	L	

Figure 5.18 Transient1 Data Area: Detailed Node Information Acquisition Request

For details on each item in the figure above, refer to Table 5.12.

The following shows the format of Detailed node information acquisition response frame.

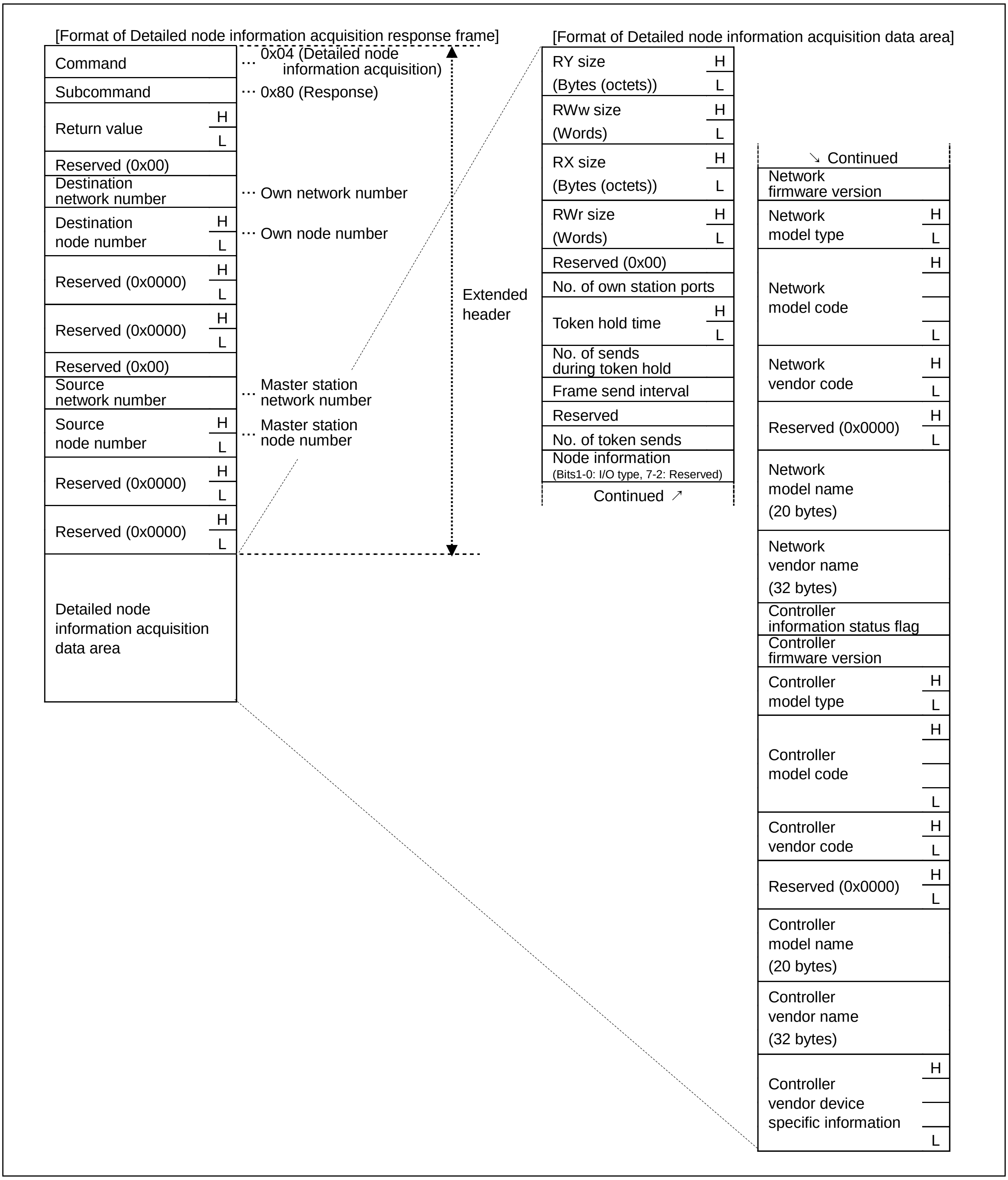


Figure 5.19 Transient1 Data Area: Detailed Node Information Acquisition Response

Each data area item of Detailed node information acquisition response frame shown in the table below is set to the value acquired by the function gerR_IN32_GetUnitInformation. Refer to Section 6.4.1(2) "gerR_IN32_Initialize".

Table 5.19 Data Area Items of Detailed Node Information Acquisition Response

Item	Description	Value	Remarks
RY size (bytes (octets))	RY size of own station	Minimum value: 0 Maximum value: 256	-
RWw size (words)	RWw size of own station	Minimum value: 0 Maximum value: 1024	-
RX size (bytes (octets))	RX size of own station	Minimum value: 0 Maximum value: 256	-
RWr size (words)	RWr size of own station	Minimum value: 0 Maximum value: 1024	-
Reserved	Reserved	Fixed value: 0x00	-
No. of own station ports	Number of ports of own station	1 to 2	-
Token hold time	Maximum value (μs) of token hold time of own station	1 to 32767	-
No. of sends during token hold	Number of frame sending other than token frame sending during token hold	1 to 255	-
Frame send interval	Frame interval after token frame reception to MyStatus frame sending	1 to 255	-
Reserved	Reserved	Fixed value: 0x00	-
No. of token sends	Number of repeated sending of token frame during token hold	1 to 255	-
Node information (I/O type)	I/O type	Mixed: 0x00 Input: 0x01 Output: 0x02 Composite: 0x03	-
Network firmware version	Network firmware version	0 to 255	-
Network model type	Network model type	0x0001 to 0xFFFF	-
Network model code	Network model code	0x00000000 to 0xFFFFFFFF	-
Network vendor code	Network vendor code	0x0000 to 0xFFFF	-
Reserved	Reserved	Fixed value: 0x0000	-
Network model name	Network model name	Model name (20 bytes)	-
Network vendor name	Network vendor name	Vendor name (32 bytes)	-
Controller information status flag	Controller information (from "Controller firmware version" to "Controller vendor device specific information") status flag	Disable: 0 Enable: 1	-
Controller firmware version	Controller firmware version	0 to 255	-
Controller model type	Controller model type	0x0001 to 0xFFFF	-
Controller model code	Controller model code	0x00000000 to 0xFFFFFFFF	-
Controller vendor code	Controller vendor code	0x0000 to 0xFFFF	-
Reserved	Reserved	Fixed value: 0x0000	-
Controller model name	Controller model name	Model name (20 bytes)	-
Controller vendor name	Controller vendor name	Vendor name (32 bytes)	-
Controller vendor device specific information	Controller vendor device specific information	0x00000000 to 0xFFFFFFFF	-

5.3.3 TransientAck frame format

The following table provides an overview of the TransientAck frame format.

Table 5.20 TransientAck Frame Format Overview

No.	Item	Size (Bytes)	Remarks
1	MAC header	14	Refer to Section 5.3.1
	CC-Link IE header	14	
2	TransientAck data area	28	Fixed value: 0x00000001[Note]
3	DCS	4	Data Check Sequence[Note]
4	FCS	4	Frame Check Sequence[Note]

Note. Automatically calculated and added by R-IN32M3-CL.

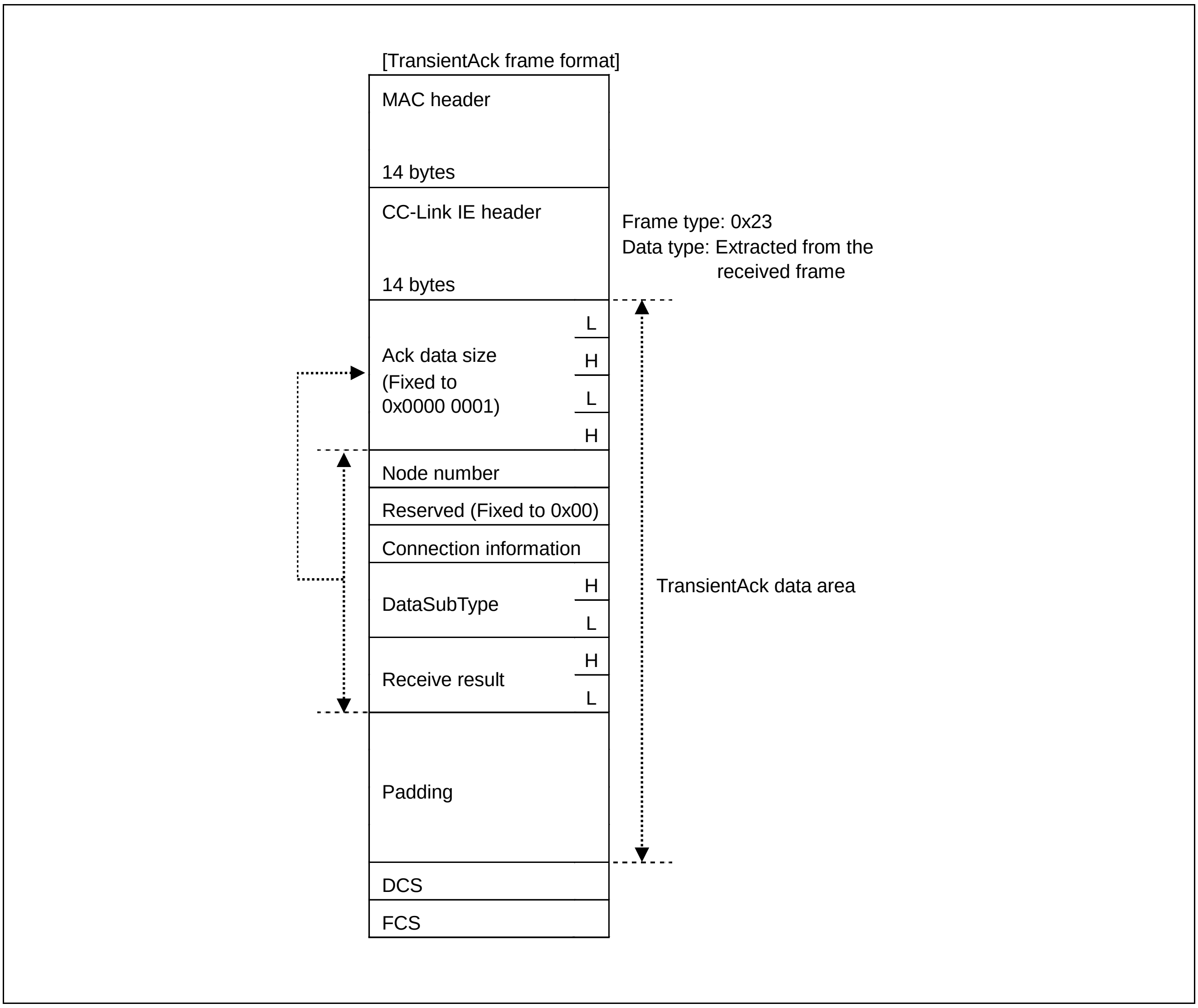


Figure 5.20 TransientAck Frame Format Overview

(1) MAC header, CC-Link IE header

Refer to Section 5.3.1 "Transient frame common format".

(2) TransientAck data area

Table 5.21 TransientAck Data Area Items

Item	Description	Value	Remarks
Ack data size	Data size from node number to receive result	Fixed to 0x0000 0001	-
Node number	Node number of TransientAck frame send destination	Node number of received Transient1 or Transient2 frame send source	When a transient frame is received from the master station (send source node number: 0x0000), set the destination node number after converting the value to "0x007D".
Reserved	Reserved	Fixed to 0x00	-
Connection information	Connection information loopback value of Ack send target frame (Connection Information)	Connection information of received Transient1 or Transient2 frame	-
Data sub-type	Data sub-type of received Transient1 frame	0x0002: Transient1 0x0000: Transient2	For Transient2, set the data sub-type to the fixed value of 0x0000.
Receive result	Receive result (RET) of Transient1 frame or Transient2 frame	0x0000: Normal Other than 0x0000: Abnormal	-
Padding	Padding (16 bytes)	-	To satisfy the minimum Ethernet frame size of 64 bytes, padding is automatically performed by R-IN32M3-CL.

5.3.4 CC-Link compatible transient frame format

The following shows the format of CC-Link compatible transient frame.

Table 5.22 Overview of CC-Link Compatible Transient Frame Format

No.	Item		Size (Bytes)	Remarks
1	MAC header		14	Refer to Section 5.3.1
	CC-Link IE header		14	
2	Transient2 header	Request	26	-
		Response	28	-
3	Transient2 data area	Request	0 to 960	-
		Response	0 to 960	-
4	DCS		4	Data Check Sequence[Note]
5	FCS		4	Frame Check Sequence[Note]

Note. Automatically calculated and added by R-IN32M3-CL.

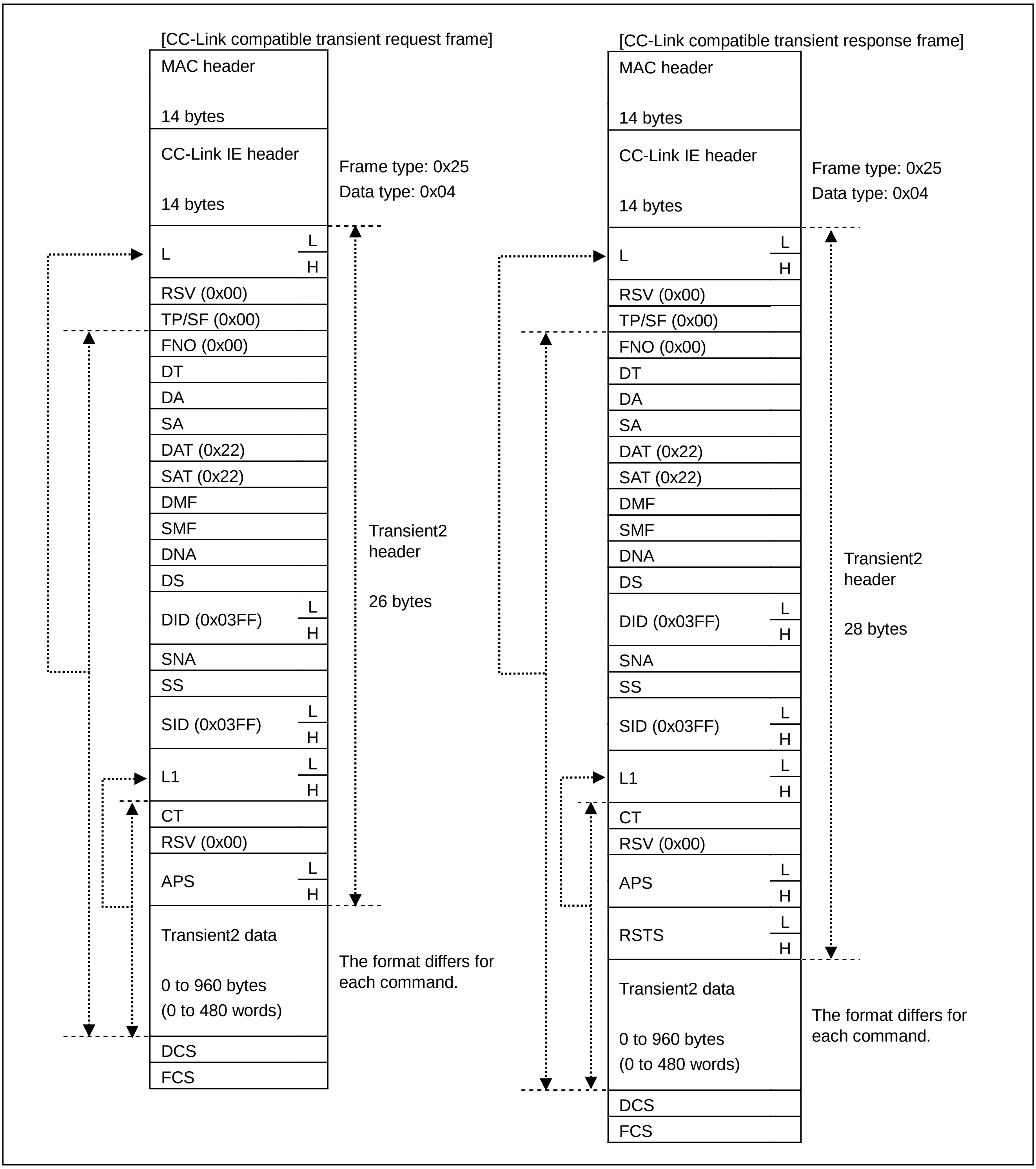


Figure 5.21 Overview of CC-Link Compatible Transient Frame Format

(1) MAC header, CC-Link IE header

Refer to Section 5.3.1 "Transient frame common format"

(2) Transient2 header

Table 5.23 Transient2 Header Items

Item	Description		Value	Remarks
L	Frame length (bytes)		22 to 982: CC-Link compatible transient 41 to 1440: SLMP	CC-Link compatible transient: FNO to Transient2 data SLMP: RSV to SLMP data
RSV	Reserved		Fixed to 0x00	-
TP/SF	Not used (type/sequence number)		Fixed to 0x00	-
FNO	Not used (divided frame number)		Fixed to 0x00	-
DT	Not used (data frame type)		Fixed to 0x00	-
DA	Destination node number		0x01 to 0x78 (1 to 120): Station number 0x7D: Specified control station/master station 0x7E: Current control station/master station 0xFF: Global request	Same value as DS
SA	Source node number		0x01 to 0x78 (1 to 120): Station number	Same value as SS
DAT	Destination application type		Fixed to 0x22	-
SAT	Source application type		Fixed to 0x22	-
DMF	Destination module flag		0x00: CC-Link compatible transient 0x03: SLMP	-
SMF	Source module flag		0x00: CC-Link compatible transient 0x03: SLMP	-
DNA	Destination network number		0x01 to 0xEF (1 to 239)	-
DS	Destination node number		0x01 to 0x78 (1 to 120): Station number 0x7D: Specified control station/master station 0x7E: Current control station/master station 0xFF: Global request	Same value as DA
DID	Destination identification number		Fixed to 0x03FF	-
SNA	Source network number		0x01 to 0xEF (1 to 239)	-
SS	Source node number		0x01 to 0x78 (1 to 120)	Same value as SA
SID	Source identification number		Fixed to 0x03FF	-
L1	Data length (bytes)		4 to 972	Size (bytes) from CT to DATA
CT	Command type		0x04 to 0x1F: CC-Link compatible transient 0x30: SLMP request 0xB0: SLMP response	For command type of CC-Link compatible transient, refer to Table 5.25.
RSV	Reserved		Fixed to 0x00	-
APS	Application number	Bits 15-8	Fixed to 0x00	Set a number in the sequential order to identify the order of the frame when the source station sends a request.
		Bits 7-0	0x00 to 0xFF	
RSTS	Return code		0x0000: Normal Other than 0x0000: Error code	During response only. For details on return code, refer to (a) in this section.

(a) Return code (RSTS)

The return code (RSTS) is an area where the server stores the error code in the response frame when an error exists in the client request frame.

[When the own station is a client]

During the response frame receive processing, the error code of an error detected in the request frame sent by the own station is stored.

Refer to the user's manual of the request destination device (Mitsubishi Electric or partner manufacturer product) and correct the request frame creation processing or the request send processing.

[When the own station is a server]

During the response frame send processing, store the error code of an error detected in the request frame sent by the client.

The error code can be defined by a user. The following table lists error code examples.

Table 5.24 Examples of Error Codes Stored in Return Code

No.	Error Code	Description	Action
1	0000h	Normal	-
2	D203h	Transient data read/write address specification error	Correct the read/write addresses in the transient request source, and perform the processing again.
3	D213h	Transient data command error	Correct the request command in the transient request source, and perform the processing again.
4	D218h	Transient data read/write data size error	Correct the read/write data size in the transient request source, and perform the processing again.
5	D219h	Transient data attribute code error	Correct the attribute code in the transient request source, and perform the processing again.
6	D21Ah	Transient data access code error	Correct the access code in the transient request source, and perform the processing again.
7	D2AEh	Transient data destination station number error	Transient data addressed to a different network/station number has been received. Check the network number and the destination station number, and perform the processing again.
8	D2A0h	Receive buffer full error	Check the network status by executing the CC-Link IE Field Network diagnostics. When transient data reception of the destination station is overloaded, have the send source send the data after a desired period of time has elapsed.
9	D2A1h	Send buffer full error	Decrease the transient transmission frequency, and perform the processing again. Check that there is no error in the cable and switching hub connections in the request source.
10	D2A3h	Transient data frame length (L) error	Correct the corresponding error in the Transient2 header, and perform the processing again.
11	D2A4h	Transient data reserved (RSV) error	
12	D2A5h	Transient data destination station number (DA) error	
13	D2A6h	Transient data source station number (SA) error	
14	D2A7h	Transient data destination application type (DAT) error	
15	D2A8h	Transient data source application type (SAT) error	
16	D2A9h	Transient data destination network number (DNA) error	
17	D2AAh	Transient data destination station number (DS) error	
18	D2ABh	Transient data source network number (SNA) error	
19	D2ACh	Transient data source number (SS) error	
20	D2ADh	Transient data length (L1) error	

(b) Command type (CT)

The following shows the data structure of the CC-Link compatible transient command type (CT).

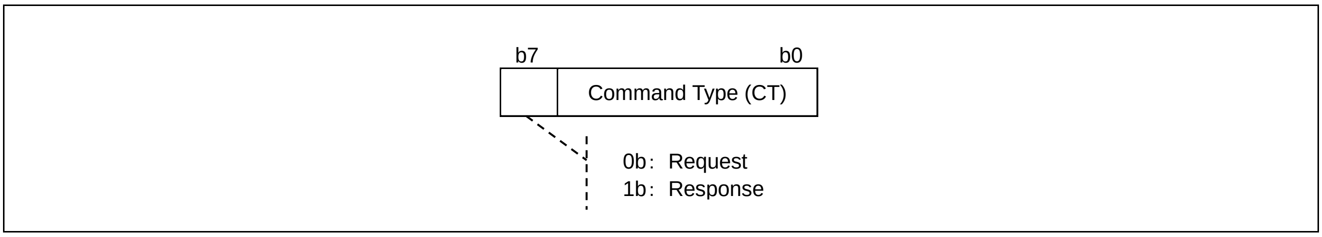


Figure 5.22 Data Structure of Command Type (CT)

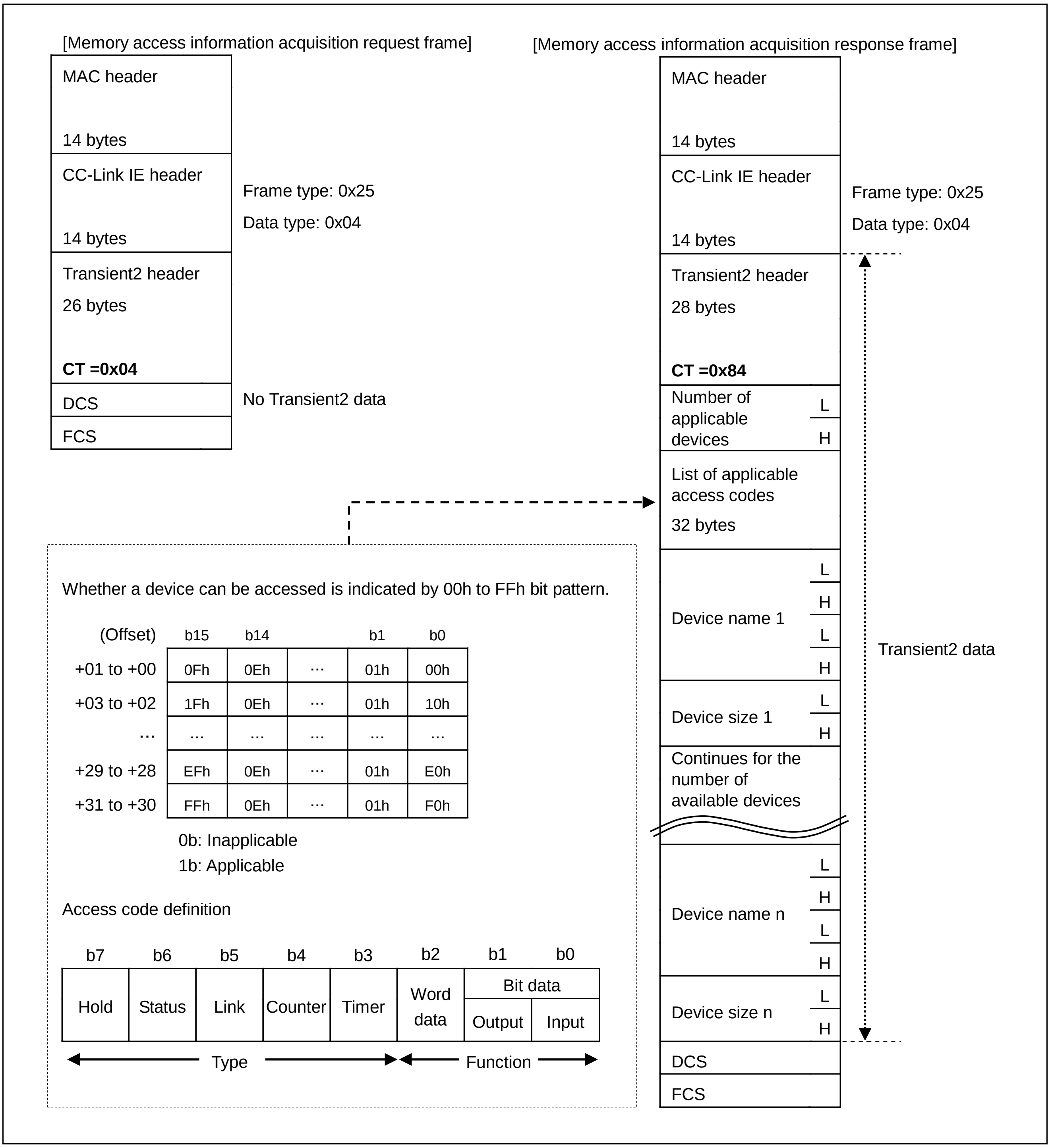
Table 5.25 CC-Link Compatible Transient Command Type List

CT	Command type
0x04	Memory access information acquisition request
0x84	Memory access information acquisition response
0x08	RUN request
0x88	RUN response
0x09	STOP request
0x89	STOP response
0x10	Memory read request
0x90	Memory read response
0x12	Memory write request
0x92	Memory write response

(3) Transient2 data area

(a) Memory access information acquisition

The memory access information acquisition request allows you to acquire applicable devices of the destination controller and access codes.



Whether a device can be accessed is indicated by 00h to FFh bit pattern.

(Offset)	b15	b14		b1	b0
+01 to +00	0Fh	0Eh	···	01h	00h
+03 to +02	1Fh	0Eh	···	01h	10h
···	···	···	···	···	···
+29 to +28	EFh	0Eh	···	01h	E0h
+31 to +30	FFh	0Eh	···	01h	F0h

0b: Inapplicable
1b: Applicable

Access code definition

b7	b6	b5	b4	b3	b2	b1	b0
Hold	Status	Link	Counter	Timer	Word data	Bit data	
						Output	Input

Figure 5.23 Overview of Memory Access Information Acquisition Frame Format

(b) RUN

The RUN request changes the operating status of another station to RUN.

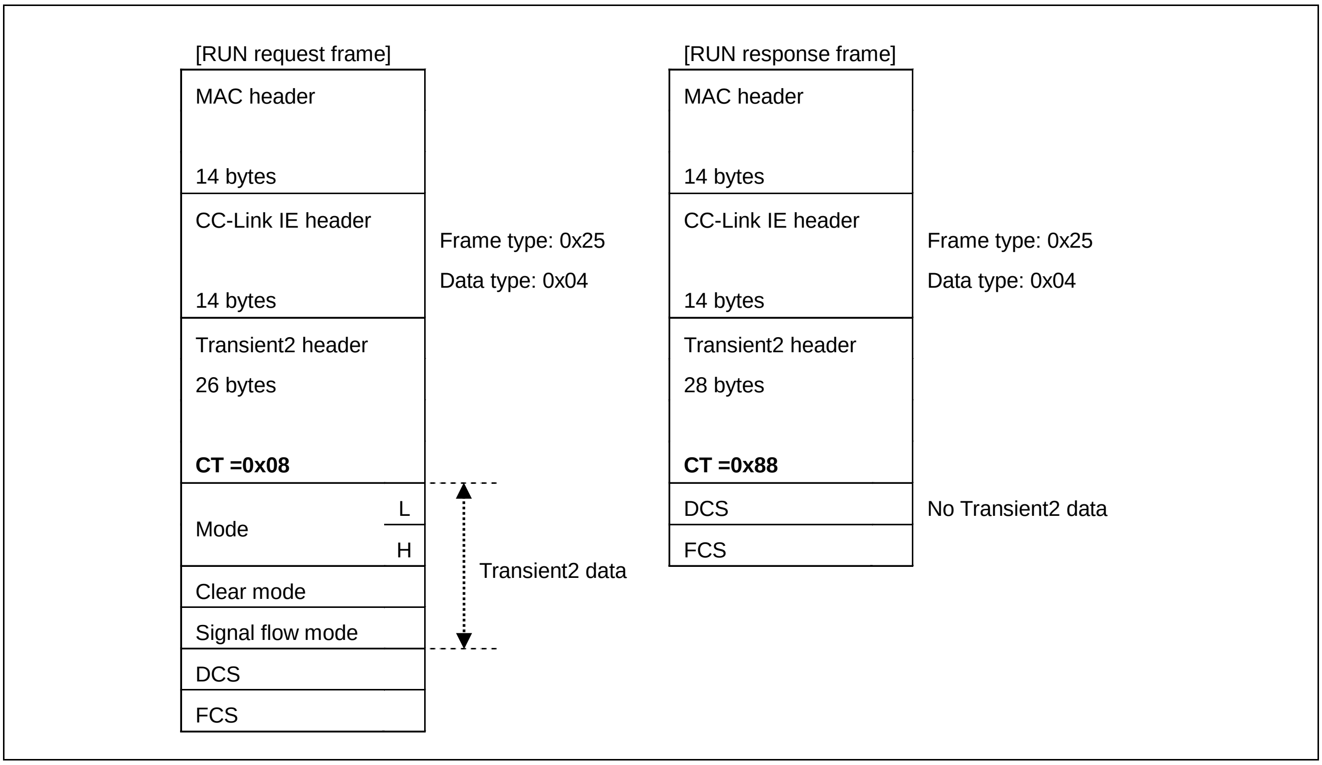


Figure 5.24 Overview of RUN Frame Format

Table 5.26 RUN Request Setting List

Item	Setting	Value
Mode	Normal RUN	0x0003
	Forced RUN	0x0001
Clear mode	Clear all	0x02
	Clear all areas other than latch range	0x01
	Do not clear device	0x00
Signal flow mode	Fixed value	0x00

(c) STOP

The STOP request changes the operating status of another station to STOP.

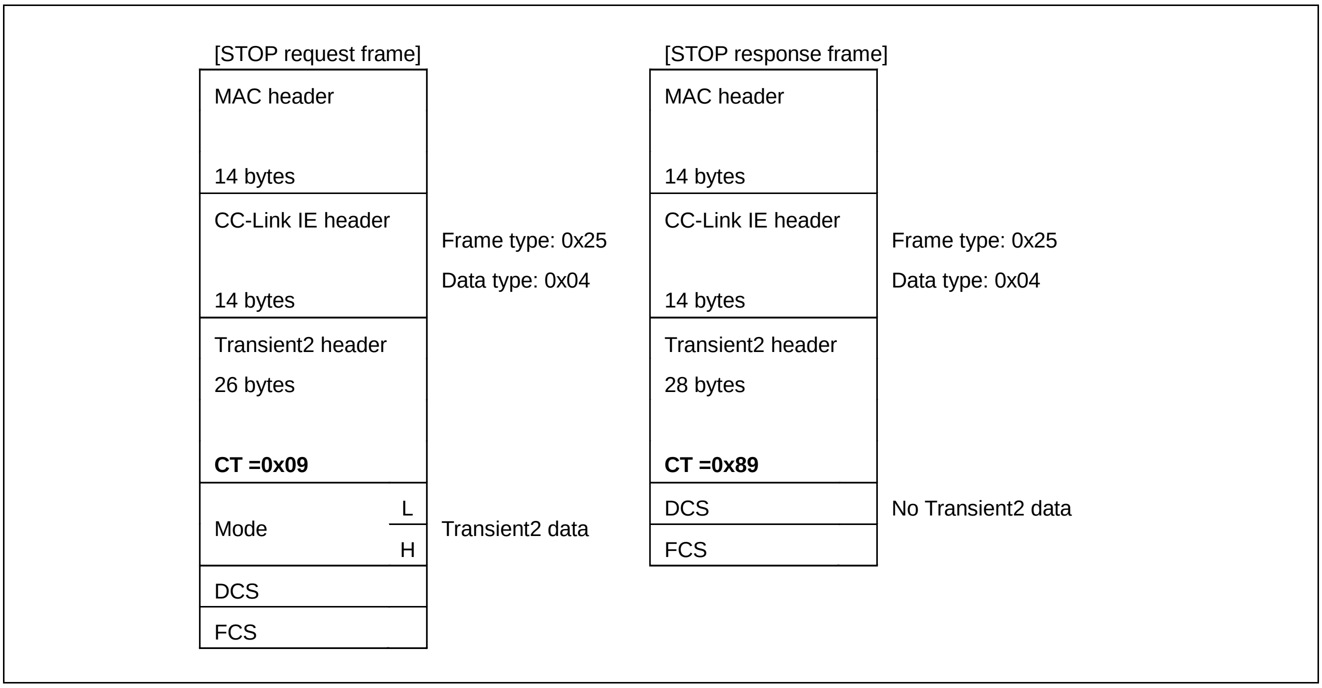


Figure 5.25 Overview of STOP Frame Format

Table 5.27 STOP Request Setting List

Item	Setting	Value
Mode	Normal STOP	0x0003
	Forced STOP	0x0001

(d) Memory read

The memory read request retrieves data from devices of another station.

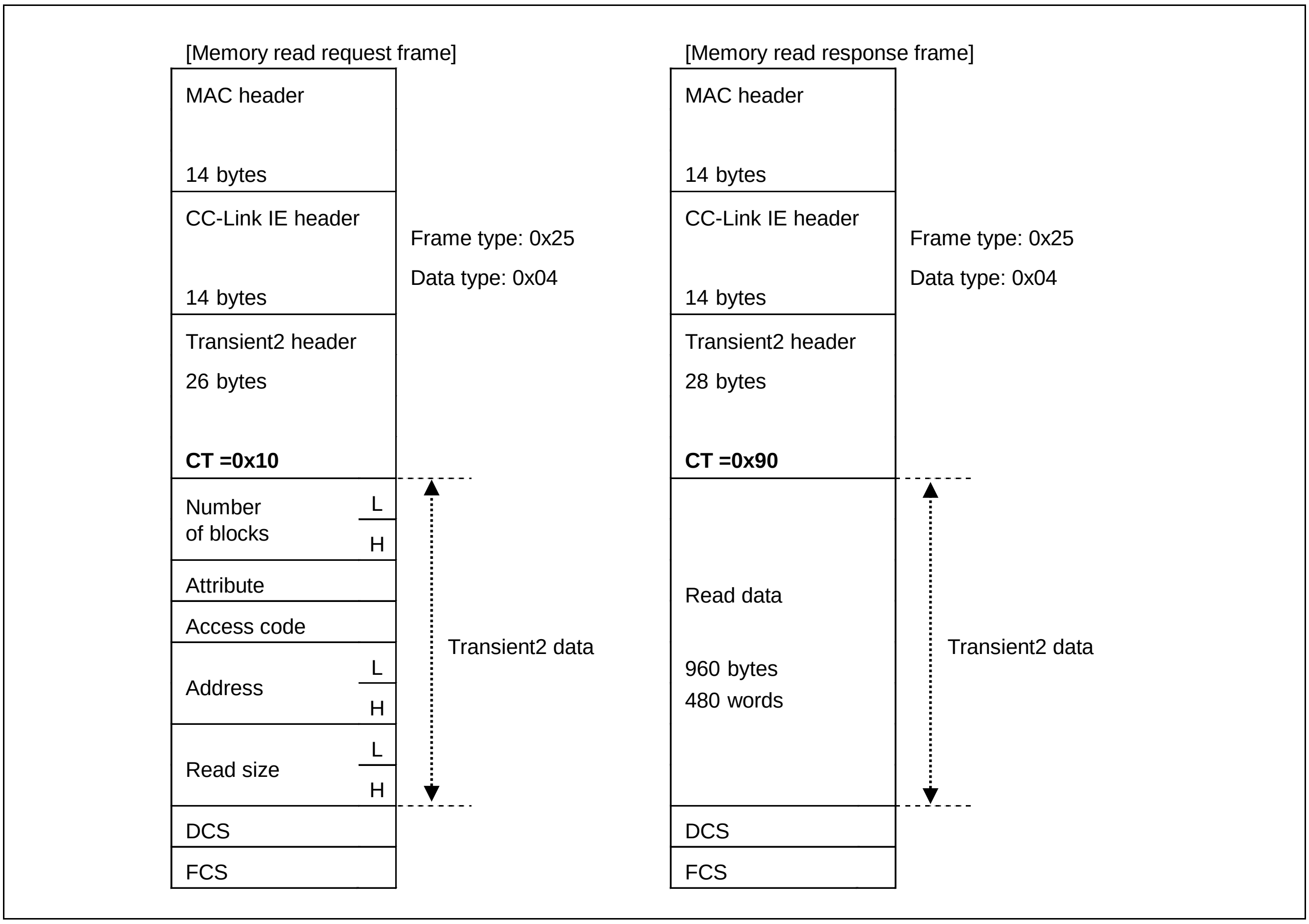


Figure 5.26 Overview of Memory Read Frame Format

Table 5.28 Memory Read Request Setting List

Item	Setting	Value
Number of blocks	Number of blocks from Attribute to Read Size.	Fixed to 0x0001
Attribute	Refer to Figure 5.29	-
Access code	Refer to Figure 5.28	-
Address	Address of device	0 to 65535
Read size	Unit: Words	1 to 480

Remark.
- **This frame is sent when the dedicated instruction RIRD is executed in a Mitsubishi Electric programmable controller.**
- **When sending a request to a Mitsubishi Electric product, set the attribute to 0x05 and access code to a value according to Table 5.30.**

(e) Memory write

The memory write request writes data to devices of another station.

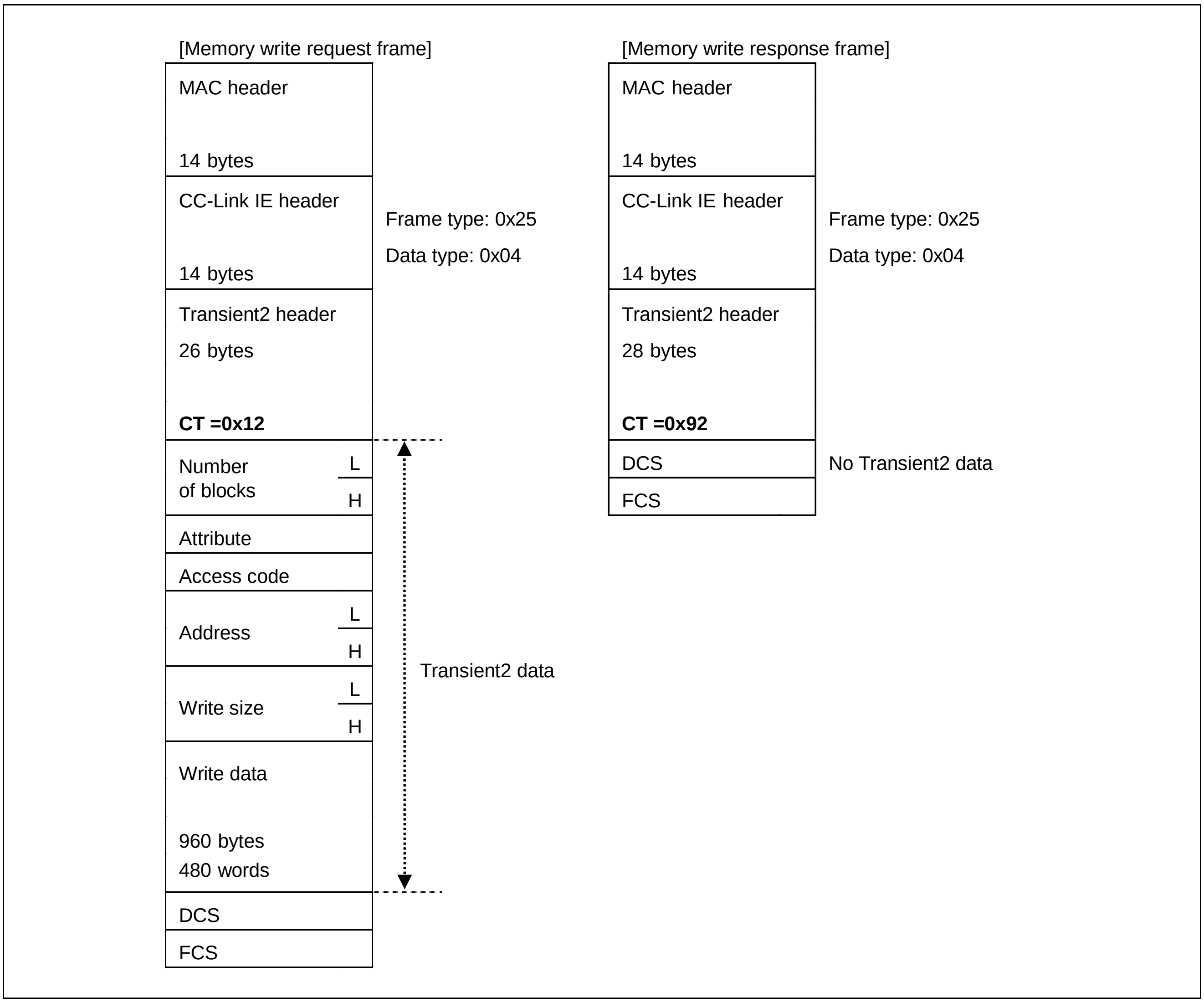


Figure 5.27 Overview of Memory Write Frame Format

Table 5.29 Memory Write Request Setting List

Item	Setting	Value
Number of blocks	Number of blocks from Attribute to Write Size	Fixed to 0x0001
Attribute	Refer to Figure 5.29	-
Access code	Refer to Figure 5.28	-
Address	Address of device	0 to 65535
Write size	Unit: Words	1 to 480

Remark. **・This frame is sent when the dedicated instruction RIWT is executed in a Mitsubishi Electric programmable controller.**

・When sending a request to a Mitsubishi Electric product, set the attribute to 0x05 and access code to a value according to Table 5.30.

(f) Access codes and attributes

The following are the definitions of an access code and an attribute.

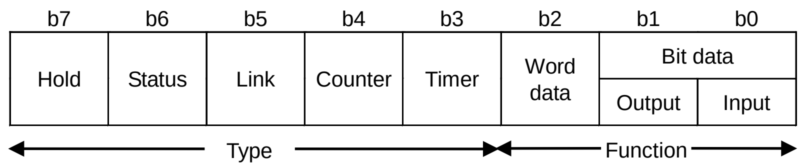


Figure 5.28 Access Code Definition

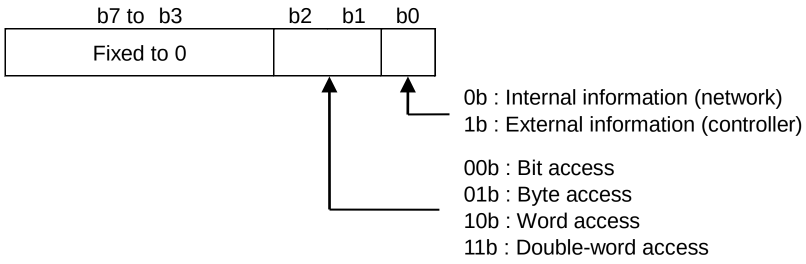


Figure 5.29 Attribute Definition

[When the own station is a server]

Define the device/buffer memory areas of the own station so that another station (Mitsubishi Electric product or developed device) can access them by using the memory read/write commands.

[When the own station is a client]

Refer to the following table when accessing another station (Mitsubishi Electric product) from the own station by using the memory read/write commands.

The number of device points (size) differs depending on the programmable controller. For the accessible range, refer to the user's manual of the programmable controller used.

When accessing to a station other than Mitsubishi Electric products, refer to the user's manual of the station.

Table 5.30 Mitsubishi Electric Product Access Code List

Device	Symbol	Device Type		Unit	Access Code [Note1]	Attribute Code [Note1]
		Bit	Word			
Input relay	X	○	-	Hexadecimal	0x01	0x05
Output relay	Y	○	-	Hexadecimal	0x02	
Special relay	SM	○	-	Decimal	0x43	
Special register	SD	-	○	Decimal	0x44	
Internal relay	M	○	-	Decimal	0x03	
Latch relay	L	○	-	Decimal	0x83	
Timer (contact)	T	○	-	Decimal	0x09	
Timer (coil)	T	○	-	Decimal	0x0A	
Timer (current value)	T	-	○	Decimal	0x0C	
Retentive timer (contact)	ST	○	-	Decimal	0x89	
Retentive timer (coil)	ST	○	-	Decimal	0x8A	
Retentive timer (current value)	ST	-	○	Decimal	0x8C	
Counter (contact)	C	○	-	Decimal	0x11	
Counter (coil)	C	○	-	Decimal	0x12	
Counter (current value)	C	-	○	Decimal	0x14	
Data register	D[Note2]	-	○	Decimal	0x04	
File register	R	-	○	Decimal	0x84	
Link relay	B	○	-	Hexadecimal	0x23	
Link register	W [Note2]	-	○	Hexadecimal	0x24	
Link special relay	SB	○	-	Hexadecimal	0x63	
Link special register	SW	-	○	Hexadecimal	0x64	

Note 1. If the target station is a station other than the master/local module, refer to the user's manual of the target station for the access codes and attribute codes.

2. The extended data register (D65536 and later) and the extended link register (W10000 and later) cannot be specified.

5.3.5 SLMP frame format

The following table provides an overview of the SLMP frame format.

Table 5.31 SLMP Frame Format Overview

No.	Item			Size (Bytes)	Remarks
1	MAC header			14	Refer to Section 5.3.1
	CC-Link IE header			14	
2	Transient1 header			16	Refer to Section 5.3.2(2)
3	Transient1 data area[Note2]	Transient2 header	Request	26	Refer to Section 5.3.4(2)
			Response	28	
4		SLMP header		15	-
5		SLMP data	Request	0 to 1425	-
			Response	0 to 1423	
6	DCS			4	Data Check Sequence[Note1]
7	FCS			4	Frame Check Sequence[Note1]

Note 1. Automatically calculated and added by R-IN32M3-CL.

2. When Transient1 data area is used as "SLMP". Refer to the Section 5.3.2 "CC-Link IE Field specific transient frame format" when using the Transient1 data area as "CC-Link IE Field specific transient".

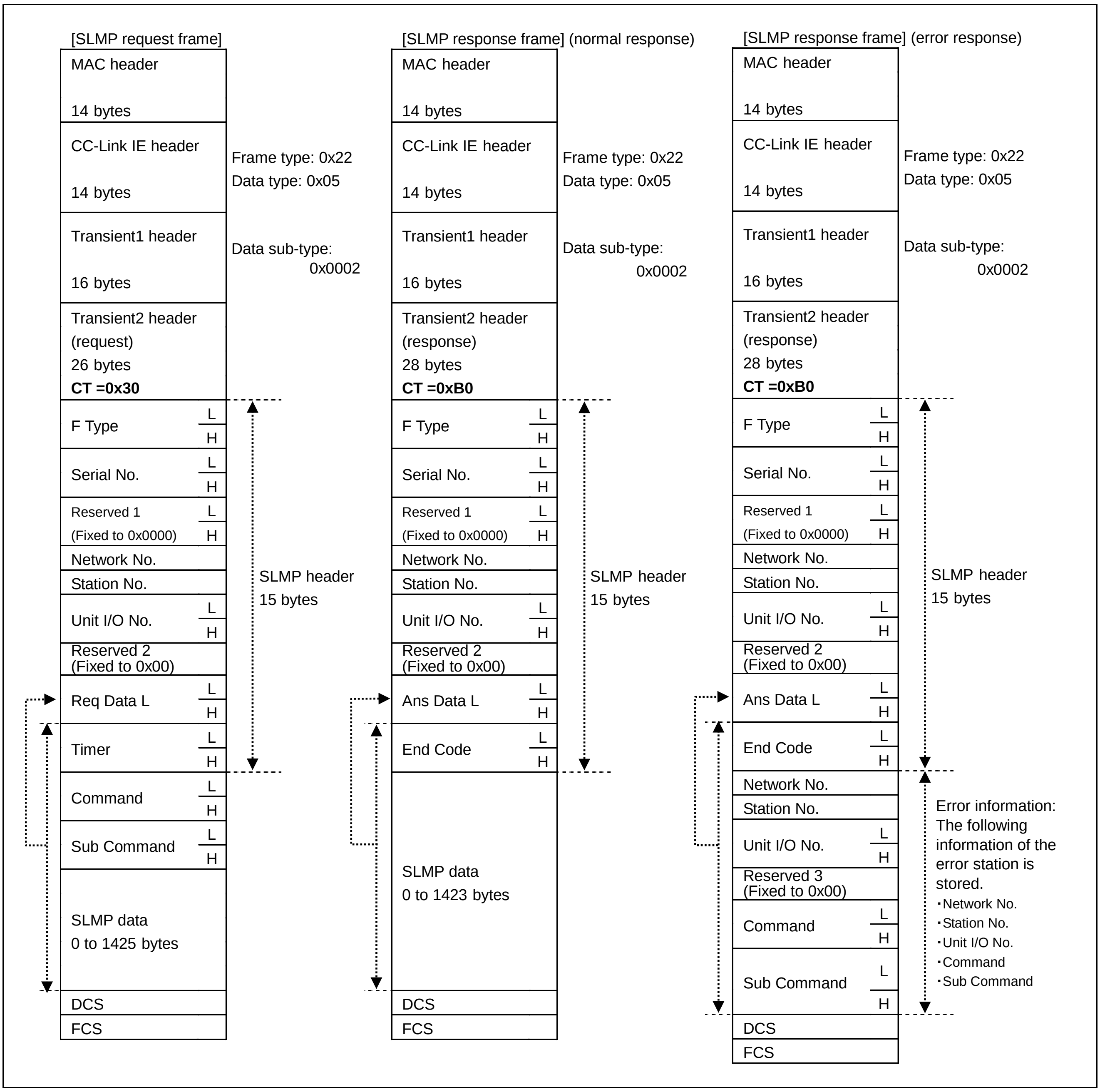


Figure 5.30 SLMP Frame Format Overview

(1) MAC header, CC-Link IE header

Refer to Section 5.3.1 "Transient frame common format".

(2) Transient1 header

Refer to Section 5.3.2(2) "Transient1 header".

(3) Transient2 header

Refer to Section 5.3.4(2) "Transient2 header".

(4) SLMP header

Table 5.32 SLMP Header Items

Item	Description	Value	Remarks
F Type	Frame type	0x0054: During request 0x00D4: During response	-
Serial No.	Serial number	0x0000 to 0xFFFF	Set a number to identify the frame. Set the same value for a request frame and the corresponding response frame.
Network No.	Destination network number	0x00: Own station 0x01 to 0xEF (1 to 239): Other station	Set the network number of the destination station.
Station No.	Destination station number	0x01 to 0x78 (1 to 120): Station number 0x7D: Specified control station/master station 0x7E: Current control station/master station 0xFF: Own station[Note]	Set the destination station number.
Unit I/O No.	Destination module I/O number	Fixed to 0x03FF	Set the access destination CPU module.
Req Data L	Request data length	-	Set the request data size, from Timer to the end of the data area, in bytes.
Ans Data L	Response data length	-	Set the response data size, from End Code to the end of the data area, in bytes.
Timer	Monitoring timer	0x0001 to 0xFFFF 0x0000: Unlimited	Request frame only. Set the wait time (in increments of 250 ms) for the client to receive a response from the server. Recommended values: Own station: 0001h to 0028h (0.25 to 10 s) Other stations: 0002h to 00F0h (0.5 to 60 s)
End Code	End code	0x0000: Normal end Other than 0x0000: Error code	Response frame only. For details on end codes, refer to (a) in this section.

Note. Effective only when the Network No. is set to 0x00

(a) End code (End Code)

The end code (End Code) is an area where the server stores the error code in the response frame when an error exists in the client request frame.

[When the own station is a client]

During the response frame receive processing, the error code of an error detected in the request frame sent by the own station is stored.
Refer to the user's manual of the request destination device and correct the request frame creation processing or the request send processing.

[When the own station is a server]

During the response frame send processing, store the error code of an error detected in the request frame sent by the client.
The error code can be defined by a user. The following table lists error code examples.

Table 5.33 Examples of Error Codes Stored in End Code

No.	Category	Error Code	Description	Action
1	Successful completion	0000h	The request was processed normally.	-
2	General error	C059h	・There is an error in the command/subcommand specification. ・A command other than that in the specified sequence was received.	Correct the command/subcommand, and send the request again.
3		C05Ch	There is an error in the request message.	Correct the request message, and send the request again.
4		C061h	The request data length and data size do not match.	Correct the request data or the request data length, and send the request again.
5		CEE0h	Another request is being executed. The request cannot be processed.	Wait for a while, and send the request again.
6		CEE1h	The request message size exceeds the range that can be processed.	Correct the request message, and send the request again.
7		CEE2h	The response message size exceeds the range that can be processed.	Correct the request message, and send the request again.
8	Server information	CF10h	The specified server information number does not exist.	Correct the server information number, and send the request again.
9	Communication settings	CF20h	An item that cannot be set is included in the request message.	Correct the setting item (CSP+), and send the request again.
10	Parameter settings	CF30h	The specified parameter ID does not exist.	Correct the parameter and the parameter ID (CSP+), and send the request again.
11		CF31h	The write exclusive start processing has not been performed. The request cannot be processed.	Execute the write exclusive start processing, and send the request again.
12	Communication status	CF70h	An error occurred in the communication path of the relay destination. The request cannot be processed.	Check the communication path, and send the request again.
13		CF71h	A timeout occurred. The processing was interrupted.	Check the status of the destination device, and send the request again.

(5) SLMP data area

(a) SLMP memory read

SLMP memory read is used when retrieving data from the buffer memory of another station (SLMP-compatible device). The following shows the format of SLMP memory read frame.

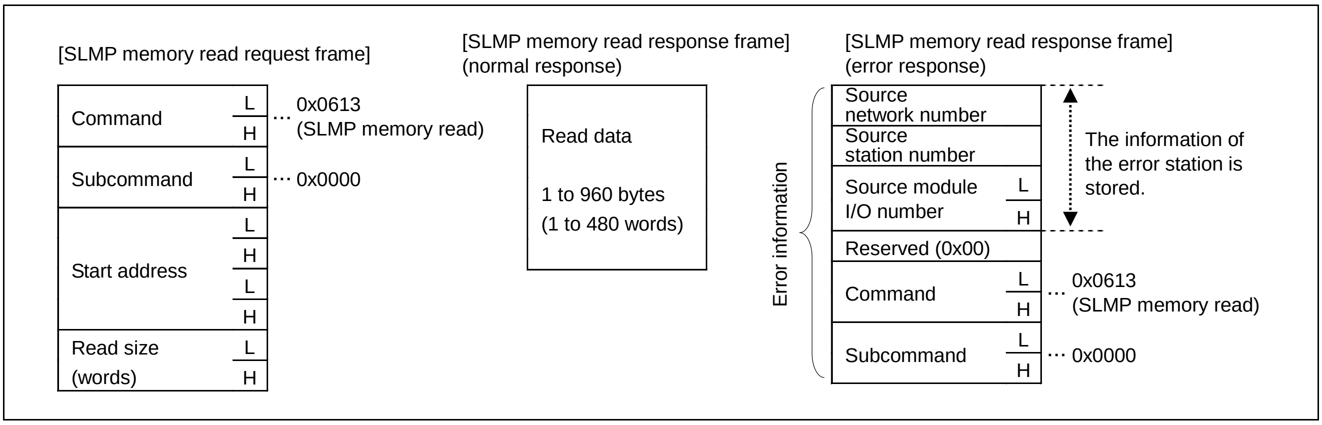


Figure 5.31 SLMP Memory Read Frame

The following table describes the details of items defined in the SLMP memory read frame format.

Table 5.34 Details of SLMP Memory Read Frame Format

Description	Value	Remarks
Command	0x0613	-
Subcommand	0x0000	-
Start address	-	Specify the start address of the buffer memory to be read.
Read size (words)	0x1 to 0x1E0 (1 to 480)	Specify the word length of the buffer memory to be read.
Source network number	0x00: Own station 0x01 to 0xEF (1 to 239): Other station	Specify the network number of the response sending station.
Source station number	0x01 to 0x78 (1 to 120): Station number 0xFF: Own station[Note]	Specify the station number of the response sending station.
Destination module I/O number	0x03FF: Fixed	Set the access destination CPU module.

Note. Effective only when the Network No. is set to 0x00

(b) SLMP memory write

SLMP memory write is used when writing data to the buffer memory of another station (SLMP-compatible device). The following shows the format of an SLMP memory write frame.

Note that when the response is returned normally, there is no SLMP data area. (The SLMP header, DCS, and FCS are required.)

