MITSUBISHI ELECTRIC

Programmable Controller

MELSEC iQ-R series

MELSEC iQ-R MELSECWinCPU Module Programming Manual

-R102WCPU-W

SAFETY PRECAUTIONS

(Read these precautions before using this product.)

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

Make sure that the end users read this manual and then keep the manual in a safe place for future reference.

CONDITIONS OF USE FOR THE PRODUCT

(1) MELSEC programmable controller ("the PRODUCT") shall be used in conditions;

i) where any problem, fault or failure occurring in the PRODUCT, if any, shall not lead to any major or serious accident; and

ii) where the backup and fail-safe function are systematically or automatically provided outside of the PRODUCT for the case of any problem, fault or failure occurring in the PRODUCT.

(2) The PRODUCT has been designed and manufactured for the purpose of being used in general industries.

MITSUBISHI ELECTRIC SHALL HAVE NO RESPONSIBILITY OR LIABILITY (INCLUDING, BUT NOT LIMITED TO ANY AND ALL RESPONSIBILITY OR LIABILITY BASED ON CONTRACT, WARRANTY, TORT, PRODUCT LIABILITY) FOR ANY INJURY OR DEATH TO PERSONS OR LOSS OR DAMAGE TO PROPERTY CAUSED BY the PRODUCT THAT ARE OPERATED OR USED IN APPLICATION NOT INTENDED OR EXCLUDED BY INSTRUCTIONS, PRECAUTIONS, OR WARNING CONTAINED IN MITSUBISHI ELECTRIC USER'S, INSTRUCTION AND/OR SAFETY MANUALS, TECHNICAL BULLETINS AND GUIDELINES FOR the PRODUCT.

("Prohibited Application")

Prohibited Applications include, but not limited to, the use of the PRODUCT in;

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

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

(3) Mitsubishi Electric shall have no responsibility or liability for any problems involving programmable controller trouble and system trouble caused by DoS attacks, unauthorized access, computer viruses, and other cyberattacks.

CONSIDERATIONS FOR USE

For products manufactured by Microsoft® Corporation in the United States

A MELSECWinCPU module is equipped with Windows® 10 IoT Enterprise manufactured by Microsoft Corporation in the United States as OS. For using the module, our company does not have any responsibility for problems and damage caused by a product manufactured by Microsoft Corporation in the United States.

For the problems or specifications of the Microsoft Corporation product, refer to the corresponding manual or consult Microsoft Corporation.

Contact information is available on the following website:

- Microsoft Corporation: support.microsoft.com/en-us/contactus

INTRODUCTION

Thank you for purchasing the Mitsubishi Electric MELSEC iQ-R series programmable controllers.

This manual describes the functions required for programming.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the MELSEC iQ-R series programmable controller to handle the product correctly.

Please make sure that the end users read this manual.

CONTENTS

RELEVANT MANUALS

Manual name [manual number]	Description	Available form
MELSEC iQ-R MELSECWinCPU Module Programming Manual [SH-082433ENG] (this manual)	Programming specifications and dedicated function library of a MELSECWinCPU module	e-Manual PDF
MELSEC iQ-R MELSECWinCPU Module User's Manual [SH-082431ENG]	Performance specifications, procedure before operation, functions, devices, parameters, and troubleshooting of a MELSECWinCPU module	Print book e-Manual PDF
CW Configurator Operating Manual [SH-081382ENG]	System configuration, parameter settings, and operation method for the online function of CW Configurator	e-Manual PDF
MELSEC iQ-R Module Configuration Manual [SH-081262ENG]	The combination of the MELSEC iQ-R series modules, common information on the installation/wiring in the system, and specifications of the power supply module, base unit, SD memory card, and battery	Print book e-Manual PDF

Point

e-Manual refers to the Mitsubishi Electric FA electronic book manuals that can be browsed using a dedicated tool.

e-Manual has the following features:

- Required information can be cross-searched in multiple manuals.
- Other manuals can be accessed from the links in the manual.
- Hardware specifications of each part can be found from the product figures.
- Pages that users often browse can be bookmarked.
- Sample programs can be copied to an engineering tool.

TERMS

Unless otherwise specified, this manual uses the following terms.

Term	Description
Bus control	A processing unit that can control communication with another CPU module or a control module via a bus
C Controller module dedicated function	A dedicated function library provided by this product. In this product, it is used to clear errors and acquire module configuration information.
Intelligent function module	A module that has functions other than an input or output, such as an A/D converter module and D/A converter module
MELSEC data link function	A dedicated function library provided by this product. It is used to access this product and another CPU module which is set as a connection target via network or in a multiple CPU system.

GENERIC TERMS AND ABBREVIATIONS

Unless otherwise specified, this manual uses the following generic terms and abbreviations.

Generic term/abbreviation	Description
CC-Link IE TSN master/local module	RJ71GN11-T2
CC-Link IE TSN module	A CC-Link IE TSN master/local module
CC-Link IE Controller Network module	Includes the following: • RJ71GP21-SX CC-Link IE Controller Network module • RJ71GP21S-SX CC-Link IE Controller Network module
CC-Link IE module	A CC-Link IE Controller Network module and a CC-Link IE TSN module
CC-Link module	RJ61BT11
CPU module	A MELSEC iQ-R series CPU module
GOT	Mitsubishi Graphic Operation Terminal
R102WCPU-W	An R102WCPU-W MELSECWinCPU module
MELSECWinCPU module	A MELSEC iQ-R series MELSECWinCPU module
Network module	Includes the following: • CC-Link IE Controller Network modules • CC-Link IE TSN modules • CC-Link modules
Bus interface	A MELSEC iQ-R bus interface
Bus interface communication	MELSEC iQ-R bus interface communication
Bus interface function	A dedicated function library provided by a MELSEC-Q series MELSECWinCPU module
Base unit	A main base unit, an extension base unit, and an RQ extension base unit
Dedicated function library	C Controller module dedicated functions and MELSEC data link functions
Power supply module	A MELSEC iQ-R series power supply module
I/O module	An input module, an output module, an I/O combined module, and an interrupt module

1 DEVELOPMENT ENVIRONMENT

This chapter explains the environment for program development.

1.1 Tool

The following development tools can be used to develop programs.

- Microsoft® Visual Studio® 2019 (Enterprise, Professional)
- Microsoft Visual Studio 2017 (Enterprise, Professional)

1.2 Programming Language

The following programming languages can be used to develop user applications for a MELSECWinCPU module.

- C/C++
- Visual Basic®
- C#

MEMO

2 PROGRAMMING

This chapter explains programming-related specifications to develop user applications.
This product supports the development of 32-bit user applications only.

Point

Only the essential procedures for developing MELSECWinCPU module programs are explained. For general development procedures of Visual Studio, refer to manuals corresponding to the Microsoft Corporation product or consult Microsoft Corporation.

2.1 Programming Creating Procedure

This section shows the procedure for creating programs by using a library.

1. Install a development tool (development environment for Windows).

☞ Page 7 Tool

2. Copy a library file and a header file.

Each of the files is stored in a folder of a MELSECWinCPU module as shown below.

Drive name	Folder name		Description
C	WinCPU	INCLUDE	Folder to store header files
		LIB	Folder to store library files

3. Create a program project.

Select Windows for the platform.

4. Set an environment for using a library.

☞ Page 9 Registering a library file

5. Create a program.

- ☞ Page 11 C Controller Module Dedicated Functions
- ☞ Page 12 MELSEC Data Link Functions

Registering a library file

The following explains the procedure for registering a library file to a development tool.

Visual C++

■Registering a header file

1. Select "Solution Explorer" ⇨ a project name, then right-click it and select [Properties] from the shortcut menu.

2. Select the following items in the "Property Pages" screen, and click "<Edit>" for "Include Directories."

- Configuration: All Configurations
- Platform: All Platforms
- Configuration Properties: VC++ Directories

3. Click the [] button in the "Include Directories" screen, and click the [...] button.

4. Select a folder where a Visual C++® header file is stored in the "Select Directory" screen, and click the [Select Folder] button.

5. Check that the folder selected in the "Include Directories" screen is added, and click the [OK] button.

■Registering a library file

1. Select the following items in the "Property Pages" screen, and click "<Edit>" for "Library Directories."

- Configuration: All Configurations
- Platform: Win32
- Configuration Properties: VC++ Directories

2. Click the [] button in the "Library Directories" screen, and click the [...] button.

3. Select a folder where a C++ (32 bit) development library file is stored in the "Select Directory" screen, and click the [Select Folder] button.

4. Check that the folder selected in the "Library Directories" screen is added, and click the [OK] button.

5. Select the following items in the "Property Pages" screen, and click "<Edit>" for "Additional Dependencies."

- Configuration: All Configurations
- Platform: All Platforms
- Configuration Properties: "Input" under "Linker"

6. Enter any of the following library file names in the "Additional Dependencies" screen.

Enter a name for each library to be used.

Library name	Library file name
C Controller module dedicated function	CCPUFuncWinCPU32.lib
MELSEC data link function	MDFuncWinCPU32.lib

Point

To register two libraries at the same time, enter the library file names using a semicolon (;), which is shown as below:
(Example) CCPUFuncWinCPU32.lib;MDFuncWinCPU32.lib

7. Check that the entered file name is registered in the "Property Pages" screen, and click the [OK] button.

Visual Basic and Visual C#

1. Select "Solution Explorer" ⇨ a project name, then right-click it and select [Add] ⇨ [Existing Item] from the shortcut menu.

2. Select a header file of Visual Basic or Visual C#® in the "Add Existing Item" screen and click the [Add As Link] from the [Add] button.

Development language	Library name	Header file
Visual Basic	C Controller module dedicated function	CCPUFuncWinCPU.vb
	MELSEC data link function	MDFuncWinCPU.vb
Visual C#	C Controller module dedicated function	CCPUFuncWinCPU.cs
	MELSEC data link function	MDFuncWinCPU.cs

3. Check that the selected header file is added under the selected project name in "Solution Explorer."

2.2 C Controller Module Dedicated Functions

C Controller module dedicated functions are a dedicated function library for controlling a MELSECWinCPU module.
In a MELSECWinCPU module, the functions are used for clearing errors, acquiring module configuration information, etc.

Program processing

The following shows the processing of a user program with C Controller module dedicated functions.

1. Errors are cleared and system configuration information is acquired with C Controller module dedicated functions.
2. Target devices continue to be accessed as long as a user program keeps running.
   The user program will terminate when it stops accessing the target devices.

Considerations

The following shows the considerations when using C Controller module dedicated functions.

Considerations for programming

■Header file

When creating programs in C or C++, include the Windows header file (windows.h).

■Passing arguments (IN/OUT) by reference

When creating programs in Visual C# using the IN/OUT or OUT argument, add a keyword shown in the following table to the argument and pass it by reference.

Argument (IN/OUT)	Keyword
IN/OUT	ref
OUT	out

■Same interrupt event number

When executing the CCPU_WaitEvent function and the CCPU_WaitUnitEvent function, do not use the same interrupt event number for multiple processes and threads.

Number of user programs which can be executed simultaneously

When running a user program which uses any of the following functions, maximum 64 processes and threads can be executed simultaneously.
Exceeding the maximum number may result in the extended processing time of other processes and threads and in the extended time for timeout detection.

- CCPU_ClearError function
- CCPU_GetSerialNo function
- CCPU_GetUnitInfo function
- CCPU_Reset function
- CCPU_WaitEvent function
- CCPU_WaitUnitEvent function
- mdReceiveEx function (message receive)
- mdSendEx function (message send)
- mdBdDevSetEx function
- mdBdDevRstEx function
- mdBdSendEx function
- mdBdReceiveEx function
- mdBdRandREx function
- mdBdRandWEx function
- mdBdReadLinkDeviceEx function
- mdBdWriteLinkDeviceEx function

2.3 MELSEC Data Link Functions

MELSEC data link functions are an integrated communication library which is independent of the communication protocols. A program for accessing devices of this product and for accessing the device memory of another programmable controller CPU can be created.

The communication functions supported by the MELSEC data link functions are as follows:

Communication function	Description	Reference
Bus interface communication	To access this product or a CPU module mounted on the same base unit	Page 16 Bus interface communication
CC-Link IE Controller Network communication	To access a CPU module on CC-Link IE Controller Network via a CC-Link IE Controller Network module	Page 20 CC-Link IE Controller Network communication
CC-Link IE TSN communication	To access a CPU module on CC-Link IE TSN via a CC-Link IE TSN module	Page 26 CC-Link IE TSN communication
CC-Link communication	To access a CPU module on CC-Link via a CC-Link module	Page 29 CC-Link communication

Program processing

The following shows the user program processing with a MELSEC data link function.

Point

For a path acquired by executing the mdOpen function, make sure to close its communication line by executing the mdClose function.

1. A communication line is opened. (mdOpen function)
2. Devices of the own station or other stations are accessed with MELSEC data link functions.
3. Target devices continue to be accessed as long as a program keeps running.
   The program will proceed to the next step to end the program.
4. The communication line is closed. (mdClose function)

Multi-thread communications

■Executing the mdOpen function and the mdClose function

When using MELSEC data link functions from multiple threads within the same process, execute the mdOpen function at the start of the process, and then use the path, which is acquired by the mdOpen function, for each thread.

In addition, execute the mdClose function after terminating all threads.

Executing the mdClose function without terminating all threads may cause the following symptom:

- A MELSEC data link function error occurs in the other threads.

Number of user programs which can be executed simultaneously

When running a user program which uses any of the following functions, maximum 64 processes and threads can be executed simultaneously.

Exceeding the maximum number may result in the extended processing time of other processes and threads and in the extended time for timeout detection.

- CCPU_ClearError function
- CCPU_GetSerialNo function
- CCPU_GetUnitInfo function
- CCPU_Reset function
- CCPU_WaitEvent function
- CCPU_WaitUnitEvent function
- mdReceiveEx function (message receive)
- mdSendEx function (message send)
- mdBdDevSetEx function
- mdBdDevRstEx function
- mdBdSendEx function
- mdBdReceiveEx function
- mdBdRandREx function
- mdBdRandWEx function
- mdBdReadLinkDeviceEx function
- mdBdWriteLinkDeviceEx function

Considerations

The following shows the considerations when using MELSEC data link functions.

Considerations for programming

■Header file

When creating programs in C or C++, include the Windows header file (windows.h).

■Open/close processing of a communication line (the mdOpen/mdClose function)

Perform the open and close processing of communication line (the mdOpen function and the mdClose function) only once at the start and end of each user program. Opening and closing the line for each communication decreases the communication performance.

■Initial access using the transient transmission function

When initially accessing other stations using the transient transmission function, the function acquires programming controller devices from a CPU on other stations.
Therefore, the initial function execution time will be prolonged.

■Total number of access target stations by the transient transmission function

Accessing 257 or more other stations using the transient transmission function may decrease the communication performance. Reduce the total number to 256 or less.

■Effects from Windows and other applications

The operation of a user program may be affected by Windows processing and other applications.
Create a program by considering that the execution time and interval of MELSEC data link functions may be longer.

■Forced termination during execution of MELSEC data link functions

When forcibly terminating a user application in which MELSEC data link functions are executed, the following cases may occur.

- A user application to be forcibly terminated cannot be ended.
- An error occurs in a MELSEC data link function being executed in other applications.
- Engineering tools manufactured by Mitsubishi Electric Corporation are affected.

■Passing arguments (IN/OUT) by reference

When creating programs in Visual C# using the IN/OUT or OUT argument, add a keyword shown in the following table to the argument and pass it by reference.

Argument (IN/OUT)	Keyword
IN/OUT	ref
OUT	out

■Data inconsistency

Data inconsistency may occur if a write or read data value with the size exceeding 65000 bytes is specified with functions shown below.

Reduce the write or read data size to less than 65000 bytes. Alternatively, create an interlock program.

- mdBdSendEx function
- mdBdReceiveEx function
- mdBdRandREx function
- mdBdRandWEx function
- mdBdReadLinkDeviceEx function
- mdBdWriteLinkDeviceEx function

■Executing a MELSEC data link function simultaneously from multiple user programs

When executing a MELSEC data link function simultaneously from multiple user programs, a request for the transient transmission processing of the next user program is sent after that processing of the previously executed user program is completed. Therefore, executing multiple user programs simultaneously may lead to extended time from the execution to the completion of a MELSEC data link function in some of the user programs.

Corresponding MELSEC data link functions are shown as below:

- mdControl function
- mdDevRstEx function
- mdDevSetEx function
- mdGetLabelInfo function
- mdRandREx function
- mdRandRLabelEx function
- mdRandWEx function
- mdRandWLabelEx function
- mdReceiveEx function
- mdSendEx function
- mdTypeRead function

■Communication timeout value of the MELSEC data link function setting function

A communication timeout value which is set using the MELSEC data link function setting function is not a timeout time from the execution to the completion of a MELSEC data link function. It is a timeout time to monitor the time taken from when a request is sent to an access target to when the response is returned via transient transmission. Set an appropriate time according to the time when an access target system returns the response.

Corresponding MELSEC data link functions are shown as below:

- mdControl function
- mdDevRstEx function
- mdDevSetEx function
- mdGetLabelInfo function
- mdRandREx function
- mdRandRLabelEx function
- mdRandWEx function
- mdRandWLabelEx function
- mdReceiveEx function
- mdSendEx function
- mdTypeRead function

Accessible range and accessible devices

This section explains the accessible range and accessible devices when using MELSEC data link functions.
The accessible range of the devices varies depending on specifications and settings of an access target module. For the accessible range, check the specifications of the access target module.

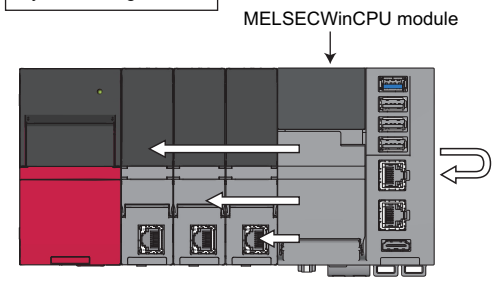
Bus interface communication

The following explains the range and devices accessible for bus interface communication.

■Accessible range

An accessible range for bus interface communication includes the own station (a MELSECWinCPU module), and a CPU module and C Controller module in a multiple CPU system.

System configuration



■Accessible devices

The devices accessible for communication via a bus are shown below.

Point

- 'Batch' and 'Random' in the following table indicate as follows:
  Batch: Batch write (mdSendEx function), batch read (mdReceiveEx function)
  Random: Random write (mdRandWEx function), random read (mdRandREx function), bit set (mdDevSetEx function), bit reset (mdDevRstEx function), random write by using a label name (mdRandWLabelEx function), random read by using a label name (mdRandRLabelEx function)
- Only bit devices can be accessed by bit set (mdDevSetEx function) and bit reset (mdDevRstEx function).
- The fixed scan communication area can be accessed only when the multiple CPU setting is configured.
- Device extension specifications (digit specification, bit specification, and index specification) cannot be used.

Accessing the host CPU

The following table shows the accessible devices when accessing the host CPU.

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU
			R102WCPU-W
Input relay	X	Batch/random	○
Output relay	Y	Batch/random	○
Internal relay	M	Batch/random	○
Special relay	SM	Batch/random	○
Data register	D	Batch/random	○
Special register	SD	Batch/random	○
Link relay	B	Batch/random	○
Link register	W	Batch/random	○
Intelligent function module device, module access device	Un\G	Batch/random	○
CPU buffer memory	U3En\G	Batch	○
		Random	×
Fixed scan communication area	U3En\HG	Batch	○
		Random	×

Point

When accessing the host CPU with a function whose name starts with 'mdBd,' the devices shown above can be accessed.

- mdBdDevRstEx function
- mdBdDevSetEx function
- mdBdRandREx function
- mdBdRandWEx function
- mdBdReceiveEx function
- mdBdSendEx function

Accessing other CPUs

The following tables show the accessible devices when accessing other CPUs (a CPU module and a C Controller module in a multiple CPU system).

