MITSUBISHI ELECTRIC

Programmable Controller

MELSEC iQ-R series

MELSEC iQ-R C Intelligent Function 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.

In this manual, the safety precautions are classified into two levels: "⚠ WARNING" and "⚠ CAUTION".

⚠ **WARNING**	Indicates that incorrect handling may cause hazardous conditions, resulting in death or severe injury.
⚠ **CAUTION**	Indicates that incorrect handling may cause hazardous conditions, resulting in minor or moderate injury or property damage.

Under some circumstances, failure to observe the precautions given under "⚠ CAUTION" may lead to serious consequences.

Observe the precautions of both levels because they are important for personal and system safety.

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) Mitsubishi 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 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'S USER, 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 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 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 representative in your region.

(3) Mitsubishi 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 intelligent function 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 intelligent function 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 MELSEC iQ-R series programmable controllers.
This manual describes the functions required for programming of the relevant products listed below.
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.

Relevant products

RD55UP06-V, RD55UP12-V

CONTENTS

RELEVANT MANUALS

Manual name [manual number]	Description	Available form
MELSEC iQ-R C Intelligent Function Module Programming Manual [SH-081568ENG] (this manual)	Programming specifications and dedicated function libraries of a C intelligent function module	e-Manual PDF
MELSEC iQ-R C Intelligent Function Module User's Manual (Startup) [SH-081566ENG]	Specifications, procedure before operation, wiring, and operation examples of a C intelligent function module	Print book e-Manual PDF
MELSEC iQ-R C Intelligent Function Module User's Manual (Application) [SH-081567ENG]	Functions, input/output signals, buffer memory, parameter setting, and troubleshooting of a C intelligent function module	Print book 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

Point

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

e-Manual has the following features:

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

TERMS

Unless otherwise specified, this manual uses the following terms.

Term	Description
C intelligent function module	A generic term for MELSEC iQ-R series C intelligent function module.
C intelligent function module dedicated function	A dedicated function library offered by the C intelligent function module. It is used to control the C intelligent function module.
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).
CW-Sim Standalone	An abbreviation for VxWorks simulator that can operate the C Controller module and C intelligent function module programs on a personal computer to which CW Workbench not installed.
Dedicated function library	A generic term for C intelligent function module dedicated function and MELSEC iQ-R series data link function.
Engineering tool	Another term of the software package for the MELSEC programmable controllers. This manual explains the GX Works3.
GX Works3	A generic product name for SWnDND-GXW3. ('n' indicates version.)
MELSEC iQ-R series data link function	A data link function library offered by the C intelligent function module. It is used to access an own station or the modules on the network.
VxWorks	A product name for the real-time operating system manufactured by Wind River Systems, Inc.

1 COMMON ITEMS

A user program is created by using the VxWorks standard API functions*1 and dedicated function library offered by the C intelligent function module in accordance with the specification of VxWorks, the operating system of C intelligent function 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 offered by a C intelligent function module are as follows:

- C intelligent function module dedicated function
- MELSEC iQ-R series data link function

Point

For the execution procedure of user programs, refer to the following manual.
MELSEC iQ-R C Intelligent Function Module User's Manual (Startup)

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 intelligent function module dedicated function	CITLFunc.h
MELSEC iQ-R series data link function	MDRFunc.h

Point

A header file is stored in a C intelligent function module.
(MELSEC iQ-R C Intelligent Function Module User's Manual (Startup))

1.2 C Intelligent Function Module Dedicated Functions

C intelligent function module dedicated functions of the dedicated function libraries are used to control C intelligent function module.

These functions can be used for reading status of the module or accessing resources such as LED control.

Program processing

The following procedure shows the processing flow of the user program using C intelligent function module.

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

Considerations

The following shows the considerations when using the C intelligent function module dedicated function.

Considerations for user WDT (User watchdog timer)

■ A 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 CITL_StartWDT function.
- Lower the number of tasks with high CPU utilization or make them deactivated.
- Review the user program.

Reset the C intelligent function 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.

1.3 MELSEC iQ-R Series Data Link Functions

MELSEC iQ-R series 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 iQ-R series data link functions.
The communication functions supported by the MELSEC iQ-R series 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.
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 procedure shows the processing flow of the user program using MELSEC iQ-R series data link function.

When accessing with a device name

1. Start a task.
2. Open a communication line. (mdrOpen function)
3. Perform dummy access (such as device/model name reading) to an access target.
4. Access the target by using the MELSEC iQ-R series 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. (mdrClose function)
7. Complete the task.

When accessing with a label name

1. Start a task.
2. Open a communication line. (mdrOpen function)
3. Acquire device information (label assignment information) from a target CPU module. (mdrGetLabelInfo function)
4. Access the target CPU module by using the acquired device information (label assignment information). (mdrRandRLabel/mdrRandWLabel 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. (mdrClose function)
8. Complete the task.

Considerations

The following shows the considerations when using the MELSEC iQ-R series data link functions.

Considerations for programming

■ Open/close processing of a communication line (mdrOpen/mdrClose function)

Perform the open/close processing of communication line (the mdrOpen/mdrClose 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 mdrOpen function

At the first execution of the function after using the mdrOpen 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 intelligent function 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.

■ mdrGetLabelInfo function call

The mdrGetLabelInfo 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 mdrRandRLabel/mdrRandWLabel function , call the mdrGetLabelInfo function again.

■ taskDelete execution

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

■ Error by access concentration in a CPU module

When using the MELSEC iQ-R series data link function to access a CPU module from multiple modules or a built-in Ethernet port of the CPU module in the system where communication processing such as device access to a CPU module is performed frequently, an error may occur in communication processing on other modules due to the concentration of processing to a CPU module.
When verifying the operation at the system construction and if an error occurs in communication processing, take a following measure before running the system.

- When executing the MELSEC iQ-R series data link function in multiple tasks, do not execute the function at the same time by exclusion control, or execute the function in one task.
- Lengthen the execution interval of the MELSEC iQ-R series data link function to avoid errors in communication processing.

Accessible range

This section shows the accessible CPU module , device, and route by using the C intelligent function module.

Accessible CPU modules

■ MELSEC iQ-R series

Module		Model
RCPU	Programmable controller CPU	R04CPU, R08CPU, R16CPU, R32CPU, R120CPU
	Process CPU	R08PCPU, R16PCPU, R32PCPU, R120PCPU
	CC-Link IE built-in CPU	R04ENCPU, R08ENCPU, R16ENCPU, R32ENCPU, R120ENCPU
C Controller module		R12CCPU-V*1

*1 It can be accessed only when using an RD55UP06-V.

■ MELSEC-Q series

Module		Model
QCPU (Q mode)	Basic model QCPU	Q00JCPU, Q00CPU, Q01CPU
	High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
	Universal model QCPU	Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU, Q03UDVCPU, Q04UD(P)VCPU, Q06UD(P)VCPU, Q13UD(P)VCPU, Q26UD(P)VCPU
	Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
C Controller module		Q12DCCPU-V*1, Q24DHCCPU-V, Q24DHCCPU-VG, Q24DHCCPU-LS

*1 Only a serial number of which the first 5 digits are "12042" or later

■ MELSEC-L series

Module	Model
LCPU	L02CPU, L02CPU-P, L02SCPU, L06CPU, L26CPU, L26CPU-BT, L26CPU-PBT

Accessible routes

The access target CPU modules are as follows:

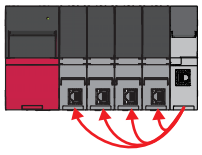
Access target	Accessible CPU modules
(1)	RCPU
(2)	MELSEC iQ-R series C Controller module*1
(3)	QCPU (Q mode)
(4)	LCPU
(5)	MELSEC-Q series C Controller module*1

*1 The module cannot be used as a relay station.

■ Accessing own station (control CPU, other multiple CPU)

Access the CPU module of the station on which C intelligent function module is mounted.

Own station



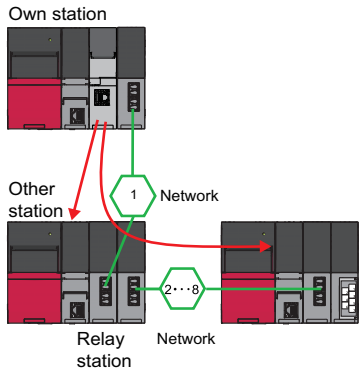
○: Accessible, ×: Not accessible, —: Not applicable

Communication route	Access target				
	(1)	(2)	(3)	(4)	(5)
Bus interface communication	○ (CPU No.1 to 4)	○ (CPU No.1 to 4)	—	—	—

■ The access via single network (specify a network number and station number.)

Access by specifying the network number and station number of the target station.

When the access target CPU module can be specified by the network number and station number (CPU number) from the station on which C intelligent function module is mounted in the status where the access target CPU module is connected to the network, the access to the CPU module that is mounted on the eighth network is available via a relay station of RCPU or QCPU (Q mode).



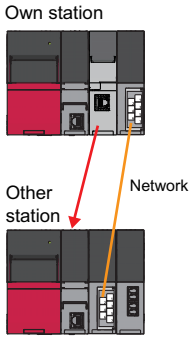
○: Accessible, ×: Not accessible, —: Not applicable

Communication route	Access target				
	(1)	(2)	(3)	(4)	(5)
CC-Link IE Controller Network	○ (CPU No.1 to 4)	○ (CPU No.1 to 4)	○ (CPU No.1 to 4)	—	○ (CPU No.1 to 4)
CC-Link IE Field Network				○	
MELSECNET/H			○ (CPU No.1 to 4)	—	○ (CPU No.1 to 4)

■ The access via single network (specify a start I/O number and a station number of the target station.)

Access by specifying a start I/O number of module to access the target station and a station number of the target station.

The following shows the route when the access target CPU module and C intelligent function module of the mounting side are directly connected.



○: Accessible, ×: Not accessible, —: Not applicable

Communication route	Access target				
	(1)	(2)	(3)	(4)	(5)
CC-Link	○ (CPU No.1 to 4)	○ (CPU No.1 to 4)	○ (CPU No.1 to 4)	○	○ (CPU No.1 to 4)

Accessible devices

The access target CPU modules are as follows:

Access target	Accessible CPU modules
(1)	RCPU
(2)	MELSEC iQ-R series C Controller module
(3)	QCPU (Q mode)
(4)	LCPU
(5)	MELSEC-Q series C Controller module

○: Accessible, ×: Not accessible

Device name (Device)[*1]		Access target				
		(1)	(2)	(3)	(4)	(5)
Function input (FX)		×	×	×	×	×
Function output (FY)		×	×	×	×	×
Function register (FD)		×	×	×	×	×
Special relay (SM)		○	○	○	○	○
Special register (SD)		○	○	○	○	○
Input relay (X)		○	○	○	○	○
Output relay (Y)		○	○	○	○	○
Internal relay (M)		○	○	○	○	○[*2]
Latch relay (L)		○	×	○	○	×
Annunciator (F)		○	×	○	○	×
Edge relay (V)		○	×	○	○	×
Link relay (B)		○	○	○	○	○[*3]
Data register (D)		○	○	○	○	○[*2]
Link register (W)		○	○	○	○	○[*3]
Extended internal relay (M)		×	×	○	○	×
Extended data register (D)[*4]		×	×	○	○	×
Extended link register (W)[*4]		×	×	○	○	×
Timer	Contact (TS)	○	×	○	○	×
	Coil (TC)	○	×	○	○	×
	Current value (T/TN) [*5]	○	×	○	○	×
Long timer	Contact (LTS)	○	×	×	×	×
	Coil (LTC)	○	×	×	×	×
	Current value (LT/LTN)[*5]	○	×	×	×	×
Counter	Contact (CS)	○	×	○	○	×
	Coil (CC)	○	×	○	○	×
	Current value (C/CN)[*5]	○	×	○	○	×
Long counter	Contact (LCS)	○	×	×	×	×
	Coil (LCC)	○	×	×	×	×
	Current value (LC/LCN)[*5]	○	×	×	×	×
Retentive timer	Contact (STS, SS[*6])	○	×	○	○	×
	Coil (STC, SC[*6])	○	×	○	○	×
	Current value (ST/STN, ST/SN[*6])[*5]	○	×	○	○	×
Long retentive timer	Contact (LSTS)	○	×	×	×	×
	Coil (LSTC)	○	×	×	×	×
	Current value (LST/LSTN)[*5]	○	×	×	×	×
Link special relay (SB)		○	×	○	○	×
Link special register (SW)		○	×	○	○	×
Step relay (S)		×	×	×	×	×
Direct input (DX)		×	×	×	×	×
Direct output (DY)		×	×	×	×	×
Accumulator (A)		×	×	×	×	×

Device name (Device)[*1]		Access target				
		(1)	(2)	(3)	(4)	(5)
Index register (Z)		○	×	○	○	×
Long index register (LZ)		○	×	×	×	×
File register	(R)	○	×	○[*7]	○	×
	(ZR)[*8]	○	○	○[*7]	○	×
	(ER□\R)[*9]	×	×	×	×	×
Link direct device[*10]	Link input (J□\X)	○	○	○	○	○
	Link output (J□\Y)	○	○	○	○	○
	Link relay (J□\B)	○	○	○	○	○
	Link special relay (J□\SB)	○	○	○	○	○
	Link register (J□\W)	○	○	○	○	○
	Link special register (J□\SW)	○	○	○	○	○
Refresh data register (RD)		○	×	×	×	×
Module access device	Module access device/Intelligent function module device (U□\G)[*11]	○	○	○	○	○
	Multiple CPU shared device (U3E□\G)[*12]	×	○	○	○	○
CPU buffer memory access device[*12]	CPU buffer memory access device (U3E□\G)	○	×	×	×	×
	CPU buffer memory access device (Fixed cycle communication area) (U3E□\HG)	○	×	×	×	×
Global label (GV) [*13] (No device assigned)		○	×	×	×	×

*1 The file registers for each local device and program in which the program name is specified cannot be accessed.
*2 For Q12DCCPU-V (Basic mode), select "Use device function" on a C Controller module.
*3 For Q12DCCPU-V, only Q12DCCPU-V (Extended mode) can be accessed.
*4 The extended data register (D) and extended link register (W) can be accessed by the following two methods.
(1): Access by directly specifying the device name of the extended data register (D) and extended link register (W)
(2): Access to the file register (ZR) area assigned to the extended data register (D) and extended link register (W)
*5 Either of the device names can be specified.
*6 This is the device name in QCPU (Q mode), LCPU, and MELSEC-Q series C Controller module.
*7 It is not accessible when using Q00JCPU or Q00UJCPU.
*8 When accessing out of the range of the file register (ZR) area, the value of -1(FFFFH) is sampled.
*9 "□": Specify the block number.
*10 "□": Specify the network number.
*11 "□": Specify the start I/O number÷ 10H.
*12 "□": Specify the CPU number (CPU No.1: 0, CPU No.2: 1, CPU No.3: 2, CPU No.4: 3)
*13 Only the mdrRandRLabel/mdrRandWLabel function can be used.