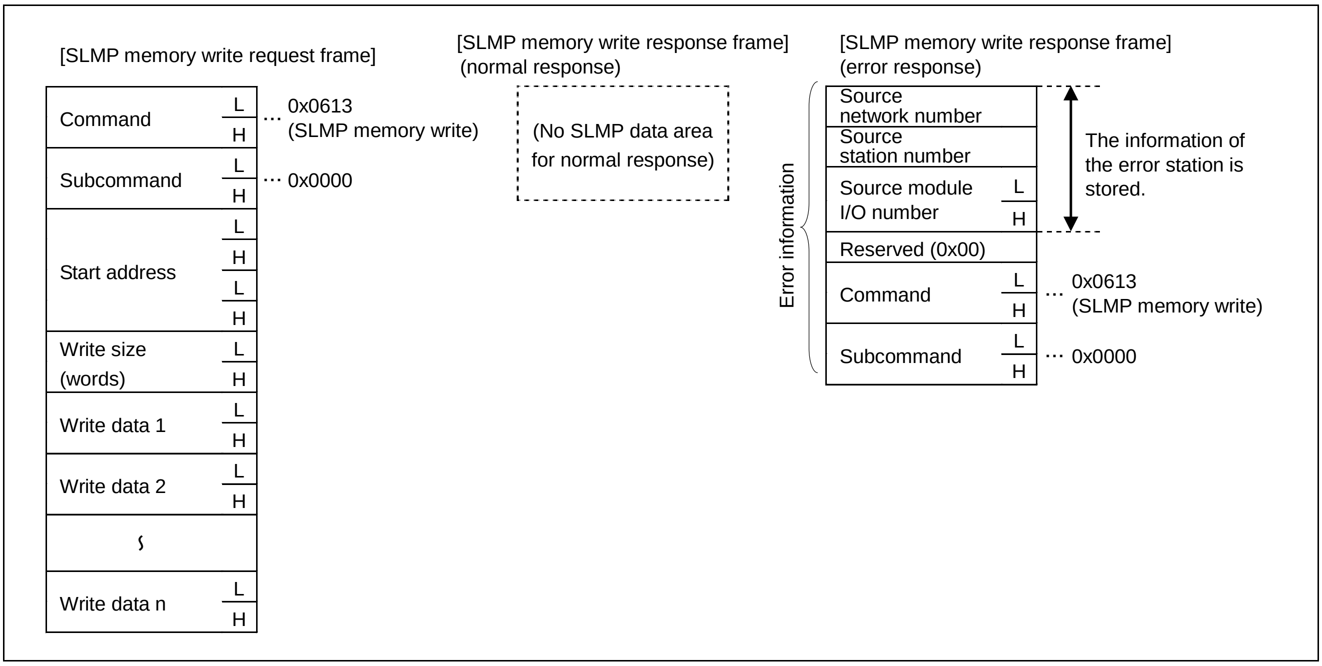


Figure 5.32 SLMP Memory Write Frame

The following table describes the details of items defined in the SLMP memory write frame format.

Table 5.35 Details of SLMP Memory Write Frame Format

Description	Value	Remarks
Command	0x1613	-
Subcommand	0x0000	-
Start address	-	Specify the start address of the buffer memory to be written.
Write size (words)	0x0001 to 0x01E0 (1 to 480)	Specify the word length of the buffer memory to be written.
Write data	-	Set the data to be written.
Source network number	0x00: Own station 0x01 to 0xEF (1 to 239): Other station	Specify the network number of the response sending station.
Source station number	0x01 to 0x78 (1 to 120): Station number 0x7D: Specified control station/master station 0x7E: Current control station/master station 0xFF: Own station[Note]	Specify the station number of the response sending station.
Destination module I/O number	0x03FF: Fixed	Set the access destination CPU module.

Note. Effective only when the Network No. is set to 0x00

5.4 MyStatus Overview

MyStatus is used to report the status of nodes connected to the network.

R-IN32M3-CL sets own station information in MyStatus frame and notify the master station of it. It also receives MyStatus frame from the master station and monitors the status of the master station.

5.4.1 Sending MyStatus

The user program sets the own station information in arguments of the function gerR_IN32_SetNodeStatus, and the R-IN32M3-CL driver sets the information in MyStatus frame and sends it to the master station. The following table lists the own station information that is set in MyStatus frame by UserSendMyStatus (MyStatus send processing).

Table 5.36 Information Related to Sending MyStatus

No.	Item	Description
1	Detailed application operation status	Stores the operation status of the user application. 0000h: Detailed application operation status notification not supported 0001h: Application stopped 0002h: Application running 0003h: Application does not exist Other than the above: Not used
2	Detailed application error status	Stores the error status when a user application error occurs. 0000h: No error 0001h: Minor error 0002h: Moderate error 0003h: Major error Other than the above: Not used

5.4.2 Receiving MyStatus

The R-IN32M3-CL driver receives MyStatus frames from the master station.

The following table lists the master station information that is acquired in MyStatus frame by UserReceiveCyclic (MyStatus from master station and cyclic receive processing).

Table 5.37 Information Related to Receiving MyStatus

No.	Item	Description
1	Master station application operation status	Stores the operation status of the master station application.Note1 0b: Application stopped 1b: Application running
2	Master station application error status	Stores the error status of the master station application.Note2 0b: No error 1b: Error

Note 1. When a Mitsubishi Electric master station is used, the following status of the programmable controller CPU module will be stored.

[Application stopped]

Operation stop of a sequence program (when the RUN/STOP switch is set to "STOP" or a moderate/major error occurs).

[Application running]

Operation execution of a sequence program (when the RUN/STOP switch is set to "RUN").

2. When a Mitsubishi Electric master station is used, the following status of the programmable controller CPU module will be stored.

[No error]

No error, or an error in which the CPU module continues operation such as a battery error (minor error).

[Error]

An error in which the CPU module stops operation such as a WDT error (moderate error), and an error in which the CPU module stops operation such as a hardware failure (major error).

6. DEVELOPING FIRMWARE

6.1 Development Procedure

This section describes the procedure for developing firmware using the sample code on the CD-ROM provided with this document.

The sample code comprises the program parts described in Table 6.1. While customization of the R-IN32M3-CL driver main unit is not required, other program parts must be customized in accordance with the hardware of a device to be developed (target).

Table 6.1 List of Program Parts Included in Sample Code

Program Part Name	Overview	Need for Change
User program	An application program created by the user. A program is used as reference for checking the communication function logic of an intelligent device station (sample program), and therefore customize it as necessary.	Customization required
R-IN32M3-CL driver interface functions	Functions called when a function of the R-IN32M3-CL driver is used from the user program.	Not required
R-IN32M3-CL driver target-dependent functions	Functions that must be customized in accordance with the hardware environment of the target user.	Customization required
R-IN32M3-CL driver callback functions	Functions used when the user program requests callback from the R-IN32M3-CL driver. Describes the processing on the user program side for events that occur in the R-IN32M3-CL driver.	Customization required
R-IN32M3-CL driver main body	The main body of the driver area that is called by R-IN32M3-CL driver interface functions and controls R-IN32M3-CL.	Not required

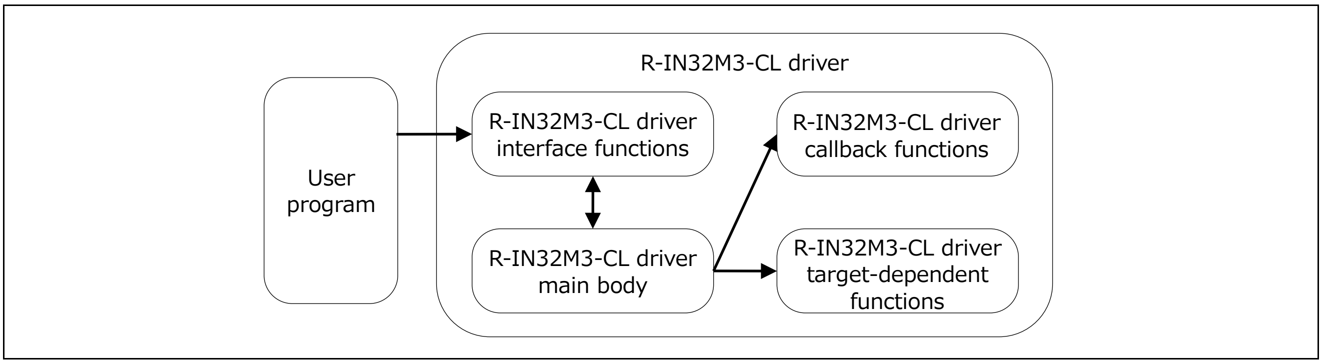

R-IN32M3-CL driver
User program
R-IN32M3-CL driver interface functions
R-IN32M3-CL driver callback functions
R-IN32M3-CL driver main body
R-IN32M3-CL driver target-dependent functions

Figure 6.1 Sample Code Configuration

Caution. The sample code provided in this reference manual has been verified that a compilation error does not occur based on "GCC (GNU C Compiler) Version 4.3.4". The sample code is not operating system or MPU dependent. Customize the sample code in accordance with the user environment.

The following describes the procedure for developing firmware.

Step 1: Creating a user program

Create a user program while referring to Section 6.2.1 "Main processing".

Step 2: Customizing the R-IN32M3-CL driver target-dependent functions

Customizes the R-IN32M3-CL driver target-dependent functions in accordance with the hardware of the device to be developed. For details, refer to Section 6.5 "Customizing the R-IN32M3-CL Driver Target-Dependent Functions".

Step 3: Customizing the R-IN32M3-CL driver callback functions

Customize the R-IN32M3-CL driver callback functions in accordance with the hardware of the device to be developed. For details, refer to Section 6.6 "Customizing the R-IN32M3-CL Driver Callback Functions".

Step 4: Creating the R-IN32M3-CL library

Compile the files for the R-IN32M3-CL driver main body and the R-IN32M3-CL driver target-dependent functions, execute the librarian, and create the R-IN32M3-CL driver library files.

Step 5: Connecting the user program and library files

Connect the user program, the customized R-IN32M3-CL driver callback functions, and the library files, and then create the load module file.

Step 6: Load the load module file into the device to be developed (target).

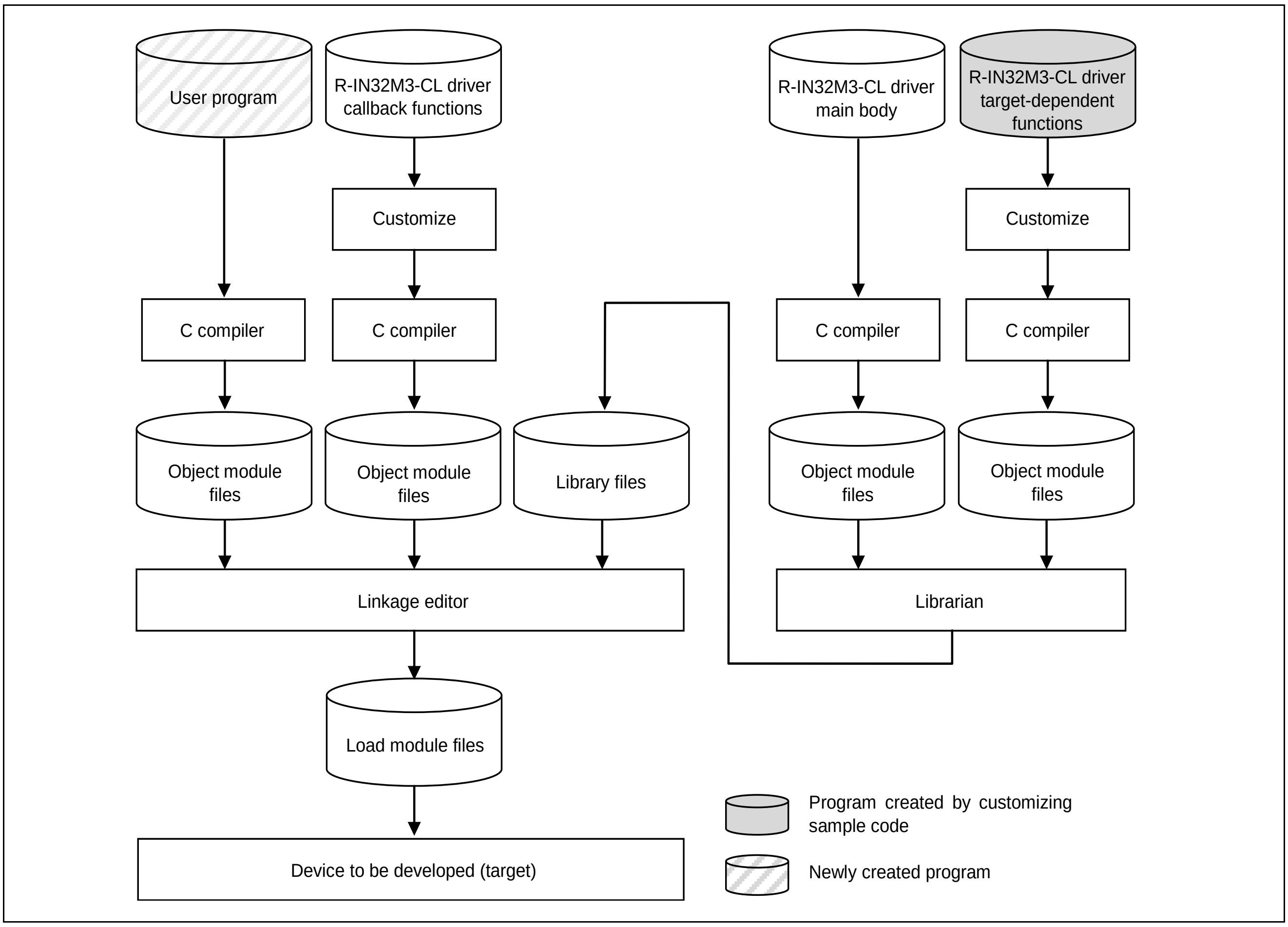


Figure 6.2 Firmware Development Procedure

6.1.1 Sample code file list

The following table lists the sample code files.

Table 6.2 Sample Code File List (1/2)

Folder		File	Description
sample	include	R_IN32M3Callback.h	R-IN32M3-CL driver callback functions header file (Refer to Section 6.6)
	obj	makefile	Makefile (For user program construction)
	src	R_IN32M3_Callback.c	R-IN32M3-CL driver callback functions source file (Refer to Section 6.6)
		R_IN32M3_HWTest.c	User program (Hardware test processing) source file
		R_IN32M3_HWTest.h	User program (Hardware test processing) header file
		R_IN32M3_sample.c	User program (Main processing and others) source file
		R_IN32M3_sample.h	User program (Main processing and others) header file
		R_IN32M3_Transient.c	User program (Transient send/receive processing) source file
		R_IN32M3_Transient.h	User program (Transient send/receive processing) header file
driver	include	R_IN32M3Driver.h	R-IN32M3-CL driver header file
		R_IN32M3Function.h	R-IN32M3-CL driver target-dependent functions header file (Refer to Section 6.5.1)
		R_IN32M3Types.h	R-IN32M3-CL driver interface functions header file (Refer to Section 6.4)
	obj	makefile	Makefile (For R_IN32M3Driver.a construction)
	src	R_IN32_Interface.c	R-IN32M3-CL driver interface functions source file (Refer to Section 6.3)
		R_IN32R.c	R-IN32M3-CL driver target-dependent functions source file (Refer to Section 6.5.2)
		R_IN32R.h	R-IN32M3-CL driver target-dependent functions header file (Refer to Section 6.5.2)
		R_IN32C_l.h	R-IN32M3-CL driver main body
		R_IN32C_Library.c	
		R_IN32C_MainState.c	
		R_IN32C_PortState.c	
		R_IN32C_R_IN32DInterface.c	
		R_IN32C_Time.c	
		R_IN32D.h	
		R_IN32D_cyc.c	
		R_IN32D_cyc_l.h	
		R_IN32D_ihnd.c	
		R_IN32D_ini.c	
		R_IN32D_intr.c	
		R_IN32D_intr_l.h	
		R_IN32D_led.c	
		R_IN32D_phy.c	
		R_IN32D_phy_l.h	
		R_IN32D_RcvCnt.c	
		R_IN32D_RcvCnt_l.h	

Table 6.2 Sample Code File List (2/2)

Folder		File	Description
driver	src	R_IN32D_RcvPrm.c	R-IN32M3-CL driver main body
		R_IN32D_RcvPrm_l.h	
		R_IN32D_reg.c	
		R_IN32D_reg_l.h	
		R_IN32D_sub.c	
		R_IN32D_sub_l.h	
		R_IN32D_tran.c	
		R_IN32D_tran_l.h	
		R_IN32M3.h	
		R_IN32M3_0.h	
		R_IN32M3_1.h	
		R_IN32M3_2.h	
		R_IN32M3_3.h	
		R_IN32S.c	
		R_IN32S.h	
		R_IN32T.h	
		R_IN32T_ASIC.c	
		R_IN32T_ASIC.h	
		R_IN32T_Cmu.h	
		R_IN32T_CmuNCycRcv.c	
		R_IN32T_CmuOutLpBak.c	
		R_IN32T_CmuSub.h	
		R_IN32T_CmuSub3.c	
		R_IN32T_Com.c	
		R_IN32T_Com.h	
		R_IN32T_Data.c	
		R_IN32T_Data.h	
		R_IN32T_FrmForm.h	
		R_IN32T_MACIP.c	
		R_IN32T_MACIP.h	
		R_IN32T_RegChk.c	
		R_IN32T_RegChk.h	
		R_IN32T_RING.c	
		R_IN32T_RING.h	
		R_IN32T_TxFrame.c	
		R_IN32T_TxFrame.h	
		R_IN32U.h	
		R_IN32U_Init.c	
		R_IN32.h	
		R_IN32_Frame.h	
		R_IN32C.h	
		R_IN32C_Cyclic.c	
		R_IN32C_Data.c	
		R_IN32C_Indication.c	
		R_IN32C_Init.c	

6.2 Sample Flowcharts

This section provides the list of sample flowcharts for the user program. Processing described in each flowchart is a sample processing to implement functions of intelligent device station. Customize the user program using the sample flowcharts as a reference.

Table 6.3 List of Sample Flowcharts Related to Initial Processing and Cyclic Transmission (R_IN32M3_sample.c File)

No.	Overview	Reference	Implementation Required	Remarks
1	Main processing	Section 6.2.1	◎	
2	Initialization processing	Section 6.2.2	◎	
3	Communication start processing	Section 6.2.3	◎	
4	PHY check processing	Section 6.2.4	△	Necessity of implementation varies according to PHY.
5	PHY setting change processing	Section 6.2.5	△	
6	Own station error processing	Section 6.2.6	△	
7	Cyclic transmission stop processing	Section 6.2.7	△	
8	Event processing	Section 6.2.8	◎	
9	MyStatus from master station and cyclic receive processing	Section 6.2.9	◎	
10	MyStatus send processing	Section 6.2.10	◎	
11	Cyclic send processing	Section 6.2.11	◎	
12	Communication status update processing	Section 6.2.12	◎	
13	Cyclic transmission status update processing	Section 6.2.13	△	
14	MIB information acquisition processing	Section 6.2.14	△	

Remark. ◎: Required, ○: Recommended, △: Optional

Table 6.4 List of Sample Flowcharts Related to Transient Transmission (R_IN32M3_Transient.c File) (1/2)

No.	Overview	Reference	Implementation Required	Remarks
1	Transient1, Transient2, and TransientAck receive processing	Section 6.2.15	◎	
2	Transient2 request frame creation processing	Section 6.2.16	△	Required when the own station becomes a client of Transient2.
3	Transient1, Transient2, and TransientAck send processing	Section 6.2.17	◎	
4	Transient1 receive data processing	Section 6.2.18	◎	
5	Transient1 receive data reconstruction start processing	Section 6.2.19	◎	
6	Transient1 receive data reconstruction processing	Section 6.2.20	◎	
7	Node information distribution frame receive processing	Section 6.2.21	△	Required when the own station becomes a client of Transient2 or 8 SLMP.
8	Node information distribution frame check processing	Section 6.2.22	△	
9	Statistical information acquisition request frame receive processing	Section 6.2.23	△	
10	Statistical information acquisition response frame creation processing	Section 6.2.24	△	
11	Detailed node information acquisition request frame receive processing	Section 6.2.25	◎	
12	Detailed node information acquisition response frame creation processing	Section 6.2.26	◎	
13	Option information acquisition request frame receive processing	Section 6.2.27	○	Required to support extension functions.Note
14	Selected station information acquisition request frame receive processing	Section 6.2.28	○	Required to support the CC-Link IE Field Network diagnostic function.
15	Communication test request frame receive processing	Section 6.2.29	○	
16	Cable test request frame receive processing	Section 6.2.30	○	
17	Transient2 receive data processing	Section 6.2.31	△	Required when the own station becomes a server or a client of Transient2.
18	Transient2 receive data check processing	Section 6.2.32	△	
19	TransientAck receive data processing	Section 6.2.33	◎	
20	TransientAck frame creation processing	Section 6.2.34	◎	
21	Transient2 response frame creation processing	Section 6.2.35	△	Required when the own station becomes a server of Transient2.
22	Transient2 memory read request frame creation processing	Section 6.2.36	△	Required when the own station becomes a client of Transient2 memory read.

Table 6.4 List of Sample Flowcharts Related to Transient Transmission (R_IN32M3_Transient.c File) (2/2)

No.	Overview	Reference	Implementation Required	Remarks
23	Transient2 memory write request receive processing	Section 6.2.37	△	Required when the own station becomes a server of Transient2 memory write.
24	Transient2 memory read response receive processing	Section 6.2.38	△	Required when the own station becomes a client of Transient2 memory read.
25	SLMP memory read request frame receive processing	Section 6.2.39	△	Required when the own station becomes a server of SLMP memory read.
26	SLMP memory write request frame receive processing	Section 6.2.40	△	Required when the own station becomes a server of SLMP memory write.
27	SLMP memory read request frame creation processing	Section 6.2.41	△	Required when the own station becomes a client of SLMP memory read.
28	Transient1 request send division determination processing	Section 6.2.42	△	Required to send SLMP request frame of 1518 bytes or more.
29	Transient1 request frame creation processing	Section 6.2.43	△	Required when the own station becomes a client of SLMP.
30	SLMP memory read response receive processing	Section 6.2.44	△	Required when the own station becomes a client of SLMP memory read.

Remark. ◎: Required, ○: Recommended, △: Optional

Note. Extended functions of CC-Link IE Field Network including the SLMP frame send/receive function and CC-Link IE Field Network diagnostic function.

Caution. The R_IN32M3_Transient.c file describes Transient2 memory read/write and SLMP memory read/write as sample processing of each command.
If you want to implement commands other than the above, add the processing for each command while referring to Section 5.2 "Transient Transmission Overview" and the relevant manual "SLMP Reference Manual" (BAP-C3002-001).

Table 6.5 List of Sample Flowcharts Related to Hardware Test (R_IN32M3_HWTest.c File)

No.	Overview	Reference	Implementation Required	Remarks
1	Hardware test (IEEE 802.3ab compliance test)	Section 6.2.45	◎	
2	Hardware test (loopback communication test)	Section 6.2.46	○	

Remark. ◎: Required, ○: Recommended, △: Optional

6.2.1 Main processing

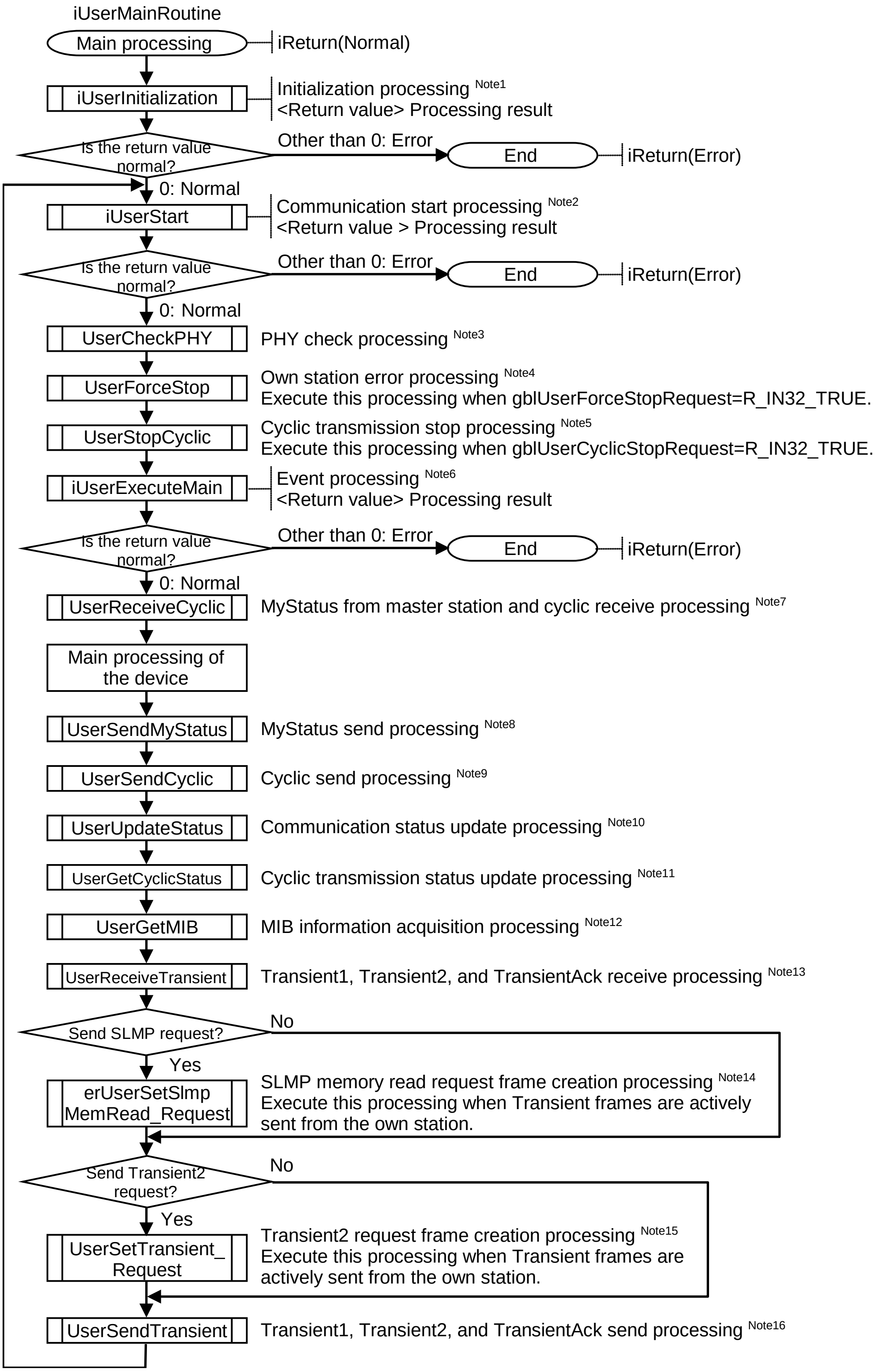

iUserMainRoutine
Main processing
iReturn(Normal)
iUserInitialization
Initialization processing Note1
<Return value> Processing result
Is the return value normal?
Other than 0: Error
End
iReturn(Error)
0: Normal
iUserStart
Communication start processing Note2
<Return value > Processing result
Is the return value normal?
Other than 0: Error
End
iReturn(Error)
0: Normal
UserCheckPHY
PHY check processing Note3
UserForceStop
Own station error processing Note4
Execute this processing when gblUserForceStopRequest=R_IN32_TRUE.
UserStopCyclic
Cyclic transmission stop processing Note5
Execute this processing when gblUserCyclicStopRequest=R_IN32_TRUE.
iUserExecuteMain
Event processing Note6
<Return value> Processing result
Is the return value normal?
Other than 0: Error
End
iReturn(Error)
0: Normal
UserReceiveCyclic
MyStatus from master station and cyclic receive processing Note7
Main processing of the device
UserSendMyStatus
MyStatus send processing Note8
UserSendCyclic
Cyclic send processing Note9
UserUpdateStatus
Communication status update processing Note10
UserGetCyclicStatus
Cyclic transmission status update processing Note11
UserGetMIB
MIB information acquisition processing Note12
UserReceiveTransient
Transient1, Transient2, and TransientAck receive processing Note13
Send SLMP request?
No
Yes
erUserSetSlmp MemRead_Request
SLMP memory read request frame creation processing Note14
Execute this processing when Transient frames are actively sent from the own station.
Send Transient2 request?
No
Yes
UserSetTransient_ Request
Transient2 request frame creation processing Note15
Execute this processing when Transient frames are actively sent from the own station.
UserSendTransient
Transient1, Transient2, and TransientAck send processing Note16

Figure 6.3 Flowchart for Main Processing

Note 1. For details, refer to Section 6.2.2 "Initialization processing".

2. For details, refer to Section 6.2.3 "Communication start processing".

3. For details, refer to Section 6.2.4 "PHY check processing".

4. For details, refer to Section 6.2.6 "Own station error processing".

5. For details, refer to Section 6.2.7 "Cyclic transmission stop processing".

6. For details, refer to Section 6.2.8 "Event processing".

7. For details, refer to Section 6.2.9 "MyStatus from master station and cyclic receive processing".

8. For details, refer to Section 6.2.10 "MyStatus send processing".

9. For details, refer to Section 6.2.11 "Cyclic send processing".

10. For details, refer to Section 6.2.12 "Communication status update processing".

11. For details, refer to Section 6.2.13 "Cyclic transmission status update processing".

12. For details, refer to Section 6.2.14 "MIB information acquisition processing".

13. For details, refer to Section 6.2.15 "Transient1, Transient2, and TransientAck receive processing".

14. For details, refer to Section 6.2.41 "SLMP memory read request frame creation processing".

15. For details, refer to Section 6.2.16 "Transient2 request frame creation processing".

16. For details, refer to Section 6.2.17 "Transient1, Transient2, and TransientAck send processing".

6.2.2 Initialization processing

This function initializes R-IN32M3-CL, enables and disables the R-IN32M3-CL internal WDT, and sets the node number and network number.

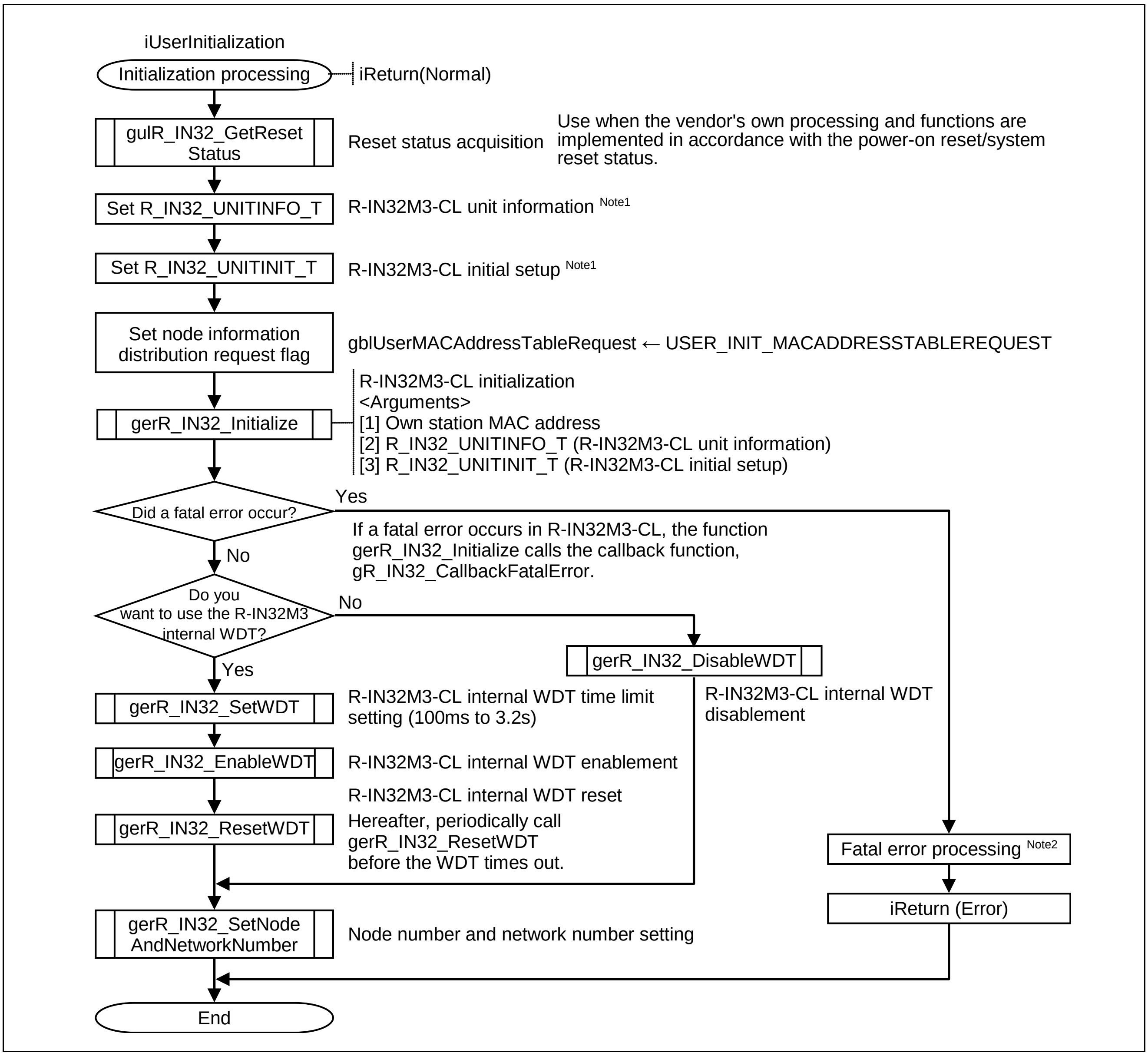

iUserInitialization
Initialization processing
iReturn(Normal)
gulR_IN32_GetReset Status
Reset status acquisition
Use when the vendor's own processing and functions are implemented in accordance with the power-on reset/system reset status.
Set R_IN32_UNITINFO_T
R-IN32M3-CL unit information Note1
Set R_IN32_UNITINIT_T
R-IN32M3-CL initial setup Note1
Set node information distribution request flag
gblUserMACAddressTableRequest ← USER_INIT_MACADDRESSTABLEREQUEST
gerR_IN32_Initialize
R-IN32M3-CL initialization
<Arguments>
[1] Own station MAC address
[2] R_IN32_UNITINFO_T (R-IN32M3-CL unit information)
[3] R_IN32_UNITINIT_T (R-IN32M3-CL initial setup)
Did a fatal error occur?
Yes
No
If a fatal error occurs in R-IN32M3-CL, the function gerR_IN32_Initialize calls the callback function, gR_IN32_CallbackFatalError.
Do you want to use the R-IN32M3 internal WDT?
No
Yes
gerR_IN32_DisableWDT
R-IN32M3-CL internal WDT disablement
gerR_IN32_SetWDT
R-IN32M3-CL internal WDT time limit setting (100ms to 3.2s)
gerR_IN32_EnableWDT
R-IN32M3-CL internal WDT enablement
gerR_IN32_ResetWDT
R-IN32M3-CL internal WDT reset
Hereafter, periodically call gerR_IN32_ResetWDT before the WDT times out.
Fatal error processing Note2
iReturn (Error)
gerR_IN32_SetNode AndNetworkNumber
Node number and network number setting
End

Figure 6.4 Flowchart for Initialization Processing

Note 1. For details, refer to Section 6.4.1(2) "gerR_IN32_Initialize".

2. For example, add processing such as calling UserForceStop (Own station error processing), and setting the own station to bypass mode.

Caution. [gblUserMACAddressTableRequest]

"gblUserMACAddressTableRequest" (node information distribution request flag) is used to determine whether or not the own station receives Node information distribution frame is to be received.

- **When own station wants to receive node information (when own station wants to send a transient request)**
  Set both "blMACAddressTableRequest" (initial value of node information distribution request) and "gblUserMACAddressTableRequest" (node information distribution request flag) of R_IN32_UNITINIT_T to "R_IN32_TRUE".
- **When own station does not want to receive node information (when own station does not want to send a transient request)**
  Set both "blMACAddressTableRequest" (initial value of node information distribution request) and "gblUserMACAddressTableRequest" (node information distribution request flag) of R_IN32_UNITINIT_T to "R_IN32_FALSE".

6.2.3 Communication start processing

This function instructs R-IN32M3-CL to start communication.

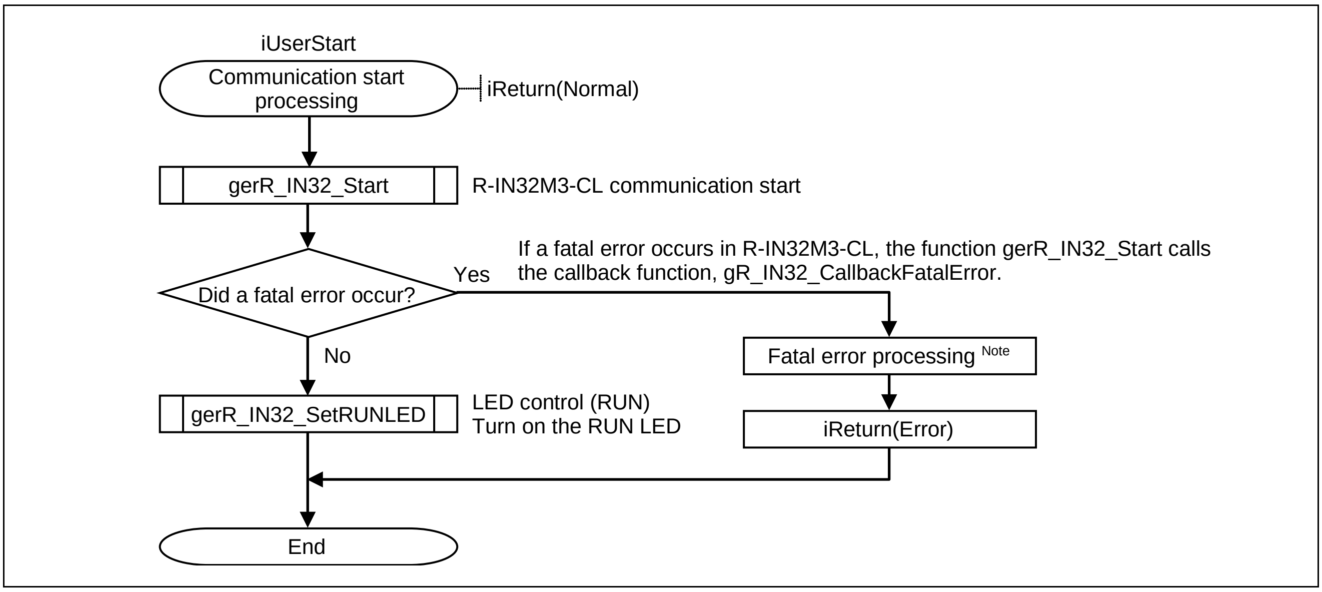

iUserStart
Communication start processing
iReturn(Normal)
gerR_IN32_Start
R-IN32M3-CL communication start
If a fatal error occurs in R-IN32M3-CL, the function gerR_IN32_Start calls the callback function, gR_IN32_CallbackFatalError.
Did a fatal error occur?
Yes
No
Fatal error processing Note
gerR_IN32_SetRUNLED
LED control (RUN)
Turn on the RUN LED
iReturn(Error)
End

Figure 6.5 Flowchart for Communication Start Processing

Note. For example, add processing such as calling UserForceStop (Own station error processing), and setting the own station to bypass mode.

6.2.4 PHY check processing

R-IN32M3-CL requires 1-Gbps/full-duplex linkup. This function checks if PHY is linked under settings other than 1-Gbps/full duplex.

If PHY is linked under settings other than 1-Gbps/full duplex, this function changes the PHY setting.

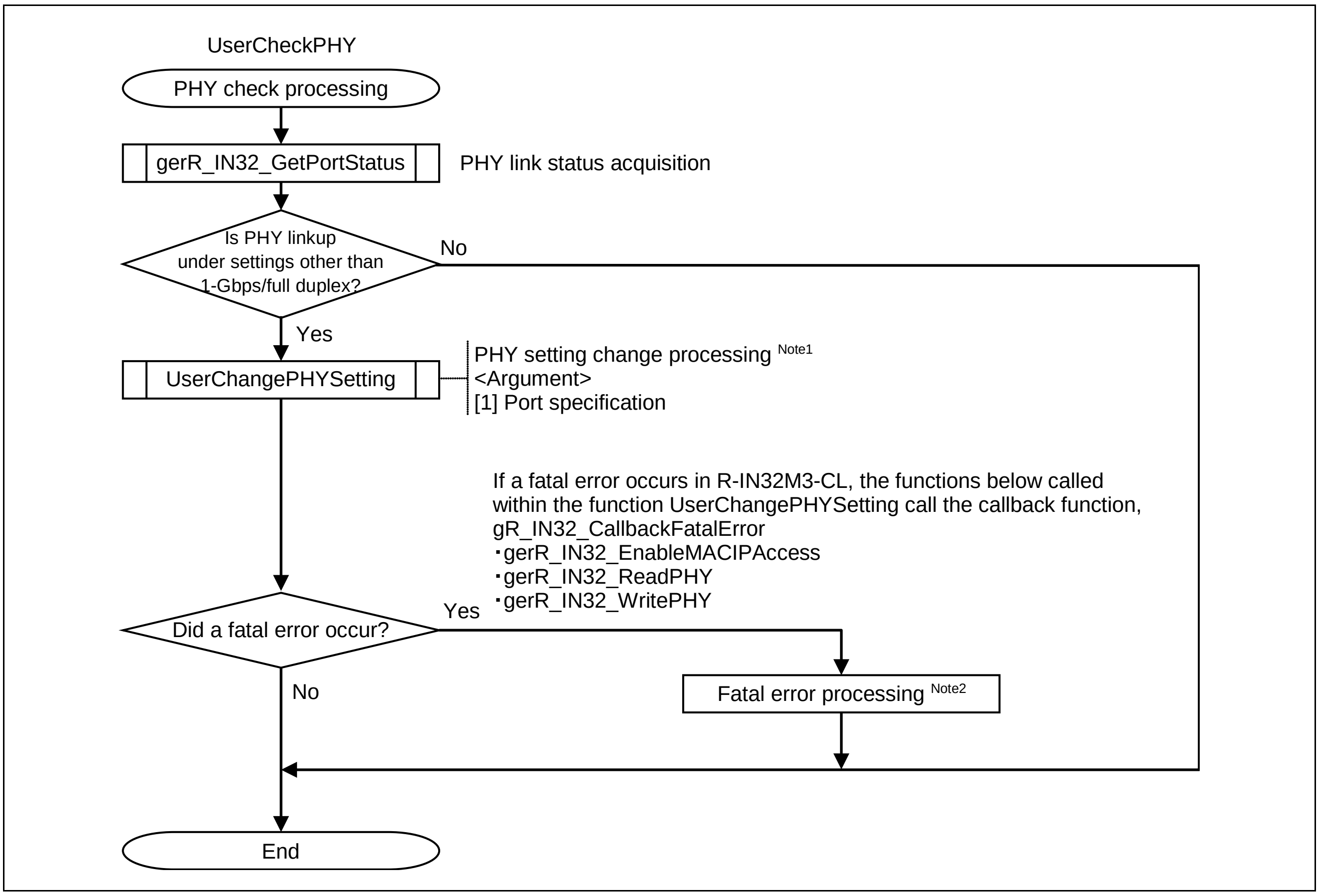

UserCheckPHY
PHY check processing
gerR_IN32_GetPortStatus
PHY link status acquisition
Is PHY linkup under settings other than 1-Gbps/full duplex?
No
Yes
UserChangePHYSetting
PHY setting change processing Note1
<Argument>
[1] Port specification
If a fatal error occurs in R-IN32M3-CL, the functions below called within the function UserChangePHYSetting call the callback function, gR_IN32_CallbackFatalError
･gerR_IN32_EnableMACIPAccess
･gerR_IN32_ReadPHY
･gerR_IN32_WritePHY
Yes
Did a fatal error occur?
No
Fatal error processing Note2
End

Figure 6.6 Flowchart for PHY Check Processing

Note 1. For details, refer to Section 6.2.5 "PHY setting change processing".

2. For example, add processing such as calling UserForceStop (Own station error processing), and setting the own station to bypass mode.

Caution. Implement the above processing on both port 1 and port 2.
Implementation is not required if the PHY used permits linkup fixed to 1-Gbps/full duplex according to hardware settings.

6.2.5 PHY setting change processing

This function sets PHY so that it only permits linkup under 1-Gbps/full duplex settings.

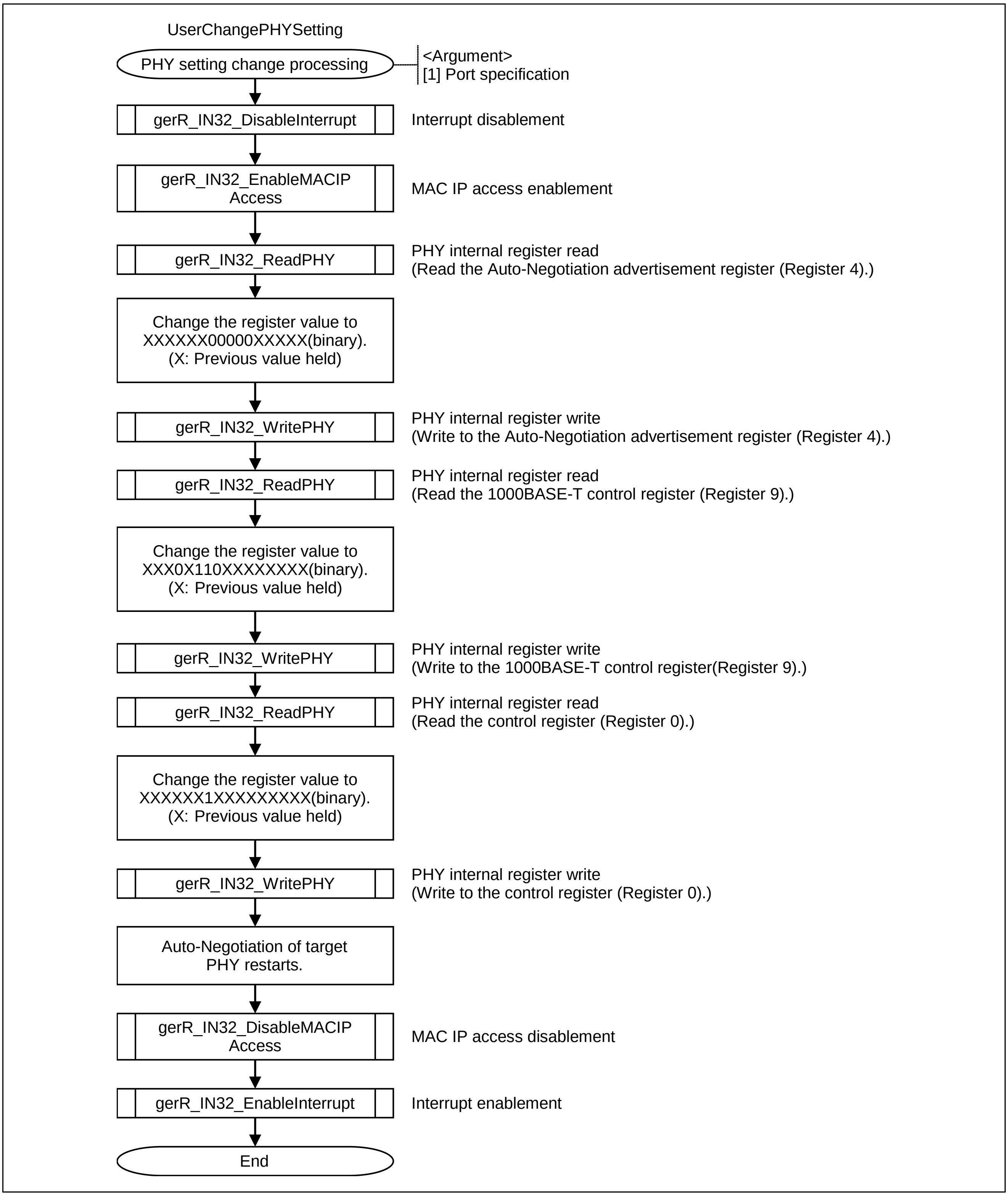

UserChangePHYSetting
PHY setting change processing
<Argument>
[1] Port specification
gerR_IN32_DisableInterrupt
Interrupt disablement
gerR_IN32_EnableMACIP Access
MAC IP access enablement
gerR_IN32_ReadPHY
PHY internal register read
(Read the Auto-Negotiation advertisement register (Register 4).)
Change the register value to XXXXXX00000XXXXX(binary). (X: Previous value held)
gerR_IN32_WritePHY
PHY internal register write
(Write to the Auto-Negotiation advertisement register (Register 4).)
gerR_IN32_ReadPHY
PHY internal register read
(Read the 1000BASE-T control register (Register 9).)
Change the register value to XXX0X110XXXXXXXX(binary). (X: Previous value held)
gerR_IN32_WritePHY
PHY internal register write
(Write to the 1000BASE-T control register(Register 9).)
gerR_IN32_ReadPHY
PHY internal register read
(Read the control register (Register 0).)
Change the register value to XXXXXX1XXXXXXXXX(binary). (X: Previous value held)
gerR_IN32_WritePHY
PHY internal register write
(Write to the control register (Register 0).)
Auto-Negotiation of target PHY restarts.
gerR_IN32_DisableMACIP Access
MAC IP access disablement
gerR_IN32_EnableInterrupt
Interrupt enablement
End

Figure 6.7 Flowchart for PHY Setting Change Processing

6.2.6 Own station error processing

This function changes the state of the own station to an error when a vendor-defined error occurs. (This processing is optional.)

When an error occurs on the own station, R-IN32M3-CL changes to bypass mode. In bypass mode, communication frames that have entered the port are not received by R-IN32M3-CL but are forwarded as is to another port.

To clear the own station error, power-on reset or system reset is required.

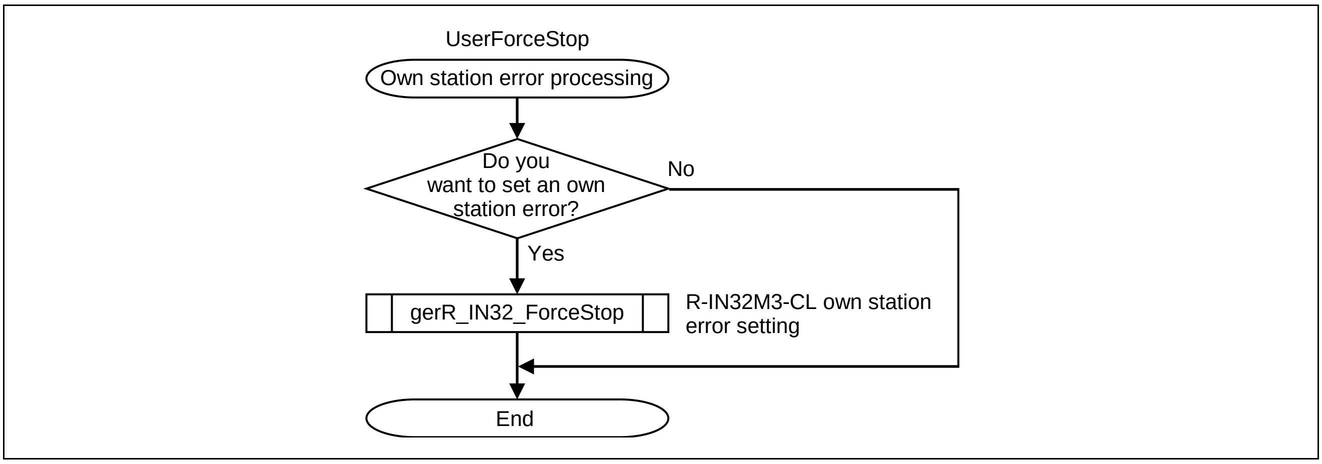

UserForceStop
Own station error processing
Do you want to set an own station error?
No
Yes
gerR_IN32_ForceStop
R-IN32M3-CL own station error setting
End

Figure 6.8 Flowchart for Own Station Error Processing

6.2.7 Cyclic transmission stop processing

This function controls the stop and restart of cyclic transmission for device-side reasons. (This processing is optional.)

Even if you stop cyclic transmission, transient transmission is possible. (Token passing continues.)

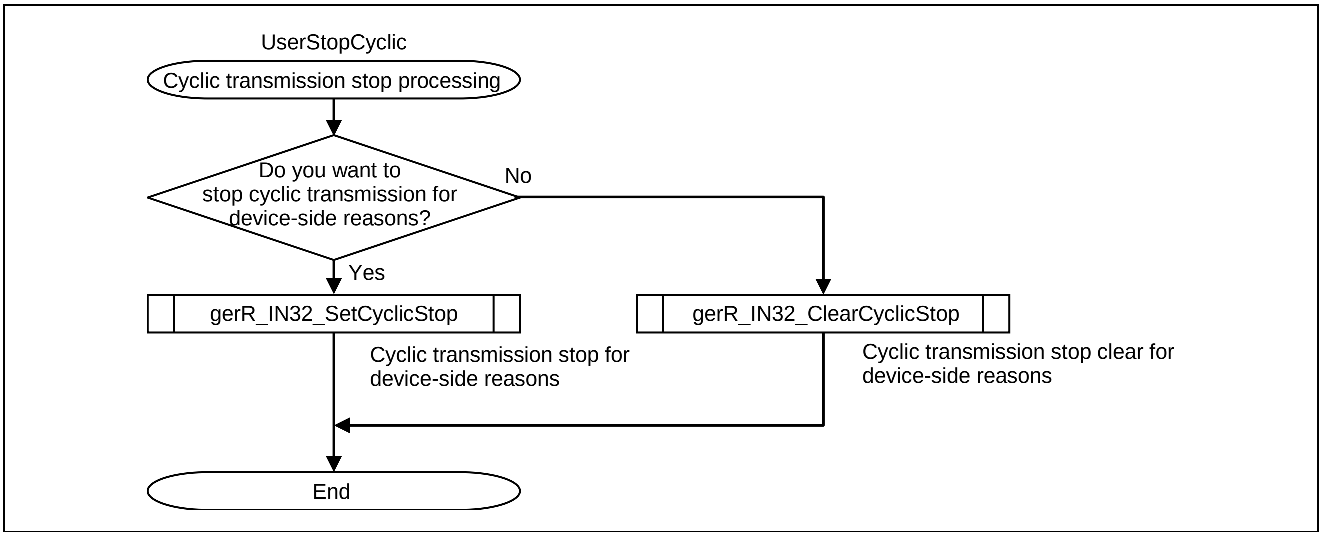

UserStopCyclic
Cyclic transmission stop processing
Do you want to stop cyclic transmission for device-side reasons?
No
Yes
gerR_IN32_SetCyclicStop
gerR_IN32_ClearCyclicStop
Cyclic transmission stop for device-side reasons
Cyclic transmission stop clear for device-side reasons
End

Figure 6.9 Flowchart for Cyclic Transmission Stop Processing

6.2.8 Event processing

This function detects MPU interrupts (R-IN32M3-CL events), processes the events, and updates MIB information.

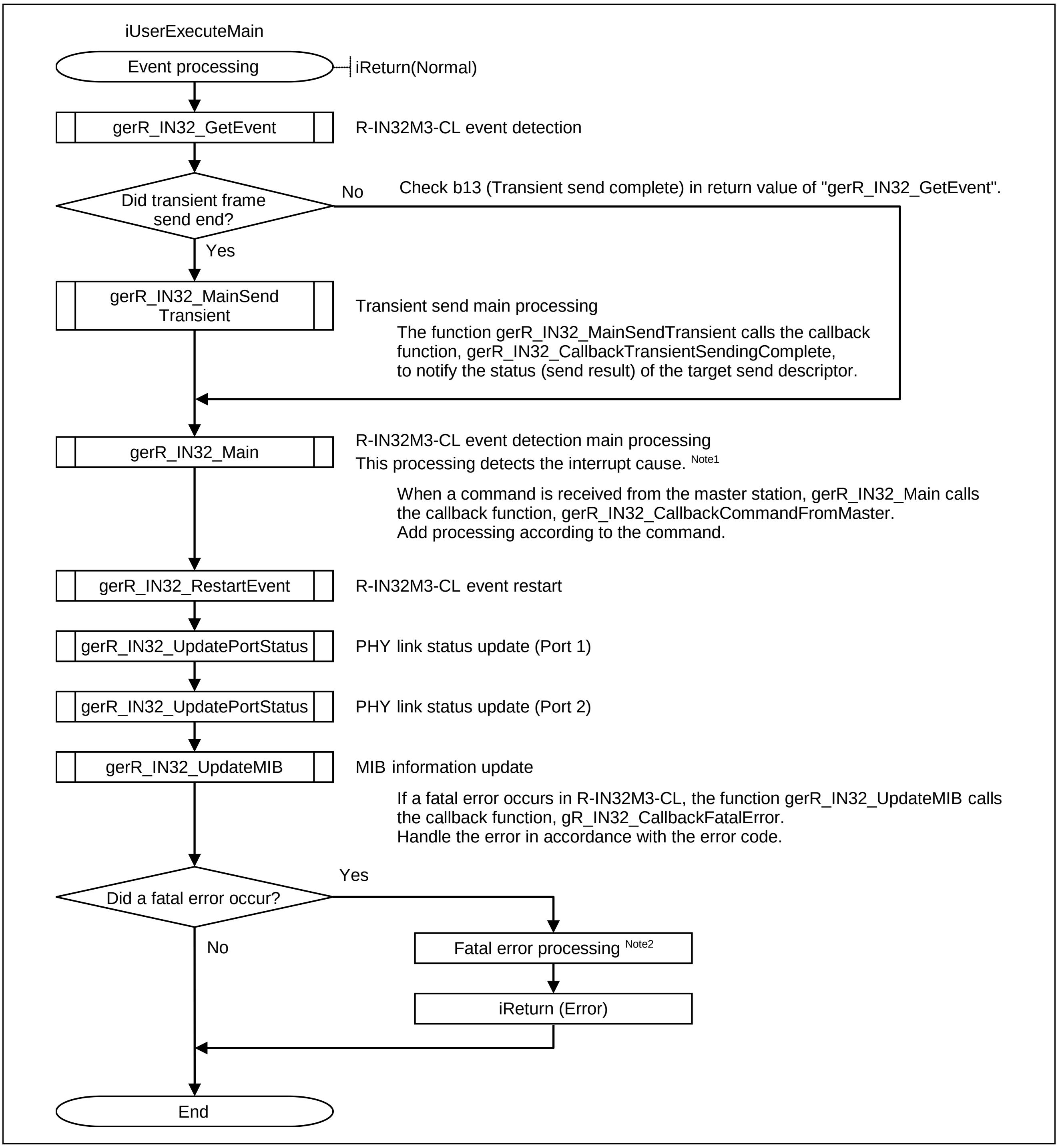

iUserExecuteMain
Event processing
iReturn(Normal)
gerR_IN32_GetEvent
R-IN32M3-CL event detection
Did transient frame send end?
No
Check b13 (Transient send complete) in return value of "gerR_IN32_GetEvent".
Yes
gerR_IN32_MainSend Transient
Transient send main processing
The function gerR_IN32_MainSendTransient calls the callback function, gerR_IN32_CallbackTransientSendingComplete, to notify the status (send result) of the target send descriptor.
gerR_IN32_Main
R-IN32M3-CL event detection main processing
This processing detects the interrupt cause. Note1
When a command is received from the master station, gerR_IN32_Main calls the callback function, gerR_IN32_CallbackCommandFromMaster.
Add processing according to the command.
gerR_IN32_RestartEvent
R-IN32M3-CL event restart
gerR_IN32_UpdatePortStatus
PHY link status update (Port 1)
gerR_IN32_UpdatePortStatus
PHY link status update (Port 2)
gerR_IN32_UpdateMIB
MIB information update
If a fatal error occurs in R-IN32M3-CL, the function gerR_IN32_UpdateMIB calls the callback function, gR_IN32_CallbackFatalError.
Handle the error in accordance with the error code.
Did a fatal error occur?
Yes
No
Fatal error processing Note2
iReturn (Error)
End

Figure 6.10 Flowchart for Event Processing

Note 1. For details, refer to Section 6.4.3 "Event".