No.	Access target CPU
(1)	RnCPU
(2)	R12CCPU-V

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU	
			(1)	(2)
Input relay	X	Batch/random	○	○
Output relay	Y	Batch/random	○	○
Latch relay	L	Batch/random	○	×
Internal relay	M	Batch/random	○	○
Special relay	SM	Batch/random	○	○
Annunciator	F	Batch/random	○	×
Timer (contact)	T	Batch/random	○	×
Long timer (contact)	LT	Batch/random	○	×
Timer (coil)	T	Batch/random	○	×
Long timer (coil)	LT	Batch/random	○	×
Counter (contact)	C	Batch/random	○	×
Long counter (contact)	LC	Batch/random	○	×
Counter (coil)	C	Batch/random	○	×
Long counter (coil)	LC	Batch/random	○	×
Timer (current value)	T	Batch/random	○	×

Device		Access method	Access target CPU	
			(1)	(2)
Long timer (current value)	LT	Batch/random	○	×
Counter (current value)	C	Batch/random	○	×
Long counter (current value)	LC	Batch/random	○	×
Data register	D	Batch/random	○	○
Special register	SD	Batch/random	○	○
Index register	Z	Batch/random	○	×
Long index register	LZ	Batch/random	○	×
File register	R	Batch/random	○	×
	ZR	Batch/random	○	○
Refresh data register	RD	Batch/random	○	×
Link relay	B	Batch/random	○	○
Link register	W	Batch/random	○	○
Link special relay	SB	Batch/random	○	×
Retentive timer (contact)	ST	Batch/random	○	×
Long retentive timer (contact)	LST	Batch/random	○	×
Retentive timer (coil)	ST	Batch/random	○	×
Long retentive timer (coil)	LST	Batch/random	○	×
Link special register	SW	Batch/random	○	×
Edge relay	V	Batch/random	○	×
Own station random access buffer	—	Batch/random	×	×
Retentive timer (current value)	ST	Batch/random	○	×
Long retentive timer (current value)	LST	Batch/random	○	×
Remote register for sending	RWw	Batch/random	×	×
Remote register for receiving	RWr	Batch/random	×	×
Own station buffer memory	—	Batch/random	×	×
Link direct device (link input)[*1]	Jn\X	Batch/random	○	○
Link direct device (link output)[*1]	Jn\Y	Batch/random	○	○
Link direct device (link relay)[*1]	Jn\B	Batch/random	○	○
Link direct device (link register)[*1]	Jn\W	Batch/random	○	○
Link direct device (link special relay)[*1]	Jn\SB	Batch/random	○	○
Link direct device (link special register)[*1]	Jn\SW	Batch/random	○	○
Intelligent function module device, module access device	Un\G	Batch/random	○	○
Other station buffer memory	—	Batch/random	×	×
Other station random access buffer	—	Batch/random	×	×
Remote input	RX	Batch/random	×	×
Remote output	RY	Batch/random	×	×
Remote register	RW	Batch/random	×	×
Link special relay	SB	Batch/random	×	×
Link special register	SW	Batch/random	×	×
CPU buffer memory	U3En\G	Batch	○	○
		Random	×	×
Fixed scan communication area	U3En\HG	Batch	○	○
		Random	×	×
Global label (No device assigned)[*2]	GV	Batch	×	×
		Random	○	×
Safety input	SA\X	Batch/random	×	×
Safety output	SA\Y	Batch/random	×	×
Safety internal relay	SA\M	Batch/random	×	×
Safety link relay	SA\B	Batch/random	×	×
Safety timer	SA\T	Batch/random	×	×
Safety retentive timer	SA\ST	Batch/random	×	×
Safety counter	SA\C	Batch/random	×	×

Device		Access method	Access target CPU	
			(1)	(2)
Safety data register	SA\D	Batch/random	×	×
Safety link register	SA\W	Batch/random	×	×
Safety special relay	SA\SM	Batch/random	×	×
Safety special register	SA\SD	Batch/random	×	×

*1 When accessing a link device directly, it is accessed as a link direct device (J□\□) depending on network module specifications. The methods for accessing a link direct device by bus interface communication are the same as those for accessing a link direct device by transient transmission using the network module access function.
For details on accessing a link direct device (J□\□), refer to the following:
MELSEC iQ-R MELSECWinCPU Module User's Manual

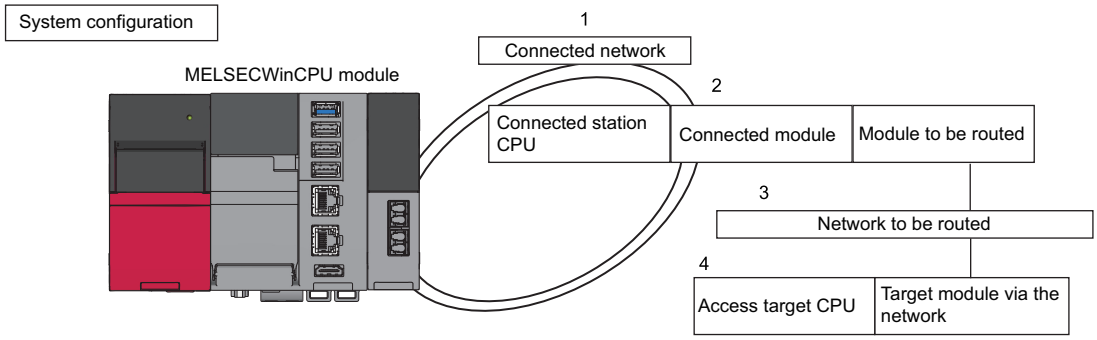
*2 Can be specified only with the mdRandRLabelEx function and the mdRandWLabelEx function.

CC-Link IE Controller Network communication

The following explains the range and devices accessible for communication via a CC-Link IE Controller Network module.

■Accessible range

The system configuration in the accessible range and the accessibility of each access target CPU via a CC-Link IE Controller Network module are shown below.



■Accessibility

Accessibility is shown in the following tables. The own station and the connected station CPU are accessible.

No.	Access target CPU	
(1)	MELSEC iQ-R series	Programmable controller
(2)	MELSEC-Q series	
(3)	MELSEC-L series	
(4)	MELSEC iQ-R series	C Controller module
(5)	MELSEC-Q series	
(6)	MELSEC iQ-R series	MELSECWinCPU module
(7)	MELSEC-Q series	

○: Accessible, ×: Not accessible

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU							
			Programmable controller			C Controller module		MELSECWinCPU module		Interface board for a personal computer
			(1)	(2)	(3)	(4)	(5)	(6)	(7)	
CC-Link IE Controller Network	MELSEC iQ-R series programmable controller	CC-Link IE Controller Network	○	○	×	○	○	○	×	×
		CC-Link IE Field Network	○	○	○	○	○*1	×	×	×
		CC-Link IE TSN	○	×	×	○	×	○	×	×
		MELSECNET/H network	○	○	×	○	○	×	×	×
		MELSECNET/10 network	○	○	×	○	○	×	×	×
		Ethernet	○	○	○	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×
CC-Link IE Controller Network	MELSEC iQ-R series C Controller module	CC-Link IE Controller Network	×	×	×	×	×	×	×	×
		CC-Link IE Field Network	×	×	×	×	×	×	×	×
		CC-Link IE TSN	×	×	×	×	×	×	×	×
		MELSECNET/H network	×	×	×	×	×	×	×	×
		MELSECNET/10 network	×	×	×	×	×	×	×	×
		Ethernet	×	×	×	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU							
			Programmable controller			C Controller module		MELSECWinCPU module		Interface board for a personal computer
			(1)	(2)	(3)	(4)	(5)	(6)	(7)	
CC-Link IE Controller Network	MELSEC iQ-R series MELSECWin CPU module	CC-Link IE Controller Network	×	×	×	×	×	×	×	×
		CC-Link IE Field Network	×	×	×	×	×	×	×	×
		CC-Link IE TSN	×	×	×	×	×	×	×	×
		MELSECNET/H network	×	×	×	×	×	×	×	×
		MELSECNET/10 network	×	×	×	×	×	×	×	×
		Ethernet	×	×	×	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×
CC-Link IE Controller Network	MELSEC-Q series programmable controller (Q mode)	CC-Link IE Controller Network*2	○	○*3	×	○	○	○	×	×
		CC-Link IE Field Network*2	○	○*3	○	○	○*1	×	×	×
		MELSECNET/H network	○	○*3	×	○	○	×	×	×
		MELSECNET/10 network	○	○*3	×	○	○	×	×	×
		MELSECNET(Ⅱ)	×	×	×	×	×	×	×	×
		Ethernet	○	○	○	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×
CC-Link IE Controller Network	MELSEC-Q series C Controller module	CC-Link IE Controller Network	×	×	×	×	×	×	×	×
		CC-Link IE Field Network	×	×	×	×	×	×	×	×
		MELSECNET/H network	×	×	×	×	×	×	×	×
		MELSECNET/10 network	×	×	×	×	×	×	×	×
		MELSECNET(Ⅱ)	×	×	×	×	×	×	×	×
		Ethernet	×	×	×	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×

*1 The following CPUs are accessible:
Q12DCCPU-V (Extended mode)
Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*2 The station number 65 or later is accessible only when all control CPUs on the network to be routed are universal model QCPUs.

*3 Connected station CPUs (Q00J, Q00, and Q01CPU) are not accessible.

■Accessible devices

The devices accessible for communication via a CC-Link IE Controller Network module are shown below.

Point

- The 'Batch' and 'Random' in the table shown below indicate as follows:
  Batch: Batch write (mdSendEx function), batch read (mdReceiveEx function), message send (mdSendEx function), message receive (mdReceiveEx function), link device batch write (mdBdWriteLinkDeviceEx function), and link device batch read (mdBdReadLinkDeviceEx function)
  Random: Random write (mdRandWEx function), random read (mdRandREx function), bit set (mdDevSetEx function), bit reset (mdDevRstEx function), random write by using a label name (mdRandWLabelEx function), and random read by using a label name (mdRandRLabelEx function)
- Only bit devices can be accessed by bit set (mdDevSetEx function) and bit reset (mdDevRstEx function).
- Device extension specifications (digit specification, bit specification, and index specification) cannot be used.

Accessing the own station

Accessible devices of a CC-Link IE Controller Network module controlled by a MELSECWinCPU module are shown in the following table.

An error will occur when specifying a device other than an accessible device.

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU
			R102WCPU-W
RECV function*1	—	Batch	○
		Random	×
Own station direct link input*2	LX	Batch	○
		Random	×
Own station direct link output*2	LY	Batch	○
		Random	×
Own station direct link relay*2	LB	Batch	○
		Random	×
Own station direct link register*2	LW	Batch	○
		Random	×
Own station direct link special relay*2	SB	Batch	○
		Random	×
Own station direct link special register*2	SW	Batch	○
		Random	×

*1 Can be accessed only with the mdReceiveEx function (message receive).

*2 Can be accessed with the mdBdWriteLinkDeviceEx function and the mdBdReadLinkDeviceEx function.

<When using a function whose name starts with 'mdBd'>

The following functions can access the same devices as when accessing the host CPU by bus interface communication.
(☞ Page 16 Bus interface communication)

- mdBdDevSetEx function
- mdBdDevRstEx function
- mdBdSendEx function
- mdBdReceiveEx function
- mdBdRandWEx function
- mdBdRandREx function

Accessing other stations

The accessible devices of the CC-Link IE Controller Network module on other stations are shown in the following tables.

No.	Access target CPU
(1)	Q00JCPU, Q00CPU, Q01CPU, Q02(H)CPU, Q02UCPU, Q06HCPU, Q02PHCPU, Q06PHCPU, Q12HCPU, Q25HCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU, Q00UCPU, Q01UCPU, Q00UJCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU, Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU, Q26UDVCPU Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, Q26UDPVCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, Q26DHCCPU-LS
(3)	Personal computer
(4)	L02CPU, L26CPU-BT, L02CPU-P, L26CPU-PBT, L02SCPU, L26CPU, L06CPU
(5)	R04CPU, R08CPU, R16CPU, R32CPU, R120CPU, R08PCPU, R16PCPU, R32PCPU, R120PCPU, R04ENCPU, R08ENCPU, R16ENCPU, R32ENCPU, R120ENCPU, R08SFCPU, R16SFCPU, R32SFCPU, R120SFCPU
(6)	R12CCPU-V
(7)	R102WCPU-W

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU						
			(1)	(2)	(3)	(4)	(5)	(6)	(7)
Input relay	X	Batch/random	○	○*1	×	○	○	○	○
Output relay	Y	Batch/random	○	○*1	×	○	○	○	○
Latch relay	L	Batch/random	○	×	×	○	○	×	×
Internal relay	M	Batch/random	○	○*1	×	○	○	○	○
Special relay	SM	Batch/random	○	○*1	×	○	○	○	○
Annunciator	F	Batch/random	○	×	×	○	○	×	×
Timer (contact)	T	Batch/random	○	×	×	○	○	×	×
Long timer (contact)	LT	Batch/random	×	×	×	×	○	×	×
Timer (coil)	T	Batch/random	○	×	×	○	○	×	×
Long timer (coil)	LT	Batch/random	×	×	×	×	○	×	×
Counter (contact)	C	Batch/random	○	×	×	○	○	×	×
Long counter (contact)	LC	Batch/random	×	×	×	×	○	×	×
Counter (coil)	C	Batch/random	○	×	×	○	○	×	×
Long counter (coil)	LC	Batch/random	×	×	×	×	○	×	×
Timer (current value)	T	Batch/random	○	×	×	○	○	×	×
Long timer (current value)	LT	Batch/random	×	×	×	×	○	×	×
Counter (current value)	C	Batch/random	○	×	×	○	○	×	×
Long counter (current value)	LC	Batch/random	×	×	×	×	○	×	×
Data register	D	Batch/random	○	○*1	×	○	○	○	○
Special register	SD	Batch/random	○	○*1	×	○	○	○	○
Index register	Z	Batch/random	○	×	×	○	○	×	×
Long index register	LZ	Batch/random	×	×	×	×	○	×	×
File register	R	Batch/random	○*2	×	×	○	○	×	×
	ZR	Batch/random	○*2	×	×	○	○	○	×
Refresh data register	RD	Batch/random	×	×	×	×	○	×	×
Link relay	B	Batch/random	○	○*3	×	○	○	○	○
Link register	W	Batch/random	○	○*3	×	○	○	○	○
Link special relay	SB	Batch/random	○	×	×	○	○	×	×
Retentive timer (contact)	ST	Batch/random	○	×	×	○	○	×	×
Long retentive timer (contact)	LST	Batch/random	×	×	×	×	○	×	×
Retentive timer (coil)	ST	Batch/random	○	×	×	○	○	×	×
Long retentive timer (coil)	LST	Batch/random	×	×	×	×	○	×	×
Link special register	SW	Batch/random	○	×	×	○	○	×	×
Edge relay	V	Batch/random	○	×	×	○	○	×	×
Own station random access buffer	—	Batch/random	×	×	×	×	×	×	×
Retentive timer (current value)	ST	Batch/random	○	×	×	○	○	×	×

Device		Access method	Access target CPU						
			(1)	(2)	(3)	(4)	(5)	(6)	(7)
Long retentive timer (current value)	LST	Batch/random	×	×	×	×	○	×	×
Own station link register (for sending)	—	Batch/random	×	×	×	×	×	×	×
Own station link register (for receiving)	—	Batch/random	×	×	×	×	×	×	×
Own station buffer memory	—	Batch/random	×	×	×	×	×	×	×
SEND function (with arrival confirmation)*4	—	Batch	○	○	×	○	○	○	○
		Random	×	×	×	×	×	×	×
SEND function (without arrival confirmation)*4	—	Batch	○	○	×	○	○	○	○
		Random	×	×	×	×	×	×	×
Link direct device (link input)*5	Jn\X	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link output)*5	Jn\Y	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link relay)*5	Jn\B	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link register)*5	Jn\W	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link special relay)*5	Jn\SB	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link special register)*5	Jn\SW	Batch/random	○	○*1	×	○	○	○	○
Intelligent function module device, module access device	Un\G	Batch/random	○	○*1	×	○	○	○	○
CPU shared memory, CPU buffer memory (CPU No.1 area)	U3E0\G	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
CPU shared memory, CPU buffer memory (CPU No.2 area)	U3E1\G	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
CPU shared memory, CPU buffer memory (CPU No.3 area)	U3E2\G	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
CPU shared memory, CPU buffer memory (CPU No.4 area)	U3E3\G	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.1 area)	U3E0\HG	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.2 area)	U3E1\HG	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.3 area)	U3E2\HG	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.4 area)	U3E3\HG	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Other station buffer memory	—	Batch/random	×	×	×	×	×	×	×
Other station random access buffer	—	Batch/random	×	×	×	×	×	×	×
Remote input for CC-Link	RX	Batch/random	×	×	×	×	×	×	×
Remote output for CC-Link	RY	Batch/random	×	×	×	×	×	×	×
Other station link register	—	Batch/random	×	×	×	×	×	×	×
Link special relay for CC-Link	SB	Batch/random	×	×	×	×	×	×	×
Link special register for CC-Link	SW	Batch/random	×	×	×	×	×	×	×
Global label (No device assigned)*6	GV	Batch	×	×	×	×	×	×	×
		Random	×	×	×	×	○	×	×
Safety input	SA\X	Batch/random	×	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×	×
Safety internal relay	SA\M	Batch/random	×	×	×	×	×	×	×
Safety link relay	SA\B	Batch/random	×	×	×	×	×	×	×
Safety timer	SA\T	Batch/random	×	×	×	×	×	×	×
Safety retentive timer	SA\ST	Batch/random	×	×	×	×	×	×	×
Safety counter	SA\C	Batch/random	×	×	×	×	×	×	×
Safety data register	SA\D	Batch/random	×	×	×	×	×	×	×
Safety link register	SA\W	Batch/random	×	×	×	×	×	×	×
Safety special relay	SA\SM	Batch/random	×	×	×	×	×	×	×
Safety special register	SA\SD	Batch/random	×	×	×	×	×	×	×

*1 The following CPUs are accessible:
Q12DCCPU-V with a serial number of which the first 5 digits are '12042' or later
Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*2 Q00JCPU is not accessible.

*3 The following CPUs are accessible:
Q12DCCPU-V (Extended mode)
Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*4 This is a function to send a message to a network module on other stations via a CC-Link IE Controller Network module. The function cannot access to a multiple CPU system (when a logical station number is specified).

*5 When accessing a link device directly, it is accessed as a link direct device (J□\□) depending on network module specifications.
For details on accessing a link direct device (J□\□), refer to the following:
MELSEC iQ-R MELSECWinCPU Module User's Manual

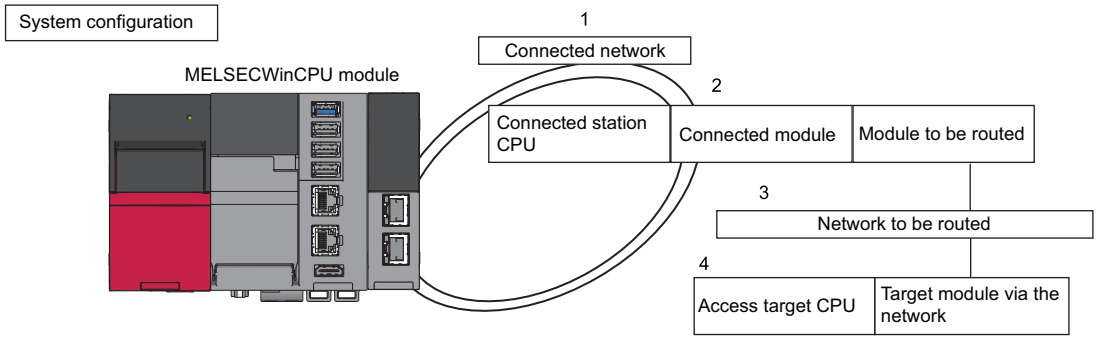
*6 Can be specified only with the mdRandRLabelEx function and the mdRandWLabelEx function.

CC-Link IE TSN communication

The following explains the range and devices accessible for communication via a CC-Link IE TSN module.

■Accessible range

The system configuration in the accessible range and the accessibility of each access target CPU via a CC-Link IE TSN module are shown below.



■Accessibility

Accessibility is shown in the following tables. The own station and the connected station CPU are accessible.

No.	Access target CPU	
(1)	MELSEC iQ-R series	Programmable controller
(2)	MELSEC-Q series	
(3)	MELSEC-L series	
(4)	MELSEC iQ-R series	C Controller module
(5)	MELSEC-Q series	
(6)	MELSEC iQ-R series	MELSECWinCPU module
(7)	MELSEC-Q series	

○: Accessible, ×: Not accessible

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU							
			Programmable controller			C Controller module		MELSECWinCPU module		Interface board for a personal computer
			(1)	(2)	(3)	(4)	(5)	(6)	(7)	
CC-Link IE TSN	MELSEC iQ-R series programmable controller	CC-Link IE Controller Network	○	○	×	○	○	○	×	×
		CC-Link IE Field Network	○	○	○	○	○	×	×	×
		CC-Link IE TSN	○	×	×	○	×	○	×	×
		MELSECNET/H network	○	○	×	○	○	×	×	×
		MELSECNET/10 network	○	○	×	○	○	×	×	×
		Ethernet	○	○	○	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×
CC-Link IE TSN	MELSEC iQ-R series C Controller module	CC-Link IE Controller Network	×	×	×	×	×	×	×	×
		CC-Link IE Field Network	×	×	×	×	×	×	×	×
		CC-Link IE TSN	×	×	×	×	×	×	×	×
		MELSECNET/H network	×	×	×	×	×	×	×	×
		MELSECNET/10 network	×	×	×	×	×	×	×	×
		Ethernet	×	×	×	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU							
			Programmable controller			C Controller module		MELSECWinCPU module		Interface board for a personal computer
			(1)	(2)	(3)	(4)	(5)	(6)	(7)	
CC-Link IE TSN	MELSEC iQ-R series MELSECWin CPU module	CC-Link IE Controller Network	×	×	×	×	×	×	×	×
		CC-Link IE Field Network	×	×	×	×	×	×	×	×
		CC-Link IE TSN	×	×	×	×	×	×	×	×
		MELSECNET/H network	×	×	×	×	×	×	×	×
		MELSECNET/10 network	×	×	×	×	×	×	×	×
		Ethernet	×	×	×	×	×	×	×	×
		Serial communication	×	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×	×

■Accessible devices

The devices accessible for communication via a CC-Link IE TSN module are shown below.

Point

- The 'Batch' and 'Random' in the table shown below indicate as follows:
  Batch: Batch write (mdSendEx function), batch read (mdReceiveEx function), message send (mdSendEx function), message receive (mdReceiveEx function), link device batch write (mdBdWriteLinkDeviceEx function), and link device batch read (mdBdReadLinkDeviceEx function)
  Random: Random write (mdRandWEx function), random read (mdRandREx function), bit set (mdDevSetEx function), bit reset (mdDevRstEx function), random write by using a label name (mdRandWLabelEx function), and random read by using a label name (mdRandRLabelEx function)
- Only bit devices can be accessed by bit set (mdDevSetEx function) and bit reset (mdDevRstEx function).
- Device extension specifications (digit specification, bit specification, and index specification) cannot be used.

Accessing the own station

The following table shows the accessible devices of a CC-Link IE TSN module controlled by a MELSECWinCPU module.
An error will occur when specifying a device other than an accessible device.
○: Accessible, ×: Not accessible

Device		Access method	Access target CPU
			R102WCPU-W
RECV function*1	—	Batch	○
		Random	×
Own station remote input*2	RX	Batch	○
		Random	×
Own station remote output*2	RY	Batch	○
		Random	×
Own station direct link relay*2	LB	Batch	○
		Random	×
Own station direct link register*2	LW	Batch	○
		Random	×
Own station remote register (for sending)*2	RWw	Batch	○
		Random	×
Own station remote register (for receiving)*2	RWr	Batch	○
		Random	×
Own station direct link special relay*2	SB	Batch	○
		Random	×
Own station direct link special register*2	SW	Batch	○
		Random	×

*1 Can be accessed only with the mdReceiveEx function (message receive).
*2 Can be accessed with the mdBdWriteLinkDeviceEx function and the mdBdReadLinkDeviceEx function.