Argument specification

This section shows the argument specification of the MELSEC iQ-R series data link functions.

Channel

A channel implies a network and communication route to be used when communicating with a C intelligent function module.
A channel number is set for each module in a user program.
A channel to be used for MELSEC iQ-R series data link functions is as follows:

Channel number	Network	Communication route
12	Bus interface	Used for communication via bus.

CPU number, network number, start I/O number, station number

CPU numbers, Network numbers, start I/O numbers and station numbers to be specified to MELSEC iQ-R series data link functions are as follows:

Access route		CPU number	Network number	Start I/O number	Station number
Bus interface	Own station	• 0: Control CPU specification • 1 to 4: Multiple CPU specification	—*1	—*1	—*1
CC-Link IE Controller Network	Via single network		1 to 239		1 to 120, 0*2, 125*2
CC-Link IE Field Network					0 to 120
MELSECNET/H network					1 to 64, 0*2, 125*2
CC-Link			—*1	0000H to 00FEH	0 to 63

*1 No error will occur even if the value is set.
*2 A specified control station of the network, which is specified to the network number, is accessed. To access a station that is actually operating as the control station, specify the station number.

Device type

The following table shows the device types specified to the MELSEC iQ-R series data link functions.
Devices are defined in the header file (MDRFunc.h).

Point

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

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

Device name (Device)		Device type		
		Code		Device name
		Decimal	Hexadecimal	
Timer	Contact (TS)	7	7H	DevTT
	Coil (TC)	8	8H	DevTC
	Current value (T/TN)	11	BH	DevTN
Long timer	Contact (LTS)	41	29H	DevLTT
	Coil (LTC)	42	2AH	DevLTC
	Current value (LT/LTN)	43	2BH	DevLTN
Counter	Contact (CS)	9	9H	DevCT
	Coil (CC)	10	AH	DevCC
	Current value (C/CN)	12	CH	DevCN
Long counter	Contact (LCS)	44	2CH	DevLCT
	Coil (LCC)	45	2DH	DevLCC
	Current value (LC/LCN)	46	2EH	DevLCN
Retentive timer	Contact (STS, SS)	26	1AH	DevSTT
	Coil (STC, SC)	27	1BH	DevSTC
	Current value (ST/STN, ST/SN)	35	23H	DevSTN
Long retentive timer	Contact (LSTS)	47	2FH	DevLSTT
	Coil (LSTC)	48	30H	DevLSTC
	Current value (LST/LSTN)	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
Edge relay (V)		30	1EH	DevQV
Module refresh register (RD)		39	27H	DevRD
Global label (GV) *5	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_C
	For bit F	616	268H	DevGV_F

Device name (Device)		Device type		
		Code		Device name
		Decimal	Hexadecimal	
Link direct device[*3,*4] Argument value of device name (1 to 255): Network number	Link input (J□\X)	1001 to 1255	3E9H to 4E7H	DevLX(1) to DevLX(255)
	Link output (J□\Y)	2001 to 2255	7D1H to 8CFH	DevLY(1) to DevLY(255)
	Link relay (J□\B)	23001 to 23255	59D9H to 5AD7H	DevLB(1) to DevLB(255)
	Link register (J□\W)	24001 to 24255	5DC1H to 5EBFH	DevLW(1) to DevLW(255)
	Link special relay (J□\SB)	25001 to 25255	61A9H to 62A7H	DevLSB(1) to DevLSB(255)
	Link special register (J□\SW)	28001 to 28255	6D61H to 6E5FH	DevLSW(1) to DevLSW(255)
Intelligent function module device, module access device[*3] Argument value of device name (0 to 255): Start I/O number ÷ 16.		29000 to 29255	7148H to 7247H	DevSPG(0) to DevSPG(255)

*1 For Q12DCCPU-V, it is categorized as the device type for Q bus interface.

*2 The devices cannot be used for the mdrDevRst/mdrDevSet/mdrRandR/mdrRandW functions.

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

*4 "□": indicates a network number.

*5 Only the mdrRandRLabel/mdrRandWLabel function can be used.

1.4 Considerations on Interrupt Service Routine (ISR)

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

Point

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

2 FUNCTION LIST

This chapter shows the functions used for C intelligent function modules.

2.1 C Intelligent Function Module Dedicated Functions

The C intelligent function module dedicated functions are as listed below.

C intelligent function module dedicated functions

Function name	Description	Reference
CITL_ChangeFileSecurity	To change the file access restriction status of a C intelligent function module.	☞ Page 22 CITL_ChangeFileSecurity
CITL_ClearError	To clear errors of a C intelligent function module.	☞ Page 23 CITL_ClearError
CITL_DisableYInt	To disable the routine registered with the CITL_EntryYInt function.	☞ Page 24 CITL_DisableYInt
CITL_EnableYInt	To enable the routine registered with the CITL_EntryYInt function.	☞ Page 25 CITL_EnableYInt
CITL_EntryDedicatedInstFunc	To register a routine to be executed using the dedicated instruction (G(P).CEXECUTE).	☞ Page 26 CITL_EntryDedicatedInstFunc
CITL_EntryTimerEvent	To register a timer event.	☞ Page 27 CITL_EntryTimerEvent
CITL_EntryWDTInt	To register a routine to be called when a user WDT error interrupt occurs.	☞ Page 29 CITL_EntryWDTInt
CITL_EntryYInt	To register a routine to be called when an output signal (Y) interrupt occurs.	☞ Page 30 CITL_EntryYInt
CITL_FromBuf	To read data from the buffer memory of a C intelligent function module.	☞ Page 31 CITL_FromBuf
CITL_GetCollectData	To acquire data sampled in data sampling in each sequence scan.	☞ Page 32 CITL_GetCollectData
CITL_GetCounterMicros	To acquire a 1 μs counter value of a C intelligent function module.	☞ Page 34 CITL_GetCounterMicros
CITL_GetCounterMillis	To acquire a 1 ms counter value of a C intelligent function module.	☞ Page 35 CITL_GetCounterMillis
CITL_GetErrInfo	To acquire the error information of a C intelligent function module.	☞ Page 36 CITL_GetErrInfo
CITL_GetFileSecurity	To acquire the file access mode of a C intelligent function module.	☞ Page 37 CITL_GetFileSecurity
CITL_GetIDInfo	To acquire the individual identification information of a C intelligent function module.	☞ Page 38 CITL_GetIDInfo
CITL_GetLEDStatus	To acquire the LED status of a C intelligent function module.	☞ Page 39 CITL_GetLEDStatus
CITL_GetSerialNo	To acquire the serial number of a C intelligent function module.	☞ Page 40 CITL_GetSerialNo
CITL_GetSwitchStatus	To acquire the switch status of a C intelligent function module.	☞ Page 41 CITL_GetSwitchStatus
CITL_GetTime	To acquire the clock data (local time) of a C intelligent function module.	☞ Page 42 CITL_GetTime
CITL_GetUnitStatus	To acquire the operating status of a C intelligent function module.	☞ Page 43 CITL_GetUnitStatus
CITL_MountMemoryCard	To mount the SD memory card inserted to a C intelligent function module.	☞ Page 44 CITL_MountMemoryCard
CITL_RegistEventLog	To register an event log in the event history of a control CPU module.	☞ Page 45 CITL_RegistEventLog
CITL_ResetWDT	To reset the user WDT of a C intelligent function module.	☞ Page 46 CITL_ResetWDT
CITL_SetCollectData	To set data to be sampled in data sampling in each sequence scan.	☞ Page 47 CITL_SetCollectData
CITL_SetLEDStatus	To set the LED status of a C intelligent function module.	☞ Page 49 CITL_SetLEDStatus
CITL_SetSyncTimeStatus	To set the operating status of time synchronization of a C intelligent function module.	☞ Page 50 CITL_SetSyncTimeStatus
CITL_ShutdownRom	To shut down the standard ROM of a C intelligent function module.	☞ Page 51 CITL_ShutdownRom
CITL_StartCollectData	To start data sampling in each sequence scan.	☞ Page 52 CITL_StartCollectData
CITL_StartWDT	To set and start the user WDT of a C intelligent function module.	☞ Page 53 CITL_StartWDT
CITL_StopCollectData	To stop data sampling in each sequence scan.	☞ Page 54 CITL_StopCollectData
CITL_StopWDT	To stop the user WDT of a C intelligent function module.	☞ Page 55 CITL_StopWDT
CITL_SyncTime	To synchronize the time of a C intelligent function module with that of a control CPU module.	☞ Page 56 CITL_SyncTime
CITL_SysClkRateGet	To read the system clock rate specified with the CITL_SysClkRateSet function from the flash ROM.	☞ Page 57 CITL_SysClkRateGet
CITL_SysClkRateSet	To store the specified system clock rate in the flash ROM.	☞ Page 58 CITL_SysClkRateSet

Function name	Description	Reference
CITL_ToBuf	To write data to the buffer memory of a C intelligent function module.	☞ Page 59 CITL_ToBuf
CITL_UnmountMemoryCard	To unmount the SD memory card inserted to a C intelligent function module.	☞ Page 60 CITL_UnmountMemoryCard
CITL_WaitCollectDataRecvEvent	To wait for data to be sampled in data sampling in each sequence scan.	☞ Page 61 CITL_WaitCollectDataRecvEvent
CITL_WaitSwitchEvent	To wait for a switch interrupt event of C intelligent function module to occur.	☞ Page 62 CITL_WaitSwitchEvent
CITL_WaitTimerEvent	To wait for a timer event to occur.	☞ Page 63 CITL_WaitTimerEvent
CITL_WaitYEvent	To wait for the output signal (Y) interrupt event notification.	☞ Page 64 CITL_WaitYEvent
CITL_X_In_Bit	To read an input signal (X) in bit (1-point) units.	☞ Page 65 CITL_X_In_Bit
CITL_X_In_Word	To read an input signal (X) in word (16-point) units.	☞ Page 66 CITL_X_In_Word
CITL_X_Out_Bit	To write to an input signal (X) in bit (1-point) units.	☞ Page 67 CITL_X_Out_Bit
CITL_X_Out_Word	To write to an input signal (X) in word (16-point) units.	☞ Page 68 CITL_X_Out_Word
CITL_Y_In_Bit	To read an output signal (Y) in bit (1-point) units.	☞ Page 69 CITL_Y_In_Bit
CITL_Y_In_Word	To read an output signal (Y) in word (16-point) units.	☞ Page 70 CITL_Y_In_Word

C intelligent function module dedicated functions for ISR

Function name	Description	Reference
CITL_DisableYInt_ISR	To disable the routine registered with the CITL_EntryYInt function.	☞ Page 71 CITL_DisableYInt_ISR
CITL_EnableYInt_ISR	To enable the routine registered with the CITL_EntryYInt function.	☞ Page 72 CITL_EnableYInt_ISR
CITL_FromBuf_ISR	To read data from the buffer memory of a C intelligent function module.	☞ Page 73 CITL_FromBuf_ISR
CITL_GetCounterMicros_ISR	To acquire a 1 μs counter value of a C intelligent function module.	☞ Page 74 CITL_GetCounterMicros_ISR
CITL_GetCounterMillis_ISR	To acquire a 1 ms counter value of a C intelligent function module.	☞ Page 75 CITL_GetCounterMillis_ISR
CITL_RegistEventLog_ISR	To register an event log in the event history of a control CPU module.	☞ Page 76 CITL_RegistEventLog_ISR
CITL_SetLEDStatus_ISR	To set the LED status of a C intelligent function module.	☞ Page 77 CITL_SetLEDStatus_ISR
CITL_ToBuf_ISR	To write data to the buffer memory of a C intelligent function module.	☞ Page 78 CITL_ToBuf_ISR
CITL_X_In_Word_ISR	To read an input signal (X) in word (16-point) units.	☞ Page 79 CITL_X_In_Word_ISR
CITL_X_Out_Word_ISR	To write to an input signal (X) in word (16-point) units.	☞ Page 80 CITL_X_Out_Word_ISR
CITL_Y_In_Word_ISR	To read an output signal (Y) in word (16-point) units.	☞ Page 81 CITL_Y_In_Word_ISR

2.2 MELSEC iQ-R Series Data Link Functions

The MELSEC iQ-R series data link functions are as listed below.

Function name	Description	Reference
mdrClose	To close a communication line (channel).	☞ Page 82 mdrClose
mdrControl	To perform remote operations (RUN/STOP/PAUSE) for a CPU module.	☞ Page 83 mdrControl
mdrDevRst	To reset bit devices.	☞ Page 84 mdrDevRst
mdrDevSet	To set bit devices.	☞ Page 85 mdrDevSet
mdrGetLabelInfo	To acquire device information corresponding to label names.	☞ Page 86 mdrGetLabelInfo
mdrInit	To initialize the communication route information.	☞ Page 89 mdrInit
mdrOpen	To open a communication line (channel).	☞ Page 90 mdrOpen
mdrRandR	To read devices randomly.	☞ Page 91 mdrRandR
mdrRandRLabel	To read devices corresponding to labels randomly.	☞ Page 94 mdrRandRLabel
mdrRandW	To write devices randomly.	☞ Page 98 mdrRandW
mdrRandWLabel	To write devices corresponding to labels randomly.	☞ Page 100 mdrRandWLabel
mdrReceive	To read devices in a batch.	☞ Page 103 mdrReceive
mdrSend	To write devices in a batch.	☞ Page 104 mdrSend
mdrTypeRead	To read the model code of CPU module.	☞ Page 105 mdrTypeRead

3 DETAILS OF FUNCTION

This chapter shows the details of the C intelligent function module dedicated function and the MELSEC iQ-R series data link function.

