



Programmable Controller

MELSEC iQ-R
series

MELSEC iQ-R C Controller Module Programming Manual

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. If products are used in a different way from that specified by manufacturers, the protection function of the products may not work properly.

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

Considerations for the Wind River Systems product

C Controller module has an embedded real-time operating system, VxWorks, manufactured by Wind River Systems, Inc. in the United States. We, Mitsubishi, make no warranty for the Wind River Systems product and will not be liable for any problems and damages caused by the Wind River Systems product during use of C Controller module.

For the problems or specifications of the Wind River Systems product, refer to the corresponding manual or consult Wind River Systems, Inc.

Contact information is available on the following website.

- Wind River Systems, Inc.: www.windriver.com

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

SAFETY PRECAUTIONS	1
CONDITIONS OF USE FOR THE PRODUCT	1
CONSIDERATIONS FOR USE	2
INTRODUCTION	2
RELEVANT MANUALS	7
TERMS	8
GENERIC TERMS AND ABBREVIATIONS	8
CHAPTER 1 COMMON ITEMS	10
1.1 Header Files	10
1.2 C Controller Module Dedicated Functions	11
Program processing	11
Considerations	11
Argument specification	13
1.3 MELSEC Data Link Functions	14
Program processing	14
Considerations	15
Accessible range and devices	17
Argument specification	44
1.4 Motion Module Dedicated Class	52
Program processing	52
Classes	55
Members	57
Data type	58
Considerations	62
1.5 Considerations on Interrupt Service Routine (ISR)	63
CHAPTER 2 FUNCTION LIST	64
2.1 C Controller Module Dedicated Functions	64
C Controller module dedicated functions	64
C Controller module dedicated functions for ISR	66
2.2 MELSEC Data Link Functions	67
2.3 Motion Module Dedicated Class	68
CHAPTER 3 DETAILS OF FUNCTION	69
3.1 C Controller Module Dedicated Functions	69
CCPU_ChangeCCIEFBCycPrm	69
CCPU_ChangeFileSecurity	70
CCPU_ClearError	71
CCPU_Control	72
CCPU_DedicatedDInst	73
CCPU_DedicatedGInst	75
CCPU_DedicatedJInst	77
CCPU_DedicatedMInst	79
CCPU_DisableInt	82
CCPU_EnableInt	83
CCPU_EndCCIEFBDataAssurance	84
CCPU_EntryCCIEFBRefEndFunc	85

CCPU_EntryInt	86
CCPU_EntryTimerEvent	88
CCPU_EntryWDTInt	90
CCPU_FromBuf	91
CCPU_FromBufHG	92
CCPU_GetCCIEFBDiagnosticInfo	93
CCPU_GetConstantProcessStatus	95
CCPU_GetCounterMicros	96
CCPU_GetCounterMillis	97
CCPU_GetCpuStatus	98
CCPU_GetDotMatrixLED	100
CCPU_GetErrInfo	102
CCPU_GetFileSecurity	103
CCPU_GetIDInfo	104
CCPU_GetLEDStatus	105
CCPU_GetOpSelectMode	107
CCPU_GetPowerStatus	108
CCPU_GetRTC	109
CCPU_GetSerialNo	110
CCPU_GetSwitchStatus	111
CCPU_GetUnitInfo	112
CCPU_LockFWUpdate	115
CCPU_MountMemoryCard	116
CCPU_ReadDevice	117
CCPU_ReadLinkDevice	118
CCPU_ReadMCUnitLabel	119
CCPU_ReadMCUnitLabelBit	121
CCPU_RegistEventLog	122
CCPU_Reset	123
CCPU_ResetDevice	124
CCPU_ResetWDT	125
CCPU_RestoreDefaultCCIEFBCycPrm	126
CCPU_SetDevice	127
CCPU_SetDotMatrixLED	128
CCPU_SetLEDStatus	130
CCPU_SetOpSelectMode	131
CCPU_SetRTC	132
CCPU_ShutdownRom	133
CCPU_StartCCIEFBDataAssurance	134
CCPU_StartWDT	135
CCPU_StopWDT	136
CCPU_SysClkRateGet	137
CCPU_SysClkRateSet	138
CCPU_ToBuf	139
CCPU_ToBufHG	140
CCPU_UnlockFWUpdate	141
CCPU_UnmountMemoryCard	142
CCPU_WaitEvent	143
CCPU_WaitSwitchEvent	145
CCPU_WaitTimerEvent	146
CCPU_WaitUnitEvent	147

CCPU_WriteDevice	149
CCPU_WriteLinkDevice	150
CCPU_WriteMCUnitLabel	151
CCPU_WriteMCUnitLabelBit	155
CCPU_X_In_BitEx	156
CCPU_X_In_WordEx	157
CCPU_Y_In_BitEx	158
CCPU_Y_In_WordEx	159
CCPU_Y_Out_BitEx	160
CCPU_Y_Out_WordEx	161
3.2 C Controller module dedicated functions for ISR	162
CCPU_DisableInt_ISR	162
CCPU_EnableInt_ISR	163
CCPU_FromBuf_ISR	164
CCPU_FromBufHG_ISR	165
CCPU_GetCounterMicros_ISR	166
CCPU_GetCounterMillis_ISR	167
CCPU_GetDotMatrixLED_ISR	168
CCPU_ReadDevice_ISR	170
CCPU_RegistEventLog_ISR	171
CCPU_ResetDevice_ISR	172
CCPU_SetDevice_ISR	173
CCPU_SetDotMatrixLED_ISR	174
CCPU_SetLEDStatus_ISR	176
CCPU_ToBuf_ISR	177
CCPU_ToBufHG_ISR	179
CCPU_WriteDevice_ISR	180
CCPU_X_In_Word_ISR	181
CCPU_Y_In_Word_ISR	183
CCPU_Y_Out_Word_ISR	185
3.3 MELSEC Data Link Functions	187
mdClose	187
mdControl	188
mdDevRstEx	189
mdDevSetEx	190
mdGetLabelInfo	191
mdInit	194
mdOpen	195
mdRandREx	196
mdRandRLabelEx	199
mdRandWEx	202
mdRandWLabelEx	204
mdReceiveEx	206
mdReceiveEx	207
mdSendEx	209
mdSendEx	210
mdTypeRead	212
3.4 Motion Module Dedicated Class	215
MCFB::RefreshLabels	215
MCFBExecute::SetExecute	216
MCFBEnable::SetEnable	218

MCv_Jog::SetEnableJog	220
-----------------------------	-----

CHAPTER 4 ERROR CODE LIST 222

4.1 Common Error Codes	222
4.2 C Controller Module Dedicated Functions	224
4.3 MELSEC Data Link Functions	229
4.4 Motion Module Dedicated Class	232
4.5 Error Codes Different From Conventional Functions	233

APPENDIX 234

Appendix 1 Example for Replacing Ladder by C Language	234
Program example	234
Appendix 2 How to Replace a Q12DCCPU-V	236
Replacement of projects	236
Replacement of VxWorks standard API functions	236
Replacement of functions	236
Replacement of device type	236
Compilation of replaced project	239
Appendix 3 Correspondence Table to Q12DCCPU-V Functions	240
C Controller module dedicated functions	240
C Controller module dedicated functions for ISR	240
Bus interface functions	241
Bus interface functions for ISR	242
MELSEC data link functions	243
Appendix 4 How to Replace a Q06CCPU-V	244
Replacement of projects	244
Replacement of VxWorks standard API functions	244
Replacement of functions	244
Replacement of device type	244
Compilation of replaced project	247
Appendix 5 Correspondence Table to Q06CCPU-V Functions	248
Bus interface functions	248
MELSEC data link functions	249

INDEX 250

FUNCTION INDEX 251

REVISIONS	252
WARRANTY	253
INFORMATION AND SERVICES	254
TRADEMARKS	254

RELEVANT MANUALS

Manual name [manual number]	Description	Available form
MELSEC iQ-R C Controller Module Programming Manual [SH-081371ENG] (this manual)	Programming specifications and dedicated function library of a C Controller module	e-Manual PDF
CW Workbench/CW-Sim Operating Manual [SH-081373ENG]	System configuration, specifications, functions, and troubleshooting of CW Workbench/CW-Sim	e-Manual PDF



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.

TERMS

Unless otherwise specified, this manual uses the following terms.

Term	Description
MELSEC data link function	A data link function library offered by C Controller module. It is used to access other CPU modules 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 module	A generic term for the following modules: <ul style="list-style-type: none">• RJ71GN11-T2 CC-Link IE TSN master/local module• RJ71GN11-SX CC-Link IE TSN master/local module• Motion module
CC-Link IE Controller Network module	A generic term for the following modules: <ul style="list-style-type: none">• RJ71GP21-SX CC-Link IE Controller Network modules• RJ71GP21S-SX CC-Link IE Controller Network modules• RJ71EN71 (when a CC-Link IE Controller Network function is used.)
CC-Link IE Field Network module	A generic term for the following modules: <ul style="list-style-type: none">• RJ71GF11-T2 CC-Link IE Field Network master/local modules• RJ71EN71 (when a CC-Link IE Field Network function is used).• Simple motion modules
CC-Link module	An abbreviation for CC-Link module, RJ61BT11.
CW Configurator	A generic product name for SWnDND-RCCPU. ('n' indicates its version.)
CW Workbench	An abbreviation for C Controller module and C intelligent function module engineering tool, CW Workbench.
CW-Sim	An abbreviation for VxWorks simulator that can operate and debug the C Controller module and C intelligent function module programs on a personal computer on which CW Workbench installed, without connecting to the actual machine (target).
C Controller module	A generic term for MELSEC iQ-R series C Controller module.
C Controller module dedicated function	A dedicated function library offered by C Controller module. It is used to control the C Controller module.
MCFB setting function	A generic term for the following functions: <ul style="list-style-type: none">• MCFBExecute::SetExecute• MCFBEnable::SetEnable• MCv_Jog::SetEnableJog
R12CCPU-V	An abbreviation for R12CCPU-V C Controller module.
RCPU	A generic term for R04CPU, R04ENCPU, R08CPU, R08PCPU, R08ENCPU, R08SFCPU, R16CPU, R16PCPU, R16ENCPU, R16SFCPU, R32CPU, R32PCPU, R32ENCPU, R32SFCPU, R120CPU, R120PCPU, R120ENCPU, and R120SFCPU.
VxWorks	A product name for the real-time operating system manufactured by Wind River Systems, Inc.
Intelligent function module	A generic term for modules which has functions other than input and output, such as A/D converter module and D/A converter module.
High Performance model QCPU	A generic term for Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU.
Interface board for a personal computer	A generic term for the following modules: <ul style="list-style-type: none">• NZ81GN11-T2 CC-Link IE TSN interface board• NZ81GN11-SX CC-Link IE TSN interface board
Bus interface	An abbreviation for a MELSEC iQ-R bus interface.
Bus interface communication	An abbreviation for MELSEC iQ-R bus interface communication.
Bus interface function	A generic term for a dedicated function library (QBF functions) provided by a MELSEC-Q series C Controller module.
Process CPU	A generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU, and Q25PHCPU.
Basic model QCPU	A generic term for Q00JCPU, Q00CPU, and Q01CPU.
Universal model QCPU	A generic term for Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDECPU, Q03UDVCPU, Q04UDHCPU, Q04UDEHCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDHCPU, Q06UDEHCPU, Q06UDVCPU, Q06UDPVCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDEHCPU, Q13UDVCPU, Q13UDPVCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDEHCPU, Q26UDVCPU, Q26UDPVCPU, Q50UDEHCPU, Q100UDEHCPU.

Generic term/abbreviation	Description
Dedicated function library	A generic term for C Controller module dedicated functions, MELSEC data link functions, and motion module dedicated classes.
Redundant CPU	A generic term for Q12PRHCPU and Q25PRHCPU.

1 COMMON ITEMS

A user program is created by using the VxWorks standard API functions^{*1} and dedicated function library provided by a C Controller module in accordance with the specification of VxWorks, the operating system of C Controller module.

^{*1} For details on the VxWorks standard API functions, refer to the following programmer's guide supported.

 VxWorks "KERNEL PROGRAMMER'S GUIDE"

Dedicated function libraries provided by a C Controller module are as follows:

- C Controller module dedicated function
- MELSEC Data Link function
- Motion module dedicated class^{*1}

^{*1} Available to use in a C Controller module with the firmware version '15' or later.

Point

For the execution procedure of user programs, refer to the following:

 MELSEC iQ-R C Controller Module User's Manual

1.1 Header Files

Include the following header files in a user program to use the dedicated function library.

Dedicated function library	Header file
C Controller module dedicated function	CCPUFunc.h
MELSEC data link function	MDFunc.h
Motion module dedicated class	MCFBFunc.h

Point

A header file is stored in a C Controller module.

 MELSEC iQ-R C Controller Module User's Manual

1.2 C Controller Module Dedicated Functions

C Controller dedicated functions of the dedicated function libraries are used to control C Controller module. These functions can be used for reading status of C Controller module or accessing resources such as LED control and clock.

Program processing

The following shows the user program processing using the C Controller module dedicated function.

1. Start a task.
2. Read the status of C Controller module and access the resources such as LED control and clock by using the C Controller module dedicated function.
3. Complete the task.

Considerations

Considerations for user WDT (user watchdog timer)

■User WDT error occurrence

If a user WDT cannot be reset due to a user program runaway, a user WDT error will occur.

In this case, take the following corrective actions.

- Increase the user WDT period set with the CCPU_StartWDT function.
- Lower the number of tasks with high CPU utilization or make them deactivated.
- Review the user program.

Reset the C Controller module once the corrective actions have been taken.

Point

In the user program, a user WDT can be used to monitor the hardware and status of user program, and processing timeout for accessing and controlling each module.

■User WDT setting range

The user WDT period can be set within the range of 100 ms to 10,000 ms.

■Output when a user WDT error occurs

When a user WDT error occurs, the output turns OFF.

Considerations for dedicated functions

■Argument

- For the first argument, "pclInstName", categorizing the dedicated instructions (D/DP/J/JP/G/GP/M) is not required.
- Devices of the own station cannot be specified. Reserve the required area with a user program, and specify the start address of the area to the relevant argument.
- Specify devices of other stations by using a character string. Set the value (the number of elements to which one is added for the termination code) to the number of elements for the array.

(Example) To specify D4: `char cDev[3] = {"D4"};`

Set '3', the total number of '2' for D4 and '1' for termination code, to the number of elements for the character string.

- When the data type is device name (control data, input data, and output data), specify the argument using an array.
- The size is required to be set according to the number of elements for the array, which is specified to the argument. When data type is 16-bit binary, BCD 4-digit, or real number, set the size to '1'. When data type is 32-bit binary or BCD 8-digit, set the size to '2'.
- When data type is bit (completion device), specify the argument using an array. Specify '1' for completion or '0' for incompletion to the first array. In the second array, '0' or '1' is stored for the normal completion or abnormal completion, respectively.
- For arguments without the setting data, set the setting data to NULL, and set the size to '0'.
- Errors which occur in the dedicated function are not registered in the event history.

■Executing a dedicated function

Before executing a dedicated function, check the status of a target module specified in the second argument. The response may not be returned depending on the target module status.

Function name	Target
CCPU_DedicatedDInst	A CPU module specified by the start I/O number of the target CPU module
CCPU_DedicatedMInst	
CCPU_DedicatedGInst	A module where the module dedicated instruction is executed that is specified by the start input/output number
CCPU_DedicatedJInst	A network module specified by a network number.

Argument specification

This section shows the argument specifications of the C Controller module dedicated functions.

Device type

The following tables show the device types specified to the C Controller module dedicated functions.
Devices are defined in the header file "CCPUFunc.h".

Point

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

■Device types for accessing internal user devices and internal system devices

Device type can be specified to the argument sDevType of the CCPU_WriteDevice/CCPU_ReadDevice/CCPU_SetDevice/CCPU_ResetDevice/CCPU_WriteDevice_ISR/CCPU_ReadDevice_ISR/CCPU_SetDevice_ISR/CCPU_ResetDevice_ISR functions.


Device name (Device)	Device type		
	Code		Device name
	Decimal	Hexadecimal	
Internal relay (M)	4	4H	Dev_CCPU_M
Special relay (SM)	5	5H	Dev_CCPU_SM
Data register (D)	13	DH	Dev_CCPU_D
Special register (SD)	14	EH	Dev_CCPU_SD
Link relay (B)	23	17H	Dev_CCPU_B
Link register (W)	24	18H	Dev_CCPU_W
File register (ZR)	220	DCH	Dev_CCPU_ZR

■Device types for accessing link devices

Device type can be specified to the argument sDevType of the CCPU_WriteLinkDevice/CCPU_ReadLinkDevice functions.

Device name (Device)	Device type		
	Code		Device name
	Decimal	Hexadecimal	
Link input (Jn\X)*1	1000	3E8H	Dev_LX
Link output (Jn\Y)*1	2000	7D0H	Dev_LY
Link relay (Jn\B)*1	23000	59D8H	Dev_LB
Link register (Jn\W)*1	24000	5DC0H	Dev_LW
Link special relay (Jn\SB)*1,*2	25000	61A8H	Dev_LSB
Link special register (Jn\SW)*1,*2	28000	6D60H	Dev_LSW

*1 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using C Controller module dedicated functions, refer to the following:
 MELSEC iQ-R C Controller Module User's Manual

*2 For motion modules (RD78G/GH), only link special relays (Jn\SB) and link special registers (Jn\SW) can be accessed.

1.3 MELSEC Data Link Functions

MELSEC data link functions are the integrated communication function libraries which are independent of the communication protocols.

A program to communicate with a CPU module can be created regardless of a target hardware or communication protocols by using the MELSEC data link functions.

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

Communication function	Description
Bus interface communication	Accesses a CPU module mounted on the same base unit.
CC-Link IE Controller Network communication	Accesses a CPU module on the CC-Link IE Controller Network via a CC-Link IE Controller Network module.
CC-Link IE Field Network communication	Accesses a CPU module on the CC-Link IE Field Network via a CC-Link IE Field Network module.
CC-Link IE TSN communication	Accesses a CPU module on the CC-Link IE TSN via a CC-Link IE TSN module.
MELSECNET/H network communication	Accesses a CPU module on the MELSECNET/H network via a MELSECNET/H network module.
CC-Link communication	Accesses a CPU module on the CC-Link via a CC-Link module.

Program processing

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

When accessing with a device name

1. Start a task.
2. Open a communication line. (mdOpen function)
3. Perform dummy access (such as device/model name reading) to an access target.
4. Access the target by using the MELSEC data link function.
5. To stop accessing the target, go to the procedure 6.
To access the target again, go back to the procedure 4.
6. Close the communication line. (mdClose function)
7. Complete the task.

When accessing with a label name

1. Start a task.
2. Open a communication line. (mdOpen function)
3. Acquire device information (label assignment information) from a target CPU module. (mdGetLabelInfo function)
4. Access the target CPU module by using the acquired device information (label assignment information).
(mdRandRLabelEx/mdRandWLabelEx function)
5. Check if there is no change in the device information (label assignment information) of the target CPU module.
If it is changed, go back to the procedure 3.
6. To stop accessing the target, go to the procedure 7.
To access the target again, go back to the procedure 4.
7. Close the communication line. (mdClose function)
8. Complete the task.

Considerations

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

Considerations for programming

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

Perform the open/close processing of communication line (the mdOpen/mdClose function) only once at the start of task (task activation) and at the end of task (task completion) respectively in each user program. Opening/closing the line every communication decreases the communication performance.

■Execution after using the mdOpen function

At the first execution of the function after using the mdOpen function, it takes longer to execute the function since the CPU module information needs to be acquired. The succeeding processing time can be shortened by performing dummy access at the first time.

■Access to other stations on the same task

Accessing 33 or more other stations simultaneously on the same task of C Controller module using a user program may decrease the communication performance. To access other stations simultaneously on the same task, limit it to 32 or less stations.

■mdGetLabelInfo function call

The mdGetLabelInfo function does not need to be called each time to access a target CPU module.

Only if the error occurs (Error code: -81) when accessing by using the mdRandRLabelEx/mdRandWLabelEx function, call the mdGetLabelInfo function again.

■taskDelete execution

Do not execute the taskDelete in a task using the MELSEC data link function. Also, do not delete a task using the MELSEC data link function with the taskDelete. Otherwise, the MELSEC data link function may not operate properly.

Accessing devices of the own station and of a CPU module on other stations

An interlock may be required to be provided depending on the link status of the own station and other stations.

■Access to devices on the own station

Create a user program that provides an interlock to enable reading/writing data when the following conditions are satisfied to access devices via each network module.

Module to be routed	Condition that requires interlock
CC-Link IE Controller Network module	When the following conditions are met: <ul style="list-style-type: none"> • The bit of the own station data link error status (SB0049) is OFF (data linking). • The bit corresponding to the communication target station that is stored in 'Data link status of each station' (SW00B0 to SW00B7) is OFF (normal communication).
CC-Link IE Field Network module	
CC-Link IE TSN module	
MELSECNET/H network module	When the following conditions are met: <ul style="list-style-type: none"> • The module status (SB20) is OFF (normal). • The bit of the own station baton pass status (SB47) is OFF (normal). • The bit of the own station data link status (SB49) is OFF (data linking).
CC-Link module	When the following conditions are met: <ul style="list-style-type: none"> • The module error (Xn0) is OFF (normal). • 'Module READY' (XnF) is ON (activated). • The bits of the own station data link status (Xn1) is ON (data linking).

Even if the above conditions are not satisfied; however, read/write processing to the own station is completed normally.

■Other station transient access (other station CPU module remote operation and device access)

Create a user program that provides an interlock to enable reading/writing data when the following conditions are satisfied to access devices via each network module.

Module to be routed	Condition that requires interlock
CC-Link IE Controller Network module	When the following conditions are met: <ul style="list-style-type: none">• The bit of the own station baton pass status (SB47) is OFF (normal).• The baton pass status of a station to be accessed (the bit corresponding to the communication target station which is stored from SWA0 to A7) is OFF (normal communication).
CC-Link IE Field Network module	
CC-Link IE TSN module	When the following conditions are met: <ul style="list-style-type: none">• The bit of the own station data link error status (SB0049) is OFF (data linking).• The bit corresponding to the communication target station that is stored in 'Data link status of each station' (SW00B0 to SW00B7) is OFF (normal communication).
MELSECNET/H network module	When the following conditions are met: <ul style="list-style-type: none">• The condition that turns the interlock ON is met when accessing devices on the own station.• The baton pass status of a station to be accessed (the bit corresponding to the communication target station which is stored from SW70 to 73) is OFF (normal communication).• The data link status (the bit corresponding to the communication target station which is stored from SW74 to 77) is OFF (normal communication).
CC-Link module	When the following conditions are met: <ul style="list-style-type: none">• The condition that turns the interlock ON is met when accessing devices on the own station.• The data link status of a station to be accessed (the bit corresponding to the communication target station which is stored from SW80 to 83) is OFF (normal communication).

Timeout value of MELSEC data link functions

A timeout value of MELSEC data link functions represents a value of one communication processing with an access target. The timeout detection time may be longer than the set timeout value because communication processing is performed multiple times for some functions to use or when initially accessing the access target. In addition, when communication processing is performed multiple times, it may take longer time than the timeout value depending on the response timing of the access target, even if the processing is completed normally.



When initially accessing the access target, the following processes will be performed multiple times regardless of the functions to use. Therefore, communication processing will also be performed multiple times.

- Determining an access target
- Acquiring information for the programming controller CPU of an access target

Accessible range and devices

This section explains the accessible range and accessible devices when using MELSEC data link functions.

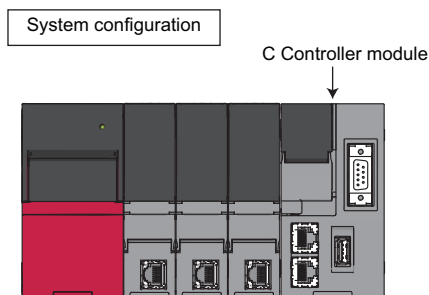
The accessible range of the devices varies depending on specifications and settings of an access target module. For the accessible range, check the access target module specifications.

Bus interface communication

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

■ Accessible range

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



■ 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)
- Bit devices can be accessed by using 'bit set' (mdDevSetEx function) and 'bit reset' (mdDevRstEx function).
- The fixed cycle 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 module.

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU
			R12CCPU-V
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	○
File register	ZR	Batch/random	○
Intelligent function module device, module access device	Un\G	Batch/random	○
CPU buffer memory	U3En\G	Batch	○
		Random	×
Fixed cycle communication area	U3En\HG	Batch	○
		Random	×

Accessing the other CPU

The following table shows the accessible devices when accessing the other CPUs (that is a CPU module and a C Controller module in a multiple CPU system).

No.	Access target CPU
(1)	RCPU
(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	○	×
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	○	×

Device		Access method	Access target CPU	
			(1)	(2)
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 cycle communication area	U3En\HG	Batch	○	○
		Random	×	×
Global label	GV	Batch	×	×
		Random	○	×
Safety input	SA\X	Batch/random	×	×
Safety output	SA\Y	Batch/random	×	×
Safety internal relay	SA\IM	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 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

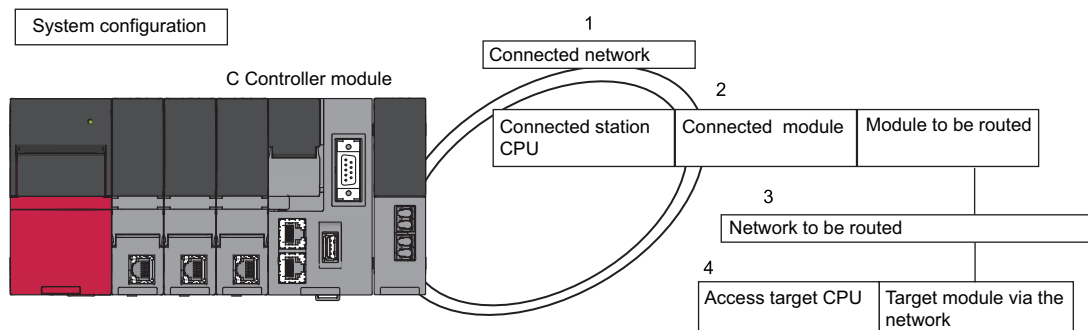
 MELSEC iQ-R C Controller Module User's Manual

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.



■Applicable access

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

○: Accessible, ×: Not accessible

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU						
			Programmable controller			C Controller module		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
CC-Link IE Controller Network	MELSEC iQ- R series programmabl e 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	×	×	×	×	×	×	×
	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		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
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	×	×	×	×	×	×	×
	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 It is not accessible when the connected station CPU is Q00J/Q00/Q01CPU.

■ Accessible devices

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

1

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)
- Bit devices can be accessed by using '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 accessible devices of the CC-Link IE Controller Network module controlled by a C Controller module are shown in the following table.

○: Accessible, ×: Not accessible

Device	Access method	Access target CPU
		R12CCPU-V
RECV function	Batch	○
	Random	×

For details on the replacement from device types specified with an existing product, refer to the following:

☞ Page 236 Replacement of device type

<For other than the RECV function>

To access a CC-Link IE Controller Network module controlled by a C Controller module, use the method explained in the following section. Accessing the own station by using a CC-Link IE Controller Network communication will cause the 'station number/network number error.'

☞ Page 236 Replacement of device type

Accessing other stations

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

No.	Access target CPU
(1)	Basic model QCPU, high performance model QCPU, process CPU, redundant CPU, universal model QCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS
(3)	Interface board for a personal computer
(4)	L02CPU, L26CPU-BT, L02CPU-P, L26CPU-PBT, LJ72GF15-T2, NZ2GF-ETB, L02SCPU, L26CPU, and L06CPU
(5)	RCPU
(6)	R12CCPU-V

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	×	×	○	○	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	○ ^{*1}	×	○	○	○
Link register	W	Batch/random	○	○ ^{*1}	×	○	○	○
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	×	×	×	×	○	×
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) ^{*3}	—	Batch	○	○	○	○	○	○
		Random	×	×	×	×	×	×
SEND function (without arrival confirmation) ^{*3}	—	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)	U3E0\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.2 area)	U3E1\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.3 area)	U3E2\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.4 area)	U3E3\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
Fixed cycle communication area (CPU No.1 area)	U3E0\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.2 area)	U3E1\HG	Batch	×	×	×	×	○	○
		Random					×	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
Fixed cycle communication area (CPU No.3 area)	U3E2\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle 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	GV	Batch	×	×	×	×	×	×
		Random	×	×	×	×	○	×
Safety input	SA\X	Batch/random	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×
Safety internal relay	SA\IM	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 (Extended mode)

Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*2 Q00JCPU is not accessible.

*3 This is a function to send a message to a network module on other stations via a CC-Link IE Controller Network module. Access to a multiple CPU system (when a logical station number is specified) is not available.

*4 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

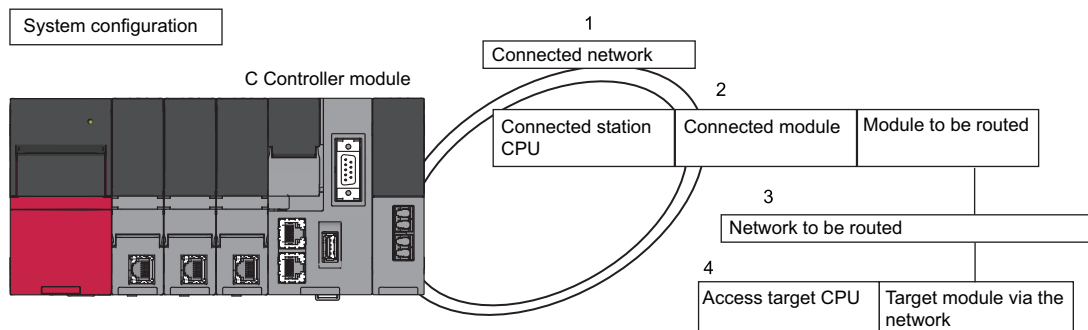
📖 MELSEC iQ-R C Controller Module User's Manual

CC-Link IE Field Network communication

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

■Accessible range

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



■Applicable access

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

○: Accessible, ×: Not accessible

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU						
			Programmable controller			C Controller module		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
CC-Link IE Field Network*1	MELSEC iQ- R series programmabl e controller	CC-Link IE Controller Network	○	○	×	○	○	×	×
		CC-Link IE Field Network	○	○	○	○	○*2	×	×
		CC-Link IE TSN	○	×	×	○	×	×	○
		MELSECNET/H network	○	○	×	○	○	×	×
		MELSECNET/10 network	○	○	×	○	○	×	×
		Ethernet	×	×	○	×	×	×	×
		Serial communication	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×
	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		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
CC-Link IE Field Network* ¹	MELSEC-Q series programmabl e controller (Q mode)	CC-Link IE Controller Network* ³	○	○* ⁴	×	○	○	×	×
		CC-Link IE Field Network* ³	○	○* ⁴	○	○	○* ²	×	×
		MELSECNET/H network	○	○* ⁴	×	○	○	×	×
		MELSECNET/10 network	○	○* ⁴	×	○	○	×	×
		MELSECNET(II)	×	×	×	×	×	×	×
		Ethernet	×	×	○	×	×	×	×
		Serial communication	×	×	×	×	×	×	×
		CC-Link	×	×	×	×	×	×	×
	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 A simple motion module (RD77GF4, RD77GF8, and RD77GF16) is not accessible.

*² The following CPUs are accessible:

Q12DCCPU-V (Extended mode)

Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

*³ The station number 65 or later is accessible only when all control CPUs on the network to be routed are universal model QCPUs.

*⁴ It is not accessible when the connected station CPU is Q00J/Q00/Q01CPU.

■ Accessible devices

The devices accessible for communication via a CC-Link IE Field Network master/local module are shown below.



- '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)
- Bit devices can be accessed by using '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 accessible devices of the CC-Link IE Field Network module controlled by a C Controller module are shown in the following table.

○: Accessible, ×: Not accessible

Device	Access method	Access target CPU
		R12CCPU-V
RECV function	Batch	○
	Random	×

For details on the replacement from device types specified with an existing product, refer to the following:

☞ Page 236 Replacement of device type

<For other than the RECV function>

To access a CC-Link IE Field Network module controlled by a C Controller module, use the method explained in the following section. Accessing the own station by using CC-Link IE Field Network communication will cause the 'station number/network number error.'

☞ Page 236 Replacement of device type

Accessing other stations

The accessible devices of the CC-Link IE Field Network module on the other station are shown in the following table.

No.	Access target CPU
(1)	Basic model QCPU, high performance model QCPU, process CPU, redundant CPU, universal model QCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS
(3)	Interface board for a personal computer
(4)	L26CPU-BT, L02CPU, L02CPU-P, L26CPU-PBT, LJ72GF15-T2, NZ2GF-ETB, L02SCPU, L26CPU, and L06CPU
(5)	RCPU
(6)	R12CCPU-V

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	×	×	○	○	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	○ ^{*1}	×	○	○	○
Link register	W	Batch/random	○	○ ^{*1}	×	○	○	○
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	×	×	×	×	○	×
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)	—	Batch	○	○ ^{*1}	○	○	○	○
		Random	×	×	×	×	×	×
SEND function (without arrival confirmation)	—	Batch	○	○ ^{*1}	○	○	○	○
		Random	×	×	×	×	×	×
Link direct device (link input) ^{*4}	Jn\X	Batch/random	○	○	×	×	○	○
Link direct device (link output) ^{*4}	Jn\Y	Batch/random	○	○	×	×	○	○
Link direct device (link relay) ^{*4}	Jn\B	Batch/random	○	○ ^{*3}	×	×	○	○
Link direct device (link register) ^{*4}	Jn\W	Batch/random	○	○ ^{*3}	×	×	○	○
Link direct device (link special relay) ^{*4}	Jn\SB	Batch/random	○	○ ^{*3}	×	○	○	○
Link direct device (link special register) ^{*4}	Jn\SW	Batch/random	○	○ ^{*3}	×	○	○	○
Intelligent function module device, module access device	Un\G	Batch/random	○	○ ^{*3}	×	○	○	○
CPU shared memory, CPU buffer memory (CPU No.1 area)	U3E0\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.2 area)	U3E1\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.3 area)	U3E2\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.4 area)	U3E3\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
Fixed cycle communication area (CPU No.1 area)	U3E0\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.2 area)	U3E1\HG	Batch	×	×	×	×	○	○
		Random					×	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
Fixed cycle communication area (CPU No.3 area)	U3E2\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle 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	GV	Batch	×	×	×	×	×	×
		Random	×	×	×	×	○	×
Safety input	SA\X	Batch/random	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×
Safety internal relay	SA\I	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 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

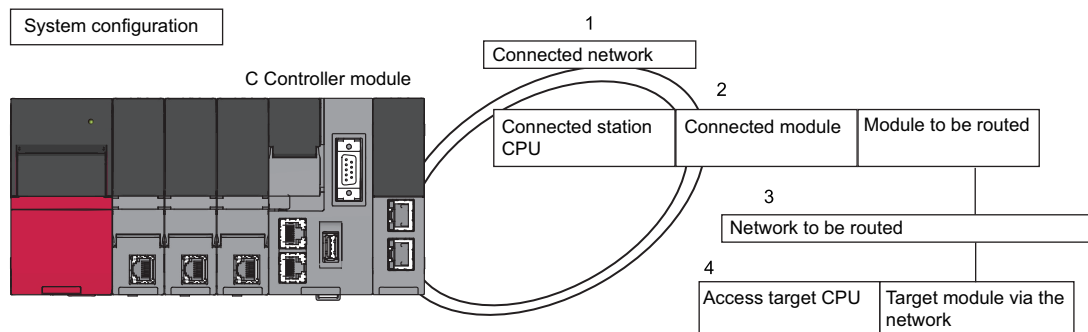
📖 MELSEC iQ-R C Controller Module User's Manual

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.



■ Applicable access

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

○: Accessible, ×: Not accessible

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU						
			Programmable controller			C Controller module		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
CC-Link IE TSN	MELSEC iQ- R series programmabl e 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	×	×	×	×	×	×	×
	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		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
CC-Link IE TSN	Interface board for a personal computer	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 The following CPUs are accessible:
Q12DCCPU-V (Extended mode)
Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS

■ Accessible devices

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

Accessing the own station

The accessible devices of the CC-Link IE TSN module controlled by a C Controller module are shown in the following table.

○: Accessible, ×: Not accessible

Device	Access method	Access target CPU
		R12CCPU-V
RECV function	Batch	○
	Random	×

<For other than the RECV function>

To access a CC-Link IE TSN module controlled by a C Controller module, use the method explained in the following section.

Accessing the own station by using CC-Link IE TSN communication will cause the 'station number/network number error.'

📖 Page 236 Replacement of device type

Accessing other stations

The accessible devices of the CC-Link IE TSN module on the other station are shown in the following table.

No.	Access target CPU
(1)	Basic model QCPU, high performance model QCPU, process CPU, redundant CPU, universal model QCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS
(3)	Interface board for a personal computer
(4)	L26CPU-BT, L02CPU, L02CPU-P, L26CPU-PBT, LJ72GF15-T2, NZ2GF-ETB, L02SCPU, L26CPU, and L06CPU
(5)	RCPU
(6)	R12CCPU-V

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	○*1	×	○	○	○
Link register	W	Batch/random	○	○*1	×	○	○	○

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	×	×	×	×	○	×
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)	—	Batch	○	○*1	○	○	○	○
		Random	×	×	×	×	×	×
SEND function (without arrival confirmation)	—	Batch	○	○*1	○	○	○	○
		Random	×	×	×	×	×	×
Link direct device (link input)*4	Jn\X	Batch/random	○	○	×	×	○	○
Link direct device (link output)*4	Jn\Y	Batch/random	○	○	×	×	○	○
Link direct device (link relay)*4	Jn\B	Batch/random	○	○*3	×	×	○	○
Link direct device (link register)*4	Jn\W	Batch/random	○	○*3	×	×	○	○
Link direct device (link special relay)*4	Jn\SB	Batch/random	○	○*3	×	○	○	○
Link direct device (link special register)*4	Jn\SW	Batch/random	○	○*3	×	○	○	○
Intelligent function module device, module access device	Un\G	Batch/random	○	○*3	×	○	○	○
CPU shared memory, CPU buffer memory (CPU No.1 area)	U3E0\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.2 area)	U3E1\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.3 area)	U3E2\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
CPU shared memory, CPU buffer memory (CPU No.4 area)	U3E3\G	Batch	○	○	×	×	○	○
		Random	×	×			×	×
Fixed cycle communication area (CPU No.1 area)	U3E0\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.2 area)	U3E1\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.3 area)	U3E2\HG	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle 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	GV	Batch	×	×	×	×	×	×
		Random					○	
Safety input	SA\X	Batch/random	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
Safety internal relay	SA\IM	Batch/random	×	×	×	×	×	×
Safety link relay	SA\IB	Batch/random	×	×	×	×	×	×
Safety timer	SA\T	Batch/random	×	×	×	×	×	×
Safety retentive timer	SA\ST	Batch/random	×	×	×	×	×	×
Safety counter	SA\IC	Batch/random	×	×	×	×	×	×
Safety data register	SA\ID	Batch/random	×	×	×	×	×	×
Safety link register	SA\IW	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 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

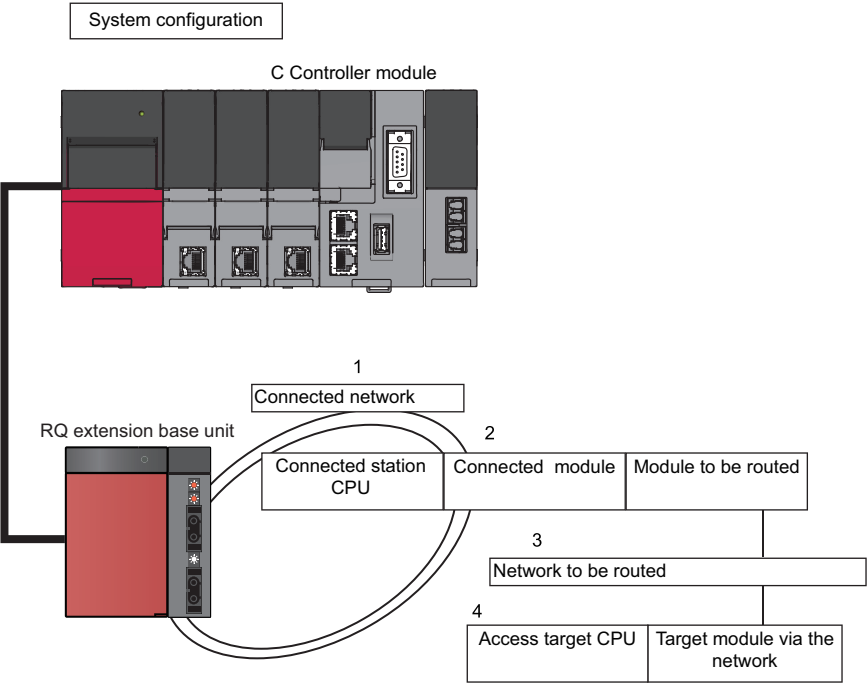
 MELSEC iQ-R C Controller Module User's Manual

MELSECNET/H network communication

The following explains the range and devices accessible for communication via a MELSECNET/H network module.

■Accessible range

The system configuration in the accessible range and the accessibility of each access target CPU via a MELSECNET/H network module are shown below.