<When using a function whose name starts with 'mdBd'>

The following functions can access the same devices as when accessing the host CPU by bus interface communication.
(☞ Page 16 Bus interface communication)

- mdBdDevSetEx function
- mdBdDevRstEx function
- mdBdSendEx function
- mdBdReceiveEx function
- mdBdRandWEx function
- mdBdRandREx function

Accessing other stations

When accessing the network of other stations from a CC-Link IE TSN module controlled by a MELSECWinCPU module, accessible CPUs and devices are the same as that for a CC-Link IE Controller Network module.
☞ Page 20 CC-Link IE Controller Network communication

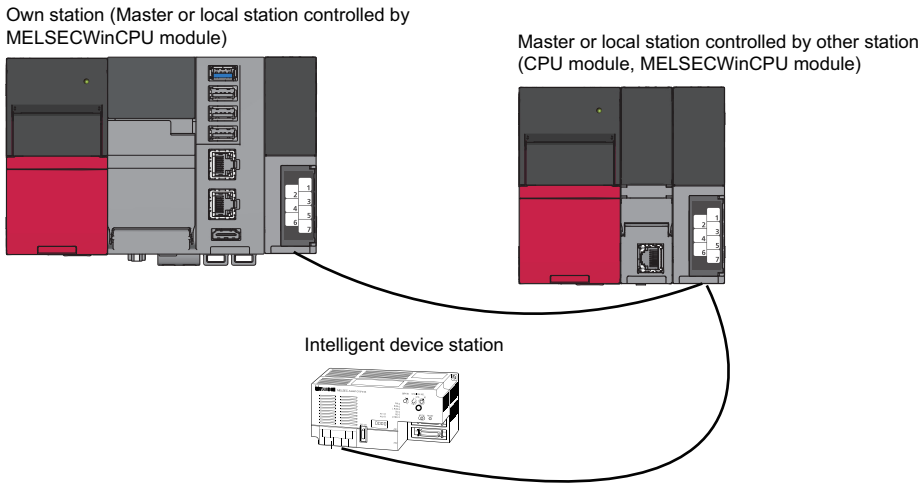
CC-Link communication

The following explains the range and devices accessible for CC-Link communication.

■Accessible range

The accessible range for CC-Link communication includes the own station (the master station or local stations controlled by a MELSECWinCPU module), the master station or local stations controlled by other stations (a CPU module, C Controller module, and MELSECWinCPU module), and an intelligent device station.

System configuration



Point

When the own station number is 64, other stations cannot be accessed.
Only the own station can be accessed.

■Accessible devices

The devices accessible for communication via a CC-Link module are shown below.

Point

- 'Batch' and 'Random' in the following table indicate as follows:
  Batch: Batch write (mdSendEx function), batch read (mdReceiveEx function)
  Random: Random write (mdRandWEx function), random read (mdRandREx function), bit set (mdDevSetEx function), bit reset (mdDevRstEx function), random write by using a label name (mdRandWLabelEx function), random read by using a label name (mdRandRLabelEx function)
- Only bit devices can be accessed by bit set (mdDevSetEx function) and bit reset (mdDevRstEx function).
- Device extension specifications (digit specification, bit specification, and index specification) cannot be used.

Accessing the own station

To access a CC-Link module controlled by a MELSECWinCPU module, use the method explained in the following section.
Accessing the own station by using CC-Link communication will cause the 'station number/network number error.'
☞ Page 108 Replacement of device types

<When using a function whose name starts with 'mdBd'>

The following functions can access the same devices as when accessing the host CPU by bus interface communication.
(☞ Page 16 Bus interface communication)

- mdBdDevSetEx function
- mdBdDevRstEx function
- mdBdSendEx function
- mdBdReceiveEx function
- mdBdRandWEx function
- mdBdRandREx function

Accessing other stations

The accessible devices of the CC-Link module on other stations are shown in the following tables.

No.	Access target CPU
(1)	Q00JCPU, Q00CPU, Q01CPU, Q02(H)CPU, Q02UCPU, Q06HCPU, Q02PHCPU, Q06PHCPU, Q12HCPU, Q25HCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU, Q00UCPU, Q01UCPU, Q00UJCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU, Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU, Q26UDVCPU Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, Q26UDPVCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, Q26DHCCPU-LS
(3)	Intelligent device station
(4)	L02CPU, L26CPU-BT, L02CPU-P, L26CPU-PBT, L02SCPU, L26CPU, L06CPU
(5)	R04CPU, R08CPU, R16CPU, R32CPU, R120CPU, R08PCPU, R16PCPU, R32PCPU, R120PCPU, R04ENCPU, R08ENCPU, R16ENCPU, R32ENCPU, R120ENCPU, R08SFCPU, R16SFCPU, R32SFCPU, R120SFCPU
(6)	R12CCPU-V
(7)	R102WCPU-W

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU						
			(1)	(2)	(3)	(4)	(5)	(6)	(7)
Input relay	X	Batch/random	○	○*1	×	○	○	○	○
Output relay	Y	Batch/random	○	○*1	×	○	○	○	○
Latch relay	L	Batch/random	○	×	×	○	○	×	×
Internal relay	M	Batch/random	○	○*1	×	○	○	○	○
Special relay	SM	Batch/random	○	○*1	×	○	○	○	○
Annunciator	F	Batch/random	○	×	×	○	○	×	×
Timer (contact)	T	Batch/random	○	×	×	○	○	×	×
Long timer (contact)	LT	Batch/random	×	×	×	×	○	×	×
Timer (coil)	T	Batch/random	○	×	×	○	○	×	×
Long timer (coil)	LT	Batch/random	×	×	×	×	○	×	×
Counter (contact)	C	Batch/random	○	×	×	○	○	×	×
Long counter (contact)	LC	Batch/random	×	×	×	×	○	×	×
Counter (coil)	C	Batch/random	○	×	×	○	○	×	×
Long counter (coil)	LC	Batch/random	×	×	×	×	○	×	×
Timer (current value)	T	Batch/random	○	×	×	○	○	×	×
Long timer (current value)	LT	Batch/random	×	×	×	×	○	×	×
Counter (current value)	C	Batch/random	○	×	×	○	○	×	×
Long counter (current value)	LC	Batch/random	×	×	×	×	○	×	×
Data register	D	Batch/random	○	○*1	×	○	○	○	○
Special register	SD	Batch/random	○	○*1	×	○	○	○	○
Index register	Z	Batch/random	○	×	×	○	○	×	×
Long index register	LZ	Batch/random	×	×	×	×	○	×	×
File register	R	Batch/random	○*2	×	×	○	○	×	×
	ZR	Batch/random	○*2	×	×	○	○	○	×
Refresh data register	RD	Batch/random	×	×	×	×	○	×	×
Link relay	B	Batch/random	○	○*3	×	○	○	○	○
Link register	W	Batch/random	○	○*3	×	○	○	○	○
Link special relay	SB	Batch/random	○	×	×	○	○	×	×
Retentive timer (contact)	ST	Batch/random	○	×	×	○	○	×	×
Long retentive timer (contact)	LST	Batch/random	×	×	×	×	○	×	×
Retentive timer (coil)	ST	Batch/random	○	×	×	○	○	×	×
Long retentive timer (coil)	LST	Batch/random	×	×	×	×	○	×	×
Link special register	SW	Batch/random	○	×	×	○	○	×	×
Edge relay	V	Batch/random	○	×	×	○	○	×	×
Own station random access buffer	—	Batch/random	×	×	×	×	×	×	×
Retentive timer (current value)	ST	Batch/random	○	×	×	○	○	×	×

Device		Access method	Access target CPU						
			(1)	(2)	(3)	(4)	(5)	(6)	(7)
Long retentive timer (current value)	LST	Batch/random	×	×	×	×	○	×	×
Remote register for sending	RWw	Batch/random	×	×	×	×	×	×	×
Remote register for receiving	RWr	Batch/random	×	×	×	×	×	×	×
Own station buffer memory	—	Batch/random	×	×	×	×	×	×	×
SEND function (with arrival confirmation)	—	Batch/random	×	×	×	×	×	×	×
SEND function (without arrival confirmation)	—	Batch/random	×	×	×	×	×	×	×
Link direct device (link input)*4	Jn\X	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link output)*4	Jn\Y	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link relay)*4	Jn\B	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link register)*4	Jn\W	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link special relay)*4	Jn\SB	Batch/random	○	○*1	×	○	○	○	○
Link direct device (link special register)*4	Jn\SW	Batch/random	○	○*1	×	○	○	○	○
Intelligent function module device, module access device	Un\G	Batch/random	○	○*1	×	○	○	○	○
CPU shared memory, CPU buffer memory (CPU No.1 area)	—	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
CPU shared memory, CPU buffer memory (CPU No.2 area)	—	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
CPU shared memory, CPU buffer memory (CPU No.3 area)	—	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
CPU shared memory, CPU buffer memory (CPU No.4 area)	—	Batch	○	○*1	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.1 area)	—	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.2 area)	—	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.3 area)	—	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Fixed scan communication area (CPU No.4 area)	—	Batch	×	×	×	×	○	○	○
		Random	×	×	×	×	×	×	×
Other station buffer memory	—	Batch/random	×	×	×	×	×	×	×
Other station random access buffer	—	Batch/random	×	×	×	×	×	×	×
Remote input	RX	Batch/random	×	×	×	×	×	×	×
Remote output	RY	Batch/random	×	×	×	×	×	×	×
Remote register	RW	Batch/random	×	×	×	×	×	×	×
Remote special relay	SB	Batch/random	×	×	×	×	×	×	×
Link special register	SW	Batch/random	×	×	×	×	×	×	×
Global label (No device assigned)*5	GV	Batch	×	×	×	×	×	×	×
		Random	×	×	×	×	○	×	×
Safety input	SA\X	Batch/random	×	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×	×
Safety internal relay	SA\M	Batch/random	×	×	×	×	×	×	×
Safety link relay	SA\B	Batch/random	×	×	×	×	×	×	×
Safety timer	SA\T	Batch/random	×	×	×	×	×	×	×
Safety retentive timer	SA\ST	Batch/random	×	×	×	×	×	×	×
Safety counter	SA\C	Batch/random	×	×	×	×	×	×	×
Safety data register	SA\D	Batch/random	×	×	×	×	×	×	×
Safety link register	SA\W	Batch/random	×	×	×	×	×	×	×
Safety special relay	SA\SM	Batch/random	×	×	×	×	×	×	×
Safety special register	SA\SD	Batch/random	×	×	×	×	×	×	×

*1 The following CPUs are accessible:
Q12DCCPU-V with a serial number of which the first 5 digits are '12042' or later
Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*2 Q00JCPU is not accessible.

*3 The following CPUs are accessible:
Q12DCCPU-V (Extended mode)
Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*4 When accessing a link device directly, it is accessed as a link direct device (J□\□) depending on network module specifications.
For details on accessing a link direct device (J□\□), refer to the following:
MELSEC iQ-R MELSECWinCPU Module User's Manual

*5 Can be specified only with the mdRandRLabelEx function and the mdRandWLabelEx function.

Argument specifications

This section shows the argument specifications of the MELSEC data link functions.

Channel number

A channel indicates a network and communication route to be used when communicating with a MELSECWinCPU module.
It needs to be set for each module in a user program.
Channels to be used for MELSEC data link functions are as follows:

Channel No.	Network	Communication route
12	Bus interface	Via a bus
151 to 158	CC-Link IE Controller Network	Via a CC-Link IE Controller Network module controlled by a MELSECWinCPU module
281 to 288	CC-Link IE TSN	Via a CC-Link IE TSN module controlled by a MELSECWinCPU module
81 to 88	CC-Link	Via a CC-Link module controlled by a MELSECWinCPU module

Network number and station number

■Network number and station number for MELSEC data link functions (excluding the mdControl function and the mdTypeRead function)

Network numbers and station numbers to be specified to MELSEC data link functions are as follows:

Communication	Specification method		Network No.	Station No.
Bus interface	Own station		0 (0H)	255 (FFH)
	Other station			1 (CPU No.1), 2 (CPU No.2), 3 (CPU No.3), and 4 (CPU No.4)
CC-Link IE Controller Network	Own station		0 (0H)	255 (FFH)
	Other station	Station number	1 (1H) to 239 (EFH)	1 (1H) to 120 (78H) 0 (0H)*1, 125 (7DH)*1
		Group number 1 to 32*2*3		129 (81H) to 160 (A0H)
		All stations*2		240 (F0H)
	Logical station number*4		0 (0H)	65 (41H) to 239 (EFH)
CC-Link IE TSN	Own station		0 (0H)	255 (FFH)
	Other station	Station number	1 (1H) to 239 (EFH)	1 (1H) to 120 (78H) 0 (0H)*6, 125 (7DH)*5*6
		Group number 1 to 32*2*3		129 (81H) to 160 (A0H)
		All stations*2		240 (F0H)
	Logical station number*4		0 (0H)	65 (41H) to 239 (EFH)
CC-Link	Other station		0 (0H)	0 (0H) to 63 (3FH)*7
	Logical station number*4			65 (41H) to 239 (EFH)

*1 When '0 (0H)' or '125 (7DH)' is specified for the station number, a specified control station of the network, which is specified for the network number, is accessed. To access the current control station (a station that is actually operating as the control station), specify the station number from 1 (1H) to 120 (78H).

*2 Can be specified when the SEND function (mdSendEx (message send function)) with 'no arrival confirmation' specification is used.

*3 Can be specified when using CC-Link IE Controller Network and CC-Link IE TSN.

*4 A logical number which is specified for "station number" in a user program (MELSEC data link functions)
It is used to access another station CPU (another CPU in a multiple CPU system) from a target module (channel number).
However, note that an error will occur and an access target CPU module cannot be accessed if it does not support a network module on the access route.
To access directly to a CPU module which controls other stations on MELSECNET/10 network, MELSECNET/H network, CC-Link IE Controller Network, CC-Link IE Field Network, and CC-Link IE TSN, the logical station number is not required to be set. Use the station numbers of other stations on MELSECNET/10 network, MELSECNET/H network, CC-Link IE Controller Network, CC-Link IE Field Network and CC-Link IE TSN.
Also, for direct access to CC-Link other stations (station number 0 to 63) or the CPU module which controls CC-Link other stations, setting of the logical station number is not required. Specify or use the station number of CC-Link.
A logical number can be set in the target settings of CW Configurator module parameters.

*5 When '0 (0H)' or '125 (7DH)' is specified for the station number, a master station of the network, which is specified for the network number, is accessed. To access a master operating station (a station that is operating as the master station when the submaster function is used), specify the station number from 1 (1H) to 120 (78H).

*6 When '0 (0H)' or '125 (7DH)' is specified for the station number, a master station of the network, which is specified for the network number, is accessed.

*7 The station number 64 cannot be specified for CC-Link communication.
In addition, when the station number of the own station is 64, other stations cannot be set. (Only the own station can be accessed.)

■Network number and station number for MELSEC data link functions (the mdControl function and the mdTypeRead function)

Network numbers and station numbers to be specified to the mdControl function or the mdTypeRead function are as follows:

Communication	Station number specification method
Bus interface	Own station: 255 (FFH) Other station: 1 (CPU No.1), 2 (CPU No.2), 3 (CPU No.3), 4 (CPU No.4)
CC-Link IE Controller Network	Own station: 255 (FFH) Other station:*1*2*6
CC-Link IE TSN	Own station: 255 (FFH) Other station:*1*3*6
CC-Link	Own station: 255 (FFH) Other station: 0 (0H) to 63 (3FH), 65 (41H) to 239 (EFH)*4*5*6

*1 Station number setting for a CC-Link IE Controller Network module and CC-Link IE TSN module

Upper	Lower

Upper/ lower	Setting item	Setting value	Description
Upper	Network number	1 (1H) to 239 (EFH)	Set this to specify other stations in the own network or each station on other networks. • Set this to issue a sending request to CC-Link IE Field Network, CC-Link IE Controller Network, CC-Link IE TSN, MELSECNET/H network, or MELSECNET/10 network.
Lower	Station number	1 (1H) to 120 (78H)	Set this to specify the station number of other stations. • For MELSECNET/H network, the setting range is from 1 to 64. • For CC-Link IE Field Network, CC-Link IE Controller Network, or CC-Link IE TSN, the setting range is from 0 to 120.
		0 (0H), 125 (7DH)	• For MELSECNET/H network or CC-Link IE Controller Network, the access target is a control station.*2 • For CC-Link IE Field Network or CC-Link IE TSN, the access target is a master station.*3

Logical station number specification method*6
Set '0' in the upper byte (network number) of the station number above, and specify a logical station number in the lower byte (station number). The setting range of the logical station number is 65 (41H) to 239 (EFH). The number can be set in the target settings of CW Configurator module parameters.

*2 When '0 (0H)' or '125 (7DH)' is specified for the station number, a specified control station of the network, which is specified for the network number, is accessed. To access the current control station (a station that is actually operating as the control station), specify the station number from 1 (1H) to 120 (78H).

*3 When '0 (0H)' or '125 (7DH)' is specified for the station number, the master station of the network, which is specified for the network number, is accessed.

*4 Station number setting for CC-Link module

Upper	Lower

Upper/ lower	Setting item	Setting value	Description
Upper	Network number	0	Set the value for CC-Link.
Lower	Station number	0 (0H) to 63 (3FH)	Set the station number of other stations.

Logical station number setting method
Set '0' in the upper byte (network number) of the station number above, and specify a logical station number in the lower byte (station number). The setting range of the logical station number is 65 (41H) to 239 (EFH). The number can be set in the target settings of CW Configurator module parameters.

*5 The station number 64 cannot be specified for CC-Link communication.
In addition, when the station number of the own station is 64, other stations cannot be set. (Only the own station can be accessed.)

*6 It is used to access another station CPU (another CPU in a multiple CPU system) from a target module (channel number).
However, note that an error will occur and an access target CPU module cannot be accessed if it does not support a network module on the access route.
To access directly to a CPU module which controls other stations on MELSECNET/10 network, MELSECNET/H network, CC-Link IE Controller Network, CC-Link IE Field Network, and CC-Link IE TSN, the logical station number is not required to be set. Use the station numbers of other stations on MELSECNET/10 network, MELSECNET/H network, CC-Link IE Controller Network, CC-Link IE Field Network and CC-Link IE TSN.
Also, for direct access to CC-Link other stations (station number 0 to 63) or the CPU module which controls CC-Link other stations, setting of the logical station number is not required. Specify or use the station number of CC-Link.

Device type

The following tables show the device types specified to the MELSEC data link functions.
Devices are defined in the header file "MDFuncWinCPU.h."

Point

Either a code or a device name can be specified as a device type.

■Common device types

Device name (device)		Device type		
		Code		Device name
		Decimal	Hexadecimal	
Input relay (X)		1	1H	DevX
Output relay (Y)		2	2H	DevY
Latch relay (L)		3	3H	DevL
Internal relay (M)		4	4H	DevM
Special relay (SM)		5	5H	DevSM
CPU buffer memory*1,*2	CPU No.1 area (U3E0\G)	501	1F5H	DevSPB1
	CPU No.2 area (U3E1\G)	502	1F6H	DevSPB2
	CPU No.3 area (U3E2\G)	503	1F7H	DevSPB3
	CPU No.4 area (U3E3\G)	504	1F8H	DevSPB4
Fixed scan communication area*1,*2	CPU No.1 area (U3E0\HG)	511	1FFH	DevHSPB1
	CPU No.2 area (U3E1\HG)	512	200H	DevHSPB2
	CPU No.3 area (U3E2\HG)	513	201H	DevHSPB3
	CPU No.4 area (U3E3\HG)	514	202H	DevHSPB4
Annunciator (F)		6	6H	DevF
Timer	Contact (T)	7	7H	DevTT
	Coil (T)	8	8H	DevTC
	Current value (T)	11	BH	DevTN
Long timer	Contact (LT)	41	29H	DevLTT
	Coil (LT)	42	2AH	DevLTC
	Current value (LT)	43	2BH	DevLTN
Counter	Contact (C)	9	9H	DevCT
	Coil (C)	10	AH	DevCC
	Current value (C)	12	CH	DevCN
Long counter (contact)	Contact (LC)	44	2CH	DevLCT
	Coil (LC)	45	2DH	DevLCC
	Current value (LC)	46	2EH	DevLCN
Retentive timer	Contact (ST)	26	1AH	DevSTT
	Coil (ST)	27	1BH	DevSTC
	Current value (ST)	35	23H	DevSTN
Long retentive timer	Contact (LST)	47	2FH	DevLSTT
	Coil (LST)	48	30H	DevLSTC
	Current value (LST)	49	31H	DevLSTN
Data register (D)		13	DH	DevD
Special register (SD)		14	EH	DevSD
Index register (Z)*3		20	14H	DevZ
Long index register (LZ)*3		38	26H	DevLZ
File register (R)*3		22	16H	DevR
File register (ZR)*3		220	DCH	DevZR
Link relay (B)		23	17H	DevB
Link register (W)		24	18H	DevW
Link special relay (SB)*3		25	19H	DevQSB
Link special register (SW)*3		28	1CH	DevQSW

Device name (device)		Device type		
		Code		Device name
		Decimal	Hexadecimal	
Edge relay (V)		30	1EH	DevQV
Refresh data register (RD)		39	27H	DevRD
Global label (GV)*4	For word, double word, and quad word size	600	258H	DevGV
	For bit 0	601	259H	DevGV_0
	For bit 1	602	25AH	DevGV_1
	For bit 2	603	25BH	DevGV_2
	For bit 3	604	25CH	DevGV_3
	For bit 4	605	25DH	DevGV_4
	For bit 5	606	25EH	DevGV_5
	For bit 6	607	25FH	DevGV_6
	For bit 7	608	260H	DevGV_7
	For bit 8	609	261H	DevGV_8
	For bit 9	610	262H	DevGV_9
	For bit A	611	263H	DevGV_A
	For bit B	612	264H	DevGV_B
	For bit C	613	265H	DevGV_C
	For bit D	614	266H	DevGV_D
	For bit E	615	267H	DevGV_E
	For bit F	616	268H	DevGV_F
Link direct device*3,*5 Argument value for a device name (1 to 255): Network number	Link input (Jn\X)	1001 to 1255	3E9H to 4E7H	DevLX(1) to DevLX(255)
	Link output (Jn\Y)	2001 to 2255	7D1H to 8CFH	DevLY(1) to DevLY(255)
	Link relay (Jn\B)	23001 to 23255	59D9H to 5AD7H	DevLB(1) to DevLB(255)
	Link register (Jn\W)	24001 to 24255	5DC1H to 5EBFH	DevLW(1) to DevLW(255)
	Link special relay (Jn\SB)	25001 to 25255	61A9H to 62A7H	DevLSB(1) to DevLSB(255)
	Link special register (Jn\SW)	28001 to 28255	6D61H to 6E5FH	DevLSW(1) to DevLSW(255)
Intelligent function module device*3, module access device*3 Argument value for a device name (0 to 255): Start I/O No. divided by 16		29000 to 29255	7148H to 7247H	DevSPG(0) to DevSPG(255)
SEND function (with arrival confirmation) and RECV function		101	65H	DevMAIL
SEND function (without arrival confirmation)		102	66H	DevMAILNC

*1 For Q12DCCPU-V, it is categorized as the device type for Q bus interface.
(It cannot be accessed with CC-Link communication and CC-Link IE Controller Network communication.)