3.1 C Intelligent Function Module Dedicated Functions

This section shows the details of the C intelligent function module dedicated function.

C intelligent function module dedicated functions

CITL_ChangeFileSecurity

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

■ Format

short CITL_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.) • 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 37 CITL_GetFileSecurity

CITL_ClearError

This function clears errors of a C intelligent function module.

■ Format

short CITL_ClearError(long* plErrorInfo)

■ Argument

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

■ Description

- This function clears errors occurred in a C intelligent function module.
- When no error occurs, this function ends normally.
- When a module major or moderate error occurs, the error cannot be cleared. (This function ends normally.)

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 36 CITL_GetErrInfo

CITL_DisableYInt

This function disables the routine registered with the CITL_EntryYInt function.

■ Format

short CITL_DisableYInt (short sYNo)

■ Argument

Argument	Name	Description	IN/OUT
sYNo	Output signal (Y) number	Specify the output signal (Y) number. (If -1 is specified, disable all the registered routines.)	IN

■ Description

- This function disables the routine registered with the CITL_EntryYInt function. (The routine is not executed when an output signal (Y) interrupt occurs.)
- Specify the output signal (Y) number (sYNo) specified in the CITL_EntryYInt function in the output signal (Y) number (sYNo).
- The output signal (Y) interrupt event notification wait function (CITL_WaitYEvent function)and the function executing interrupt routine when output signal (Y) interrupts (Defined by CITL_EntryYInt/CITL_EnableYInt/CITL_DisableYInt function) operate independently. These functions operate independently even if interrupt occurs by the same output signals (Y).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 25 CITL_EnableYInt
☞ Page 30 CITL_EntryYInt

CITL_EnableYInt

This function enables the routine registered with the CITL_EntryYInt function.

■ Format

short CITL_EnableYInt (short sYNo)

■ Argument

Argument	Name	Description	IN/OUT
sYNo	Output signal (Y) number	Specify the output signal (Y) number. (If -1 is specified, enable all the registered routines.)	IN

■ Description

- This function enables the routine registered with the CITL_EntryYInt function. (The routine is executed when an output signal (Y) interrupt occurs.)
- Specify the output signal (Y) number (sYNo) specified in the CITL_EntryYInt function in the output signal (Y) number (sYNo).
- The output signal (Y) interrupt event notification wait function (CITL_WaitYEvent function)and the function executing interrupt routine when output signal (Y) interrupts (Defined by CITL_EntryYInt/CITL_EnableYInt/CITL_DisableYInt function) operate independently. These functions operate independently even if interrupt occurs by the same output signals (Y).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 24 CITL_DisableYInt
☞ Page 30 CITL_EntryYInt

CITL_EntryDedicatedInstFunc

This function registers a routine to be executed with the dedicated instruction (G(P).CEXECUTE).

■ Format

short CITL_EntryDedicatedInstFunc (CITL_CEXECUTEFUNCPTR pCEXECUTEFuncPtr)

■ Argument

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

The data type of registered routine (pCEXECUTEFuncPtr) is defined by the header file (CITLFunc.h) as follows:

- void (*CITL_CEXECUTEFUNCPTR) (unsigned short* pusReqData, unsigned short* pusReqSize, unsigned short* pusAnsData, unsigned short* pusAnsSize)

Argument	Name	Description	IN/OUT
pusReqData	Request data	Receives the requested data specified by the dedicated instruction.	IN
pusReqSize	Request data size	Receives the size of the requested data specified by the dedicated instruction.	IN
pusAnsData	Response data	Returns the response data to the dedicated instruction.	OUT
pusAnsSize	Response data size	Returns the size of the response data to the dedicated instruction.	OUT

■ Description

- This function registers a routine to be executed in a C intelligent function module when executing the dedicated instruction (G(P).CEXECUTE).
- The registered routine is operated on a task with the following settings.
  Task priority: 100
  Stack size: 40000 byte
  Task option: VX_FP_TASK
- When NULL is specified to the registered routine (pCEXECUTEFuncPtr), the routine is deregistered.
- When CITL_EntryDedicatedInstFunc function is executed several times, the last registered routine will be in effect.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

CITL_EntryTimerEvent

This function registers a timer event.

■ Format

short CITL_EntryTimerEvent (long* plEvent)

■ Argument

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

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

Specification position	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 (Clear: 0, Cycle: 1 to 60,000 [ms])	
plEvent[3]	Synchronization type of the first timer event (Batch synchronization: 0, Individual synchronization: 1)	
plEvent[4]	Second timer event number (1 to 16)	Second timer event setting
plEvent[5]	Cycle of the second timer event (Clear: 0, Cycle: 1 to 60,000 [ms])	
plEvent[6]	Synchronization type of the second timer event (Batch synchronization: 0, Individual synchronization: 1)	
plEvent[7]	Third timer event number (1 to 16)	Third timer event setting
plEvent[8]	Cycle of the third timer event (Clear: 0, Cycle: 1 to 60,000 [ms])	
plEvent[9]	Synchronization type of the third timer event (Batch synchronization: 0, Individual synchronization: 1)	
⋮	⋮	⋮
plEvent[46]	16th timer event number (1 to 16)	16th timer event setting
plEvent[47]	Cycle of the 16th timer event (Clear: 0, Cycle: 1 to 60,000 [ms])	
plEvent[48]	Synchronization type of the 16th timer event (Batch synchronization: 0, Individual synchronization: 1)	

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

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

■ Description

- The CITL_EntryTimerEvent function sets the cycle and synchronization type for the timer event registration.
- When '0' is specified to 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 event. For the synchronization type, refer to the description of the CITL_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 timer event registered by this function waits for the event with the CITL_WaitTimerEvent function.
- All the timer events are cleared at the initial status.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 63 CITL_WaitTimerEvent

CITL_EntryWDTInt

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

■ Format

short CITL_EntryWDTInt (short sType, CITL_FUNCPTR pFuncPtr)

■ Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (When 'Reserved' is specified, an error is returned.) • 0: User WDT • Others: Reserved	IN
pFuncPtr	Registered routine	Specify the 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 (CITLFunc.h).

■ Description

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

Precautions

- When the operating system is in an interrupt disabled state, the registered routine is not executed.
- 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 intelligent function 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 46 CITL_ResetWDT
☞ Page 53 CITL_StartWDT
☞ Page 55 CITL_StopWDT

CITL_EntryYInt

This function registers a routine to be called when an output signal (Y) interrupt occurs.

■ Format

short CITL_EntryYInt (short sYNo, CITL_FUNCPTR pFuncPtr)

■ Argument

Argument	Name	Description	IN/OUT
sYNo	Output signal (Y) number	Specify the output signal (Y) number.	IN
pFuncPtr	Registered routine	Specify the 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 (CITLFunc.h).
- Specify the output signal (Y) number in the following format.
  Output signal (Y) number: 0x10 to 0x1F

■ Description

- This function registers a routine specified to the registered routine (pFuncPtr) in the interrupt specified with the output signal (Y) number (sYNo).
- When NULL is specified to the registered routine (pFuncPtr), the routine is deregistered.
- Use the CITL_EnableYInt function to enable the routine registered with the CITL_EntryYInt function.
  Otherwise, the routine will not be called.

Precautions

- When the operating system is in an interrupt disabled state, the registered routine is not executed.
- 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 intelligent function 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 CITL_EntryYInt function is executed more than once with the same output signal (Y) number (sYNo) specified, the routine, which was specified by the registered routine (pFuncPtr) at last, will be registered. (Multiple routines cannot be registered.)
- The routine is disabled after the registration is done by the CITL_EntryYInt function.
- When the routine registered by the CITL_EntryYInt function is running, calling the routine registered in WDT error interrupt is delayed.
- The output signal (Y) interrupt event notification wait function (CITL_WaitYEvent function)and the function executing interrupt routine when output signal (Y) interrupts (Defined by CITL_EntryYInt/CITL_EnableYInt/CITL_DisableYInt function) operate independently. These functions operate independently even if interrupt occurs by the same output signals (Y).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 24 CITL_DisableYInt
☞ Page 25 CITL_EnableYInt

CITL_FromBuf

This function reads data from the buffer memory of a C intelligent function module.

■ Format

short CITL_FromBuf (unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

■ Argument

Argument	Name	Description	IN/OUT
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 buffer memory of a C intelligent function module, and stores it in the data storage destination (pusDataBuf). Data is read by specifying an offset address from the start of the buffer memory of a C intelligent function module.

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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 59 CITL_ToBuf

CITL_GetCollectData

This function acquires data sampled in data sampling in each sequence scan.

■ Format

short CITL_GetCollectData (short* psBuf, unsigned long ulBufSize, unsigned long* pulRecordNum)

■ Argument

Argument	Name	Description	IN/OUT
psBuf	Acquired data storage destination	Specify the storage destination of acquired data.	OUT
ulBufSize	Acquired data storage destination size	Specify the size of the area reserved in the acquired data storage destination in byte units.	IN
pulRecordNum	Storage destination for the number of records	Specify the storage destination for the number of records.	OUT

Specify the acquired data storage destination size (ulBufSize) as follows:

- 1 record size × sizeof(short) × Number of records to be acquired

■ Description

- Acquired data is stored in the acquired data storage destination (psBuf) in word units in the order specified with the CITL_SetCollectData function. To 1 point of sampled points, a bit device and a word device are stored per 1 point, a double-word device is stored per 1/2 points (two words are used) for one word in the acquired data storage destination (psBuf).
- Sampled records are stored all or from the oldest one for the size specified to the acquired data storage destination size (ulBufSize) in the acquired data storage destination (psBuf). When a numerical value which cannot be divided by the data size required for 1 record is specified to the acquired data storage destination size (ulBufSize), data that can be stored in the acquired data storage destination size (ulBufSize) is stored in the acquired data storage destination (psBuf).
- The number of records of data stored in the acquired data storage destination (psBuf) is stored in the storage destination for the number of records (pulRecordNum).
- The following table shows the data to be stored in the acquired data storage destination (psBuf). (When outputting an index of header information, a date and time, and the data missing status)

Storage position*1	Description	Record*2
psBuf[0]	Index	1 record
psBuf[1]		
psBuf[2]	Date and time (Number of elapsed seconds from 00:00:00 on January 1st 1970)	
psBuf[3]		
psBuf[4]	Date and time (Nanoseconds less than a second from the number of elapsed seconds (unit: 100 μs))	
psBuf[5]		
psBuf[6]	Data missing status (0: Data is not missing, 1: Data is missing)	
psBuf[7]	Sampled data (start)	
⋮	⋮	
psBuf[7+(n-1)]	Sampled data (end)	
⋮	⋮	⋮
psBuf[(k+n)(m-1)]	⋮	m record
psBuf[(k+n)(m-1)+1]	⋮	
psBuf[(k+n)(m-1)+2]	⋮	
⋮	⋮	
psBuf[(k+n)(m-1)+7]	⋮	
⋮	⋮	
psBuf[(k+n)(m-1)+7+(n-1)]	Sampled data (end)	

*1 Information and data when using a double word (2 words) are stored from the lower word.

*2 A record refers to data to be sampled in one sequence scan. (Data set with the CITL_SetCollectData function)

Precautions

If CPU parameters of a control CPU module are changed during data sampling in each sequence scan, data sampling is stopped. Since the assignment of devices may be changed due to the change of CPU parameters, set target data with the C intelligent function module dedicated function (CITL_SetCollectData) again when restarting data sampling.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 47 CITL_SetCollectData
☞ Page 52 CITL_StartCollectData
☞ Page 54 CITL_StopCollectData
☞ Page 61 CITL_WaitCollectDataRecvEvent

CITL_GetCounterMicros

This function acquires a 1 μs counter value of a C intelligent function module.

■ Format

short CITL_GetCounterMicros(unsigned long* pulMicros)

■ Argument

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

■ Description

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

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 35 CITL_GetCounterMillis

CITL_GetCounterMillis

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

■ Format

short CITL_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 a C intelligent function module, and stores it 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 34 CITL_GetCounterMicros

CITL_GetErrInfo

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

■ Format

short CITL_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 intelligent function 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.
  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.

Storage position	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

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 23 CITL_ClearError

CITL_GetFileSecurity

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

■ Format

short CITL_GetFileSecurity(short* psMode)

■ Argument

Argument	Name	Description	IN/OUT
psMode	File access mode	Stores the file access mode. • 0: Access restriction clear mode • 1: Access restriction mode	OUT

■ Description

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

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 22 CITL_ChangeFileSecurity

CITL_GetIDInfo

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

■ Format

short CITL_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 intelligent function 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 individual identification information is stored in the individual identification information storage destination (pucGetData) as shown below.

Storage position	Description
pucGetData[0]	Individual identification information (CH1)
pucGetData[1]	
pucGetData[2]	
pucGetData[3]	
pucGetData[4]	
pucGetData[5]	
pucGetData[6]	Individual identification information (CH2)*1
pucGetData[7]	
pucGetData[8]	
pucGetData[9]	
pucGetData[10]	
pucGetData[11]	

*1 RD55UP12-V only

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 40 CITL_GetSerialNo

CITL_GetLEDStatus

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

■ Format

short CITL_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.) • 0: RUN LED • 1: ERR LED • 2: CARD RDY LED • 3: USER LED • 4 to 6: Reserved • -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 intelligent function 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 size (ulBufSize).
- The LED status to be stored in the LED status storage destination (pusLedInfo) is as follows.

Stored information	LED status
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].)