■Applicable access

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

○: Accessible, ×: Not accessible

1. Connected network	2. Connected station CPU	3. Network to be routed	4. Access target CPU						
			Programmable controller			C Controller module		MELSEC WinCPU module	Interface board for a personal computer
			MELSEC iQ-R series	MELSEC -Q series	MELSEC -L series	MELSEC iQ-R series	MELSEC -Q series	MELSEC -Q series	
• MELSECNET/H network • MELSECNET/10 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	×	×	×	×	×	×	×
	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	×	×	×	×	×	×	×
• MELSECNET/H network • MELSECNET/10 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	×	×	×	×	×	×	×
	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 It is not accessible when the connected station CPU is Q00J/Q00/Q01CPU.

■ Accessible devices

The devices accessible for communication via a MELSECNET/H network 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)
- Bit devices can be accessed by using '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 accessible devices of the MELSECNET/H network module controlled by C Controller module are shown in the following table.

○: Accessible, ×: Not accessible

Device	Access method	Access target CPU
		R12CCPU-V
RECV function	Batch	○
	Random	×

For details on the replacement from device types specified with an existing product, refer to the following:

☞ Page 236 Replacement of device type

<For other than the RECV function>

To access a MELSECNET/H network module controlled by a C Controller module, use the methods explained in the following section. Accessing the own station by using MELSECNET/H network communication will cause the 'station number/network number error.'

☞ Page 236 Replacement of device type

Accessing other stations

The accessible devices of the MELSECNET/H network module on the other station are shown in the following table.

No.	Access target CPU
(1)	Basic model QCPU, high performance model QCPU, process CPU, redundant CPU, universal model QCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS
(3)	Interface board for a personal computer
(4)	L26CPU-BT, L02CPU, L02CPU-P, L26CPU-PBT, LJ72GF15-T2, NZ2GF-ETB, L02SCPU, L26CPU, and L06CPU
(5)	RCPU
(6)	R12CCPU-V

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	×	×	○	○	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	×	×	○	○	×
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	○	○ ^{*4}	×	×	○	×
Link direct device (link output) ^{*5}	Jn\Y	Batch/random	○	○ ^{*4}	×	×	○	×
Link direct device (link relay) ^{*5}	Jn\B	Batch/random	○	○ ^{*4}	×	×	○	×
Link direct device (link register) ^{*5}	Jn\W	Batch/random	○	○ ^{*4}	×	×	○	×
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	○	○	×	×	○	×
		Random	×	×			×	
CPU shared memory, CPU buffer memory (CPU No.2 area)	U3E1\G	Batch	○	○	×	×	○	×
		Random	×	×			×	
CPU shared memory, CPU buffer memory (CPU No.3 area)	U3E2\G	Batch	○	○	×	×	○	×
		Random	×	×			×	
CPU shared memory, CPU buffer memory (CPU No.4 area)	U3E3\G	Batch	○	○	×	×	○	×
		Random	×	×			×	
Fixed cycle communication area (CPU No.1 area)	U3E0\HG	Batch	×	×	×	×	○	×
		Random					×	
Fixed cycle communication area (CPU No.2 area)	U3E1\HG	Batch	×	×	×	×	○	×
		Random					×	

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
Fixed cycle communication area (CPU No.3 area)	U3E2\HG	Batch	×	×	×	×	○	×
		Random					×	
Fixed cycle 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	GV	Batch	×	×	×	×	×	×
		Random	×	×	×	×	×	×
Safety input	SA\X	Batch/random	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×
Safety internal relay	SA\I	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 A message is sent to a network module on the other station via a MELSECNET/H network module.

Access to a multiple CPU system (when a logical station number is specified) is not available.

*5 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

 MELSEC iQ-R C Controller Module User's Manual

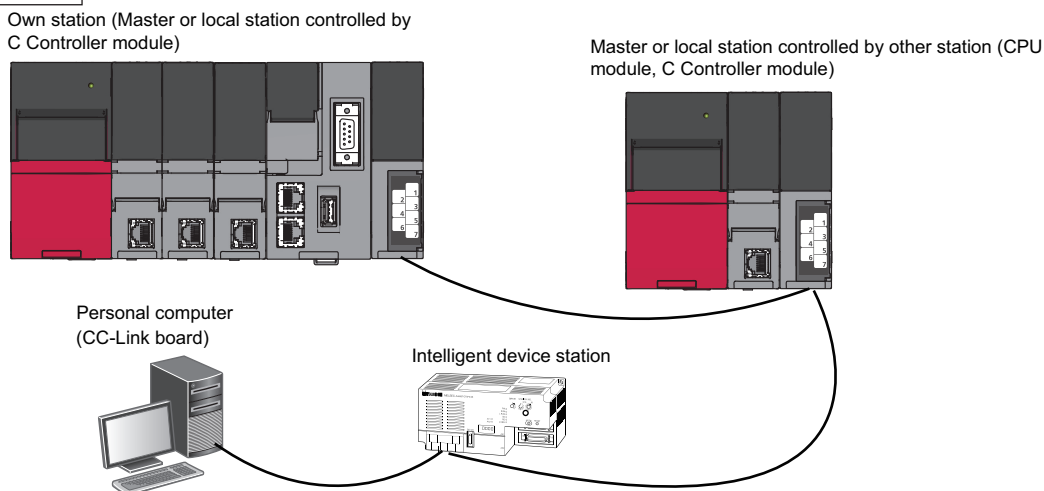
CC-Link communication

The following explains the accessible range and devices 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 C Controller module), the master station or local stations controlled by other stations (CPU module and C Controller module), an intelligent device station, and a personal computer on which a CC-Link board is installed.

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)
- Bit devices can be accessed by using '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 C Controller 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 236 Replacement of device type

Accessing other stations

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

No.	Access target CPU
(1)	Basic model QCPU, high performance model QCPU, process CPU, redundant CPU, universal model QCPU
(2)	Q12DCCPU-V, Q24DHCCPU-V, Q24DHCCPU-LS, Q24DHCCPU-VG, and Q26DHCCPU-LS
(3)	Personal computer and intelligent device station
(4)	L02CPU, L26CPU-BT, L02CPU-P, L26CPU-PBT, L02SCPU, L26CPU, and L06CPU
(5)	RCPU
(6)	R12CCPU-V

○: Accessible, ×: Not accessible

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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	○	×	×	○	○	×
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	×	×	×	×	×	×

Device		Access method	Access target CPU					
			(1)	(2)	(3)	(4)	(5)	(6)
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 cycle communication area (CPU No.1 area)	—	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.2 area)	—	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.3 area)	—	Batch	×	×	×	×	○	○
		Random					×	×
Fixed cycle communication area (CPU No.4 area)	—	Batch	×	×	×	×	○	○
		Random					×	×
Global label	GV	Batch	×	×	×	×	×	×
		Random	×	×	×	×	○	×
Safety input	SA\X	Batch/random	×	×	×	×	×	×
Safety output	SA\Y	Batch/random	×	×	×	×	×	×
Safety internal relay	SA\IM	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\IC	Batch/random	×	×	×	×	×	×
Safety data register	SA\ID	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 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 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

📖 MELSEC iQ-R C Controller Module User's Manual

Argument specification

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

Channel

A channel implies a network and communication route to be used when communicating with a C Controller 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 number	Network	Communication route
12	Bus interface	Used for communication via a bus.
151 to 158	CC-Link IE Controller Network	Used for communication via a CC-Link IE Controller Network module controlled by a C Controller module.
181 to 188	CC-Link IE Field Network	Used for communication via a CC-Link IE Field Network module controlled by a C Controller module.
281 to 288	CC-Link IE TSN	Used for communication via a CC-Link IE TSN module controlled by a C Controller module.
51 to 54	MELSECNET/H network	Used for communication via a MELSECNET/H network module controlled by a C Controller module.
81 to 88	CC-Link	Used for communication via a CC-Link module controlled by a C Controller 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:

To specify station numbers for the mdControl function and mdTypeRead function, refer to "Network number and station number for MELSEC data link functions (the mdControl function and the mdTypeRead function)".

Communication	Specification method		Network number	Station number
Bus interface	Own station		0 (0H)	255 (FFH)* ³
	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)* ⁷ , 125 (7DH)* ⁷
		Group number 1 to 32* ^{4,*5}		129 (81H) to 160 (A0H)
		All stations* ⁴		240 (F0H)
	Logical station number* ¹		0 (0H)	65 (41H) to 239 (EFH)
CC-Link IE Field Network	Own station		0 (0H)	255 (FFH)
	Other station	Station number	1 (1H) to 239 (EFH)	0 (0H) to 120 (78H), 125 (7DH)* ⁶
		All stations* ⁴		240 (F0H)
	Logical station number* ¹		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)* ⁸ , 125 (7DH)* ⁸
		Group number 1 to 32* ^{4,*5}		129 (81H) to 160 (A0H)
		All stations* ⁴		240 (F0H)
	Logical station number* ¹		0 (0H)	65 (41H) to 239 (EFH)
MELSECNET/H network	Own station		0 (0H)	255 (FFH)
	Other station	Station number	1 (1H) to 239 (EFH)	1 (1H) to 64 (40H) 0 (0H)* ⁷ , 125 (7DH)* ⁷
		Group number 1 to 32* ^{4,*5}		129 (81H) to 160 (A0H)
		All stations* ⁴		240 (F0H)
	Logical station number* ¹		0 (0H)	65 (41H) to 239 (EFH)
CC-Link	Other station		0 (0H)	0 (0H) to 63 (3FH)* ²
	Logical station number* ¹			65 (41H) to 239 (EFH)

*1 Logical station numbers are logical numbers which specifies "station number" in a user program (MELSEC data link functions). Logical station numbers are used to access from an applicable module (channel number) to the other station CPU (other CPU of a multiple CPU system). However, if the access target CPU module is not compatible with a network module on the access route, an error will occur and the applicable module will not be able to access to the CPU module.

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.

Logical station numbers can be set in the target setting of a network module. For parameters of a network module, set them in CW Configurator. (CW Configurator Operating Manual)

· [Navigation window] ⇒ [Parameter] ⇒ [Module Information] ⇒ a target module ⇒ [Application Settings] ⇒ [Target settings]

*2 The station number 64 cannot be specified for CC-Link communication.

*3 Communication to C Controller module (own station) by using the MELSEC data link functions is possible; however, it may take longer to execute the functions compared to the C Controller module dedicated functions. Use the C Controller module dedicated functions to create a user program in which performance should be ensured (such as control program).

*4 The group number and all stations specification is valid when the SEND function (mdSendEX (message send function)) with 'no arrival confirmation' specification is used.

*5 The group number can be specified when using CC-Link IE Controller Network, MELSECNET/H network, and CC-Link IE TSN.

*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. 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).

- *7 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).
- *8 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.

■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 Field Network	Own station: 255 (FFH) Other station: *1,*3,*6
CC-Link IE TSN	Own station: 255 (FFH) Other station: *1,*3,*6
MELSECNET/H network	Own station: 255 (FFH) Other station: *1,*2,*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, CC-Link IE Field Network module, MELSECNET/H 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 any of 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 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 and CC-Link IE Controller Network, the access target is a control station.*2 • For CC-Link IE TSN and CC-Link IE Field Network, the access target is a master station.*3

Method for specifying a logical station number*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).

Logical station numbers can be set in the target setting of a network module. For parameters of a network module, set them in CW Configurator. (CW Configurator Operating Manual)

· [Navigation window] ⇒ [Parameter] ⇒ [Module Information] ⇒ a target module ⇒ [Application Settings] ⇒ [Target settings]

*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, a specified control station of the network, which is specified for the network number, is accessed. For CC-Link IE Field Network, 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).

*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).

Logical station numbers can be set in the target setting of a network module. For parameters of a network module, set them in CW Configurator. (CW Configurator Operating Manual)

· [Navigation window] ⇒ [Parameter] ⇒ [Module Information] ⇒ a target module ⇒ [Application Settings] ⇒ [Target settings]

*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 Logical station numbers are logical numbers which specifies "station number" in a user program (MELSEC data link functions).

Logical station numbers are used to access from an applicable module (channel number) to the other station CPU (other CPU of a multiple CPU system). However, if the access target CPU module is not compatible with a network module on the access route, an error will occur and the applicable module will not be able to access to the CPU module.

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.

Logical station numbers can be set in the target setting of a network module. For parameters of a network module, set them in CW Configurator. (CW Configurator Operating Manual)

· [Navigation window] ⇒ [Parameter] ⇒ [Module Information] ⇒ a target module ⇒ [Application Settings] ⇒ [Target settings]

Device type

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



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 cycle 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 of 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 of 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 is not accessible in CC-Link communication, CC-Link IE Controller Network communication and CC-Link IE Field 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 and mdRandWLabelEx functions can be used.

*5 To directly access link devices, access them as link direct devices (J□\□) depending on network module specifications. An access target device varies depending on a device number to be specified. Therefore, a device number which is actually accessed may be different from the specified device number.

For specification method for accessing link direct devices (J□\□) using MELSEC data link functions, refer to the following:

 MELSEC iQ-R C Controller Module User's Manual

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

The device types shown in the following table can be specified in the user program:

- For sending/receiving 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 Field Network modules

The device types shown in the following table can be specified in the user program:

- For sending/receiving 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 table can be specified in the user program:

- For sending/receiving 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 MELSECNET/H network modules

The device types shown in the following table can be specified in the user program:

- For sending/receiving message

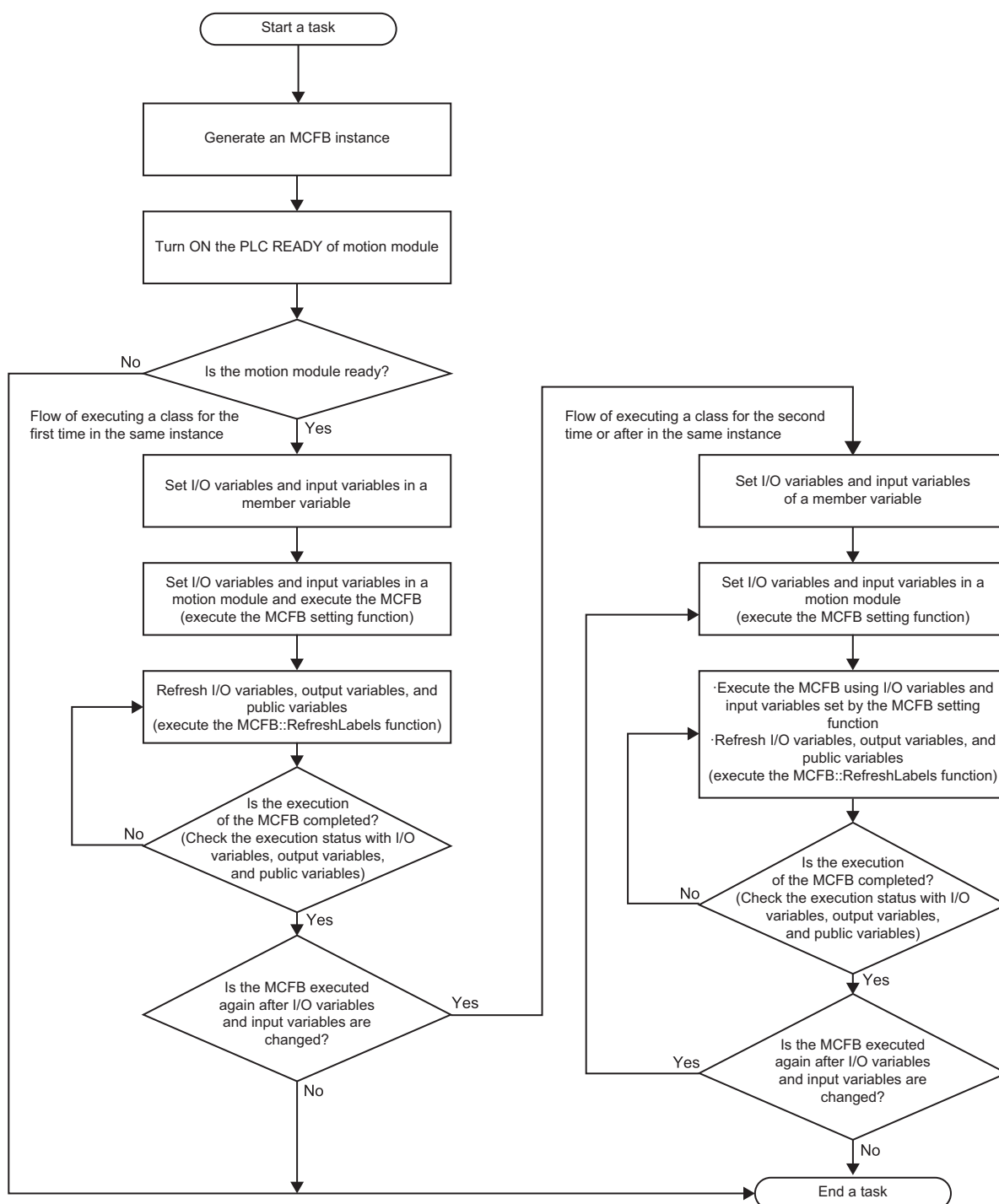
Device	Device type		
	Code		Device name
	Decimal	Hexadecimal	
SEND function (with arrival confirmation) and RECV function	101	65H	DevMAIL
SEND function (without arrival confirmation)	102	66H	DevMAILNC

1.4 Motion Module Dedicated Class

This library is an interface for executing MCFBs of a motion module.
 Before using this library, make sure to fully understand MCFB specifications.
 For specifications of MCFBs, refer to the following:
 MELSEC iQ-R Motion Module User's Manual (Application)

Program processing

The following shows the user program processing using a motion module dedicated class.



A default value is stored in input variables, output variables, and public variables by creating an MCFB instance. Programming can be shortened by setting input variables only for variables required to be changed from the default value.

Execution procedure

The following shows the procedure for executing the Enable-type MCFB (MC_Power) using a motion module dedicated class. The procedure is the same as when executing the Execute-type MCFB. Before using the motion module dedicated class, make sure to fully understand the MCFB specifications which correspond to each instruction class.

■When executing an Enable-type MCFB

Set 'TRUE' for the input variable 'Enable' and then execute the FB. After that, wait for the output variable 'ReadyStatus' to change to 'TRUE.'

Then, set 'FALSE' for the 'Enable' and wait for the 'ReadyStatus' to change to 'FALSE.'

```
void mcPower_Test (void){
    short sRet = 0;
    unsigned short usData = 0;

    /*Generate an MCFB instance*/
    MC_Power mcPower;

    /*Turn ON the PLC READY of motion module*/
    sRet = CCPU_Y_Out_BitEx( 0, 0x0000, TRUE);
    if( sRet != 0 ){
        return;
    }
    taskDelay(10);

    /*Motion module preparation completed*/
    sRet = CCPU_X_In_BitEx( 0, 0x0000, &usData);
    if( (sRet != 0) || (sData == FALSE) ){
        return;
    }

    /*Set I/O variables and input variables of a member variable*/
    mcPower.Axis.AxisNo = 1;
    mcPower.Axis.StartIO = 0x0000;
    mcPower.ServoON = TRUE;

    /*Servo ON*/
    /*Set I/O variables and input variables to the motion module and execute the MCFB*/
    sRet = mcPower.SetEnable(TRUE);
    if( sRet != 0 ){
        return;
    }
    /*Refresh I/O variables, output variables, and public variables*/
    while(1){
        sRet = mcPower.RefreshLabels();
        /*Check that the MCFB processing is completed*/
        if( sRet == 0 ){
            if( mcPower.Error ==FALSE){
                if(mcPower.ReadyStatus == TRUE){
                    break;
                }
            }else{
                break;
            }
        }else{
            return;
        }
    }
}
```

```

        taskDelay(1);
    }

    /*Wait for the motion module side to be refreshed*/
    taskDelay(1);

    /*Servo OFF*/
    /*Set I/O variables and input variables of a member variable*/
    /*No member variables to be set due to the FALSE execution*/

    /*Set I/O variables and input variables to the motion module*/
    sRet = mcPower.SetEnable(FALSE);
    if( sRet != 0 ){
        return;
    }

    /*Execute the MCFB with I/O variables and input variables set in the MCFB setting function*/
    /*Refresh I/O variables, output variables, and public variables*/
    while(1){
        sRet = mcPower.RefreshLabels();

        /*Check that the MCFB processing is completed*/
        if( sRet == 0 ){
            if( mcPower.Error ==FALSE){
                if(mcPower.ReadyStatus == FALSE){
                    break;
                }
            }else{
                break;
            }
        }else{
            return;
        }

        taskDelay(1);
    }
}

```


Classes

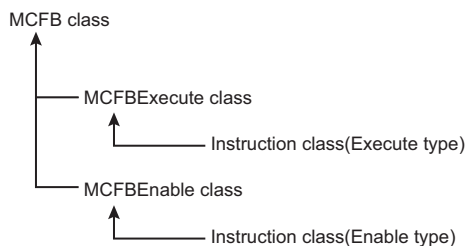
The following explains classes provided in motion module dedicated classes.

Category	Name	Description
MCFB class	MCFB	A class for providing a function to execute an MCFB. The corresponding MCFB version is as follows: • MELSEC iQ-R motion module library version '03D'
Execution class	MCFBExecute	A class for executing an Execute type or Enable type MCFB.
	MCFBEnable	For details on the Execute type and Enable type, refer to the following:  MELSEC iQ-R Motion Module User's Manual (Application)
Instruction class	MC_AbortTrigger	A class for executing an MCFB which has the same name as a class. I/O variables, input variables, output variables, and public variables of MCFBs are defined as a member variable of each corresponding instruction class. When an instruction class instance is created, default values will be stored in the I/O variables, input variables, output variables, and public variables. For details on MCFB specifications and default values to be stored, refer to the following:  MELSEC iQ-R Motion Module User's Manual (Application)
	MC_CamIn	
	MC_CamTableSelect	
	MC_CombineAxes	
	MC_GearIn	
	MC_GroupDisable	
	MC_GroupEnable	
	MC_GroupReset	
	MC_GroupStop	
	MC_Home	
	MC_MoveAbsolute	
	MC_MoveRelative	
	MC_MoveVelocity	
	MC_Reset	
	MC_SetPosition	
	MC_Stop	
	MC_TorqueControl	
	MC_TouchProbe	
	MC_WriteParameter	
	MCv_ChangeCycle	
	MCv_MotionErrorReset	
	MCv_MoveCircularInterpolateAbsolute	
	MCv_MoveCircularInterpolateRelative	
	MCv_MoveLinearInterpolateAbsolute	
	MCv_MoveLinearInterpolateRelative	
	MCv_ReadProfileData	
	MCv_SetTorqueLimit	
	MCv_SpeedControl	
	MCv_SpeedLimitFilter	
	MCv_WriteProfileData	
	MC_GroupSetOverride	
	MC_Power	
	MC_ReadParameter	
	MC_SetOverride	
	MCv_AllPower	
	MCv_BacklashCompensationFilter	
	MCv_DirectionFilter	
	MCv_Jog	
	MCv_SmoothingFilter	

Hierarchical class diagram

Functions and variables defined at the upper class are inherited by the lower class. (Class inheritance relationship)

The following table shows inheritance relationships between classes.



Execution class	Instruction class
MCFBExecute	MC_CamIn
	MC_CombineAxes
	MC_GearIn
	MC_GroupStop
	MC_Home
	MC_MoveAbsolute
	MC_MoveRelative
	MC_MoveVelocity
	MCv_SpeedControl
	MC_Stop
	MC_TorqueControl
	MCv_MoveCircularInterpolateAbsolute
	MCv_MoveCircularInterpolateRelative
	MCv_MoveLinearInterpolateAbsolute
	MCv_MoveLinearInterpolateRelative
	MCv_SpeedLimitFilter
	MC_AbortTrigger
	MC_CamTableSelect
	MC_GroupDisable
	MC_GroupEnable
	MC_GroupReset
	MC_Reset
	MC_SetPosition
	MC_TouchProbe
	MC_WriteParameter
	MCv_ChangeCycle
	MCv_MotionErrorReset
	MCv_SetTorqueLimit
	MCv_ReadProfileData
	MCv_WriteProfileData
MCFBEnable	MCv_BacklashCompensationFilter
	MCv_DirectionFilter
	MCv_Jog
	MCv_SmoothingFilter
	MC_GroupSetOverride
	MC_Power
	MC_ReadParameter
	MC_SetOverride
	MCv_AllPower

Members

The following explains members of each class provided in motion module dedicated classes.

MCFB class

■Function list

Function name	Description
RefreshLabels	To execute an MCFB with I/O variables and input variables set in the MCFB setting function. After the execution, I/O variables, input variables and public variables are refreshed.

■Variable list

None

MCFBExecute class

■Function list

Function name	Description
SetExecute	To set I/O variables and input variables of an Execute-type MCFB in a motion module.

■Variable list

None

MCFBEnable class

■Function list

Function name	Description
SetEnable	To set I/O variables and input variables of an Enable-type MCFB in a motion module.

■Variable list

None


Instruction class

■Function list

Function name	Description
SetEnableJog (For the MCv_Jog class only)	To set I/O variables and input variables of the Enable-type MCv_Jog in a motion module.

■Variable list

For details on instruction class variables, refer to the following function block specifications:

 MELSEC iQ-R Motion Module User's Manual (Application)


Data type

This section shows data types which can be used in this library.

ENUM enumerators

The following table shows I/O variables, input variables, output variables, and public variables which can be used in this library.


For details on ENUM enumerators and structures (axis variables), refer to the following:

 MELSEC iQ-R Motion Module User's Manual (Application)

Type name, enumerator		Setting value	Description
MC_START_MODE			
—	mcImmediate	0	Immediately
	mcAbsolute	1	Absolute
	mcRelative	2	Relative
MC_SOURCE			
—	mcSetValue	1	Set value
	mcActualValue	2	Actual value
	mcLatestSetValue	101	Latest set value
	mcLatestActualValue	102	Latest actual value
MC_BUFFER_MODE			
—	mcAborting	0	Aborting
	mcBuffered	1	Buffered
	mcBlendingLow	2	BlendingLow
	mcBlendingPrevious	3	BlendingPrevious
	mcBlendingNext	4	BlendingNext
	mcBlendingHigh	5	BlendingHigh
MC_COMBINE_MODE			
—	mcAddAxes	0	Add positions of 2 input axes
	mcSubAxes	1	Subtract positions of 2 input axes
MC_DIRECTION			
—	mcPositiveDirection	1	Positive direction
	mcNegativeDirection	2	Negative direction
	mcShortestWay	3	Shortest path
	mcCurrentDirection	4	Current direction
MC_VELOCITY_LIMIT_MODE			
—	Ignore	0	Ignore
	ClampWithRamp	1	Clamp
	Truncate	2	Truncate
	ImmediateStop	3	Immediate stop
	ClampWithoutRamp	4	Clamp (Without Ramp at Deceleration)
MC_CIRC_MODE			
—	mcBorder	0	Border point designation
	mcCenter	1	Center point designation
	mcRadius	2	Radius designation
MC_CIRC_PATHCHOICE			
—	mcCW	0	CW
	mcCCW	1	CCW
	mcShortWay	2	Short way
	mcLongWay	3	Long way
	mcCWLongWay	4	CW long way
	mcCCWLongWay	5	CCW long way
MC_INTERPOLATE_SPEED_MODE			

Type name, enumerator		Setting value	Description
—	VectorSpeed	0	Vector speed
	LongAxisSpeed	1	Long axis speed
	ReferenceAxisSpeed	2	Reference axis speed
MC_EXECUTION_MODE			
—	mcImmediately	0	Execute immediately
	mcQueued	1	Execute at completion
	mcNextExecute	2	Execute at next start
	mcSpeculatively	3	Execute speculatively
MC_RECORD_MODE*1			
—	mcOneShot	0	One shot mode
	mcRecordCount	1	Number of designation times mode
	mcRingBuffer	2	Ring buffer mode

*1 The ENUM enumerator name is different from the one described in the following:

 MELSEC iQ-R Motion Module User's Manual (Application)

Precautions

The following ENUM type is not defined.

- MC_SIGNAL_LOGIC

To access the I/O variables, input variables, output variables, and public variables of ENUM type, specify values according to the type of variable to be used.

■Corresponding classes

The following table shows classes provided in this library and their corresponding ENUM enumerators.

Corresponding class	Type name	Enumerator
MC_CamIn	MC_START_MODE	<ul style="list-style-type: none"> • mcImmediate(0) • mcAbsolute(1) • mcRelative(2)
<ul style="list-style-type: none"> • MC_CamIn • MC_CombineAxes • MC_GearIn • MCv_DirectionFilter • MCv_SmoothingFilter • MCv_SpeedLimitFilter 	MC_SOURCE	<ul style="list-style-type: none"> • mcSetValue(1) • mcActualValue(2) • mcLatestSetValue(101) • mcLatestActualValue(102)
MCv_BacklashCompensationFilter		<ul style="list-style-type: none"> • mcSetValue(1) • mcLatestSetValue(101)
<ul style="list-style-type: none"> • MC_CamIn • MC_MoveAbsolute • MC_MoveRelative • MCv_MoveCircularInterpolateAbsolute • MCv_MoveCircularInterpolateRelative • MCv_MoveLinearInterpolateAbsolute • MCv_MoveLinearInterpolateRelative 	MC_BUFFER_MODE	<ul style="list-style-type: none"> • mcAborting(0) • mcBuffered(1) • mcBlendingLow(2) • mcBlendingPrevious(3) • mcBlendingNext(4) • mcBlendingHigh(5)
<ul style="list-style-type: none"> • MC_CombineAxes • MC_GearIn • MC_MoveVelocity • MC_TorqueControl 		<ul style="list-style-type: none"> • mcAborting(0) • mcBuffered(1)
MCv_SpeedControl		<ul style="list-style-type: none"> • mcAborting(0) • mcBuffered(1) • mcBlendingLow(2)
MC_CombineAxes	MC_COMBINE_MODE	<ul style="list-style-type: none"> • mcAddAxes(0) • mcSubAxes(1)
<ul style="list-style-type: none"> • MC_MoveAbsolute • MCv_MoveLinearInterpolateAbsolute 	MC_DIRECTION	<ul style="list-style-type: none"> • mcPositiveDirection(1) • mcNegativeDirection(2) • mcShortestWay(3)
<ul style="list-style-type: none"> • MC_MoveVelocity • MCv_SpeedControl • MC_TorqueControl • MCv_BacklashCompensationFilter 		<ul style="list-style-type: none"> • mcPositiveDirection(1) • mcNegativeDirection(2)
<ul style="list-style-type: none"> • MCv_DirectionFilter • MCv_SpeedLimitFilter 	MC_VELOCITY_LIMIT_MODE	<ul style="list-style-type: none"> • Ignore(0) • ClampWithRamp(1) • Truncate(2) • ImmediateStop(3) • ClampWithoutRamp(4)
<ul style="list-style-type: none"> • MCv_MoveCircularInterpolateAbsolute • MCv_MoveCircularInterpolateRelative 	MC_CIRC_MODE	<ul style="list-style-type: none"> • mcBorder(0) • mcCenter(1) • mcRadius(2)
<ul style="list-style-type: none"> • MCv_MoveCircularInterpolateAbsolute • MCv_MoveCircularInterpolateRelative 	MC_CIRC_PATHCHOICE	<ul style="list-style-type: none"> • mcCW(0) • mcCCW(1) • mcShortWay(2) • mcLongWay(3) • mcCWLongWay(4) • mcCCWLongWay(5)
<ul style="list-style-type: none"> • MCv_MoveLinearInterpolateAbsolute • MCv_MoveLinearInterpolateRelative 	MC_INTERPOLATE_SPEED_MODE	<ul style="list-style-type: none"> • VectorSpeed(0) • LongAxisSpeed(1) • ReferenceAxisSpeed(2)
<ul style="list-style-type: none"> • MC_CamTableSelect • MCv_ChangeCycle • MCv_SetTorqueLimit • MCv_WriteProfileData 	MC_EXECUTION_MODE	<ul style="list-style-type: none"> • mcImmediately (0) • mcQueued(1) • mcSpeculatively(3)
MC_SetPosition		<ul style="list-style-type: none"> • mcQueued(1) • mcSpeculatively(3)
MC_WriteParameter		<ul style="list-style-type: none"> • mcImmediately (0) • mcQueued(1)
MC_TouchProbe	MC_RECORD_MODE	<ul style="list-style-type: none"> • OneShot(0) • RecordCount(1) • RingBuffer(2)

Structures

The following table shows structures can be used in this library.

For details on the other structures, refer to the following:

 MELSEC iQ-R Motion Module User's Manual (Application)

Type name, enumerator	Description	Type
AXIS_REF		
—	AxisNo	Axis No.
	StartIO	I/O No. (The first 3 digits of the 4 digits written in hexadecimal)
AXES_GROUP_REF		
—	GroupNo	Axes group No.
	StartIO	I/O No. (The first 3 digits of the 4 digits written in hexadecimal)
INSTANCE_ID		
—	StartIO	IO No.
	Number	Instance ID
TARGET_REF		
—	StartIO	IO No.
	Target	Target
SIGNAL_SELECT		
—	Source	Signal
	Detection	Signal detection method
	CompensationTime	Compensation time
	FilterTime	Filter time
MC_TRIGGER_REF		
—	Signal	Trigger signal
MC_INPUT_REF		
—	Signal	Input signal
FILE_LOCATION		
—	FileName	File name
	Path	Folder specification
PROFILE_DATA		
—	Location	Calculation profile data storage location
	ID	Profile ID
MC_CAM_REF		
—	ProfileData	Profile

Restriction

This library defines structures which are used for I/O variables, input variables, output variables, and public variables of each class with the corresponding MCFB version. Therefore, when these variables are added or changed in the corresponding version or later, the added or changed variables cannot be used.

The corresponding MCFB version is as follows:

- MELSEC iQ-R motion module library version '03D'

Precautions

The following structures are not defined as structures in header files.

- MC_CAM_ID
- PROFILE_ID

To access I/O variables, input variables, output variables, and public variables of the structures, access them as the 'unsigned short' type.



Considerations

Initial execution of the MCFB setting function.

A C Controller module creates an internal instance for sending/receiving data to/from a motion module at the initial execution of the MCFB setting function. Since the C Controller module communicates with the motion module when creating the instance, it may take longer time for the function to be executed. Therefore, the initial execution time for the function becomes longer. Ensure to execute the MCFB setting function once by executing the dummy MCFB setting function (by setting 'false' for the argument) in each MCFB class.

Considerations for executing MCFBs

Note the following when initially executing the MCFB setting function or executing an MCFB with the MCFB::RefreshLabels function.

- Ensure to execute the MCFB with the MCFB::RefreshLabels function after setting I/O variables and input variables by the MCFB setting function.
- Before executing the MCFB, check that the 'ready' signal of motion module is turned ON. The MCFB will not be run if the signal does not turn ON.
The 'motion module not-ready error' occurs at the initial execution of the MCFB setting function. However, the error will be detected only at the initial execution of the setting function.
- Check that the MCFB which corresponds to each instruction class is in the executable status and execute it.
For the executable status of MCFBs, check specifications of each MCFB. ( MELSEC iQ-R Motion Module User's Manual (Application))
- Depending on motion module specifications, the MCFB::RefreshLabels function needs to be executed for multiple times to start processing on the motion module side. After executing an MCFB, make sure to check values of I/O variables, output variables, and public variables by executing the MCFB::RefreshLabels function repeatedly until the processing of the MCFB is completed.
For the completion status of MCFBs, check specifications of each MCFB. ( MELSEC iQ-R Motion Module User's Manual (Application))

Processing at refresh

Check that the 'ready' signal of the motion module is turned ON before executing the function. The function will be completed normally even if the 'ready' signal does not turn ON. However, I/O variables, output variables, and public variables will not be updated because the processing of the motion module side does not operate.

1.5 Considerations on Interrupt Service Routine (ISR)

1

Fully understand the restrictions of VxWorks, operating system, before creating a routine which is executed in an interrupt service routine (ISR: InterruptServiceRoutine). To use another dedicated function by synchronizing it to an interrupt, implement the notification processing in a user program and perform it in a task.

Point

Setting an inappropriate value or executing a function other than a C Controller module dedicated function for ISR from an interrupt service routine may cause the VxWorks runaway.

2 FUNCTION LIST

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

2.1 C Controller Module Dedicated Functions

The C Controller module dedicated functions are as listed below.