*2 The devices cannot be used for the mdRandREx, mdRandWEx, mdDevSetEx, and mdDevRstEx functions.

*3 Even if a non-existent device is specified in the mdRandREx function, the function may end normally.
(All of the bits turn ON in read data. For word devices, the read data is '-1'.)

*4 Only the mdRandRLabelEx function and the mdRandWLabelEx function can be used.

*5 When accessing a link device directly, it is accessed as a link direct device (J□\□) depending on network module specifications.
For details on accessing a link direct device (J□\□), refer to the following:
MELSEC iQ-R MELSECWinCPU Module User's Manual

■Device types for accessing CC-Link IE Controller Network modules

The device types shown in the following tables can be specified in user programs:

• For direct access

The own station link devices of a CC-Link IE Controller Network module, which is controlled by a MELSECWinCPU module, can be specified with the mdBdWriteLinkDeviceEx function and the mdBdReadLinkDeviceEx function.

Device name (device)	Device type		
	Code		Device name
	Decimal	Hexadecimal	
Own station direct link input (LX)	1000	3E8H	DevLX(0)
Own station direct link output (LY)	2000	7D0H	DevLY(0)
Own station direct link relay (LB)	23000	59D8H	DevLB(0)
Own station direct link register (LW)	24000	5DC0H	DevLW(0)
Own station direct link special relay (SB)	25000	61A8H	DevLSB(0)
Own station direct link special register (SW)	28000	6D60H	DevLSW(0)

• For sending/receiving a message

Device	Device type		
	Code		Device name
	Decimal	Hexadecimal	
SEND function (with arrival confirmation) and RECV function	101	65H	DevMAIL
SEND function (without arrival confirmation)	102	66H	DevMAILNC

■Device types for accessing CC-Link IE TSN modules

The device types shown in the following tables can be specified in user programs:

• For direct access

The own station link devices of a CC-Link IE TSN module, which is controlled by a MELSECWinCPU module, can be specified with the mdBdWriteLinkDeviceEx function and the mdBdReadLinkDeviceEx function.

Device name (device)	Device type		
	Code		Device name
	Decimal	Hexadecimal	
Own station remote input (RX)	1000	3E8H	DevLX(0)
Own station remote output (RY)	2000	7D0H	DevLY(0)
Own station link relay (LB)	23000	59D8H	DevLB(0)
Own station link register (LW)*1	24000	5DC0H	DevLW(0)
Own station remote register (for sending) (RWw)*1 Own station remote register (for receiving) (RWr)*1	24000	5DC0H	DevLW(0)
Own station direct link special relay (SB)	25000	61A8H	DevLSB(0)
Own station direct link special register (SW)	28000	6D60H	DevLSW(0)

*1 Own station remote registers (RWw and RWr) and own station link register (LW) can be accessed within the following range.
RWw0 to 1FFF: LW0 to LW1FFF
RWr0 to 1FFF: LW2000 to LW3FFF
LW0 to 3FFF: LW4000 to LW7FFF

• For sending/receiving a message

Device	Device type		
	Code		Device name
	Decimal	Hexadecimal	
SEND function (with arrival confirmation) and RECV function	101	65H	DevMAIL
SEND function (without arrival confirmation)	102	66H	DevMAILNC

3 FUNCTION LIST

This chapter describes the functions that can be used for a MELSECWinCPU module.

3.1 C Controller Module Dedicated Functions

The C Controller module dedicated functions are as listed below.

Function name	Function	Reference
CCPU_ClearError	To clear errors of a MELSECWinCPU module	Page 42 CCPU_ClearError
CCPU_GetSerialNo	To acquire the serial number of a MELSECWinCPU module	Page 43 CCPU_GetSerialNo
CCPU_GetUnitInfo	To acquire the module configuration information	Page 44 CCPU_GetUnitInfo
CCPU_Reset	To reset the bus of a MELSECWinCPU module (CPU No.1)	Page 47 CCPU_Reset
CCPU_WaitEvent	To wait for an interrupt event notification from another CPU module	Page 48 CCPU_WaitEvent
CCPU_WaitUnitEvent	To wait for an interrupt event notification from modules	Page 50 CCPU_WaitUnitEvent

3.2 MELSEC Data Link Functions

The MELSEC data link functions are as listed below.

Function name	Function	Reference
mdBdDevRstEx	To reset bit devices (for the own station only)	Page 52 mdBdDevRstEx
mdBdDevSetEx	To set bit devices (for the own station only)	Page 53 mdBdDevSetEx
mdBdRandREx	To read data from devices randomly (for the own station only)	Page 54 mdBdRandREx
mdBdRandWEx	To write data to devices randomly (for the own station only)	Page 56 mdBdRandWEx
mdBdReadLinkDeviceEx	To read data from own station link devices of each network module	Page 58 mdBdReadLinkDeviceEx
mdBdReceiveEx	To read data from devices in a batch (for the own station only)	Page 59 mdBdReceiveEx
mdBdSendEx	To write data to devices in a batch (for the own station only)	Page 60 mdBdSendEx
mdBdWriteLinkDeviceEx	To write data to own station link devices of each network module	Page 61 mdBdWriteLinkDeviceEx
mdClose	To close a communication line (channel)	Page 62 mdClose
mdControl	To perform remote operations (remote RUN/STOP/PAUSE) for the CPU module	Page 63 mdControl
mdDevRstEx	To reset bit devices	Page 64 mdDevRstEx
mdDevSetEx	To set bit devices	Page 65 mdDevSetEx
mdGetLabelInfo	To acquire device information corresponding to a label name	Page 66 mdGetLabelInfo
mdOpen	To open a communication line (channel)	Page 70 mdOpen
mdRandREx	To read data from devices randomly	Page 71 mdRandREx
mdRandRLabelEx	To read data from devices corresponding to labels randomly	Page 74 mdRandRLabelEx
mdRandWEx	To write data to devices randomly	Page 77 mdRandWEx
mdRandWLabelEx	To write data to devices corresponding to labels randomly	Page 79 mdRandWLabelEx
mdReceiveEx	To read data from devices in a batch	Page 82 mdReceiveEx
mdReceiveEx (message receive)	To receive messages (RECV function)	Page 84 mdReceiveEx (message receive)
mdSendEx	To write data to devices in a batch	Page 87 mdSendEx
mdSendEx (message send)	To send messages (SEND function)	Page 89 mdSendEx (message send)
mdTypeRead	To read the model code of a CPU module	Page 91 mdTypeRead

4 DETAILS OF FUNCTION

This chapter explains the details of C Controller module dedicated functions and MELSEC data link functions.

4.1 C Controller Module Dedicated Functions

This section explains the details of C Controller module dedicated functions.

CCPU_ClearError

This function clears errors of a MELSECWinCPU module.

Format

■Visual C++

short CCPU_ClearError(long* errorinfo);

■Visual Basic

Short CCPU_ClearError(Integer errorinfo);

■Visual C#

short CCPU_ClearError(int errorinfo);

Argument

Argument	Name	Description	IN/OUT
errorinfo	Error information	Unused (Even if a value is specified, the operation is not affected.)	IN

Description

- This function clears errors occurred in a MELSECWinCPU module.
- When no error occurs, the function ends normally.
- When a stop error has occurred, the error cannot be cleared. (The function ends normally.)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

CCPU_GetSerialNo

This function acquires the serial number of a MELSECWinCPU module.

Format

■Visual C++

short CCPU_GetSerialNo(char* getdata, unsigned long datasize);

■Visual Basic

Short CCPU_GetSerialNo(Sbyte getdata(0), UInteger datasize);

■Visual C#

short CCPU_GetSerialNo(sbyte[] getdata, uint datasize);

Argument

Argument	Name	Description	IN/OUT
getdata	Serial number storage destination	Specify the serial number storage destination.	OUT
datasize	Serial number storage destination size	Specify the size of the serial number storage destination in byte units.*1 (When '0' is specified, this function ends normally without processing.)	IN

*1 The serial number of a MELSECWinCPU module has 16 digits. Therefore, reserve an area of 16 bytes or more in the serial number storage destination and specify 16 bytes for the serial number storage destination size (datasize).

Description

- This function acquires a serial number (16 digits) of a MELSECWinCPU module, and stores it in the serial number storage destination (getdata).
- The function acquires information for the size specified to the serial number storage destination size (datasize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

CCPU_GetUnitInfo

This function acquires the module configuration information.

Format

■Visual C++

short CCPU_GetUnitInfo(unsigned short* unitinfo1, unsigned short* unitinfo2, unsigned short* unitinfo3);

■Visual Basic

Short CCPU_GetUnitInfo(UShort unitinfo1(0), UShort unitinfo2(0), UShort unitinfo3(0));

■Visual C#

short CCPU_GetUnitInfo(ushort[] unitinfo1, ushort[] unitinfo2, ushort[] unitinfo3);

Argument

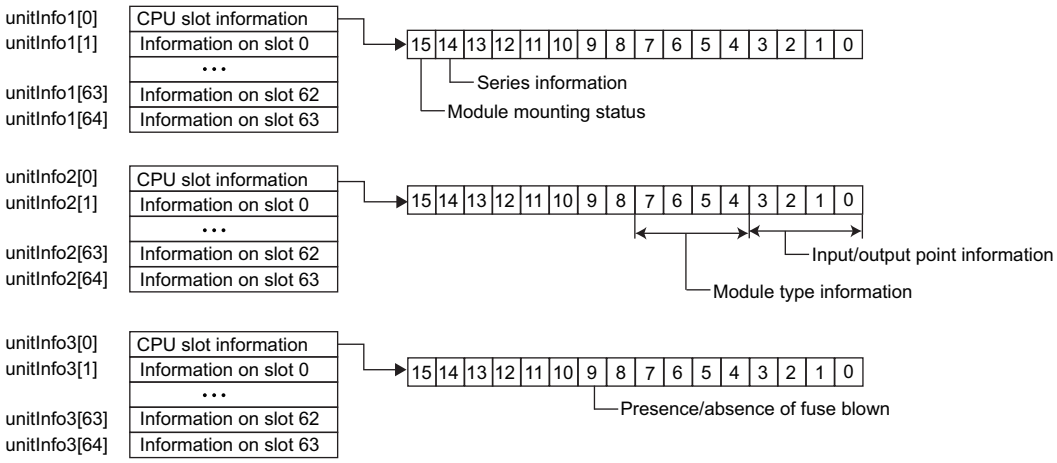
Argument	Name	Description	IN/OUT
unitinfo1	Module configuration information 1	Specify the storage destination of module configuration information 1.	OUT
unitinfo2	Module configuration information 2	Specify the storage destination of module configuration information 2.	OUT
unitinfo3	Module configuration information 3	Specify the storage destination of module configuration information 3.	OUT

Description

This function reads module configuration information (for 65 slots) and stores the information to the module configuration information 1 (unitinfo1), module configuration information 2 (unitinfo2), and module configuration information 3 (unitinfo3). Module configuration information to be stored differs depending on the series information.

Series information is MELSEC iQ-R series

For series information, check the 14th bit of module configuration information 1 (unitinfo1) [0-64].



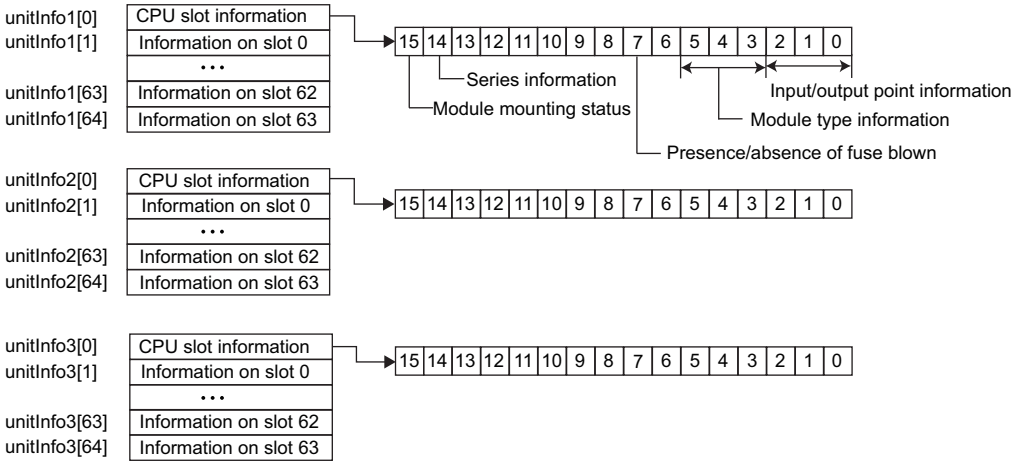
unitinfo		Description		
		Storage position		Status
unitinfo1[0-64]		bit15	Module mounting status	• 0: Not mounted • 1: Mounted
		bit14	Series information	1: MELSEC iQ-R series (0: MELSEC-Q series)
		bit13-0	Reserved	—
unitinfo2[0-64]	No module mounted*1	bit15-0	Reserved	—
	A module mounted*2	bit15-8	Reserved	—
		bit7-4	Module type information	• 0000: Input module • 0001: Power supply module • 0010: Output module • 0011: Base unit • 0100: Reserved • 0101: Reserved • 0110: I/O combined module • 0111: Empty • 1000: Intelligent module • 1001: CPU • 1010: Bus extension module • 1011: Reserved • 1100: Reserved • 1101: Reserved • 1110: Reserved • 1111: Module other than above
		bit3-0	Input/output point information	• 0000: 16 points • 0001: 32 points • 0010: 48 points • 0011: 64 points • 0100: 128 points • 0101: 256 points • 0110: 512 points • 0111: 1024 points • 1000: 2048 points • 1001: 4096 points • 1111: 0 points
unitinfo3[0-64]	No module mounted*1	bit15-0	Reserved	—
	A module mounted*2	bit15-10	Reserved	—
		bit9	Presence/absence of fuse blown	• 0: Normal • 1: Fuse blown
		bit8-0	Reserved	—

*1 Indicates when '0' is stored in the 'bit 15' of unitinfo1[N]. (N indicates the same array elements.)

*2 Indicates when '1' is stored in the 'bit 15' of unitinfo1[N]. (N indicates the same array elements.)

Series information is MELSEC-Q series

For series information, check the 14th bit of module configuration information 1 (unitinfo1) [0-64].



unitinfo		Description		
		Storage position		Status
unitinfo1[0-64]	—	bit15	Module mounting status	• 0: Not mounted • 1: Mounted
	No module mounted*1	bit14-0	Reserved	—
	A module mounted*2	bit14	Series information	0: MELSEC-Q series (1: MELSEC iQ-R series)
		bit7	Presence/absence of fuse blown	• 0: Normal • 1: Fuse blown
		bit6	Reserved	—
		bit5-3	Module type information	• 000: Input module • 001: Output module • 010: I/O combined module • 011: Intelligent function module • 111: Module other than above
		bit2-0	Input/output point information	• 000: 16 points • 001: 32 points • 010: 48 points • 011: 64 points • 100: 128 points • 101: 256 points • 110: 512 points • 111: 1024 points
unitinfo2[0-64]		bit15-0	Reserved	—
unitinfo3[0-64]		bit15-0	Reserved	—

*1 Indicates when '0' is stored in the 'bit 15' of unitinfo1[N]. (N indicates the same array elements.)
*2 Indicates when '1' is stored in the 'bit 15' of unitinfo1[N]. (N indicates the same array elements.)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

CCPU_Reset

This function resets the bus of a MELSECWinCPU module (CPU No.1).

Format

■Visual C++

short CCPU_Reset(void);

■Visual Basic

Short CCPU_Reset(void);

■Visual C#

short CCPU_Reset(void);

Argument

None

Description

- This function resets the bus of a MELSECWinCPU module (CPU No.1).
  It can be used to reset and restart a system due to an error.
- When setting a MELSECWinCPU module as the bus master CPU, reset the bus.
  For resetting the bus, refer to the following:
  MELSEC iQ-R MELSECWinCPU Module User's Manual

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Precautions

The CCPU_Reset function can be executed only when all the following conditions are satisfied.

■When the host CPU is set as the bus master CPU (CPU No.1)

- "Enable" is set for the "Remote Bus Reset Setting" parameter in the bus master CPU (CPU No.1).
- The Y output status of the bus master CPU (CPU No.1) is in the Y STOP state.

■When the host CPU is set as a CPU other than the bus master CPU (CPU No.1)

The CCPU_Reset function cannot be executed.

CCPU_WaitEvent

This function waits for an interrupt event notification from another CPU module.

Format

■Visual C++

short CCPU_WaitEvent(short* eventsetting, unsigned long timeout, short* seteventno);

■Visual Basic

Short CCPU_WaitEvent(Short eventsetting(0), UInteger timeout, Short seteventno(0));

■Visual C#

short CCPU_WaitEvent(short[] eventsetting, uint timeout, short[] seteventno);

Argument

Argument	Name	Description	IN/OUT
eventsetting	Interrupt event setting	Specify the interrupt event.	IN
timeout	Timeout value	Specify the timeout value in milliseconds (0H to FFFFFFFFH). (When FFFFFFFFH is specified, the function waits for an event infinitely.)	IN
seteventno	Occurred event	An occurred event is stored. (The CPU number and event number (interrupt pointer number) of the notified interrupt event are stored.)	OUT

■Specification method for the interrupt event setting (eventsetting)

eventsetting	Description	
eventsetting[0]	Number of interrupt event settings (1 to 64)	
eventsetting[1]	CPU number of the first interrupt event (1 to 4)	First event setting
eventsetting[2]	Event number (interrupt pointer number) of the first interrupt event (0 to 15)	
eventsetting[3]	CPU number of the second interrupt event (1 to 4)	Second event setting
eventsetting[4]	Event number (interrupt pointer number) of the second interrupt event (0 to 15)	
eventsetting[5]	CPU number of the third interrupt event (1 to 4)	Third event setting
eventsetting[6]	Event number (interrupt pointer number) of the third interrupt event (0 to 15)	
⋮	⋮	⋮

The following values are stored in the occurred event (seteventno).

seteventno	Description
seteventno[0]	CPU number of the notified interrupt event
seteventno[1]	Event number (interrupt pointer number) of the notified interrupt event

Description

- This function waits for an interrupt event specified to the interrupt event setting (eventsetting) for the time specified to the timeout value (timeout).
- When multiple interrupt events occur, the interrupt events are notified in ascending order of the event number.
- If an interrupt event has already been notified at the time when the CCPU_WaitEvent function is called, the function immediately ends normally. When the bus is reset, any interrupt events that occurred prior to the reset are discarded.
- If multiple interrupt events have been notified for the same event number (interrupt pointer number) at the time when the CCPU_WaitEvent function is called, the function processes the events as a single interrupt event notification.
- Set the event number (interrupt pointer number) without duplication. Otherwise, an error will be returned.
- Specify a programmable controller CPU or C Controller module to the CPU number. Otherwise, an error will be returned.
- Design a program so that the CCPU_WaitEvent function is not called from multiple user programs simultaneously by specifying the same event number (interrupt pointer number) in the user programs. Otherwise, a user program to which the interrupt event is notified is unpredictable.

Ex.

Settings of 'eventsetting' to wait for interrupt event 0 and 1 for CPU No.1, and interrupt event 10 for CPU No.2

eventsetting[0] = 3;

eventsetting[1] = 1;

eventsetting[2] = 0;

eventsetting[3] = 1;

eventsetting[4] = 1;

eventsetting[5] = 2;

eventsetting[6] = 10;

When interrupt event 10 for CPU No.2 occurs, '2' and '10' are returned to seteventno[0] and seteventno[1], respectively.

Precautions

- Do not set the clock data of a MELSECWinCPU module while executing the CCPU_WaitEvent function. Otherwise, the CCPU_WaitEvent function does not operate properly. (The process of the function may not be completed.)
- Windows does not have functions which ensure that processing completes within the certain time. Therefore, timeout detection may not be performed at the specified timeout value depending on Windows system processing and processing on other applications.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 50 CCPU_WaitUnitEvent

CCPU_WaitUnitEvent

This function waits for an interrupt event notification from modules.

Format

■Visual C++

short CCPU_WaitUnitEvent(short* eventsetting, unsigned long timeout, short* seteventno);

■Visual Basic

Short CCPU_WaitUnitEvent(Short eventsetting(0), UInteger timeout, Short seteventno);

■Visual C#

short CCPU_WaitUnitEvent(short[] eventsetting, uint timeout, short seteventno);

Argument

Argument	Name	Description	IN/OUT
eventsetting	Event setting	Specify the interrupt event.	IN
timeout	Timeout value	Specify the timeout value in milliseconds (0H to FFFFFFFFH). (When FFFFFFFFH is specified, the function waits for an event infinitely.)	IN
seteventno	Occurred event	An occurred event is stored. The event number (interrupt pointer number) of the notified interrupt event is stored.	OUT

■Specification method for the event setting (eventsetting)

eventsetting	Description
eventsetting[0]	Number of interrupt event settings (1 to 64)
eventsetting[1]	Interrupt pointer number of the first interrupt event (0 to 15, 50 to 1023)
eventsetting[2]	Interrupt pointer number of the second interrupt event (0 to 15, 50 to 1023)
eventsetting[3]	Interrupt pointer number of the third interrupt event (0 to 15, 50 to 1023)
⋮	⋮

Description

- This function waits for an interrupt event specified to the event setting (eventsetting) for the time specified to the timeout value (timeout).
- When multiple interrupt events occur, the interrupt events are notified in ascending order of the event number.
- If an interrupt event has already been notified at the time when the CCPU_WaitUnitEvent function is called, the function immediately ends normally. When the bus is reset, any interrupt events that occurred prior to the reset are discarded.
- If multiple interrupt events have been notified for the same event number (interrupt pointer number) at the time when the CCPU_WaitUnitEvent function is called, the function processes the events as a single interrupt event notification.
- Set the event number (interrupt pointer number) without duplication. Otherwise, an error will be returned.
- Design a program so that the CCPU_WaitUnitEvent function is not called from multiple user programs simultaneously by specifying the same event number (interrupt pointer number) in the user programs. Otherwise, a user program to which the interrupt event is notified is unpredictable.
- An interrupt event may not be notified when a stop error occurs in a MELSECWinCPU module.
- Create the following user program to discontinue the processing of a user program when a stop error occurs.
  A program that reads the SD203 value after executing the CCPU_WaitUnitEvent function and ends processing if the Y output status is in the Y STOP state.
- When an interrupt event is notified (return value of this function is normal), the event number of the notified interrupt event is returned to the occurred event (seteventno).