Storage position	Description
pusLedInfo[0]	RUN LED status
pusLedInfo[1]	ERR LED status
pusLedInfo[2]	CARD RDY LED status
pusLedInfo[3]	USER LED status
pusLedInfo[4]	Reserved
pusLedInfo[5]	
pusLedInfo[6]	

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 36 CITL_GetErrInfo

CITL_GetSerialNo

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

■ Format

short CITL_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. (When '0' is specified, this function ends normally without processing.)	IN

■ Description

- This function acquires the serial number (16-digits) of a C intelligent function 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

CITL_GetSwitchStatus

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

■ Format

short CITL_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 intelligent function 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.

Storage position		Description	Status
plStatusBuf[0]	bit31 to 6	Reserved	—
	bit5 to 3	MODE/SELECT switch status	• 000: MODE state • 010: NEUTRAL state • 100: SELECT state • Others: Reserved
	bit2 to 0	Reserved	—

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

CITL_GetTime

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

■ Format

short CITL_GetTime(short* psGetData, unsigned long ulBufSize)

■ Argument

Argument	Name	Description	IN/OUT
psGetData	Clock data storage destination	Specify the storage destination of the clock data.	OUT
ulBufSize	Clock data storage destination size	Specify the clock data 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 intelligent function 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 information to be stored in the clock data storage destination (psGetData) is as follows.
  (Available range: January 1, 1980 to December 31, 2079)

Storage position	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)

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

CITL_GetUnitStatus

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

■ Format

short CITL_GetUnitStatus(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 intelligent function module, and stores it in 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.)

Storage position		Description	Status
plStatusBuf[0]	bit31 to 8	Reserved	—
	bit7 to 4		
	bit3 to 0		
plStatusBuf[1]	bit31 to 16	Reserved	—
	bit15 to 9		
	bit8, 7	Time synchronization operating status	• 0: Automatic synchronization • 1: Stop
	bit6, 5	Reserved	—
	bit4, 3	SD memory card status	• 0: Inserted (mounted) • 1: Inserted (unmounted) • 2: Not inserted
	bit2	Reserved	—
	bit1		
	bit0	Standard ROM shutdown status	• 0: Shutdown not performed • 1: Shutdown completed
plStatusBuf[2]	bit31 to 0	Index value for number of the standard ROM write cycle	—

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 36 CITL_GetErrInfo

CITL_MountMemoryCard

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

■ Format

short CITL_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.) • 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.
- The CITL_MountMemoryCard function is available when the status of the SD memory card is "Inserted (unmounted)". (The status of the SD memory card can be checked by the CITL_GetUnitStatus 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 CITL_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.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 43 CITL_GetUnitStatus
☞ Page 60 CITL_UnmountMemoryCard

CITL_RegistEventLog

This function registers an event log in the event history of a control CPU module.

■ Format

short CITL_RegistEventLog (long lEventCode, char* pcEventMsg)

■ Argument

Argument	Name	Description	IN/OUT
lEventCode	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 control CPU module.

The contents to be registered on the event history screen of the engineering tool 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	• For an RD55UP06-V: RD55UP06-V (Fixed) • For an RD55UP12-V: RD55UP12-V (Fixed)
Start I/O number	Input/output number of the C intelligent function module that executed the CITL_RegistEventLog function.
Detailed event code information	Detailed code (hexadecimal) specified to the detailed code (lEventCode)
Detailed event log information	Detailed information specified to the detailed information (pcEventMsg)
Cause	The event history was registered from the C intelligent function module dedicated function. (Fixed)

- The event history can be stored for the size of the event history file specified with an engineering tool.
  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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

CITL_ResetWDT

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

■ Format

short CITL_ResetWDT (short sType)

■ Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (When 'Reserved' is specified, an error is returned.) • 0: User WDT • Others: Reserved	IN

■ Description

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

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 29 CITL_EntryWDTInt
☞ Page 53 CITL_StartWDT
☞ Page 55 CITL_StopWDT

CITL_SetCollectData

This function sets data to be sampled in data sampling in each sequence scan.

■ Format

short CITL_SetCollectData (long* plSetData, unsigned short usSetHeaderInfo, unsigned short usRecordNum, unsigned long* pulRecordSize)

■ Argument

Argument	Name	Description	IN/OUT
plSetData	Storage destination for data set in the data sampling in each sequence scan setting	Specify the number of blocks, device type, start device number, and number of sampling points of a device to be sampled.	IN
usSetHeaderInfo	Header information setting	Specify header information not to output at the time of data sampling. (If 0 is specified, all header information is output.)	IN
usRecordNum	Number of retainable records	Specify the size of the temporary area (number of retainable records) to store devices to be sampled.	IN
pulRecordSize	Record size storage destination	Specify the storage destination of the record size calculated by the storage destination for data set in the data sampling in each sequence scan setting (plSetData) and the header information setting (usSetHeaderInfo). (When NULL is specified, the record size is not stored.)	OUT

- Set so that the total number of points specified for each block is 32768 points or less.
  Otherwise, a size error occurs.
- Specify a value of '1' or more for the number of sampling points. When '0' is specified for the number of sampling points or the number of blocks, an error is returned.
- When a value other than NULL is specified for the storage destination for data set in the data sampling in each sequence scan setting (plSetData), the temporary area will be reserved according to the specified settings. If the area is already reserved, the reserved area will be released.
  When NULL is specified for the storage destination for data set in the data sampling in each sequence scan setting (plSetData), settings will be cleared and the temporary area will be released.
- Specify a device which can be accessed with a MELSEC iQ-R series data link function for the device type.
- The specification method of the storage destination for data set in the data sampling in each sequence scan setting (plSetData) is as follows:

Storage position	Description	Block
plSetData[0]	Number of blocks	—
plSetData[1]	Device type	Block 1
plSetData[2]	Start device number	
plSetData[3]	Number of sampling points	
plSetData[4]	Device type	Block 2
plSetData[5]	Start device number	
plSetData[6]	Number of sampling points	
⋮	⋮	⋮
plSetData[3(n-1)+1]	Device type	Block n
plSetData[3(n-1)+2]	Start device number	
plSetData[3(n-1)+3]	Number of sampling points	

- The specification method of the header information setting (usSetHeaderInfo) is as follows:

Definition names can be combined by the OR operator.

Definition name	Description
REMOVE_INDEX	An index is not output.
REMOVE_DATE	A date and time is not output.
REMOVE_DATA_MISS	Data missing information is not output.

- Specify a value of '1' or more for the number of retainable records (usRecordNum).
  When '0' is specified, an error is returned.

■ Description

- This function sets data to be sampled in each sequence scan according to the storage destination for data set in the data sampling in each sequence scan setting (plSetData).
- When executing the CITL_SetCollectData function multiple times, the last executed setting is enabled.
- This function sets data to be sampled in each sequence scan only. To start data sampling in each sequence scan, execute the CITL_StartCollectData function.

Precautions

Setting is applied only when it ends normally.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 32 CITL_GetCollectData
☞ Page 52 CITL_StartCollectData
☞ Page 54 CITL_StopCollectData
☞ Page 61 CITL_WaitCollectDataRecvEvent

CITL_SetLEDStatus

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

■ Format

short CITL_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.) • 0: USER LED • Others: Reserved	IN
usLedInfo	LED status information	Specify the LED status information.	IN

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

Stored information	LED status
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 the C intelligent function module to the status specified by the LED status information (usLedInfo).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 39 CITL_GetLEDStatus

CITL_SetSyncTimeStatus

This function sets the operating status of time synchronization of a C intelligent function module.

■ Format

short CITL_SetSyncTimeStatus (unsigned short usSyncTimeStatus)

■ Argument

Argument	Name	Description	IN/OUT
usSyncTimeStatus	Time synchronization operating status	Specify the operating status of time synchronization.	IN

The specification method of the time synchronization operating status (usSyncTimeStatus) is as follows:

Either a definition name or a value can be specified as the time synchronization operating status.

Operating status	Definition name	Value
Automatic synchronization	SYNCTIME_AUTO	0
Stop	SYNCTIME_STOP	1

■ Description

- This function sets the operating status of time synchronization of a C intelligent function module. (The initial status is automatic synchronization.)
- When specifying automatic synchronization to the time synchronization operating status (usSyncTimeStatus), the time is synchronized with that of a control CPU module every 500 ms.
- When specifying stop to the time synchronization operating status (usSyncTimeStatus), synchronization with the time of a control CPU module every 500 ms is stopped.
- The operating status of time synchronization can be acquired with the CITL_GetUnitStatus function.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 43 CITL_GetUnitStatus

☞ Page 56 CITL_SyncTime

CITL_ShutdownRom

This function shuts down the standard ROM of a C intelligent function module.

■ Format

short CITL_ShutdownRom (void)

■ Argument

None

■ Description

- This function shuts down the standard ROM of a C intelligent function module.
  (The shutdown status can be checked with the CITL_GetUnitStatus function.)
- It is used to shut down the standard ROM before turning the power of a C intelligent function module OFF. After the shutdown, file operations (creating, deleting, and overwriting a file) to the standard ROM cannot be performed. Reference to the standard ROM is possible.
- Before calling CITL_ShutdownRom function, it is necessary to ensure that the access to (reading from/writing to) the standard ROM is stopped and all files are closed. Otherwise, data in the standard ROM may be corrupted 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 standard ROM.
  Also, an error occurs when configuring the settings on the "Service and Account Settings" screen.
- If the standard ROM is already shut down, this function ends normally without processing.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 43 CITL_GetUnitStatus
☞ Page 44 CITL_MountMemoryCard
☞ Page 60 CITL_UnmountMemoryCard

CITL_StartCollectData

This function starts data sampling in each sequence scan.

■ Format

short CITL_StartCollectData (void)

■ Argument

None

■ Description

This function starts data sampling in each sequence scan.

This function performs according to the contents set with the CITL_SetCollectData function.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 32 CITL_GetCollectData

☞ Page 47 CITL_SetCollectData

☞ Page 54 CITL_StopCollectData

☞ Page 61 CITL_WaitCollectDataRecvEvent

CITL_StartWDT

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

■ Format

short CITL_StartWDT(short sType, short sInterval)

■ Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (When 'Reserved' is specified, an error is returned.) • 0: User WDT • Others: Reserved	IN
sInterval	WDT interval	Specify the interval of WDT in 10 ms units. (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 sInterval×10 ms and starts the user WDT.
- When the WDT is not reset periodically within the set time (by execution of the CITL_ResetWDT function), the user WDT error will occur. When the user WDT error occurs, the C intelligent function module will be in the moderate error status. (The RUN LED turns ON, and the ERR LED starts flashing.)
- When CITL_StartWDT function is executed while the WDT is running, an error will be returned.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 29 CITL_EntryWDTInt
☞ Page 46 CITL_ResetWDT
☞ Page 55 CITL_StopWDT

CITL_StopCollectData

This function stops data sampling in each sequence scan.

■ Format

short CITL_StopCollectData (void)

■ Argument

None

■ Description

This function stops data sampling in each sequence scan.

To restart data sampling in each sequence scan, execute the CITL_StartCollectData function.

Precautions

When executing the CITL_StopCollectData function, the buffer memory (data missing status) of a C intelligent function module is initialized.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 32 CITL_GetCollectData
☞ Page 47 CITL_SetCollectData
☞ Page 52 CITL_StartCollectData
☞ Page 61 CITL_WaitCollectDataRecvEvent

CITL_StopWDT

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

■ Format

short CITL_StopWDT(short sType)

■ Argument

Argument	Name	Description	IN/OUT
sType	WDT type	Specify the WDT type. (When 'Reserved' is specified, an error is returned.) • 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 29 CITL_EntryWDTInt
☞ Page 46 CITL_ResetWDT
☞ Page 53 CITL_StartWDT

CITL_SyncTime

This function synchronizes the time of a C intelligent function module with that of a control CPU module.

■ Format

short CITL_SyncTime (void)

■ Argument

None

■ Description

This function synchronizes the time of a C intelligent function module with that of a control CPU module.

■ Return value

Return value	Description
0 (0000H)	Normal

■ Relevant function

☞ Page 43 CITL_GetUnitStatus
☞ Page 50 CITL_SetSyncTimeStatus

CITL_SysClkRateGet

This function reads the system clock rate specified with the CITL_SysClkRateSet function from the flash ROM.

■ Format

short CITL_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 CITL_SysClkRateSet function from the flash ROM.

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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 58 CITL_SysClkRateSet

CITL_SysClkRateSet

This function saves the specified system clock rate in the flash ROM.

■ Format

short CITL_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. • 0: Default value (60 Hz) • 60 to 1000: Specified clock rate value	IN
psRestart	Restart necessity flag	Stores the necessity to restart a C intelligent function module after the execution of this function. (When 'NULL' is specified, the restart necessity flag is not stored.) • 0: Restart is not required. (The C intelligent function module has already been running at the specified clock rate.) • 1: Restart is required. (The C intelligent function 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 flash ROM.
  The specified system clock rate will be enabled after restarting a C intelligent function 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), restart the C intelligent function module by stopping the application processing and resetting the CPU module or turning the power OFF and ON.
- For more details on system clock rate, refer to the manual for VxWorks.

Precautions

- Execute this function only once after a C intelligent function 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 57 CITL_SysClkRateGet

CITL_ToBuf

This function writes data to the buffer memory of a C intelligent function module.

■ Format

short CITL_ToBuf (unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf, unsigned long ulBufSize)

■ Argument

Argument	Name	Description	IN/OUT
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 buffer memory of a C intelligent function module. Data is written by specifying an offset address from the start of the buffer memory of a C intelligent function module.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 31 CITL_FromBuf

CITL_UnmountMemoryCard

This function unmounts the SD memory card inserted to a C intelligent function module.

■ Format

short CITL_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.) • 1: SD memory card • Others: Reserved	IN

■ Description

- This function unmounts the drive specified to the target drive (sDrive).
- The CARD RDY LED is flashing, which indicates that process of unmounting the memory card is in progress, and later upon successful completion, the CARD RDY LED turns OFF.
- The CITL_UnmountMemoryCard function is available when the status of the SD memory card is "Inserted (mounted)". (The status of the SD memory card can be checked by the CITL_GetUnitStatus function.)
- When the status of the SD memory card has been already unmounted, this function ends normally without processing.