C Controller module dedicated functions

Function name	Function	Reference
CCPU_ChangeCCIEFBCycPrm	To change the operation parameter of the cyclic transmission of the CC-Link IE Field Network Basic function.	Page 69 CCPU_ChangeCCIEFBCycPrm
CCPU_ChangeFileSecurity	To change the file access restriction status of a C Controller module.	Page 70 CCPU_ChangeFileSecurity
CCPU_ClearError	To clear errors of a C Controller module.	Page 71 CCPU_ClearError
CCPU_Control	To perform remote operations (remote RUN/STOP/PAUSE) for the CPU module.	Page 72 CCPU_Control
CCPU_DedicatedDInst	To execute dedicated instructions categorized as 'D' or 'DP'.	Page 73 CCPU_DedicatedDInst
CCPU_DedicatedGInst	To execute dedicated instructions categorized as 'G' or 'GP'.	Page 75 CCPU_DedicatedGInst
CCPU_DedicatedJInst	To execute dedicated instructions categorized as 'J' or 'JP'.	Page 77 CCPU_DedicatedJInst
CCPU_DedicatedMInst	To execute dedicated instructions categorized as 'M' or 'MP'.	Page 79 CCPU_DedicatedMInst
CCPU_DisableInt	To disable the routine registered with the CCPU_EntryInt function.	Page 82 CCPU_DisableInt
CCPU_EnableInt	To enable the routine registered with the CCPU_EntryInt function.	Page 83 CCPU_EnableInt
CCPU_EndCCIEFBDataAssurance	To end data assurance for one link scan of CC-Link IE Field Network Basic.	Page 84 CCPU_EndCCIEFBDataAssurance
CCPU_EntryCCIEFBRefEndFunc	To register a routine to be called when the link scan of CC-Link IE Field Network Basic is completed.	Page 85 CCPU_EntryCCIEFBRefEndFunc
CCPU_EntryInt	To register a routine to be called when an interrupt occurs.	Page 86 CCPU_EntryInt
CCPU_EntryTimerEvent	To register a timer event.	Page 88 CCPU_EntryTimerEvent
CCPU_EntryWDTInt	To register a routine to be called when a user WDT error interrupt occurs.	Page 90 CCPU_EntryWDTInt
CCPU_FromBuf	To read data from the CPU buffer memory of the CPU module and the buffer memory of the intelligent function module which are mounted on the specified module position. (FROM instruction)	Page 91 CCPU_FromBuf
CCPU_FromBufHG	To read data from the fixed cycle communication area of the CPU module mounted on the specified module position.	Page 92 CCPU_FromBufHG
CCPU_GetCCIEFBDiagnosticInfo	To acquire the diagnostic information of CC-Link IE Field Network Basic.	Page 93 CCPU_GetCCIEFBDiagnosticInfo
CCPU_GetConstantProcessStatus	To acquire the fixed cycle processing status of a C Controller module.	Page 95 CCPU_GetConstantProcessStatus
CCPU_GetCounterMicros	To acquire a 1 μ s counter value of a C Controller module.	Page 96 CCPU_GetCounterMicros
CCPU_GetCounterMillis	To acquire a 1 ms counter value of a C Controller module.	Page 97 CCPU_GetCounterMillis
CCPU_GetCpuStatus	To acquire the operating status of a C Controller module.	Page 98 CCPU_GetCpuStatus
CCPU_GetDotMatrixLED	To acquire the value displayed on the dot matrix LED of a C Controller module.	Page 100 CCPU_GetDotMatrixLED
CCPU_GetErrInfo	To acquire the error information of a C Controller module.	Page 102 CCPU_GetErrInfo
CCPU_GetFileSecurity	To acquire the file access mode of a C Controller module.	Page 103 CCPU_GetFileSecurity
CCPU_GetIDInfo	To acquire the individual identification information of a C Controller module.	Page 104 CCPU_GetIDInfo
CCPU_GetLEDStatus	To acquire the LED status of a C Controller module.	Page 105 CCPU_GetLEDStatus
CCPU_GetOpSelectMode	To acquire the operation selection mode of a C Controller module.	Page 107 CCPU_GetOpSelectMode
CCPU_GetPowerStatus	To acquire the power status of a C Controller module.	Page 108 CCPU_GetPowerStatus
CCPU_GetRTC	To acquire the clock data (local time) of a C Controller module.	Page 109 CCPU_GetRTC
CCPU_GetSerialNo	To acquire the serial number of a C Controller module.	Page 110 CCPU_GetSerialNo
CCPU_GetSwitchStatus	To acquire the switch status of a C Controller module.	Page 111 CCPU_GetSwitchStatus
CCPU_GetUnitInfo	To acquire the module configuration information.	Page 112 CCPU_GetUnitInfo
CCPU_LockFWUpdate	To prohibit the execution of the firmware update of a C Controller module.	Page 115 CCPU_LockFWUpdate
CCPU_MountMemoryCard	To mount the SD memory card inserted to a C Controller module.	Page 116 CCPU_MountMemoryCard

Function name	Function	Reference
CCPU_ReadDevice	To read data from internal user devices and internal system devices of a C Controller module.	Page 117 CCPU_ReadDevice
CCPU_ReadLinkDevice	To read data from the own station link devices of CC-Link IE Controller Network module, CC-Link IE Field Network module, Ethernet module (when CC-Link IE Field Network is selected), MELSECNET/H network module, and CC-Link IE TSN module.	Page 118 CCPU_ReadLinkDevice
CCPU_ReadMCUnitLabel	To read data from module labels of a C Controller module in word units.	Page 119 CCPU_ReadMCUnitLabel
CCPU_ReadMCUnitLabelBit	To read data from the module labels of a C Controller module in bit units.	Page 121 CCPU_ReadMCUnitLabelBit
CCPU_RegistEventLog	To register an event log in the event history of a C Controller module.	Page 122 CCPU_RegistEventLog
CCPU_Reset	To reset the bus master CPU (CPU No.1).	Page 123 CCPU_Reset
CCPU_ResetDevice	To reset internal user devices and internal system devices (bit devices) of C Controller module.	Page 124 CCPU_ResetDevice
CCPU_ResetWDT	To reset the user WDT of a C Controller module.	Page 125 CCPU_ResetWDT
CCPU_RestoreDefaultCCIEFBCycPrm	To restore the operation parameter of cyclic transmission of CC-Link IE Field Network Basic to the default value (which is set in the parameter).	Page 126 CCPU_RestoreDefaultCCIEFBCycPrm
CCPU_SetDevice	To set internal user devices and internal system devices (bit devices) of C Controller module.	Page 127 CCPU_SetDevice
CCPU_SetDotMatrixLED	To set a value to be displayed on the dot matrix LED of C Controller module.	Page 128 CCPU_SetDotMatrixLED
CCPU_SetLEDStatus	To set the LED status of a C Controller module.	Page 130 CCPU_SetLEDStatus
CCPU_SetOpSelectMode	To set the operation selection mode of C Controller module.	Page 131 CCPU_SetOpSelectMode
CCPU_SetRTC	To set the clock data (local time) of C Controller module.	Page 132 CCPU_SetRTC
CCPU_ShutdownRom	To shut down the program memory and the data memory of a C Controller module.	Page 133 CCPU_ShutdownRom
CCPU_StartCCIEFBDataAssurance	To start data assurance for one link scan of CC-Link IE Field Network Basic.	Page 134 CCPU_StartCCIEFBDataAssurance
CCPU_StartWDT	To set and start the user WDT of a C Controller module.	Page 135 CCPU_StartWDT
CCPU_StopWDT	To stop the user WDT of a C Controller module.	Page 136 CCPU_StopWDT
CCPU_SysClkRateGet	To read the system clock rate specified with the CCPU_SysClkRateSet function from the backup RAM.	Page 137 CCPU_SysClkRateGet
CCPU_SysClkRateSet	To store the specified system clock rate in the backup RAM.	Page 138 CCPU_SysClkRateSet
CCPU_ToBuf	To write data to the CPU buffer memory of the CPU module (host CPU) and the buffer memory of the intelligent function module which are mounted on the specified module position. (TO instruction)	Page 139 CCPU_ToBuf
CCPU_ToBufHG	To write data to the fixed cycle communication area of the CPU module mounted on the specified module position.	Page 140 CCPU_ToBufHG
CCPU_UnlockFWUpdate	To remove the prohibition on the firmware update of a C Controller module.	Page 141 CCPU_UnlockFWUpdate
CCPU_UnmountMemoryCard	To unmount the SD memory card and USB Mass Storage Class-compliant device connected to a C Controller module.	Page 142 CCPU_UnmountMemoryCard
CCPU_WaitEvent	To wait for an interrupt event notification from other CPUs.	Page 143 CCPU_WaitEvent
CCPU_WaitSwitchEvent	To wait for a switch interrupt event of C Controller module to occur.	Page 145 CCPU_WaitSwitchEvent
CCPU_WaitTimerEvent	To wait for a timer event to occur.	Page 146 CCPU_WaitTimerEvent
CCPU_WaitUnitEvent	To wait for an interrupt event notification from modules.	Page 147 CCPU_WaitUnitEvent
CCPU_WriteDevice	To write data to internal user devices and internal system devices of C Controller module.	Page 149 CCPU_WriteDevice
CCPU_WriteLinkDevice	To write data to own station link devices of CC-Link IE Controller Network module, CC-Link IE Field Network module, Ethernet module (when CC-Link IE Field Network is selected), MELSECNET/H network module and CC-Link IE TSN module.	Page 150 CCPU_WriteLinkDevice
CCPU_WriteMCUnitLabel	To write data to module labels of a C Controller module in word units.	Page 151 CCPU_WriteMCUnitLabel
CCPU_WriteMCUnitLabelBit	To write data to module labels of a C Controller module in bit units.	Page 155 CCPU_WriteMCUnitLabelBit
CCPU_X_In_BitEx	To read an input signal (X) in bit (1-point) units.	Page 156 CCPU_X_In_BitEx
CCPU_X_In_WordEx	To read an input signal (X) in word (16-point) units.	Page 157 CCPU_X_In_WordEx
CCPU_Y_In_BitEx	To read an output signal (Y) in bit (1-point) units.	Page 158 CCPU_Y_In_BitEx
CCPU_Y_In_WordEx	To read an output signal (Y) in word (16-point) units.	Page 159 CCPU_Y_In_WordEx
CCPU_Y_Out_BitEx	To output an output signal (Y) in bit (1-point) units.	Page 160 CCPU_Y_Out_BitEx
CCPU_Y_Out_WordEx	To output an output signal (Y) in word (16-point) units.	Page 161 CCPU_Y_Out_WordEx

C Controller module dedicated functions for ISR

Function name	Function	Reference
CCPU_DisableInt_ISR	To disable the routine registered with the CCPU_EntryInt function.	Page 162 CCPU_DisableInt_ISR
CCPU_EnableInt_ISR	To enable the routine registered with the CCPU_EntryInt function.	Page 163 CCPU_EnableInt_ISR
CCPU_FromBuf_ISR	To read data from the CPU buffer memory of the CPU module and the buffer memory of the intelligent function module which are mounted on the specified module position. (FROM instruction)	Page 164 CCPU_FromBuf_ISR
CCPU_FromBufHG_ISR	To read data from the fixed cycle communication area of the CPU module mounted on the specified module position.	Page 165 CCPU_FromBufHG_ISR
CCPU_GetCounterMicros_ISR	To acquire a 1 μ s counter value of a C Controller module.	Page 166 CCPU_GetCounterMicros_ISR
CCPU_GetCounterMillis_ISR	To acquire a 1 ms counter value of a C Controller module.	Page 167 CCPU_GetCounterMillis_ISR
CCPU_GetDotMatrixLED_ISR	To acquire the value displayed on the dot matrix LED of a C Controller module.	Page 168 CCPU_GetDotMatrixLED_ISR
CCPU_ReadDevice_ISR	To read data from internal user devices and internal system devices of a C Controller module.	Page 170 CCPU_ReadDevice_ISR
CCPU_RegistEventLog_ISR	To register an event log in the event history of a C Controller module.	Page 171 CCPU_RegistEventLog_ISR
CCPU_ResetDevice_ISR	To reset internal user devices and internal system devices (bit devices) of C Controller module.	Page 170 CCPU_ReadDevice_ISR
CCPU_SetDevice_ISR	To set internal user devices and internal system devices (bit devices) of C Controller module.	Page 173 CCPU_SetDevice_ISR
CCPU_SetDotMatrixLED_ISR	To set a value to be displayed on the dot matrix LED of C Controller module.	Page 174 CCPU_SetDotMatrixLED_ISR
CCPU_SetLEDStatus_ISR	To set the LED status of a C Controller module.	Page 176 CCPU_SetLEDStatus_ISR
CCPU_ToBuf_ISR	To write data to the CPU buffer memory of the CPU module (host CPU) and the buffer memory of the intelligent function module which are mounted on the specified module position. (TO instruction)	Page 177 CCPU_ToBuf_ISR
CCPU_ToBufHG_ISR	To write data to the fixed cycle communication area of the CPU module mounted on the specified module position.	Page 179 CCPU_ToBufHG_ISR
CCPU_WriteDevice_ISR	To write data to internal user devices and internal system devices of C Controller module.	Page 180 CCPU_WriteDevice_ISR
CCPU_X_In_Word_ISR	To read an input signal (X) in word (16-point) units.	Page 181 CCPU_X_In_Word_ISR
CCPU_Y_In_Word_ISR	To read an output signal (Y) in word (16-point) units.	Page 183 CCPU_Y_In_Word_ISR
CCPU_Y_Out_Word_ISR	To output an output signal (Y) in word (16-point) units.	Page 185 CCPU_Y_Out_Word_ISR

2.2 MELSEC Data Link Functions

The MELSEC data link functions are as listed below.

Function name	Function	Reference
mdClose	To close a communication line (channel).	Page 187 mdClose
mdControl	To perform remote operations (remote RUN/STOP/PAUSE) for the CPU module.	Page 188 mdControl
mdDevRstEx	To reset bit devices.	Page 189 mdDevRstEx
mdDevSetEx	To set bit devices.	Page 190 mdDevSetEx
mdGetLabelInfo	To acquire device information corresponding to label names.	Page 191 mdGetLabelInfo
mdInit	To initialize the communication route information.	Page 194 mdInit
mdOpen	To open a communication line (channel).	Page 195 mdOpen
mdRandREx	To read devices randomly.	Page 196 mdRandREx
mdRandRLabelEx	To read devices corresponding to labels randomly.	Page 199 mdRandRLabelEx
mdRandWEx	To write devices randomly.	Page 202 mdRandWEx
mdRandWLabelEx	To write devices corresponding to labels randomly.	Page 204 mdRandWLabelEx
mdReceiveEx	To read devices in a batch.	Page 206 mdReceiveEx
mdReceiveEx	To receive messages. (RECV function)	Page 207 mdReceiveEx
mdSendEx	To write devices in a batch.	Page 209 mdSendEx
mdSendEx	To send messages. (SEND function)	Page 210 mdSendEx
mdTypeRead	To read the model code of CPU module.	Page 212 mdTypeRead

2.3 Motion Module Dedicated Class

The following table shows motion module dedicated classes.

Function name	Function	Reference
MCFB::RefreshLabels	To execute an MCFB with I/O variables and input variables set in the MCFB setting function. After the execution, I/O variables, input variables and public variables are refreshed.	Page 215 MCFB::RefreshLabels
MCFBExecute::SetExecute	To set I/O variables and input variables of an Execute-type MCFB in a motion module.	Page 216 MCFBExecute::SetExecute
MCFBEnable::SetEnable	To set I/O variables and input variables of an Enable-type MCFB in a motion module.	Page 218 MCFBEnable::SetEnable
MCv_Jog::SetEnableJog	To set I/O variables and input variables of the Enable-type MCv_Jog in a motion module.	Page 220 MCv_Jog::SetEnableJog

3 DETAILS OF FUNCTION

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

3.1 C Controller Module Dedicated Functions

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

CCPU_ChangeCCIEFBCycPrm

This function changes operation parameter of cyclic transmission of the CC-Link IE Field Network Basic function.

Format

short CCPU_ChangeCCIEFBCycPrm (unsigned short usGroupNo, unsigned short usLinkScanTime, unsigned short usTimeout, unsigned short usTimeoutRetryCnt)


Description

- This function changes the operation parameter of cyclic transmission for the specified group.
- If a value out of the range is specified to the link scan time (usLinkScanTime) and the number of detections of disconnected remote stations (usTimeoutRetryCnt), an error which indicates that the value is out of the range will be returned.
- Before the execution of the CCPU_ChangeCCIEFBCycPrm function, set the parameters of CC-Link IE Field Network Basic with CW Configurator. Otherwise, an error response will be returned.
- The timeout time (usTimeout) of the specified remote station operates in tick units.
- The parameters set with the CCPU_ChangeCCIEFBCycPrm function are applied in the next link scan. The CCPU_ChangeCCIEFBCycPrm function waits for the completion of the operation parameter application.

Argument

Argument	Name	Description	IN/OUT
usGroupNo	Group No.	Specify a group number to change the operation parameter of link scan. (0 to 4) (When '0' is specified, the same value is set to all groups.)	IN
usLinkScanTime	Link scan time	Specify the link scan time of a cyclic transmission in milliseconds. (0 to 10000 ms)	IN
usTimeout	Remote station timeout time	Specify a timeout time (ms) for detecting the disconnection of remote stations. (10 to 65535 ms)	IN
usTimeoutRetryCnt	Number of detections of disconnected remote stations	Specify the number of timeouts for detecting the disconnection of remote stations. (3 to 10)	IN

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 126 CCPU_RestoreDefaultCCIEFBCycPrm

CCPU_ChangeFileSecurity

This function changes the file access restriction status of a C Controller module.

Format

```
short CCPU_ChangeFileSecurity(short sMode, char* pcPass);
```


Argument

Argument	Name	Description	IN/OUT
sMode	File access mode	Specify the file access mode. (When 'Reserved' is specified, this function ends normally without processing.) <ul style="list-style-type: none">• 0: Access restriction clear mode• 1: Access restriction mode• Others: Reserved	IN
pcPass	Password	Specify the security password.	IN

Description

- Specify the file access restriction status to the file access mode (sMode).
- To change the file access mode (sMode), use the security password.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 103 CCPU_GetFileSecurity

CCPU_ClearError

This function clears errors of a C Controller module.

Format

short CCPU_ClearError (long* pErrorInfo)


Argument

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

Description

- This function clears errors occurred in a C Controller module.
- When no error occurs, the CCPU_ClearError function ends normally.
- When a stop error has occurred, the error cannot be cleared. (The CCPU_ClearError 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 222 ERROR CODE LIST

Relevant function

- Page 102 CCPU_GetErrInfo

CCPU_Control

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

Format

short CCPU_Control (short sCpuNo, short sCode)

Argument


Argument	Name	Description	IN/OUT
sCpuNo	CPU number	Specify the target CPU module number. Specify '0' to perform the remote operation for the host CPU. (When the CPU module specified with 1 to 4 is the host CPU, an error is returned.) <ul style="list-style-type: none">• 0: Host CPU• 1 to 4: Other CPUs	IN
sCode	Remote operation specification code	Specify the remote operation to be performed. <ul style="list-style-type: none">• 0: Remote RUN• 1: Remote STOP• 2: Remote PAUSE	IN

Description

- This function performs remote operations (remote RUN/STOP/PAUSE) for a CPU module or a C Controller module specified to the CPU number (sCpuNo).
- The RUN/STOP/RESET switch has a higher priority to change the operating status of a C Controller module. Therefore, if the RUN/STOP/RESET switch is put in the STOP state, the operating status will remain STOP irrespective of the specified remote operation.

However, since the remote operation by the CCPU_Control function is effective even when the RUN/STOP/RESET switch is put in the STOP state, a C Controller module operates according to the last specified remote operation once the RUN/STOP/RESET switch is switched from STOP to RUN.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

CCPU_DedicatedDInst

This function executes dedicated instructions categorized as 'D' or 'DP'.

Format

short CCPU_DedicatedDInst (char* pcInstName, short sCPUNo, short* psArg1, short sArg1Size, short* psArg2, short sArg2Size, short* psArg3, short sArg3Size, short* psArg4, short sArg4Size, short* psArg5, short sArg5Size, short* psArg6, short sArg6Size, short* psArg7, short sArg7Size, short* psArg8, short sArg8Size, short* psArg9, short sArg9Size)

Argument

Argument	Name	Description	IN/OUT
pcInstName	Instruction name	Specify the instruction name of the dedicated instruction to be executed.	IN
sCPUNo	Start input/output number of the target CPU	Start input/output number of the target CPU divided by 16 • CPU No.1 to 4: 3E0H to 3E3H	IN
psArg1	Setting data (1st) ^{*1}	Specify the first setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg1Size	Setting data size (1st) ^{*1}	Specify the size of the first setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg2	Setting data (2nd) ^{*1}	Specify the second setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg2Size	Setting data size (2nd) ^{*1}	Specify the size of the second setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg3	Setting data (3rd) ^{*1}	Specify the third setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg3Size	Setting data size (3rd) ^{*1}	Specify the size of the third setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg4	Setting data (4th) ^{*1}	Specify the fourth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg4Size	Setting data size (4th) ^{*1}	Specify the size of the fourth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg5	Setting data (5th) ^{*1}	Specify the fifth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg5Size	Setting data size (5th) ^{*1}	Specify the size of the fifth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg6	Setting data (6th) ^{*1}	Specify the sixth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg6Size	Setting data size (6th) ^{*1}	Specify the size of the sixth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg7	Setting data (7th) ^{*1}	Specify the seventh setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg7Size	Setting data size (7th) ^{*1}	Specify the size of the seventh setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg8	Setting data (8th) ^{*1}	Specify the eighth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg8Size	Setting data size (8th) ^{*1}	Specify the size of the eighth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg9	Setting data (9th) ^{*1}	Specify the ninth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg9Size	Setting data size (9th) ^{*1}	Specify the size of the ninth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN

*1 Out of the setting data for the dedicated instruction to be executed, setting the "start input/output number of the target CPU divided by 16" is not required.

Description

The dedicated instructions that can be specified to the instruction name (pcInstName) are as shown below.

For the specifications of each dedicated function and the completion status, refer to the programming manual for each module.

Instruction name	Description	Instruction	Supported firmware version of C Controller module
CHGA	To request a motion CPU to change the current value.	D.CHGA, DP.CHGA	'01' or later
CHGAS	To request a motion CPU to change the current value.	D.CHGAS, DP.CHGAS	
CHGV	To request a motion CPU to change the speed value.	D.CHGV, DP.CHGV	
CHGVS	To request a motion CPU to change the speed value.	D.CHGVS, DP.CHGVS	
CHGT	To request a motion CPU to change the torque limit value.	D.CHGT, DP.CHGT	
DDRD	To read data from the device of a motion CPU.	D.DDRD, DP.DDRD	
GINT*1	To issue an interrupt to a motion CPU or a C Controller module.	D.GINT, DP.GINT	
SFCS	To request a motion CPU to start a motion SFC program.	D.SFCS, DP.SFCS	
SVST	To request a motion CPU to start a servo program.	D.SVST, DP.SVST	
DDWR	To write data to the device of a motion CPU.	D.DDWR, DP.DDWR	
MCNST	To request a motion CPU to start a machine program operation.	D.MCNST, DP.MCNST	
BITWR	To write data to the bit device of a motion CPU.	D.BITWR, DP.BITWR	
SVSTD	To issue a direct positioning start request to a motion CPU.	D.SVSTD, DP.SVSTD	

*1 The arguments to issue an interrupt from a C Controller module by using the CCPU_DedicatedDInst function are the same as those to issue an interrupt to a C Controller module by using the CCPU_DedicatedMInst function.

For details on the arguments, refer to the following:

☞ Page 79 CCPU_DedicatedMInst

Precautions

- To execute the CCPU_DedicatedDInst function, reserve an area to store the completion device of the dedicated instruction and specify it to the argument. If the area and its size are not specified to the argument, an error occurs. (The instruction is not executed properly.)
- Depending on the type of dedicated instruction, the return value of the CCPU_DedicatedDInst function may be normal even if wrong argument or size is specified. Make sure to check the completion status by referring to the manual for the dedicated instruction.
- Specifying an incorrect argument may result in unexpected operation. Make sure to refer to the manual for dedicated instruction to specify the argument.
- The completion processing of the dedicated instruction is performed in the fixed cycle processing. To reduce the processing time of the function, set a small value for the refresh cycle.
- The operation varies depending on the firmware version of C Controller module. Ensure to check the supported firmware version of the C Controller module.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 222 ERROR CODE LIST

Relevant functions

- Page 75 CCPU_DedicatedGInst
- Page 77 CCPU_DedicatedJInst
- Page 79 CCPU_DedicatedMInst

CCPU_DedicatedInst

This function executes dedicated instructions categorized as 'G' or 'GP'.

Format

short CCPU_DedicatedInst (char* pcInstName, short sloNo, short* psArg1, short sArg1Size, short* psArg2, short sArg2Size, short* psArg3, short sArg3Size, short* psArg4, short sArg4Size, short* psArg5, short sArg5Size, short* psArg6, short sArg6Size, short* psArg7, short sArg7Size, short* psArg8, short sArg8Size, short* psArg9, short sArg9Size)

Argument

Argument	Name	Description	IN/OUT
pcInstName	Instruction name	Specify the instruction name of the dedicated instruction to be executed.	IN
sloNo	Start input/output number of the own station	Specify the start input/output number of the own station divided by 16. (00H to FEH)	IN
psArg1	Setting data (1st) ^{*1}	Specify the first setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg1Size	Setting data size (1st) ^{*1}	Specify the size of the first setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg2	Setting data (2nd) ^{*1}	Specify the second setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg2Size	Setting data size (2nd) ^{*1}	Specify the size of the second setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg3	Setting data (3rd) ^{*1}	Specify the third setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg3Size	Setting data size (3rd) ^{*1}	Specify the size of the third setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg4	Setting data (4th) ^{*1}	Specify the fourth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg4Size	Setting data size (4th) ^{*1}	Specify the size of the fourth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg5	Setting data (5th) ^{*1}	Specify the fifth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg5Size	Setting data size (5th) ^{*1}	Specify the size of the fifth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg6	Setting data (6th) ^{*1}	Specify the sixth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg6Size	Setting data size (6th) ^{*1}	Specify the size of the sixth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg7	Setting data (7th) ^{*1}	Specify the seventh setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg7Size	Setting data size (7th) ^{*1}	Specify the size of the seventh setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg8	Setting data (8th) ^{*1}	Specify the eighth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg8Size	Setting data size (8th) ^{*1}	Specify the size of the eighth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg9	Setting data (9th) ^{*1}	Specify the ninth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg9Size	Setting data size (9th) ^{*1}	Specify the size of the ninth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN

*1 Out of the setting data for the dedicated instruction to be executed, setting the "start input/output number of the own station" is not required.

Description

The dedicated instructions that can be specified to the instruction name (pcInstName) are as shown below.


For the specifications of each dedicated function and the completion status, refer to the programming manual for each module.

Instruction name	Description	Instruction	Supported firmware version of C Controller module
SEND	Sends data to other station programmable controller.	GP.SEND	'01' or later
RECV	Receives data from other station programmable controller.	GP.RECV	
CCPASET	Sets parameters of a master station, a submaster station, and local stations of CC-Link IE Field Network module and CC-Link IE TSN.	G.CCPASET, GP.CCPASET	
CEXECUTE	Sends a command to execute processing in a motion module.	G.CEXECUTE, GP.CEXECUTE	'15' or later
REMFRIP	Reads data from the buffer memory of remote station in word units (target station IP address specified) (16-bit address specified).	GP.REMFRIP	
REMFRDIP	Reads data from the buffer memory of remote station in word units (target station IP address specified) (32-bit address specified).	GP.REMFRDIP	
REMTOIP	Writes data to the buffer memory of remote station in word units (target station IP address specified) (16-bit address specified).	GP.REMTOIP	
REMTODIP	Writes data to the buffer memory of remote station in word units (target station IP address specified) (32-bit address specified).	GP.REMTODIP	
CCPASETX	Sets parameters to the RJ71GN11-T2.	G.CCPASETX, GP.CCPASETX	
UINI	Sets the station number and IP address to the own station where these are not set yet.	G.UINI, GP.UINI	

Precautions

- To execute this function, make sure to reserve an area to store the completion device of the dedicated instruction and specify it to the argument. If the area and its size are not specified to the argument, an error occurs. (The instruction is not executed properly.)
- Depending on the type of dedicated instruction, the return value of this function may be normal even if wrong argument or size is specified. Make sure to check the completion status by referring to the manual for the dedicated instruction.
- Specifying an incorrect argument may result in unexpected operation. Make sure to refer to the manual for dedicated instruction to specify the argument.
- The completion processing of the dedicated instruction is performed in the fixed cycle processing. To reduce the processing time of the function, set a small value for the refresh cycle.
- The operation varies depending on the firmware version of C Controller module. Ensure to check the supported firmware version of the C Controller module.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 73 CCPU_DedicatedDInst
- Page 77 CCPU_DedicatedJInst
- Page 79 CCPU_DedicatedMInst

CCPU_DedicatedJInst

This function executes dedicated instructions categorized as 'J' or 'JP'.

Format

short CCPU_DedicatedJInst (char* pcInstName, short sNetNo, short* psArg1, short sArg1Size, short* psArg2, short sArg2Size, short* psArg3, short sArg3Size, short* psArg4, short sArg4Size, short* psArg5, short sArg5Size, short* psArg6, short sArg6Size, short* psArg7, short sArg7Size, short* psArg8, short sArg8Size, short* psArg9, short sArg9Size)

Argument

Argument	Name	Description	IN/OUT
pcInstName	Instruction name	Specify the instruction name of the dedicated instruction to be executed.	IN
sNetNo	Network number of the own station	Specify the network number of own station. (1 to 239)	IN
psArg1	Setting data (1st) ^{*1}	Specify the first setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg1Size	Setting data size (1st) ^{*1}	Specify the size of the first setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg2	Setting data (2nd) ^{*1}	Specify the second setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg2Size	Setting data size (2nd) ^{*1}	Specify the size of the second setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg3	Setting data (3rd) ^{*1}	Specify the third setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg3Size	Setting data size (3rd) ^{*1}	Specify the size of the third setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg4	Setting data (4th) ^{*1}	Specify the fourth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg4Size	Setting data size (4th) ^{*1}	Specify the size of the fourth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg5	Setting data (5th) ^{*1}	Specify the fifth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg5Size	Setting data size (5th) ^{*1}	Specify the size of the fifth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg6	Setting data (6th) ^{*1}	Specify the sixth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg6Size	Setting data size (6th) ^{*1}	Specify the size of the sixth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg7	Setting data (7th) ^{*1}	Specify the seventh setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg7Size	Setting data size (7th) ^{*1}	Specify the size of the seventh setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg8	Setting data (8th) ^{*1}	Specify the eighth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg8Size	Setting data size (8th) ^{*1}	Specify the size of the eighth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg9	Setting data (9th) ^{*1}	Specify the ninth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg9Size	Setting data size (9th) ^{*1}	Specify the size of the ninth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN

^{*1} Out of the setting data for the dedicated instruction to be executed, setting the "network number of the own station" is not required.

Description

The dedicated instructions that can be specified to the instruction name (pclInstName) are as shown below.
For the specifications of each dedicated function and the completion status, refer to the programming manual for each module.


Instruction name	Description	Instruction	Supported firmware version of C Controller module
SEND	Sends data to other station programmable controller.	JP.SEND	'01' or later
RECV	Receives data from other station programmable controller.	JP.RECV	
REMT0 ^{*1}	Writes data to the buffer memory of intelligent device station/remote device station.	JP.REMTO	
REMFR ^{*1}	Reads data from the buffer memory of intelligent device station/remote device station.	JP.REMFR	
REMFRD	Reads data from the buffer memory of remote station in word units (32-bit address specified).	JP.REMFRD	'15' or later
REMTOD	Writes data to the buffer memory of remote station in word units (32-bit address specified).	JP.REMTOD	

^{*1} The error code is stored in SW0080 to SW009F instead of the completion status.
Acquire the error codes stored in SW0080 to SW009F by using the CCPU_ReadLinkDevice function.

Precautions

- To execute this function, make sure to reserve an area to store the completion device of the dedicated instruction and specify it to the argument. If the area and its size are not specified to the argument, an error occurs. (The instruction is not executed properly.)
- Depending on the type of dedicated instruction, the return value of this function may be normal even if wrong argument or size is specified. Make sure to check the completion status by referring to the manual for the dedicated instruction.
- Specifying an incorrect argument may result in unexpected operation. Make sure to refer to the manual for dedicated instruction to specify the argument.
- The completion processing of the dedicated instruction is performed in the fixed cycle processing. To reduce the processing time of the function, set a small value for the refresh cycle.
- The operation varies depending on the firmware version of C Controller module. Ensure to check the supported firmware version of the C Controller module.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 73 CCPU_DedicatedDInst
- Page 75 CCPU_DedicatedGInst
- Page 79 CCPU_DedicatedMInst

CCPU_DedicatedMInst

This function executes dedicated instructions categorized as 'M' or 'MP'.

Format

short CCPU_DedicatedMInst (char* pcInstName, short sCPUNo, short* psArg1, short sArg1Size, short* psArg2, short sArg2Size, short* psArg3, short sArg3Size, short* psArg4, short sArg4Size, short* psArg5, short sArg5Size, short* psArg6, short sArg6Size, short* psArg7, short sArg7Size, short* psArg8, short sArg8Size, short* psArg9, short sArg9Size)

Argument

Argument	Name	Description	IN/OUT
pcInstName	Instruction name	Specify the instruction name of the dedicated instruction to be executed.	IN
sCPUNo	Start input/output number of the target CPU	Start input/output number of the target CPU divided by 16 (CPU No.1: 3E0H, CPU No.2: 3E1H, CPU No.3: 3E2H, CPU No.4: 3E3H)	IN
psArg1	Setting data (1st) ^{*1}	Specify the first setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg1Size	Setting data size (1st) ^{*1}	Specify the size of the first setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg2	Setting data (2nd) ^{*1}	Specify the second setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg2Size	Setting data size (2nd) ^{*1}	Specify the size of the second setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg3	Setting data (3rd) ^{*1}	Specify the third setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg3Size	Setting data size (3rd) ^{*1}	Specify the size of the third setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg4	Setting data (4th) ^{*1}	Specify the fourth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg4Size	Setting data size (4th) ^{*1}	Specify the size of the fourth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg5	Setting data (5th) ^{*1}	Specify the fifth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg5Size	Setting data size (5th) ^{*1}	Specify the size of the fifth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg6	Setting data (6th) ^{*1}	Specify the sixth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg6Size	Setting data size (6th) ^{*1}	Specify the size of the sixth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg7	Setting data (7th) ^{*1}	Specify the seventh setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg7Size	Setting data size (7th) ^{*1}	Specify the size of the seventh setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg8	Setting data (8th) ^{*1}	Specify the eighth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg8Size	Setting data size (8th) ^{*1}	Specify the size of the eighth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN
psArg9	Setting data (9th) ^{*1}	Specify the ninth setting data for the dedicated instruction to be executed. Specify 'NULL' if there is no setting data.	IN/OUT
sArg9Size	Setting data size (9th) ^{*1}	Specify the size of the ninth setting data for the dedicated instruction to be executed in word units. Specify '0' if there is no setting data.	IN

*1 Out of the setting data for the dedicated instruction to be executed, setting the "start input/output number of the target CPU divided by 16" is not required.

Description

The dedicated instructions that can be specified to the instruction name (pcInstName) are as shown below.

For the specifications of each dedicated function and the completion status, refer to the programming manual for each module.

Instruction name	Description	Instruction	Supported firmware version of C Controller module
GINT	To issue an interrupt to a motion CPU or a C Controller module.	M.GINT, MP.GINT	'01' or later
MCNST	To request a motion CPU to start a machine program operation.	M.MCNST, MP.MCNST	
BITWR	To write data to the bit device of a motion CPU.	M.BITWR, MP.BITWR	
SVSTD	To issue a direct positioning start request to a motion CPU.	M.SVSTD, MP.SVSTD	

To issue an interrupt from a C Controller module, specify the following values to each argument.

Argument	Description
pcInstName	Specify "GINT".
sCPUNo	Specify the target CPU module. • CPU No.1: 3E0H • CPU No.2: 3E1H • CPU No.3: 3E2H • CPU No.4: 3E3H
psArg1	Specify the interrupt pointer number to psArg1[0]. (0 to 15)
sArg1Size	Specify '1'.
psArg2	Specify the area to store the completion device (2 words). • Completion of the instruction processing: psArg2[1]=0, psArg2[0]=1 • Abnormal completion of the instruction processing: psArg2[1]=1, psArg2[0]=1
sArg2Size	Specify '2'.
psArg3	Specify the area to store the completion status (1 word). When the interrupt pointer number specified to psArg1[0] is other than 0 to 15, the psArg3[0] will be '2282H'.
sArg3Size	Specify '1'.
psArg4 to psArg9	Specify NULL.
sArg4Size to sArg9Size	Specify '0'.

Point


The completion status and completion device can be omitted at the same time. (Omitting only one of them is not available.)

To omit them, specify "0" to sArg2Size and sArg3Size, and "NULL" to psArg2 and psArg3.

Precautions

- To execute this function, reserve an area to store the completion device of the dedicated instruction and specify it to the argument. If the area and its size are not specified to the argument, an error occurs. (The instruction is not executed properly.)
- Depending on the type of dedicated instruction, the return value of this function may be normal even if wrong argument or size is specified. Make sure to check the completion status by referring to the manual for the dedicated instruction.
- Specifying an incorrect argument may result in unexpected operation. Make sure to refer to the manual for dedicated instruction to specify the argument.
- The completion processing of the dedicated instruction is performed in the fixed cycle processing. To reduce the processing time of the function, set a small value for the refresh cycle.
- The operation varies depending on the firmware version of C Controller module. Ensure to check the supported firmware version of the C Controller module.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 73 CCPU_DedicatedDInst
- Page 75 CCPU_DedicatedGInst
- Page 77 CCPU_DedicatedJInst

CCPU_DisableInt

This function disables the routine registered with the CCPU_EntryInt function.

Format

short CCPU_DisableInt (short sSINo)


Argument

Argument	Name	Description	IN/OUT
sSINo	Interrupt pointer number	Specify the interrupt pointer number.	IN

Description

- This function disables the routine registered with the CCPU_EntryInt function. (The routine is not executed when an interrupt occurs.)
- Specify the interrupt pointer number (sSINo) specified in the CCPU_EntryInt function to Interrupt pointer number (sSINo).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 86 CCPU_EntryInt
- Page 83 CCPU_EnableInt

CCPU_EnableInt

This function enables the routine registered with the CCPU_EntryInt function.

Format

short CCPU_EnableInt (short sSINo)


Argument

Argument	Name	Description	IN/OUT
sSINo	Interrupt pointer number	Specify the interrupt pointer number.	IN

Description

- This function enables the routine registered with the CCPU_EntryInt function. (The routine is executed when an interrupt occurs.)
- Specify the interrupt pointer number (sSINo) specified in the CCPU_EntryInt function to Interrupt pointer number (sSINo).
- Since an interrupt does not occur while a stop error is occurring in a C Controller module, the routine registered with the CCPU_EntryInt function will not be executed even if it is enabled.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 86 CCPU_EntryInt
- Page 82 CCPU_DisableInt

CCPU_EndCCIEFBDataAssurance

This function ends data assurance for one link scan of CC-Link IE Field Network Basic.

Format

short CCPU_EndCCIEFBDataAssurance (unsigned short usGroupNo)


Argument

Argument	Name	Description	IN/OUT
usGroupNo	Group No.	Specify a group number to end data assurance. (1 to 4)	IN

Description

- This function ends data assurance for one link scan of CC-Link IE Field Network Basic for the specified group.
- When the CCPU_EndCCIEFBDataAssurance function is executed, the cyclic transmission of the CC-Link IE Field Network Basic function is restarted.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 134 CCPU_StartCCIEFBDataAssurance

CCPU_EntryCCIEFBRefEndFunc

This function registers a routine to be called when the link scan of CC-Link IE Field Network Basic is completed.

Format

short CCPU_EntryCCIEFBRefEndFunc (CCPU_REFENDFUNCPtr pREFENDFuncPtr)

Argument

Argument	Name	Description	IN/OUT
pREFENDFuncPtr	Registered routine	Specify a routine to be registered. (The routine is deregistered by specifying NULL.)	IN

The data type of the registered routine (pREFENDFuncPtr) is defined in the header file 'CCPUFunc.h' as follows.

- Format

short (*CCPU_REFENDFUNCPtr) (unsigned short usGroupNo)

- Argument

Argument	Name	Description	IN/OUT
usGroupNo	Group No.	Specify the group number of remote stations whose link scan is completed.	OUT

- Return value

Return value	Description
0 (0000H)	Normal

Description

- This function registers a routine to be called when the link scan of CC-Link IE Field Network Basic is completed.
- Only one routine is executed at once. After the link scan of another group is completed during the execution of a routine, the module waits for the execution of the next routine until the current routine is completed.
- If the CCPU_EntryCCIEFBRefEndFunc function is executed once or more, the most recently registered routine will be valid.
- The registered routine operates on the task^{*1} where the CC-Link IE Field Network Basic function operates. Do not perform processing which causes a block or long processing which occupies a CPU because these processings can affect link scan time. Execute a long processing which occupies a CPU for a long time on another task.
The affect on link scan time can be checked with a maximum link scan time of a buffer memory.
- Do not call the CCPU_ChangeCCIEFBCycPrm function or CCPU_RestoreDefaultCCIEFBCycPrm function in the routine to be registered. Otherwise, the registered routine is not returned and therefore the link scan of the target group stops.


*1 The registered routine operates on the following task.

Task priority: 48

Stack size: 4096 byte

Task option: VX_FP_TASK

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

CCPU_EntryInt

This function registers a routine to be called when an interrupt occurs.

Format

short CCPU_EntryInt (short sSINo, CCPU_FUNCPtr pFuncPtr)

Argument

Argument	Name	Description	IN/OUT
sSINo	Interrupt pointer number	Specify the interrupt pointer number.	IN
pFuncPtr	Registered routine	Specify a routine to be registered. (The routine is deregistered by specifying NULL.)	IN

- The data type of the registered routine (pFuncPtr) is defined as void type in the header file, "CCPUFunc.h".

The specification method for the interrupt pointer number (sSINo) is as follows:

sSINo	Description
0 to 15	Interrupt by module
44	Inter-module synchronization interrupt
45	Multiple CPU synchronization interrupt
50 to 1023	Interrupt by module


Description

- This function registers the routine specified to the registered routine (pFuncPtr) in the interrupt specified with interrupt pointer number (sSINo).
- When NULL is specified to the registered routine (pFuncPtr), the routine is deregistered.
- Use the CCPU_EnableInt function to enable the routine registered with the CCPU_EntryInt function. Otherwise, the routine will not be called.

Precautions

- The registered routine is not executed while the operating system is in the interrupt-disabled state.
- For processing a routine to be registered in the registered routine (pFuncPtr), note the following:
A routine to be registered must not have an argument. (Do not pass an argument from an interrupt.)
When registering a routine, observe the considerations on the interrupt service routine (ISR).
Register minimal processing of a routine so that the processing time is as short as possible.
Only the C Controller module dedicated function for ISR can be used for a routine to be registered. Do not use any other function. (An error of a function to be registered is not checked.)
- When the CCPU_EntryInt function is executed multiple times by specifying the same interrupt pointer number (sSINo), the routine specified to the registered routine (pFuncPtr) last is registered. (Multiple routines cannot be registered.)
- The routine is disabled after the registration is done by this function.
- Calling an interrupt from other CPUs, a multiple CPU synchronization interrupt, and a WDT error interrupt is delayed while the routine registered with the CCPU_EntryInt function is in operation.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 83 CCPU_EnableInt
- Page 82 CCPU_DisableInt

CCPU_EntryTimerEvent

This function registers a timer event.

Format

short CCPU_EntryTimerEvent (long* plEvent)

Argument

Argument	Name	Description	IN/OUT
plEvent	Registered event	Specify a timer event to be registered.	IN

The specification method for the registered event (plEvent) is as follows:

plEvent	Description	
plEvent[0]	Number of timer event settings (1 to 16)	
plEvent[1]	First timer event number (1 to 16)	First timer event setting
plEvent[2]	Cycle of the first timer event (0: Clear, 1 to 60,000: Cycle [ms])	
plEvent[3]	Synchronization type of the first timer event (0: Batch synchronization, 1: Individual synchronization)	
plEvent[4]	Second timer event number (1 to 16)	Second timer event setting
plEvent[5]	Cycle of the second timer event (0: Clear, 1 to 60,000: Cycle [ms])	
plEvent[6]	Synchronization type of the second timer event (0: Batch synchronization, 1: Individual synchronization)	
plEvent[7]	Third timer event number (1 to 16)	Third timer event setting
plEvent[8]	Cycle of the third timer event (0: Clear, 1 to 60,000: Cycle [ms])	
plEvent[9]	Synchronization type of the third timer event (0: Batch synchronization, 1: Individual synchronization)	
⋮	⋮	⋮


When setting the timer event cycle, only the following specification method is applicable.

- For 1 to 1000: Specify multiples of 5 (by 5 milliseconds)
- For 1000 to 60,000: Specify multiples of 1000 (1 s units)

Description

- This function sets the cycle and synchronization type for the timer event registration.
- When '0' is specified for the cycle of the registered event (plEvent), the timer event is deregistered (the occurrence is cleared). Deregistration will clear the events that have occurred before that.
- Up to 16 timer events can be set. The cycle (1 to 60,000 [ms]) and synchronization type (batch synchronization or individual synchronization) can be specified for each timer event. For the synchronization type, refer to the description of the CCPU_WaitTimerEvent function.
- Specify the timer event number without duplication. Otherwise, an error will be returned.
- To change the cycle of a timer event number that the cycle is already set, clear it (specify '0' to the cycle), and then register the cycle (specify the cycle) again. Otherwise, an error will be returned.
- The registered timer event can be placed into the wait state with the CCPU_WaitTimerEvent function.
- All the timer events are cleared at the initial status.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 146 CCPU_WaitTimerEvent

CCPU_EntryWDTInt

This function registers a routine to be called when a user WDT error interrupt occurs.

Format

short CCPU_EntryWDTInt (short sType, CCPU_FUNCPTR pFuncPtr)

Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (If reserve is specified, an error is returned.) <ul style="list-style-type: none">• 0: User WDT• Others: Reserved	IN
pFuncPtr	Registered routine	Specify a routine to be registered. (The routine is deregistered by specifying NULL.)	IN

- The data type of the registered routine (pFuncPtr) is defined as void type in the header file, "CCPUFunc.h".

Description

- This function registers a routine to call when a user WDT error interrupt of C Controller module occurs.
- Specify the routine to be registered to the registered routine (pFuncPtr).
- When this function is executed several times, the last registered routine will be in effect.
- The routine registered with this function is executed as an interrupt service routine (ISR) when a user WDT error occurs. (If the CCPU_ResetWDT function is not executed within the time interval specified in the CCPU_StartWDT function, the WDT error interrupt will occur.)