Ex.

Settings of 'eventsetting' when waiting for interrupt event 0, interrupt event 1, interrupt event 50, and interrupt event 51

eventsetting[0] = 4;

eventsetting[1] = 0;

eventsetting[2] = 1;

eventsetting[3] = 50;

eventsetting[4] = 51;

When event 51 occurs, 51 is returned to 'seteventno.'

The event numbers (interrupt pointer numbers) are as follows.

Event number (Interrupt pointer number)	Interrupt factor	Notes
0 to 15	Interrupt by module	—
16 to 49	Reserved	—
50 to 1023	Interrupt by module	Set with CW Configurator

Precautions

- Do not set the clock data of a MELSECWinCPU module while executing the CCPU_WaitUnitEvent function. Otherwise, the function does not operate properly. (The process of the function may not be completed.)
- Windows does not have functions which ensure that processing completes within the certain time. Therefore, timeout detection may not be performed at the specified timeout value depending on Windows system processing and processing on other applications.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 48 CCPU_WaitEvent

4.2 MELSEC Data Link Functions

This section explains the details of MELSEC data link functions.

mdBdDevRstEx

This function resets (turns OFF) bit devices of the own station.

Format

■Visual C++

long mdBdDevRstEx(long path, long devtyp, long devno);

■Visual Basic

Integer mdBdDevRstEx(Integer path, Integer devtyp, Integer devno);

■Visual C#

int mdBdDevRstEx(int path, int devtyp, int devno);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
devtyp	Device type	Specify the device type of a bit device. ☞ Page 37 Device type	IN
devno	Specified device number	Specify the device number of a bit device.	IN

Description

- This function resets (turns OFF) bit devices of the own station specified to the device type (devtyp) and device number (devno).
- The function is dedicated for bit devices such as link relay (B) and internal relay (M).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 53 mdBdDevSetEx

mdBdDevSetEx

This function sets (turns ON) bit devices of the own station.

Format

■Visual C++

long mdBdDevSetEx(long path, long devtyp, long devno);

■Visual Basic

Integer mdBdDevSetEx(Integer path, Integer devtyp, Integer devno);

■Visual C#

int mdBdDevSetEx(int path, int devtyp, int devno);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
devtyp	Device type	Specify the device type of a bit device. ☞ Page 37 Device type	IN
devno	Specified device number	Specify the device number of a bit device.	IN

Description

- This function sets (turns ON) bit devices of the own station specified to the device type (devtyp) and device number (devno).
- The function is dedicated for bit devices such as link relay (B) and internal relay (M).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 52 mdBdDevRstEx

mdBdRandREx

This function reads data from devices on the own station randomly.

Format

■Visual C++

long mdBdRandREx(long path, long* dev, short* buf, long bufsize);

■Visual Basic

Integer mdBdRandREx(Integer path, Integer dev(0), Short buf(0), Integer bufsize);

■Visual C#

int mdBdRandREx(int path, int[] dev, short[] buf, int bufsize);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
dev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be read.	IN
buf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
bufsize	Read data storage destination size	Specify the area size reserved in the read data storage destination in byte units.	IN

■Specification method for the randomly selected device (dev)

dev	Description	
dev[0]	Number of blocks	
dev[1]	Device type of block 1	Block 1
dev[2]	Start device number of block 1	
dev[3]	Number of read points of block 1	
dev[4]	Device type of block 2	Block 2
dev[5]	Start device number of block 2	
dev[6]	Number of read points of block 2	
⋮	⋮	⋮
The number of blocks can be specified within the range of 1 to 32767.		

Description

This function reads the device specified with the randomly selected device (dev).

The read data is stored in the read data storage destination (buf) in word units in the order specified to the randomly selected device (dev).

A bit device is stored per 16 points and a word device is stored per 1 point.

Precautions

Data inconsistency occurs in the following cases:

- The number of specified blocks is different.
- A number other than a multiple of 16 is specified to the start device number or the number of points of a bit device.

Example

The following shows an example for executing this function.

Ex.

Reading values from M100 to M115, D10 to D13, and M0 to M13

Device where data is randomly read	Current value
M100 to M115	All bits are OFF.
D10 to D13	10 is stored in D10, 200 in D11, 300 in D12, and 400 in D13.
M0 to M13	All bits are ON.

The following tables show the examples of values specified to the randomly selected device (dev) and read data storage destination size (bufsize) as well as values read to the read data storage destination (buf).

■Values specified to the randomly selected device (dev)

dev	Specified value	Description	
dev[0]	3	Number of blocks = 3	—
dev[1]	DevM	Device type = M	Block 1: M100 to M115
dev[2]	100	Start device number = 100	
dev[3]	16	Number of read points = 16	
dev[4]	DevD	Device type = D	Block 2: D10 to D13
dev[5]	10	Start device number = 10	
dev[6]	4	Number of read points = 4	
dev[7]	DevM	Device type = M	Block 3: M0 to M3
dev[8]	0	Start device number = 0	
dev[9]	14	Number of read points = 14	

■Values read to the read data storage destination (buf)

buf	Read device	Read value	Description
buf[0]	M100 to M115	0	All the bit devices from M100 to M115 are OFF. (Bit information for 16 points can be stored.)
buf[1]	D10	10	D10 = 10
buf[2]	D11	200	D11 = 200
buf[3]	D12	300	D12 = 300
buf[4]	D13	400	D13 = 400
buf[5]	M0 to M13	3FFFH	All the bit devices from M0 to M13 are ON.

■Value specified to the read data storage destination size (bufsize)

(buf[0] to buf[5] = 6) × 2 = 12

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 56 mdBdRandWEx

mdBdRandWEx

This function writes data to devices on the own station randomly.

Format

■Visual C++

long mdBdRandWEx(long path, long* dev, short* buf, long bufsize);

■Visual Basic

Integer mdBdRandWEx(Integer path, Integer dev(0), Short buf(0), Integer bufsize);

■Visual C#

int mdBdRandWEx(int path, int[] dev, short[] buf, int bufsize);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
dev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be written.	IN
buf	Write data storage destination	Specify the storage destination (address) of write data. (Reserve a continuous area for the write data storage destination.)	IN
bufsize	Write data storage destination size	Unused (Even if a value is specified, the operation is not affected.)	IN

■Specification method for the randomly selected device (dev)

dev	Description	
dev[0]	Number of blocks	
dev[1]	Device type of block 1	Block 1
dev[2]	Start device number of block 1	
dev[3]	Number of write points of block 1	
dev[4]	Device type of block 2	Block 2
dev[5]	Start device number of block 2	
dev[6]	Number of write points of block 2	
⋮	⋮	⋮
The number of blocks can be specified within the range of 1 to 32767.		

Description

This function writes data to a device specified to the randomly selected device (dev).

The data to be written is stored in the write data storage destination (buf) in word units.

A bit device is stored per 16 points and a word device is stored per 1 point.

Precautions

Data inconsistency occurs in the following cases:

- The number of specified blocks is different.
- A number other than a multiple of 16 is specified to the start device number or the number of points of a bit device.

Example

The following shows an example for executing this function.

Ex.

Turning OFF all the bits from M100 to M115 and writing 10 to D10, 200 to D11, and 400 to D13

Device where data is randomly written	Description
M100 to M115	Turn all the bits OFF.
D10 to D13	Store 10 in D10, 200 in D11, 300 in D12, and 400 in D13.

The following tables show the examples of values specified to the randomly selected device (dev) and write data storage destination (buf).

■Values specified to the randomly selected device (dev)

dev	Specified value	Description	
dev[0]	3	Number of blocks = 3	—
dev[1]	DevM	Device type of block 1 = M	Block 1: M100 to M115
dev[2]	100	Start device number of block 1 = 100	
dev[3]	16	Number of write points of block 1 = 16	
dev[4]	DevD	Device type of block 2 = D	Block 2: D10 to D13
dev[5]	10	Start device number of block 2 = 10	
dev[6]	4	Number of write points of block 2 = 4	

■Values specified to the write data (buf)

buf	Specified value	Description
buf[0]	0	Turn all bit devices from M100 to M115 OFF.
buf[1]	10	D10 = 10
buf[2]	200	D11 = 200
buf[3]	300	D12 = 300
buf[4]	400	D13 = 400

■Number of bytes of write data storage destination (buf)

(buf[0] to buf[4] = 5) × 2 = 10

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 54 mdBdRandREx

mdBdReadLinkDeviceEx

This function reads data from the own station link devices of a CC-Link IE Controller Network module and CC-Link IE TSN module.

Format

■Visual C++

long mdBdReadLinkDeviceEx (long path, long devtyp, long devno, long size, short* databuf, long bufsize);

■Visual Basic

Integer mdBdReadLinkDeviceEx(Integer path, Integer devtyp, Integer devno, Integer size, Short databuf(0), Integer bufsize);

■Visual C#

int mdBdReadLinkDeviceEx(int path, int devtyp, int devno, int size, short[] databuf, int bufsize);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
devtyp	Device type	Specify the device type of a device. ☞ Page 37 Device type	IN
devno	Start device number	Specify the start device number. (Only multiples of 16 can be specified to bit devices.)	IN
size	Data size	Specify the read data size in word units.	IN
databuf	Data storage destination	Specify the storage destination of read data.	OUT
bufsize	Data storage destination size	Specify the data storage destination size in word units.	IN

Description

- This function reads data of subsequent devices starting from a link device specified to the device type (devtyp) and the start device number (devno) of a network module which is specified to the channel path (path). The data is read for the size specified to the data size (size) and stored in the data storage destination (databuf).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 61 mdBdWriteLinkDeviceEx

mdBdReceiveEx

This function reads data from devices on the own station in a batch.

Format

■Visual C++

long mdBdReceiveEx(long path, long devtyp, long devno, long* size, short* data);

■Visual Basic

Integer mdBdReceiveEx(Integer path, Integer devtyp, Integer devno, Integer size, Short data(0));

■Visual C#

int mdBdReceiveEx(int path, int devtyp, int devno, int size, short[] data);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
devtyp	Device type	Specify the device type of a device. ☞ Page 37 Device type	IN
devno	Start device number	Specify the start device number of devices to be read in a batch. (For bit devices, set the number in multiples of 16.)	IN
size	Read data size	Specify a read data size in byte units. (Specify a value in multiples of 2. If the value other than that is specified, the size specification error (-255) will occur.)	IN/OUT
data	Read data storage destination	Specify the storage destination (address) of read data.	OUT

Description

- This function reads data in the order from a device specified to the device type (devtyp) and the start device number (devno). The data is read for the size specified to the read data size (size).
- When the read data size (size) exceeds the device range, a readable size is returned to the read data size (size).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 60 mdBdSendEx

mdBdSendEx

This function writes data to devices on the own station in a batch.

Format

■Visual C++

long mdBdSendEx(long path, long devtyp, long devno, long* size, short* data);

■Visual Basic

Integer mdBdSendEx(Integer path, Integer devtyp, Integer devno, Integer size, Short data(0));

■Visual C#

int mdBdSendEx(int path, int devtyp, int devno, int size, short[] data);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
devtyp	Device type	Specify the device type for devices to be written in a batch. ☞ Page 37 Device type	IN
devno	Start device number	Specify the start device number of devices to which the data is written in a batch. (For bit devices, set the number in multiples of 16.)	IN
size	Write data size	Specify a write data size in byte units. (Specify a value in multiples of 2. If the value other than that is specified, the size specification error (-255) will occur.)	IN/OUT
data	Write data storage destination	Specify the storage destination (address) of write data. (Reserve a continuous area for the write data storage destination.)	IN

Description

- This function writes data in the order from a device specified to the device type (devtyp) and the start device number (devno). The data is written for the size specified to the write data size (size).
- The function checks arguments and verifies whether the sum of address and size determined by the arguments is within the device range. When the write data size (size) exceeds the device range, a writable size is returned to the write data size (size).
- Accessible devices are the devices which can access the own station.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 59 mdBdReceiveEx

mdBdWriteLinkDeviceEx

This function writes data to the own station link devices of a CC-Link IE Controller Network module and CC-Link IE TSN module.

Format

■Visual C++

long mdBdWriteLinkDeviceEx (long path, long devtyp, long devno, long size, short* databuf, long bufsize);

■Visual Basic

Integer mdBdWriteLinkDeviceEx(Integer path, Integer devtyp, Integer devno, Integer size, Short databuf(0), Integer bufsize);

■Visual C#

int mdBdWriteLinkDeviceEx(int path, int devtyp, int devno, int size, short[] databuf, int bufsize);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
devtyp	Device type	Specify the device type of a device. ☞ Page 37 Device type	IN
devno	Start device number	Specify the start device number of a device to which data is written. (Only multiples of 16 can be specified to bit devices.)	IN
size	Data size	Specify the write data size in word units.	IN
databuf	Data storage destination	Specify the storage destination of write data.	IN
bufsize	Data storage destination size	Unused (Even if a value is specified, the operation is not affected.)	IN

Description

- This function writes data of the data storage destination (databuf) to the subsequent devices starting from a link device specified to the device type (devtyp) and the start device number (devno) of a network module which is specified to the channel path (path). The data is written for the size specified to the data size (size).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 58 mdBdReadLinkDeviceEx

mdClose

This function closes a communication line (channel).

Format

■Visual C++

short mdClose(long path);

■Visual Basic

Short mdClose(Integer path);

■Visual C#

short mdClose(int path);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN

Description

- This function closes the channel opened by the mdOpen function.
- When using multiple channels, close the channel number one by one.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen

mdControl

This function performs remote operations (remote RUN/STOP/PAUSE) for a CPU module.

Format

■Visual C++

short mdControl(long path, short stno, short code);

■Visual Basic

Short mdControl(Integer path, Short stno, Short code);

■Visual C#

short mdControl(int path, short stno, short code);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
stno	Station number	Specify the network number and station number of a target module. ☞ Page 34 Network number and station number	IN
code	Instruction code	Specify the contents of the remote operation in numerical value.	IN

■Specification method for the instruction code (code)

Instruction code (code) (decimal)	Description
0	Remote RUN
1	Remote STOP
2	Remote PAUSE

Description

- This function changes the status of a CPU module specified to the station number (stno) to the one specified to the instruction code (code).
- The function cannot be executed for C Controller modules.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose

mdDevRstEx

This function resets (turns OFF) bit devices.

Format

■Visual C++

long mdDevRstEx(long path, long netno, long stno, long devtyp, long devno);

■Visual Basic

Integer mdDevRstEx(Integer path, Integer netno, Integer stno, Integer devtyp, Integer devno);

■Visual C#

int mdDevRstEx(int path, int netno, int stno, int devtyp, int devno);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
devtyp	Device type	Specify the device type of a bit device. ☞ Page 37 Device type	IN
devno	Specified device number	Specify the device number of a bit device.	IN

Description

- This function resets (turns OFF) bit devices of a module specified to the network number (netno), station number (stno), device type (devtyp), and specified device number (devno).
- The function is dedicated for bit devices such as link relay (B) and internal relay (M).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 65 mdDevSetEx

mdDevSetEx

This function sets (turns ON) bit devices.

Format

■Visual C++

long mdDevSetEx(long path, long netno, long stno, long devtyp, long devno);

■Visual Basic

Integer mdDevSetEx(Integer path, Integer netno, Integer stno, Integer devtyp, Integer devno);

■Visual C#

int mdDevSetEx(int path, int netno, int stno, int devtyp, int devno);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
devtyp	Device type	Specify the device type of a bit device. ☞ Page 37 Device type	IN
devno	Specified device number	Specify the device number of a bit device.	IN

Description

- This function sets (turns ON) bit devices of a module specified to the network number (netno), station number (stno), device type (devtyp), and specified device number (devno).
- The function is dedicated for bit devices such as link relay (B) and internal relay (M).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 64 mdDevRstEx

mdGetLabelInfo

This function acquires device information corresponding to a label name.

Format

■Visual C++

long mdGetLabelInfo (long path, long netno, long stno, long lbcnt, void* lblst, long* devlst, unsigned long long* lbcode);

■Visual Basic

Integer mdGetLabelInfo(Integer path, Integer netno, Integer stno, Integer lbcnt, String lblst(0), Integer devlst(0), ULong lbcode);

■Visual C#

int mdGetLabelInfo (long int path, long int netno, long int stno, long int lbcnt, string[] lblst, long* int[] devlst, ulong lbcode);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
lbcnt	Number of labels	Specify the number of labels. (The number of labels can be specified within the range of 1 to 32767.)	IN
lblst	Label name array	Specify the storage address of label name for each label. (Specify a label name in Unicode (UTF-16).)	IN
devlst	Device name array	Specify a device to store the acquired device information. (Device information that is assigned to labels specified for the label name array (lblst) is stored in a randomly selected device format.)	OUT
lbcode	Label code	A value to identify whether the label of a CPU module is changed or not is stored. (A change in label settings can be checked by whether this value is changed or not. However, the value changes even when converting all in a CPU module.) Specification method for the label code (lbcode): Specify values to the argument (label code) of the mdRandRLabelEx function and mdRandWLabelEx function.	OUT

Device information assigned to labels specified to the label name array (lblst) is stored in a device specified to the device name array (devlst) in a randomly selected device format listed below.

devlst	Description	
devlst[0]	Number of blocks	
devlst[1]	Device type	Block 1
devlst[2]	Start device number	
devlst[3]	Number of read points	
devlst[4]	Device type	Block 2
devlst[5]	Start device number	
devlst[6]	Number of read points	
⋮	⋮	⋮
devlst[3(n-1)+1]	Device type	Block n
devlst[3(n-1)+2]	Start device number	
devlst[3(n-1)+3]	Number of read points	

The number of blocks can be specified within the range of 1 to 32767.

One block comprises of three elements such as a device type, the start device number, and the number of read points. The total number of blocks will be stored in the first element of the device name array (devlst).

Description

- This function reads labels of a CPU module specified to the network number (netno) and the station number (stno).
- Reserve the area for the device name array (devlst) in the call source.
- Reserve the area of the device name array (devlst) for the size equal to (lbcnt × 3 + 1).
- If any labels whose label information cannot be acquired exist in label names specified to the label name array (lblst), the mdGetLabelInfo function returns any of the errors shown below. In addition, the value '0' is stored for the device type, the start device number, and the number of read points of the label.

Error code	Error occurrence
-82 (FFAEH)	• A non-existent label was specified. • Devices assigned to labels do not support random read/write. • The specification method for devices assigned to labels is incorrect.
-84 (FFACH)	The specification method for devices assigned to labels is incorrect.

- The error response is returned in order of detection.
  If two labels (Label1: non-existent label name, Label2: incorrect device specification method by digit specification) are specified, only an error of Label1 (the first detected label) (-82) is returned.
- Even if the mdGetLabelInfo function returns the error (-82 or -84), the value is stored in the device name array (devlst) for the label that acquired device information successfully.

■Specification method for a label name specified to the device name array (lblst)

For the specification method for the label name to be specified for the device name array (lblst), refer to the following:
MELSEC iQ-R MELSECWinCPU Module User's Manual

Precautions

- Create a file with Unicode (UTF-16) character strings using an application (such as Notepad) of Windows.
- When a device is specified such as the bit specification of word device or the digit specification of label, the device information cannot be acquired.
- When a label to which a device is not assigned is specified, DevGV is stored in the device type.
- The device type 'DevGV' can be specified only with the functions which support label access (the mdRandRLabelEx and mdRandWLabelEx function).
- For details on the label communication function, refer to the following:
  MELSEC iQ-R MELSECWinCPU Module User's Manual

Example

The following shows an example for executing this function.

Ex.

Reading five labels (Label 1 to 5)

■Preparing for label names (Unicode character strings)

1. Describe a target label name in a text file, and save the file by specifying Unicode (UTF-16).
2. Read the label name from the saved text file in a binary format on a user program and store the address of the label name, which is to be passed onto the label name array (lblst), in the memory.

The following tables show the examples of values specified to the label name array (lblst) and data read to the device name array (devlst).

■Values specified to the label name array (lblst)

lblst	Specified value	Description
lblst[0]	First (Label1) label name storage address	Label name
lblst[1]	Second (Label2) label name storage address	Label name
lblst[2]	Third (Label3) label name storage address	Label name
lblst[3]	Fourth (Label4) label name storage address	Label name
lblst[4]	Fifth (Label5) label name storage address	Label name

■Values read to the label name array (devlst)

devlst	Read value	Description
devlst[0]	5	Number of blocks
devlst[1]	DevD	Device type
devlst[2]	10	Start device number
devlst[3]	1	Number of read points
devlst[4]	DevD	Device type
devlst[5]	11	Start device number
devlst[6]	1	Number of read points
devlst[7]	DevM	Device type
devlst[8]	100	Start device number
devlst[9]	1	Number of read points
devlst[10]	DevM	Device type
devlst[11]	101	Start device number
devlst[12]	1	Number of read points
devlst[13]	DevM	Device type
devlst[14]	102	Start device number
devlst[15]	1	Number of read points

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: *1 ☞ Page 94 ERROR CODES

*1 For a return value which does not exist in the reference, refer to the manual for the CPU module. (📖MELSEC iQ-R CPU Module User's Manual (Application))

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 74 mdRandRLabelEx
- Page 79 mdRandWLabelEx

mdOpen

This function opens a communication line (channel).