Precautions

Before calling the CITL_UnmountMemoryCard function, create a program so that accessing to the target drive is stopped and all files are closed. Otherwise, data may be corrupted or a file system error may occur.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 43 CITL_GetUnitStatus
☞ Page 44 CITL_MountMemoryCard

CITL_WaitCollectDataRecvEvent

This function waits for data to be sampled in data sampling in each sequence scan.

■ Format

short CITL_WaitCollectDataRecvEvent (unsigned short usWaitRecord, unsigned long ulTimeout)

■ Argument

Argument	Name	Description	IN/OUT
usWaitRecord	Number of wait records	Specify the number of records waiting to be sampled.	IN
ulTimeout	Timeout value	Specify the timeout value in ms units (0H to FFFFFFFFH). (When FFFFFFFFH is specified, the function waits for an event infinitely.)	IN

■ Description

- This function waits until the number of records specified to the number of wait records (usWaitRecord) is sampled in the temporary area for data sampling in each sequence scan.
- When data has already been sampled in the temporary area at the time of executing the CITL_WaitCollectDataRecvEvent function, this function ends normally without waiting for data to be sampled. Acquire the target data in the temporary area and execute the function again. When data sampling in each sequence scan is stopped, an error occurs.
- The specified timeout value is rounded to the tick unit. Specify a timeout value of one tick or more.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 32 CITL_GetCollectData
☞ Page 47 CITL_SetCollectData
☞ Page 52 CITL_StartCollectData
☞ Page 54 CITL_StopCollectData

CITL_WaitSwitchEvent

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

■ Format

short CITL_WaitSwitchEvent(short sSwitch, unsigned long ulTimeout)

■ Argument

Argument	Name	Description	IN/OUT
sSwitch	Switch interrupt event type	Specify the switch interrupt event type. • 0: Reserved • 1: Reserved • 2: MODE switch interrupt event	IN
ulTimeout	Timeout	Specify the timeout value in ms units (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 the time when this function is called, this function returns immediately.
- If the same switch interrupt event has been notified several times at the time of calling the CITL_WaitSwitchEvent function, it is treated as a single switch interrupt event.
- The specified timeout value is rounded to the tick unit. Specify a timeout value of one tick or more.

Precautions

For the MODE switch interrupt event, an event issuance status cannot be judged from the appearance. To check the issued status of MODE switch interrupt event, implement the processing such as receiving a switch interrupt event using the CITL_WaitSwitchEvent function and making the USER LED turn ON.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 39 CITL_GetLEDStatus

CITL_WaitTimerEvent

This function waits for a timer event to occur.

■ Format

short CITL_WaitTimerEvent (long lEventNo)

■ Argument

Argument	Name	Description	IN/OUT
lEventNo	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 (lEventNo) to occur.
- The occurrence cycle of the timer event number (1 to 16) can be set, changed, or cleared by the CITL_EntryTimerEvent function.
- When reset operation is performed, any event that has occurred prior to reset is discarded.
- Using the CITL_WaitTimerEvent 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 the CITL_WaitTimerEvent 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 CITL_EntryTimerEvent function.

Precautions

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

- If the synchronization type is batch synchronization, this function is called later, cancel the waiting status of all the tasks waiting for an event. However, if there is no task in the waiting status at the time of event occurrence, the waiting status is not canceled even if the CITL_WaitTimerEvent function is called later.
- 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). However, if there is no task in the wait state at the time of event occurrence, the wait state will not be canceled even if the CITL_WaitTimerEvent function is called later.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 27 CITL_EntryTimerEvent

CITL_WaitYEvent

This function waits for the output signal (Y) interrupt event notification.

■ Format

short CITL_WaitYEvent (short* psYNo, unsigned long ulTimeout, unsigned short* pusSetEventNo)

■ Argument

Argument	Name	Description	IN/OUT
psYNo	Output signal (Y) number	Specify the output signal (Y) number.	IN
ulTimeout	Timeout value	Specify the timeout value in ms units (0H to FFFFFFFFH). (When FFFFFFFFH is specified, the function waits for an event infinitely.)	IN
pusSetEventNo	Occurred output signal (Y) event	Stores the occurred event. (Stores the output signal (Y) number of the notified interrupt event.)	OUT

- The specification method of the output signal (Y) number is as follows:

Storage position	Description
psYNo[0]	Number of interrupt event settings (1 to 16)
psYNo[1]	Output signal (Y) number of the first interrupt event (0x10 to 0x1F)
psYNo[2]	Output signal (Y) number of the second interrupt event (0x10 to 0x1F)
⋮	⋮
psYNo[8]	Output signal (Y) number of the 8th interrupt event (0x10 to 0x1F)

- The occurred output signal (Y) event (pusSetEventNo) is stored as follows:

Storage position	Description
pusSetEventNo[0]	Output signal (Y) number of the notified interrupt event

■ Description

- This function waits for an interrupt event specified to the output signal (Y) number (psYNo) for the time specified to the timeout value (ulTimeout).
- When multiple interrupt events occur, the interrupt events are notified in ascending order of the output signal (Y) number.
- If an interrupt event has already been notified at the time when this function is called, this function returns immediately. 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 interrupt event number (the output signal (Y) number) at the time of calling the CITL_WaitYEvent function, it is treated as a single interrupt event notification.
- Set the output signal (Y) 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.
- Design a program so that this function is not called simultaneously by specifying the same interrupt event (output signal (Y) number) from multiple tasks. Otherwise, the execution of the interrupt event notified task is unpredictable.
- The output signal (Y) interrupt event notification wait function (CITL_WaitYEvent function)and the function executing interrupt routine when output signal (Y) interrupts (Defined by CITL_EntryYInt/CITL_EnableYInt/CITL_DisableYInt function) operate independently. These functions operate independently even if interrupt occurs by the same output signals (Y).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

CITL_X_In_Bit

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

■ Format

short CITL_X_In_Bit (unsigned short usXNo, unsigned short* pusData)

■ Argument

Argument	Name	Description	IN/OUT
usXNo	Input signal	Specify the input signal (X). (0 to 31)	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). • 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 value of an input signal (X) read to the data storage destination (pusData) is stored.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 66 CITL_X_In_Word
☞ Page 67 CITL_X_Out_Bit
☞ Page 68 CITL_X_Out_Word
☞ Page 69 CITL_Y_In_Bit
☞ Page 70 CITL_Y_In_Word

CITL_X_In_Word

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

■ Format

short CITL_X_In_Word (unsigned short usXNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

■ Argument

Argument	Name	Description	IN/OUT
usXNo	Start input signal	Specify a start input signal (X). (Only 0x00 and 0x10 can be specified.)	IN
usSize	Read data size	Specify the read data size in word units. • When start input signal is 0x00: Only 1 and 2 can be specified. • When start input signal is 0x10: Only 1 can be specified.	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 a start input signal (X) specified to the start input signal (usXNo), and stores it in the data storage destination (pusDataBuf).
- Specify the area size of the data storage destination (pusDataBuf) to the data storage destination size (usBufSize).
- When 0x10 is specified to the start input signal (usXNo) and two words are specified to the read data size (usSize), the CITL_X_In_Word function returns an I/O access size error and the value is not read.
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

Storage position	Description
pusDataBuf[0]	Data of usXNo+FH to usXNo
pusDataBuf[1]	Data of usXNo+1FH to usXNo+10H

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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 65 CITL_X_In_Bit
☞ Page 67 CITL_X_Out_Bit
☞ Page 68 CITL_X_Out_Word
☞ Page 69 CITL_Y_In_Bit
☞ Page 70 CITL_Y_In_Word

CITL_X_Out_Bit

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

■ Format

short CITL_X_Out_Bit (unsigned short usXNo, unsigned short usData)

■ Argument

Argument	Name	Description	IN/OUT
usXNo	Input signal	Specify the input signal (X). (0 to 31)	IN
usData	Write data	Specify the written data. (Specify the value of bit 0.) • 0: OFF • 1: ON	IN

■ Description

- This function writes to an input signal (X) specified to the input signal (usXNo) in bit (1-point) units. (Turns ON/OFF.)
- An input signal (X) turns ON/OFF according to a value specified to bit 0 in the data storage destination (usData). (Values of bit 1 to 7 are ignored.)
- Only the area available for user can be written. Even though the data is written by the CITL_X_Out_Bit function in the area other than the area available for user, no error will occur, but the value will not be written.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 65 CITL_X_In_Bit
☞ Page 66 CITL_X_In_Word
☞ Page 68 CITL_X_Out_Word
☞ Page 69 CITL_Y_In_Bit
☞ Page 70 CITL_Y_In_Word

CITL_X_Out_Word

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

■ Format

short CITL_X_Out_Word (unsigned short usXNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

■ Argument

Argument	Name	Description	IN/OUT
usXNo	Start input signal	Specify a start input signal (X). (Only 0x00 and 0x10 can be specified.)	IN
usSize	Write data size	Specify the write data size in word units. • When start input signal is 0x00: Only 1 and 2 can be specified. • When start input signal is 0x10: Only 1 can be specified.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN
usBufSize	Data storage destination size	Specify '0'.	IN

■ Description

- This function writes data in the data storage destination (pusDataBuf) from a start input signal (X) specified to the start input signal (usXNo) to an input signal (X) for the size specified to the data size (usSize).
- When 0x10 is specified to the start input signal (usXNo) and two words are specified to the write data size (usSize), the CITL_X_In_Word function returns an I/O access size error and the value is not written.
- Only the area available for user can be written. Even though the data is written by the CITL_X_Out_Word function in the area other than the area available for user, no error will occur, but the value will not be written.
- Store write data in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

Storage position	Description
pusDataBuf[0]	Data of usXNo+FH to usXNo
pusDataBuf[1]	Data of usXNo+1FH to usXNo+10H

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 65 CITL_X_In_Bit
☞ Page 66 CITL_X_In_Word
☞ Page 67 CITL_X_Out_Bit
☞ Page 69 CITL_Y_In_Bit
☞ Page 70 CITL_Y_In_Word

CITL_Y_In_Bit

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

■ Format

short CITL_Y_In_Bit (unsigned short usYNo, unsigned short* pusData)

■ Argument

Argument	Name	Description	IN/OUT
usYNo	Output signal	Specify the output signal (Y). (0 to 31)	IN
pusData	Data storage destination	Specify the storage destination of read data. The following values are stored depending on the value of an output signal (Y). • 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 value of a read output signal (Y) is stored in the data storage destination (pusData).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 65 CITL_X_In_Bit
☞ Page 66 CITL_X_In_Word
☞ Page 67 CITL_X_Out_Bit
☞ Page 68 CITL_X_Out_Word
☞ Page 70 CITL_Y_In_Word

CITL_Y_In_Word

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

■ Format

short CITL_Y_In_Word (unsigned short usYNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

■ Argument

Argument	Name	Description	IN/OUT
usYNo	Start output signal	Specify a start output signal (Y). (Only 0x00 and 0x10 can be specified.)	IN
usSize	Read data size	Specify the read data size in word units. • When start output signal is 0x00: Only 1 and 2 can be specified. • When start output signal is 0x10: Only 1 can be specified.	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 a start output signal (Y) specified to the start output signal (usYNo), and stores it in the data storage destination (pusDataBuf).
- Specify the area size of the data storage destination (pusDataBuf) to the data storage destination size (usBufSize).
- When 0x10 is specified to the start output signal (usYNo) and two words are specified to the read data size (usSize), the CITL_Y_In_Word function returns an I/O access size error and the value is not read.
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

Storage position	Description
pusDataBuf[0]	Data of usYNo+FH to usYNo
pusDataBuf[1]	Data of usYNo+1FH to usYNo+10H

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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 65 CITL_X_In_Bit
☞ Page 66 CITL_X_In_Word
☞ Page 67 CITL_X_Out_Bit
☞ Page 70 CITL_Y_In_Word
☞ Page 69 CITL_Y_In_Bit

C intelligent function module dedicated functions for ISR

CITL_DisableYInt_ISR

This function disables the routine registered with the CITL_EntryYInt function.

■ Format

short CITL_DisableYInt_ISR(short sYNo)

■ Argument

Argument	Name	Description	IN/OUT
sYNo	Output signal (Y) number	Specify the output signal (Y) number. (If -1 is specified, disable all the registered routines.)	IN

■ Description

- This function disables the routine registered with the CITL_EntryYInt function.
  (The registered routine is not executed when an output signal (Y) interrupt occurs.)
- Specify the output signal (Y) number (sYNo) specified in the CITL_EntryYInt function in the output signal (Y) number (sYNo).

■ ⚠WARNING

CITL_DisableYInt_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- Do not specify an unregistered output signal (Y).

■ Return value

Return value	Description
0 (0000H)	Normal

■ Relevant function

☞ Page 30 CITL_EntryYInt

☞ Page 72 CITL_EnableYInt_ISR

CITL_EnableYInt_ISR

This function enables the routine registered with the CITL_EntryYInt function.

■ Format

short CITL_EnableYInt_ISR (short sYNo)

■ Argument

Argument	Name	Description	IN/OUT
sYNo	Output signal (Y) number	Specify the output signal (Y) number. (If -1 is specified, enable all the registered routines.)	IN

■ Description

- This function enables the routine registered with the CITL_EntryYInt function. (The registered routine is executed when an output signal (Y) interrupt occurs.)
- Specify the output signal (Y) number (sYNo) specified in the CITL_EntryYInt function in the output signal (Y) number (sYNo).

■ ⚠WARNING

CITL_EnableYInt_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- Do not specify an unregistered output signal (Y).

■ Return value

Return value	Description
0 (0000H)	Normal

■ Relevant function

☞ Page 30 CITL_EntryYInt

☞ Page 71 CITL_DisableYInt_ISR

CITL_FromBuf_ISR

This function reads data from the buffer memory of a C intelligent function module.

■ Format

short CITL_FromBuf_ISR (unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf)

■ Argument

Argument	Name	Description	IN/OUT
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 buffer memory of a C intelligent function module, and stores it in the data storage destination (pusDataBuf). Data is read by specifying an offset address from the start of the buffer memory of a C intelligent function module.