2. For example, add processing such as calling UserForceStop (Own station error processing), and setting the own station to bypass mode.

6.2.9 MyStatus from master station and cyclic receive processing

This function acquires the status of the master station from the received MyStatus frame and acquires cyclic data (RY, RWw) from the received cyclic frame.

Perform "Hold/Clear processing" in accordance with the status of the master station that is acquired from the MyStatus frame (in accordance with whether the master station is stopped, an error occurred, or the like).

Hold/Clear processing is processing in which the developed device continues (Hold) or stops (Clear) output when the developed device controls external output and cyclic transmission has stopped for reasons such as a master station application stop/error, or data link disconnection.

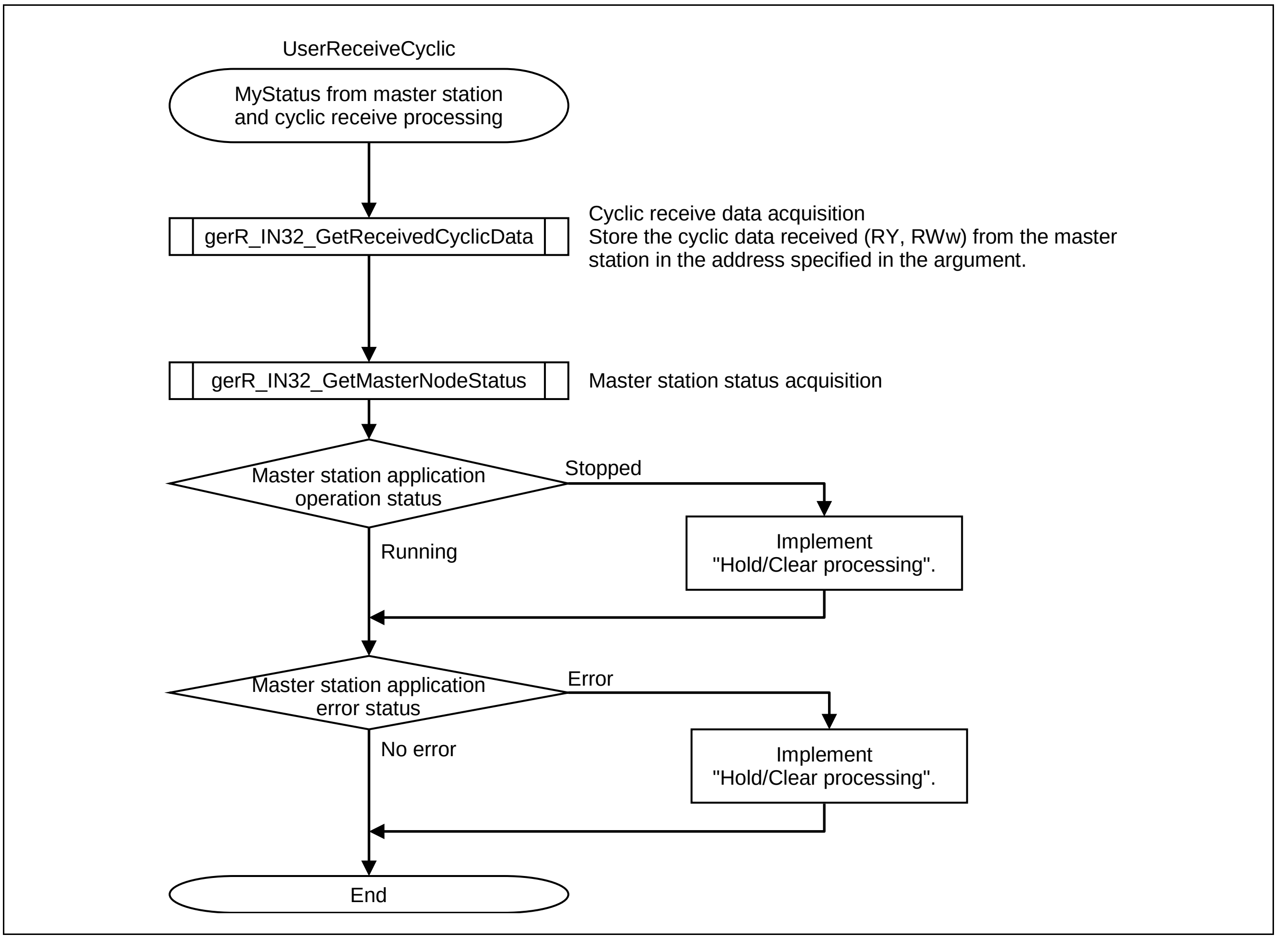

UserReceiveCyclic
MyStatus from master station and cyclic receive processing
gerR_IN32_GetReceivedCyclicData
Cyclic receive data acquisition
Store the cyclic data received (RY, RWw) from the master station in the address specified in the argument.
gerR_IN32_GetMasterNodeStatus
Master station status acquisition
Master station application operation status
Stopped
Running
Implement "Hold/Clear processing".
Master station application error status
Error
No error
Implement "Hold/Clear processing".
End

Figure 6.11 Flowchart for MyStatus from Master Station and Cyclic Receive Processing

Caution. Consider 1) and 2) below and implement the Hold/Clear processing as a fail-safe.

1) Cyclic data (RY, RWw) sent by the master station

In the case of a master station application stop/error, cyclic data that the master station sends is held or cleared depending on the master station setting. (When a Mitsubishi Electric master station is used, Hold/Clear processing is set in "output status setting for CPU module STOP" and "output status setting for CPU stop error".)

The slave station (own station) cannot previously detect whether cyclic data that the master station sends is held or cleared.

2) Cyclic data (RY, RWw) acquired by the R-IN32M3-CL driver depending on the master station application status

Cyclic data received in a slave station (own station) is acquired by the R-IN32M3-CL driver (gerR_IN32_GetReceivedCyclicData). Contents of acquired cyclic data differ depending on the operation/error status of the master station application.

Master station application		Cyclic data acquired by the R-IN32M3-CL driver
Operation status	Error status	
Running	No error	Cyclic data that the master station is "currently" sending
Stopped	No error	
Running[Note]	Error[Note]	Not acquired (At the address specified by the argument, cyclic data stored in point of time before an error occurs in the master station application remains.)
Stopped	Error	

Note. When a Mitsubishi Electric master station is used, the programmable controller CPU module cannot be in a state of "Operating" and "Error" at the same time.

6.2.10 MyStatus send processing

This function creates MyStatus frame. The set frame is automatically sent by R-IN32M3-CL.

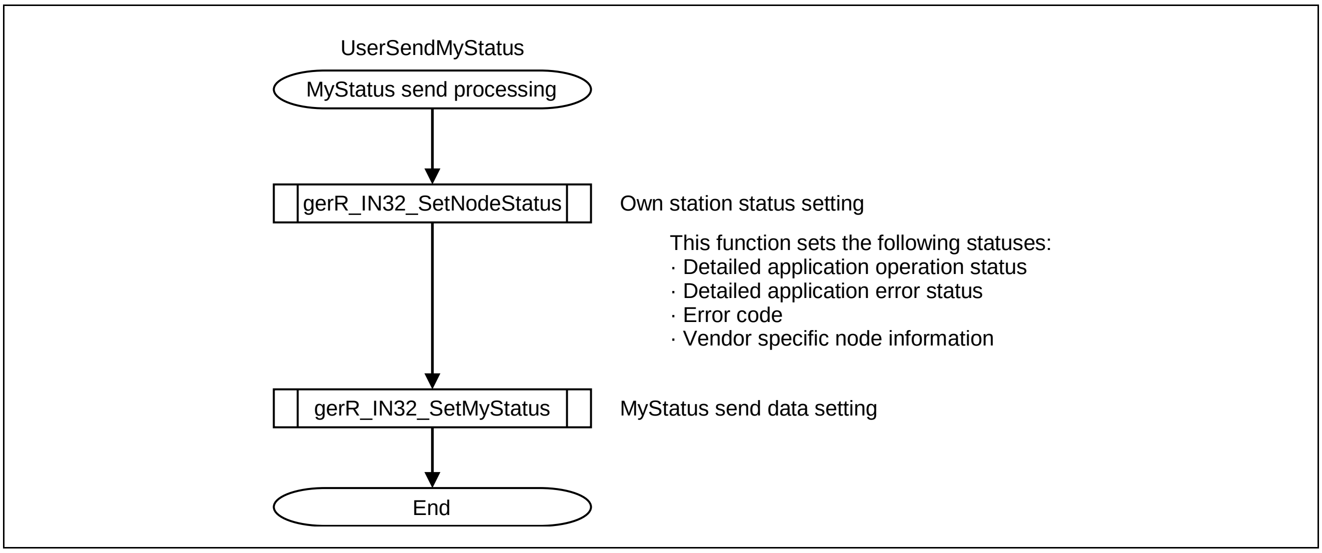

UserSendMyStatus
MyStatus send processing
gerR_IN32_SetNodeStatus
Own station status setting
This function sets the following statuses:
· Detailed application operation status
· Detailed application error status
· Error code
· Vendor specific node information
gerR_IN32_SetMyStatus
MyStatus send data setting
End

Figure 6.12 Flowchart for MyStatus Send Processing

6.2.11 Cyclic send processing

This function sends cyclic send data (RX and RWr).

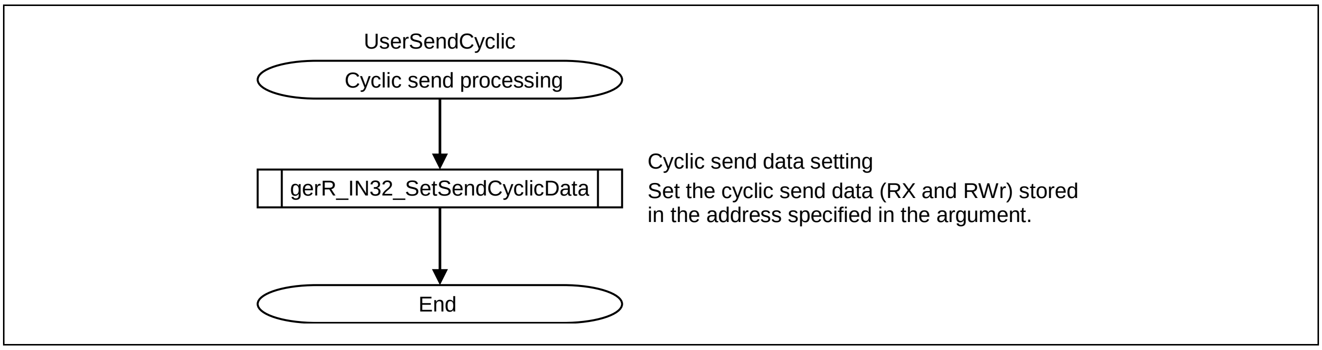

UserSendCyclic
Cyclic send processing
gerR_IN32_SetSendCyclicData
Cyclic send data setting
Set the cyclic send data (RX and RWr) stored
in the address specified in the argument.
End

Figure 6.13 Flowchart for Cyclic Send Processing

6.2.12 Communication status update processing

This function acquires the data link status of the own station, and controls the Hold/Clear processing and the on/off status of the D LINK LED and the ERR. LED in accordance with the data link status.

Hold/Clear processing is processing in which the developed device continues (Hold) or stops (Clear) output when the developed device controls external output and cyclic transmission has stopped for reasons such as a master station application stop/error, or data link disconnection.

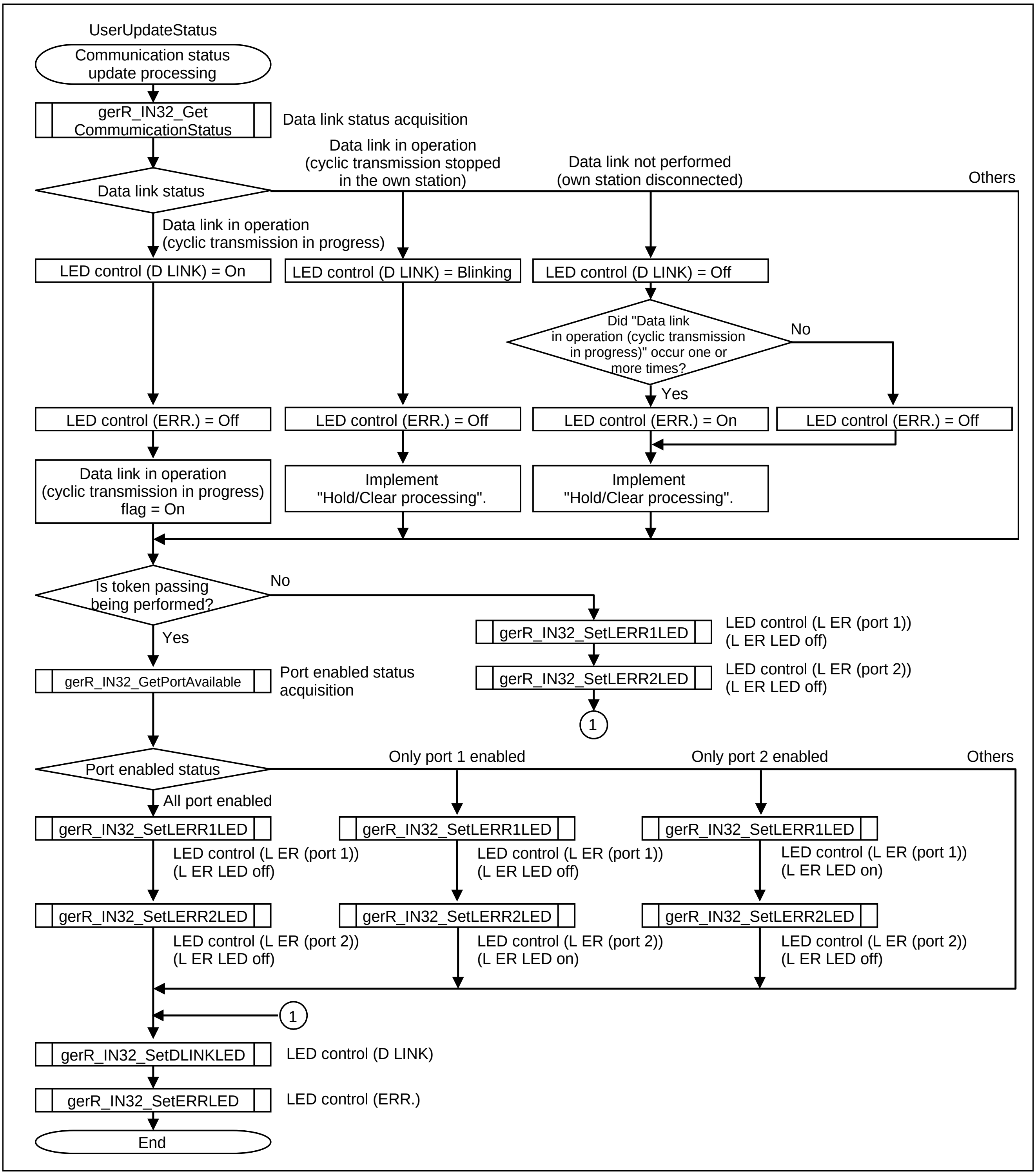

UserUpdateStatus
Communication status update processing
gerR_IN32_Get CommumicationStatus
Data link status acquisition
Data link status
Data link in operation (cyclic transmission stopped in the own station)
Data link not performed (own station disconnected)
Others
Data link in operation (cyclic transmission in progress)
LED control (D LINK) = On
LED control (D LINK) = Blinking
LED control (D LINK) = Off
Did "Data link in operation (cyclic transmission in progress)" occur one or more times?
No
Yes
LED control (ERR.) = Off
LED control (ERR.) = Off
LED control (ERR.) = On
LED control (ERR.) = Off
Data link in operation (cyclic transmission in progress) flag = On
Implement "Hold/Clear processing".
Implement "Hold/Clear processing".
Is token passing being performed?
No
Yes
gerR_IN32_SetLERR1LED
LED control (L ER (port 1)) (L ER LED off)
gerR_IN32_SetLERR2LED
LED control (L ER (port 2)) (L ER LED off)
1
gerR_IN32_GetPortAvailable
Port enabled status acquisition
Port enabled status
Only port 1 enabled
Only port 2 enabled
Others
All port enabled
gerR_IN32_SetLERR1LED
LED control (L ER (port 1)) (L ER LED off)
gerR_IN32_SetLERR1LED
LED control (L ER (port 1)) (L ER LED off)
gerR_IN32_SetLERR1LED
LED control (L ER (port 1)) (L ER LED on)
gerR_IN32_SetLERR2LED
LED control (L ER (port 2)) (L ER LED off)
gerR_IN32_SetLERR2LED
LED control (L ER (port 2)) (L ER LED on)
gerR_IN32_SetLERR2LED
LED control (L ER (port 2)) (L ER LED off)
1
gerR_IN32_SetDLINKLED
LED control (D LINK)
gerR_IN32_SetERRLED
LED control (ERR.)
End

Figure 6.14 Flowchart for Communication Status Update Processing

Caution. Consider the following and implement the Hold/Clear processing as a fail-safe.

Cyclic data (RY, RWw) acquired by the R-IN32M3-CL driver depending on the data link status

Cyclic data received in a slave station (own station) is acquired by the R-IN32M3-CL driver (gerR_IN32_GetReceivedCyclicData). Contents of acquired cyclic data differ depending on data link status.

Data link status	Cyclic data acquired by the R-IN32M3-CL driver
Data link not performed (The own station is disconnected.)	Not acquired (At the address specified by the argument, cyclic data stored in point of time before the own station is disconnected remains.)
Data link in operation (Cyclic transmission is stopped in the own station.)[Note]	Cyclic data that the master station is "currently" sending

Note. The slave station receives RY, RWw and does not send RX, RWr.

6.2.13 Cyclic transmission status update processing

This function acquires the cyclic transmission size specified by the master station and the cyclic transmission status.

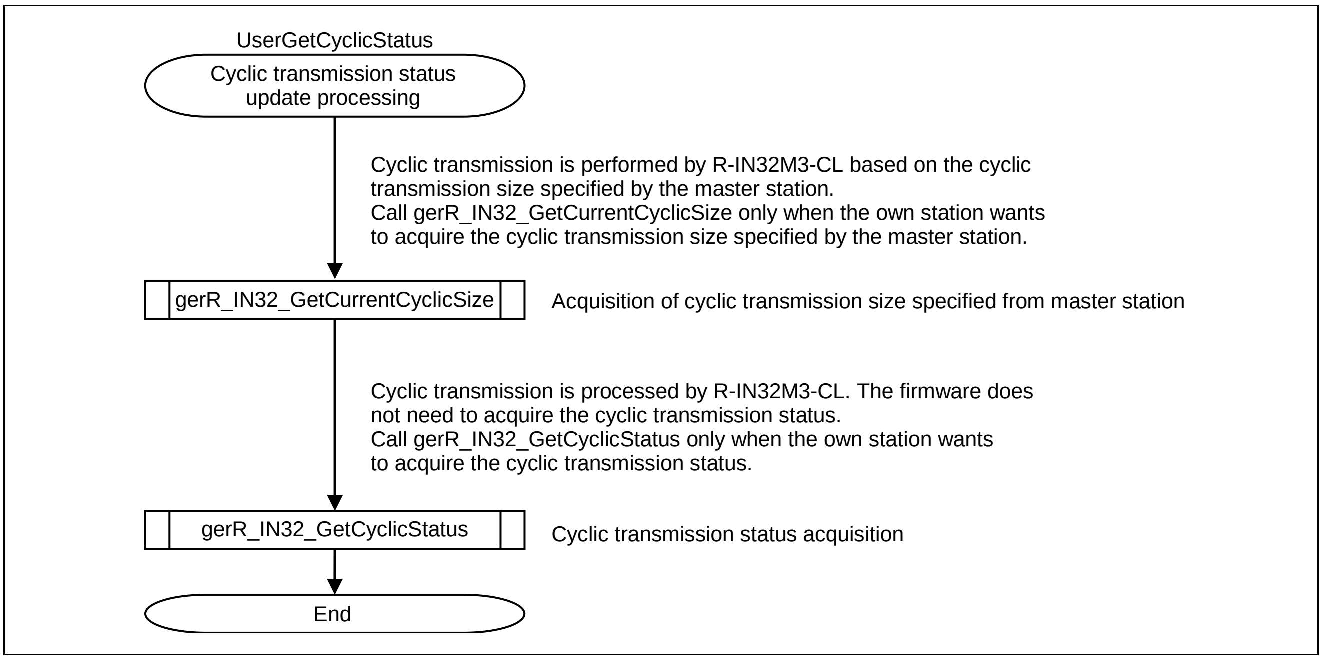

UserGetCyclicStatus
Cyclic transmission status update processing
Cyclic transmission is performed by R-IN32M3-CL based on the cyclic transmission size specified by the master station.
Call gerR_IN32_GetCurrentCyclicSize only when the own station wants to acquire the cyclic transmission size specified by the master station.
gerR_IN32_GetCurrentCyclicSize
Acquisition of cyclic transmission size specified from master station
Cyclic transmission is processed by R-IN32M3-CL. The firmware does not need to acquire the cyclic transmission status.
Call gerR_IN32_GetCyclicStatus only when the own station wants to acquire the cyclic transmission status.
gerR_IN32_GetCyclicStatus
Cyclic transmission status acquisition
End

Figure 6.15 Flowchart for Cyclic Transmission Status Update Processing

6.2.14 MIB information acquisition processing

This function acquires or clears MIB information.

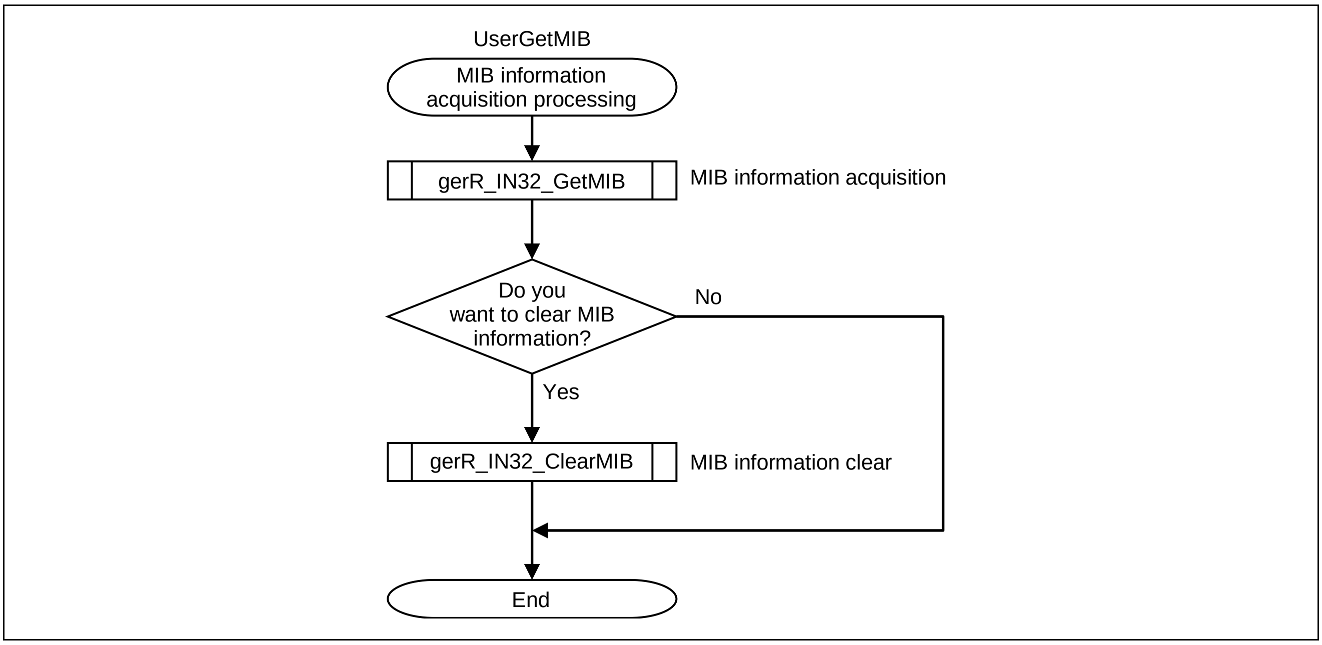

UserGetMIB
MIB information acquisition processing
gerR_IN32_GetMIB
MIB information acquisition
Do you want to clear MIB information?
No
Yes
gerR_IN32_ClearMIB
MIB information clear
End

Figure 6.16 Flowchart for MIB Information Acquisition Processing

Caution. MIB information is non-disclosed information. Do not disclose the information to the end user.

(1) List of MIB Information of Ring Control Area

Table 6.6 List of MIB Information of Ring Control Area

No.	MIB Information	Description
1	No. of HEC error frames	Counts the number of HEC errors in received frames.
2	No. of DCS/FCS error frames	Counts the number of DCS/FCS errors in received frames.
3	No. of undersize error frames	Counts the number of received error frames with a size less than 28 bytes.
4	No. of forwarded frames	Counts the number of forwarded frames.
5	No. of upper layer transmission frames	Counts the number of frames transmitted to upper layers.
6	No. of discarded frames due to full forward buffer	Counts the number of frames discarded due to a full forward buffer.
7	No. of discarded frames due to full upper layer transmission buffer	Counts the number of frames discarded due to a full upper layer transmission buffer.

(2) List of MIB Information of MAC IP Area

Table 6.7 List of MIB Information of MAC IP Area

No.	MIB Information	Description
1	No. of received frames	Counts all frame receptions, including error frames. Error frames: FCS error, undersized, oversized frames
2	No. of sent frames	Counts the number of sent frames.
3	No. of received undersized frames	Counts the number of received frames with a size less than 64 bytes.
4	No. of received oversized frames	Counts the number of received frames with a size exceeding 1518 bytes.
5	No. of received frame FCS errors	Counts the number of received frames with an FCS error.
6	No. of received frame fragment errors	Counts the number of received frames with fragment errors. Fragment error: A frame with less than 64 bytes and an FCS error
7	No. of frames detected within minimum IFG	Counts the number of frames detected within the minimum inter-frame gap (IFG).
8	No of received frames with SFD or less	Counts the number of received frames that ended at a field up to SFD and were not recognized as a valid frame.
9	No. of reception code errors	Counts the number of GMII reception data errors detected (RECV_*_ERR=1[Note]). Counts a RECV_*_ERR[Note] that occurred multiple times in an idle state (RECV_*_DV=1[Note]) as one error. Note: The asterisk ("*") indicates a wild character. (A: Port 1, B: Port 2)
10	No. of received invalid carrier errors	Counts the number of invalid carriers that occurred in an idle state. Counts multiple invalid carriers that occurred in an idle state as one error.
11	No. of received carrier extension errors	Counts the number of carrier extensions that occurred in an idle state. Counts multiple carrier extensions that occurred in an idle state as one error.

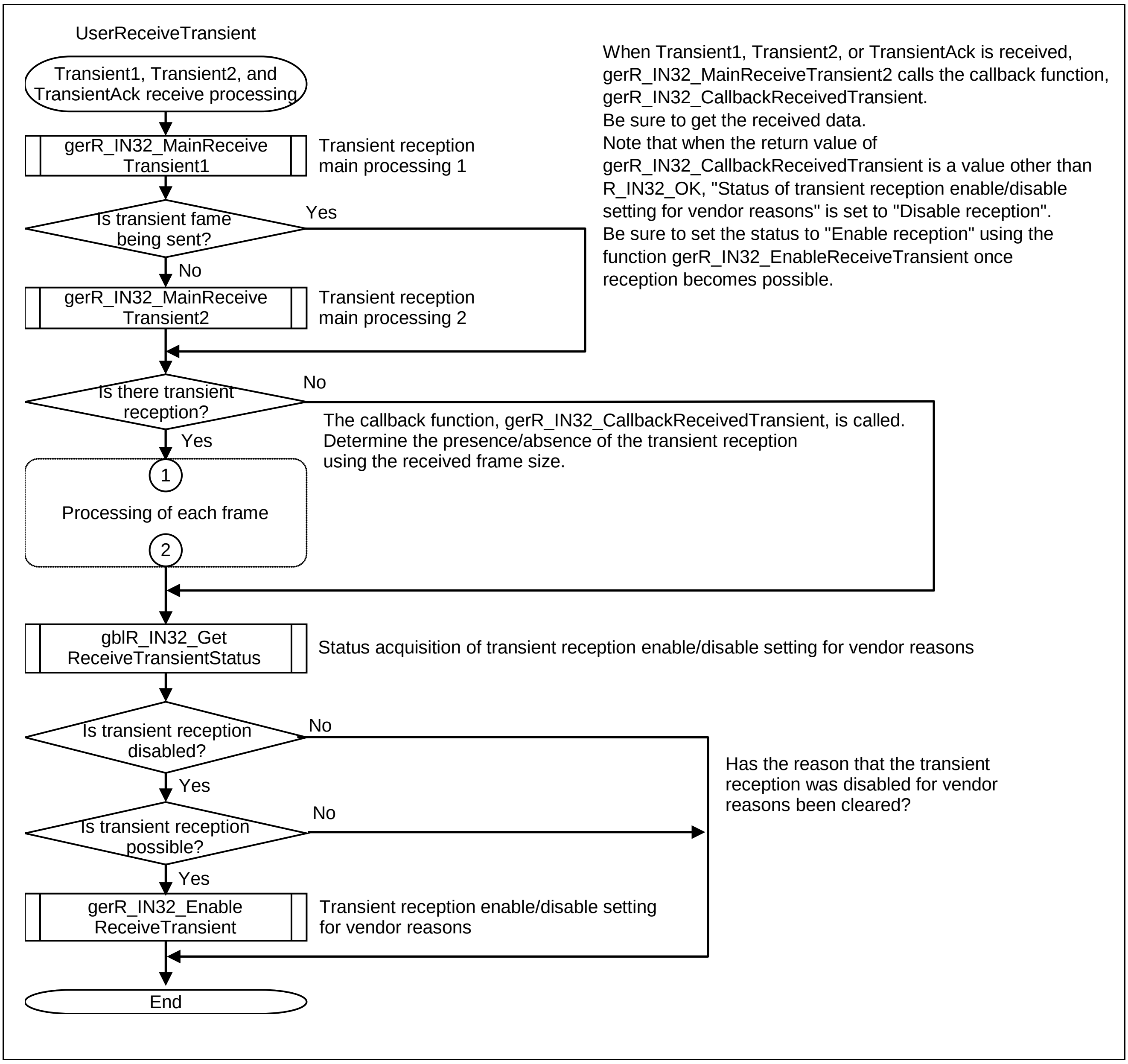
(3) List of Other MIB Information

Table 6.8 List of Other MIB Information

No.	MIB Information	Description
1	No. of link downs (port 1)	Counts the number of link downs of port 1.
2	No. of link downs (port 2)	Counts the number of link downs of port 2.
3	No. of master watch timer errors	Counts the number of timeouts of the master watch timer.
4	No. of received cyclic frames	Counts the number of cyclic frames received by R-IN32M3-CL.
5	No. of received transient frames	Counts the number of transient frames received by R-IN32M3-CL.
6	No. of received transient frames discarded	Counts the number of received transient frames discarded by R-IN32M3-CL.

6.2.15 Transient1, Transient2, and TransientAck receive processing

This function receives Transient1, Transient2, and TransientAck frames and processes the data.


UserReceiveTransient
Transient1, Transient2, and TransientAck receive processing
gerR_IN32_MainReceiveTransient1
Transient reception main processing 1
Is transient fame being sent?
Yes
No
gerR_IN32_MainReceiveTransient2
Transient reception main processing 2
Is there transient reception?
No
Yes
1
Processing of each frame
2
gblR_IN32_GetReceiveTransientStatus
Status acquisition of transient reception enable/disable setting for vendor reasons
Is transient reception disabled?
No
Yes
Is transient reception possible?
No
Yes
gerR_IN32_EnableReceiveTransient
Transient reception enable/disable setting for vendor reasons
Has the reason that the transient reception was disabled for vendor reasons been cleared?
End

When Transient1, Transient2, or TransientAck is received, gerR_IN32_MainReceiveTransient2 calls the callback function, gerR_IN32_CallbackReceivedTransient.
Be sure to get the received data.
Note that when the return value of gerR_IN32_CallbackReceivedTransient is a value other than R_IN32_OK, "Status of transient reception enable/disable setting for vendor reasons" is set to "Disable reception".
Be sure to set the status to "Enable reception" using the function gerR_IN32_EnableReceiveTransient once reception becomes possible.

The callback function, gerR_IN32_CallbackReceivedTransient, is called.
Determine the presence/absence of the transient reception using the received frame size.

Figure 6.17 Flowchart for Transient1, Transient2, and TransientAck Receive Processing (1/2)

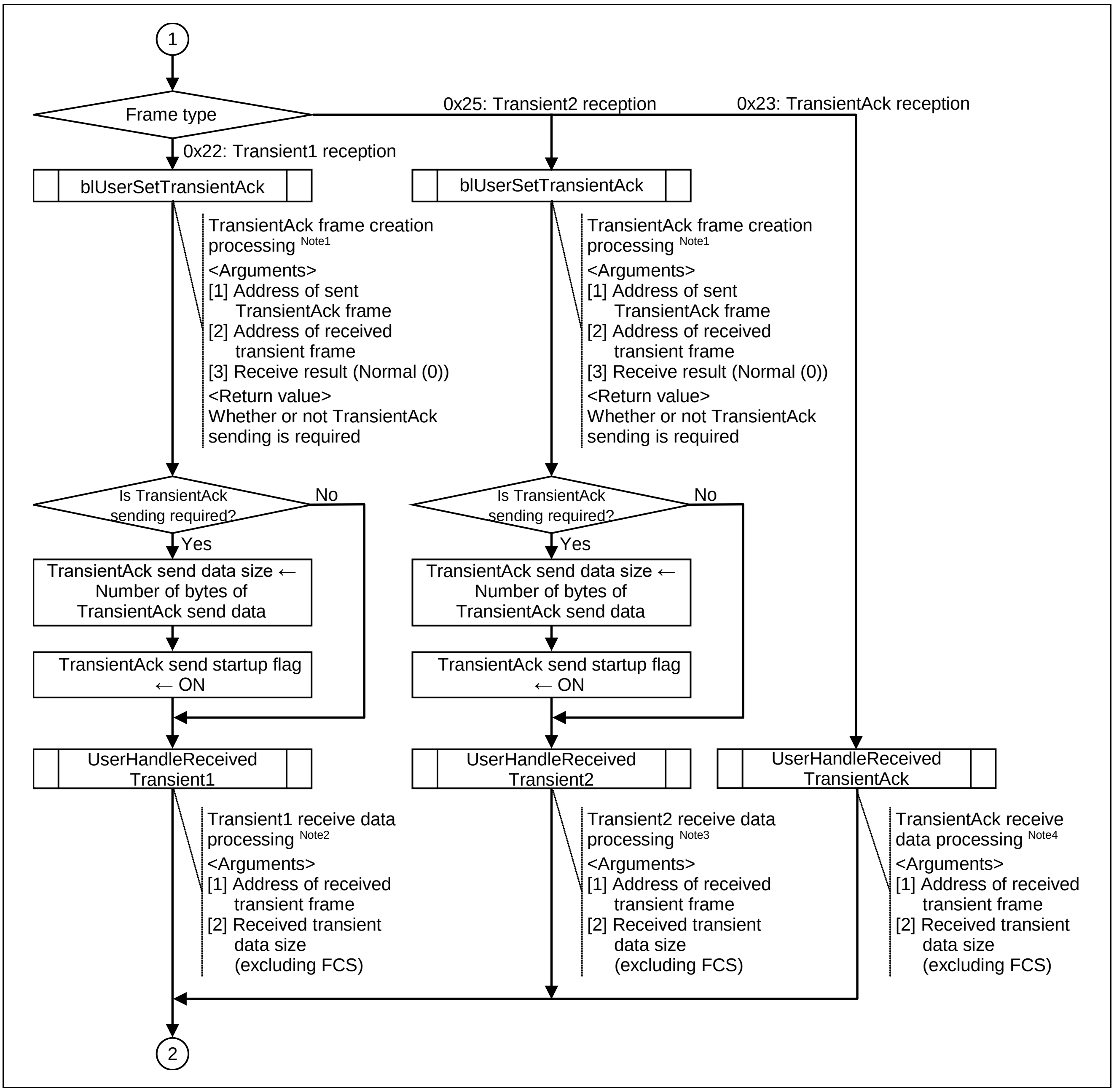

1
Frame type
0x25: Transient2 reception
0x23: TransientAck reception
0x22: Transient1 reception
blUserSetTransientAck
blUserSetTransientAck
TransientAck frame creation processing Note1
<Arguments>
[1] Address of sent TransientAck frame
[2] Address of received transient frame
[3] Receive result (Normal (0))
<Return value>
Whether or not TransientAck sending is required
TransientAck frame creation processing Note1
<Arguments>
[1] Address of sent TransientAck frame
[2] Address of received transient frame
[3] Receive result (Normal (0))
<Return value>
Whether or not TransientAck sending is required
Is TransientAck sending required?
No
Yes
Is TransientAck sending required?
No
Yes
TransientAck send data size ← Number of bytes of TransientAck send data
TransientAck send data size ← Number of bytes of TransientAck send data
TransientAck send startup flag ← ON
TransientAck send startup flag ← ON
UserHandleReceived Transient1
UserHandleReceived Transient2
UserHandleReceived TransientAck
Transient1 receive data processing Note2
<Arguments>
[1] Address of received transient frame
[2] Received transient data size (excluding FCS)
Transient2 receive data processing Note3
<Arguments>
[1] Address of received transient frame
[2] Received transient data size (excluding FCS)
TransientAck receive data processing Note4
<Arguments>
[1] Address of received transient frame
[2] Received transient data size (excluding FCS)
2

Figure 6.17 Flowchart for Transient1, Transient2, and TransientAck Receive Processing (2/2)

Note 1. For details, refer to Section 6.2.34 "TransientAck frame creation processing".

2. For details, refer to Section 6.2.18 "Transient1 receive data processing".

3. For details, refer to Section 6.2.31 "Transient2 receive data processing".

4. For details, refer to Section 6.2.33 "TransientAck receive data processing".

6.2.16 Transient2 request frame creation processing

This function creates Transient2 memory read request frame. This processing is an example of the processing for creating Transient2 request frame.

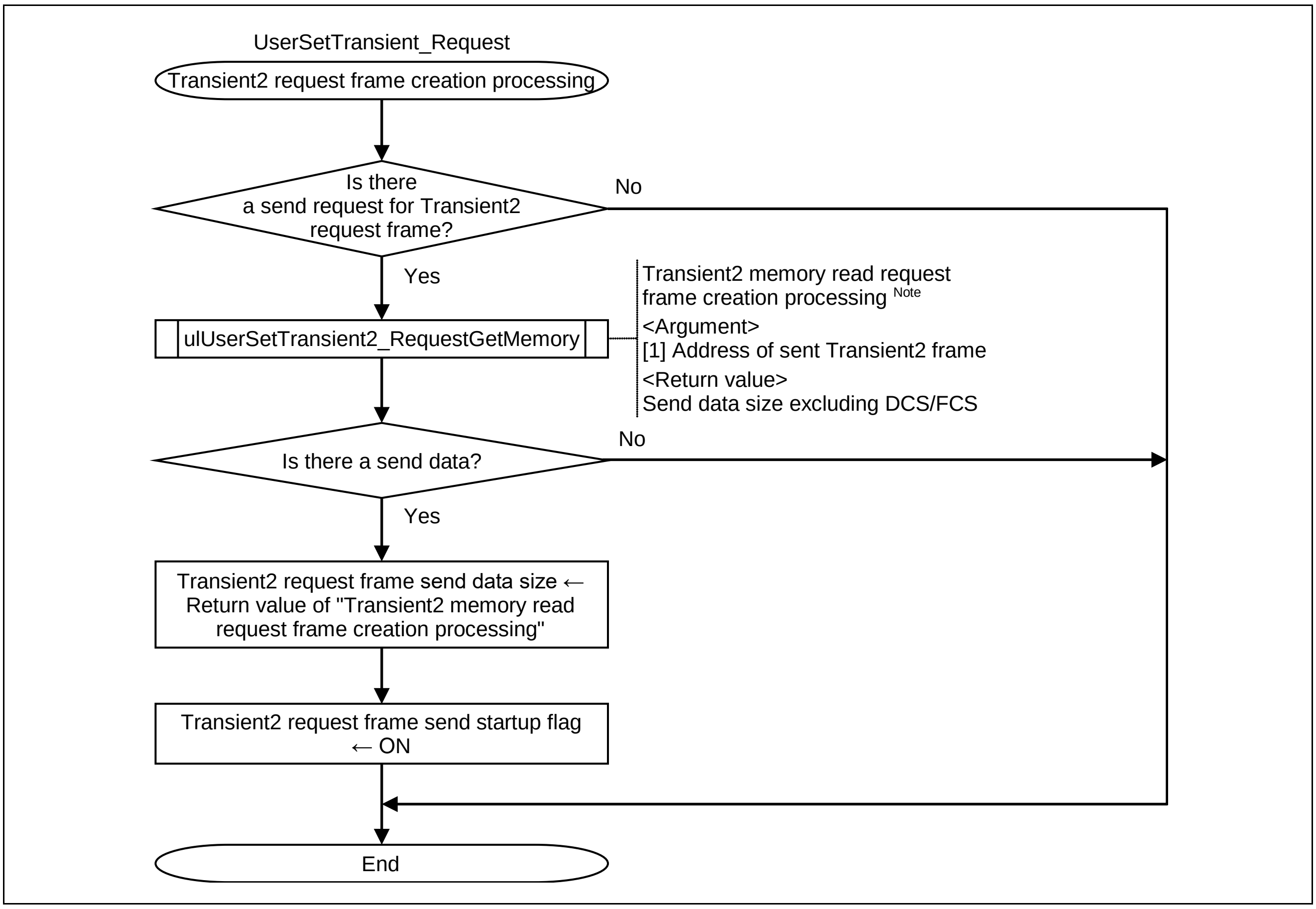

UserSetTransient_Request
Transient2 request frame creation processing
Is there a send request for Transient2 request frame?
No
Yes
ulUserSetTransient2_RequestGetMemory
Transient2 memory read request frame creation processing Note
<Argument>
[1] Address of sent Transient2 frame
<Return value>
Send data size excluding DCS/FCS
Is there a send data?
No
Yes
Transient2 request frame send data size ← Return value of "Transient2 memory read request frame creation processing"
Transient2 request frame send startup flag ← ON
End

Figure 6.18 Flowchart for Transient2 Request Frame Creation Processing

Note. For details, refer to Section 6.2.36 "Transient2 memory read request frame creation processing".

The above flowchart shows an example of "Transient2 memory read request".

Implement the processing as necessary.

6.2.17 Transient1, Transient2, and TransientAck send processing

This function sends Transient1, Transient2, and TransientAck frames.

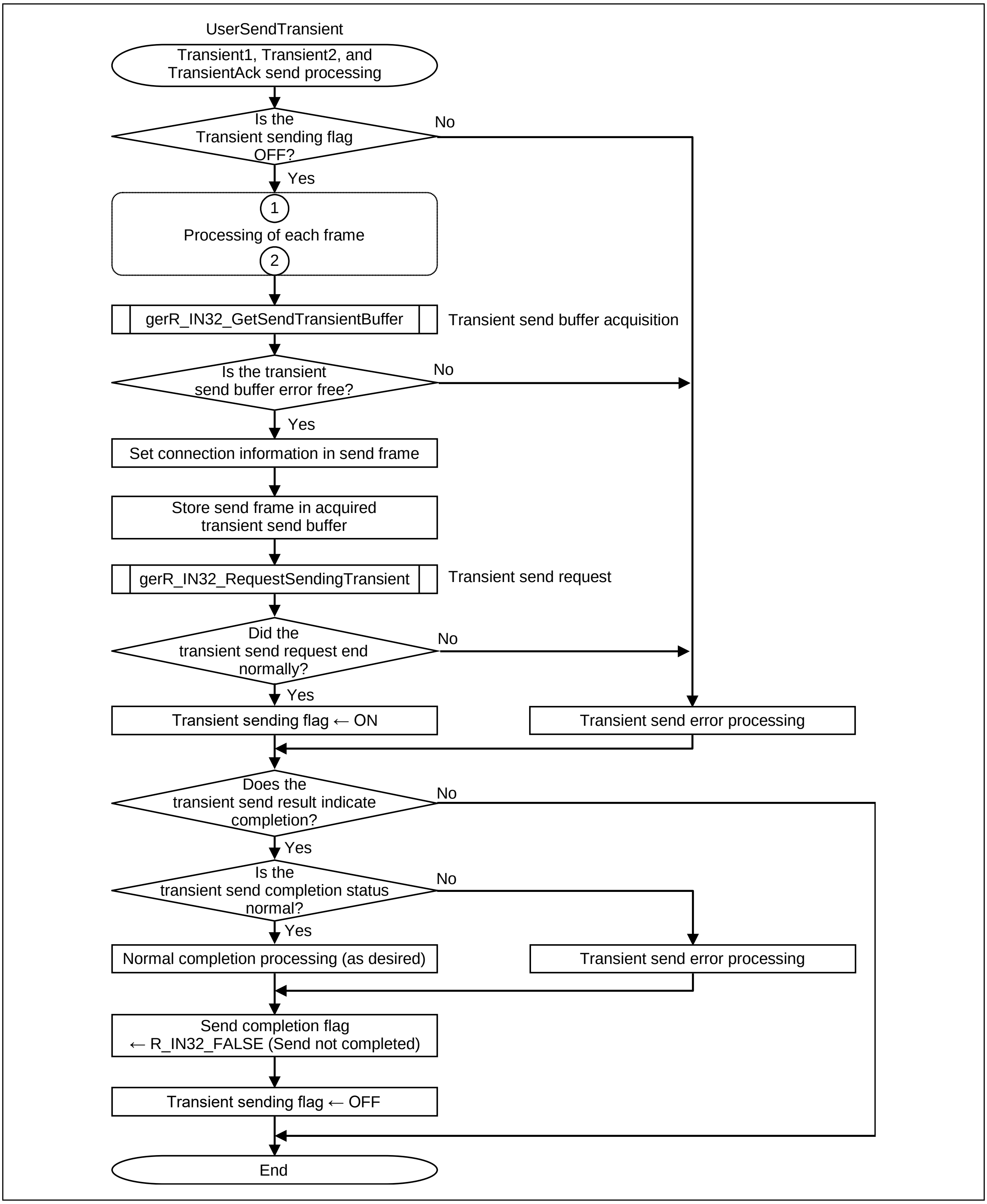

UserSendTransient
Transient1, Transient2, and TransientAck send processing
Is the Transient sending flag OFF?
No
Yes
1
Processing of each frame
2
gerR_IN32_GetSendTransientBuffer
Transient send buffer acquisition
Is the transient send buffer error free?
No
Yes
Set connection information in send frame
Store send frame in acquired transient send buffer
gerR_IN32_RequestSendingTransient
Transient send request
Did the transient send request end normally?
No
Yes
Transient sending flag ← ON
Transient send error processing
Does the transient send result indicate completion?
No
Yes
Is the transient send completion status normal?
No
Yes
Normal completion processing (as desired)
Transient send error processing
Send completion flag ← R_IN32_FALSE (Send not completed)
Transient sending flag ← OFF
End

Figure 6.19 Flowchart for Transient1, Transient2, and TransientAck Send Processing (1/2)

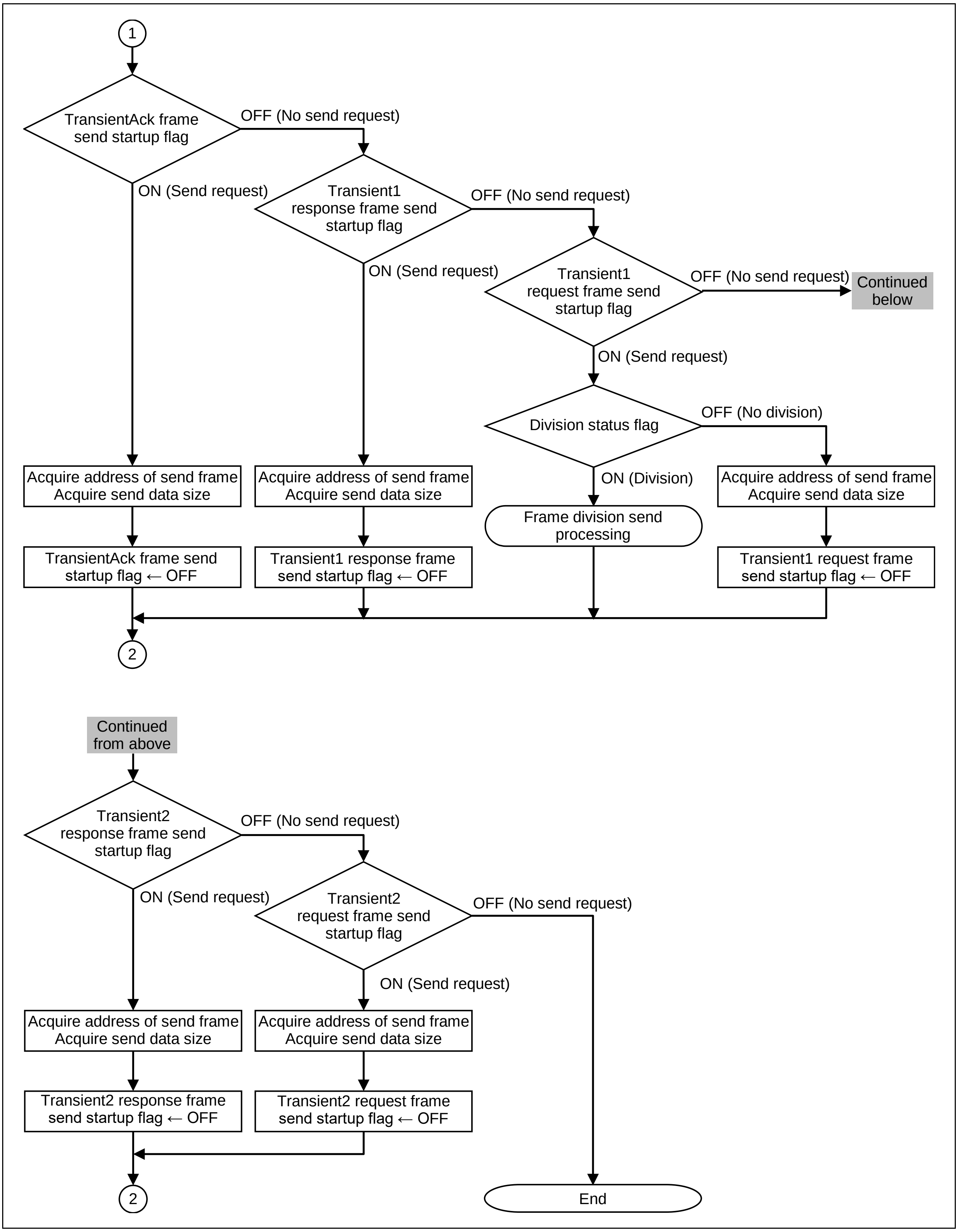

1
TransientAck frame send startup flag
OFF (No send request)
ON (Send request)
Transient1 response frame send startup flag
OFF (No send request)
ON (Send request)
Transient1 request frame send startup flag
OFF (No send request)
Continued below
ON (Send request)
Division status flag
OFF (No division)
ON (Division)
Acquire address of send frame Acquire send data size
Acquire address of send frame Acquire send data size
Frame division send processing
Acquire address of send frame Acquire send data size
TransientAck frame send startup flag ← OFF
Transient1 response frame send startup flag ← OFF
Transient1 request frame send startup flag ← OFF
2
Continued from above
Transient2 response frame send startup flag
OFF (No send request)
ON (Send request)
Transient2 request frame send startup flag
OFF (No send request)
ON (Send request)
Acquire address of send frame Acquire send data size
Acquire address of send frame Acquire send data size
Transient2 response frame send startup flag ← OFF
Transient2 request frame send startup flag ← OFF
2
End

Figure 6.19 Flowchart for Transient1, Transient2, and TransientAck Send Processing (2/2)

[Sending data by dividing data into blocks]

When the transient data requested to be sent is 1466 to 2048 bytes, the transient data can be divided and sent. Implement this processing in accordance with specifications of the developed device.

The following shows an image of the process for divided sending.

For details regarding the Transient1 frame, refer to Section 5.3.2 "CC-Link IE Field specific transient frame format".

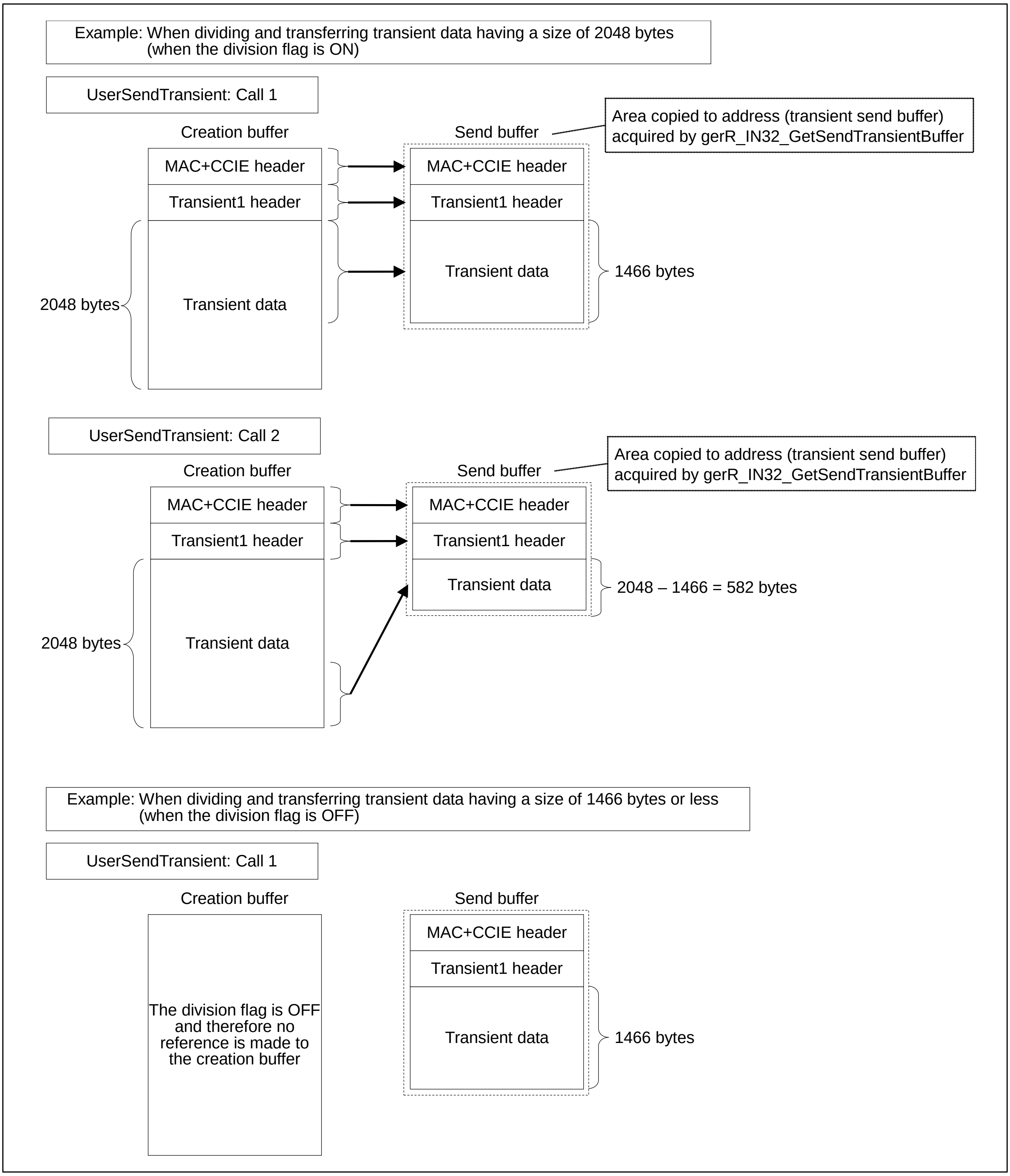

Example: When dividing and transferring transient data having a size of 2048 bytes (when the division flag is ON)
UserSendTransient: Call 1
Creation buffer
Send buffer
Area copied to address (transient send buffer) acquired by gerR_IN32_GetSendTransientBuffer
MAC+CCIE header
Transient1 header
2048 bytes
Transient data
MAC+CCIE header
Transient1 header
Transient data
1466 bytes
UserSendTransient: Call 2
Creation buffer
Send buffer
Area copied to address (transient send buffer) acquired by gerR_IN32_GetSendTransientBuffer
MAC+CCIE header
Transient1 header
2048 bytes
Transient data
MAC+CCIE header
Transient1 header
Transient data
2048 – 1466 = 582 bytes
Example: When dividing and transferring transient data having a size of 1466 bytes or less (when the division flag is OFF)
UserSendTransient: Call 1
Creation buffer
Send buffer
The division flag is OFF and therefore no reference is made to the creation buffer
MAC+CCIE header
Transient1 header
Transient data
1466 bytes

Figure 6.20 Transient Frame Divided Sending Procedure

6.2.18 Transient1 receive data processing

This function analyzes a received Transient1 frame and performs processing in accordance with the analysis result. In addition, this function reconstructs data when a Transient1 frame is received divided.