Precautions


- The registered routine is not executed while the operating system is in the interrupt-disabled state.
- For processing a routine to be registered in the registered routine (pFuncPtr), note the following:
 - A routine to be registered must not have an argument. (Do not pass an argument from an interrupt.)
 - When registering a routine, observe the considerations on the interrupt service routine (ISR).
 - Register minimal processing of a routine so that the processing time is as short as possible.
 - Only the C Controller module dedicated function for ISR can be used for a routine to be registered. Do not use any other function. (An error of a function to be registered is not checked.)

■ ⚠ WARNING

When a routine that does not observe the considerations on interrupt service routine (ISR) is registered, the operating system may be runaway.

Make sure to use the routine after carefully verifying the operation and performance.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 135 CCPU_StartWDT
- Page 125 CCPU_ResetWDT
- Page 136 CCPU_StopWDT

CCPU_FromBuf

This function reads data from the CPU buffer memory of the CPU module and the buffer memory of the intelligent function module which are mounted on the specified module position. (FROM instruction)

Format

short CCPU_FromBuf (unsigned short usloNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
usloNo	Module position	Specify the module position. Start I/O number divided by 16 (0H to FFH, 3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
ulBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN

Description

- This function reads data for the size specified to the data size (ulSize) from the CPU buffer memory of a CPU module and the buffer memory of an intelligent function module which are specified to the module position (usloNo), and stores it in the data storage destination (pusDataBuf).


Data is read by specifying an offset address from the start of the CPU buffer memory of a CPU module and the buffer memory of an intelligent function module to the offset (ulOffset).

- To access the CPU buffer memory of the module in a multiple CPU system (CPU No.1 to No.4), specify 3E0H to 3E3H (CPU No.1 to No.4) to the module position (usloNo). However, the CPU buffer memory can be accessed only when the multiple CPU setting is configured.

Precautions

Note that the size of data storage destination (ulBufSize) should be equal to or bigger than the data size (ulSize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 139 CCPU_ToBuf

CCPU_FromBufHG

This function reads data from the fixed cycle communication area of the CPU module mounted on the specified module position.

Format

short CCPU_FromBufHG(unsigned short usIoNo, unsigned long short ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
usIoNo	Module position	Specify the module position. Start I/O number divided by 16 (3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
ulBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN


Description

- This function reads data for the size specified to the data size (ulSize) from the fixed cycle communication area of a CPU module specified to the module position (usIoNo), and stores it in the data storage destination (pusDataBuf). Data is read by specifying an offset address from the start of the fixed cycle communication area to the offset (ulOffset).
- The fixed cycle communication area can be accessed only when the fixed cycle communication area setting under the multiple CPU setting is configured.

Precautions

Note that the size of data storage destination (ulBufSize) should be equal to or bigger than the data size (ulSize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 91 CCPU_FromBuf
- Page 139 CCPU_ToBuf
- Page 140 CCPU_ToBufHG

CCPU_GetCCIEFBDiagnosticInfo

This function acquires the diagnostic information of CC-Link IE Field Network Basic.

Format

short CCPU_GetCCIEFBDiagnosticInfo (unsigned short usSlave, short* psStatusBuf, unsigned long ulBufSize)

Argument


Argument	Name	Description	IN/OUT
usSlave	Remote station number	Specify a remote station to acquire diagnostic information. (1 to 64)	IN
psStatusBuf	Diagnostic information storage destination	Specify a storage destination for diagnostic information.	OUT
ulBufSize	Diagnostic information storage destination size	Specify the size of the area which is reserved in the diagnostic information storage destination in word units.	IN

Description

- This function acquires CC-Link IE Field Network Basic information of the remote station specified to the remote station number (usSlave), and stores it to the diagnostic information storage destination (psStatusBuf).
- It also acquires the information for the size specified to the diagnostic information storage destination size (ulBufSize).
- The information to be stored to the diagnostic information storage destination (psStatusBuf) is as follows.

psStatusBuf	Description		
psStatusBuf[0]	Diagnostic information status flag		Diagnostic information status (valid or invalid) of the specified remote station b0 to b7: Status of diagnostic information 1 b8 to b15: Status of diagnostic information 2 <ul style="list-style-type: none">• 0: Invalid• 1: Valid
psStatusBuf[1]	Diagnostic information 1	Number of occupied stations	Number of occupied stations of the specified remote station
psStatusBuf[2]		Group No.	Group number of the specified remote station
psStatusBuf[3]		IP address (lower)	IP address (lower) of the specified remote station
psStatusBuf[4]		IP address (upper)	IP address (upper) of the specified remote station
psStatusBuf[5] to psStatusBuf[10]		Reserved	
psStatusBuf[11]		Accumulated number of timeouts	Accumulated number of timeouts of the specified remote station
psStatusBuf[12]		Accumulated number of disconnection detections	Accumulated number of disconnection detections of the specified remote station
psStatusBuf[13] to psStatusBuf[15]		Reserved	
psStatusBuf[16]	Diagnostic information 2	Manufacturer code	Manufacturer code of the specified remote station
psStatusBuf[17]		Reserved	
psStatusBuf[18]		Model code (lower)	Model code (lower) of the specified remote station
psStatusBuf[19]		Model code (upper)	Model code (upper) of the specified remote station
psStatusBuf[20]		Device version	Device version of the specified remote station
psStatusBuf[21]		Reserved	
psStatusBuf[22]		Module information	Module information of the specified remote station
psStatusBuf[23]		Error code	Error code of the latest error occurred on the specified remote station
psStatusBuf[24]		Detailed module information (lower)	Detailed module information (lower) of the specified remote station
psStatusBuf[25]		Detailed module information (upper)	Detailed module information (upper) of the specified remote station
psStatusBuf[26] to psStatusBuf[32]		Reserved	

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

CCPU_GetConstantProcessStatus

This function acquires the fixed cycle processing status of C Controller module.

Format

```
short CCPU_GetConstantProcessStatus(unsigned short * pusStatusBuf, unsigned long ulBufSize)
```

Argument

Argument	Name	Description	IN/OUT
pusStatusBuf	Fixed cycle processing status storage destination	Specify the fixed cycle processing status storage destination.	OUT
ulBufSize	Fixed cycle processing status storage destination size	Specify the fixed cycle processing status storage destination size in word units. (When '0' is specified, this function ends normally without processing.)	IN

Description


- This function acquires the fixed cycle processing status of C Controller module and stores it in the fixed cycle processing status storage destination (pusStatusBuf).
- It also acquires the information for the size specified to the fixed cycle processing status storage destination size (ulBufSize).
- The information to be stored in the fixed cycle processing status storage destination (pusStatusBuf) is as follows.

pusStatusBuf	Description
pusStatusBuf[0]	Fixed cycle processing cycle [ms] (setting value)*1
pusStatusBuf[1]	Current fixed cycle processing time [ms]
pusStatusBuf[2]	Current fixed cycle processing time [μs]
pusStatusBuf[3]	Minimum fixed cycle processing time [ms]
pusStatusBuf[4]	Minimum fixed cycle processing time [μs]
pusStatusBuf[5]	Maximum fixed cycle processing time [ms]
pusStatusBuf[6]	Maximum fixed cycle processing time [μs]

*1 The fixed cycle processing includes the refresh processing with network modules, the reset processing of watchdog timer, and the self-diagnostic processing. For details on the functions, refer to the following:

 MELSEC iQ-R C Controller 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 222 ERROR CODE LIST

Relevant function

- Page 102 CCPU_GetErrInfo

CCPU_GetCounterMicros

This function acquires a 1 μ s counter value of a C Controller module.

Format

short CCPU_GetCounterMicros(unsigned long* pulMicros)


Argument

Argument	Name	Description	IN/OUT
pulMicros	1 μ s counter value storage destination	Specify the storage destination of the 1 μ s counter value.	OUT

Description

- This function acquires a 1 μ s counter value of C Controller module and stores the value in the 1 μ s counter value storage destination (pulMicros).
- The 1 μ s counter value increases by 1 every 1 μ s after the power is turned ON.
- The count cycles between 0 and 4294967295.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 97 CCPU_GetCounterMillis

CCPU_GetCounterMillis

This function acquires a 1 ms counter value of a C Controller module.

Format

short CCPU_GetCounterMillis(unsigned long* pulMillis)


Argument

Argument	Name	Description	IN/OUT
pulMillis	1 ms counter value storage destination	Specify the storage destination of the 1 ms counter value.	OUT

Description

- This function acquires a 1 ms counter value of C Controller module and stores the value in the 1 ms counter value storage destination (pulMillis).
- The 1 ms counter value increases by 1 every 1 ms after the power is turned ON.
- The count cycles between 0 and 4294967295.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 96 CCPU_GetCounterMicros

CCPU_GetCpuStatus

This function acquires the operating status of a C Controller module.

Format

short CCPU_GetCpuStatus(long* plStatusBuf, unsigned long ulBufSize)

Argument


Argument	Name	Description	IN/OUT
plStatusBuf	Operating status storage destination	Specify the storage destination of the operating status.	OUT
ulBufSize	Operating status storage destination size	Specify the size of area reserved in the operating status storage destination in double word units. (When '0' is specified, this function ends normally without processing.)	IN

Description

- This function acquires the operating status of a C Controller module, and stores it to the operating status storage destination (plStatusBuf).
- It also acquires the information for the size specified to the operating status storage destination size (ulBufSize).
- The information to be stored in the operating status storage destination (plStatusBuf) is as follows.
(If information to be stored is not supported, '0' is set as its status.)

plStatusBuf	Description		
	Storage position		Status
plStatusBuf[0]	bit31 to 8	Reserved	—
	bit7 to 4	Cause of STOP/PAUSE	<ul style="list-style-type: none">• 0: RUN/STOP/RESET switch• 1: Reserved• 2: Reserved• 3: Execution of the CCPU_Control function from a user program• 4: Error• 5: Remote operations• Others: Reserved
	bit3 to 0	CPU operating status	<ul style="list-style-type: none">• 0: RUN state• 1: Reserved• 2: STOP state• 3: PAUSE state• Others: Reserved
plStatusBuf[1]	bit31 to 16	Reserved	—
	bit15 to 7	Reserved	—
	bit6	USB Mass Storage Class-compliant device status	<ul style="list-style-type: none">• 0: Inserted (mounted)• 1: Inserted (unmounted)• 2: Not inserted
	bit5		
	bit4	SD memory card status	<ul style="list-style-type: none">• 0: Inserted (mounted)• 1: Inserted (unmounted)• 2: Not inserted
	bit3		
	bit2	Reserved	—
	bit1	Program memory shutdown status	<ul style="list-style-type: none">• 0: Shutdown not performed• 1: Shutdown completed
	bit0	Data memory shutdown status	<ul style="list-style-type: none">• 0: Shutdown not performed• 1: Shutdown completed
plStatusBuf[2]	bit31 to 0	Index value for number of data memory write cycle	—

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 102 CCPU_GetErrInfo

CCPU_GetDotMatrixLED

This function acquires the value displayed on the dot matrix LED of a C Controller module, and stores it to the LED data storage destination (pcData).

Format

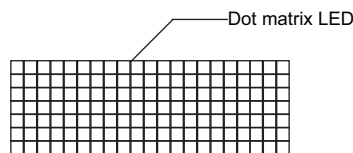
short CCPU_GetDotMatrixLED(char* pcData, unsigned long ulDataSize)

Argument

Argument	Name	Description	IN/OUT
pcData	LED data storage destination	Specify the storage destination of LED data.	OUT
ulDataSize	LED data storage destination size	Specify the LED data storage destination size in byte units. (When '0' is specified, this function ends normally without processing.)	IN

Description

- This function acquires the value displayed on the dot matrix LED, and stores it in the LED data storage destination (pcData).
- It also acquires the information for the size specified to the LED data storage destination size (ulDataSize).
- The value displayed on the dot matrix LED is stored in the LED data storage destination (pcData) as shown below.



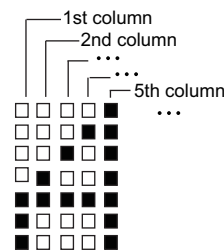
pcData[0] to pcData[19]: Data of the dot matrix LED (7×20)

The value displayed in the following format is acquired.

Data format for each column: Bit pattern in which '0' is for the upper one bit, and '1' (when LED is ON) or '0' (when LED is OFF) is for lower seven bits

Ex.

The bit pattern shown below is displayed on the dot matrix LED:



1st column: 0000 0111b = 07H → pcData[0] = 0x07

2nd column: 0000 1100b = 0cH → pcData[1] = 0x0c


3rd column: 0001 0100b = 14H → pcData[2] = 0x14

4th column: 0010 0100b = 24H → pcData[3] = 0x24

5th column: 0111 1111b = 7fH → pcData[4] = 0x7f

6th column to 20th column: 0000 0000b = 00H → pcData[5] to pcData[19] = 0x00

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 128 CCPU_SetDotMatrixLED

CCPU_GetErrInfo

This function acquires the error information of a C Controller module.

Format

short CCPU_GetErrInfo(unsigned short* pusErrorInfo, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
pusErrorInfo	Error information storage destination	Specify the error information storage destination .	OUT
ulBufSize	Error information storage destination size	Specify the error information storage destination size in word units. (When '0' is specified, this function ends normally without processing.)	IN

Description


- This function acquires the error information of a C Controller module, and stores it in the error information storage destination (pusErrorInfo).
- It also acquires the information for the size specified to the error information storage destination size (ulBufSize).
- The information to be stored in the error information storage destination (pusErrorInfo) is as follows.

pusErrorInfo	Description
pusErrorInfo[0]	Self-diagnostics error code 1
pusErrorInfo[1]	Self-diagnostics error code 2
pusErrorInfo[2]	Self-diagnostics error code 3
pusErrorInfo[3]	Self-diagnostics error code 4
pusErrorInfo[4]	Self-diagnostics error code 5
pusErrorInfo[5]	Self-diagnostics error code 6
pusErrorInfo[6]	Self-diagnostics error code 7
pusErrorInfo[7]	Self-diagnostics error code 8
pusErrorInfo[8]	Self-diagnostics error code 9
pusErrorInfo[9]	Self-diagnostics error code 10
pusErrorInfo[10]	Self-diagnostics error code 11
pusErrorInfo[11]	Self-diagnostics error code 12
pusErrorInfo[12]	Self-diagnostics error code 13
pusErrorInfo[13]	Self-diagnostics error code 14
pusErrorInfo[14]	Self-diagnostics error code 15
pusErrorInfo[15]	Self-diagnostics error code 16

Point

Up to 16 error codes for errors occurred in the self-diagnostics are stored in order from pusErrorInfo[0]. The error code which has already been stored is not stored.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 71 CCPU_ClearError

CCPU_GetFileSecurity

This function acquires the file access mode of a C Controller module.

Format

```
short CCPU_GetFileSecurity(short* psMode);
```


Argument

Argument	Name	Description	IN/OUT
psMode	File access mode	Stores the file access mode. <ul style="list-style-type: none">• 0: Access restriction clear mode• 1: Access restriction mode	OUT

Description

This function acquires the current file access mode, and stores it to the file access mode (psMode).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 70 CCPU_ChangeFileSecurity

CCPU_GetIDInfo

This function acquires the individual identification information of a C Controller module.

Format

```
short CCPU_GetIDInfo(unsigned char *pucGetData, unsigned long ulBufSize);
```

Argument


Argument	Name	Description	IN/OUT
pucGetData	Individual identification information storage destination	Specify the individual identification information storage destination.	OUT
ulBufSize	Individual identification information storage destination size	Specify the individual identification information storage destination size in word units.	IN

Description

- This function acquires the individual identification information of a C Controller module, and stores it in the individual identification information storage destination (pucGetData).
- It also acquires the information for the size specified to the individual identification information storage destination size (ulBufSize).
- The information to be stored in the individual identification information storage destination (pucGetData) is as follows.

pucGetData	Description
pucGetData[0]	Individual identification information for CH1
pucGetData[1]	
pucGetData[2]	
pucGetData[3]	
pucGetData[4]	
pucGetData[5]	
pucGetData[6]	Individual identification information for CH2
pucGetData[7]	
pucGetData[8]	
pucGetData[9]	
pucGetData[10]	
pucGetData[11]	

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 110 CCPU_GetSerialNo

CCPU_GetLEDStatus

This function acquires the LED status of a C Controller module.

Format

short CCPU_GetLEDStatus(long lLed, unsigned short* pusLedInfo, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
lLed	Target LED	Specify the target LED. (When 'Reserved' is specified, this function ends normally without processing.) <ul style="list-style-type: none">• 0: READY LED• 1: ERROR LED• 2: BUS RUN LED• 3: CARD RDY LED• 4: USER LED• 5: USB RDY LED• 6: RS SD/RD LED• -1: All of the LEDs above• Others: Reserved	IN
pusLedInfo	LED status storage destination	Specify the storage destination of the LED status.	OUT
ulBufSize	LED status storage destination size	Specify the LED status storage destination size in word units. (When '0' is specified, this function ends normally without processing.)	IN

Description

- This function acquires the LED status of the C Controller module specified to the target LED (lLed), and stores it in the LED status storage destination (pusLedInfo).
- It also acquires the information for the size specified to the LED status storage destination (ulBufSize).
- If the LED is not supported, '0' is set to the LED status.
- The LED status to be stored in the LED status storage destination (pusLedInfo) is as follows.


pusLedInfo	Description
0	OFF
1	ON (Red)
2	Flashing at low speed (Red)
3	Flashing at high speed (Red)
4	ON (Green)
5	Flashing at low speed (Green)
6	Flashing at high speed (Green)

- When '-1' is specified to the target LED (lLed), the LED status stored in the LED status storage destination (pusLedInfo) is as follows:

(When '0' to '6' is specified, the specified LED status is stored in pusLedInfo[0].)

pusLedInfo	Description
pusLedInfo[0]	READY LED status
pusLedInfo[1]	ERROR LED status
pusLedInfo[2]	BUS RUN LED status
pusLedInfo[3]	CARD RDY LED status
pusLedInfo[4]	USER LED status
pusLedInfo[5]	USB RDY LED status
pusLedInfo[6]	RS SD/RD LED status

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 102 CCPU_GetErrInfo

CCPU_GetOpSelectMode

This function acquires the operation selection mode of a C Controller module.

Format

short CCPU_GetOpSelectMode(long IModelInfo, long* pISelectMode)

Argument

Argument	Name	Description	IN/OUT
IModelInfo	Mode information	Specify the mode information. <ul style="list-style-type: none">• 1: Operation selection mode when the MODE/SELECT switch is held in the SELECT position• 2: Display mode of a dot matrix LED• Others: Reserved	IN
pISelectMode	Operation selection mode	Specify the storage destination of the acquired operation mode.	OUT

Description


- This function stores the operation selection mode of a C Controller module in the operation selection mode (pISelectMode).
- The information which is stored to the operation selection mode (pISelectMode) when 1 is specified to the mode information (IModelInfo) is as follows:

pISelectMode	Description
1	Notifies event to the user program.
2	Unmounts SD memory card forcibly.
3	Unmounts USB Mass Storage Class-compliant devices forcibly.
4	Unmounts SD memory card/USB Mass Storage Class-compliant devices forcibly.

- The information which is stored to the operation selection mode (pISelectMode) when 2 is specified to the mode information (IModelInfo) is as follows:

pISelectMode	Description
1	Displays the specified content on the dot matrix LED.
2	Displays an error code on the dot matrix LED.
3	Displays the IP address of CH1 on the dot matrix LED.
4	Displays the IP address of CH2 on the dot matrix LED.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 131 CCPU_SetOpSelectMode

CCPU_GetPowerStatus

This function acquires the power status of C Controller module.

Format

short CCPU_GetPowerStatus(long* plStatusBuf, unsigned long ulBufSize)

Argument


Argument	Name	Description	IN/OUT
plStatusBuf	Power status storage destination	Specify the storage destination of the power status.	OUT
ulBufSize	Power status storage destination size	Specify the power status storage destination size in double word units. (When '0' is specified, this function ends normally without processing.)	IN

Description

- This function acquires the power status of C Controller module and stores it in the power status storage destination (plStatusBuf).
- It also acquires the information for the size specified to the power status storage destination size (ulBufSize).
- The information to be stored in the power status storage destination (plStatusBuf) is as follows.

plStatusBuf	Description		
	Storage position		Status
plStatusBuf[0]	bit31 to 16	Reserved	—
	bit15 to 0	Number of detected momentary power failures	—

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 102 CCPU_GetErrInfo

CCPU_GetRTC

This function acquires the clock data (local time) of a C Controller module.

Format

short CCPU_GetRTC(short* psGetData,unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
psGetData	Clock data storage destination	Specify the storage destination of the clock data (local time).	OUT
ulBufSize	Clock data storage destination size	Specify the clock data (local time) storage destination size in word units. (When '0' is specified, this function ends normally without processing.)	IN

Description


- This function acquires the clock data (local time) of a C Controller module, and stores it in the clock data storage destination (psGetData).
- It also acquires the information for the size specified to the clock data storage destination size (ulBufSize).
- The clock data (local time) are stored in the clock data storage destination (psGetData) as follows.

(Available range: January 1, 1980 to December 31, 2079)

psGetData	Description
psGetData[0]	Year data (1980 to 2079)
psGetData[1]	Month data (1 to 12)
psGetData[2]	Day data (1 to 31)
psGetData[3]	Hour data (0 to 23)
psGetData[4]	Minute data (0 to 59)
psGetData[5]	Second data (0 to 59)
psGetData[6]	Day data (0 to 6) (0: Sunday, 1: Monday, 2: Tuesday, 3: Wednesday, 4: Thursday, 5: Friday, 6: Saturday)
psGetData[7] ^{*1}	Time zone (Unit: minute)
psGetData[8] ^{*1}	Daylight saving time status flag (0 to 1) (0: Not during daylight saving time, 1: During daylight saving time)

*1 Information can be acquired when using a C Controller module with the firmware version '06' or later. For the firmware version '05' or earlier, the area is not overwritten even when information is acquired.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 132 CCPU_SetRTC

CCPU_GetSerialNo

This function acquires the serial number of a C Controller module.

Format

short CCPU_GetSerialNo(char* pcGetData, unsigned long ulDataSize)

Argument


Argument	Name	Description	IN/OUT
pcGetData	Serial number storage destination	Specify the serial number storage destination.	OUT
ulDataSize	Serial number storage destination size	Specify 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 C Controller 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 (ulDataSize).

Description

- This function acquires the serial number (16-digits) of a C Controller module, and stores it in the serial number storage destination (pcGetData).
- It also acquires the information for the size specified to the serial number storage destination size (ulDataSize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 104 CCPU_GetIDInfo

CCPU_GetSwitchStatus

This function acquires the switch status of a C Controller module.

Format

short CCPU_GetSwitchStatus(long* plStatusBuf, unsigned long ulBufSize)

Argument


Argument	Name	Description	IN/OUT
plStatusBuf	Switch status storage destination	Specify the switch status storage destination.	OUT
ulBufSize	Switch status storage destination size	Specify the switch status storage destination size in double word units. (When '0' is specified, this function ends normally without processing.)	IN

Description

- This function acquires the switch status of a C Controller module, and stores it in the switch status storage destination (plStatusBuf).
- It also acquires the information for the size specified to the switch status storage destination size (ulBufSize).
- The information to be stored in the switch status storage destination (plStatusBuf) is as follows.

plStatusBuf	Description		
	Storage position		Status
plStatusBuf[0]	bit31-6	Reserved	—
	bit5-3	MODE/SELECT switch status	<ul style="list-style-type: none">• 000: MODE state• 010: NEUTRAL state• 100: SELECT state• Others: Reserved
	bit2-0	RUN/STOP/RESET switch status	<ul style="list-style-type: none">• 000: RESET state• 010: STOP state• 100: RUN state• Others: Reserved

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

CCPU_GetUnitInfo

This function acquires the module configuration information.

Format

short CCPU_GetUnitInfo (unsigned short* pusUnitInfo1, unsigned short* pusUnitInfo2, unsigned short* pusUnitInfo3)

Argument

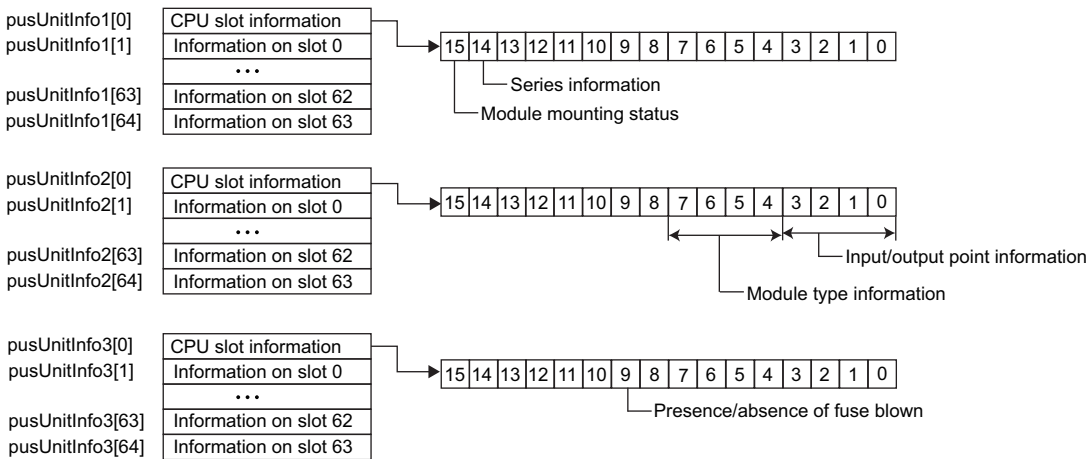
Argument	Name	Description	IN/OUT
pusUnitInfo1	Module configuration information 1	Specify the storage destination of module configuration information 1.	OUT
pusUnitInfo2	Module configuration information 2	Specify the storage destination of module configuration information 2.	OUT
pusUnitInfo3	Module configuration information 3	Specify the storage destination of module configuration information 3.	OUT

Description

This function reads module configuration information (65 slots) and stores to the module configuration information 1 (pusUnitInfo1), module configuration information 2 (pusUnitInfo2), and module configuration information 3 (pusUnitInfo3). Module configuration information to be stored differs depending on the series information.

Series information is MELSEC iQ-R series

The series can be checked in the 14th bit of pusUnitInfo1[0-64].



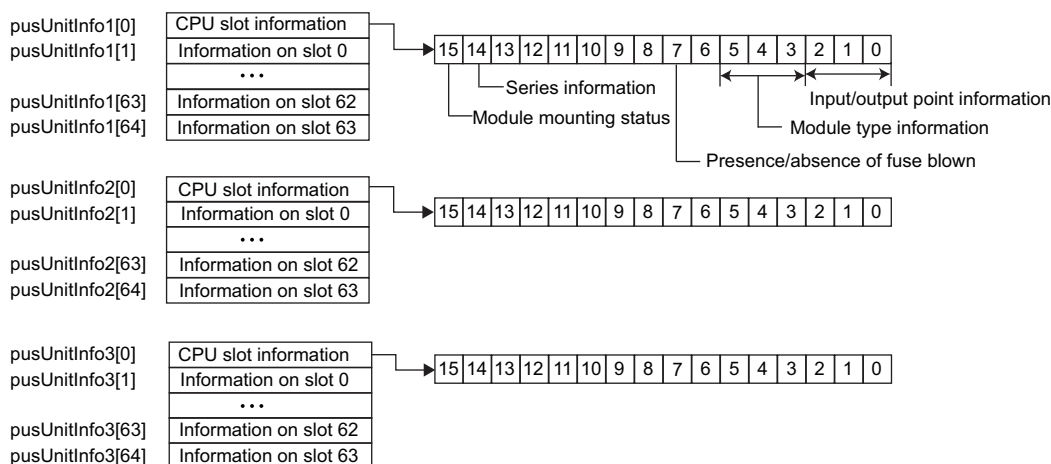
pusUnitInfo		Description		
		Storage position		Status
pusUnitInfo1[0-64]		bit15	Module mounting status	<ul style="list-style-type: none"> • 0: Not mounted • 1: Mounted
		bit14	Series information	1: MELSEC iQ-R series (0: MELSEC-Q series)
		bit13-0	Reserved	—
pusUnitInfo2[0-64]	No module mounted ^{*1}	bit15-0	Reserved	—
	A module mounted ^{*2}	bit15-8	Reserved	—
		bit7-4	Module type information	<ul style="list-style-type: none"> • 0000: Input module • 0001: Power • 0010: Output module • 0011: Base unit • 0100: Reserved • 0101: Reserved • 0110: I/O combined module • 0111: Empty • 1000: Intelligent function 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	<ul style="list-style-type: none"> • 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
pusUnitInfo3[0-64]	No module mounted ^{*1}	bit15-0	Reserved	—
	A module mounted ^{*2}	bit15-10	Reserved	—
		bit9	Presence/absence of fuse blown	<ul style="list-style-type: none"> • 0: Normal • 1: Fuse blown
		bit8-0	Reserved	—

*1 Indicates when '0' is stored in the 'bit 15' of pusUnitInfo1[N]. (N indicates the same array elements.)

*2 Indicates when '1' is stored in the 'bit 15' of pusUnitInfo1[N]. (N indicates the same array elements.)

Series information is MELSEC-Q series

The series can be checked in the 14th bit of pusUnitInfo1[0-64].



pusUnitInfo		Description		
		Storage position		Status
pusUnitInfo1[0-64]	—	bit15	Module mounting status	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• 0: Normal• 1: Fuse blown
		bit6	Reserved	—
		bit5-3	Module type information	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• 000: 16 points• 001: 32 points• 010: 48 points• 011: 64 points• 100: 128 points• 101: 256 points• 110: 512 points• 111: 1024 points
pusUnitInfo2[0-64]		bit15-0	Reserved	—
pusUnitInfo3[0-64]		bit15-0	Reserved	—

*1 Indicates when '0' is stored in the 'bit 15' of pusUnitInfo1[N]. (N indicates the same array elements.)

*2 Indicates when '1' is stored in the 'bit 15' of pusUnitInfo1[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 222 ERROR CODE LIST

CCPU_LockFWUpdate

This function prohibits the execution of the firmware update of a C Controller module.

Format

short CCPU_LockFWUpdate(char* pcPass)


Argument

Argument	Name	Description	IN/OUT
pcPass	Password	Specify a password for removing the prohibition on the firmware update.	IN

Description

- This function prohibits the execution of the firmware update of a C Controller module.
- 8 to 16 characters and symbols can be set for a password (pcPass). (Character is case sensitive.)
- Specify NULL as termination character for a password (pcPass).
- If the CCPU_LockFWUpdate function is executed while firmware update is prohibited, an error will be returned.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 141 CCPU_UnlockFWUpdate

CCPU_MountMemoryCard

This function mounts the SD memory card inserted to a C Controller module.

Format

short CCPU_MountMemoryCard (short sDrive)

Argument

Argument	Name	Description	IN/OUT
sDrive	Target drive	Specify a target drive. (When 'Reserved' is specified, this function ends normally without processing.) <ul style="list-style-type: none">• 1: SD memory card• Others: Reserved	IN

Description

- This function mounts the drive specified to the target drive (sDrive).
- The CARD RDY LED keeps flashing during the mount processing, and it turns ON once the mount processing is completed.
- This function can be executed when the status of an SD memory card is "Inserted (unmounted)".
(The status can be checked with the CCPU_GetCpuStatus function.)
- When an SD memory card has already been mounted, this function ends normally without processing.

Point

Use this function to access an SD memory card again without removing it after unmounting the SD memory card with the CCPU_UnmountMemoryCard function while the power is ON.

This function does not need to be executed since an SD memory card is automatically mounted when it is replaced.

Precautions

The USB Mass Storage Class-compliant device cannot be mounted using this function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: Page 222 ERROR CODE LIST

Relevant functions

- Page 142 CCPU_UnmountMemoryCard
- Page 98 CCPU_GetCpuStatus


CCPU_ReadDevice

This function reads data from internal user devices and internal system devices of C Controller module.

Format

short CCPU_ReadDevice (short sDevType, unsigned long ulDevNo, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number. (Only multiples of 16 can be specified for bit devices.)	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
ulBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN


Description

This function reads data of a device after the one specified to the device type (sDevType) and the start device number (ulDevNo) for the size specified to the data size (ulSize), and stores it in the data storage destination (pusDataBuf).

Precautions

Note that the size of data storage destination (ulBufSize) should be equal to or bigger than the data size (ulSize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 149 CCPU_WriteDevice


CCPU_ReadLinkDevice

This function reads data from the own station link devices of CC-Link IE Controller Network module, CC-Link IE Field Network module, Ethernet module (when CC-Link IE Field Network is selected), MELSECNET/H network module, and CC-Link IE TSN module which are controlled by a C Controller module.

Format

short CCPU_ReadLinkDevice (unsigned short usIoNo, short sDevType, unsigned long ulDevNo, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
usIoNo	Module position	Specify the module position as follows. Start I/O number divided by 16 (0H to FFH)	IN
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number. (Only multiples of 16 can be specified for bit devices.)	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
ulBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN


Description

This function reads data of subsequent devices after the device, which is specified in the device type (sDevType) and the start device number (ulDevNo), from the following modules specified in the module position (usIoNo): a CC-Link IE Controller Network module, CC-Link IE Field Network module, Ethernet module (when CC-Link IE Field Network is selected), MELSECNET/H network module, and CC-Link IE TSN module. The function reads the device data for the size specified in the data size (ulSize) and stores it in the data storage destination (pusDataBuf).

Precautions

- Note that the size of data storage destination (ulBufSize) should be equal to or bigger than the data size (ulSize).
- To access a motion module using CC-Link IE TSN, specify link devices (SB and SW). Specifying link devices other than SB and SW will cause an error.
- To access link devices of a network module controlled by another CPU, access to another CPU via bus interface using the MELSEC data link function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 150 CCPU_WriteLinkDevice

CCPU_ReadMCUnitLabel

This function reads data from module labels of a C Controller module in word units.

Format

short CCPU_ReadMCUnitLabel (unsigned long ulUnitLabel, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize, unsigned long ulUnitLabelID)

Argument

Argument	Name	Description	IN/OUT
ulUnitLabel	Module label	Specify a module label.	IN
ulOffset	Offset	Specify the offset from the specified module label in word units.	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
ulBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN
ulUnitLabelID	Module label ID	Specify a module label ID.	IN

Description

- This function reads data from the module label of C Controller module, which is specified for the module label (ulUnitLabel), to the specified data storage destination (pusDataBuf) for data size specified in the data size (ulSize).
- To use the CCPU_ReadMCUnitLabel function, include a header file that has the information of module labels output by CW Configurator, and specify a module label.

■Specifying a module label

- For the module label (ulUnitLabel), specify a module label other than bit type. Otherwise, unintended data is read.
- For the module label (ulUnitLabel), specify a module label which is set in CW Configurator in the structure format. For the example of specifying a module label, refer to 'Specification method for a module label' in the 'CCPU_WriteMCUnitLabel function' section. (📖 Page 152 Specification method for a module label)
- To specify an array label for the module label (ulUnitLabel) and read data from each array element, specify an offset for the offset (ulOffset) in word units. For an offset value to be specified, calculate from a data type and an array element number that are specified for the module label (ulUnitLabel).

Offset value = The number of words corresponded to a label data type × an array element number

The number of words varies depending on the label data type. For each label size, refer to the 'Specification method for data size' in the 'CCPU_WriteMCUnitLabel function' section. (📖 Page 153 Specification method for data size)

■Specifying a module label ID

- For the module label ID (ulUnitLabelID), specify a macro defined in the header file, which includes the information of module labels output by CW Configurator. For an example of specifying a module label ID, refer to 'Specification method for a module label ID' in the 'CCPU_WriteMCUnitLabel function' section. (📖 Page 153 Specification method for a module label ID)
- When specifying '0' for the module label ID (ulUnitLabelID), data will be read from the module label without checking the module label ID.


■Specifying data size

- For an example of specifying the data size (ulSize), refer to the 'Specification method for data size' in the 'CCPU_WriteMCUnitLabel function' section. (📖 Page 153 Specification method for data size)

Precautions

When reading data from module labels, data is read from the module label (ulUnitLabel) and offset (ulOffset) for the size specified in the size (ulSize). However, this function does not check whether the data is within the specified module label. Therefore, data may be read from module labels other than the specified ones depending on the offset (ulOffset) and size (ulSize). (For the offset (ulOffset) and size (ulSize) check, this function only checks whether the sum of module label (ulUnitLabel), offset (ulOffset) and data size (ulSize) is exceeding the module label area size (for all modules).)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 151 CCPU_WriteMCUnitLabel

CCPU_ReadMCUnitLabelBit

This function reads data from module labels of a C Controller module in bit units.

Format

short CCPU_ReadMCUnitLabelBit (unsigned long ulUnitLabel, unsigned short* pusDataBuf, unsigned long ulUnitLabelID)

Argument

Argument	Name	Description	IN/OUT
ulUnitLabel	Module label	Specify a module label.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
ulUnitLabelID	Module label ID	Specify a module label ID.	IN


Description

- This function reads data from the module label of C Controller module, which is specified for the module label (ulUnitLabel), to the specified data storage destination (pusDataBuf).
- To use the CCPU_ReadMCUnitLabelBit function, include a header file that has the information of module labels output by CW Configurator, and specify a module label.
- The read data is stored in the data storage destination (pusDataBuf). (0: OFF/1: ON)


■Specifying a module label

- For the module label (ulUnitLabel), specify a bit-type module label. Specifying a module label other than bit type causes unintended data to be read.

■Specifying a module label ID

- For the module label ID (ulUnitLabelID), specify a macro defined in the header file, which includes the information of module labels output by CW Configurator. For an example of specifying a module label ID, refer to 'Specification method for a module label ID' in the 'CCPU_WriteMCUnitLabel function' section. ( Page 153 Specification method for a module label ID)
- When specifying '0' for the module label ID (ulUnitLabelID), data will be read from the module label without checking the module label ID.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 155 CCPU_WriteMCUnitLabelBit

CCPU_RegistEventLog

This function registers an event log in the event history of a C Controller module.

Format

short CCPU_RegistEventLog (long IEventCode, char* pcEventMsg)

Argument

Argument	Name	Description	IN/OUT
IEventCode	Detailed code	Specify a detailed event code to be registered in the event history.	IN
pcEventMsg	Detailed information	Specify detailed information character string data of an event to be registered in the event history. (The detailed information character string data of an event can be specified up to 200 bytes. When 'NULL' is specified, the detailed information is not registered.)	IN

Description


This function registers an event log in the event history of a C Controller module.

The contents to be registered on the event history screen of CW Configurator are as follows:

Item	Description
Occurrence Date	Event registered date and time
Event Type	Operation (fixed)
Status	Information (fixed)
Event Code	25000 (fixed)
Overview	Registration from the user program (fixed)
Source	R12CCPU-V (fixed)
Start I/O No.	Input/output number of the C Controller module that executed the CCPU_RegistEventLog function.
Detailed event code information	Detailed code (hexadecimal) specified to the detailed code (IEventCode)
Detailed event log information	Detailed information specified to the detailed information (pcEventMsg)
Cause	The event history was registered from the C Controller module detailed function. (Fixed)

- The event history can be stored for the size of the event history file specified with CW Configurator.
Note that data is deleted in order from older data if the specified file size is exceeded.
- An error occurs if the character string data specified to the detailed information (pcEventMsg) is 201 bytes or bigger.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

CCPU_Reset

This function resets the bus master CPU (CPU No.1).

Format

short CCPU_Reset (void)

Argument

None

Description


- This function resets the bus master CPU (CPU No.1).
- Use this function only to reset the bus master CPU due to an error or others.
- Do not execute this function while a file in the program memory, SD memory card, and USB Mass Storage Class-compliant device is being accessed. Doing so may result in data corruption or file system failure.
- Perform the following before executing this function while a file is being accessed.

File access destination	Description
Program memory	Close a user file.
SD memory card, USB Mass Storage Class-compliant device	Close a user file, and unmount an SD memory card and a USB Mass Storage Class-compliant device.


- This function can be executed only when all the following conditions are satisfied.
If the condition is not satisfied, the error codes indicated in the brackets below will be returned.

Host CPU	Description
Bus master CPU (CPU No.1)	"Enable" is set to "Remote Reset" in the bus master CPU (CPU No.1). (Unset: 16523) The operating status of the bus master CPU (CPU No.1) is in the STOP state. (RUN/PAUSE state: 16400)
CPUs other than the bus master (CPU No.1)	The bus master CPU (CPU No.1) is a CPU module: (C Controller module: -222) "Enable" is set to "Remote Reset" in the bus master CPU (CPU No.1). (Unset: -222) The operating status of the bus master CPU (CPU No.1) is in the STOP state. (RUN/PAUSE state: -222)

Precautions

- When the remote STOP is performed to the bus master CPU (CPU No.1) from other peripheral devices (such as GX Works3), the bus master CPU (CPU No.1) cannot be reset with the CCPU_Reset function.
For the remote operation and the operating status of a C Controller module, refer to the following:
 MELSEC iQ-R C Controller Module User's Manual
- When this function is executed, no value is returned because a C Controller module is restarted from an operating system.
(All programs are forcibly terminated.)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 142 CCPU_UnmountMemoryCard
- Page 133 CCPU_ShutdownRom


CCPU_ResetDevice

This function resets internal user devices and internal system devices (bit devices) of C Controller module.

Format

short CCPU_ResetDevice (short sDevType, unsigned long ulDevNo)


Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number.	IN

Description

This function resets (turns OFF) the device of a C Controller module specified to the device type (sDevType) and the start device number (ulDevNo).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 127 CCPU_SetDevice

CCPU_ResetWDT

This function resets the user WDT of a C Controller module.

Format

short CCPU_ResetWDT (short sType)

Argument


Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (If reserve is specified, an error is returned.) <ul style="list-style-type: none">• 0: User WDT• Others: Reserved	IN

3

Description

- This function resets the user WDT.
- When this function is executed without starting the user WDT, an error will be returned.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 135 CCPU_StartWDT
- Page 136 CCPU_StopWDT
- Page 90 CCPU_EntryWDTInt

CCPU_RestoreDefaultCCIEFBCycPrm

This function restores the operation parameter of cyclic transmission of CC-Link IE Field Network Basic to the default value (which is set in the parameter).

Format

short CCPU_RestoreDefaultCCIEFBCycPrm (void)


Argument

None

Description

- This function restores the all group operation parameters of cyclic transmission to their default value (which are set in the parameter).
- The parameters set with the CCPU_RestoreDefaultCCIEFBCycPrm function are applied in the next link scan. The CCPU_RestoreDefaultCCIEFBCycPrm function waits for the completion of the operation parameter application.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 69 CCPU_ChangeCCIEFBCycPrm


CCPU_SetDevice

This function sets internal user devices and internal system devices (bit devices) of C Controller module.

Format

short CCPU_SetDevice (short sDevType, unsigned long ulDevNo)


Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Device number	Specify the device number.	IN

Description

This function sets (turns ON) the device of a C Controller module specified to the device type (sDevType) and the start device number (ulDevNo).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 124 CCPU_ResetDevice

CCPU_SetDotMatrixLED

This function sets a value to be displayed on the dot matrix LED of C Controller module.

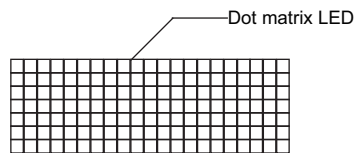
Format

short CCPU_SetDotMatrixLED(unsigned short usLedMode, char* pcData)

Argument

Argument	Name	Description	IN/OUT
usLedMode	Output mode	Specify the output mode to the dot matrix LED. (When 'Reserved' is specified, this function ends normally without processing.) <ul style="list-style-type: none">• 0: Dot mode• 1: ASCII mode• Others: Reserved	IN
pcData	LED data	Specify the LED data.	IN

- Specify the LED data (pcData) as follows.
- Mode 0: Dot mode



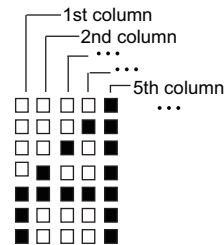
pcData[0] to pcData[19]: Data of the dot matrix LED (7 × 20)

The data specified in the following format is displayed.

Data format for each column: Bit pattern in which '0' is for the upper one bit, and '1' (when LED is ON) or '0' (when LED is OFF) is for lower seven bits

Ex.

The bit pattern shown below is output to the dot matrix LED:



1st column: 0000 0111b = 07H → pcData[0] = 0x07
2nd column: 0000 1100b = 0cH → pcData[1] = 0x0c
3rd column: 0001 0100b = 14H → pcData[2] = 0x14
4th column: 0010 0100b = 24H → pcData[3] = 0x24
5th column: 0111 1111b = 7fH → pcData[4] = 0x7f
6th column to 20th column: 0000 0000b = 00H → pcData[5] to pcData[19] = 0x00

·Mode 1: ASCII mode

The specified character strings are displayed in pcData[0] to pcData[3].

Available characters (ASCII code) are shown below.

×: character string specification not allowed

Bit		Upper four bits															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Lower four bits	0	×	×	SP	0	×	P	×	×	×	×	×	×	×	×	×	×
	1	×	×	×	1	A	Q	×	×	×	×	×	×	×	×	×	×
	2	×	×	×	2	B	R	×	×	×	×	×	×	×	×	×	×
	3	×	×	×	3	C	S	×	×	×	×	×	×	×	×	×	×
	4	×	×	×	4	D	T	×	×	×	×	×	×	×	×	×	×
	5	×	×	%	5	E	U	×	×	×	×	×	×	×	×	×	×
	6	×	×	×	6	F	V	×	×	×	×	×	×	×	×	×	×
	7	×	×	×	7	G	W	×	×	×	×	×	×	×	×	×	×
	8	×	×	×	8	H	X	×	×	×	×	×	×	×	×	×	×
	9	×	×	×	9	I	Y	×	×	×	×	×	×	×	×	×	×
	A	×	×	×	×	J	Z	×	×	×	×	×	×	×	×	×	×
	B	×	×	×	×	K	×	×	×	×	×	×	×	×	×	×	×
	C	×	×	×	×	L	×	×	×	×	×	×	×	×	×	×	×
	D	×	×	-	×	M	×	×	×	×	×	×	×	×	×	×	×
	E	×	×	.	×	N	×	×	×	×	×	×	×	×	×	×	×
	F	×	×	/	×	O	×	×	×	×	×	×	×	×	×	×	×

When a character other than above is specified, an error will be returned.

When the character strings are null-terminated, data after the NULL character are not displayed (blank). (They are displayed with left-aligned.)


Description

This function displays the value specified to the LED data (pcData) on the dot matrix LED according to the output mode (usLedMode).

Precautions

- To display data on the dot matrix LED, selecting 'USER' in the operation selection mode is required. (MELSEC iQ-R C Controller Module User's Manual)
- When checking an operation or checking the selected operation with the MODE/SELECT switch, an error occurs at the time of executing the CCPU_SetDotMatrixLED function even when "USER" is selected in the operation selection mode.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 100 CCPU_GetDotMatrixLED

CCPU_SetLEDStatus

This function sets the LED status of a C Controller module.

Format

short CCPU_SetLEDStatus(long lLed, unsigned short usLedInfo)

Argument

Argument	Name	Description	IN/OUT
lLed	Target LED	Specify the target LED. (When 'Reserved' is specified, this function ends normally without processing.) <ul style="list-style-type: none">• 0: USER LED• Others: Reserved	IN
usLedInfo	LED status information	Specify the LED status information.	IN


The specification method for the LED status information (usLedInfo) is as follows:

usLedInfo	Description
0	OFF
1	ON (Red)
2	Flashing at low speed (Red)
3	Flashing at high speed (Red)
4	ON (Green)
5	Flashing at low speed (Green)
6	Flashing at high speed (Green)

Description

This function controls the USER LED of C Controller module to the status specified to the LED status information (usLedInfo).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 105 CCPU_GetLEDStatus

CCPU_SetOpSelectMode

This function sets the operation selection mode of C Controller module.

Format

short CCPU_SetOpSelectMode(long IModelInfo, long ISelectMode)

Argument

Argument	Name	Description	IN/OUT
IModelInfo	Mode information	Specify the mode information.	IN
ISelectMode	Operation selection mode	Specify the operation selection mode.	IN


The specification method for the mode information (IModelInfo) and operation selection mode (ISelectMode) is as follows:

IModelInfo	ISelectMode	Description
1	1	Notifies event to the user program when holding the MODE/SELECT switch in the SELECT position.
	2	Unmounts SD memory card forcibly when holding the MODE/SELECT switch in the SELECT position.
	3	Unmounts USB Mass Storage Class-compliant devices forcibly when holding the MODE/SELECT switch in the SELECT position.
	4	Unmounts SD memory card/USB Mass Storage Class-compliant devices forcibly when holding the MODE/SELECT switch in the SELECT position.
	Others	Reserved
2	1	Displays the specified content on the dot matrix LED.
	2	Displays an error code on the dot matrix LED.
	3	Displays the IP address of CH1 on the dot matrix LED.
	4	Displays the IP address of CH2 on the dot matrix LED.
	Others	Reserved
Others	—	Reserved

Description

- This function sets the operation selection mode of C Controller module to the status specified to the operation selection mode (ISelectMode).
- The operation selection mode setting will be valid after this function is executed.
- If the operation selection mode is changed with both of this function and the switch operation, the mode set the last will be valid.
- An error occurs if this function is executed while the MODE/SELECT switch is being operated to select an operation.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 107 CCPU_GetOpSelectMode

CCPU_SetRTC

This function sets the clock data (local time) of C Controller module.

Format

short CCPU_SetRTC(short* psSetData)

Argument

Argument	Name	Description	IN/OUT
psSetData	Clock data	Specify the clock data (local time) to be set.	IN

- Specify the clock data (local time) to the clock data (psSetData) as follows.

(Available range: January 1, 1980 to December 31, 2079)

psSetData	Description
psSetData[0]	Year data (1980 to 2079)
psSetData[1]	Month data (1 to 12)
psSetData[2]	Day data (1 to 31)
psSetData[3]	Hour data (0 to 23)
psSetData[4]	Minute data (0 to 59)
psSetData[5]	Second data (0 to 59)


Description

- This function sets the clock data (local time) specified to the clock data (psSetData) to C Controller module.
- If the clock data (psSetData) is out of the range, an error is returned.
- Once the clock data (local time) is set, the history set to the event history is registered.
- When the daylight saving time function is enabled, an error is returned if clock data less than one hour from the start date and time of daylight saving time is set.

Precautions

- The clock data (local time) set with this function are not reflected to the clock of the operating system (VxWorks).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 109 CCPU_GetRTC

CCPU_ShutdownRom

This function shuts down the program memory and data memory of a C Controller module.


Format

short CCPU_ShutdownRom (void)


Argument

None

Description

- This function shuts down the program memory and data memory of a C Controller module. The BUS RUN LED starts flashing at high speed after the shutdown. (The shutdown status can be checked with the CCPU_GetCpuStatus function.)
- This function is used to shut down the program memory and data memory before turning the power of a C Controller module OFF. File operations (creating, deleting, and overwriting a file) on the program memory and data memory are disabled after the shutdown. However, reference to the program memory and data memory is possible. For details on the program memory and data memory, refer to the following:
( MELSEC iQ-R C Controller Module User's Manual)
- When calling this function, make sure to stop accessing files in the program memory and data memory, and close all files. Otherwise, data corruption or a file system error may occur.
- Always turn the power of the system OFF or reset the CPU module after checking that a shutdown is completed. If operation is continued, an error occurs when accessing files in the program memory and the data memory.
- Shutdown processing is performed in the order from the program memory to the data memory. If the program memory fails to be shut down, the data memory will not be shut down.
- When the program memory and data memory are in the shutdown completed status, this function ends normally without processing.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 116 CCPU_MountMemoryCard
- Page 142 CCPU_UnmountMemoryCard

CCPU_StartCCIEFBDataAssurance

This function starts data assurance for one link scan of CC-Link IE Field Network Basic.

Format

short CCPU_StartCCIEFBDataAssurance (unsigned short usGroupNo, unsigned long ulTimeout)


Argument

Argument	Name	Description	IN/OUT
usGroupNo	Group No.	Specify a group number to start data assurance. (1 to 4)	IN
ulTimeout	Timeout value	Specify a timeout time until data assurance is started in milliseconds. (When FFFFFFFFH is specified, the function waits for an event infinitely.)	IN

Description

- This function starts data assurance for one link scan of CC-Link IE Field Network Basic for the specified group.
- When the CCPU_StartCCIEFBDataAssurance function is executed, the cyclic transmission of CC-Link IE Field Network Basic is stopped.
- During the link refresh of CC-Link IE Field Network Basic, the module waits for the completion of link refresh (until the timeout value (ulTimeout) elapses).
- Data assurance can be ended using the CCPU_EndCCIEFBDataAssurance function only for the thread where the data assurance has been started using the CCPU_StartCCIEFBDataAssurance function. When the CCPU_StartCCIEFBDataAssurance function is executed in multiple threads, the function executed later waits for the completion of the previously executed data assurance function (waits until the timeout value (ulTimeout) reaches the time set with the CCPU_StartCCIEFBDataAssurance function).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 84 CCPU_EndCCIEFBDataAssurance

CCPU_StartWDT

This function sets and starts the user WDT of a C Controller module.

Format

short CCPU_StartWDT(short sType, short sInterval)


Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (If reserve is specified, an error is returned.) • 0: User WDT • Others: Reserved	IN
sInterval	WDT interval	Specify the interval of WDT by 10 milliseconds. (Available range is between 10 to 1000 (100 to 10000 [ms]).)	IN

Description

- The user WDT is the timer for detecting a hardware failure or program error.
- This function sets an interval of the WDT to the WDT interval (sInterval) × 10 ms and starts the user WDT.
- If the user WDT is not reset periodically within the set time (by execution of the CCPU_ResetWDT function), a user WDT error will occur. When a user WDT error occurs, a C Controller module will be in the stop error state. (The BUS RUN LED turns OFF, and the ERROR LED starts flashing.)
- When this function is executed while the WDT is running, an error will be returned.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 125 CCPU_ResetWDT
- Page 136 CCPU_StopWDT
- Page 90 CCPU_EntryWDTInt

CCPU_StopWDT

This function stops the user WDT of a C Controller module.

Format

short CCPU_StopWDT(short sType)


Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (If reserve is specified, an error is returned.) <ul style="list-style-type: none">• 0: User WDT• Others: Reserved	IN

Description

- This function stops the user WDT.
- When this function is executed without starting the user WDT, it ends normally.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 135 CCPU_StartWDT
- Page 125 CCPU_ResetWDT
- Page 90 CCPU_EntryWDTInt

CCPU_SysClkRateGet

This function reads the system clock rate specified with the CCPU_SysClkRateSet function from the backup RAM.

Format

```
short CCPU_SysClkRateGet(short* psTicks)
```

Argument

Argument	Name	Description	IN/OUT
psTicks	Clock rate	Stores the system clock rate in the unit of clock frequency (Hz) per one second. • 0: Default value (60 Hz) • 60 to 1000: Specified clock rate value	OUT

Description


- This function reads the system clock rate specified with the CCPU_SysClkRateSet function from the backup RAM.
- When a C Controller module is initialized, the system clock rate is set to the default value (60 Hz).

Precautions

The read value may not correspond to the system clock rate in operation.

To check the system clock rate in operation, use the sysClkRateGet function of VxWorks.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 138 CCPU_SysClkRateSet

CCPU_SysClkRateSet

This function saves the specified system clock rate in the backup RAM.

Format

short CCPU_SysClkRateSet(short sTicks, short* psRestart)

Argument

Argument	Name	Description	IN/OUT
sTicks	Clock rate	Specify the system clock rate in the unit of clock frequency (Hz) per one second. <ul style="list-style-type: none">• 0: Default value (60 Hz)• 60 to 1000: Specified clock rate value	IN
psRestart	Restart necessity flag	Stores the necessity to restart a C Controller module after the execution of this function. (When 'NULL' is specified, the restart necessity flag is not stored.) <ul style="list-style-type: none">• 0: Restart is not required. (C Controller module has already been running at the specified clock rate.)• 1: Restart is required. (C Controller module operates at the specified clock rate after restarting it.)	OUT


Description

- This function saves the system clock rate specified to the clock rate (sTicks) in the backup RAM.
The specified system clock rate will be enabled after restarting a C Controller module.
- When the output to the restart necessity flag (psRestart) is '0' (restart is not required), continue the application processing.
- When the output to the restart necessity flag (psRestart) is '1' (restart is required), stop the application processing, and reset the C Controller module or turn the power OFF and ON.
- For details on the system clock rate, refer to the manual for VxWorks.
- When a C Controller module is initialized, the system clock rate is set to the default value (60 Hz).

Precautions

- Execute this function only once after a C Controller module is started.
If this function is executed by specifying the same clock rate value as the first time, the restart necessity flag (psRestart) will be '0' (restart is not required) regardless of the system clock rate value in operation.
- Use this function to change the system clock rate.
If the sysClkRateSet function of VxWorks is used, the operation of VxWorks will be unstable.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 137 CCPU_SysClkRateGet

CCPU_ToBuf

This function writes data to the CPU buffer memory of the CPU module (host CPU) and the buffer memory of the intelligent function module which are mounted on the specified module position. (TO instruction)

Format

short CCPU_ToBuf (unsigned short usloNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)


Argument

Argument	Name	Description	IN/OUT
usloNo	Module position	Specify the module position as follows. For the CPU buffer memory, only the host CPU can be accessed. Start I/O number divided by 16 (0H to FFH, 3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN
ulBufSize	Data storage destination size	Specify '0'.	IN

Description

- This function writes data in the data storage destination (pusDataBuf) for the size specified to the data size (ulSize) to the CPU buffer memory of a CPU module (host CPU) and the buffer memory of an intelligent function module which are specified to the module position (usloNo).
Data is written by specifying an offset address from the start of the CPU buffer memory of a CPU module (host CPU) and the buffer memory of an intelligent function module to the offset (ulOffset).
- To access the CPU buffer memory (host CPU) of the module in a multiple CPU system (CPU No.1 to No.4), specify 3E0H (CPU No.1) to 3E3H (CPU No.4) to the module position (usloNo). However, the CPU buffer memory (host CPU) can be accessed only when the multiple CPU setting is configured.
- When executing this function while the operating status of a CPU module is not RUN, the STOP/PAUSE error (-28640) occurs.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 91 CCPU_FromBuf

CCPU_ToBufHG

This function writes data to the fixed cycle communication area of the CPU module mounted on the specified module position.

Format

short CCPU_ToBufHG(unsigned short usIoNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)


Argument

Argument	Name	Description	IN/OUT
usIoNo	Module position	Specify the module position as follows. Start I/O number divided by 16 (3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN
ulBufSize	Data storage destination size	Specify '0'.	IN

Description

- This function writes data in the data storage destination (pusDataBuf) for the size specified to the data size (ulSize) to the fixed cycle communication area of a CPU module specified to the module position (usIoNo). Data is written by specifying an offset address from the start of the fixed cycle communication area to the offset (ulOffset).
- The fixed cycle communication area can be accessed only when the fixed cycle communication area setting under the multiple CPU setting is configured.
- When executing this function while the operating status of a CPU module is not RUN, the STOP/PAUSE error (-28640) occurs.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 91 CCPU_FromBuf
- Page 139 CCPU_ToBuf
- Page 92 CCPU_FromBufHG

CCPU_UnlockFWUpdate

This function removes the prohibition on the firmware update of a C Controller module.

Format

short CCPU_UnlockFWUpdate(char* pcPass)


Argument

Argument	Name	Description	IN/OUT
pcPass	Password	Specify a password for removing the prohibition on the firmware update.	IN

Description

- This function removes the prohibition on the firmware update of a C Controller module.
- If a password set to 'pcPass' is incorrect, an error will be returned.
- When the password is forgotten, initialize the C Controller module.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 115 CCPU_LockFWUpdate

CCPU_UnmountMemoryCard

This function unmounts the SD memory card and USB Mass Storage Class-compliant device inserted to a C Controller module.

Format

short CCPU_UnmountMemoryCard (short sDrive)

Argument

Argument	Name	Description	IN/OUT
sDrive	Target drive	Specify a target drive. (When 'Reserved' is specified, this function ends normally without processing.) <ul style="list-style-type: none">• 1: SD memory card• 2: USB Mass Storage Class-compliant device• Others: Reserved	IN


Description

- This function unmounts the drive specified to the target drive (sDrive).
- The CARD RDY LED keeps flashing during the unmount processing of the SD memory card, and it turns OFF after the unmount processing is completed.
- The USB RDY LED keeps flashing during the unmount processing of the USB Mass Storage Class-compliant device, and it turns OFF after the unmount processing is completed.
- This function can be executed when the status of the drive specified to the target drive (sDrive) is "Inserted (mounted)". (The status can be checked with the CCPU_GetCpuStatus function.)
- When the drive specified to the target drive (sDrive) has already been unmounted, this function ends normally without processing.

Precautions

Design a program so that accessing files in the target drive is stopped and all files are closed before calling this function. If this function is called while the files are open, data corruption or a file system error may occur.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 116 CCPU_MountMemoryCard
- Page 98 CCPU_GetCpuStatus

CCPU_WaitEvent

This function waits for an interrupt event notification from other CPUs.

Format

short CCPU_WaitEvent (short* psEvent, unsigned long ulTimeout, short* psSetEventNo)

Argument

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

- The specification method for the interrupt event setting (psEvent) is as follows:

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

- The following value is stored in the occurred event (psSetEventNo).

psSetEventNo	Description
psSetEventNo[0]	CPU number of the notified interrupt event
psSetEventNo[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 (psEvent) for the time specified to the timeout value (ulTimeout).
- 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 this function is called, this function immediately ends normally. When a reset operation is performed, any interrupt event that occurred prior to reset is discarded.
- If multiple interrupt events have been notified for the same event number (interrupt pointer number) at a time when this function is called, it is notified as a single interrupt event.
- Set the event number (interrupt pointer number) without duplication. Otherwise, an error will be returned.
- The specified timeout value is rounded to the tick unit. Specify a timeout value of one tick or more.
- Specify the programmable controller CPU or C Controller module to the CPU number. Otherwise, an error will be returned.
- Design a program so that this function is not called simultaneously by specifying the same event number (interrupt pointer number) from multiple tasks. Otherwise, the execution of the interrupt event notified task is unpredictable.

Ex.

Setting of psEvent to wait for interrupt event 0 and 1 for CPU No.1, and interrupt event 10 for CPU No.2


```
psEvent[0] = 3;  
psEvent[1] = 1;  
psEvent[2] = 0;  
psEvent[3] = 1;  
psEvent[4] = 1;  
psEvent[5] = 2;  
psEvent[6] = 10;
```

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

Precautions

Do not set the clock data of a C Controller module while executing this function. Otherwise, this function does not operate properly. (This function may not be completed.)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 147 CCPU_WaitUnitEvent

CCPU_WaitSwitchEvent

This function waits for a switch interrupt event of C Controller module to occur.

Format

short CCPU_WaitSwitchEvent(short sSwitch, unsigned long ulTimeout)

Argument

Argument	Name	Description	IN/OUT
sSwitch	Switch interrupt event type	Specify the switch interrupt event type. • 0: RUN switch interrupt event • 1: STOP switch interrupt event • 2: SELECT switch interrupt event	IN
ulTimeout	Timeout	Specify the timeout value in milliseconds (0H to FFFFFFFFH). (When FFFFFFFFH is specified, the function waits for an event infinitely.)	IN


Description

- This function waits for a switch interrupt event specified to the switch interrupt event type (sSwitch).
- If an interrupt event has already been notified at a time when this function is called, this function immediately ends normally.
- If the same switch interrupt event has been notified several times at a time when this function is called, the user program executes processing as a single switch interrupt event notification.
- The specified timeout value is rounded to the tick unit. Specify a timeout value of one tick or more.

Precautions

- To issue the switch interrupt event by holding the MODE/SELECT switch in the SELECT position, selecting "EVENT" in the operation selection mode is required. (IMELSEC iQ-R C Controller Module User's Manual)
- For the SELECT switch interrupt event, an event issuance status cannot be judged from the appearance. To check the issued status of SELECT switch interrupt event, implement the processing such as receiving a switch interrupt event using this function and making the USER LED turn ON.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

CCPU_WaitTimerEvent

This function waits for a timer event to occur.

Format

short CCPU_WaitTimerEvent (long IEventNo)

Argument

Argument	Name	Description	IN/OUT
IEventNo	Timer event number	Specify a timer event number that waits for a timer event to occur. (1 to 16)	IN

Description


- This function waits for a timer event specified to the timer event number (IEventNo) to occur.
- The occurrence cycle of the timer every number (1 to 16) can be set, changed, or cleared by the CCPU_EntryTimerEvent function.
- When reset operation is performed, any event that has occurred prior to reset is discarded.
- Using this function enables a cycle timer task. However, even though an event occurs, the waiting task may not be operated immediately due to the system status (such as the interrupt).
- If waiting for an event with this function to a cleared timer event, the wait status will not be cleared until an event occurs after the registration of the event (and the specified cycle has elapsed) with CCPU_EntryTimerEvent function.

Precautions

Note that operation of waiting for event (function completion) using this function will vary. This operation variation depends on the specified value of synchronization type of the timer event number with the CCPU_EntryTimerEvent function.

- When the synchronization type is batch synchronization, the wait state of all tasks that are waiting for an event is canceled. However, if there is no task in the wait state at event occurrence, the wait state will not be canceled even if the CCPU_WaitTimerEvent function is called.
- When the synchronization type is individual synchronization, the wait state of one task in the tasks that are waiting for an event is canceled. If multiple tasks are waiting for the same event, the wait state will be canceled in order of priority of a task (in order of execution of wait when the priority is same). If there is no task in the wait state at the time of event occurrence, the wait state will be canceled when the CCPU_WaitTimerEvent function is called later.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 88 CCPU_EntryTimerEvent

CCPU_WaitUnitEvent

This function waits for an interrupt event notification from modules.

Format

short CCPU_WaitUnitEvent (short* psEvent, unsigned long ulTimeout, short* psSetEventNo)

Argument

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

- The specification method for the event setting (psEvent) is as follows:

psEvent	Description
psEvent[0]	Number of interrupt event settings (1 to 64)
psEvent[1]	Interrupt pointer number of the first interrupt event (0 to 15, 50 to 1023)
psEvent[2]	Interrupt pointer number of the second interrupt event (0 to 15, 50 to 1023)
psEvent[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 (psEvent) for the time specified to the timeout value (ulTimeout).
- 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 this function is called, this function immediately ends normally. When a reset operation is performed, any interrupt event that occurred prior to reset is discarded.
- If multiple interrupt events have been notified for the same event number (interrupt pointer number) at a time when this function is called, it is notified as a single interrupt event.
- Set the event number (interrupt pointer number) without duplication. Otherwise, an error will be returned.
- The specified timeout value is rounded to tick unit. Specify a timeout value of one tick or more.
- Design a program so that this function is not called simultaneously by specifying the same interrupt event (interrupt pointer number) from multiple tasks. Otherwise, the execution of the interrupt event notified task is unpredictable.
- 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 (psSetEventNo).
- An interrupt event is not notified while a stop error is occurring in a C Controller module.

Ex.

Setting of psEvent when waiting for interrupt event 0, interrupt event 1, interrupt event 50, and interrupt event 51

```
psEvent[0] = 4;
```

```
psEvent[1] = 0;
```

```
psEvent[2] = 1;
```

```
psEvent[3] = 50;
```

```
psEvent[4] = 51;
```

When event 51 occurs, 51 is returned to psSetEventNo.


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 C Controller module while executing this function. Otherwise, this function does not operate properly. (This function may not be completed.)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 143 CCPU_WaitEvent


CCPU_WriteDevice

This function writes data to internal user devices and internal system devices of C Controller module.

Format

short CCPU_WriteDevice (short sDevType, unsigned long ulDevNo, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)


Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number. (Only multiples of 16 can be specified for bit devices.)	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN
ulBufSize	Data storage destination size	Specify '0'.	IN

Description

This function writes data in the data storage destination (pusDataBuf) for the size specified to the data size (ulSize) to a device after the one specified to the device type (sDevType) and the start device number (ulDevNo).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 117 CCPU_ReadDevice


CCPU_WriteLinkDevice

This function writes data to the own station link devices of the following modules: CC-Link IE Controller Network module, CC-Link IE Field Network module, Ethernet module (when CC-Link IE Field Network is selected), MELSECNET/H network module, and CC-Link IE TSN module which are controlled by a C Controller module.

Format

short CCPU_WriteLinkDevice (unsigned short usIoNo, short sDevType, unsigned long ulDevNo, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

Argument

Argument	Name	Description	IN/OUT
usIoNo	Module position	Specify the module position as follows. Start I/O number divided by 16 (0H to FFH)	IN
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number. (Only multiples of 16 can be specified for bit devices.)	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN
ulBufSize	Data storage destination size	Specify '0'.	IN


Description

- This function writes data to subsequent devices after the device, which is specified in the device type (sDevType) and the start device number (ulDevNo) of the following modules specified in the module position (usIoNo): a CC-Link IE Controller Network module, CC-Link IE Field Network module, Ethernet module (when CC-Link IE Field Network is selected), MELSECNET/H network module, and CC-Link IE TSN module. The function writes data in the data storage destination (pusDataBuf) for the size specified in the data size (ulSize).

Precautions

- To access a motion module using CC-Link IE TSN, specify link devices (SB and SW). Specifying link devices other than SB and SW will cause an error.
- To access link devices of a network module controlled by another CPU, access to another CPU via bus interface using the MELSEC data link function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 118 CCPU_ReadLinkDevice

CCPU_WriteMCUnitLabel

This function writes data to module labels of a C Controller module in word units.

Format

short CCPU_WriteMCUnitLabel (unsigned long ulUnitLabel, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize, unsigned long ulUnitLabelID)

Argument

Argument	Name	Description	IN/OUT
ulUnitLabel	Module label	Specify a module label.	IN
ulOffset	Offset	Specify the offset from the specified module label in word units.	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN
ulBufSize	Data storage destination size	Specify '0'.	IN
ulUnitLabelID	Module label ID	Specify a module label ID.	IN


Description

- This function writes data, which is stored in the data storage destination (pusDataBuf), to the module label of a C Controller module, which is specified for the module label (ulUnitLabel), for the size specified in the data size (ulSize).
- To use the CCPU_WriteMCUnitLabel function, include a header file that has the information of module labels output by CW Configurator, and specify a module label.

■ Specification method for a module label

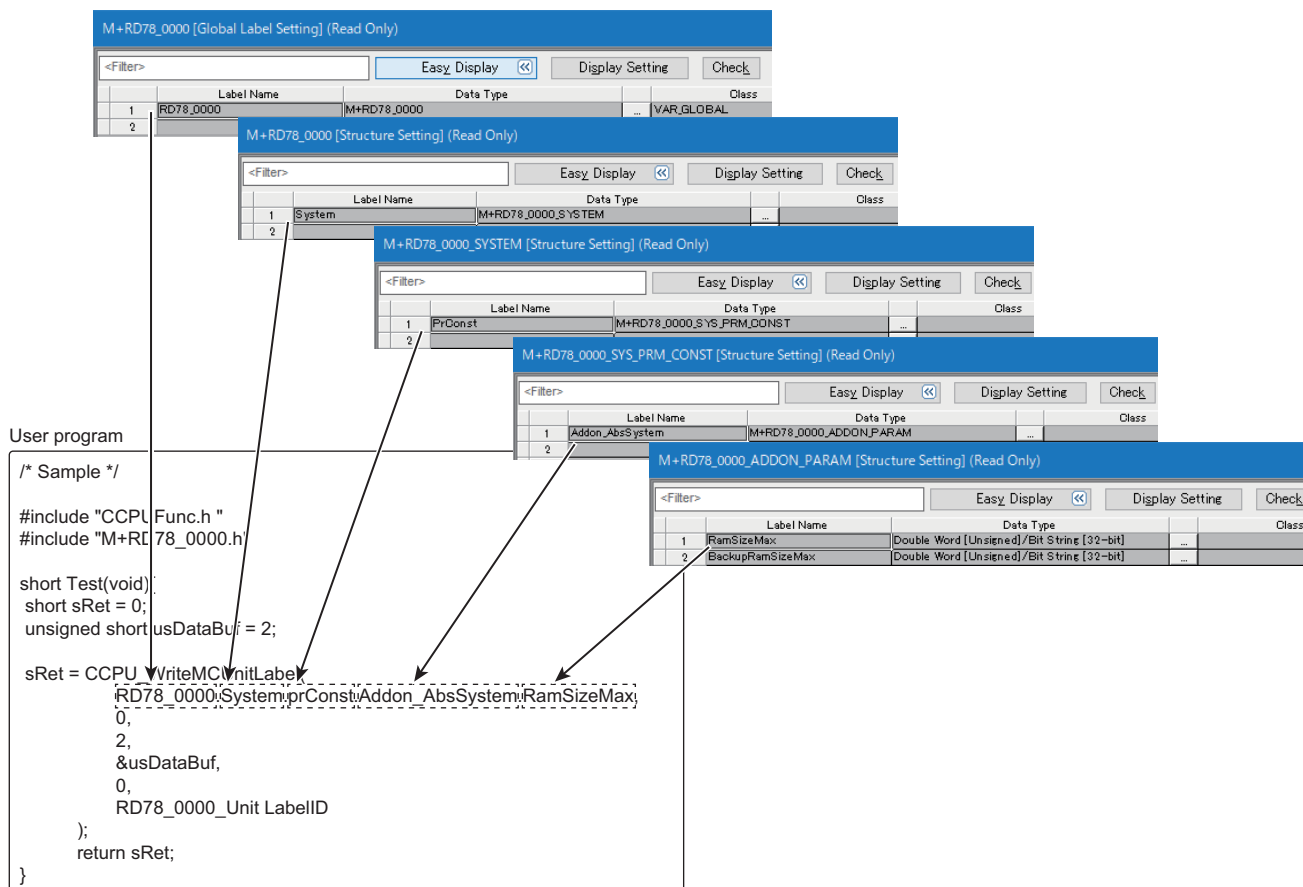
- For the module label (ulUnitLabel), specify a module label other than bit type. Specifying a bit type module label may cause an unpredictable operation.
- To specify an array label for the module label (ulUnitLabel) and write data to each array element, specify an offset for the offset (ulOffset) in word units. For an offset value to be specified, calculate from a label data type and an array element number that are specified for the module label (ulUnitLabel).

Offset value = the number of words corresponded to a label data type × an array element number

The number of words varies depending on the label data type. For each label size, refer to the 'Specification method for data size.' ( Page 153 Specification method for data size)

- For the module label (ulUnitLabel), specify a module label which is set in CW Configurator in the structure format.

The following shows an example for specifying a module label.



The diagram illustrates the configuration of module labels for a user program. It shows four configuration windows for different module labels, each with a table of settings. Arrows indicate the mapping from the user program's function call to the specific labels defined in these windows.

User program

```

/* Sample */
#include "CCPU_Func.h"
#include "M+RD78_0000.h"

short Test(void)
{
    short sRet = 0;
    unsigned short usDataBuf = 2;

    sRet = CCPU_WriteMCUnitLabel(
        RD78_0000_System, prConst, Addon_AbsSystem, RamSizeMax,
        0,
        2,
        &usDataBuf,
        0,
        RD78_0000_UnitLabelID
    );
    return sRet;
}
    
```

M+RD78_0000 [Global Label Setting] (Read Only)

Label Name	Data Type	Class
1 RD78_0000	M+RD78_0000	VAR_GLOBAL
2		

M+RD78_0000 [Structure Setting] (Read Only)

Label Name	Data Type	Class
1 System	M+RD78_0000_SYSTEM	
2		

M+RD78_0000_SYSTEM [Structure Setting] (Read Only)

Label Name	Data Type	Class
1 PrConst	M+RD78_0000_SYS_PRM_CONST	
2		

M+RD78_0000_SYS_PRM_CONST [Structure Setting] (Read Only)

Label Name	Data Type	Class
1 Addon_AbsSystem	M+RD78_0000_ADDON_PARAM	
2		

M+RD78_0000_ADDON_PARAM [Structure Setting] (Read Only)

Label Name	Data Type	Class
1 RamSizeMax	Double Word [Unsigned]/Bit String [32-bit]	
2 BackupRamSizeMax	Double Word [Unsigned]/Bit String [32-bit]	

■ Specification method for data size

- For the data size (ulSize), specify the number of words according to the label data type.

The following shows an example for specifying data size.

Label data type	Number of words
Word (signed, unsigned)	1
Double words (signed, unsigned)	2
Single precision real number	2
Double precision real number	4
Time*1	2

- *1 Different from the time type in a programmable controller CPU, it cannot be written or read in the 'T#23d23h59m59s999ms' format.
 To write data, specify double-words [signed] (in ms unit).
 When data is read, the data is read in double-words [signed] (in ms unit).
 A value (ms) to be specified can be calculated by the following calculation formula:
 · (d (date) specified value × 86,400,000) + (h (hour) specified value × 3,600,000) + (m (minute) specified value × 60,000) + (s (second) specified value × 1,000) + ms (millisecond) specified value)
 Example: To write 'T#1h30m' (3,780,000 (ms)), the calculation formula is as follows:
 · 3,780,000 (ms) = (0 × 86,400,000) + (1 × 3,600,000) + (30 × 60,000) + (0 × 1,000) + 0

■ Specification method for a module label ID

- For the module label ID (ulUnitLabelID), specify a macro defined in the header file, which includes the information of module labels output by CW Configurator.
- When specifying '0' for the module label ID (ulUnitLabelID), data will be read from the module label without checking the module label ID.

The following shows an example for specifying a module label ID.

Header File

```
/* M+RD78_0000 */
#define RD78_0000_UnitLabelID 1213UL

typedef struct{
  struct{
    struct{
      unsigned long RamSizeMax;
      unsigned long BackupRamSizeMax;
    } Addon_AbsSystem;
  } PrConst;
} System;
} RD78_0000_TAG;

static const RD78_0000_TAG RD78_0000 = {{{{0x100000, 0x100020}}}};
```

User Program

```
/* Sample */
#include "CCPUFunc.h"
#include "M+RD78_0000.h"

short Test(void){
  short sRet = 0;
  unsigned short usDataBuf = 2;

  sRet = CCPU_WriteModuleUnitLabel(
    RD78_0000_System.prConst.Addon_AbsSystem.RamSizeMax,
    0,
    2,
    &usDataBuf,
    0,
    RD78_0000_UnitLabelID);
  return sRet;
}
```


(1) Defined macro

(2) Module label ID (ulUnitLabelID)

Precautions

- When specifying '0' for the module label ID (ulUnitLabelID), this function does not check the address in the label area where the public labels (module labels), which are written to a C Controller module, and the module labels, which are specified in a function for accessing module labels, are assigned. Therefore, an unpredictable operation may occur.
- For the module label ID (ulUnitLabelID), one module label ID is defined for a system where a C Controller module controls a motion module. Therefore, when changing a public label (module label) of motion module, the headers of all the motion modules must be updated.
- When writing data to module labels, data is written to the module label (ulUnitLabel) and offset (ulOffset) for the size specified in the size (ulSize). However, this function does not check whether the data is within the specified module label. Therefore, data may be written to module labels other than the specified ones depending on the offset (ulOffset) and size (ulSize). (For the offset (ulOffset) and size (ulSize) check, this function checks only whether the sum of module label (ulUnitLabel), offset (ulOffset) and data size (ulSize) is exceeding the module label area size (for all modules).)

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 119 CCPU_ReadMCUnitLabel

CCPU_WriteMCUnitLabelBit

This function writes module labels of a C Controller module in bit units.

Format

short CCPU_WriteMCUnitLabelBit (unsigned long ulUnitLabel, unsigned short usDataBuf, unsigned long ulUnitLabelID)

Argument

Argument	Name	Description	IN/OUT
ulUnitLabel	Module label	Specify a module label.	IN
usDataBuf	Data storage destination	Specify the storage destination of write data.	IN
ulUnitLabelID	Module label ID	Specify a module label ID.	IN


Description

- This function writes data, which is stored in the data storage destination (usDataBuf), to the module label of a C Controller module specified in the module label (ulUnitLabel).
- To use the CCPU_WriteMCUnitLabelBit function, include a header file that has the information of module labels output by CW Configurator, and specify a module label.
- Specify 0 (OFF) or 1 (ON) for the data storage destination (usDataBuf). When a value other than 0 (OFF) or 1 (ON) is specified, only a specified value on the 0th bit of data storage destination (usDataBuf) is enabled. (Values of bit 1 to 15 are ignored.)


■Specifying a module label

- For the module label (ulUnitLabel), specify a bit type module label. Specifying a module label other than bit type may cause an unpredictable operation.

■Specifying a module label ID

- For the module label ID (ulUnitLabelID), specify a macro defined in the header file, which includes the information of module labels output by CW Configurator. For an example of specifying a module label ID, refer to 'Specification method for a module label ID' in the 'CCPU_WriteMCUnitLabel function' section. ( Page 153 Specification method for a module label ID)
- When specifying '0' for the module label ID (ulUnitLabelID), data will be read from the module label without checking the module label ID.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 121 CCPU_ReadMCUnitLabelBit
- Page 151 CCPU_WriteMCUnitLabel

CCPU_X_In_BitEx

This function reads an input signal (X) in bit (1-point) units.

Format

short CCPU_X_In_BitEx (short sFlg, unsigned short usXNo, unsigned short* pusData)


Argument

Argument	Name	Description	IN/OUT
sFlg	Access flag	Specify an access flag. <ul style="list-style-type: none">• 0: Normal access• Others: Reserved	IN
usXNo	Input signal	Specify an input signal (X).	IN
pusData	Data storage destination	Specify the storage destination of read data. Either of the following values is stored depending on the value of the input signal (X). <ul style="list-style-type: none">• 0: OFF• 1: ON	OUT

Description

- This function reads an input signal (X) specified to the input signal (usXNo) in bit (1-point) units.
- A read value of an input signal (X) is stored in the data storage destination (pusData).
- This function operates to the mounted module corresponding to the specified input signal (usXNo) regardless of the type of the parameter setting (I/O assignment). When the specified area is "empty", this function ends normally without processing (read data: 0). When it is "output module", the I/O assignment error occurs.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 157 CCPU_X_In_WordEx
- Page 160 CCPU_Y_Out_BitEx
- Page 161 CCPU_Y_Out_WordEx
- Page 158 CCPU_Y_In_BitEx
- Page 159 CCPU_Y_In_WordEx

CCPU_X_In_WordEx

This function reads an input signal (X) in word (16-point) units.

Format

short CCPU_X_In_WordEx (short sFlg, unsigned short usXNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

Argument

Argument	Name	Description	IN/OUT
sFlg	Access flag	Specify an access flag. • 0: Normal access • Others: Reserved	IN
usXNo	Start input signal	Specify a start input signal (X). (Specify a multiple of 16.)	IN
usSize	Read data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
usBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN

Description


- This function reads an input signal (X) for the size specified to the read data size (usSize) from the start input signal (X) specified to the start input signal (usXNo), and stores it in the data storage destination (pusDataBuf).
- Specify an area size of the data storage destination (pusDataBuf) to the data storage destination size (usBufSize).
- This function operates to the mounted module corresponding to the specified input signal (usXNo) regardless of the type of the parameter setting (I/O assignment). When the specified area is "empty", this function ends normally without processing (read data: 0). When it is "output module", the I/O assignment error occurs.
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

pusDataBuf	Description
pusDataBuf[0]	Data of usXNo+FH to usXNo
pusDataBuf[1]	Data of usXNo+1FH to usXNo+10H
⋮	⋮
pusDataBuf[usSize-1]	Data of usXNo+(usSize-1)×16+FH to usXNo+(usSize-1)×16

Precautions

Note that the size of data storage destination (usBufSize) should be equal to or bigger than the read data size (usSize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 156 CCPU_X_In_BitEx
- Page 160 CCPU_Y_Out_BitEx
- Page 161 CCPU_Y_Out_WordEx
- Page 158 CCPU_Y_In_BitEx
- Page 159 CCPU_Y_In_WordEx

CCPU_Y_In_BitEx

This function reads an output signal (Y) in bit (1-point) units.

Format

short CCPU_Y_In_BitEx (short sFlg, unsigned short usYNo, unsigned short* pusData)


Argument

Argument	Name	Description	IN/OUT
sFlg	Access flag	Specify an access flag. <ul style="list-style-type: none">• 0: Normal access• Others: Reserved	IN
usYNo	Output signal	Specify an output signal (Y).	IN
pusData	Data storage destination	Specify the storage destination of read data. Either of the following values is stored depending on the value of the output signal (Y). <ul style="list-style-type: none">• 0: OFF• 1: ON	OUT

Description

- This function reads an output signal (Y) specified to the output signal (usYNo) in bit (1-point) units.
- A read value of an output signal (Y) is stored in the data storage destination (pusData).
- This function operates to the mounted module corresponding to the specified output signal (usYNo) regardless of the type of the parameter setting (I/O assignment). When the specified area is "empty", this function ends normally without processing (read data: 0). When it is "input module", the I/O assignment error occurs.
- No error will occur even if this function is executed when the operating status of a CPU module is STOP or PAUSE. An output signal (Y) is read at execution of this function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 156 CCPU_X_In_BitEx
- Page 157 CCPU_X_In_WordEx
- Page 160 CCPU_Y_Out_BitEx
- Page 161 CCPU_Y_Out_WordEx
- Page 159 CCPU_Y_In_WordEx

CCPU_Y_In_WordEx

This function reads an output signal (Y) in word (16-point) units.

Format

short CCPU_Y_In_WordEx (short sFlg, unsigned short usYNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

Argument

Argument	Name	Description	IN/OUT
sFlg	Access flag	Specify an access flag. • 0: Normal access • Others: Reserved	IN
usYNo	Start output signal	Specify a start output signal (Y). (Specify a multiple of 16.)	IN
usSize	Read data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT
usBufSize	Data storage destination size	Specify the data storage destination size in word units.	IN

Description


- This function reads an output signal (Y) for the size specified to the read data size (usSize) from the start output signal (Y) specified to the start output signal (usYNo), and stores it in the data storage destination (pusDataBuf).
- Specify an area size of the data storage destination (pusDataBuf) to the data storage destination size (usBufSize).
- This function operates to the mounted module corresponding to the specified output signal (usYNo) regardless of the type of the parameter setting (I/O assignment). When the specified area is "empty", this function ends normally without processing (read data: 0). When it is "input module", the I/O assignment error occurs.
- No error will occur even if this function is executed when the operating status of a CPU module is STOP or PAUSE. An output signal (Y) is read at execution of this function.
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

pusDataBuf	Description
pusDataBuf[0]	Data of usYNo+FH to usYNo
pusDataBuf[1]	Data of usYNo+1FH to usYNo+10H
⋮	⋮
pusDataBuf[usSize-1]	Data of usYNo+(usSize-1)×16+FH to usYNo+(usSize-1)×16

Precautions

Note that the size of data storage destination (usBufSize) should be equal to or bigger than the read data size (usSize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 156 CCPU_X_In_BitEx
- Page 157 CCPU_X_In_WordEx
- Page 160 CCPU_Y_Out_BitEx
- Page 161 CCPU_Y_Out_WordEx
- Page 158 CCPU_Y_In_BitEx

CCPU_Y_Out_BitEx

This function outputs an output signal (Y) in bit (1-point) units.

Format

short CCPU_Y_Out_BitEx (short sFlg, unsigned short usYNo, unsigned short usData)


Argument

Argument	Name	Description	IN/OUT
sFlg	Access flag	Specify an access flag. • 0: Normal access • Others: Reserved	IN
usYNo	Output signal	Specify an output signal (Y).	IN
usData	Data storage destination	Specify the storage destination of output data. (Specify the value of bit 0.) • 0: OFF • 1: ON	IN

Description

- This function outputs (turns ON/OFF) an output signal (Y) specified to the output signal (usYNo) in bit (1-point) units.
- Specify 0 (OFF) or 1 (ON) for the data storage destination (usData). When a value other than 0 (OFF) or 1 (ON) is specified, only a specified value on the 0th bit of data storage destination (usData) is enabled. (Values of bit 1 to 15 are ignored.)
- When executing this function while the operating status of a CPU module is not RUN, the STOP/PAUSE error occurs.
- When this function is executed to "input module", the I/O assignment error occurs.
- Do not specify an output module controlled by other CPUs to the output signal (usYNo).
If it is specified, no operation is performed to the output module.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 156 CCPU_X_In_BitEx
- Page 157 CCPU_X_In_WordEx
- Page 161 CCPU_Y_Out_WordEx
- Page 158 CCPU_Y_In_BitEx
- Page 159 CCPU_Y_In_WordEx

CCPU_Y_Out_WordEx

This function outputs an output signal (Y) in word (16-point) units.

Format

short CCPU_Y_Out_WordEx (short sFlg, unsigned short usYNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

Argument


Argument	Name	Description	IN/OUT
sFlg	Access flag	Specify an access flag. • 0: Normal access • Others: Reserved	IN
usYNo	Start output signal	Specify a start output signal (Y). (Specify a multiple of 16.)	IN
usSize	Output size	Specify the output size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of output data.	IN
usBufSize	Data storage destination size	Specify '0'.	IN

Description

- This function outputs (turns ON/OFF) data in the data storage destination (pusDataBuf) from a start output signal (Y) specified to the start output signal (usYNo) to an output signal (Y) for the size specified to the data size (usSize).
- When executing this function while the operating status of a CPU module is not RUN, the STOP/PAUSE error occurs.
- When this function is executed to "input module", the I/O assignment error occurs.
- Do not specify an output module controlled by other CPUs to the output signal (usYNo).
If it is specified, no operation is performed to the output module.
- Store output data in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

pusDataBuf	Description
pusDataBuf[0]	Data of usYNo+FH to usYNo
pusDataBuf[1]	Data of usYNo+1FH to usYNo+10H
⋮	⋮
pusDataBuf[usSize-1]	Data of usYNo+(usSize-1)×16+FH to usYNo+(usSize-1)×16

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 156 CCPU_X_In_BitEx
- Page 157 CCPU_X_In_WordEx
- Page 160 CCPU_Y_Out_BitEx
- Page 158 CCPU_Y_In_BitEx
- Page 159 CCPU_Y_In_WordEx

3.2 C Controller module dedicated functions for ISR

CCPU_DisableInt_ISR

This function disables the routine registered with the CCPU_EntryInt function.

Format

short CCPU_DisableInt (short sSINo)

Argument

Argument	Name	Description	IN/OUT
sSINo	Interrupt pointer number	Specify the interrupt pointer number.	IN

Description

- This function disables the routine registered with the CCPU_EntryInt function. (The routine is not executed when an interrupt occurs.)
- Specify the interrupt pointer number (sSINo) specified in the CCPU_EntryInt function to Interrupt pointer number (sSINo).

WARNING

If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 86 CCPU_EntryInt
- Page 163 CCPU_EnableInt_ISR

CCPU_EnableInt_ISR

This function enables the routine registered with the CCPU_EntryInt function.

Format

short CCPU_EnableInt (short sSINo)

Argument

Argument	Name	Description	IN/OUT
sSINo	Interrupt pointer number	Specify the interrupt pointer number.	IN

Description

- This function enables the routine registered with the CCPU_EntryInt function. (The routine is executed when an interrupt occurs.)
- Specify the interrupt pointer number (sSINo) specified in the CCPU_EntryInt function to Interrupt pointer number (sSINo).
- Since an interrupt does not occur while a stop error is occurring in a C Controller module, the routine registered with the CCPU_EntryInt function will not be executed even if it is enabled.

WARNING

If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.

Return value

Return value	Description
0 (0000H)	Normal

Relevant functions

- Page 86 CCPU_EntryInt
- Page 162 CCPU_DisableInt_ISR

CCPU_FromBuf_ISR

This function reads data from the CPU buffer memory of the CPU module and the buffer memory of the intelligent function module which are mounted on the specified module position. (FROM instruction)

Format

short CCPU_FromBuf_ISR (unsigned short usloNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usloNo	Module position	Specify the module position as follows. Start I/O number divided by 16 (0H to FFH, 3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

Description

- This function reads data for the size specified to the data size (ulSize) from the CPU buffer memory of a CPU module and the buffer memory of an intelligent function module which are specified to the module position (usloNo), and stores it in the data storage destination (pusDataBuf).
Data is read by specifying an offset address from the start of the CPU buffer memory of a CPU module and the buffer memory of an intelligent function module to the offset (ulOffset).
- To access the CPU buffer memory of the module in a multiple CPU system (CPU No.1 to No.4), specify 3E0H (CPU No.1) to 3E3H (CPU No.4) to the module position (usloNo). However, the CPU buffer memory can be accessed only when the multiple CPU setting is configured.


Restriction

Do not execute this function in a routine other than an interrupt routine.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the read data is a multiple of 2.
A data area for the size (words) of the read data is reserved.
A non-existent CPU buffer memory is not specified.
A non-existent buffer memory is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 177 CCPU_ToBuf_ISR

CCPU_FromBufHG_ISR

This function reads data from the fixed cycle communication area of the CPU module mounted on the specified module position.

Format

short CCPU_FromBufHG_ISR (unsigned short usloNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usloNo	Module position	Specify the module position as follows. Start I/O number divided by 16 (3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

Description

- This function reads data for the size specified to the data size (ulSize) from the fixed cycle communication area of a CPU module specified to the module position (usloNo), and stores it in the data storage destination (pusDataBuf). Data is read by specifying an offset address from the start of the fixed cycle communication area to the offset (ulOffset).
- The fixed cycle communication area can be accessed only when the fixed cycle communication area setting under the multiple CPU setting is configured.


Restriction

Do not execute this function in a routine other than an interrupt routine.

! WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the read data is a multiple of 2.
A data area for the size (words) of the read data is reserved.
A non-existent fixed cycle communication area is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 164 CCPU_FromBuf_ISR
- Page 177 CCPU_ToBuf_ISR
- Page 179 CCPU_ToBufHG_ISR

CCPU_GetCounterMicros_ISR

This function acquires a 1 μ s counter value of a C Controller module.

Format

short CCPU_GetCounterMicros_ISR(unsigned long* pulMicros)

Argument

Argument	Name	Description	IN/OUT
pulMicros	1 μ s counter value storage destination	Specify the storage destination of the 1 μ s counter value.	OUT

Description

- This function acquires a 1 μ s counter value of C Controller module and stores the value in the 1 μ s counter value storage destination (pulMicros).
- The 1 μ s counter value increases by 1 every 1 μ s after the power is turned ON.
- The count cycles between 0 and 4294967295.

Restriction

Do not execute this function in a routine other than an interrupt routine.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 167 CCPU_GetCounterMillis_ISR

CCPU_GetCounterMillis_ISR

This function acquires a 1 ms counter value of a C Controller module.

Format

short CCPU_GetCounterMillis_ISR(unsigned long* pulMillis)

Argument

Argument	Name	Description	IN/OUT
pulMillis	1 ms counter value storage destination	Specify the storage destination of the 1 ms counter value.	OUT

Description

- This function acquires a 1 ms counter value of C Controller module and stores the value in the 1 ms counter value storage destination (pulMillis).
- The 1 ms counter value increases by 1 every 1 ms after the power is turned ON.
- The count cycles between 0 and 4294967295.

Restriction

Do not execute this function in a routine other than an interrupt routine.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 166 CCPU_GetCounterMicros_ISR

CCPU_GetDotMatrixLED_ISR

This function acquires the value displayed on the dot matrix LED of a C Controller module, and stores it to the LED data storage destination (pcData).

Format

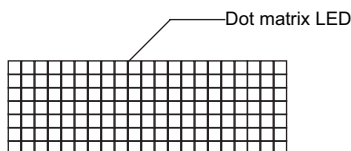
short CCPU_GetDotMatrixLED_ISR (char* pcData, unsigned long ulDataSize)

Argument

Argument	Name	Description	IN/OUT
pcData	LED data storage destination	Specify the storage destination of LED data.	OUT
ulDataSize	LED data storage destination size	Specify the LED data storage destination size in byte units.	IN

Description

- This function acquires the value displayed on the dot matrix LED, and stores it in the LED data storage destination (pcData).
- It also acquires the information for the size specified to the LED data storage destination size (ulDataSize).
- The value displayed on the dot matrix LED is stored in the LED data storage destination (pcData) as shown below.



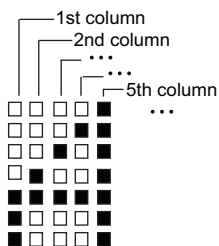
pcData[0] to pcData[19]: Data of the dot matrix LED (7 × 20)

The value displayed in the following format is acquired.

Data format for each column: Bit pattern in which '0' is for the upper one bit, and '1' (when LED is ON) or '0' (when LED is OFF) is for lower seven bits

Ex.

The bit pattern shown below is displayed on the dot matrix LED:



1st column: 0000 0111b = 07H → pcData[0] = 0x07

2nd column: 0000 1100b = 0cH → pcData[1] = 0x0c

3rd column: 0001 0100b = 14H → pcData[2] = 0x14

4th column: 0010 0100b = 24H → pcData[3] = 0x24

5th column: 0111 1111b = 7fH → pcData[4] = 0x7f

6th column to 20th column: 0000 0000b = 00H → pcData[5] to pcData[19] = 0x00

Restriction

Do not execute this function in a routine other than an interrupt routine.

WARNING

If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 174 CCPU_SetDotMatrixLED_ISR


CCPU_ReadDevice_ISR

This function reads data from internal user devices and internal system devices of C Controller module.

Format

short CCPU_ReadDevice_ISR (short sDevType, unsigned long ulDevNo, unsigned long ulSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number. (Only multiples of 16 can be specified for bit devices.)	IN
ulSize	Data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

Description

This function reads data of a device after the one specified to the device type (sDevType) and the start device number (ulDevNo) for the size specified to the data size (ulSize), and stores it in the data storage destination (pusDataBuf).

Restriction

Do not execute this function in a routine other than an interrupt routine.

! WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
A data area for the size (words) of the read data is reserved.
A device which is out of the range is not specified.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 180 CCPU_WriteDevice_ISR

CCPU_RegistEventLog_ISR

This function registers an event log in the event history of a C Controller module.

Format

short CCPU_RegistEventLog_ISR (long IEventCode, char* pcEventMsg)

Argument

Argument	Name	Description	IN/OUT
IEventCode	Detailed code	Specify a detailed event code to be registered in the event history.	IN
pcEventMsg	Detailed information	Specify detailed information character string data of an event to be registered in the event history. (The detailed information character string data of an event can be specified up to 200 bytes. When 'NULL' is specified, the detailed information is not registered.)	IN

Description

This function registers an event log in the event history of a C Controller module.

The contents to be registered on the event history screen of CW Configurator are as follows:

Item	Description
Occurrence Date	Event registered date and time
Event Type	Operation (fixed)
Status	Information (fixed)
Event Code	25000 (fixed)
Overview	Registration from the user program (fixed)
Source	R12CCPU-V (fixed)
Start I/O No.	Input/output number of the C Controller module that executed the CCPU_RegistEventLog_ISR function.
Detailed event code information	Detailed code (hexadecimal) specified to the detailed code (IEventCode)
Detailed event log information	Detailed information specified to the detailed information (pcEventMsg)
Cause	The event history was registered from the C Controller module detailed function. (Fixed)

- The event history can be stored for the size of the event history file specified with CW Configurator.
Note that data is deleted in order from older data if the specified file size is exceeded.


Restriction

Do not execute this function in a routine other than an interrupt routine.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
Detailed information which is out of the range is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST


CCPU_ResetDevice_ISR

This function resets internal user devices and internal system devices (bit devices) of C Controller module.

Format

short CCPU_ResetDevice_ISR(short sDevType, unsigned long ulDevNo)

Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number.	IN

Description

This function resets (turns OFF) the device of a C Controller module specified to the device type (sDevType) and the start device number (ulDevNo).

Restriction

Do not execute this function in a routine other than an interrupt routine.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
A device which is out of the range is not specified.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 173 CCPU_SetDevice_ISR


CCPU_SetDevice_ISR

This function sets internal user devices and internal system devices (bit devices) of C Controller module.

Format

short CCPU_SetDevice_ISR (short sDevType, unsigned long ulDevNo)

Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Device number	Specify the device number.	IN

Description

This function sets (turns ON) the device of a C Controller module specified to the device type (sDevType) and the start device number (ulDevNo).

Restriction

Do not execute this function in a routine other than an interrupt routine.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
A device which is out of the range is not specified.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 172 CCPU_ResetDevice_ISR

CCPU_SetDotMatrixLED_ISR

This function sets a value to be displayed on the dot matrix LED of C Controller module.

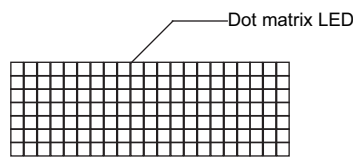
Format

short CCPU_SetDotMatrixLED_ISR (unsigned short usLedMode, char* pcData)

Argument

Argument	Name	Description	IN/OUT
usLedMode	Output mode	Unused (Even if a value is specified, the operation is not affected.)	IN
pcData	LED data	Specify the LED data.	IN

Specify the LED data (pcData) as follows.



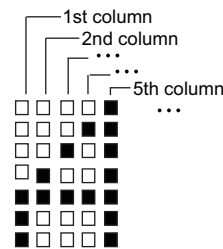
pcData[0] to pcData[19]: Data of the dot matrix LED (7×20)

The data specified in the following format is displayed.

Data format for each column: Bit pattern in which '0' is for the upper one bit, and '1' (when LED is ON) or '0' (when LED is OFF) is for lower seven bits

Ex.

The bit pattern shown below is output to the dot matrix LED:



1st column: 0000 0111b = 07H → pcData[0] = 0x07

2nd column: 0000 1100b = 0cH → pcData[1] = 0x0c

3rd column: 0001 0100b = 14H → pcData[2] = 0x14

4th column: 0010 0100b = 24H → pcData[3] = 0x24

5th column: 0111 1111b = 7fH → pcData[4] = 0x7f

6th column to 20th column: 0000 0000b = 00H → pcData[5] to pcData[19] = 0x00

Description

This function displays the value specified to the LED data (pcData) on the dot matrix LED.

Restriction

- Do not execute this function in a routine other than an interrupt routine.
- Do not execute this function when a mode other than "USER" is selected in the operation selection mode.
An unintended value may be displayed on the dot matrix LED.
- Do not execute this function while checking an operation or checking the selected operation with the MODE/SELECT switch.
An unintended value may be displayed on the dot matrix LED.

Precautions

When this function is executed, an image of the dot matrix LED in which data are being written may be displayed on the [Module Diagnostics (CPU diagnostics)] screen of CW Configurator.

WARNING

If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.

Return value

Return value	Description
0 (0000H)	Normal

Relevant function

- Page 168 CCPU_GetDotMatrixLED_ISR

CCPU_SetLEDStatus_ISR

This function sets the LED status of a C Controller module.

Format

short CCPU_SetLEDStatus_ISR(long lLed, unsigned short usLedInfo)

Argument

Argument	Name	Description	IN/OUT
lLed	Target LED	Unused (Even if a value is specified, the operation is not affected.)	IN
usLedInfo	LED status information	Specify the LED status information.	IN

The specification method for the LED status information (usLedInfo) is as follows:

usLedInfo	Description
0	OFF
1	ON (Red)
2	Flashing at low speed (Red)
3	Flashing at high speed (Red)
4	ON (Green)
5	Flashing at low speed (Green)
6	Flashing at high speed (Green)

Description

This function controls the USER LED of C Controller module to the status specified to the LED status information (usLedInfo).

Restriction

Do not execute this function in a routine other than an interrupt routine.

WARNING

If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.

Return value

Return value	Description
0 (0000H)	Normal

Relevant functions

- Page 130 CCPU_SetLEDStatus

CCPU_ToBuf_ISR

This function writes data to the CPU buffer memory of the CPU module (host CPU) and the buffer memory of the intelligent function module which are mounted on the specified module position. (TO instruction)

Format

short CCPU_ToBuf_ISR (unsigned short usloNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usloNo	Module position	Specify the module position as follows. For the CPU buffer memory, only the host CPU can be accessed. Start I/O number divided by 16 (0H to FFH, 3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN

Description

- This function writes data in the data storage destination (pusDataBuf) for the size specified to the data size (ulSize) to the CPU buffer memory of a CPU module (host CPU) and the buffer memory of an intelligent function module which are specified to the module position (usloNo).
Data is written by specifying an offset address from the start of the CPU buffer memory of a CPU module (host CPU) and the buffer memory of an intelligent function module to the offset (ulOffset).
- To access the CPU buffer memory (host CPU) of the module in a multiple CPU system (CPU No.1 to No.4), specify 3E0H (CPU No.1) to 3E3H (CPU No.4) to the module position (usloNo). However, the CPU buffer memory (host CPU) can be accessed only when the multiple CPU setting is configured.
- When executing this function while the operating status of a CPU module is not RUN, the STOP/PAUSE error (-28640) occurs.


Restriction

- Do not execute this function in a routine other than the one registered in the interrupt.
- When data is written to the same CPU buffer memory (host CPU) from a routine other than an interrupt routine, the output value may be overlapped, resulting in an invalid value. Manage the resource so that data is not written to the same CPU buffer memory (host CPU).

! WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the write data is a multiple of 2.
A non-existent CPU buffer memory (host CPU) is not specified.
A non-existent buffer memory is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 164 CCPU_FromBuf_ISR

CCPU_ToBufHG_ISR

This function writes data to the fixed cycle communication area of the CPU module mounted on the specified module position.

Format

short CCPU_ToBufHG_ISR (unsigned short usloNo, unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usloNo	Module position	Specify the module position as follows. Start I/O number divided by 16 (3E0H to 3E3H)	IN
ulOffset	Offset	Specify the offset in word units.	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN

Description

- This function writes data in the data storage destination (pusDataBuf) for the size specified to the data size (ulSize) to the fixed cycle communication area of a CPU module specified to the module position (usloNo). Data is written by specifying an offset address from the start of the fixed cycle communication area to the offset (ulOffset).
- The fixed cycle communication area can be accessed only when the fixed cycle communication area setting under the multiple CPU setting is configured.
- When executing this function while the operating status of a CPU module specified to the module position (usloNo) is not RUN, the STOP/PAUSE error (-28640) occurs.


Restriction

- Do not execute this function in a routine other than an interrupt routine.
- When data is written to the same fixed cycle communication area from a routine other than an interrupt routine, the output value may be overlapped, resulting in an invalid value. Manage the resource so that data is not written to the same fixed cycle communication area.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the write data is a multiple of 2.
A non-existent fixed cycle communication area is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 164 CCPU_FromBuf_ISR
- Page 177 CCPU_ToBuf_ISR
- Page 165 CCPU_FromBufHG_ISR


CCPU_WriteDevice_ISR

This function writes data to internal user devices and internal system devices of C Controller module.

Format

short CCPU_WriteDevice_ISR (short sDevType, unsigned long ulDevNo, unsigned long ulSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
sDevType	Device type	Specify the device type.  Page 13 Argument specification	IN
ulDevNo	Start device number	Specify the start device number. (Only multiples of 16 can be specified for bit devices.)	IN
ulSize	Data size	Specify the write data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN

Description

- This function writes data in the data storage destination (pusDataBuf) for the size specified to the data size (ulSize) to a device after the one specified to the device type (sDevType) and the start device number (ulDevNo).

Restriction

- Do not execute this function in a routine other than an interrupt routine.
- When data is written to the same device from a routine other than an interrupt routine, the output value may be overlapped, resulting in an invalid value. Manage the resource so that data is not written to the same device.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the write data is a multiple of 2.
A device which is out of the range is not specified.

Return value

Return value	Description
0 (0000H)	Normal

Relevant functions

- Page 170 CCPU_ReadDevice_ISR

CCPU_X_In_Word_ISR

This function reads an input signal (X) in word (16-point) units.

Format

short CCPU_X_In_Word_ISR (unsigned short usXNo, unsigned short usSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usXNo	Start input signal	Specify a start input signal (X). (Specify a multiple of 16.)	IN
usSize	Read data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

Description

- This function operates to the mounted module corresponding to the specified start input signal (usXNo) regardless of the type of the parameter setting (I/O assignment). When the specified area is "empty" or "output module", this function ends normally without processing (read data: 0).
- The input status controlled by other CPUs is not imported.
(The setting to import the out-group input status for the multiple CPU setting is ignored.)
- This function reads an input signal (X) for the size specified to the read data size (usSize) from the start input signal (X) specified to the start input signal (usXNo), and stores it in the data storage destination (pusDataBuf).
- Specify the start input signal (usXNo) in multiples of 16. (The remainder divided by 16 is discarded.)
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

pusDataBuf	Description
pusDataBuf[0]	Data of usXNo+FH to usXNo
pusDataBuf[0]	Data of usXNo+1FH to usXNo+10H
⋮	⋮
pusDataBuf[usSize-1]	Data of usXNo+(usSize-1)×16+FH to usXNo+(usSize-1)×16


Restriction

- Do not execute this function in a routine other than an interrupt routine.
- Do not execute this function to an I/O assignment on which an intelligent function module or an interrupt module is mounted.

WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the read data is a multiple of 2.
A data area for the size (words) of the read data is reserved.
An input signal (X) which is out of the range (excluding 0H to FFFH) is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 183 CCPU_Y_In_Word_ISR
- Page 185 CCPU_Y_Out_Word_ISR

CCPU_Y_In_Word_ISR

This function reads an output signal (Y) in word (16-point) units.

Format

short CCPU_Y_In_Word_ISR (unsigned short usYNo, unsigned short usSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usYNo	Start output signal	Specify a start output signal (Y). (Specify a multiple of 16.)	IN
usSize	Read data size	Specify the read data size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

Description

- This function operates to the mounted module corresponding to the specified output signal (usYNo) regardless of the type of the parameter setting (I/O assignment). When the specified area is "empty" or "input module", this function ends normally without processing (read data: 0).
- The input status controlled by other CPUs is not imported.
(The setting to import the out-group input status for the multiple CPU setting is ignored.)
- This function reads an output signal (Y) for the size specified to the read data size (usSize) from the start output signal (Y) specified to the start output signal (usYNo), and stores it in the data storage destination (pusDataBuf).
- Specify the start output signal (usYNo) in multiples of 16. (The remainder divided by 16 is discarded.)
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

pusDataBuf	Description
pusDataBuf[0]	Data of usYNo+FH to usYNo
pusDataBuf[1]	Data of usYNo+1FH to usYNo+10H
⋮	⋮
pusDataBuf[usSize-1]	Data of usYNo+(usSize-1)×16+FH to usYNo+(usSize-1)×16


Restriction

- Do not execute this function in a routine other than an interrupt routine.
- Do not execute this function to an I/O assignment on which an intelligent function module or an interrupt module is mounted.

! WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the read data is a multiple of 2.
A data area for the size (words) of the read data is reserved.
An output signal (Y) which is out of the range (excluding 0H to FFFH) is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant function

- Page 181 CCPU_X_In_Word_ISR
- Page 185 CCPU_Y_Out_Word_ISR

CCPU_Y_Out_Word_ISR

This function outputs an output signal (Y) in word (16-point) units.

Format

short CCPU_Y_Out_Word_ISR (unsigned short usYNo, unsigned short usSize, unsigned short* pusDataBuf)

Argument

Argument	Name	Description	IN/OUT
usYNo	Start output signal	Specify a start output signal (Y). (Specify a multiple of 16.)	IN
usSize	Output size	Specify the output size in word units.	IN
pusDataBuf	Data storage destination	Specify the storage destination of output data.	IN

Description

- This function outputs (turns ON/OFF) data in the data storage destination (pusDataBuf) from a start output signal (Y) specified to the start output signal (usYNo) to an output signal (Y) for the size specified to the data size (usSize).
- Specify the start output signal (usYNo) in multiples of 16. (The remainder divided by 16 is discarded.)
- Do not specify an output module controlled by other CPUs to the output signal (usYNo).
If it is specified, no operation is performed to the output module.
- Store output data in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

pusDataBuf	Description
pusDataBuf[0]	Data of usYNo+FH to usYNo
pusDataBuf[1]	Data of usYNo+1FH to usYNo+10H
⋮	⋮
pusDataBuf[usSize-1]	Data of usYNo+(usSize-1)×16+FH to usYNo+(usSize-1)×16


Restriction

- Do not execute this function in a routine other than an interrupt routine.
- Do not execute this function to an I/O assignment on which an intelligent function module or an interrupt module is mounted.
- When data is output to the same output signal (Y) from a routine other than an interrupt routine, the output value may be overlapped, resulting in an invalid value. Manage the resource so that data is not output to the same output signal (Y).

! WARNING

- If any function in which an invalid argument is specified is executed, an error such as hardware failure (3C02H) may occur on C Controller module.
- This function does not check the specified argument.
When creating a program, note the following:
An address specified to the write data is a multiple of 2.
An output signal (Y) which is out of the range (excluding 0H to FFFH) is not specified.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 181 CCPU_X_In_Word_ISR
- Page 183 CCPU_Y_In_Word_ISR

3.3 MELSEC Data Link Functions

This section explains the details of the MELSEC data link function.

mdClose

This function closes a communication line (channel).

Format

short mdClose(long IPath)


Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN

Description

- This function closes the channel opened by the mdOpen function.
- When using multiple channels, close the channel 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 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen


mdControl

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

Format

short mdControl(long IPath, short sStNo, short sCode)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
sStNo	Station number	Specify the network number and station number of the target module.  Page 44 Argument specification	IN
sCode	Instruction code	Specify the contents of the remote operation in numerical value.	IN

The specification method for the instruction code (sCode) is as follows:

sCode (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 (sStNo) to the one specified to the instruction code (sCode).

Restriction

This function cannot be executed for a C Controller module, PC CPU module, MELSECWinCPU module, or interface board for a personal computer.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose

mdDevRstEx

This function resets bit devices.

Format

long mdDevRstEx(long IPath, long INetNo, long IStNo, long IDevType, long IDevNo)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module.	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
IDevType	Device type	Specify the device type of bit device. ☞ Page 44 Argument specification	IN
IDevNo	Device number	Specify the device number of bit device.	IN

Description

- This function resets (turns OFF) the bit device of the module specified to the network number (INetNo), the station number (IStNo), the device type (IDevType), and the device number (IDevNo).
- This function is dedicated for bit devices such as link relay (B), 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 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 190 mdDevSetEx

mdDevSetEx

This function sets bit devices.

Format

long mdDevSetEx (long IPath, long INetNo, long IStNo, long IDevType, long IDevNo)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module.	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
IDevType	Device type	Specify the device type of bit device. ☞ Page 44 Argument specification	IN
IDevNo	Device number	Specify the device number of bit device.	IN

Description

- This function sets (turns ON) the bit device of the module specified to the network number (INetNo), the station number (IStNo), the device type (IDevType), and the device number (IDevNo).
- This function is dedicated for bit devices such as link relay (B), 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 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 189 mdDevRstEx

mdGetLabelInfo

This function acquires device information corresponding to label names.

Format

```
long mdGetLabelInfo (long IPath, long INetNo, long IStNo, long ILbCnt, void* pLbLst, long* pIDevLst, unsigned long long* pullLbCode)
```

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
ILbCnt	Number of labels	Specify the number of labels. (Up to 10240) Number of labels can be specified up to 10240.	IN
pLbLst	Label name array	Specify the storage address of label name for each label. Specify a label name in Unicode (UTF-16).	IN
pIDevLst	Device name array	Specify a device to store device information which is acquired. (Device information assigned to labels specified to the label name array (pLbLst) is stored in a randomly selected device format.)	OUT
pullLbCode	Label code	A value to identify whether the label of a CPU module is changed or not is stored. (Whether the label setting is changed or not can be checked by whether this value is changed or not. However, even when converting all in a CPU module, the value changes.)	OUT

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

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

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

Description

- This function reads labels of a CPU module specified to the network number (INetNo) and the station number (IStNo).
- Reserve the area for the device name array (pIDevLst) in the call source.
- Reserve the area for (ILbCnt × 3 + 1) for the size of area of the device name array (pIDevLst).
- If any of the labels of which the label information cannot be acquired exists in the label name specified to the label name array (pLbLst), this function returns any of the following errors. For the device type, start device number, and the number of read points of the label, '0' is stored.

Error code	Error occurrence
-82 (FFB2H)	<ul style="list-style-type: none"> • 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 (FFB4H)	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 (pIDevLst) for the label that acquired device information successfully.
- The specification method for the label name to specify to the device name array (pLbLst) is as follows:

○: Possible, ×: Impossible

Label type	Specification possibility	Specification method	Specification example
Label of the simple data type	○	Specify the label name.	Label1
Element specification of the array label	○	Specify in the following format. <ul style="list-style-type: none"> • One-dimensional array: Label name [m] • Two-dimensional array: Label name [m, n] • Three-dimensional array: Label name [m, n, l] 	<ul style="list-style-type: none"> • One-dimensional array: Label1 [10] • Two-dimensional array: Label2 [10, 20] • Three-dimensional array: Label3 [10, 20, 30]
Whole specification of the structure label	×	—	—
Member of the structure label	○	Specify in the following format. Label name.Element name. to Element name	Str1.Elem1. to Elem3
Array member of the structure label	○	Specify in the following format. Label name.Element name [m]	Str1.Elem[10]
Bit specification of label	×	—	—
Digit specification of label	×	—	—
Label of timer type, retentive timer type, and counter type	○	Specify in the following format. <ul style="list-style-type: none"> • Contact: Label name.S • Coil: Label name.C • Current value: Label name.N 	<ul style="list-style-type: none"> • Contact: Label1.S • Coil: Label2.C • Current value: Label3.N

Precautions

- In CW Workbench, Unicode character strings cannot be entered and source codes including Unicode character strings cannot be compiled. Create a file with Unicode (UTF-16) character strings entered in 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.
- DevGV can be specified only with the functions supporting label access (mdRandRLabelEx/mdRandWLabelEx).
- For accessible CPU modules, refer to the following:

 MELSEC iQ-R C Controller Module User's Manual

Example

The following tables show the examples of values specified to the label name array (pLbLst) and data read to the device name array (pDevLst). (For five labels to be read: Label 1 to 5).

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


- Values specified to pLbLst


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

- Value to be read to pDevLst

pDevLst	Read value	Description
pDevLst[0]	5	Number of blocks
pDevLst[1]	DevD	Device type
pDevLst[2]	10	Start device number
pDevLst[3]	1	Number of read points
pDevLst[4]	DevD	Device type
pDevLst[5]	11	Start device number
pDevLst[6]	1	Number of read points
pDevLst[7]	DevM	Device type
pDevLst[8]	100	Start device number
pDevLst[9]	1	Number of read points
pDevLst[10]	DevM	Device type
pDevLst[11]	101	Start device number
pDevLst[12]	1	Number of read points
pDevLst[13]	DevM	Device type
pDevLst[14]	102	Start device number
pDevLst[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 222 ERROR CODE LIST

*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 195 mdOpen
- Page 187 mdClose
- Page 199 mdRandRLabelEx
- Page 204 mdRandWLabelEx

mdInit

This function initializes communication route information.

Format

short mdInit(long IPath)


Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN

Description

This function clears communication route information using the path of the specified channel.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose


mdOpen

This function opens a communication line (channel).

Format

short mdOpen(short sChan, short sMode, long* plPath)


Argument

Argument	Name	Description	IN/OUT
sChan	Channel	Specify a communication line (channel).  Page 44 Argument specification	IN
sMode	Mode	Specify '-1'.	IN
plPath	Path of channel	Specify the storage destination (address) of the path of the channel. (The path of the opened channel is stored.)	OUT

Description

- When executing a MELSEC data link function, use the path of a channel opened with this function.
- To end a user program, close the opened path of a channel with the mdClose function.
- When using multiple channels, open the channel 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 222 ERROR CODE LIST

Relevant function

- Page 187 mdClose

mdRandREx

This function reads devices randomly.

Format

long mdRandREx(long IPath, long INetNo, long IStNo, long* pIDev, short* psBuf, long IBufSize)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
pIDev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be read.	IN
psBuf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
IBufSize	Read data storage destination size	Specify the area size reserved in the read data storage destination in byte units.	IN

The specification method for the randomly selected device (pIDev) is as follows:

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

Description

- This function reads devices specified to the randomly selected device (pIDev) from a module specified to the network number (INetNo) and the station number (IStNo).
- The read data is stored in the read data storage destination (psBuf) in word units in order of the specification to the randomly selected device (pIDev). A bit device is stored per 16 points, a word device is stored per 1 point, and a double-word device is stored in word units.
- Specify so that the total number of read points specified for each block is 10240 points or less. Otherwise, a size error (-5) occurs.
- Communication time varies significantly depending on the contents specified to the randomly selected device (pIDev). To reduce communication time, use the mdReceiveEx function.
- To access the own station, set the station number to 255. When the actual station number is used, an error will occur.

Example

The following tables show the examples of values specified to the randomly selected device (plDev), values read to the read data storage destination (psBuf), and the number of bytes of read data.

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

Values specified to the randomly selected device (plDev)

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


Values read to the read data storage destination (psBuf)

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

Number of bytes of read data

$(\text{psBuf}[0] \text{ to } \text{psBuf}[10] = 11) \times 2 = 22$

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 202 mdRandWEx

mdRandRLabelEx

This function reads devices corresponding to labels randomly.

Format

```
long mdRandRLabelEx(long IPath, long INetNo, long IStNo, long* pIDev, short* psBuf, long lBufSize, unsigned long long ullLbCode)
```

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
pIDev	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 with the mdGetLabelInfo function.)	IN
psBuf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
lBufSize	Read data storage destination size	Specify the area size reserved in the read data storage destination in byte units.	IN
ullLbCode	Label code	Specify the label code acquired with the mdGetLabelInfo function.	IN

The specification method for the randomly selected device (pIDev) is as follows:

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

- One block comprises of three elements such as device type, start device number, and number of read points, the total number of blocks will be stored in the first element of the randomly-specified device (pIDev).

Description

- This function reads devices specified to the randomly selected device (pIDev) from a module specified to the network number (INetNo) and the station number (IStNo).
- The read data is stored in the read data storage destination (psBuf) in word units in order of the specification to the randomly selected device (pIDev). A bit device and a word device are stored per 1 point, and a double-word device is stored in word units.
- Specify so that the total number of read points specified for each block is 10240 points or less. Otherwise, a size error (-5) occurs.
- When '0' is specified to the label code (ullLbCode), the device is read without checking the label code.

Example

The following tables show the examples of values specified to the randomly selected device (plDev), values read to the read data storage destination (psBuf), and the number of bytes of read data.

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

Values specified to the randomly selected device (plDev)

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


Values read to the read data storage destination (psBuf)


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

Number of bytes of read data

$(\text{psBuf}[0] \text{ to } \text{psBuf}[10] = 11) \times 2 = 22$

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: *1  Page 222 ERROR CODE LIST

*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 functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 191 mdGetLabelInfo
- Page 204 mdRandWLabelEx

mdRandWEx

This function writes devices randomly.

Format

long mdRandWEx(long IPath, long INetNo, long IStNo, long* pIDev, short* psBuf, long lBufSize)

Argument

Argument	Name	Description	IN/OUT
IPath	Channel	Specify the path of the channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
pIDev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be written.	IN
psBuf	Write data storage destination	Specify the storage destination (address) of write data. Reserve a continuous area for the write data storage destination.	IN
lBufSize	Write data storage destination size	Unused (Even if a value is specified, the operation is not affected.)	IN

The specification method for the randomly selected device (pIDev) is as follows:

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

Description

- This function writes data to a device, which is specified to the randomly selected device (pIDev), of a module specified to the network number (INetNo) and the station number (IStNo).
- The data to be written is stored to the write data storage destination (psBuf) 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 in word units.
- Specify so that the total number of write points specified for each block is 10240 points or less. Otherwise, a size error (-5) occurs.
- Note that the extension comment information will be deleted when the data is written to the block to which an extension comment is assigned (extension file register).
- Also, note that sub 2 or sub 3 program will be deleted when data is written to a block (extension file register) overlapping with the program setting area for sub 2 or sub 3.

Example

The following tables show the examples of values specified to the randomly selected device (plDev) and the write data storage destination (psBuf), and the number of bytes of write data.

Device to be written randomly	Description
M100 to M115	Turns all the bits OFF.
D10 to D13	Stores 10 in D10, 200 in D11, 300 in D12, and 400 in D13.
LCN100 to LCN101	Stores 0x1 in LCN100, and 0x10000 in LCN101.

Values specified to the randomly selected device (plDev)

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


Values specified to the write data storage destination (psBuf)

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

Number of bytes of write data

(psBuf[0] to psBuf[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 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 196 mdRandREx

mdRandWLabelEx

This function writes devices corresponding to labels randomly.

Format

long mdRandWLabelEx(long IPath, long INetNo, long IStNo, long* plDev, short* psBuf, long lBufSize, unsigned long long ullLbCode)

Argument

Argument	Name	Description	IN/OUT
IPath	Channel	Specify the path of the channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 13 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 13 Argument specification	IN
plDev	Randomly selected device	Specify the number of blocks, device type, start device number, and device points of devices to be written.	IN
psBuf	Write data storage destination	Specify the storage destination (address) of write data. Reserve a continuous area for the write data storage destination.	IN
lBufSize	Write data storage destination size	Unused (Even if a value is specified, the operation is not affected.)	IN
ullLbCode	Label code	Specify the label code acquired with the mdGetLabelInfo function.	IN

The specification method for the randomly selected device (plDev) is as follows:

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

- One block comprises of three elements such as device type, start device number, and number of write points, the total number of blocks will be stored in the first element of the randomly-specified device (plDev).

Description

- This function writes data to a device, which is specified to the randomly selected device (plDev), of a module specified to the network number (INetNo) and the station number (IStNo).
- The data to be written is stored to the write data storage destination (psBuf) in word units. A bit device and a word device are stored per 1 point, and a double-word device is stored in word units.
- Specify so that the total number of write points specified for each block is 10240 points or less. Otherwise, a size error (-5) occurs.
- Note that the extension comment information will be deleted when the data is written to the block to which an extension comment is assigned (extension file register).
- Also, note that sub 2 or sub 3 program will be deleted when data is written 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 (ullLbCode), the device is written without checking the label code.

Example

The following tables show the examples of values specified to the randomly selected device (plDev) and the write data storage destination (psBuf), and the number of bytes of write data.

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

Values specified to the randomly selected device (plDev)

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


Values specified to the write data storage destination (psBuf)


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

Number of bytes of write data

(psBuf[0] to psBuf[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 222 ERROR CODE LIST

*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 functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 191 mdGetLabelInfo
- Page 199 mdRandRLabelEx

mdReceiveEx

This function reads devices in a batch.

Format

long mdReceiveEx(long IPath, long INetNo, long IStNo, long IDevType, long IDevNo, long* pISize, short* psData)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
IDevType	Device type	Specify the device type for device to be read in batch.	IN
IDevNo	Start device number	Specify the start device number for device to be read in batch. (For a bit device, set a device number in multiples of 8).	IN
pISize	Read data size	Specify the read data size in byte units. (Specify the value in multiples of 4 when a double-word device (LZ, LTN, LCN, LSTN) is specified, or 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
psData	Read data storage destination	Specify the storage destination (address) of read data.	OUT

Description

- This function reads data for the size specified to the read data size (pISize) from a device specified to the device type (IDevType) and the start device number (IDevNo) of a module specified to the network number (INetNo) and the station number (IStNo).
- When the read data size exceeds the device range, a readable size is returned to the read data size (pISize).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 209 mdSendEx(Device batch write function)

mdReceiveEx

This function receives messages. (RECV function)

Format

long mdReceiveEx(long IPath, long INetNo, long IStNo, long IDevType, long IDevNo, long* pSize, short* psData)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify '0' (0H).	IN
IStNo	Station number	Specify own station 255 (FFH).	IN
IDevType	Device type	Specify the device type for device to be read in batch. Only "RECV function: 101 (65H and DevMAIL)" is available.	IN
IDevNo	Channel number	Specify the channel number. • CC-Link IE Controller Network: 1 to 8 • CC-Link IE Field Network: 1 to 2 • CC-Link IE TSN: 1 to 8 • MELSECNET/H network: 1 to 8	IN
pSize	Receive message size	Specify the 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
psData	Receive data storage destination	Specify the storage destination (address) of receive data. Data equivalent to 6 + a value specified to pSize (bytes) are specified.	OUT

Description


- This function supports the RECV instruction, the dedicated instruction for a CC-Link IE Controller Network module, a CC-Link IE Field Network module, CC-Link IE TSN, or a MELSECNET/H network module, when a value specified for the path of channel (IPath) is a value returned with the mdOpen function by specifying CC-Link IE Controller Network (channel No.151 to 158), CC-Link IE Field Network (channel No.181 to 188), CC-Link IE TSN (channel No.281 to 288), or MELSECNET/H network (channel No.51 to 54), and "RECV function: 101" is specified for the device type.
- This function receives the message to the channel specified to the channel number (IDevNo) from among the messages sent to a CC-Link IE Controller Network module, CC-Link IE Field Network module, CC-Link IE TSN, or a MELSECNET/H network module.
- Check that the RECV execution request flag of a network module is ON before executing this function.
- For more advanced RECV function, use the C Controller module dedicated functions. This mdReceiveEx function reads the message to the specified channel number in the order it was received.
- When the actual size of a received message is smaller than the value specified to the receive message size (pSize), data of the actual size is stored in the receive data storage destination (psData[3] or higher), and the data size of the received message is returned to the receive message size (pSize).
- When the actual size of a received message is bigger than the value specified to the receive message size (pSize), data up to the specified size is stored in the receive data storage destination (psData[3] or higher).
- A received message is stored in the receive data storage destination (psData).
Information on a message send source (network number, station number, and used channel of a send station) is stored in psData[0] to psData[2]. Therefore, the storage size of a received message is '6' + (pSize) byte.

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

- The arguments of this function correspond to the control data (device) of the dedicated instruction (RECV) as shown below:

Device	Item	Corresponding argument and return value
+0	Error completion type	—
+1	Completion status	sRet
+2	Own station storage channel	IDevNo
+3	Channel used by send station	psData[2]
+4	Send station network number	psData[0]
+5	Send station number	psData[1]
+6	Not used	—
+7	Not used	—
+8	Arrival monitoring time	—
+9	Receive data length	plSize
+10	Not used	—
+11	Clock setting flag	—
+12	Clock data (Set only in an abnormal state.)	—
+13		—
+14		—
+15		—
+16	Error detection 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 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 210 mdSendEx(Message send function)
- Page 75 CCPU_DedicatedGInst
- Page 77 CCPU_DedicatedJInst

mdSendEx

This function writes devices in a batch.

Format

long mdSendEx(long IPath, long INetNo, long IStNo, long IDevType, long IDevNo, long* pISize, short* psData)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. ☞ Page 44 Argument specification	IN
IDevType	Device type	Specify the device type for device to be written in batch.	IN
IDevNo	Start device number	Specify the start device number to be written in batch. (For a bit device, set a device number in multiples of 8).	IN
pISize	Write data size	Specify the write data size in byte units. (Specify the value in multiples of 4 when a double-word device (LZ, LTN, LCN, LSTN) is specified, or 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
psData	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 for the size specified to the write data size (pISize) from a device specified to the device type (IDevType) and the start device number (IDevNo) of a module specified to the network number (INetNo) and the station number (IStNo).
- It checks the arguments and verifies whether the address + size determined by the arguments is within the device memory range.
- When the write data size exceeds the device range, a writable size is returned to the write data size (pISize).
- Note that the extension comment information will be deleted when the data is written to the block to which an extension comment is assigned (extension file register).

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following: ☞ Page 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 206 mdReceiveEx(Device batch read function)

mdSendEx

This function sends messages. (SEND function)

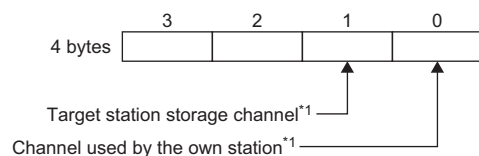
Format

long mdSendEx(long IPath, long INetNo, long IStNo, long IDevType, long IDevNo, long* plSize, short* psData)

Argument

Argument	Name	Description	IN/OUT
IPath	Path of channel	Specify the path of the opened channel.	IN
INetNo	Network number	Specify the network number of target module. A logical station number cannot be specified. ☞ Page 44 Argument specification	IN
IStNo	Station number	Specify the station number of target module. A logical station number cannot be specified. ☞ Page 44 Argument specification	IN
IDevType	Device type	Specify the device type for device to be written in batch. When "Group number" or "All stations" is specified for the station number, only "Without arrival confirmation" is valid. • With arrival confirmation: 101 (65H, DevMAIL) • Without arrival confirmation: 102 (66H, DevMAILNC)	IN
IDevNo	Channel number	Specify the channel number.	IN
plSize	Send data size	Specify the send data size in byte units. (2 to 1920) (Specify the size with an even number.)	IN/OUT
psData	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: 1 to 8
 CC-Link IE Field Network: 1 to 2
 CC-Link IE TSN: 1 to 8
 MELSECNET/H network: 1 to 8


Description

- This function supports the SEND instruction, the dedicated instruction for a CC-Link IE Controller Network module, CC-Link IE Field Network module, CC-Link IE TSN, or MELSECNET/H network module, when a value specified for the path of channel (IPath) is a value returned with the mdOpen function by specifying CC-Link IE Controller Network (channel No.151 to 158), CC-Link IE Field Network (channel No.181 to 188), CC-Link IE TSN (channel No.281 to 288), or MELSECNET/H network (channel No.51 to 54), and "With arrival confirmation: 101" or "Without arrival confirmation: 102" is specified for the device type.
- This function sends a message from a CC-Link IE Controller Network module, CC-Link IE Field Network module, CC-Link IE TSN, or MELSECNET/H network module to the target (network number/station/channel) specified for the station number (IStNo) and the device type (IDevNo).
- For more advanced SEND functions, use the C Controller module dedicated functions.
- An error occurs if a message data is sent to a channel currently in use.

- The arguments of this function correspond to the control data (device) of the dedicated instruction (SEND) as shown below:

Device	Item	Corresponding argument and return value
+0	Execution/error completion type	IDevType
+1	Completion status	sRet
+2	Own station channel	IDevNo
+3	Target station storage channel	IDevNo
+4	Target station network number	INetNo
+5	Target station number	IStNo
+6	Not used	—
+7	Number of retransmission (retry)	—
+8	Arrival monitoring time	—
+9	Send data length	pSize
+10	Not used	—
+11	Clock setting flag	—
+12	Clock data	—
+13		—
+14		—
+15		—
+16	Error detection 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 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose
- Page 207 mdReceiveEx(Message receive function)
- Page 75 CCPU_DedicatedGInst
- Page 77 CCPU_DedicatedJInst


mdTypeRead

This function reads the model code of a CPU module.

Format

short mdTypeRead(long lPath, short sStNo, short* psCode)

Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sStNo	Station number	Specify the network number and station number of the target module.  Page 44 Argument specification	IN
psCode	Model code	Specify the storage destination (address) of the model code. (Stores the read model code.)	OUT

Description


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

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
0090H	Interface board for a personal computer
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
02E9H	Q04UDEHCPU
02EAH	Q06UDEHCPU
02EBH	Q13UDEHCPU
02ECH	Q26UDEHCPU
02EDH	Q50UDEHCPU
02EEH	Q100UDEHCPU

Model code (hexadecimal)	CPU module
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
4C00H	R16MTCPU
4C01H	R32MTCPU
4C02H	R64MTCPU
4841H	R08PCPU
4842H	R16PCPU
4843H	R32PCPU
4844H	R120PCPU
4891H	R08SFCPU
4892H	R16SFCPU
4893H	R32SFCPU
4894H	R120SFCPU
4C00H	R16MTCPU
4C01H	R32MTCPU
4C02H	R64MTCPU

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

Relevant functions

- Page 195 mdOpen
- Page 187 mdClose

3.4 Motion Module Dedicated Class

This section shows the details of the motion module dedicated classes.

MCFB::RefreshLabels

This function executes an MCFB with I/O variables and input variables set in the MCFB setting function. After the execution, I/O variables, input variables and public variables are refreshed.




Format

short MCFB::RefreshLabels(void)


Argument

None

Description

- This function executes an MCFB with I/O variables and input variables set in the MCFB setting function. After the execution, the function acquires values of I/O variables, output variables, and public variables of the MCFB and passes the values to member variables of each instruction class.
To execute an MCFB, check that the MCFB is in the executable status. If the MCFB is not in the executable status (including the case when a motion module is not ready), an error may not be detected. ( Page 62 Considerations for executing MCFBs)
- For operational specifications for executing an MCFB, check the MCFB which corresponds to each instruction class. ( MELSEC iQ-R Motion Module User's Manual (Application))
- After the MCFB is executed, only the values of I/O variables, output variables, and public variables are updated.
- For I/O variables, output variables, and public variables that can be acquired, refer to the following: ( MELSEC iQ-R Motion Module User's Manual (Application))
- Output variables and public variables may not be updated depending on the operation timing of a motion module. After executing the MCFB setting function, execute the MCFB::RefreshLabels function repeatedly and make sure to check values of I/O variables, output variables, and public variables.
- If the MCFB setting function has never been executed, or the execution is not successful, the 'function not executed error' will occur. Check that the MCFB setting function is completed normally and retry again.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

MCFBExecute::SetExecute

This function sets I/O variables and input variables of an Execute-type MCFB in a motion module.

Format


short MCFBExecute::SetExecute(bool Execute)

Argument

Argument	Name	Description	IN/OUT
Execute	Execution command	Specify a value to be set for Execute, which is an input variable of an Execute-type MCFB.	IN


Description

- This function operates differently when executing it at the first time or from the second time onwards in the same instance.

Item	Description
First time	Creates an internal instance for sending/receiving data to/from a motion module. After creating the instance, the function sets I/O variables and input variables in a motion module and executes an MCFB. Since the C Controller module communicates with a motion module when creating the instance, an error occurs if the 'ready' signal of the motion module does not turn ON. ( Page 62 Considerations for executing MCFBs)
Second time or after	Sets I/O variables and input variables in a motion module. The MCFB is executed at the execution of the MCFB::RefreshLabels function.

After executing the MCFB, execute the MCFB::RefreshLabels function repeatedly until the MCFB processing is completed and make sure to check values of I/O variables, output variables, and public variables.

For operational specifications for executing an Execute-type MCFB, refer to the following:


( MELSEC iQ-R Motion Module User's Manual (Application))

- After executing the MCFBExecute::SetExecute function, execute the MCFB with the MCFB::RefreshLabels function in the same instance. Executing the MCFBExecute::SetExecute function continuously for multiple times causes 'refresh not executed error.'
However, for the initial execution of the MCFBExecute::SetExecute function, I/O variables and input variables must be set before executing the MCFB. Therefore, as long as a motion module is refreshed, the MCFBExecute::SetExecute function will be completed normally even if it is executed continuously after the initial execution. After the normal completion of the function, values specified for the I/O variables and input variables are set, and the MCFB is executed with the set variables at the next execution of the MCFB::RefreshLabels function.
- When executing the MCFBExecute::SetExecute function immediately after the MCFB::RefreshLabels function is executed in the same instance, the 'refresh not executed error' occurs depending on the refresh timing of a motion module.
To change I/O variables and input variables, change the variables after the appropriate time passes from the execution of the MCFB::RefreshLabels function in accordance with a system.
- When the 'refresh not executed error' occurs, the member variable values of I/O variables and input variables at when the MCFBExecute::SetExecute function is successfully executed last time will be enabled.
- The 'input/output number specification error' occurs if the input/output number of an instance is changed from the one specified at the first execution of the MCFBExecute::SetExecute function and that function is executed in the same instance.

Precautions

To change I/O variables and input variables set in a motion module, the variables need to be set again after executing an MCFB with the MCFB::RefreshLabels function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

MCFBEnable::SetEnable

This function sets I/O variables and input variables of an Enable-type MCFB in a motion module.

Format

short MCFBEnable::SetEnable(bool Enable)

Argument

Argument	Name	Description	IN/OUT
Enable	Execution command	Specify a value to be set for Enable, which is an input variable of an Enable-type MCFB.	IN

Description

- This function operates differently when executing it at the first time or from the second time onwards in the same instance.

Item	Description
First time	Creates an internal instance for sending/receiving data to/from a motion module. After creating the instance, the function sets I/O variables and input variables in a motion module and executes an MCFB. Since the C Controller module communicates with a motion module when creating the instance, an error occurs if the 'ready' signal of the motion module does not turn ON. (Page 62 Considerations for executing MCFBs)
Second time or after	Sets I/O variables and input variables in a motion module. The MCFB is executed at the execution of the MCFB::RefreshLabels function.

After executing the MCFB, execute the MCFB::RefreshLabels function repeatedly until the MCFB processing is completed and make sure to check values of I/O variables, output variables, and public variables.

For operational specifications for executing an Enable-type MCFB, refer to the following:


(MELSEC iQ-R Motion Module User's Manual (Application))

- After executing the MCFBEnable::SetEnable function, execute the MCFB with the MCFB::RefreshLabels function in the same instance. Executing the MCFBEnable::SetEnable function continuously for multiple times causes 'refresh not executed error.'
However, for the initial execution of the MCFBEnable::SetEnable function, I/O variables and input variables must be set before executing the MCFB. Therefore, as long as a motion module is refreshed, the MCFBEnable::SetEnable function will be completed normally even if it is executed continuously after the initial execution. After the normal completion of the function, values specified for the I/O variables and input variables are set, and the MCFB is executed with the set variables at the next execution of the MCFB::RefreshLabels function.
- When executing the MCFBEnable::SetEnable function immediately after the MCFB::RefreshLabels function is executed in the same instance, the 'refresh not executed error' occurs depending on the refresh timing of a motion module.
To change I/O variables and input variables, change the variables after the appropriate time passes from the execution of the MCFB::RefreshLabels function in accordance with a system.
- When the 'refresh not executed error' occurs, the member variable values of I/O variables and input variables at when the MCFBEnable::SetEnable function is successfully executed last time will be enabled.
- The 'input/output number specification error' occurs if the input/output number of an instance is changed from the one specified at the first execution of the MCFBEnable::SetEnable function and that function is executed in the same instance.
- The MCv_Jog::SetEnable function is not supported in the MCv_Jog class. Use the MCv_Jog::SetEnableJog function.
Executing the MCv_Jog::SetEnable function in the MCv_Jog class causes 'unsupported function execution error.'

Precautions

To change I/O variables and input variables set in a motion module, the variables need to be set again after executing an MCFB with the MCFB::RefreshLabels function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

MCv_Jog::SetEnableJog

This function sets I/O variables and input variables of an Enable-type MCv_Jog in a motion module.


Format


short MCv_Jog::SetEnableJog(bool JogForward, bool JogBackward)

Argument

Argument	Name	Description	IN/OUT
JogForward	Forward JOG command	Specify a value to be set in JogForward, which is the input variable of MCv_Jog of an Enable-type MCFB.	IN
JogBackward	Backward JOG command	Specify a value to be set in JogBackward, which is the input variable of MCv_Jog of an Enable-type MCFB.	IN


Description

- This function sends a command for executing MCv_Jog. For operational specifications of MCv_Jog, refer to the following:
( MELSEC iQ-R Motion Module User's Manual (Application))
- This function operates differently when executing it at the first time or from the second time onwards in the same instance.

Item	Description
First time	Creates an internal instance for sending/receiving data to/from a motion module. After creating the instance, the function sets I/O variables and input variables in a motion module and executes an MCFB. Since the C Controller module communicates with a motion module when creating the instance, an error occurs if the 'ready' signal of the motion module does not turn ON. ( Page 62 Considerations for executing MCFBs)
Second time or after	Sets I/O variables and input variables in a motion module. The MCFB is executed at the execution of the MCFB::RefreshLabels function.

After executing the MCFB, execute the MCFB::RefreshLabels function repeatedly until the MCFB processing is completed and make sure to check values of I/O variables, output variables, and public variables.

For operational specifications for executing an Enable-type MCFB, refer to the following:


( MELSEC iQ-R Motion Module User's Manual (Application))

- After executing the MCv_Jog::SetEnableJog function, execute the MCFB with the MCFB::RefreshLabels function in the same instance. Executing the MCv_Jog::SetEnableJog function continuously for multiple times causes 'refresh not executed error.'
However, for the initial execution of the MCv_Jog::SetEnableJog function, I/O variables and input variables must be set before executing the MCFB. Therefore, as long as a motion module is refreshed, the MCv_Jog::SetEnableJog function will be completed normally even if it is executed continuously after the initial execution. After the normal completion of the function, values specified for the I/O variables and input variables are set, and the MCFB is executed with the set variables at the next execution of the MCFB::RefreshLabels function.
- When executing the MCv_Jog::SetEnableJog function immediately after the MCFB::RefreshLabels function is executed in the same instance, the refresh not executed error occurs depending on the refresh timing of a motion module.
To change I/O variables and input variables, change the variables after the appropriate time passes from the execution of the MCFB::RefreshLabels function in accordance with a system.
- When the 'refresh not executed error' occurs, the member variable values of I/O variables and input variables at when the MCv_Jog::SetEnableJog function is successfully executed last time will be enabled.
- The 'input/output number specification error' occurs if the input/output number of an instance is changed from the one specified at the first execution of the MCv_Jog::SetEnableJog function and that function is executed in the same instance.

Precautions

To change I/O variables and input variables set in a motion module, the variables need to be set again after executing an MCFB with the MCFB::RefreshLabels function.

Return value

Return value	Description
0 (0000H)	Normal
Other than 0	Error For details on the error, refer to the following:  Page 222 ERROR CODE LIST

4 ERROR CODE LIST

This chapter shows the codes for errors occurred in the dedicated function library and the corrective actions.

4.1 Common Error Codes

The following table shows the common error codes for the dedicated function library.

Error code*1		Description	Corrective action
Decimal	Hexadecimal		
1	0001H	■Driver not started The driver is not started.	<ul style="list-style-type: none"> • Check the channel number. • Correct the error that occurred when the driver is started. • Check the status of the system drive of the C Controller module. • Check if the operating system is running normally.
2	0002H	■Timeout error <ul style="list-style-type: none"> • 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. 	<ul style="list-style-type: none"> • Review the operating status and mounting condition of the access target station. • 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.	<ul style="list-style-type: none"> • Check the path of the channel, network number, and station number. • Check if the function performed in the target station is supported.
77	004DH	■Memory reservation error ■Resource shortage error ■Task over error <ul style="list-style-type: none"> • Reserving sufficient memory failed, or there are too many tasks using the dedicated function library. • A function other than the C Controller module dedicated function for the interrupt service routine (ISR) is executed in ISR. 	<ul style="list-style-type: none"> • The memory may be insufficient. End another running task or reduce the access size. • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON. • Reduce the number of tasks which use the dedicated function library and retry the operation. • To execute other dedicated functions synchronously with an interrupt, implement the notification processing in a user program and perform the processing in a task. • Review the size or number specified to the arguments of the user program.
102	0066H	■Data send error ■Restart error Sending data failed, or an attempt was made to send data during restart.	<ul style="list-style-type: none"> • Retry. • Retry after completion of the restart. • Check if the C Controller module is running normally. • Reset the C Controller system.
16384 to 20479	4000H to 4FFFH	■Errors detected in the access target CPU module	Refer to the user's manual for 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 and station number of the request destination. When the target station is the interface board for a personal computer, use the mdTypeRead function and mdSendEx function (SEND function) only. Do not use any other functions.
-28158	9202H	■WDT error WDT (system/user) error occurred.	<ul style="list-style-type: none"> • Reset the C Controller module, or turn the power OFF and ON. • Check the event history in the module diagnostics of CW Configurator and remove the causes of WDT (system or user) error. After that, reset the C Controller module or turn the power OFF and ON again.

Error code*1		Description	Corrective action
Decimal	Hexadecimal		
-28413	9103H	■Target CPU down error The target CPU module is down.	Check the operating status of the target CPU and troubleshoot the error according to the user's manual for 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 and troubleshoot the error according to the user's manual for 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.	<ul style="list-style-type: none"> • Check the mounting condition of the target CPU module. • Change the target CPU number specified in the user program.
-28622	9032H	■Target module busy error <ul style="list-style-type: none"> • 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. 	Add the processing to wait for the completion of the target operation or to retry the operation in the user program.
-28628	902CH	■Pointer address specification error An incorrect address was specified to the argument pointer.	Check the address of the specified pointer.
-28629	902BH	■WDT not started error An attempt was made to reset a WDT before starting it.	Reset the WDT after starting it.
-28630	902AH	■WDT startup error An attempt was made to start WDT while the other WDT is starting up.	Start the WDT after stopping the WDT which is starting up.
-28632	9028H	■I/O number error <ul style="list-style-type: none"> • The specified I/O number is out of range. • No accessible module is mounted on the specified I/O number. 	Check the specified I/O number.
-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 (C Controller module/PC CPU module/MELSECWinCPU module).
-28634	9026H	■Intelligent function module down error The intelligent function module has an error.	<ul style="list-style-type: none"> • Check the mounting condition of the target CPU module. • Replace the intelligent function module or base unit.
-28640	9020H	■STOP/PAUSE error The request of output or of writing to the buffer memory is issued when the operating status of the CPU module is STOP or PAUSE.	Change the operation status of the CPU module to RUN.
-28653	9013H	■I/O assignment error <ul style="list-style-type: none"> • 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 (C Controller module/PC CPU module/MELSECWinCPU module).
-28660	900CH	■Access size error The specified size is out of range.	Review the specified offset and size.

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

4.2 C Controller Module Dedicated Functions

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

Error code		Description	Corrective action
Decimal	Hexadecimal		
-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 specified access size of I/O signal (I/O number and read/write size in words).
-205	FF33H	■I/O number error The specified I/O number is out of range.	Check the specified I/O number.
-208	FF30H	■Offset error • The specified offset is out of range. • An AnS series module (buffer memory) was accessed.	• Check the specified offset. • Check the specified I/O number.
-209	FF2FH	■Buffer memory size error • The specified offset and its size is out of range. • The address of data storage buffer pointer is 0. • The specified size is 0.	• Check the specified buffer memory size. • Check the offset and its size. • Check the specified data storage buffer pointer.
-210	FF2EH	■Read area size error The read area size is smaller than the read size.	• 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.
-220	FF24H	■WDT type error The specified WDT type is out of range.	Check the specified WDT type.
-222	FF22H	■Bus master CPU reset error Remote reset of the bus master CPU (CPU No.1) failed.	• Enable the setting to allow remote reset (set "Remote Reset" to "Enable") for the bus master CPU (CPU No.1). • Change the operating status of the bus master CPU (CPU No.1) to STOP. • Check if the bus master CPU (CPU No.1) is a programmable controller CPU or C Controller module.
-223	FF21H	■Memory reservation error Reserving sufficient memory failed.	Check if sufficient memory is available.
-224	FF20H	■LED setting value error The specified LED setting value is out of range.	Check the specified LED setting value.
-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 control code failed.	• Retry. • Check if the C Controller module is running normally. • Reset the C Controller system.
-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 (1 to 4).	• Change the specified CPU number. • 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 the function waits for the event.	• Check if a program is forcibly being terminated. • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON.
-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.
-236	FF14H	■Remote operation specification code error The remote operation specification code is out of range.	Check the specified remote operation specification code.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-237	FF13H	■Detailed information character string specification error The length of the specified character string was out of range or characters which cannot be specified was specified.	Correct the length of the specified character string or character string data.
		■Application code specification error Five or more digits of the hexadecimal number was specified for the application code.	Change the specified application code.
-238	FF12H	■Event log registration error Registering an event log failed.	Reset the C Controller module, or turn the power OFF and ON.
-239	FF11H	■Drive insertion error Either of the following functions executed with no drive inserted. • CCPU_UnmountMemoryCard • CCPU_MountMemoryCard	Check if a drive is inserted.
-240	FF10H	■Clock data incorrect error The clock data to be set or the read clock data is incorrect.	<ul style="list-style-type: none"> • Check the clock data to be set. • If this error occurs when reading the clock data, set the data again.
-241	FF0FH	■Cycle specification error <ul style="list-style-type: none"> • The specified cycle is out of range. • The cycle was set even when it had already been set. 	<ul style="list-style-type: none"> • Check the specified cycle. • Check if the cycle has been already set.
-242	FF0EH	■Synchronization type specification error The specified synchronization type is out of range.	Check the specified synchronization type.
-246	FF0AH	■Timer event registration error Registering a timer event failed.	<ul style="list-style-type: none"> • Retry. • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON.
-253	FF03H	■Device number specification error <ul style="list-style-type: none"> • 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 <ul style="list-style-type: none"> • The specified number of words is out of range. • The specified size is 0. • The sum of values for specified module label, offset, and data size exceeds the range of module label (label memory). 	<ul style="list-style-type: none"> • Correct the specified start device number and number of words. • Correct the module label, offset and data size specified in the user program.
-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.	<ul style="list-style-type: none"> • 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.
-257	FEFFH	■Interrupt event type specification error The value specified to the interrupt event type is out of range.	Check the specified value.
-259	FEFDH	■Interrupt service routine unregistered error <ul style="list-style-type: none"> • The processing was not registered when enabling the processing corresponding to an event (interrupt). • The specified CPU number is incorrect. 	<ul style="list-style-type: none"> • Register the processing for the event (interrupt) and perform the operation again. • Check the specified CPU number.
-260	FEFCH	■Drive mount error ■Drive unmount error The mount processing or unmount processing of the drive failed.	<ul style="list-style-type: none"> • Retry. • Check if the drive is damaged. • Replace the drive.
-264	FEF8H	■Pointer error The address of the specified pointer is incorrect.	Check the address of the specified pointer.
-267	FEF5H	■Authentication error The password is incorrect.	Check the specified password.
-269	FEF3H	■Network number error The specified network number is out of range.	Check the specified network number.
-281	FEE7H	■Instruction name error The instruction name is incorrect.	Check the specified instruction name.
-282	FEE6H	■Mode information error The specified mode information is out of range.	Check the specified mode information.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-283	FEE5H	■Operation selection mode The specified operation selection mode is out of range.	Check the specified operation selection mode.
-288	FEE0H	■Individual identification information read error Reading individual identification information failed.	<ul style="list-style-type: none"> • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON.
-289	FEDFH	■Dot matrix LED mode selection error An operation other than "USER" was selected by using the CCPU_SetOpSelectMode function or with the MODE/SELECT switch.	Select "USER" by using the CCPU_SetOpSelectMode function or with the MODE/SELECT switch.
-290	FEDEH	■MODE is being selected by switch operation The CCPU_SetOpSelectMode function or the CCPU_SetDotMatrixLED function was executed while an operation was being selected with the MODE/SELECT switch.	Execute the function after selecting an operation.
-291	FEDDH	■Fixed cycle communication area unreserved error An attempt was made to access a CPU module in which fixed cycle communication area is not reserved.	Use the fixed cycle communication function in the multiple CPU setting of system parameter, and check if 1K word or more is set for the fixed cycle communication area.
-292	FEDCH	■Program memory shutdown error The shutdown processing of the program memory failed.	<ul style="list-style-type: none"> • Check if files in the program memory are being accessed. • Check if all files in the program memory have been closed.
-293	FEDBH	■Data memory shutdown error The shutdown processing of the data memory failed.	<ul style="list-style-type: none"> • Check if files in the data memory are being accessed. • Check if all files in the data memory have been closed.
-295	FED9H	■Selected operation is being checked The CCPU_SetDotMatrixLED function was executed while checking the selected operation.	Execute the CCPU_SetDotMatrixLED function after checking the operation.
-296	FED8H	■Setting data size error The setting data size is out of range.	Check the setting data size.
-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.	
-299	FED5H	■Input/output number, network number incorrect specification <ul style="list-style-type: none"> • 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. 	
-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.
-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.	
-303	FED1H	■Input/output number, network number incorrect specification An I/O module or intelligent function module controlled by other CPU modules was specified.	<ul style="list-style-type: none"> • 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.
-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.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-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 consult your local Mitsubishi 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.	<ul style="list-style-type: none"> • Correct the argument of the function. • Check if the firmware version of the C Controller module supports the dedicated instruction specified in the program.
-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.
-312	FEC8H	■Operation error The CCPU_DedicatedDInst function was executed in the state where the fixed cycle communication function is set to "Not Use" in the multiple CPU setting of system parameters.	Change the fixed cycle communication function in the multiple CPU setting to "Use".
-313	FEC7H	■Operation error In a multiple CPU system, the number of data points which exceeds the usable system area size in each CPU was specified.	Correct the number of data points for the CCPU_DedicatedDInst 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 consult your local Mitsubishi representative.
-315	FEC5H		
-316	FEC4H		
-317	FEC3H	■Other CPU module major error An error was detected in other CPU modules when a function was executed.	Reset the CPU module. 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 consult your local Mitsubishi representative.
-318	FEC2H	■System bus error An error was detected on the system bus.	<ul style="list-style-type: none"> • Take measures to reduce noise. • Reset the CPU module. 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 consult your local Mitsubishi representative.
-319	FEC1H	■Hardware failure A hardware failure was detected.	<ul style="list-style-type: none"> • Take measures to reduce noise. • Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-320	FEC0H	■Clock rate specification error The specified clock rate is out of range.	Check the specified clock rate.
-321	FEBFH	■System bus error An error was detected on the system bus.	<ul style="list-style-type: none"> • Take measures to reduce noise. • Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-322	FEBEH	■System bus error An error was detected on the system bus.	<ul style="list-style-type: none"> • Check the connection status of the extension cable. • Take measures to reduce noise. • Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-323	FEBDH	■System bus error An error was detected on the system bus.	<ul style="list-style-type: none"> • Take measures to reduce noise. • Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-324	FEBCH	■System bus error An error was detected on the system bus.	<ul style="list-style-type: none"> • Take measures to reduce noise. • Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-325	FEBBH	■System bus error An error was detected on the system bus.	<ul style="list-style-type: none"> Take measures to reduce noise. Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-326	FEBAH	■System bus error	<ul style="list-style-type: none"> Take measures to reduce noise. Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-327	FEB9H	■Module major error <ul style="list-style-type: none"> 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. 	<ul style="list-style-type: none"> Check the connection status of the extension cable. Reset the CPU module. If the same error code is displayed again, the possible cause is a hardware failure of the CPU module. Please consult your local Mitsubishi representative.
-328	FEB8H	■Group No. specification error The specified group number is out of range.	Check the specified group number.
-329	FEB7H	■Link time specification error The specified link scan time is out of range.	Check the specified link scan time.
-330	FEB6H	■Number of detections of disconnected remote stations error The specified number of detections of disconnected remote stations is out of range.	Check the specified number of detections of disconnected remote stations.
-331	FEB5H	■CC-Link IE Field Network Basic parameter unset The function is executed without setting CC-Link IE Field Network Basic parameters.	Set the CC-Link IE Field Network Basic parameters with CW Configurator.
-332	FEB4H	■Remote station timeout time specification error	Check the timeout time for the disconnection detection of the specified remote station.
-335	FEB1H	■Assurance of link scan data waiting time timed out The specified waiting time elapsed.	Increase the timeout time specified to the argument.
-336	FEB0H	■Remote station number specification error The specified remote station number does not exist in the remote station.	Check the specified remote station number.
-340	FEACH	■Incorrect password The specified password for removing the prohibition on the firmware update is incorrect.	Check the specified password for removing the prohibition on the firmware update.
-342	FEAAH	■Number of characters exceeded The number of characters in the specified password for removing the prohibition on the firmware update is out of the range.	Check the number of characters in the specified password for removing the prohibition on the firmware update.
-343	FEA9H	■Unavailable character in password Unavailable character is specified for the password for removing the prohibition on the firmware update.	Check the specified password for removing the prohibition on the firmware update.
-344	FEA8H	■Firmware update enabled Firmware update prohibition is removed while the firmware update is not prohibited.	No action is required because the firmware update is enabled.
-345	FEA7H	■Firmware update prohibited Firmware update is prohibited again while the firmware update has already been prohibited.	When changing a password, remove the prohibition on the firmware update, and then execute the function again.
-350	FEA2H	■Module label ID mismatch error The specified module label ID and the module label ID written to a C Controller module are different.	Check that the module label ID ^{*1} written to a C Controller module and the specified module label ID ^{*2} match. For how to use module labels, refer to the following: ■MELSEC iQ-R C Controller Module User's Manual
-351	FEA1H	■Module label error The specified module label does not exist.	Specify the module label (public label) written to the C Controller module.

*1 Can be checked with 'Module label ID' (SD1596 to 1597). (■MELSEC iQ-R C Controller Module User's Manual)










*2 Can be checked with a header file used in user programs.

4.3 MELSEC Data Link Functions

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

Error code*1		Description	Corrective action
Decimal	Hexadecimal		
-1	FFFFH	■Path error <ul style="list-style-type: none"> The specified path is unavailable. The taskDelete was executed in the task using a MELSEC data link function. The task using a MELSEC data link function was deleted with the taskDelete. 	<ul style="list-style-type: none"> Use a path pointer returned with the mdOpen function. Check if the taskDelete was executed in the task using 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 <ul style="list-style-type: none"> 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 for reading/writing device randomly exceeds the device range. 	<ul style="list-style-type: none"> Check the start device number of the specified device. Check the device number plus the number of points. Specify the start device number of bit device in multiples of 8. Check if the specified device is available in the target station.
-3	FFFDH	■Device type error The specified device type is unavailable.	<ul style="list-style-type: none"> Check if the device is accessible to the specified network number and station number. (☞ Page 17 Accessible range and devices) Check if the specified device is available in the target station.
-5	FFFBH	■Size error <ul style="list-style-type: none"> 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. The total of the points that are specified for each block number of the mdRandREx function or the mdRandWEx function exceeds 10240. 	<ul style="list-style-type: none"> Check the specified device size. Check the device number and the size. Specify an even-number byte. Reduce the total points that are specified for each block number of the mdRandREx function or the mdRandWEx function 10240 or less.
-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.	<ul style="list-style-type: none"> 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.	<ul style="list-style-type: none"> 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 <ul style="list-style-type: none"> The specified station number or network number is out of range. A device which cannot be accessed by the target station is specified. 	<ul style="list-style-type: none"> 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.	<ul style="list-style-type: none"> 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/RCV channel number error The channel number specified in the SEND/RCV function is out of range.	Specify the channel number within the range. <ul style="list-style-type: none"> CC-Link IE Controller Network: 1 to 8 CC-Link IE Field Network: 1 to 2 CC-Link IE TSN: 1 to 8 MELSECNET/H network: 1 to 8


Error code*1		Description	Corrective action
Decimal	Hexadecimal		
-31	FFE1H	■Module load error Loading modules required for executing functions failed.	<ul style="list-style-type: none"> • The memory may be insufficient. End another running task or reduce the access size. • Check the status of the system drive of the C Controller module.
-32	FFE0H	■Resource timeout error The resource is being used by another task/thread and is not released within 30 seconds.	<ul style="list-style-type: none"> • Retry. • The memory may be insufficient. End another running task. • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON.
-33	FFDFH	■Communication target unsupported error The module specified as the communication target by a network number and station number is not supported.	<ul style="list-style-type: none"> • 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 read error Reading parameter files from the registry failed.	<ul style="list-style-type: none"> • Check if the access target is correctly set with CW Configurator. • Check if the setting for the channel number is enabled. • Reset the C Controller module or turn the power OFF and ON after checking the parameters with CW Configurator again and writing them. • Check if the access target module supports dedicated function libraries. (MELSEC iQ-R C Controller Module User's Manual)
-37	FFDBH	■Communications initialization error Initializing the setting for communication failed.	<ul style="list-style-type: none"> • Retry. • The memory may be insufficient. End another running task. • Check the available memory capacity. • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON.
-42	FFD6H	■Close error Communications cannot be closed.	<ul style="list-style-type: none"> • Retry. • Check if the C Controller module is running normally. • Reset the C Controller module, or turn the power OFF and ON.
-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.
-80	FFB0H	■Connection destination CPU error The connection destination CPU is not an RCP.	Connect an RCP.
-81	FFAFH	■Label code mismatch error Label assignment information of the CPU module was changed.	Acquire label information using the mdGetLabelInfo function again.
-82	FFAEH	■Label incorrect value error An incorrect label name was specified. <ul style="list-style-type: none"> • 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.
-475 to -3839	FE25H to F101H	Refer to the following: Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)	
-4097 to -8192	EFFFH to E000H	Refer to the following: MELSEC iQ-R CC-Link IE Controller Network User's Manual (Application) MELSEC-Q CC-Link IE Controller Network Reference Manual	

Error code*1		Description	Corrective action
Decimal	Hexadecimal		
-8193 to -12288	DFFFH to D000H	Refer to the following:  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:  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:  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 function of which the return value is a long-type, the value will be eight digits in hexadecimal.

4.4 Motion Module Dedicated Class

The following table shows error codes for the motion module dedicated class.

Error code*1		Description	Corrective action
Decimal	Hexadecimal		
1024	400H	■I/O number specification error The input/output number of an instance is changed from the one specified at the first normal completion of the MCFB setting function, and the MCFB setting function or MCFB::RefreshLabels function is executed again in the same instance.	Specify the input/output number specified at the first normal completion of the MCFB setting function and execute the function or MCFB::RefreshLabels function.
1025	401H	■Input/output number out of range error An out-of-range value is specified for the input/output number when executing the MCFB setting function.	Specify a value within 0H to FEH for the input/output number.
1026	402H	■Refresh not executed error <ul style="list-style-type: none"> The MCFB::RefreshLabels function is not executed after executing the MCFB setting function. The refresh processing is not performed on the motion module side. 	<ul style="list-style-type: none"> Execute the MCFB setting function again after executing the MCFB::RefreshLabels function. When the error occurs after executing the MCFB::RefreshLabels function, execute the MCFB setting function after the appropriate time passes in accordance with a system.
1027	403H	■Function not executed error The MCFB setting function is not executed or the execution failed in the same instance.	After the execution of MCFB setting function is completed normally, execute the MCFB::RefreshLabels function again.
1028	404H	■Memory reservation error <ul style="list-style-type: none"> Reserving sufficient memory failed. A motion module dedicated class is executed in the interrupt service routine (ISR). 	<ul style="list-style-type: none"> The memory may be insufficient. End another running task. Reduce the number of tasks which use the dedicated function library and retry the operation. Check if the C Controller module is running normally. Reset the C Controller system. To execute other dedicated functions synchronously with an interrupt, implement the notification processing in a user program and perform the processing in a task.
1029	405H	■Unsupported function execution error An unsupported function is executed.	Execute a supported function.
1030	406H	■Motion module not ready at the initial execution <ul style="list-style-type: none"> The 'read' signal of the motion module is turned OFF at the initial execution of the MCFB setting function. The module corresponding to the specified input/output number is not a motion module. 	<ul style="list-style-type: none"> Check that the 'ready' signal is turned ON and execute the MCFB setting function again. For the 'ready' signal of motion module, refer to the 'MELSEC iQ-R Motion Module User's Manual (Application).' Check that the module corresponding to the input/output number, which is specified when executing the MCFB setting function, is a motion module.
-345 to -201	FEA7H to FF37H	Error codes returned by a C Controller module dedicated function	<ul style="list-style-type: none"> Check that the module corresponding to the input/output number, which is specified when executing the MCFB setting function, is a motion module. If the specified input/output number corresponds to the motion module, check the status of the motion module. For the motion module status, refer to 'MELSEC iQ-R Motion Module User's Manual (Application).'
-28672 to -28150	9000H to 920AH	Common error codes for C Controller module dedicated functions and MELSEC data link functions	Refer to the corresponding error code.  Page 222 Common Error Codes

4.5 Error Codes Different From Conventional Functions

Error code (return value) for replaced functions may differ from the one for conventional functions. Be sure to refer to the list of error code in this manual ( Page 222 ERROR CODE LIST) to perform the troubleshooting.

APPENDIX

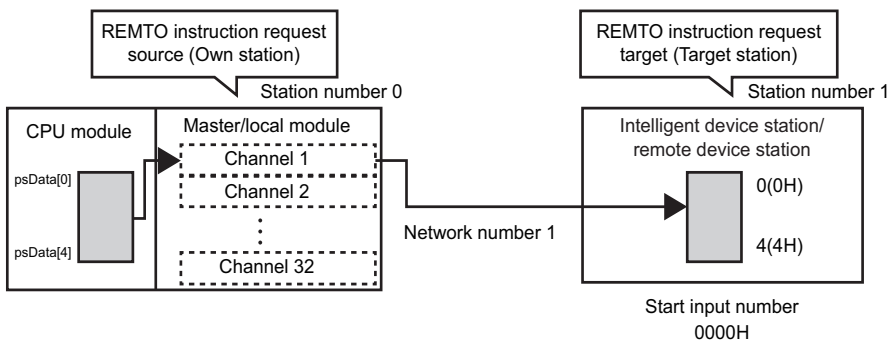
Appendix 1 Example for Replacing Ladder by C Language

This section shows program examples for replacing ladder by C language.

Program example

The following shows a program for writing the data in psData[5] of the station number 0 (own station) to the buffer memory (address: 0 to 4) of the intelligent device station/remote device station on the station number 1 (target station).

System configuration example



The CPU module is a programmable controller CPU

■Example for using the ladder program

(0)	SM400		JP.REMTO	J1	K1	K1	K0	K0	D0	K5	M0
(15)											{END}

Precautions

For a programmable controller CPU, in the program example, the data for five words (D0 to D4) is written since the start address of the buffer memory is specified with a device.

The CPU module is C Controller module

■Example for using the C Controller module dedicated function

```
short CCPU_DedicatedJInstSample(void){
    short sRet=0; /*(1)*/
    char pcInstName[8]="REMT0"; /*(2)*/
    short sNetNo=1; /*(3)*/
    short sChan=1; /*(4)*/
    short sStNo=1; /*(5)*/
    short sloNo=0x0000; /*(6)*/
    short sAdd=0; /*(7)*/
    short psData[5]={1,2,3,4,5}; /*(8)*/
    short sSize=5; /*(9)*/
    short psCmp[2]={0,0}; /*(10)*/


    /*(11)*/
    sRet=CCPU_DedicatedJInst(
        pcInstName,
        sNetNo,
        1,
        &sChan,
        1,
        &sStNo,
        1,
        &sloNo,
        1,
        &sAdd,
        1,
        psData,
        5,
        &sSize,
        1,
        psCmp,
        2,
        NULL,
        0,
        NULL,
        0
    );
    return sRet;
}
```

- (1) Return value of the CCPU_DedicatedJInst function
- (2) Instruction
- (3) Target network No. (1 to 239)
- (4) Own station channel (1 to 32)
- (5) Target station number (1 to 120)
- (6) Start input/output number of an intelligent function module (0x0000 to 0x00FE)
- (7) Start address of the buffer memory (0 to 65535)
- (8) Data to be written
- (9) Number of units of data to be written (1 to 240 words)
- (10) Instruction completion result
- (11) Execution of a dedicated instruction

Appendix 2 How to Replace a Q12DCCPU-V

Replacement of projects

Import the Q12DCCPU-V projects by using the Import function of CW Workbench (SW1DND-CWWR-E/EZ/EVZ). Select the "Build Support and Specs" tab on the screen of property for the imported project, and change "Active build spec" to "ARMARCH7gnu_SMP".*1

*1 For details on importing projects and changing "Active build spec", refer to the following:
 CW Workbench/CW-Sim Operating Manual

Replacement of VxWorks standard API functions

The operating system of R12CCPU-V has been upgraded from Q12DCCPU-V.
(Q12DCCPU-V: VxWorks 6.4 → R12CCPU-V: VxWorks 6.9)

To replace VxWorks standard API function, refer to "MIGRATION GUIDE" of VxWorks.*1

*1 PDF file of VxWorks "MIGRATION GUIDE" is included in CW Workbench.

Replacement of functions

If any of the functions listed in Page 240 Correspondence Table to Q12DCCPU-V Functions is used in the user program, replace the function.*1

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

Replacement of device type


The device types listed on the following table are deleted from R12CCPU-V.

If any of the following device type is used in the user program to be replaced, perform the alternative method shown in the "Alternative method" column. The methods described in the following section are available as the alternative methods.

 Page 238 Alternative method

Bus interface functions

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

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Link input internal buffer	—	QBFDDev_LXBuf	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method.  Page 238 Refreshing device, The access target is a network module on the own station.
Link output internal buffer	—	QBFDDev_LYBuf	
Link relay internal buffer	—	QBFDDev_LBBuf	
Link register internal buffer	—	QBFDDev_LWBuf	

MELSEC data link functions

■Device types for accessing CC-Link modules

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Own station remote input	RX	DevX	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ Page 238 Refreshing device, The access target is a network module on the own station. ☞ Page 238 Module access device, The access target is 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	
Own station link special relay ^{*1}	SB	DevSM	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ Page 238 Module access device, The access target is a network module on the own station.
Own station link special register ^{*2}	SW	DevSD	
Own station link special relay ^{*1}	SB	DevQSB	
Own station link special register ^{*2}	SW	DevQSW	
Own station random access buffer	—	DevMRB	
Own station buffer memory	—	DevSPB	
Other station buffer memory	—	DevRBM	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ Page 238 Module access device, The access target is a network module on the own station.
Other station random access buffer	—	DevRAB	
Other station remote input	—	DevRX	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ Page 238 Refreshing device, The access target is a network module on the own station. ☞ Page 238 Module access device, The access target is a network module on the own station.
Other station remote output	—	DevRY	
Other station link register	—	DevRW	
Other station link special relay	—	DevSB	
Other station link special register	—	DevSW	

*1 The own station link special relay (SB) has two device type definitions; DevSM and DevQSB. Either of them can be specified for the same operation.

*2 The own station link special register (SW) has two device type definitions; DevSD and DevQSW. Either of them can be specified for the same operation.

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

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Own station link input internal buffer (LX buffer)	—	DevX	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ Page 238 Refreshing device, The access target is a network module on the own station. ☞ Page 238 CCPU_ReadLinkDevice/CCPU_WriteLinkDevice
Own station link output internal buffer (LY buffer)	—	DevY	
Own station link relay internal buffer (LB buffer)	—	DevB	
Own station link register internal buffer (LW buffer)	—	DevW	
Own station direct link input	LX	DevLX(0)	
Own station direct link output	LY	DevLY(0)	
Own station direct link relay	LB	DevLB(0)	
Own station direct link register	LW	DevLW(0)	
Own station direct link special relay ^{*1}	SB	DevSM, DevQSB, DevLSB(0)	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ Page 238 CCPU_ReadLinkDevice/CCPU_WriteLinkDevice
Own station direct link special register ^{*2}	SW	DevSD, DevQSW, DevLSW(0)	
Buffer memory	—	—	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ Page 238 Module access device, The access target is a network module on the own station.

*1 The own station direct link special relay (SB) has three device type definitions; DevSM, DevQSB, and DevLSB(0). Any of them can be specified for the same operation.

*2 The own station direct link special register (SW) has three device type definitions; DevSD, DevQSW, and DevLSW(0). Any of them can be specified for the same operation.

A

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

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Own station remote input	RX	DevLX(0)	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ Page 238 Refreshing device, The access target is a network module on the own station. ☞ Page 238 Module access device, The access target is a network module on the own station. ☞ Page 238 CCPU_ReadLinkDevice/CCPU_WriteLinkDevice
Own station remote output	RY	DevLY(0)	
Own station remote register (for sending)	RWw	DevLW(0)	
Own station remote register (for receiving)	RWr		
Own station direct link special relay ^{*1}	SB	DevSM, DevQSB, DevLSB(0)	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ Page 238 CCPU_ReadLinkDevice/CCPU_WriteLinkDevice
Own station direct link special register ^{*2}	SW	DevSD, DevQSW, DevLSW(0)	
Buffer memory	—	DevSPB	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ Page 238 Module access device, The access target is a network module on the own station.

*1 The own station direct link special relay (SB) has three device type definitions; DevSM, DevQSB, and DevLSB(0). Any of them can be specified for the same operation.

*2 The own station direct link special register (SW) has three device type definitions; DevSD, DevQSW, and DevLSW(0). Any of them can be specified for the same operation.

Alternative method

■Refreshing device

Access target	Alternative method
The access target is a network module on the own station.	Configure the refresh setting so that a device of C Controller module, M, B, D, W, or ZR is refreshed by a link device of a network module.
	Use the MELSEC data link functions to access a device of C Controller module, M, B, D, W or ZR.
The access target is a network module on the other station.	Configure the refresh setting for CPU module on the other station so that a device of C Controller module on the other station is refreshed by a link device of a network module.
	Specify the other station to the network number and station number in the MELSEC data link functions to access a device of CPU module on the other station.

■Module access device

Access target	Alternative method
The access target is a network module on the own station.	Open a communication line by specifying 'bus interface' to the channel in the mdOpen function.
	Specify the module access device (DevSPG) to the device type in the MELSEC data link functions, and access the area to which link devices are assigned ^{*1} in the buffer memory of a network module.
The access target is a network module on the other station.	Open a communication line by specifying the channel name corresponding to each network to the channel in the mdOpen function.
	Specify the other station to the network number and station number in the MELSEC data link functions.
	Specify the module access device (DevSPG) to the device type in the MELSEC data link functions, and access the area to which link devices are assigned ^{*1} in the buffer memory of a network module.

*1 For details on the buffer memory address to which link devices are assigned, refer to the manual for the network module to be accessed.

■CCPU_ReadLinkDevice/CCPU_WriteLinkDevice

- Access the own station link device of a network module by using the CCPU_ReadLinkDevice/CCPU_WriteLinkDevice function. For details, refer to the relevant functions.

☞ Page 118 CCPU_ReadLinkDevice, Page 150 CCPU_WriteLinkDevice

Compilation of replaced project

Compile the replaced project in CW Workbench.

Appendix 3 Correspondence Table to Q12DCCPU-V Functions

○: Conventional functions can be used. ×: Conventional functions cannot be used.

—: Replacement of functions is not required. Not available: No functions are available to be replaced.

C Controller module dedicated functions

Function name (Q12DCCPU-V)	Mode type	Availability in R12CCPU-V	Function name (replaced)
CCPU_ClearError	Extended mode	○	—
CCPU_EntryWDTInt	Extended mode	○	—
CCPU_Get7SegLED	Extended mode	×	CCPU_GetDotMatrixLED
CCPU_GetCpuStatus	Extended mode	○	—
CCPU_GetErrInfo	Extended mode	○	—
CCPU_GetLEDStatus	Extended mode	○	—
CCPU_GetPowerStatus	Extended mode	○	—
CCPU_GetRefreshStatus	Extended mode	×	CCPU_GetConstantProcessStatus
CCPU_GetRTC	Extended mode	○	—
CCPU_GetSwitchStatus	Extended mode	○	—
CCPU_MountMemoryCard	Extended mode	○	—
CCPU_ReadSRAM	Extended mode	×	CCPU_ReadDevice ^{*1}
CCPU_RegistEventLog	Extended mode	○	—
CCPU_ResetWDT	Extended mode	○	—
CCPU_Set7SegLED	Extended mode	×	CCPU_SetDotMatrixLED
CCPU_SetLEDStatus	Extended mode	○	—
CCPU_SetRTC	Extended mode	○	—
CCPU_StartWDT	Extended mode	○	—
CCPU_StopWDT	Extended mode	○	—
CCPU_UnmountMemoryCard	Extended mode	○	—
CCPU_WriteSRAM	Extended mode	×	CCPU_WriteDevice ^{*1}
CCPU_ChangeFileSecurity	Extended mode	○	—
CCPU_GetFileSecurity	Extended mode	○	—
CCPU_CommunicateMCPProtocol	Extended mode	×	—
CCPU_SetOpenNoMCPProtocol	Extended mode	×	—

*1 Use ZR device as a substitute.

C Controller module dedicated functions for ISR

Function name (Q12DCCPU-V)	Mode type	Availability in R12CCPU-V	Function name (replaced)
CCPU_Get7SegLED_ISR	Extended mode	×	CCPU_GetDotMatrixLED_ISR
CCPU_Set7SegLED_ISR	Extended mode	×	CCPU_SetDotMatrixLED_ISR
CCPU_ReadSRAM_ISR	Extended mode	×	CCPU_ReadDevice_ISR ^{*1}
CCPU_WriteSRAM_ISR	Extended mode	×	CCPU_WriteDevice_ISR ^{*1}
CCPU_SetLEDStatus_ISR	Extended mode	○	—

*1 Use ZR device as a substitute.

Bus interface functions

Function name (Q12DCCPU-V)	Mode type	Availability in R12CCPU-V	Function name (replaced)
QBF_Close	Basic mode/Extended mode	×	Not available
QBF_ControlEx	Basic mode/Extended mode	×	CCPU_Control
QBF_ControlProgram	Basic mode/Extended mode	×	Not available
QBF_FromBuf	Basic mode/Extended mode	×	CCPU_FromBuf, CCPU_FromBufHG
QBF_GINT	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionCHGA	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionCHGT	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionCHGT2	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionCHGV	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionDDRD	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionDDWR	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionSFCS	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionSVST	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_Open	Basic mode/Extended mode	×	Not available
QBF_ReadDevice	Basic mode/Extended mode	×	CCPU_ReadDevice
QBF_ReadLinkDevice	Basic mode/Extended mode	×	CCPU_ReadLinkDevice
QBF_RECV	Basic mode/Extended mode	×	CCPU_DedicatedGInst, CCPU_DedicatedJInst
QBF_RefreshLinkDevice	Basic mode/Extended mode	×	Not available
QBF_Reset	Basic mode/Extended mode	×	CCPU_Reset
QBF_ResetDevice	Basic mode/Extended mode	×	CCPU_ResetDevice
QBF_SEND	Basic mode/Extended mode	×	CCPU_DedicatedGInst, CCPU_DedicatedJInst
QBF_SetDevice	Basic mode/Extended mode	×	CCPU_SetDevice
QBF_ToBuf	Basic mode/Extended mode	×	CCPU_ToBuf, CCPU_ToBufHG
QBF_UnitInfo	Basic mode/Extended mode	×	CCPU_GetUnitInfo
QBF_WaitEvent	Basic mode/Extended mode	×	CCPU_WaitEvent
QBF_WaitUnitEvent	Basic mode/Extended mode	×	CCPU_WaitUnitEvent
QBF_WriteDevice	Basic mode/Extended mode	×	CCPU_WriteDevice
QBF_WriteLinkDevice	Basic mode/Extended mode	×	CCPU_WriteLinkDevice
QBF_X_In_BitEx	Basic mode/Extended mode	×	CCPU_X_In_BitEx
QBF_X_In_WordEx	Basic mode/Extended mode	×	CCPU_X_In_WordEx
QBF_Y_In_BitEx	Basic mode/Extended mode	×	CCPU_Y_In_BitEx
QBF_Y_In_WordEx	Basic mode/Extended mode	×	CCPU_Y_In_WordEx
QBF_Y_Out_BitEx	Basic mode/Extended mode	×	CCPU_Y_Out_BitEx
QBF_Y_Out_WordEx	Basic mode/Extended mode	×	CCPU_Y_Out_WordEx
QBF_MotionCHGVS	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_MotionCHGAS	Basic mode/Extended mode	×	CCPU_DedicatedDInst
QBF_REMTO	Extended mode	×	CCPU_DedicatedJInst
QBF_REMFR	Extended mode	×	CCPU_DedicatedJInst
QBF_DisableCpuInt	Basic mode/Extended mode	×	Not available
QBF_DisableMultiCPUSyncInt	Basic mode/Extended mode	×	CCPU_DisableInt
QBF_DisableUnitInt	Basic mode/Extended mode	×	CCPU_DisableInt
QBF_EnableCpuInt	Basic mode/Extended mode	×	Not available
QBF_EnableMultiCPUSyncInt	Basic mode/Extended mode	×	CCPU_EnableInt
QBF_EnableUnitInt	Basic mode/Extended mode	×	CCPU_EnableInt
QBF_EntryCpuInt	Basic mode/Extended mode	×	Not available
QBF_EntryMultiCPUSyncInt	Basic mode/Extended mode	×	CCPU_EntryInt
QBF_EntryUnitInt	Basic mode/Extended mode	×	CCPU_EntryInt

Function name (Q12DCCPU-V)	Mode type	Availability in R12CCPU-V	Function name (replaced)
QBF_ClearError	Basic mode/Extended mode	×	CCPU_ClearError
QBF_Control	Basic mode/Extended mode	×	CCPU_Control
QBF_Control7SegLED	Basic mode/Extended mode	×	CCPU_SetDotMatrixLED
QBF_ControlLED	Basic mode/Extended mode	×	CCPU_SetLEDStatus
QBF_EntryTimerEvent	Basic mode/Extended mode	×	CCPU_EntryTimerEvent
QBF_EntryWDTInt	Basic mode/Extended mode	×	CCPU_EntryWDTInt
QBF_GetTime	Basic mode/Extended mode	×	CCPU_GetRTC
QBF_MountCfCard	Basic mode/Extended mode	×	CCPU_MountMemoryCard
QBF_ReadSRAM	Basic mode/Extended mode	×	CCPU_ReadDevice ^{*1}
QBF_ReadStatusEx	Basic mode/Extended mode	×	CCPU_GetCpuStatus
QBF_RegistEventLog	Basic mode/Extended mode	×	CCPU_RegistEventLog, CCPU_RegistEventLog_ISR
QBF_ResetWDT	Basic mode/Extended mode	×	CCPU_ResetWDT
QBF_SetTime	Basic mode/Extended mode	×	CCPU_SetRTC
QBF_ShutdownRom	Basic mode/Extended mode	×	CCPU_ShutdownRom
QBF_StartWDT	Basic mode/Extended mode	×	CCPU_StartWDT
QBF_StopWDT	Basic mode/Extended mode	×	CCPU_StopWDT
QBF_UnmountCfCard	Basic mode/Extended mode	×	CCPU_UnmountMemoryCard
QBF_WaitTimerEvent	Basic mode/Extended mode	×	CCPU_WaitTimerEvent
QBF_WriteSRAM	Basic mode/Extended mode	×	CCPU_WriteDevice ^{*1}

^{*1} Use ZR device as a substitute.

Bus interface functions for ISR

Function name (Q12DCCPU-V)	Mode type	Availability in R12CCPU-V	Function name (replaced)
QBF_DisableCpuInt_ISR	Basic mode/Extended mode	×	Not available
QBF_DisableMultiCPUSyncInt_ISR	Basic mode/Extended mode	×	CCPU_DisableInt_ISR
QBF_DisableUnitInt_ISR	Basic mode/Extended mode	×	CCPU_DisableInt_ISR
QBF_EnableCpuInt_ISR	Basic mode/Extended mode	×	Not available
QBF_EnableMultiCPUSyncInt_ISR	Basic mode/Extended mode	×	CCPU_EnableInt_ISR
QBF_EnableUnitInt_ISR	Basic mode/Extended mode	×	CCPU_EnableInt_ISR
QBF_FromBuf_ISR	Basic mode/Extended mode	×	CCPU_FromBuf_ISR
QBF_ReadDevice_ISR	Basic mode/Extended mode	×	CCPU_ReadDevice_ISR
QBF_ResetDevice_ISR	Basic mode/Extended mode	×	CCPU_ResetDevice_ISR
QBF_SetDevice_ISR	Basic mode/Extended mode	×	CCPU_SetDevice_ISR
QBF_ToBuf_ISR	Basic mode/Extended mode	×	CCPU_ToBuf_ISR
QBF_WriteDevice_ISR	Basic mode/Extended mode	×	CCPU_WriteDevice_ISR
QBF_X_In_Word_ISR	Basic mode/Extended mode	×	CCPU_X_In_Word_ISR
QBF_Y_In_Word_ISR	Basic mode/Extended mode	×	CCPU_Y_In_Word_ISR
QBF_Y_Out_Word_ISR	Basic mode/Extended mode	×	CCPU_Y_Out_Word_ISR
QBF_ControlLED_ISR	Basic mode/Extended mode	×	CCPU_SetLEDStatus_ISR
QBF_Control7SegLED_ISR	Basic mode/Extended mode	×	CCPU_SetDotMatrixLED_ISR
QBF_WriteSRAM_ISR	Basic mode/Extended mode	×	CCPU_WriteDevice_ISR ^{*1}
QBF_ReadSRAM_ISR	Basic mode/Extended mode	×	CCPU_ReadDevice_ISR ^{*1}

^{*1} Use ZR device as a substitute.

MELSEC data link functions


Function name (Q12DCCPU-V)	Mode type	Availability in R12CCPU-V	Function name (replaced)
mdClose	Basic mode/Extended mode	○	—
mdControl	Basic mode/Extended mode	○	—
mdDevRstEx	Basic mode/Extended mode	○	—
mdDevSetEx	Basic mode/Extended mode	○	—
mdInit	Basic mode/Extended mode	○	—
mdOpen	Basic mode/Extended mode	○	—
mdRandREx	Basic mode/Extended mode	○	—
mdRandWEx	Basic mode/Extended mode	○	—
mdReceiveEx (Device batch read function)	Basic mode/Extended mode	○	—
mdReceiveEx (Message receive function)	Basic mode/Extended mode	○	—
mdSendEx (Device batch write function)	Basic mode/Extended mode	○	—
mdSendEx (Message send function)	Basic mode/Extended mode	○	—
mdTypeRead	Basic mode/Extended mode	○	—
mdDevRst	Basic mode	×	mdDevRstEx
mdDevSet	Basic mode	×	mdDevSetEx
mdRandR	Basic mode	×	mdRandREx
mdRandW	Basic mode	×	mdRandWEx
mdReceive (Device batch read function)	Basic mode	×	mdReceiveEx (Device batch read function)
mdReceive (Message receive function)	Basic mode	×	mdReceiveEx (Message receive function)
mdSend (Device batch write function)	Basic mode	×	mdSendEx (Device batch write function)
mdSend (Message send function)	Basic mode	×	mdSendEx (Message send function)

A

Appendix 4 How to Replace a Q06CCPU-V

Replacement of projects

Create a new project in CW Workbench and add source code. *1

*1 For creating a project, refer to the following:
 CW Workbench/CW-Sim Operating Manual

Replacement of VxWorks standard API functions


The operating system of R12CCPU-V has been upgraded from Q06CCPU-V.
(Q06CCPU-V:VxWorks 5.4 → R12CCPU-V: VxWorks 6.9)

To replace VxWorks standard API function, refer to "MIGRATION GUIDE" of VxWorks. *1
*1 PDF file of VxWorks "MIGRATION GUIDE" is included in CW Workbench.

Replacement of functions


If any of the functions listed in Page 248 Correspondence Table to Q06CCPU-V Functions is used in the user program, replace the function. *1
*1 Check the specifications of the function before replacement since changing arguments may be required for the replacement in some case.

Replacement of device type

The device types listed on the following table are deleted from R12CCPU-V.
If any of the following device type is used in the user program to be replaced, perform the alternative method shown in the "Alternative method" column. The methods described in the following section are available as the alternative methods.
 Page 246 Alternative method

Bus interface functions

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

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Link input internal buffer	—	QBFDDev_LXBuf	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method.  The access target is a network module on the own station.(refreshing device)
Link output internal buffer	—	QBFDDev_LYBuf	
Link relay internal buffer	—	QBFDDev_LBBuf	
Link register internal buffer	—	QBFDDev_LWBuf	

MELSEC data link functions

■Device types for accessing CC-Link modules

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Own station remote input	RX	DevX	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ The access target is a network module on the own station.(refreshing device) ☞ The access target is a network module on the own station.(module access device)
Own station remote output	RY	DevY	
Own station link register (for sending)	—	DevWw	
Own station link register (for receiving)	—	DevWr	
Own station link special relay ^{*1}	SB	DevSM	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ The access target is a network module on the own station.(module access device)
Own station link special register ^{*2}	SW	DevSD	
Own station link special relay ^{*1}	SB	DevQSB	
Own station link special register ^{*2}	SW	DevQSW	
Own station random access buffer	—	DevMRB	
Own station buffer memory	—	DevSPB	
Other station buffer memory	—	DevRBM	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ The access target is a network module on the other station.(module access device)
Other station random access buffer	—	DevRAB	
Other station remote input	—	DevRX	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ The access target is a network module on the other station.(refreshing device) ☞ The access target is a network module on the other station.(module access device)
Other station remote output	—	DevRY	
Other station link register	—	DevRW	
Other station link special relay	—	DevSB	
Other station link special register	—	DevSW	

*1 The own station link special relay (SB) has two device type definitions; DevSM and DevQSB. Either of them can be specified for the same operation.

*2 The own station link special register (SW) has two device type definitions; DevSD and DevQSW. Either of them can be specified for the same operation.

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

Device type deleted from R12CCPU-V			Alternative method
Device		Device name specification	
Own station link input internal buffer (LX buffer)	—	DevX	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with either of the methods below. ☞ The access target is a network module on the own station.(refreshing device) ☞ Page 246 CCPU_ReadLinkDevice/CCPU_WriteLinkDevice
Own station link output internal buffer (LY buffer)	—	DevY	
Own station link relay internal buffer (LB buffer)	—	DevB	
Own station link register internal buffer (LW buffer)	—	DevW	
Own station direct link input	LX	DevLX(0)	
Own station direct link output	LY	DevLY(0)	
Own station direct link relay	LB	DevLB(0)	
Own station direct link register	LW	DevLW(0)	
Own station direct link special relay ^{*1}	SB	DevSM, DevQSB, DevLSB(0)	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ Page 246 CCPU_ReadLinkDevice/CCPU_WriteLinkDevice
Own station direct link special register ^{*2}	SW	DevSD, DevQSW, DevLSW(0)	
Buffer memory	—	—	The area (device) corresponding to the device type deleted from R12CCPU-V can be accessed with the following method. ☞ The access target is a network module on the own station.(module access device)

*1 The own station direct link special relay (SB) has three device type definitions; DevSM, DevQSB, and DevLSB(0). Any of them can be specified for the same operation.

*2 The own station direct link special register (SW) has three device type definitions; DevSD, DevQSW, and DevLSW(0). Any of them can be specified for the same operation.

Alternative method

■Refreshing device

Access target	Alternative method
The access target is a network module on the own station.	Configure the refresh setting so that a device of C Controller module, M, B, D, W, or ZR is refreshed by a link device of a network module. Use the MELSEC data link functions to access a device of C Controller module, M, B, D, W or ZR.
The access target is a network module on the other station.	Configure the refresh setting for CPU module on the other station so that a device of C Controller module on the other station is refreshed by a link device of a network module. Specify the other station to the network number and station number in the MELSEC data link functions to access a device of CPU module on the other station.

■Module access device

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

*1 For details on the buffer memory address to which link devices are assigned, refer to the manual for the network module to be accessed.

■CCPU_ReadLinkDevice/CCPU_WriteLinkDevice

- Access the own station link device of a network module by using the CCPU_ReadLinkDevice/CCPU_WriteLinkDevice function. For details, refer to the relevant functions.

📖 Page 118 CCPU_ReadLinkDevice, Page 150 CCPU_WriteLinkDevice

Compilation of replaced project

Compile the replaced project in CW Workbench.

Appendix 5 Correspondence Table to Q06CCPU-V Functions

○: Conventional functions can be used. ×: Conventional functions cannot be used.

—: Replacement of functions is not required. Not available: No functions are available to be replaced.

Bus interface functions

Function name (Q06CCPU-V)	Availability in R12CCPU-V	Function name (replaced)
QBF_Close	×	Not available
QBF_Control	×	CCPU_Control
QBF_ControlEx	×	CCPU_Control
QBF_ControlLED	×	CCPU_SetLEDStatus
QBF_ControlProgram	×	Not available
QBF_EntryTimerEvent	×	CCPU_EntryTimerEvent
QBF_EntryWDTInt	×	CCPU_EntryWDTInt
QBF_FromBuf	×	CCPU_FromBuf, CCPU_FromBufHG
QBF_GetTime	×	CCPU_GetRTC
QBF_GINT	×	CCPU_DedicatedDInst
QBF_MotionCHGA	×	CCPU_DedicatedDInst
QBF_MotionCHGT	×	CCPU_DedicatedDInst
QBF_MotionCHGV	×	CCPU_DedicatedDInst
QBF_MotionDDRD	×	CCPU_DedicatedDInst
QBF_MotionDDWR	×	CCPU_DedicatedDInst
QBF_MotionSFCS	×	CCPU_DedicatedDInst
QBF_MotionSVST	×	CCPU_DedicatedDInst
QBF_MountCfCard	×	CCPU_DedicatedDInst
QBF_Open	×	Not available
QBF_ReadLinkDevice	×	Not available
QBF_ReadSRAM	×	CCPU_ReadDevice ^{*1}
QBF_ReadStatusEx	×	CCPU_GetCpuStatus
QBF_RECV	×	CCPU_DedicatedGInst, CCPU_DedicatedJInst
QBF_RefreshLinkDevice	×	CCPU_RefreshLinkDevice
QBF_RegistEventLog	×	CCPU_RegistEventLog, CCPU_RegistEventLog_ISR
QBF_Reset	×	CCPU_Reset
QBF_ResetWDT	×	CCPU_ResetWDT
QBF_SEND	×	CCPU_DedicatedGInst, CCPU_DedicatedJInst
QBF_SetTime	×	CCPU_SetRTC
QBF_ShutdownRom	×	Not available
QBF_StartWDT	×	CCPU_StartWDT
QBF_StopWDT	×	CCPU_StopWDT
QBF_ToBuf	×	CCPU_ToBuf, CCPU_ToBufHG
QBF_UnitInfo	×	CCPU_GetUnitInfo
QBF_UnmountCfCard	×	CCPU_UnmountMemoryCard

*1 Use ZR device as a substitute.

MELSEC data link functions

Function name (Q06CCPU-V)	Availability in R12CCPU-V	Function name (replaced)
mdClose	○	—
mdControl	○	—
mdDevRstEx	○	—
mdDevSetEx	○	—
mdInit	○	—
mdOpen	○	—
mdRandREx	○	—
mdRandWEx	○	—
mdReceiveEx (Device batch read function)	○	—
mdReceiveEx (Message receive function)	○	—
mdSendEx (Device batch write function)	○	—
mdSendEx (Message send function)	○	—
mdTypeRead	○	—
mdDevRst	×	mdDevRstEx
mdDevSet	×	mdDevSetEx
mdRandR	×	mdRandREx
mdRandW	×	mdRandWEx
mdReceive (Device batch read function)	×	mdReceiveEx (Device batch read function)
mdReceive (Message receive function)	×	mdReceiveEx (Message receive function)
mdSend (Device batch write function)	×	mdSendEx (Device batch write function)
mdSend (Message send function)	×	mdSendEx (Message send function)

INDEX

B

Bus interface communication 14,17

C

C Controller module dedicated function 8,10
CC-Link communication 14,41
CC-Link IE Controller Network communication . 14,21
CC-Link IE Field Network communication 14,26
CC-Link IE TSN communication 14,31
Channel 44
CW Workbench 8
CW-Sim 8

D

Dedicated function library. 9,10
Device type 13,49
Dummy access 14,15

H

Header file 10,13,49,86,90

I

ISR 63

M

MELSEC data link function. 8,10
MELSECNET/H network communication 14,36
Motion module dedicated class. 10

T

Task 15

U

User watchdog timer 11

V

VxWorks 8
VxWorks standard API function 10

FUNCTION INDEX

C

CCPU_ChangeCCIEFBCycPrm	69
CCPU_ChangeFileSecurity	70
CCPU_ClearError	71
CCPU_Control	72
CCPU_DedicatedDInst	73
CCPU_DedicatedGInst	75
CCPU_DedicatedJInst	77
CCPU_DedicatedMInst	79
CCPU_DisableInt	82
CCPU_DisableInt_ISR	162
CCPU_EnableInt	83
CCPU_EnableInt_ISR	163
CCPU_EndCCIEFBDataAssurance	84
CCPU_EntryCCIEFBRefEndFunc	85
CCPU_EntryInt	86
CCPU_EntryTimerEvent	88
CCPU_EntryWDTInt	90
CCPU_FromBuf	91
CCPU_FromBufHG	92
CCPU_FromBufHG_ISR	165
CCPU_FromBuf_ISR	164
CCPU_GetCCIEFBDiagnosticInfo	93
CCPU_GetConstantProcessStatus	95
CCPU_GetCounterMicros	96
CCPU_GetCounterMicros_ISR	166
CCPU_GetCounterMillis	97
CCPU_GetCounterMillis_ISR	167
CCPU_GetCpuStatus	98
CCPU_GetDotMatrixLED	100
CCPU_GetDotMatrixLED_ISR	168
CCPU_GetErrInfo	102
CCPU_GetFileSecurity	103
CCPU_GetIDInfo	104
CCPU_GetLEDStatus	105
CCPU_GetOpSelectMode	107
CCPU_GetPowerStatus	108
CCPU_GetRTC	109
CCPU_GetSerialNo	110
CCPU_GetSwitchStatus	111
CCPU_GetUnitInfo	112
CCPU_LockFWUpdate	115
CCPU_MountMemoryCard	116
CCPU_ReadDevice	117
CCPU_ReadDevice_ISR	170
CCPU_ReadLinkDevice	118
CCPU_ReadMCUnitLabel	119
CCPU_ReadMCUnitLabelBit	121
CCPU_RegistEventLog	122
CCPU_RegistEventLog_ISR	171
CCPU_Reset	123
CCPU_ResetDevice	124
CCPU_ResetDevice_ISR	172
CCPU_ResetWDT	125
CCPU_RestoreDefaultCCIEFBCycPrm	126
CCPU_SetDevice	127
CCPU_SetDevice_ISR	173
CCPU_SetDotMatrixLED	128
CCPU_SetDotMatrixLED_ISR	174
CCPU_SetLEDStatus	130
CCPU_SetLEDStatus_ISR	176

CCPU_SetOpSelectMode	131
CCPU_SetRTC	132
CCPU_ShutdownRom	133
CCPU_StartCCIEFBDataAssurance	134
CCPU_StartWDT	135
CCPU_StopWDT	136
CCPU_SysClkRateGet	137
CCPU_SysClkRateSet	138
CCPU_ToBuf	139
CCPU_ToBufHG	140
CCPU_ToBufHG_ISR	179
CCPU_ToBuf_ISR	177
CCPU_UnlockFWUpdate	141
CCPU_UnmountMemoryCard	142
CCPU_WaitEvent	143
CCPU_WaitSwitchEvent	145
CCPU_WaitTimerEvent	146
CCPU_WaitUnitEvent	147
CCPU_WriteDevice	149
CCPU_WriteDevice_ISR	180
CCPU_WriteLinkDevice	150
CCPU_WriteMCUnitLabel	151
CCPU_WriteMCUnitLabelBit	155
CCPU_X_In_BitEx	156
CCPU_X_In_WordEx	157
CCPU_X_In_Word_ISR	181
CCPU_Y_In_BitEx	158
CCPU_Y_In_WordEx	159
CCPU_Y_In_Word_ISR	183
CCPU_Y_Out_BitEx	160
CCPU_Y_Out_WordEx	161
CCPU_Y_Out_Word_ISR	185

M

MCFB::RefreshLabels	215
MCFBEnable::SetEnable	218
MCFBExecute::SetExecute	216
MCv_Jog::SetEnableJog	220
mdClose	187
mdControl	188
mdDevRstEx	189
mdDevSetEx	190
mdGetLabelInfo	191
mdInit	194
mdOpen	195
mdRandREx	196
mdRandRLabelEx	199
mdRandWEx	202
mdRandWLabelEx	204
mdReceiveEx	206,207
mdSendEx	209,210
mdTypeRead	212

REVISIONS

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

Revision date	*Manual number	Description
February 2015	SH(NA)-081371ENG-A	First edition
April 2015	SH(NA)-081371ENG-B	Structural change of the manual
May 2015	SH(NA)-081371ENG-C	■Added or modified parts TERMS, Section 3.1
October 2015	SH(NA)-081371ENG-D	■Added or modified parts INTRODUCTION, TERMS, Section 1.2, Section 1.3, Section 2.2, Section 3.2, Section 4.3
September 2016	SH(NA)-081371ENG-E	■Added or modified parts TERMS, Section 1.3, Section 3.1, Section 3.2, Section 4.3
October 2017	SH(NA)-081371ENG-F	■Added or modified parts Section 2.1, Section 3.1, Section 4.2
December 2017	SH(NA)-081371ENG-G	■Added or modified parts Section 2.1, Section 3.1, Section 4.2
September 2018	SH(NA)-081371ENG-H	■Added or modified part Section 4.2
March 2019	SH(NA)-081371ENG-I	■Added or modified part Section 3.1
May 2021	SH(NA)-081371ENG-J	■Added or modified parts TERMS, Chapter 1, Section 1.1, Section 1.2, Section 1.3, Section 1.4, Section 2.1, Section 2.3, Chapter 3, Section 3.1, Section 3.3, Section 3.4, Section 4.1, Section 4.2, Section 4.3, Section 4.4, Appendix 2, Appendix 3, Appendix 4, Appendix 5
September 2022	SH(NA)-081371ENG-K	■Added or modified parts Section 3.1, Section 3.3
October 2023	SH(NA)-081371ENG-L	■Added or modified parts GENERIC TERMS AND ABBREVIATIONS, Section 1.3, Section 3.1, Section 3.3, Section 4.1, Section 4.3
June 2024	SH(NA)-081371ENG-M	■Added or modified parts Section 3.1, Section 4.1, Section 4.2, Section 4.3

Japanese manual number: SH-081370-M

This manual confers no industrial property rights or any rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

© 2015 MITSUBISHI ELECTRIC CORPORATION

WARRANTY

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place. Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

- (1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.
- (2) Even within the gratis warranty term, repairs shall be charged for in the following cases.
 1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
 2. Failure caused by unapproved modifications, etc., to the product by the user.
 3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
 4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
 5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
 6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
 7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

2. Onerous repair term after discontinuation of production

- (1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued. Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.
- (2) Product supply (including repair parts) is not available after production is discontinued.

3. Overseas service

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

4. Exclusion of loss in opportunity and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to:

- (1) Damages caused by any cause found not to be the responsibility of Mitsubishi.
- (2) Loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products.
- (3) Special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products.
- (4) Replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

5. Changes in product specifications

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

INFORMATION AND SERVICES

For further information and services, please contact your local Mitsubishi Electric sales office or representative.
Visit our website to find our locations worldwide.

MITSUBISHI ELECTRIC Factory Automation Global Website

Locations Worldwide

www.MitsubishiElectric.com/fa/about-us/overseas/

TRADEMARKS

Microsoft and Windows are trademarks of the Microsoft group of companies.

VxWorks, and Wind River are either registered trademarks or trademarks of Wind River Systems, Inc.

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)-081371ENG-M(2406)

MODEL: RCCPU-P-E

mitsubishi electric corporation

HEAD OFFICE: TOKYO BLDG., 2-7-3, MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS: 1-14, YADA-MINAMI 5-CHOME, HIGASHI-KU, NAGOYA 461-8670, JAPAN

When exported from Japan, this manual does not require application to the
Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.