Format

■Visual C++

short mdOpen (short chan, short mode, long *path);

■Visual Basic

Short mdOpen(Short chan, Short mode, Integer path);

■Visual C#

short mdOpen(short chan, short mode, int path);

Argument

Argument	Name	Description	IN/OUT
chan	Channel	Specify a communication line (channel). ☞ Page 33 Channel number	IN
mode	Mode	Specify '-1.'	IN
path	Channel path	Specify the storage destination (address) of the channel path. (The opened line path is returned.)	OUT

Description

- This function opens a communication line which corresponds to the specified channel number. Make sure to execute this function before using MELSEC data link functions.
- The path of a channel, which is to be used for an argument of other functions, is returned to the pointer of the opened line path.
- To end a user program, close the opened channel path with the mdClose function.
- When using multiple channels, open the channel number one by one.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 62 mdClose

mdRandREx

This function reads data from devices randomly.

Format

■Visual C++

long mdRandREx(long path, long netno, long stno, long* dev, short* buf, long bufsize);

■Visual Basic

Integer mdRandREx(Integer path, Integer netno, Integer stno, Integer dev(0), Short buf(0), Integer bufsize);

■Visual C#

int mdRandREx(int path, int netno, int stno, int[] dev, short[] buf, int bufsize);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
dev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be read.	IN
buf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
bufsize	Read data storage destination size	Specify the area size reserved in the read data storage destination in byte units.	IN

■Specification method for the randomly selected device (dev)

dev	Description	
dev[0]	Number of blocks	
dev[1]	Device type of block 1	Block 1
dev[2]	Start device number of block 1	
dev[3]	Number of read points of block 1	
dev[4]	Device type of block 2	Block 2
dev[5]	Start device number of block 2	
dev[6]	Number of read points of block 2	
⋮	⋮	⋮
The number of blocks can be specified within the range of 1 to 32767.		

Description

- This function reads data of devices specified to the randomly selected device (dev) from a module which is specified to the network number (netno) and station number (stno).
  The read data is stored in the read data storage destination (buf) in word units in the order specified to the randomly selected device (dev).
  A bit device is stored per 16 points, a word device is stored per 1 point, and a double-word device is stored per word.

Example

The following shows an example for executing this function.

Ex.

Reading current values from M100 to M115, D10 to D13, M0 to M13, T10, LCV100, and LCN101

Device where data is randomly read	Current value
M100 to M115	All bits are OFF.
D10 to D13	10 is stored in D10, 200 in D11, 300 in D12, and 400 in D13.
M0 to M13	All bits are ON.
T10 current value	10 (1 second) is stored in T10.
LCN100 to LCN101	0x1 is stored in LCN100 and 0x10000 is in LCN101.

The following tables show the examples of values specified to the randomly selected device (dev) and read data storage destination size (bufsize) as well as values read to the read data storage destination (buf).

■Values specified to the randomly selected device (dev)

dev	Specified value	Description	
dev[0]	5	Number of blocks = 5	—
dev[1]	DevM	Device type of block 1 = M	Block 1: M100 to M115
dev[2]	100	Start device number of block 1 = 100	
dev[3]	16	Number of read points of block 1 = 16	
dev[4]	DevD	Device type of block 2 = D	Block 2: D10 to D13
dev[5]	10	Start device number of block 2 = 10	
dev[6]	4	Number of read points of block 2 = 4	
dev[7]	DevM	Device type of block 3 = M	Block 3: M0 to M13
dev[8]	0	Start device number of block 3 = 0	
dev[9]	14	Number of read points of block 3 = 14	
dev[10]	DevTN	Device type of block 4 = T	Block 4: T10
dev[11]	10	Start device number of block 4 = 10	
dev[12]	1	Number of read points of block 4 = 1	
dev[13]	DevLCN	Device type of block 5 = LCN	Block 5: LCN100 to LCN101
dev[14]	100	Start device number of block 5 = 100	
dev[15]	2	Number of read points of block 5 = 2	

■Values read to the read data storage destination (buf)

buf	Read device	Read value	Description
buf[0]	M100 to M115	0	All the bit devices from M100 to M115 are OFF.
buf[1]	D10	10	D10 = 10
buf[2]	D11	200	D11 = 200
buf[3]	D12	300	D12 = 300
buf[4]	D13	400	D13 = 400
buf[5]	M0 to M13	3FFFH	All the bit devices from M0 to M13 are ON.
buf[6]	T10	10	T10 = 10
buf[7]	LCN100	0x1	L of LCN100 = 0x0001
buf[8]			H of LCN100 = 0x0000
buf[9]	LCN101	0x10000	L of LCN101 = 0x0000
buf[10]			H of LCN101 = 0x0001

■Value specified to the read data storage destination size (bufsize)

(buf[0] to buf[10] = 11) × 2 = 22

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 77 mdRandWEx

mdRandRLabelEx

This function reads data from devices corresponding to labels randomly.

Format

■Visual C++

long mdRandRLabelEx(long path, long netno, long stno, long* dev, short* buf, long bufsize, unsigned long long lbcode);

■Visual Basic

Integer mdRandRLabelEx(Integer path, Integer netno, Integer stno, Integer dev(0), short buf(0), Integer bufsize, ULong lbcode);

■Visual C#

int mdRandRLabelEx(int path, int netno, int stno, int[] dev, short[] buf, int bufsize, ulong lbcode);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
dev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be read. (Specify the value acquired by the mdGetLabelInfo function.)	IN
buf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
bufsize	Read data storage destination size	Specify the area size reserved in the read data storage destination in byte units.	IN
lbcode	Label code	Specify the label code acquired by the mdGetLabelInfo function. (A change in label settings can be checked by whether this value is changed or not.)	IN

■Specification method for the randomly selected device (dev)

dev	Description	
dev[0]	Number of blocks	
dev[1]	Device type	Block 1
dev[2]	Start device number	
dev[3]	Number of read points	
dev[4]	Device type	Block 2
dev[5]	Start device number	
dev[6]	Number of read points	
⋮	⋮	⋮
dev[3(n-1)+1]	Device type	Block n
dev[3(n-1)+2]	Start device number	
dev[3(n-1)+3]	Number of read points	

The number of blocks can be specified within the range of 1 to 32767.

One block comprises of three elements such as a device type, the start device number, and the number of read points. Store the total number of blocks in the first element of the randomly selected device (dev).

For each element of blocks, the device name array (devlst) which is acquired by the mdGetLabelInfo function can be used.

Description

- This function reads devices specified to the randomly selected device (dev) from a module which is specified to the network number (netno) and the station number (stno).
- The read data is stored in the read data storage destination (buf) in word units in the order specified to the randomly selected device (dev). A bit device and a word device are stored per 1 point, and a double-word device is stored per word.
- When '0' is specified to the label code (lbcode), the device is read without the label code being checked.

Example

The following shows an example for executing this function.

Ex.

Reading current values from M100, D10 to D13, M0, T10, LCV100, and LCN100 to LCN101

Device where data is randomly read	Current value
M100	Bit is OFF.
D10 to D13	10 is stored in D10, 200 in D11, 300 in D12, and 400 in D13.
M0	Bit is ON.
T10 current value	10 (1 second) is stored in T10.
LCN100 to LCN101	0x1 is stored in LCN100 and 0x10000 is in LCN101.

The following tables show the examples of values specified to the randomly selected device (dev) and read data storage destination size (bufsize) as well as values read to the read data storage destination (buf).

■Values specified to the randomly selected device (dev)

dev	Specified value	Description	
dev[0]	5	Number of blocks = 5	—
dev[1]	DevM	Device type = M	Block 1: M100
dev[2]	100	Start device number = 100	
dev[3]	1	Number of read points = 1	
dev[4]	DevD	Device type = D	Block 2: D10 to D13
dev[5]	10	Start device number = 10	
dev[6]	4	Number of read points = 4	
dev[7]	DevM	Device type = M	Block 3: M0
dev[8]	0	Start device number = 0	
dev[9]	1	Number of read points = 1	
dev[10]	DevTN	Device type = T	Block 4: T10
dev[11]	10	Start device number = 10	
dev[12]	1	Number of read points = 1	
dev[13]	DevLCN	Device type = LCN	Block 5: LCN100 to LCN101
dev[14]	100	Start device number = 100	
dev[15]	2	Number of read points = 2	

■Values read to the read data storage destination (buf)

buf	Read device	Read value	Description
buf[0]	M100	0	M100 = OFF
buf[1]	D10	10	D10 = 10
buf[2]	D11	200	D11 = 200
buf[3]	D12	300	D12 = 300
buf[4]	D13	400	D13 = 400
buf[5]	M0	1	M0 = ON
buf[6]	T10	10	T10 = 10
buf[7]	LCN100	0x1	L of LCN100 = 0x0001
buf[8]			H of LCN100 = 0x0000
buf[9]	LCN101	0x10000	L of LCN101 = 0x0000
buf[10]			H of LCN101 = 0x0001

■Value specified to the read data storage destination size (bufsize)

(buf[0] to buf[10] = 11) × 2 = 22

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: *1 ☞ Page 94 ERROR CODES

*1 For a return value which does not exist in the reference, refer to the manual for the programmable controller CPU. (📖MELSEC iQ-R CPU Module User's Manual (Application))

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 66 mdGetLabelInfo
- Page 79 mdRandWLabelEx

mdRandWEx

This function writes data to devices randomly.

Format

■Visual C++

long mdRandWEx(long path, long netno, long stno, long* dev, short* buf, long bufsize);

■Visual Basic

Integer mdRandWEx(Integer path, Integer netno, Integer stno, Integer dev(0), Short buf(0), Integer bufsize);

■Visual C#

int mdRandWEx(int path, int netno, int stno, int[] dev, short[] buf, int bufsize);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
dev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be written.	IN
buf	Write data storage destination	Specify the storage destination (address) of write data. (Reserve a continuous area for the write data storage destination.)	IN
bufsize	Write data storage destination size	Unused (Even if a value is specified, the operation is not affected.)	IN

■Specification method for the randomly selected device (dev)

dev	Description	
dev[0]	Number of blocks	
dev[1]	Device type of block 1	Block 1
dev[2]	Start device number of block 1	
dev[3]	Number of write points of block 1	
dev[4]	Device type of block 2	Block 2
dev[5]	Start device number of block 2	
dev[6]	Number of write points of block 2	
⋮	⋮	⋮
The number of blocks can be specified within the range of 1 to 32767.		

Description

- This function writes data to a device, which is specified to the randomly selected device (dev), of a module specified to the network number (netno) and station number (stno).
  The data to be written is stored in the write data storage destination (buf) in word units.
  A bit device is stored per 16 points, a word device is stored per 1 point, and a double-word device is stored per word.
- Communication time varies significantly depending on the contents specified to the randomly selected device (dev).
  To reduce communication time, use the mdSendEx function (☞ Page 87 mdSendEx).
- Note that the extension comment information will be deleted when writing data to the block to which an extension comment is assigned (extension file register).
- Note that sub 2 or sub 3 program will be deleted when writing data to a block (extension file register) overlapping with the program setting area for sub 2 or sub 3.

Example

The following shows an example for executing this function.

Ex.

Turning OFF all the bits from M100 to M115 and writing 10 to D10, 200 to D11, 400 to D13, 0x1 to LCN100, and 0x10000 to LCN101

Device where data is randomly written	Description
M100 to M115	Turn all the bits OFF.
D10 to D13	Store 10 in D10, 200 in D11, 300 in D12, and 400 in D13.
LCN100 to LCN101	Store 0x1 in LCN100, and 0x10000 in LCN101.

The following tables show the examples of values specified to the randomly selected device (dev) and write data storage destination (buf).

■Values specified to the randomly selected device (dev)

dev	Specified value	Description	
dev[0]	3	Number of blocks = 3	—
dev[1]	DevM	Device type of block 1 = M	Block 1: M100 to M115
dev[2]	100	Start device number of block 1 = 100	
dev[3]	16	Number of write points of block 1 = 16	
dev[4]	DevD	Device type of block 2 = D	Block 2: D10 to D13
dev[5]	10	Start device number of block 2 = 10	
dev[6]	4	Number of write points of block 2 = 4	
dev[7]	DevLCN	Device type of block 3 = LCN	Block 3: LCN100 to LCN101
dev[8]	100	Start device number of block 3 = 100	
dev[9]	2	Number of write points of block 3 = 2	

■Values specified to the write data storage destination (buf)

buf	Specified value	Description
buf[0]	0	Turn all bit devices from M100 to M115 OFF.
buf[1]	10	D10 = 10
buf[2]	200	D11 = 200
buf[3]	300	D12 = 300
buf[4]	400	D13 = 400
buf[5]	0x0001	L of LCN100
buf[6]	0x0000	H of LCN100
buf[7]	0x0000	L of LCN101
buf[8]	0x0001	H of LCN101

■Number of bytes of write data storage destination (buf)

(buf[0] to buf[8] = 9) × 2 = 18

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 71 mdRandREx

mdRandWLabelEx

This function writes data to devices corresponding to labels randomly.

Format

■Visual C++

long mdRandWLabelEx(long path, long netno, long stno, long* dev, short* buf, long bufsize, unsigned long long lbcode);

■Visual Basic

Integer mdRandWLabelEx(Integer path, Integer netno, Integer stno, Integer dev(0), short buf(0), Integer bufsize, ULong lbcode);

■Visual C#

int mdRandWLabelEx(int path, int netno, int stno, int[] dev, short[] buf, int bufsize, ulong lbcode);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
dev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be written. (Specify the value acquired by the mdGetLabelInfo function.)	IN
buf	Write data storage destination	Specify the storage destination (address) of write data. (Reserve a continuous area for the write data storage destination.)	IN
bufsize	Write data storage destination size	Unused (Even if a value is specified, the operation is not affected.)	IN
lbcode	Label code	Specify the label code acquired by the mdGetLabelInfo function.	IN

■Specification method for the randomly selected device (dev)

dev	Description	
dev[0]	Number of blocks	
dev[1]	Device type	Block 1
dev[2]	Start device number	
dev[3]	Number of write points	
dev[4]	Device type	Block 2
dev[5]	Start device number	
dev[6]	Number of write points	
⋮	⋮	⋮
dev[3(n-1)+1]	Device type	Block n
dev[3(n-1)+2]	Start device number	
dev[3(n-1)+3]	Number of write points	

The number of blocks can be specified within the range of 1 to 32767.

One block comprises of three elements such as a device type, the start device number, and the number of write points. Store the total number of blocks in the first element of the randomly selected device (dev).

For each element of blocks, the device name array (devlst) which is acquired by the mdGetLabelInfo function can be used.

Description

- This function writes data to a device, which is specified to the randomly selected device (dev), of a module specified to the network number (netno) and the station number (stno).
- The data to be written is stored in the write data storage destination (buf) in word units. A bit device and a word device are stored per 1 point, and a double-word device is stored per word.
- Note that the extension comment information will be deleted when writing data to the block to which an extension comment is assigned (extension file register).
- Note that sub 2 or sub 3 program will be deleted when writing data to a block (extension file register) overlapping with the program setting area for sub 2 or sub 3.
- When '0' is specified to the label code (lbcode), the device is written without checking the label code.

Example

The following shows an example for executing this function.

Ex.

Turning OFF the bit of M100 and writing 10 to D10, 200 to D11, 300 to D12, 0x1 to LCN100, and 0x10000 to LCN101

Device where data is randomly written	Description
M100	Turns the bit OFF.
D10 to D13	Store 10 in D10, 200 in D11, 300 in D12, and 400 in D13.
LCN100 to LCN101	Store 0x1 in LCN100, and 0x10000 in LCN101.

The following tables show the examples of values specified to the randomly selected device (dev) and write data storage destination (buf).

■Values specified to the randomly selected device (dev)

dev	Specified value	Description	
dev[0]	3	Number of blocks = 3	—
dev[1]	DevM	Device type = M	Block 1: M100
dev[2]	100	Start device number = 100	
dev[3]	1	Number of write points = 1	
dev[4]	DevD	Device type = D	Block 2: D10 to D13
dev[5]	10	Start device number = 10	
dev[6]	4	Number of write points = 4	
dev[7]	DevLCN	Device type = LCN	Block 3: LCN100 to LCN101
dev[8]	100	Start device number = 100	
dev[9]	2	Number of write points = 2	

■Values specified to the write data storage destination (buf)

buf	Specified value	Description
buf[0]	0	M100 = OFF
buf[1]	10	D10 = 10
buf[2]	200	D11 = 200
buf[3]	300	D12 = 300
buf[4]	400	D13 = 400
buf[5]	0x0001	L of LCN100
buf[6]	0x0000	H of LCN100
buf[7]	0x0000	L of LCN101
buf[8]	0x0001	H of LCN101

■Number of bytes of write data storage destination (buf)

(buf[0] to buf[8] = 9) × 2 = 18

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: *1 ☞ Page 94 ERROR CODES

*1 For a return value which does not exist in the reference, refer to the manual for the programmable controller CPU. (📖MELSEC iQ-R CPU Module User's Manual (Application))

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 66 mdGetLabelInfo
- Page 74 mdRandRLabelEx

mdReceiveEx

This function reads data from devices in a batch.

Format

■Visual C++

long mdReceiveEx(long path, long netno, long stno, long devtyp, long devno, long* size, short* data);

■Visual Basic

Integer mdReceiveEx(Integer path, Integer netno, Integer stno, Integer devtyp, Integer devno, Integer size, Short data(0));

■Visual C#

int mdReceiveEx(int path, int netno, int stno, int devtyp, int devno, int size, short[] data);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
devtyp	Device type	Specify the device type for devices to be read in a batch.	IN
devno	Start device number	Specify the start number of devices to be read in a batch. (For bit devices, set the number in multiples of 8.)	IN
size	Read data size	Specify a read data size in byte units. (Specify the value in multiples of 4 when a double-word device (LZ, LTN, LCN, or LSTN) is specified. Alternatively, specify the value in multiples of 2 when a word device or bit device is specified. If the value other than that is specified, the size error (-5) will occur.)	IN/OUT
data	Read data storage destination	Specify the storage destination (address) of read data.	OUT

Description

- This function reads data in the order from a device specified to the device type (devtyp) and the start device number (devno) of a module which is specified to the network number (netno) and station number (stno). The data is read for the size specified to the read data size (size).
- When the read data size (size) exceeds the device range, a readable size is returned to the read data size (size).
- When a double-word device (LZ, LTN, LCN, or LSTN) is specified to the device type (devtyp), the following read data is stored to the read data storage destination (data).

Ex.

When the device type (devtyp) is LZ and read data size (size) is 8

Array	Value
data(0)	L of LZ0
data(1)	H of LZ0
data(2)	L of LZ1
data(3)	H of LZ1

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 87 mdSendEx

mdReceiveEx (message receive)

This function receives messages. (RECV function)

Format

■Visual C++

long mdReceiveEx(long path, long netno, long stno, long devtyp, long devno, long* size, short* data);

■Visual Basic

Integer mdReceiveEx(Integer path, Integer netno, Integer stno, Integer devtyp, Integer devno, Integer size, Short data(0));

■Visual C#

int mdReceiveEx(int path, int netno, int stno, int devtyp, int devno, int size, short[] data);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify '0' (0H).	IN
stno	Station number	Specify own station '255' (FFH).	IN
devtyp	Device type	Specify "RECV function: 101 (65H and DevMAIL)."	IN
devno	Channel number	Specify a channel number. • CC-Link IE Controller Network and CC-Link IE TSN: 1 to 8	IN
size	Receive message size	Specify a receive message size in byte units. (2 to 1920) (Specify the size with an even number. If it is specified with an odd number, the size error (-5) will occur.)	IN/OUT
data	Receive data storage destination	Specify the storage destination (address) of receive data. Data equivalent to '6 + (size)' bytes are stored.	OUT

Description

- The RECV instruction is the dedicated instruction for a CC-Link IE Controller Network module and CC-Link IE TSN module. This function supports the RECV instruction under the following conditions: a value specified to the channel path (path) is a value returned from the mdOpen function by specifying CC-Link IE Controller Network (channel No.151 to 158) or CC-Link IE TSN (channel No.281 to 288), and "RECV function: 101" is specified to the device type.
  For details on the RECV instruction, refer to the following:
  (MELSEC iQ-R Programming Manual (Module Dedicated Instructions))
- The function receives the message of a channel specified to the channel number (devno) from among the messages sent to a CC-Link IE Controller Network module and CC-Link IE TSN module.
- The function reads the message to the specified channel number (devno) in the order it was received.
- When specifying the same own station channel using the message communication function, execute the function after the previously executed function is completed. If the message communication function is executed simultaneously from multiple programs by specifying the same own station channel, an error will occur in the function executed later.
- When the actual size of a received message is smaller than the value specified to the receive message size (size), the actual data size is stored in the receive data storage destination (data[3] or higher), and the data size of the received message is returned to the receive message size (size).
- When the actual size of a received message is larger than the value specified to the receive message size (size), data up to the specified size is stored in the receive data storage destination (data[3] or higher).
- A received message is stored in the receive data storage destination (data).
  Information on a message send source (the network number (netno), station number (stno), and used channel number (devno) of a send station) is stored in data[0] to data[2]. Therefore, the storage size of a received message is '6 + (size)' byte.

data	Description
data[0]	Send station network number
data[1]	Send station number
data[2]	Channel used by send station
data[3] or higher	• Received message (actual data) • (2 to 1920 bytes)

- Arguments of the mdReceiveEx function corresponding to the control data (device) of the dedicated instruction (RECV) are as shown below:

Device	Item	Corresponding argument and return value
+0	Error completion type	—
+1	Completion status	ret (return value)
+2	Own station storage channel	devno
+3	Channel used by send station	data[2]
+4	Send station network number	data[0]
+5	Send station number	data[1]
+6	Not used	—
+7	Not used	—
+8	Arrival monitoring time	—
+9	Receive data length	size
+10	Not used	—
+11	Clock setting flag	—
+12	Clock data (Set only in an abnormal state.)	—
+13		—
+14		—
+15		—
+16	Error-detected network number	—
+17	Error-detected station number	—

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 89 mdSendEx (message send)

mdSendEx

This function writes data to devices in a batch.