Restriction

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

■ ⚠WARNING

CITL_FromBuf_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- The offset (ulOffset) is a multiple of 2.
- The data area for the size (words) of the read data is reserved.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 78 CITL_ToBuf_ISR

CITL_GetCounterMicros_ISR

This function acquires a 1 μs counter value of a C intelligent function module.

■ Format

short CITL_GetCounterMicros_ISR (unsigned long* pulMicros)

■ Argument

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

■ Description

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

Restriction

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

■ Return value

Return value	Description
0 (0000H)	Normal

■ Relevant function

☞ Page 75 CITL_GetCounterMillis_ISR

CITL_GetCounterMillis_ISR

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

■ Format

short CITL_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 a C intelligent function module, and stores it 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 CITL_GetCounterMillis_ISR function in a routine other than the one registered in the interrupt.

■ Return value

Return value	Description
0 (0000H)	Normal

■ Relevant function

☞ Page 74 CITL_GetCounterMicros_ISR

CITL_RegistEventLog_ISR

This function registers an event log in the event history of a control CPU module.

■ Format

short CITL_RegistEventLog_ISR (long lEventCode, char* pcEventMsg)

■ Argument

Argument	Name	Description	IN/OUT
lEventCode	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 control CPU module.

The contents to be registered on the event history screen of the engineering tool 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	• For an RD55UP06-V: RD55UP06-V (Fixed) • For an RD55UP12-V: RD55UP12-V (Fixed)
Start I/O number	Input/output number of the C intelligent function module that executed the CITL_RegistEventLog_ISR function.
Detailed event code information	Detailed code (hexadecimal) specified to the detailed code (lEventCode)
Detailed event log information	Detailed information specified to the detailed information (pcEventMsg)
Cause	The event history was registered from the C intelligent function module dedicated function. (Fixed)

- The event history can be stored for the size of the event history file specified with an engineering tool. 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.

Restriction

Do not execute CITL_RegistEventLog_ISR function in a routine other than the one registered in the interrupt.

■ ⚠WARNING

CITL_RegistEventLog_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- The detailed information within the range is specified.

■ Return value

Return value	Description
0 (0000H)	Normal

CITL_SetLEDStatus_ISR

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

■ Format

short CITL_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 of the LED status information (usLedInfo) is as follows:

Stored information	LED status
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 the C intelligent function module to the status specified by the LED status information (usLedInfo).

Restriction

Do not execute CITL_SetLEDStatus_ISR function in a routine other than the one registered in the interrupt.

■ Return value

Return value	Description
0 (0000H)	Normal

CITL_ToBuf_ISR

This function writes data to the buffer memory of a C intelligent function module.

■ Format

short CITL_ToBuf_ISR (unsigned long ulOffset, unsigned long ulSize, unsigned short* pusDataBuf)

■ Argument

Argument	Name	Description	IN/OUT
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 buffer memory of a C intelligent function module. Data is written by specifying an offset address from the start of the buffer memory of a C intelligent function module.

Restriction

- Do not execute CITL_ToBuf_ISR function in a routine other than the one registered in the interrupt.
- When data is written to the same buffer memory 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 buffer memory.

■ ⚠WARNING

CITL_ToBuf_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- The offset (ulOffset) is a multiple of 2.
- Do not specify outside the buffer memory or system area.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 73 CITL_FromBuf_ISR

CITL_X_In_Word_ISR

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

■ Format

short CITL_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). (Only 0x00 and 0x10 can be specified.)	IN
usSize	Read data size	Specify the read data size in word units. • When start input signal is 0x00: Only 1 and 2 can be specified. • When start input signal is 0x10: Only 1 can be specified.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

■ Description

- This function reads an input signal (X) for the size specified to the read data size (usSize) from the start input signal (usXNo), and stores it in the data storage destination (pusDataBuf).
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

Storage position	Description
pusDataBuf[0]	Data of usXNo+FH to usXNo
pusDataBuf[1]	Data of usXNo+1FH to usXNo+10H

Restriction

Do not execute CITL_X_In_Word_ISR function in a routine other than the one registered in the interrupt.

■ ⚠WARNING

CITL_X_In_Word_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- The data area for the size (words) of the read data is reserved.
- The input signal (X) within the range (0H to 1FH) is specified.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 80 CITL_X_Out_Word_ISR
☞ Page 81 CITL_Y_In_Word_ISR

CITL_X_Out_Word_ISR

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

■ Format

short CITL_X_Out_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). (Only 0x00 and 0x10 can be specified.)	IN
usSize	Write data size	Specify the write data size in word units. • When start input signal is 0x00: Only 1 and 2 can be specified. • When start input signal is 0x10: Only 1 can be specified.	IN
pusDataBuf	Data storage destination	Specify the storage destination of write data.	IN

■ Description

- This function writes to the input signal (X) with a specific write data size (usSize) from the start input signal (usXNo) depending on the data storage destination (pusDataBuf). (Turns ON/OFF.)
- Only the area available for user can be written. Even though the data is written by the CITL_X_Out_Word function in the area other than the area available for user, no error will occur, but the value will not be written.
- Store write data in the data storage destination (pusDataBuf) in ascending order from the lower bit.

Storage position	Description
pusDataBuf[0]	Data of usXNo+FH to usXNo
pusDataBuf[1]	Data of usXNo+1FH to usXNo+10H

Restriction

Do not execute CITL_X_Out_Word_ISR function in a routine other than the one registered in the interrupt.

■ ⚠WARNING

CITL_X_Out_Word_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- The input signal (X) within the range (0H to 1FH) is specified.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 79 CITL_X_In_Word_ISR

☞ Page 81 CITL_Y_In_Word_ISR

CITL_Y_In_Word_ISR

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

■ Format

short CITL_Y_In_Word_ISR (unsigned short usYNo, unsigned short usSize, unsigned short* pusDataBuf, unsigned short usBufSize)

■ Argument

Argument	Name	Description	IN/OUT
usYNo	Start output signal	Specify a start output signal (Y). (Only 0x00 and 0x10 can be specified.)	IN
usSize	Read data size	Specify the read data size in word units. • When start output signal is 0x00: Only 1 and 2 can be specified. • When start output signal is 0x10: Only 1 can be specified.	IN
pusDataBuf	Data storage destination	Specify the storage destination of read data.	OUT

■ Description

- This function reads an output signal (Y) for the size specified to the read data size (usSize) from the start output signal (usYNo), and stores it in the data storage destination (pusDataBuf).
- Read data is stored in the data storage destination (pusDataBuf) in ascending order from the lower bit as shown below.

Storage position	Description
pusDataBuf[0]	Data of usYNo+FH to usYNo
pusDataBuf[1]	Data of usYNo+1FH to usYNo+10H

Restriction

Do not execute CITL_Y_In_Word_ISR function in a routine other than the one registered in the interrupt.

■ ⚠WARNING

CITL_Y_In_Word_ISR function does not check the specified argument.

Create a program with the following conditions in mind.

- The data area for the size (words) of the read data is reserved.
- The output signal (Y) within the range (0H to 1FH) is specified.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 79 CITL_X_In_Word_ISR

☞ Page 80 CITL_X_Out_Word_ISR

3.2 MELSEC iQ-R Series Data Link Functions

This section shows the details of the MELSEC iQ-R series data link function.

mdrClose

This function closes a communication line (channel).

■ Format

short mdrClose(long lPath)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN

■ Description

This function closes a channel opened with the mdrOpen function.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 90 mdrOpen

mdrControl

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

■ Format

short mdrControl(long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, short sCode)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the CPU number of the target CPU module. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
sCode	Instruction code	Specify the contents of the remote operation in numerical value.	IN

The specification method of the instruction code (sCode) is as follows:

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

■ Description

This function changes the status of a CPU module specified to the station number (lStNo) to the one specified to the instruction code (sCode).

Restriction

This function cannot be executed for C Controller module, PC CPU module, and WinCPU module.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

☞ Page 90 mdrOpen

mdrDevRst

This function resets bit devices.

■ Format

short mdrDevRst(long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long lDevType, long lDevNo)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the CPU number of the target CPU module. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
lDevType	Device type	Specify the device type of bit device. ☞ Page 14 Argument specification	IN
lDevNo	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 (lNetNo),the start I/O number (lIoNo),the station number (lStNo), the CPU number (sCPU),the device type (lDevType),and the device number (lDevNo).
- The mdrDevRst function is dedicated function for bit devices such as link relay (B) and internal relay (M).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

☞ Page 85 mdrDevSet

☞ Page 90 mdrOpen

mdrDevSet

This function sets bit devices.

■ Format

short mdrDevSet (long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long lDevType, long lDevNo)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the CPU number of the target CPU module. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
lDevType	Device type	Specify the device type of bit device. ☞ Page 14 Argument specification	IN
lDevNo	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 (lNetNo),the start I/O number (lIoNo),the station number (lStNo), the CPU number (sCPU),the device type (lDevType),and the device number (lDevNo).
- The mdrDevSet function is dedicated function for bit devices such as link relay (B) and internal relay (M).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

☞ Page 84 mdrDevRst

☞ Page 90 mdrOpen

mdrGetLabelInfo

This function acquires device information corresponding to label names.

■ Format

short mdrGetLabelInfo (long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long lLbCnt, void* pLbLst, long* plDevLst, unsigned long long* pullLbCode)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the CPU number of the target CPU module. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
lLbCnt	Number of labels	Specify the number of labels. (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
plDevLst	Device name array	Specify a device to store the acquired device information. (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 (plDevLst) in a randomly selected device format listed below.

Storage position	Description	Block
plDevLst[0]	Number of blocks	—
plDevLst[1]	Device type	Block 1
plDevLst[2]	Start device number	
plDevLst[3]	Number of read points	
plDevLst[4]	Device type	Block 2
plDevLst[5]	Start device number	
plDevLst[6]	Number of read points	
⋮	⋮	⋮
plDevLst[3n+1]	Device type	Block n
plDevLst[3n+2]	Start device number	
plDevLst[3n+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 (plDevLst).

■ Description

- This function reads labels of a CPU module specified to the network number (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- Reserve the area for the label name array (plDevLst) in the call source.
- Reserve the area for (lLbCnt × 3 + 1) for the size of area of the device name array (plDevLst).
- 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 number of read points of the label, '0' is stored.

Error number	Occurrence condition
-82(FFB2H)	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. (Devices are index-modified or indirectly specified.)
-84(FFB4H)	The specification method for devices assigned to labels is incorrect. (Devices are bit-specified or digit-specified.)

- 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 the first detected Label1 error (-82) is returned.
- Even if the mdrGetLabelInfo function returns the error (-82 or -84), the value is stored in the device name array (plDevLst) for the label that acquired device information successfully.
- The specification method of the label name to specify to the label 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. • One-dimensional array: Label name [m] • Two-dimensional array: Label name [m, n] • Three-dimensional array: Label name [m, n, l]	• 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. • Contact: Label name.S • Coil: Label name.C • Current value: Label name.N	• 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 (mdrRandRLabel/mdrRandWLabel).
- For accessible CPU modules, refer to the following manual.
  MELSEC iQ-R C Intelligent Function Module User's Manual (Application)

■ Example

The following tables show the examples of values set to the label name array (pLbLst) and data read to the device name array (plDevLst). (For five labels to be read: Label1 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 binary format from the saved text file on a user program, and store the address of the label name passed to the label name array (pLbLst) on the memory.

• Values set to the label name array (pLbLst)

Setting target	Setting 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

• Data to be read to the device name array (plDevLst)

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

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter.*1 ☞ Page 108 ERROR CODE LIST

*1 For return values which do 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 82 mdrClose
☞ Page 90 mdrOpen
☞ Page 94 mdrRandRLabel
☞ Page 100 mdrRandWLabel

mdrInit

This function initializes communication route information.

■ Format

short mdrInit (long lPath)

■ Argument

Argument	Name	Description	IN/OUT
lPath	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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

☞ Page 90 mdrOpen

mdrOpen

This function opens a communication line (channel).

■ Format

short mdrOpen (short sChan, short sMode, long* plPath, long lTimeOut)

■ Argument

Argument	Name	Description	IN/OUT
sChan	Channel	Specify a communication line (channel). ☞ Page 14 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
lTimeOut	Timeout value	Specify the timeout value of MELSEC iQ-R series data link function for MELSEC iQ-R series bus interface (Channel No.12) • Setting range: 1 to 360 sec (1 sec unit)	IN

■ Description

- The path of the channel opened by the mdrOpen function is used when MELSEC iQ-R series data link functions are executed.
- To end the program, close the path of the opened channel using the mdrClose function.

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

mdrRandR

This function reads devices randomly.

■ Format

short mdrRandR(long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long* plDev, short* psBuf, long lBufSize)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
plDev	Randomly selected device	Specify the number of blocks, device type, start device number, and points of devices to be read.	IN
psBuf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
lBufSize	Read data storage destination size	Specify the size of area reserved in the read data storage destination in byte units.	IN

The specification method of the randomly selected device (plDev) is as follows:

Specification position	Description	Block
plDev[0]	Number of blocks	—
plDev[1]	Device type	Block 1
plDev[2]	Start device number	
plDev[3]	Number of read points	
plDev[4]	Device type	Block 2
plDev[5]	Start device number	
plDev[6]	Number of read points	
⋮	⋮	⋮
plDevLst[3n+1]	Device type	Block n
plDevLst[3n+2]	Start device number	
plDevLst[3n+3]	Number of read points	

■ Description

- This function reads devices specified to the randomly selected device (plDev) from a module specified to the network number (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- 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 (plDev). 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) will occur.
- Communication time varies significantly depending on the contents specified to the randomly selected device (plDev). To reduce communication time, use the mdrReceive 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	'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)

Specification position	Specified value	Description	Block
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)

Storage position	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

(psBuf[0] to psBuf[10] = 11) × 2 = 22

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose
☞ Page 90 mdrOpen
☞ Page 98 mdrRandW

mdrRandRLabel

This function reads devices corresponding to labels randomly.