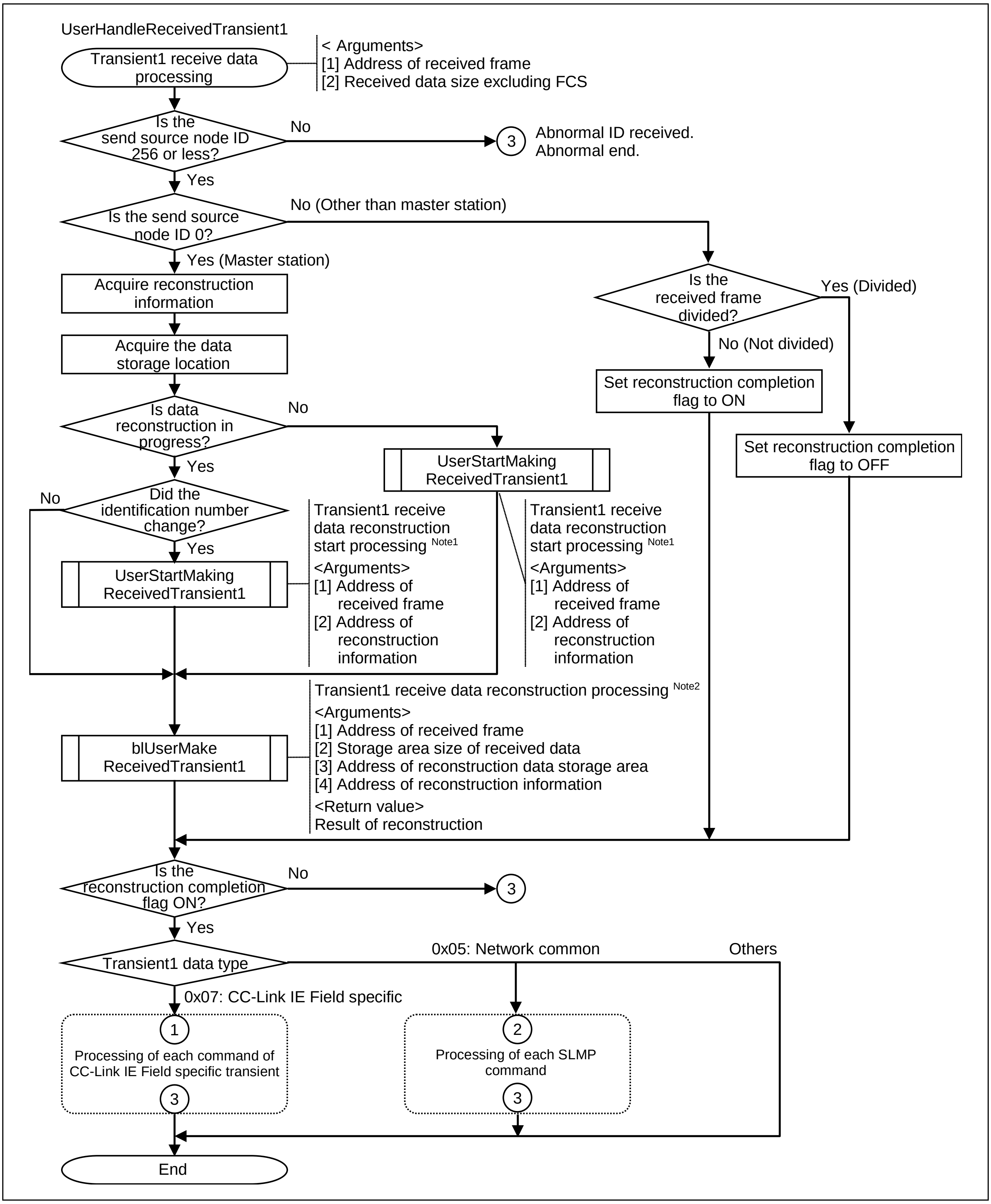

UserHandleReceivedTransient1
Transient1 receive data processing
< Arguments>
[1] Address of received frame
[2] Received data size excluding FCS
Is the send source node ID 256 or less?
No
3
Abnormal ID received.
Abnormal end.
Yes
Is the send source node ID 0?
No (Other than master station)
Yes (Master station)
Acquire reconstruction information
Acquire the data storage location
Is the received frame divided?
Yes (Divided)
No (Not divided)
Set reconstruction completion flag to ON
Set reconstruction completion flag to OFF
Is data reconstruction in progress?
No
Yes
UserStartMaking ReceivedTransient1
No
Did the identification number change?
Yes
UserStartMaking ReceivedTransient1
Transient1 receive data reconstruction start processing Note1
<Arguments>
[1] Address of received frame
[2] Address of reconstruction information
Transient1 receive data reconstruction start processing Note1
<Arguments>
[1] Address of received frame
[2] Address of reconstruction information
Transient1 receive data reconstruction processing Note2
<Arguments>
[1] Address of received frame
[2] Storage area size of received data
[3] Address of reconstruction data storage area
[4] Address of reconstruction information
<Return value>
Result of reconstruction
blUserMake ReceivedTransient1
Is the reconstruction completion flag ON?
No
3
Yes
Transient1 data type
0x05: Network common
Others
0x07: CC-Link IE Field specific
1
Processing of each command of CC-Link IE Field specific transient
3
2
Processing of each SLMP command
3
End

Figure 6.21 Flowchart for Transient1 Receive Data Processing (1/3)

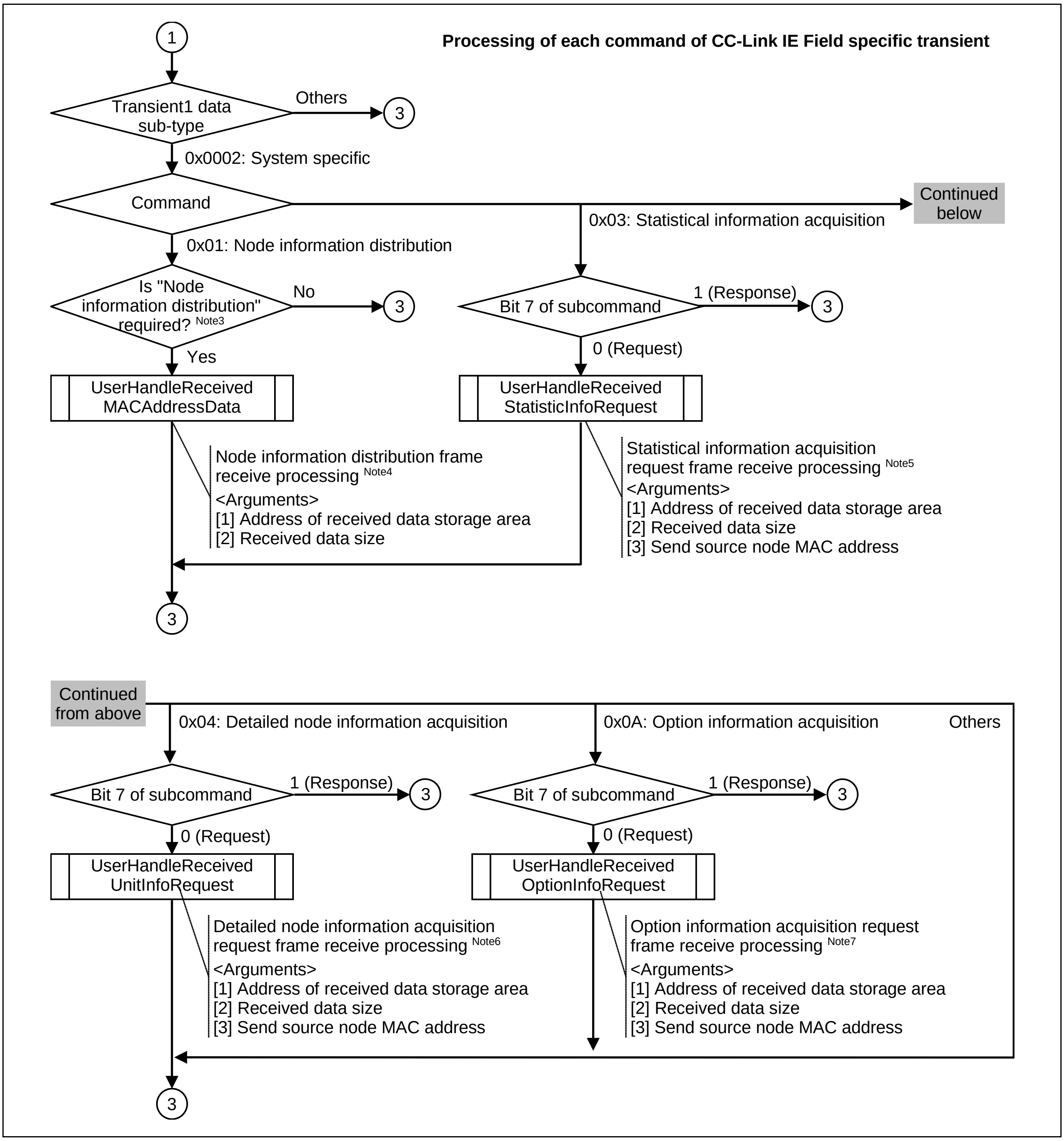

Processing of each command of CC-Link IE Field specific transient
1
Transient1 data sub-type
Others
3
0x0002: System specific
Command
Continued below
0x03: Statistical information acquisition
0x01: Node information distribution
Is "Node information distribution" required? Note3
No
3
Yes
Bit 7 of subcommand
1 (Response)
3
0 (Request)
UserHandleReceived MACAddressData
UserHandleReceived StatisticInfoRequest
Node information distribution frame receive processing Note4
<Arguments>
[1] Address of received data storage area
[2] Received data size
Statistical information acquisition request frame receive processing Note5
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
3
Continued from above
0x04: Detailed node information acquisition
0x0A: Option information acquisition
Others
Bit 7 of subcommand
1 (Response)
3
Bit 7 of subcommand
1 (Response)
3
0 (Request)
0 (Request)
UserHandleReceived UnitInfoRequest
UserHandleReceived OptionInfoRequest
Detailed node information acquisition request frame receive processing Note6
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
Option information acquisition request frame receive processing Note7
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
3

Figure 6.21 Flowchart for Transient1 Receive Data Processing (2/3)

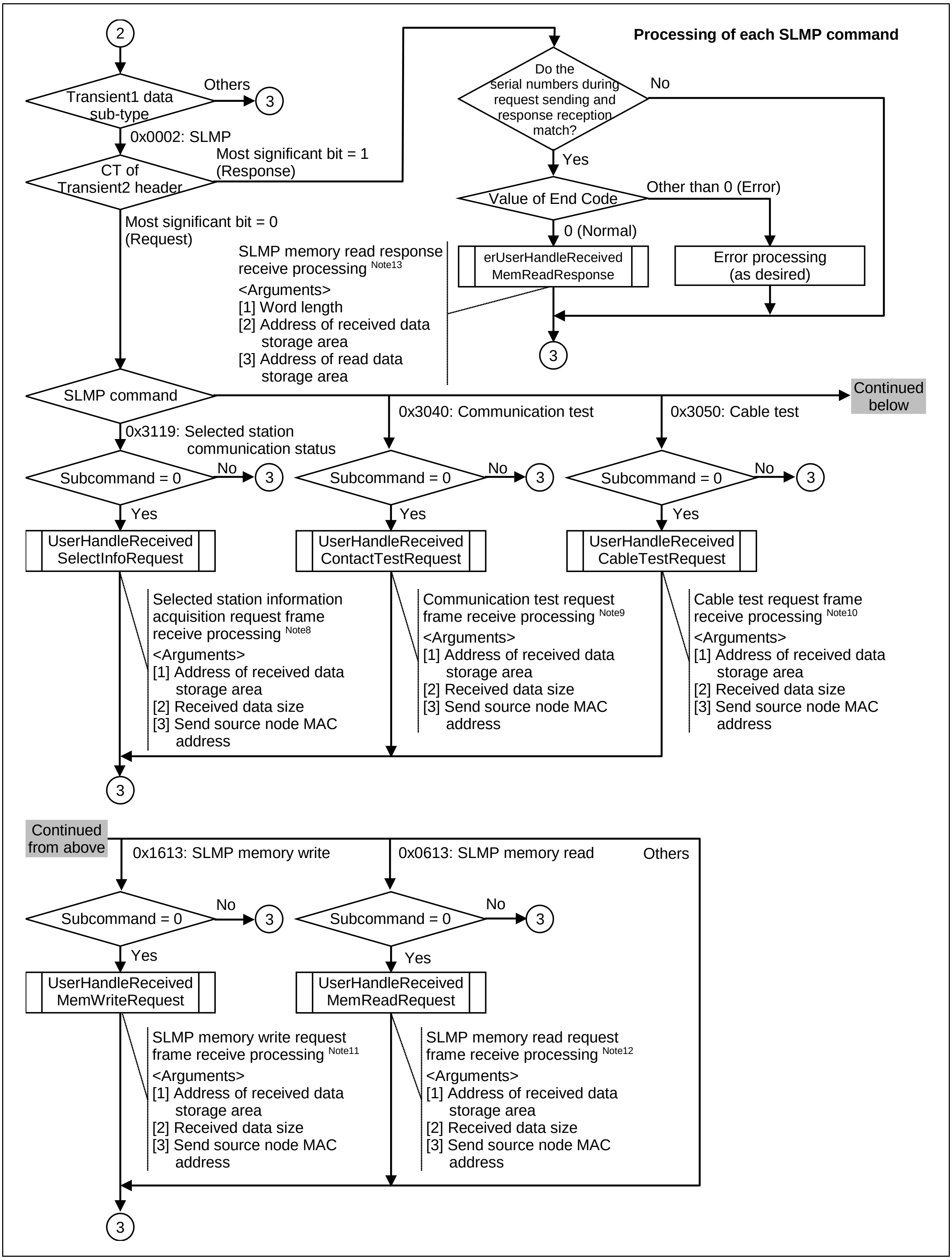

Processing of each SLMP command
2
Transient1 data sub-type
Others
3
0x0002: SLMP
CT of Transient2 header
Most significant bit = 1 (Response)
Most significant bit = 0 (Request)
Do the serial numbers during request sending and response reception match?
No
Yes
Value of End Code
Other than 0 (Error)
0 (Normal)
erUserHandleReceived MemReadResponse
Error processing (as desired)
SLMP memory read response receive processing Note13
<Arguments>
[1] Word length
[2] Address of received data storage area
[3] Address of read data storage area
SLMP command
Continued below
0x3119: Selected station communication status
0x3040: Communication test
0x3050: Cable test
Subcommand = 0
No
Yes
UserHandleReceived SelectInfoRequest
UserHandleReceived ContactTestRequest
UserHandleReceived CableTestRequest
Selected station information acquisition request frame receive processing Note8
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
Communication test request frame receive processing Note9
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
Cable test request frame receive processing Note10
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
Continued from above
0x1613: SLMP memory write
0x0613: SLMP memory read
Others
UserHandleReceived MemWriteRequest
UserHandleReceived MemReadRequest
SLMP memory write request frame receive processing Note11
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
SLMP memory read request frame receive processing Note12
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address

Figure 6.21 Flowchart for Transient1 Receive Data Processing (3/3)

Note 1. For details, refer to Section 6.2.19 "Transient1 receive data reconstruction start processing".

2. For details, refer to Section 6.2.20 "Transient1 receive data reconstruction processing".

3. If R_IN32_FALSE is set by the initial value of (g) Node information distribution request in B) R_IN32_UNITINIT_T initial setup of gerR_IN32_Initialize, "Node information distribution" is not required.
In this case, specify gblUserMACAddressTableRequest to R_IN32_FALSE.

4. For details, refer to Section 6.2.21 "Node information distribution frame receive processing".

5. For details, refer to Section 6.2.23 "Statistical information acquisition request frame receive processing".

6. For details, refer to Section 6.2.25 "Detailed node information acquisition request frame receive processing".

7. For details, refer to Section 6.2.27 "Option information acquisition request frame receive processing".

8. For details, refer to Section 6.2.28 "Selected station information acquisition request frame receive processing".

9. For details, refer to Section 6.2.29 "Communication test request frame receive processing".

10. For details, refer to Section 6.2.30 "Cable test request frame receive processing".

11. For details, refer to Section 6.2.40 "SLMP memory write request frame receive processing".

12. For details, refer to Section 6.2.39 "SLMP memory read request frame receive processing".

13. For details, refer to Section 6.2.44 "SLMP memory read response receive processing".

[SLMP request reception from master station]

The CC-Link IE Field Network diagnostics and parameter processing/command execution of slave stations can be performed using the engineering tool. These functions can be used by the own station responding to an SLMP request frame from the master station.

The following shows an image of the processing procedure in which the server sends SLMP response frame in response to SLMP request frame from the master station.
An example of selected station information is given here. The processing for sending and receiving is the same as that for the communication test, cable test, and the commands described in CSP+.

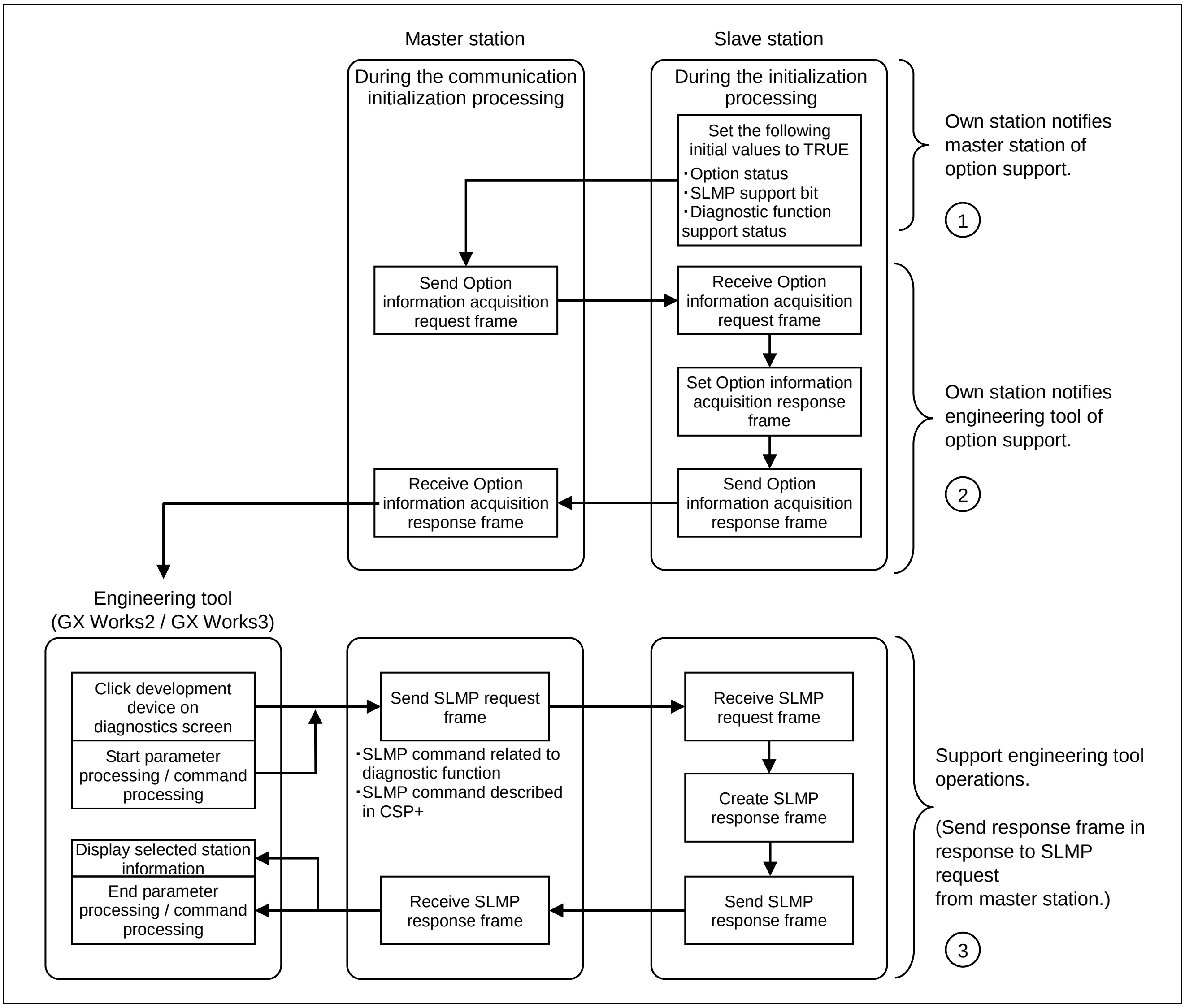

Master station
Slave station
During the communication initialization processing
During the initialization processing
Set the following initial values to TRUE
・Option status
・SLMP support bit
・Diagnostic function support status
Own station notifies master station of option support.
1
Send Option information acquisition request frame
Receive Option information acquisition request frame
Set Option information acquisition response frame
Own station notifies engineering tool of option support.
2
Receive Option information acquisition response frame
Send Option information acquisition response frame
Engineering tool (GX Works2 / GX Works3)
Click development device on diagnostics screen
Start parameter processing / command processing
Send SLMP request frame
・SLMP command related to diagnostic function
・SLMP command described in CSP+
Receive SLMP request frame
Create SLMP response frame
Support engineering tool operations.
(Send response frame in response to SLMP request from master station.)
Display selected station information
End parameter processing / command processing
Receive SLMP response frame
Send SLMP response frame
3

Figure 6.22 SMP Request Reception Procedure

1) R_IN32_UNITINIT_T setup (R-IN32M3-CL initial setup)

Set the following members of R_IN32_UNITINIT_T to "R_IN32_TRUE". (Refer to Section 6.4.1(2) "gerR_IN32_Initialize".)

・ulOptionSupport (Initial value of option status)

・ulSlmpSupport (Initial value of SLMP support bit)

・ulSlmpDiagnosisSupport (Initial value of diagnostic function support status)

2) Response to Option information acquisition request frame

UserHandleReceivedOptionInfoRequest (Option information acquisition request frame receive processing) responds to the Option information acquisition request frame from the master station.

3) Response to Selected station information acquisition request frame

UserHandleReceivedSelectInfoRequest (Selected station information acquisition request frame receive processing) responds to the Selected station information acquisition request frame from the master station.

6.2.19 Transient1 receive data reconstruction start processing

This function starts reconstructing the divided Transient1 receive frame.

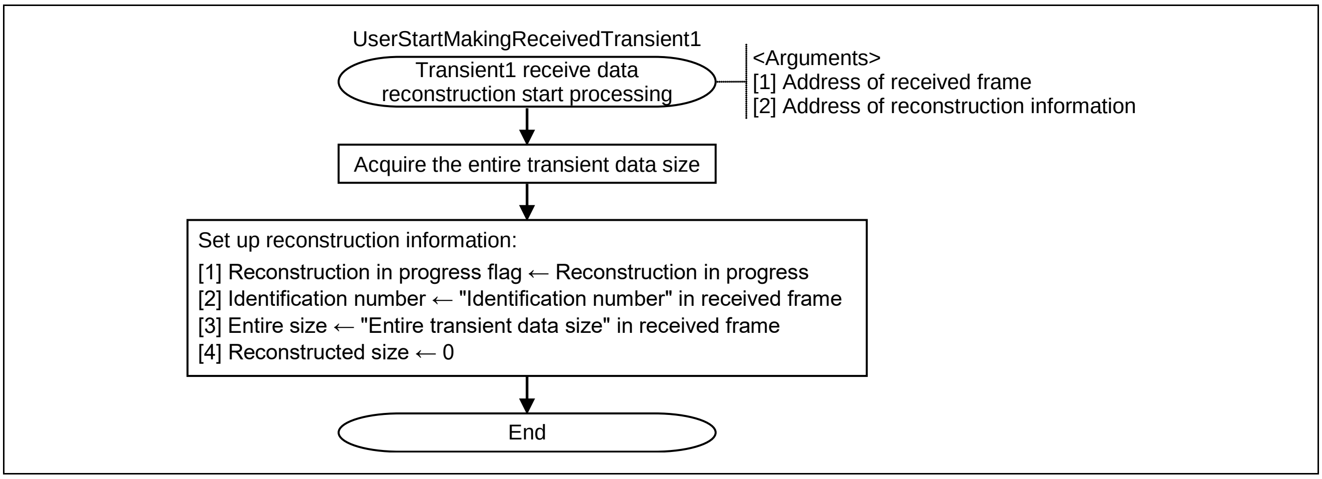

UserStartMakingReceivedTransient1
Transient1 receive data reconstruction start processing
<Arguments>
[1] Address of received frame
[2] Address of reconstruction information
Acquire the entire transient data size
Set up reconstruction information:
[1] Reconstruction in progress flag ← Reconstruction in progress
[2] Identification number ← "Identification number" in received frame
[3] Entire size ← "Entire transient data size" in received frame
[4] Reconstructed size ← 0
End

Figure 6.23 Flowchart for Transient1 Receive Data Reconstruction Start Processing

6.2.20 Transient1 receive data reconstruction processing

This function reconstructs the data of the Transient1 frame.

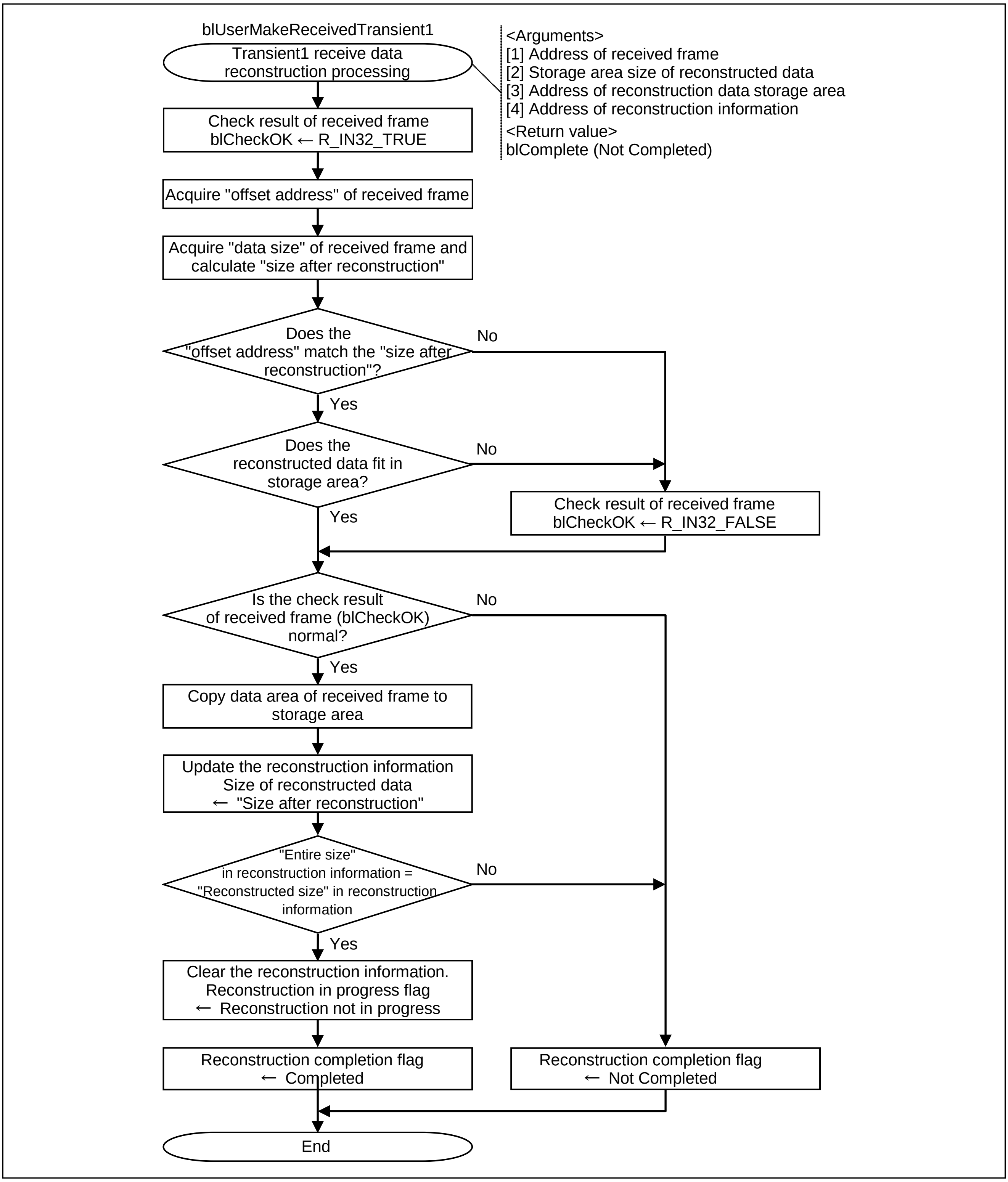

blUserMakeReceivedTransient1
Transient1 receive data reconstruction processing
<Arguments>
[1] Address of received frame
[2] Storage area size of reconstructed data
[3] Address of reconstruction data storage area
[4] Address of reconstruction information
<Return value>
blComplete (Not Completed)
Check result of received frame blCheckOK ← R_IN32_TRUE
Acquire "offset address" of received frame
Acquire "data size" of received frame and calculate "size after reconstruction"
Does the "offset address" match the "size after reconstruction"?
No
Yes
Does the reconstructed data fit in storage area?
No
Yes
Check result of received frame blCheckOK ← R_IN32_FALSE
Is the check result of received frame (blCheckOK) normal?
No
Yes
Copy data area of received frame to storage area
Update the reconstruction information Size of reconstructed data ← "Size after reconstruction"
"Entire size" in reconstruction information = "Reconstructed size" in reconstruction information
No
Yes
Clear the reconstruction information. Reconstruction in progress flag ← Reconstruction not in progress
Reconstruction completion flag ← Completed
Reconstruction completion flag ← Not Completed
End

Figure 6.24 Flowchart for Transient1 Receive Data Reconstruction Processing

6.2.21 Node information distribution frame receive processing

This function receives a Node information distribution frame and registers the information of each node.

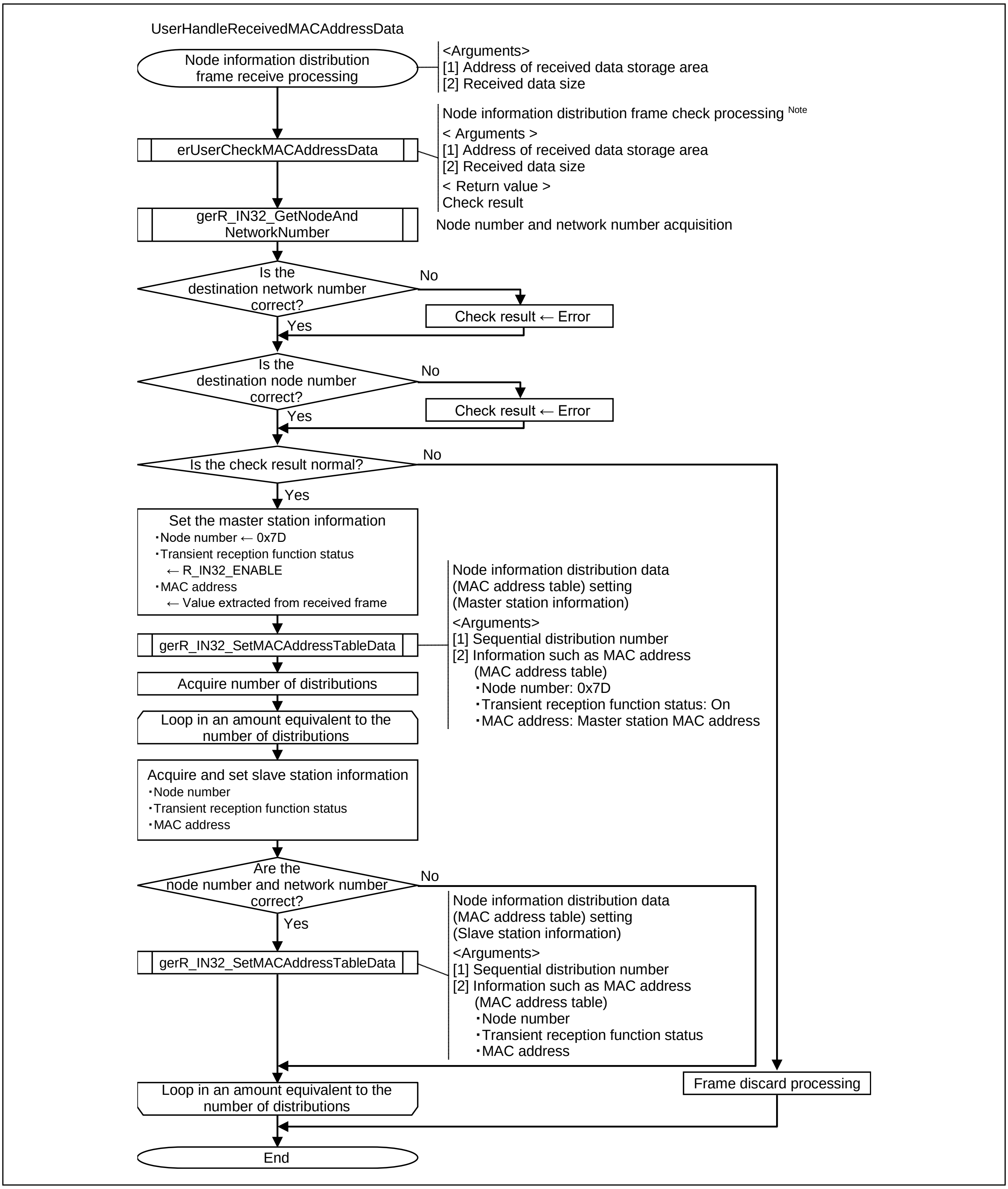

UserHandleReceivedMACAddressData
Node information distribution frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
Node information distribution frame check processing Note
< Arguments >
[1] Address of received data storage area
[2] Received data size
< Return value >
Check result
erUserCheckMACAddressData
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Check result ← Error
Yes
Is the destination node number correct?
No
Check result ← Error
Yes
Is the check result normal?
No
Yes
Set the master station information
・Node number ← 0x7D
・Transient reception function status ← R_IN32_ENABLE
・MAC address ← Value extracted from received frame
Node information distribution data (MAC address table) setting (Master station information)
<Arguments>
[1] Sequential distribution number
[2] Information such as MAC address (MAC address table)
・Node number: 0x7D
・Transient reception function status: On
・MAC address: Master station MAC address
gerR_IN32_SetMACAddressTableData
Acquire number of distributions
Loop in an amount equivalent to the number of distributions
Acquire and set slave station information
・Node number
・Transient reception function status
・MAC address
Are the node number and network number correct?
No
Yes
Node information distribution data (MAC address table) setting (Slave station information)
<Arguments>
[1] Sequential distribution number
[2] Information such as MAC address (MAC address table)
・Node number
・Transient reception function status
・MAC address
gerR_IN32_SetMACAddressTableData
Frame discard processing
Loop in an amount equivalent to the number of distributions
End

Figure 6.25 Flowchart for Node Information Distribution Frame Receive Processing

Note. For details, refer to Section 6.2.22 "Node information distribution frame check processing".

6.2.22 Node information distribution frame check processing

This function checks the data in Node information distribution frame.

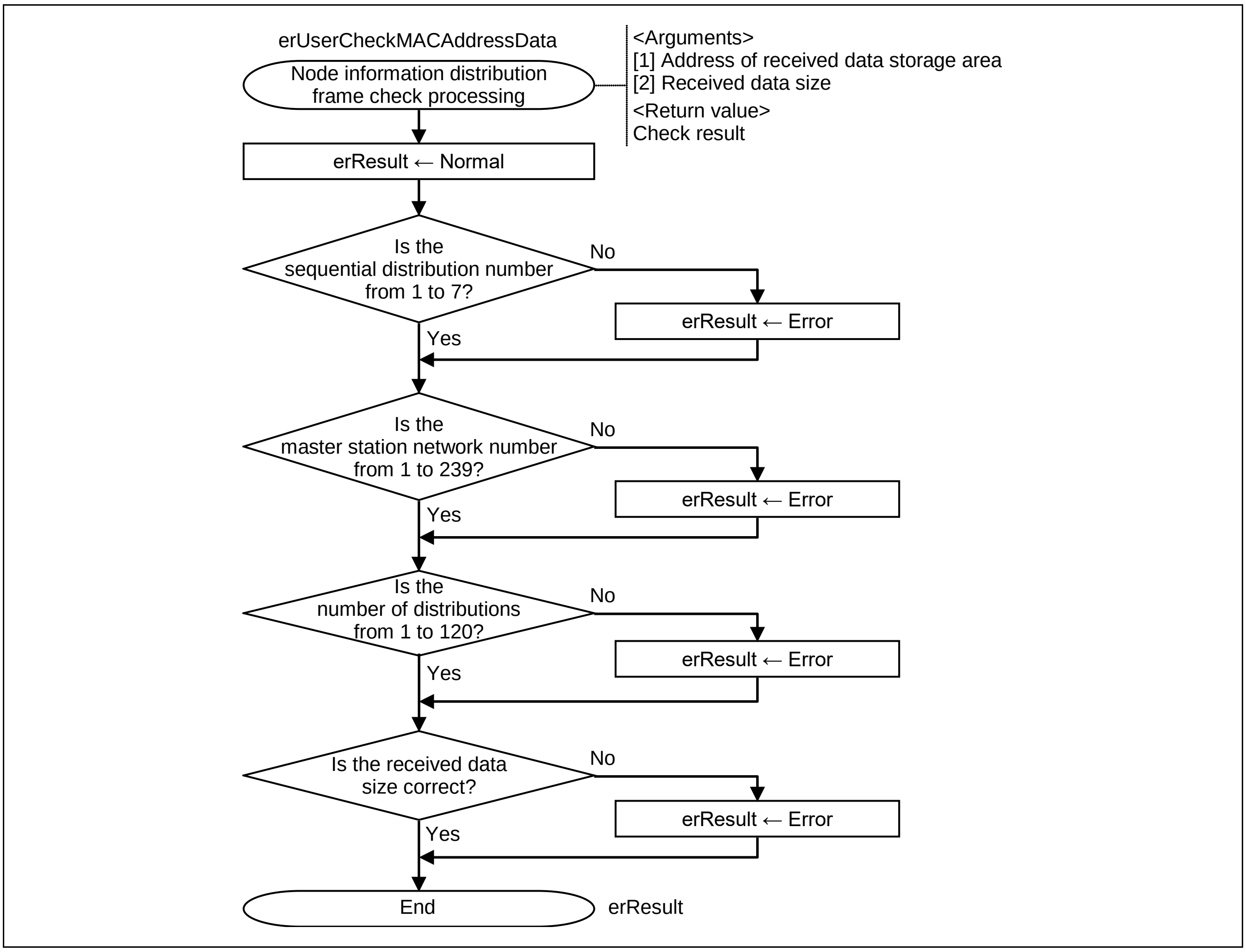

erUserCheckMACAddressData
Node information distribution frame check processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
<Return value>
Check result
erResult ← Normal
Is the sequential distribution number from 1 to 7?
No
erResult ← Error
Yes
Is the master station network number from 1 to 239?
No
erResult ← Error
Yes
Is the number of distributions from 1 to 120?
No
erResult ← Error
Yes
Is the received data size correct?
No
erResult ← Error
Yes
End
erResult

Figure 6.26 Flowchart for Node Information Distribution Frame Check Processing

6.2.23 Statistical information acquisition request frame receive processing

This function performs processing when Statistical information acquisition request frame is received.

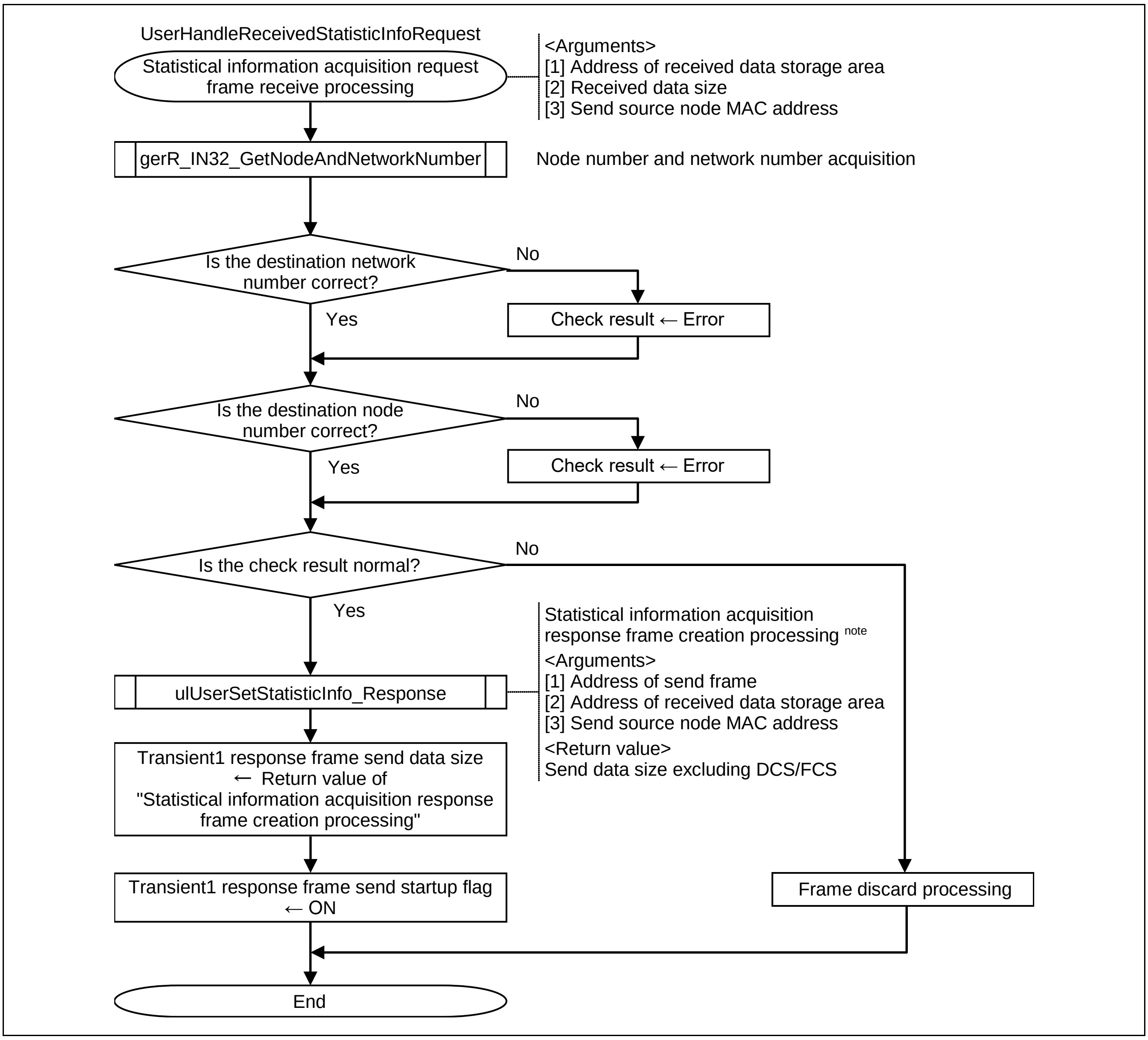
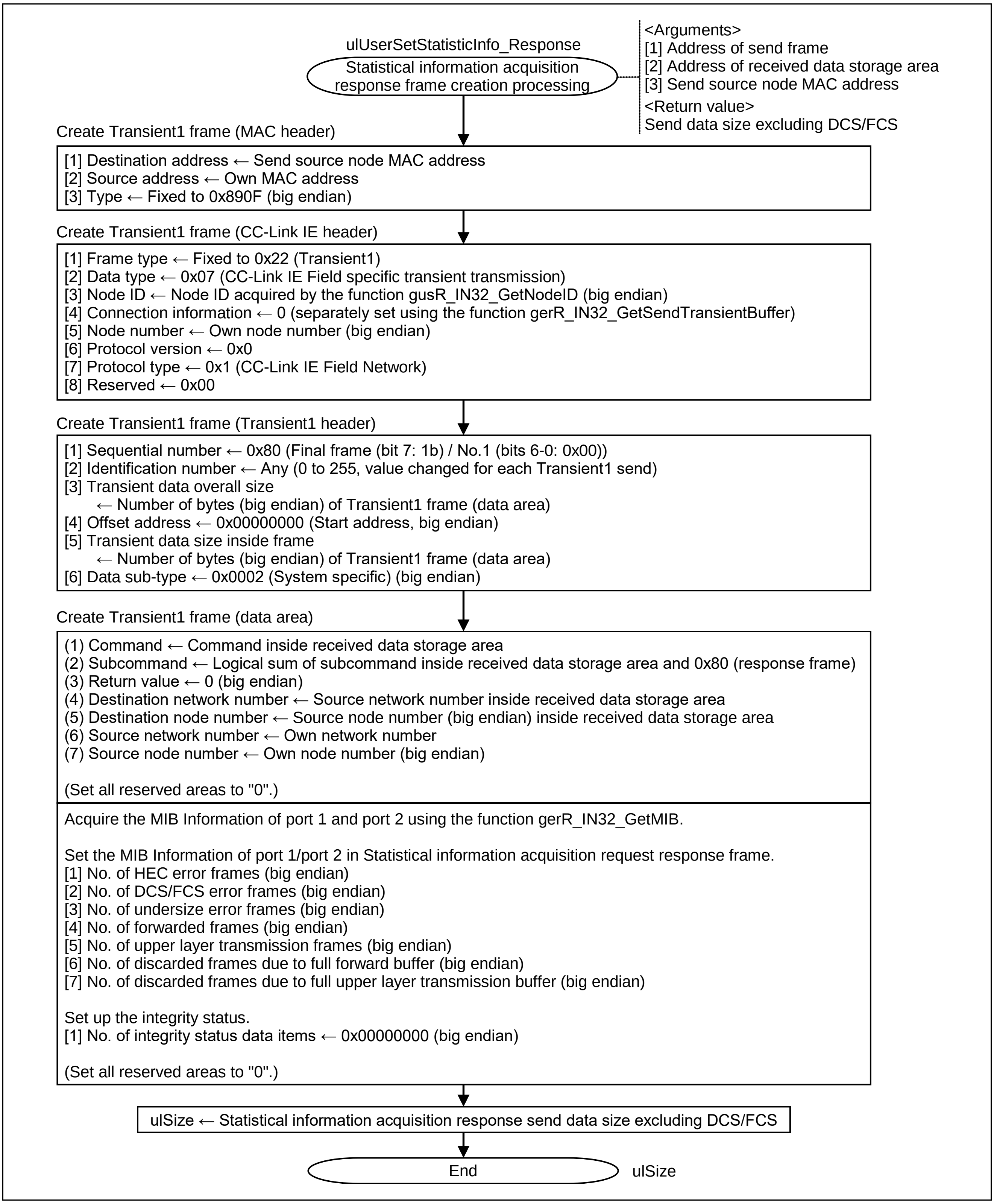

UserHandleReceivedStatisticInfoRequest
Statistical information acquisition request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Yes
Check result ← Error
Is the destination node number correct?
No
Yes
Check result ← Error
Is the check result normal?
No
Yes
Statistical information acquisition response frame creation processing note
<Arguments>
[1] Address of send frame
[2] Address of received data storage area
[3] Send source node MAC address
<Return value>
Send data size excluding DCS/FCS
ulUserSetStatisticInfo_Response
Transient1 response frame send data size ← Return value of "Statistical information acquisition response frame creation processing"
Transient1 response frame send startup flag ← ON
Frame discard processing
End

Figure 6.27 Flowchart for Statistical Information Acquisition Request Frame Receive Processing

Note. For details, refer to Section 6.2.24 "Statistical information acquisition response frame creation processing".

6.2.24 Statistical information acquisition response frame creation processing

This function creates Statistical information acquisition response frame.



ulUserSetStatisticInfo_Response

Statistical information acquisition response frame creation processing

<Arguments>
[1] Address of send frame
[2] Address of received data storage area
[3] Send source node MAC address
<Return value>
Send data size excluding DCS/FCS

Create Transient1 frame (MAC header)

[1] Destination address ← Send source node MAC address
[2] Source address ← Own MAC address
[3] Type ← Fixed to 0x890F (big endian)

Create Transient1 frame (CC-Link IE header)

[1] Frame type ← Fixed to 0x22 (Transient1)
[2] Data type ← 0x07 (CC-Link IE Field specific transient transmission)
[3] Node ID ← Node ID acquired by the function gusR_IN32_GetNodeID (big endian)
[4] Connection information ← 0 (separately set using the function gerR_IN32_GetSendTransientBuffer)
[5] Node number ← Own node number (big endian)
[6] Protocol version ← 0x0
[7] Protocol type ← 0x1 (CC-Link IE Field Network)
[8] Reserved ← 0x00

Create Transient1 frame (Transient1 header)

[1] Sequential number ← 0x80 (Final frame (bit 7: 1b) / No.1 (bits 6-0: 0x00))
[2] Identification number ← Any (0 to 255, value changed for each Transient1 send)
[3] Transient data overall size
← Number of bytes (big endian) of Transient1 frame (data area)
[4] Offset address ← 0x00000000 (Start address, big endian)
[5] Transient data size inside frame
← Number of bytes (big endian) of Transient1 frame (data area)
[6] Data sub-type ← 0x0002 (System specific) (big endian)

Create Transient1 frame (data area)

(1) Command ← Command inside received data storage area
(2) Subcommand ← Logical sum of subcommand inside received data storage area and 0x80 (response frame)
(3) Return value ← 0 (big endian)
(4) Destination network number ← Source network number inside received data storage area
(5) Destination node number ← Source node number (big endian) inside received data storage area
(6) Source network number ← Own network number
(7) Source node number ← Own node number (big endian)

(Set all reserved areas to "0".)

Acquire the MIB Information of port 1 and port 2 using the function gerR_IN32_GetMIB.

Set the MIB Information of port 1/port 2 in Statistical information acquisition request response frame.
[1] No. of HEC error frames (big endian)
[2] No. of DCS/FCS error frames (big endian)
[3] No. of undersize error frames (big endian)
[4] No. of forwarded frames (big endian)
[5] No. of upper layer transmission frames (big endian)
[6] No. of discarded frames due to full forward buffer (big endian)
[7] No. of discarded frames due to full upper layer transmission buffer (big endian)

Set up the integrity status.
[1] No. of integrity status data items ← 0x00000000 (big endian)

(Set all reserved areas to "0".)

ulSize ← Statistical information acquisition response send data size excluding DCS/FCS

End

ulSize

Figure 6.28 Flowchart for Statistical Information Acquisition Response Frame Creation Processing

6.2.25 Detailed node information acquisition request frame receive processing

This function performs processing when Detailed node information acquisition request frame is received.

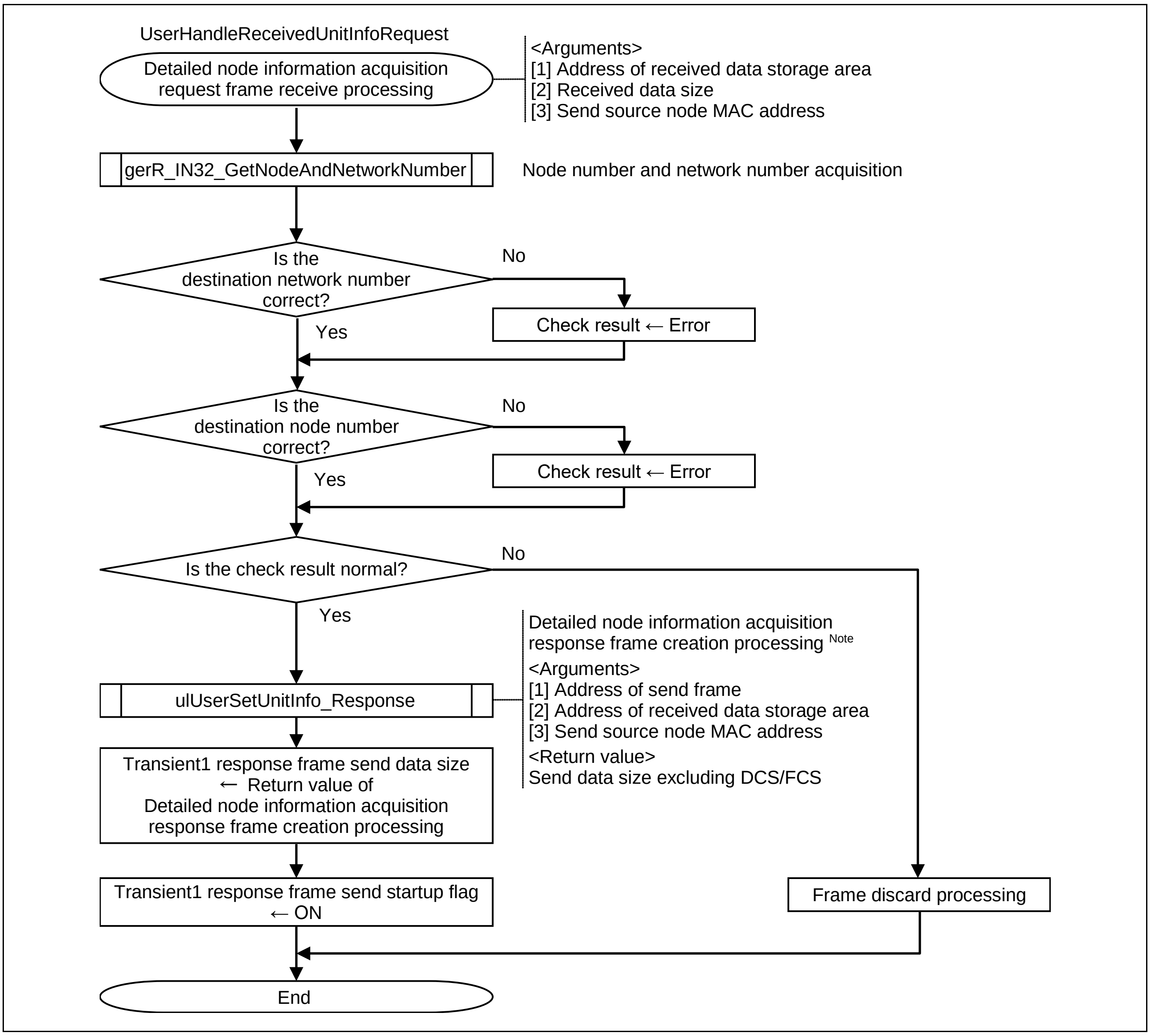
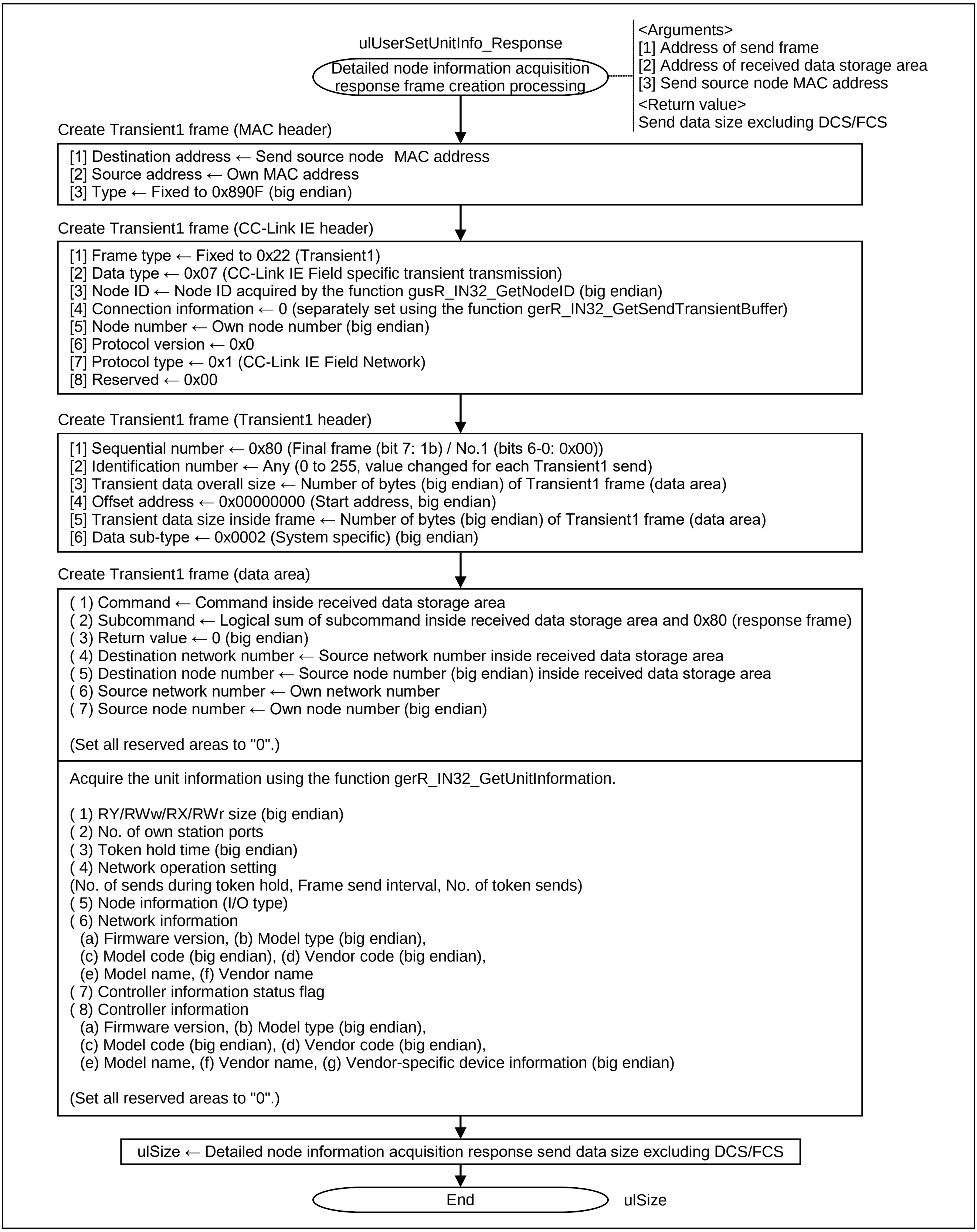

UserHandleReceivedUnitInfoRequest
Detailed node information acquisition request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Check result ← Error
Yes
Is the destination node number correct?
No
Check result ← Error
Yes
Is the check result normal?
No
Yes
ulUserSetUnitInfo_Response
Detailed node information acquisition response frame creation processing Note
<Arguments>
[1] Address of send frame
[2] Address of received data storage area
[3] Send source node MAC address
<Return value>
Send data size excluding DCS/FCS
Transient1 response frame send data size ← Return value of Detailed node information acquisition response frame creation processing
Transient1 response frame send startup flag ← ON
Frame discard processing
End

Figure 6.29 Flowchart for Detailed Node Information Acquisition Request Frame Receive Processing

Note. For details, refer to Section 6.2.26 "Detailed node information acquisition response frame creation processing".