Format

■Visual C++

long mdSendEx(long path, long netno, long stno, long devtyp, long devno, long* size, short* data);

■Visual Basic

Integer mdSendEx(Integer path, Integer netno, Integer stno, Integer devtyp, Integer devno, Integer size, Short data(0));

■Visual C#

int mdSendEx(int path, int netno, int stno, int devtyp, int devno, int size, short[] data);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. ☞ Page 34 Network number and station number	IN
devtyp	Device type	Specify the device type for devices to be written in a batch.	IN
devno	Start device number	Specify the start number of devices to be written in a batch. (For bit devices, set the number in multiples of 8.)	IN
size	Write data size	Specify a write data size in byte units. (Specify the value in multiples of 4 when a double-word device (LZ, LTN, LCN, or LSTN) is specified. Alternatively, specify the value in multiples of 2 when a word device or bit device is specified. If the value other than that is specified, the size error (-5) will occur.)	IN/OUT
data	Write data storage destination	Specify the storage destination (address) of write data. (Reserve a continuous area for the write data storage destination.)	IN

Description

- This function writes data in the order from a device specified to the device type (devtyp) and the start device number (devno) of a module which is specified to the network number (netno) and the station number (stno). The data is written for the size specified to the write data size (size).
- The function checks arguments and verifies whether the sum of address and size determined by the arguments is within the device range.
  When the write data size (size) exceeds the device range, a writable size is returned to the write data size (size).
- Note that the extension comment information will be deleted when writing data to the block to which an extension comment is assigned (extension file register).
- Note that sub 2 or sub 3 program will be deleted when writing data to a block (extension file register) overlapping with the program setting area for sub 2 or sub 3.
- When a double-word device (LZ, LTN, LCN, or LSTN) is specified to the device type (devtyp), store the following write data to the read data storage destination (data).

Ex.

When the device type (devtyp) is LZ and write data size (size) is 8

Array	Value
data(0)	L of LZ0
data(1)	H of LZ0
data(2)	L of LZ1
data(3)	H of LZ1

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 82 mdReceiveEx

mdSendEx (message send)

This function sends messages. (SEND function)

Format

■Visual C++

long mdSendEx(long path, long netno, long stno, long devtyp, long devno, long* size, short* data);

■Visual Basic

Integer mdSendEx(Integer path, Integer netno, Integer stno, Integer devtyp, Integer devno, Integer size, Short data(0));

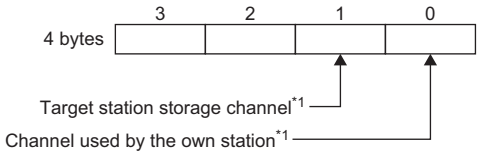
■Visual C#

long mdSendEx(int path, int netno, int stno, int devtyp, int devno, int size, short[] data);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
netno	Network number	Specify the network number of a target module. A logical station number cannot be specified. ☞ Page 34 Network number and station number	IN
stno	Station number	Specify the station number of a target module. A logical station number cannot be specified. ☞ Page 34 Network number and station number	IN
devtyp	Device type	Select the sending means from the following items. When "Group number" or "All stations" is specified to the station number, only "Without arrival confirmation" is valid. • With arrival confirmation: 101 (65H, DevMAIL) • Without arrival confirmation: 102 (66H, DevMAILNC)	IN
devno	Channel number	Specify a channel number.	IN
size	Send data size	Specify a send data size in byte units. (2 to 1920) (Specify the size with an even number.)	IN/OUT
data	Send data storage destination	Specify the storage destination (address) of send data. (Reserve a continuous area for the send data storage destination.)	IN

• Specify the channel number as follows.



*1 CC-Link IE Controller Network and CC-Link IE TSN module: 1 to 8

Description

- The SEND instruction is the dedicated instruction for a CC-Link IE Controller Network module and CC-Link IE TSN module. This function supports the SEND instruction under the following conditions: a value specified to the channel path (path) is a value returned from the mdOpen function by specifying CC-Link IE Controller Network (channel No.151 to 158) or CC-Link IE TSN (channel No.281 to 288), and "With arrival confirmation: 101" or "Without arrival confirmation: 102" is specified to the device type.
  For details on the SEND instruction, refer to the following:
  (📖MELSEC iQ-R Programming Manual (Module Dedicated Instructions))
- The function sends a message from a CC-Link IE Controller Network module or CC-Link IE TSN module to the targets (network number, station, and channel) specified to the network number (netno), station number (stno), and device type (devtyp).
- When sending data to the same target station storage channel of the receiving station with arrival confirmation, make sure to send the data after the receiving station reads the previously sent data with the message receive function (or the RECV instruction). Otherwise, an error will occur. If an error is detected, send the data again after a while.
- When specifying the same own station channel using the message communication function, execute the function after the previously executed function is completed. If the message communication function is executed simultaneously from multiple programs by specifying the same own station channel, an error will occur in the function executed later.
- Arguments of the mdSendEx function corresponding to the control data (device) of the dedicated instruction (SEND) are as shown below:

Device	Item	Corresponding argument and return value
+0	Execution/error completion type	devtyp
+1	Completion status	ret (return value)
+2	Own station channel	devno
+3	Target station storage channel	devno
+4	Target station network number	netno
+5	Target station number	stno
+6	Not used	—
+7	Number of retransmissions (retries)	—
+8	Arrival monitoring time	—
+9	Send data length	size
+10	Not used	—
+11	Clock setting flag	—
+12	Clock data	—
+13		—
+14		—
+15		—
+16	Error-detected network number	—
+17	Error-detected station number	—

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose
- Page 84 mdReceiveEx (message receive)

mdTypeRead

This function reads the model code of a CPU module.

Format

■Visual C++

short mdTypeRead(long path, short stno, short* buf);

■Visual Basic

Short mdTypeRead(Integer path, Short stno, Short buf);

■Visual C#

short mdTypeRead(int path, short stno, short buf);

Argument

Argument	Name	Description	IN/OUT
path	Channel path	Specify the path of the opened channel. (Use the path which is returned when the mdOpen function is executed.)	IN
stno	Station number	Specify the network number and station number of a target module. ☞ Page 34 Network number and station number	IN
buf	Model code	Specify the storage destination (address) of a model code. (The model code of a read CPU module is stored.)	OUT

Description

This function reads the model code of the CPU module with the station number specified to the station number (stno).

For CPU modules other than the following, the model code is undefined.

Model code (hexadecimal)	CPU module
0041H	Q02CPU, Q02HCPU
0042H	Q06HCPU
0043H	Q12HCPU
0044H	Q25HCPU
0049H	Q12PHCPU
004AH	Q25PHCPU
004BH	Q12PRHCPU
004CH	Q25PRHCPU
004DH	Q02PHCPU
004EH	Q06PHCPU
0250H	Q00JCPU
0251H	Q00CPU
0252H	Q01CPU
0260H	Q00UJCPU
0261H	Q00UCPU
0262H	Q01UCPU
0263H	Q02UCPU
0266H	Q10UDHCPU
0267H	Q20UDHCPU
0268H	Q03UDCPU
0269H	Q04UDHCPU
026AH	Q06UDHCPU
026BH	Q13UDHCPU
026CH	Q26UDHCPU
02E6H	Q10UDEHCPU
02E7H	Q20UDEHCPU
02E8H	Q03UDECPU

Model code (hexadecimal)	CPU module
02E9H	Q04UDEHCPU
02EAH	Q06UDEHCPU
02EBH	Q13UDEHCPU
02ECH	Q26UDEHCPU
02EDH	Q50UDEHCPU
02EEH	Q100UDEHCPU
0362H	Q04UDPVCPU
0363H	Q06UDPVCPU
0364H	Q13UDPVCPU
0365H	Q26UDPVCPU
0366H	Q03UDVCPU
0367H	Q04UDVCPU
0368H	Q06UDVCPU
036AH	Q13UDVCPU
036CH	Q26UDVCPU
0541H	L02CPU
0543H	L02SCPU
0544H	L06CPU
0545H	L26CPU
0548H	L26CPU-BT
0549H	L02CPU-P
054AH	L26CPU-PBT
0641H	LJ72GF15-T2
0642H	NZ2GF-ETB
2014H	Q172DCPU(-S1)
2015H	Q173DCPU(-S1)
2018H	Q172DSCPU
2019H	Q173DSCPU
2043H	Q12DCCPU-V
2044H	Q24DHCCPU-V
2045H	Q24DHCCPU-LS
2046H	Q24DHCCPU-VG
2047H	Q26DHCCPU-LS
4800H	R04CPU
4801H	R08CPU
4802H	R16CPU
4803H	R32CPU
4804H	R120CPU
4805H	R04ENCPU
4806H	R08ENCPU
4807H	R16ENCPU
4808H	R32ENCPU
4809H	R120ENCPU
4820H	R12CCPU-V
4841H	R08PCPU
4842H	R16PCPU
4843H	R32PCPU
4844H	R120PCPU
4891H	R08SFCPU
4892H	R16SFCPU
4893H	R32SFCPU
4894H	R120SFCPU
4C00H	R16MTCPU

Model code (hexadecimal)	CPU module
4C01H	R32MTCPU
4C02H	R64MTCPU
4C20H	R102WCPU-W

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 94 ERROR CODES

Relevant function

- Page 70 mdOpen
- Page 62 mdClose

5 ERROR CODES

This chapter shows error codes and corrective actions.

5.1 Common Error Codes

The following table shows the common error codes.

Error code[*1]		Error description	Corrective action
Dec	Hex		
1	0001H	■Driver not started The driver is not started.	• Check the channel path. • Check the error that occurred when the driver is started. • Check if the MELSECWinCPU module is running normally. • Check if the operating system is running normally.
2	0002H	■Timeout error • A timeout occurred while waiting for the response. • During CC-Link communication, the request was issued to other stations when the station number of the own station is '64'. • The module specified as the communication target is not supported.	• Review the operating status of the access target station and condition of the module. • Retry on the user program. • Increase the timeout value of MELSEC data link function. • When issuing a request to other stations during CC-Link communication, set the station number of the own station other than '64'. • Check if the module specified as the communication target is supported.
66	0042H	■Already opened error The specified channel has already been opened.	The open processing is required only one time. (If this error occurred, the path of the correct channel will be returned to the argument.)
67	0043H	■Already closed error The specified channel has already been closed.	The close processing is required only one time.
69	0045H	■Unsupported function performing error An unsupported function in the target station was performed.	• Check the path of the channel, network number, and station number. • Check if the function performed in the target station is supported.
70	0046H	■Station number error • The specified station number is incorrect. • The request for other stations was issued to the own station, or the network number was not '0' even though the station number was the own station (FFH).	Correct the network number and station number specified in the user program.
77	004DH	■Memory reservation error ■Resource shortage error ■Task over error • Reserving sufficient memory failed, or there are too many tasks that use MELSEC data link functions or C Controller module dedicated functions.	• The memory may be insufficient. End another running task or reduce the access size. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware. • Reduce the number of tasks that use MELSEC data link functions or C Controller module dedicated functions and retry. • Review the size or number specified to the arguments of the user program.
85	0055H	■Network channel number error (When a SEND/RECV request is issued) Channel number error	Check the specified channel number when the SEND/RECV request is issued.
102	0066H	■Data send error ■Starting error Sending data failed, or an attempt was made to send data during the initial processing of the bus control or during the bus reset.	• Retry. • Retry after the READY LED turns ON. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
103	0067H	■Reception error Receiving data failed, or an attempt was made to receive data during the bus reset.	• Retry. • Retry after the READY LED turns ON. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
130	0082H	■Device number error • The specified device number is out of range. • The specified bit device number is not a multiple of 8.	Check the device number.

Error code[*1]		Error description	Corrective action
Dec	Hex		
131	0083H	■Number of device points error • The specified number of device points is out of range. • The specified number of bit device points is not a multiple of 8.	Check the specified number of device points.
16384 to 20479	4000H to 4FFFH	■Errors detected in the access target CPU module	Refer to the user's manual of the access target CPU module.
-25056	9E20H	■Processing code error A request which cannot be performed in the request destination was issued.	Check the network number of the request destination.
-26334	9922H	■Reset error • The bus was reset while accessing another station. • The bus was reset while monitoring with CW Configurator.	• Retry. • Monitor again.
-26336	9920H	■Routing request error for unsupported station A routing request to another loop was issued to a station which does not support the routing function.	Check the settings of routing parameters.
-28150	920AH	■Device access error during data link stop The devices (RX, RY, RWw, and RWr) of the own station were accessed when the data link was not performed.	• Check the specified device start number and size, or the device range of the parameter for the master station. • Restart the data link. (Note that data is written/read despite this error, however, the contents of the data will not be guaranteed.)
-28151	9209H	■Abnormal data reception error Abnormal response data received.	Check if an error occurred in the request destination CPU module or link module. (If the status is normal, resend the request.)
-28158	9202H	■Hardware error A WDT (system) error or a CRAM CRC error occurred.	• Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware. • Check the event history in the module diagnostics of CW Configurator and remove the causes of the error. After that, turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28410	9106H	■Target CPU busy error The target CPU module is busy.	• Add the processing to wait for the completion of the target operation or to retry the operation in the user program. • Increase the timeout time specified to the argument in the user program.
-28412	9104H	■Target CPU unsupported error An unsupported request was issued to the target CPU module.	Change the target CPU number specified in the user program.
-28413	9103H	■Target CPU down error The target CPU module is down.	Check the operating status of the target CPU module and troubleshoot the error according to the user's manual of the CPU module.
-28414	9102H	■Target CPU abnormal start error A request was issued to the CPU module which is not operating normally.	Check the operating status of the target CPU module and troubleshoot the error according to the user's manual of the CPU module.
-28415	9101H	■Target CPU major error A request was issued to the CPU module in which a major error occurred.	Check the operating status of the target CPU module and troubleshoot the error according to the user's manual of the CPU module.
-28416	9100H	■Target CPU mounting error A request was issued by specifying the CPU number in the state where no CPU module is mounted.	• Check the mounting condition of the target CPU module. • Check that the CPU number specified to the device type is correct.
-28605	9043H	■Resource shortage error Reserving sufficient memory failed, or a user program which uses MELSEC data link functions is forcibly terminated.	• Restart Windows. • If the same error occurs, turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28607	9041H	■System error	• Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28608	9040H	■System error	• Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28622	9032H	■Target module busy error ■Target module offline error • The target module is busy. • The own station channel or the target station storage channel is used for other instructions, or multiple identical instructions are being executed. • An attempt was made to access the target module when it was offline.	• Add the processing to wait for the completion of the target operation or to retry the operation in the user program. • Check whether any error occurred on the network module, or it is offline.

Error code[*1]		Error description	Corrective action
Dec	Hex		
-28624	9030H	■Function unsupported error • Any processing was performed to the module which does not support the station-based block data assurance function for cyclic data. • Any processing was performed to the module on which the station-based block data assurance function for cyclic data is not set. • An attempt was made to access a module which is not controlled by the host CPU module.	• Check if the target CC-Link module supports the station-based block data assurance function for cyclic data. • Check if the station-based block data assurance function for cyclic data is set for the target module. • Check if the control CPU of the target module is the host CPU module.
-28625	902FH	■Intelligent function module offline error An attempt was made to access the intelligent function module when it was offline.	Check the status of the intelligent function module and access the module while it is online.
-28626	902EH	■Control data setting value out of range error The specified control data is out of range.	Review the user program and check the value set to the control data.
-28627	902DH	■Transient unsupported error Transient transmission cannot be performed via the specified communication route and target. (Another station was specified when the station number of the own station is '64' during CC-Link communication.)	• Check the communication route and target which support the transient request. • Change the station number of the own station.
-28628	902CH	■Pointer address specification error An incorrect address was specified to the argument pointer.	• Check the address of the specified pointer. • Check whether the address matches with the argument pointer type. • If a structure member address is specified, check the pragma pack specification (#pragma pack) and review the structure declaration to match alignment.
-28631	9029H	■Buffer access range error • The specified offset is out of range. • The specified offset and its size is out of range.	• Check the specified offset. • Check the specified buffer size. • Check the offset and its size.
-28632	9028H	■I/O number error • The specified I/O number is out of range. • No accessible module is mounted on the accessed slot with the specified I/O number. • Parameters were changed.	• Check that the start I/O number specified to the device type is correct. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check that a module with the start I/O number specified to the device type is running normally. • Close the opened channel and open it again.
-28633	9027H	■Non-controlled module read error An attempt was made to access a module which is not controlled by the host CPU when reading data from a non-controlled module is not allowed.	Check if the control CPU of the specified module is the host CPU module (MELSECWinCPU module).
-28634	9026H	■Intelligent function module down error The intelligent function module has an error.	• Check the mounting condition of the target CPU module. • Replace the intelligent function module or base unit.
-28635	9025H	■Intelligent function module error No intelligent function module is mounted on the accessed slot with the specified I/O number.	• Check the specified I/O number and the slot. • Check the mounting condition of the target CPU module.
-28636	9024H	■Control bus error The control bus to the intelligent function module has an error.	• Check if an error occurs in the bus master CPU (CPU No.1) in the multiple CPU system. • Check the mounting condition of the target CPU module. • Replace the intelligent function module or base unit.
-28638	9022H	■Multiple CPU unsupported operation error	Reset the bus master CPU (CPU No.1) or the bus.
-28640	9020H	■Y STOP/PAUSE error The request for outputting or writing to the buffer memory is issued when the operating status of the CPU module is Y STOP or PAUSE.	Change the Y output status to the Y OUT state.
-28653	9013H	■I/O assignment error • An attempt was made to read the input value (X) from an output module. • An attempt was made to write the output value (Y) to an input module. • An attempt was made to read the output value (Y) from an input module.	Check the input number (X) and output number (Y).
-28654	9012H	■Non-controlled module write error An attempt was made to access a module which is not controlled by the host CPU.	Check if the control CPU of the specified module is the host CPU module (MELSECWinCPU module).

Error code[*1]		Error description	Corrective action
Dec	Hex		
-28660	900CH	■Access size error The specified size is out of range.	Review the specified offset and size.
-28661	900BH	■Inaccessible error Inaccessible area was specified.	Review the specified offset and size.
-28662	900AH	■CPU number specification error The specified CPU number is out of range or unavailable.	• Review the specified CPU number. • Check the operating status of the specified CPU module.
-28664	9008H	■Data send area occupied	Retry.
-28665	9007H	■No registration data error	Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28666	9006H	■Data length error	Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28668	9004H	■Reply data stored error	Resend the request.
-28669	9003H	■Area number error The specified area number, offset address, and mode are out of range.	Review the area number, offset address, and mode.
-28671	9001H	■Module identification error	• Review the parameters. • Check the specified module. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-28672	9000H	■Processing code error	Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.

*1 When the return value of the function is of the long type, the value will be eight digits in hexadecimal.

5.2 C Controller Module Dedicated Functions

The following table shows the error codes of the C Controller module dedicated functions.

Error code		Error description	Corrective action
Dec	Hex		
-201	FF37H	■Module identification error The specified module identification is unavailable.	Check the specified module identification.
-203	FF35H	■I/O signal error The specified I/O signal is out of range.	Check the specified I/O signal.
-204	FF34H	■I/O access size error The specified access size of I/O signal is out of range.	• Check the size (I/O signal and read/write size) for accessing the specified I/O signal. • Check if the size is specified according to the argument description. (Even number, in a multiple of 2, and in a multiple of 4 etc.)
-205	FF33H	■I/O number error • The specified I/O number is out of range. • No accessible module is mounted on the accessed slot with the specified I/O number. • Parameters were changed.	• Check that the start I/O number specified to the device type is correct. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check that a module with the start I/O number specified to the device type is running normally. • Close the opened channel and open it again.
-208	FF30H	■Offset error The specified offset (the start device number) is out of range.	• Check the specified offset (the start device number). • Check the I/O number specified to the device type is correct.
-209	FF2FH	■Buffer memory size error • The specified offset (the start device number) and its size are out of range. • The address of data storage buffer pointer is 0. • The specified size is 0.	• Check whether the offset (the start device number) and the sum of the offset (the start device number) and size exceeds the buffer memory size of a module specified with the start I/O number. • Check the specified data storage buffer pointer.
-210	FF2EH	■Read area size error The read area size is too small.	• Check the read size. • Check the read area size.
-211	FF2DH	■Time setting error The specified time is out of range.	Check the specified time.
-214	FF2AH	■Intelligent function module error A slot with the specified I/O number was accessed in the state where no intelligent module was mounted.	• Check the specified I/O number and the slot. • Check the mounting condition of the target CPU module.
-217	FF27H	■Driver not started The driver is not started.	Check if the driver is started.
-222	FF22H	■Bus master CPU reset error The remote RESET of the bus master CPU (CPU No.1) failed.	Check that the MELSECWinCPU module where the remote RESET is being executed is the bus master CPU (CPU No.1).
-223	FF21H	■Memory reservation error Reserving sufficient memory failed.	• Reduce the memory size to be specified. • Restart the MELSECWinCPU system.
-225	FF1FH	■Event number specification error The specified event number is out of range or duplicated.	Check the specified event number.
-227	FF1DH	■Control code send error Sending the control code failed.	• Retry. • Check if the MELSECWinCPU module is running normally. • Reset the bus of the MELSECWinCPU module.
-231	FF19H	■Event timeout error A timeout occurred while waiting for an event.	• Increase the timeout time. • Check if the interrupt event number (interrupt pointer number) is set correctly.
-232	FF18H	■CPU number specification error • The specified CPU number is incorrect. • The specified CPU cannot issue the request. • The host CPU module in which the remote operation is performed is specified by a number for specifying other stations.	• Check that the CPU number specified to the device type is correct. • Do not issue a request, that generated an error, to the specified CPU. • Specify '0' (host CPU) to perform the remote operation on the host CPU module.
-234	FF16H	■Event wait error An error other than timeout occurred while waiting for the event.	• Check if a program is forcibly being terminated. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-235	FF15H	■Number of event settings specification error The specified number of event settings is out of range.	Check the number of specified event settings.