■ Format

short mdrRandRLabel (long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long* plDev, short* psBuf, long lBufSize, unsigned long long ullLbCode)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
plDev	Randomly selected device	Specify the number of blocks, device type, start device number, and points of devices to be read. (Specify the value acquired with the mdrGetLabelInfo function.)	IN
psBuf	Read data storage destination	Specify the storage destination (address) of read data.	OUT
lBufSize	Read data storage destination size	Specify the size of area reserved in the read data storage destination in byte units.	IN
ullLbCode	Label code	Specify the label code acquired with the mdrGetLabelInfo function.	IN

The specification method of the randomly selected device (plDev) is as follows:

Stored information position	Stored information	Block
plDev[0]	Number of blocks	—
plDev[1]	Device type	Block 1
plDev[2]	Start device number	
plDev[3]	Number of read points	
plDev[4]	Device type	Block 2
plDev[5]	Start device number	
plDev[6]	Number of read points	
⋮	⋮	⋮
plDevLst[3n+1]	Device type	Block n
plDevLst[3n+2]	Start device number	
plDevLst[3n+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 (plDev).

■ Description

- This function reads devices specified to the randomly selected device (plDev) from a module specified to the network number (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- 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 (plDev). 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) will occur.
- 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	'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)

Setting target	Specified value	Description	Block
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	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
plDev[8]	0	Start device number = 0	
plDev[9]	1	Number of read points = 1	
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	

Data to be read to the read data storage destination (psBuf)

Read position	Read device	Read value	Description
psBuf[0]	M100	0	The bit device for M100 is 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	The bit device for M0 is 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

(psBuf[0] to psBuf[10] = 11) × 2 = 22

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter.*1 ☞ Page 108 ERROR CODE LIST

*1 For return values which do 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 82 mdrClose
☞ Page 86 mdrGetLabelInfo
☞ Page 90 mdrOpen
☞ Page 100 mdrRandWLabel

mdrRandW

This function writes devices randomly.

■ Format

short mdrRandW(long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long* plDev, short* psBuf, long lBufSize)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU 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.	IN
lBufSize	Write data storage destination size	Specify the size of area reserved in the write data storage destination in byte units.	IN

The specification method of the randomly selected device (plDev) is as follows:

Stored information position	Stored information	Block
plDev[0]	Number of blocks	—
plDev[1]	Device type	Block 1
plDev[2]	Start device number	
plDev[3]	Number of write points	
plDev[4]	Device type	Block 2
plDev[5]	Start device number	
plDev[6]	Number of write points	
⋮	⋮	⋮
plDev[3n+1]	Device type	Block n
plDev[3n+2]	Start device number	
plDev[3n+3]	Number of write points	

■ 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 (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- 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) will occur.
- 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	Current value
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)

Write position	Specified value	Description	Block
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 5: 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)

Write position	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 = 0x0001
psBuf[6]	0x0000	Upper bit of LCN100 = 0x0000
psBuf[7]	0x0000	Lower bit of LCN101 = 0x0000
psBuf[8]	0x0001	Upper bit of LCN101 = 0x0001

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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

☞ Page 90 mdrOpen

☞ Page 91 mdrRandR

mdrRandWLabel

This function writes devices corresponding to labels randomly.

■ Format

short mdrRandWLabel (long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long* plDev, short* psBuf, long lBufSize, unsigned long long ullLbCode)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU 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.	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 mdrGetLabelInfo function.	IN

The specification method of the randomly selected device (plDev) is as follows:

Stored information position	Stored information	Block
plDev[0]	Number of blocks	—
plDev[1]	Device type	Block 1
plDev[2]	Start device number	
plDev[3]	Number of write points	
plDev[4]	Device type	Block 2
plDev[5]	Start device number	
plDev[6]	Number of write points	
⋮	⋮	⋮
plDev[3n+1]	Device type	Block n
plDev[3n+2]	Start device number	
plDev[3n+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 (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- 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) will occur.
- 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	Current value
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)

Setting target	Specified value	Description	Block
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 5: LCN100 to LCN101
plDev[8]	100	Start device number = 100	
plDev[9]	2	Number of write points = 2	

Data specified for the write data storage destination (psBuf)

Write position	Specified value	Description
psBuf[0]	0	Turns the bit device for 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 = 0x0001
psBuf[6]	0x0000	Upper bit of LCN100 = 0x0000
psBuf[7]	0x0000	Lower bit of LCN101 = 0x0000
psBuf[8]	0x0001	Upper bit of LCN101 = 0x0001

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 (0000H)	Error For details on the error, refer to the following chapter.*1 ☞ Page 108 ERROR CODE LIST

*1 For return values which do 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 82 mdrClose
☞ Page 86 mdrGetLabelInfo
☞ Page 90 mdrOpen
☞ Page 94 mdrRandRLabel

mdrReceive

This function reads devices in a batch.

■ Format

short mdrReceive(long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long lDevType, long lDevNo, long* plSize, short* psData)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
lDevType	Device type	Specify the device type for device to be read in batch.	IN
lDevNo	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
plSize	Read data size	Specify the read data size in byte units. (Specify the value in multiples of 4 when 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 (plSize) from a device, which is specified to the device type (lDevType) and the start device number (lDevNo), of a module specified to the network number (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- When the read data size exceeds the device range, a readable size is returned to the read data size (plSize).

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose

☞ Page 90 mdrOpen

☞ Page 104 mdrSend

mdrSend

This function writes devices in a batch.

■ Format

short mdrSend(long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, long lDevType, long lDevNo, long* plSize, short* psData)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU specification	IN
lDevType	Device type	Specify the device type for device to be written in batch.	IN
lDevNo	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
plSize	Write data size	Specify the write data size in byte units. (Specify the value in multiples of 4 when 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 (plSize) starting from a device, which is specified to the device type (lDevType) and the start device number (lDevNo), of a module specified to the network number (lNetNo), the start I/O number (lIoNo), the station number (lStNo), and the CPU number (sCPU).
- 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 (plSize).
- 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 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose
☞ Page 90 mdrOpen
☞ Page 103 mdrReceive

mdrTypeRead

This function reads the model code of a CPU module.

■ Format

short mdrTypeRead (long lPath, short sRoute, long lNetNo, long lIoNo, long lStNo, short sCPU, short* psCode)

■ Argument

Argument	Name	Description	IN/OUT
lPath	Path of channel	Specify the path of the opened channel.	IN
sRoute	Access route	Specify the access route to target module. • 0: CC-Link IE Controller Network • 1: CC-Link IE Field Network • 2: MELSECNET/H • 3: CC-Link • 4: Bus interface	IN
lNetNo	Network number	Specify the network number of target module. ☞ Page 14 Argument specification	IN
lIoNo	Start I/O number	Specify the start I/O number divided by 16 of the target module. ☞ Page 14 Argument specification	IN
lStNo	Station number	Specify the station number of target module. ☞ Page 14 Argument specification	IN
sCPU	CPU number	Specify the target CPU number. • 0: Control CPU specification • 1 to 4: Multiple CPU 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 code of the CPU module with the station number specified to the station number (lStNo).

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

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

Model code (hexadecimal)	CPU module
02E8H	Q03UDECPU
02E9H	Q04UDEHCPU
02EAH	Q06UDEHCPU
02EBH	Q13UDEHCPU
02ECH	Q26UDEHCPU
02EDH	Q50UDEHCPU
02EEH	Q100UDEHCPU
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
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
4C00H	R16MTCPU
4C01H	R32MTCPU
4C02H	R64MTCPU

■ Return value

Return value	Description
0 (0000H)	Normal
Other than 0 (0000H)	Error For details on the error, refer to the following chapter. ☞ Page 108 ERROR CODE LIST

■ Relevant function

☞ Page 82 mdrClose
☞ Page 90 mdrOpen

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		Description	Corrective action
Decimal	Hexadecimal		
1	0001H	■Driver not started The driver is not started.	• Check the channel number. • Correct the error that occurred when the driver is started. • Check the status of the system drive of the C intelligent function module. • Check if the operating system is running normally.
2	0002H	■Timeout error • A timeout occurred while waiting for the response. • During CC-Link communication, the request was issued to other stations when the station number of the own station is '64'. • The module specified as the communication target is not supported.	• Check the operating status and mounting condition of the access target station. • Retry on the user program. • Increase the timeout value of MELSEC iQ-R series data link function. • When issuing a request to other stations during CC-Link communication, set the station number of the own station other than '64'. • Check if the module specified as the communication target is supported.
66	0042H	■Already opened error The specified channel has already been opened.	The open processing is required only one time. (If this error occurred, the path of the correct channel will be returned to the argument.)
67	0043H	■Already closed error The specified channel has already been closed.	The close processing is required only one time.
69	0045H	■Unsupported function performing error An unsupported function in the target station was performed.	• Check the path of the channel, network number, and station number. • Check if the function performed in the target station is supported.
70	0046H	■Station number error • The specified station number is incorrect. • The request for other stations was issued to the own station, or the network number was not '0' even though the station number was the own station (FFH).	Correct the network number and station number specified in the user program.
77	004DH	■Memory reservation error ■Resource shortage error ■Task over error Securing sufficient memory failed. Or, there are too many tasks using the dedicated function library.	• The memory may be insufficient. End another running task or reduce the access size. • Reduce the number of tasks using the dedicated function library and retry the operation. • Check the size or number specified to the arguments of the user program. • Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
102	0066H	■Data send error ■Restart error Sending data failed. Or, an attempt was made to send data during restart.	• Retry. • Retry after completion of the restart. • Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
103	0067H	■Reception error Receiving data failed.	• Retry. • Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
130	0082H	■Device number error • The specified device number is out of range. • The specified bit device number is not multiple of 8.	Check the device number.

Error code		Description	Corrective action
Decimal	Hexadecimal		
131	0083H	■Number of device points error • The specified number of device points is out of range. • The specified number of bit device points is not a multiple of 8.	Check the device points.
16384 to 20479[*1]	4000H to 4FFFH	■Errors detected in the access target CPU module	Refer to the user's manual of the access target CPU module.
-25056	9E20H	■Processing code error A request which cannot be performed in the request destination was issued.	Check the network number and station number of the request destination.
-26336	9920H	■Routing request error for unsupported station A routing request to another loop was issued to a station which does not support the routing function.	Check the settings of routing parameters.
-28150	920AH	■Device access error during data link stop The devices (RX, RY, RWw, and RWr) of the own station were accessed when the data link was not performed.	• Check the specified start device number and size, or the device range of the parameter for the master station. • Restart the date link. (Note that data is written/read despite this error, however, the contents of the data will not be guaranteed.)
-28151	9209H	■Abnormal data reception error Abnormal response data received.	Check if an error occurred in the request destination CPU module or link module. (If the status is normal, resend the request.)
-28158	9202H	■WDT error WDT (system/user) error occurred.	Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-28410	9106H	■Target CPU busy error The target CPU module is busy.	• Add the processing to wait for the completion of the target operation or to retry the operation in the user program. • Increase the timeout time specified to the argument in the user program.
-28412	9104H	■Target CPU unsupported error An unsupported request was issued to the target CPU module.	Change the target CPU module specified in the user program.
-28413	9103H	■Target CPU down error The target CPU module is down.	Check the operating status of the target CPU and troubleshoot the error according to the user's manual of the CPU module.
-28414	9102H	■Target CPU abnormal start error A request was issued to the CPU module which is not operating normally.	Check the operating status of the target CPU and troubleshoot the error according to the user's manual of the CPU module.
-28415	9101H	■Target CPU major error A request was issued to the CPU module in which a major error occurred.	Check the operating status of the target CPU and troubleshoot the error according to the user's manual of the CPU module.
-28416	9100H	■Target CPU mounting error A request was issued by specifying the CPU number in the state where no CPU module is mounted.	• Check the mounting condition of the target CPU module. • Change the target CPU number specified in the user program.
-28624	9030H	■Function unsupported error • Any processing was performed to the module which does not support the station-based block data assurance function for cyclic data. • Any processing was performed to the module on which the station-based block data assurance function for cyclic data is not set. • An attempt was made to access a module which was not controlled by the host CPU module.	• Check if the target CC-Link module supports the station-based block data assurance function for cyclic data. • Check if the station-based block data assurance function for cyclic data is set for the target module. • Check if the control CPU of the target module is the host CPU module.
-28625	902FH	■Intelligent function module offline error An attempt was made to access the intelligent function module when it was offline.	Check the status of the intelligent function module and access the module while it is online.
-28626	902EH	■Control data setting value out of range error The specified control data is out of range.	Check the value set to the control data of the user program.
-28627	902DH	■Transient unsupported error Transient transmission cannot be performed via the specified communication route and target. (Another station was specified when the station number of the own station is '64' during CC-Link communication.)	• Check the communication route and target which support the transient request. • Change the station number of the own station.
-28628	902CH	■Pointer address specification error An incorrect address was specified to the argument pointer.	Check the address of the specified pointer.
-28629	902BH	■WDT not started error An attempt was made to reset a WDT before starting it.	Reset the WDT after starting it.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-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.
-28631	9029H	■Buffer access range error • The specified offset is out of range. • The specified offset and its size is out of range.	• Check the specified offset. • Check the specified buffer size. • Check the offset and its size.
-28632	9028H	■I/O number error • The specified I/O number is out of range. • No accessible module is mounted on the specified I/O number.	Check the specified I/O number.
-28634	9026H	■Intelligent function module down error The intelligent function module has an error.	• Check the mounting condition of the target CPU module. • Replace the intelligent function module or base unit.
-28635	9025H	■Intelligent function module error No intelligent function module is mounted on the accessed slot with the specified I/O number.	• Check the specified I/O number and the slot. • Check the mounting condition of the target CPU module.
-28636	9024H	■Control bus error The control bus to the intelligent function module has an error.	• Check if an error occurs in the bus master CPU (CPU No.1) in the multiple CPU system. • Check the mounting condition of the target CPU module. • Replace the intelligent function module or base unit.
-28638	9022H	■Multiple CPU unsupported operation error	Reset the bus master CPU (CPU No.1).
-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 operating status of the CPU module to RUN.
-28653	9013H	■I/O assignment error • An attempt was made to read the input value (X) from an output module. • An attempt was made to write the output value (Y) to an input module. • An attempt was made to read the output value (Y) from an input module.	Check the input number (X) and output number (Y).
-28660	900CH	■Access size error The specified size is out of range.	Check the specified offset and size.
-28661	900BH	■Inaccessible error Inaccessible area was specified.	Check the specified offset and size.
-28662	900AH	■CPU number specification error The CPU number is out of range or unavailable.	• Check the specified CPU number. • Check the operating status of the specified CPU module.
-28663	9009H	■Base unit number specification error The specified base unit number is out of range.	Check the specified base unit number.
-28664	9008H	■Data send area occupied	Retry.
-28665	9007H	■No registration data error	Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-28666	9006H	■Data length error	Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-28668	9004H	■Reply data stored error	Resend the request.
-28669	9003H	■Area number error The specified area number, offset address, or mode is out of range.	Check the specified area number, offset address, or mode.
-28671	9001H	■Module identification error	• Check the parameters. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-28672	9000H	■Processing code error	Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.

*1 When the access route specifies a wrong I/O number in the configuration of CC-Link, errors in the following range may occur. Check if the correct I/O number is specified.
· Specify other than network module: 4000H to 4FFFH
· Specify the serial communication module: 7000H to 7FFFH
· Specify the CC-Link IE Field Network module: D000H to DFFFH
· Specify the CC-Link IE Controller Network module: E000H to EFFFH
· Specify the MELSECNET/H module: F000H to FFFFH

4.2 C Intelligent Function Module Dedicated Functions

The following table shows the error codes of the C intelligent function 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).
-208	FF30H	■Offset error • The specified offset is out of range. • An AnS series module (buffer memory) has been 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.
-220	FF24H	■WDT type error The specified WDT type is out of range.	Check the specified WDT type.
-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.
-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.
-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 intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-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.
-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.
-239	FF11H	■Memory card mounting error Either of the following functions executed with no specified memory card inserted. • CITL_MountMemoryCard • CITL_UnmountMemoryCard	Check if a memory card is inserted.
-240	FF10H	■Clock data incorrect error The clock data to be set or the read clock data is incorrect.	• 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 • The specified cycle is out of range. • The cycle was set even when it had already been set.	• 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.	• Retry. • Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-257	FEFFH	■Interrupt event type specification error The value specified to the interrupt event type is out of range.	Check the specified value.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-258	FEFEH	■Output signal (Y) number specification error The value specified to the output signal (Y) number is out of range.	Check the specified value.
-259	FEFDH	■Interrupt service routine unregistered error The processing was not registered when enabling the processing corresponding to an event (interrupt).	Register the processing for the event (interrupt) and perform the operation again.
-260	FEFCH	■Memory card mount error ■Memory card unmount error The mount processing or unmount processing of the specified memory card failed.	• Retry. • Check if the specified memory card is damaged. • Replace the memory card.
-264	FEF8H	■Pointer error The address of the specified pointer is incorrect.	Check the address of the specified pointer.
-267	FEF5H	■Authentication error The specified password is incorrect.	Check the specified password.
-288	FEE0H	■Individual identification information read error Reading individual identification information failed.	• Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-292	FEDCH	■Standard ROM shutdown error The shutdown processing of the standard ROM failed.	• Check if files in the standard ROM are being accessed. • Check whether all files in the standard ROM have been closed.
-320	FEC0H	■Clock rate specification error The specified clock rate is out of range.	Check the specified clock rate.
-328	FEB8H	■Maximum setting size exceeded error The setting size of data sampling in each sequence scan is out of range.	Check the maximum setting size of data sampling in each sequence scan.
-329	FEB7H	■Setting data error The setting data of data sampling in each sequence scan is incorrect.	• Check the setting data of data sampling in each sequence scan. • Set target data within the range in which devices of the control CPU are assigned. • Review the device type. • Review the number of blocks. • Review the number of retainable records.
-330	FEB6H	■Data sampling setting error The data to be sampled was set during data sampling in each sequence scan.	Stop data sampling in each sequence scan, and set data to be sampled again.
-331	FEB5H	■Sampling setting unregistered error Acquisition of sampled data failed because data sampling in each sequence scan was not set.	Check if data sampling in each sequence scan is set.
-332	FEB4H	■Sampling setting unregistered error Starting sampling failed because data sampling in each sequence scan was not set.	Check if data sampling in each sequence scan is set.
-333	FEB3H	■Memory reservation error Reserving memory for the number of retainable records for data sampling in each sequence scan failed.	Check the number of retainable records.
-334	FEB2H	■Number of retainable records exceeded error The number of wait records exceeds the number of retainable records that is set in the data sampling in each sequence scan setting.	Check the number of wait records.
-335	FEB1H	■Function unsupported error The control CPU does not support the sequence scan synchronization sampling function.	Replace the control CPU with a CPU module supporting the sequence scan synchronization sampling function (high speed sampling).
-336	FEB0H	■Maximum number of settings error The total number of device points of each module using the sequence scan synchronization sampling function of the control CPU exceeds the maximum number of points.	When multiple modules use the sequence scan synchronization sampling function, review the setting so that the total number of sampling device points of each module is the maximum number of points or less of the control CPU.
-337	FEAFH	■Maximum number of settings error The number of modules using the sequence scan synchronization sampling function of the control CPU exceeds the maximum number.	Configure a system so that the number of modules using the sequence scan synchronization sampling function of the control CPU does not exceed the maximum number.
-338	FEAEH	■Sampling wait error A request was received in the state where the module could not wait for data sampling in each sequence scan.	Perform it again after starting data sampling in each sequence scan.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-339	FEADH	■Sampling wait suspension error Since data sampling in each sequence scan was stopped, the wait processing was interrupted.	When the processing to wait for sampling is required again, perform it after starting data sampling in each sequence scan.
-340	FEACH	■Sampling wait cancellation error Since the status of the control CPU was placed into the STOP/PAUSE state, the sampling wait state was canceled.	When the processing to wait for sampling is required again, perform it after starting data sampling in each sequence scan.
-341	FEABH	■CPU parameter change error Acquisition of data failed since data sampling was stopped due to the change of CPU parameters of the control CPU.	Set data sampling in each sequence scan again.
-342	FEAAH	■Time synchronization operating status setting error The value specified to the time synchronization operating status is out of range.	Check the value specified to the time synchronization operating status.
-343	FEA9H	■Sampling setting error • The specified device does not exist. • The specified device cannot be used for data sampling in each sequence scan.	Review the device.
-344	FEA8H	■Sampling setting error The specified device is out of range.	Set the device within the range.
-345	FEA7H	■Maximum number of points error The total number of device points of each module using the sequence scan synchronization sampling function of the control CPU exceeds the maximum number of points. (The number of device points is calculated by rounding up in 8192 points.)	• Review the number of target device points for the sequence scan synchronization sampling function. • Review the settings of each module using the sequence scan synchronization sampling function of the control CPU so that the total number of device points does not exceed the maximum number.
-346	FEA6H	■Sampling setting error The data sampling in each sequence scan setting failed.	• Restore the number of device points for data sampling in each sequence scan to the one before updating the setting. • Cycle the power of the system where the C intelligent function module is mounted or reset the CPU module. • Review the number of device points for data sampling in each sequence scan. • Set data sampling in each sequence scan again.

4.3 MELSEC iQ-R Series Data Link Functions

The following table shows the error codes of the MELSEC iQ-R series data link functions.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-1	FFFFH	■Path error • The specified path is unavailable. • The taskDelete was executed in the task using a MELSEC iQ-R series data link function. • The task using a MELSEC iQ-R series data link function was deleted with the taskDelete.	• Use a path pointer returned with the mdrOpen function. • Check if the taskDelete was executed in the task using a MELSEC iQ-R series data link function. • Check if the task using a MELSEC iQ-R series data link function was deleted with the taskDelete.
-2	FFFEH	■Device number error • The specified device number is out of range. • The specified bit device number is not a multiple of 8. • The device number and the points for the same block specified for reading/writing device randomly exceeds the device range.	• Check the start device number of the specified device. • Check the device number plus the number of points. • Specify the start device number of bit device in multiples of 8. • Check that the specified device is available in the CPU module on the target station.
-3	FFFDH	■Device type error The specified device type is unavailable.	• Correct the specified device type. • Check if the specified device is available in the target station.
-5	FFFBH	■Size error • The device number and the size exceeds the device range. • The device number and the size exceeds the range for the same block. • The access was made with an odd-number bytes. • The total of the points that are specified for each block number of the mdrRandR function or the mdrRandW function exceeds 10240.	• 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 mdrRandR function or the mdrRandW 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 mdrOpen function is unavailable.	Check the specified channel number.
-11	FFF5H	■Insufficient buffer area error The area size of the read data storage destination is smaller than the read data size.	Check the area size of the read data storage destination and the read data size.
-12	FFF4H	■Block number error The specified block number is unavailable.	• Check the block number (device type) of the specified device. • Check if the specified device and block number are available in the target station.
-13	FFF3H	■Write protect error The specified block number of the extension file register overlaps with the write protect area of the memory card.	• Check the block number (device type) of the extension file register. • Check the write protect switch of the memory card.
-16	FFF0H	■Station number/network number error • The specified station number or network number is out of range. • A device which is not accessible by the target station is specified.	• Check the specified station number and network number. • Check the devices which can be accessed by the target station.
-17	FFEFH	■All stations/group number specification error A function which does not support specifying all stations and group number was specified.	• Check if the function allows specifying all stations and group number. • When "All stations" or "Group number" is specified to the station number, specify "Without arrival confirmation" to the device type.
-18	FFEEH	■Remote specification error The specification code specified with the mdrControl function is unavailable.	Check the specified specification code.
-31	FFE1H	■Module load error Loading modules required for executing functions failed.	• The memory may be insufficient. End another running task or reduce the access size. • Check the status of the system drive of the C intelligent function module.
-32	FFE0H	■Resource timeout error The resource is being used by other tasks/threads and is not released within 30 seconds.	• Retry. • The memory may be insufficient. End another running task. • Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.

Error code		Description	Corrective action
Decimal	Hexadecimal		
-33	FFDFH	■Communication target unsupported error The module specified as the communication target by the network number and station number is not supported.	Check if the module specified as the communication target by a network number and station number is supported.
-36	FFDCH	■Registry write error Writing parameter files to the registry failed.	• Check if the standard ROM has already been shutdown. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-42	FFD6H	■Close error Communications cannot be closed.	• Retry. • Check if the C intelligent function module is running normally. • Reset the CPU module, or turn the power OFF and ON to reset the C intelligent function module.
-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 iQ-R series data link function service error MELSEC iQ-R series data link function service is disabled.	Enable the MELSEC iQ-R series data link function service with an engineering tool.
-53	FFCBH	■Timeout value error The specified timeout value is out of range.	Check the specified time out value.
-54	FFCAH	■I/O number error The specified I/O number is out of range.	Check the specified I/O number.
-55	FFC9H	■Logical station number error The specified logical station number is out of range.	Check the specified logical station number.
-56	FFC8H	■Target CPU error The specified target CPU is out of range.	Check the specified target CPU module.
-57	FFC7H	■Access route error The specified access route is out of range.	Check the specified access route.
-80	FFB0H	■Connection destination CPU error The connection destination CPU is not an RCPU.	Connect an RCPU.
-81	FFAFH	■Label code mismatch error Label assignment information of the CPU module was changed.	Acquire label information using the mdrGetLabelInfo function again.
-82	FFAEH	■Label incorrect value error An incorrect label name was specified. • Non-existent label name • A label name assigned to a device that 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.
-4096 to -1	F000H to FFFFH[*1]	📖 Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)	
-4097 to -8192	EFFFH to E000H[*1]	Refer to the following manual. 📖MELSEC iQ-R CC-Link IE Controller Network User's Manual (Application) 📖 MELSEC-Q CC-Link IE Controller Network Reference Manual	
-8193 to -12288	DFFFH to D000H[*1]	Refer to the following manual. 📖MELSEC iQ-R CC-Link IE Field Network User's Manual (Application) 📖 MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual 📖MELSEC-L CC-Link IE Field Network Master/Local Module User's Manual	
-16384 to -12289	C000H to CFFFH[*1]	Refer to the following manuals. 📖MELSEC iQ-R Ethernet User's Manual (Application)	
-16385 to -20480	BFFFH to B000H[*1]	Refer to the following manual. 📖MELSEC iQ-R CC-Link System Master/Local Module User's Manual (Application) 📖 MELSEC-Q CC-Link System Master/Local Module User's Manual 📖MELSEC-L CC-Link System Master/Local Module User's Manual	
28672 to 32767	7000H to 7FFFH[*1]	Refer to the following manuals. 📖MELSEC iQ-R Serial Communication Module User's Manual(Application)	

*1 When the access route specifies a wrong I/O number in the configuration of CC-Link, errors in the following range may occur. Check if the correct I/O number is specified.
· Specify other than network module: 4000H to 4FFFH
· Specify the serial communication module: 7000H to 7FFFH
· Specify the CC-Link IE Field Network module: D000H to DFFFH
· Specify the CC-Link IE Controller Network module: E000H to EFFFH
· Specify the MELSECNET/H module: F000H to FFFFH

MEMO

INDEX

FUNCTION INDEX

C

M

I

REVISIONS

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

Revision date	*Manual number	Description
December 2015	SH(NA)-081568ENG-A	First edition
January 2017	SH(NA)-081568ENG-B	■Added or modified parts Section 2.1, Section 3.1, Section 4.2
July 2020	SH(NA)-081568ENG-C	■Added or modified parts CONDITIONS OF USE FOR THE PRODUCT, INTRODUCTION, Section 1.3, Section 3.1, Section 3.2, Section 4.2

Japanese manual number: SH-081565-C

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.

TRADEMARKS

VxWorks and Wind River are either registered trademarks or trademarks of Wind River Systems, Inc.
Windows is either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
The company names, system names and product names mentioned in this manual are either registered trademarks or trademarks of their respective companies.
In some cases, trademark symbols such as '™' or '®' are not specified in this manual.

SH(NA)-081568ENG-C(2007)
MODEL:RD55UP06-V-P-E

MITSUBISHI ELECTRIC CORPORATION

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