6.2.26 Detailed node information acquisition response frame creation processing

This function creates Detailed node information acquisition response frame.



ulUserSetUnitInfo_Response

Detailed node information acquisition response frame creation processing

<Arguments>
[1] Address of send frame
[2] Address of received data storage area
[3] Send source node MAC address
<Return value>
Send data size excluding DCS/FCS

Create Transient1 frame (MAC header)

[1] Destination address ← Send source node MAC address
[2] Source address ← Own MAC address
[3] Type ← Fixed to 0x890F (big endian)

Create Transient1 frame (CC-Link IE header)

[1] Frame type ← Fixed to 0x22 (Transient1)
[2] Data type ← 0x07 (CC-Link IE Field specific transient transmission)
[3] Node ID ← Node ID acquired by the function gusR_IN32_GetNodeID (big endian)
[4] Connection information ← 0 (separately set using the function gerR_IN32_GetSendTransientBuffer)
[5] Node number ← Own node number (big endian)
[6] Protocol version ← 0x0
[7] Protocol type ← 0x1 (CC-Link IE Field Network)
[8] Reserved ← 0x00

Create Transient1 frame (Transient1 header)

[1] Sequential number ← 0x80 (Final frame (bit 7: 1b) / No.1 (bits 6-0: 0x00))
[2] Identification number ← Any (0 to 255, value changed for each Transient1 send)
[3] Transient data overall size ← Number of bytes (big endian) of Transient1 frame (data area)
[4] Offset address ← 0x00000000 (Start address, big endian)
[5] Transient data size inside frame ← Number of bytes (big endian) of Transient1 frame (data area)
[6] Data sub-type ← 0x0002 (System specific) (big endian)

Create Transient1 frame (data area)

(1) Command ← Command inside received data storage area
(2) Subcommand ← Logical sum of subcommand inside received data storage area and 0x80 (response frame)
(3) Return value ← 0 (big endian)
(4) Destination network number ← Source network number inside received data storage area
(5) Destination node number ← Source node number (big endian) inside received data storage area
(6) Source network number ← Own network number
(7) Source node number ← Own node number (big endian)

(Set all reserved areas to "0".)

Acquire the unit information using the function gerR_IN32_GetUnitInformation.

(1) RY/RWw/RX/RWr size (big endian)
(2) No. of own station ports
(3) Token hold time (big endian)
(4) Network operation setting
(No. of sends during token hold, Frame send interval, No. of token sends)
(5) Node information (I/O type)
(6) Network information
(a) Firmware version, (b) Model type (big endian),
(c) Model code (big endian), (d) Vendor code (big endian),
(e) Model name, (f) Vendor name
(7) Controller information status flag
(8) Controller information
(a) Firmware version, (b) Model type (big endian),
(c) Model code (big endian), (d) Vendor code (big endian),
(e) Model name, (f) Vendor name, (g) Vendor-specific device information (big endian)

(Set all reserved areas to "0".)

ulSize ← Detailed node information acquisition response send data size excluding DCS/FCS

End

ulSize

Figure 6.30 Flowchart for Detailed Node Information Acquisition Response Frame Creation Processing

6.2.27 Option information acquisition request frame receive processing

This function performs processing when Option information acquisition request frame is received.
The processing is to notify the master station that the own station supports SLMP frame.

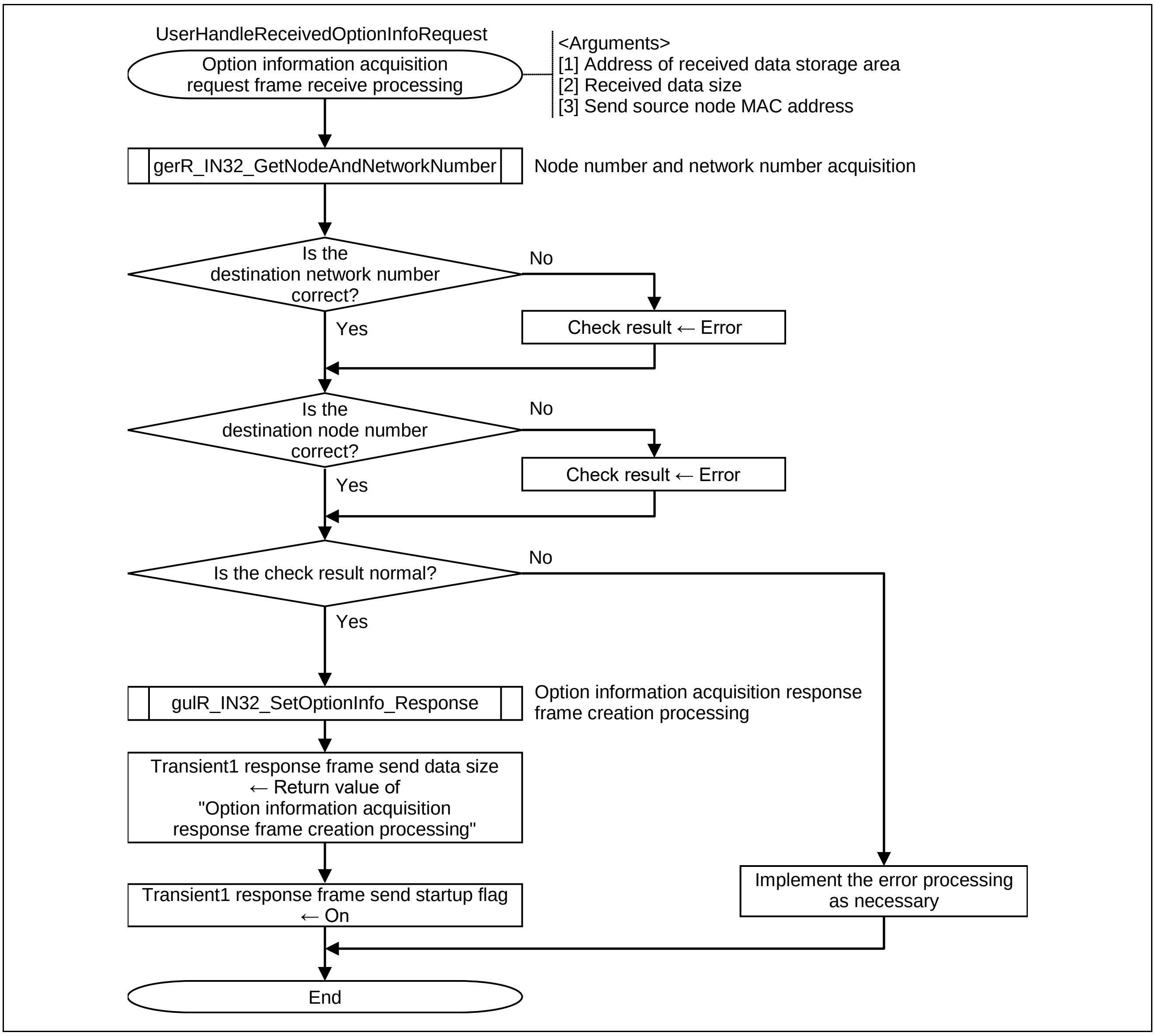

UserHandleReceivedOptionInfoRequest
Option information acquisition request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Check result ← Error
Yes
Is the destination node number correct?
No
Check result ← Error
Yes
Is the check result normal?
No
Yes
gulR_IN32_SetOptionInfo_Response
Option information acquisition response frame creation processing
Transient1 response frame send data size ← Return value of "Option information acquisition response frame creation processing"
Transient1 response frame send startup flag ← On
Implement the error processing as necessary
End

Figure 6.31 Flowchart for Option Information Acquisition Request Frame Receive Processing

6.2.28 Selected station information acquisition request frame receive processing

This function performs processing when Selected station information acquisition request frame is received.

The processing is required to support "Selected station communication status monitor" of CC-Link IE Field diagnostic function.

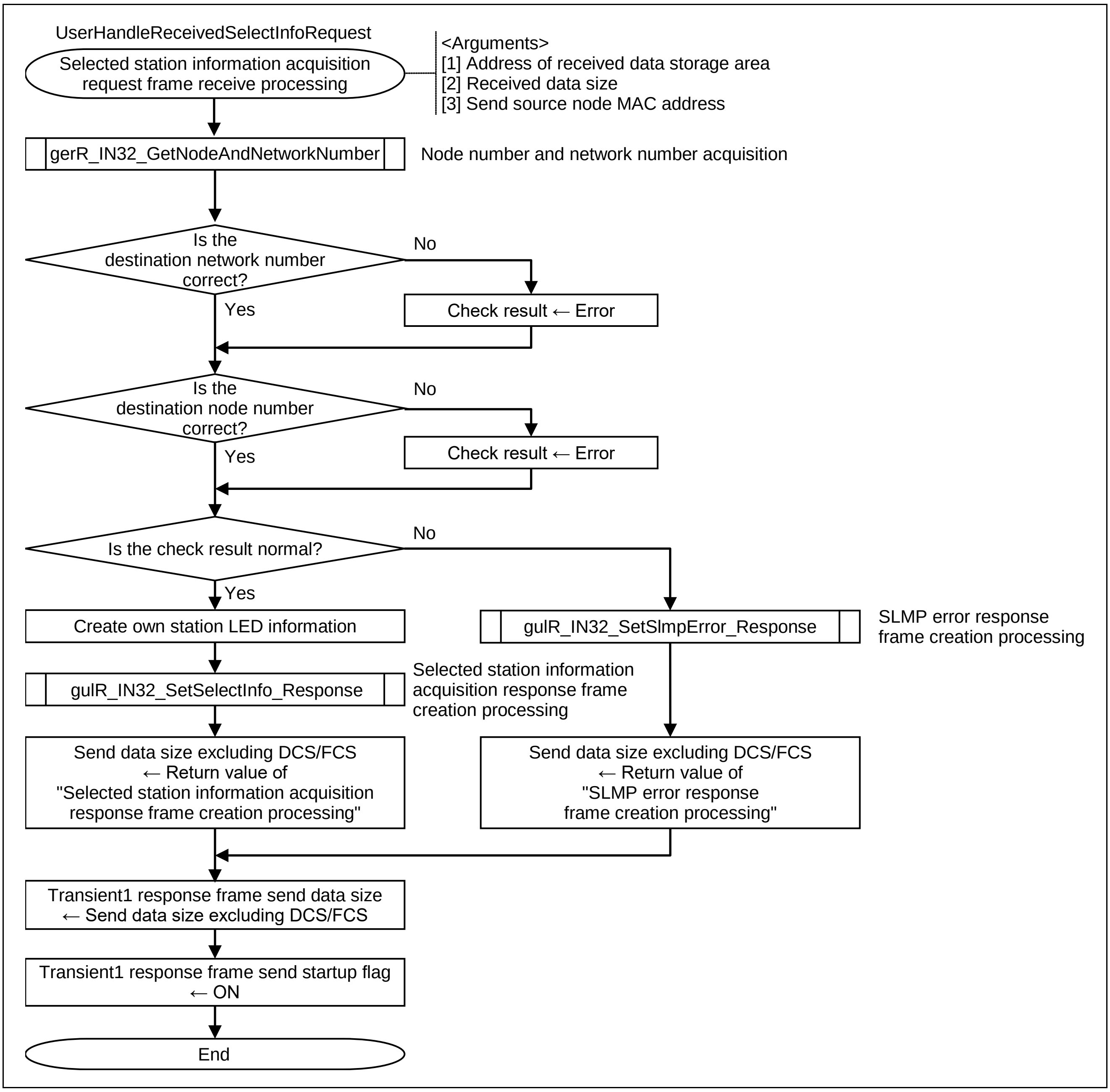

UserHandleReceivedSelectInfoRequest
Selected station information acquisition request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Yes
Check result ← Error
Is the destination node number correct?
No
Yes
Check result ← Error
Is the check result normal?
No
Yes
Create own station LED information
gulR_IN32_SetSlmpError_Response
SLMP error response frame creation processing
gulR_IN32_SetSelectInfo_Response
Selected station information acquisition response frame creation processing
Send data size excluding DCS/FCS ← Return value of "Selected station information acquisition response frame creation processing"
Send data size excluding DCS/FCS ← Return value of "SLMP error response frame creation processing"
Transient1 response frame send data size ← Send data size excluding DCS/FCS
Transient1 response frame send startup flag ← ON
End

Figure 6.32 Flowchart for Selected Station Information Acquisition Request Frame Receive Processing

6.2.29 Communication test request frame receive processing

This function performs processing when Communication test request frame is received.
The processing is required to support "Communication test" of CC-Link IE Field diagnostic function.

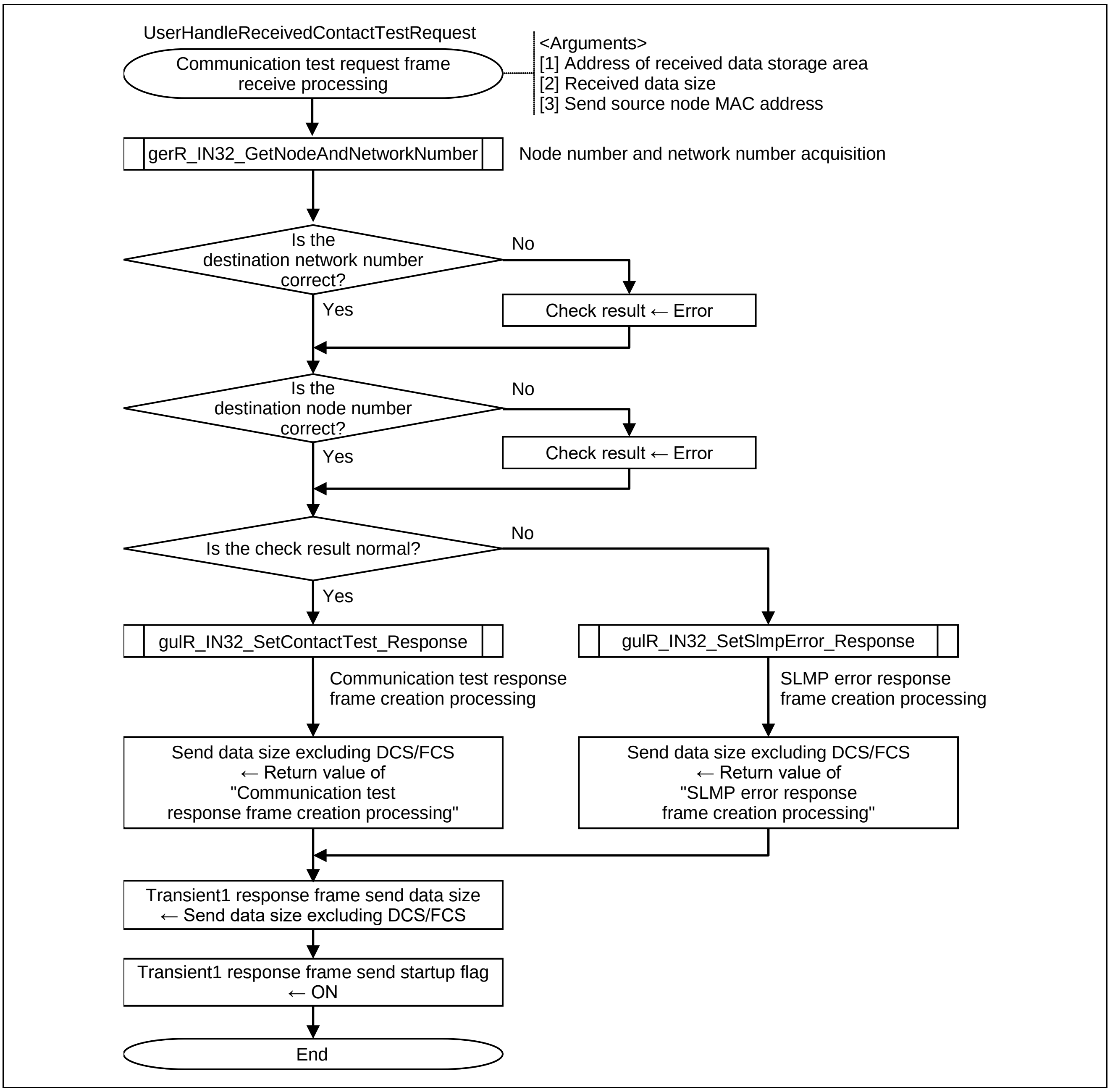

UserHandleReceivedContactTestRequest
Communication test request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Yes
Check result ← Error
Is the destination node number correct?
No
Yes
Check result ← Error
Is the check result normal?
No
Yes
gulR_IN32_SetContactTest_Response
gulR_IN32_SetSlmpError_Response
Communication test response frame creation processing
SLMP error response frame creation processing
Send data size excluding DCS/FCS ← Return value of "Communication test response frame creation processing"
Send data size excluding DCS/FCS ← Return value of "SLMP error response frame creation processing"
Transient1 response frame send data size ← Send data size excluding DCS/FCS
Transient1 response frame send startup flag ← ON
End

Figure 6.33 Flowchart for Communication Test Request Frame Receive Processing

6.2.30 Cable test request frame receive processing

This function performs processing when Cable test request frame is received.
The processing is required to support "Cable test" of CC-Link IE Field diagnostic function.

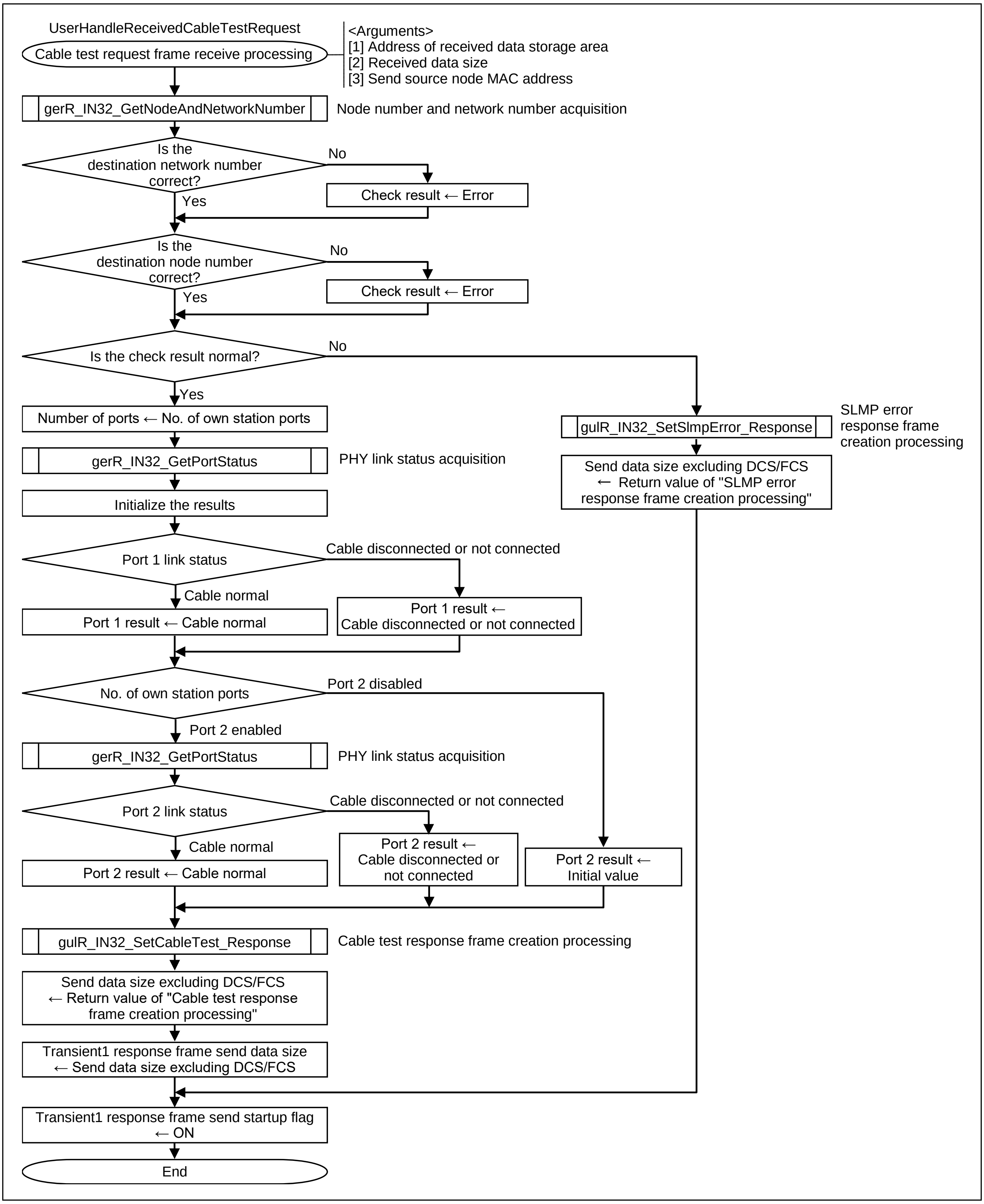

UserHandleReceivedCableTestRequest
Cable test request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Yes
Check result ← Error
Is the destination node number correct?
No
Yes
Check result ← Error
Is the check result normal?
No
Yes
Number of ports ← No. of own station ports
gerR_IN32_GetPortStatus
PHY link status acquisition
Initialize the results
gulR_IN32_SetSlmpError_Response
SLMP error response frame creation processing
Send data size excluding DCS/FCS ← Return value of "SLMP error response frame creation processing"
Port 1 link status
Cable disconnected or not connected
Cable normal
Port 1 result ← Cable normal
Port 1 result ← Cable disconnected or not connected
No. of own station ports
Port 2 disabled
Port 2 enabled
gerR_IN32_GetPortStatus
PHY link status acquisition
Port 2 link status
Cable disconnected or not connected
Cable normal
Port 2 result ← Cable normal
Port 2 result ← Cable disconnected or not connected
Port 2 result ← Initial value
gulR_IN32_SetCableTest_Response
Cable test response frame creation processing
Send data size excluding DCS/FCS ← Return value of "Cable test response frame creation processing"
Transient1 response frame send data size ← Send data size excluding DCS/FCS
Transient1 response frame send startup flag ← ON
End

Figure 6.34 Flowchart for Cable Test Request Frame Receive Processing

6.2.31 Transient2 receive data processing

This function analyzes a received Transient2 frame and creates or receives a response frame in accordance with the analysis results.

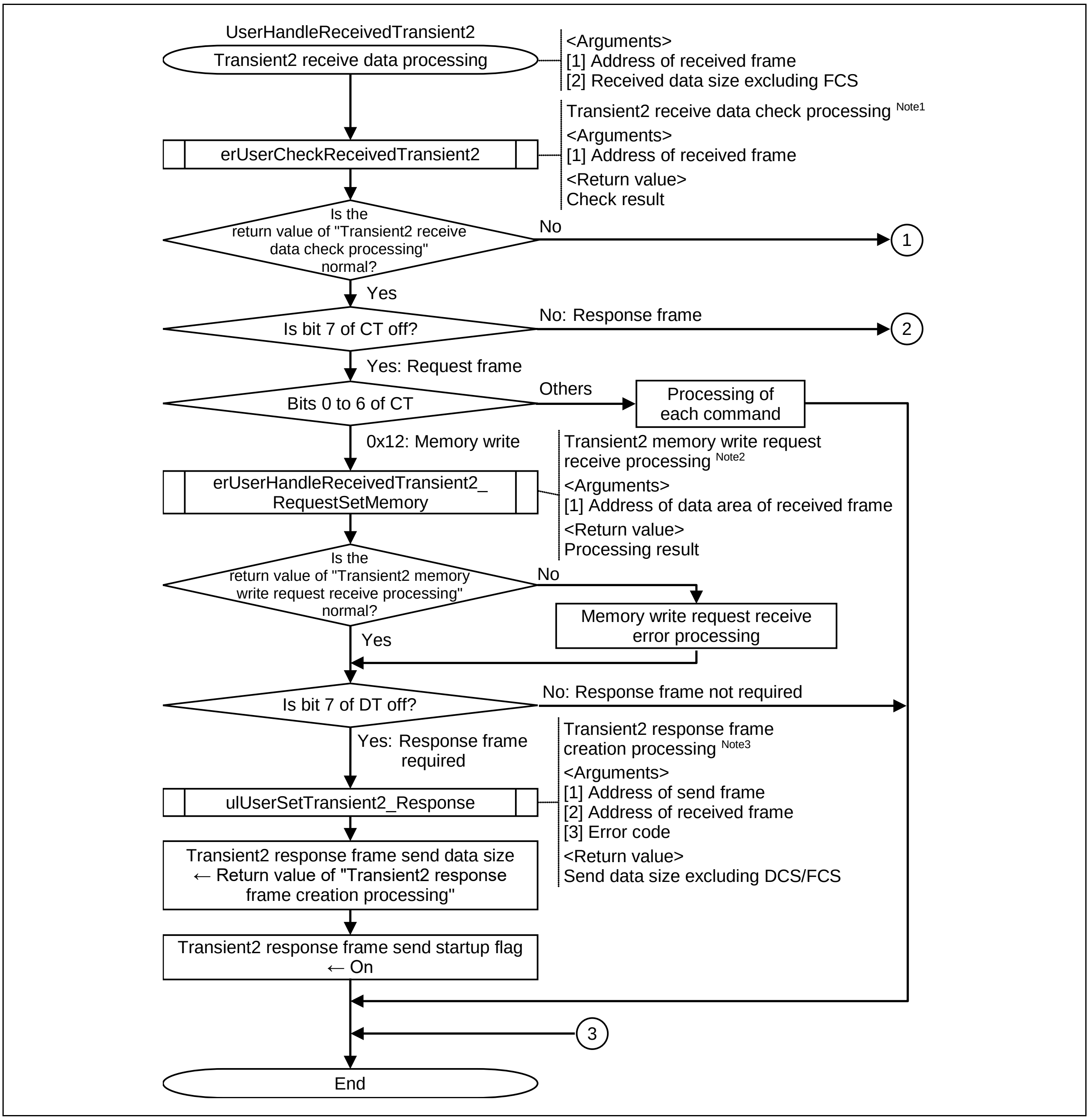

UserHandleReceivedTransient2
Transient2 receive data processing
<Arguments>
[1] Address of received frame
[2] Received data size excluding FCS
Transient2 receive data check processing Note1
<Arguments>
[1] Address of received frame
<Return value>
Check result
erUserCheckReceivedTransient2
Is the return value of "Transient2 receive data check processing" normal?
No
1
Yes
Is bit 7 of CT off?
No: Response frame
2
Yes: Request frame
Bits 0 to 6 of CT
Others
Processing of each command
0x12: Memory write
Transient2 memory write request receive processing Note2
<Arguments>
[1] Address of data area of received frame
<Return value>
Processing result
erUserHandleReceivedTransient2_RequestSetMemory
Is the return value of "Transient2 memory write request receive processing" normal?
No
Memory write request receive error processing
Yes
Is bit 7 of DT off?
No: Response frame not required
Yes: Response frame required
Transient2 response frame creation processing Note3
<Arguments>
[1] Address of send frame
[2] Address of received frame
[3] Error code
<Return value>
Send data size excluding DCS/FCS
ulUserSetTransient2_Response
Transient2 response frame send data size ← Return value of "Transient2 response frame creation processing"
Transient2 response frame send startup flag ← On
3
End

Figure 6.35 Flowchart for Transient2 Receive Data Processing (1/2)

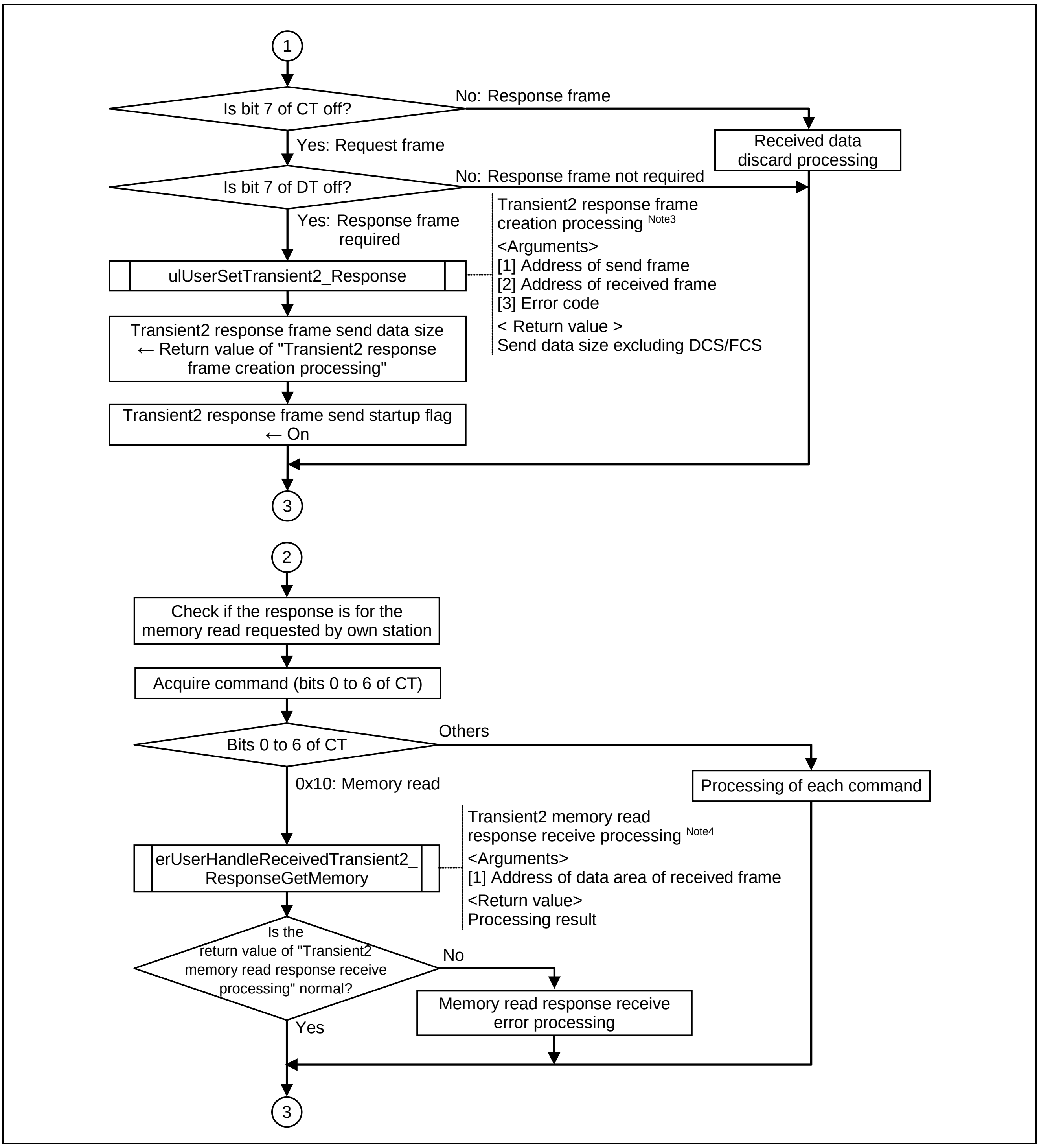

1
Is bit 7 of CT off?
No: Response frame
Yes: Request frame
Received data discard processing
Is bit 7 of DT off?
No: Response frame not required
Yes: Response frame required
ulUserSetTransient2_Response
Transient2 response frame creation processing Note3
<Arguments>
[1] Address of send frame
[2] Address of received frame
[3] Error code
< Return value >
Send data size excluding DCS/FCS
Transient2 response frame send data size ← Return value of "Transient2 response frame creation processing"
Transient2 response frame send startup flag ← On
3
2
Check if the response is for the memory read requested by own station
Acquire command (bits 0 to 6 of CT)
Bits 0 to 6 of CT
Others
Processing of each command
0x10: Memory read
erUserHandleReceivedTransient2_ResponseGetMemory
Transient2 memory read response receive processing Note4
<Arguments>
[1] Address of data area of received frame
<Return value>
Processing result
Is the return value of "Transient2 memory read response receive processing" normal?
No
Yes
Memory read response receive error processing
3

Figure 6.35 Flowchart for Transient2 Receive Data Processing (2/2)

Note 1. For details, refer to Section 6.2.32 "Transient2 receive data check processing".
2. For details, refer to Section 6.2.37 "Transient2 memory write request receive processing".
3. For details, refer to Section 6.2.35 "Transient2 response frame creation processing".
4. For details, refer to Section 6.2.38 "Transient2 memory read response receive processing".

6.2.32 Transient2 receive data check processing

This function checks if the received Transient2 frame is addressed to the own station by checking the destination node number (DA/DS (Destination Address No./Destination Station No.)) and destination network number (DNA (Destination Network Address)).

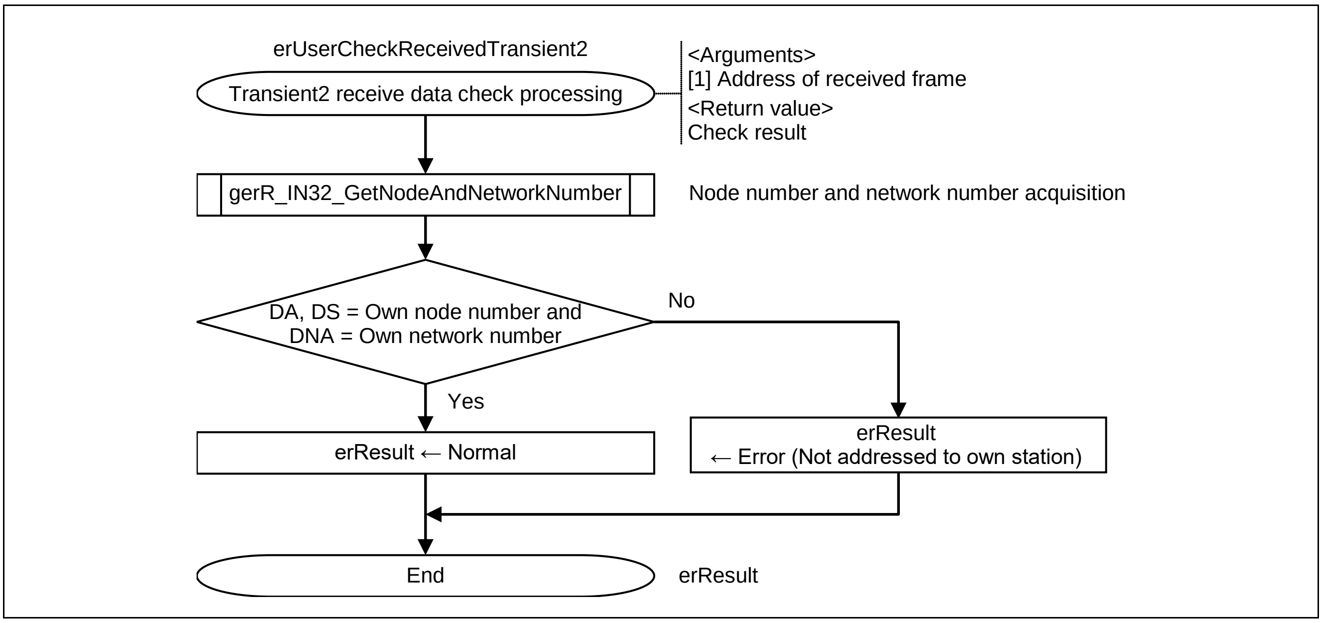

erUserCheckReceivedTransient2
Transient2 receive data check processing
<Arguments>
[1] Address of received frame
<Return value>
Check result
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
DA, DS = Own node number and DNA = Own network number
No
Yes
erResult ← Normal
erResult ← Error (Not addressed to own station)
End
erResult

Figure 6.36 Flowchart for Transient2 Receive Data Check Processing

6.2.33 TransientAck receive data processing

This function processes the received TransientAck frame.

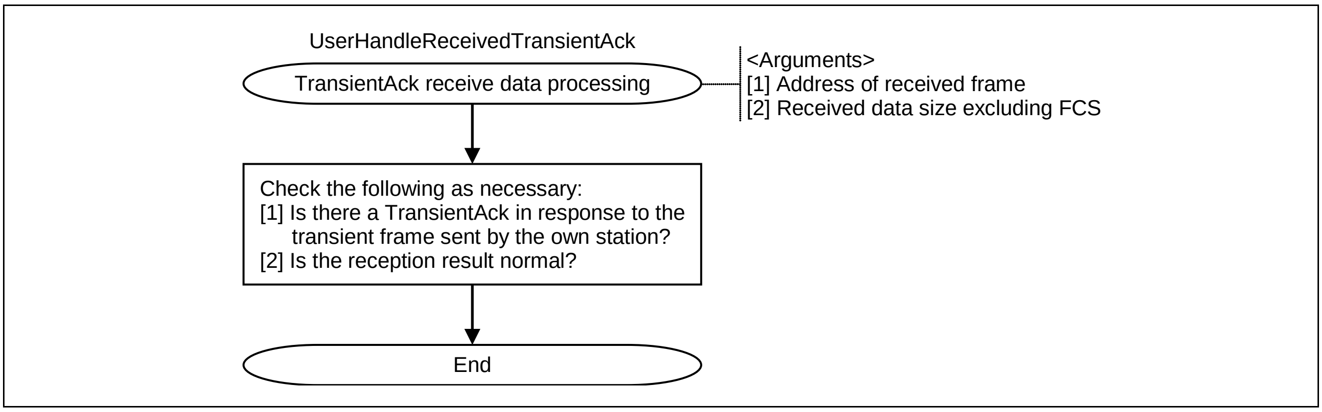

UserHandleReceivedTransientAck
TransientAck receive data processing
<Arguments>
[1] Address of received frame
[2] Received data size excluding FCS
Check the following as necessary:
[1] Is there a TransientAck in response to the transient frame sent by the own station?
[2] Is the reception result normal?
End

Figure 6.37 Flowchart for TransientAck Receive Data Processing

6.2.34 TransientAck frame creation processing

This function creates TransientAck frame.

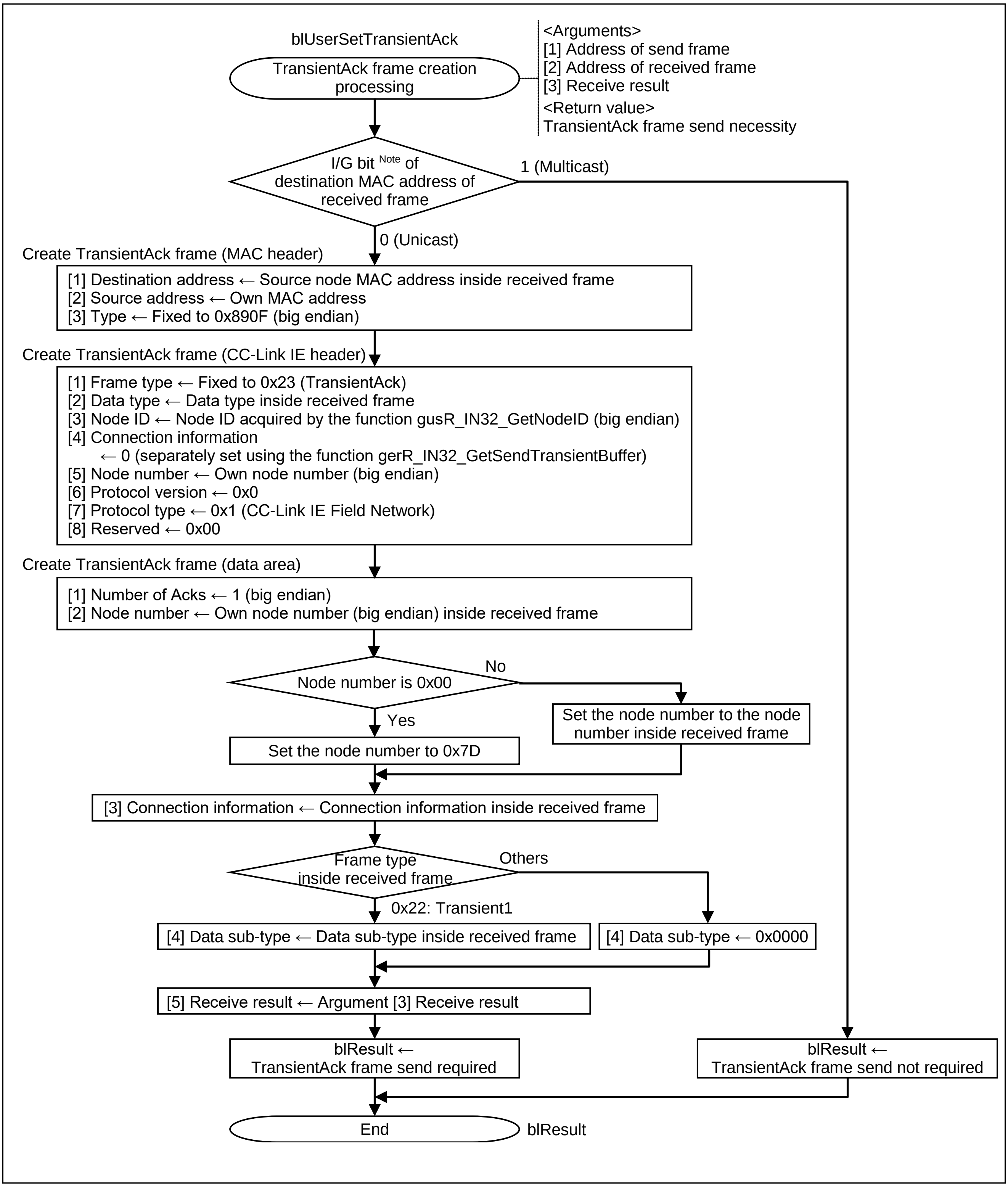

blUserSetTransientAck
TransientAck frame creation processing
<Arguments>
[1] Address of send frame
[2] Address of received frame
[3] Receive result
<Return value>
TransientAck frame send necessity
I/G bit Note of destination MAC address of received frame
1 (Multicast)
0 (Unicast)
Create TransientAck frame (MAC header)
[1] Destination address ← Source node MAC address inside received frame
[2] Source address ← Own MAC address
[3] Type ← Fixed to 0x890F (big endian)
Create TransientAck frame (CC-Link IE header)
[1] Frame type ← Fixed to 0x23 (TransientAck)
[2] Data type ← Data type inside received frame
[3] Node ID ← Node ID acquired by the function gusR_IN32_GetNodeID (big endian)
[4] Connection information
← 0 (separately set using the function gerR_IN32_GetSendTransientBuffer)
[5] Node number ← Own node number (big endian)
[6] Protocol version ← 0x0
[7] Protocol type ← 0x1 (CC-Link IE Field Network)
[8] Reserved ← 0x00
Create TransientAck frame (data area)
[1] Number of Acks ← 1 (big endian)
[2] Node number ← Own node number (big endian) inside received frame
Node number is 0x00
No
Yes
Set the node number to the node number inside received frame
Set the node number to 0x7D
[3] Connection information ← Connection information inside received frame
Frame type inside received frame
Others
0x22: Transient1
[4] Data sub-type ← Data sub-type inside received frame
[4] Data sub-type ← 0x0000
[5] Receive result ← Argument [3] Receive result
blResult ← TransientAck frame send required
blResult ← TransientAck frame send not required
End
blResult

Figure 6.38 Flowchart for TransientAck Frame Creation Processing

Note. The I/G bit is the least significant bit of the first byte (octet) of the MAC address.

6.2.35 Transient2 response frame creation processing

This function creates Transient2 response frame.

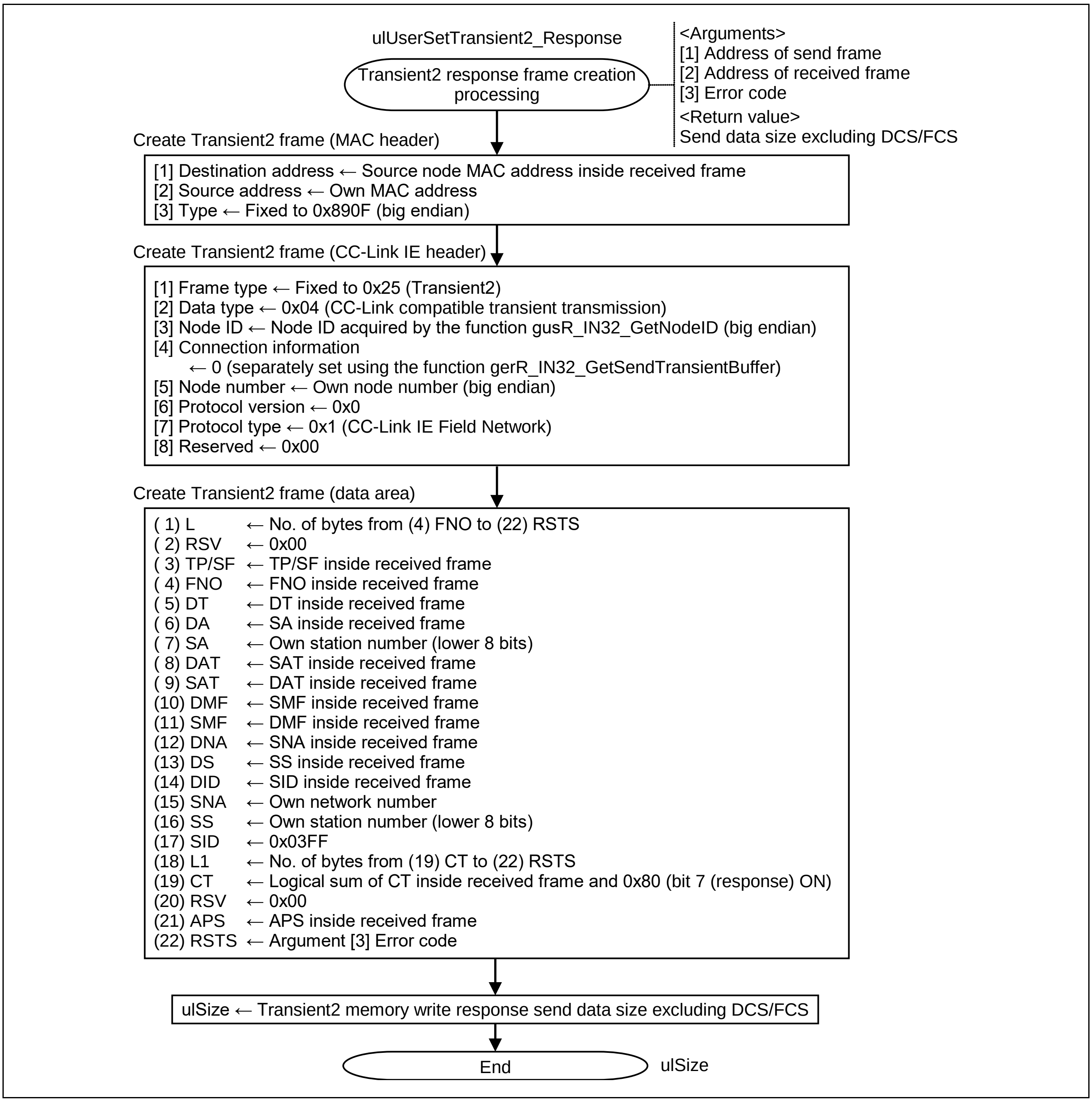

ulUserSetTransient2_Response
Transient2 response frame creation processing
<Arguments>
[1] Address of send frame
[2] Address of received frame
[3] Error code
<Return value>
Send data size excluding DCS/FCS
Create Transient2 frame (MAC header)
[1] Destination address ← Source node MAC address inside received frame
[2] Source address ← Own MAC address
[3] Type ← Fixed to 0x890F (big endian)
Create Transient2 frame (CC-Link IE header)
[1] Frame type ← Fixed to 0x25 (Transient2)
[2] Data type ← 0x04 (CC-Link compatible transient transmission)
[3] Node ID ← Node ID acquired by the function gusR_IN32_GetNodeID (big endian)
[4] Connection information
← 0 (separately set using the function gerR_IN32_GetSendTransientBuffer)
[5] Node number ← Own node number (big endian)
[6] Protocol version ← 0x0
[7] Protocol type ← 0x1 (CC-Link IE Field Network)
[8] Reserved ← 0x00
Create Transient2 frame (data area)
(1) L ← No. of bytes from (4) FNO to (22) RSTS
(2) RSV ← 0x00
(3) TP/SF ← TP/SF inside received frame
(4) FNO ← FNO inside received frame
(5) DT ← DT inside received frame
(6) DA ← SA inside received frame
(7) SA ← Own station number (lower 8 bits)
(8) DAT ← SAT inside received frame
(9) SAT ← DAT inside received frame
(10) DMF ← SMF inside received frame
(11) SMF ← DMF inside received frame
(12) DNA ← SNA inside received frame
(13) DS ← SS inside received frame
(14) DID ← SID inside received frame
(15) SNA ← Own network number
(16) SS ← Own station number (lower 8 bits)
(17) SID ← 0x03FF
(18) L1 ← No. of bytes from (19) CT to (22) RSTS
(19) CT ← Logical sum of CT inside received frame and 0x80 (bit 7 (response) ON)
(20) RSV ← 0x00
(21) APS ← APS inside received frame
(22) RSTS ← Argument [3] Error code
ulSize ← Transient2 memory write response send data size excluding DCS/FCS
End
ulSize

Figure 6.39 Flowchart for Transient2 Response Frame Creation Processing

6.2.36 Transient2 memory read request frame creation processing

This function creates Transient2 memory read request frame.

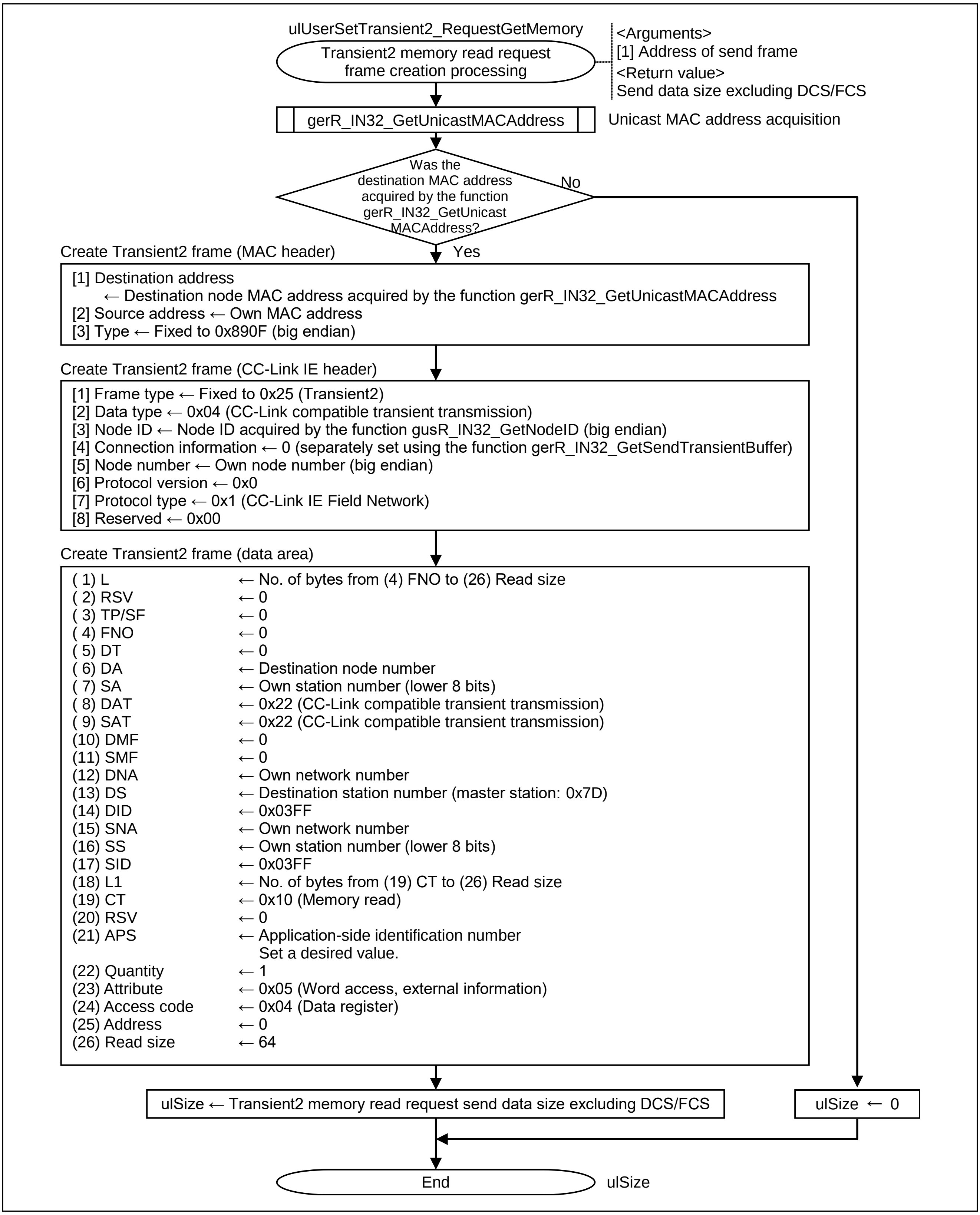

ulUserSetTransient2_RequestGetMemory
Transient2 memory read request frame creation processing
<Arguments>
[1] Address of send frame
<Return value>
Send data size excluding DCS/FCS
gerR_IN32_GetUnicastMACAddress
Unicast MAC address acquisition
Was the destination MAC address acquired by the function gerR_IN32_GetUnicast MACAddress?
No
Yes
Create Transient2 frame (MAC header)
[1] Destination address
← Destination node MAC address acquired by the function gerR_IN32_GetUnicastMACAddress
[2] Source address ← Own MAC address
[3] Type ← Fixed to 0x890F (big endian)
Create Transient2 frame (CC-Link IE header)
[1] Frame type ← Fixed to 0x25 (Transient2)
[2] Data type ← 0x04 (CC-Link compatible transient transmission)
[3] Node ID ← Node ID acquired by the function gusR_IN32_GetNodeID (big endian)
[4] Connection information ← 0 (separately set using the function gerR_IN32_GetSendTransientBuffer)
[5] Node number ← Own node number (big endian)
[6] Protocol version ← 0x0
[7] Protocol type ← 0x1 (CC-Link IE Field Network)
[8] Reserved ← 0x00
Create Transient2 frame (data area)
(1) L ← No. of bytes from (4) FNO to (26) Read size
(2) RSV ← 0
(3) TP/SF ← 0
(4) FNO ← 0
(5) DT ← 0
(6) DA ← Destination node number
(7) SA ← Own station number (lower 8 bits)
(8) DAT ← 0x22 (CC-Link compatible transient transmission)
(9) SAT ← 0x22 (CC-Link compatible transient transmission)
(10) DMF ← 0
(11) SMF ← 0
(12) DNA ← Own network number
(13) DS ← Destination station number (master station: 0x7D)
(14) DID ← 0x03FF
(15) SNA ← Own network number
(16) SS ← Own station number (lower 8 bits)
(17) SID ← 0x03FF
(18) L1 ← No. of bytes from (19) CT to (26) Read size
(19) CT ← 0x10 (Memory read)
(20) RSV ← 0
(21) APS ← Application-side identification number
Set a desired value.
(22) Quantity ← 1
(23) Attribute ← 0x05 (Word access, external information)
(24) Access code ← 0x04 (Data register)
(25) Address ← 0
(26) Read size ← 64
ulSize ← Transient2 memory read request send data size excluding DCS/FCS
ulSize ← 0
End
ulSize

Figure 6.40 Flowchart for Transient2 Memory Read Request Frame Creation Processing

This flowchart describes the following processing in the memory read function.

(1) Destination node number	0x7D (Master station)
(2) Access code	0x04 (Data register)
(3) Attribute	0x05 (Word access, external information)
(4) Address	0
(5) Read size	64

6.2.37 Transient2 memory write request receive processing

This function performs processing when Transient2 memory write request frame is received.

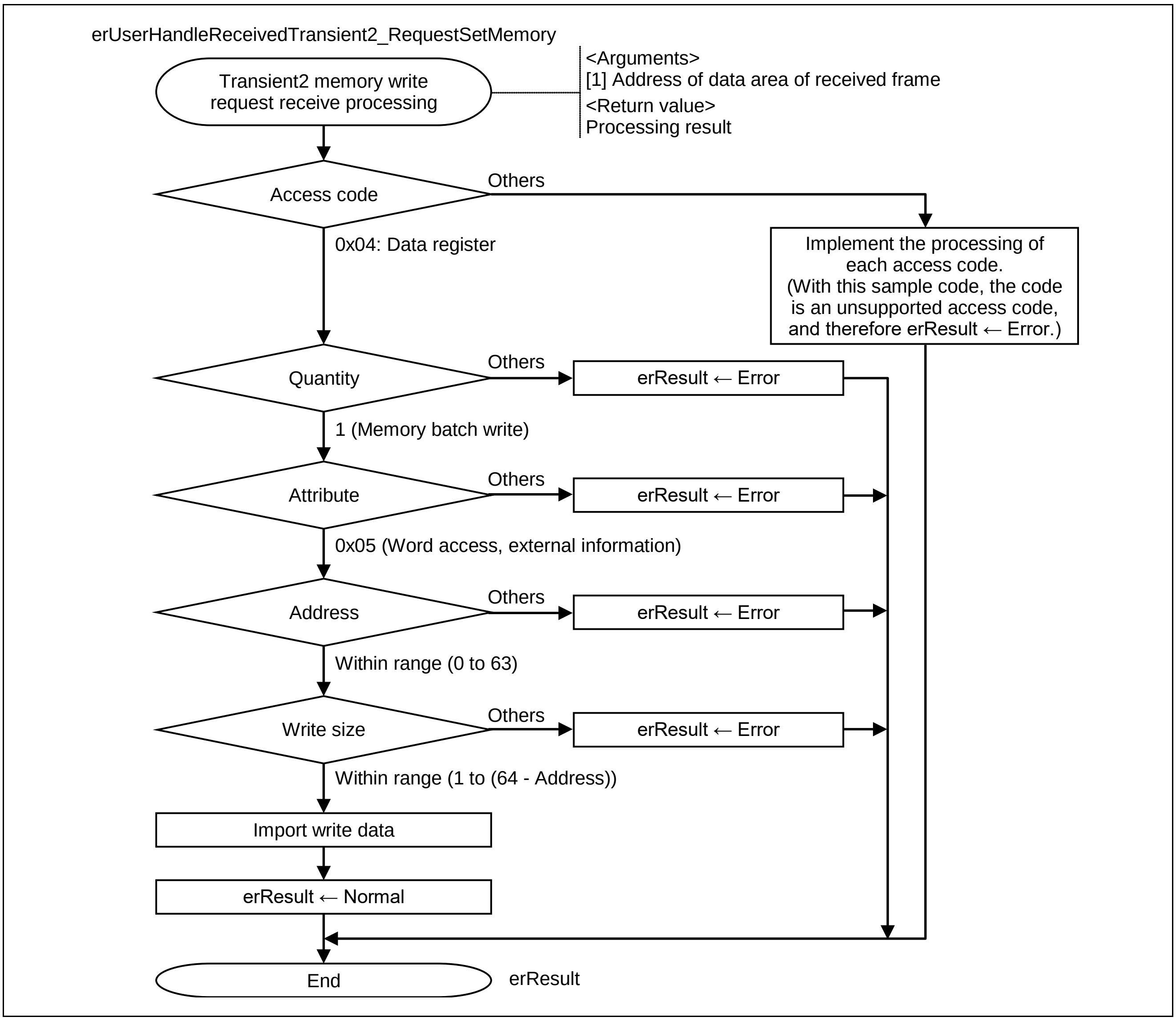

erUserHandleReceivedTransient2_RequestSetMemory
Transient2 memory write request receive processing
<Arguments>
[1] Address of data area of received frame
<Return value>
Processing result
Access code
Others
0x04: Data register
Implement the processing of each access code.
(With this sample code, the code is an unsupported access code, and therefore erResult ← Error.)
Quantity
Others
erResult ← Error
1 (Memory batch write)
Attribute
Others
erResult ← Error
0x05 (Word access, external information)
Address
Others
erResult ← Error
Within range (0 to 63)
Write size
Others
erResult ← Error
Within range (1 to (64 - Address))
Import write data
erResult ← Normal
End
erResult

Figure 6.41 Flowchart for Transient2 Memory Write Request Receive Processing

This flowchart describes the following processing in the memory write function.

Any other processing results in error.

(1) Memory batch write

(2) Access code	0x04 (Data register)
(3) Attribute	0x05 (Word access, external information)
(4) Address	0 to 63
(5) Write size	1 to (64 - Address)

6.2.38 Transient2 memory read response receive processing

This function performs processing when Transient2 memory read request frame is received.

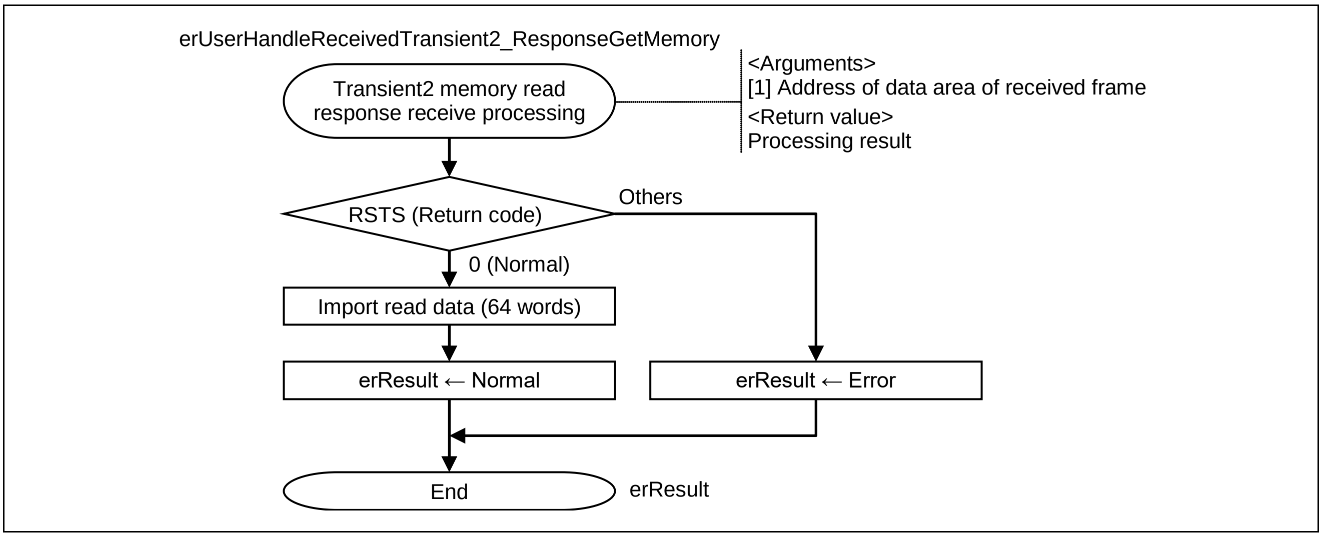

erUserHandleReceivedTransient2_ResponseGetMemory
Transient2 memory read response receive processing
<Arguments>
[1] Address of data area of received frame
<Return value>
Processing result
RSTS (Return code)
Others
0 (Normal)
Import read data (64 words)
erResult ← Normal
erResult ← Error
End
erResult

Figure 6.42 Flowchart for Transient2 Memory Read Response Receive Processing

This flowchart describes the receive processing for the following requests in the memory read function.

(1) Destination node number	0x7D (Master station)
(2) Access code	0x04 (Data register)
(3) Attribute	0x05 (Word access, external information)
(4) Address	0
(5) Read size	64

6.2.39 SLMP memory read request frame receive processing

This function performs processing when SLMP memory read request frame is received.

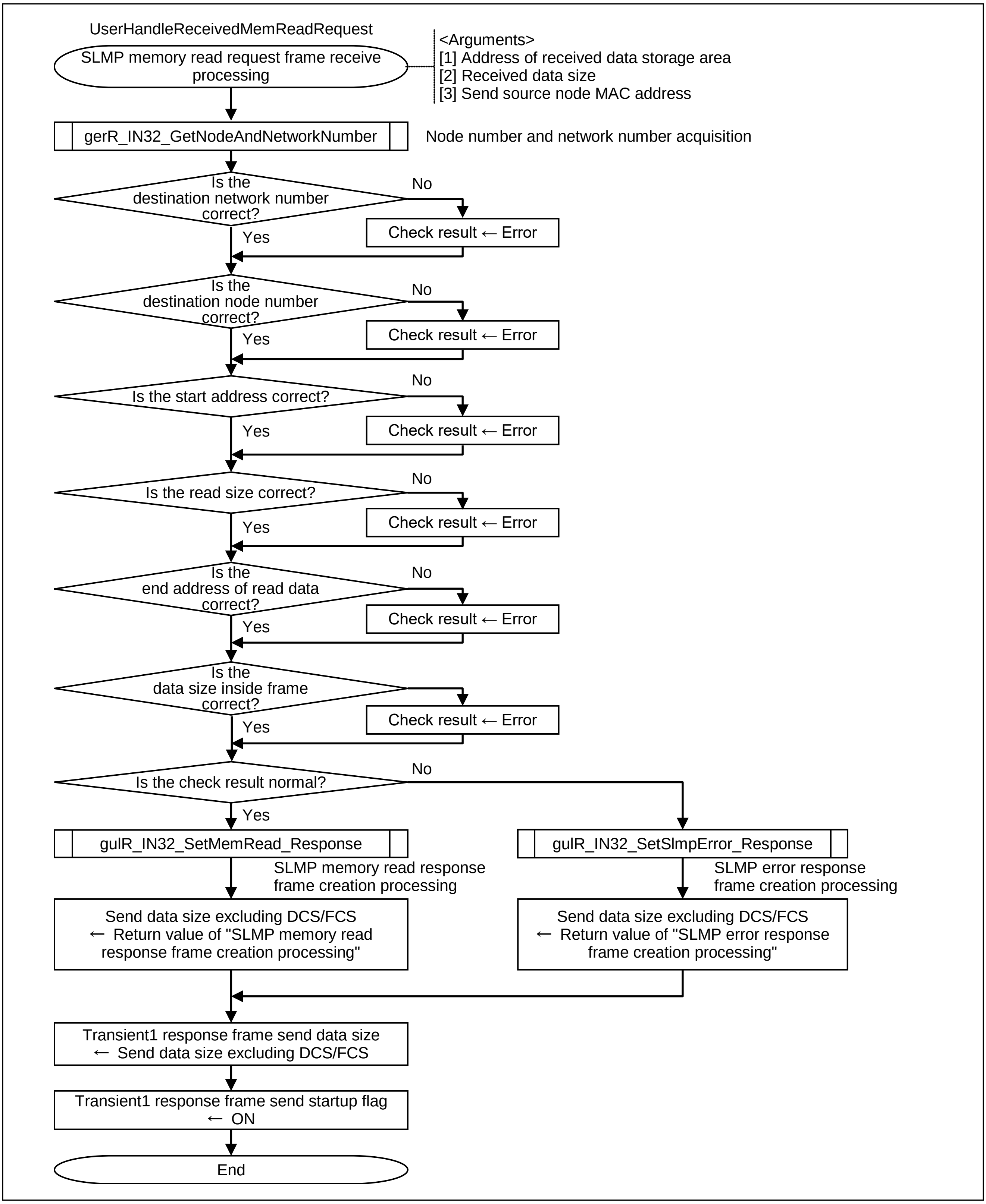

UserHandleReceivedMemReadRequest
SLMP memory read request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Check result ← Error
Yes
Is the destination node number correct?
No
Check result ← Error
Yes
Is the start address correct?
No
Check result ← Error
Yes
Is the read size correct?
No
Check result ← Error
Yes
Is the end address of read data correct?
No
Check result ← Error
Yes
Is the data size inside frame correct?
Check result ← Error
Yes
Is the check result normal?
No
Yes
gulR_IN32_SetMemRead_Response
SLMP memory read response frame creation processing
gulR_IN32_SetSlmpError_Response
SLMP error response frame creation processing
Send data size excluding DCS/FCS ← Return value of "SLMP memory read response frame creation processing"
Send data size excluding DCS/FCS ← Return value of "SLMP error response frame creation processing"
Transient1 response frame send data size ← Send data size excluding DCS/FCS
Transient1 response frame send startup flag ← ON
End

Figure 6.43 Flowchart for SLMP Memory Read Request Frame Receive Processing

6.2.40 SLMP memory write request frame receive processing

This function performs processing when SLMP memory write request frame is received.

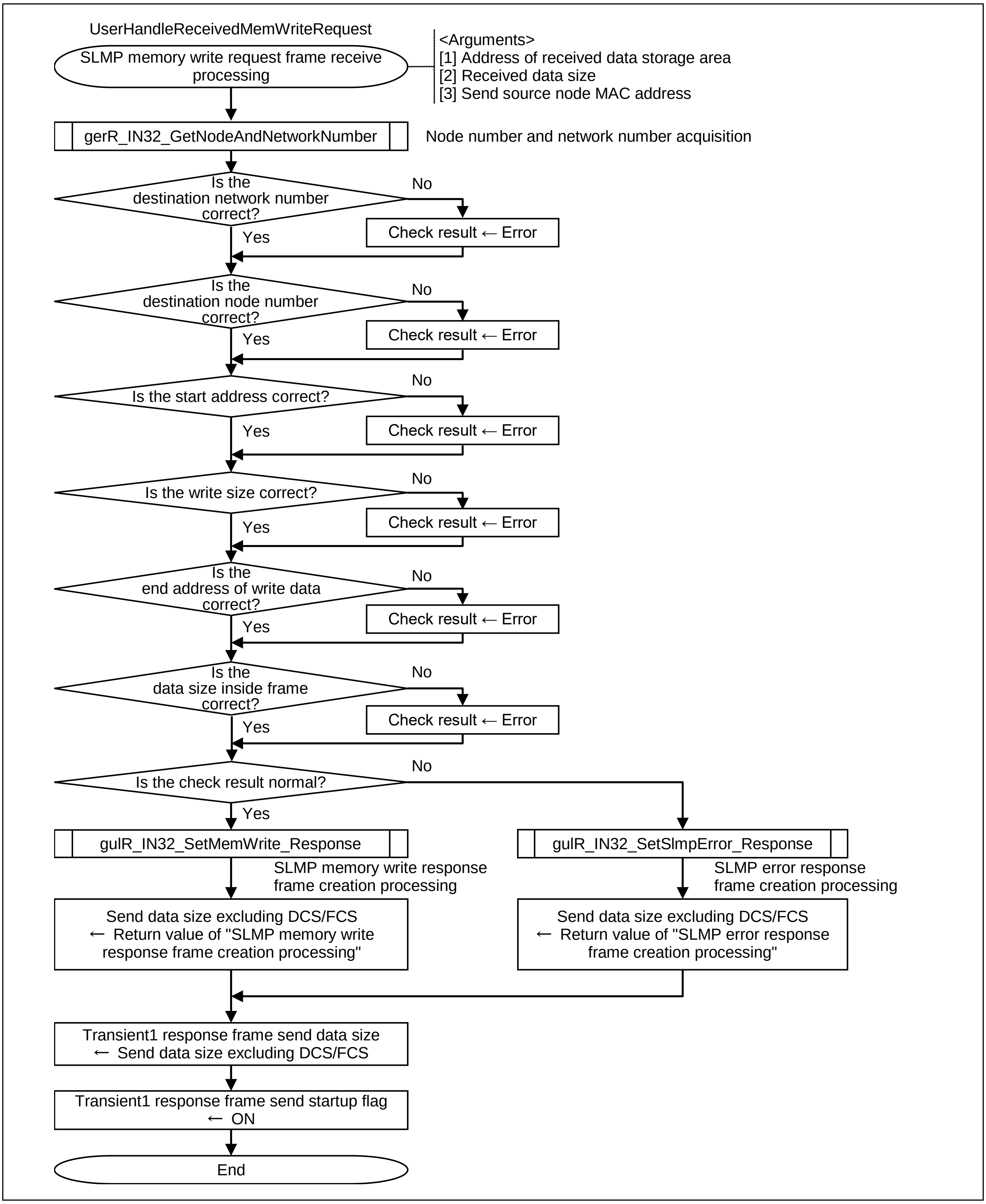

UserHandleReceivedMemWriteRequest
SLMP memory write request frame receive processing
<Arguments>
[1] Address of received data storage area
[2] Received data size
[3] Send source node MAC address
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Is the destination network number correct?
No
Yes
Check result ← Error
Is the destination node number correct?
No
Yes
Check result ← Error
Is the start address correct?
No
Yes
Check result ← Error
Is the write size correct?
No
Yes
Check result ← Error
Is the end address of write data correct?
No
Yes
Check result ← Error
Is the data size inside frame correct?
No
Yes
Check result ← Error
Is the check result normal?
No
Yes
gulR_IN32_SetMemWrite_Response
SLMP memory write response frame creation processing
gulR_IN32_SetSlmpError_Response
SLMP error response frame creation processing
Send data size excluding DCS/FCS ← Return value of "SLMP memory write response frame creation processing"
Send data size excluding DCS/FCS ← Return value of "SLMP error response frame creation processing"
Transient1 response frame send data size ← Send data size excluding DCS/FCS
Transient1 response frame send startup flag ← ON
End

Figure 6.44 Flowchart for SLMP Memory Write Request Frame Receive Processing

6.2.41 SLMP memory read request frame creation processing

This function creates SLMP memory read request frame to be sent to another station. This processing is an example of the processing for creating SLMP request frame.

For other commands, add processing as required.

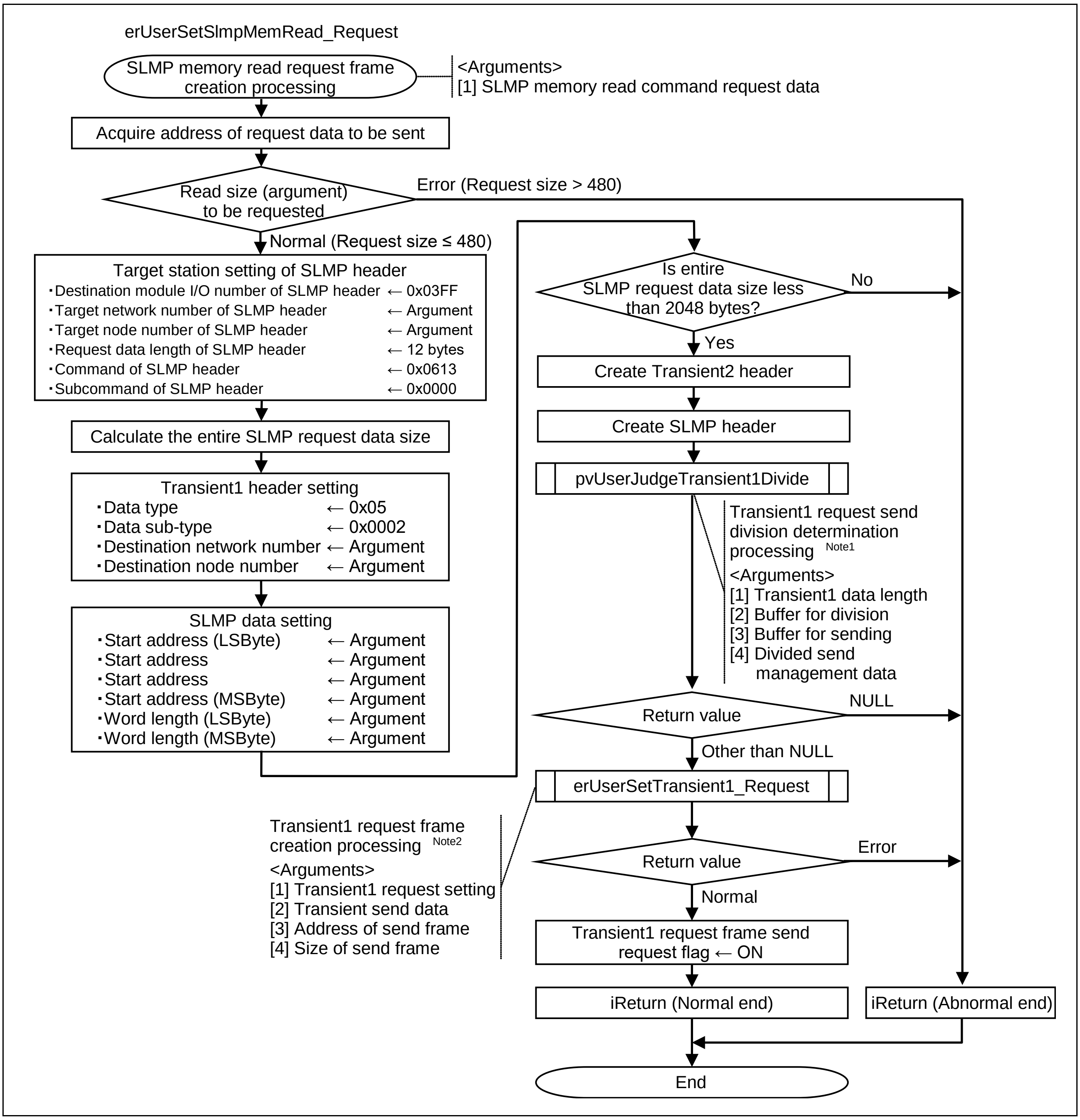

erUserSetSlmpMemRead_Request
SLMP memory read request frame creation processing
<Arguments>
[1] SLMP memory read command request data
Acquire address of request data to be sent
Read size (argument) to be requested
Error (Request size > 480)
Normal (Request size ≤ 480)
Target station setting of SLMP header
･Destination module I/O number of SLMP header ← 0x03FF
･Target network number of SLMP header ← Argument
･Target node number of SLMP header ← Argument
･Request data length of SLMP header ← 12 bytes
･Command of SLMP header ← 0x0613
･Subcommand of SLMP header ← 0x0000
Calculate the entire SLMP request data size
Transient1 header setting
･Data type ← 0x05
･Data sub-type ← 0x0002
･Destination network number ← Argument
･Destination node number ← Argument
SLMP data setting
･Start address (LSByte) ← Argument
･Start address ← Argument
･Start address ← Argument
･Start address (MSByte) ← Argument
･Word length (LSByte) ← Argument
･Word length (MSByte) ← Argument
Is entire SLMP request data size less than 2048 bytes?
No
Yes
Create Transient2 header
Create SLMP header
pvUserJudgeTransient1Divide
Transient1 request send division determination processing Note1
<Arguments>
[1] Transient1 data length
[2] Buffer for division
[3] Buffer for sending
[4] Divided send management data
Return value
NULL
Other than NULL
erUserSetTransient1_Request
Transient1 request frame creation processing Note2
<Arguments>
[1] Transient1 request setting
[2] Transient send data
[3] Address of send frame
[4] Size of send frame
Return value
Error
Normal
Transient1 request frame send request flag ← ON
iReturn (Normal end)
iReturn (Abnormal end)
End

Figure 6.45 Flowchart for SLMP Memory Read Request Frame Creation Processing

Note 1. For details, refer to Section 6.2.42 "Transient1 request send division determination processing".

2. For details, refer to Section 6.2.43 "Transient1 request frame creation processing".

6.2.42 Transient1 request send division determination processing

This function determines if a frame should be divided prior to sending when creating a Transient1 request frame.

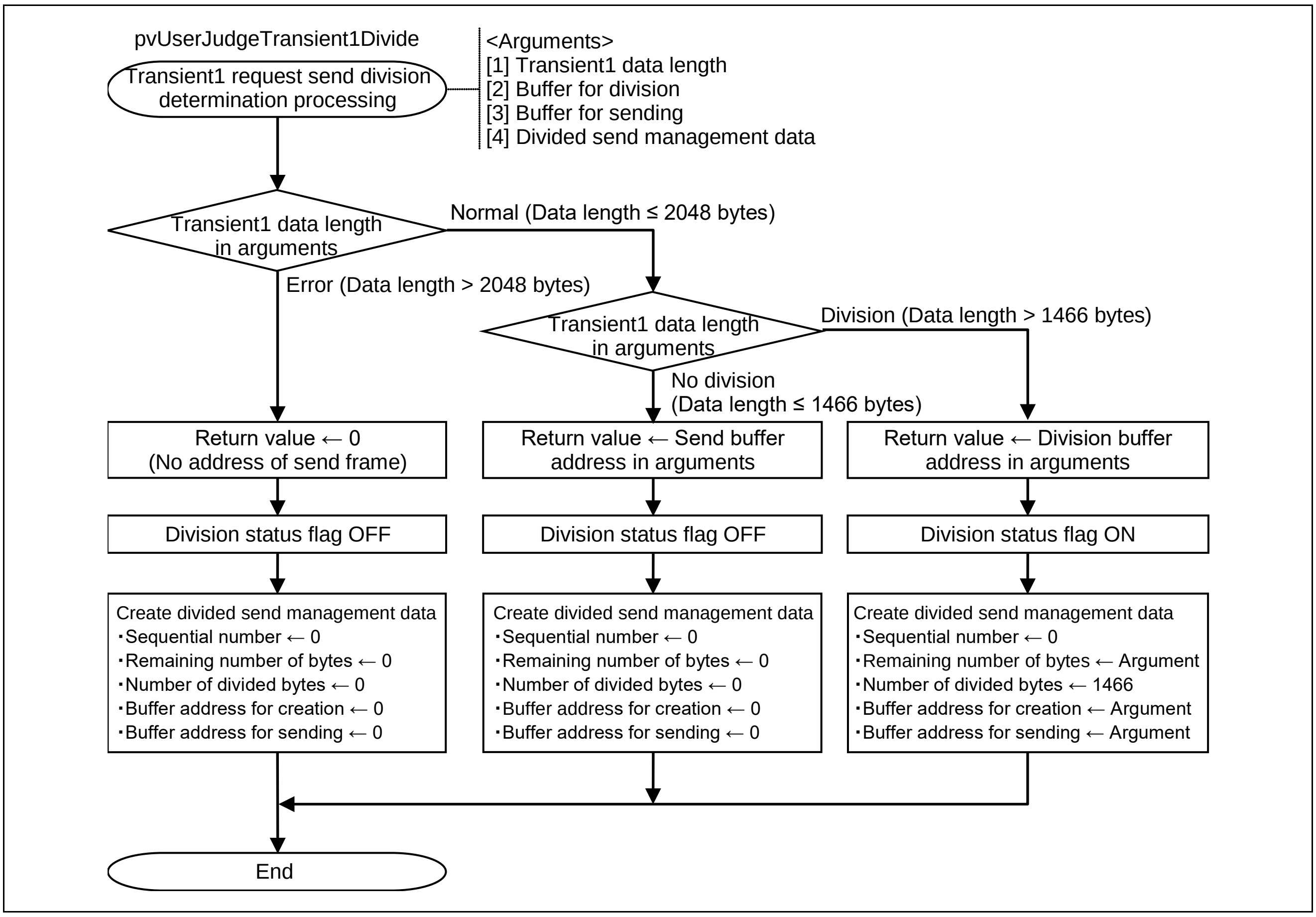

pvUserJudgeTransient1Divide
Transient1 request send division determination processing
<Arguments>
[1] Transient1 data length
[2] Buffer for division
[3] Buffer for sending
[4] Divided send management data
Transient1 data length in arguments
Normal (Data length ≤ 2048 bytes)
Error (Data length > 2048 bytes)
Transient1 data length in arguments
Division (Data length > 1466 bytes)
No division
(Data length ≤ 1466 bytes)
Return value ← 0
(No address of send frame)
Return value ← Send buffer address in arguments
Return value ← Division buffer address in arguments
Division status flag OFF
Division status flag OFF
Division status flag ON
Create divided send management data
・Sequential number ← 0
・Remaining number of bytes ← 0
・Number of divided bytes ← 0
・Buffer address for creation ← 0
・Buffer address for sending ← 0
Create divided send management data
・Sequential number ← 0
・Remaining number of bytes ← 0
・Number of divided bytes ← 0
・Buffer address for creation ← 0
・Buffer address for sending ← 0
Create divided send management data
・Sequential number ← 0
・Remaining number of bytes ← Argument
・Number of divided bytes ← 1466
・Buffer address for creation ← Argument
・Buffer address for sending ← Argument
End

Figure 6.46 Flowchart for Transient1 Request Send Division Determination Processing

6.2.43 Transient1 request frame creation processing

This function creates a request frame (from the MAC header to the Transient1 header) when an SLMP memory read request is sent from the own station to another station.

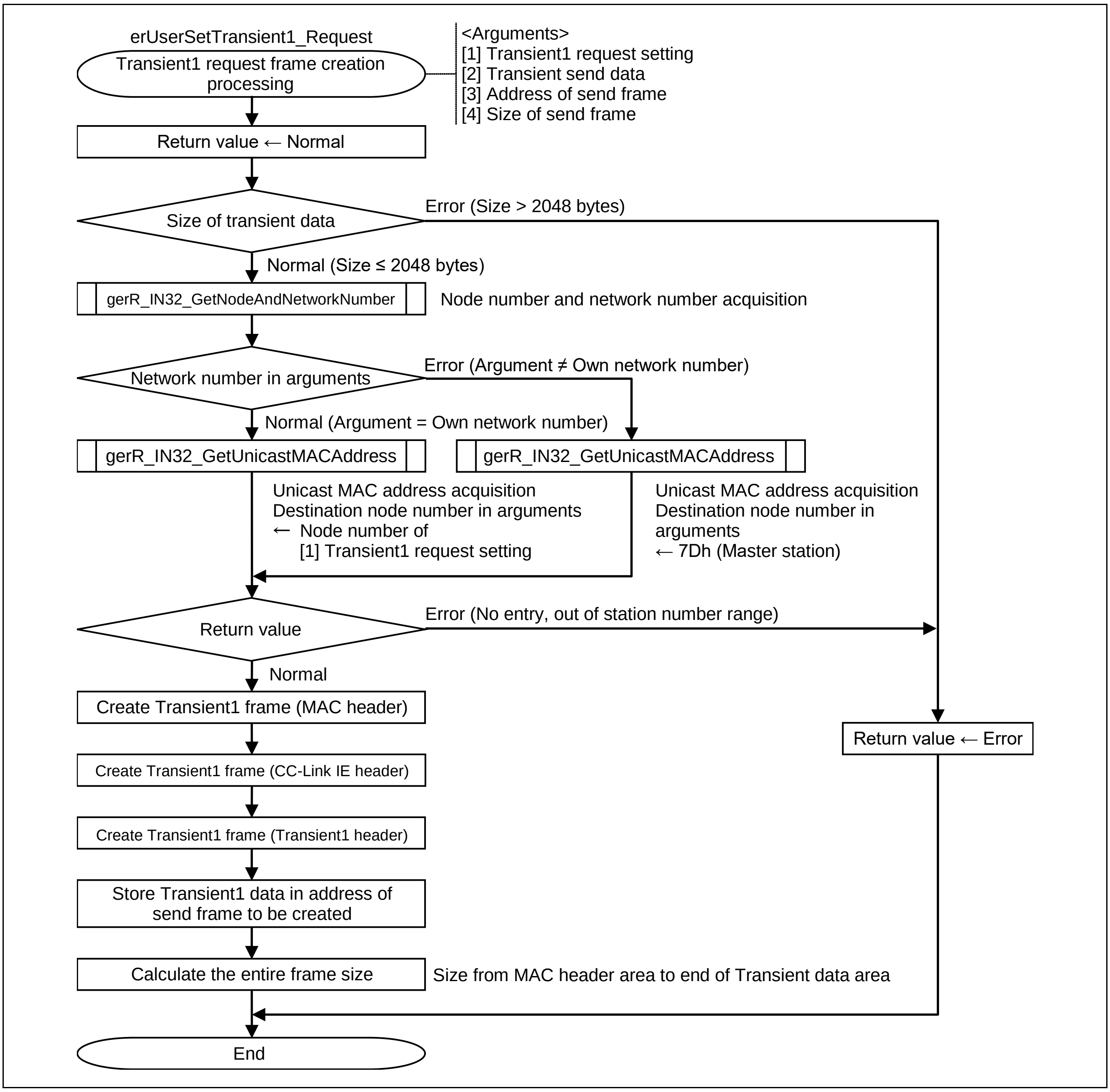

erUserSetTransient1_Request
Transient1 request frame creation processing
<Arguments>
[1] Transient1 request setting
[2] Transient send data
[3] Address of send frame
[4] Size of send frame
Return value ← Normal
Size of transient data
Error (Size > 2048 bytes)
Normal (Size ≤ 2048 bytes)
gerR_IN32_GetNodeAndNetworkNumber
Node number and network number acquisition
Network number in arguments
Error (Argument ≠ Own network number)
Normal (Argument = Own network number)
gerR_IN32_GetUnicastMACAddress
gerR_IN32_GetUnicastMACAddress
Unicast MAC address acquisition
Destination node number in arguments
← Node number of
[1] Transient1 request setting
Unicast MAC address acquisition
Destination node number in arguments
← 7Dh (Master station)
Return value
Error (No entry, out of station number range)
Normal
Create Transient1 frame (MAC header)
Return value ← Error
Create Transient1 frame (CC-Link IE header)
Create Transient1 frame (Transient1 header)
Store Transient1 data in address of send frame to be created
Calculate the entire frame size
Size from MAC header area to end of Transient data area
End

Figure 6.47 Flowchart for Transient1 Request Frame Creation Processing

6.2.44 SLMP memory read response receive processing

This function receives response frames for SLMP memory read requested by the own station to other stations.

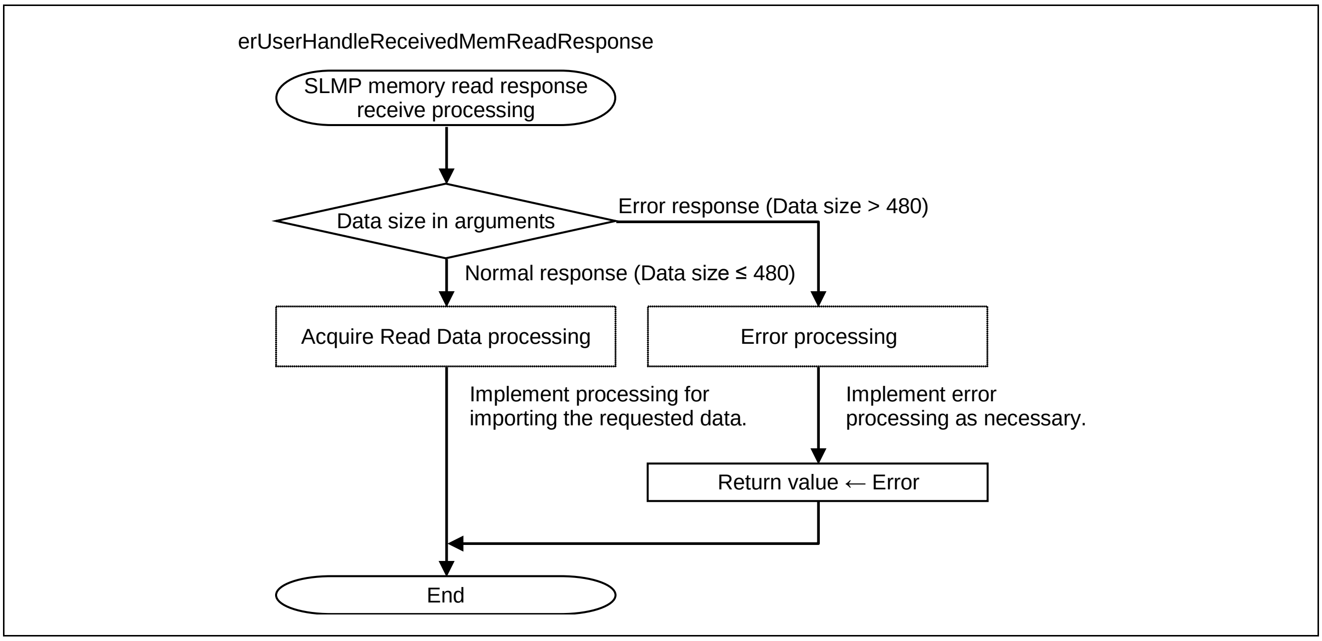

erUserHandleReceivedMemReadResponse
SLMP memory read response receive processing
Data size in arguments
Error response (Data size > 480)
Normal response (Data size ≤ 480)
Acquire Read Data processing
Error processing
Implement processing for importing the requested data.
Implement error processing as necessary.
Return value ← Error
End

Figure 6.48 Flowchart for SLMP Memory Read Response Receive Processing

6.2.45 Hardware test (IEEE 802.3ab compliance test)

This function performs the IEEE 802.3ab compliance test.

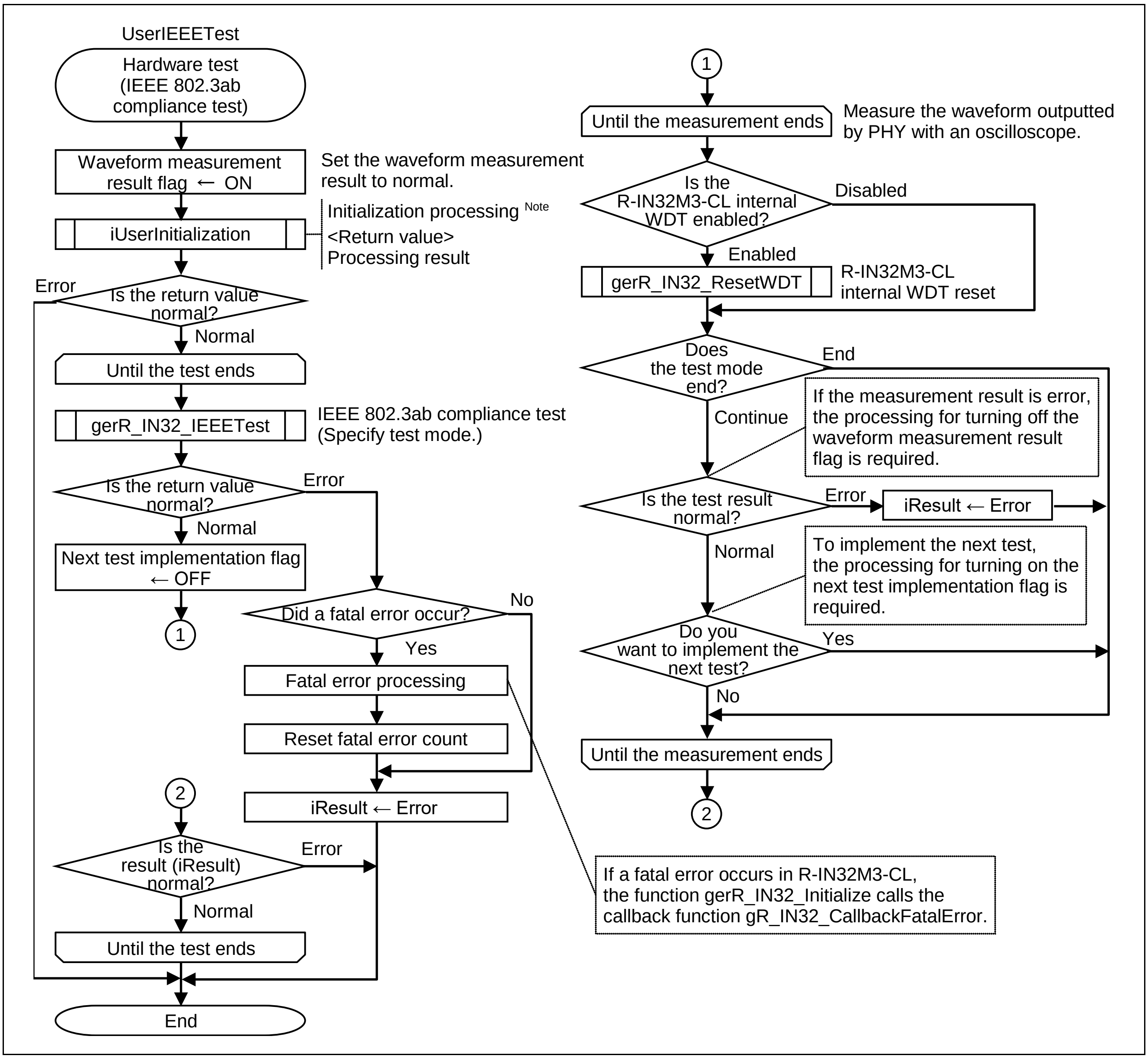

UserIEEETest
Hardware test (IEEE 802.3ab compliance test)
Waveform measurement result flag ← ON
Set the waveform measurement result to normal.
iUserInitialization
Initialization processing Note
<Return value>
Processing result
Error
Is the return value normal?
Normal
Until the test ends
gerR_IN32_IEEETest
IEEE 802.3ab compliance test (Specify test mode.)
Is the return value normal?
Error
Normal
Next test implementation flag ← OFF
1
Did a fatal error occur?
No
Yes
Fatal error processing
Reset fatal error count
iResult ← Error
2
Is the result (iResult) normal?
Error
Normal
Until the test ends
End
1
Until the measurement ends
Measure the waveform outputted by PHY with an oscilloscope.
Is the R-IN32M3-CL internal WDT enabled?
Disabled
Enabled
gerR_IN32_ResetWDT
R-IN32M3-CL internal WDT reset
Does the test mode end?
End
Continue
If the measurement result is error, the processing for turning off the waveform measurement result flag is required.
Is the test result normal?
Error
iResult ← Error
Normal
To implement the next test, the processing for turning on the next test implementation flag is required.
Do you want to implement the next test?
Yes
No
Until the measurement ends
2
If a fatal error occurs in R-IN32M3-CL, the function gerR_IN32_Initialize calls the callback function gR_IN32_CallbackFatalError.

Figure 6.49 Flowchart for Hardware Test (IEEE 802.3ab Compliance Test)

Note. For details, refer to Section 6.2.2 "Initialization processing".

Caution. The function needs to be implemented to implement the tests described in the CC-Link IE Field Network Intelligent Device Station Conformance Test Specifications (BAP-C0401-037).

The function gerR_IN32R_IEEETest (refer to Section 6.5.2 "Creating the R-IN32M3-CL driver target-dependent functions") is called within the gerR_IN32_IEEETest processing. Be sure to customize gerR_IN32R_IEEETest in accordance with the specifications of the PHY used.

6.2.46 Hardware test (loopback communication test)

The loopback communication test involves the internal loopback communication test and external loopback communication test.

Ports that might be failed can be resolved based on each test result.

Table 6.9 Troubleshooting Based on Loopback Communication Test

Target Port Resulting in R_IN32_ERR by Internal Loopback Communication Test (gerR_IN32_InternalLoopBackTest)	Source Port Resulting in R_IN32_ERR by External Loopback Communication Test (gerR_IN32_ExternalLoopBackTest)	Port Suspected of Failure
Port 1	Port 1	Port 1 XMIT
	Port 2	Port 1 RECV
Port 2	Port 1	Port 2 RECV
	Port 2	Port 2 XMIT

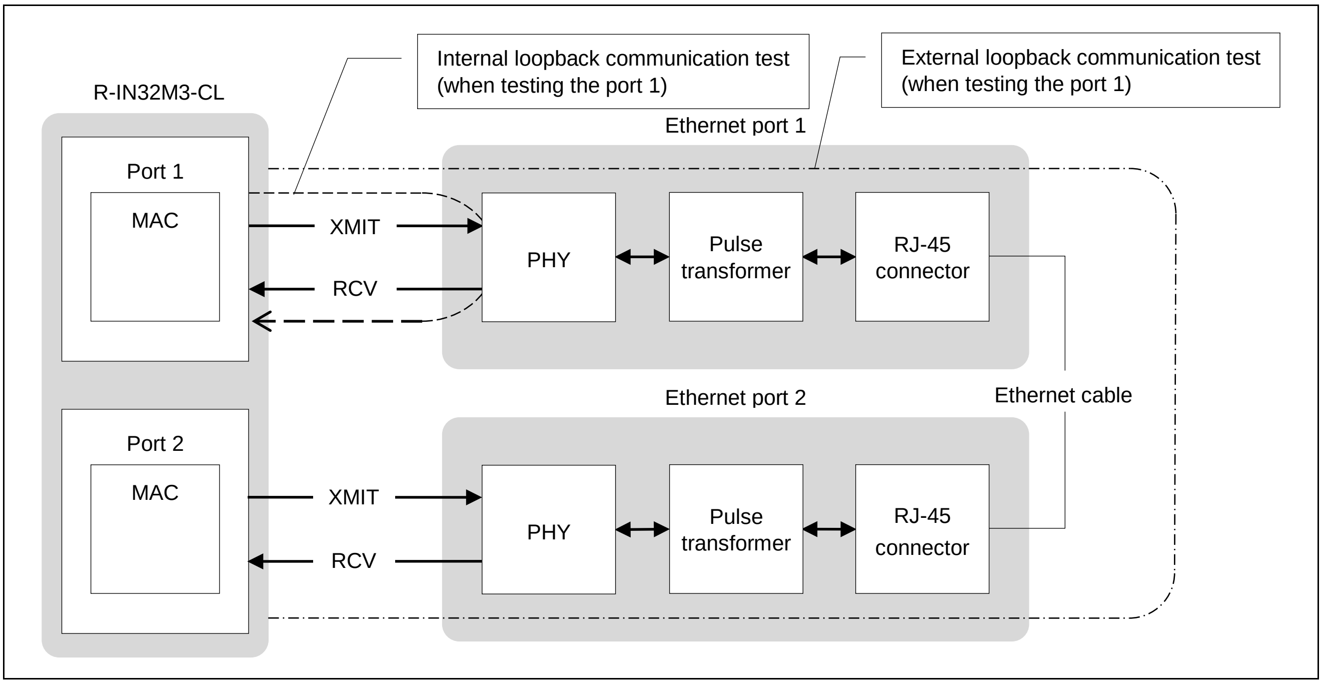


Figure 6.50 Port Schematic Diagram

Implement the test in accordance with the precautions of each test item.

Table 6.10 Test Item Precautions

No.	Test Item	Precautions
1	Internal loopback communication test	・When the internal loopback communication test is implemented, the PHY link shuts down. It takes 3 or more seconds for the PHY link to go up again. Be sure to execute reset processing so that WDT does not time out. (When you use the R-IN32M3-CL internal WDT, call the function gerR_IN32_ResetWDT.) ・Implement the internal loopback communication test as independent processing, not in main processing (iUserMainRoutine). (Example: Separately implement the normal operation mode to start main processing and the internal loopback communication test mode.)
2	External loopback communication test	Connect the port 1 and port 2 using an Ethernet cable.

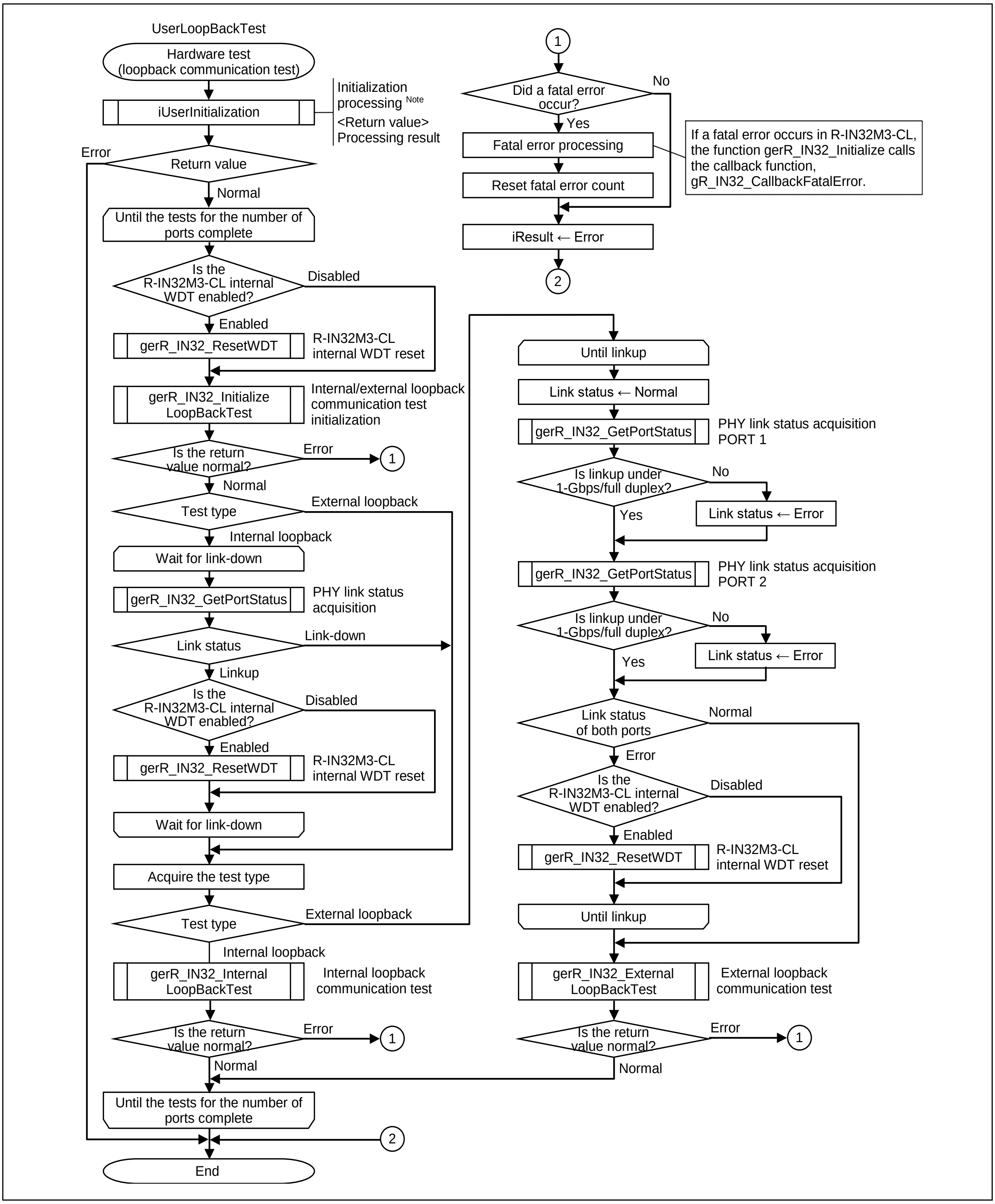

UserLoopBackTest
Hardware test (loopback communication test)
iUserInitialization
Initialization processing Note
<Return value> Processing result
Error
Return value
Normal
Until the tests for the number of ports complete
Is the R-IN32M3-CL internal WDT enabled?
Disabled
Enabled
gerR_IN32_ResetWDT
R-IN32M3-CL internal WDT reset
gerR_IN32_Initialize LoopBackTest
Internal/external loopback communication test initialization
Is the return value normal?
Error
1
Normal
Test type
External loopback
Internal loopback
Wait for link-down
gerR_IN32_GetPortStatus
PHY link status acquisition
Link status
Link-down
Linkup
Is the R-IN32M3-CL internal WDT enabled?
Disabled
Enabled
gerR_IN32_ResetWDT
R-IN32M3-CL internal WDT reset
Wait for link-down
Acquire the test type
Test type
External loopback
Internal loopback
gerR_IN32_Internal LoopBackTest
Internal loopback communication test
Is the return value normal?
Error
1
Normal
Until the tests for the number of ports complete
2
End
1
Did a fatal error occur?
No
Yes
Fatal error processing
If a fatal error occurs in R-IN32M3-CL, the function gerR_IN32_Initialize calls the callback function, gR_IN32_CallbackFatalError.
Reset fatal error count
iResult ← Error
2
Until linkup
Link status ← Normal
gerR_IN32_GetPortStatus
PHY link status acquisition PORT 1
Is linkup under 1-Gbps/full duplex?
No
Yes
Link status ← Error
gerR_IN32_GetPortStatus
PHY link status acquisition PORT 2
Is linkup under 1-Gbps/full duplex?
No
Yes
Link status ← Error
Link status of both ports
Normal
Error
Is the R-IN32M3-CL internal WDT enabled?
Disabled
Enabled
gerR_IN32_ResetWDT
R-IN32M3-CL internal WDT reset
Until linkup
gerR_IN32_External LoopBackTest
External loopback communication test
Is the return value normal?
Error
1
Normal

Figure 6.51 Flowchart for Hardware Test (Loopback Communication Test)

Note. For details, refer to Section 6.2.2 "Initialization processing".

6.3 R-IN32M3-CL Driver Interface Function List

The following lists the interface functions of the R-IN32M3-CL driver.

Table 6.11 R-IN32M3-CL Driver Interface Function List (1/3)

Function Category	Function Name	Function Type	Overview
Initial setup	gulR_IN32_GetResetStatus	ULONG	Reset status acquisition
	gerR_IN32_Initialize	ERRCODE	R-IN32M3-CL initialization
	gerR_IN32_SetNodeAndNetworkNumber	ERRCODE	Node number and network number setting
	gerR_IN32_Start	ERRCODE	R-IN32M3-CL communication start
Watchdog timer	gerR_IN32_ResetWDT	ERRCODE	R-IN32M3-CL internal WDT reset
	gerR_IN32_DisableWDT	ERRCODE	R-IN32M3-CL internal WDT disablement
	gerR_IN32_EnableWDT	ERRCODE	R-IN32M3-CL internal WDT enablement
	gerR_IN32_SetWDT	ERRCODE	R-IN32M3-CL internal WDT time limit setting
Event	gerR_IN32_GetEvent	ERRCODE	R-IN32M3-CL event detection
	gerR_IN32_Main	ERRCODE	R-IN32M3-CL event detection main processing
	gerR_IN32_RestartEvent	ERRCODE	R-IN32M3-CL event restart
	gerR_IN32_UpdatePortStatus	ERRCODE	PHY link status update
	gerR_IN32_UpdateMIB	ERRCODE	MIB information update
Cyclic transmission	gerR_IN32_SetCyclicStop	ERRCODE	Cyclic transmission stop for device-side reasons
	gerR_IN32_ClearCyclicStop	ERRCODE	Cyclic transmission stop clear for device-side reasons
	gerR_IN32_GetReceivedCyclicData	ERRCODE	Cyclic receive data acquisition
	gerR_IN32_GetMasterNodeStatus	ERRCODE	Master station status acquisition
	gerR_IN32_SetMyStatus	ERRCODE	MyStatus send data setting
	gerR_IN32_SetSendCyclicData	ERRCODE	Cyclic send data setting
Own station status setup	gerR_IN32_SetNodeStatus	ERRCODE	Own station status setting
	gerR_IN32_ForceStop	ERRCODE	Own station error setting
Own station status acquisition	gerR_IN32_GetNodeAndNetworkNumber	ERRCODE	Node number and network number acquisition
	gerR_IN32_GetCurrentCyclicSize	ERRCODE	Acquisition of cyclic transmission size specified from master station
	gerR_IN32_GetCommumicationStatus	ERRCODE	Data link status acquisition
	gerR_IN32_GetPortStatus	ERRCODE	PHY link status acquisition
	gerR_IN32_GetCyclicStatus	ERRCODE	Cyclic transmission status acquisition
	gerR_IN32_GetMIB	ERRCODE	MIB information acquisition
	gerR_IN32_ClearMIB	ERRCODE	MIB information clear
	gerR_IN32_GetPortAvailable	ERRCODE	Port enabled status acquisition

Table 6.11 R-IN32M3-CL Driver Interface Function List (2/3)

Function Category	Function Name	Function Type	Overview
LED control	gerR_IN32_SetLERR1LED	ERRCODE	LED control (L ER (port 1))
	gerR_IN32_SetLERR2LED	ERRCODE	LED control (L ER (port 2))
	gerR_IN32_SetERRLED	ERRCODE	LED control (ERR.)
	gerR_IN32_SetDLINKLED	ERRCODE	LED control (D LINK)
	gerR_IN32_SetUSER1LED	ERRCODE	LED control (User LED 1)
	gerR_IN32_SetUSER2LED	ERRCODE	LED control (User LED 2)
	gerR_IN32_SetRUNLED	ERRCODE	LED control (RUN)
	gerR_IN32_DisableLED	ERRCODE	LED control function disablement
	gerR_IN32_EnableLED	ERRCODE	LED control function enablement
Network time	gerR_IN32_GetNetworkTime	ERRCODE	Network time (serial value) acquisition
	gerR_IN32_SetNetworkTime	ERRCODE	Network time (serial value) setting
	gerR_IN32_NetworkTimeToDate	ERRCODE	Network time (serial value) to clock information conversion
	gerR_IN32_DateToNetworkTime	ERRCODE	Clock information to network time (serial value) conversion
MDIO access	gerR_IN32_EnableMACIPAccess	ERRCODE	MAC IP access enablement
	gerR_IN32_DisableMACIPAccess	ERRCODE	MAC IP access disablement
	gerR_IN32_WritePHY	ERRCODE	PHY internal register write
	gerR_IN32_ReadPHY	ERRCODE	PHY internal register read
Transient reception processing	gerR_IN32_MainReceiveTransient1	ERRCODE	Transient reception main processing 1
	gerR_IN32_MainReceiveTransient2	ERRCODE	Transient reception main processing 2
	gerR_IN32_EnableReceiveTransient	ERRCODE	Transient reception enable/disable setting for vendor reasons
	gblR_IN32_GetReceiveTransientStatus	BOOL	Status acquisition of transient reception enable/disable setting for vendor reasons
	gerR_IN32_SetMACAddressTableData	ERRCODE	Node information distribution data (MAC address table) setting

Table 6.11 R-IN32M3-CL Driver Interface Function List (3/3)

Function Category	Function Name	Function Type	Overview
Transient send processing	gerR_IN32_GetUnitInformation	ERRCODE	Unit information acquisition
	gusR_IN32_GetNodeID	USHORT	Node ID acquisition
	gerR_IN32_GetMulticastMACAddress	ERRCODE	Multicast MAC address acquisition
	gerR_IN32_GetUnicastMACAddress	ERRCODE	Unicast MAC address acquisition
	gerR_IN32_GetSendTransientBuffer	ERRCODE	Transient send buffer acquisition
	gerR_IN32_RequestSendingTransient	ERRCODE	Transient send request
	gerR_IN32_MainSendTransient	ERRCODE	Transient send main processing
	gulR_IN32_SetOptionInfo_Response	ULONG	Option information acquisition response frame creation processing
	gulR_IN32_SetSelectInfo_Response	ULONG	Selected station information acquisition response frame creation processing
	gulR_IN32_SetSlmpError_Response	ULONG	SLMP error response frame creation processing
	gulR_IN32_SetContactTest_Response	ULONG	Communication test response frame creation processing
	gulR_IN32_SetCableTest_Response	ULONG	Cable test response frame creation processing
	gulR_IN32_SetMemRead_Response	ULONG	SLMP memory read response frame creation processing
	gulR_IN32_SetMemWrite_Response	ULONG	SLMP memory write response frame creation processing
Interrupt	gerR_IN32_DisableInterrupt	ERRCODE	Interrupt disablement
	gerR_IN32_EnableInterrupt	ERRCODE	Interrupt enablement
Hardware test	gerR_IN32_IEEETest	ERRCODE	IEEE 802.3ab compliance test
	gerR_IN32_InitializeLoopBackTest	ERRCODE	Internal/external loopback communication test initialization
	gerR_IN32_InternalLoopBackTest	ERRCODE	Internal loopback communication test
	gerR_IN32_ExternalLoopBackTest	ERRCODE	External loopback communication test

6.4 R-IN32M3-CL Driver Interface Function Details

The R-IN32M3-CL driver interface functions are called from a user program written in C language.
This section describes how to use the R-IN32M3-CL driver interface functions and the details of related functions.

This section uses the following definitions based on the sample code.

(1) Parameter data type and size

The R-IN32M3-CL driver interface functions use the parameter data and types below.

```
#define  VOID            void;
typedef  char            CHAR;
typedef  unsigned char   UCHAR;
typedef  short           SHORT;
typedef  unsigned short  USHORT;
typedef  int             INT;
typedef  unsigned int    UINT;
typedef  long            LONG;
typedef  unsigned long   ULONG;
typedef  int             ERRCODE;
typedef  int             BOOL;
```

(2) Error code definitions

The R-IN32M3-CL driver interface functions use the error codes returned as return values below.

```
#define  R_IN32_OK                 0    /*!< Normal */
#define  R_IN32_ERR                (-1) /*!< Abnormal end */
#define  R_IN32_ERR_OTHER          (-2) /*!< Abnormal end (Error occurred in driver inside library.) */
#define  R_IN32_ERR_OUTOFRANGE     (-3) /*!< Out of range */
#define  R_IN32_ERR_EMPTY          (-4) /*!< Empty */
#define  R_IN32_ERR_OVERFLOW       (-5) /*!< Overflow */
#define  R_IN32_ERR_NOENTRY        (-6) /*!< No entry */
#define  R_IN32_ERR_NOPERMIT       (-7) /*!< Not permitted */
#define  R_IN32_ERR_NODATA         (-8) /*!< No data */
#define  R_IN32_ERR_NOMYSTATUS     (-9) /*!< No valid MyStatus */
```

(3) Other definitions

```
#define  R_IN32_TRUE   1
#define  R_IN32_FALSE  0
```

6.4.1 Initial setup

(1) gulR_IN32_GetResetStatus

Function	Reset status acquisition			
Call format	ULONG gulR_IN32_GetResetStatus (VOID)			
Arguments	Name	Variable name	Description	I/O
	None			
Return value	R_IN32_RESET_PWRON(1): Power-on reset R_IN32_RESET_SYSTEM(2): System reset			
Description	This function acquires the reset status. Call this function before gerR_IN32_Initialize.			

(2) gerR_IN32_Initialize

Function	R-IN32M3-CL initialization			
Call format	ERRCODE gerR_IN32_Initialize (const UCHAR* puchMACAddr, const R_IN32_UNITINFO_T* pstUnitInfo, const R_IN32_UNITINIT_T *pstUnitInit)			
Arguments	Name	Variable name	Description	I/O
	const UCHAR	*puchMACAddr	Own station MAC address Set as follows for 12-34-56-78-90-AB: puchMACAddr[0]: 0x12 puchMACAddr[1]: 0x34 puchMACAddr[2]: 0x56 puchMACAddr[3]: 0x78 puchMACAddr[4]: 0x90 puchMACAddr[5]: 0xAB	Input
	const R_IN32_UNITINFO_T	*pstUnitInfo	R-IN32M3-CL unit information initial setup For details, refer to Section A) "R_IN32_UNITINFO_T initial setup".	Input
	const R_IN32_UNITINIT_T	*pstUnitInit	R-IN32M3-CL initial setup For details, refer to Section B) "R_IN32_UNITINIT_T initial setup".	Input
Return value	R_IN32_OK: Normal end			
Description	This function performs R-IN32M3-CL initialization and PHY reset. Calling this function disables the R-IN32M3-CL internal WDT. When you want to use the R-IN32M3-CL internal WDT, be sure to call the function gerR_IN32_EnableWDT. For details, refer to Section 6.4.2 "Watchdog timer". *: When a fatal error occurs in R-IN32M3-CL, this function calls the function below created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

Arguments of gerR_IN32_Initialize

The following describes the structure of R_IN32_UNITINFO_T based on the sample code.

```
/* R-IN32M3-CL unit information */
typedef struct R_IN32_UNITINFO_TAG {
    /* Cyclic transmission size maximum value */
    ULONG    ulMaxRySize;                  /*!< RY size [bytes (octets)] */
    ULONG    ulMaxRWwSize;                 /*!< RWw size (words) */
    ULONG    ulMaxRxSize;                  /*!< RX size [bytes (octets)] */
    ULONG    ulMaxRWrSize;                 /*!< RWr size (words) */

    /* Station information 1 */
    ULONG    ulMyStationPortTotalNumber;   /*!< No. of own station ports */
    ULONG    ulTokenHoldTime;              /*!< Token hold time */

    /* Station information 2 */
    ULONG    ulIOType;                     /*!< Node information (I/O type) */

    /* Network information */
    ULONG    ulNetVersion;                 /*!< Network firmware version */
    ULONG    ulNetModelType;               /*!< Network model type */
    ULONG    ulNetUnitModelCode;           /*!< Network model code */
    ULONG    ulNetVendorCode;              /*!< Network vendor code */
    UCHAR    auchNetUnitModelName[20];     /*!< Network model name */
    UCHAR    auchNetVendorName[32];        /*!< Network vendor name */
    USHORT   usHwVersion;                  /*!< Network hardware version */
    USHORT   usDeviceVersion;              /*!< Network device version */

    /* Controller information */
    BOOL     blInfomationFlag;             /*!< Controller information status flag */
    ULONG    ulCtrlVersion;                /*!< Controller firmware version */
    ULONG    ulCtrlModelType;              /*!< Controller model type */
    ULONG    ulCtrlUnitModelCode;          /*!< Controller model code */
    ULONG    ulCtrlVendorCode;             /*!< Controller vendor code */
    UCHAR    auchCtrlUnitModelName [20];   /*!< Controller model name */
    UCHAR    auchCtrlVendorName [32];      /*!< Controller vendor name */
    ULONG    ulVendorInformation;          /*!< Controller vendor device specific information */
} R_IN32_UNITINFO_T;
```

A) R_IN32_UNITINFO_T initial setup

The items initially set by R_IN32_UNITINFO_T are as follows:

(a) RY size [bytes (octets)]

Specifies the RY size (bytes) communicable by the own station in increments of 1 byte (multiple of 1).

The maximum value for an intelligent device station is 256 bytes.

(b) RWw size (words)

Specifies the RWw size (words) communicable by the own station in increments of 2 words (multiple of 2).

The maximum value for an intelligent device station is 1024 words.

(c) RX size [bytes (octets)]

Specifies the RX size (bytes) communicable by the own station in increments of 1 byte (multiple of 1).

The maximum value for an intelligent device station is 256 bytes.

(d) RWr size (words)

Specifies the RWr size (words) communicable by the own station in increments of 2 words (multiple of 2).

The maximum value for an intelligent device station is 1024 words.

(e) No. of own station ports

Specifies the number of physical communication ports of the own station.

For an intelligent device station developed with R-IN32M3-CL, set "2" or "1".

(f) Token hold time

Specifies the maximum time the own station holds a token after token passing begins, in μs.

For an intelligent device station developed with R-IN32M3-CL, set 23 (μsec).

(g) Node information (I/O type)

Specifies the I/O type.

00b (0x0) indicates mixed, 01b (0x1) indicates input, 10b (0x2) indicates output, and 11b (0x3) indicates composite.

Mixed is used in a case when the input and output are mixed and the input and output use the same address.

Composite is used in a case where the input and output are mixed and the input and output do not use the same address.

(h) Network firmware version

Specifies the firmware version of the network. The firmware version is any version defined by the vendor.

(i) Network model type

Specifies the model type specified by the CC-Link Partner Association.

(j) Network model code

Specifies the model code of the network.

The model code is any code defined by the vendor. Manage the code so that it is unique within the same vendor code.

(k) Network vendor code

Specifies the vendor code acquired when the vendor became a member of the CC-Link Partner Association, in BCD.

(If the vendor code is 5678, 0x5678 is specified.)

(l) Network model name

Specifies the model name of the network (in 20-byte character string (ASCII code)).

The model name is any name defined by the vendor. Manage the name so that it is unique within the same vendor code.

(m) Network vendor name

Specifies the vendor name of the network (in 32-byte character string (ASCII code)).

The vendor name is any name defined by the vendor.

(n) Network hardware version

Specifies the hardware version of the network. The hardware version is any version defined by the vendor.

(o) Network device version

Specifies the device version (Version).

The device version (Version) indicates the version of the functions of the developed device. Used for associating the developed device with CSP+ files.[Note]

(p) Controller information status flag

Enables/Disables controller information ((q) Controller firmware version to (w) Controller vendor device specific information). R_IN32_FLASE indicates disable and R_IN32_TRUE indicates enable.

Disabled when there is only a communication function.

(q) Controller firmware version

Specifies the firmware version of the controller. The firmware version is any version defined by the vendor.

(r) Controller model type

Specifies the model type specified by the CC-Link Partner Association.

(s) Controller model code

Specifies the model code of the controller.

The model code is any code defined by the vendor. Manage the code so that it is unique within the same vendor code.

(t) Controller vendor code

Specifies the vendor code acquired when the vendor became a member of the CC-Link Partner Association, in BCD.

(If the vendor code is 5678, 0x5678 is specified.)

(u) Controller model name

Specifies the model name of the controller. (in 20-byte character string (ASCII code)).

The model name is any name defined by the vendor. Manage the name so that it is unique within the same vendor code.

(v) Controller vendor name

Specifies the vendor name of the controller. (in 32-byte character string (ASCII code)).

The vendor name is any name defined by the vendor.

(w) Controller vendor device specific information

Specifies the vendor device specific information of the controller.

The vendor device specific information is any information defined by the vendor.

Note. The device version of CSP+ is described below.
For details, refer to "DEVICE_INFO Part" in the "Control & Communication System Profile Specification".

[Network and Controller: Supplement Information]

1) Definition of network and controller

Network: A communication section comprising R-IN32M3-CL and the peripheral circuit in the own station

Controller: A functional section which is unique to the vendor (such as I/O section, temperature adjustment section and robot section) in the own station

The following describes examples.

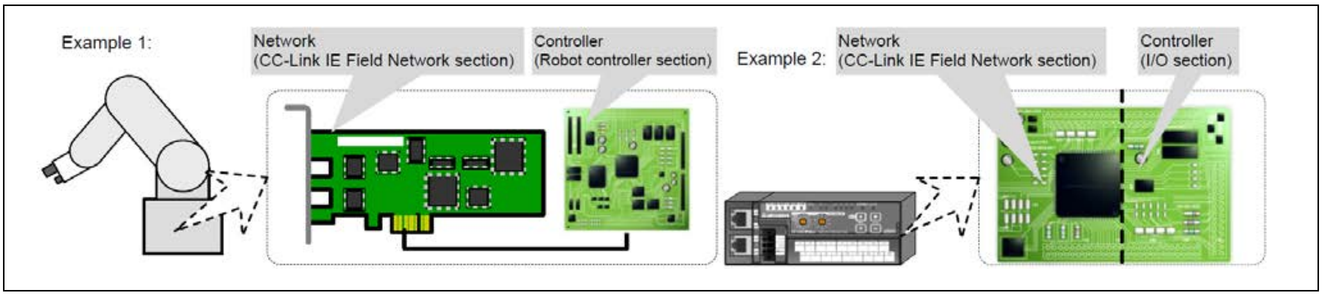

Example 1:
Network
(CC-Link IE Field Network section)
Controller
(Robot controller section)
Example 2:
Network
(CC-Link IE Field Network section)
Controller
(I/O section)

Figure 6.52 Network and Controller Example

2) Setting of network

Network setting is required. The following items are checked in the conformance test.

(h) Network firmware version

(j) Network model code

(i) Network model type

(k) Network vendor code

3) Setting of controller

Controller setting is optional.

Set the controller in the following cases. (In other cases, controller setting is not required.)

- When performing the parameter processing/command execution of slave station after verifying the vendor code/model code described in CSP+ against the controller information of the connected slave stations.
- When the developed device (network) is a communication optional item for a product (controller) such as series products.
- When the vendor of controller and network is different.

[Device Version: Supplemental Information]

1) Background

When the software version of a R-IN32M3-CL application product is upgraded, specification changes sometimes occur, such as the addition of slave station parameter processing or command execution.

When the specifications of a R-IN32M3-CL application product change, the CSP+ file also needs to be updated in accordance with the specification change.

2) Purpose of device version

The information that identifies the specifications before and after a change is the device version. The device version is used to indicate the specifications of the R-IN32M3-CL application product that correspond to each CSP+ file.

(a) Purpose of use by the engineering tool

The engineering tool manages all CSP+ files having different device versions, making it possible to provide optimum functions and UI in accordance with the used version of the R-IN32M3-CL application product.

(b) Purpose of use by end user

The end user can select the CSP+ file for the device actually used upon comparing the device versions described in the CSP+ file and the version of the R-IN32M3-CL application product used.

Arguments of gerR_IN32_Initialize

The following describes the structure of R_IN32_UNITINIT_T based on the sample code.

```
/* R-IN32M3-CL initial setup */
typedef struct R_IN32_UNITINIT_TAG {
    BOOL     blNMIUse;                    /*!< NMI interrupt use */
    BOOL     blInterruptUse;              /*!< MPU interrupt function use */
    BOOL     blFailedProcess1;            /*!< Failed process setting 1 */
    BOOL     blFailedProcess2;            /*!< Failed process setting 2 */
    ULONG    ulNodeType;                  /*!< Node type */
    BOOL     blTransientReceiveEnable;    /*!< Transient reception function */
    BOOL     blMACAddressTableRequest;    /*!< Node information distribution request */
    ULONG    ulRunStatus;                 /*!< Initial value of detailed application operation status */
    ULONG    ulErrorStatus;               /*!< Initial value of detailed application error status */
    ULONG    ulErrorCode;                 /*!< Initial value of application error code */
    ULONG    ulUserInformation;           /*!< Initial value of vendor specific node information */
    ULONG    ulOptionSupport;             /*!< Option status */
    ULONG    ulSlmpSupport;               /*!< SLMP support bit */
    ULONG    ulSlmpDiagnosisSupport;      /*!< Diagnostic function support status */
} R_IN32_UNITINIT_T;
```

B) R_IN32_UNITINIT_T initial setup

The items initially set by R_IN32_UNITINIT_T are as follows:

(a) NMI interrupt use (Only when you want to use the R-IN32M3-CL internal WDT function)

Specify "R_IN32_TRUE" when you want to use the R-IN32M3-CL internal WDT function, and "R_IN32_FALSE" when you do not.

Specifying "R_IN32_TRUE" changes the NMIL pin to "Low" when the R-IN32M3-CL internal WDT overflows.

(b) MPU interrupt function use

Specify "R_IN32_TRUE" when you want to use the R-IN32M3-CL MPU interrupt function, and "R_IN32_FALSE" when you do not.

Specifying "R_IN32_TRUE" changes the INTL pin to "Low" when a R-IN32M3-CL interrupt occurs.

(c) Failed process setting 1

Specify "R_IN32_TRUE".

When any of the signals below are true, R-IN32M3-CL changes to bypass mode. (Communication frames are neither sent nor received. A received frame is forwarded as is to another port.)

[1] When the WDTIL pin is True (Low)

[2] When the R-IN32M3-CL internal WDT times out

To clear bypass mode, power-on reset or system reset is required.

(d) Failed process setting 2

Specify "R_IN32_TRUE".

When an own station error is set (gerR_IN32_ForceStop function is called), R-IN32M3-CL changes to bypass mode. (Communication frames are neither sent nor received. A received frame is forwarded as is to another port.)

To clear the own station error, power-on reset or system reset is required.

For gerR_IN32_ForceStop function details, refer to Section 6.4.5(2) "gerR_IN32_ForceStop".

(e) Node type

Specifies the node type of the own station. Specify intelligent device station (0x33).

(f) Transient reception function

Specify "R_IN32_TRUE".

This item specifies whether or not the transient reception function is present. "R_IN32_FALSE" indicates the function is not present, and "R_IN32_TRUE" indicates the function is present.

(g) Node information distribution request

Node information indicates the correspondence between the MAC addresses and node numbers of other stations. When this is set to "R_IN32_TRUE", node information is distributed from the master station by multicast. Set this item to "R_IN32_TRUE" when a transient transmission client function is implemented, and to "R_IN32_FALSE" when it is not.

Caution 1. When "R_IN32_FALSE" is specified, also specify "gblUserMACAddressTableRequest", which judges whether the node information distribution is required or not, to "R_IN32_FALSE" in Transient1 receive data processing (refer to Section 6.2.17 "Transient1, Transient2, and TransientAck send processing").

2. When a response is returned to the send source, the response can be returned using the send source MAC address.
When transient frames are actively sent, the MAC address table is used.
The MAC address table is created using Node information distribution frame (Transient1 frame) distributed from the master station.

(h) Initial value of detailed application operation status

Specifies the initial value of the detailed application operation status within nodeStatus of the MyStatus frame.

Table 6.12 List of Initial Values of Detailed Application Operation Status

Value	Communication Operation
R_IN32_RUNSTS_UNSUPPORTED	Detailed application operation status notification not supported
R_IN32_RUNSTS_STOP	Application stopped
R_IN32_RUNSTS_RUN	Application running
R_IN32_RUNSTS_NOTEXIST	Application user does not exist

(i) Initial value of detailed application error status

Specifies the initial value of the detailed application error status within nodeStatus of the MyStatus frame.

Table 6.13 List of Initial Values of Detailed Application Error Status

Value	Communication Operation
R_IN32_ERRSTS_NONE	No error
R_IN32_ERRSTS_WARNING	Minor error
R_IN32_ERRSTS_ERROR	Moderate error
R_IN32_ERRSTS_FATALERROR	Major error

(j) Initial value of application error code

Specifies the initial value of errorCode of the MyStatus frame.

(k) Initial value of vendor specific node information

Specifies the initial value of vendorSpfNodeInfo of the MyStatus frame.

(l) Option status

Set this item to "R_IN32_TRUE" (recommended) when options are supported, and to "R_IN32_FALSE" when options are not supported.

Remark. An option is an extended function of CC-Link IE Field Network, and includes the SLMP frame send/receive function and CC-Link IE Field Network diagnostic function.

(m) SLMP support bit

Set this item to "R_IN32_TRUE" (recommended) when SLMP frames are sent and received, and to "R_IN32_FALSE" when they are not.

Caution. To send/receive SLMP frames, set both this and the "Initial value of option status" to "R_IN32_TRUE".

(n) Diagnostic function support status

Set this item to "R_IN32_TRUE" (recommended) when the CC-Link IE Field Network diagnostic function is supported, and to "R_IN32_FALSE" when it is not.

Caution. To support the CC-Link IE Field Network diagnostic function, set this item as well as the "Initial value of option status" and the "Initial value of SLMP support bit" to "R_IN32_TRUE".

(3) gerR_IN32_SetNodeAndNetworkNumber

Function	Node number and network number setting			
Call format	ERRCODE gerR_IN32_SetNodeAndNetworkNumber (UCHAR uchNetworkNumber, USHORT usNodeNumber)			
Arguments	Name	Variable name	Description	I/O
	UCHAR	uchNetworkNumber	Network number (value range: 1 to 239)	Input
	USHORT	usNodeNumber	Node number (value range: 1 to 120)	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end (status error in library) R_IN32_ERR_OUTOFRANGE: Node number out of range or network number out of range			
Description	This function sets the node number and network number in R-IN32M3-CL. When the return value is R_IN32_ERR_ OUTOFRANGE, the node number and network number are not set. Add error processing to the call source function. *: This function needs to be called after the processing described in Section 8.2.2 "Initialization processing" before calling gerR_IN32_Start by Section 8.2.3 "Communication start processing". Calling this function before executing the above processing results in a R_IN32_ERR (abnormal end; status error in library).			

(4) ger R_IN32_Start

Function	R-IN32M3-CL communication start			
Call format	ERRCODE gerR_IN32_Start (VOID)			
Arguments	Name	Variable name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function provides instructions to start communication to R-IN32M3-CL. *: When a fatal error occurs in R-IN32M3-CL, this function calls the function below created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

6.4.2 Watchdog timer

(1) gerR_IN32_ResetWDT

Function	R-IN32M3-CL internal WDT reset			
Call format	ULONG gerR_IN32_ResetWDT (VOID)			
Arguments	Name	Variable name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function resets the R-IN32M3-CL internal WDT. *: If you want to call a function within Section 6.4.2 "Watchdog timer" after this function is called, wait 1.032 µs or longer.			

(2) gerR_IN32_DisableWDT

Function	R-IN32M3-CL internal WDT disablement			
Call format	ULONG gerR_IN32_DisableWDT (VOID)			
Arguments	Name	Variable name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function disables the R-IN32M3-CL internal WDT. *: If you want to call a function within Section 8.4.2 "Watchdog timer" after this function is called, wait 1.032 µs or longer. R-IN32M3-CL enables the R-IN32M3-CL internal WDT immediately after reset. (Initial value of R-IN32M3-CL internal WDT time limit setting: 3.2 s.) The R-IN32M3-CL internal WDT is disabled when the function gerR_IN32_Initialize is called. Implement one of the following when the period until startup of gerR_IN32_Initialize takes time: ・Call this function to disable the R-IN32M3-CL internal WDT. ・Call gerR_IN32_ResetWDT to reset the R-IN32M3-CL internal WDT. (Make sure that the R-IN32M3-CL internal WDT does not time out.)			

(3) gerR_IN32_EnableWDT

Function	R-IN32M3-CL internal WDT enablement			
Call format	ULONG gerR_IN32_EnableWDT (VOID)			
Arguments	Name	Variable name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function enables the R-IN32M3-CL internal WDT. *: If you want to call a function within Section 8.4.2 "Watchdog timer" after this function is called, wait 1.032 µs or longer. R-IN32M3-CL disables the R-IN32M3-CL internal WDT when the function gerR_IN32_Initialize is called. Be sure to call this function when you want to use the R-IN32M3-CL internal WDT.			

(4) gerR_IN32_SetWDT

Function	R-IN32M3-CL internal WDT time limit setting			
Call format	ULONG gerR_IN32_SetWDT (USHORT usWDTCOUNT)			
Arguments	Name	Variable name	Description	I/O
	USHORT	usWDTCOUNT	R-IN32M3-CL internal WDT time limit setting 0x0000: 100ms 0x0001: 200ms 0x0002: 300ms : 0x001F: 3.2 s	Input
Return value	R_IN32_OK: Normal end			
Description	This function sets the R-IN32M3-CL internal WDT time limit. *: If you want to call a function within Section 8.4.2 "Watchdog timer" after this function is called, wait 1.032 μs or longer. If the R-IN32M3-CL internal WDT time limit setting is changed by this function while the R-IN32M3-CL internal WDT is running (after the function gerR_IN32_EnableWDT is called), the R-IN32M3-CL internal WDT runs using the new time limit setting when the function gerR_IN32_ResetWDT is called. (Until the function gerR_IN32_ResetWDT is called, the R-IN32M3-CL internal WDT runs using the R-IN32M3-CL internal WDT time limit setting prior to the change.)			

6.4.3 Event

(1) gerR_IN32_GetEvent

Function	R-IN32M3-CL event detection			
Call format	ERRCODE gerR_IN32_GetEvent (R_IN32_EVTPRM_INTERRUPT_T *pstEvent)			
Arguments	Name	Variable name	Description	I/O
	R_IN32_EVTPRM_INTERRUPT_T	*pstEvent	Interrupt cause	Output
Return value	R_IN32_OK: Normal end			
Description	This function detects R-IN32M3-CL events.			

Arguments of gerR_IN32_GetEvent

The following describes the configuration of R_IN32_EVTPRM_INTERRUPT_T based on the sample code.

```
/* Interrupt cause */
typedef struct R_IN32_EVTPRM_INTERRUPT_TAG {
    union {
        ULONG       ulAll;
        struct {
            ULONG   b1ZCommConnect:                1;    /* b0     : Connect communication */
            ULONG   b1ZCommDisconnect:             1;    /* b1     : Disconnect communication */
            ULONG   b1ZCommConnectToDisconnect:    1;    /* b2     : Connect communication
                                                                       → Disconnect communication */
            ULONG   b1ZCommDisconnectToConnect:    1;    /* b3     : Disconnect communication
                                                                       → Connect communication */
            ULONG   b1ZChangeStNoNetNo:            1;    /* b4     : Change node number and network number */
            ULONG   b1ZChangeActCommand:           1;    /* b5     : Change run command */
            ULONG   b1ZPrmFrmRcv_OK:               1;    /* b6     : Parameter frame reception */
            ULONG   b1ZReserve1:                   1;    /* b7     : Reserved */
            ULONG   b1ZPrmChkFrmRcv_OK:            1;    /* b8     : ParamCheck frame reception
                                                                       (when parameters match) */
            ULONG   b3ZReserve2:                   3;    /* b9-11  : Reserved */
            ULONG   b1ZRecvNonCyclic:              1;    /* b12    : Transient reception */
            ULONG   b1ZSendFinNonCyclic:           1;    /* b13    : Transient send complete */
            ULONG   b7ZReserve3:                   7;    /* b14-20 : Reserved */
            ULONG   b1ZMasterWatchTimeout:         1;    /* b21    : Master watch timer timeout occurred */
            ULONG   bAZReserve4:                   10;   /* b22-31 : Reserved */
        } stBit;
    } uniFlag;
} R_IN32_EVTPRM_INTERRUPT_T;
```

(2) gerR_IN32_Main

Function	R-IN32M3-CL event detection main processing			
Call format	ERRCODE gerR_IN32_Main (const R_IN32_EVTPRM_INTERRUPT_T *pstEvent)			
Arguments	Name	Variable Name	Description	I/O
	const R_IN32_EVTPRM_INTERRUPT_T	*pstEvent	Interrupt cause	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end (status error in library)			
Description	This function performs processing in response to a R-IN32M3-CL event. *: This function needs to be called after the processing described in Section 8.2.2 "Initialization processing" and Section 8.2.3 "Communication start processing". Calling this function before executing the above processing results in a R_IN32_ERR (abnormal end; status error in library).			

(3) gerR_IN32_RestartEvent

Function	R-IN32M3-CL event restart			
Call format	ERRCODE gerR_IN32_RestartEvent (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function restarts events stopped by R-IN32M3-CL event detection (gerR_IN32_GetEvent).			

(4) gerR_IN32_UpdatePortStatus

Function	PHY link status update			
Call format	ERRCODE gerR_IN32_UpdatePortStatus (ULONG ulPort)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulPort	Port specification R_IN32_PORT1(0): Port 1 R_IN32_PORT2(1): Port 2	Input
Return value	R_IN32_OK: Normal end			
Description	This function updates the PHY link status.			

(5) gerR_IN32_UpdateMIB

Function	MIB information update			
Call format	ERRCODE gerR_IN32_UpdateMIB (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end [MIB information collection error (status error in library / mismatch)] R_IN32_ERR_OTHER: Abnormal end [MIB information collection error (error occurred in driver inside library)]			
Description	This function updates the MIB information. *: When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

6.4.4 Cyclic transmission

(1) gerR_IN32_SetCyclicStop

Function	Cyclic transmission stop for device-side reasons			
Call format	ERRCODE gerR_IN32_SetCyclicStop (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function stops cyclic transmission for device-side reasons. If you want to clear the stop status, call the function gerR_IN32_ClearCyclicStop.			

(2) ger R_IN32_ClearCyclicStop

Function	Cyclic transmission stop clear for device-side reasons			
Call format	ERRCODE gerR_IN32_ClearCyclicStop (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function clears cyclic transmission stop that was called by the function gerR_IN32_SetCyclicStop.			

(3) ger R_IN32_GetReceivedCyclicData

Function	Cyclic receive data acquisition			
Call format	ERRCODE gerR_IN32_GetReceivedCyclicData (VOID *pRyDst, VOID *pRWwDst, BOOL blEnable)			
Arguments	Name	Variable Name	Description	I/O
	VOID	*pRyDst	RY area	Output
	VOID	*pRWwDst	RWw area[*1]	Output
	BOOL	blEnable	Enables/Disables copying. R_IN32_TRUE: Enable R_IN32_FALSE: Disable	Input
Return value	R_IN32_OK: Normal end (received data present) R_IN32_ERR: Abnormal end (no received data)			
Description	This function stores cyclic receive data from the master station in the addresses indicated by pRyDst and pRWwDst. Note, however, that when blEnable is set to R_IN32_FALSE, the cyclic receive data is discarded. (The return value changes to R_IN32_ERR.) *: R_IN32_ERR: Abnormal end (no received data) While a R_IN32_ERR occurs when no cyclic communication is received from the previous call of the function gerR_IN32_GetReceivedCyclicData to the current call of the function gerR_IN32_GetReceivedCyclicData, this does not indicate an error. *1: Set the start address of the RWw area in increments of 4 bytes (0 or multiple of 4).			

(4) ger R_IN32_GetMasterNodeStatus

Function	Master station status acquisition			
Call format	ERRCODE gerR_IN32_GetMasterNodeStatus (BOOL *pblRunSts, BOOL *pblErrSts, ULONG *pulErrCode)			
Arguments	Name	Variable Name	Description	I/O
	BOOL	*pblRunSts	Application operation status R_IN32_TRUE: Running R_IN32_FALSE: Stopped	Output
	BOOL	*pblErrSts	Application error status R_IN32_TRUE: Error R_IN32_FALSE: No error	Output
	ULONG	*pulErrCode	Master station error code	Output
Return value	R_IN32_OK: Normal end (MyStatus frame received from master station) R_IN32_ERR: Abnormal end [MyStatus frame not received from master station due to no data link (data link disconnected)]			
Description	This function acquires the status of the master station from the MyStatus frame received from the master station. When the MyStatus frame is not received from the master station due to no data link (data link disconnected), the arguments are as follows: pblRunSts R_IN32_FALSE pblErrSts R_IN32_FALSE pulErrCode 0			

(5) ger R_IN32_SetMyStatus

Function	MyStatus send data setting			
Call format	ERRCODE gerR_IN32_SetMyStatus (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function sets the own station status specified by the function gerR_IN32_SetNodeStatus in R-IN32M3-CL.			

(6) ger R_IN32_SetSendCyclicData

Function	Cyclic send data setting			
Call format	ERRCODE gerR_IN32_SetSendCyclicData (const VOID *pRxSrc, const VOID *pRWwSrc, BOOL blEnable)			
Arguments	Name	Variable Name	Description	I/O
	const VOID	*pRxSrc	RX area	Input
	const VOID	*pRWwSrc	RWw area*1	Input
	BOOL	blEnable	Enables/Disables update. R_IN32_TRUE: Enable R_IN32_FALSE: Disable	Input
Return value	R_IN32_OK: Normal end			
Description	This function sets the cyclic send data stored in the addresses indicated in pRxSrc and pRWwSrc in R-IN32M3-CL. Note, however, that when blEnable is set to R_IN32_FALSE, cyclic send data is not set. (The return value changes to R_IN32_ERR.) *1: Set the start address of the RWw area in increments of 4 bytes (0 or multiple of 4).			

6.4.5 Own station status setup

(1) gerR_IN32_SetNodeStatus

Function	Own station status setting			
Call format	ERRCODE gerR_IN32_SetNodeStatus (ULONG ulRunSts, ULONG ulErrSts, ULONG ulErrCode, ULONG ulUserInformation)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulRunSts	Detailed application operation status R_IN32_RUNSTS_UNSUPPORTED(0): Detailed application operation status notification not supported R_IN32_RUNSTS_STOP(1): Application stopped R_IN32_RUNSTS_RUN(2): Application running R_IN32_RUNSTS_NOTEXIST(3): Application user does not exist	Input
	ULONG	ulErrSts	Detailed application error status R_IN32_ERRSTS_NONE(0): No error R_IN32_ERRSTS_WARNING(1): Minor error R_IN32_ERRSTS_ERROR(2): Moderate error R_IN32_ERRSTS_FATALERROR(3): Major error	Input
	ULONG	ulErrCode	Error code	Input
	ULONG	ulUserInformation	Vendor specific node information	Input
Return value	R_IN32_OK: Normal end			
Description	This function sets the own station status as information to be sent in a MyStatus frame.			

(2) gerR_IN32_ForceStop

Function	Own station error setting			
Call format	ERRCODE gerR_IN32_ForceStop (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function sets an own station error in R-IN32M3-CL. To clear the own station error, power-on reset or system reset is required.			

6.4.6 Own station status acquisition

(1) gerR_IN32_GetNodeAndNetworkNumber

Function	Node number and network number acquisition			
Call format	ERRCODE gerR_IN32_GetNodeAndNetworkNumber (USHORT *pusNodeNumber, UCHAR *puchNetworkNumber)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	*pusNodeNumber	Node number	Output
	UCHAR	*puchNetworkNumber	Network number	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the node number and network number.			

(2) gerR_IN32_GetCurrentCyclicSize

Function	Acquisition of cyclic transmission size specified from master station			
Call format	ERRCODE gerR_IN32_GetCurrentCyclicSize (R_IN32_CYCLIC_SIZE_T *pstCyclicSize)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32_CYCLIC_SIZE_T	*pstCyclicSize	Cyclic transmission size ulRySize: RY size [bytes (octets)] ulRWwSize: RWw size [bytes (octets)] ulRxSize: RX size [bytes (octets)] ulRWrSize: RWr size [bytes (octets)]	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the cyclic transmission size specified from the master station in Parameter frame. The functions gerR_IN32_GetReceivedCyclicData and gerR_IN32_SetSendCyclicData input and output cyclic send/receive data in the size acquired by this function.			

Arguments of gerR_IN32_GetCurrentCyclicSize

The following describes the structure of R_IN32_CYCLIC_SIZE_T based on the sample code.

```
/* Cyclic transmission size */
typedef struct R_IN32_CYCLIC_SIZE_TAG {
        ULONG   ulRySize;       /*!< RY size [bytes (octets)] */
        ULONG   ulRWwSize;      /*!< RWw size [bytes (octets)] */
        ULONG   ulRxSize;       /*!< RX size [bytes (octets)] */
        ULONG   ulRWrSize;      /*!< RWr size [bytes (octets)] */
} R_IN32_CYCLIC_SIZE_T;
```

(3) gerR_IN32_GetCommumicationStatus

Function	Data link status acquisition			
Call format	ERRCODE gerR_IN32_GetCommumicationStatus (ULONG *pulCommSts)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	*pulCommSts	Data link status R_IN32_COMMSTS_CYC_DLINK(2): Data link in operation (cyclic transmission in progress) R_IN32_COMMSTS_TOKEN_PASS(1): Data link in operation (cyclic transmission stopped) R_IN32_COMMSTS_DISCONNECT(0): Data link not performed (disconnected)	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the data link status. Turn the D LINK LED on/off according to the data link status. R_IN32_COMMSTS_CYC_DLINK: LED on Others: LED off *: For D LINK LED on/off control, refer to 6.2.12 "Communication status update processing".			

(4) gerR_IN32_GetPortStatus

Function	PHY link status acquisition			
Call format	ERRCODE gerR_IN32_GetPortStatus (ULONG ulPort, ULONG *pulLinkStatus, ULONG *pulSpeed, ULONG *pulDuplex)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulPort	Port specification R_IN32_PORT1(0): Port 1 R_IN32_PORT2(1): Port 2	Input
	ULONG	*pulLinkStatus	Link status R_IN32_LINKUP(1): Link-up R_IN32_LINKDOWN(0): Link-down	Output
	ULONG	*pulSpeed	Speed R_IN32_SPEED_1G(0): 1 Gbps R_IN32_SPEED_100M(1): 100 Mbps R_IN32_SPEED_10M(2): 10 Mbps (Enabled when the second argument *pulLinkStatus is R_IN32_LINKUP(1). Do not use when the second argument is R_IN32_LINKDOWN(0).)	Output
	ULONG	*pulDuplex	Full duplex / Half duplex R_IN32_DUPLEX_FULL(0): Full duplex R_IN32_DUPLEX_HALF(1): Half duplex (Enabled when the second argument *pulLinkStatus is R_IN32_LINKUP(1). Do not use when the second argument is R_IN32_LINKDOWN(0).)	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the PHY link status.			

(5) gerR_IN32_GetCyclicStatus

Function	Cyclic transmission status acquisition			
Call format	ULONG gerR_IN32_GetCyclicStatus (R_IN32_CYCLIC_STA_T *pstCyclicStatus)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32_CYCLIC_STA_T	*pstCyclicStatus	Cyclic transmission status Bit2-0 Cyclic transmission parameter hold status 001b: Parameter normally received 010b: Not received or ID mismatch 011b: Checking 100b: Parameter abnormally received Bit3 Cyclic transmission parameter check status 0: Checked 1: Checking Bit4 Node number invalid setting status 0: Within range 1: Out of range Bit5 Reserved node setting status 0: Non-reserved node 1: Reserved node Bit6 Cyclic transmission implementation instruction (batch) setting status 0: Run 1: Stop Bit7 Cyclic transmission implementation instruction (individual) setting status 0: Run 1: Stop Bit8 Reserved Bit9 Cyclic transmission continuation not possible error status 0: No error 1: Cyclic transmission continuation not possible error Bit10 Node number duplication status 0: No duplication 1: Duplication Bit11 Reserved Bit12 Node type invalid / Specified size invalid status 0: Normal 1: Invalid Bit13 Reserved Bit14 Disconnection status 0: Cyclic communications in progress or token passing in progress 1: Disconnected Bit15 Stop status due to own reasons 0: Not stopped 1: Cyclic transmission stopped due to reason other than the above	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the cyclic transmission status.			

Arguments of gerR_IN32_GetCyclicStatus

The following describes the structure of R_IN32_CYCLIC_STA_T based on the sample code.

```
/* Cyclic transmission status */
typedef struct R_IN32_CYCLIC_STA_TAG {
  union {
    USHORTusAll;
    struct {
        USHORT   b3ZComonParamkeepCond:         3;  /* b2-0  : Cyclic transmission parameter hold status */
        USHORT   b1ZParamCheckCond:             1;  /* b3    : Cyclic transmission parameter check status */
        USHORT   b1ZMyNodeNoRangeOut:           1;  /* b4    : Node number invalid setting status */
        USHORT   b1ZMyNodeReserveSetup:         1;  /* b5    : Reserved node setting status */
        USHORT   b1ZCyclicOpeInstructPackage:   1;  /* b6    : Cyclic transmission implementation
                                                                  instruction (batch) setting status */
        USHORT   b1ZCyclicOpeInstructVarious:   1;  /* b7    : Cyclic transmission implementation
                                                                  instruction (individual) setting status */
        USHORT   b1ZReserved1:                  1;  /* b8    : Reserved */
        USHORT   b1ZMyMpuAbnomal:               1;  /* b9    : Cyclic transmission continuation not
                                                                  possible error status */
        USHORT   b1ZMyNodeNumberDuplicate:      1;  /* b10   : Node number duplication status */
        USHORT   b1ZReserved2:                  1;  /* b11   : Reserved */
        USHORT   b1ZNodeTypeWrong:              1;  /* b12   : Node type invalid / Specified size
                                                                  invalid  status */
        USHORT   b1ZReserved3:                  1;  /* b13   : Reserved */
        USHORT   b1ZDLinkState:                 1;  /* b14   : Disconnection status */
        USHORT   b1ZCyclicState:                1;  /* b15   : Stop status due to own reasons*/
    } stBit;
  } uniCycSta;
} R_IN32_CYCLIC_STA_T;
```

(6) gerR_IN32_GetMIB

Function	MIB information acquisition			
Call format	ERRCODE gerR_IN32_GetMIB (R_IN32_MIB_T *pstMIB)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32_MIB_T	*pstMIB	R-IN32M3-CL MIB information	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires MIB information.			

Caution. MIB information is non-disclosed information. Disclose the information only to the vendor.

Arguments of gerR_IN32_GetMIB

The following describes the structure of R_IN32_MIB_T based on the sample code.

```
/* MIB Information */
typedef struct R_IN32_MIB_TAG {
        R_IN32_MIBSDRD_T      stSDRD;                /*!< Send/receive area counter value */
        R_IN32_MIBMACIP_T     stMACIP1;              /*!< MAC IP area counter value (port 1) */
        R_IN32_MIBMACIP_T     stMACIP2;              /*!< MAC IP area counter value (port 2) */
        R_IN32_MIBRGCNT_T     stRING1;               /*!< Ring control area counter value (port 1) */
        R_IN32_MIBRGCNT_T     stRING2;               /*!< Ring control area counter value (port 2) */
        ULONG                 ulP1DownCounter;       /*!< Link down counter (port 1) */
        ULONG                 ulP2DownCounter;       /*!< Link down counter (port 2) */
        ULONG                 ulMasterWatchCount;    /*!< Master watch timer error counter */
} R_IN32_MIB_T;
```

The following describes the configuration of the tags included in R_IN32_MIB_T.

```
/* MIB information (counter) */
typedef struct R_IN32_MIBSDRD_TAG {
        ULONG   ulCyclicRecNomalFrameCnt;    /*!< Received cyclic frame counter */
        ULONG   ulNonCyclicRecValidCnt;      /*!< Received transient frame counter */
        ULONG   ulNonCyclicRecRejectCnt;     /*!< Received transient frame discarded counter */
} R_IN32_MIBSDRD_T;
```

```
/* MIB information (ring control area) */
typedef struct R_IN32_MIBRGCNT_TAG {
        ULONG   ulHecErr;         /*!< MIB1: No. of HEC error frames */
        ULONG   ulDcsFcsErr;      /*!< MIB2: No. of DCS/FCS error frames */
        ULONG   ulUnderErr;       /*!< MIB3: No. of undersize error frames */
        ULONG   ulRpt;            /*!< MIB4: No. of forwarded frames */
        ULONG   ulUp;             /*!< MIB5: No. of upper layer transmission frames */
        ULONG   ulRptFullDrop;    /*!< MIB6: No. of discarded frames due to full forward buffer */
        ULONG   ulUpFullDrop;     /*!< MIB7: No. of discarded frames due to full upper layer transmission buffer */
} R_IN32_MIBRGCNT_T;
```

```
/* MIB information (MAC IP) */
typedef struct R_IN32_MIBMACIP_TAG {
        ULONG  ulRFrm;          /*!< Received frame counter */
        ULONG  ulTFrm;          /*!< Sent frame counter */
        ULONG  ulRUnd;          /*!< Received undersized frame counter */
        ULONG  ulROvr;          /*!< Received oversized frame counter */
        ULONG  ulRFcs;          /*!< Received frame FCS error counter */
        ULONG  ulRFgm;          /*!< Received frame fragment error counter */
        ULONG  ulRIFGErr;       /*!< Minimum IFG frame detection counter */
        ULONG  ulREps;          /*!< Received frame with SFD or less detection counter */
        ULONG  ulRCde;          /*!< Reception code error counter */
        ULONG  ulRFce;          /*!< Received invalid carrier error counter */
        ULONG  ulRCEE;          /*!< Received carrier extension error counter */
} R_IN32_MIBMACIP_T;
```

(7) gerR_IN32_ClearMIB

Function	MIB information clear			
Call format	ERRCODE gerR_IN32_ClearMIB (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function clears MIB information.			

(8) gerR_IN32_GetPortAvailable

Function	Port enabled status acquisition			
Call format	ERRCODE gerR_IN32_GetPortAvailable (ULONG* pulPortAvailable)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	*pulPortAvailable	Port enabled status R_IN32_MYPORT_PORTALL(0x00): All owned ports enabled R_IN32_MYPORT_PORT_1(0x01): Only port 1 enabled R_IN32_MYPORT_PORT_2(0x02): Only port 2 enabled	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the enabled status of each port set in the master station.			

6.4.7 LED control

(1) gerR_IN32_SetLERR1LED

Function	LED control (L ER (port 1))			
Call format	ERRCODE gerR_IN32_SetLERR1LED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off the L ER LED of port 1. *: The LED cannot be turned on when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(2) gerR_IN32_SetLERR2LED

Function	LED control (L ER (port 2))			
Call format	ERRCODE gerR_IN32_SetLERR2LED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off the L ER LED of port 2. *: The LED cannot be turned on when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(3) gerR_IN32_SetERRLED

Function	LED control (ERR.)			
Call format	ERRCODE gerR_IN32_SetERRLED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on R_IN32_LED_BLINK: LED blinking	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off the ERR LED. *: The LED cannot be turned off or set to blinking when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(4) gerR_IN32_SetDLINKLED

Function	LED control (D LINK)			
Call format	ERRCODE gerR_IN32_SetDLINKLED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on R_IN32_LED_BLINK: LED blinking	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off the D LINK LED. *: The LED cannot be turned on or set to blinking when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(5) gerR_IN32_SetUSER1LED

Function	LED control (User LED 1)			
Call format	ERRCODE gerR_IN32_SetUSER1LED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on R_IN32_LED_BLINK: LED blinking	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off User LED 1. *: The LED cannot be turned on or set to blinking when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(6) gerR_IN32_SetUSER2LED

Function	LED control (User LED 2)			
Call format	ERRCODE gerR_IN32_SetUSER2LED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on R_IN32_LED_BLINK: LED blinking	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off User LED 2. *: The LED cannot be turned on or set to blinking when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(7) gerR_IN32_SetRUNLED

Function	LED control (RUN)			
Call format	ERRCODE gerR_IN32_SetRUNLED (ULONG ulCtrl)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulCtrl	LED control R_IN32_LED_OFF: LED off R_IN32_LED_ON: LED on	Input
Return value	R_IN32_OK: Normal end			
Description	This function turns on and off the RUN LED. *: The LED cannot be turned on when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(8) gerR_IN32_DisableLED

Function	LED control function disablement			
Call format	ERRCODE gerR_IN32_DisableLED (USHORT usBitPattern)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	usBitPattern	LED control function disablement (On: Disable, Off: Hold previous value) Bit 0: Disable RUN LED Bit 2: Disable User LED 2 Bit 4: Disable User LED 1 Bit 6: Disable D LINK LED Bit 8: Disable ERR. LED Bit10: Disable port 1 L ER LED Bit11: Disable port 2 L ER LED (Bits 1, 3, 5, 7, 9, and 12 to 15: Not used)	Input
Return value	R_IN32_OK: Normal end			
Description	This function disables the LED function. *: The function cannot be disabled when a R-IN32M3-CL internal WDT, external WDT, or own station error occurs.			

(9) gerR_IN32_EnableLED

Function	LED control function enablement			
Call format	ERRCODE gerR_IN32_EnableLED (USHORT usBitPattern)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	usBitPattern	LED control function enablement (On: Enable, Off: Hold previous value) Bit 0: Enable RUN LED Bit 2: Enable User LED 2 Bit 4: Enable User LED 1 Bit 6: Enable D LINK LED Bit 8: Enable ERR. LED Bit10: Enable port 1 L ER LED Bit11: Enable port 2 L ER LED (Bits 1, 3, 5, 7, 9, and 12 to 15: Not used)	Input
Return value	R_IN32_OK: Normal end			
Description	This function enables the LED function.			

6.4.8 Network time

(1) gerR_IN32_GetNetworkTime

Function	Network time (serial value) acquisition			
Call format	ERRCODE gerR_IN32_GetNetworkTime (USHORT *pusSerial)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	*pusSerial	Network time pusSerial[0]: Network time (bits 15-0) pusSerial[1]: Network time (bits 31-16) pusSerial[2]: Network time (bits 47-32)	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the network time (serial value in increments of 15.2587890625 µs given a starting point of January 1, 2000, 00:00:00).			

(2) gerR_IN32_SetNetworkTime

Function	Network time (serial value) setting			
Call format	ERRCODE gerR_IN32_SetNetworkTime (const USHORT *pusSerial)			
Arguments	Name	Variable Name	Description	I/O
	const USHORT	*pusSerial	Network time pusSerial[0]: Network time (bits 15-0) pusSerial[1]: Network time (bits 31-16) pusSerial[2]: Network time (bits 47-32)	Input
Return value	R_IN32_OK: Normal end			
Description	This function sets the network time (serial value in increments of 15.2587890625 µs given a starting point of January 1, 2000, 00:00:00).			

(3) gerR_IN32_NetworkTimeToDate

Function	Network time (serial value) to clock information conversion			
Call format	ERRCODE gerR_IN32_NetworkTimeToDate (R_IN32_TIMEINFO_T *pstTimeInfo, const USHORT *pusSerial)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32_TIMEINFO_T	*pstTimeInfo	Clock information	Output
	const USHORT	*pusSerial	Network time pusSerial[0]: Network time (bits 31-16) pusSerial[1]: Network time (bits 47-32)	Input
Return value	R_IN32_OK: Normal end			
Description	This function converts the network time (serial value in increments of seconds given a starting point of January 1, 2000, 00:00:00) to clock information [year/month/day/hour/minute/second/microsecond (fixed to 0)/day of the week].			

Arguments of gerR_IN32_NetworkTimeToDate

The following describes the structure of R_IN32_TIMEINFO_T based on the sample code

```
/* Clock information */
typedef struct R_IN32_TIMEINFO_TAG {
        USHORT    usYear;     /*!< Year (2000 - 2136)*/
        USHORT    usMonth;    /*!< Month ( 1 - 12)*/
        USHORT    usDay;      /*!< Day ( 1 - 31)*/
        USHORT    usHour;     /*!< Hour ( 0 - 23)*/
        USHORT    usMin;      /*!< Minute ( 0 - 59)*/
        USHORT    usSec;      /*!< Second ( 0 - 59)*/
        USHORT    usMsec;     /*!< msec ( 0 - 999)*/
        USHORT    usWday;     /*!< Day of the week (0 (Sunday) - 6 (Saturday))*/
} R_IN32_TIMEINFO_T;
```

(4) gerR_IN32_DateToNetworkTime

Function	Clock information to network time (serial value) conversion			
Call format	ERRCODE gerR_IN32_DateToNetworkTime (const R_IN32_TIMEINFO_T *pstTimeInfo, USHORT *pusSerial)			
Arguments	Name	Variable Name	Description	I/O
	const R_IN32_TIMEINFO_T	*pstTimeInfo	Clock information	Input
	USHORT	*pusSerial	Network time pusSerial[0]: Network time (bits 15-0) pusSerial[1]: Network time (bits 31-16) pusSerial[2]: Network time (bits 47-32)	Output
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function converts clock information (year/month/day/hour/minute/second) to network time (serial value in increments of seconds given a starting point of January 1, 2000, 00:00:00). (ausSerial[0]: Network time (bits 15-0) is fixed to 0.) *: A year other than 2000 to 2136 results in a R_IN32_ERR. The R-IN32M3-CL driver does not check for any errors other than the above. Implement error processing in the user program to ensure that there are no leap year or date errors.			

6.4.9 MDIO access

(1) gerR_IN32_EnableMACIPAccess

Function	MAC IP access enablement			
Call format	ERRCODE gerR_IN32_EnableMACIPAccess (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end (MDIO command end wait error)			
Description	This function enables MAC IP access. *: Shorten the period from MAC IP access enablement (gerR_IN32_EnableMACIPAccess) to MAC IP access disablement (gerR_IN32_DisableMACIPAccess) to the extent possible. (If the vendor uses interrupts, use the function with the interrupts disabled from MAC IP access enablement to MAC IP access disablement.) When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

(2) gerR_IN32_DisableMACIPAccess

Function	MAC IP access disablement			
Call format	ERRCODE gerR_IN32_DisableMACIPAccess (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function disables the MAC IP access.			

(3) gerR_IN32_WritePHY

Function	PHY internal register write			
Call format	ERRCODE gerR_IN32_WritePHY (ULONG ulPort, ULONG ulAddr, ULONG ulData)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulPort	Port subject to register writing R_IN32_PORT1(0): Port 1 R_IN32_PORT2(1): Port 2	Input
	ULONG	ulAddr	PHY register address	Input
	ULONG	ulData	Data to be written to PHY	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end (MDIO command end wait error)			
Description	This function writes to the PHY internal register in MDIO. *: Use this function during the period from MAC IP access enablement (gerR_IN32_EnableMACIPAccess) to MAC IP access disablement (gerR_IN32_DisableMACIPAccess). When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

(4) gerR_IN32_ReadPHY

Function	PHY internal register read			
Call format	ERRCODE gerR_IN32_ReadPHY (ULONG ulPort, ULONG ulAddr, ULONG *ulData)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulPort	Port subject to register reading R_IN32_PORT1(0): Port 1 R_IN32_PORT2(1): Port 2	Input
	ULONG	ulAddr	PHY register address	Input
	ULONG	*ulData	Data read from PHY	Output
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end (MDIO command end wait error)			
Description	This function reads the PHY internal register in MDIO. *: Use this function during the period from MAC IP access enablement (gerR_IN32_EnableMACIPAccess) to MAC IP access disablement (gerR_IN32_DisableMACIPAccess). When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

6.4.10 Transient reception processing

(1) gerR_IN32_MainReceiveTransient1

Function	Transient reception main processing 1			
Call format	ERRCODE gerR_IN32_MainReceiveTransient1 (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function acquires the transient frames received by R-IN32M3-CL.			

(2) gerR_IN32_MainReceiveTransient2

Function	Transient reception main processing 2			
Call format	ERRCODE gerR_IN32_MainReceiveTransient2 (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function delivers the received transient frames acquired by the function gerR_IN32_MainReceiveTransient1 to the user program using the callback function gerR_IN32_CallbackReceivedTransient.			

(3) gerR_IN32_EnableReceiveTransient

Function	Transient reception enable/disable setting for vendor reasons			
Call format	ULONG gerR_IN32_EnableReceiveTransient (BOOL blEnable)			
Arguments	Name	Variable Name	Description	I/O
	BOOL	blEnable	Reception enable/disable setting R_IN32_TRUE: Enable reception R_IN32_FALSE: Disable reception	Input
Return value	R_IN32_OK: Normal end			
Description	This function enables or disables transient reception for vendor reasons. When the return value of the function below created by the vendor is R_IN32_ERR, "Status of transient reception enable/disable setting for vendor reasons" is set to "Disable reception". Be sure to set the status to "Enable reception" using this function once reception becomes possible. gerR_IN32_CallbackReceivedTransient			

(4) gblR_IN32_GetReceiveTransientStatus

Function	Status acquisition of transient reception enable/disable setting for vendor reasons			
Call format	BOOL gblR_IN32_GetReceiveTransientStatus (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	Status of reception enable/disable setting R_IN32_TRUE: Reception enabled R_IN32_FALSE: Reception disabled			
Description	This function acquires the status of transient reception enable/disable setting for vendor reasons.			

(5) gerR_IN32_SetMACAddressTableData

Function	Node information distribution data (MAC address table) setting			
Call format	ERRCODE gerR_IN32_SetMACAddressTableData (UCHAR uchSeqNumber, R_IN32_MACADDRESSDATA_T *pstMacAddrDat)			
Arguments	Name	Variable Name	Description	I/O
	UCHAR	uchSeqNumber	Sequential distribution number (value range: 1 to 7)	Input
	R_IN32_MACADDRESSDATA_T	*pstMacAddrDat	Information such as MAC address (MAC address table)	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR_OUTOFRANGE: Node number out of range or sequential distribution number out of range			
Description	This function sets the information (MAC address table), such as the MAC address, acquired by node information distribution from the master station, and the sequential distribution number. *: Register the node number of the master station as 0x7D. If R_IN32_FALSE is set by the initial value of (g) Node information distribution request in B) R_IN32_UNITINIT_T initial setup of the function gerR_IN32_Initialize, this function does not need to be called.			

Arguments of gerR_IN32_SetMACAddressTableData

The following describes the structure of R_IN32_MACADDRESSDATA_T based on the sample code.

```
/* Information such as MAC address (MAC address table) */
typedef struct _R_IN32_MACADDRESSDATA_TAG {
        USHORT    usNodeNumber;                  /*!< Node number (1 to 120, master station: 0x7D) */
        UCHAR     uchTransientReceiveEnable;     /*!< Transient reception function
                                                 (R_IN32_ENABLE/R_IN32_DISABLE) */
        UCHAR     auchMacAddress[6];             /*!< MAC address */
} R_IN32_MACADDRESSDATA_T;
```

6.4.11 Transient send processing

(1) gerR_IN32_GetUnitInformation

Function	Unit information acquisition			
Call format	ERRCODE gerR_IN32_GetUnitInformation (R_IN32_UNITINFO_T *pstUnitInfo, R_IN32_UNITNETWORKSETTING_T*pstUnitNetworkSetting)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32_UNITINFO_T	*pstUnitInfo	Unit information	Output
	R_IN32_UNITNETWORK SETTING_T	*pstUnitNetworkSetting	Network operation setting	Output
Return value	R_IN32_OK: Normal end			
Description	This function acquires the setting information of the own station. The acquired setting information is used when creating Detailed node information acquisition response frame.			

Arguments of gerR_IN32_GetUnitInformation

The following describes the structure of R_IN32_UNITNETWORKSETTING_T based on the sample code.

```
/* Network operation setting */
typedef struct R_IN32_UNITNETWORKSETTING_TAG {
        ULONG      ulFrameSendCount;       /*!< No. of sends during token hold */
        ULONG      ulFrameSendInterval;    /*!< Frame send interval */
        ULONG      ulTokenSendCount;       /*!< No. of token sends */
} R_IN32_UNITNETWORKSETTING_T;
```

(2) gusR_IN32_GetNodeID

Function	Node ID acquisition			
Call format	USHORT gusR_IN32_GetNodeID (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	Node ID			
Description	This function acquires the node ID. The acquired node ID is used when performing transient send.			

(3) gerR_IN32_GetMulticastMACAddress

Function	Multicast MAC address acquisition			
Call format	ERRCODE gerR_IN32_GetMulticastMACAddress (UCHAR *puchMACAddr)			
Arguments	Name	Variable Name	Description	I/O
	UCHAR	*puchMACAddr	Multicast address When 13-34-56-78-90-AB is set, the following addresses are returned: puchMACAddr[0]: 0x13 puchMACAddr[1]: 0x34 puchMACAddr[2]: 0x56 puchMACAddr[3]: 0x78 puchMACAddr[4]: 0x90 puchMACAddr[5]: 0xAB	Output
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end [The multicast MAC address cannot be acquired due to no data link (data link disconnected).]			
Description	This function acquires the multicast MAC address. The acquired multicast MAC address is used as the destination address when transient send is performed to all nodes connected to the network.			

(4) gerR_IN32_GetUnicastMACAddress

Function	Unicast MAC address acquisition			
Call format	ERRCODE gerR_IN32_GetUnicastMACAddress (USHORT usNodeNumber,UCHAR *puchMACAddr)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	usNodeNumber	Node number (value range: 1 to 120, master station: 0x7D)	Input
	UCHAR	*puchMACAddr	Unicast address When 12-34-56-78-90-AB is set, the following addresses are returned: puchMACAddr[0]: 0x12 puchMACAddr[1]: 0x34 puchMACAddr[2]: 0x56 puchMACAddr[3]: 0x78 puchMACAddr[4]: 0x90 puchMACAddr[5]: 0xAB	Output
Return value	R_IN32_OK: Normal end R_IN32_ERR_NOENTRY: No entry R_IN32_ERR_OUTOFRANGE: Node number out of range			
Description	This function acquires the unicast MAC address corresponding to the node number from Node information distribution frame received from the master station. *: When there is no data link (data link disconnected), the unicast MAC address cannot be acquired. (The return value becomes R_IN32_ERR_NOENTRY.) Set the node number of the master station to 0x7D.			

(5) gerR_IN32_GetSendTransientBuffer

Function	Transient send buffer acquisition			
Call format	ERRCODE gerR_IN32_GetSendTransientBuffer (USHORT usSize, VOID** ppvSendBuffAddr, UCHAR *puchSendBuffNo, UCHAR *puchConnectionInfo)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	usSize	Send data size excluding DCS/FCS	Input
	VOID	**ppvSendBuffAddr	Transient send buffer address	Output
	UCHAR	*puchSendBuffNo	Transient send buffer number	Output
	UCHAR	*puchConnectionInfo	Transient connection information	Output
Return value	R_IN32_OK: Normal end (transient send buffer acquisition) R_IN32_ERR: Abnormal end (transient send buffer acquisition error)			
Description	This function inquires whether or not there is space in the transient send area for send of the "send data size", and returns the following information if there is space: ・Transient send buffer address ・Transient send buffer number ・Transient connection information *: In the following case, transient send cannot be performed and the process ends in error (R_IN32_ERR: Abnormal end): ・When there is no data link (data link disconnected) ・When the send data size is greater than 1510 bytes When you want to perform transient send, execute the following: ・Acquire the transient send buffer number using this function. ・Store the send data in the acquired transient send buffer. ・Request transient send using the function gerR_IN32_RequestSendingTransient.			

(6) gerR_IN32_RequestSendingTransient

Function	Transient send request			
Call format	ERRCODE gerR_IN32_RequestSendingTransient (UCHAR uchSendBuffNo, USHORT usSize)			
Arguments	Name	Variable Name	Description	I/O
	UCHAR	uchSendBuffNo	Transient send buffer number	Input
	USHORT	usSize	Send data size excluding DCS/FCS	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end (transient send request error)			
Description	This function specifies send to the transient send buffer number acquired by the function gerR_IN32_GetSendTransientBuffer. Before executing this function, perform the following: ・Acquire the transient send buffer using the function gerR_IN32_GetSendTransientBuffer. ・Store the send data in the acquired transient send buffer. *: In the following case, transient send cannot be performed and the process ends in error (R_IN32_ERR: Abnormal end): ・When there is no data link (data link disconnected) Any error that occurs after send is requested by this function is notified by the return value of the function gerR_IN32_MainSendTransient. Set the send data size to the same size as the value specified in gerR_IN32_GetSendTransientBuffer.			

(7) gerR_IN32_MainSendTransient

Function	Transient send main processing			
Call format	ULONG gerR_IN32_MainSendTransient (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	This function acquires the transient send end result. This function calls the function gerR_IN32_CallbackTransientSendingComplete to issue a notification regarding the status (send result) of the target send descriptor.			

(8) gulR_IN32_ SetOptionInfo_Response

Function	Option information acquisition response frame creation processing			
Call format	ULONG gulR_IN32_SetOptionInfo_Response (VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA, const USHORT usSupportFunction)			
Arguments	Name	Variable Name	Description	I/O
	VOID*	pvSendFrame	Address of send frame	Output
	const VOID*	pvReceivedData	Address of received data storage area	Input
	const UCHAR*	puchSA	Send source node MAC address	Input
	const USHORT	usSupportFunction	SLMP support status USER_SUPPORT_FUNCTION (1): SLMP supported	Input
Return value	Send data size (excluding DCS/FCS)			
Description	This function creates Option information acquisition response frame. Specify USER_SUPPORT_FUNCTION (1) in SLMP support status (usSupportFunction). For the sample code, do not change SLMP support status value since USER_SUPPORT_FUNCTION (1) is specified by default.			

(9) gulR_IN32_SetSelectInfo_Response

Function	Selected station information acquisition response frame creation processing			
Call format	ULONG gulR_IN32_SetSelectInfo_Response (VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA, const USER_LED_INFO* pstUserLedInfo)			
Arguments	Name	Variable Name	Description	I/O
	VOID*	pvSendFrame	Address of send frame	Output
	const VOID*	pvReceivedData	Address of received data storage area	Input
	const UCHAR*	puchSA	Send source node MAC address	Input
	const USER_SELECTINFO_LED_INFO_T*	pstUserLedInfo	Own station LED information [LED color] USER_SELECTINFO_LED_UNUSED(0): LED not used USER_SELECTINFO_LED_GREEN(1): Green USER_SELECTINFO_LED_RED(2): Red USER_SELECTINFO_LED_ORANGE(3) : Orange [LED status] USER_SELECTINFO_LED_UNUSED(0): LED not used USER_SELECTINFO_LED_OFF(1): Off USER_SELECTINFO_LED_ON(2): On USER_SELECTINFO_LED_BLINK(3): Blinking	Input
Return value	Send data size (excluding DCS/FCS)			
Description	This function creates Selected station information acquisition response frame. To display the LED information of the own station, specify the own station LED information corresponding to the own station status.			

Arguments of gulR_IN32_SetSelectInfo_Response

The following describes the structure of USER_SELECTINFO_LED_INFO_T based on the sample code.

```
/* Own station LED information */
typedef struct_USER_SELECTINFO_LED_INFO_TAG {
        UCHAR               uchRow;             /* No. of LED array rows */
        UCHAR               uchColumn;          /* No. of LED array columns */
        USER_LED_INFO_T     stLedInf[8];        /* LED information 1 to 8 */
} USER_SELECTINFO_LED_INFO_T;
```

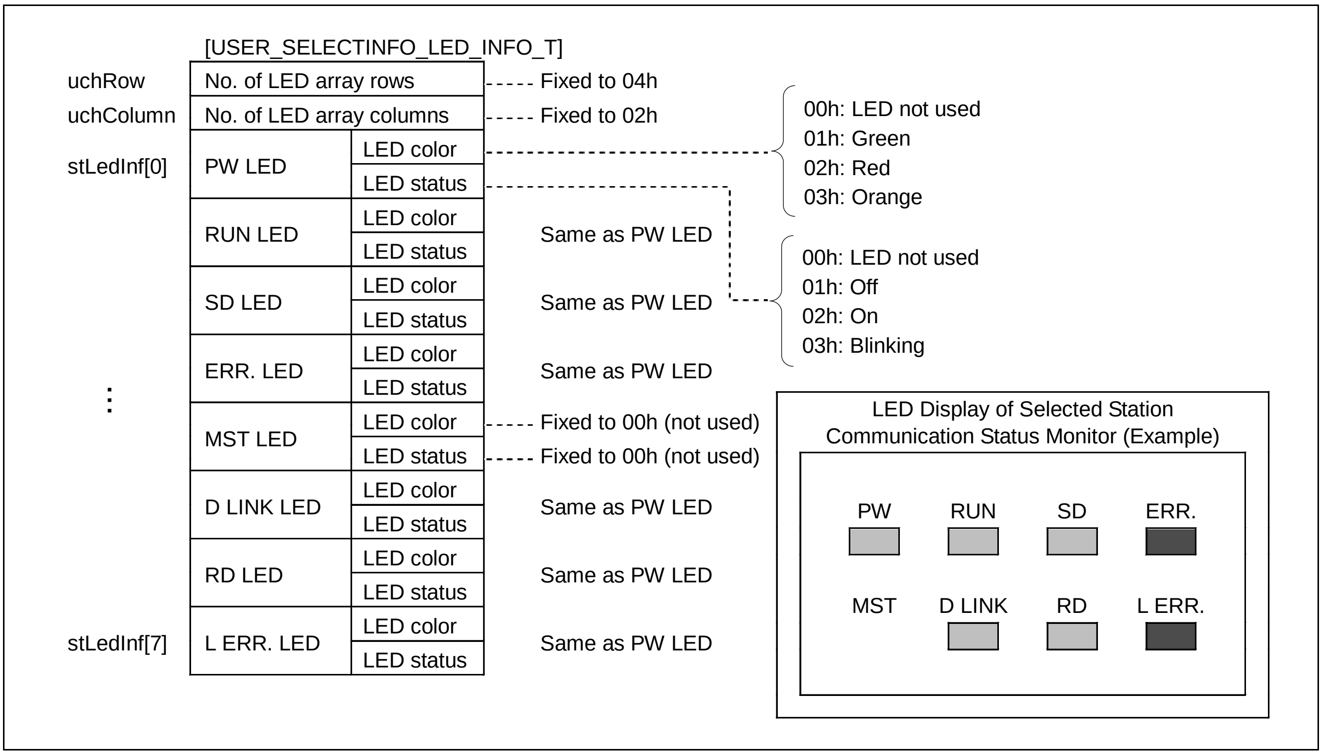


Figure 6.53 Own Station LED Information

Caution. When the actual LED status (on/off/blinking) changes at an interval shorter than the communication

interval of selected station information acquisition, the change in the LED status is not transmitted to the engineering tool.

(When the Mitsubishi Electric engineering tool is used, the communication interval of selected station information acquisition is approximately 5 seconds.)

In this case, the LED indication in the diagnostic window differs from the actual LED status.

Example: The status of LED which is repeatedly turned on/off at high-speed such as SD and RD LEDs changes at shorter intervals than the communication interval of selected station information acquisition. Therefore, the LED indication in the diagnostic window differs from the actual LED status.

(10) gulR_IN32_SetSlmpError_Response

Function	SLMP error response frame creation processing			
Call format	ULONG gulR_IN32_SetSlmpError_Response (VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA, const USHORT usFinCode)			
Arguments	Name	Variable Name	Description	I/O
	VOID*	pvSendFrame	Address of send frame	Output
	const VOID*	pvReceivedData	Address of received data storage area	Input
	const UCHAR*	puchSA	Send source node MAC address	Input
	const USHORT	usFinCode	End code 0x0000: Normal end 0x0001 to 0xFFFF: Error code (user-defined)	Input
Return value	Send data size (excluding DCS/FCS)			
Description	This function creates SLMP command error response frame. For the end code, the error code is specified by the server to the request frame sent from the client. 1) When the own station is a client, during the response frame receive processing, the error code of an error detected in the request frame sent by the own station is stored. 2) When the own station is a server, during the response frame send processing, specify the error code of an error detected in the request frame sent by the client.			

(11) gulR_IN32_SetContactTest_Response

Function	Communication test response frame creation processing			
Call format	ULONG gulR_IN32_SetContactTest_Response (VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA)			
Arguments	Name	Variable Name	Description	I/O
	VOID*	pvSendFrame	Address of send frame	Output
	const VOID*	pvReceivedData	Address of received data storage area	Input
	const UCHAR*	puchSA	Send source node MAC address	Input
Return value	Send data size (excluding DCS/FCS)			
Description	This function creates Communication test response frame.			

(12) gulR_IN32_SetCableTest_Response

Function	Cable test response frame creation processing			
Call format	ULONG gulR_IN32_SetCableTest_Response (VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA)			
Arguments	Name	Variable Name	Description	I/O
	VOID*	pvSendFrame	Address of send frame	Output
	const VOID*	pvReceivedData	Address of received data storage area	Input
	const UCHAR*	puchSA	Send source node MAC address	Input
	const USER_CABLETEST_ RESULT_T*	pstTestResult	Results [Number of ports] gulR_IN32U_MAX_PORT_NUMBER: No. of ports of own station [Results of the cable test] USER_CABLE_TEST_OK(0): Cable normal USER_CABLE_TEST_NG(2): Cable disconnected, or not connected	Input
Return value	Send data size (excluding DCS/FCS)			
Description	This function creates Cable test response frame. Specify the number of ports and the cable test results.			

Arguments of gulR_IN32_SetCableTest_Response

The following describes the structure of USER_CABLETEST_RESULT_T based on the sample code.

```
/* SLMP cable test (for response) frame format */
typedef struct _USER_CABLETEST_RESULT_TAG {
    USHORT    usPortNum;                                          /* No. of ports */
    USHORT    auchPortResult[USER_CABLE_TEST_RESULT_MAX];         /* Results */
} USER_CABLETEST_RESULT_T;
```

(13) gulR_IN32_SetMemRead_Response

Function	SLMP memory read response frame creation processing			
Call format	ULONG gulR_IN32_SetMemRead_Response (VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA)			
Arguments	Name	Variable Name	Description	I/O
	VOID*	pvSendFrame	Address of send frame	Output
	const VOID*	pvBufferMemory	Address of buffer memory	Output
	const VOID*	pvReceivedData	Address of received data storage area	Input
	const UCHAR*	puchSA	Send source node MAC address	Input
Return value	Send data size (excluding DCS/FCS)			
Description	This function creates SLMP memory read response frame.			

(14) gulR_IN32_SetMemWrite_Response

<table>
<tr><td>Function</td><td colspan="4">SLMP memory write response frame creation processing</td></tr>
<tr><td>Call format</td><td colspan="4">ULONG gulR_IN32_SetMemWrite_Response
(VOID* pvSendFrame, const VOID* pvReceivedData, const UCHAR* puchSA)</td></tr>
<tr><td rowspan="4">Arguments</td><td>Name</td><td>Variable Name</td><td>Description</td><td>I/O</td></tr>
<tr><td>VOID*</td><td>pvSendFrame</td><td>Address of send frame</td><td>Output</td></tr>
<tr><td>const VOID*</td><td>pvReceivedData</td><td>Address of received data storage area</td><td>Input</td></tr>
<tr><td>const UCHAR*</td><td>puchSA</td><td>Send source node MAC address</td><td>Input</td></tr>
<tr><td>Return value</td><td colspan="4">Send data size (excluding DCS/FCS)</td></tr>
<tr><td>Description</td><td colspan="4">This function creates SLMP memory write response frame.</td></tr>
</table>

6.4.12 Interrupts

(1) gerR_IN32_DisableInterrupt

Function	Interrupt disablement			
Call format	ERRCODE gerR_IN32_DisableInterrupt (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	Interrupt disablement			

(2) gerR_IN32_EnableInterrupt

Function	Interrupt enablement			
Call format	ERRCODE gerR_IN32_EnableInterrupt (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end			
Description	Interrupt enablement			

6.4.13 Hardware test

(1) gerR_IN32_IEEETest

Function	IEEE 802.3ab compliance test			
Call format	ERRCODE gerR_IN32_IEEETest (USHORT usMode)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	usMode	IEEE 802.3ab compliance test mode R_IN32_IEEE_MODE1(1): MODE1 R_IN32_IEEE_MODE2(2): MODE2 R_IN32_IEEE_MODE3(3): MODE3 R_IN32_IEEE_MODE4(4): MODE4 R_IN32_IEEE_END(5): Test end	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function sets the waveform output for test mode in PHY in accordance with the IEEE 802.3ab compliance test mode of the argument. Within this function, gerR_IN32R_IEEETest (refer to Section 6.5.2 "Creating the R-IN32M3-CL driver target-dependent functions") is called. Be sure to customize gerR_IN32R_IEEETest in accordance with the specifications of the PHY used. *: When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

(2) gerR_IN32_InitializeLoopBackTest

Function	Internal/external loopback communication test initialization			
Call format	ERRCODE gerR_IN32_InitializeLoopBackTest (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function performs Initialization processing for executing the internal/external loopback communication test. *: When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

(3) gerR_IN32_InternalLoopBackTest

Function	Internal loopback communication test			
Call format	ERRCODE gerR_IN32_InternalLoopBackTest (ULONG ulPort)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulPort	Test target port R_IN32_PORT1(0): Port 1 R_IN32_PORT2(1): Port 2	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function sends a frame from the test target port specified in the argument, and verifies the received result by internal loopback. *: When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

(4) gerR_IN32_ExternalLoopBackTest

Function	External loopback communication test			
Call format	ERRCODE gerR_IN32_ExternalLoopBackTest (ULONG ulPort)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulPort	Test target port R_IN32_PORT1(0): Port 1 R_IN32_PORT2(1): Port 2	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function sends a frame from the test target port specified in the argument, and verifies the received result using the other port. When implementing this test, connect port 1 and port 2 using an Ethernet cable. *: When the return value of this function is a value other than R_IN32_OK, the function calls the following function created by the vendor. Be sure to execute error processing in accordance with the error code. gR_IN32_CallbackFatalError			

6.5 Customizing the R-IN32M3-CL Driver Target-Dependent Functions

6.5.1 Changing the header file

Change each item defined in the header file "R_IN32M3Function.h" in accordance with the system environment of the vendor.

(1) R-IN32M3-CL address setting

1) R-IN32M3-CL start address

Specifies the address for R-IN32M3-CL access by the R-IN32M3-CL driver.

```
#define    R_IN32_BASE_ADR       0x0FA00000        /* R-IN32M3-CL start address */
```

(2) PHY reset setting

Defines the setup for resetting PHY during initialization.

1) PHY reset assert time setting

Sets the time at which the R-IN32M3-CL driver is to assert the PHY reset signal in units of μs.
The assertion time varies depending on the PHY used. Refer to the manual of the PHY used.

```
#define    R_IN32_WAITUS_PHYRESET_ASSERT   10000UL   /* PHY reset assertion time */
```

2) Time after PHY reset clear to normal operation

Specifies the time after PHY reset is cleared by the R-IN32M3-CL driver to normal PHY operation, in units of μ s.
The time after reset clear to normal PHY operation varies depending on the PHY used. Refer to the manual of the PHY used.

```
#define    R_IN32_WAITUS_PHYRESET_END        5000UL    /* Time after PHY reset clear
                                                          to normal operation */
```

(3) Number of transient reception buffers

Defines the number of transient reception buffers.
The R-IN32M3-CL driver uses an area (memory) equivalent to R_IN32_TRANSIENT_BUFFER_NUM × 1520 bytes.
Set a value greater than or equal to 2.

```
#define    R_IN32_TRANSIENT_BUFFER_NUM      (64)         /* No. of transient  reception buffers */
```

6.5.2 Creating the R-IN32M3-CL driver target-dependent functions

Caution. Be sure to implement the target-dependent functions described in Table 6.14 "R-IN32M3-CL Driver Target-Dependent Function List".

The R-IN32M3-CL driver target-dependent functions must be customized in accordance with the target hardware environment. The following lists the functions to be customized by the vendor.

Table 6.14 R-IN32M3-CL Driver Target-Dependent Function List

Function Category	Function Name	Function Type	Overview
Wait processing	gR_IN32R_WaitUS	VOID	Time wait
Time measurement	gR_IN32R_StartStopwatchTimer	VOID	Time measurement start
	gR_IN32R_GetElapsedTime	VOID	Elapsed time acquisition
Interrupt	gR_IN32R_DisableInt	VOID	Interrupt disablement
	gR_IN32R_EnableInt	VOID	Interrupt enablement
Hardware test	gerR_IN32R_IEEETest	ERRCODE	IEEE 802.3ab compliance test

(1) gR_IN32R_WaitUS

Function	Time wait			
Call format	VOID gR_IN32R_WaitUS (ULONG ulWaitTime)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulWaitTime	Waiting time (µs)	Input
Return value	None			
Description	This function waits for the waiting time specified in the argument to elapse. *: The maximum waiting time used by the R-IN32M3-CL driver is 10 ms (10000 UL). If the assertion time of the used PHY is longer than 10 ms (10000 UL), change the value below so that that value can be counted: #define R_IN32_WAITUS_PHYRESET_ASSERT 10000UL /* PHY reset assertion time */ (For details, refer to Section 6.5.1(2) "PHY reset setting".)			

(2) gR_IN32R_StartStopwatchTimer

Function	Time measurement start			
Call format	VOID gR_IN32R_StartStopwatchTimer (R_IN32R_STOPWATCH_T *pstStopWatch,ULONG ulUnit)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32R_STOPWATCH_T*	pstStopWatch	Stopwatch work area	I/O
	ULONG	ulUnit	Measurement unit (1: µs)	Input
Return value	None			
Description	This function starts time measurement.			

Arguments of gR_IN32R_StartStopwatchTimer

The following describes the configuration of R_IN32R_STOPWATCH_T based on the sample code.

```
typedef struct _R_IN32R_STOPWATCH_TAG {
        ULONG       ulUnit;                /* Unit of measured time */
        ULONG       ulFirstTmr1Cnt;        /* General-purpose timer 1 counter value (at startup) */
        ULONG       ulLastTmr1Cnt;         /* General-purpose timer 1 counter value (previous value) */
} R_IN32R_STOPWATCH_T;
```

(3) gR_IN32R_GetElapsedTime

Function	Elapsed time acquisition			
Call format	VOID gR_IN32R_GetElapsedTime (R_IN32R_STOPWATCH_T *pstStopWatch, ULONG *pulElapsedTime)			
Arguments	Name	Variable Name	Description	I/O
	R_IN32R_STOPWATCH_T	*pstStopWatch	Stopwatch work area	I/O
	ULONG	*pulElapsedTime	Elapsed time (Unit: The unit specified by the function gR_IN32R_StartStopwatchTimer)	Output
Return value	None			
Description	This function acquires the elapsed time after Time measurement start function gR_IN32R_StartStopwatchTimer is called. The R-IN32M3-CL driver monitors timeouts using the functions gR_IN32R_StartStopwatchTimer and gR_IN32R_GetElapsedTime. *: Implement the function so that Unsigned Long (0 to 4294967295) can be counted. If timeout monitoring is not required, set *ulElapsedTime to "0" (elapsed time: 0 μs).			

(4) gR_IN32R_DisableInt

Function	Interrupt disablement			
Call format	VOID gR_IN32R_DisableInt (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	None			
Description	Interrupt disablement *:This function is a dummy function. Regard the processing as no processing.			

(5) gR_IN32R_EnableInt

Function	Interrupt enablement			
Call format	VOID gR_IN32R_EnableInt (VOID)			
Arguments	Name	Variable Name	Description	I/O
	None			
Return value	None			
Description	Interrupt enablement *:This function is a dummy function. Regard the processing as no processing.			

(6) gerR_IN32R_IEEETest

Function	IEEE 802.3ab compliance test			
Call format	ERRCODE gerR_IN32R_IEEETest (USHORT usIEEETestMode)			
Arguments	Name	Variable Name	Description	I/O
	USHORT	usIEEETestMode	IEEE 802.3ab compliance test mode R_IN32R_IEEE_MODE1(1): MODE1 R_IN32R_IEEE_MODE2(2): MODE2 R_IN32R_IEEE_MODE3(3): MODE3 R_IN32R_IEEE_MODE4(4): MODE4 R_IN32R_IEEE_END(5): Test end	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function sets the waveform output for test mode in PHY in accordance with the IEEE 802.3ab compliance test mode of the argument. This function assumes that the PHY "88E111-B2-BAB1C100" of Marvell Semiconductor is used. If you use a different PHY, customize the function in accordance with the PHY specifications.			

6.6 Customizing the R-IN32M3-CL Driver Callback Functions

The internal processing of R-IN32M3-CL driver callback functions needs to be customized by the vendor. The following describes the callback functions called by the R-IN32M3-CL driver.

Table 6.15 R-IN32M3-CL Driver Callback Function List

Function Category	Function Name	Function Type	Overview
Error processing	gR_IN32_CallbackFatalError	VOID	R-IN32M3-CL fatal error acquisition
Own station status acquisition	gerR_IN32_CallbackCommandFromMaster	ERRCODE	Command acquisition from master station
Transient send/ receive	gerR_IN32_CallbackReceivedTransient	ERRCODE	Received transient frame acquisition
	gerR_IN32_CallbackTransientSendingComplete	ERRCODE	Transient send completion status acquisition

(1) gR_IN32_CallbackFatalError

Function	R-IN32M3-CL fatal error acquisition			
Call format	VOID gR_IN32_CallbackFatalError (ULONG ulErrorCode, ULONG ulErrorInfo)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	ulErrorCode	Fatal error code	Input
	ULONG	ulErrorInfo	Fatal error information (Address of function when error occurred)	Input
Return value	None			
Description	This function acquires R-IN32M3-CL fatal errors. The R-IN32M3-CL driver calls this function when a R-IN32M3-CL fatal error is detected. Function internal processing is freely implemented by the vendor.			

Table 6.16 List of Fatal Error Codes of gR_IN32_CallbackFatalError Function

Fatal Error Code (ulErrorCode)	Fatal Error Information (ulErrorInfo)	Fatal Error Description	Action
D529	Driver internal call source function Address of the function gerR_IN32D_ClearTxRxRAM	Communication LSI error	・The error is most likely a malfunction caused by interference such as noise. Check the distance between lines and cables as well as device grounding, and implement noise countermeasures accordingly. ・Implement a module unit test. If the error occurs again, most likely the hardware is faulty.
D52A	Driver internal call source function Address of the function erR_IN32D_MDIO_WaitCommandComplete	Communication LSI error	
D52B	Driver internal call source function Address of the function erR_IN32D_ResetMAC	Communication LSI error	
D52C	Driver internal call source function Address of the function gerR_IN32D_StartRing	Communication LSI error	

(2) gerR_IN32_CallbackCommandFromMaster

Function	Command acquisition from master station			
Call format	ERRCODE gerR_IN32_CallbackCommandFromMaster (ULONG pulCommand)			
Arguments	Name	Variable Name	Description	I/O
	ULONG	pulCommand	Command status from master station ulCommand Bit 0: Cyclic transmission stop instruction (node number out of range) 1: Stop instruction Bit 1: Cyclic transmission stop instruction (reserved node setting) 1: Stop instruction Bit 2: Cyclic transmission stop instruction (master station instruction) 1: Stop instruction Bit 3: Cyclic transmission stop instruction (node number duplication) 1: Stop instruction Bits 15 to 4: Reserved Bit 16: Node type invalid (own station node type does not match node type specified by master station.) 1: Node type invalid Bit 17: Specified size invalid (The cyclic transmission size specified by the master station is greater than the allowable maximum size (size specified by the function gerR_IN32_Initialize) for own station cyclic transmission.) 1: Specified size invalid Bits 31 to 18: Reserved	Input
Return value	R_IN32_OK: Normal end			
Description	This function acquires commands by Parameter frame reception from the master station. The R-IN32M3-CL driver calls this function when Parameter frame is received from the master station. Function internal processing is freely implemented by the vendor.			

(3) gerR_IN32_CallbackReceivedTransient

Function	Received transient frame acquisition			
Call format	ERRCODE gerR_IN32_CallbackReceivedTransient (VOID *pvRcv, USHORT usFrameSize)			
Arguments	Name	Variable Name	Description	I/O
	VOID	*pvRcv	Reception buffer	Input
	USHORT	usFrameSize	Frame size excluding FCS	Input
Return value	R_IN32_OK: Normal end R_IN32_ERR: Abnormal end			
Description	This function acquires received transient frames. The R-IN32M3-CL driver calls this function when a transient frame is received. Function internal processing is freely implemented by the vendor. *: Set the start address of the reception buffer in increments of 4 bytes (0 or multiple of 4). When the return value is a value other than R_IN32_OK, "Status of transient reception enable/disable setting for vendor reasons" is set to "Disable reception". Be sure to set the status to "Enable reception" using the function gerR_IN32_EnableReceiveTransient once reception becomes possible.			

(4) gerR_IN32_CallbackTransientSendingComplete

Function	Transient send completion status acquisition			
Call format	ERRCODE gerR_IN32_CallbackTransientSendingComplete (UCHAR uchSendBuffNo, ERRCODE erSendStatus)			
Arguments	Name	Variable Name	Description	I/O
	UCHAR	uchSendBuffNo	Transient send buffer number	Input
	ERRCODE	erSendStatus	Status of target transient send buffer (send result) R_IN32_OK: Transient send normal completion R_IN32_ERR: Transient send abnormal completion	Input
Return value	R_IN32_OK: Normal end			
Description	This function acquires the send status (send result) of the transient send buffer. The R-IN32M3-CL driver calls this function when send of a transient frame ends. Function internal processing is freely implemented by the vendor.			

7. LINK DEVICE SYSTEM AREA

A part of link devices in an intelligent device station connected to the CC-Link IE Field Network can be defined as a system area. A system area is used to notify other stations of the status of the own station and to instruct operation from the master station to the own station.

Defining a part of link devices as a system area is optional. To define a system area, assign the bits of the link device as indicated in Table 7.1 "System Area Bit Assignments (Example)".

The following table shows an example of defining a system area for the remote input (RX) and the remote output (RY). When defining a system area for the remote registers (RWr, RWw), replace RX with RWr and RY with RWw.

Table 7.1 System Area Bit Assignments (Example)

	Bit	Name	Bit	Name
System area	RX(S+0)	Reserved	RY(S+0)	Reserved
	RX(S+1)		RY(S+1)	
	RX(S+2)		RY(S+2)	
	RX(S+3)		RY(S+3)	
	RX(S+4)		RY(S+4)	
	RX(S+5)		RY(S+5)	
	RX(S+6)		RY(S+6)	
	RX(S+7)	Warning status flag	RY(S+7)	
	RX(S+8)	Initial data processing request flag	RY(S+8)	Initial data processing complete flag
	RX(S+9)	Initial data setting complete flag	RY(S+9)	Initial data setting request flag
	RX(S+A)	Error status flag	RY(S+A)	Error reset request flag
	RX(S+B)	Remote ready	RY(S+B)	Reserved
	RX(S+C)	Reserved	RY(S+C)	
	RX(S+D)		RY(S+D)	
	RX(S+E)		RY(S+E)	
	RX(S+F)		RY(S+F)	

Remark. S : Start number of system area

If you define a part of link devices as a system area, describe the definition information of the link devices in the CC-Link Control & Communication System Profile.

7.1 System Area Details

The following describes the details on each bit of the system area using the remote input (RX) and the remote output (RY) as an example.

(1) Remote ready: RX(S+B)

This bit indicates that data can be sent and received between the master station and the own station.

Turn on the bit after power-on or hardware reset.

Turn off the bit when data cannot be sent or received between the master station and the own station due to Error status flag.

However, leave the bit on when Warning status flag is on.

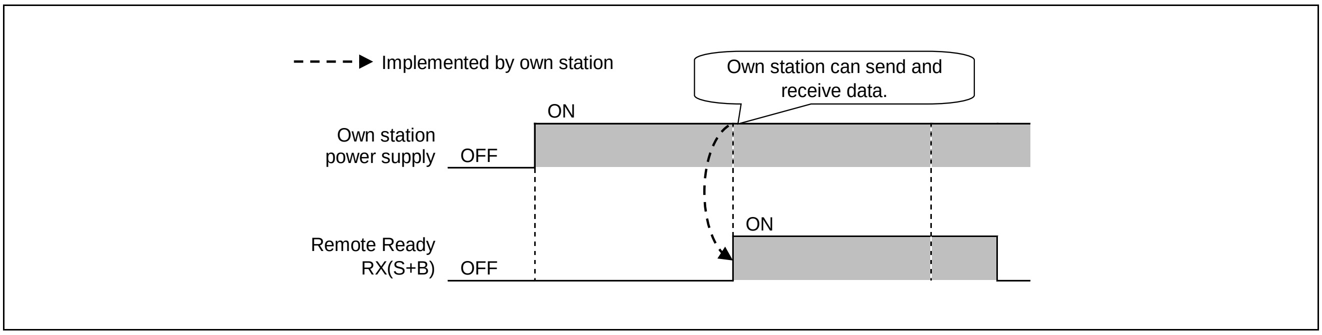

Implemented by own station
Own station can send and receive data.
ON
Own station power supply
OFF
ON
Remote Ready RX(S+B)
OFF

Figure 7.1 Timing Chart: Remote ready

(2) Initial data processing request flag: RX(S+8), Initial data processing complete flag: RY(S+8)

These bits are used to request initial data processing from the own station to the master station after power-on or hardware reset of the own station.

After the initial data processing completes, turn on Remote ready.

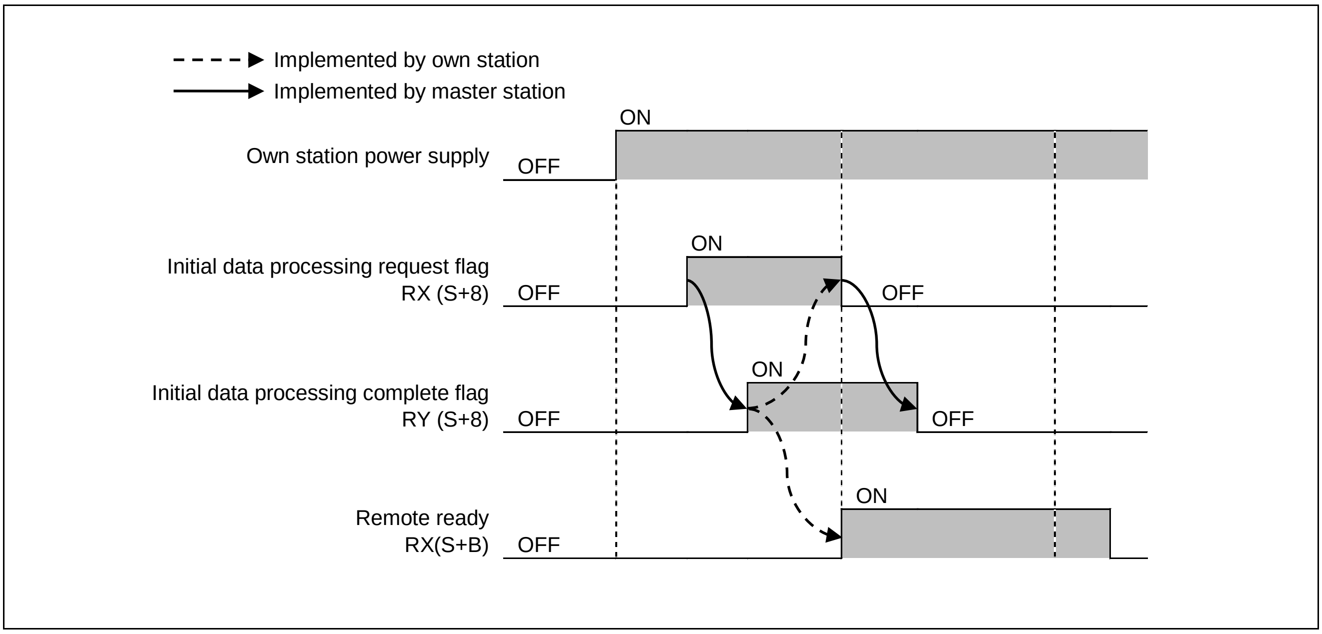


Figure 7.2 Timing Chart: Initial Data Processing Request/Complete Flag

(3) Initial data setting complete flag: RX(S+9), Initial data setting request flag: RY(S+9)

These bits are used to request initial data setting from the master station to the own station.
After the initial data are set, turn on Remote ready.

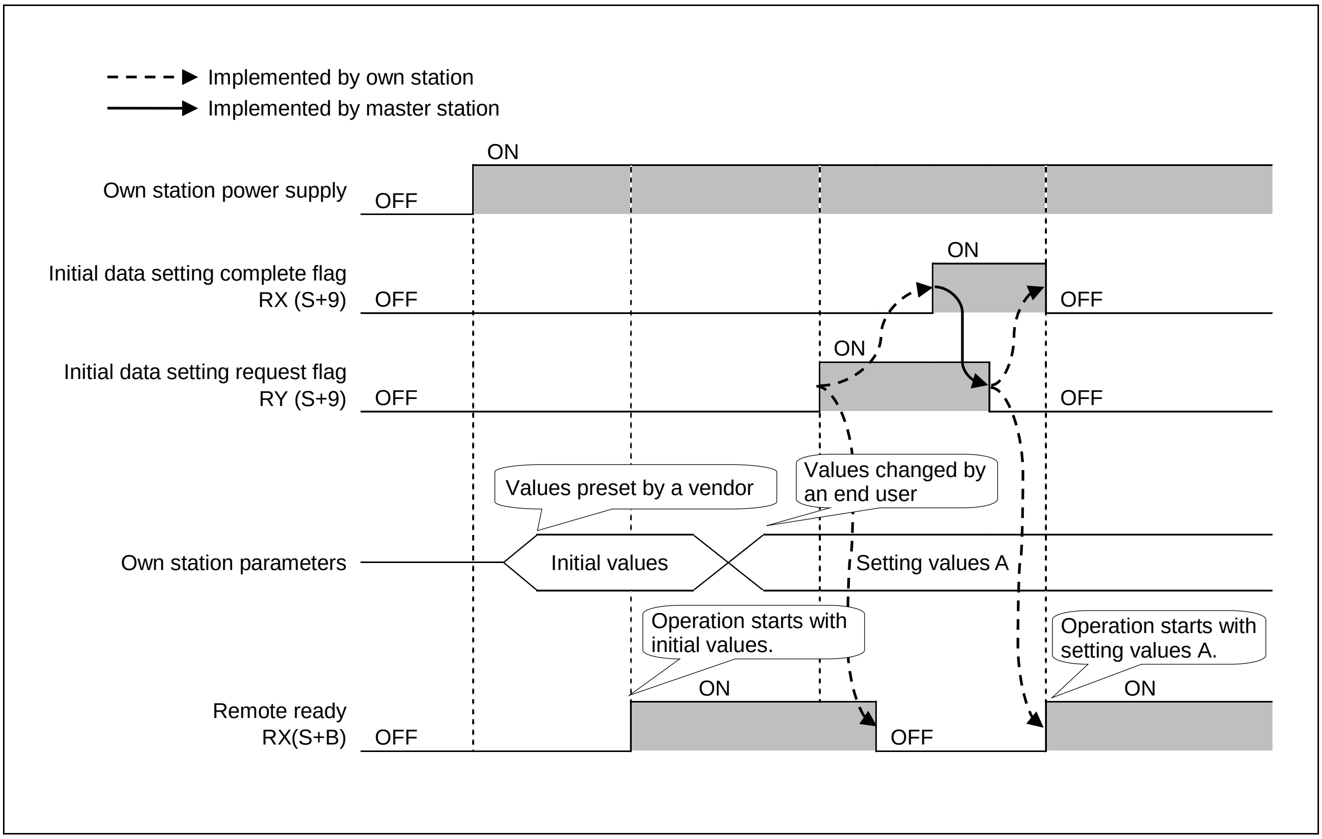

Implemented by own station
Implemented by master station
Own station power supply
OFF
ON
Initial data setting complete flag RX (S+9)
OFF
ON
OFF
Initial data setting request flag RY (S+9)
OFF
ON
OFF
Values preset by a vendor
Values changed by an end user
Own station parameters
Initial values
Setting values A
Operation starts with initial values.
Operation starts with setting values A.
Remote ready RX(S+B)
OFF
ON
OFF
ON

Figure 7.3 Timing Chart: Initial Data Setting Complete/Request Flag

(4) Implementation of Initial data processing request/complete flag and Initial data setting complete/request flag

When these flags are implemented, turn on Remote ready after both the initial data processing and the initial data setting processing complete.

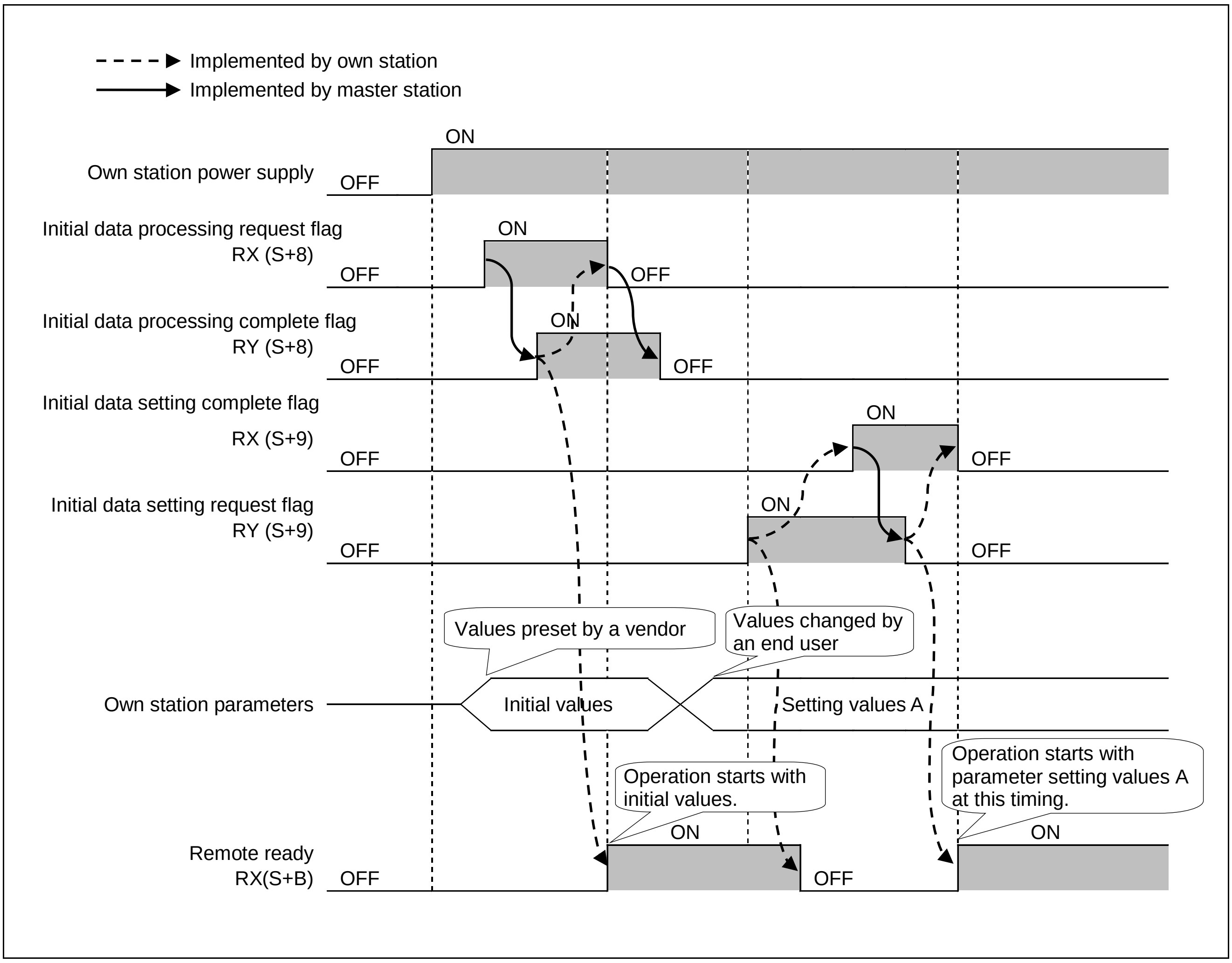

Implemented by own station
Implemented by master station
Own station power supply
ON
OFF
Initial data processing request flag
RX (S+8)
Initial data processing complete flag
RY (S+8)
Initial data setting complete flag
RX (S+9)
Initial data setting request flag
RY (S+9)
Values preset by a vendor
Values changed by an end user
Own station parameters
Initial values
Setting values A
Operation starts with initial values.
Operation starts with parameter setting values A at this timing.
Remote ready
RX(S+B)

Figure 7.4 Timing Chart: Initial Data Processing and Setting

(5) Error status flag: RX(S+A), Error reset request flag: RY(S+A)

These bits are used to notify or clear a moderate/major error of the own station. (The station can no longer continue its operation.)

Turn on Error status flag when a moderator/major error occurs in the own station.

The master station clears the error status and turns on the Error reset request flag.

The own station turns off Error status flag and clears the error code storage area.

Turn off Remote ready from error occurrence to error clear.

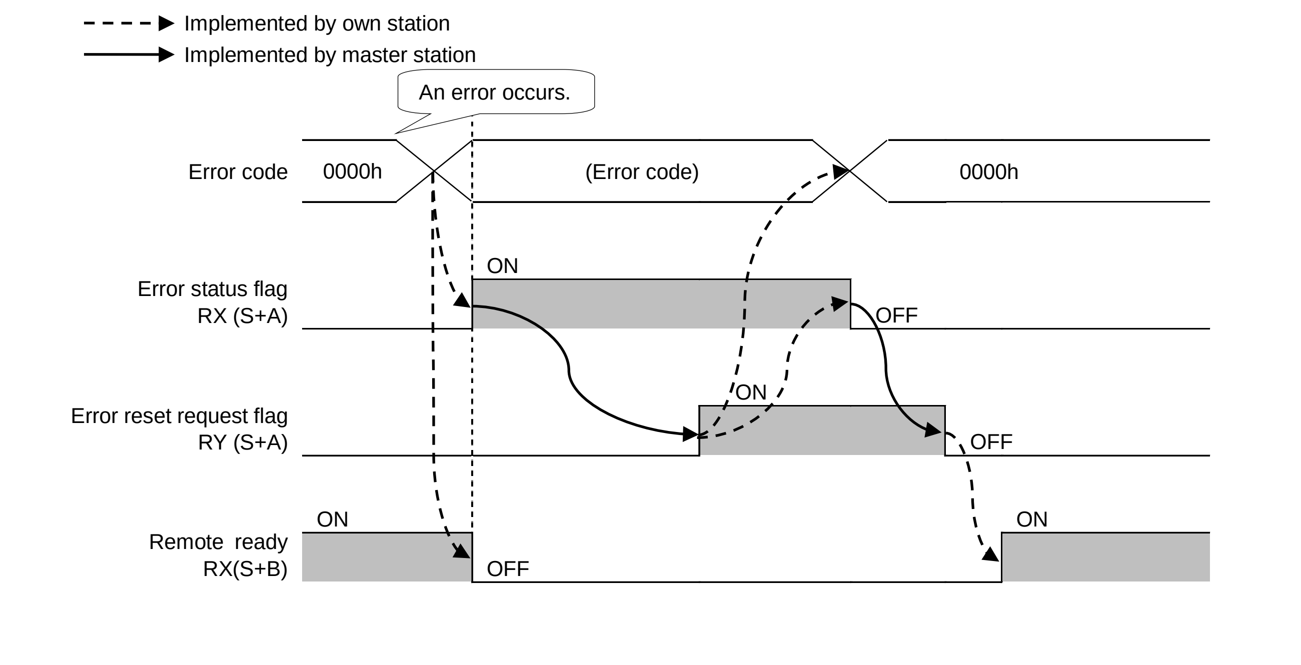


Figure 7.5 Timing Chart: Error Status Flag, Error Reset Request Flag

(6) Warning status flag: RX(S+7)

This bit is used to notify a minor error of the own station. (The station can continue its operation.)
Turn on this flag when a minor error occurs in the own station.
When the master station eliminates the error cause, the own station clears the warning code and turn off this flag.
Leave Remote ready on from warning occurrence to warning clear.

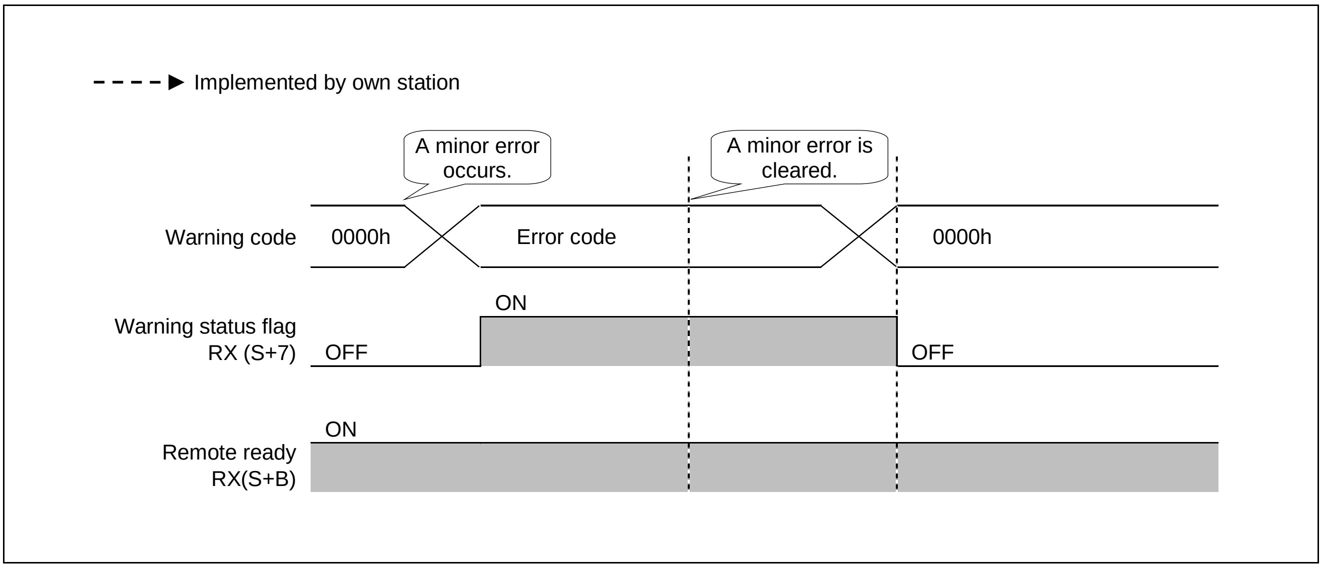

- - - -► Implemented by own station
A minor error occurs.
A minor error is cleared.
Warning code
0000h
Error code
0000h
Warning status flag RX (S+7)
OFF
ON
OFF
Remote ready RX(S+B)
ON

Figure 7.6 Timing Chart: Warning Status Flag

REVISION HISTORY | R-IN32M3 Series CC-Link IE Field Intelligent device station

Rev.	Date	Description	
		Page	Summary
1.00	Jul 26, 2013	-	First edition issued
2.00	Dec 25, 2014	2	Modification of PHY address 2 of Table 1.2 Circuit Design Check Sheet
		6	Modification of product name of Figure 2.1 External AND Logic for Turning L.ERR On
		9	Modification of product name Figure 3.1 Transient1 Response Procedure (Request Source: Master Station)
		10	Modification of product name Figure 3.2 Transient2 Response Procedure
		140	Modification of header file name and R-IN32M3-CL head address of 4.6.1 Changing the header file
5.00	Jan 31, 2018	-	Full-fledged revision Overall changes : - Changed structure of Chapters Added chapter ... Chapter 1,2 and 7 Deleted chapter ... Chapter 5 before the change Changed with contents ... Chapter 4,5 and 6 (Chapter 2,3 and 4 before the change) Changed without contents ... Chapter 3 (Chapter 5 before the change) - Unified format and wording (Without changing contents) - Changed description and expression of sentences and figures (Without changing contents)
		21-24	Chapter 4 "STATUS DISPLAY FUNCTION" : - Changed structure of sections Detailed the note in Section 4.1, and created Section 4.1.1 - Detailed description of contents of Section 4.1, 4.2.1 and 4.3
		25-76	Chapter 5 "DATA COMMUNICATION METHOD OF CC-LINK IE FIELD NETWORK" : - Changed structure of sections - Detailed the part of Section 5.2 and creaded Section 5.2.1, 5.2.3 and 5.2.4 - Added Section 5.2.2 (Description of Transmit1 request sending procedure) - Added Section 5.3.5 (Description of SLMP frame format) - Added Section 5.4 (Description of MyStatus) - Detailed the contents of Section 5.3.1, and separated into Section 5.3.1 and new Section 5.3.2 - Detailed description of contents - Added description of Option information acquisition of Transient1, SLMP and MyStatus - Changed frame name to distinguish between the previous Transient 1 frame and SLMP Form "Transient1" to "CC-Link IE Field specific transient transmission" From "Transient2" to "CC-Link compatible transient transmission"
		77-80	Section 6.1 "Development Procedure" : - Changed structure of sections Deleted the previous Section 6.3 and created Section 6.1.1 - Detailed description of contents

		81-142	Section 6.2 "Sample Flowcharts" : - Changed structure of sections - Added Section 6.2.27 (Processing of Option information acquisition request frame receive) - Added Section 6.2.28 (Processing of Selected station information acquisition request frame receive) - Added Section 6.2.29 (Processing of Communication test request frame receive) - Added Section 6.2.30 (Processing of Cable test request frame receive) - Added Section 6.2.39 (Processing of SLMP memory read request frame receive) - Added Section 6.2.40 (Processing of SLMP memory write request frame receive) - Added Section 6.2.41 (Processing of SLMP memory read request frame creation) - Added Section 6.2.42 (Processing of Transient1 request send division determination) - Added Section 6.2.43 (Processing of Transient1 request frame creation) - Added Section 6.2.44 (Processing of SLMP memory read response receive)
			- Added the above additional processing to the list of sample flowcharts - Changed the processing name from "General flowchart" to "Main processing" - Added examples of own station errors as notes in each section where own error is described - Added a note about "gblUserMACAddressTableRequest" to Section 6.2.2 - Added description of Hold/Clear process and note to Section 6.2.9 - Added description of Hold/Clear processing and note to Section 6.2.12 - Added description of sending data by dividing data into blocks to Section 6.2.17 - Added description of SLMP request reception from master station to Section 6.2.18 - Added description of troubleshooting based on Loopback Communication Test to Section 6.2.46
		143-145	Section 6.3 "Interface Function List for R-IN32M3-CL Driver" : - Added the following functions to the R-IN32M3-CL driver interface function list gerR_IN32_GetPortAvailable, gulR_IN32_SetOptionInfo_Response, gulR_IN32_SetSelectInfo_Response, gulR_IN32_SetSlmpError_Response, gulR_IN32_SetContactTest_Response, gulR_IN32_SetCableTest_Response, gulR_IN32_SetMemRead_Response, gulR_IN32_SetMemWrite_Response
		146-194	Section 6.4 "R-IN32M3-CL Driver Interface Function Details" : - Changed structure of sections - Added Section 6.4.6 (8) (gerR_IN32_GetPortAvailable function) - Added Section 6.4.11 (8) (gulR_IN32_SetOptionInfo_Response function) - Added Section 6.4.11 (9) (gulR_IN32_SetSelectInfo_Response function) - Added Section 6.4.11 (10) (gulR_IN32_SetSlmpError_Response function) - Added Section 6.4.11 (11) (gulR_IN32_SetContactTest_Response function) - Added Section 6.4.11 (12) (gulR_IN32_SetCableTest_Response function) - Added Section 6.4.11 (13) (gulR_IN32_SetMemRead_Response function) - Added Section 6.4.11 (14) (gulR_IN32_SetMemWrite_Response function)

			- Modified description field in Section 6.4.1 (2) - Added / deleted the following structure members to R_IN32_UNITINFO_T whichi is the argument of Section 6.4.1 (2) Added : usHwVersion, usDeviceVersion Deleted : blNodeAndNetworkNumberFromMasterPermission - Added ((n)(o)) / deleted description of initial setting of above members in Section 6.4.1 (2) A) - Changed description of initial setting of the following in 6.4.1 (2) A) - (e) ... Changed the setting to "2 or 1" - (f) ... Changed the setting to "23μsec" - (h) ... Added description - Added note about the device version of CSP+ - Added supplement information of network and controller - Added supplement information of device version - Added / modified the following structure members to R_IN32_UNITINIT_T whichi is argument of Section 6.4.1 (2) Added : ulOptionSupport, ulSlmpSupport, ulSlmpDiagnosisSupport Modified : ulErrorStatus, ulErrorCode - Added ((l)(m)(n)) / modified ((i)(j)) description of initial setting of above members in Section 6.4.1 (2) B) - Changed description of initial setting of the following in 6.4.1 (2) B) - (f) ... Changed the setting to "R_IN32_TRUE" - (g) ... Detailed description - Detailed description field in Section 6.4.1 (3) - Added contents to description field in Section 6.4.13 (1)
		195-198	Section 6.5 "Customizing the R-IN32M3-CL Driver Target-Dependent Functions" : - Detailed description field in Section 6.5.2 (6)
		199-201	Section 6.6 "Customizing the R-IN32M3-CL Driver Callback Functions" : - Removed "gerR_IN32_CallbackNodeAndNetworkNumber" function from the R-IN32M3-CL driver callback function list - Deleted the section describing the above function

[Memo]

R-IN32M3 Series User’s Manual:
(CC-Link IE Field Intelligent device station)

Publication Date:	Rev.1.00	Jul 26, 2013
	Rev.5.00	Jan 31, 2018

Published by: Renesas Electronics Corporation

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 5.0

R-IN32M3 Series

User's Manual

(CC-Link IE Field Intelligent device station)

Renesas Electronics Corporation