Error code		Error description	Corrective action
Dec	Hex		
-236	FF14H	■Remote operation specification code error The remote operation specification code is out of range.	Check the specified remote operation specification code.
-253	FF03H	■Device number specification error • The specified device number is out of range. • The specified bit device number is not a multiple of 16.	Correct the start device number of the specified device.
-254	FF02H	■Device type specification error The specified device type is unavailable.	Check the specified device type.
-255	FF01H	■Size specification error • The specified size is out of range. • The specified size is 0.	• Check the specified start device number and size. • Check if the size is specified according to the argument description. (Even number, in a multiple of 2, and in a multiple of 4 etc.)
-256	FF00H	■Response completion wait timeout error A timeout occurred while waiting for completion of a response of a processing requested to other CPU modules.	• Increase the timeout time specified to the argument. • Review and correct the user program (including other tasks which execute motion CPU interaction functions). • Review the program used for the request destination CPU module and correct it to perform the processing requested from other CPU modules, for example, by adding the WAIT instruction.
-258	FEFEH	■Interrupt pointer number specification error The value specified as the interrupt pointer number is out of range.	Check the specified value.
-263	FEF9H	■Caller flag error The value specified to the caller flag is out of range.	Review the specified value, and specify a value within the range.
-264	FEF8H	■Pointer error The address of the specified pointer is incorrect.	Check the address of the specified pointer.
-265	FEF7H	■Target system specification error The value specified in the target system is out of range.	Check the specified value.
-269	FEF3H	■Network number error The specified network number is out of range.	Check the specified network number.
-270	FEF2H	■Channel number error The specified channel number is out of range.	Check the specified channel number.
-271	FEF1H	■Target station number error The specified target station number is out of range.	Check the specified target station number.
-291	FEDDH	■Fixed scan communication area unreserved error An attempt was made to access a CPU module in which fixed scan communication area is not reserved.	Use the fixed scan communication function in the multiple CPU setting of system parameter, and check if 1K word or more is set for the fixed scan communication area.
-297	FED7H	■Input/output number, network number incorrect specification An input/output number out of the range (other than 000H to FFFH or 3E0H to 3E3H) was specified.	Correct the argument of the function.
-298	FED6H	■Input/output number, network number incorrect specification An input/output number with no module was specified.	• Correct the argument of the function. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check the operating status of the specified module.
-299	FED5H	■Input/output number, network number incorrect specification • An input/output number of a module which does not support the function was specified. • The dedicated instruction was specified with the specified module or mode.	• Check the applicability of the dedicated function (such as the support status and executable mode) by referring to the manual for relevant modules. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check the operating status of the specified module.
-300	FED4H	■Input/output number, network number incorrect specification An input/output number of a module, that cannot be specified, was specified.	• Correct the argument of the function. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check the operating status of the specified module.
-301	FED3H	■Input/output number, network number incorrect specification A network number out of the range (other than 1 to 239) was specified.	
-302	FED2H	■Input/output number, network number incorrect specification A network number which does not exist was specified.	

Error code		Error description	Corrective action
Dec	Hex		
-303	FED1H	■Input/output number, network number incorrect specification An I/O module or intelligent function module controlled by other CPU modules was specified.	• Correct the argument of the function. • Delete a module controlled by other CPU modules, which has been specified by the link direct device, from the program. • Specify the network module controlled by the host CPU module with a link direct device. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check the operating status of the specified module.
-304	FED0H	■Input/output number, network number incorrect specification The target module cannot be identified with a function to specify an I/O module or intelligent function module. (The character strings to specify the target module are incorrect.)	• Correct the argument of the function. • Check if the network module corresponding to a channel specified with the mdOpen function is running normally. • Check the operating status of the specified module.
-305	FECFH	■Input/output number, network number incorrect specification The specified I/O module or intelligent function module is in the state where it cannot execute the function.	The possible cause is a hardware failure of the I/O module or intelligent function module specified. Please contact your local Mitsubishi Electric sales office or representative.
-306	FECEH	■Device, buffer memory incorrect specification The specified device exceeded the usable range.	Correct the argument of the function.
-307	FECDH	■Device, buffer memory incorrect specification A device that cannot be specified was specified.	Correct the argument of the function.
-308	FECCH	■Program fault The specified argument structure is incorrect.	Correct the argument of the function.
-309	FECBH	■Program fault The specified number of devices is incorrect.	Correct the argument of the function.
-310	FECAH	■Operation error An unusable character string for a function is specified.	Correct the argument of the function.
-311	FEC9H	■Operation error Data out of the specifiable range was entered.	Correct the argument of the function.
-314	FEC6H	■Module major error An error was detected in the intelligent function module when a function was executed.	The possible cause is a hardware failure of the intelligent function module where the error was detected. Please contact your local Mitsubishi Electric sales office or representative.
-315	FEC5H		
-316	FEC4H	■Other CPU module major error An error was detected in other CPU modules when a function was executed.	Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the host CPU module or other CPU modules where the error was detected. Please contact your local Mitsubishi Electric sales office or representative.
-317	FEC3H		
-318	FEC2H	■System bus error An error was detected on the system bus.	• Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module, I/O module, intelligent function module, base unit, or extension cable. Please contact your local Mitsubishi Electric sales office or representative.
-319	FEC1H	■Hardware failure A hardware failure was detected.	• Take measures to reduce noise. • Turn the power of the CPU module OFF and ON or reset the hardware. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.
-321	FEBFH	■System bus error An error was detected on the system bus.	• Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.
-322	FEBEH	■System bus error An error was detected on the system bus.	• Check the connection status of the extension cable. • Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.
-323	FEBDH	■System bus error An error was detected on the system bus.	• Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.

Error code		Error description	Corrective action
Dec	Hex		
-324	FEBCH	■System bus error An error was detected on the system bus.	• Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.
-325	FEBBH	■System bus error An error was detected on the system bus.	• Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.
-326	FEBAH	■System bus error An error was detected on the system bus.	• Take measures to reduce noise. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.
-327	FEB9H	■Module major error • A major error was notified from the intelligent function module. • The I/O module or intelligent function module is not mounted properly or was removed during operation.	• Check the connection status of the extension cable. • Reset the bus. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please contact your local Mitsubishi Electric sales office or representative.

5.3 MELSEC Data Link Functions

The following table shows the error codes of MELSEC data link functions.

Error code[*1]		Error description	Corrective action
Dec	Hex		
-1	FFFFH	■Path error • The specified path is unavailable. • The taskDelete was executed in the task with a MELSEC data link function. • The task using a MELSEC data link function was deleted with the taskDelete.	• Use a path pointer returned with the mdOpen function. • Check if the taskDelete was executed in the task with a MELSEC data link function. • Check if the task using a MELSEC data link function was deleted with the taskDelete.
-2	FFFEH	■Device number error • The specified device number is out of range. • The specified bit device number is not a multiple of 8. • The device number and the points for the same block specified to reading/writing device randomly exceeds the device range.	• Check the start device number of the specified device. • Check the device number plus the number of points. • To specify a bit device, specify a start device number with a multiple of 8. • Check if the specified device is available in the CPU module on the target station.
-3	FFFDH	■Device type error The specified device type is unavailable.	• Check if the device is accessible to the specified network number and station number. (☞ Page 33 Argument specifications) • Check if the specified device is available in the target station.
-5	FFFBH	■Size error • The device number and the size exceeds the device range. • The device number and the size exceeds the range for the same block. • The access was made with an odd-number bytes.	• Check the specified device size. • Check the device number and the size. • Specify an even-number byte.
-6	FFFAH	■Number of blocks error The number of blocks specified to the function for reading/writing device randomly is out of range.	Check the number of the specified blocks.
-8	FFF8H	■Channel number error The channel number specified with the mdOpen function is unavailable.	Check the specified channel number.
-11	FFF5H	■Insufficient buffer area error The area size of the read data storage destination is smaller than the read data size.	Check the area size of the read data storage destination and the read data size.
-12	FFF4H	■Block number error The specified block number is unavailable.	• Check the block number (device type) of the specified device. • Check if the specified device and block number are available in the target.
-13	FFF3H	■Write protect error The specified block number of the extension file register overlaps with the write protect area of the memory card.	• Check the block number (device type) of the extension file register. • Check the write protect switch of the memory card.
-16	FFF0H	■Station number/network number error • The specified station number or network number is out of range. • A device which cannot be accessed by the target station is specified.	• Check the specified station number and network number. • Check the devices which can be accessed by the target station.
-17	FFEFH	■All stations/group number specification error A function which does not support specifying all stations and group number was specified.	• Check if the function allows specifying all stations and group number. • When "All stations" or "Group number" is specified to the station number, specify "Without arrival confirmation" to the device type.
-18	FFEEH	■Remote operation error The specification code specified with the mdControl function is unavailable.	Check the specified specification code.
-19	FFEDH	■SEND/RECV channel number error The channel number specified in the SEND/RECV function is out of range.	Specify the channel number within the range. • CC-Link IE Controller Network and CC-Link IE TSN: 1 to 8
-31	FFE1H	■Module load error Loading modules required for executing functions failed.	• The memory may be insufficient. End another running task or reduce the access size. • Check the status of the system drive of the MELSECWinCPU module.

Error code[*1]		Error description	Corrective action
Dec	Hex		
-32	FFE0H	■Resource timeout error The resource is being used by another task/thread and is not released within 30 seconds.	• Retry. • The memory may be insufficient. End another running task. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-33	FFDFH	■Communication target unsupported error The module specified as the communication target by a network number and station number is not supported.	• Check if the module specified as the communication target by a network number and station number is supported. • Check the settings of the access target set in CW Configurator.
-34	FFDEH	■Registry open error Opening parameter files in the registry failed.	Check if the access target is correctly set with CW Configurator.
-35	FFDDH	■Registry/parameter read error Reading a registry or parameter file is failed.	• Check if the access target is correctly set with CW Configurator. • Check if the setting for the channel number is enabled. • Reset the bus after checking the parameters with CW Configurator again and writing them. • For an access target, check the 'Accessing other stations' section of each network described in the following section: ☞ Page 16 Accessible range and accessible devices • Check if the access target is correct.
-37	FFDBH	■Communications initialization error Initializing the setting for communication failed.	• Retry. • The memory may be insufficient. End another running task. • Check the available memory capacity. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-42	FFD6H	■Close error Communications cannot be closed.	• Retry. • Check if the MELSECWinCPU module is running normally. • Turn the power of the MELSECWinCPU module system OFF and ON or reset the hardware.
-43	FFD5H	■ROM operation error A TC setting value was written to the CPU module during ROM operation.	Change the TC setting value during RAM operation.
-52	FFCCH	■MELSEC data link function service error MELSEC data link function service is disabled.	Enable the MELSEC data link function service with CW Configurator.
-53	FFCBH	■Timeout value error The specified timeout value is out of range.	Check the specified time out value.
-54	FFCAH	■I/O number error The specified I/O number is out of range.	Check the specified I/O number.
-55	FFC9H	■Logical station number error The specified logical station number is out of range.	Check the specified logical station number.
-56	FFC8H	■Target CPU error The specified target CPU is out of range.	Check the specified target CPU.
-80	FFB0H	■Target CPU error The CPU module on the target station does not support labels.	• Check the specified network number and station number. • Check that the label communication function can access the CPU module on the target station.
-81	FFAFH	■Label code mismatch error Label assignment information of the CPU module was changed.	Acquire label information with the mdGetLabelInfo function again.
-82	FFAEH	■Label incorrect value error An incorrect label name was specified. • Non-existent label name • Label name assigned to a device which does not support random read/write. • Label name assigned to a device which is specified by the inappropriate specification method (index modification or indirect specification)	Check the specified label name or the device specification method.
-83	FFADH	■Size error The number of labels exceeded the range.	Check the number of labels.
-84	FFACH	■Device specification method error A device was specified by inappropriate specification method (bit specification or digit specification).	Check the device specification method.
-200 to -327	FF38H to FEB9H	Refer to the following functions. ☞ Page 98 C Controller Module Dedicated Functions	

Error code[*1]		Error description	Corrective action
Dec	Hex		
-475 to -3839	FE25H to F101H	Refer to the following manual. Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)	
-4097 to -8192	EFFFH to E000H	Refer to the following manual. MELSEC iQ-R CC-Link IE Controller Network User's Manual (Application) MELSEC-Q CC-Link IE Controller Network Reference Manual	
-8193 to -12288	DFFFH to D000H	Refer to the following manual. MELSEC iQ-R CC-Link IE Field Network User's Manual (Application) MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual MELSEC-L CC-Link IE Field Network Master/Local Module User's Manual MELSEC iQ-R CC-Link IE TSN User's Manual (Application)	
-12289 to -16384	CFFFH to C000H	Refer to the following manual. MELSEC iQ-R Ethernet User's Manual (Application) MELSEC iQ-R CC-Link IE TSN User's Manual (Application)	
-16385 to -20480	BFFFH to B000H	Refer to the following manual. MELSEC iQ-R CC-Link System Master/Local Module User's Manual (Application) MELSEC-Q CC-Link System Master/Local Module User's Manual MELSEC-L CC-Link System Master/Local Module User's Manual	

*1 When the return value of the function is of the long type, the value will be eight digits in hexadecimal.

APPENDIX

Appendix 1 Method for Replacing Existing Products

This section explains methods for replacing existing products (Q10WCPU-W1-J and Q10WCPU-W1-CFJ) with a MELSEC iQ-R MELSECWinCPU module (R102WCPU-W).

Replacement of projects

For details on upgrading programming development environment, refer to manuals corresponding to the Microsoft Corporation product or consult Microsoft Corporation.

Existing product	MELSEC iQ-R MELSECWinCPU module
Microsoft Visual Studio 2010 (Visual C++, Visual Basic) Microsoft Visual Studio 2008 (Visual C++, Visual Basic)	Microsoft Visual Studio 2019 (Enterprise, Professional) Microsoft Visual Studio 2017 (Enterprise, Professional)

For the procedure of creating programs for a MELSEC iQ-R MELSECWinCPU module, refer to the following:

☞ Page 9 Programming Creating Procedure

Replacement of library files and header files

Replace a library file and header file with the one shown in the following table.

Programming language	Library name	Existing product	MELSEC iQ-R MELSECWinCPU module
C++	MELSEC data link function	MdFunc32.lib	MDFuncWinCPU32.lib
	Bus interface function	QBFFunc32.lib	• MDFuncWinCPU32.lib • CCPUFuncWinCPU32.lib
Visual Basic	MELSEC data link function	Mdfunc.vb	MDFuncWinCPU.vb
	Bus interface function	QbfFunc32.vb	• MDFuncWinCPU.vb • CCPUFuncWinCPU.vb

A

Replacement of functions

For functions which cannot be used with a MELSEC iQ-R MELSECWinCPU module, replace them with the ones shown in the following tables.

MELSEC data link functions

Function of existing products	Availability in a MELSEC iQ-R MELSECWinCPU module	Replaced with[*1]
mdOpen	○	—
mdClose	○	—
mdSend	×	mdSendEx function, mdBdSendEx function, or mdBdWriteLinkDeviceEx function
mdReceive	×	mdReceiveEx function, mdBdReceiveEx function, or mdBdReadLinkDeviceEx function
mdRandW	×	mdRandWEx function or mdBdRandWEx function
mdRandR	×	mdRandREx function or mdBdRandREx function
mdDevSet	×	mdDevSetEx function or mdBdDevSetEx function
mdDevRst	×	mdDevRstEx function or mdBdDevRstEx function
mdTypeRead	○	—
mdControl	○	—
mdInit	×	—
mdBdModSet	×	—
mdBdModRead	×	—
mdBdLedRead	×	—
mdBdSwRead	×	—
mdBdVerRead	×	—
mdSendEx	○	—[*2]
mdReceiveEx	○	—[*2]
mdRandWEx	○	—[*2]
mdRandREx	○	—[*2]
mdDevSetEx	○	—
mdDevRstEx	○	—

*1 Check the specifications of the function before replacement since changing arguments may be required for the replacement in some cases.

*2 To access own station devices or buffer memory, use a function whose name starts with 'mdBd.'

Bus interface functions

Function of existing products	Availability in a MELSEC iQ-R MELSECWinCPU module	Replaced with*1
QBF_Open	×	—
QBF_Close	×	—
QBF_X_In_Bit	×	—*2
QBF_X_In_Word	×	—*2
QBF_X_In	×	—*2
QBF_Y_Out_Bit	×	—*2
QBF_Y_Out_Word	×	—*2
QBF_Y_Out	×	—*2
QBF_Y_In_Bit	×	—*2
QBF_Y_In_Word	×	—*2
QBF_Y_In	×	—*2
QBF_ToBuf	×	—*2
QBF_FromBuf	×	—*2
QBF_UnitInfo	×	CCPU_GetUnitInfo function
QBF_StartWDT	×	—
QBF_ResetWDT	×	—
QBF_StopWDT	×	—
QBF_ReadStatus	×	—
QBF_ReadStatusEx	×	—
QBF_ControlLED	×	—
QBF_Reset	×	CCPU_Reset function
QBF_WaitEvent	×	CCPU_WaitEvent function
QBF_WaitUnitEvent	×	CCPU_WaitUnitEvent function
QBF_ControlProgram	×	—

*1 Check the specifications of the function before replacement since changing arguments may be required for the replacement in some cases.

*2 To access own station devices or buffer memory, use a function whose name starts with 'mdBd.'

A

Replacement of device types

The following shows device types, which cannot be used for a MELSEC iQ-R MELSECWinCPU module, and their alternative methods.

To use any of the following device types in user programs, replace them in accordance with the alternative methods.

MELSEC data link functions

■Common device types

Device type not available in a MELSEC iQ-R MELSECWinCPU module		Alternative method
Device name	**Device name**	
Extension file register	DevER0 to 256	Access devices by replacing the block number with ZR devices in 32K point units. ■Before replacement • Device type: DevERnnn (nnn = 0 to 256) • Start device number: mmm ■After replacement • Device type: DevZR • Start device number: (nnn × 32768) + mmm
Timer setting value main	DevTM	These devices cannot be used in an access target CPU for this product. Access the devices after replacing them with other devices.
Timer setting value sub1	DevTS	
Timer setting value sub2	DevTS2	
Timer setting value sub3	DevTS3	
Counter setting value main	DevCM	
Counter setting value sub1	DevCS	
Counter setting value sub2	DevCS2	
Counter setting value sub3	DevCS3	
Accumulator	DevA	
Index register	DevV	

The following shows examples for replacing an extension file register.

Example: DevER0 with the start device number 0

- Start device: DevZR
- Start device number: (0 × 32768) + 0 = 0

Example: DevER0 with the start device number 256

- Start device: DevZR
- Start device number: (0 × 32768) + 256 = 256

Example: DevER1 with the start device number 0

- Start device: DevZR
- Start device number: (1 × 32768) + 0 = 32768

Example: DevER10 with the start device number 5

- Start device: DevZR
- Start device number: (10 × 32768) + 5 = 327685

■Device types for CC-Link modules

Device type not available in a MELSEC iQ-R MELSECWinCPU module		Alternative method
Device name (device)	**Device name**	
Own station remote input (RX)	DevX	The devices can be accessed by the methods described in the following: • ☞ Page 109 Refreshing devices(When accessing a network module on the own station) • ☞ Page 109 Specifying a module access device(When accessing a network module on the own station)
Own station remote output (RY)	DevY	
Own station link register (for sending) (—)	DevWw	
Own station link register (for receiving) (—)	DevWr	

Device type not available in a MELSEC iQ-R MELSECWinCPU module		Alternative method
Device name (device)	**Device name**	
Own station link special relay (SB)	DevSM	The devices can be accessed by the method described in the following: ☞ Page 109 Specifying a module access device(When accessing a network module on the own station)
	DevQSB	
Own station link special register (SW)	DevSD	
	DevQSW	
Own station random access buffer (—)	DevMRB	
Own station buffer memory (—)	DevSPB	
Other station buffer memory (—)	DevRBM	The devices can be accessed by the method described in the following: ☞ Page 109 Specifying a module access device(When accessing a network module on other stations)
Other station random access buffer (—)	DevRAB	
Other station remote input (RX)	DevRX	The devices can be accessed by the methods described in the following: • ☞ Page 109 Refreshing devices(When accessing a network module on other stations) • ☞ Page 109 Specifying a module access device(When accessing a network module on other stations)
Other station remote output (RY)	DevRY	
Other station link register (—)	DevRW	
Other station link special relay (SB)	DevSB	
Other station link special register (SW)	DevSW	

Alternative methods

■Refreshing devices

Access target	Alternative method
When accessing a network module on the own station	Configure the refresh setting so that link devices (M, B, D, and W) of a MELSECWinCPU module are refreshed by link devices of a network module.
	Access devices (M, B, D and W) of a MELSECWinCPU module with a MELSEC data link function.
When accessing a network module on other stations	Configure the refresh setting for a CPU module on other stations so that devices of the CPU module are refreshed by link devices of a network module.
	Specify another station to the network number and station number of a MELSEC data link function to access devices of a CPU module on other stations.

■Specifying a module access device

Access target	Alternative method
When accessing a network module on the own station	Open a communication line by specifying 'bus interface' to the mdOpen function channel.
	Specify the module access device (DevSPG) for the device type of MELSEC data link functions and access the area*1 to which link devices are assigned in the buffer memory of a network module.
When accessing a network module on other stations	Open a communication line by specifying 'bus interface' to the mdOpen function channel.
	Specify another station to the network number and station number of a MELSEC data link function.
	Specify the module access device (DevSPG) for the device type of MELSEC data link functions and access the area*1 to which link devices are assigned in the buffer memory of a network module.

*1 For the buffer memory address to which link devices are assigned, refer to the manual for an access target network module.

■Using the mdBdReadLinkDeviceEx function or the mdBdWriteLinkDeviceEx function

Access own station link devices of a network module with the mdBdReadLinkDeviceEx function or the mdBdWriteLinkDeviceEx function. For details, refer to the following corresponding functions:

☞ Page 58 mdBdReadLinkDeviceEx, Page 61 mdBdWriteLinkDeviceEx

MEMO

INDEX

I

FUNCTION INDEX

C

M

REVISIONS

*The manual number is given on the bottom left of the back cover.

Revision date	*Manual number	Description
January 2022	SH(NA)-082433ENG-A	First edition

Japanese manual number: SH-082430-A

TRADEMARKS

Microsoft, Visual Basic, Visual C++, Visual C#, and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.
In some cases, trademark symbols such as '™' or '®' are not specified in this manual.

SH(NA)-082433ENG-A(2201)
MODEL: R102WCPU-W-P-E

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN