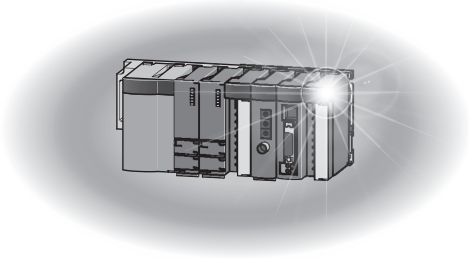
MITSUBISHI ELECTRIC

Programmable Controller

MELSEC Q series

QnUCPU User's Manual (Function Explanation, Program Fundamentals)

-Q00U(J)CPU
-Q01UCPU
-Q02UCPU
-Q03UDVCPU
-Q03UD(E)CPU
-Q04UDVCPU
-Q04UDPVCPU
-Q04UD(E)HCPU
-Q06UDVCPU
-Q06UDPVCPU
-Q06UD(E)HCPU
-Q10UD(E)HCPU
-Q13UDVCPU
-Q13UDPVCPU
-Q13UD(E)HCPU
-Q20UD(E)HCPU
-Q26UDVCPU
-Q26UDPVCPU
-Q26UD(E)HCPU
-Q50UDEHCPU
-Q100UDEHCPU



●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.

[Design Precautions]

⚠WARNING

● Configure safety circuits external to the programmable controller to ensure that the entire system operates safely even when a fault occurs in the external power supply or the programmable controller. Failure to do so may result in an accident due to an incorrect output or malfunction.

(1) Configure external safety circuits, such as an emergency stop circuit, protection circuit, and protective interlock circuit for forward/reverse operation or upper/lower limit positioning.

(2) The programmable controller stops its operation upon detection of the following status, and the output status of the system will be as shown below.

	Q/L series module	AnS/A series module
Overcurrent or overvoltage protection of the power supply module is activated.	All outputs are turned off	All outputs are turned off
The CPU module detects an error such as a watchdog timer error by the self-diagnostic function.	All outputs are held or turned off according to the parameter setting.	All outputs are turned off

All outputs may turn on when an error occurs in the part, such as I/O control part, where the CPU module cannot detect any error. To ensure safety operation in such a case, provide a safety mechanism or a fail-safe circuit external to the programmable controller. For a fail-safe circuit example, refer to General Safety Requirements in the QCPU User's Manual (Hardware Design, Maintenance and Inspection).

(3) Outputs may remain on or off due to a failure of an output module relay or transistor. Configure an external circuit for monitoring output signals that could cause a serious accident.

[Design Precautions]

⚠WARNING

- In an output module, when a load current exceeding the rated current or an overcurrent caused by a load short-circuit flows for a long time, it may cause smoke and fire. To prevent this, configure an external safety circuit, such as a fuse.
- Configure a circuit so that the programmable controller is turned on first and then the external power supply.
  If the external power supply is turned on first, an accident may occur due to an incorrect output or malfunction.
- For the operating status of each station after a communication failure, refer to relevant manuals for the network.
  Incorrect output or malfunction due to a communication failure may result in an accident.
- When changing data of the running programmable controller from a peripheral connected to the CPU module or from a personal computer connected to an intelligent function module, configure an interlock circuit in the sequence program to ensure that the entire system will always operate safely. For program modification and operating status change, read relevant manuals carefully and ensure the safety before operation.
  Especially, when a remote programmable controller is controlled by an external device, immediate action cannot be taken if a problem occurs in the programmable controller due to a communication failure.
  To prevent this, configure an interlock circuit in the sequence program, and determine corrective actions to be taken between the external device and CPU module in case of a communication failure.

[Design Precautions]

⚠CAUTION

- Do not install the control lines or communication cables together with the main circuit lines or power cables.
  Keep a distance of 100mm or more between them.
  Failure to do so may result in malfunction due to noise.
- When a device such as a lamp, heater, or solenoid valve is controlled through an output module, a large current (approximately ten times greater than normal) may flow when the output is turned from off to on.
  Take measures such as replacing the module with one having a sufficient current rating.
- After the CPU module is powered on or is reset, the time taken to enter the RUN status varies depending on the system configuration, parameter settings, and/or program size. Design circuits so that the entire system will always operate safely, regardless of the time.

[Installation Precautions]

⚠CAUTION

- Use the programmable controller in an environment that meets the general specifications in the QCPU User's Manual (Hardware Design, Maintenance and Inspection).
  Failure to do so may result in electric shock, fire, malfunction, or damage to or deterioration of the product.
- To mount the module, while pressing the module mounting lever in the lower part of the module, fully insert the module fixing projection(s) into the hole(s) in the base unit and press the module until it snaps into place.
  Incorrect mounting may cause malfunction, failure, or drop of the module.
  When using the programmable controller in an environment of frequent vibrations, fix the module with a screw.
  Tighten the screw within the specified torque range.
  Undertightening can cause drop of the screw, short circuit, or malfunction.
  Overtightening can damage the screw and/or module, resulting in drop, short circuit, or malfunction.
- When using an extension cable, connect it to the extension cable connector of the base unit securely.
  Check the connection for looseness.
  Poor contact may cause incorrect input or output.
- When using a memory card, fully insert it into the memory card slot.
  Check that it is inserted completely.
  Poor contact may cause malfunction.
- When using an SD memory card, fully insert it into the SD memory card slot.
  Check that it is inserted completely.
  Poor contact may cause malfunction.
- Securely insert an extended SRAM cassette into the cassette connector of a CPU module.
  After insertion, close the cassette cover to prevent the cassette from coming off.
  Failure to do so may cause malfunction.
- Shut off the external power supply (all phases) used in the system before mounting or removing a module. Failure to do so may result in damage to the product.
  A module can be replaced online (while power is on) on any MELSECNET/H remote I/O station or in the system where a CPU module supporting the online module change function is used.
  Note that there are restrictions on the modules that can be replaced online, and each module has its predetermined replacement procedure.
  For details, refer to the relevant sections in the QCPU User's Manual (Hardware Design, Maintenance and Inspection) and in the manual for the corresponding module.
- Do not directly touch any conductive parts and electronic components of the module, memory card, SD memory card, or extended SRAM cassette.
  Doing so can cause malfunction or failure of the module.
- When using a Motion CPU module and modules designed for motion control, check that the combinations of these modules are correct before applying power.
  The modules may be damaged if the combination is incorrect.
  For details, refer to the user's manual for the Motion CPU module.

[Wiring Precautions]

⚠WARNING

- Shut off the external power supply (all phases) used in the system before installation and wiring.
  Failure to do so may result in electric shock or damage to the product.
- After wiring, attach the included terminal cover to the module before turning it on for operation.
  Failure to do so may result in electric shock.

[Wiring Precautions]

⚠CAUTION

- Individually ground the FG and LG terminals of the programmable controller with a ground resistance of 100Ω or less.
  Failure to do so may result in electric shock or malfunction.
- Use applicable solderless terminals and tighten them within the specified torque range. If any spade solderless terminal is used, it may be disconnected when the terminal screw comes loose, resulting in failure.
- Check the rated voltage and terminal layout before wiring to the module, and connect the cables correctly.
  Connecting a power supply with a different voltage rating or incorrect wiring may cause a fire or failure.
- Securely connect the connector to the module. Failure to do so may cause malfunction.
- Connectors for external connection must be crimped or pressed with the tool specified by the manufacturer, or must be correctly soldered.
  Incomplete connections could result in short circuit, fire, or malfunction.
- Do not install the control lines or communication cables together with the main circuit lines or power cables.
  Keep a distance of 100mm or more between them.
  Failure to do so may result in malfunction due to noise.
- Place the cables in a duct or clamp them.
  If not, dangling cable may swing or inadvertently be pulled, resulting in damage to the module or cables or malfunction due to poor contact.
- Check the interface type and correctly connect the cable.
  Incorrect wiring (connecting the cable to an incorrect interface) may cause failure of the module and external device.
- Tighten the terminal screw within the specified torque range.
  Undertightening can cause short circuit, fire, or malfunction.
  Overtightening can damage the screw and/or module, resulting in drop, short circuit, or malfunction.
- Prevent foreign matter such as dust or wire chips from entering the module.
  Such foreign matter can cause a fire, failure, or malfunction.
- A protective film is attached to the top of the module to prevent foreign matter, such as wire chips, from entering the module during wiring.
  Do not remove the film during wiring.
  Remove it for heat dissipation before system operation.

[Wiring Precautions]

⚠CAUTION

- When disconnecting the cable from the module, do not pull the cable by the cable part.
  For the cable with connector, hold the connector part of the cable.
  For the cable connected to the terminal block, loosen the terminal screw.
  Pulling the cable connected to the module may result in malfunction or damage to the module or cable.
- Mitsubishi Electric programmable controllers must be installed in control panels.
  Connect the main power supply to the power supply module in the control panel through a relay terminal block.
  Wiring and replacement of a power supply module must be performed by maintenance personnel who is familiar with protection against electric shock. For wiring methods, refer to the QCPU User's Manual (Hardware Design, Maintenance and Inspection).

[Startup and Maintenance Precautions]

⚠WARNING

- Do not touch any terminal while power is on.
  Doing so will cause electric shock or malfunction.
- Correctly connect the battery connector.
  Do not charge, disassemble, heat, short-circuit, solder, or throw the battery into the fire. Also, do not expose it to liquid or strong shock.
  Doing so will cause the battery to produce heat, explode, ignite, or leak, resulting in injury and fire.
- Shut off the external power supply (all phases) used in the system before cleaning the module or retightening the terminal screws, connector screws, or module fixing screws.
  Failure to do so may result in electric shock or cause the module to fail or malfunction.

[Startup and Maintenance Precautions]

⚠CAUTION

- Before performing online operations (especially, program modification, forced output, and operation status change) for the running CPU module from the peripheral connected, read relevant manuals carefully and ensure the safety.
  Improper operation may damage machines or cause accidents.
- Do not disassemble or modify the modules.
  Doing so may cause failure, malfunction, injury, or a fire.
- Use any radio communication device such as a cellular phone or PHS (Personal Handy-phone System) more than 25cm away in all directions from the programmable controller.
  Failure to do so may cause malfunction.

[Startup and Maintenance Precautions]

⚠CAUTION

- Shut off the external power supply (all phases) used in the system before mounting or removing a module. Failure to do so may cause the module to fail or malfunction.
  A module can be replaced online (while power is on) on any MELSECNET/H remote I/O station or in the system where a CPU module supporting the online module change function is used.
  Note that there are restrictions on the modules that can be replaced online, and each module has its predetermined replacement procedure.
  For details, refer to the relevant sections in the QCPU User's Manual (Hardware Design, Maintenance and Inspection) and in the manual for the corresponding module.
- After the first use of the product, do not perform each of the following operations more than 50 times (IEC 61131-2/JIS B 3502 compliant).
  Exceeding the limit may cause malfunction.
  - Mounting/removing the module to/from the base unit
  - Inserting/removing the extended SRAM cassette to/from the CPU module
  - Mounting/removing the terminal block to/from the module
- After the first use of the product, do not insert/remove the SD memory card to/from the CPU module more than 500 times. Exceeding the limit may cause malfunction.
- Do not drop or apply shock to the battery to be installed in the module.
  Doing so may damage the battery, causing the battery fluid to leak inside the battery.
  If the battery is dropped or any shock is applied to it, dispose of it without using.
- Before handling the module, touch a grounded metal object to discharge the static electricity from the human body.
  Failure to do so may cause the module to fail or malfunction.

[Disposal Precautions]

⚠CAUTION

- When disposing of this product, treat it as industrial waste.
  When disposing of batteries, separate them from other wastes according to the local regulations.
  (For details of the battery directive in EU member states, refer to the QCPU User's Manual (Hardware Design, Maintenance and Inspection).)

[Transportation Precautions]

⚠CAUTION

- When transporting lithium batteries, follow the transportation regulations.
  (For details of the regulated models, refer to the QCPU User's Manual (Hardware Design, Maintenance and Inspection).)

●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.

INTRODUCTION

This manual, "QnUCPU User's Manual (Function Explanation, Program Fundamentals)" describes the memory maps, functions, programs, I/O number assignment, and devices of the Universal model QCPU.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the Q series programmable controller to handle the product correctly.
When applying the program examples introduced in this manual to the actual system, ensure the applicability and confirm that it will not cause system control problems.

■ Relevant CPU module

CPU module	Model
Universal model QCPU	Q00U(J)CPU, Q01UCPU, Q02UCPU, Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU, Q50UDEHCPU, Q100UDEHCPU

Remark

This manual does not describe the specifications of the power supply modules, base units, extension cables, memory cards, SD memory cards, extended SRAM cassettes, batteries as well as the lists of error codes, special relay, and special register. For details, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

For multiple CPU systems, refer to the following.

QCPU User's Manual (Multiple CPU System)

Memo

CONTENTS

PART 2 FUNCTIONS

PART 3 DEVICES, CONSTANTS

CHAPTER 4 DEVICES 336

MANUALS

To understand the main specifications, functions, and usage of the CPU module, refer to the basic manuals. Read other manuals as well when using a different type of CPU module and its functions. Order each manual as needed, referring to the following list.

● :Basic manual, ○ :Other CPU module manuals/Use them to utilize functions.

(1) CPU module user's manual

Manual name <manual number (model code)>	Description	Manual type
QCPU User's Manual (Hardware Design, Maintenance and Inspection) <SH-080483ENG (13JR73)>	Specifications of the hardware (CPU modules, power supply modules, base units, extension cables, memory cards, SD memory cards, extended SRAM cassettes, and batteries), system maintenance and inspection, troubleshooting, and error codes	●
QCPU User's Manual (Multiple CPU System) <SH-080485ENG (13JR75)>	Information on building multiple CPU systems (system configurations, I/O numbers, communications between CPU modules, and communications with I/O modules and intelligent function modules)	●
QnUCPU User's Manual (Communication via Built-in Ethernet Port) <SH-080811ENG (13JZ29)>	Detailed description of communication via the built-in Ethernet ports of the CPU module	○
QnUDVCPU/LCPU User's Manual (Data Logging Function) <SH-080893ENG (13JZ39)>	Detailed description of the data logging function of the CPU module	○

(2) Programming manual

Manual name <manual number (model code)>	Description	Manual type
QCPU/LCPU Programming Manual (Common Instruction) <SH-080809ENG (13JW10)>	Detailed description and usage of instructions used in programs	●
MELSEC-Q/L/QnA Programming Manual (SFC) <SH-080041 (13JF60)>	System configuration, specifications, functions, programming, and error codes for SFC (MELSAP3) programs	○
MELSEC-Q/L Programming Manual (MELSAP-L) <SH-080076 (13JF61)>	System configuration, specifications, functions, programming, and error codes for SFC (MELSAP-L) programs	○
MELSEC-Q/L Programming Manual (Structured Text) <SH-080366E (13JF68)>	System configuration and programming using structured text language	○
MELSEC-Q/L/QnA Programming Manual (PID Control Instructions) <SH-080040 (13JF59)>	Dedicated instructions for PID control	○
MELSEC-Q Programming/Structured Programming Manual (Process Control Instructions) <SH-080316E (13JF67)>	Dedicated instructions for process control	○

(3) Operating manual

Manual name <manual number (model code)>	Description	Manual type
GX Works2 Version1 Operating Manual (Common) <SH-080779ENG (13JU63)>	System configuration, parameter settings, and online operations of GX Works2, which are common to Simple projects and Structured projects	●
GX Developer Version 8 Operating Manual <SH-080373E (13JU41)>	Operating methods of GX Developer, such as programming, printing, monitoring, and debugging	○

(4) I/O module and intelligent function module manual

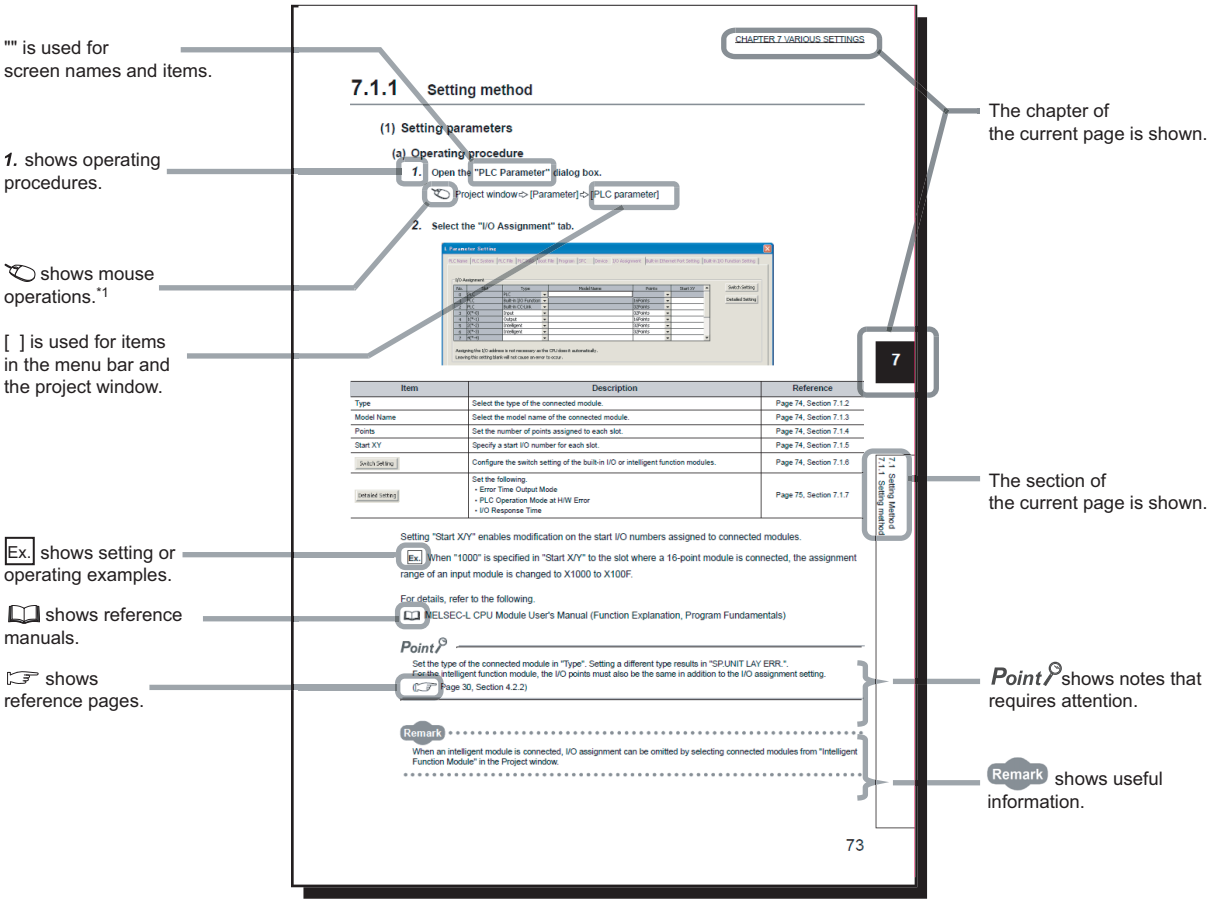
Manual name <manual number (model code)>	Description	Manual type
MELSEC-Q CC-Link IE Controller Network Reference Manual <SH-080668ENG (13JV16)>	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of the CC-Link IE Controller Network module	○
MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual <SH-080917ENG (13JZ47)>	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of the CC-Link IE Field Network module	○
Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network) <SH-080049 (13JF92)>	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of a MELSECNET/H network system (PLC to PLC network)	○
Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network) <SH-080124 (13JF96)>	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of a MELSECNET/H network system (remote I/O network)	○
Q Corresponding Ethernet Interface Module User's Manual (Basic) <SH-080009 (13JL88)>	Specifications, procedures for data communication with external devices, line connection (open/close), fixed buffer communication, random access buffer communication, and troubleshooting of the Ethernet module	○
MELSEC-Q/L Ethernet Interface Module User's Manual (Application) <SH-080010 (13JL89)>	E-mail function, programmable controller CPU status monitoring function, communication via CC-Link IE Field Network, CC-Link IE Controller Network, MELSECNET/H, or MELSECNET/10, communication using the data link instructions, and file transfer function (FTP server) of the Ethernet module	○
MELSEC-Q CC-Link System Master/Local Module User's Manual <SH-080394E (13JR64)>	System configuration, performance specifications, functions, handling, wiring, and troubleshooting of the QJ61BT11N	○
Q Corresponding Serial Communication Module User's Manual (Basic) <SH-080006 (13JL86)>	Overview, system configuration, specifications, procedures before operation, basic data communication method with external devices, maintenance and inspection, and troubleshooting for using the serial communication module	○
MELSEC-Q/L Serial Communication Module User's Manual (Application) <SH-080007 (13JL87)>	Special functions (specifications, usage, and settings) and data communication method with external devices of the serial communication module	○

(5) Others

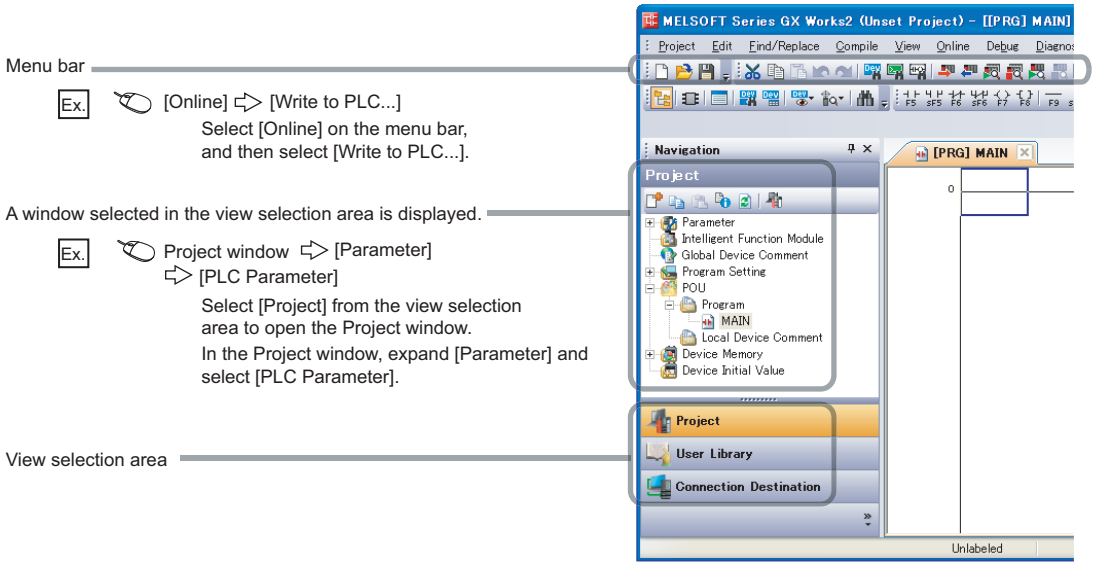
Manual name <manual number (model code)>	Description	Manual type
MELSEC Communication Protocol Reference Manual <SH-080008 (13JF89)>	Communication method using the MC protocol, which reads/writes data to/from the CPU module via the serial communication module or Ethernet module	○
iQ Sensor Solution Reference Manual <SH-081133ENG (13JV28)>	Operating methods of iQ Sensor Solution, such as programming and monitoring	○
CC-Link IE Field Network Basic Reference Manual <SH-081684ENG (13JX62)>	Specifications, procedures before operation, system configuration, programming, functions, parameter settings, and troubleshooting of CC-Link IE Field Network Basic	○

MANUAL PAGE ORGANIZATION

In this manual, pages are organized and the symbols are used as shown below. The following page illustration is for explanation purpose only, and is different from the actual pages.



*1 The mouse operation example is provided below. (For GX Works2)



Icon Universal model QCPU	Description
Universal	Icons indicate that specifications described on the page contain some precautions.

TERMS

Unless otherwise specified, this manual uses the following generic terms and abbreviations.
*☐ indicates a part of the model or version.
(Example): Q33B, Q35B, Q38B, Q312B → Q3☐B

Generic term/abbreviation	Description
■ CPU module type	
CPU module	Generic term for the Universal model QCPU
High Performance model QCPU	Generic term for the Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
Process CPU	Generic term for the Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
Universal model QCPU	Generic term for the Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q03UDVCPU, Q03UDECPU, Q04UDHCPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU
Built-in Ethernet port QCPU	Generic term for the Q03UDVCPU, Q03UDECPU, Q04UDVCPU, Q04UDPVCPU, Q04UDEHCPU, Q06UDVCPU, Q06UDPVCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDVCPU, Q13UDPVCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDVCPU, Q26UDPVCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU
High-speed Universal model QCPU	Generic term for the Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU, and Q26UDVCPU
Universal model Process CPU	Generic term for the Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, and Q26UDPVCPU
Motion CPU	Generic term for the Mitsubishi Electric motion controllers: Q172CPUN, Q173CPUN, Q172HCPU, Q173HCPU, Q172CPUN-T, Q173CPUN-T, Q172HCPU-T, Q173HCPU-T, Q172DCPU, Q173DCPU, Q172DCPU-S1, Q173DCPU-S1, Q172DSCPU, and Q173DSCPU
PC CPU module	Abbreviation for the MELSEC-Q series-compatible PC CPU module manufactured by CONTEC Co., Ltd., PPC-CPU852(MS)-512
C Controller module	Generic term for the C Controller modules: Q06CCPU-V, Q06CCPU-V-B, Q12DCCPU-V, Q24DHCCPU-V, and Q24DHCCPU-LS
■ CPU module model	
Qn(H)CPU	Generic term for the Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
QnPHCPU	Generic term for the Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
QnU(D)(H)CPU	Generic term for the Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, and Q26UDHCPU
QnUD(H)CPU	Generic term for the Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, and Q26UDHCPU
QnUDVCPU	Generic term for the Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU, and Q26UDVCPU
QnUDPVCPU	Generic term for the Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, and Q26UDPVCPU
QnUDE(H)CPU	Generic term for the Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU
QnUD(E)(H)CPU	Generic term for the Q03UDCPU, Q03UDECPU, Q04UDHCPU, Q04UDEHCPU, Q06UDHCPU, Q06UDEHCPU, Q10UDHCPU, Q10UDEHCPU, Q13UDHCPU, Q13UDEHCPU, Q20UDHCPU, Q20UDEHCPU, Q26UDHCPU, Q26UDEHCPU, Q50UDEHCPU, and Q100UDEHCPU

Generic term/abbreviation	Description
■ Base unit type	
Base unit	Generic term for the main base unit, extension base unit, slim type main base unit, redundant power main base unit, redundant power extension base unit, and multiple CPU high speed main base unit
Main base unit	Generic term for the Q3□B, Q3□SB, Q3□RB, and Q3□DB
Extension base unit	Generic term for the Q5□B, Q6□B, Q6□RB, QA1S5□B, QA1S6□B, QA1S6ADP+A1S5□B/A1S6□B, QA6□B, and QA6ADP+A5□B/A6□B
Slim type main base unit	Another name for the Q3□SB
Redundant power main base unit	Another name for the Q3□RB
Redundant power extension base unit	Another name for the Q6□RB
Multiple CPU high speed main base unit	Another name for the Q3□DB
■ Base unit model	
Q3□B	Generic term for the Q33B, Q35B, Q38B, and Q312B main base units
Q3□SB	Generic term for the Q32SB, Q33SB, and Q35SB slim type main base units
Q3□RB	Another name for the Q38RB main base unit for redundant power supply system
Q3□DB	Generic term for the Q35DB , Q38DB and Q312DB multiple CPU high speed main base units
Q5□B	Generic term for the Q52B and Q55B extension base units
Q6□B	Generic term for the Q63B, Q65B, Q68B, and Q612B extension base units
Q6□RB	Another name for the Q68RB extension base unit for redundant power supply system
QA1S5□B	Another name for the QA1S51B extension base unit
QA1S6□B	Generic term for the QA1S65B and QA1S68B extension base units
QA6□B	Generic term for the QA65B and QA68B extension base units
A5□B	Generic term for the A52B, A55B, and A58B extension base units
A6□B	Generic term for the A62B, A65B, and A68B extension base units
QA6ADP+A5□B/A6□B	Abbreviation for A large type extension base unit where the QA6ADP is mounted
QA1S6ADP+A1S5□B/A1S6□B	Abbreviation for A small type extension base unit where the QA1S6ADP is mounted
■ Power supply module	
Power supply module	Generic term for the Q series power supply module, slim type power supply module, and redundant power supply module
Q series power supply module	Generic term for the Q61P-A1, Q61P-A2, Q61P, Q61P-D, Q62P, Q63P, Q64P, and Q64PN power supply modules
Slim type power supply module	Abbreviation for the Q61SP slim type power supply module
Redundant power supply module	Generic term for the Q63RP and Q64RP power supply modules for redundant power supply system
■ Network module	
CC-Link IE module	Generic term for the CC-Link IE Controller Network module and CC-Link IE Field Network module
MELSECNET/H module	Abbreviation for the MELSECNET/H network module
Ethernet module	Abbreviation for the Ethernet interface module
CC-Link module	Abbreviation for the CC-Link system master/local module
■ Network	
CC-Link IE	Generic term for the CC-Link IE Controller Network and CC-Link IE Field Network
MELSECNET/H	Abbreviation for the MELSECNET/H network system
■ Memory extension	
Memory card	Generic term for the SRAM card, Flash card, and ATA card
SRAM card	Generic term for the Q2MEM-1MBSN, Q2MEM-1MBS, Q2MEM-2MBSN, Q2MEM-2MBS, Q3MEM-4MBS, and Q3MEM-8MBS SRAM cards

Generic term/abbreviation	Description
Flash card	Generic term for the Q2MEM-2MBF and Q2MEM-4MBF Flash cards
ATA card	Generic term for the Q2MEM-8MBA, Q2MEM-16MBA, and Q2MEM-32MBA ATA cards
SD memory card	Generic term for the NZ1MEM-2GBSD, NZ1MEM-4GBSD, NZ1MEM-8GBSD, NZ1MEM-16GBSD, L1MEM-2GBSD, and L1MEM-4GBSD Secure Digital memory cards. This is a non-volatile memory card.
Extended SRAM cassette	Generic term for the Q4MCA-1MBS, Q4MCA-2MBS, Q4MCA-4MBS, and Q4MCA-8MBS extended SRAM cassettes
■ Software package	
Programming tool	Generic term for GX Works2 and GX Developer
GX Works2	Product name for MELSEC programmable controller software package
GX Developer	
■ Others	
MC protocol	Abbreviation for the MELSEC communication protocol. The MELSEC communication protocol is a communication method to access from an external device to the CPU module according to the communication procedure for the Q series programmable controller (such as a serial communication module, Ethernet module).
QA6ADP	Abbreviation for the QA6ADP QA conversion adapter module
QA1S6ADP	Generic term for the QA1S6ADP and QA1S6ADP-S1 Q-AnS base unit conversion adapters
Extension cable	Generic term for the QC05B, QC06B, QC12B, QC30B, QC50B, and QC100B extension cables
Battery	Generic term for the Q6BAT, Q7BATN, Q7BAT, and Q8BAT CPU module batteries, Q2MEM-BAT SRAM card battery, and Q3MEM-BAT SRAM card battery
GOT	Generic term for Mitsubishi Electric Graphic Operation Terminal, GOT-A*** series, GOT-F*** series, GOT1000 series, and GOT2000 series

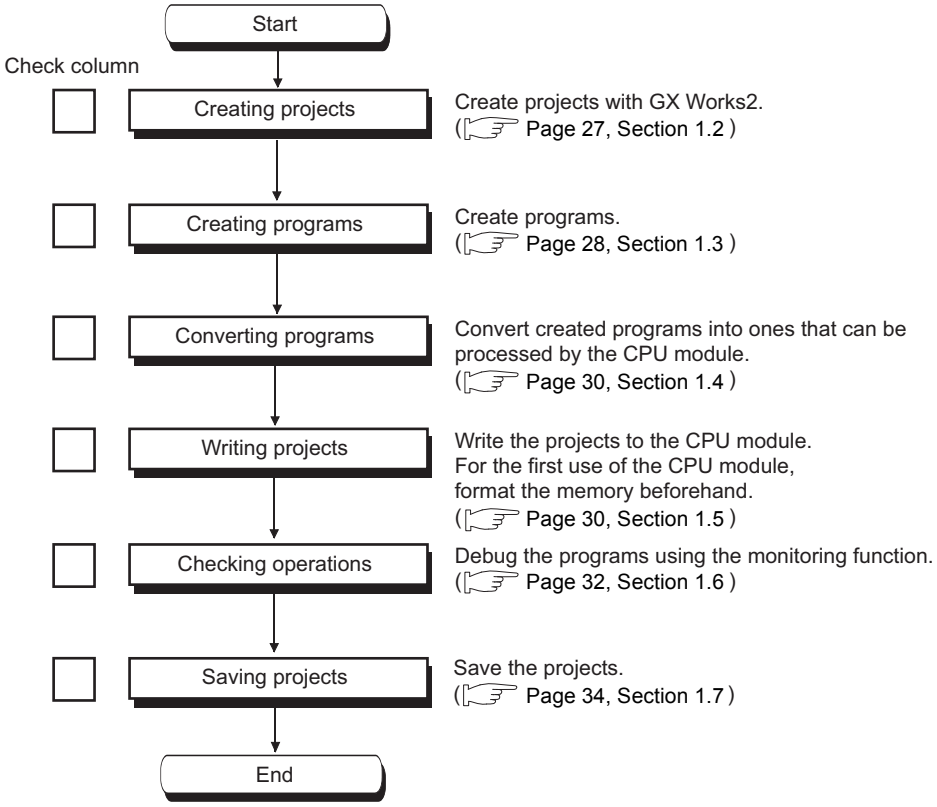
Memo

PART 1 PROGRAMMING

In this part, fundamental knowledge of programming is described.

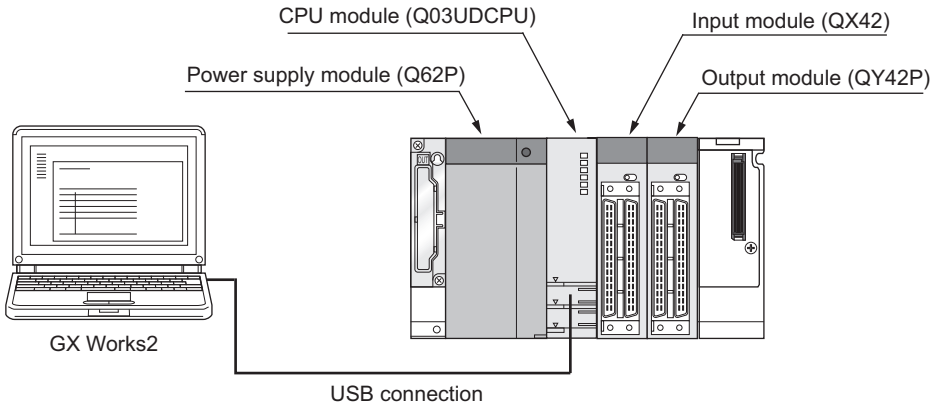
CHAPTER 1 BASIC PROCEDURE FOR PROGRAMMING

This chapter describes the basic procedure for programming.



1.1 System Configuration Example

The following system configuration is used for description throughout this chapter.



* Wiring of the power supply module and I/O modules are omitted in this illustration.

1.2 Creating a Project

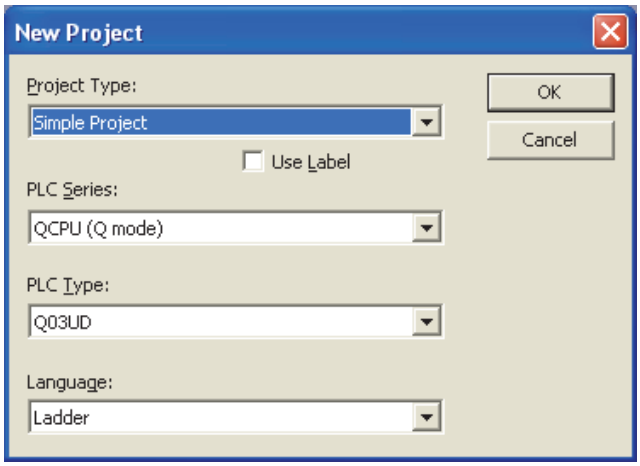
A project is a set of information, such as programs and parameters, which is necessary to operate a programmable controller.

The following two projects are available.

- Simple project
- Structured project

Create a new project using GX Works2.

[Project] ⇨ [New...]



Item	Description
Project Type	Select a type of project to create. In this chapter, "Simple Project" is selected.
Use Label	Select this checkbox when using a label for programming. In this chapter, this is not selected.
PLC Series	Select a series of the CPU module to use in the project. In this chapter, "QCPU (Q mode)" is selected.
PLC Type	Select a type of the CPU module (CPU module model) to use in the project. In this chapter, "Q03UD" is selected.
Language	Select a language of the program data to use for the new project. In this chapter, "Ladder" is selected.

Point

When perform communication between a programming tool and a CPU module through GOT or a network module, check the PLC type because the modules could be connected with incorrect model names. If the modules are connected with incorrect model names, data may not be written or read properly.

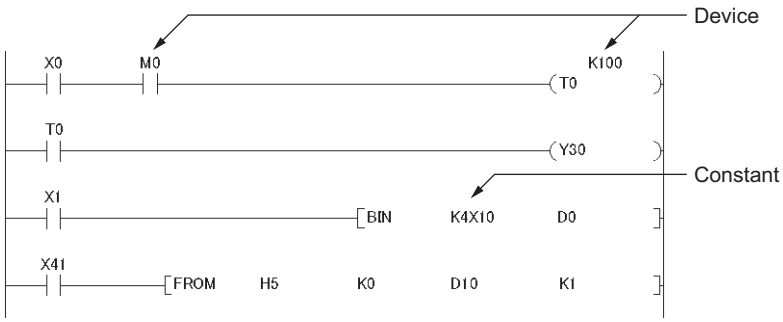
1.3 Creating a Program

1.3.1 Prior knowledge for creating a program

(1) Device and constants

Devices and constants, such as shown below, are used for creating a program.

(☞ Page 336, CHAPTER 4)



(2) Concept of I/O numbers

I/O numbers are automatically assigned.

Power supply module	CPU module	Input module	Output module	Empty
		64 points	64 points	
		X0000 to X003F	Y0040 to Y007F	

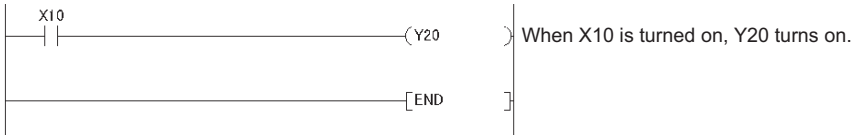
Users can also assign I/O numbers according to their purposes. (☞ Page 52, Section 2.2)

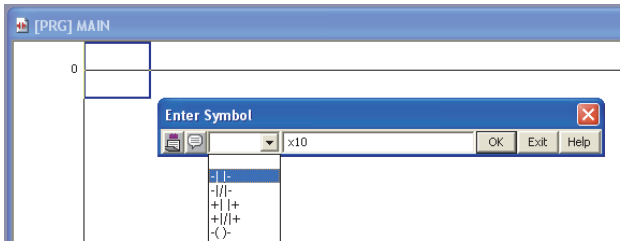
(3) Program configuration

A main routine program, subroutine program, (☞ Page 69, Section 2.4.3), and interrupt program (☞ Page 82, Section 2.9) can be included in a program.

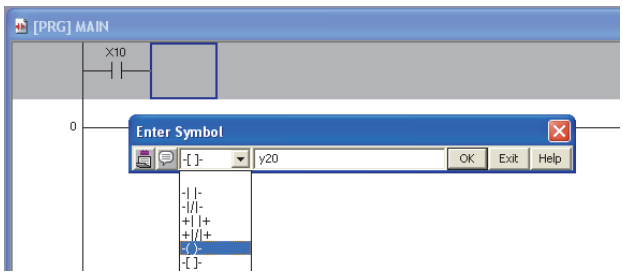
1.3.2 How to create a program

This section shows how to create the following sample program.

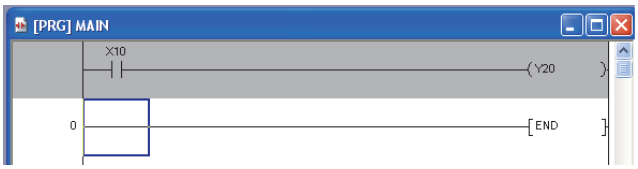




1. **To enter X10, type X10 at the original cursor position and select the contact shown in the left figure.**



2. **To enter Y20, type Y20 and select the coil shown in the left figure.**



The program has been created. In the next procedure, convert the program.

1.4 Converting a Program

Operation of a program is defined after converting its ladder.

[Compile] ⇨ [Build]

The program has been converted. In the next procedure, write the program to a CPU module.

Point

- To use a label, the program must be compiled.
  GX Works2 Version 1 Operating Manual (Common)
- After modifying a program, it must be compiled.

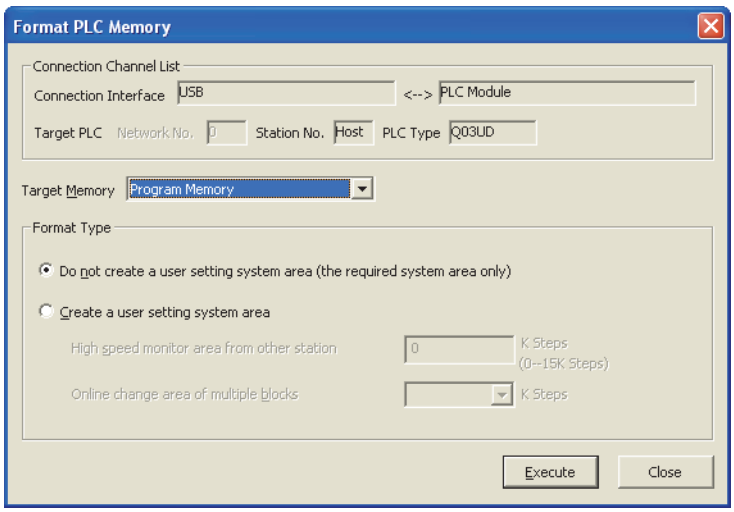
1.5 Writing a Project to the CPU Module

Write a project to the CPU module. Note that, if the project is new, the memory (☞ Page 35, Section 2.1.1) needs to be formatted first.

1.5.1 Formatting a memory

To format a memory, open the "Format PLC Memory" dialog box. In this chapter, a program memory is formatted so that a program can be written to it.

[Online] ⇨ [PLC Memory Operation] ⇨ [Format PLC Memory...]



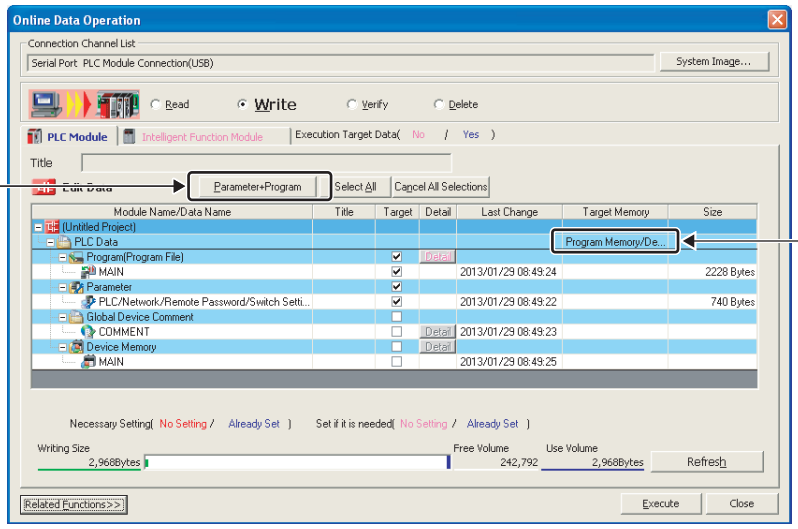
To check the capacity of the memory after formatting, open the "Online Data Operation" dialog box.

1.5.2 Writing to the CPU module

Open the "Online Data Operation" dialog box. In this chapter, a project is written to the program memory.

[Online] ⇨ [Write to PLC...]

2) Selecting this will automatically select the parameter and program checkboxes.



1) Select the program memory.

The project has been written. In the next procedure, execute the program.

Point

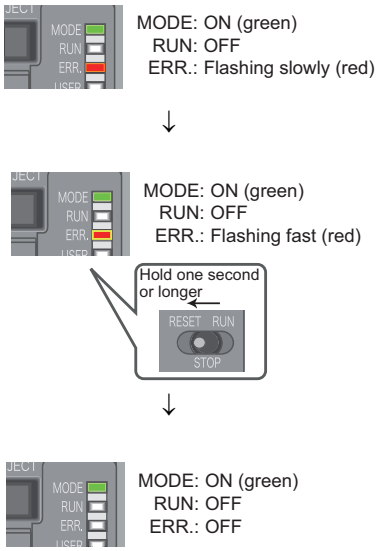
Note that parameter setting is required to operate CPU modules. In this chapter, the procedure for parameter setting is not introduced since default values are used. (Page 431, Appendix 1)

1.6 Checking an Operation of the CPU Module

To check an operation, execute the program written to the CPU module. In this chapter, operation is checked through the monitoring screen of GX Works2.

(1) Executing a program

Before operating the CPU module, data written to the CPU module must be validated. To validate, power off and then on or reset the CPU module.

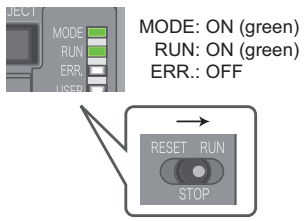


1. **Before resetting the CPU module, check the current LED status.**

2. **Move the switch on the front of the CPU module to the RESET position. (One second or longer)**

3. **Hold the switch until the ERR. LED turns off after flashing.**

In the next procedure, run the CPU module. To run, use the switch on the CPU module.



4. **Move the switch to the RUN position.**

When the RUN LED is lit green, the program is being executed successfully.

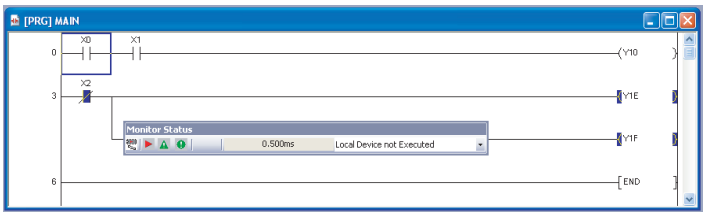
Point

By remote operation, CPU modules can be operated without using switches. (Page 131, Section 3.6)

(2) Checking operation

Conductivity and power distribution status of contacts and coils can be checked by switching GX Works2 to the monitor mode.

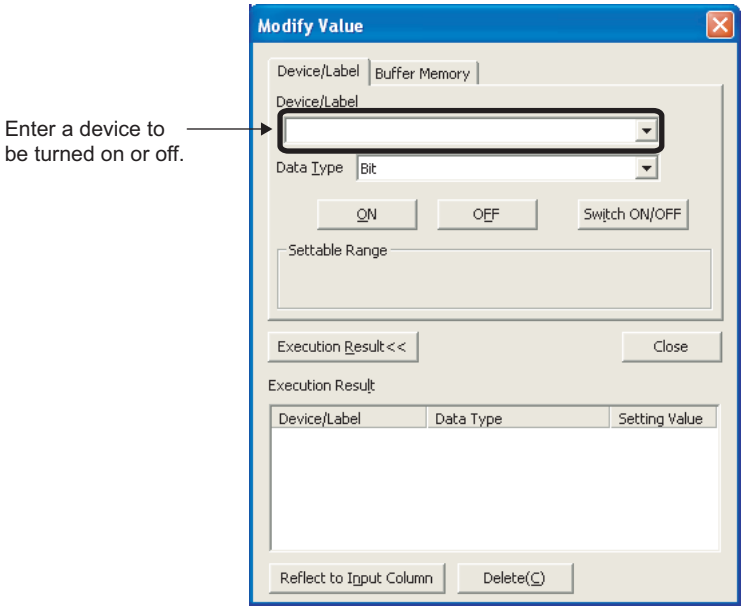
[Online] ⇨ [Monitor] ⇨ [Start Monitoring]



When X0 and X1 are turned on, Y10 turns on. (to turn on X0 and X1, place the cursor on them and double-click while holding the Shift key.) While contacts and coils are conducting, they are shown in blue.

Debug can be performed by forcibly turn on or off devices in the "Modify Value" dialog box.

[Debug] ⇨ [Modify Value...]



For details on current value changing, refer to the following.

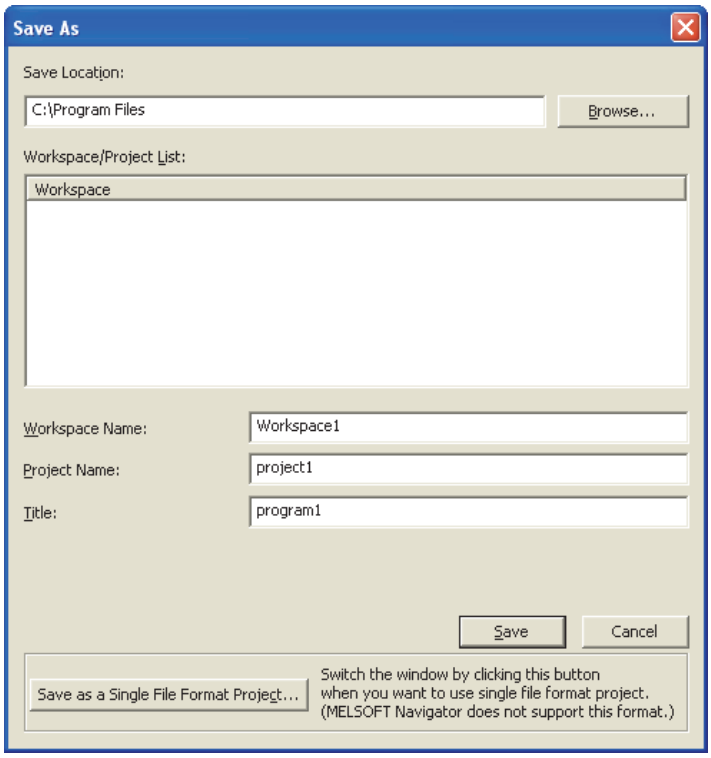
GX Works2 Version 1 Operating Manual (Common)

If a program is edited during debugging, the program can be written to the CPU module even while the CPU module is in the RUN status. (Page 168, Section 3.12)

1.7 Saving a Project

To save a project, open the "Save As" dialog box.

[Project] ⇨ [Save As...]



Item	Description
Save Location	Enter the storage destination folder (drive or path) of the workspace. Folders can be browsed for selection by clicking the Browse... button.
Workspace/Project List	Select a workspace. Double-click "Workspace" to display a project list.
Workspace Name	Enter a name for the workspace.
Project Name	Enter a name for the project.
Title*1	Enter a title for the project.

*1 Projects can also be saved without titles.

CHAPTER 2 APPLICATION OF PROGRAMMING

2.1 Memory and Files

2.1.1 Memory configuration and storable data

Memory configuration differs depending on the CPU module (refer to the following).

CPU module	Memory configuration
Q00UJCPU	Program memory, standard ROM
Q00UCPU, Q01UCPU	Program memory, standard RAM, standard ROM
Q02UCPU, QnUD(H)CPU, QnUDE(H)CPU	Program memory, standard RAM, standard ROM, memory card (SRAM card, Flash card, or ATA card)
QnUDVCPU, QnUDPVCPU	Program memory, standard RAM, standard ROM, SD memory card

(1) Program memory

This memory stores programs and parameters required in processing of the CPU module.

(a) Processing a program

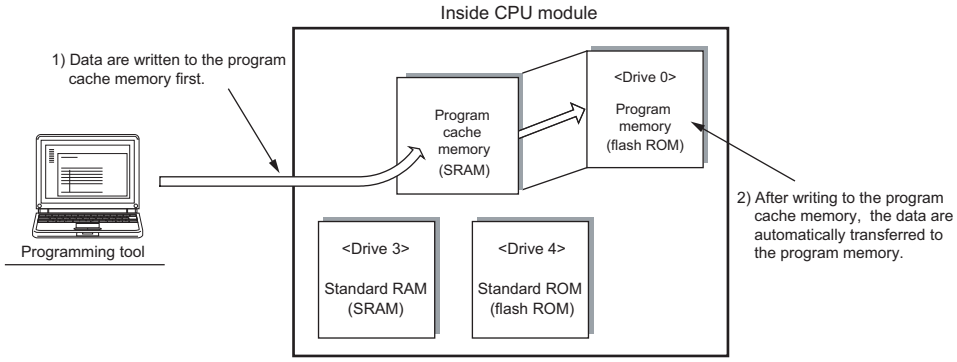
When a program is executed, data in the program memory are transferred to the program cache memory[*1] at the following timings.

- Initial processing at power-on
- Initial processing at reset

*1 The program cache memory is used for program operations.

(b) Writing to the program memory

When a program is written to the program memory, it is temporarily written to the program cache memory, and then automatically transferred back to the program memory.



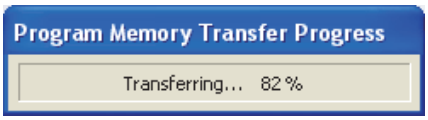
Point

While the CPU module is in the RUN status, automatic data transfer to the program memory can be disabled by setting. (Page 173, Section 3.12.3)

(c) Transfer confirmation to the program memory

Program transfer to the program memory can be checked by the following.

- Checking the status in the progress screen
  The following figure is the progress screen in a programming tool.



- Checking with the special relay and the special register
  The status can be checked using SM681 and SD681.

(d) Checking whether data are transferred to the program memory Note 2.1

Whether data are transferred from the program cache memory to the program memory can be checked using SM165.

(2) Standard RAM

This memory stores file register files, local device files, sampling trace files, and module error history files.
For the High-speed Universal model QCPU and Universal model Process CPU, the size of standard RAM is extended by installing an extended SRAM cassette.

(3) Standard ROM

This memory stores data such as device comments and PLC user data.

Note 2.1 Universal

When checking the data transfer status with the Q02UCPU, Q03UDCPU, Q04UDHCPU, or Q06UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

(4) Memory card

This memory is used to extend memory in a CPU module. Three types of memory cards are applicable.

- SRAM card
- Flash card
- ATA card

(a) SRAM card

Data can be read from or written to a file register file stored in an SRAM card by sequence programs.
This card is used when:

- the number of file register points is greater than the standard RAM capacity, or
- the sampling trace function is used. (☞ Page 184, Section 3.14)

When storing file registers to the SRAM card, the file registers can be written or read by the sequence program up to 4086K points.

(b) Flash card

This card is used when changing data is not required. Data are written to the card using a programming tool and read from the card by sequence programs. (Data cannot be written to the card by sequence programs.) Up to 2039K points of file register data can be stored.

(c) ATA card

This card is used for PLC user data (general-purpose data).
With the file access instruction (such as the SP. FWRITE instruction) in the sequence program, access the PLC user data in the ATA card in CSV format/binary format.

(5) SD memory card

This memory stores programs and parameters. To execute programs stored in an SD memory card, perform boot operation. (☞Page 104, Section 2.11) To execute the data logging function, this card must be inserted.

(6) Memory capacities and necessity of formatting

The following tables list the memory capacities and necessity of formatting of each memory.
Format a memory that requires formatting using a programming tool before use.

Memory device		Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU	Q03UD/Q03UDECPU	Q04UDH/Q04UDEHCPU	Q06UDH/Q06UDEHCPU	Formatting
Program memory		40K bytes (10K steps)	40K bytes (10K steps)	60K bytes (15K steps)	80K bytes (20K steps)	120K bytes (30K steps)	160K bytes (40K steps)	240K bytes (60K steps)	Necessary
Standard ROM		256K bytes	512K bytes	512K bytes	512K bytes	1024K bytes	1024K bytes	1024K bytes	Unnecessary
Standard RAM		-	128K bytes	128K bytes	128K bytes	192K bytes	256K bytes	768K bytes	Necessary[*1]
Memory card	SRAM card	-			Q2MEM-1MBSN, Q2MEM-1MBS: 1M byte Q2MEM-2MBSN, Q2MEM-2MBS: 2M bytes Q3MEM-4MBS: 4M bytes Q3MEM-8MBS: 8M bytes				Necessary
	Flash card	-			Q2MEM-2MBF: 2M bytes Q2MEM-4MBF: 4M bytes				Unnecessary
	ATA card	-			Q2MEM-8MBA: 8M bytes Q2MEM-16MBA: 16M bytes Q2MEM-32MBA: 32M bytes				Necessary

Memory device		Q10UDH/Q10UDEHCPU	Q13UDH/Q13UDEHCPU	Q20UDH/Q20UDEHCPU	Q26UDH/Q26UDEHCPU	Q50UDEHCPU	Q100UDEHCPU	Formatting
Program memory		400K bytes (100K steps)	520K bytes (130K steps)	800K bytes (200K steps)	1040K bytes (260K steps)	2000K bytes (500K steps)	4000K bytes (1000K steps)	Necessary
Standard ROM		2048K bytes	2048K bytes	4096K bytes	4096K bytes	8192K bytes	16384K bytes	Unnecessary
Standard RAM		1024K bytes	1024K bytes	1280K bytes	1280K bytes	1536K bytes	1792K bytes	Necessary[*1]
Memory card	SRAM card	Q2MEM-1MBSN, Q2MEM-1MBS: 1M byte Q2MEM-2MBSN, Q2MEM-2MBS: 2M bytes Q3MEM-4MBS: 4M bytes Q3MEM-8MBS: 8M bytes						Necessary
	Flash card	Q2MEM-2MBF: 2M bytes Q2MEM-4MBF: 4M bytes						Unnecessary
	ATA card	Q2MEM-8MBA: 8M bytes Q2MEM-16MBA: 16M bytes Q2MEM-32MBA: 32M bytes						Necessary

Memory device		Q03UDVCPU	Q04UDVCPU/ Q04UDPVCPU	Q06UDVCPU/ Q06UDPVCPU	Q13UDVCPU/ Q13UDPVCPU	Q26UDVCPU/ Q26UDPVCPU	Formatting
Program memory		120K bytes (30K steps)	160K bytes (40K steps)	240K bytes (60K steps)	520K bytes (130K steps)	1040K bytes (260K steps)	Necessary
Standard ROM		1025.5K bytes			2051K bytes	4102K bytes	Unnecessary
Standard RAM*2		192K bytes	256K bytes	768K bytes	1024K bytes	1280K bytes	Necessary*1
	With an extended SRAM cassette (1M)	1216K bytes	1280K bytes	1792K bytes	2048K bytes	2304K bytes	
	With an extended SRAM cassette (2M)	2240K bytes	2304K bytes	2816K bytes	3072K bytes	3328K bytes	
	With an extended SRAM cassette (4M)	4288K bytes	4352K bytes	4864K bytes	5120K bytes	5376K bytes	
	With an extended SRAM cassette (8M)	8384K bytes	8448K bytes	8960K bytes	9216K bytes	9472K bytes	
SD memory card		Capacity of the SD memory card used					Necessary

*1 When the memory contents become indefinite in initial status or due to the end of battery life, the memory is automatically formatted after the CPU module is powered off and then on or is reset.

*2 This is the capacity when an extended SRAM cassette is not used.

Point

- When files are written to each memory, the unit of stored file size depends on the target CPU module and memory area. (Page 50, Section 2.1.3 (4))
- In memory capacity calculation, 1 step is equal to 4 bytes.

(7) Memory and data to be stored

The following table lists data that can be stored in each memory.

◎ :Required, ○ :Storable, × :Not storable

Item	CPU module built-in memory			Memory card (RAM)	Memory card (ROM)		Memory card (SD)	File name and extension	Remarks
	Program memory	Standard RAM	Standard ROM	SRAM card	Flash card	ATA card	SD memory card		
	Drive 0[*1]	Drive 3[*1]	Drive 4[*1]	Drive 1[*1]	Drive 2[*1]		Drive 2[*1]		
Parameter	○	○[*14]	○	○	○	○	○	PARAM.QPA	1 data/drive
Intelligent function module parameter[*2]	○	○[*14]	○	○	○	○	○	IPARAM.QPA	1 data/drive
Program	◎	○[*14]	○[*3]	○[*4]	○[*4]	○[*4]	○[*4]	***.QPG	-
Device comment	○[*5]	○[*14]	○[*6]	○[*6]	○[*6]	○[*6]	○[*6]	***.QCD	-
Initial device value	○	○[*14]	○	○	○	○	○	***.QDI	-
File register	×	○[*7][*8]	×	○	○[*9]	×	×	***.QDR	-
Local device	×	○[*7]	×	○	×	×	×	***.QDL	1 data/CPU module
Sampling trace file	×	○[*7]	×	○	×	×	×	***.QTD	-
Device data storage file	×	×	○	×	×	×	×	DEVSTORE.QST	-
Module error collection file	×	○[*7]	×	×	×	×	×	IERRLOG.QIE	-
Boot setting file	○	○[*14]	○	○	○	○	○	AUTOEXEC.QBT	-
Remote password	○	○[*14]	○	○	○	○	○	00000000.QTM	-
Latch data backup file	×	×	○	×	×	×	×	LCHDAT00.QBP	-
Backup data file	×	×	×	○	○	○	○	MEMBKUP0.QBP	-
Data logging setting file	×	×	○	×	×	×	○	LOGCOM.QLG, LOG01.QLG to LOG10.QLG	-
Data logging file	×	×	○[*13]	×	×	×	○	***.CSV	-
PLC user data	×	×	○	×	×	○[*10]	○[*10]	***.CSV/BIN	-
Symbolic information[*11]	○	○[*14]	○	○	×	○	○	*12	-
Drive heading	○	○[*14]	○	○	○	○	○	QN.DAT	-
System file for the iQ Sensor Solution function (data backup/restoration)	×	×	×	×	×	×	○	SSBRINF.QSI	-
Backup data file for the iQ Sensor Solution function (data backup/restoration)	×	×	×	×	×	×	○	***.QBR[*15]	-
Predefined protocol setting file	×	×	○	×	×	×	○	ECPRTCL.QPT	-

Item	CPU module built-in memory			Memory card (RAM)	Memory card (ROM)		Memory card (SD)	File name and extension	Remarks
	Program memory	Standard RAM	Standard ROM	SRAM card	Flash card	ATA card	SD memory card		
	Drive 0*1	Drive 3*1	Drive 4*1	Drive 1*1	Drive 2*1		Drive 2*1		
System information file for CPU module data backup/restoration	×	×	×	×	×	×	○	BKUPINF.QSL	-
System data file for CPU module data backup/restoration	×	×	×	×	×	×	○	BKUPDAT.QBK	-
Device data file for CPU module data backup/restoration	×	×	×	×	×	×	○	DEVDATA.QDT	-
Operation history file	×	×	○	×	×	×	○	OPERATE.QOL	-

*1 A drive number is used to specify memory where data are written or read by external devices using a sequence program or MC protocol. Since memory names are used to specify memory using a programming tool, drive numbers do not need to be considered.

*2 Store the intelligent function module parameters in the same drive with the parameters. The intelligent function module parameters stored in a different drive are not valid.

*3 A program stored in the standard ROM cannot be executed. Store the program to the program memory before execution.

*4 To execute a program stored in a memory card or SD memory card, make the setting in the Boot file tab of the PLC parameter dialog box.

*5 The device comments cannot be read by instructions in a sequence program.

*6 Several scans are required to read device comments using a sequence program.

*7 CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU store only one file for each item. The High-speed Universal model QCPU and Universal model Process CPU store more than one file for each item.

*8 For the number of storable file register points, refer to Page 392, Section 4.7.

*9 A sequence program allows reading only. No data can be written from the sequence program.

*10 Data can be written or read with the following instructions.
- SP.FREAD (batch-reads data from the specified file in the memory card.)
- SP.FWRITE (batch-writes data to the specified file in the memory card.)

*11 This is the data in which the information of label program configuration is stored.
GX Works2 Version 1 Operating Manual (Common)

*12 CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU: The file name and extension will be SRCINF1M.CAB or SRCINF2M.CAB for Simple projects (with a label), and SRCINF1I.CAB or SRCINF2I.CAB for Structured projects.
High-speed Universal model QCPU and Universal model Process CPU: The file name and extension will be SRCINF1M.C32 or SRCINF2M.C32 for Simple projects (with a label), and SRCINF1I.C32 or SRCINF2I.C32 for Structured projects.

*13 This file cannot be specified as a data storage file when the data logging function is used. To write data to the file, execute the write PLC user data function.

*14 Only the High-speed Universal model QCPU and Universal model Process CPU can store these data in the memory.

*15 This file name depends on the connection type of the iQ Sensor Solution data backup/restoration function. (iQ Sensor Solution Reference Manual)

Point

For methods for writing data to each memory (online operation), refer to the following.
Manual for the programming tool used

2.1.2 Parameter-valid drive

CPU modules operate according to parameter settings. Systems automatically select parameters from those stored in the drives for CPU module operation, according to the following priority order. A user does not have to select them.

(1) Priority of the parameter-valid drives

The CPU module operates according to parameters stored in a higher priority drive.

- Q00U(J)CPU, Q01UCPU

Priority		Drive where parameters are stored
High ↑ ↓ Low	1	Drive 0 (program memory)
	2	Drive 4 (standard ROM)

- Q02UCPU, QnUD(H)CPU, QnUDE(H)CPU

Priority		Drive where parameters are stored
High ↑ ↓ Low	1	Drive 0 (program memory)
	2	Drive 1 (memory card RAM)
	3	Drive 2 (memory card ROM)
	4	Drive 4 (standard ROM)

- QnUDVCPU, QnUDPVCPU

Priority		Drive where parameters are stored
High ↑ ↓ Low	1	Drive 0 (program memory)
	2	Drive 2 (memory card SD)*1
	3	Drive 3 (standard RAM)
	4	Drive 4 (standard ROM)

*1 When the CPU module is locked with a security key and parameters are stored in a memory card (SD) (no parameters stored in the program memory), "MISSING PARA" (error code: 2200) occurs.

Point

- If a parameter file with a boot setting exists in a memory card or SD memory card, the file will be transferred according to the setting. If the transfer target memory is set to the program memory, the file will be transferred to the program cache memory as well.
- To check the parameter file that the CPU module uses, see "Parameter Valid Drive Information" of "PLC Status Information" on the PLC Diagnostics window. (Page 466, Appendix 2)

  [Diagnostics] ⇨ [PLC Diagnostics]

(2) When to determine valid parameters

The CPU module automatically searches for parameters in the following timing and operates by the settings of the parameters stored in the drives:

- the CPU module is powered off and then on, or
- it is reset.

When storing parameters to a drive by executing the write to PLC function from a programming tool, the timing for validating the parameters differs depending on the drive.

(a) When parameters are stored to the drive different from the one that stores the parameters in operation

The parameters are validated according to the priority set to the drive after the CPU module is powered off and then on or is reset.

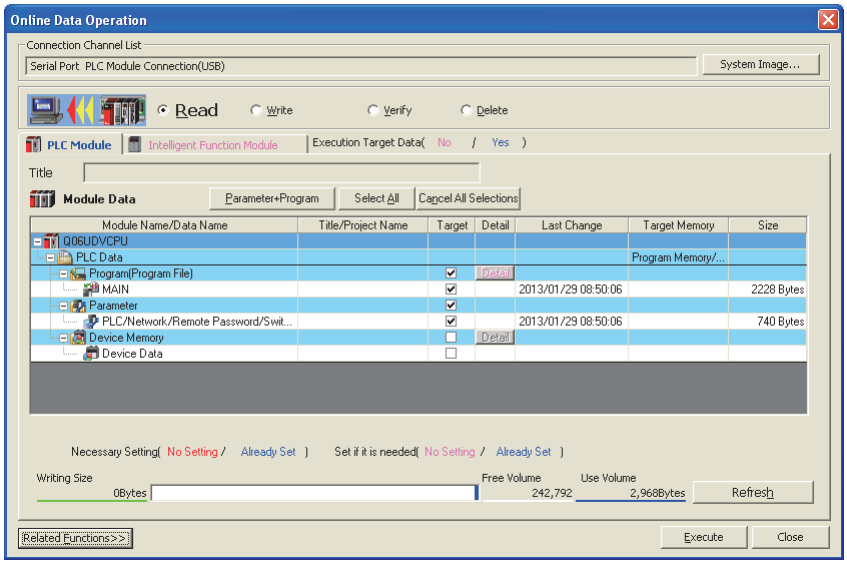
(b) When parameters are stored to the drive same as the one that stores the parameters in operation

Only the setting made in the Device tab of the PLC parameter dialog box is validated after “Write to PLC” is performed.
To validate all parameter settings, power off and then on or reset the CPU module.

2.1.3 Files

The files written to the CPU module have information such as a file name, file size, and written date. These information can be checked on the window displayed by selecting [Read from PLC] from the menu of a programming tool.

[Online] ⇨ [Read from PLC...]



Item	Description
File name	• File name structure and file specification Each file name is composed of a name (up to 8 characters in one byte/4 characters in double bytes) and an extension (3 characters in one byte). Create a file name with upper-case characters only. An extension is automatically appended according to the type set when the file was created. • Characters that cannot be used for a file name The following reserved words for Microsoft® Windows® cannot be used as a file name. COM1 to COM9, PRN, LPT1 to LPT9, NULL, AUX, CLOCK$, CON • How to specify a file name in the sequence program Since the sequence program is not case-sensitive in one-byte characters, the file can be named by both upper-case and lower-case characters. (Both "ABC" and "abc" are treated as "ABC".) In double-byte characters, an upper-case character and lower-case character are distinguished. Name a file by an upper-case character. ("ABC" and "abc" are distinguished.)
Last Change	The date and time when a file was written to the CPU module is shown. The date and time are based on the clock set on the programming tool (personal computer) side.
Size	The size of a file when it was written from a programming tool to the CPU module is shown in units of bytes. To display the latest data, click the Refresh button. At least 64 bytes (136 bytes for a program) are added to the file created by a user except a file register file. (☞ Page 46, Section 2.1.3 (2))

(1) Precautions for handling files

(a) Power-off or reset during file operation

If the CPU module is powered off or is reset during file operation, files in each memory remain as is. (To hold files in the memory card or SD memory card used, do not remove the card during power-off. Power off and on the CPU module with the card being inserted.)

Point

When the programmable controller is powered off during an operation in which a file is moved, the data in operation are held in the internal memory of the CPU module. The held data are recovered at power-on. To hold the internal memory data, battery backup is required.

(b) Simultaneous write from multiple programming tools to the same file

While data are being written to a file using a programming tool, another programming tool cannot access to the file until the writing ends. Also, a file is being accessed by a programming tool, another programming tool cannot write data to the file until the access ends. To write data to the same file from multiple programming tools, wait until the current processing ends and perform the processing one at a time.

(c) Simultaneous write from multiple programming tools to different files

Up to 10 programming tools can simultaneously access to different files in a CPU module.

(d) Access to the SD memory card

Accessing the SD memory card by using the SP.FREAD or SP.FWRITE instruction may cause the scan time to increase as the number of files stored in the SD memory card increases.

(2) File size

The size of a file used in the CPU module depends on the file type. When a file is written to the memory area, the unit of the stored file size depends on the CPU module and memory area to be written.

(☞ Page 50, Section 2.1.3 (4)) Calculate the rough size of each file with reference to the following table.

<table>
<tr><th colspan="2">File type</th><th>File size (unit: byte)</th></tr>
<tr><td colspan="2">Drive heading</td><td>72</td></tr>
<tr><td rowspan="7">Parameter</td><td>Default[*1]</td><td>• CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU: 464
• High-speed Universal model QCPU and Universal model Process CPU: 740</td></tr>
<tr><td>With boot setting[*2]</td><td>84 + (18 × (number of files))</td></tr>
<tr><td>With CC-Link IE Controller Network setting</td><td>72 + (total parameter sizes of each module) + (size of the routing setting) + (size of the data link transfer setting)
<table><tr><td>Parameter size of each module</td><td>Up to 10368 (When only LB/LW(1) is set, 1826 + 16 × (number of refresh transfer points).)</td></tr><tr><td>Size of the routing setting</td><td>6 + 8 × (number of routing settings)</td></tr><tr><td>Size of the data link transfer setting</td><td>6 + 12 × (number of transfer settings) + 86 × (number of modules)</td></tr></table></td></tr>
<tr><td>With CC-Link IE Field Network setting</td><td>72 + (total parameter sizes of each module) + (size of the routing setting) + (size of the data link transfer setting)
<table><tr><td>Parameter size of each module</td><td>Up to 13074</td></tr><tr><td>Size of the routing setting</td><td>6 + 8 × (number of routing settings)</td></tr><tr><td>Size of the data link transfer setting</td><td>6 + 16 × (number of RX transfer settings) + 16 × (number of RWr transfer settings)</td></tr></table></td></tr>
<tr><td>With MELSECNET/H setting</td><td>Increase up to 6180/module</td></tr>
<tr><td>With Ethernet setting</td><td>Increase up to 922/module</td></tr>
<tr><td>With CC-Link setting</td><td>Increase up to the values in the following table (The values indicate an increase per module.)
<table><tr><td rowspan="2">CC-Link setting</td><td colspan="3">Mode setting</td></tr><tr><td>Ver.1 mode</td><td>Ver.2 mode</td><td>Ver.2 additional mode</td></tr><tr><td>1st module</td><td>550 bytes</td><td>572 bytes</td><td>624 bytes</td></tr><tr><td>2nd to 4th modules</td><td>536 bytes</td><td>558 bytes</td><td>610 bytes</td></tr><tr><td>5th module</td><td>550 bytes</td><td>566 bytes</td><td>618 bytes</td></tr><tr><td>6th to 8th modules</td><td>536 bytes</td><td>558 bytes</td><td>610 bytes</td></tr></table></td></tr>
<tr><td>With remote password setting</td><td>92 + (number of target modules × 10), increase up to 172/module</td></tr>
<tr><td colspan="2">Sequence program</td><td>• CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU:
148[*3] + (4 × ((number of steps) + (number of steps of reserved area for online change)))
• High-speed Universal model QCPU and Universal model Process CPU:
224[*3] + (4 × ((number of steps) + (number of steps of reserved area for online change)))</td></tr>
</table>

File type	File size (unit: byte)
Device comment	• CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU: 74 + 8 + (total comment data size of each device) • High-speed Universal model QCPU and Universal model Process CPU: 74 + 72 + 8 + (total comment data size of each device) Comment data size per device = 10 + 10240 × a + 40 × b • a: Quotient of ((number of device points)/256) • b: Remainder of ((number of device points)/256)
Initial device value	• CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU: 66 + 44 × n + 2 × m + 8 • High-speed Universal model QCPU and Universal model Process CPU: 66 + 44 × n + 2 × m + 72 + 8 • m: Total number of device points set to the initial device value • n: Number of settings of the initial device value
User setting area	Setting value when formatted (0 to 15K)
File register	2 × (number of file register points)
Sampling trace file	362 + (number of word device points + number of bit device points) × 12 + (N1 + N2 + N3 + number of word device points × 2 + (number of bit device points/16) × 2) × the number of traces (total number of executions) • Apply the following values to N1 to N3 according to the items selected under "Trace additional Information" on the Trace condition settings window. (☞ Page 187, Section 3.14 (8) (b)) • N1: When "Time" is selected, apply "4". • N2: When "Step no." is set, apply "10". • N3: When "Program name" is selected, apply "8".
Device data backup file	Setting value when formatted (2 to 1024K)
Module error collection file	76 + (64 × (value set for the number of storable errors))
Local device[*4]	70 + 6 × (set device type) + (Am + Av + B + Ct + Cst + Cc) × n • Am, Av = (((a1 + a2) ÷ 16) - ((a1 + 1) ÷ 16) + 1) × 2 • B = b × 2 • Ct, Cst, Cc = ((((c1 + c2) × 2) ÷ 16 - ((c1 × 2 + 1) ÷ 16) + 1) × 2 + c2 × 2 • Am, Av: Save area sizes of M (internal relay) and V (edge relay), respectively a1: Start device number of M or V a2: Number of points of M or V • B: Save area size of D (data register) and Z (index register) b: Total number of points of D and Z • Ct, Cst, Cc: Save area sizes of T (timer), ST (retentive timer), and C (counter), respectively c1: Start device number of T, ST, or C a2: Number of points of T, ST, or C • N: Number of programs (only the ones using local devices[*5])
Data logging setting file	Refer to the following. QnUDVCPU/LCPU User's Manual (Data Logging Function)
System file for the iQ Sensor Solution function (data backup/restoration)	Refer to the following. iQ Sensor Solution Reference Manual
Backup data file for the iQ Sensor Solution function (data backup/restoration)	
Predefined protocol setting file	65532

File type	File size (unit: byte)
System information file for CPU module data backup/restoration	68 + (34 × (N + 1) + 34 × M) + L • N: Number of target drives • M: Number of target files • L: Total size of file names of target files (including punctuation mark (.) and extension)
System data file for CPU module data backup/restoration	64 + (6+ (a × 34)) + (6 + (a × (336 + (12 × b) + (12 × c))) + (6 + (d × 198)) + 22 + 5440 + (6 + e × 6418)) • a: Sampling trace registration status • b: Number of word device points of sampling trace information • c: Number of bit device points of sampling trace information • d: Data logging registration status • e: Module error collection setting status • For the status of a, d, and e, put “1” when they are registered/set and put “0” when they are not registered/set to obtain the above calculation formula.
Device data file for CPU module data backup/restoration	64 + 2 + 4 + 72 + (N × 8) + 11328 + ((a + (b ÷ 16) + (c × 2 ÷ 16)) × 2) • N: Number of target devices (for the timer, retentive timer, and counter, the contact and coil are counted as one device. • a: Number of word device points • b: Number of bit device points (except for the timer, retentive timer, and counter) • c: Number of bit device points (the timer, retentive timer, and counter)
Operation history file	The file size depends on the parameter setting. (1K to 1024K bytes)

*1 The value will be adjusted by the system so that the total number of bytes of PLC parameters and network parameters becomes multiple of four.

*2 The value will be adjusted by the system so that the total number of bytes becomes multiple of four.

*3 These are default values. (Values differ depending on the parameter setting).

*4 After the decimal point of a value obtained by a division part in the formula is rounded up.

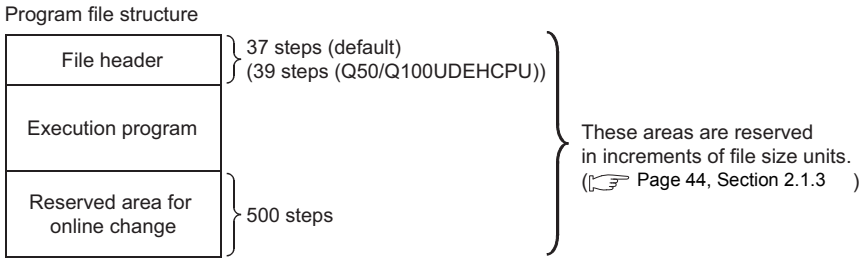
*5 For the Q02UCPU, Q03UDCPU, Q04UDHCPU, or Q06UDHCPU whose serial number (first five digits) is "10011" or earlier, apply the number of execution programs.

Remark

For a calculation example of memory capacity, refer to Page 51, Section 2.1.3 (4) (c).

(3) Program file structure

A program file consists of a file header, execution program, and reserved area for online change.



The sizes of the programs stored in the CPU module program memory are the total of above three areas.

Item	Description
File header	This area stores data such as the name, size, and created date of files. The file header size ranges from 25 to 41 steps (100 to 164 bytes) depending on the setting made in the Device tab of the PLC Parameter dialog box.
Execution program	This area stores the created program.
Reserved area for online change	This area is used when the number of steps is increased after writing data in the RUN status. (Default: 500 steps (2000 bytes) The setting value can be changed in the "Program Detail Setting" dialog box. (It can be changed while online change is performed.) After the online change is complete, remaining number of steps for this area is displayed.

(4) Memory capacity

When a file is written to the memory area, the unit of the stored file depends on the CPU module and memory area to be written. This unit is referred to as a file size unit.

(a) File size unit for each memory area

The following table lists the file size unit of the CPU module and memory area to be written.

<table>
<tr><th rowspan="2">CPU module model</th><th colspan="4">Memory area</th></tr>
<tr><th>Program memory</th><th>Standard RAM</th><th>standard ROM</th><th>Flash card*1</th></tr>
<tr><td>Q00UJCPU</td><td rowspan="9">1 step/4 bytes</td><td>-</td><td>64 steps/256 bytes</td><td rowspan="2">-</td></tr>
<tr><td>Q00UCPU, Q01UCPU</td><td rowspan="6">512 bytes</td><td rowspan="2">128 steps/512 bytes</td></tr>
<tr><td>Q02UCPU, Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU</td><td rowspan="3">128 steps/512 bytes</td></tr>
<tr><td>Q10UD(E)HCPU, Q13UD(E)HCPU</td><td>256 steps/1024 bytes</td></tr>
<tr><td>Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU</td><td>512 steps/2048 bytes</td></tr>
<tr><td>Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU</td><td>128 steps/ 512 bytes</td><td rowspan="3">-</td></tr>
<tr><td>Q13UDVCPU, Q13UDPVCPU</td><td>256 steps/ 1024 bytes</td></tr>
<tr><td>Q26UDVCPU, Q26UDPVCPU</td><td>512 steps/ 2048 bytes</td></tr>
</table>

*1 The file size unit of the Flash card is applied when a file is written to the Flash card by "Export to ROM Format".

(b) File size unit for each memory card or SD memory card

The following table lists the file size unit for each memory card or SD memory card.

<table>
<tr><th></th><th>Type</th><th>Model</th><th>File size unit (cluster size)</th></tr>
<tr><td rowspan="9">Memory card</td><td rowspan="4">SRAM card</td><td>Q2MEM-1MBSN, Q2MEM-1MBS</td><td>512 bytes</td></tr>
<tr><td>Q2MEM-2MBSN, Q2MEM-2MBS</td><td>1024 bytes</td></tr>
<tr><td>Q3MEM-4MBS</td><td>1024 bytes</td></tr>
<tr><td>Q3MEM-8MBS</td><td>4096 bytes</td></tr>
<tr><td rowspan="2">Flash card*1</td><td>Q2MEM-2MBF</td><td>1024 bytes</td></tr>
<tr><td>Q2MEM-4MBF</td><td>1024 bytes</td></tr>
<tr><td rowspan="3">ATA card</td><td>Q2MEM-8MBA</td><td>4096 bytes</td></tr>
<tr><td>Q2MEM-16MBA</td><td>4096 bytes</td></tr>
<tr><td>Q2MEM-32MBA</td><td>2048 bytes</td></tr>
<tr><td colspan="2" rowspan="6">SD memory card</td><td>NZ1MEM-2GBSD</td><td>32K bytes</td></tr>
<tr><td>NZ1MEM-4GBSD</td><td>32K bytes</td></tr>
<tr><td>NZ1MEM-8GBSD</td><td>32K bytes</td></tr>
<tr><td>NZ1MEM-16GBSD</td><td>32K bytes</td></tr>
<tr><td>L1MEM-2GBSD</td><td>32K bytes</td></tr>
<tr><td>L1MEM-4GBSD</td><td>32K bytes</td></tr>
</table>

*1 The file size unit of the Flash card is applied when:

- A file is written to the Flash card by Write to PLC (Flash ROM).
- A file is written to the Flash card using a programming tool without accessing the CPU module.

(c) Calculation example of memory capacity

Ex. The following describes a calculation example of memory capacity when parameters and a program are written to the program memory.

- Conditions
  1) CPU module to be written: Q26UDHCPU
  2) Writing file: Table below

File name	File size*1
PARAM.QPA (parameter file)	464 bytes
MAIN.QPG (sequence program)	525 steps / 2100 bytes

*1 For file size, refer to Page 50, Section 2.1.3 (4) (a).

3) Reserved area for online change: 500 steps/2000 bytes

- Memory capacity calculation

The memory capacity is calculated in units of file sizes of the CPU module to be written. The file size unit of the Q26UDHCPU in this example is 1 step/4 bytes. (☞ Page 50, Section 2.1.3 (4) (a))

1. **Calculation of parameter file size**

Since the parameter file size is 464 bytes, 116 steps/464 bytes is occupied on the program memory.

2. **Calculation of program size**

The program size is found by: Program size + reserved area for online change. Since a program is stored in units of file sizes (1 step), only the amount equal to the program size is occupied.

3. **Result**

The calculation results of the memory capacities are as shown below.

File name	File size		Memory capacity
PARAM.QPA	464 bytes		116 steps (464 bytes)
MAIN.QPG	Sequence program size	525 steps	1025 steps (4100 bytes)
	Reserved area for online change	500 steps	
	Total	1025 steps	
Total memory capacity			1141 steps (4564 bytes)

2.2 Base Unit Assignment

2.2.1 Base mode

Use this mode when assigning the number of available slots to the main base unit and extension base units.
The following two modes are available.

- Auto mode
- Detail mode

(1) Auto mode

Use this mode when assigning the number of slots equal to that on the base unit used.

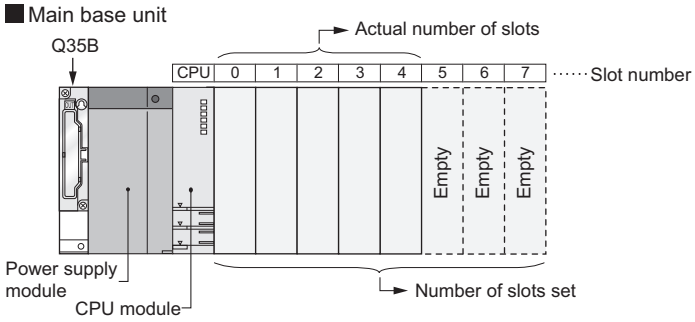
(2) Detail mode

Use the detail mode when assigning the number of slots for each base unit.
Any number of slots can be assigned irrespective of the actual number of slots on the base unit to be used.

(a) Setting the number of slots greater than the actual one

Slots are occupied by the number of slots set.
The slots after actually used ones are regarded as empty slots.

Ex. Three slots will be the empty slots when a 5-slot base unit is used and the number of available slots are set to eight.

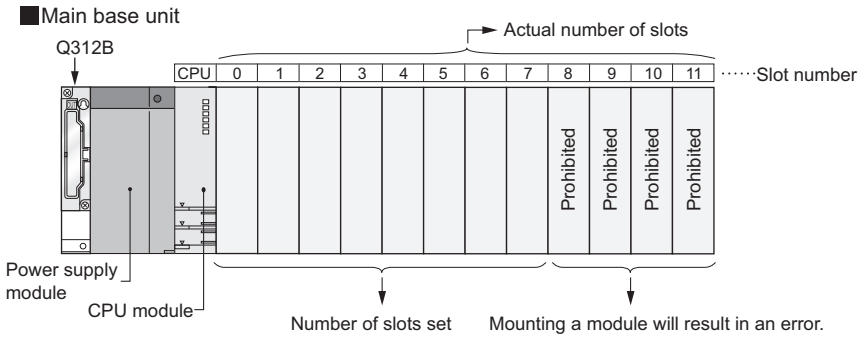


The number of points for the empty slots will be either value set on the PLC system tab, or on the I/O Assignment tab in the PLC parameter dialog box. (The default is 16 points.)

(b) Setting the number of slots smaller than the actual one

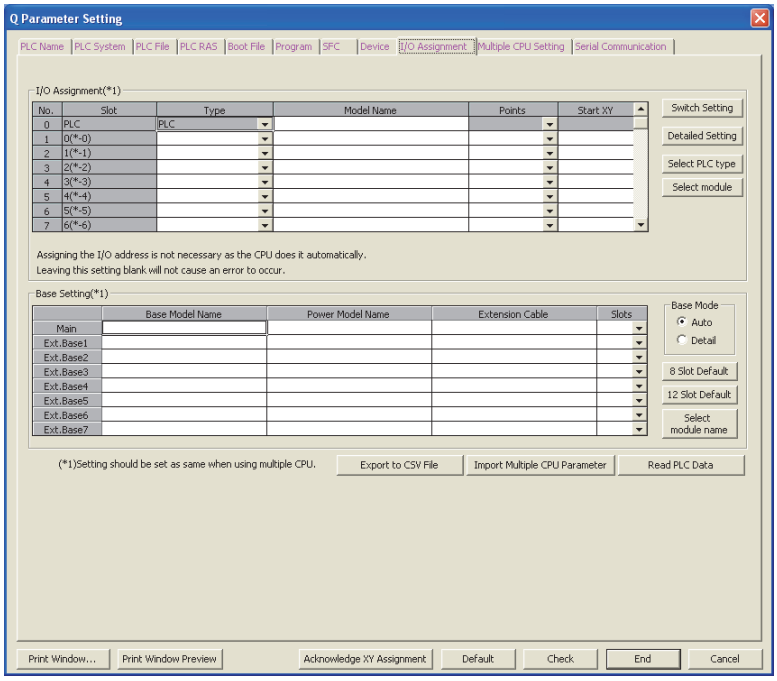
Set the smaller number than the actual number of slots when slots with no module mounted need not be recognized.

Ex. Four slots from the right end of the base unit will be the prohibited slots when using a 12-slot base unit and setting the number of available slots to eight. (Mounting a module on a prohibited slot causes "SP.UNIT LAY ERR.".)



2.2.2 Base unit assignment setting

Set base units on the I/O Assignment tab of the PLC parameter dialog box.



Item		Description
Base Mode	Auto	Select a mode for base unit assignment either from auto mode or detail mode.
	Detail	
Base Setting	Base Model Name	Enter the model names of base units, power supply modules, and extension cables to be used within 16 characters for reference or when printing out parameters. CPU modules do not use the entered model names.
	Power Model Name	
	Extension Cable	
	Slots	When "Detail" is set, select the number of slots on the base unit used.
8 Slot Default button		When "Detail" is set, select either of these items for batch-setting the base units to the specified number of slots
12 Slot Default button		

Point

- In auto mode, when any extension base unit number is skipped at the setting using the base number setting connector, an empty extension base unit cannot be reserved. To reserve empty extension base units for future extension, select detail mode.
- In detail mode, set the number of slots to all base units used. Failure to do so may result in incorrect I/O assignment setting.

2.3 I/O Number Assignment

The I/O number indicates addresses used for sequence programs in the following cases.

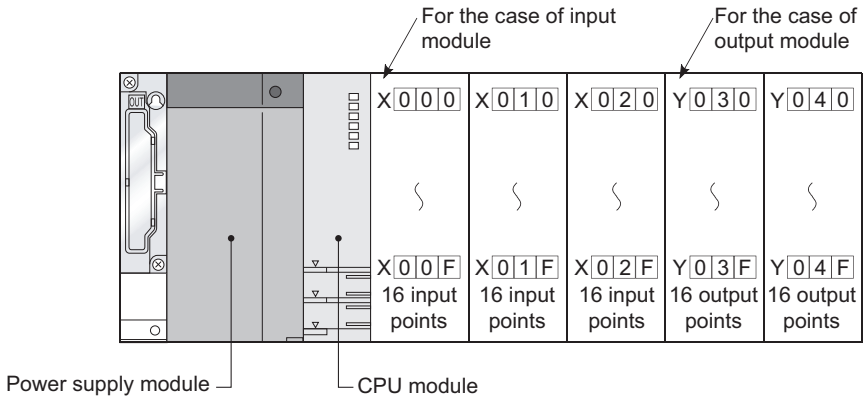
- Input of on/off data to the CPU module
- Output of on/off data from the CPU module to the external device

(1) Input and output of on/off data

The input (X) is used to input on/off data to the CPU module, and the output (Y) is used to output on/off data from the CPU module.

(2) I/O number representation

The I/O numbers are represented in hexadecimal. When a 16-point I/O module is used, the I/O number for each slot will be 16 point-sequence number from □□ 0 to □□ F as shown in the following figure. "X" and "Y" is prefixed to the I/O number of input modules and the I/O number of output modules, respectively.

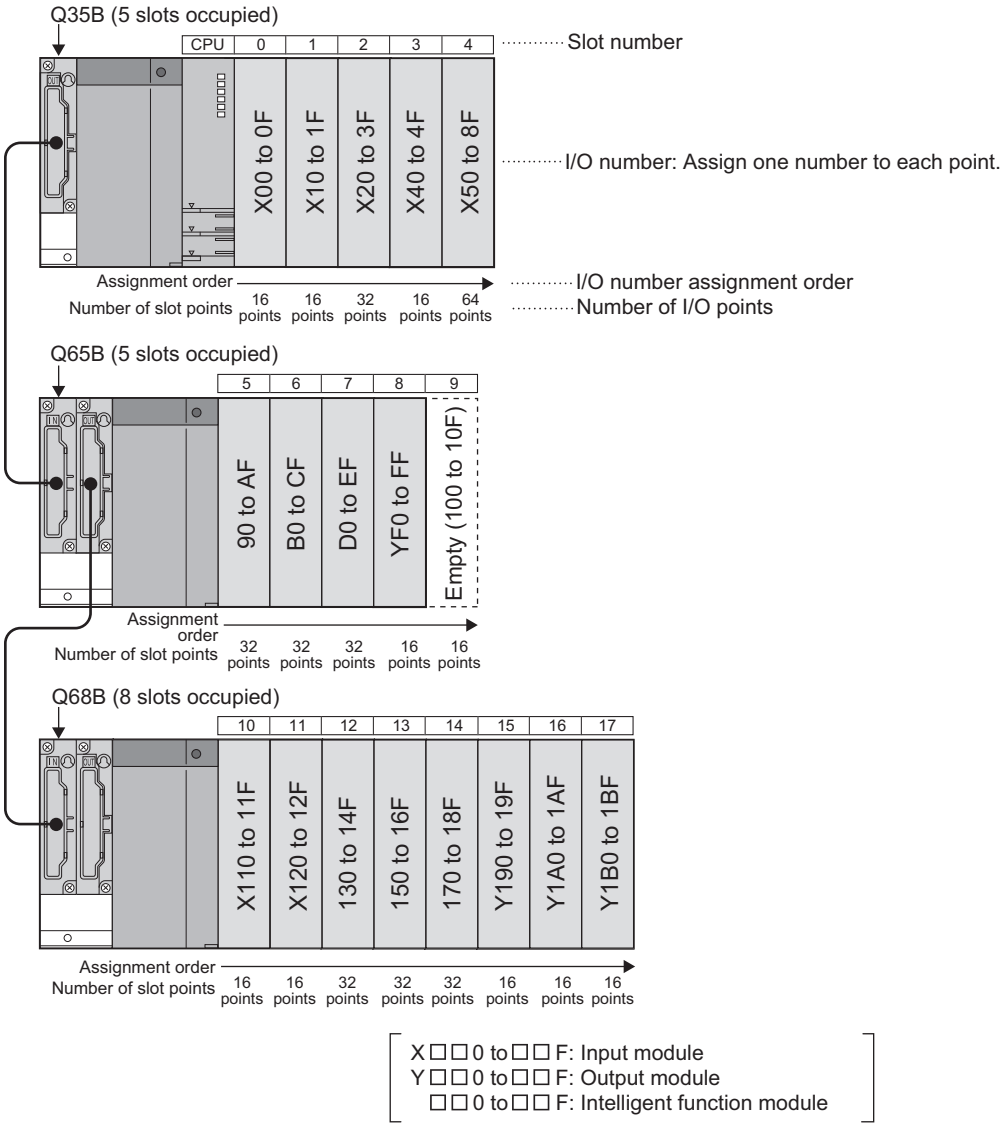


2.3.1 Concept of I/O number assignment

The CPU module assigns I/O numbers at power on or reset, according to the I/O assignment setting.

(1) I/O number assignment

The following figure shows an example of I/O number assignment to base units in the system where the CPU module is mounted on the main base unit.



(a) Assignment order

For the main base unit, the I/O numbers are assigned to the modules from left to right in a sequential order, starting from 0_H assigned to the module on the right of the CPU module.

For extension base units, the I/O numbers are continued from the last number of the I/O number of the main base unit.

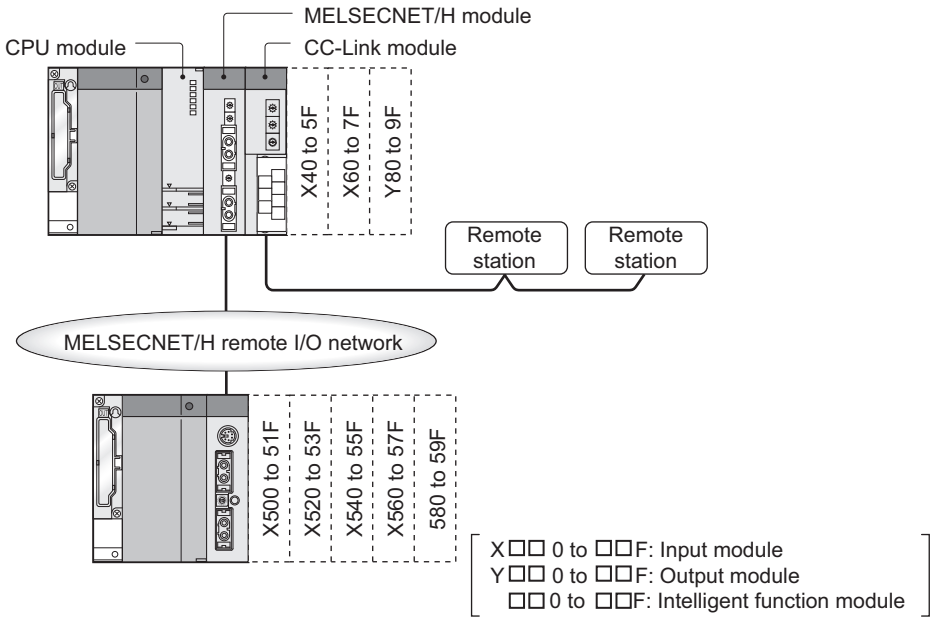
(b) I/O number of each slot

Each slot on the base unit occupies I/O numbers by the number of I/O points of the mounted modules.

(2) I/O assignment on a remote I/O stations

The devices of input (X) and output (Y) in the CPU module can be assigned to I/O modules and intelligent function modules, which allows to control the modules in the remote I/O system such as CC-Link IE Field Network, MELSECNET/H remote I/O network and CC-Link.

Also, the input (X) and output (Y) devices can be used as refresh-target devices for the link I/O (LX, LY) of CC-Link IE Field Network master/local modules and MELSECNET/H modules.



(a) I/O numbers available on remote I/O stations

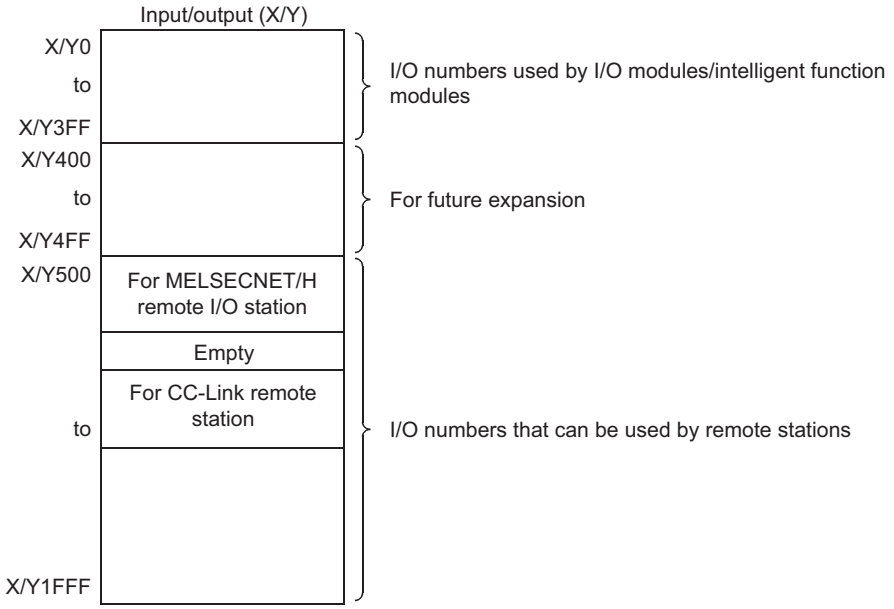
When the input (X) and output (Y) of the CPU module are used for the I/O numbers in the remote station, assign the I/O numbers later than those used for the I/O modules and intelligent function modules on the CPU module side.

Ex. When X/Y0 to X/Y3FF (1024 points) are used for the I/O modules and intelligent function modules on the CPU module side, X/Y400 and later can be used in the remote stations.

(b) Precautions for using remote station I/O numbers

• Setting for future extension

When the input (X) and output (Y) of the CPU module are used for the I/O numbers on the remote station, consider future extension of I/O modules and/or intelligent function modules on the CPU module side.



When X/Y0 to 3FF (1024 points) are used by I/O modules and/or intelligent function modules and X/Y400 to 4FF (256 points) are secured for future extension

• When CC-Link IE Field Network, MELSECNET/H, or CC-Link is used

Assign I/O numbers for the refresh-target devices (in the CPU module) of CC-Link IE Field Network or MELSECNET/H so that they do not overlap with those for the CC-Link remote stations.

Point

- When network parameter setting has not been made in the CC-Link system, X/Y 1000 to 17FF (2048 points) are assigned to the CC-Link system master/local modules of lower numbers.
- There are no restrictions on the I/O number assignment order for the CC-Link IE Field Networks, MELSECNET/H remote I/O networks, CC-Link.
- Free space can be provided in the area for the CC-Link IE Field Network remote station, MELSECNET/H remote I/O station, and CC-Link remote station.

2.3.2 Setting I/O numbers

Set the I/O numbers on the I/O Assignment tab.

(1) Purpose of I/O number assignment

(a) Reserving points for future module changes

The number of points can be flexibly set so that the I/O number modification can be avoided when changing the current module to another in the future.

Ex. 32 points can be assigned for future use to the slot where an input module with 16 points is currently mounted.

(b) Preventing I/O numbers from changing

The change in the I/O numbers can be prevented when an I/O module or intelligent function module, whose occupied I/O points are other than 16, is removed due to failure.

(c) Changing the I/O numbers to those used in the program

When the I/O numbers used in the actual system differ from those in the designed program, the I/O numbers of each module on the base unit can be changed to the ones in the designed program.

Point

- If any of the I/O modules whose number of I/O points are other than 16 fails without I/O assignment setting, the I/O numbers assigned following to the failed module may change, leading to a malfunction. For this reason, making the I/O assignment setting is recommended.
- I/O assignment setting allows the following settings as well. (The I/O assignment is required for the input response time and switch settings.)
  - Input response time setting (I/O response time) (☞ Page 139, Section 3.7)
  - Error time output mode setting (☞ Page 141, Section 3.8)
  - CPU module operation setting during a hardware error of intelligent function modules (☞ Page 142, Section 3.9)
  - Switch setting of intelligent function modules and interrupt modules (☞ Page 143, Section 3.10)

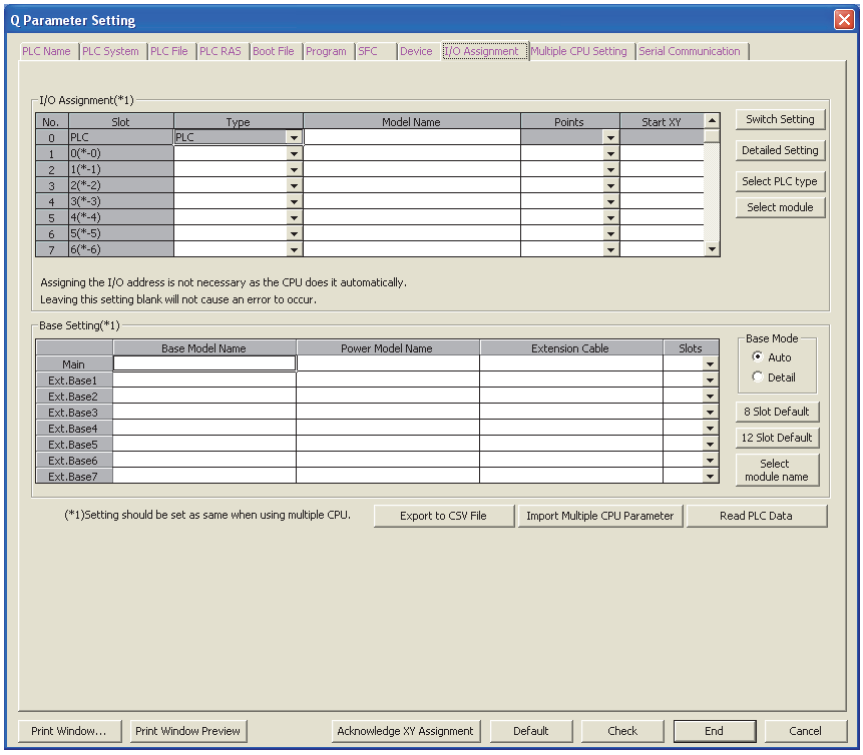
(2) I/O assignment

The I/O assignment is set on the I/O Assignment tab of the PLC parameter dialog box.

On the I/O Assignment tab, the following items can be set for each slot on the base unit.

- "Type" (module type)
- "Points" (I/O points)
- "Start XY" (start I/O number)

Ex. To change the I/O number of the specified slot, setting is allowed only to the number of points.
For other items that are not set, settings are completed based on the installation status of the base unit.



Item	Description
Slot	The slot number and location of the slot are displayed. When the base unit is set to auto mode, the base unit number is indicated in "*", and the slot number is counted from slot 0 of the main base unit.
Type	Select the type of the mounted module from the following: • Empty (empty slot) • Input (input module) • Hi. Input (high-speed input module) • Output (output module) • I/O Mix (I/O combined module) • Intelligent (intelligent function module) • Interrupt (interrupt module) When the type is not specified for a slot, the type of the actually mounted module applies.
Model Name	Enter the model names of mounted modules within 16 characters. CPU modules do not use entered model names. (Use the entered model names for reference.)
Points	When changing the number of I/O points for each slot, select the points from the following: • 0 Point • 16 Points • 32 Points • 48 Points • 64 Points • 128 Points • 256 Points • 512 Points • 1024 Points When the number of points is not specified for a slot, the number of points of the actually mounted module applies. For empty slots, the number of points set on the PLC System tab of the PLC parameter dialog box is assigned (default: 16 Points).
Start XY	When changing the I/O number of each slot, enter a new start I/O number. When this item is not specified for a slot, the I/O number counting from the last number of the current setting is assigned.

(3) Precautions

(a) Type setting

The type set in the I/O Assignment tab must be the same as that of the mounted module. Setting a different type may cause incorrect operation. For an intelligent function module, the number of I/O points must also be the same to the I/O assignment setting. The following table lists the operations when the mounted module type differs from the one set in the I/O Assignment tab.

Mounted module	I/O assignment setting	Result
Input module, output module, I/O combined module	• Intelligent • Interrupt	Error (SP.UNIT.LAY.ERR.)
Intelligent function module	• Input • Hi. Input • Output • I/O Mix	Error (SP.UNIT.LAY.ERR.)
Empty slot	• Input • Hi. Input • Output • I/O Mix • Intelligent • Interrupt	Empty slot
All modules	• Empty	Empty slot
Other combinations	-	Error does not occur but incorrect operation may be caused. Or, error (PARAMETER ERROR (error code: 3000)) is detected.

(b) I/O points of slots

The number of I/O points set in the I/O Assignment tab takes priority to that of mounted modules.

- When the number of I/O points is set to the number which is less than that of mounted I/O modules
  The available number of I/O points for the mounted I/O modules will be reduced.

Ex. When the number of I/O points is set to 16 points in the I/O Assignment tab of the PLC parameter dialog box for the slot where a 32-point input module is mounted, the second half 16 points of the module becomes unavailable.

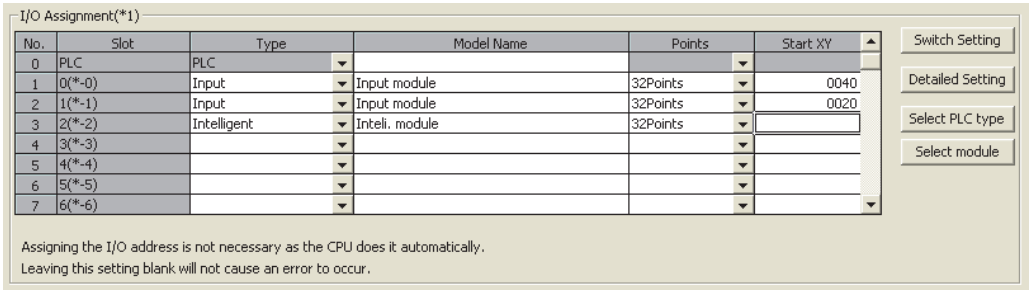
- When the number of I/O points is set to the number which is exceeding that of mounted I/O modules
  The exceeded number of points will not be used in the I/O modules.
- Last I/O number
  Set the last I/O number within the maximum number of I/O points. Failure to do so may cause an error ("SP. UNIT LAY ERR."). ("***" is displayed in "I/O Address" on the System Monitor screen.)
- When setting 0 points for empty slots
  Setting "Empty" for Type and "0 Point" for Points even occupies one slot. To set slots after the specific slot number unoccupied, set the number of slots in detail mode.
  (☞ Page 52, Section 2.2.1)

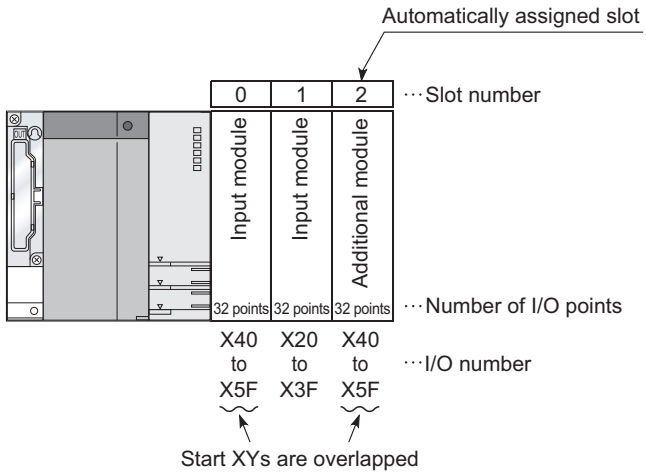
(c) Start XY setting

When the start XY has not been entered, the CPU module automatically assigns it. For this reason, the start XY setting of each slot may be duplicated with the one assigned by the CPU module in the case of 1) or 2) below.

1. **Start XY values are not in the correct order.**
2. **Slots with and without the start XY setting (automatically assigned slot) are mixed**

The following figure shows an example of start XY duplication.





Do not set duplicated start XY for each slot.
Specify start XY in the additional module to prevent the duplication of start XY.

Ex. Input "0060" to "start XY" in slot2.

Duplication of start XY will result in "SP. UNIT LAY ERR." (An error occurs even the module is not mounted.)

(d) When using AnS/A series compatible extension base units

When using the Q5□B/Q6□B in combination with the AnS/A series compatible extension base units, QA1S5□B, QA1S6□B, and QA6□B, take the following precautions.

- Connect the extension base units in the order of the Q5□B/Q6□B, QA1S5□B/QA1S6□B, QA6□B from the closest position to the main base unit.
- The QA1S51B does not have an extension cable connector (OUT) and therefore cannot be used in combination with the QA6□B.
- Batch-assign I/O numbers of the modules mounted on the base units for each series: Q series → A series or Aseries → Q series. Failure to do so will result in "SP.UNITLAY ERR.".

When the QA6ADP+A1S5□B/A1S6□B is used, refer to the following.

QA6ADP QA Conversion Adapter Module User's Manual

When the QA1S6ADP+A1S5□B/A1S6□B is used, refer to the following.

QA1S6ADP Q-AnS Base Unit Conversion Adapter User's Manual

QA1S6ADP-S1 Q-AnS Base Unit Conversion Adapter User's Manual

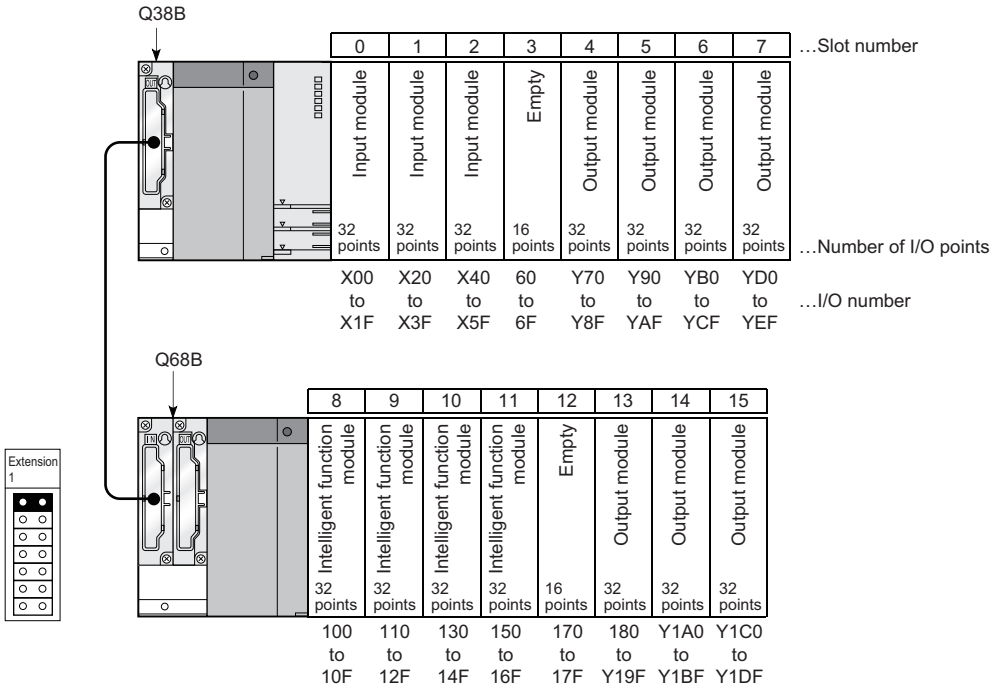
2.3.3 I/O number setting example

I/O number setting examples are provided as follows.

(1) Changing the number of points of an empty slot from 16 to 32

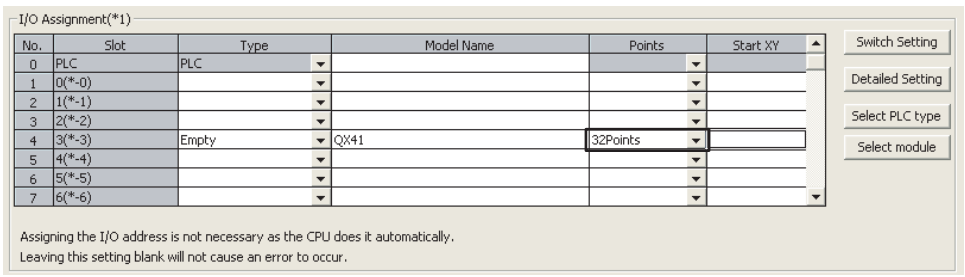
Reserve 32 points for the currently empty slot (Slot 3) so that the I/O numbers of Slot No. 4 and later do not change when a 32-point input module is mounted there in the future.

(a) System configuration and I/O number assignment before I/O assignment setting

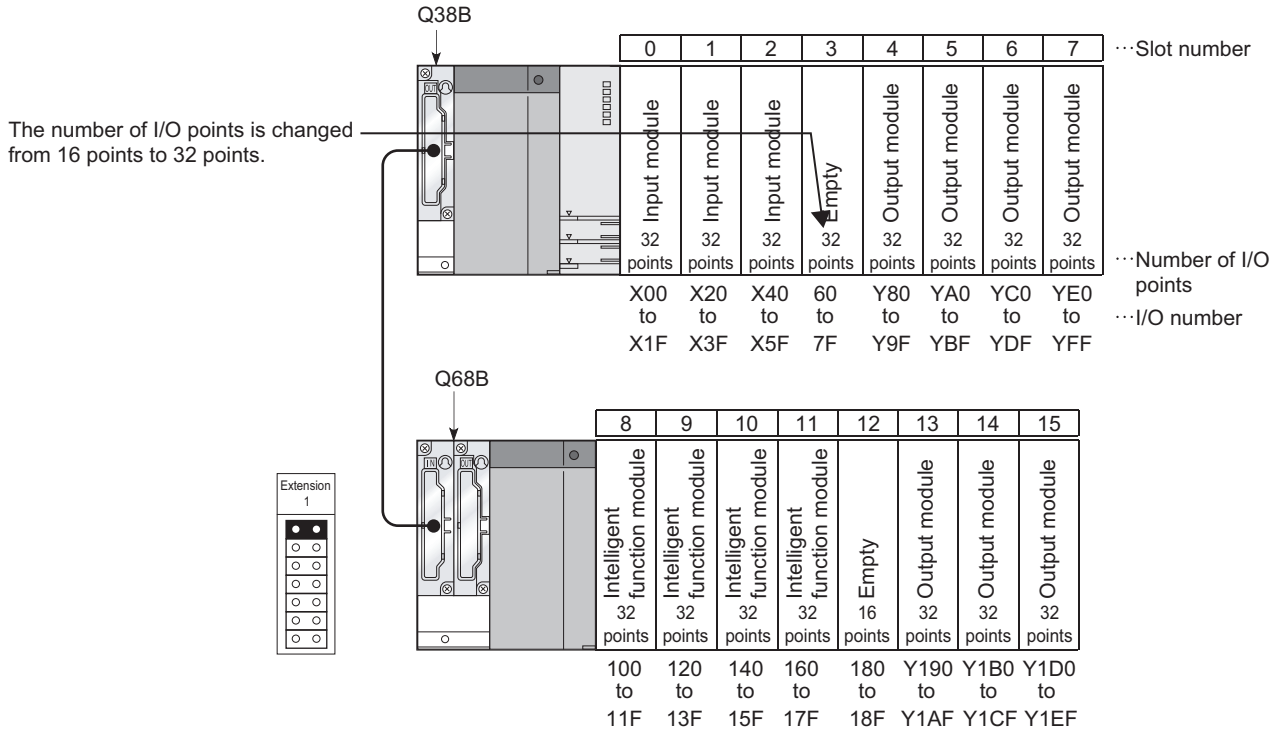


(b) I/O assignment

Select "32 Points" for "Points" of the slot 3 in the I/O Assignment tab of the PLC parameter dialog box. (When "Type" is not specified, the type of the mounted module will be set.)



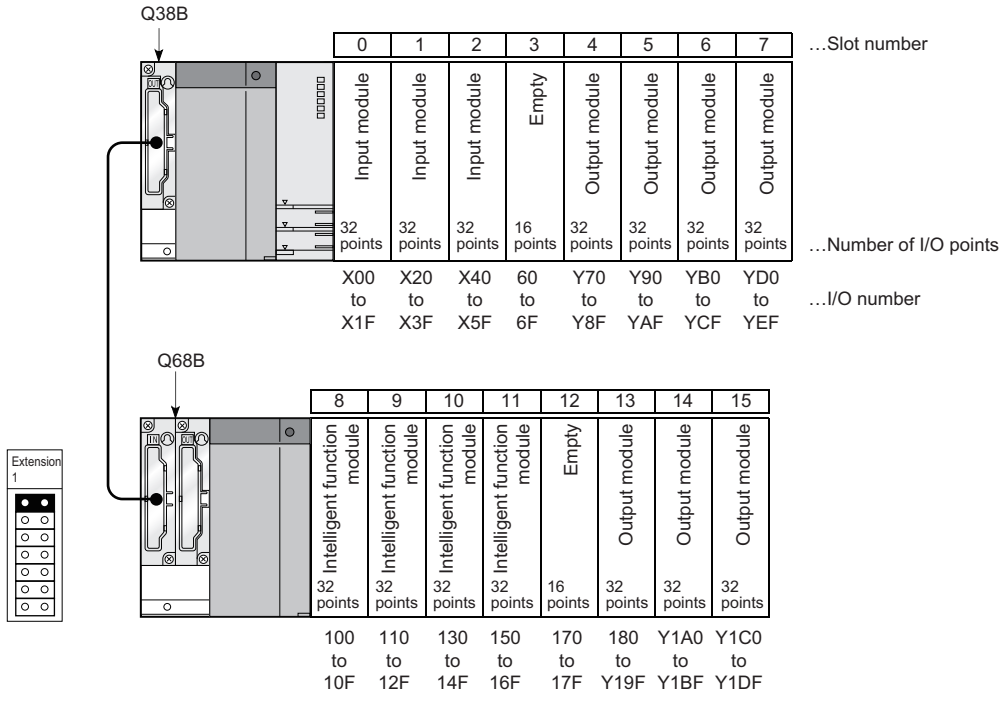
(c) I/O number assignment after the I/O assignment setting



(2) Changing the I/O number of an empty slot

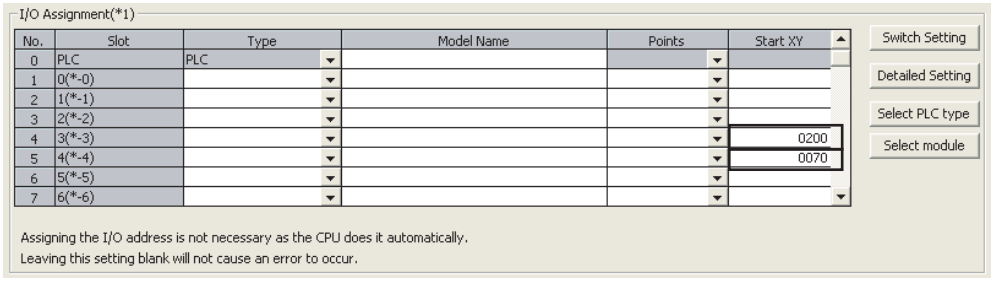
Change the I/O number of the currently empty slot (Slot 3) to X200 through 21F so that the I/O numbers of Slot 4 and later do not change when a 32-point input module is mounted there in the future.

(a) System configuration and I/O number assignment before I/O assignment setting

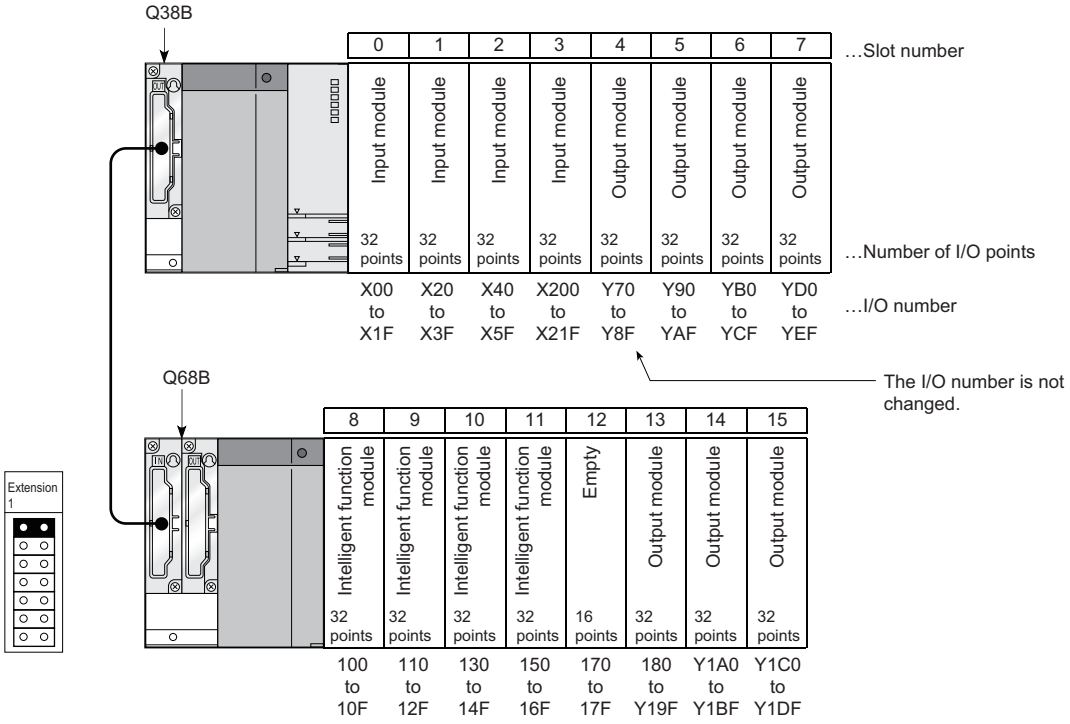


(b) I/O assignment

Set "200" for "Start XY" of the slot 3 and "70" for "Start XY" of the slot 4 in the I/O Assignment tab of the PLC parameter dialog box. (When the start I/O number is not set, the I/O number following the slot 3 will be set.)



(c) I/O number assignment after the I/O assignment setting

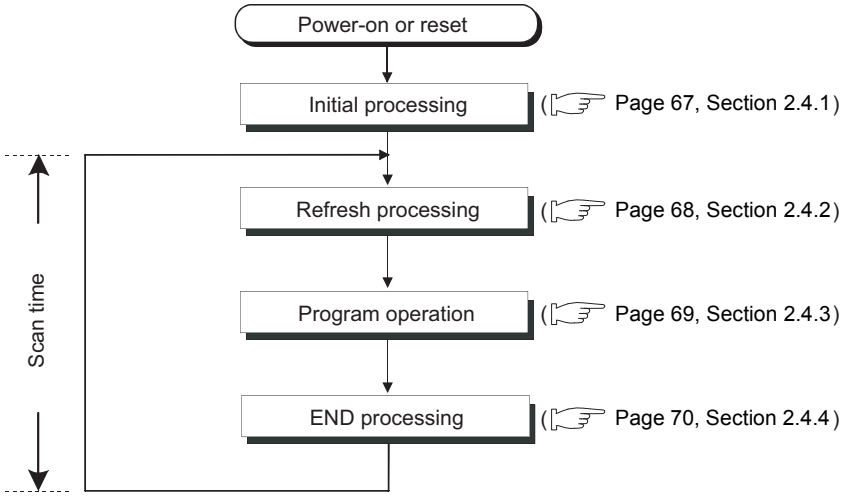


2.3.4 Checking I/O numbers

Information on mounted modules and their I/O numbers can be checked on the System Monitor screen in a programming tool.

2.4 Scan Time Structure

A CPU module sequentially performs the following processing in the RUN status. Scan time is the time required for all processing and executions to be performed.



2.4.1 Initial Processing

The CPU module performs pre-processing required for program operations. The processing is performed only once when any of the operations described in the following table is performed. When initial processing is completed, the CPU module will enter the status set using the RUN/STOP/RESET switch. (☞ Page 71, Section 2.5)

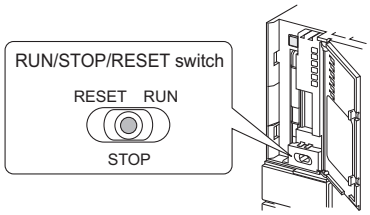
○ : Performed, × : Not performed

Initial processing item	CPU module status		
	Powered-on	Reset	Changed from STOP to RUN*1
The I/O module initialization	○	○	×
Boot from a memory card or SD memory card	○	○	×
PLC parameter check	○	○	○
Multiple CPU system parameter consistency check	○	○	○
Initialization of devices outside the latch range (bit device: off, word device: 0)	○	○	×
Automatic I/O number assignment of mounted modules	○	○	○
CC-Link IE and MELSECNET/H information setting	○	○	×
Intelligent function module switch setting	○	○	×
CC-Link information setting	○	○	×
Ethernet information setting	○	○	×
Initial device value setting	○	○	○
Serial communication function setting	○	○	×

*1 The operation indicates that the status is changed back to RUN without resetting the module after any parameter or program was changed in the STOP status. (The RUN/STOP/RESET switch is set from STOP to RUN (the RUN LED will flash), then back to STOP and to RUN again.) Note that the PLS, □P instruction (instruction for pulse conversion) may not be executed properly with the above operation. This is because the previous information may not be inherited depending on the program changes.

Point

If any parameter or program is changed in the STOP status, reset the CPU module using the RUN/STOP/RESET switch.



2.4.2 I/O Refresh (Refresh Processing with Input/Output Modules)

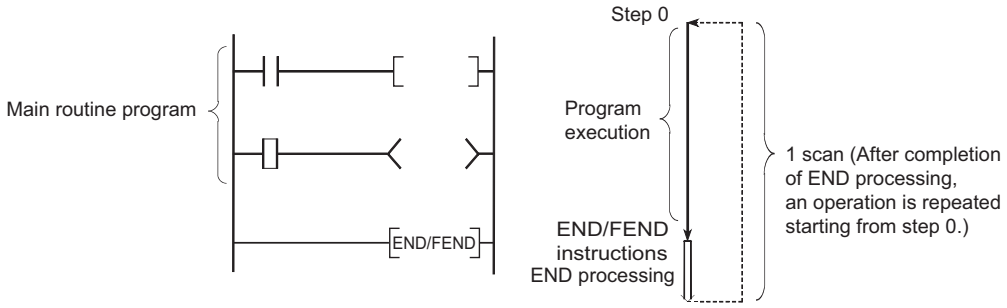
The CPU module performs the following before sequence program operations.

- On/off data input from the input module or intelligent function module to the CPU module
- On/off data output from the CPU module to the output module or intelligent function module

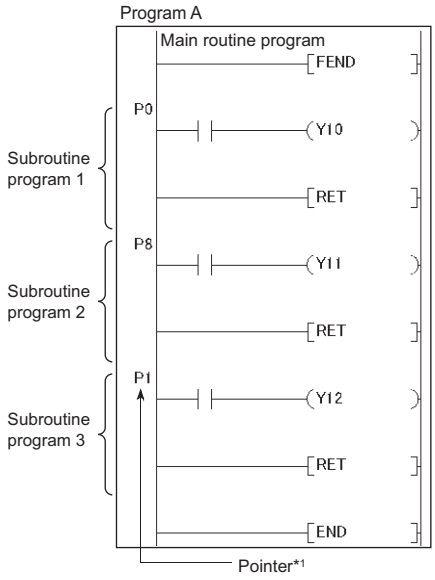
When the constant scan time is set, I/O refresh is performed after the constant scan waiting time has elapsed. (I/O refresh is performed at each constant scan cycle.)

2.4.3 Program Operation

The CPU module repeatedly executes the program stored in the module from step 0 to the END or FEND instruction. This program is referred to as a main routine program. This program is executed from step 0 in every scan.



A main routine program can be divided into subroutine programs. A subroutine program is from a pointer (P□) to the RET instruction, and is created between the FEND and END instructions. This program is executed only when called by an instruction, such as CALL(P) and FCALL(P), from a main routine program.



*1 Pointer numbers do not need to be specified in ascending order.

Use a subroutine program for purposes such as the following:

- To organize a program executed several times in one scan as a subroutine program so that the entire number of steps can be reduced
- To organize programs executed under the specific condition as a subroutine program so that the scan time can be reduced

Point

- Subroutine programs can be managed as one separate program (standby type program). (Page 95, Section 2.10.3)
- Subroutine programs can be configured with the nesting. (Page 407, Section 4.9)
- Using an interrupt pointer in a subroutine program changes the program to an interrupt program. (Page 82, Section 2.9)

2.4.4 END Processing

The CPU module performs refresh processing with network modules and communication with external devices.
END processing includes the following.

- Refresh with network modules
- Refresh with CC-Link IE Field Network Basic
- Auto refresh with intelligent function module
- Intelligent function module dedicated instruction processing
- Device data latch processing
- Service processing
- Watchdog timer reset
- Auto refresh between multiple CPU modules
- Device data collection using the sampling trace function (only when trace point is set to every scan (after END instruction execution))
- Self-diagnostics processing
- Special relay/special register value setting (only for those that should be set during END processing)

Point

When the constant scan function (☞ Page 119, Section 3.2) is used, the results of processing performed in END processing are held for the period between after END processing is completed and until the next scan starts.

2.5 Operation Processing in the RUN, STOP, or PAUSE Status

There are three types of operating status of the CPU module.

- RUN status
- STOP status
- PAUSE status

This section describes program operation processing in the CPU module based on its operating status.

(1) Operation processing in the RUN status

RUN status is a status where sequence program operations are repeatedly performed in a loop between the step 0 and the END (FEND) instruction.

(a) Output status when entering the RUN status

The CPU module outputs either of the following according to the output mode parameter setting when its status is changed to RUN. (☞ Page 125, Section 3.4)

- Output (Y) status saved in the STOP status
- Result of operations performed after one scan

(b) Processing time required before operations

The processing time required for the CPU module to start sequence program operations after its operating status is changed from STOP to RUN varies depending on the system configuration and/or parameter settings. (It takes one to three seconds normally.)

(2) Operation processing in the STOP status

STOP status is a status where sequence program operations are stopped by the RUN/STOP/RESET switch or the remote STOP function.

The CPU module status will be changed to STOP when a stop error occurs.

(a) Output status when entering the STOP status

When entering the STOP status, the CPU module saves data in the output (Y) and turns off all outputs.

The device memory other than that of the output (Y) will be held.

(3) Operation processing in the PAUSE status

PAUSE status is a status where sequence program operations are stopped by the remote PAUSE function after operations are performed for one scan, holding the output and device memory status.

(4) Operation processing when operating status of the CPU module changed

RUN/STOP status	CPU module operation processing			
	Sequence program operation processing	External output	Device memory	
			M, L, S, T, C, D	Y
RUN → STOP	The CPU module executes the program until the END instruction and stops.	The CPU module saves the output (Y) status immediately before its status is changed to STOP and turns off all the outputs.	The CPU module holds the device memory status immediately before its status is changed to STOP.	The CPU module saves the output (Y) status immediately before its status is changed to STOP and turns off all the outputs.
STOP → RUN	The CPU module executes the program from the step 0.	The CPU module outputs data according to the output mode parameter setting. (☞ Page 125, Section 3.4)	The CPU module holds the device memory status immediately before its status is changed to STOP. Note that the CPU module uses initial device values if those values are preset. Local device data are cleared.	The CPU module outputs data according to the output mode parameter setting. (☞ Page 125, Section 3.4)

Point

The CPU module performs the following in any of the RUN, STOP, or PAUSE status.

- Refresh processing with I/O modules
- Refresh processing with network modules
- Refresh processing with CC-Link IE Field Network Basic
- Auto refresh processing with intelligent function modules
- Self-diagnostics processing
- Service processing
- Intelligent function module dedicated instruction processing (completion processing only)
- Operation processing of Multiple CPU high speed transmission function
- Setting values for special relay/special register (only for those that should be set during END processing)

Even if the CPU module is in the STOP or PAUSE status, the following operations can be executed.

- I/O monitor or test operation from a programming tool
- Read/Write data from/to external devices using the MC protocol
- Communication with other stations using CC-Link IE or MELSECNET/H
- Communication with CC-Link remote stations

2.6 Operation Processing during Momentary Power Failure

When the input voltage supplied to the power supply module drops below the specified range, the CPU module detects a momentary power failure and performs the following operation.

(1) When a momentary power failure occurs for a period shorter than the allowable power failure time

The CPU module registers error data and suspends the operation processing.
The CPU module, however, continues measurement in the timer device and holds the output status.

(a) When resume start is specified for the SFC program

Data in the system is saved.

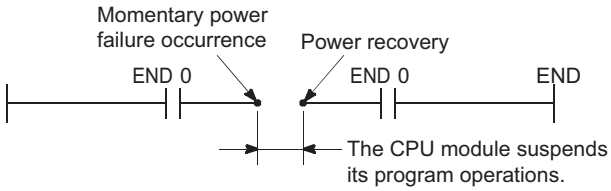
(b) When power is recovered after a momentary power failure

The CPU module restarts its operation processing.

(c) Watchdog timer (WDT) measurement during a momentary power failure

Even if operation processing is suspended due to a momentary power failure, the CPU module continues the measurement of the watchdog timer (WDT).

Ex. When the WDT setting of PLC parameter is 200ms and the scan time is 190ms, if a momentary power failure occurs for 15ms, "WDT ERROR" occurs.



(2) When a momentary power failure occurs for a period longer than the allowable power failure

The CPU module starts its operations initially.
Operation processing will be the same as that when any of the following is performed.

- Programmable controller is powered on.
- The CPU module is reset by the RUN/STOP/RESET switch.
- The CPU module is reset by a programming tool (the remote reset operation).

Point

- In a redundant power supply system, the CPU module does not suspend its operations if a momentary power failure occurs in either of the power supply modules. However, if a momentary power failure occurs under the condition where the power is supplied to only one of the power supply modules, operations are suspended.
- Information of a momentary power failure occurred in a redundant power supply system will be stored in SM1782 to SM1783 and SD1782 to SD1783. On the other hand, information of a momentary power failure occurred in a single power supply system will be stored in SM53 and SD53.

2.7 Data Clear Processing

This section describes how to clear data in the CPU module and settings required for clearing latch data.

(1) Clearing data

Data in the CPU module are cleared when the reset operation (using the RUN/STOP/RESET switch or by powering off and on the module) is performed. However, the following data cannot be cleared by these operations:

- Data in the program memory
- Data in the standard ROM
- Data in a memory card or SD memory card
- Data in latch-specified devices (☞ Page 75, Section 2.7 (4))

(2) Clearing data that cannot be cleared by the reset operation

(a) Data in the program memory

Clear the data by:

- Selecting the "Clear Program Memory" checkbox in the Boot File tab of the PLC parameter dialog box.
- Configuring settings on the screen opened by selecting [Online] → [Delete PLC Data] in a programming tool.

(b) Data in the standard ROM

Data stored on the standard ROM are automatically cleared when new data is written on it.

(c) Data in a memory card or SD memory card

Clear the data by configuring settings on the screen opened by selecting [Online] → [Delete PLC Data] in a programming tool.

(d) Data in latch-specified devices

Refer to Page 75, Section 2.7 (4).

(3) Device latch specification

Set a latch range for each latch-target device in the Device tab of the PLC parameter dialog box.

(☞ Page 123, Section 3.3 (4))

(a) Latch range setting

Two different types of latch ranges can be set using a programming tool:

- Latch clear operation enable range (Latch (1) Start/End)
  This is a range within which data can be cleared by a latch clear operation.
- Latch clear operation disable range (Latch (2) Start/End)
  This is a range within which data cannot be cleared by a latch clear operation.

(4) Clearing latch data

(a) Data in the latch clear operation enable range (Latch (1) Start/End)

Perform either of the following.

- Remote latch clear

  Perform the operation using a programming tool. (Page 137, Section 3.6.4)

- Latch clear by using the special relay and special register areas Note 2.2

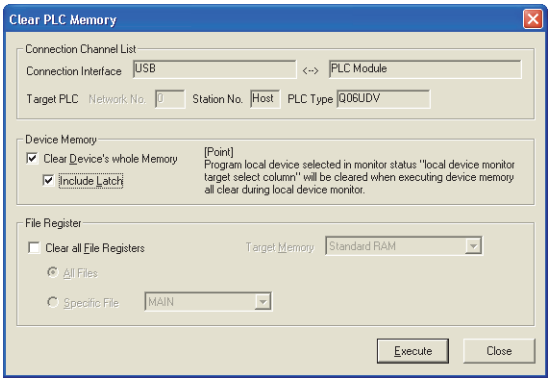
  1. **Change the operating status of the CPU module to STOP.**
  2. **Set "5A01$_H$" in SD339.**
  3. **Turn on SM339.**

(b) Data in the latch clear operation disable range (Latch (2) Start/End) and in the file register

Perform any of the following.

- Reset data by using the RST instruction.
- Transfer K0 by using the MOV or FMOV instruction. (MELSEC-Q/L Programming Manual (Common Instruction))
- Set parameters ("Clear Device's whole Memory" or "Clear all File Registers").

  [Online]⇨[PLC Memory Operation]⇨[Clear PLC Memory]⇨"Clear Device's whole Memory"/"Clear all File Registers"



Point

Latching device data increases the scan time. When latching device data, consider the increase in scan time.
(Page 478, Appendix 3.2 (6))

Note 2.2 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU support latch clear operation by using the special relay and special register areas. Before executing the function, check the version of the CPU module used.
(Page 466, Appendix 2)

2.8 I/O Processing and Response Delay

The CPU module performs I/O processing in the refresh mode.
Using the direct access input/output in a sequence program, however, allows the CPU module to perform I/O processing in the direct mode at the time of each instruction execution.
This section describes these I/O processing modes of the CPU module and response delays.

(a) Refresh mode (☞ Page 77, Section 2.8.1)

Refresh mode is a mode for the CPU module to access input/output modules and perform I/O processing collectively before the start of sequence program operations.

(b) Direct mode (☞ Page 80, Section 2.8.2)

Direct mode is a mode for the CPU module to access input/output modules and perform I/O processing at the timing when each instruction is executed in a sequence program.
To access input/output modules in the direct mode, use the direct access input or direct access output in a sequence program.

(1) Differences between refresh mode and direct mode

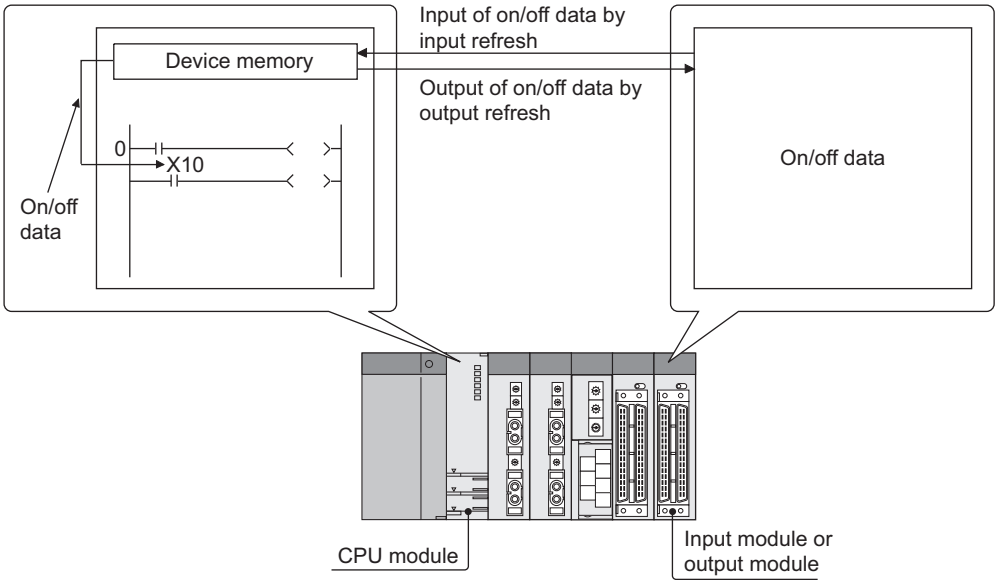
The direct mode directly accesses I/O modules at execution of an instruction. Therefore, data is input faster than when it is input in refresh mode. Processing time required for each instruction, however, takes longer. The following table lists the availability of the refresh mode and the direct mode for each input and output.

Item	Refresh mode	direct mode
Input/output modules	Available	Available
Input/output of intelligent function modules		
Input/output of MELSEC-I/O LINK Remote I/O System Master Module (AJ51T64/A1SJ51T64)*1		
Remote input/output in CC-Link IE, CC-Link IE Field Network Basic, MELSECNET/H, or CC-Link	Available	Not available

*1 The module must be mounted on the AnS/A series compatible extension base unit (QA1S5□B, QA1S6□B, QA6□B, QA6ADP+A5□B/A6□B, or QA1S6ADP+A1S5□B/A1S6□B). (The CPU module whose serial number (first five digits) is "13102" or later must be used. However, the QnUDPV cannot be used.)

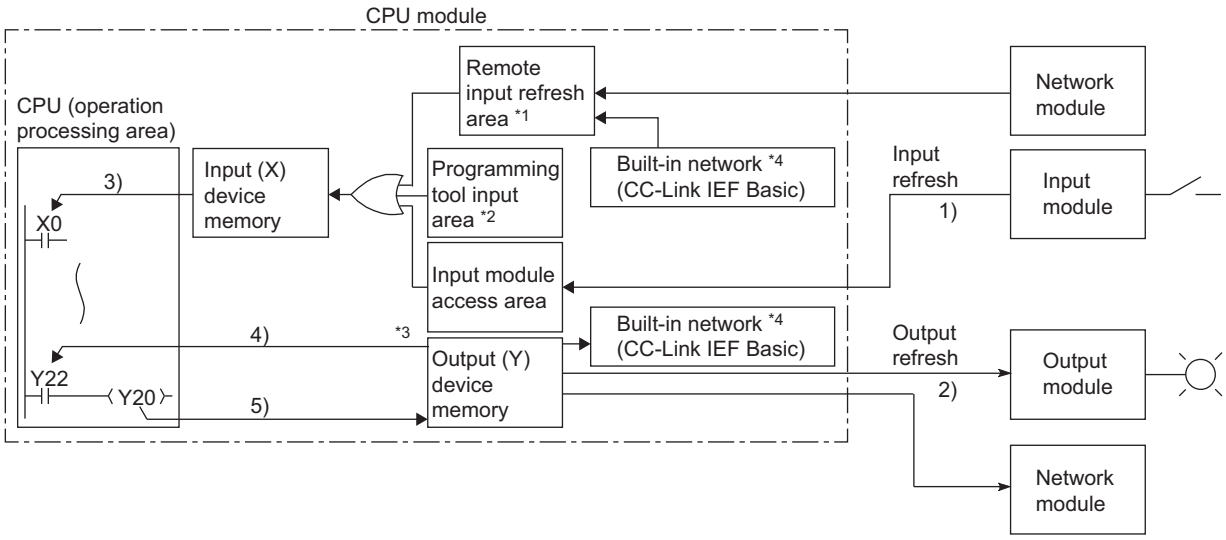
2.8.1 Refresh mode

In a refresh mode, the CPU module batch-performs I/O processing before the start of sequence program operations.



(1) Outline of the processing

The following describes the details of the refresh processing.



*1 The remote input refresh area indicates the area to be used when auto refresh is set to the input (X) in the CC-Link IE, CC-Link IE Field Network Basic, MELSECNET/H, or CC-Link. Data in the remote input refresh area will be refreshed automatically during END processing.

*2 Data in the programming tool input area can be turned on or off by the following:
- •Test operation of a programming tool
- •Writing data from the network module
- •Writing data from an external device using the MC protocol
- • Writing data using the simple PLC communication function

*3 Data in the output (Y) device memory can be turned on or off by the following:
- •Test operation of a programming tool
- •Writing data from an external device using the MC protocol
- •Writing data from the network module
- • Writing data using the simple PLC communication function

*4 This applies only to the QnUDVCPU and QnUDPVCPU.

Item	Description
Input refresh	Before program operation, input data are collectively read out from the input modules (1), the OR processing with the programming tool input area and remote input refresh area is executed, and then the data are stored in the input (X) device memory.
Output refresh	Before program operation, data in the output (Y) device memory (2) are collectively output to the output module.
Execution of an input contact instruction	Input data in the input (X) device memory (3) are read out and the program is executed.
Execution of an output contact instruction	Output data in the output (Y) device memory (4) are read out and the program is executed.
Execution of the OUT instruction	The operation result of the program (5) are stored to the output (Y) device memory.

(a) Input

On/off data of an input module are batch-input to the area for communication with the input module in the CPU module before the start of sequence program operations.

The CPU module performs sequence program operations using the on/off data stored in the input (X) device memory.

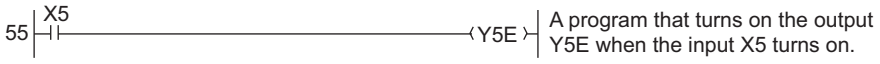
(b) Output

The operation results of the sequence program is output to the output (Y) device memory in the CPU module every time program operation is performed. Then, the CPU module batch-outputs the on/off data in the output (Y) device memory to an output module before the start of sequence program operations.

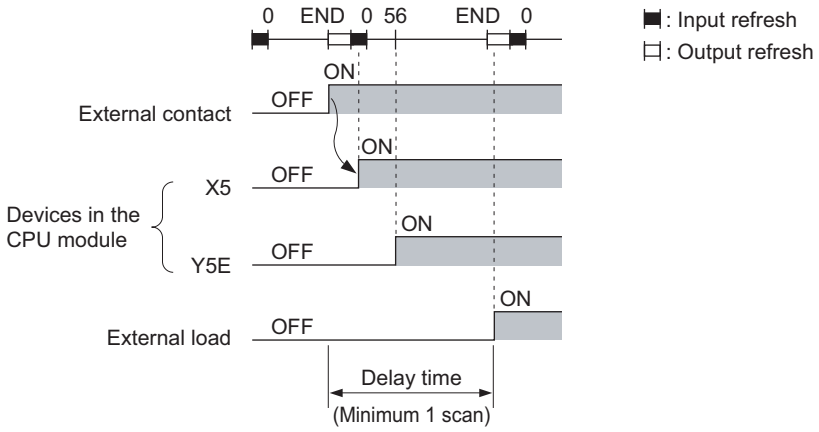
(2) Response delay

An output response which corresponds to the status change in the input module delays for two scans (maximum) depending on the on timing of an external contact.

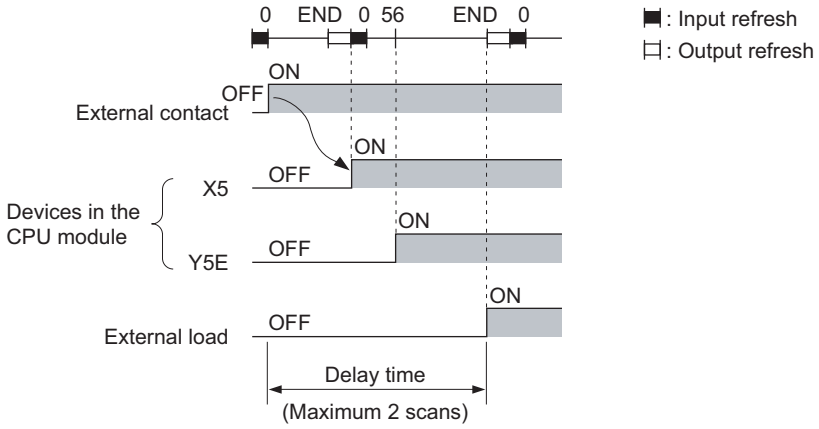
[Example]



- Y5E turns on the earliest

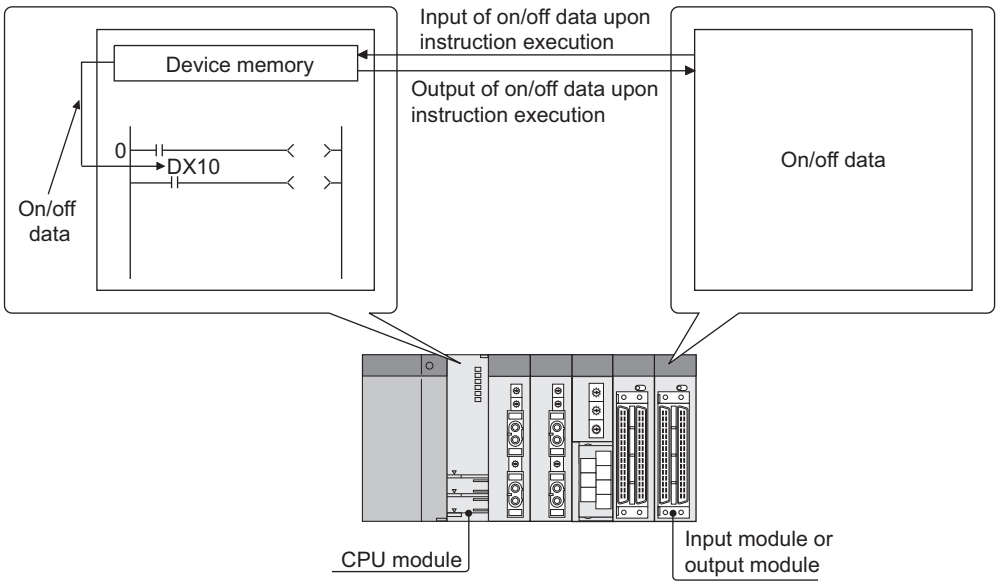


- Y5E turns on the latest



2.8.2 Direct mode

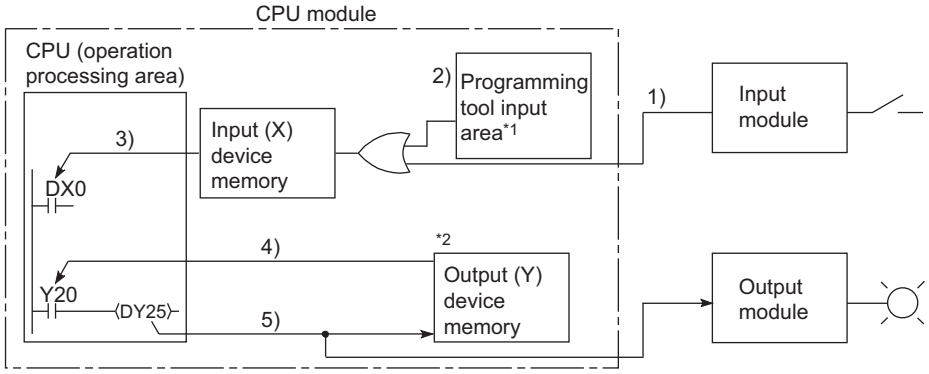
In a direct mode, the CPU module performs I/O processing when each instruction is executed in a sequence program.



With this mode, the CPU module uses the direct access input (DX) and direct access output (DY) to perform I/O processing.

(1) Outline of the processing

The following describes the details of the Direct processing.



*1 Data in the programming tool input area can be turned on or off by the following:
- •Test operation of a programming tool
- •Writing data from the network module
- •Writing data from an external device using the MC protocol
- • Writing data using the simple PLC communication function

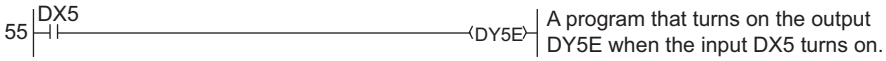
*2 Data in the output (Y) device memory can be turned on or off by the following:
- •Test operation of a programming tool
- •Writing data from an external device using the MC protocol
- •Writing data from the network module
- • Writing data using the simple PLC communication function

Item	Description
Execution of an input contact instruction	The OR processing is performed with the input information of the input module (1)) and the input data of the programming tool input area (2)) or remote input refresh area. The result is stored in the input (X) device memory and is used as input data (3)) to execute the program.
Execution of an output contact instruction	Output data in the output (Y) device memory (4)) are read out and the program is executed.
Execution of the OUT instruction	The operation result of the program (5)) are output to the output module, and stored in the output (Y) device memory.

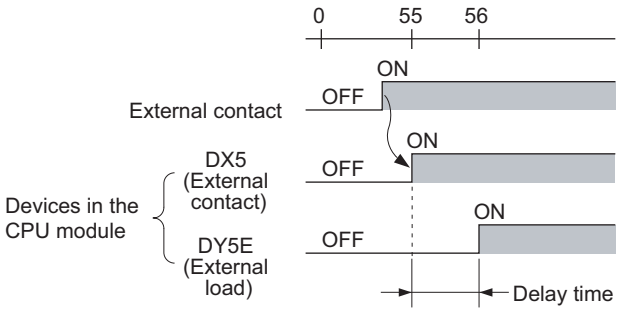
(2) Response delay

An output response which corresponds to the status change in the input module delays for one scans (maximum) depending on the on timing of an external contact.

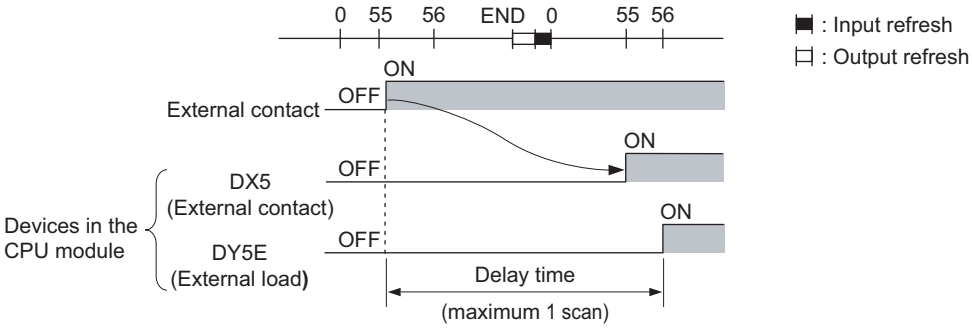
[Example]



- DY5E turns on the earliest

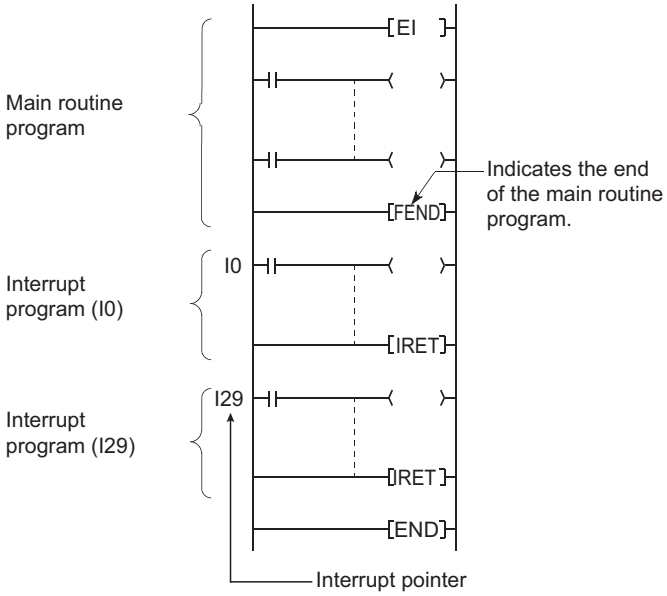


- DY5E turns on the latest



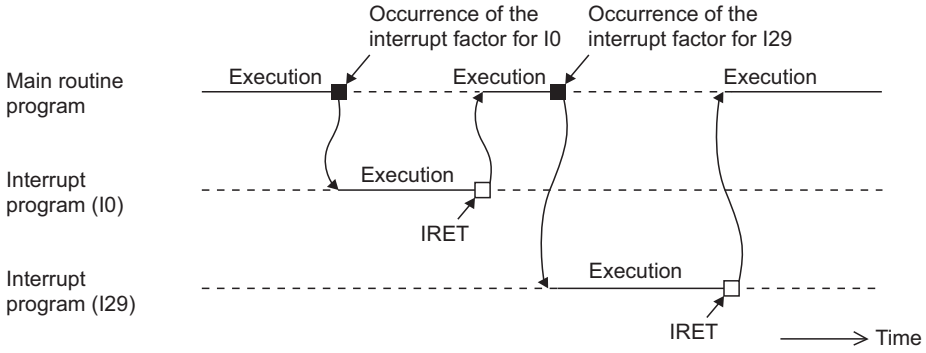
2.9 Interrupt Program

An interrupt program is from an interrupt pointer (I□) to the IRET instruction.



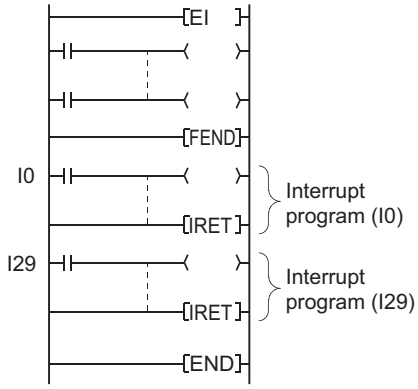
The interrupt pointer (I□) number varies depending on the interrupt factor. (☞ Page 412, Section 4.11)

When an interrupt factor occurs, the interrupt program of the interrupt pointer number corresponding to that factor is executed. (Interrupt programs are executed only when the interrupt factor occurs.



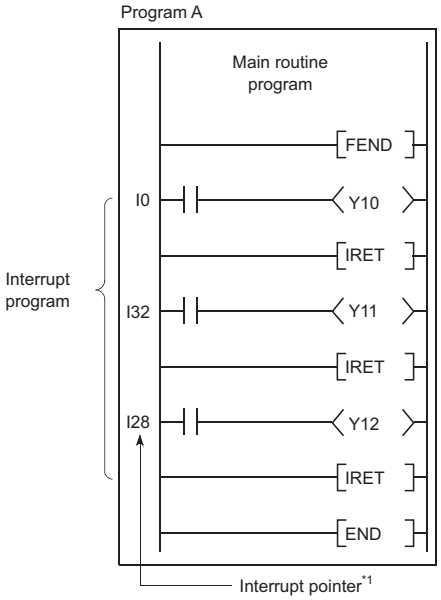
Point

Only one interrupt program can be created with one interrupt pointer number.



(1) Programming of interrupt programs

Create interrupt programs between the FEND and END instructions in the main routine program.



*1 The pointer numbers do not need to be specified in ascending order.

Point

Interrupt programs can be managed as one separate program (stand-by type program). (☞ Page 95, Section 2.10.3)

(a) Before executing an interrupt program

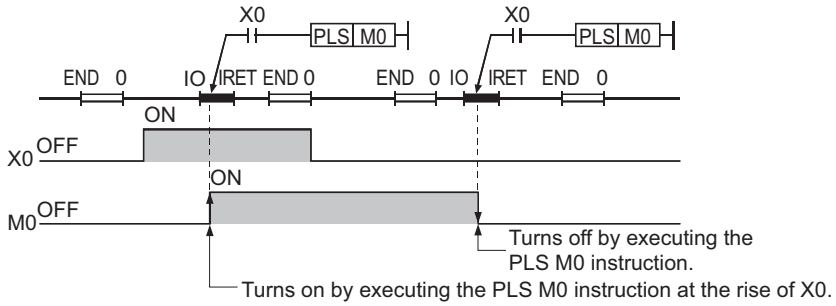
Before executing the interrupt programs of the interrupt pointers I0 to I15, I28 to I31, I45, I49, and I50 to I255, enable interrupts with the EI instruction. For details on the EI instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

(b) Restrictions on programming

- PLS and PLF instructions

  The PLS and PLF instructions perform off processing in the next scan after the instruction is executed. Therefore, the device which is turned on by the instruction remains on until the same instruction is reexecuted.



- EI and DI instructions

  During execution of an interrupt program, interrupts are disabled (DI) so that any other interrupt processing will not be executed. Do not execute the EI or DI instruction during interrupt program execution.

- Timer (T) and counter (C)

  Do not use the timer (T) and counter (C) in interrupt programs. If more than one interrupts occur in one scan, the timer (T) and counter (C) in the interrupt program cannot measure the time correctly. The OUT C□ instruction status causes the counter (C) measure the number of interrupts incorrectly.

- Instructions not available in interrupt programs

  Refer to sections corresponding to each instruction in the following.

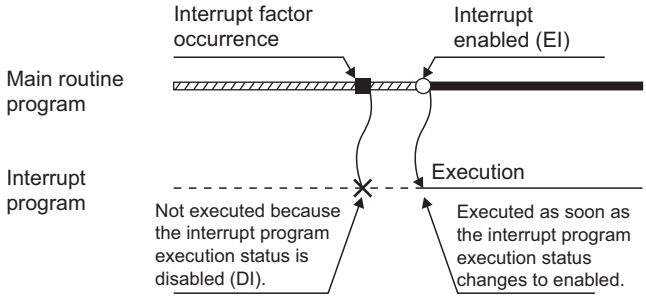
  MELSEC-Q/L Programming Manual (Common Instruction)

(2) Operation when an interrupt factor occurs

There are restrictions on interrupt programs depending on the interrupt factor occurrence timing.

(a) When an interrupt factor occurs before the interrupt program execution status is enabled

The CPU module stores the interrupt factor occurred.
As soon as the interrupt program execution status is enabled, the CPU module executes the interrupt program corresponding to the stored interrupt factor.

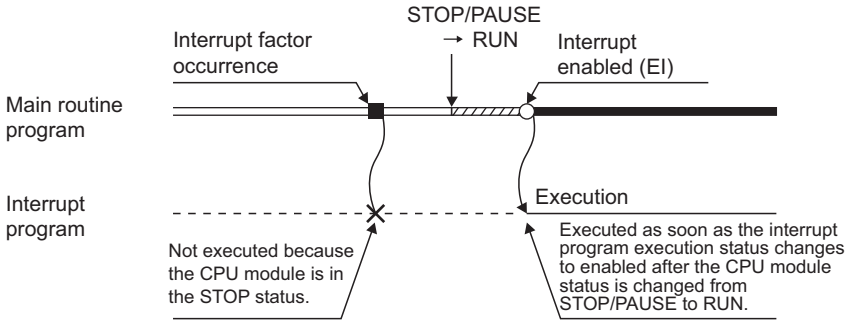


When the same interrupt factor occurs more than one time before the interrupt program execution status is enabled, the interrupt factors of I0 to I15, I28 to I31, I45, I50 to I255 and fixed scan execution type programs are stored only once. For details on the IMASK instruction, refer to the following.
However, the factors occurred by IMASK instruction at mask status are all discarded.

MELSEC-Q/L Programming Manual (Common Instruction)

(b) When an interrupt factor occurs in the STOP or PAUSE status

The CPU module executes the interrupt program as soon as the interrupt program execution status is enabled after the CPU module status is changed to RUN.

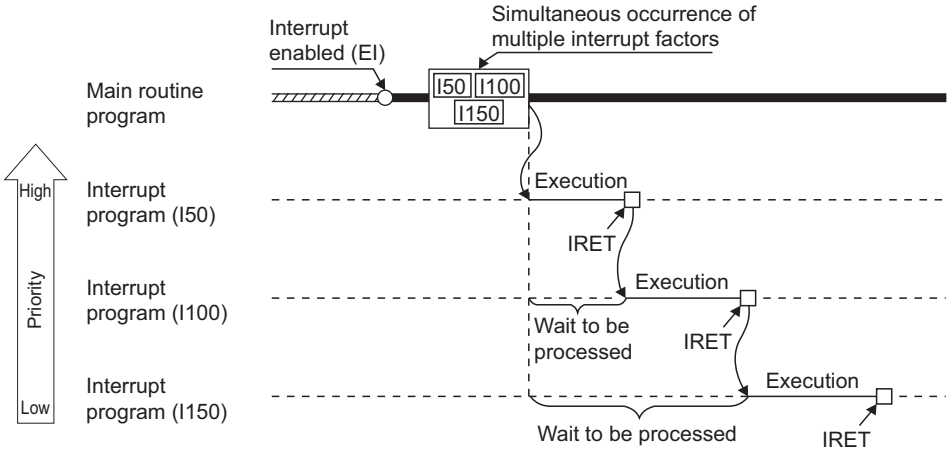


(c) When multiple interrupt factors occur simultaneously in the interrupt program execution enabled status

The interrupt programs are executed in the order of interrupt pointers (I□) with high priority.

(☞ Page 413, Section 4.11.1)

Other interrupt programs have to wait until processing of the interrupt program being executed is completed.



(d) When the same interrupt factor as that of the interrupt program being executed occurs

When the same interrupt factor as that of the interrupt program being executed occurs more than one time before completion of interrupt program processing, the interrupt factors of I0 to I15, I45, and I50 to I255 are stored only once, and then the interrupt program corresponding to each stored interrupt factor is executed after completion of current interrupt program execution.

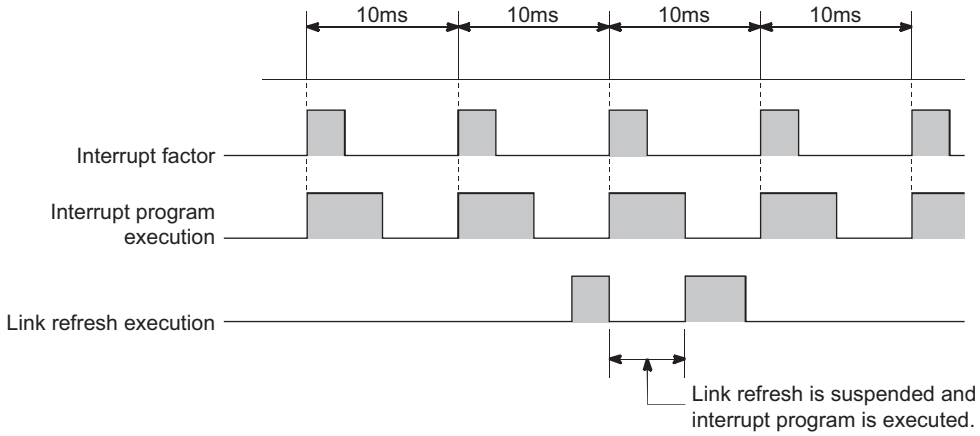
The interrupt factors of I28 to I31 and fixed scan execution type programs are all stored, and then all the interrupt programs corresponding to interrupt factors are executed after completion of current interrupt program execution.

(e) When an interrupt factor occurs during link refresh

The link refresh is suspended and an interrupt program is executed.

Even if the Block data assurance per station setting is enabled in the CC-Link IE or MELSECNET/H network, this setting does not work when a device set as a refresh target is used in the interrupt program.

In the interrupt program, do not use any refresh target device.



For the Block data assurance per station setting, refer to the following.

Manual for each network module

(f) Interrupt during END processing

When the constant scan function is used and an interrupt factor occurs during the waiting time in END processing, an interrupt program corresponding to the interrupt factor is executed.

(g) When an interrupt factor occurs during access to another module

When an interrupt factor occurs during access to another module (during service processing or instruction processing), the interrupt program becomes standby status until the service processing or the instruction in execution is completed.

To shorten the wait time of the interrupt, reduce the amount of data that access to other modules.

(3) Processing at program execution type change

When the program execution type is changed from the scan execution type to the interrupt, the CPU module saves and restores the following data. (☞ Page 390, Section 4.6.3)

- Data in the index register
- File register block number

Whether to save and restore the data above can be set in the PLC parameter dialog box.

If the data is not saved or restored, the overhead time of the corresponding interrupt program can be shortened.

(☞ Page 472, Appendix 3.2 (3))

(4) Precautions

(a) When the same device is used

During execution of an instruction in a main routine program, an interrupt program may be executed, suspending the processing of the instruction being executed.

If the same device is used for the main routine program and interrupt program, device data may become inconsistent. In this case, take the following measures to prevent device data inconsistency.

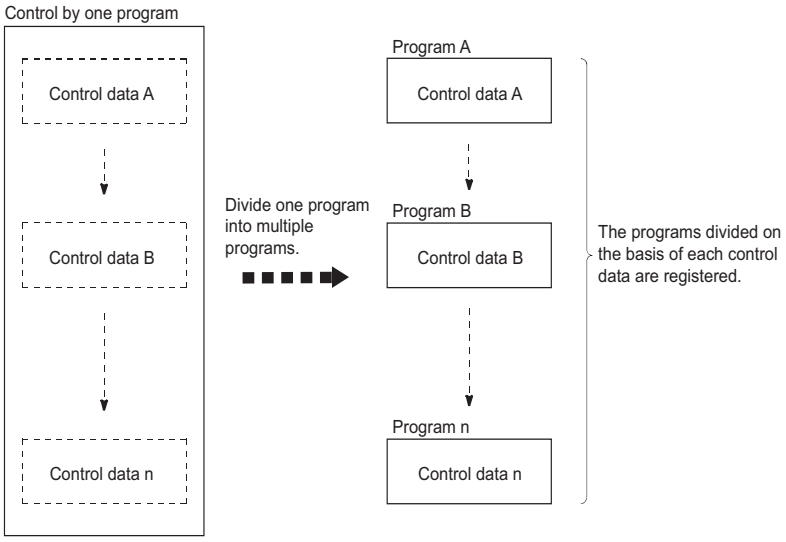
- Moving device data to another device
  Do not directly specify the device where the data is written by the interrupt program in the main routine program. Use the data in another device by moving the data with the transfer instruction.
- Disabling interrupts with the DI instruction
  Disable interrupts with the DI instruction if instructions that may cause inconvenience for the main routine program are used. However, interrupts do not occur during access to the device of the corresponding argument of the instruction. For this reason, data inconsistency will not occur in units of arguments.

2.10 Settings When Program is Divided

When one sequence program is divided into multiple programs, execution conditions, such as executing a program only once at start-up or executing a program at fixed intervals, can be set for each program.

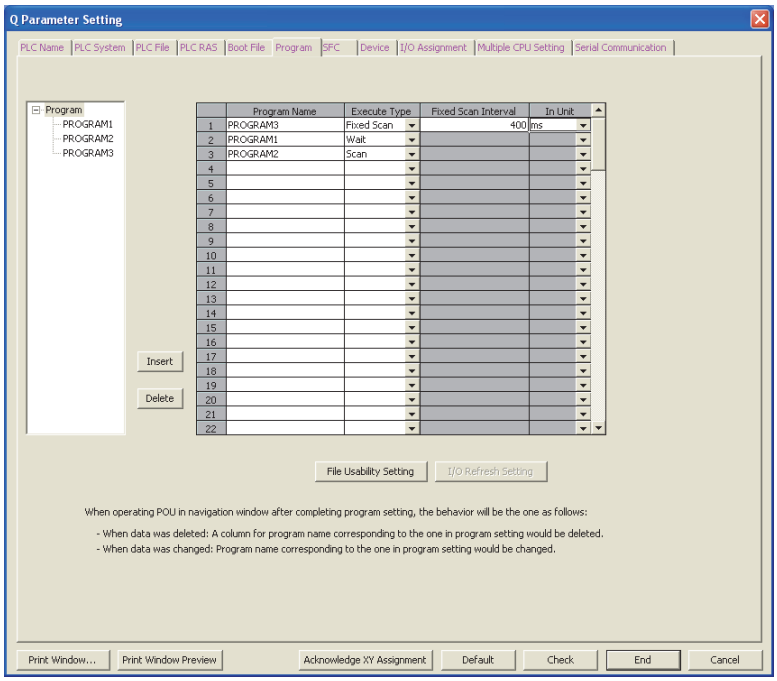
(1) Control by multiple programs dividing one program

The CPU module can store multiple programs divided on the basis of each control unit.
This enables programming of one sequence program by two or more designers.



(2) Settings required for execution of multiple programs

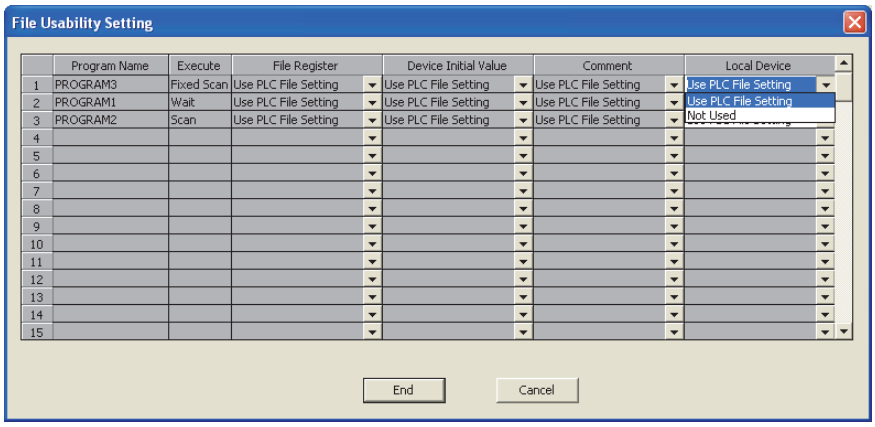
To execute multiple programs, names (file names) and execution conditions of the programs must be set. Set them in the Program tab of the PLC parameter dialog box.



Item		Description
Program Name		Enter the name (file name) of the program to be executed in the CPU module. The programs are executed according to the setting order.
Execute Type		Select an execution type of the program set under "Program Name". The CPU module executes programs whose execute type has been set here according to the setting order.
	Initial execution type ("Initial")	This program is executed only once when the CPU module is powered on or its status is switched from STOP to RUN. (☞ Page 92, Section 2.10.1)
	Scan execution type ("Scan")	This program is executed once in every scan, starting in the scan following the scan in which an initial execution type program is executed. (☞ Page 94, Section 2.10.2)
	Standby type ("Wait")	This program is executed only when its execution is requested. (☞ Page 95, Section 2.10.3)
	Fixed scan execution type ("Fixed Scan")	This program is executed at the intervals specified under "Fixed Scan Interval" and "In Unit". (☞ Page 98, Section 2.10.4) • Fixed Scan Interval Enter the execution interval of fixed scan execution type program. The setting range depends on the setting unit. • When the unit is "ms" : 0.5 to 999.5ms (in increments of 0.5ms) • When the unit is "s" : 1 to 60s (in increments of 1s) • In Unit Select the unit ("ms" or "s") of the fixed scan interval.

(a) File usability setting Note 2.3

For each program, determine whether to use the file specified for the local device in the PLC file tab of the PLC parameter dialog box.



The default is set to "Use PLC File Setting".

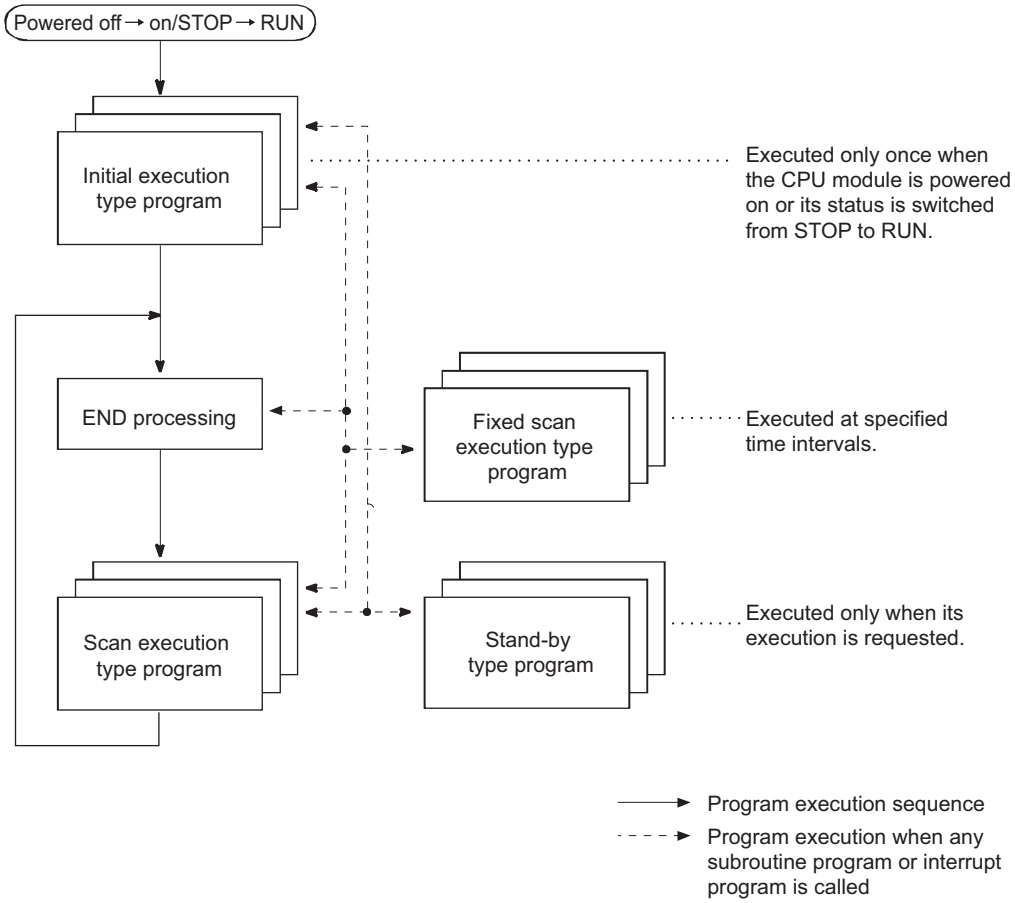
When "Not Used" is selected, data in the local device is not saved or restored when the program execution type is changed.

Note 2.3 Universal

The Q00UJCPU does not support the file usability setting. When using the setting for the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

(3) Program sequence in the CPU module

The following figure shows the program sequence after the CPU module is powered on or its status is changed from STOP to RUN.

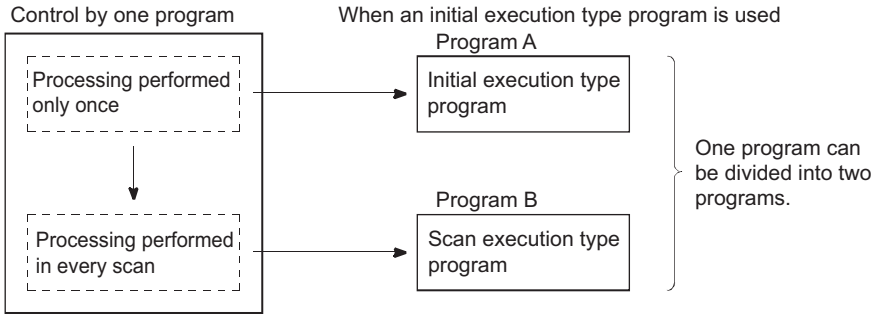


Point

Use initial execution type program, stand-by type program, and fixed scan execution type program as required.

2.10.1 Initial execution type program

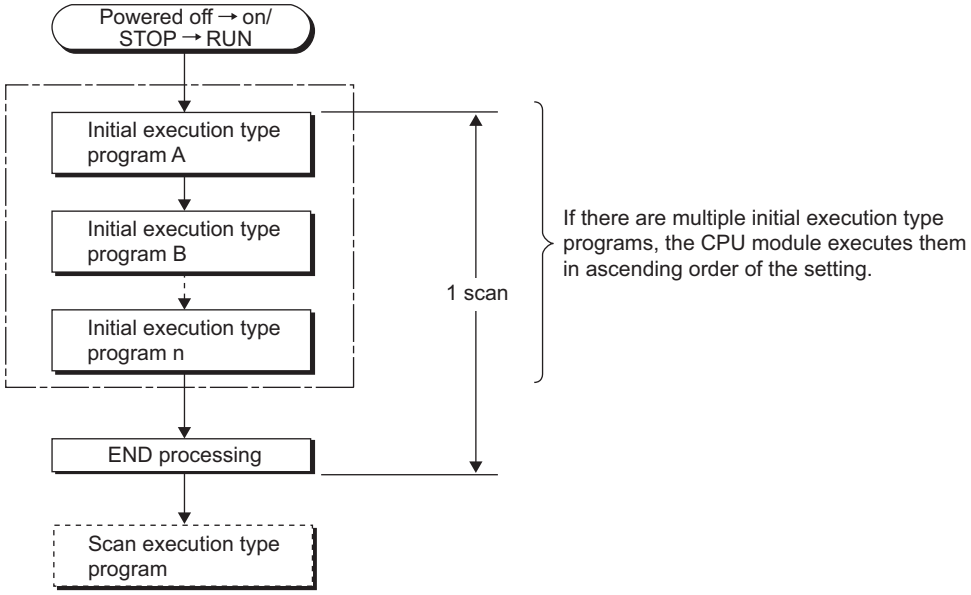
Initial execution type program is executed only once when the CPU module is powered on or its operating status is changed from STOP to RUN. This type of program can be used as a program that need not be executed from the next scan and later once it is executed, like initial processing to an intelligent function module.



(1) Processing

(a) Execution order

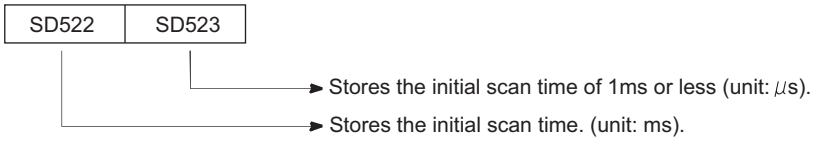
After completion of all the initial execution type program execution, END processing is performed.
In the next scan and later, scan execution type programs are executed.



(b) Initial scan time

Initial scan time is the sum of the execution time of initial execution type program and the END processing time. When multiple programs are executed, the initial scan time will be the sum of the time required for completing all the initial execution type program execution and the END processing time.

- Initial scan time storage location
  The CPU module measures the initial scan time and stores it into the special register (SD522 and SD523).The initial scan time can be checked by monitoring SD522 and SD523.



Ex. If the stored values in SD522 and SD523 are 3 and 400 respectively, the initial scan time is 3.4ms.

- Accuracy and measurement of the initial scan time
  Accuracy of the initial scan time stored in the special register is ±0.1ms. Even if the WDT instruction (instruction that resets the watchdog timer) is executed in the sequence program, the measurement of the initial scan time continues.
- Execution of an interrupt program or fixed scan execution type program
  When an interrupt program or fixed scan execution type program is executed before completion of the initial execution type program execution, the execution time of the executed program will be added to the initial scan time.

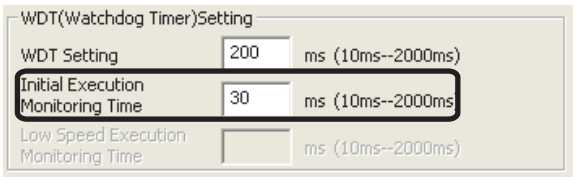
(2) Precautions on programming

Initial execution type programs do not support the instructions that require several scans (instructions with completion device).

Ex. SEND, RECV, and similar instructions

(3) Initial execution monitoring time setting

Initial execution monitoring time is a timer for monitoring initial scan time. Set a time value in the PLC RAS tab of the PLC parameter dialog box. The setting range is 10 to 2000ms (in increments of 10ms). No default value is set.



(a) When the initial scan time exceeds the preset initial execution monitoring time

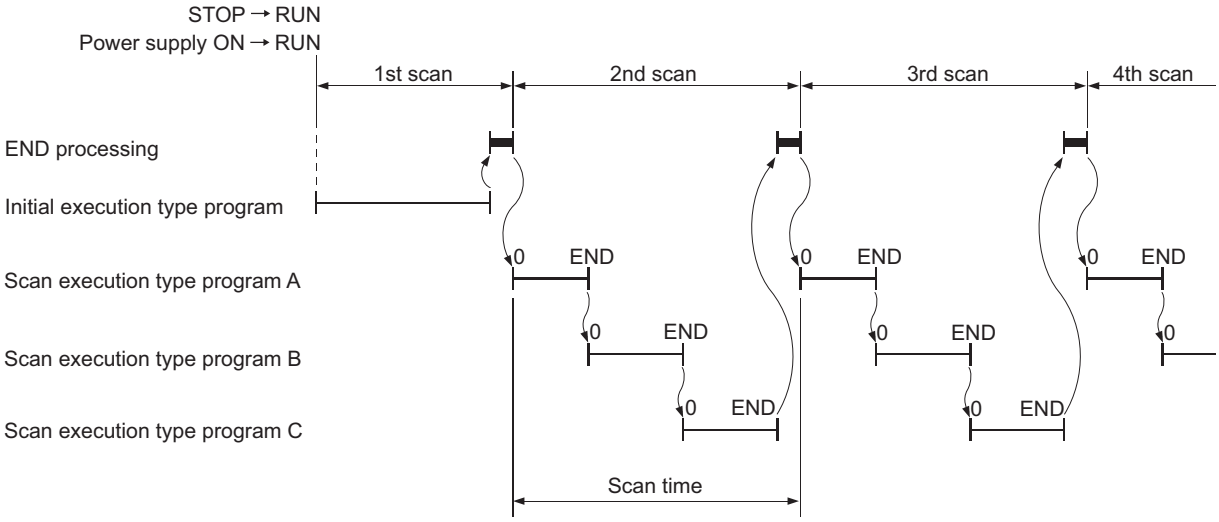
"WDT ERROR" occurs and the CPU module stops program operations.
Set a time value so that the initial execution monitoring time becomes longer than actual initial scan time.

Point

An error of the measurement value is 10ms for the initial execution monitoring time setting.
If the initial execution monitoring time (t) parameter is set to 10ms, "WDT ERROR" occurs when actual initial scan time is within the range of 10ms < t < 20ms.

2.10.2 Scan execution type program

Scan execution type program is executed once in every scan, starting in the next scan of which the initial execution type program is executed and later.



When multiple scan execution type programs are executed, the scan time will be the time required for completing all the scan execution type program execution. If an interrupt program or fixed scan execution type program is executed, execution time of the executed program will be added to the scan time.

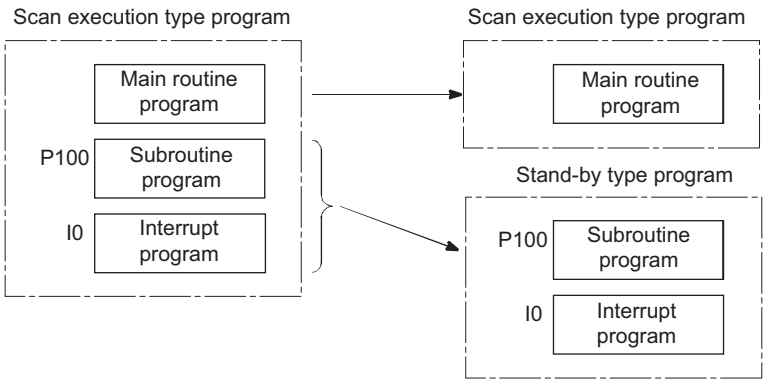
2.10.3 Stand-by type program

Stand-by type program is executed only when its execution is requested. This type of program can be changed to any desired execution type by a sequence program instruction.

(1) Application

(a) Program library

Stand-by type program is used as a program library, a collection of subroutine programs and/or interrupt programs, and managed separately from a main routine program.
Multiple subroutine programs and/or interrupt programs can be created and managed in a single stand-by type program.



(b) Program type change

Stand-by type program is used to create and store programs available in all systems. Only required programs will be executed. For example, a program preset as a stand-by ("Wait") type program in the PLC parameter dialog box can be changed to a scan execution type program and executed in the sequence program.

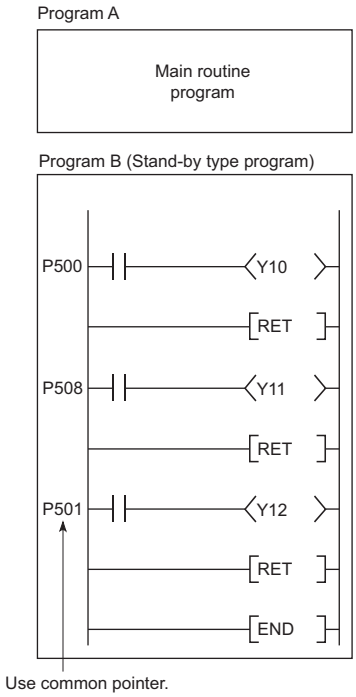
(2) Execution method

Execute stand-by type programs in either of the following methods.

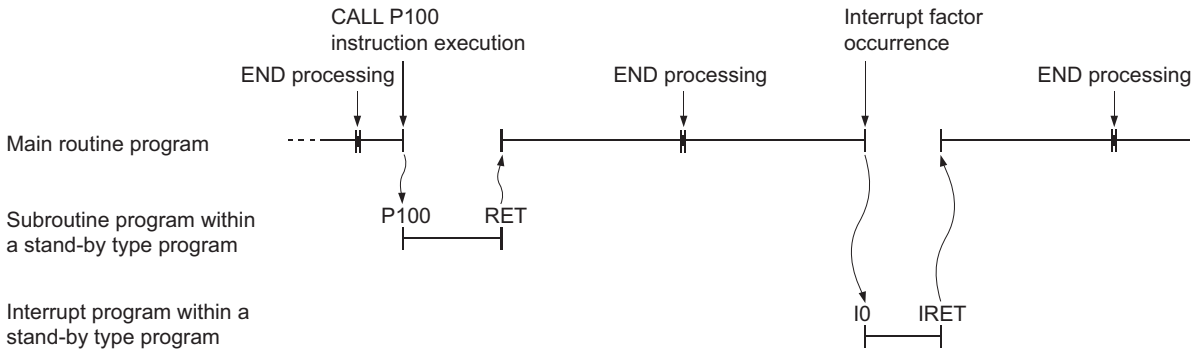
- Create subroutine and/or interrupt programs in a stand-by type program and call them using a pointer or when an interrupt occurs.
- Change a stand-by type program to any other execution type using instructions.

(a) Creating subroutine and/or interrupt programs in a single stand-by type program

When creating subroutine and/or interrupt programs in a single stand-by type program, start the program from the step 0. The FEND instruction used in creation of a subroutine or interrupt program is not required after a main routine program.



After execution of the standby type program, the CPU module re-executes the program that called a program in the standby type program. The following figure shows the operation when the subroutine and interrupt programs in the standby type program are executed.



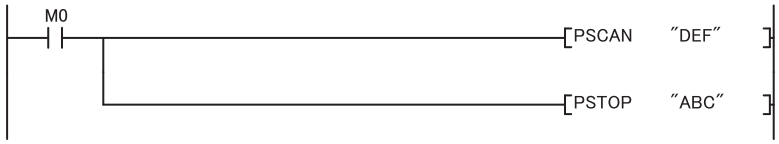
Point

- For restrictions on programming of subroutine and interrupt programs, refer to the following.
  - Subroutine program: Page 69, Section 2.4.3
  - Interrupt program: Page 82, Section 2.9
- Use common pointers. (Page 411, Section 4.10.2) If local pointers are used, subroutine programs in a stand-by type program cannot be executed from any other program.

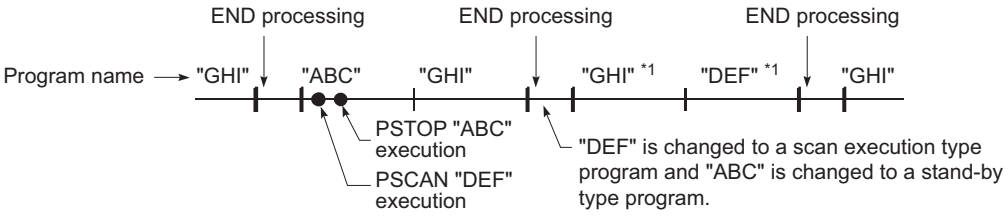
(b) Changing the program execution type using instructions

Use the PSCAN, PSTOP, or POFF instruction to change a program execution type. (☞ Page 102, Section 2.10.5)

Ex. • The PSCAN instruction changes the program "DEF" to a scan execution type program.
• The PSTOP instruction changes the program "ABC" to a standby type program.



The program execution type is changed in END processing. Therefore, the execution type will not be changed in the middle of program execution. If different types are set to the same program in the same scan, the program will be changed to the type specified by the last instruction executed.



*1 The programs "GHI" and "DEF" are executed in the order set in the Program tab of the PLC parameter dialog box.

(3) Precautions on programming

(a) Unavailable devices

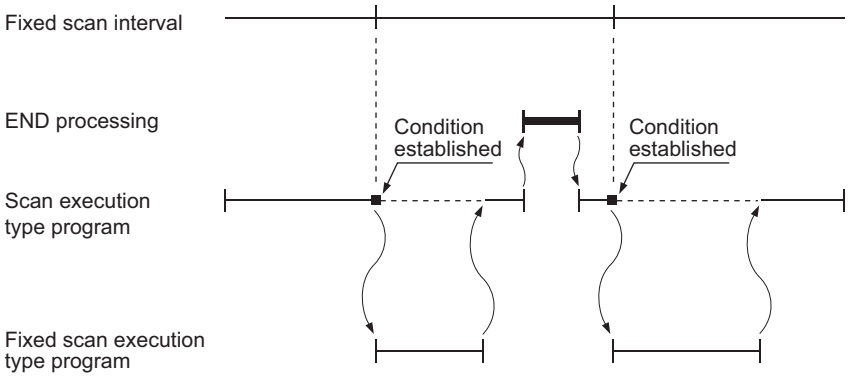
Unavailable devices depend on the program type (subroutine program or interrupt program) or the execution type changed by an instruction.

(b) Use of local devices

For execution of a subroutine program using a local device, refer to Page 422, Section 6.2.

2.10.4 Fixed scan execution type program

Fixed scan execution type program is a program executed at specified time intervals. This type of programs, unlike interrupt programs, can be interrupted in units of files without interrupt pointers or the IRET instruction. For the restrictions on programming, refer to Page 84, Section 2.9 (1) (b). (The restrictions on programming are the same as those for interrupt programs.)



Point

To execute a fixed scan execution type program, execute the EI instruction in the initial execution type program or scan execution type program to enable interrupts.

(1) Processing

(a) When two or more fixed scan execution type programs exist

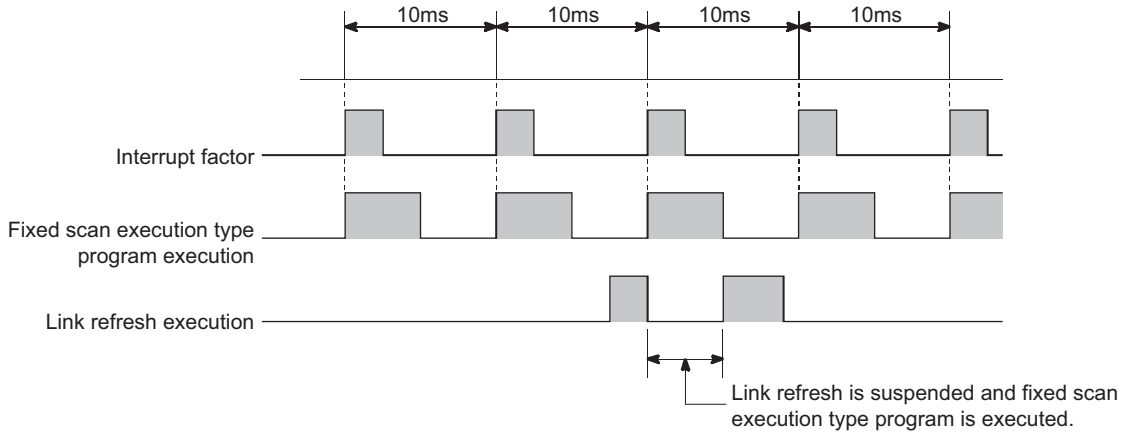
Each fixed scan execution type program is executed at specified time intervals.
If two or more fixed scan execution type programs reach the specified time at the same timing, programs will be executed in ascending order of the numbers set in the Program tab of the PLC parameter dialog box.

(b) When both fixed scan execution type program and interrupt program exist

When a fixed scan execution type program and an interrupt program (I28 to I31) reach the specified time at the same timing, the interrupt program will be given priority.

(c) When the execution condition is established during link refresh

The link refresh is suspended and a fixed scan execution type program is executed.
Even if the Block data assurance per station setting is enabled in the CC-Link IE or MELSECNET/H network, this setting does not work when a device set as a refresh target is used in the fixed scan execution type program.
In the fixed scan execution type program, do not use any refresh target device.

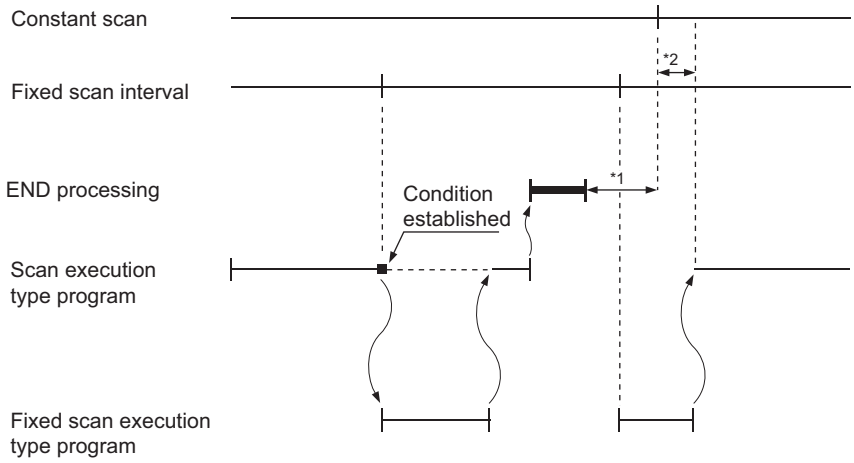


For the Block data assurance per station setting, refer to the following.

Manual for each network module

(d) When the execution condition is established during END processing

When the execution condition is established during the constant scan execution or the waiting time of the END instruction, a fixed scan execution type program is executed.



*1 Waiting time
*2 If processing is not completed within the waiting time, the scan time increases.

(2) Processing at program execution type change

For how to save and restore data in the index register when the program execution type is changed, refer to Page 87, Section 2.9 (3). (The method is the same as that for interrupt programs.)

(3) Precautions

(a) Execution interval of a fixed scan execution type program

Execution interval of a fixed scan execution type program may increase from the preset interval depending on the time set for disabling interrupts by the DI instruction (interrupt disabled time).
If the interrupt disabled time by the DI instruction becomes too long, use an interrupt program by fixed scan interrupt (I28 to I31) instead of a fixed scan execution type program.

Highest common factor of fixed scan execution interval[*1] < Interrupt disabled time ••• Condition 1)

*1 This is the highest common factor of execution interval set to multiple fixed scan execution type programs

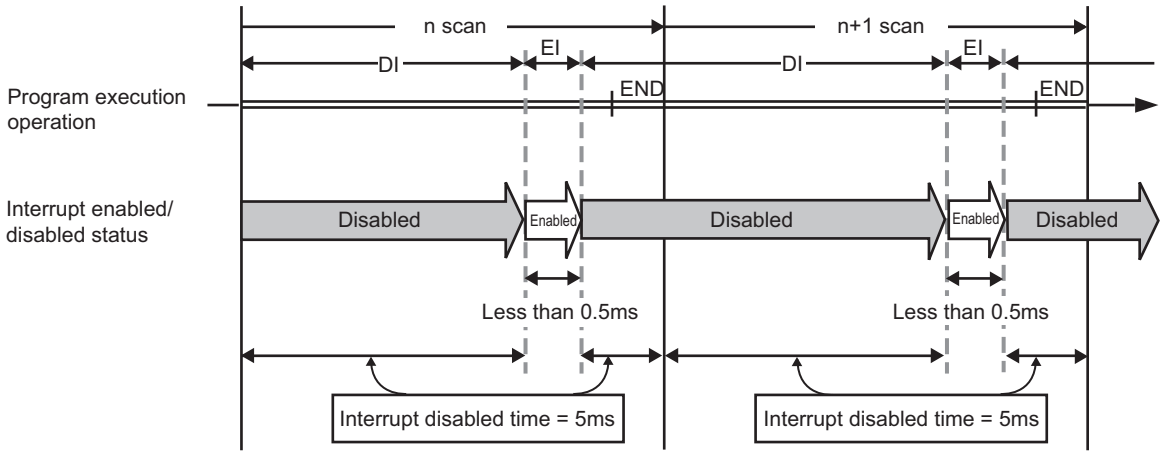
When the condition 1) is satisfied, the actual execution interval of a fixed scan execution type program may increase from the preset interval by the time shown in the expression below.

$$\frac{\text{Interrupt disabled time}}{\text{Highest common factor of fixed scan execution interval}} \times \text{Fixed scan execution interval set to the corresponding program}$$

The following shows an example of the increase in execution time of a fixed scan execution type program.

Ex.
- Fixed scan execution interval ••• 10ms, 5ms, 1ms, 0.5ms
- Highest common factor of fixed scan execution interval ••• 0.5ms
- Interrupt disabled time (DI) ••• 5ms
  (Interrupt enabled time (EI) ••• less than 0.5ms)

With the settings above, the condition 1) will be 0.5ms < 5ms.



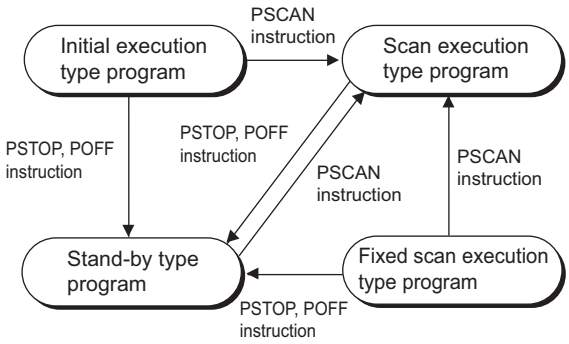
The execution time of a fixed scan execution type program whose execution interval is set to 10ms increases 100ms (5 ÷ 0.5 × 10 = 100) at the most.

2.10.5 Changing the program execution type

(1) Changing the execution type using instructions

(a) Instructions used to change the execution type

The execution type of sequence programs can be changed using instructions even during execution.
Use the PSCAN, PSTOP, or POFF instruction to change the execution type.



Execution type before change	Instruction		
	PSCAN	PSTOP	POFF
Scan execution type	Remains unchanged.	Changes to the stand-by type.	Turns off outputs in the next scan. Changes to the stand-by type in two scans later.
Initial execution type	Changes to the scan execution type.	Changes to the stand-by type.	Turns off outputs in the next scan. Changes to the stand-by type in two scans later.
Stand-by type	Changes to the scan execution type.	Remains unchanged.	No processing
Fixed scan execution type	Changes to the scan execution type.	Changes to the stand-by type.	Turns off outputs in the next scan. Changes to the stand-by type in two scans later.

Point

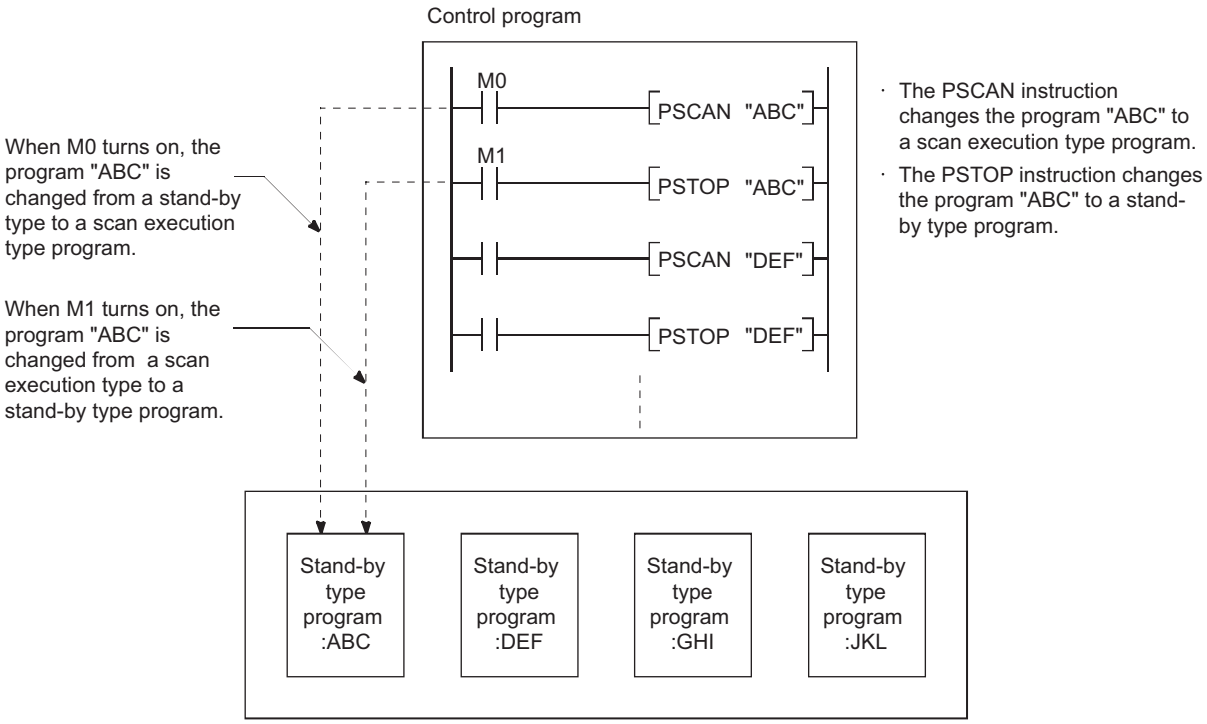
Once the fixed scan execution type program is changed to another execution type, the type cannot be returned to the fixed scan execution type.

(b) Execution type change example

In a control program, a standby type program matching the preset condition is changed to a scan execution type program in the course of program execution.

An unused scan execution type program can also be changed to a standby type program.

The following figure shows an example where the execution type of the standby type programs "ABC", "DEF", "GHI", and "JKL" are changed in the control program.

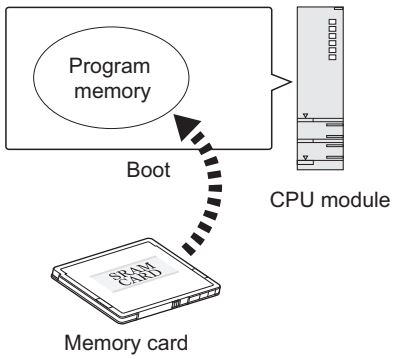


2.11 Boot Operation Note 2.4

This section describes methods for executing the program stored in a memory card or SD memory card.

(1) Executing the program in a memory card or SD memory card

To execute the program in a memory card or SD memory card, boot the program into the program memory. To execute the boot operation, set the boot target program in the Boot file tab of the PLC parameter dialog box.
(☞ Page 105, Section 2.11 (3))
The program set in parameter is booted into the program memory when the CPU module is powered off and then on or is reset.



(2) Bootable files, transfer source, and transfer destination

The following table lists combinations of bootable file, transfer source, and transfer destination.

○ : Bootable, × : Cannot be booted.

File type	Transfer source	Transfer destination	
		Program memory	Standard ROM
Parameter*1	Memory card, SD memory card*2	○	○
Sequence program		○	×
Device comment		○	○
Initial device value		○	○
Label program		○	○

*1 The intelligent function module parameter is included.
*2 If the boot operation is performed while the CPU module is locked with a security key, "BOOT ERROR" (error code: 2214) occurs.

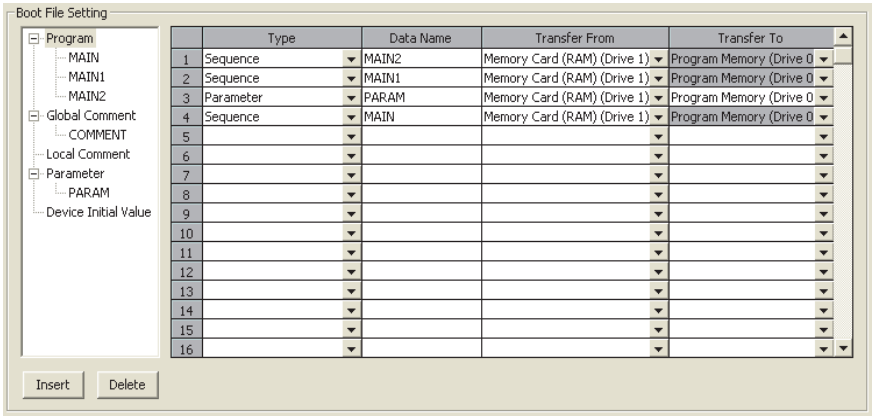
Note 2.4 Universal

The Q00UJCPU, Q00UCPU, and Q01UCPU do not support the boot operation.

(3) Procedure before boot operation

The following is the procedure to store files to be booted in a memory card or SD memory card before the boot operation.

1. **Create a program.**
2. **Configure the setting for a boot operation.**
   Set the names of files to be booted to the program memory in the Boot File tab of the PLC parameter dialog box.



3. **Insert a memory card or SD memory cart into the CPU module.**
4. **Write the parameter file and other files set in parameter to the memory card or SD memory card.**
5. **Execute the program.**
   Power on or reset the CPU module. The BOOT LED turns on after a boot operation from the specified memory is completed.
6. **Check that the boot operation has completed successfully.**
   The following status indicates normal completion of boot operation.
   - The BOOT LED turns on.
   - The special relay (SM660) turns on.
   - The data written in the transfer source memory and the one in the program memory are compared and confirmed their match by "Verify with PLC".

(4) Stopping the boot operation

To stop boot operation and operate the CPU module by the parameters and program files written to the program memory, perform the following operations.

1. **Remove the memory card or SD memory card, and write parameters without boot file setting to the program memory.**
2. **Power off and then on the programmable controller or rest the CPU module.**

(5) Precautions

(a) Storage location of parameters

- Store the parameter file set in the Boot file tab of the PLC parameter dialog box to the memory card or SD memory card. If it is stored in the program memory or standard ROM, the CPU module ignores the settings. (☞ Page 42, Section 2.1.2)
- The CPU module operates using the parameters in the program memory, not those in the memory card or SD memory card, if the following conditions are both met.
  1) A parameter file exists in the program memory.
  2) The parameter file in a memory card or SD memory card is not set for boot target in parameter.

(b) Online change during boot operation

If the program in the program memory is overwritten online while the boot operation is being performed, the change is not reflected to the program in the memory card or SD memory card.
Write the same program file to the memory card or SD memory card.

(c) Maximum number of settable boot files

Set the maximum number of settable boot files in the Boot file tab of the PLC parameter dialog box so that it may be the same with the number of files storable to the program memory.
However, the number of boot files decreases by one when:

- The parameter file (with boot settings) stored in the memory card or SD memory card is booted.
- A heading is set.

(d) Boot operation when the ATA card is used

When data are booted from the ATA card, the processing time of maximum 200ms may be required per 1K step (4K bytes).

(e) When data in the program memory are changed after the CPU module is powered off and then on or is reset

If the program memory data are changed after the sequence program is written to the program memory and the CPU module is powered off and then on or is reset, a boot operation may be active.
While the BOOT LED on the front of the CPU module is on, the boot operation is active. Refer to Page 105, Section 2.11 (4) and stop the boot operation.

(f) File size before and after the boot operation

The size unit of a file stored in each memory differs. (☞Page 50, Section 2.1.3 (4))
Note that files transferred from the memory card or SD memory card to the program memory differ in size before and after the transfer.

(g) Program written to the memory card or SD memory card

Set the programmable controller type (model name of the CPU module) for the program written to the memory card (program set in the Boot file tab) and the model name of the CPU module to be booted to the same.
Set the type (model) of the CPU module that actually performs the boot operation to the program (with boot settings) written to the memory card or SD memory card.

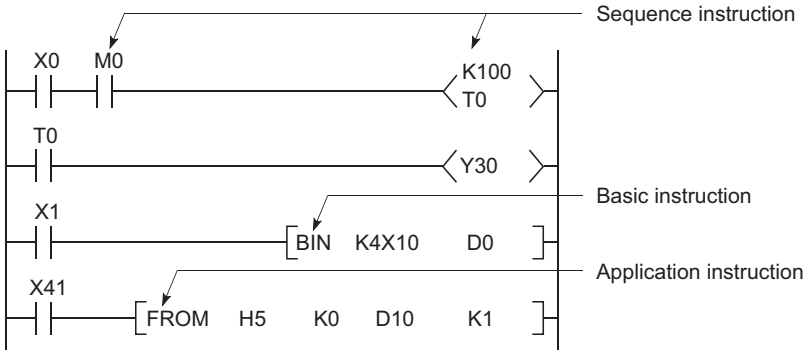
2.12 Programming Language

Programming tools support the following programming languages.

- Ladder
- Structured text
- SFC
- Structured ladder

(1) Ladder

A graphical programming language used for contacts and coils. For a project with a label, the inline ST function can be used in the ladder editor which allows a user to edit structured text programs.



Point

Data indicating the execution status of an operation in a sequence program step is referred to as "signal flow".

(2) Structured text

A text language such as C language, and is preferred by computing engineers.

(3) SFC

A graphical programming language where the execution order and conditions are clearly defined for the program.

(4) Structured ladder

A graphical programming language that is used contacts and coils.

Remark

For the projects that support these programming languages, refer to the following.

Manual for the programming tool used

2.13 Communications with Intelligent Function Modules

The intelligent function module allows the CPU module to process analog quantity and high-speed pulses that cannot be processed by the I/O modules. The following is some of the intelligent function modules.

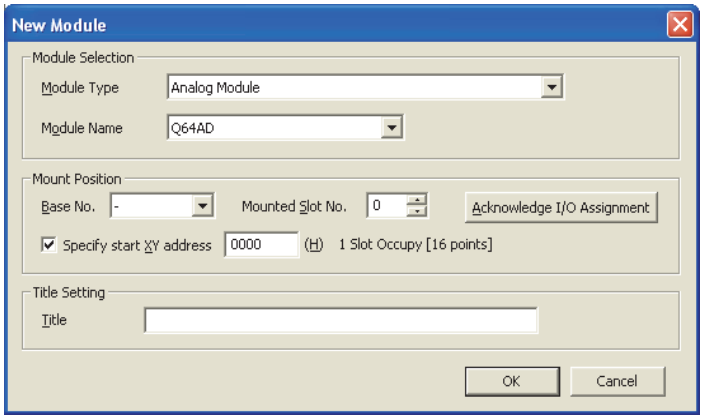
- Serial communication module
- Analog module

The intelligent function module is equipped with a memory (buffer memory) to store the data taken in from or output to external devices. The CPU module writes or reads data to or from the buffer memory of the intelligent function module.

(1) Setting method of intelligent function module parameters

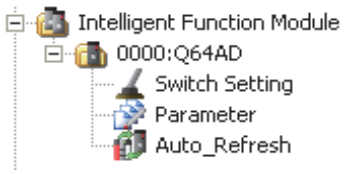
Open the "New Module" dialog box.

Project window ⇨ [Intelligent function module] ⇨ Right-click ⇨ [New Module...]



Item		Description
Module Selection	Module Type	Select a type of the CPU module.
	Module Name	Select a model name of the CPU module.
Mount Position	Base No.	Select a base No. where the CPU module is connected.
	Mounted Slot No.	Select a slot No. where the CPU module is connected.
	Acknowledge I/O Assignment	The I/O assignment settings in the PLC Parameter dialog box can be checked.
	Specify start X/Y address	Enter the start I/O number.
Title Setting	Title	Enter a title.

Upon completion of the setting above, parameters for the intelligent function module appear in the "Project" window.



To set the intelligent function module parameters, refer to the following.

Manual for the intelligent function module used

(2) Communications with the FROM and TO instructions

The FROM instruction stores data read from the buffer memory of the intelligent function module to the specified device.

The TO instruction writes data stored in the specified device to the buffer memory of the intelligent function module.

For details on the FROM and TO instructions, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

(3) Communications using the intelligent function module device

The intelligent function module device represents the buffer memory of the intelligent function module as one of the CPU module devices. (Page 384, Section 4.5.1)

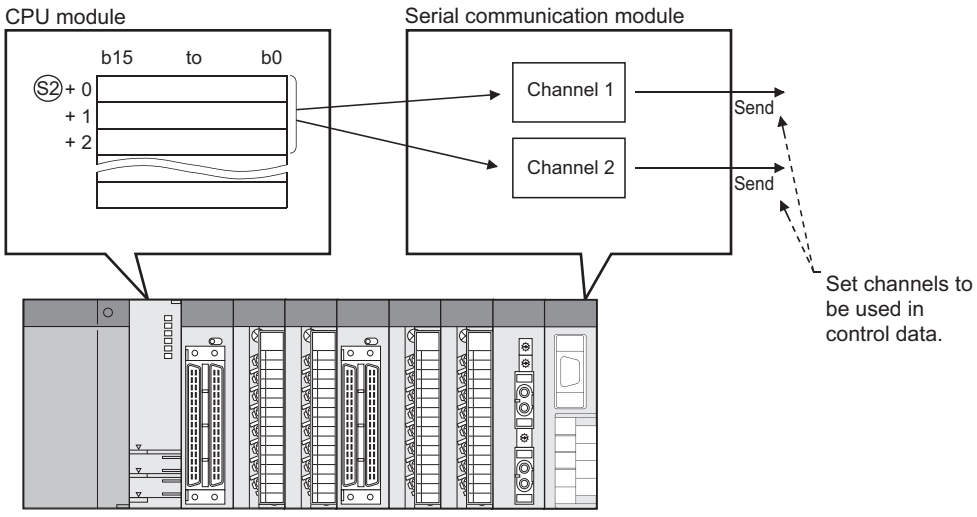
The difference from the FROM and TO instructions is that, with this device, both reading and writing data from and to the intelligent function module can be processed with one instruction.

(4) Communications using the intelligent function module dedicated instruction

This instruction enables easy programming for the use of functions of the intelligent function module.

Ex. Serial communication module dedicated instruction (OUTPUT instruction)

The OUTPUT instruction allows communications with external device by nonprocedural protocol regardless of the buffer memory address of the serial communication module.



(a) Processing of the intelligent function module dedicated instruction

When using multiple intelligent function module dedicated instructions to one intelligent function module, execute the dedicated instructions one by one after the completion device turns on. This completion device turns on for one scan when an instruction is completed. If the CPU module status is changed from RUN to STOP before the completion device turns on, the completion device does not turn on until one scan after the next RUN of CPU module.

For details on the intelligent function module dedicated instructions and the completion device, refer to the following.

Manual for the intelligent function module used

2.14 Access to the AnS/A Series Special Function Modules Note 2.5

(1) Effect of high-speed access to the special function module

Processing time in the Q series CPU module has been speeded up so that the scan time is shortened.
If the FROM or TO instruction is frequently executed to a special function module in short scan, processing in the special function module may not be completed correctly.

(2) Measures for high-speed access to the special function module

- Adjust execution intervals of the FROM and TO instructions to the processing time and conversion time using the timer and constant scan of the CPU module.
- Adjust execution intervals of the FROM and TO instructions using SM415 (2n (ms) clock) or SD415 (2nms clock setting).
  If SM415 is used as an interlock for the FROM or TO instruction, the instruction is executed every 120ms since the initial value of SD415 has been set to "30".

Point

When changing the SM415 clock, store the changed value in SD415.
For details of SM415 and SD415, refer to the following.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Note 2.5 Universal

Before using the AnS/A series-compatible special function modules, check the version of the CPU module used.
(☞ Page 466, Appendix 2)

PART 2 FUNCTIONS

In this part, functions of the CPU module are described.

CHAPTER 3 FUNCTIONS

This chapter describes the functions of the Universal model QCPU.

3.1 Function List

The following table lists the functions of the Universal model QCPU.

○ : Supported, △ : Partly supported, × : Not supported

Function	Description	Q00UJCPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H)CPU	QnUDE(H)CPU	QnUDVCPU, QnUDPVCPU	Reference
Boot operation	Transfers data stored in a memory card or SD memory card to the program memory or the standard ROM at power-on or reset.	×	×	○	○	○	○	Page 104, Section 2.11
Constant scan	Executes a program in a set time interval regardless of its scan time.	○	○	○	○	○	○	Page 119, Section 3.2
Latch function	Holds the device data even when the CPU module is powered off and on or reset.	○	○	○	○	○	○	Page 122, Section 3.3
Output status selection when the status changed from STOP to RUN	Selects the output (Y) status (outputting the same status prior to STOP or clearing the status) when the operating status of the CPU module is switched from STOP to RUN.	○	○	○	○	○	○	Page 125, Section 3.4
Clock function	Reads the internal clock data of the CPU module to use it for time management.	○	○	○	○	○	○	Page 127, Section 3.5
Remote RUN/STOP	Runs or stops the program operations in the CPU module externally.	○	○	○	○	○	○	Page 131, Section 3.6.1
Remote PAUSE	Stops the program operations in the CPU module externally, holding the status of outputs (Y).	○	○	○	○	○	○	Page 134, Section 3.6.2
Remote RESET	Resets the CPU module externally when the CPU module is in the STOP status.	○	○	○	○	○	○	Page 136, Section 3.6.3
Remote latch clear	Clears the latch data in the CPU module when the CPU module is in the STOP status.	○	○	○	○	○	○	Page 137, Section 3.6.4
Input response time selection	Selects input response time values for the Q series-compatible input modules, I/O combined modules, high-speed input modules, and interrupt modules.	○	○	○	○	○	○	Page 139, Section 3.7
Error time output mode setting	Sets whether to clear or retain the output to the Q series-compatible output modules, I/O combined modules, intelligent function modules, and interrupt modules at the time of a stop error of the CPU module.	○	○	○	○	○	○	Page 141, Section 3.8

Function	Description	Q00UJCPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H)CPU	QnUDE(H)CPU	QnUDVCPU, QnUDPVCPU	Reference
H/W error time PLC operation mode setting	Sets whether to stop or continue operations in the CPU module when a hardware error has occurred in an intelligent function module or interrupt module.	○	○	○	○	○	○	Page 142, Section 3.9
Intelligent function module switch setting	Makes settings for the intelligent function modules and interrupt modules. (Refer to manuals of intelligent function modules and interrupt modules for setting details.)	○	○	○	○	○	○	Page 143, Section 3.10
Monitor function	Reads the status of programs and devices in the CPU module using a programming tool.	○	○	○	○	○	○	Page 145, Section 3.11
Monitor condition setting	Specifies the monitoring timing of the CPU module with device condition or step number.	×	×	△ *1	△ *1	○	○	Page 146, Section 3.11.1
Local device monitor/test	Monitors and/or tests the local devices of the specified program using a programming tool.	×	○	○	△ *1	○	○	Page 151, Section 3.11.2
External input/output forced on/off	Forcibly turns on/off the external input/output of the CPU module using a programming tool.	○	○	△ *1	△ *1	○	○	Page 154, Section 3.11.3
Executional conditioned device test	Changes a device value within the specified step of a sequence program.	○	○	△ *1	△ *1	○	○	Page 159, Section 3.11.4
Online change	Writes programs when the CPU module is in the RUN status.	○	○	○	○	○	○	Page 168, Section 3.12
Program monitor list	Displays the scan time and execution status of the program being executed.	○	○	○	○	○	○	Page 180, Section 3.13.1
Interrupt program monitor list	Displays the number of executions of interrupt programs.	○	○	○	○	○	○	Page 180, Section 3.13.2
Scan time measurement	Measures the execution time of the area specified by the steps in a program.	○	○	△ *1	△ *1	○	○	Page 181, Section 3.13.3
Sampling trace function	Continuously samples the specified device data at a preset timing.	×	○	○	○	○	○	Page 184, Section 3.14
Debug function from multiple programming tools	Enables simultaneous debugging by multiple programming tools.	○	○	○	○	○	○	Page 189, Section 3.15
Watchdog timer	Monitors operational delays caused by hardware failure or program error of the CPU module.	○	○	○	○	○	○	Page 193, Section 3.16
Self-diagnostic function	Self-diagnoses the CPU module to see whether an error exists or not.	○	○	○	○	○	○	Page 195, Section 3.17
Error history	Stores the result of self-diagnostics to the memory as error history data.	○	○	○	○	○	○	Page 206, Section 3.18

Function	Description	Q00UJCPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H)CPU	QnUDE(H)CPU	QnUDVCPU, QnUDPVCPU	Reference
Security function	Protects data in the CPU module against tampering and theft by unauthorized persons.	○	○	○	○	○	○	Page 207, Section 3.19
Password registration	Prohibits writing/reading data to/from each file in the CPU module using a programming tool.	○	○	○	○	○	×	Page 207, Section 3.19.1
File password 32	Prohibits writing/reading data to/from each file in the CPU module using a programming tool. Sets a read password and write password for each file.	×	×	×	×	×	○	Page 209, Section 3.19.2
File access control by security key	Prevents unauthorized access to the files in the CPU module by writing a security key to the module. (The CPU module is locked with a security key.)	×	×	×	×	×	○	Page 214, Section 3.19.3
Remote password	Prevents unauthorized access from external devices.	○	○	○	○	○	○	Page 219, Section 3.19.4
Block password	Prevents access to program contents by setting a block password for each POU.	○	○	○	○	○	○	GX Works2 Version 1 Operating Manual (Common)
LED indication	Displays the operating status of the CPU module with LEDs on the front of the module.	○	○	○	○	○	○	Page 222, Section 3.20
LED indication priority	Sets whether to indicate an error with LED according to the priority of each error.	○	○	○	○	○	○	Page 223, Section 3.20.2
High-speed interrupt function	Executes an interrupt program at fixed intervals of 0.1 to 1.0ms using the high-speed interrupt pointer (I49).	×	×	×	×	×	○	Page 225, Section 3.21
Interrupt from intelligent function module	Executes an interrupt program at the time of interrupt request from the intelligent function module.	○	○	○	○	○	○	Page 232, Section 3.22
Serial communication function	Communicates data using the MC protocol by connecting the RS-232 interface of the CPU module and a personal computer or HMI from other companies using an RS-232 cable.	○	○	△ *1	△ *1	×	○	Page 233, Section 3.23
Service processing setting	Specifies the service processing count or time to be executed in END processing.	○	○	○	○	○	○	Page 241, Section 3.24.1
Initial device value	Registers data used in a program to the device of the CPU module or the buffer memory of the intelligent function module and special function module without a program.	○	○	○	○	○	○	Page 247, Section 3.25
Battery life-prolonging function	Extends the life of a battery by holding only clock data using the battery.	○	○	○	○	○	○	Page 250, Section 3.26
Memory check function	Checks whether data in the memories of the CPU module are not changed due to excessive electric noise.	○	○	○	○	○	○	Page 251, Section 3.27

Function	Description	Q00UJCPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H)CPU	QnUDE(H)CPU	QnUDVCPU, QnUDPVCPU	Reference
Program cache memory auto recovery function	Restores the error location automatically by using data in the program memory, which are stored in the flash ROM, when the memory check function detects an error in the program cache memory.	△ *1	△ *1	△ *1	△ *1	△ *1	○	Page 252, Section 3.28
Latch data backup to standard ROM	Backs up latch data such as device data and error history to the standard ROM without using a battery.	○	○	○	○	○	○	Page 254, Section 3.29
Writing/reading device data to/from standard ROM	Writes/reads device data to/from the standard ROM.	○	○	○	○	○	○	Page 259, Section 3.30
CPU module change function with memory card	Backs up all the data (only the file register files and latch-target device data) in a CPU module to a memory card or SD memory card. The data backed up can be restored to a replaced CPU module. *2	×	×	△ *1	△ *1	△ *1	○	Page 260, Section 3.31
CPU module data backup/restoration function	Backs up data such as program files, a parameter file, and device data including file registers in a CPU module to an SD memory card. The data backed up can be restored as necessary. *2	×	×	×	×	×	△ *1	Page 276, Section 3.32
Module model name read	Reads the model name of a module on a base unit.	△ *1	△ *1	△ *1	△ *1	△ *1	○	Page 297, Section 3.33
Module error collection	Collects errors occurred in the connected intelligent function modules in the CPU module.	△ *1	△ *1	△ *1	△ *1	△ *1	○	Page 298, Section 3.34
Local device batch read function	Batch-reads local device data in the CPU module and stores them in a CSV file.	×	△ *1	△ *1	△ *1	△ *1	○	Page 302, Section 3.35
Send points extension function (CC-Link IE Controller Network module)	Extends the maximum number of link points per CC-Link IE Controller Network module.	△ *1	△ *1	△ *1	△ *1	△ *1	○	Page 304, Section 3.36
Write-protect function for device data (from outside the CPU module)	Disables device data writing (including the file register) from outside the CPU module such as the programming tool, GOT, SLMP/MC protocol, and FTP to the write-protected range set in the parameter.	×	×	×	×	×	△ *1	Page 306, Section 3.37
Operation history function	Saves the operation information of device data writing/writing of files from outside the CPU module such as the programming tool, GOT, SLMP/MC protocol, and FTP into the CPU module as an operation history file, and displays it in the programming tool.	×	×	×	×	×	△ *1	Page 314, Section 3.38

Function	Description	Q00UJCPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H)CPU	QnUDE(H)CPU	QnUDVCPU, QnUDPVCPU	Reference
Built-in Ethernet function	Enables MC protocol communications and the following functions by using built-in Ethernet ports.	×	×	×	×	○	○	QnUCPU User's Manual (Communication via Built-in Ethernet Port)
File transfer function (FTP)	Enables the use of FTP (File Transfer Protocol) server function, which transfers files between the CPU module and external devices. External devices with a FTP client function can directly access to the files in the CPU module.	×	×	×	×	○	○	
Predefined protocol function	Sends and receives packets predefined by using GX Works2, enabling easy communications with external devices (such as measuring instruments and bar code readers)	×	×	×	×	×	△ *1	
Socket communication function	Communicates data (using TCP/UDP) with external devices connected on the Ethernet network. The function is executed by dedicated instructions.	×	×	×	×	△ *1	○	
Simple PLC communication function *1*2	Allows data communications between specified devices at the specified timing just by doing simple settings from a programming tool.	×	×	×	×	×	△ *1	
IP address change function	Changes an IP address of a built-in Ethernet port by storing it in the special relay and special register, not in the built-in Ethernet port setting parameter.	×	×	×	×	△ *1	○	
IP packet transfer function	Communicates with the following devices that correspond to IP address specified via a CC-Link IE Controller Network module or CC-Link IE Field Network module, using a protocol such as the FTP or HTTP via a built-in Ethernet port from an Ethernet device such as a personal computer. • External devices on the CC-Link IE Controller Network or CC-Link IE Field Network • External devices on the Ethernet network, which are connected through the built-in Ethernet ports	×	×	×	×	△ *1	○	
Reading/writing device data from/to the CPU module on another station by specifying an IP address	Reads/writes device data from/to the CPU module on another station by using the dedicated instructions.	×	×	×	×	×	△ *1	
SLMP frame send instruction	Sends MC protocol messages (QnA-compatible 3E frame and 4E frame) from the CPU module to external devices connected on the Ethernet network.	×	×	×	×	×	△ *1	

Function	Description	Q00UJCPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H)CPU	QnUDE(H)CPU	QnUDVCPU, QnUDPVCPU	Reference
Writing/reading data to/from refresh devices with the specified station number	Writes/reads data by specifying the station number of the target station, without considering the assignment of refresh devices.	×	×	×	△ *1	△ *1	△ *1	MELSEC-Q/L Programming Manual (Common Instruction)
Data logging function	Collects data from the specified device of a CPU module at a specified timing. The data logging file can be transferred from a CPU module to the FTP server using the data logging file transfer function.	×	×	×	×	×	○	QnUDVCPU/LCPU User's Manual (Data Logging Function)
iQ Sensor Solution function	The functions can be used in the iQ Sensor Solution.	×	×	×	×	×	△ *1	Page 333, Section 3.39, iQ Sensor Solution Reference Manual
Automatic detection of connected device	Detects devices supporting iQ Sensor Solution connected to the CPU module, and automatically displays them on "List of devices" and "Device map area" using a programming tool.							
System configuration check	Compares the system configuration information displayed on a programming tool with the actual system configuration, and checks if they match.							
Communication setting reflection	Reflects the communication settings (such as IP addresses) of devices supporting iQ Sensor Solution on "Device map area" to the devices connected over Ethernet in the system.							
Sensor parameter read/write	Reads/writes parameters from/to devices supporting iQ Sensor Solution.							
Monitoring	Monitors the current values (such as measurement values and input/output values), status (error existence), and error information of devices supporting iQ Sensor Solution graphically using a programming tool.							
Data backup/restoration	Backs up setting data (such as parameters) in a device supporting iQ Sensor Solution to an SD memory card. The data backed up can be restored as necessary.*2							
CC-Link IE Field Network Basic function	A set of functions that can be used in CC-Link IE Field Network Basic	×	×	×	×	×	△ *1	CC-Link IE Field Network Basic Reference Manual

*1 Availability depends on the version of the CPU module. (Page 466, Appendix 2)

*2 For details, refer to Page 573, Appendix 9.

Remark

For details on the special relay (SM) and special register (SD) used for each function, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

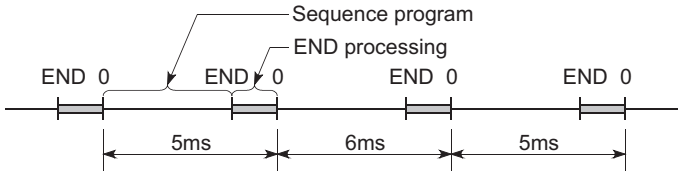
3.2 Constant Scan

Scan time differs depending on the execution status of instructions used in sequence programs. This function repeatedly executes sequence programs keeping their scan time constant.

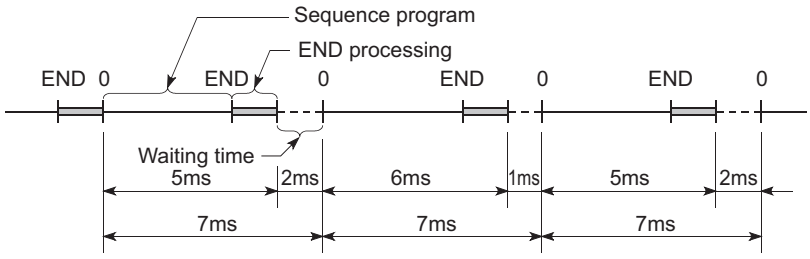
(1) Application

I/O refresh is performed before every sequence program execution.
This function is used to maintain I/O refresh intervals constant even if the execution time of each sequence program differs.

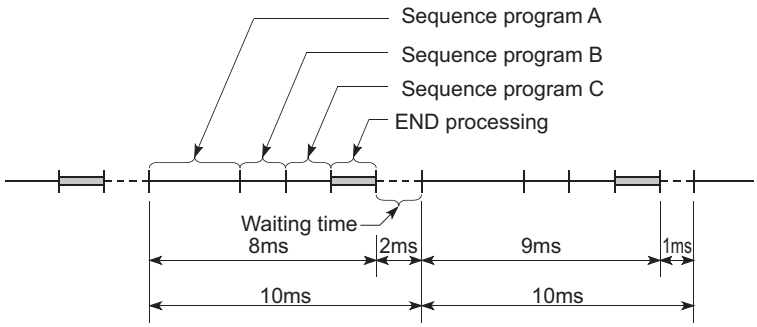
- Scan time without constant scan setting



- Scan time with constant scan setting (7ms)

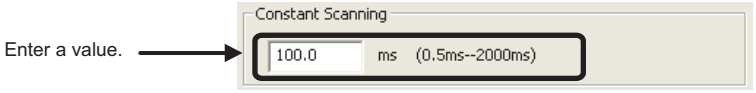


- Scan time for multiple programs with constant scan setting (10ms)



(2) Constant scan time setting

Set a constant scan time value in the PLC RAS tab of the PLC parameter dialog box. (☞ Page 442, Appendix 1.2.4)
When not executing the constant scan function, leave the constant scan time setting box blank.



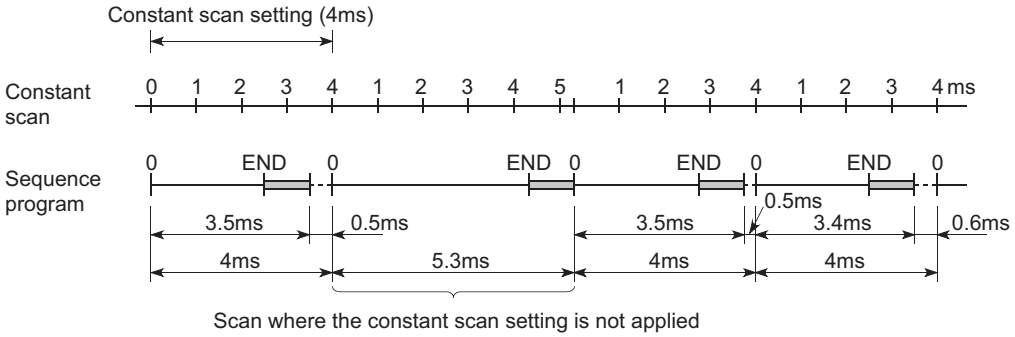
(a) Condition

The constant scan time needs to satisfy the following relational expression.

(WDT setting time) > (Constant scan setting time) > (Sequence program maximum scan time)

If the sequence program scan time is longer than the constant scan setting time, the CPU module detects "PRG. TIME OVER" (error code: 5010).

In this case, the constant scan setting will be ignored and the sequence program scan time will be applied.



If the sequence program scan time is longer than the WDT setting time, the CPU module detects "WDT ERROR".

In this case, the program execution will be stopped.

(3) Waiting time from when END processing is executed until next scan starts

Sequence program processing is stopped during the waiting time from when END processing of a sequence program is executed until next scan starts.

(a) When an interrupt factor occurs during waiting time

Either of the following programs is executed.

- Interrupt program
- Fixed scan execution type program

(b) When a service processing parameter is set

The communication service processing with peripherals (such as programming tools) and intelligent function modules is enabled during the waiting time by setting a service processing parameter. (☞ Page 241, Section 3.24.1)

(4) Constant scan accuracy

The constant scan accuracy is 0.01ms.

However, the constant scan time may increase in the following cases.

(a) When interrupt program or fixed scan execution type program is executed

Interrupts are disabled while an interrupt program or fixed scan execution type program is executed.

Even if the constant scan time runs out during execution of an interrupt program or fixed scan execution type program, the constant scan cannot be finished.

In this case, the constant scan time may exceed the time set and increase by the time of the program executed.

(b) When service processing is performed just before the end of constant scan

The constant scan time may exceed the time set and increase.

3.3 Latch Function

This function holds data in each device of the CPU module when:

- the CPU module is powered off and then on,
- the CPU module is reset, or
- power failure occurs exceeding the allowable momentary power failure time.

Data in devices of the CPU module are cleared and set back to their default (bit device: off, word device: 0) if the latch function is not used.

(1) Application

This function is used to hold the data managed by sequential control and continue control operation especially when the CPU module is powered off and then on.

(2) Program operation when the latch function is used

Program operation is the same, regardless of the latch status.

(3) Devices that can be latched

The following devices can be latched. (By default, only the latch relay is latched.)

- Latch relay (L)
- Link relay (B)
- Annunciator (F)
- Edge relay (V)
- Timer (T)
- Retentive timer (ST)
- Counter (C)
- Data register (D)
- Link register (W)

The following devices also can be latched when the file register is set to be used in parameter (PLC file tab).

- File register (R, ZR)
- Extended data register (D)
- Extended link register (W)

Point

When the battery life-prolonging function (☞ Page 250, Section 3.26) is set, the latch relay cannot be latched.

(4) Latch interval setting

Data are latched at each scan or at set intervals. Latch timing is set in parameter.

(a) Each scan

Latch data processing is performed during END processing of each scan. Since device data is latched every scan, the CPU module holds the latest device data at all times.

(b) Time setting Note 3.1

Latch data processing starts during the first END processing after a preset time has elapsed. Since the latch data processing is performed asynchronous to the sequence program, an increase in scan time is reduced.

Point

If the latch interval is shorter than a scan time, the latch timing occurs more than once within one scan. However, the latch processing is performed only once during the END processing.

(5) Latch setting

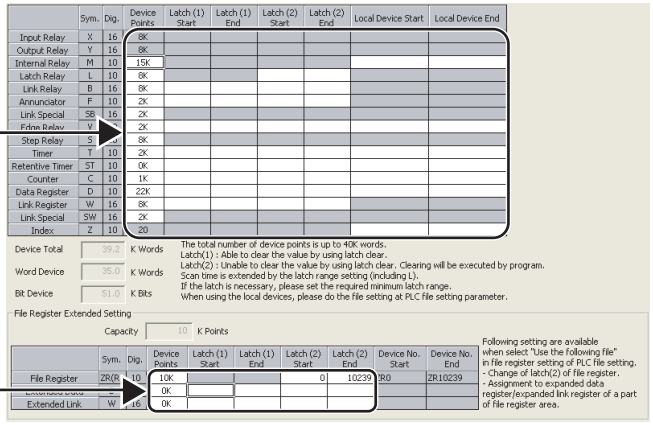
To latch data, a latch range and timing need to be set.

(a) Latch range setting

Latch-target devices and ranges are set. There are two range settings: the latch clear operation enable range setting (Latch (1)) and the latch clear operation disable range setting (Latch (2)).

Project window ⇨ [Parameter] ⇨ [PLC Parameter] ⇨ "Device" tab

Set the start and end device numbers.



Point

- If "Use the same file name as the program" is selected in the PLC File tab of the PLC parameter dialog box, latch ranges of the file register (ZR), extended data register (D), and extended link register (W) cannot be set. (All data in the file register (ZR) will be held.) The data outside the latch range will be cleared when the CPU module is powered off and on or is reset.
- When the file register file used is switched with the QDRSET instruction, the latch range setting of the file register will be disabled. After switching, regardless of the latch range setting, all data in the file register will be held.

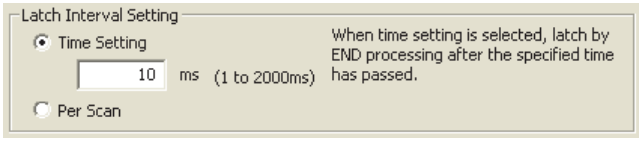
Note 3.1 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU can select "Time Setting" in parameter. The latch interval setting is fixed to "Each Scan" for other CPU modules.

(b) Latch interval setting (High-speed Universal model QCPU and Universal model Process CPU only)

A latch timing is set. (Page 447, Appendix 1.2.8)

Project window ⇨ [Parameter] ⇨ [PLC Parameter] ⇨ "Device" tab



(6) Device data latch method and influence on the scan time

Data latch processing is performed during END processing.

For this reason, the scan time increases.

Consider an influence on the scan time when latching devices. (Page 478, Appendix 3.2 (6))

Point

To minimize the scan time increase due to latch[*1], reduce the number of latch points (latch (1) setting, latch (2) setting, and latch relay) as much as possible by performing the following.

- Move latch-target data to the file register.
- Store device data that is less frequently updated in the standard ROM using the SP.DEVST instruction. (The data stored in the standard ROM can be read using the S(P).DEVLD instruction.) (Page 259, Section 3.30)
- Set the latch interval parameter to "Time Setting". (Page 124, Section 3.3 (5) (b))

*1 With the file register (including the extended data register (D) and extended link register (W)), the scan time is not increased due to latch.

(7) Device data latch clear

The following table lists the status of device data when the latch clear operation is performed.

Latch setting	Status of data
Device data without latch setting	Cleared
Device data in the "Latch (1)" range	Cleared
Device data in the "Latch (2)" range	Held[*1]

*1 For the clearing method, refer to Page 75, Section 2.7 (4).

(8) Precautions

(a) When a local device or initial device value is specified

Even data of the latch-specified devices cannot be latched.

(b) Use of battery

Device data in the latch range are held with the battery installed to the CPU module.

- Even for the boot operation, the battery is required to latch device data.
- Note that if the battery connector is disconnected from the connector of the CPU module while the power supply for the programmable controller is off, device data in the latch range will not be held and will become undefined.

3.4 Output Mode at Operating Status Change (STOP to RUN)

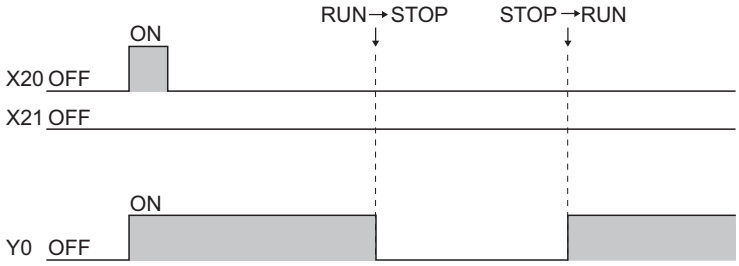
When the operating status is switched from RUN to STOP, the CPU module internally stores the outputs (Y) in the RUN status and turns off all the outputs (Y). The status of the outputs (Y) when the CPU module is changed from STOP to RUN can be selected from the following two options in the parameter setting.

- Output the output (Y) status prior to STOP. ("Previous state")
- Clear the output (Y) status. ("Recalculate (output is 1 scan later)")

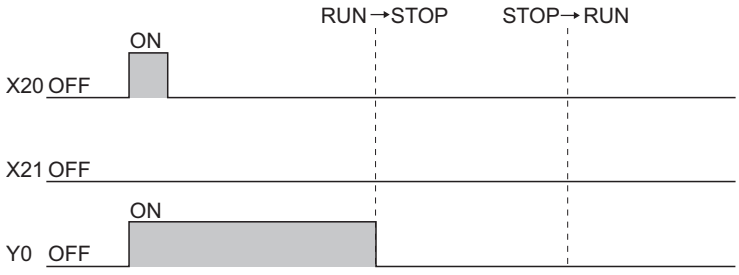
(1) Application

This function is used to determine the status of outputs (whether to resume the outputs from the previous status or not) when the operating status is changed from STOP to RUN in the holding circuit.

- When outputting the output (Y) status prior to STOP



- When clearing the output (Y) status



(2) Operation when the operating status is changed from STOP to RUN

(a) Previous state (Default)

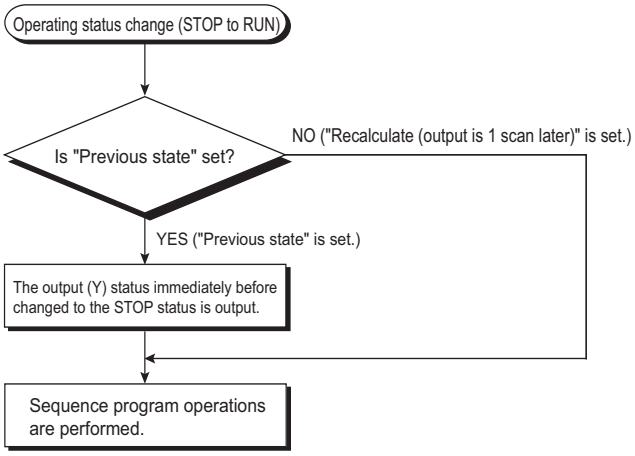
The CPU module outputs the output (Y) status immediately before changed to the STOP status and then performs sequence program operations.

(b) Recalculate (output is 1 scan later)

All outputs are turned off.

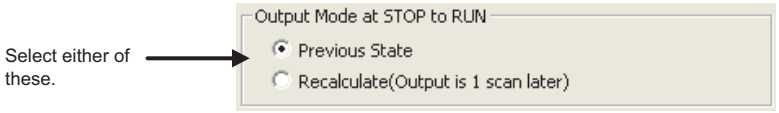
The CPU module outputs the output (Y) status after sequence program operations are completed.

For the operation of the CPU module when the output (Y) status is forcibly turned on in the STOP status, refer to Page 126, Section 3.4 (4).



(3) Setting the output mode when the operating status is changed from STOP to RUN

Set the output mode when the operating status is changed from STOP to RUN in the PLC system tab of the PLC parameter dialog box.



(4) Precautions

The following tables shows the output status of the CPU module when the operating status is changed from STOP to RUN after the outputs (Y) are forcibly turned on in the STOP status.

Output mode ("Output Mode at STOP to RUN") selected	Output status
Previous State	The output status prior to STOP is output. The on status is not held if the output status prior to STOP was off.
Recalculate (Output is 1 scan later)	The on status is held and output.

3.5 Clock Function

This function reads the internal clock data of the CPU module by a sequence program and uses it for time management. The clock data is used for time management required for some functions in the system, such as storing date into the error history.

Remark

The Built-in Ethernet port QCPU can set the time in the CPU module automatically by using the time setting function (SNTP client).

QnUCPU User's Manual (Communication via Built-in Ethernet Port)

(1) Clock operation at power off and momentary power failure

Clock operation continues by the internal battery of the CPU module even when the programmable controller is powered off or power failure occurs exceeding the allowable momentary power failure time.

(2) Clock data

The following table lists the details of clock data in the CPU module.

Data name	Description	
Year	Four digits*1 (from 1980 to 2079)	
Month	1 to 12	
Day	1 to 31 (Automatic leap year detection)	
Hour	0 to 23 (24 hours)	
Minute	0 to 59	
Second	0 to 59	
Day of the week	0	Sunday
	1	Monday
	2	Tuesday
	3	Wednesday
	4	Thursday
	5	Friday
	6	Saturday
1/1000 seconds*2	0 to 999	

*1 Storing in SD213 for the first two digits and SD210 for the last two digits of the year

*2 Can be read using Expansion clock data read (S(P).DATERD) instruction.

MELSEC-Q/L Programming Manual (Common Instruction)

(3) Changing and reading clock data

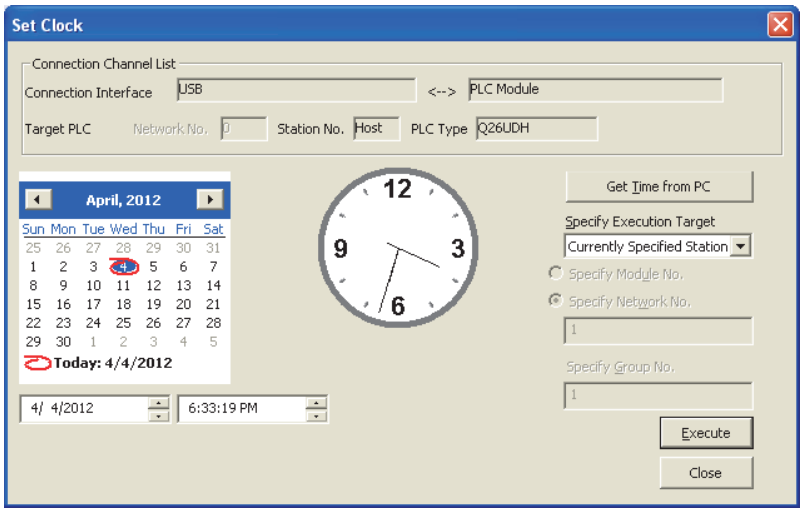
(a) Changing clock data

Clock data can be changed using either a programming tool or a program.

- Changing clock data by programming tool

  Open the "Set Clock" dialog box.

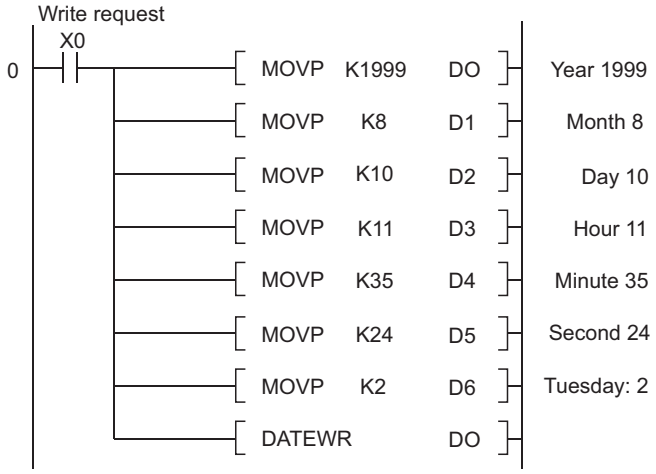
  [Online] ⇨ [Set Clock...]



- Changing clock data by a program

  Use the DATEWR instruction (instruction for writing clock data) to change the clock data.

  The following shows a program example that writes clock data set D0 to D6.



For details of the DATEWR instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

Point

- When clock data is changed, the clock of 1/1000 seconds is reset to 0.
- Year data settable by programming tool is up to 2037.

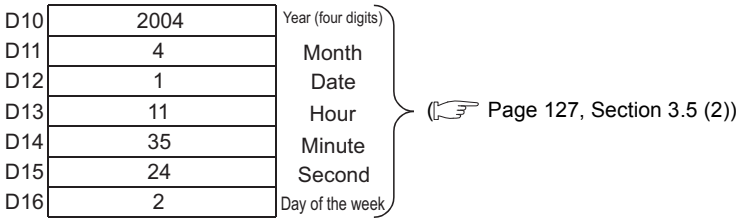
(b) Reading clock data

To read clock data to the data register, use either of the following instructions in the program.

- DATERD (instruction for reading clock data)
- S(P).DATERD (instruction for reading extended clock data)

The following figure shows a program for storing clock data that are read using the DATERD instruction to D10 to D16.

*1 The following figure shows the clock data stored in D10 to D16.



For details of the DATERD and S(P).DATERD instructions, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

Point

Clock data can also be written or read by the special relay (SM210 to SM213) and special register (SD210 to SD213). For details of the special relay and special register, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection).

(4) Precautions

(a) Initial clock data setting

No clock data is set at the factory.

Clock data is required for some functions of the CPU module used in the system, such as error history data storage, or for intelligent function modules.

Before using the CPU module for the first time, set the time correctly.

(b) Clock data correction

If clock data is corrected, rewrite all clock data to the CPU module.

(c) Clock data setting range

When changing clock data, write data within the range given in Page 127, Section 3.5 (2).

If data outside of clock range is written to the CPU module, the clock function does not operate normally.

However, the CPU module does not detect an error if the clock data is within the range.

	Write operation to the CPU module	CPU module operation
February 30	Executed	An error is not detected.
32th of month 13	Not executed	• When the DATEWR instruction is executed, "OPERATION ERROR" (error code: 4100) is detected. • When SM210 is on, SM211 turns on.

(d) Use for clock data of 1/1000 sec.

- Function that clock data of 1/1000 sec. can be used

  Only the following instructions can use the clock data of 1/1000 sec. Other instructions cannot use data of 1/1000 sec. (such as data read using SM/SD, data indicating error occurrence time stored as error history data, data read using a programming tool, and data read using dedicated instructions for other modules.)

  - S(P).DATERD
  - S(P).DATE+
  - S(P).DATE-

- When clock data is changed

  When clock data is changed using a programming tool or instructions (including dedicated instructions for other modules), the clock data of 1/1000 sec. is reset to 0.

(5) Clock data accuracy

Accuracy of the clock data varies depending on the ambient temperature as shown below.

Ambient temperature (°C)	Accuracy (Day difference, S)
0	-2.96 to +3.74 (TYP.+1.42)
+ 25	-3.18 to +3.74 (TYP.+1.50)
+ 55	-13.20 to +2.12 (TYP.-3.54)

(6) Clock data comparison

To compare clock data in a sequence program, read the clock data with the DATERD instruction (instruction for reading clock data).

Since the DATERD instruction reads the year data in four digits, the data can be compared by the comparison instruction without any modifications.

3.6 Remote Operation

The remote operation can change the operating status of the CPU module externally (using a programming tool, external devices in the MC protocol, link dedicated instructions for a CC-Link IE module or a MELSECNET/H module, or remote contacts). Four types of remote operations are available:

- Remote RUN/STOP (☞ Page 131, Section 3.6.1)
- Remote PAUSE (☞ Page 134, Section 3.6.2)
- Remote RESET (☞ Page 136, Section 3.6.3)
- Remote latch clear (☞ Page 137, Section 3.6.4)

3.6.1 Remote RUN/STOP

This operation changes the operating status of the CPU module externally to RUN or STOP, keeping the RUN/STOP/RESET switch of the CPU module in the RUN position.

(1) Application

This operation is useful to run or stop the CPU module remotely when:

- the CPU module is inaccessible, or
- the CPU module is in a control panel.

(2) Program operation

The program operation will be as follows when the remote RUN/STOP operation is performed.

(a) Remote STOP

The CPU module executes a program until the END instruction and changes its operating status to STOP.

(b) Remote RUN

The CPU module changes its operating status to RUN and executes a program from the step 0.

(3) Executing method

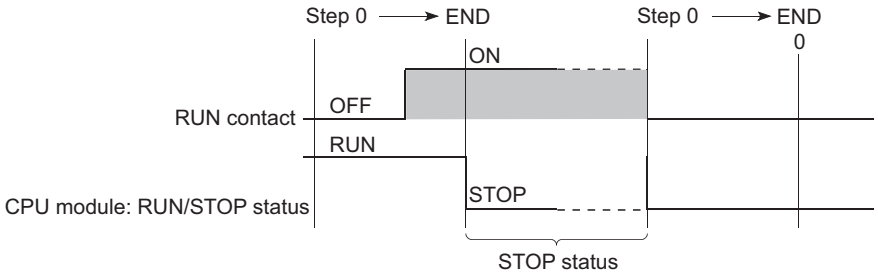
(a) Using a RUN contact

Set a RUN contact in the PLC system tab of the PLC parameter dialog box.

The settable device range is X0 to 1FFF.

The remote RUN/STOP operation can be performed by turning on/off the set RUN contact.

- When the RUN contact is turned off, the CPU module status changes to RUN.
- When the RUN contact is turned on, the CPU module status changes to STOP.



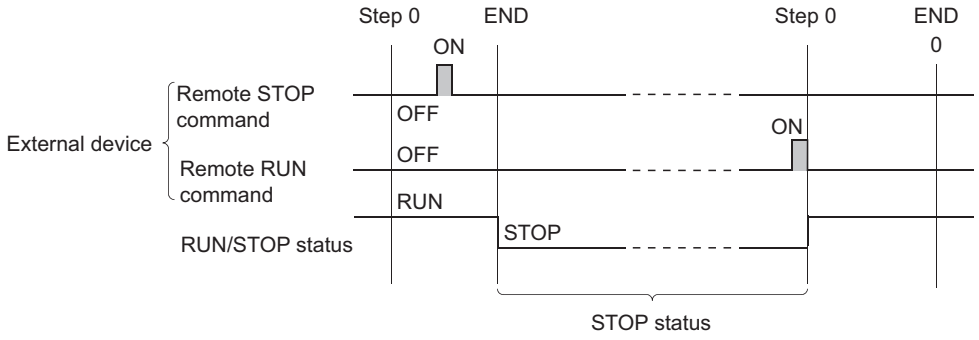
(b) Using a programming tool

Select [Online] → [Remote Operation] → "RUN" or "STOP".

(c) Using an external device in the MC protocol

Use MC protocol commands. For commands, refer to the following.

MELSEC Communication Protocol Reference Manual



(d) With link dedicated instructions of the CC-Link IE module or MELSECNET/H module

The remote RUN/STOP operation by link dedicated instructions of the CC-Link IE module or MELSECNET/H module can change the RUN/STOP status of the CPU module.

For details, refer to the following.

Manual for each network module

(4) Precautions

Pay attention to the following since the STOP status is given priority over other status.

(a) Timing of changing to the STOP status

The operating status of the CPU module is changed to STOP when the remote STOP operation is performed from any one of the following: RUN contact, programming tool, or an external device using the MC protocol.

(b) When changing the status back to RUN

To change the operating status back to RUN after the CPU module status was changed to STOP by the remote STOP operation, perform the remote RUN operation in the same order for the remote STOP operation.

Point

- The definitions of the RUN/STOP status are described below.
  - RUN status: Status where program operations are repeatedly performed in a loop between the step 0 and the END or FEND instruction.
  - STOP status: Status where program operations are stopped. All outputs (Y) turn off.
- After the CPU module is reset, the operating status of the CPU module becomes the one set using the RUN/STOP/RESET switch.

3.6.2 Remote PAUSE

This operation changes the operating status of the CPU module externally to PAUSE, keeping the RUN/STOP/RESET switch of the CPU module in the RUN position. PAUSE status is status where sequence program operations in the CPU module are stopped, holding the status (on or off) of all outputs (Y).

(1) Application

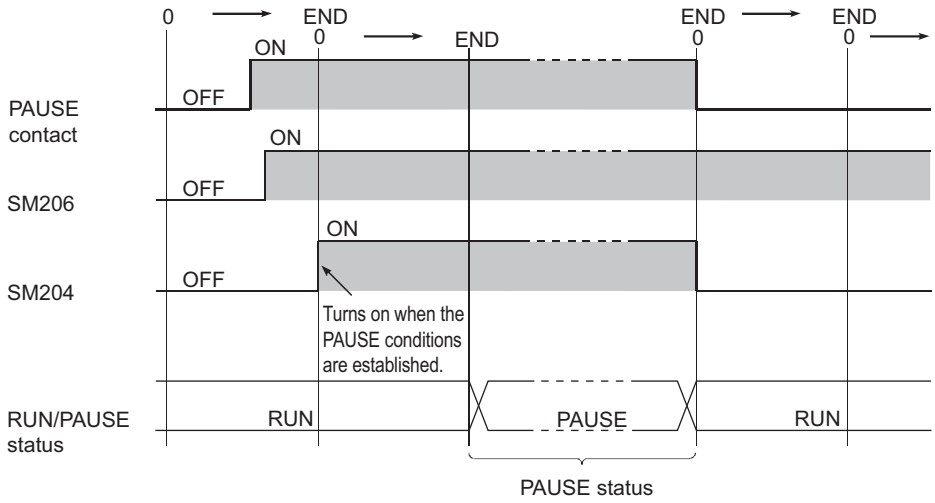
This operation is useful, especially during the process control, to hold the on status of outputs (Y) even after the operating status of the CPU module is switched from RUN to STOP.

(2) Executing method

(a) Using a PAUSE contact

Set a PAUSE contact in the PLC system tab of the PLC parameter dialog box.
The settable device range is X0 to 1FFF.

- The PAUSE contact (SM204) turns on during END processing of the scan where both the PAUSE contact and PAUSE enable coil (SM206) turn on.
  The CPU module executes one more scan until the END instruction after the scan where the PAUSE contact turns on, and then changes its operating status to PAUSE. In the PAUSE status, the program operations are stopped.
- When the PAUSE contact or SM206 is turned off, the PAUSE status will be canceled and the CPU module will restart the sequence program operation from the step 0.



Point

Setting of only a PAUSE contact is not allowed. (When setting a PAUSE contact, set a RUN contact as well.)

(b) Using a programming tool

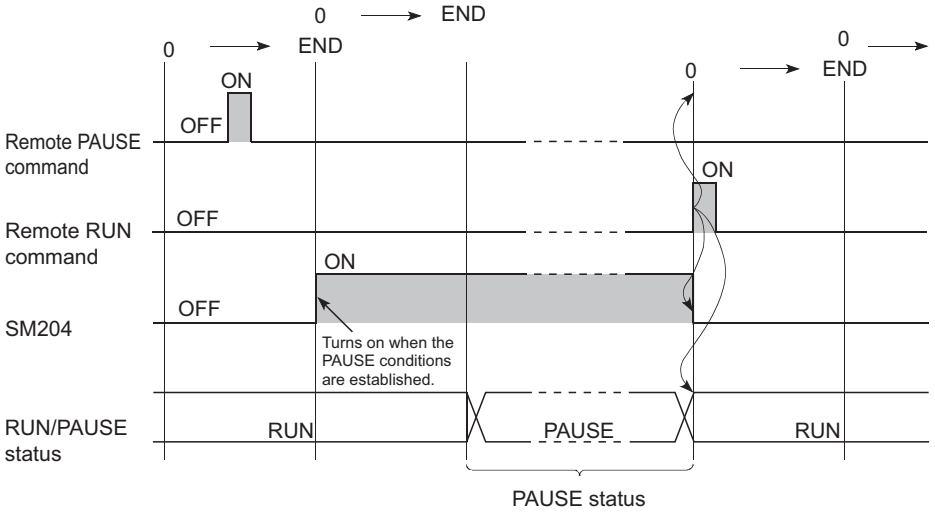
Select [Online] → [Remote Operation] → "PAUSE".

(c) Using an external device in the MC protocol

Use MC protocol commands. For commands, refer to the following.

MELSEC Communication Protocol Reference Manual

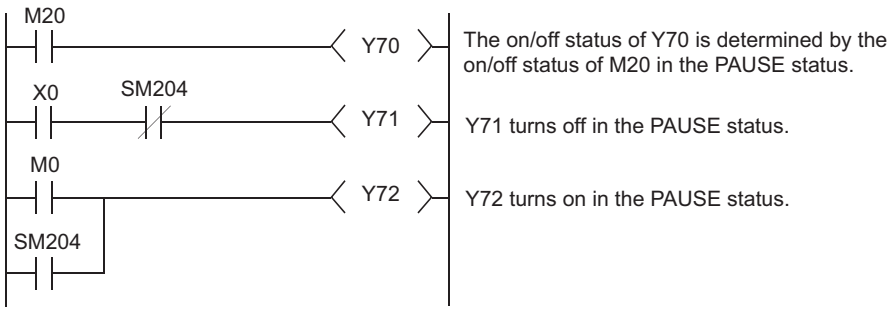
- The PAUSE contact (SM204) turns on during END processing of the scan where the remote PAUSE command is executed.
  The CPU module executes one more scan until the END instruction after the scan where the PAUSE contact turns on, and then changes its operating status to PAUSE. In the PAUSE status, the program operations are stopped.
- Upon execution of the remote RUN command, the CPU module will restart the sequence program operations from the step 0.



(3) Precautions

(a) When forcibly keeping output status (on or off)

To forcibly keep the output status (on or off) in the PAUSE status, provide an interlock with the PAUSE contact (SM204).



3.6.3 Remote RESET

This operation resets the CPU module externally when the CPU module is in the STOP status. Even if the RUN/STOP/RESET switch is in the RUN position, this operation can be performed when the module is stopped due to an error detected by the self-diagnostics function.

(1) Application

This operation is useful to reset the CPU module remotely when an error occurs in the CPU module placed in an inaccessible location.

(2) Executing method

(a) Using a programming tool

Select [Online] → [Remote Operation] → "RESET".

(b) Using an external device in the MC protocol

Use the MC protocol command. For the commands, refer to the following.

MELSEC Communication Protocol Reference Manual

Point

Before performing the remote RESET operation, select the "Allow" checkbox for the remote RESET operation in the PLC System tab of the PLC parameter dialog box, and write the parameter setting to the CPU module. Without the preset parameter setting, the operation cannot be performed.

(3) Precautions

(a) Remote RESET in the RUN status

When the CPU module is in the RUN status, the remote RESET operation cannot be performed.
To perform the operation, change the operating status of the CPU module to STOP by the remote STOP.

(b) Status after reset processing

After reset processing of the remote RESET operation is completed, the CPU module will be placed in the operating status set by the RUN/STOP/RESET switch.
If the RUN/STOP/RESET switch is set to STOP, the CPU module will be in the STOP status. If the switch is set to RUN, the CPU module will be in the RUN status.

Point

- If the remote RESET operation is performed to the CPU module which is stopped due to an error, note that the CPU module will be placed in the operating status set by the RUN/STOP/RESET switch after reset processing is completed.
- If the CPU module cannot be reset by the remote RESET operation from a programming tool, check if the remote RESET operation is set to "Allow" under the PLC System tab in the PLC parameter dialog box.
  If the "Allow" checkbox is not selected, the CPU module cannot be reset even after the remote RESET processing from a programming tool is completed.

(c) When an error occurs due to noise

Note that the CPU module may not be reset by the remote RESET operation. In this case, reset the CPU module using the RUN/STOP/RESET switch or power off and then on the CPU module.

3.6.4 Remote latch clear

This function resets the latched device data from a programming tool when the CPU module is in the STOP status.

(1) Application

This function is useful in the following cases if used together with the remote RUN/STOP operation.

- When the CPU module is inaccessible
- To clear latched device data in the CPU module in a control panel externally

(2) Executing method

(a) Using a programming tool

Select [Online] → [Remote Operation] → "Latch clear".

(b) Using an external device in the MC protocol

Use the MC protocol command. For the commands, refer to the following.

MELSEC Communication Protocol Reference Manual

To perform the remote latch clear operation, follow the following steps.

1. **Change the operating status of the CPU module to STOP by the remote STOP operation.**
2. **Clear the latched device data in the CPU module by the remote latch clear operation.**
3. **After remote latch clear processing is completed, perform the remote RUN operation to return the operating status to RUN.**

(3) Precautions

(a) Latch clear in the RUN status

The latch clear operation cannot be performed when the CPU module is in the RUN status.

(b) Latch clear enabled range

There are two kinds of latch range can be set in the Device tab of the PLC parameter dialog box: latch clear operation enable and disable range.

Remote latch clear operation resets only the data set in the "Latch (1)" (latch clear operation enable range).

For the method for resetting the device data in the latch clear operation disable range, refer to Page 75, Section 2.7 (4) (b).

(c) Devices that are reset by the remote latch clear operation

Devices that are not latched are also reset when the remote latch clear operation is performed.

3.6.5 Relationship between remote operation and RUN/STOP status of the CPU module

(1) Relationship between remote operation and RUN/STOP status of the CPU module

The following table lists the operating status of the CPU module according to the combination of remote operation and RUN/STOP status of the CPU module.

RUN/STOP status	Remote operation				
	RUN[*1]	STOP	PAUSE[*2]	RESET[*3]	Latch clear
RUN	RUN	STOP	PAUSE	Operation disabled[*4]	Operation disabled[*4]
STOP	STOP	STOP	STOP	RESET[*5]	Latch clear

*1 When performing the operation using a RUN contact, "RUN-PAUSE contact" must be set in the PLC system tab of the PLC parameter dialog box.

*2 When performing the operation using a PAUSE contact, "RUN-PAUSE contact" must be set in the PLC system tab of the PLC parameter dialog box.
In addition, the PAUSE enable coil (SM206) must be turned on.

*3 The "Allow" checkbox for the remote RESET operation must be selected in the PLC system tab of the PLC parameter dialog box.

*4 The remote RESET and remote latch clear operations are enabled if the CPU module status is changed to STOP by the remote STOP operation.

*5 The status includes a case where the CPU module is stopped due to an error.

(2) Remote operations from a single programming tool

When remote operations are performed from a single programming tool, the operating status of the CPU module will be the status of the last remote operation performed.

(3) Remote operations from multiple programming tools

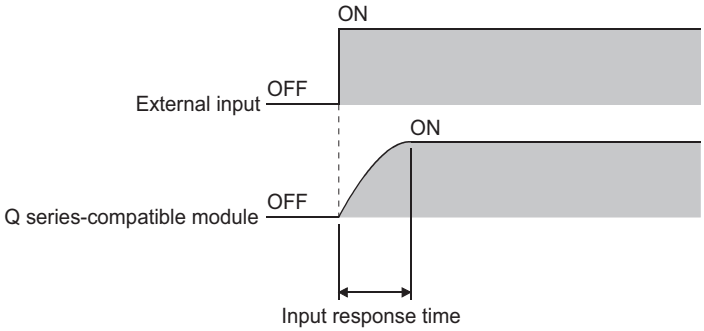
Any remote operation from other programming tools via other stations cannot be performed to the CPU module where remote operations are being performed from a programming tool connected. To perform any remote operations from other programming tools, cancel the remote operation by performing the remote RUN operation from the same programming tool that is performing the current remote operation.
For example, even if the remote STOP or RUN operation is performed from the other programming tool to the CPU module where the remote PAUSE operation has been performed by the programming tool connected, the CPU module remains in the PAUSE status. Once after the remote operation is canceled by performing the remote RUN operation from the same programming tool that is performing the remote PAUSE operation, remote operations from the other programming tool will be enabled.

3.7 Q Series-compatible Module Input Response Time Selection (I/O Response Time)

This function is used to change the input response time for each Q series-compatible module. The following table lists the modules available for input response time change and selectable time settings.

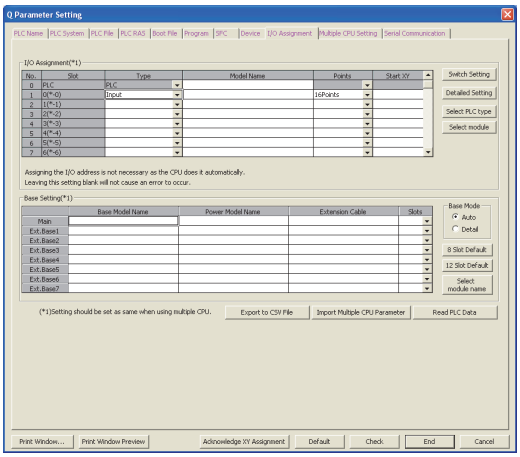
Module name	Type	Settable time
Input module	"Input"	1ms, 5ms, 10ms, 20ms, 70ms (Default: 10ms)
I/O combined module	"I/O Mix"	
High-speed input module	"Hi Input"	0.1ms, 0.2ms, 0.4ms, 0.6ms, 1ms (Default: 0.2ms)
Interrupt module	"Interrupt"	

The Q series-compatible modules in the table above take in external inputs within the set input response time.

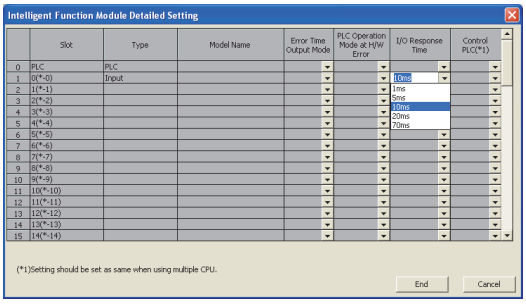


(1) Input response time setting

Set input response time values in the I/O Assignment tab of the PLC parameter dialog box.



1. **Set I/O assignment.**



2. **Click the** Detailed Setting **button.**

3. **Select an input response time.**

(2) Precautions

(a) When input response time is shortened

The shorter the input response time is, the more the CPU module is susceptible to noise.
Consider the operating environment when setting input response time values.

(b) When an AnS/A series-compatible module is used

The input response time cannot be changed.
Even if the input response time is set to the slot of the input module or interrupt module (AnS/A series-compatible), the setting is not valid.

(c) Enabling the setting

The input response time setting will be enabled when:

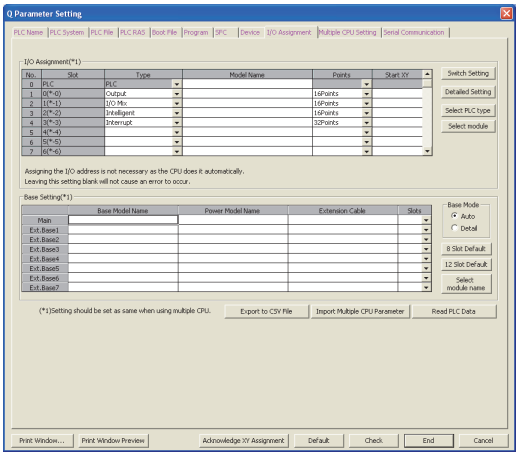
- the CPU module is powered off and then on, or
- the CPU module is reset.

3.8 Error Time Output Mode Setting

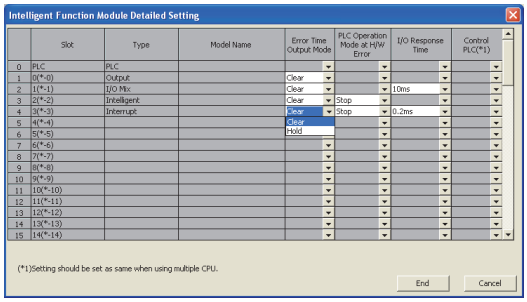
This function determines the output mode (clear or hold) from the CPU module to the Q series-compatible output modules, I/O combined modules, intelligent function modules, and/or interrupt module when a stop error occurs in the CPU module.

(1) Error time output mode setting

Set the error time output mode in the I/O Assignment tab of the PLC parameter dialog box.



1. Set I/O assignment.



2. Click the Detailed Setting button.

3. Select "Clear" or "Hold" for the slot where the error time output mode is set. (Default: "Clear")

(2) Precautions

The error time output setting will be enabled when:

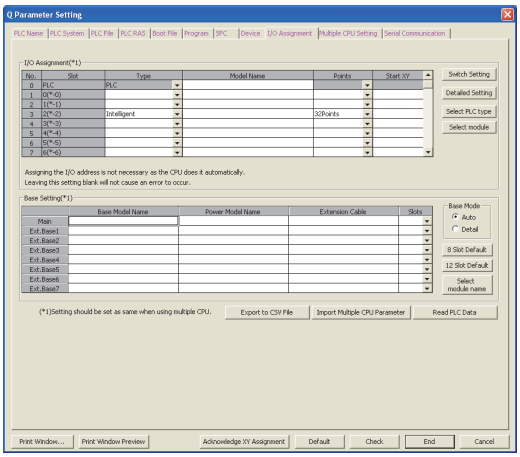
- the CPU module is powered off and then on, or
- the CPU module is reset.

3.9 H/W Error Time PLC Operation Mode Setting

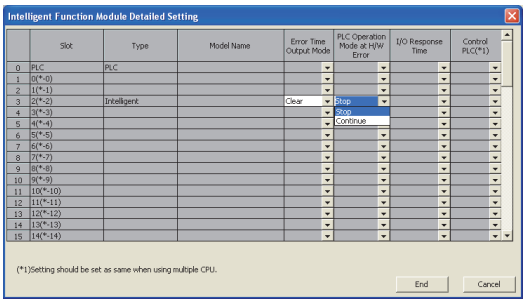
This setting determines whether to stop or continue the CPU module operation when a hardware error (CPU module detects SP.UNIT DOWN) occurs in the intelligent function module or the interrupt module.

(1) H/W error time PLC operation mode setting

Set the H/W error time PLC operation mode in the I/O Assignment tab of the PLC parameter dialog box.



1. **Set I/O assignment.**



2. **Click the** Detailed Setting **button.**

3. **Select "Stop" or "Continue" for the slot where the H/W error time PLC operation mode is set. (Default: "Stop")**

(2) Precautions

The H/W error time PLC operation setting will be enabled when:

- the CPU module is powered off and then on, or
- the CPU module is reset.

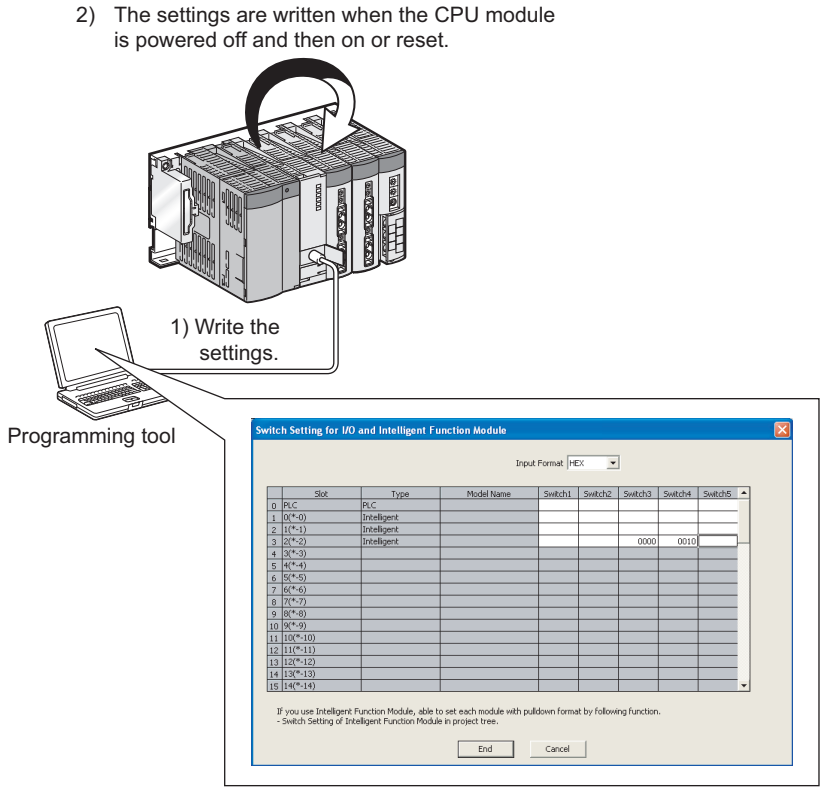
3.10 Intelligent Function Module Switch Setting

Switches of a Q series-compatible intelligent function module or an interrupt module can be set in a programming tool.

(1) Writing the switch settings

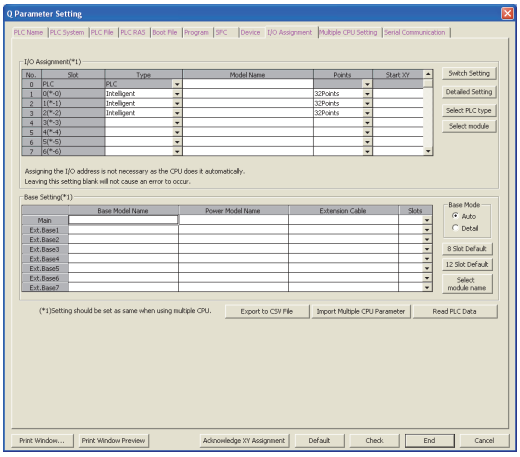
The switch settings will be written from the CPU module to each intelligent function module and interrupt module when:

- the CPU module is powered off and then on, or
- the CPU module is reset.

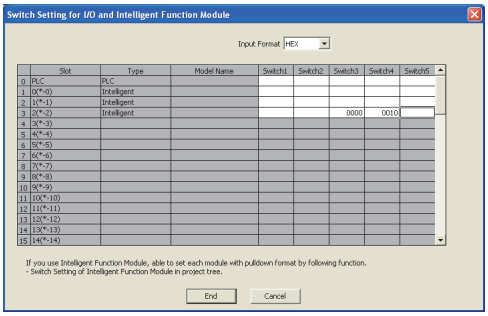


(2) Switch setting for an intelligent function module or an interrupt module

Set the switch details in the I/O Assignment tab of the PLC parameter dialog box.



1. Set I/O assignment.



2. Click the Switch Setting **button.**

3. Set the switch details for an intelligent function module or an interrupt module.

(3) Precautions

(a) When an AnS/A series-compatible module is used

Do not set the switch details for AnS/A series-compatible special function modules.
Even if values are input, the setting is ignored.

(b) Switch setting details of each module

For the switch setting details of each intelligent function module or interrupt module, refer to the manual for the intelligent function module or interrupt module used.

(c) Enabling the setting

The switch settings of each intelligent function module or interrupt module will be enabled when:

- the CPU module is powered off and then on, or
- the CPU module is reset.

3.11 Monitor Function

Programs and device data of the CPU module, and intelligent function module status can be read from a programming tool using this function.

○ : Available, △ : Available with restrictions, × : Not available

Monitor function	Availability					Reference
	Q00UJ CPU	Q00UCPU, Q01UCPU	Q02UCPU	QnUD(H) CPU	Built-in Ethernet port QCPU	
Monitor*1	○	○	○	○	○	Operating manual for the programming tool used
Monitor condition setting	×	×	△*2	△*2	○	Page 146, Section 3.11.1
Local device monitor/test	×	○	○	○	○	Page 151, Section 3.11.2
External input/output forced on/off	○	○	△*2	△*2	○	Page 154, Section 3.11.3
Executional conditioned device test	○	○	△*2	△*2	○	Page 159, Section 3.11.4

*1 This includes a ladder monitor, device batch monitor, entry data monitor, entry ladder monitor and local device monitor.

*2 Availability depends on the version of the CPU module. (Page 466, Appendix 2)

(1) Monitor request timing and displayed data

The CPU module processes monitor requests from a programming tool during END processing. For this reason, the data in the CPU module at the time of END processing will be displayed in the programming tool.

(2) Monitor with monitor condition settings

By setting a monitor condition in a programming tool during debugging, the program operation status in the CPU module can be monitored under the specified condition. Also, by setting a monitoring stop condition, a monitoring status can be held under the specified condition.

(3) Local device monitor

If multiple programs are executed and local devices are used, data in local devices of each program can also be monitored.

3.11.1 Monitor condition setting Note 3.2

This setting is used to monitor data in the CPU module under a specified condition.

(1) Setting method

There are two kinds of monitor condition setting.

- Monitor execution condition setting
- Monitor stop condition setting

For the setting method, refer to the following.

Operating manual for the programming tool used

Note 3.2 Universal

The Q00UJCPU, Q00UCPU, and Q01UCPU do not support this function. Before executing the function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

(a) When only a step number is specified

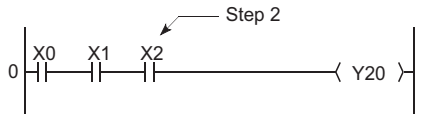
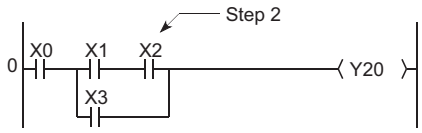
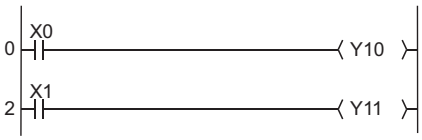
Monitor data is collected when the status immediately before execution of the specified step becomes the specified status.

The following status can be specified.

- When the operation of the specified step changes from the non-execution status to the execution status: <↑>
- When the operation of the specified step changes from the execution status to the non-execution status: <↓>
- Always only when the operation of the specified step is in execution: <ON>
- Always only when the operation of the specified step is in non-execution: <OFF>
- Always regardless of the status of the operation of the specified step: <Always>

Point

- If a step between the AND/OR blocks is specified as a monitor condition, monitor data is collected when the status previous to execution of the specified step is specified by the LD instruction. The monitor timing depends on the ladder of step specified as a monitor condition. The following shows examples of monitoring when the step 2 is on (Step No. [2] = <ON>).

Condition	Description
When the step 2 is connected by the AND instruction	The monitor execution condition is established when both X0 and X1 are on. Ladder mode  List mode 0 LD X0 1 AND X1 2 AND X2 3 OUT Y20
When the step 2 is connected in the middle of the AND/OR block	The monitor execution condition is established when X1 turns on. (The on/off status of X0 does not affect the establishment of the monitor execution condition.) Ladder mode  List mode 0 LD X0 1 LD X1 2 AND X2 3 OR X3 4 ANB 5 OUT Y20
When the start of a ladder block other than the step 0 is specified for the step number as a detailed condition	Monitor data is collected when the execution status of the instruction right before execution becomes the specified status. If (Step No. [2] = <ON>) is specified in the following ladder, monitor data is collected when OUT Y10 turns on. Ladder mode  List mode 0 LD X0 1 OUT Y10 2 LD X1 3 OUT Y11

- When "0" is specified as the step No., set the condition to "Always".
- With the High-speed Universal model QCPU and Universal model Process CPU, only the first data which satisfies the specified conditions is collected if the step between the FOR and NEXT instructions is specified.

(b) When only a device is specified

Either word device or bit device can be specified.

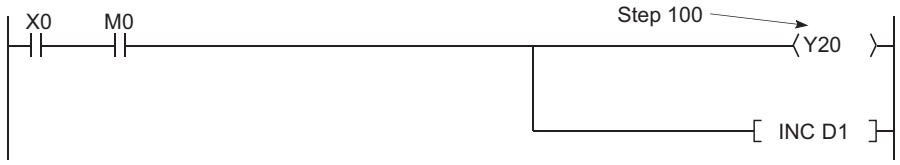
- When a word device is specified
  Monitor data is collected when the current value of the specified word device becomes the specified value. Enter the current value (in decimal or hexadecimal).
- When a bit device is specified
  Monitor data is collected when the execution status of the specified bit device becomes the specified status. Select the execution condition (on the rising edge or falling edge).

(c) When a step number and device are specified

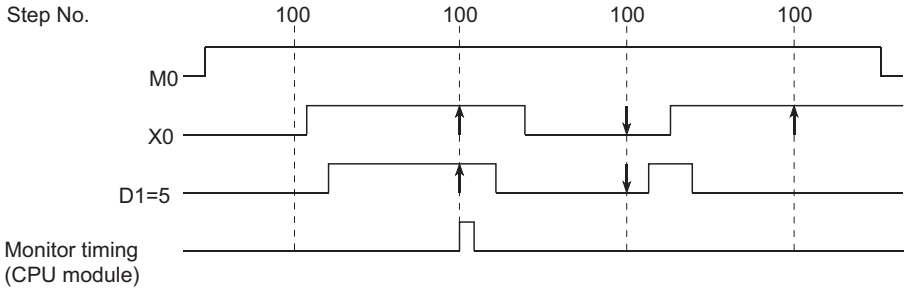
Monitor data is collected when the status previous to execution of the specified status or the status (current value) of the specified bit device (word device) becomes the specified value.

Point

- When "Step No.[100]=<↑>, Word device [D1]=[K5]" is specified as an execution condition, a monitor execution condition is established on the rising edge of the step 100 and also D1=5.



The monitor interval of a programming tool depends on the processing speed of the programming tool. For the monitor execution conditions established at the interval shorter than the monitor interval of the programming tool, monitor is executed only when the monitor execution condition is established at the monitor timing of the programming tool.



- With the High-speed Universal model QCPU and Universal model Process CPU, only the first data which satisfies the specified conditions is collected if the step between the FOR and NEXT instructions is specified.

(2) Precautions

(a) Files to be monitored

When monitor conditions are set, a programming tool monitors the file displayed on the screen. Select [Online] → [Read from PLC] in the programming tool and read data from the CPU module so that the file name in the CPU module to be monitored matches the file name displayed on the screen of the programming tool.

(b) No file register setting

If the file register is monitored when there is no file register used, "$FFFF_H$" is displayed.

(c) Device assignment

For a monitor operation, the device assignment in the CPU module and the programming tool must be the same.

(d) Monitoring the buffer memory of an intelligent function module

When monitoring the buffer memory of an intelligent function module, the scan time increases for the same reason for execution of the FROM/TO instructions.

(e) Monitoring by multiple users

When multiple users are performing monitoring at the same time, pay attention to the following.

- High speed monitor can be performed by increasing 1K step per monitor file of other stations in the system area when formatting the program memory or setting a parameter in the Boot file tab of the PLC parameter dialog box. Up to 15 stations can be set as the station monitor file, but the program space will be reduced.
- If the monitor condition or monitor stop condition is set, only one user can perform monitoring.

(f) Setting a monitor stop condition

A monitor stop condition can be set only in the ladder monitor.

(g) Specifying the same device as a condition

When specifying the same device as a monitor condition or monitor stop condition, set the on/off status as well.

(h) Specifying a step number as a monitor condition

If an instruction in the specified step is not executed in such cases described below, the monitor condition will not be established.

- The specified step is skipped with the CJ, SCJ, or JMP instruction.
- The specified step is the END instruction and never be executed because the FEND instruction also exists in the program.

(i) During monitor condition registration

Do not reset the CPU module while monitoring conditions are being registered.

(j) Monitor operation with monitor condition setting

When monitor operation with monitor condition setting is performed, other applications on the same personal computer cannot execute any online function using the same route for the monitor operation. The following applications must be noted.

- Programming tool
- Application using MX Component
- MX Sheet

If any online function is executed by other applications using the same route for the monitor operation, the following situations may occur.

- No response is returned from the CPU module for the online function executed. (An online communication function time-out occurs.)
- The CPU module detects an error (error code: 4109) for the online function executed.
- Even when the monitor condition is established in the CPU module, monitoring results cannot be updated for the monitor operation with monitor condition setting.

3.11.2 Local device monitor/test Note 3.3

This operation is useful for debugging a program, monitoring local devices (☞ Page 422, Section 6.2) in the program monitored by a programming tool.

(1) Monitoring a local device

The following table lists the monitor operation when the CPU module executes three programs "A", "B", and "C" and D0 to D99 are set as a local device.
(Three programs are to be executed in the order of A → B → C → (END processing) → A → B....)

Setting	Monitored device	
	D0 (Local device)	D100 (Global device)
Local device monitor is set	The D0 value in the specified program (local device for a program) is monitored.*1	The D100 value after execution of the specified program is monitored.*2
Local device monitor is not set	The D0 value after execution of the program "C" is monitored.	The D100 value after execution of the program "C" is monitored.

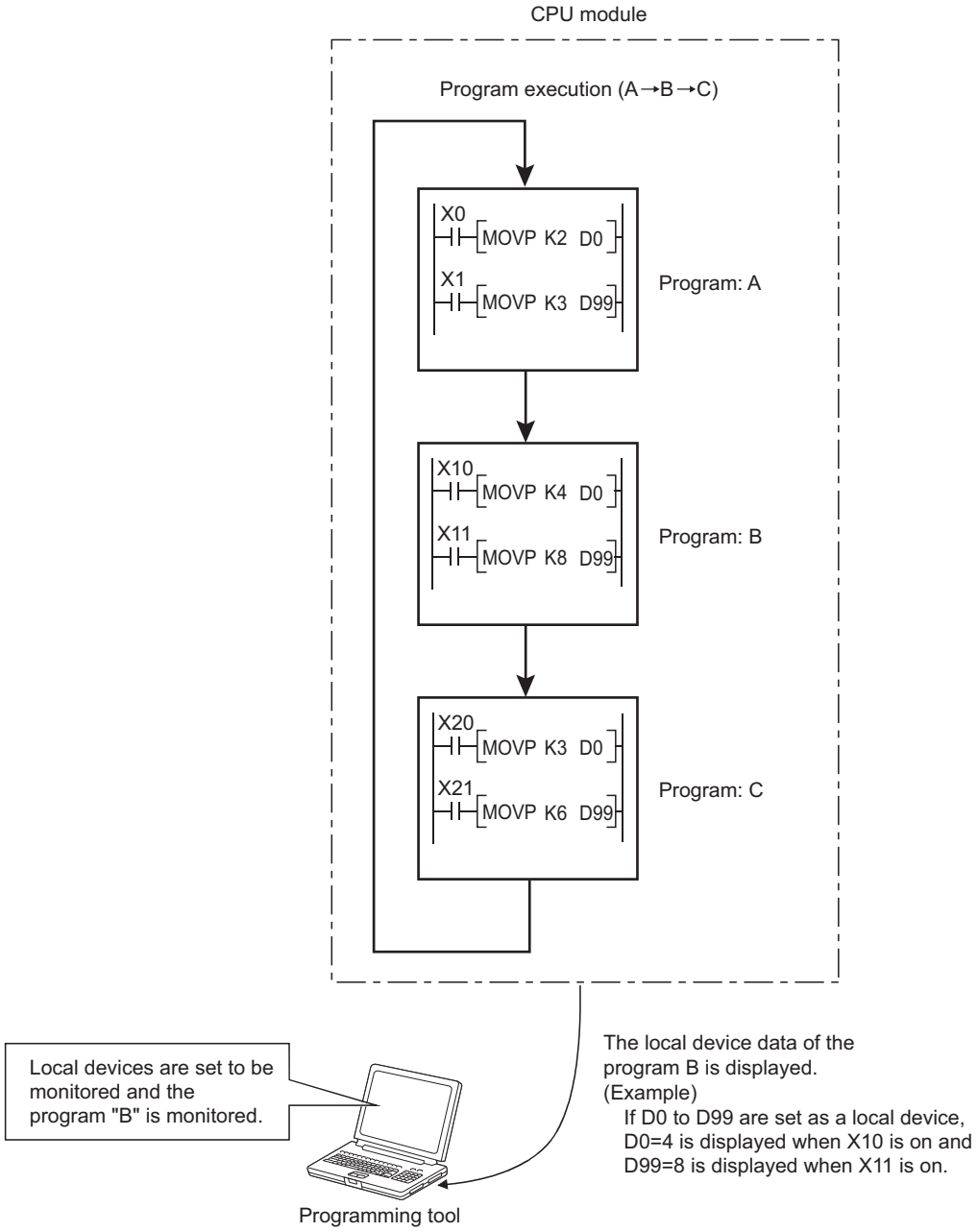
*1 When "Not Used" is set for "Local device" in File Usability Setting of the Program tab, the D0 value after execution of the specified program is monitored.

*2 When local devices are monitored using the built-in Ethernet ports of the QnUDE(H)CPU whose serial number (first five digits) is "11013" to "12051", the value in D100 after execution of the program "C" is monitored.

Note 3.3 Universal

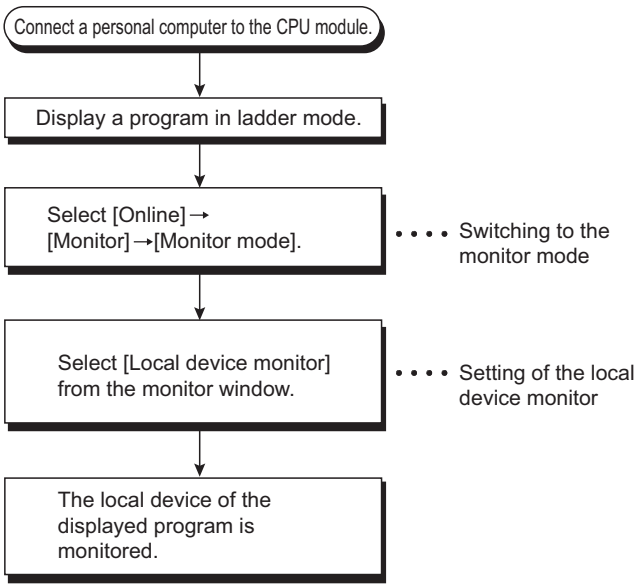
The Q00UJCPU does not support the use of local devices.

Ex. When local devices are set to be monitored and the program "B" is displayed for monitoring, the local device(s) used in the program "B" can be monitored.



(2) Monitoring procedure

The following shows the local device monitoring procedure.



(3) Precautions

(a) Local devices that can be monitored/tested by a single programming tool

One programming tool can monitor or test local devices in one program at a time. Local devices in multiple programs cannot be monitored or tested simultaneously.

(b) Number of programs that can be monitored/tested

Local devices in 16 programs can be monitored or tested simultaneously from multiple programming tools connected to the RS-232 interface of the CPU module or the serial communication module.

(c) Monitoring local devices in a stand-by type program

When local devices in a stand-by type program are monitored, data in local devices are saved and restored. For this reason, the scan time increases. (☞ Page 422, Section 6.2)

(d) Monitoring local devices in a fixed scan execution type program

When local devices in a fixed scan execution type program are monitored, data in local devices cannot be acquired and "0" is displayed.

(e) Clearing the PLC memory during local device monitoring

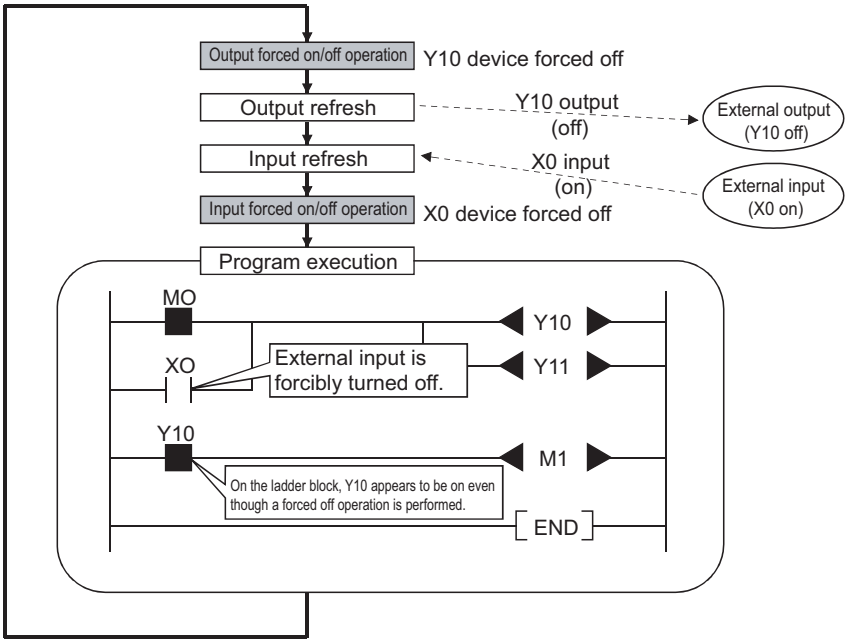
Local devices of monitoring programs are the target for the PLC memory clear operation during local device monitoring.

3.11.3 External input/output forced on/off Note 3.4

The external input/output can forcibly be turned on/off using a programming tool. The information registered for forced on/off can be cancelled by an operation from a programming tool.

(1) Input/output operation when a forced on/off operation is performed

There are three kinds of forced on/off operations: forced on ("Set forced ON"), forced off ("Set forced OFF"), and forced on/off cancellation ("Cancel it"). The following table lists the CPU module operation of when a forced on/off operation is performed.



Operation	Input (X) operation	Output (Y) operation
Forced on/off cancellation (no operation)	The CPU module performs sequence program operations using external inputs.	The CPU module outputs the results of sequence program operations externally.
Forced on	The CPU module performs sequence program operations using inputs forcibly turned on.	The CPU module outputs "on" externally regardless of the results of sequence program operations.
Forced off	The CPU module performs sequence program operations using inputs forcibly turned off.	The CPU module outputs "off" externally regardless of the results of sequence program operations.

Note 3.4 Universal

Before executing the function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used.

(Page 466, Appendix 2)

(2) Specifications

(a) CPU module status where input/output can forcibly be turned on/off

Forced on/off can be registered regardless of the operating status (RUN/STOP) of the CPU module.
Note, however, that only input can be forcibly turned on/off during a stop error.
The CPU module outputs on/off data only to Y device.

(b) Devices that can be registered

Forced on/of can be registered as many as the number of I/O device points in the CPU module.

(c) Target input/output

The following input/output are targeted for a forced on/off operation.

- Input (X) and output (Y) of modules mounted on the base unit
- Input (X) and output (Y) of the CPU module to be refreshed from LX/LY of a CC-Link IE Controller Network module or MELSECNET/H module
- Input (X) and output (Y) of the CPU module to be refreshed from RX/RY of a CC-Link IE Field Network master/local module or CC-Link module
- Input (X) and output (Y) of the CPU module to be refreshed from RX/RY of CC-Link IE Field Network Basic

When forcibly turning on/off the devices outside the above refresh ranges (for example, empty slots), only input/output in the CPU module device memory are turned on/off and the results are not output externally.

Point

In multiple CPU systems, inputs and outputs of control modules can forcibly turned on/off.
Even when inputs and outputs of non-control modules are registered for forced on/off, the input/output devices in other CPU modules and inputs and outputs of modules controlled by other CPU modules cannot be forcibly turned on/off. (The input/output devices in the own CPU module can forcibly turned on/off.)

(d) External input/output forced on/off timing

The following table lists the external input/output forced on/off timing.

Refresh area	Input	Output
Input and output of modules mounted on the base unit	• During END processing (input refresh) • At execution of the COM instruction (input refresh) • At execution of an instruction using direct access input (DX) (LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF) • At execution of the RFS or MTR instruction • At execution of an instruction used for a system interrupt (UDCNT1, UDCNT2, SPD)	• During END processing (output refresh) • At execution of the COM instruction (output refresh) • At execution of an instruction using direct access input (DX) (OUT, SET, DELTA, RST, PLS, PLF, FF, MC, SFT) • At execution of the RFS or MTR instruction • At execution of an instruction used for a system interrupt (PLSY, PWM)
Input and output of the CPU module to be refreshed from LX/LY of a CC-Link IE Controller Network module or MELSECNET/H module	• During END processing (refresh via CC-Link IE Controller Network or MELSECNET/H) • At execution of the COM instruction • At execution of the ZCOM instruction	
Input and output of the CPU module to be refreshed from RX/RY of a CC-Link IE Field Network master/local module or CC-Link module	• During END processing (auto refresh) • At execution of the COM instruction (auto refresh) • At execution of the ZCOM instruction (auto refresh)	
Input and output of the CPU module to be refreshed from RX/RY of CC-Link IE Field Network Basic	• During END processing (auto refresh) • At execution of the COM instruction (auto refresh)	

(e) Cancelling on/off registration data

The registered forced ON/OFF data can be canceled by a programming tool. Once the registered data is canceled, the status of the forced on/off registered devices will be as follows.

Forced on/off registered device		Sequence program operations (on/off) performed	Sequence program operations (on/off) not performed
Input	Input from modules mounted on the base unit	Uses the on/off status input from modules.	
	Input of the CPU module to be refreshed from LX of a CC-Link IE Controller Network module or MELSECNET/H module	Used the on/off status refreshed via CC-Link IE Controller Network module or MELSECNET/H module.	
	Input of the CPU module to be refreshed from RX of a CC-Link IE Field Network master/local module or CC-Link module	Uses the on/off status refreshed via CC-Link IE Field Network master/local module or CC-Link module.	
	Input of the CPU module to be refreshed from RX of CC-Link IE Field Network Basic	Uses the on/off status refreshed from the CPU module (when CC-Link IE Field Network Basic is used)	
	Input other than above (outside of the refresh range)	Uses the results of sequence program operations.	Holds the forced on/off status.
Output	Output from modules mounted on the base unit	Outputs the results of sequence program operations.	Holds the registered on/off status.
	Output of the CPU module to be refreshed from LY of a CC-Link IE Controller Network module or MELSECNET/H module		
	Output of the CPU module to be refreshed from RY of a CC-Link IE Field Network master/local module or CC-Link module		
	Output of the CPU module to be refreshed from RY of CC-Link IE Field Network Basic	Outputs the results of sequence program operations.*1	
	Output other than above (outside of the refresh range)	Outputs the results of sequence program operations. (The results are not output externally.)	Holds the forced on/off status.

*1 The result is output for one scan in accordance with the registered on/off state even after forced on/off registration data are canceled.

Forced on/off setting can be cleared by:

- powering off and then on the CPU module,
- resetting the CPU module by the RUN/STOP/RESET switch, or
- resetting the CPU module by the remote RESET operation.

(f) Number of devices that can be registered

Forced on/off can be registered for 32 devices in total.

(g) When output Y contact is used in a sequence program

On/off operations in a sequence program are given priority.

(h) Checking forced on/off execution status

The execution status can be checked by:

- reading the forced on/off registration status of a programming tool.
- flashing of the MODE LED (green), (The MODE LED flashes in green when at least one forced on/off is registered.) or
- the on status of the 1st bit in SD840 (Debug function usage).

Point

- The MODE LED also flashes in green when the executional conditioned device test function is used. To check the registration status using the MODE LED, check the status of the executional conditioned device test function as well. (☞ Page 160, Section 3.11.4 (3))
- When using SD840 to check the registration or cancellation status, remind that SD840 is used to check the status of the executional conditioned device test function as well.

(i) Forcibly turning input or output on/off from multiple programming tools

Forced on/off can be registered to a single CPU module from multiple programming tools connected via network. If forced on/off is registered to the same device, the last registration will be enabled. Therefore, the forced on/off status different from the actual status in the CPU module may be displayed on the programming tool that registered forced on/off earlier. When the forced on/off registration is performed from multiple programming tools, click the "Update status" button to update the registered data and execute the function.

(3) Operating procedure

For the operating procedure of forced input output registration/cancellation, refer to the following.

📖 Operating manual for the programming tool used

3.11.4 Executional conditioned device test Note 3.5

This function changes a device value within the specified step of a program.

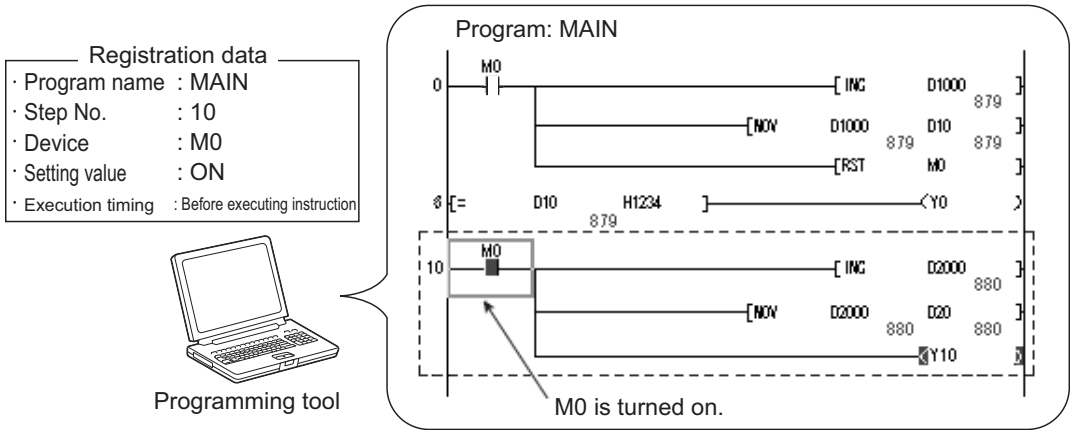
This enables debugging of the specified ladder block without modifying the program.*1

*1 The executional conditioned device test is not available for the SFC program.

(1) Operation of the executional conditioned device test

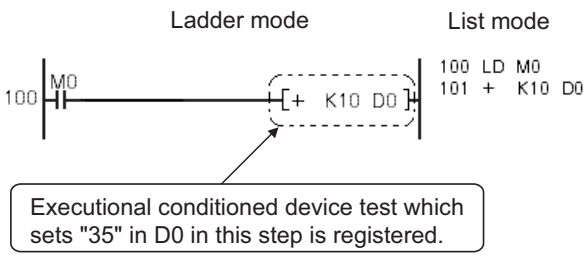
A device value will be changed based on the registration data once after the executional conditioned device test setting is registered.

The changed device value becomes enabled in the ladder blocks of the specified step number and later.

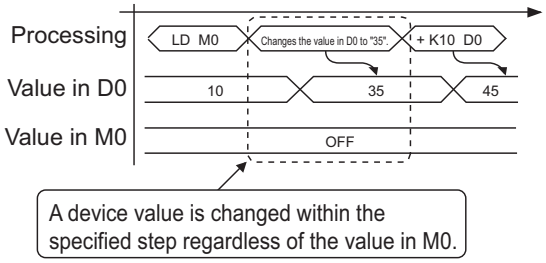


Note that a device value is changed within the specified step regardless of an execution status of the instruction in the specified step.

<Program example>



<Operation>



Note 3.5 Universal

Before executing the function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used.

(Page 466, Appendix 2)

(2) Available devices and number of settable devices

The following table lists available devices and the number of settable devices.

Type	Available device	Number of settable devices
Bit device	X, Y, M, L, B, F, SB, V, SM, T (contact), ST (contact), C (contact), J□\X, J□\Y, J□\B, J□\SB, FX, FY, DX, and DY	Up to 32 (in total)
Word device	T (current value), ST (current value), C (current value), D*1, SD, W*2, SW, R, ZR, Z, U□\G□, U3E□\G□, J□\W□, J□\SW□, and FD	
	Digit-specified bit device: X, Y, M, L, F, SM, V, B, SB, J□\X, J□\Y, J□\B, and J□\SB	
	Indirect specification (@D0): D, SD, W, SW, R, and ZR (devices specified with @)	

*1 The extended data register (D) is included.

*2 The extended link register (W) is included.

Note, when the write-protect function for device data (from outside the CPU module) is enabled, that indirect specified/index-modified devices cannot be set for the test. When the write-protected range is set to the file register (ZR (R)), the file register (R) cannot be set, too.

(3) How to check the execution status

The execution status of registered executional conditioned device test can be checked in three different ways:

- By the display on the screen for checking the registration status in a programming tool
- By the flash of the MODE LED in green
- By the on status of the first bit of SD840 (Debug function usage)

Point

- The MODE LED also flashes in green when the external input/output forced on/off function is used. To check the execution status using the MODE LED, check the status of the external input/output forced on/off function as well. (Page 158, Section 3.11.3 (2) (h))
- When using SD840 to check the execution status, remind that SD840 is used to check the status of the external input/output forced on/off function as well.

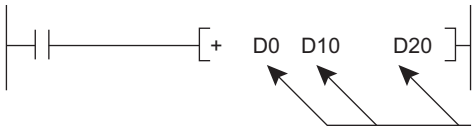
(4) Registering executional conditioned device test settings

For how to register executional conditioned device test settings, refer to the following manual.

Operating manual for the programming tool used

(a) Multiple executional conditioned device test registrations for the same step number

Multiple executional conditioned device test settings can be registered for one step number.



Devices that executional conditioned device test settings can be registered for the start step of the + instruction

However, if multiple executional conditioned device test settings with same device name and same execution timing are registered for the same step number, the registration data will be overwritten. (Even though the same device is specified, if the execution timing differs, two settings can be registered for one step.)

Point

- When setting a word device with a different data type, a device is regarded as the same device.

  Ex. When a word device is set in the order of "D100 (16 bit integer)" and then "D100 (Real number (single precision))", "D100 (Real number (single precision))" is registered.

- When setting a device with a different modification method (such as a bit-specified word device, digit-specified bit device, or index-modified device), a device is regarded as a different device.

  Ex. When a word device is set in the order of "D100.F" and then "D100Z0 (Real number (single precision)), both devices are registered.

(b) Step to be specified for executional conditioned device test registration

Any step number (0 to step number for the END instruction) in a sequence program can be specified.

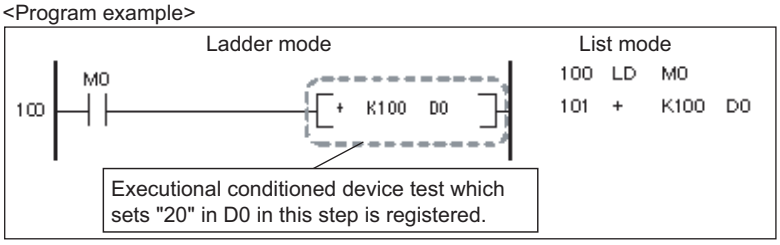
Point

Be sure to specify the start step of each instruction.

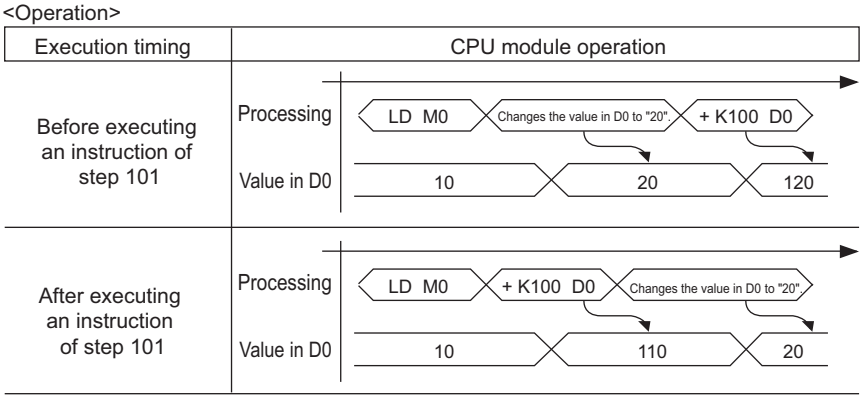
(c) Execution timing

Timing to change a device value can be specified. A device value can be changed either before or after an instruction of the specified step is executed. The following figure shows the module operation based on the execution timing.

<Program example>



<Operation>



Note that there may be a case where a device value will not be changed depending on the execution timing even though the specified step is executed. The following instructions need to be noted when registering executional conditioned device test settings.

- Instructions that do not change device values[*1]
  A device value will not be changed by executing the executional conditioned device test when the execution timing has been set to "After executing instruction", specifying the step for instructions that do not execute the next step, such as branch instructions.

*1 If the execution condition of an instruction is not satisfied, a device value will be changed based on the registration data.

The following table lists the instructions that do not change device values.

No.	Classification	Instruction	Operation
1	Stop	STOP	• When the execution condition for an instruction is satisfied. A device value will not be changed even when the specified step is executed. • When execution condition for an instruction is not satisfied. A device value will be changed after the specified step is executed.
2	Jump	CJ	
3		SCJ	
4		GOEND	
5	Repeat (Loop)	BREAK(P)	
6	Subroutine program call	CALL(P)	
7		FCALL(P)	
8		ECALL(P)	
9		EFCALL(P)	
10		XCALL	
11	End	FEND	A device value will not be changed even when the specified step is executed.
12	Jump	JMP	
13	Return from subroutine program	RET	
14	Return from interrupt program	IRET	

• FOR/NEXT instructions

When the executional conditioned device test setting is registered specifying the step for the FOR or NEXT instruction, timing of device value change is different from the timing when steps for other instructions are specified. The following table lists the device value change timing based on the execution timing.

Instruction of the specified step	Execution timing	
	Before executing instruction	After executing instruction
FOR	Executed once before the start of loop processing.	Executed once after the start of loop processing. (Device values are changed before the execution of the program between the FOR and NEXT instructions.)
NEXT	• CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU: Executed in every loop processing. (Device values are changed after the execution of the program between the FOR and NEXT instructions.) • High-speed Universal model QCPU and Universal model Process CPU: Executed once after the start of loop processing. (Device values are changed after the execution of the program between the FOR and NEXT instructions.)	Executed once after the start of loop processing.

• END instruction

When the executional conditioned device test setting is registered specifying the step for the END instruction, the execution timing is restricted to "Before executing instruction" only. If "After executing instruction" is set, the CPU module sends a registration error to the programming tool.

(d) Number of settings that can be registered simultaneously in one scan

Eight executional conditioned device test settings can be registered into the CPU module simultaneously in one scan. When nine or more executional conditioned device test settings are to be registered simultaneously, they will be registered over multiple scans.

(5) Checking/disabling executional conditioned device test settings

For how to check/disable executional conditioned device test settings, refer to the following.

Operating manual for the programming tool used

(a) Usage of the executional conditioned device test

Usage of the executional conditioned device test can be checked in the special register (SD840).

(b) Number of settings that can be disabled simultaneously in one scan

Eight executional conditioned device test settings can be disabled simultaneously in one scan. When nine or more executional conditioned device test settings are to be disabled simultaneously, they will be disabled over multiple scans.

(6) Batch-disabling executional conditioned device test settings

For how to batch-disable executional conditioned device test settings, refer to the following.

Operating manual for the programming tool used

(7) Cases where executional conditioned device test settings cannot be registered or disabled

In the following cases, executional conditioned device test setting cannot be registered or disabled.
When multiple settings are to be registered, no setting can be registered if any of the settings applies to the cases below.

(a) Executional conditioned device test settings cannot be registered

- Specified program does not exist.
- Specified step does not exist.
- Specified device does not exist.
- The number of registered executional conditioned device test settings exceeds 32.

(b) Executional conditioned device test settings cannot be disabled

- Specified program does not exist.
- Specified step does not exist.
- Specified device does not exist.
- No executional conditioned device test setting has been registered.

(8) Precautions

(a) Operations from multiple programming tools

Executional conditioned device test settings can be registered in the same CPU module from multiple programming tools connected via network. Note, however, that if multiple executional conditioned device test settings are registered with the same device name in the same step, the registration data will be overwritten. When registering executional conditioned device data settings from multiple programming tools, update the data first by clicking the Read File... button, and register the settings.

(b) Priority

If any of the following functions is set in the same step number that is specified by the executional conditioned device test setting, the executional conditioned device test is given the priority to other functions.

- Monitor condition setting
- Sampling trace function (trace point)
- Sampling trace function (trigger point)
- Scan time measurement (start step)
- Scan time measurement (end step)

(c) Disabling executional conditioned device test settings

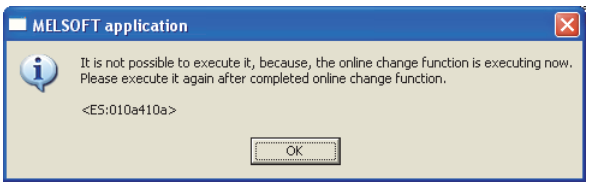
Executional conditioned device test setting can be disabled by any of the following operations, in addition to the operation of a programming tool.

- Powering off and then on the CPU module
- Resetting the CPU module
- Writing program files to the program memory while the CPU module is in the STOP status
- Clearing the program memory data while the CPU module is in the STOP status
- Formatting the program memory while the CPU module is in the STOP status

(d) Writing data while the CPU module is in the RUN status

The CPU module operation will be as follows if the executional conditioned device test and the online change function are executed simultaneously.

- When the executional conditioned device test is executed during execution of the online change function
  The online change function completes normally. However, the executional conditioned device test cannot be executed. The following message box will appear. Execute the executional conditioned device test again after the online change has completed.

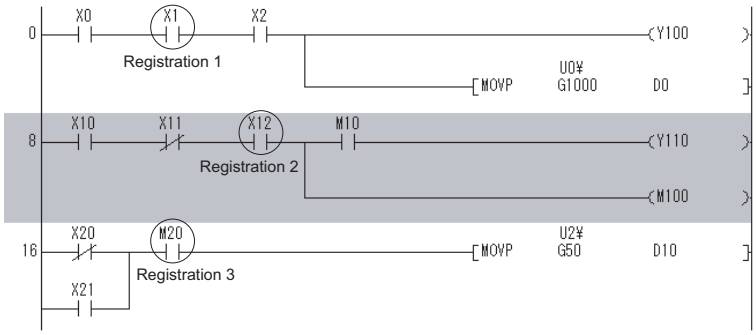


- When the online change function is executed during execution of the executional conditioned device test
  The online change function completes normally. If any executional conditioned device test setting has been registered in the program to be changed online, the corresponding setting will be disabled.
  (☞ Page 166, Section 3.11.4 (8) (e))

(e) Online change of the CPU module with executional conditioned device test registration

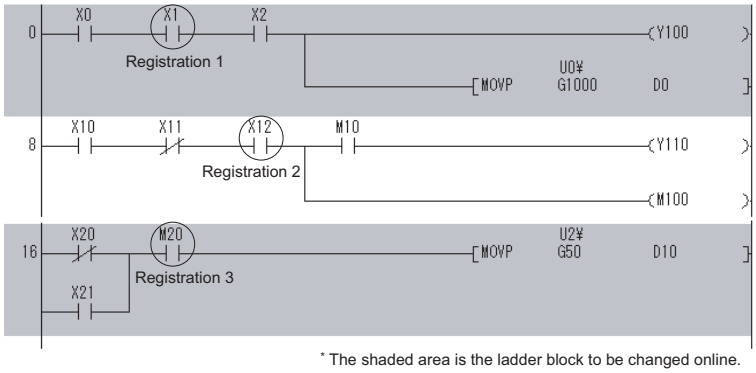
- Online change (ladder mode)
  If any executional conditioned device test setting has been registered in the ladder block to be changed online, the CPU module disables the corresponding setting.

Example 1) Step numbers of registrations 1 to 3 are specified in the executional conditioned device test settings. When the ladder block including the registration 2 is changed online, the registration 2 is disabled during execution of the online change function. Since the registrations 1 and 3 are not included in the change target program, they are not disabled.



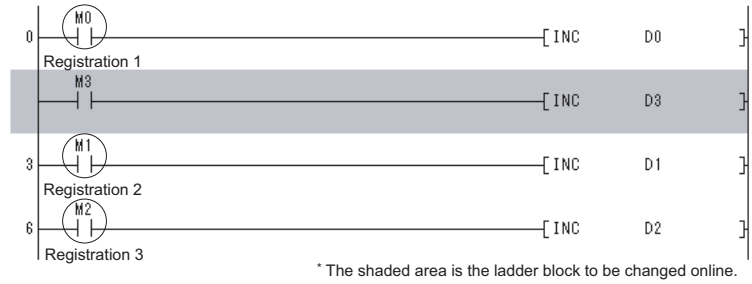
* The shaded area is the ladder block to be changed online.

Example 2) When multiple ladder blocks are changed online, ladder blocks between the change target ladder blocks will be included in the change target. For this reason, if the online change function is executed as follows, all registrations 1 to 3 are disabled.



* The shaded area is the ladder block to be changed online.

Example 3) When a ladder block is added online, the executional conditioned device test setting included in the ladder block followed after the added ladder block will be disabled. For this reason, if the online change function is executed as follows, the registration 2 is disabled.



* The shaded area is the ladder block to be changed online.

- Online change (files)
  All executional conditioned device test settings registered to the program in the online change target file are disabled.

(f) Precautions for specifying an index-modified device

If an index-modified device name is specified to register the executional conditioned device test setting, the CPU module does not check whether the specified device is within the setting range.
If the index-modified device is out of the device range or on the boundary of devices, a device value will not be changed within the specified step.

(g) Precautions for specifying an indirectly-specified device

If indirectly-specified device name is specified to register the executional conditioned device test setting, the CPU module does not check whether the specified device is within the setting range.
If the indirectly-specified device is out of the device range or on the boundary of devices, a device value will not be changed within the specified step.

(h) Precautions for specifying the file register

If the file register is specified to register the executional conditioned device test setting, the CPU module does not check the file register file assignment and the file register number range.
A file register value will not be changed within the specified step in the following cases.

- The file register file is not assigned.
- The specified file register number is out of the file register range.

3.12 Writing Programs While CPU Module is in RUN Status

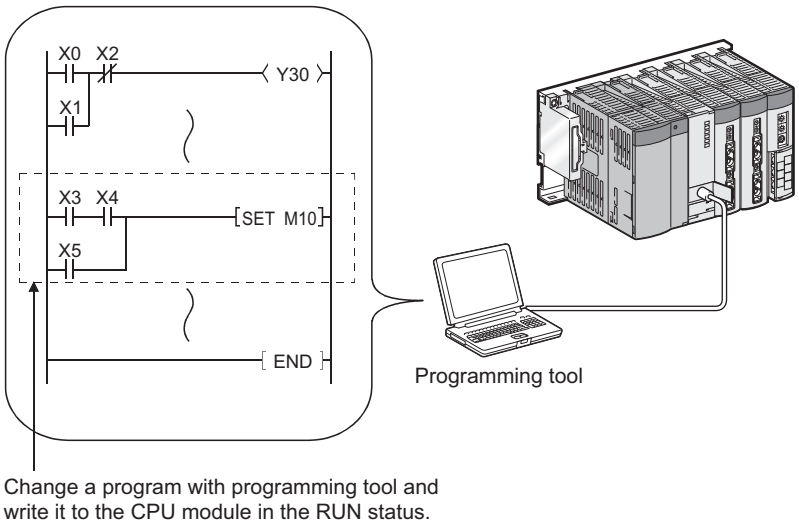
There are two ways of writing programs in the RUN status.

- Online change (ladder mode) : ☞ Page 168, Section 3.12.1
- Online change (files) : ☞ Page 171, Section 3.12.2

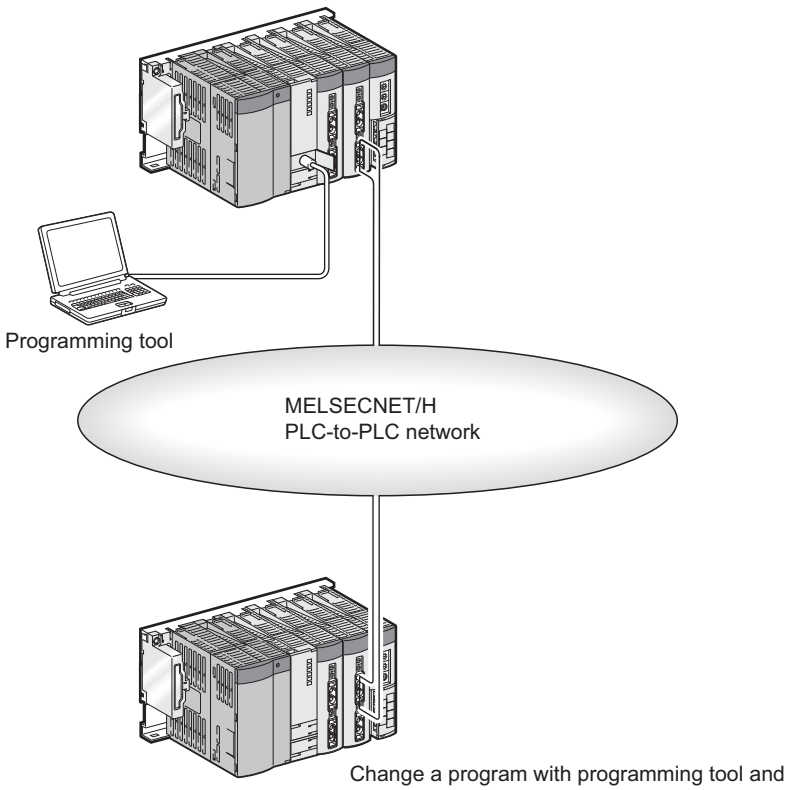
Data can also be written in the RUN status using a pointer. (☞ Page 192, Section 3.15.2)

3.12.1 Online change (ladder mode)

This function writes programs to the CPU module in the RUN status. This function enables the program in ladder mode to be changed without stopping the program operation in the CPU module.



Also, programs can be written in the RUN status from a programming tool connected to another station on the network.



(1) Memory for online change

A program cache memory (program memory) is available.

(2) Number of steps that can be batch-written by online change

Up to 512 steps can be batch-written.

(3) Changing the reserved area for online change

A program file has an area designated as reserved area for online change to support the online change that changes program file size.

The following provides precautions when the size of reserved area for online change is changed.

(a) Size of a program file

The size of a program file is addition of created program size and reserved area for online change.

(b) When program file size is increased from the secured capacity

If the size secured for the program file (size including reserved area for online change) is exceeded after a program is written in the RUN status, the reserved area for online change can be reset before the writing if the user memory area has space.

(c) Increase in the scan time

The scan time is increased when reserved area for online change is reset when programs are written in the RUN status.

For increase in the scan time, refer to Page 487, Appendix 3.3 (7).

(4) Operations prohibited when programs are written to the CPU module in the RUN status, TC setting value is changed, or data are transferred from a program cache memory to a program memory

Refer to Page 173, Section 3.12.3 (2).

(5) Instructions that do not operate normally when programs are written to the CPU module in the RUN status

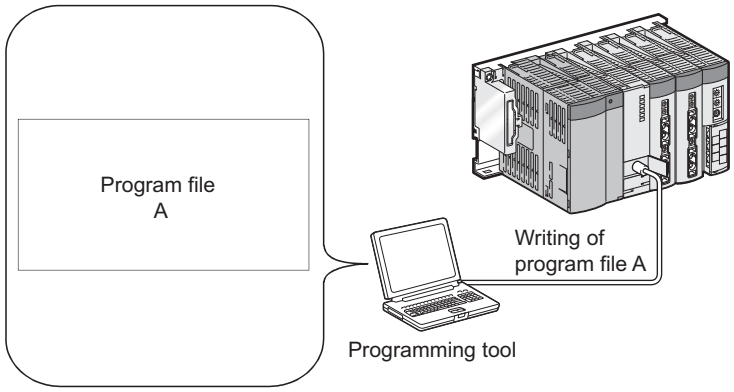
Refer to Page 174, Section 3.12.3 (3).

3.12.2 Online change (files)

This function batch-writes files listed in the following table to the CPU module in the RUN status by online operation from a programming tool.

○ : Can be written, △ : Cannot be written while being accessed, × : Cannot be written

File name	CPU module built-in memory			Memory card (RAM)	Memory card (ROM)		Memory card (SD)
	Program memory	Standard RAM	Standard ROM	SRAM card	Flash card	ATA card	SD memory card
Parameter	×	×	×	×	×	×	×
Intelligent function module parameter	×	×	×	×	×	×	×
Program	○	×	○	○	×	○	○
Device comment	○	×	△	△	×	△	△
Initial device value	×	×	×	×	×	×	×
File register	×	△	×	△	×	×	×
Local device	×	×	×	×	×	×	×
Sampling trace file	×	○	×	○	×	×	×
Programmable controller user data	×	×	○	×	×	○	○



(1) Availability

(a) For the Q00UJCPU, Q00UCPU, and Q01UCPU

The function cannot be performed in the following cases.

- A program memory does not have enough area for storing a program file to be written.
- A program memory stores the maximum number of files that can be stored.

(b) For the Q02UCPU, QnUD(H)CPU, and Built-in Ethernet port QCPU

Files can be written in the RUN status, regardless of free space in the program memory and the number of files to be stored.

(2) Increase in the scan time

The scan time increases when a program file is written to the CPU module in the RUN status.
For increase in the scan time, refer to Page 487, Appendix 3.3 (7).

(3) Online change (files) from multiple locations

Do not simultaneously write files to one CPU module in the RUN status from multiple locations.
Doing so may delete program files.

(4) Online change (files) of SFC programs

SFC programs cannot be written in units of files to the CPU module in the RUN status.

(5) Operations prohibited when programs are written to the CPU module in the RUN status, TC setting value is changed, or data are transferred from a program cache memory to a program memory

Refer to Page 173, Section 3.12.3 (2).

(6) Instructions that do not operate normally when files are written to the CPU module in the RUN status

Refer to Page 174, Section 3.12.3 (3).

3.12.3 Precautions for online change

The following shows precautions for online change.

(1) Online change during boot operation

When data are written to the CPU module in the RUN status during boot operation, the status of boot source program is not changed.

(2) Operations prohibited when programs are written to the CPU module in the RUN status, TC setting value is changed, or data are transferred from a program cache memory to a program memory

Do not perform the following operations.

(a) Power-off or reset

The following operations are not normally completed if they are performed during online change, TC setting value change, or data transfer from the program cache memory to the program memory.
If performed, write the data to the CPU module again.

- Power-off
- Reset

(b) Operations from a programming tool

The following operations cannot be performed during online change, TC setting value change, or data transfer from the program cache memory to the program memory. If performed, an error is displayed on the programming tool. Perform the following operations after online change.

- Online change (ladder mode), online change (files)
- TC setting value change
- Data transfer to the program memory Note 3.6
- Write to PLC (Flash ROM)

Note 3.6 Universal

With the Universal model QCPU whose serial number (first five digits) is "12012" or later, this function can be executed while data are being transferred from the program cache memory to the program memory.

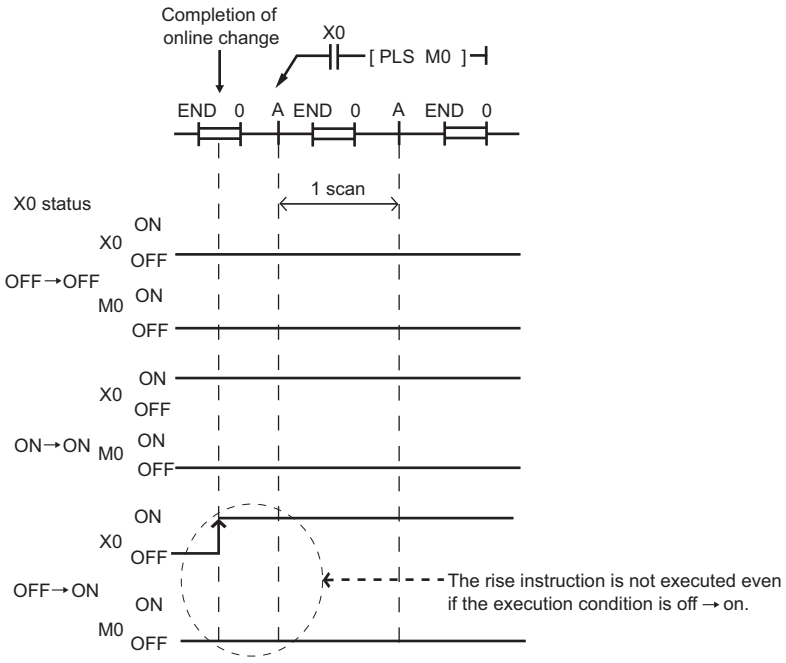
(3) Instructions that do not operate normally during online change

When data are written to the CPU module in the RUN status, the following instructions do not operate normally.

- Rise instruction
- SCJ instruction
- STMR instruction

(a) Rise instruction

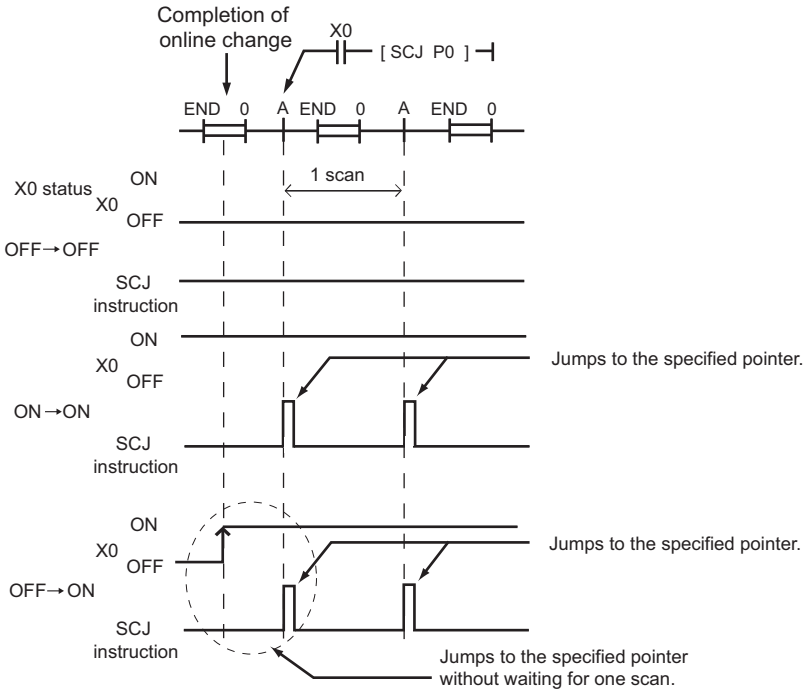
The rise instruction is not executed when the instruction is in the data written to the CPU module in the RUN status, even if the execution condition (off → on) is met.



The corresponding rise instructions are PLS and □P.

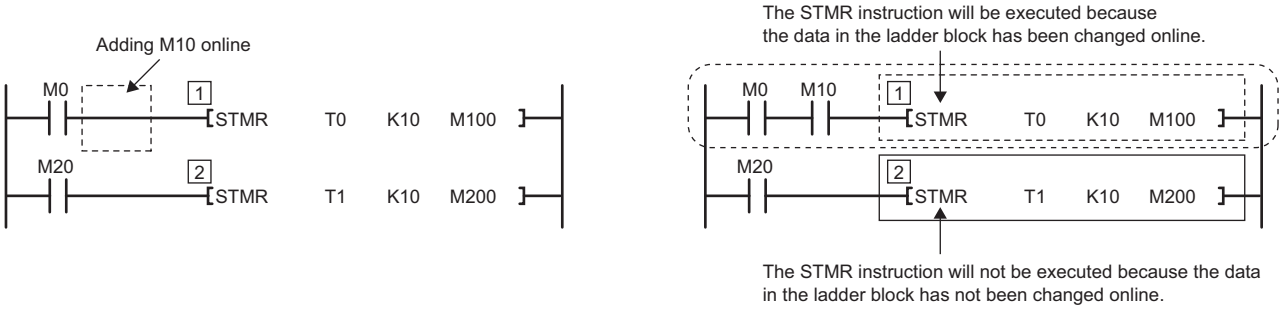
(b) SCJ instruction

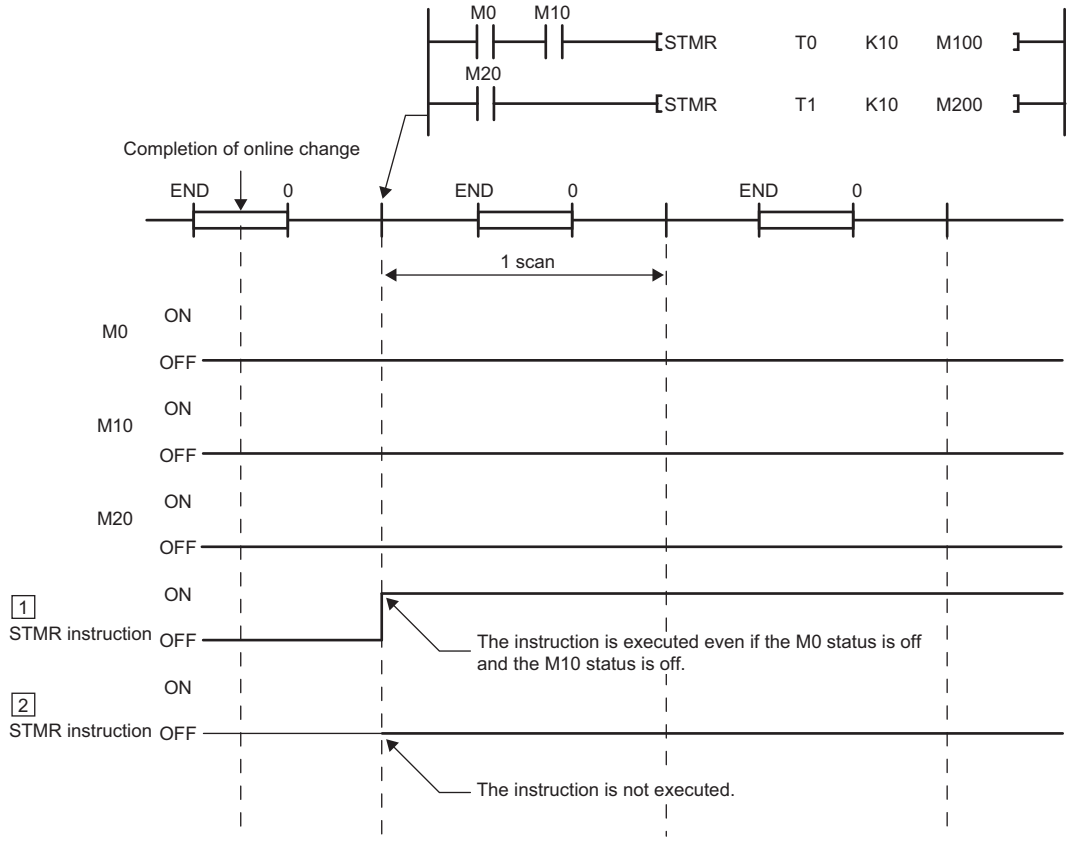
When the SCJ instruction is in the data written to the CPU module in the RUN status and the execution condition is on at completion of the writing, a jump to the specified pointer is made without a wait of one scan.



(c) STMR instruction

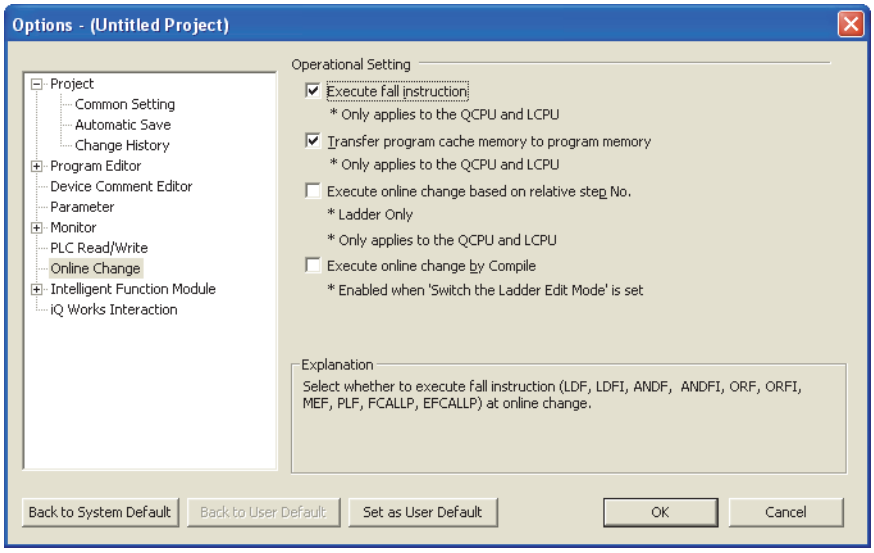
Note that the STMR instruction operates when the instruction is used within the range written data by the online program change.



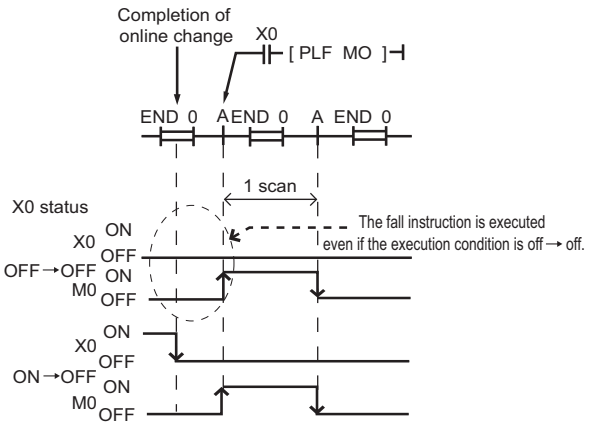


Point

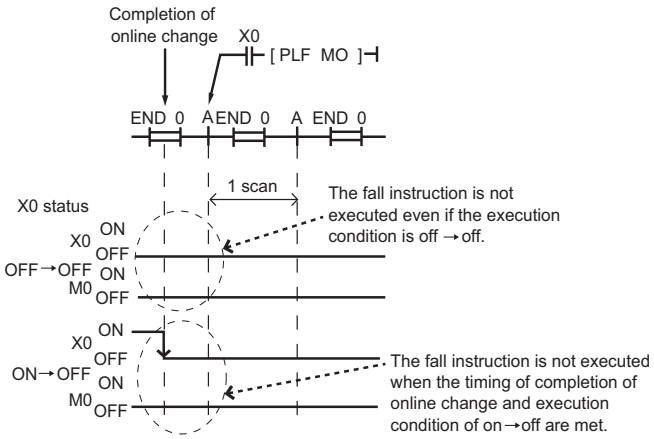
When "Execute fall instruction" is checked in the "Options" window of a programming tool, the fall instruction is executed when the instruction is in the data written to the CPU module in the RUN status, even if the execution condition (on → off) is not met. (Same operation as the High Performance model QCPU)



The corresponding fall instructions are LDF, ANDF, ORF, MEF, PLF, FCALLP, and EFCALLP. The following describes the operation with and without "Execute fall instruction" selected.



(a) Operation when selecting "Execute fall instruction"



(b) Operation when not selecting "Execute fall instruction"

(4) Writing to the program memory during online change and TC setting value change

Due to automatic data transfer to the program memory, the time takes to write data to the CPU module during the RUN status and to change TC setting value extends by the time shown in the following table.

Ts: Scan time (s)

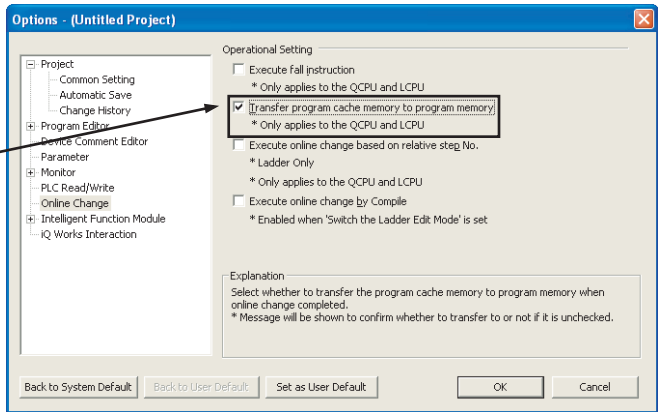
CPU module	Transfer time
Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU	Ts × 320 + 4.8 (s)
Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU	Ts × 260 + 4.7 (s)
Q10UD(E)HCPU	Ts × 439 + 6.2 (s)
Q13UD(E)HCPU	Ts × 600 + 8.0 (s)
Q20UD(E)HCPU	Ts × 839 + 11.4 (s)
Q26UD(E)HCPU	Ts × 1100 + 15.0 (s)
Q50UDEHCPU	Ts × 2450 + 17.0 (s)
Q100UDEHCPU	Ts × 4550 + 9.0 (s)
Q03UDVCPU	Ts × 5 + 0.65 (s)
Q04UDVCPU, Q04UDPVCPU	Ts × 5 + 0.85 (s)
Q06UDVCPU, Q06UDPVCPU	Ts × 5 + 1.25 (s)
Q13UDVCPU, Q13UDPVCPU	Ts × 5 + 1.85 (s)
Q26UDVCPU, Q26UDPVCPU	Ts × 5 + 3.7 (s)

Since the number of writes to the program memory (Flash ROM) is limited (up to 100,000 times), set the automatic transfer to the program memory to be disabled when data are written to the CPU module in the RUN status and changing TC setting value frequently.

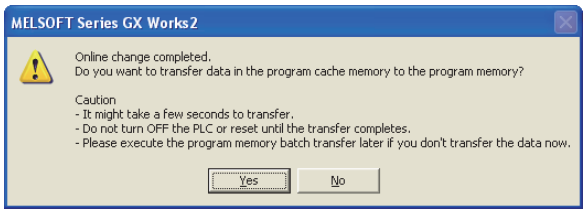
Point

Automatic data transfer to the program memory can be disabled in the "Options" window of the programming tool.

To avoid automatic transfer of program memory data, clear the checkbox. (Selected by default.)



When the automatic data transfer is disabled, the following message appears after online change.



Selecting "Yes" transfers data to the program memory. When selecting "No", execute "Program Memory Batch Download" from the programming tool.

Program transfer status can be checked in the special relay (SM165). Note 3.7

When SM165 is on, the program memory batch transfer has not completed. When SM165 is off, the program memory batch transfer has completed.

Note 3.7 Universal

When checking the transfer status with the Q02UCPU, Q03UDCPU, Q04UDHCPU, or Q06UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

3.13 Execution Time Measurement

This function displays the processing time of the program being executed.

(1) Applications and types

This function can be used to know the effect of processing time of each program on the total scan time when the system is adjusted.

There are following three types.

- Program monitor list : ☞ Page 180, Section 3.13.1
- Interrupt program list monitor : ☞ Page 180, Section 3.13.2
- Scan time measurement : ☞ Page 181, Section 3.13.3

3.13.1 Program monitor list

The scan time, number of execution times, and processing time by item can be displayed for each program.

(1) Execution

For how to execute the program monitor list, refer to the following.

📖 Operating manual for the programming tool used

(2) Precaution

(a) When the POFF instruction is in the program

When the POFF instruction is executed, a non-execution processing is performed for one scan. The number of executions includes the executions of non-execution processing. For details of the POFF instruction, refer to the following.

📖 MELSEC-Q/L Programming Manual (Common Instruction)

3.13.2 Interrupt program monitor list

This function displays the number of interrupt program executions. This function is used to check the execution status of an interrupt program.

(1) Execution

For how to execute the interrupt program monitor list, refer to the following.

📖 Operating manual for the programming tool used

3.13.3 Scan time measurement Note 3.8

This function displays the processing time of set program section during ladder monitoring. The time required for the subroutine and interrupt programs can be measured.

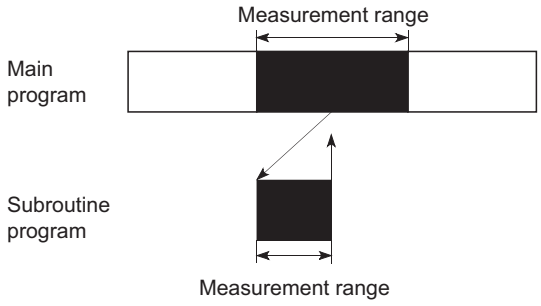
(1) Range specification of scan time measurement

There are following two types for specifying a scan time measurement range.

- Setting on the ladder monitor screen
- Setting on the scan time measurement screen

(2) When the subroutine program call instruction is in the measurement range

When the subroutine program call instruction (CALL) is in the range of scan time measurement, the scan time includes the time required for processing a subroutine program.



(3) When interrupt programs/fixed scan execution type programs are executed in the scan time measurement range

The execution time of interrupt programs and fixed scan execution type programs are added.

(4) Execution

For how to execute the scan time measurement, refer to the following.

Operating manual for the programming tool used

Note 3.8 Universal

Before executing the function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

(5) Precautions

(a) Measurement range setting

Set the measurement range so that "Start step < End step" is satisfied.

(b) Minimum unit of measurement time

The minimum unit of measurement time is 0.01ms.

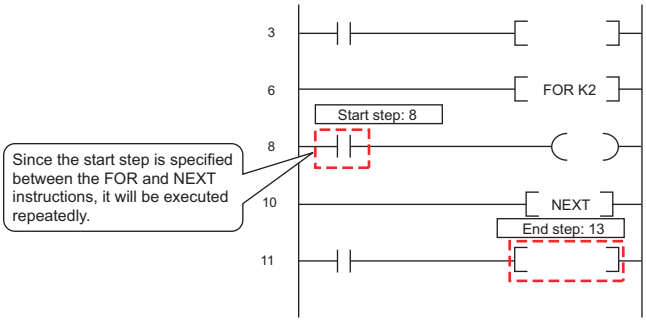
If the measurement time is less than 0.01ms, 0.000ms is displayed.

(c) When steps are specified between the FOR and NEXT instructions

Scan time required to execute the program between the specified steps is measured.

(d) When only the start step is specified between the FOR and NEXT instructions

- CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU
  Since the start step is executed repeatedly, scan time cannot be measured. (Time values are not updated on the Scan Time Measurement window of a programming tool.)



- High-speed Universal model QCPU and Universal model Process CPU
  The High-speed Universal model QCPU and Universal model Process CPU recognize only the first start step and ignores the second and later start steps. Scan time required to execute the program between the first start step and the end step is measured.

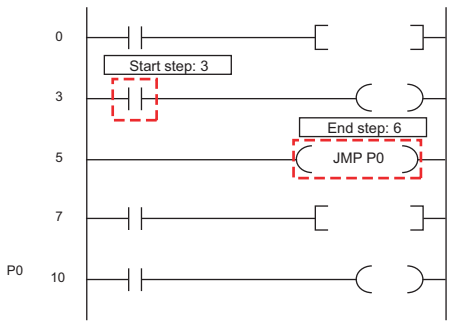
(e) When scan time cannot be measured

Scan time cannot be measured for scans executed across multiple program files.

Scan time is not updated on the Scan Time Measurement screen in the following case.

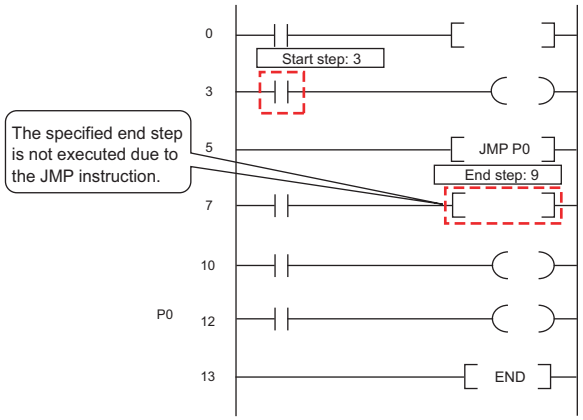
- When the branch instruction is specified to the end step

  Ex. The JMP instruction is specified to the end step.



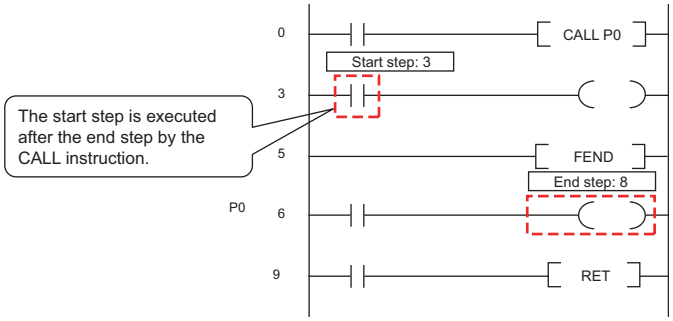
- When only the start step is executed

Ex. The specified end step is not executed by the JMP instruction.



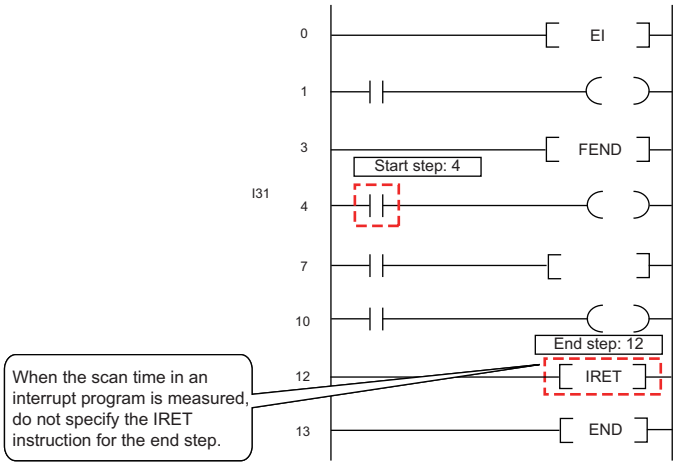
- When the end step is executed before the start step

Ex. The start step is specified as the next step of the CALL instruction and the end step is specified in a subroutine program executed by the CALL instruction.



- When the IRET instruction, FEND instruction, BREAK instruction, or RET instruction is specified for the end step

Ex. In an interrupt program by I31, the IRET instruction is specified for the end step.



3.14 Sampling Trace Function Note 3.9

This function samples the data of the specified device at a preset timing and at a preset interval (sampling cycle), and then stores the trace results in the sampling trace file.

(1) Application

The change in the device data used in the program during debugging can be checked at a specified timing. Also, this function is used to read the device data at trigger condition establishment.

(2) Sampling trace file

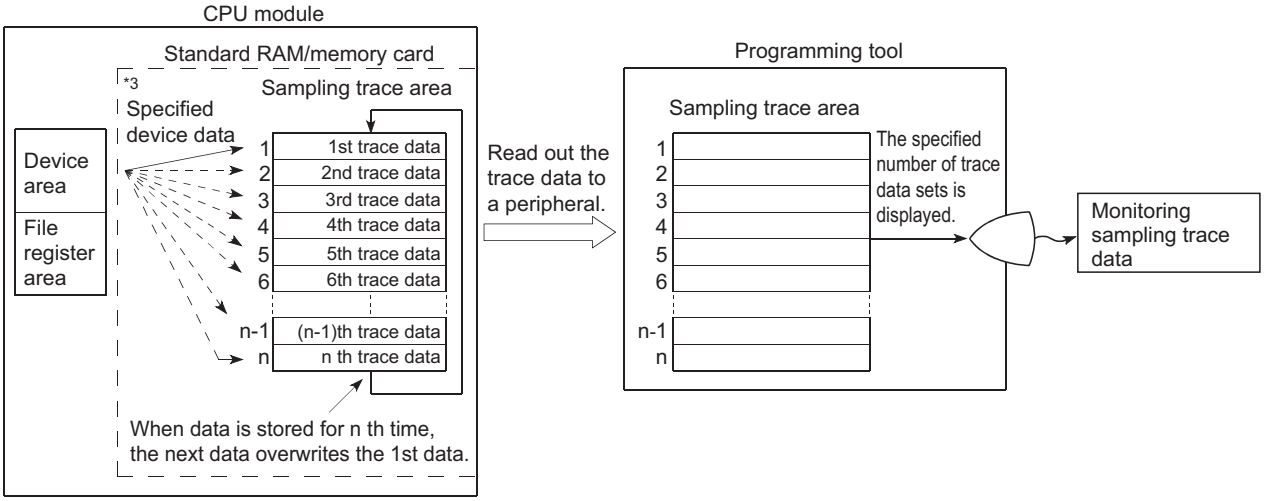
This file stores the trace setting necessary for executing the function and trace results. A sampling trace file is stored in the following memory.

CPU module	Memory
Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU	Standard RAM
Q02UCPU, QnUD(H)CPU, QnUDE(H)CPU	Standard RAM or SRAM card

(3) Sampling trace operation

(a) Operation of the CPU module

When a sampling trace trigger is issued by a programming tool, the CPU module executes traces for the preset number of times. The number of traces will be a value of which the number of bytes for the sampling trace area divided by the number of bytes of the specified device (N1 + N2 + N3 + word device points × 2 + (bit device points/16) × 2).*1 *2



*1 Round up the result of "bit device points/16" in the expression to the right of the decimal point.

*2 Add the following values to N1 to N3 according to the items selected under the trace additional information of the trace condition setting.
- N1: When "Time(sec)" is selected, add "4".
- N2: When "Step no." is selected, add "10".
- N3: When "Program name" is selected, add "8".

*3 When the trigger is issued, the CPU module samples data for the preset number of times and latches the data in the sampling trace area.

Note 3.9 Universal

The Q00UJCPU does not support this function.

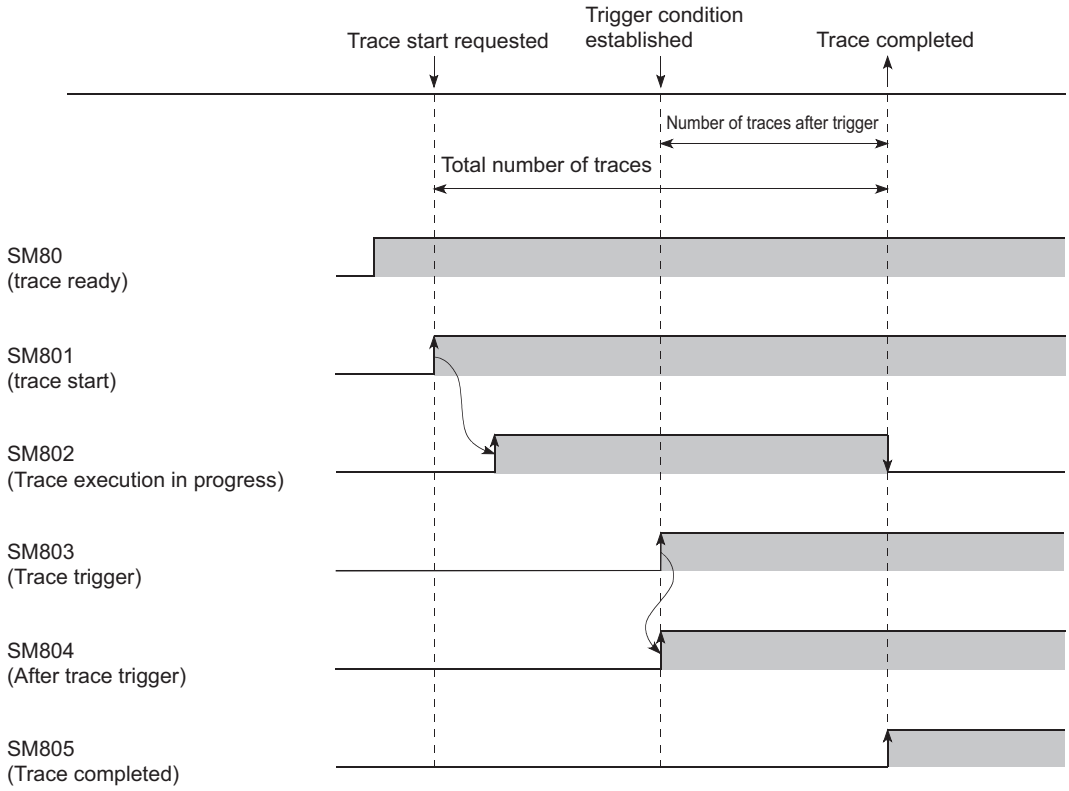
(b) Operation of the special relay

• When the sampling trace is executed normally

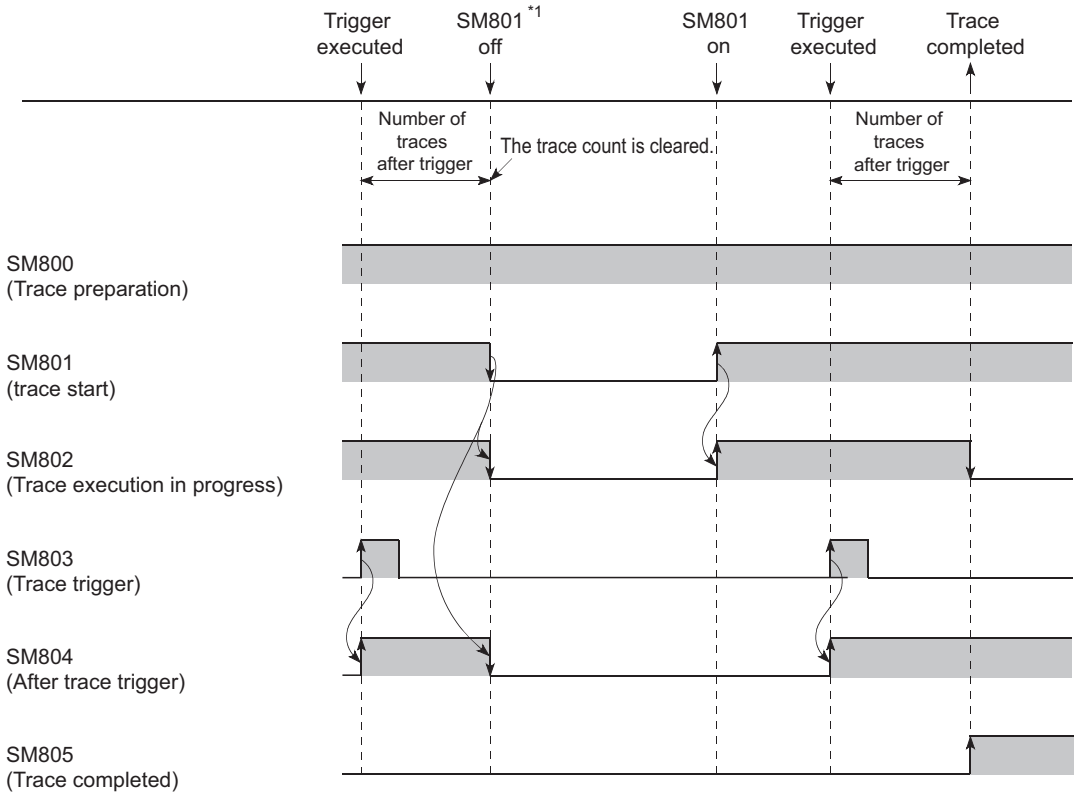
The execution status of the sampling trace can be checked in the special relays below.

Number	Name	Description
SM800	Trace preparation	Turns on when the trace setting in a programming tool is written to the CPU module. The relay is used to check whether the sampling trace execution is enabled or not.
SM801	Trace start	Turns on when the sampling trace is started.
SM802	Trace execution in progress	Turns on during sampling trace execution. The relay is used to check the sampling trace execution status.
SM803	Trace trigger	A trigger turns on upon the status change of the relay (off → on).
SM804	After trace trigger	Turns on when any of the following conditions is met. The relay is used to check the status of trigger condition. • A trigger is issued by a programming tool • The TRACE instruction is executed. • SM803 turns on. • Detailed setting (Device and Step No.)
SM805	Trace completed	Turns on when the sampling trace is completed.
SM826	Trace error	Turns on when an error occurs during sampling trace execution.

The following figure shows the operation flow of the special relays for sampling trace execution.



- When the sampling trace is interrupted
  If SM801 (Trace start) is turned off during sampling trace, execution of the sampling trace will be suspended. When the sampling trace is suspended, the trace count is cleared. The sampling trace restarts by turning on SM801.



*1 SM800 also turns off when the sampling trace is suspended by programming tool.

(4) Device/label that can be set with the sampling trace

For the device/label that can be set with the sampling trace, refer to the following.

Operating manual for the programming tool used

(5) Setting method

For how to execute the trace and set the storage location of trace data, refer to the following.

Operating manual for the programming tool used

(6) Execution method

For how to execute the sampling trace, refer to the following.

Operating manual for the programming tool used

(7) Storage method of trace data

The settings and results of trace can be stored in the CSV file format in a personal computer. For how to store the trace data, refer to the following.

Operating manual for the programming tool used

(8) Precautions

(a) Areas where sampling trace can be performed

The sampling trace can be performed from other stations on the network or serial communication module. However, it cannot be performed from multiple devices simultaneously. It can be performed from one device to the CPU module.

(b) Holding and clearing the trace setting

The trace setting (sampling trace file) registered with the CPU module is latched.
Even if the CPU module is powered off and then on or is reset, the sampling trace can be performed again with the trace setting at registration.
However, the previous trace result cannot be read.
Also in the following cases of 1) to 4), even when the trigger condition of the sampling trace is established, the latched trace setting will be cleared since the condition is not recognized as the trigger condition (SM800 (Trace preparation) turns off).
Register the trace setting again through a programming tool.

1) When selecting "Standard RAM" in "Target memory", configuring the setting that changes the local device size in the standard RAM[*1], writing parameters to the CPU module, and then performing any of the following operations

   - The CPU module is powered off and then on.
   - The CPU module is reset.
   - The CPU module is set from STOP to RUN.

*1 The operation includes when a local device is created.

2) When selecting "Standard RAM" in "Target memory" and the sampling trace file is corrupt, either of the following operations were performed.

   - The CPU module is powered off and then on.
   - The CPU module is reset.

3) When selecting "Memory card (RAM)" in "Target memory" while the SRAM card where the sampling trace file has been registered is not mounted, either of the following operations were performed.

   - The CPU module is powered off and then on.
   - The CPU module is reset.

4) When selecting "Memory card (RAM)" in "Target memory" and the sampling trace file is corrupt, either of the following operations were performed.

   - The CPU module is powered off and then on.
   - The CPU module is reset.

(c) Clearing trace execution status

The trace execution status can be cleared by either of the following operations.

- Remote latch clear (☞ Page 137, Section 3.6.4)
- Latch clear by using the special relay and special register areas(☞ Page 75, Section 2.7 (4) (a))

(d) Reading trace result in the STOP status

The trace result cannot be read while the CPU module is in the STOP status. Read the trace result while the CPU module is in the RUN status.

(e) Sampling trace registration while the trigger condition is met

Even if a trigger condition is met, the sampling trace setting can be registered by the following procedure.

1. **Turn on SM829 (Forced registration specification of trace setting).**
2. **Enable the forced execution registration.**

   [Debug] ⇨ [Sampling Trace] ⇨ [Forced Execution Registration Effective]

For the above case, start the trace in the status where the trigger condition is not met. If the trigger condition is met, the trigger may not be normally executed.

(f) When a file register is selected

When a file register is selected as a specified device for a trace setting, do not change the file register file and the block number of file register after trace registration. If doing so, trace data may not be normally collected.

(g) Data acquisition timing setting

When the data acquisition timing setting is set to "Specified Interval" or "Each Multiple CPU High Speed Transfer Cycle", pay attention to the sampling interval and sampling processing time for one sampling because the sampling trace is performed as interrupt processing. If the processing time for one sampling is long, "WDT ERROR" may occur.

(h) Performing sampling trace during execution of another sampling trace

The first sampling trace is performed normally. The second sampling trace cannot be performed.

(i) Executing online change

When sampling trace and online change are performed simultaneously, they operate as follows.

- Performing sampling trace during online change

  - When the trace point or trigger point is specified by the step number:
    The online change is completed normally but the sampling trace is not performed.
  - When the trace point and trigger point are specified by setting other than the step number:
    Both the online change and sampling trace can be performed.

- Performing online change during execution of sampling trace

  - When the trace point or trigger point is specified by the step number:
    The sampling trace is suspended but the online change is normally performed.
  - When the trace point and trigger point are specified by setting other than the step number:
    Both the online change and sampling trace can be performed.

(j) Latch clear by using the special relay and special register areas during execution of sampling trace

The latch clear operation is performed normally. However, the sampling trace will be stopped.

3.15 Debug from Multiple Programming Tools

This function allows simultaneous debugging from multiple programming tools connected to modules (such as a CPU module and serial communication module). This function is useful when debugging multiple files divided according to processes or functions.

(1) Description

The following table lists the combinations of functions that can be executed simultaneously using this function.

○ : Can be simultaneously performed, △ : Partially restricted, × : Cannot be simultaneously performed

Function in execution	Function executed later							
	Monitor *1	Program monitor list	Interrupt program monitor list	Monitor condition setting	Online change	Scan time measurement	Sampling trace	Executional conditioned device test
monitor*1	○	○	○	○	○	○	○	○
Program monitor list	○	×	○	○	○	○	○	○
Interrupt program monitor list	○	○	×	○	○	○	○	○
Monitor condition setup	○	○	○	×	△*2	○	○	○
Online change	○	○	○	×	×	×	×	×
Scan time measurement	○	○	○	○	×	×	○	○
Sampling trace	○	○	○	○	△*3	○	×	○
Executional conditioned device test	○	○	○	○	△*4	○	○	○

*1 This includes a ladder monitor, device batch monitor, entry data monitor, entry ladder monitor, and local device monitor.

*2 Cannot be performed simultaneously when the step number, or the step number and device number, are set in the monitor condition.

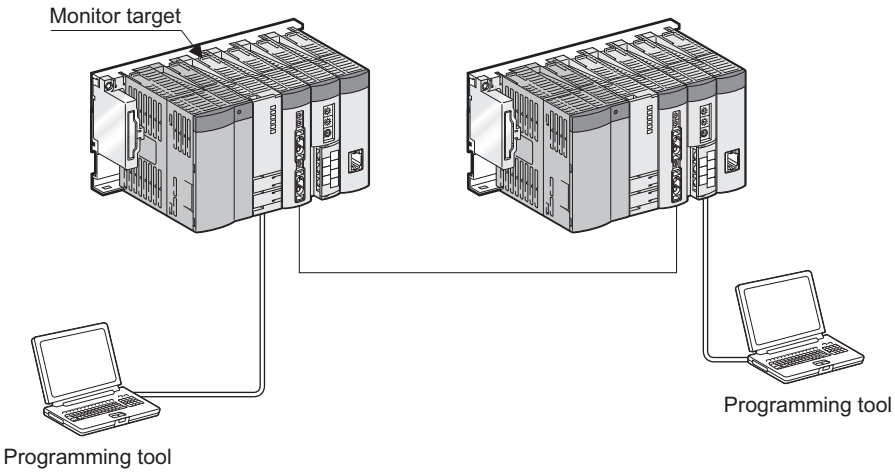
*3 Cannot be performed simultaneously unless the step number is set to the trace point or the trigger point.

*4 Cannot be performed simultaneously in the following cases.

- The data to be changed online includes the registration of an executional conditioned device test.
- When adding a ladder block by online change, registration of an executional conditioned device test is included in the ladder block immediately after the one where the ladder block is to be added.
- The program to be changed online includes registration of an executional conditioned device test.

3.15.1 Simultaneous monitoring from multiple programming tools

This function allows simultaneous monitoring from multiple programming tools connected to modules (such as a CPU module and serial communication module). Creating a user setting system area allows high-speed monitoring from multiple programming tools. (Setting a monitoring file for the host station is not required.)

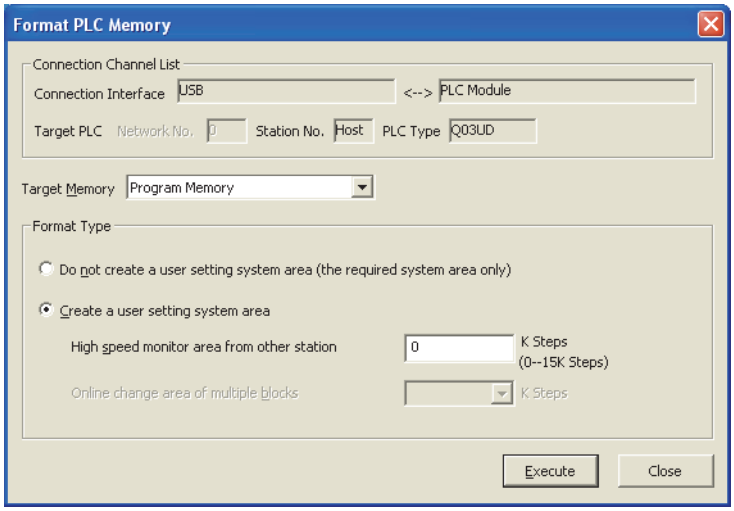


(1) Setting for simultaneous monitoring from multiple programming tools

Create a user setting system area by the following procedure.

1. **Open the "Format PLC Memory" screen.**

[Online] ⇨ [PLC Memory Operation] ⇨ [Format PLC Memory]



2. **Select "Program Memory" for "Target Memory".**

3. **Select "Create a user setting system area" in "Format Type".**

4. **Set the number of steps for the system area (in increments of 1K step) within the following range. Only 1K step is available for each monitoring file from another station.**

Maximum size of settable step	System area for monitoring from another station
Maximum 15K steps	Maximum 15

(2) Precautions

(a) Monitoring condition setting

The monitoring conditions can be set from one programming tool.

(b) Necessity of system area setting

A programming tool connected to another station can simultaneously monitor a CPU module without a user setting system area. However, the monitor speed will be slow. Since the system area is set in the program memory, the area for storing programs in the memory decreases by the size of the system area.

(c) Number of programming tools for which high-speed monitoring can be set

The number of programming tools that can simultaneously monitor a CPU module at high-speed is "the number of user setting system areas (the number of K steps) + 1".

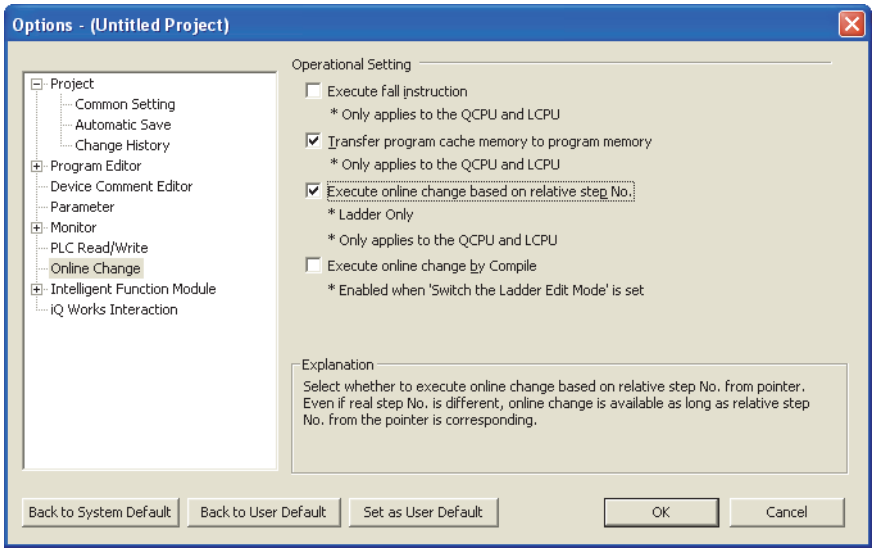
Ex. When a user setting system area is created for 15K steps, maximum 16 programming tools can simultaneously monitor a CPU module at high-speed.

3.15.2 Online change from multiple programming tools

This function allows online change from multiple programming tools.

(1) Operating procedure

Select [Tool] → [Options] → "Online Change" in the programming tool, and check the "Execute online change based on relative step No." checkbox. Set a pointer for online change in advance.



Display the ladder including the specified pointer and write the changed ladder to the CPU module during RUN.

(2) Precautions

Precautions for online change from multiple programming tools are the same as those for standard online change. (☞ Page 173, Section 3.12.3)

3.16 Watchdog Timer (WDT)

This function serves as a CPU module internal timer to detect errors of CPU module hardware and sequence programs.

(1) Setting and resetting

(a) Setting

The watchdog timer setting can be changed in the PLC RAS setting of PLC parameter.
The default is set to 200ms.
The setting range is 10 to 2000ms (in increments of 10ms).

(b) Reset

The CPU module resets the watchdog timer during END processing.

- The watchdog timer does not time up when the CPU module operates normally and the END/FEND instruction is executed within the setting value of watchdog timer.
- The watchdog timer times up when the scan time of the sequence program is extended and the END/FEND instruction could not be executed within the setting value of watchdog timer due to the hardware failure of the CPU module or execution of an interrupt program/fixed scan execution type program.

(2) When the watchdog timer times up

"WDT ERROR" is detected and the following status occurs:

- The CPU module turns off all outputs.
- The RUN LED on the front of the CPU module turns off and the ERR. LED starts flashing.
- SM1 turns on and the error code 5000 or 5001 is stored in SD0.

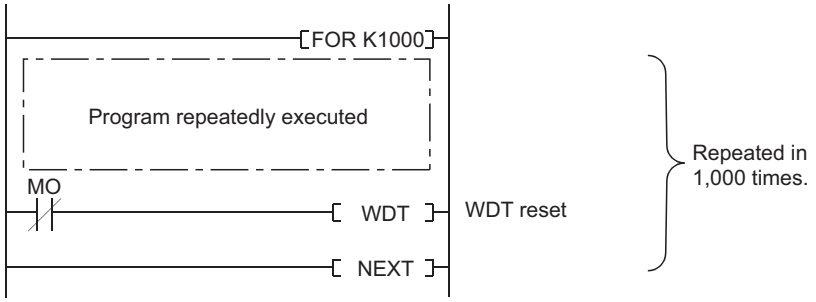
(3) Precautions

(a) Watchdog timer error

An error is observed within the range of 0 to 10ms.
Set a watchdog timer while considering such an error.

(b) Resetting a watchdog timer when a program is repeatedly executed between the FOR and NEXT instructions

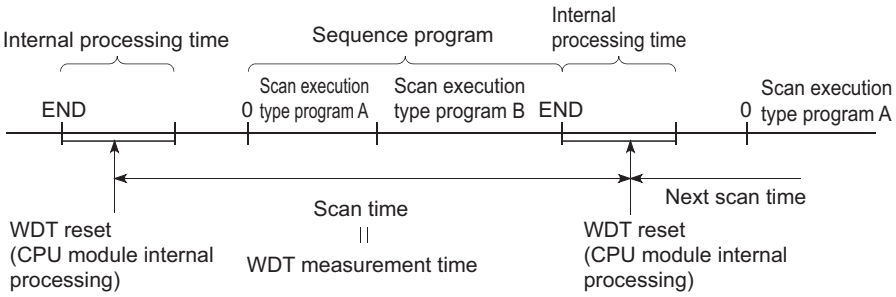
The watchdog timer can be reset by executing the WDT instruction in the sequence program.
To avoid the time up of watchdog timer while a program is repeatedly executed between the FOR and NEXT instructions, reset the watchdog timer by the WDT instruction.



(c) Scan time when using the WDT instruction

The scan time value is not reset even if the watchdog timer is reset in the sequence program.

The scan time is measured up to the END instruction.



Point

- A scan time is time required for the CPU module to operate the sequence program from step 0 and return to the step 0 in the sequence program with the same file name.
  The scan time depends on the execution status of the following:
  - Instructions used in the program
  - Interrupt program and fixed scan execution type program
- To execute the same scan time in every scan, use the constant scan function. (Page 119, Section 3.2)

3.17 Self-diagnostic Function

This function allows the CPU module to diagnose itself to check for errors. This function aims to preventive measures and prevention of malfunction of the CPU module.

(1) Self-diagnostic timing

When an error occurs at power-on or during the RUN or STOP status of the CPU module, the error is detected and displayed by the self-diagnostic function, and the CPU module stops an operation.
Note that errors cannot be detected by the function depending on error status or an instruction executed.
When the operation is not stopped by the function, configure a safety circuit external to the programmable controller so that the entire system operates safely.

(2) Checking errors

(a) LED status

When the CPU module detects an error, the ERR. LED turns on.

(b) Storage location of error information and error check

When the CPU module detects an error, the special relays (SM0, SM1) turn on and the error information (error code) is stored in the special register (SD0).
When several errors are detected, the latest error code is stored in SD0.
Use the special relays and special register in a program as an interlock for the programmable controller and mechanical system.

(3) Checking error history

The latest error code can be checked under "Error History" on the "PLC Diagnostics" screen.

[Diagnostics] ⇨ [PLC Diagnostics]

The error history data is backed up using a battery even after the programmable controller is powered off.

(4) CPU module operation at error detection

(a) Mode at error detection

When an error is detected by the self-diagnostic function, the CPU module enters either of the following modes.

- Mode that stops CPU module operation
  When an error is detected, the CPU module stops an operation and turns off all external outputs of the module set to "Clear" in "Error Time Output Mode" in "Detailed setting" of the I/O Assignment tab of the PLC parameter dialog box (Outputs (Y) in the device memory are held). Note that the external outputs of the module set to "Hold" in "Error time output mode" are held (Outputs (Y) in the device memory are held).
- Mode that continues CPU module operation
  When an error is detected, the CPU module operates programs other than the one (instruction) where an error occurred.

(b) Errors whether to continue or stop an operation can be selected

Whether to continue or stop an operation can be selected in the following errors.

- Errors whether to continue or stop an operation can be selected in the PLC RAS tab of the PLC parameter dialog box

  - Computation error (including SFC program)
  - Expanded command error (setting for future extension)
  - Fuse blown
  - Module verify error
  - Intelligent module program execution error
  - File access error
  - Memory card operation error
  - External power supply OFF (setting for future extension)

  Ex. When "Module verify error" is set to "Continue", an operation is continued from the I/O number before an error. For details of errors, refer to "Self-diagnostics list". (☞ Page 197, Section 3.17 (6))

- Error whether to continue or stop an operation can be selected in "Detailed setting" in the I/O Assignment tab of the PLC parameter dialog box.

  - Intelligent function module error

(5) Error check options

Whether to check the following errors or not can be selected in the PLC RAS tab of the PLC parameter dialog box (All the options are selected (executed) by default).

- Carry Out Battery Check
- Carry Out Fuse Blown Check
- Verify Module
- Check Device Range at Indexing
- Diagnose Redundant Power Supply System Note 3.10

Note 3.10 Universal

Before setting the diagnostic function of the redundant power supply system for the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (☞ Page 466, Appendix 2)

(6) Self-diagnostics list

The following table lists the self-diagnostics performed by the CPU module. The error messages in the "Error message" column can be checked on the screen displayed by selecting [Diagnostics] → [PLC Diagnostics] in the programming tool.

○ : Self-diagnostics is performed. × : Self-diagnostics is not performed.

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Q00UJ CPU	Q00U CPU, Q01U CPU	Q02U CPU	QnUD (E)(H) CPU	QnUDV CPU, QnUDP VCPU
					RUN	ERR.					
Hardware failure	CPU error	MAIN CPU DOWN	• Always	Stop	Off	Flashing	○	○	○	○	○
	END instruction not executed	END NOT EXECUTE	• Execution of the END instruction	Stop	Off	Flashing	○	○	○	○	○
	SFC program execution error	SFCP. END ERROR	• Execution of a SFC program	Stop	Off	Flashing	○	○	○	○	○
	RAM check	RAM ERROR	• Power-on/reset	Stop	Off	Flashing	○	○	○	○	○
	Operation circuit check	OPE.CIRCUIT ERR.	• Power-on/reset • Execution of the END instruction	Stop	Off	Flashing	○	○	○	○	○
	Fuse blown*1 *2	FUSE BREAK OFF	• Always	Stop/ continue	Off/on	Flashing /on	○	○	○	○	○
	I/O interrupt error	I/O INT. ERROR	• Occurrence of an interrupt	Stop	Off	Flashing	○	○	○	○	○
	LAN controller failure	LAN CTRL. DOWN	• Power-on/reset	Stop	Off	Flashing	×	×	×	○*4	○
	Intelligent function module error*1	SP.UNIT DOWN	• Power-on/reset • Execution of the FROM/TO instructions • Execution of the intelligent function module dedicated instruction • Execution of the END instruction	Stop/ continue	Off/on	Flashing /on	○	○	○	○	○
	Control bus error	CONTROL-BUS ERR.	• Power-on • Execution of END processing • Execution of the FROM/TO instructions • Execution of the intelligent function module dedicated instruction • Always	Stop	Off	Flashing	○	○	○	○	○
	Momentary power failure	AC/DC DOWN	• Always	Continue	On	Off	○	○	○	○	○
	Multiple CPU high speed bus error	MULTI-C.BUS ERR.	• Power-on/reset	Stop	Off	Flashing	×	×	×	○	○

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Q00UJ CPU	Q00U CPU, Q01U CPU	Q02U CPU	QnUD (E)(H) CPU	QnUDV CPU, QnUDP VCPU
					RUN	ERR.					
Hardware failure	Voltage drop of power supply for redundant base unit	SINGLE PS. DOWN	• Always	Continue	On	On	×	○	○	○	○
	Redundant power supply module failure	SINGLE PS. ERROR	• Always	Continue	On	On	×	○	○ *5	○ *5	○
	Flash ROM error	FLASH ROM ERROR	• Writing to ROM	Continue	On	On	○	○	○	○	○
Handling error	Module verification *1*2	UNIT VERIFY ERR.	• Execution of the END instruction	Stop/ continue	Off/on	Flashing /on	○	○	○	○	○
	Base assignment error	BASE LAY ERROR	• Power-on/reset	Stop	Off	Flashing	○	○	○	○	○
	Intelligent function module assignment error	SP.UNIT LAY ERR.	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	Intelligent program execution error *1	SP.UNIT ERROR	• Execution of the FROM/TO instructions	Stop/ continue	Off/on	Flashing /on	○	○	○	○	○
	Intelligent function module version error	SP.UNIT VER.ERR	• Power-on/reset	Stop	Off	Flashing	○	○	○	○	○
	No parameter	MISSING PARA.	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	Boot error	BOOT ERROR	• Power-on/reset	Stop	Off	Flashing	○	○	○	○	○
	Backup error	RESTORE ERROR	• Power-on/reset	Stop	Off	Flashing	○	○	○	○	○
	Memory card operation error *1	ICM.OPE. ERROR	• Mounting/ removal of the memory card	Stop/ continue	Off/on	Flashing /on	×	×	○	○	○
	Memory card access error	MEM.ACCESS ERROR	• Always	Continue	On	On	×	×	×	×	○
	File setting error	FILE SET ERROR	• Power-on/reset • Writing to programmable controller	Stop	Off	Flashing	○	○	○	○	○
	File access error *1	FILE OPE. ERROR	• Execution of an instruction	Stop/ continue	Off/on	Flashing /on	○	○	○	○	○
	Instruction execution disabled	CAN'T EXE.PRG.	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Q00UJ CPU	Q00U CPU, Q01U CPU	Q02U CPU	QnUD (E)(H) CPU	QnUDV CPU, QnUDP VCPU
					RUN	ERR.					
Parameter error	Parameter setting check	PARAMETER ERROR	• Power-on/reset • Switching from STOP to RUN • Writing to programmable controller	Stop	Off	Flashing	○	○	○	○	○
	Link parameter error	LINK PARA.ERROR	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	SFC parameter error	SFC PARA. ERROR	• Switching from STOP to RUN • Writing to programmable controller	Stop	Off	Flashing	○	○	○	○	○
	Intelligent function module parameter error	SP.PARA. ERROR	• Power-on/reset	Stop	Off	Flashing	○	○	○	○	○
Password error		REMOTE PASS.ERR	• Power-on/reset • Switching from STOP to RU	Stop	Off	Flashing	○	○	○	○	○
Instruction code check		INSTRUCT. CODE ERR	• Power-on/reset • Switching from STOP to RUN • Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
No END instruction		MISSING END INS.	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
Pointer setting error		CAN'T SET(P)	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
		CAN'T SET(I)	• Power-on/reset • Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Q00UJ CPU	Q00U CPU, Q01U CPU	Q02U CPU	QnUD(E)(H) CPU	QnUDV CPU, QnUDP VCPU
					RUN	ERR.					
Program error	Operation error[*1*3]	OPERATION ERROR	• Execution of an instruction	Stop/ continue	Off/on	Flashing /on	○	○	○	○	○
	FOR to NEXT instructions structure error	FOR NEXT ERROR	• Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
	CALL to RET instructions structure error	CAN'T EXECUTE(P)	• Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
	Interrupt program error	CAN'T EXECUTE(I)	• Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
	Instruction execution disabled	INST. FORMAT ERR.	• Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
	Dedicated instruction of multiple CPU high speed bus error	MULTI COM.ERROR	• Execution of an instruction	Stop	Off	Flashing	×	×	×	○	○
	SFC block configuration error	CAN' SET(BL)	• Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	SFC step configuration error	CAN'SET(S)	• Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	SFC execution error	SFC EXE. ERROR	• Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	SFC syntax error	SFCP. FORMAT ERR.	• Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○	○
	SFC block execution error	BLOCK EXE.ERROR	• Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
	SFC step execution error	STEP EXE.ERROR	• Execution of an instruction	Stop	Off	Flashing	○	○	○	○	○
CPU error	Watchdog error supervision	WDT ERROR	• Always	Stop	Off	Flashing	○	○	○	○	○
	Program time-out	PRG.TIME OVER	• Always	Continue	On	On	○	○	○	○	○
Multiple CPU systems error	Another CPU major error	MULTI CPU DOWN	• Always • Power-on/reset	Stop	Off	Flashing	×	○	○	○	○
	Multiple CPU systems execution error	MULTI EXE.ERROR	• Power-on/reset	Stop	Off	Flashing	×	○	○	○	○
	Multiple CPU systems consistency error	CPU LAY. ERROR	• Power-on/reset	Stop	Off	Flashing	×	○	○	○	○
	Another CPU minor error	MULTI CPU ERROR	• Always	Continue	On	On	×	○	○	○	○
File diagnostic check		INCORRECT FILE	• Power-on/reset • Switching from STOP to RUN • Writing to programmable controller	Stop	Off	Off	○	○	○	○	○
Annunciator check		F****	• Execution of an instruction	Continue	On	USER LED turns on.	○	○	○	○	○

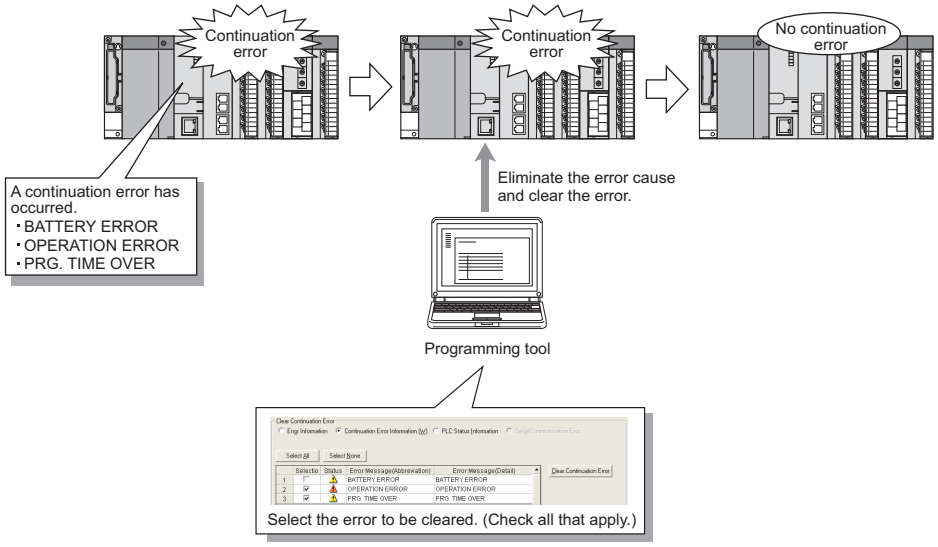
*1 The operating status can be set to "Continue" in parameter. (Default: "Stop")
*2 The check status can be selected in parameter. (Default: Checkbox selected)
*3 The error includes an operation error when a device range is checked at index modification.
*4 Only the Built-in Ethernet port QCPU supports this self-diagnostic item.

3.17.1 LEDs indicating errors

When an error occurs, the LEDs on the front of the CPU module turns on/flashes. (☞ Page 222, Section 3.20)

3.17.2 Clearing errors

Continuation errors can be cleared. The High-speed Universal model QCPU and Universal model Process CPU can clear those errors by types.



(1) Errors that can be cleared

The following errors can be cleared.

- SP.UNIT DOWN
- AC/DC DOWN
- BATTERY ERROR
- FLASH ROM ERROR
- MEM.ACCESS ERROR
- SP.UNIT ERROR
- ICM.OPE.ERROR
- FILE OPE.ERROR
- REMOTE PASS.FAIL
- SNTP OPE.ERROR
- OPERATION ERROR
- PRG.TIME OVER
- F*** (Annunciator)
- FUSE BREAK OFF
- SINGLE PS.DOWN
- SINGLE PS.ERROR
- UNIT VERIFY ERR.
- MULTI CPU ERROR
- PID ERROR

(2) Clearing method

Errors are cleared in two ways.

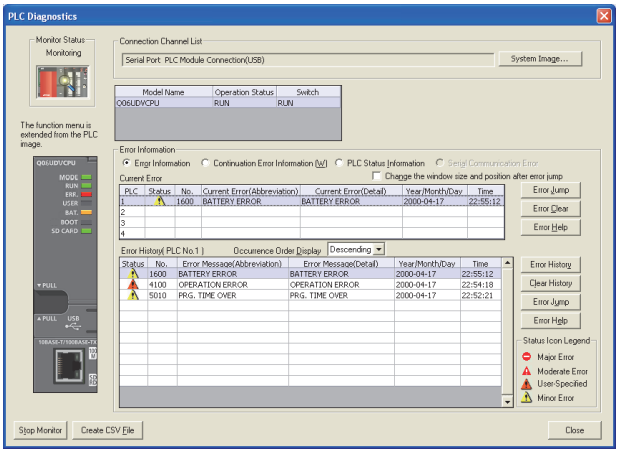
- Using a programming tool Note 3.11
- Using the special relay (SM) and special register (SD)

Note 3.11 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU can clear errors using a programming tool.

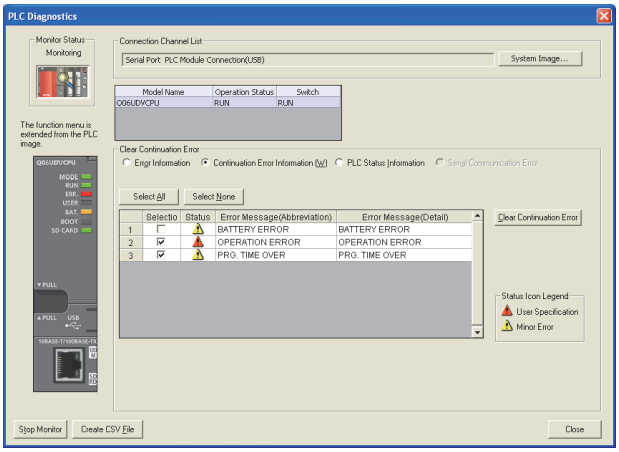
(a) Clearing errors using a programming tool (High-speed Universal model QCPU and Universal model Process CPU only)

Perform the following procedure.

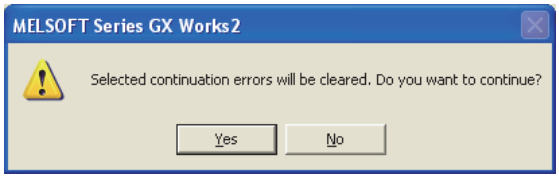


↓

↓



↓



1. Check the continuation errors detected on the PLC Diagnostics window.

2. Eliminate the error causes of the detected errors.

3. Select the "Continuation Error Information" radio button and check the checkboxes of errors to be cleared. Then, click the Clear Continuation Error button.

4. Click the Yes button to clear the errors.

5. Check that the specified errors are no longer displayed on the PLC Diagnostics window.*1

*1 The specified errors are not deleted from error history data.

(b) Clearing errors using the special relay (SM) and special register (SD)

Perform the following procedure.

- **CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU**

  1. **Eliminate the error cause.**
  2. **Store the error code corresponding to the error to be cleared in SD50.**
  3. **Turn off and then on SM50.**
  4. **The error is cleared.**

Point

When the latest error (the error stored in SD0) is cleared, error information (stored in SM0, SM1, SM5, SM16, SD0 to SD26) are cleared. If more than one error has been detected, information on other errors are also cleared and no longer obtained. To clear other errors, obtain the past error data from the error history.

- **High-speed Universal model QCPU and Universal model Process CPU**

  1. **Check the continuation error(s) detected in SD81 and SD82. (For the bit pattern, refer to the QCPU User's Manual (Hardware Design, Maintenance and Inspection).)**
  2. **Eliminate the error cause(s).**
  3. **Specify the error(s) to be cleared in SD84 and SD85.**
  4. **Turn off and then on SM84.**
  5. **Check the bit(s) corresponding to the cleared error(s) in SD81 and SD82 is off.**

Point

The High-speed Universal model QCPU and Universal model Process CPU can also clear errors by storing an error code in SD50 and turning off and on SM50. In this case, however, error types cannot be specified.

(3) Status after error clear

When the CPU module is recovered from errors, the related special relay, special register, and LEDs return to the status before the errors occurred. If the same error occurs after clearing an error, it is registered in the error history again.

(4) Precautions

- Since errors with the same message are batch-cleared regardless of their error codes, error codes not intended may also be cleared.
- To clear more than one annunciator, perform the same number of error clear operations as that of annunciators that are on.

Point

- When the clear-target error code is stored in the special register, the units digit of the code is ignored.

  Ex. When the error codes 2410, 2411, and 2412 occur and 2412 is stored in SD50 to clear, other two error codes 2410 and 2411 are also cleared.

- Only errors occurred in a CPU module can be cleared.

  Ex. Since "SP. UNIT DOWN" is an error occurred in the Q bus, the error cause will not be eliminated with the error clear methods explained in this section. To eliminate the error cause, refer to the following.

  QCPU User's Manual (Hardware Design, Maintenance and Inspection)

3.18 Error History

This function stores an error detected by the self-diagnostic function and the detection time as error history data in a memory. The error history data can be checked on the screen displayed by selecting [Diagnostics] → [PLC Diagnostics] in the programming tool.

Point

The detection time is based on the clock in the CPU module. Make sure to set the correct time before the first use of the CPU module. (Page 127, Section 3.5)

(1) Storage area

All stored logs are saved to the storage memory for error history of the CPU module.

Storage area	Number of storable logs
System memory in CPU module*1	Up to 100*2

*1 The memory is managed inside the system.

*2 When the number of storable logs are exceeded, the latest error log is stored by deletion of the oldest error log.

(2) How to clear error history

To clear the error history data stored in the memory and error history file, select [Diagnostics] → [PLC Diagnostics] in the programming tool and click the Clear History button. When the button is clicked, all the error history data stored in the storage memory of the CPU module and the error history file in a memory card are cleared.

3.19 Security Function

This function protects data in the CPU module against tampering and theft by unauthorized persons. There are four security functions available. Use those functions according to your applications and needs.

Function	Purpose	Reference
Password registration*1	To limit access to each file in the CPU module	Page 207, Section 3.19.1
File password 32*2		Page 209, Section 3.19.2
File access control by security key*2	To limit devices that can access to files in the CPU module	Page 214, Section 3.19.3
Remote password	To limit access to the CPU module from external devices	Page 219, Section 3.19.4
Block password	To limit access to each POU	GX Works2 Version 1 Operating Manual (Common)

*1 The High-speed Universal model QCPU and Universal model Process CPU do not support this function.
*2 Only the High-speed Universal model QCPU and Universal model Process CPU support these functions.

3.19.1 Password registration Note 3.12

This function disables reading and writing data, such as programs and device comments, in the CPU module using a programming tool.

(1) Password target files

A password can be set to the following files.

- Program
- Device comment
- Initial device value

(2) Operations that are controlled and the number of characters

A password can be set to the following operations. The number of characters in the password should be four (one-byte).

- Reading files
- Writing files

Remark

For characters allowed in passwords, refer to "Explanation of Registration Conditions" on the Input Password window.

Note 3.12 Universal

The High-speed Universal model QCPU and Universal model Process CPU do not support this function.

(3) Online operations that require authentication

Authentication is required to execute the following operations to password-protected files. For the authentication method, refer to Page 211, Section 3.19.2.

- Write to PLC (data writing)
- Read from PLC (data reading)
- Online change (data writing)
- Change TC setting value (data writing)
- Verify with PLC (data reading)
- Create/Change or Delete of a password (data reading and writing)
- Write to PLC (flash ROM)
- Delete PLC data (data writing)

(4) Operating procedure

For the password registration procedure, refer to the following.

Operating manual for the programming tool used

Remark

To change, delete, or unlock the password, refer to the following.

Operating manual for the programming tool used

(5) Precautions

(a) Password management

A password registered with a file cannot be read from the file. Forgetting the registered password disables the following operations.

- Program memory or memory card: Format PLC memory
- Standard ROM: Batch write

Record the registered password on paper and securely store the paper.

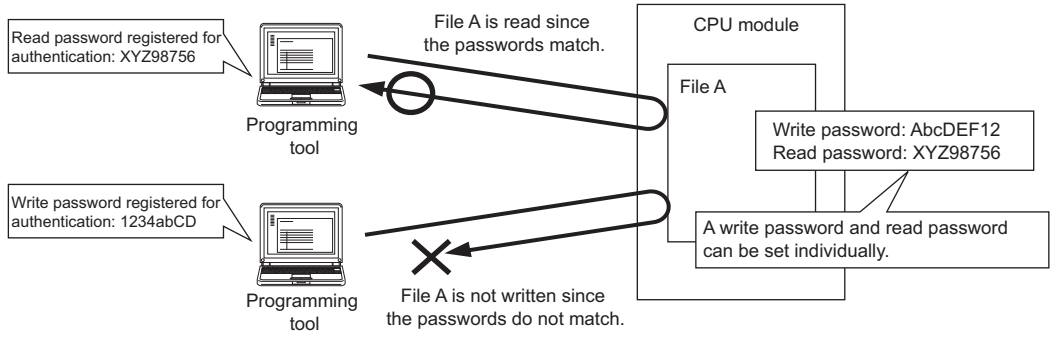
(b) Operations that overwrite files

The following operations overwrite files in the target drives (program memory and standard ROM) regardless of the password registration setting.

- Boot operation from a memory card
- CPU module change function with memory card (Backup data restoration)

3.19.2 File password 32 Note 3.13

This function sets a read password and write password for each file stored in the CPU module so that files are protected against tampering and theft by unauthorized persons.



(1) File protection timing

File protection is enabled immediately after the passwords are registered, and it is disabled immediately after the passwords are deleted.

(2) Password target files

A password can be set to the following files.

- Program
- Device comment
- Initial device value
- Parameter
- Symbolic information

(3) Operations that are controlled and the number of characters

A password can be set to the following operations. The minimum number of characters in the password should be 4 , and the maximum number should be 32.

- Reading files
- Writing files
- Reading/writing files

Remark

For characters allowed in passwords, refer to "Explanation of Registration Conditions" on the Input Password window.

Note 3.13 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU support this function.

(4) Online operations that require authentication

Authentication is required to execute the following operations to password-protected files. (☞ Page 211, Section 3.19.2)

- Write to PLC (data writing)
- Read from PLC (data reading)
- Online change (data writing)
- Change TC setting value (data writing)
- Verify with PLC (data reading)
- Create/Change or Delete of a password (data reading and writing)
- Delete PLC data (data writing)

(5) Operating procedure

For the password setting procedure, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

Remark

To change, delete, or unlock the password, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

(6) Precautions

(a) Boot from an SD memory card

The following table shows the relationship between the boot operation availability and file password 32 setting.

-: No combination available

Transfer source file		Transfer destination file		Password status	Boot operation
File	Password	File	Password		
Exist	Set	Exist	Set	Matched	Enabled
				Not matched	Disabled
			Not set	-	Disabled
		Not exist	-	-	Enabled
	Not set	Exist	Set	-	Disabled
			Not set	-	Enabled
		Not exist	-	-	Enabled
Not exist	-	-	-	-	-

If boot file settings are configured to more than one file, the files can be transferred only when all the passwords match. If all the passwords do not match, data in the SD memory card are not transferred and "BOOT ERROR" (error code: 2213) occurs.

(b) When "Clear Program Memory" is selected in parameter (Boot File tab)

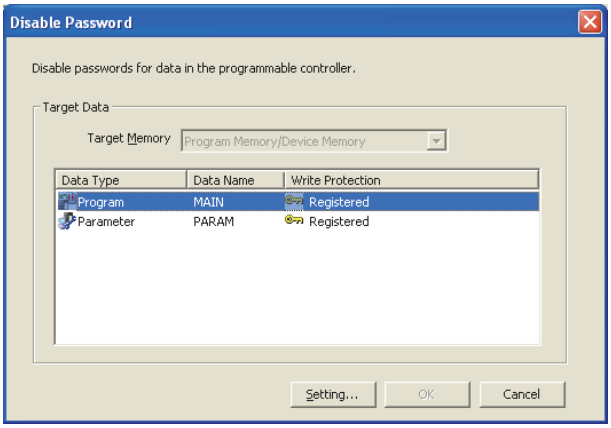
Even though a password is registered, files will be formatted.

(7) Authentication method

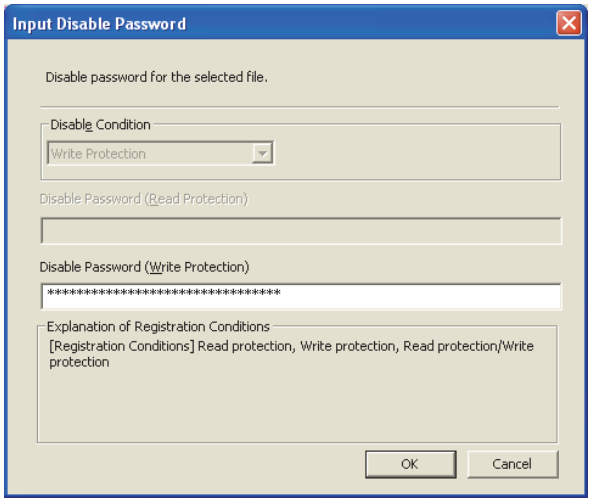
Passwords are authenticated in three ways.

- By a programming tool
- By the FTP server
- By the MC protocol

(a) Authentication by a programming tool



↓



1. Whenever an online operation requiring password authentication is executed, the "Disable Password" window appears.
Select an authentication target file, and click the "Setting" button.

2. Enter a password in the "Input Disable Password" window.

Point

The entered password is valid until the project is closed.

(b) Authentication by the FTP server

To access a password-protected file from external devices using the FTP server function, password authentication is required for each file. Authentication is required whenever files are accessed.

○: Authentication required, -: Authentication not required

Operation	FTP command	Password authentication	
		Data write	Date read
Deleting a file in a CPU module	delete	○	-
Reading a file from a CPU module	get	-	○
Deleting a file in a CPU module	mdelete	○	-
Reading a file from a CPU module	mget	-	○
Changing the name of a file in a CPU module	rename	○	-
Changing or displaying the attribute of a file in a CPU module	change	○	-

To authenticate a password, use FTP commands for password authentication. Once the password is authenticated, it is valid until the accessed external device logs out from the FTP server or the network line is disconnected. (A password does not need to be authenticated every time the authentication target FTP commands are executed.)

FTP command for password authentication	Operation
quote passwd-rd<password>	Setting, displaying, and clearing a read password (file password 32) in a CPU module
quote passwd-wr<password>	Setting, displaying, and clearing a write password (file password 32) in a CPU module
quote keyword-set<password>	Setting, displaying, and clearing a file access password in a CPU module

The executability of FTP commands for password authentication differs depending on the access path to the CPU module.

○: Executable, ×: Not executable

FTP command for password authentication	Access path	
	Via Ethernet module supporting the file password 32 function	Via Ethernet module not supporting the file password 32 function
quote passwd-rd<password>	○	×
quote passwd-wr<password>	○	×
quote keyword-set<password>	×	○

Point

- If the Ethernet module that does not support the file password 32 function is used, observe the following points.
  - Set the same password for a read password and writ password.
  - The number of characters in the password must be four.
- For details of FTP commands, refer to the following.
  QnUCPU User's Manual (Communication via Built-in Ethernet Port)

(c) Authentication by the MC protocol

To access a password-protected file from external devices using the MC protocol, the request message format of the MC protocol needs to be changed and a command for the file password 32 must be specified.

1. **Add "Keyword" at the end of a request message, and set a password in the added area.**
2. **Authenticate the password using the one set in the added area.**
3. **For the commands requiring password authentication, specify 0004 (for file password 32) in the "Subcommand" area of a request message.**

Function	Command (subcommand)
Deleting a file	1822 (0004)
Copying a file	1824 (0004)
Changing the attribute of a file	1825 (0004)
Opening a file	1827 (0004)

The executability of commands for password authentication differs depending on the access path to the CPU module.

○: Executable, ×: Not executable (Command is not supported.)

Function	Command	Access path	
		Via serial communication module supporting the file password 32 function	Via serial communication module not supporting the file password 32 function
Deleting a file	1822 (0000)	○*1	○*1
	1822 (0004)	×	○
Copying a file	1824 (0000)	○*1	○*1
	1824 (0004)	×	○
Changing the attribute of a file	1825 (0000)	○*1	○*1
	1825 (0004)	×	○
Opening a file	1827 (0000)	○*1	○*1
	1827 (0004)	×	○

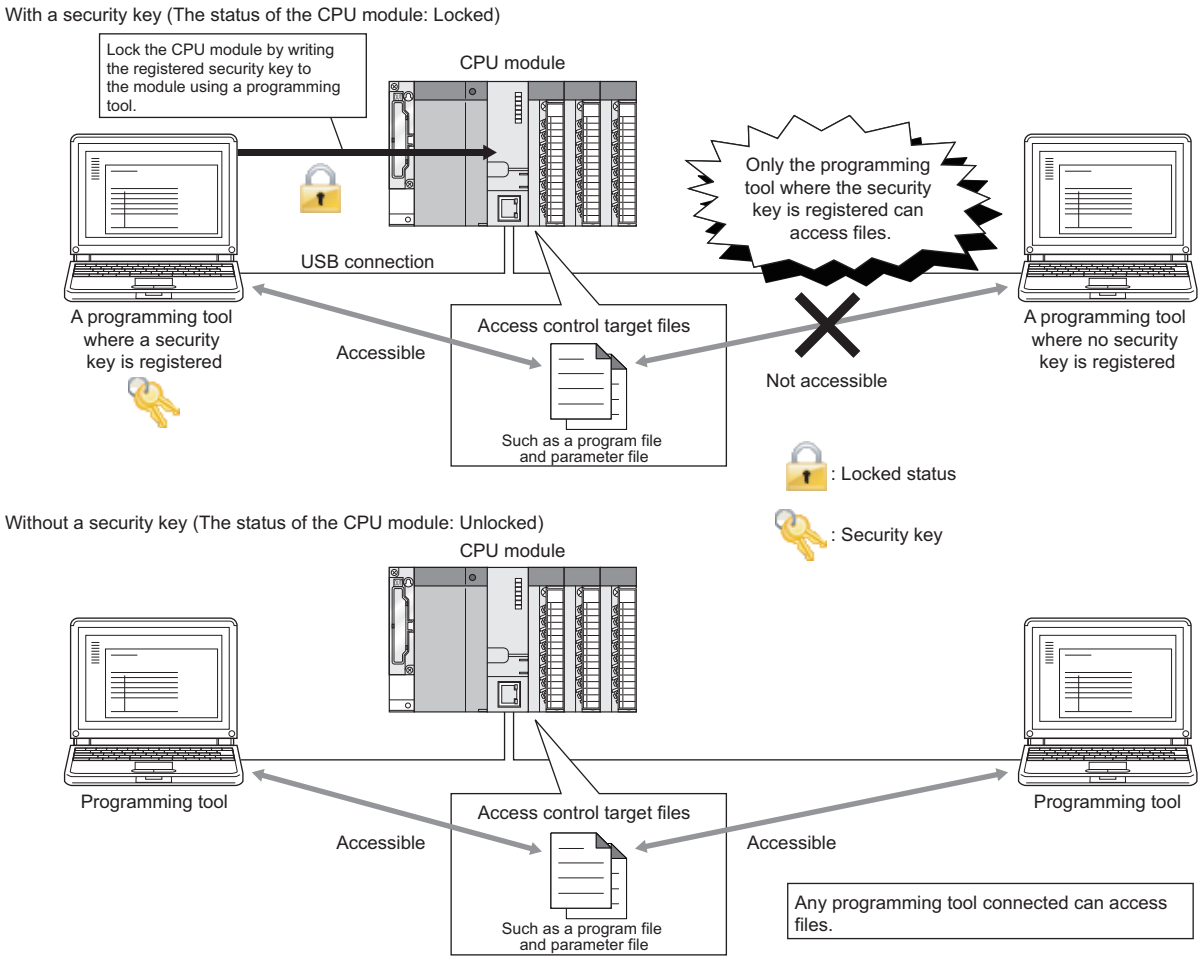
*1 The commands are executable only when no password is registered with the access-target file or the number of characters in the registered password is four.

Point

- If the serial communication module that does not support the file password 32 function is used, observe the following points.
  - Set the same password for a read password and writ password.
  - The number of characters in the password must be four.
- For details of commands, refer to the following.
  MELSEC Communication Protocol Reference Manual

3.19.3 File access control by security key Note 3.14

This function protects unauthorized access to the files in the CPU module by writing a security key*1 to the module. The CPU module is locked with a security key and the files in the module can only be accessed from a programming tool where the same security key is registered.



*1 Security key is a security code used to control file access between a programming tool and CPU module. A security key includes the following information.

- Name: A name of the security key. The number of characters used is 1 to 128.
- Date and time: Date and time when a security key is generated. The display format is [yyyy/mm/dd hh:mm].

Remark

If the security key used to lock a project or CPU module cannot be imported from a personal computer to a programming tool, the security key cannot be unlocked and the project data cannot be accessed permanently. When using a security key, note that we take no responsibility for any loss caused to the user, individual, or company resulting from lost data.

Note 3.14 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU support this function.

(1) Access control target files

With a security key, access to the following files is controlled.

- Program
- Device comment
- Parameter
- Symbolic information

(2) Access control target drives

With a security key, access to the following drives is controlled.

○: Available, ×: Not available

Drive	Read/write availability
Program memory (drive 0)	○
SD memory card (drive 2)	×
Standard RAM (including an extended SRAM cassette) (drive 3)	○
Standard ROM (drive 4)	○

Point

- Do not execute the Write to PLC, Read from PLC, or Verify with PLC function to the access control target files in the SD memory card. If executed, an error is displayed on the programming tool.
- If the CPU module is locked, device comments cannot be written to the SD memory card. Do not specify the SD memory card as a device comment storage location in the QCDSET instruction or in parameter (PLC file setting).

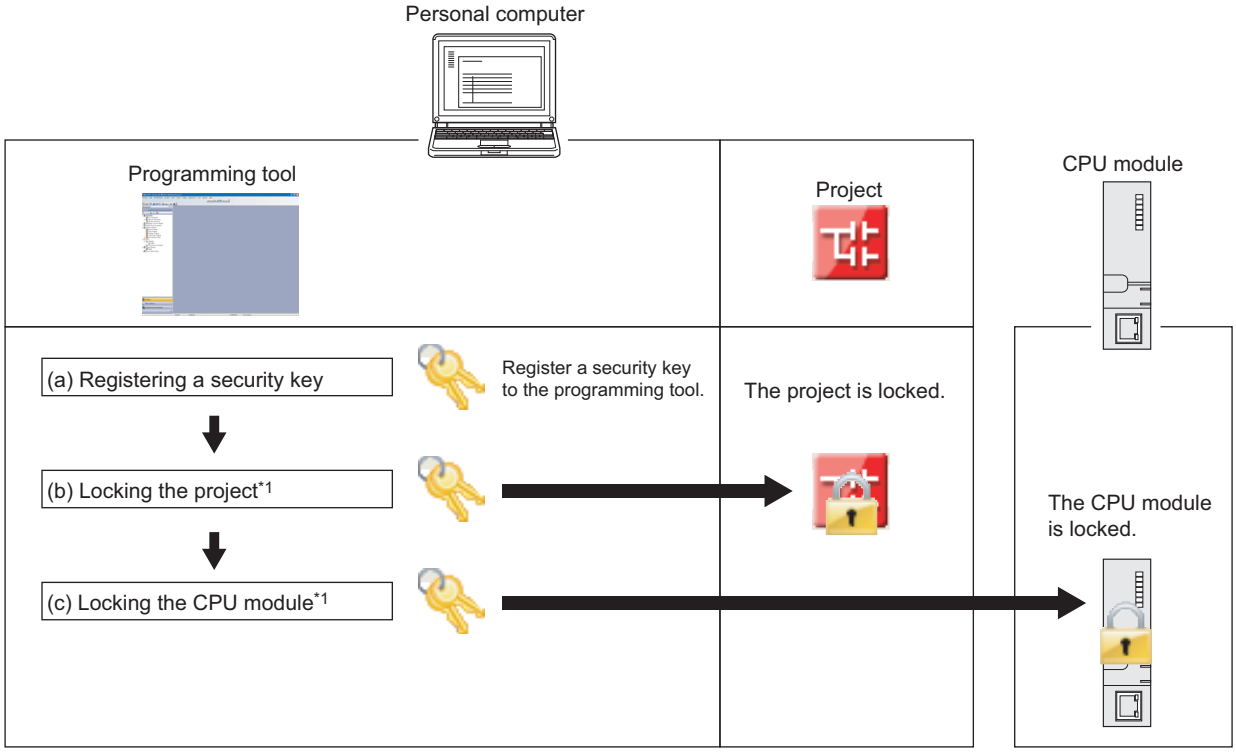
(3) Online operations that require authentication

Authentication is required to execute the following operations to access control target files. (☞ Page 217, Section 3.19.3 (5))

- Write to PLC
- Read from PLC
- Verify with PLC
- Delete PLC data
- Create/Change or Delete a password
- Online change
- Change TC setting

(4) Procedure

To control file access, register a security key in the programming tool. Then, using the registered security key, lock the CPU module.



*1 The steps (b) and (c) do not need to be proceeded in particular order.

Remark

For setting details, refer to the following.

Operating manual for the programming tool used

(a) Registering a security key

Register a security key in the programming tool.

(b) Locking the project

Lock the project using the registered security key.

(c) Locking the CPU module

Lock the CPU module using the registered security key. The CPU module can be locked while it is in the STOP or PAUSE status. The locked status is kept even during power failure and cannot be unlocked even when the PLC memory format or PLC memory clear function is executed.

Point

To lock the CPU module, use a USB device is recommended to avoid communications being intercepted.

(d) Checking the security key information

The information (name, date, and time) of the security key can be checked using a programming tool.

(e) Unlocking the CPU module

To unlock the CPU module, use the security key set with the project. Even if the security keys set with the CPU module and the project do not match, the CPU module can be unlocked. In this case, the system formats the drives (access control target drives) in the CPU module. The CPU module can be unlocked while it is in the STOP or PAUSE status.

(5) Authentication method

To read/write files from/to the locked CPU module, a security key needs to be authenticated between the CPU module and the programming tool.

(a) Writing files

To write files to the locked CPU module, the security keys set with the CPU module and the project need to match.

(b) Reading files

- To read files from the locked CPU module and create a new project, the same security key used to lock the CPU module needs to be registered in the programming tool.
- To read files from the locked CPU module to an existing project, the security keys set with the CPU module and the project need to match.

(6) Precautions

(a) Functions with restrictions

Some restrictions apply to the following functions when the CPU module is locked.

Function		Restrictions	Reference
CPU module change function with memory card		The function cannot be used.	Page 260, Section 3.31
Boot operation		The function cannot be used.	Page 104, Section 2.11
Parameter-valid drive		Parameters stored in a memory card (SD) are not valid.	Page 42, Section 2.1.2
Data logging function		Device comments are not output to the data logging file.	QnUDVCPU/LCPU User's Manual (Data Logging Function)
Accessing files from an external device other than a programming tool	File transfer function (FTP)	Access control target files are not accessed.	QnUCPU User's Manual (Communication via Built-in Ethernet Port)
	MC protocol		-
	GOT, EZSocket		-

(b) Power-off or reset of the CPU module

Do not power off or reset the CPU module during the lock or unlock processing.

(c) Operations from multiple programming tools

If the lock or unlock processing is performed simultaneously from multiple programming tools, the first operation is executed, and the second and later operations are ignored.

(d) Online operations from a different programming tool

Even if an online operation is performed from a different programming tool where the security key is registered during the lock or unlock processing, the operation is not executed.

(e) During the execution of the CPU module change function with memory card

The CPU module cannot be locked.

(f) Number of connectable programming tools

The number of programming tools that can read/write files from/to the CPU module simultaneously while the module is locked is 32.

(g) Files that are not targeted for access control

If an operation is performed from a programming tool to the initial device value and device memory files, which are not targeted for access control, while the CPU module is locked, the security key needs to be authenticated.

(h) Writing a drive heading to the program memory

A drive heading is not written to the program memory if the security key in the programming tool and the CPU module does not match.

3.19.4 Remote password

This function prevents unauthorized remote access to the CPU module.
If a remote password has been set and the CPU module is remotely accessed, entering a remote password is required.

(1) Settable modules and the number of settable modules

The following table lists the modules for which the remote password can be set and the number of settable modules.

Settable module	Number of settable modules
Built-in Ethernet port QCPU	1
Ethernet module	4
Serial communication module	8

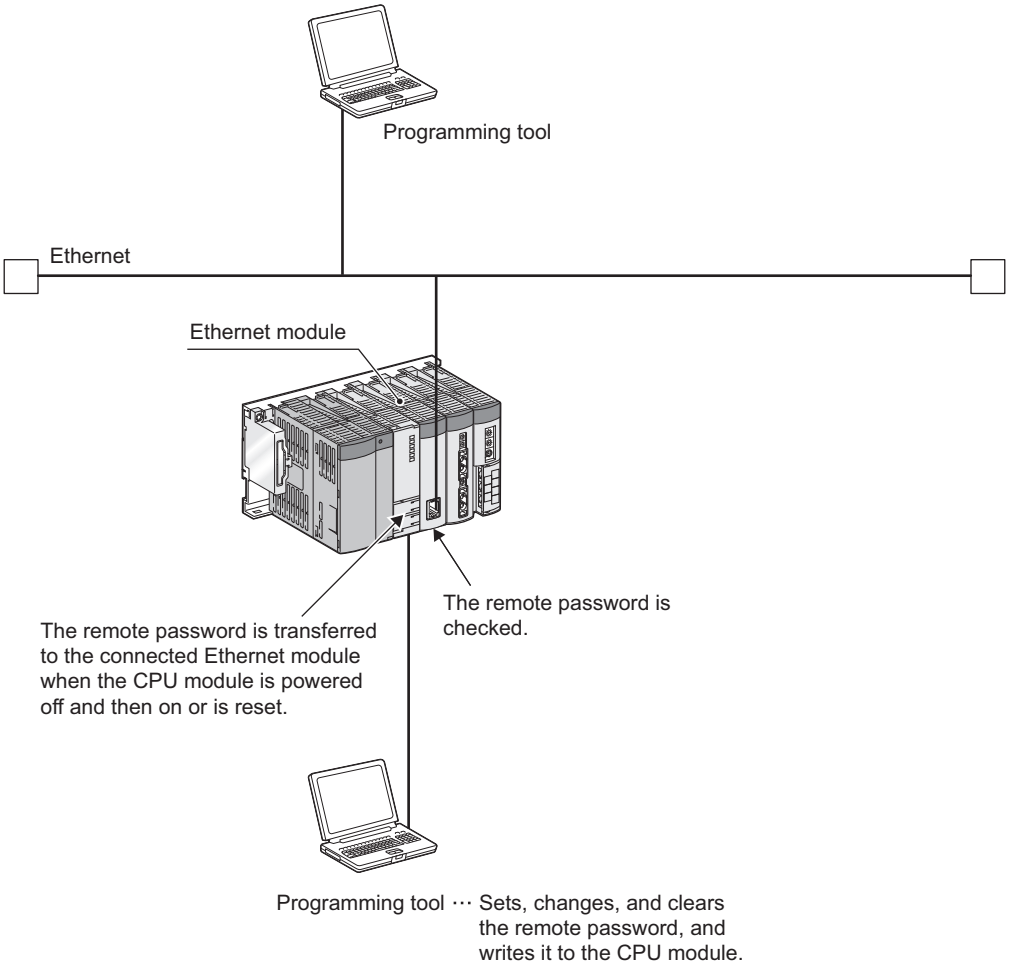
Point

- The number of settable modules in the above table indicates the number of modules for which the remote password can be set, not the number of mountable modules in the system including a CPU module. For the number of mountable modules in the system, refer to the following.
  QCPU User's Manual (Hardware Design, Maintenance and Inspection)
- For details of the remote password set for intelligent function modules, refer to the manuals for each module used.

(2) Function overview

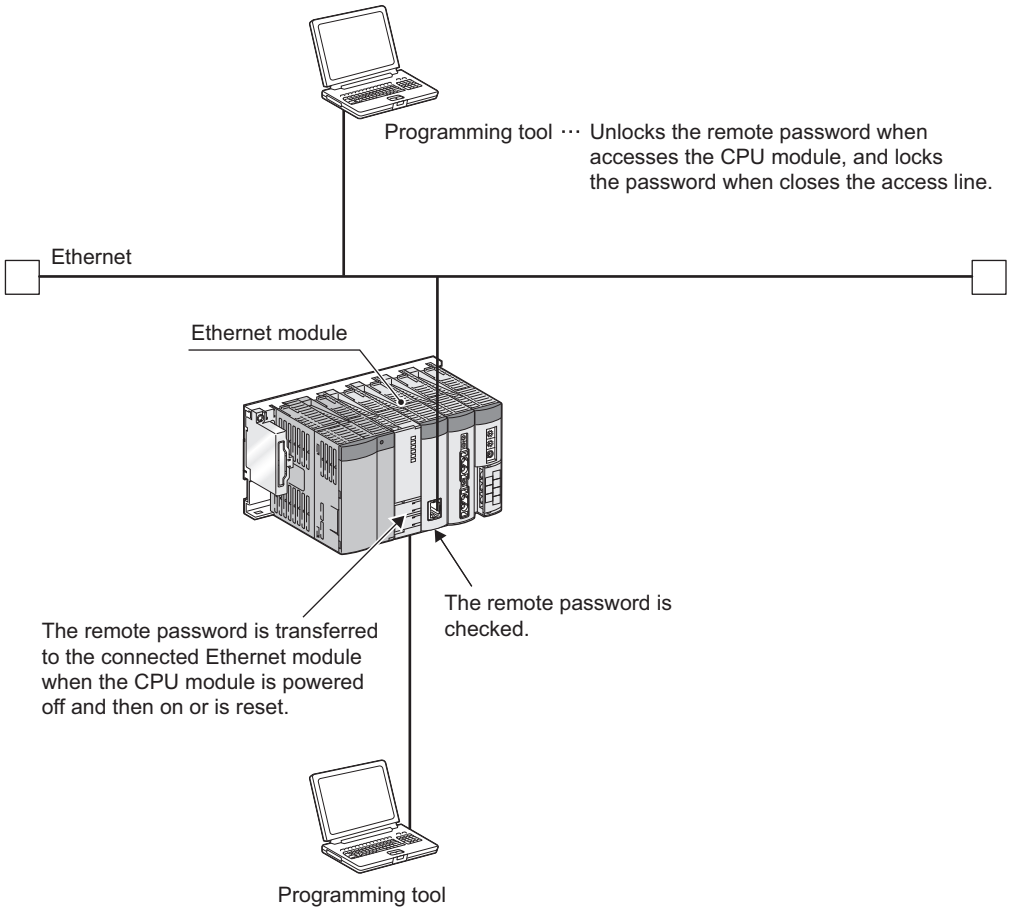
Set a remote password in parameter (☞ Page 464, Appendix 1.4), and write it to the CPU module.

The remote password is transferred to the target module (☞ Page 219, Section 3.19.4 (1)) when the CPU module is powered off and then on or is reset.



(3) Locking/unlocking the remote password

Unlock the remote password of a serial communication module or the password of an Ethernet module over Ethernet. When the entered password matches the registered password, the module is allowed to access the CPU module.



(4) Setting/changing/clearing a remote password

(a) Setting a remote password

Set a password in the Remote Password Setting window. (☞ Page 464, Appendix 1.4)

Project window ⇨ [Parameter] ⇨ [Remote Password]

Write the remote password setting to the CPU module. In a multiple CPU system, write the setting to the control CPU of the target module.

(b) Changing a remote password

Change the password in the Remote Password Setting window, and write the new remote password setting to the CPU module.

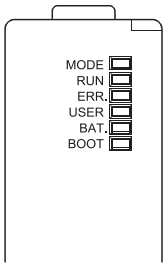
(c) Clearing a remote password

Click the Clear button in the Remote Password Setting window, and write the remote password setting to the CPU module.

3.20 LED Indication

Operating status of the CPU module can be checked by the LEDs on the front of the CPU module.
For details of LED indications, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)



3.20.1 Methods for turning off the LEDs

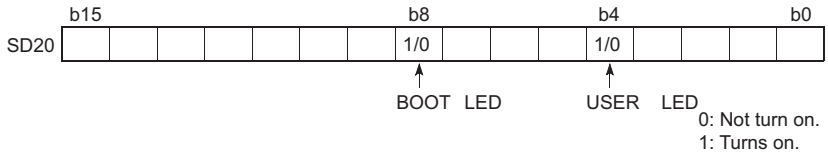
The LEDs can be turned off by the following operations (except for reset operation).

○ : Enabled, × : Disabled

Method for turning off the LED	Relevant LED			
	ERR.	USER	BAT.	BOOT
Execute the LEDR instruction after resolving the error.	○	○	○	×
After resolving the error, clear the error by the special relay SM50 and special register SD50[*1] (operation continuation error only).	○	○	○	×
Turn off the LED by the special relay SM202 and special register SD202.[*1]	×	○	×	○

*1 Description of special relays and special registers

- SM50 : Clears an error of the error code stored in SD50 when the CPU module is powered off and then on.
- SD50 : Stores a code of a error to be cleared.
  For details of error codes, refer to the following.
  QCPU User's Manual (Hardware Design, Maintenance and Inspection)
- SM202 : Turns off the LED corresponding to each bit of SD202 when the CPU module is powered off and then on.
- SD202 : Set an LED to be turned off.



Configure setting to turn off each LED as follows:

- Turning off both the BOOT LED and USER LED: SD202 = 110_H
- Turning off only the BOOT LED: SD202 = 100_H
- Turning off only the USER LED: SD202 = 10_H

There is a priority in indications of the ERR.LED, USER LED, and BAT.LED.
When a cause number of an LED is deleted in the priority, the LED will not turn on even if an error with the cause number occurs. (☞ Page 223, Section 3.20.2)

3.20.2 LED indication priority

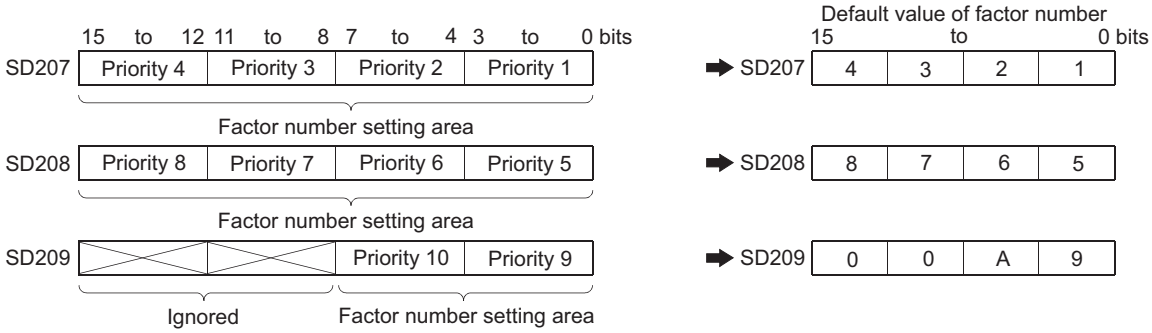
This section describes a priority for error messages stored in the LED display data (SD220 to SD227) in case of an error.

(1) Displayed error messages and their priorities

In case of multiple errors, the error messages are displayed with the following conditions.

- A stop error is always set to the LED display data (SD220 to SD227).
- An operation continuation error is displayed according to the priority cause number described in this section. Whether to indicate an error according to its priority using LED can be selected. (Set the priority using special registers, SD207 to SD209.)
- When errors having the same priority occur simultaneously, the error detected first is displayed.

The priority is determined with the special registers SD207 to SD209 as follows.



(2) Priorities and cause numbers

The following table lists the description and priority of the cause numbers set to the special registers SD207 to SD209.

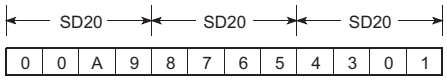
Priority	Cause number (hexadecimal)	Displayed error message	Remarks
1	1	• AC/DC DOWN	• Power-off
2	2	• UNIT VERIFY ERR. • FUSE BREAK OFF • SP.UNIT ERROR • SP.UNIT DOWN	• I/O module verification error • Fuse blown • Intelligent function module verification error
3	3	• OPERATION ERROR • SFCP OPE.ERROR • SFCP EXE.ERROR	• Operation error • SFC instruction operation error • SFC program execution error
4	4	• ICM.OPE.ERROR*1 • FILE OPE.ERROR	• Memory card operation error • File access error
		• FLASH ROM ERROR	• Flash ROM access count over error
5	5	• PRG.TIME OVER	• Constant scan setting time-out error
		• MULTI CPU ERROR*2	• Another CPU error in multiple CPU systems
6	6	• PID ERROR	• PID control instruction error
7	7	• Annunciator	-
8	8	-	-
9	9	• BATTERY ERROR	-
10	A	-	-

*1 The Q00UJCPU, Q00UCPU, and Q01UCPU cannot display the error message.
*2 The Q00UJCPU cannot display the error message.

Point

- To remain the LED off even in case of an error, set the cause number setting area (each 4 bits) of SD207 to SD209 that stores the corresponding cause number to "0".

Ex. To remain the ERR. LED off even when a fuse blown error is detected, set the cause number setting area where the cause number "2" is stored to "0".

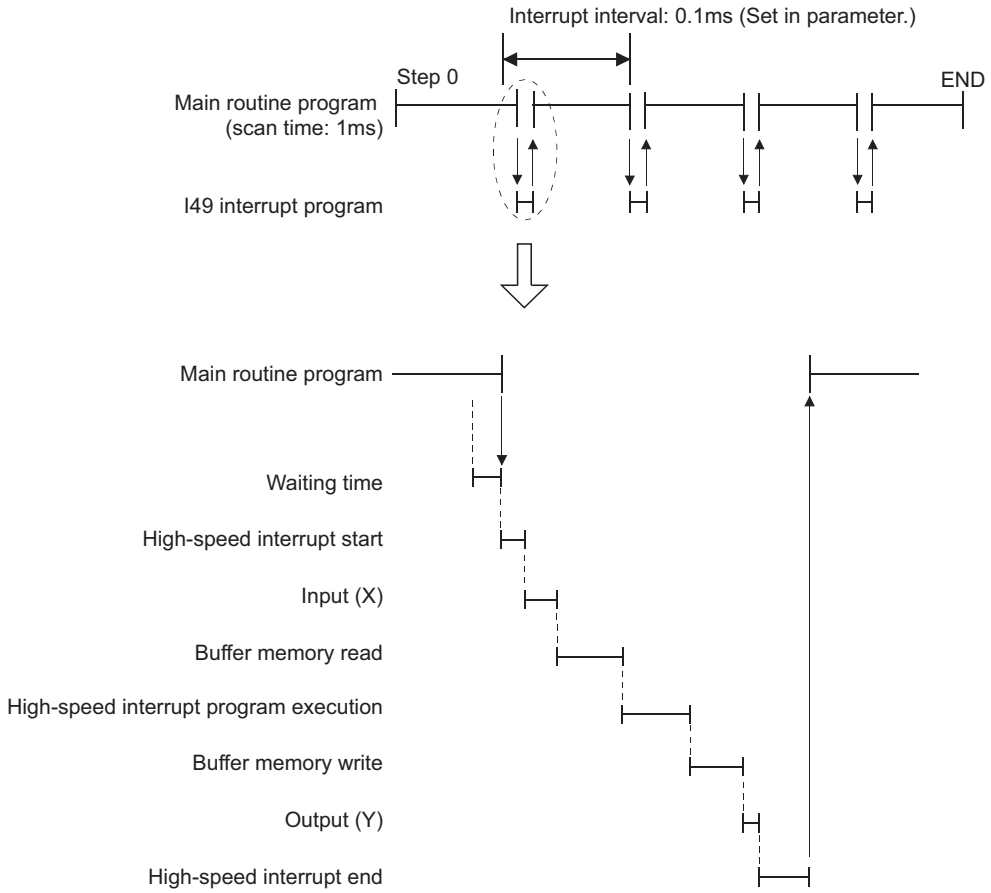


Because the cause number "2" is not set, the ERR.LED remains off even if a fuse blown is detected.
In this case, even if another error with the cause number "2" (I/O module verification error or intelligent function module verification error) is detected, the ERR.LED remains off.

- If "0" is set to the cause number setting area (setting that does not turn on the LED), SM0 (Diagnostic errors) and SM1 (Self-diagnostic error) turn on, and the error code is stored to SD0 (Diagnostic errors).

3.21 High-Speed Interrupt Function Note 3.15

This function executes an interrupt program at fixed intervals of 0.1 to 1.0ms using the high-speed interrupt pointer (I49). Also, the I/O response improves because the I/O signal data set in parameters and the data in the buffer memory of each intelligent function module are refreshed before and after these high-speed interrupt programs are executed. This enables high-accuracy control such as precise position detection.



This function consists of the following three functions.

- High-speed interrupt program execution function: Page 226, Section 3.21.1
- High-speed I/O refresh function and high-speed buffer transfer function: Page 227, Section 3.21.2

Remark

For the processing time of the high-speed interrupt function, refer to Page 491, Appendix 3 (15).

Note 3.15 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU support this function. (Page 466, Appendix 2)

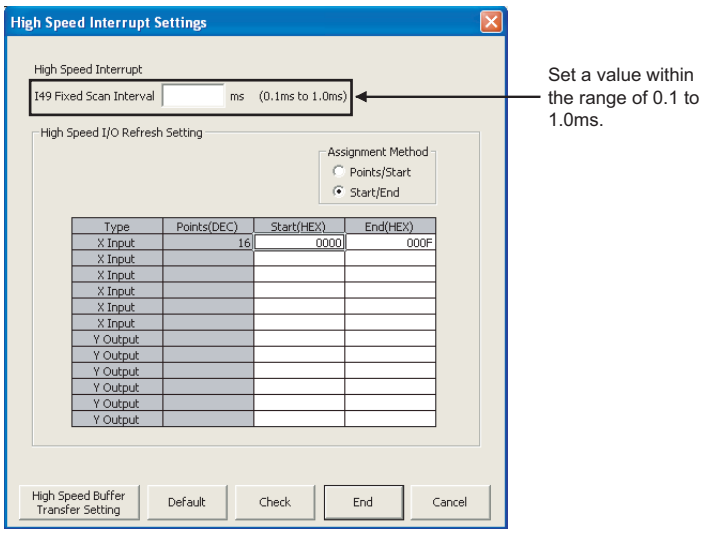
3.21.1 High-speed interrupt program execution function

This function executes interrupt programs according to the high-speed interrupt pointer (I49).

(1) Setting method

Open the High Speed Interrupt Settings window and set a value to "I49 Fixed Scan Interval" within the range of 0.1 to 1.0ms.

Project window⇨[Parameter] ⇨[PLC Parameter] ⇨"PLC System" tab⇨"System Interrupt Settings", "High Speed Interrupt Settings" button



(2) Precautions

(a) High-speed interrupts while interrupts are disabled

High-speed interrupt programs are not executed while interrupts are disabled. To execute the programs, the interrupt enable condition needs to be established.

For the functions that disable interrupts and delay the startup of a high-speed interrupt, refer to Page 231, Section 3.21.3 (3).

(b) High-speed interrupts that are ignored

If interrupts are disabled for the period longer than the set interrupt interval, there is a case that high-speed interrupts are ignored. If a high-speed interrupt occurs twice during an interrupt disabled period, the second interrupt is ignored.

(c) Executability of this function

This function is executed when all of the following conditions are met.

- The EI instruction is being executed.
- The CPU module is running.
- The high-speed interrupt pointer (I49) is not masked by the IMASK instruction. (Default: Not masked)

For the IMASK and EI instructions, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

3.21.2 High-speed I/O refresh function and high-speed buffer transfer function

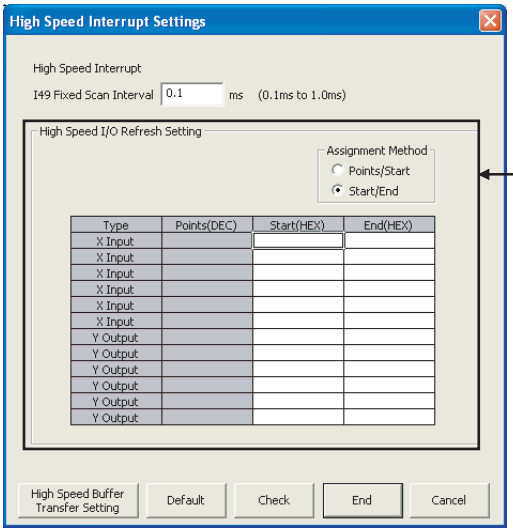
The high-speed I/O refresh function refreshes I/O signal data between I/O modules or intelligent function modules and the CPU module at the specified interrupt intervals.
The high-speed buffer transfer function refreshes data between the buffer memory in intelligent function modules and the devices in the CPU module at the specified interrupt intervals.

(1) Setting method

(a) High-speed I/O refresh function

Open the High Speed Interrupt Settings window and set the refresh ranges for X/Y.

Project window⇨[Parameter]⇨[PLC Parameter]⇨"PLC System" tab⇨"System Interrupt Settings", "High Speed Interrupt Settings" button



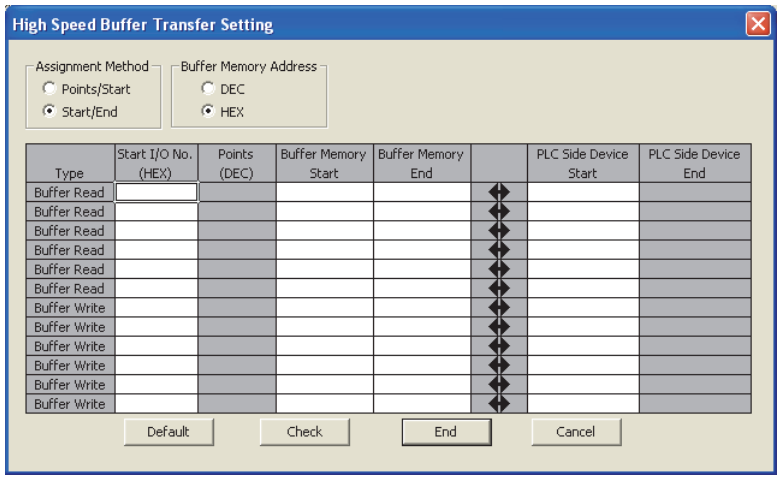
Item	Description	Restrictions	Number of settings
Assignment Method	Select an assignment method (Points/Start or Start/End).	-	-
Points(DEC)	The number of bits transferred (16 to 4096)	• I/O modules and intelligent function modules only • Numbers in the multiples of 16 only*1	Up to six settings for X input and Y output, respectively
Start(HEX)	Start device number (X0 to 0FF0 or Y0 to 0FF0)		
End(HEX)	End device number (X000F to 0FFF or Y000F to 0FFF)		

*1 This applies to both the start device number and the number of bits transferred.

(b) High-speed buffer transfer function

Open the High Speed Buffer Transfer Setting window and set the transfer ranges.

Project window⇨[Parameter]⇨[PLC Parameter]⇨"PLC System" tab⇨"System Interrupt Settings", "High Speed Interrupt Settings" button⇨"High Speed Buffer Transfer Setting" button



Item	Description	Restrictions	Number of settings
Assignment Method	Select an assignment method (Points/Start or Start/End).	-	-
Buffer Memory Address	Select an buffer memory address input system (DEC. or HEX.).		
Start I/O No. (HEX)	Start I/O number ÷ 10_H (0 to FF_H)	Intelligent function modules only	Up to six settings for read and write, respectively
Points (DEC)	Number of words transferred (1, 2 to $FFFE_H$)	• Intelligent function modules only • Even addresses and even words only*1	
Buffer Memory Start	Start address (0_H to $FFFF_H$)		
Buffer Memory End	End address (0_H to $FFFF_H$)		
PLC Side Device Start	Start device number	D, W, D (extended data register), W (extended link register), R, and ZR	
PLC Side Device End	End device number		

*1 An odd address is allowed when the number of words transferred is set to 1.

Point

Mount the target modules of this function on a main base unit. (Access time to modules mounted on a main base unit is shorter than that to modules on an extension base unit.)

(2) Precautions

(a) Executability of this function

This function is executed when all of the following conditions are met.

- The EI instruction is being executed.
- The CPU module is running.
- The high-speed interrupt pointer (I49) is not masked by the IMASK instruction. (Default: Not masked)

For the IMASK and EI instructions, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

3.21.3 Precautions

This section describes precautions for executing the high-speed interrupt function.

(1) Functions that delay the startup of high-speed interrupts

When any of the functions in the table below is being executed, high-speed interrupts cannot be executed at preset intervals.

Item	Operation when executed
Multiple CPU system configuration	The startup of a high-speed interrupt delays for about 15μs.
Connection of the QA1S5□B, QA1S6□B, and QA6□B	The startup of a high-speed interrupt delays for about 18μs for the first level and delays for about 30μs for the second and later levels.
Interrupt program (I0 to I48, I50 to I255), fixed scan execution type program	Only one interrupt is executed at a time. A high-speed interrupt starts after the execution of an interrupt program or fixed scan execution type program ends.
Operation when a continuation error occurs	The startup of a high-speed interrupt delays for the time required to detect a continuation error (instruction execution time + continuation error notification time (20μs)).
Execution of an instruction	Interrupts are disabled while an instruction is being executed. A high-speed interrupt starts after the execution of the instruction is completed.
Link direct device, instruction that accesses to module access device, MC protocol, and device test	The startup of a high-speed interrupt delays for more than 15μs.

I/O refresh, link refresh, auto refresh (intelligent function module), and multiple CPU auto refresh	The startup of a high-speed interrupt delays for the following period of time.
	• When I/O refresh is performed on the module mounted
	  • on the main base unit: 24μs maximum
	  • on the extension base unit: 40μs maximum
	• When link refresh or auto refresh is performed on the module mounted
	  • on the main base unit: 30μs maximum
	  • on the extension base unit: 70μs maximum
	• When multiple CPU auto refresh is performed on the module mounted
	  • on the main base unit: 250μs maximum

Interlink transfer	The startup of a high-speed interrupt delays for the following period of time. • Interlink transfer on the main base unit: 15μs maximum • Interlink transfer on the extension base unit: 30μs maximum
Ladder monitor, device batch monitor, and entry data monitor	The startup of a high-speed interrupt delays for the monitoring time (0.121μs x number of device points + 9μs). If self-monitoring and monitoring via any intelligent function module are requested at the same time, the startup of a high-speed interrupt delays for about 70μs.
Local device monitor	The startup of a high-speed interrupt delays for the monitoring time (340μs + performance when the local devices are used).
Buffer memory batch monitor	The startup of a high-speed interrupt delays for about 820μs. (The number of points to be monitored: 55)
Monitor condition setting	The startup of a high-speed interrupt delays for the following period of time. • When step is specified: 30μs • When internal user device is specified: 15μs
Read from PLC (during RUN)	The startup of a high-speed interrupt delays for the following period of time. • Reading data from the device memory (59K words): 250μs • Reading data from other memory: 15μs
Write to PLC (during RUN), file transfer function (FTP)	The startup of a high-speed interrupt delays for the following period of time. • Writing data to the comment file in the program memory or standard RAM: 60μs or longer • Writing data to the file register in the standard RAM: 60μs or longer • Writing data to other memory: 15μs
Forced disablement of SD memory card	The startup of a high-speed interrupt delays for about 20μs.
Program memory batch download	The startup of a high-speed interrupt delays for about 15μs.

Executional conditioned device test	The startup of a high-speed interrupt delays for the following period of time.
	• When registered/disabled: 50μs maximum
	• During execution: Tested using the same program name and step No.
	  • Internal user device: 12μs × (number of device points) + 25μs
	  • I/O device: 28μs × (number of device points) + 25μs

Item	Operation when executed
Scan time measurement	The startup of a high-speed interrupt delays for the following period of time. • When registered: 100μs • During measurement: 15μs
Sampling trace	The startup of a high-speed interrupt delays for the following period of time. • At startup: 110μs • During execution: 11μs • Internal user device (word: 50 points, bit: 50 points)
Data logging function	The startup of a high-speed interrupt delays for the following period of time. Logging type: Continuous logging (one setting, no CSV output) • Each scanning cycle (Device data match.), internal user device (128 points): 15μs • Step No. specification, internal user device (128 points): 45μs
Data logging by a high speed data logger module	The startup of a high-speed interrupt delays for the following period of time. • General data sampling: Link direct device (256 points): 1ms • High-speed data sampling: File register (256 points): 40μs
Online change	The startup of a high-speed interrupt delays for the following period of time. • Online change (ladder mode): 40μs maximum • Online change (files): 15μs maximum
Diagnostic function (such as PC diagnostics, system monitor)	The startup of a high-speed interrupt delays for about 1ms (maximum) in the multiple CPU system or if any diagnostic target module is mounted on the base unit.
Error clear	The startup of a high-speed interrupt delays for the following period of time. • Annunciator error: about 30μs • Battery error: about 25μs
IP packet transfer function	The startup of a high-speed interrupt delays for about 30μs.
Access command issuance from intelligent function modules, such as the QJ71C24 and QJ71E71, to the CPU module	The startup of a high-speed interrupt delays for the following period of time. • Read/write command: (0.060 × (number of device points) + 12) μs • Batch write command: (0.10 × (number of device points)) μs • Batch read command: (0.13 × (number of device points)) μs
Clock setting using a programming tool	The startup of a high-speed interrupt delays for about 30μs.

(2) Items disabled when the high-speed interrupt function is used

Item	Operation when used
External input/output forced on/off	This function is not executed and ignored in high-speed interrupt programs. (No error occurs.)
Index register	High-speed interrupt programs do not save nor restore data in the index register. If data in the index register are changed in a high-speed interrupt program, the data are overwritten.
Local device	High-speed interrupt programs do not save nor restore data in local devices. If data in local devices are changed in a high-speed interrupt program, the data are overwritten in the program that was being executed before the interrupt.
File register having the same name as a program	High-speed interrupt programs do not automatically change the file register name to the same name as a program. If data in the file register are changed in a high-speed interrupt program, the data are overwritten in the program that was being executed before the interrupt.
Device comment having the same name as a program	High-speed interrupt programs do not automatically change the device comment file name to the same name as a program. The following data are not updated in a high-speed interrupt program. • Comment use (SM650) • Comment drive (SD650) • Comment file name (SD651 to SD656) • Memory card in-use flag (SM604) • Memory card use conditions (SD604) • Drive 3/4 in-use flag (SM624) • Drive 3/4 use conditions (SD624)

(3) Time required for one interrupt program

If exceeded, "WDT ERROR" may occur and the operation of the high-speed interrupt program is not guaranteed.

(4) "Interrupt Program/Fixed Scan Program Setting" in PLC parameter

For high-speed interrupts, the "High Speed Execution" parameter setting is ignored even when selected.

(5) High-speed I/O refresh function and high-speed buffer transfer function

- Mount the target modules of these functions on the main base unit.
- When the high-speed buffer transfer function is executed, an error does not occur even when the file register is used exceeding the setting range. Data out of the setting range, however, are not transferred. (There is no impact on other device data.)

(6) Programming precautions

Refer to the programming precautions for interrupt programs. (☞ Page 82, Section 2.9)

(7) Occurrence of the same interrupt

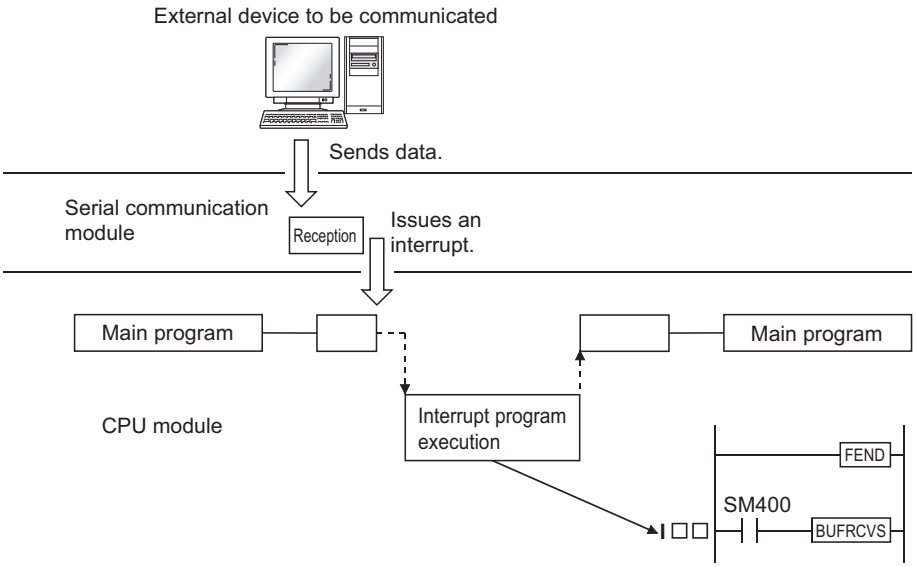
If the same interrupt factor occurs while a high-speed interrupt program (I49) is being executed, the second interrupt will be ignored.

3.22 Interrupt from Intelligent Function Module

The CPU module can execute an interrupt program (I □) by the interrupt request from the intelligent function module. For example, the serial communication module can receive data by an interrupt program when the following data communication functions are executed.

- Data reception during communication by nonprocedural protocol
- Data reception during communication by bidirectional protocol

Using an interrupt program enables a CPU module to receive data quickly.



To execute an interrupt program by an interrupt from the intelligent function module, select "Intelligent Function Module Interrupt Pointer Setting" in "Intelligent Function Module Setting" of the PLC system tab in the PLC parameter dialog box.

[PLC Parameter] ⇨ [PLC System] ⇨ [Intelligent Function Module Setting] ⇨ [Interrupt Pointer Setting] button

To configure "Intelligent Function Module Parameter" at the intelligent function module is also required.

For execution of an interrupt program by an interrupt from the intelligent function module, refer to the following.

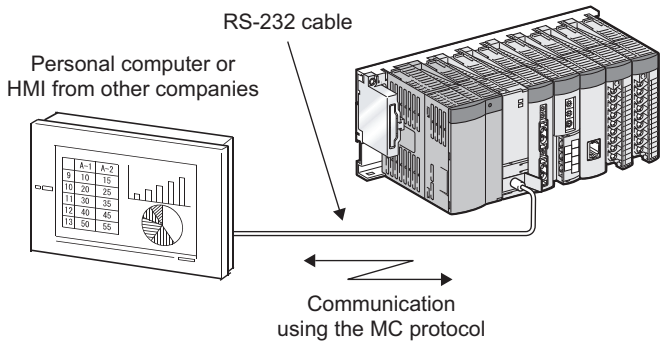
Manual for the intelligent function module used

Remark

For the numbers of interrupt pointers available for an interrupt from the intelligent function module, refer to Page 412, Section 4.11.

3.23 Serial Communication Function Note 3.16

This function communicates data using the MC protocol by connecting the RS-232 interface of the CPU module and a personal computer or HMI from other companies with an RS-232 cable. This section describes the specifications, functions, and settings of the function.



Point

- A personal computer or HMI from other companies can communicate only with a CPU module connected to it. It cannot communicate with stations connected over CC-Link IE, MELSECNET/H, Ethernet, or CC-Link.
- This function is not used to connect a programming tool or GOT to a CPU module.
- To execute this function with the Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, or Q26UDHCPU, use GX Works2. (GX Developer does not support this function.)

Note 3.16 Universal

Before executing the function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used.

(Page 466, Appendix 2)

The Built-in Ethernet port QCPU does not support this function.

(1) Specifications

(a) Transmission specifications

The following is the transmission specifications of RS-232 used for this function. Check that the specifications of the personal computer or HMI from other companies match those in the following table.

Item	Setting range	Default
Communication method	-	Full-duplex communication
Synchronization method	-	Asynchronous method
Transmission speed*1	9.6kbps, 19.2kbps, 38.4kbps, 57.6kbps, 115.2kbps	19.2kbps
Data format	-	• Start bit: 1 • Data bit: 8 • Parity bit: Odd • Stop bit: 1
MC protocol format*2 (automatic detection)	-	• Format 4 (ASCII) • Format 5 (binary)
Frame*2	-	• QnA-compatible 3C frame • QnA-compatible 4C frame
Transmission control	-	DTR/DSR control
Sum check*1	Checked/not checked	Checked
Transmission wait time*1	No waiting time, 10ms to 150ms (in increments of 10ms)	No waiting time
RUN write setting*1	Permit (selected), prohibited (deselected)	Prohibited (deselected)
Overall cable distance	-	15m

*1 The item is set in the PLC parameter of the programming tool. (☞ Page 456, Appendix 1.2.12)

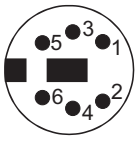
*2 Relationship between the MC protocol formats and frames is shown below.

○ : Available, × : Unavailable

Function		Format 4	Format 5
Communication in ASCII code	QnA-compatible 3C frame	○	×
	QnA-compatible 4C frame	○	×
Communication in binary code	QnA-compatible 4C frame	×	○

(b) RS-232 connector specifications

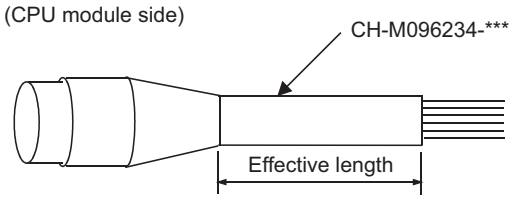
The following is the specifications of the RS-232 connector of a CPU module.

Appearance	Pin number	Signal	Signal name
 Mini-DIN 6 pins (female)	1	RD (RXD)	Receive data
	2	SD (TXD)	Send data
	3	SG	Signal ground
	4	-	-
	5	DR (DSR)	Data setting ready
	6	ER (DTR)	Data terminal ready

(c) RS-232 cable

Use either of the following RS-232 cables between a personal computer or HMI from other companies and a CPU module.

- QC30R2 (cable length: 3m)
- CH-M096234-*** (manufactured by CHUGAI Co., Ltd.)
  Cable with a Mini-DIN connector on one side and without connector on the other side
  *** indicates a cable length, which can be lengthened up to 15m in increments of 0.1m.



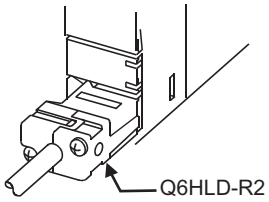
Signal layout of the CH-M096234-*** connector on CPU module side

Pin number	1	2	3	4	5	6	Metal shell
Signal	RD	SD	SG	-	DR	ER	
Wire core	Red	Black	Green/white	-	Yellow	Brown	Shield

Point

To fix the RS-232 cable to the CPU module, the use of the RS-232 connector disconnection prevention holder (Q6HLD-R2) is recommended. For the Q6HLD-R2, refer to the following.

Q6HLD-R2 Type RS-232 Connector Disconnection Prevention Holder User's Manual



(2) Commands

The following table lists the MC protocol commands that can be executed.

Function			Command	Processing	Number of processing points
Device memory	Batch read	In units of bits	0401(00 ☐ 1)	Reads bit devices in units of 1 point.	ASCII: 3584 points BIN: 7168 points
		In units of words	0401(00 ☐ 0)	Reads bit devices in units of 16 points.	480 words (7680 points)
				Reads word devices in units of 1 point.	480 points
	Batch write*1	In units of bits	1401(00 ☐ 1)	Writes bit devices in units of 1 point.	ASCII: 3584 points BIN: 7168 points
		In units of words	1401(00 ☐ 0)	Writes bit devices in units of 16 points.	480 words (7680 points)
				Writes word devices in units of 1 point.	480 points
	Random read *2*3	In units of words	0403(00 ☐ 0)	Reads bit devices in units of 16 points or 32 points by specifying the device or device number at random.	96 points
				Reads word devices in units of 1 point or 2 points by specifying the device or device number at random.	
	Test*1 (random write)	In units of bits	1402(00 ☐ 1)	Sets/resets bit devices in units of 1 point by specifying the device or device number at random.	94 points
		In units of words*2	1402(00 ☐ 0)	Sets/resets bit devices in units of 16 points or 32 points by specifying the device or device number at random.	*5
				Writes word devices in units of 1 point or 2 points by specifying the device or device number at random.	
	Monitor registration *2 *3 *4	In units of words	0801(00 ☐ 0)	Registers bit devices to be monitored in units of 16 points or 32 points.	96 points
				Registers word devices to be monitored in units of 1 point or 2 points.	96 points
	Monitor	In units of words	0802(00 ☐ 0)	Monitors devices registered for monitoring.	Number of monitor registration points

*1 To perform online change, check the "Permit" checkbox under "RUN write setting".

*2 Devices such as TS, TC, SS, SC, CS, and CC cannot be specified in units of words. For the monitor registration, an error (4032_H) occurs during the monitor operation.

*3 The monitor condition specification cannot be used for these commands.

*4 Do not execute monitor registration from multiple external devices. If executed, the last monitor registration becomes valid.

*5 Set the number of processing points within the range of the following calculation formula.
(number of word access points) × 12 + (number of double word access points) × 14 ≤ 960

- One point of a bit device corresponds to 16 bits for word access or to 32 bits for double word access.
- One point of a word device corresponds to one word for word access or to two words for double word access.

(3) Accessible devices

The following table lists the accessible devices by the serial communication function.

Category	Device		Device code*1		Device number range	
			ASCII	Binary		
Internal system device	Function input		---	---	(Cannot be accessed)	Hexadecimal
	Function output		---	---		Hexadecimal
	Function register		---	---		Decimal
	Special relay		SM	91_H	Within the device number range of the CPU module accessed. Note, however, that local devices cannot be accessed.	Decimal
	Special register		SD	$A9_H$		Decimal
Internal user device	Input		X *	$9C_H$		Hexadecimal
	Output		Y *	$9D_H$		Hexadecimal
	Internal relay		M *	90_H		Decimal
	Latch relay		L *	92_H		Decimal
	Annunciator		F *	93_H		Decimal
	Edge relay		V *	94_H		Decimal
	Link relay		B *	$A0_H$		Hexadecimal
	Data register		D *	$A8_H$		Decimal
	Link register		W *	$B4_H$		Hexadecimal
	Timer	Contact	TS	$C1_H$		Decimal
		Coil	TC	$C0_H$		
		Current value	TN	$C2_H$		
	Retentive timer	Contact	SS	$C7_H$		Decimal
		Coil	SC	$C6_H$		
		Current value	SN	$C8_H$		
	Counter	Contact	CS	$C4_H$		Decimal
		Coil	CC	$C3_H$		
		Current value	CN	$C5_H$		
	Link special relay		SB	$A1_H$		Hexadecimal
	Link special register		SW	$B5_H$		Hexadecimal
	Step relay		S *	98_H		Decimal
	Direct input*2		DX	$A2_H$		Hexadecimal
	Direct output*2		DY	$A3_H$		Hexadecimal
Index register	Index register		Z *	CC_H	Within the device number range of the CPU module accessed	Decimal
File register*3	File register		R *	AF_H		Decimal
			ZR	$B0_H$		Hexadecimal
Extended data register*3	Extended data register		D *	$A8_H$	• Binary: Within the device number range of the CPU module accessed • ASCII: 000000 to 999999 (up to 976.6K points)	Decimal
Extended link register*3	Extended link register		W *	$B4_H$	Within the device number range of the CPU module accessed	Hexadecimal

1 This is a code specified in MC protocol messages. When communicating data in ASCII code, specify the code in two characters. If the code consists of only one character, add "" (ASCII code: $2A_H$) or a space (ASCII code: 20_H) after the character.

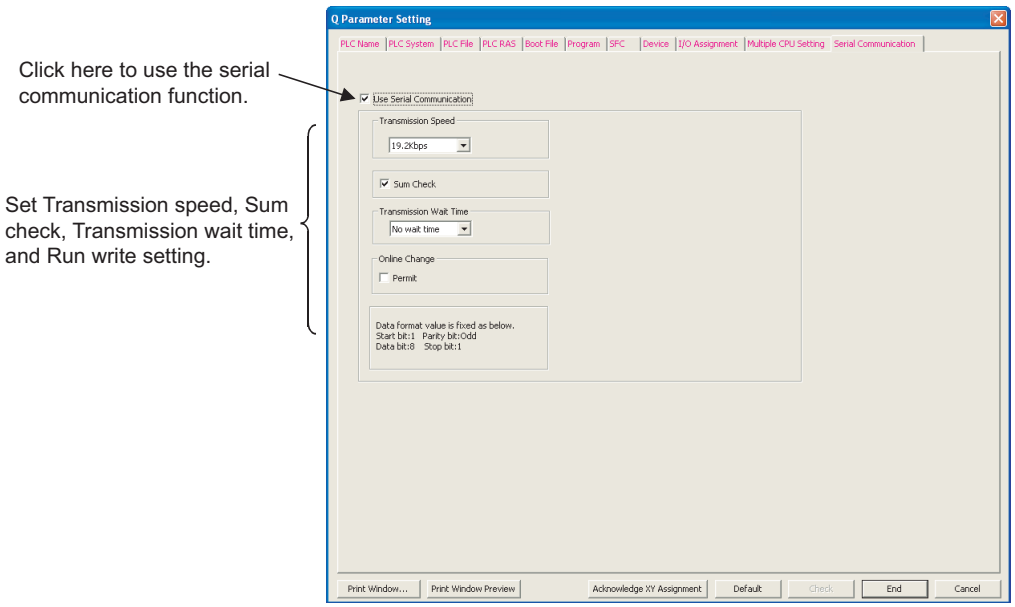
*2 Devices of DX/DY1000 or later are not available. Use X/Y devices to access devices of X/Y1000 or later.

*3 The Q00UJCPU does not support these devices.

(4) Setting transmission specifications

Set a transmission speed, sum check status, transmission wait time, and online change status for this function in the Serial Communication tab of the PLC parameter dialog box. (☞ Page 456, Appendix 1.2.12)

- Select "Use Serial Communication" to communicate with a personal computer or HMI from other companies using this function.
- Set a transmission speed, sum check status, transmission wait time, and online change status.



(5) Precautions

(a) Switching a connection from an HMI from other companies to a programming tool

A connection device can be switched from a personal computer or HMI from other companies to a programming tool during communication. However, this operation causes a communication error in the personal computer or HMI. For a startup method of the personal computer or HMI after it is reconnected to the CPU module, refer to the manual for the device used.

(b) Transmission speed set in the Transfer Setup screen

When "Use Serial Communication" is selected, the transmission speed set in the Transfer setup screen of the programming tool is ignored.

(c) Communication error

If any of the following conditions is met, no response is returned (a communication error occurs). Take a corrective action.

- The serial communication function is set not to be used.
- Communication is made at different transmission speed and data format.
- A frame to be sent has no correct starting end or terminal.
  - 3C frame format 4: ENQ/CR + LF
  - 4C frame format 4: ENQ/CR + LF
  - 4C frame format 5[*1]: DLE+STX/DLE + ETX

*1 When the "Sum Check" checkbox is selected, the sumcheck code is included.

- The frame identification number of a frame to be sent is incorrect.
- The number of transmission bytes is under the header part size.

(6) Error codes during communication with the serial communication function

The following table lists the error codes (together with their descriptions and corrective actions) sent from the CPU module to the external device when an error occurs during communication using the serial communication function.

Error code (hexadecimal)	Error item	Description	Corrective action
4000_H to $4FFF_H$	-	Error detected by the CPU module (error occurred by other than the serial communication function)	Refer to the following manual and take corrective action. QCPU User's Manual (Hardware Design, Maintenance and Inspection)
7155_H	Unregistered monitor error	A monitor request was given before monitor registration.	Give a monitor request after registering a device to be monitored.
7157_H	Request target specification error	A CPU module that does not support the serial communication function is specified as a request target module or is in the specified route.	Check that the transmission message is addressed to the CPU module that supports the serial communication function. If not, correct the address and restart communication.
7164_H	Request data error	The requested data or device specification method is incorrect.	Check the sent message/requested data of the external device, correct it, and restart communication.
7167_H	Disabled during RUN	A write command was specified while online change is disabled.	• Enable the online change and restart communication. • Set the CPU module to STOP and restart communication.
$7E40_H$	Command error	A subcommand or a command that does not exist is specified.	Check and correct the sent message of the external device and restart communication.
$7E41_H$	Data length error	The number of points specified for random write/read exceeds the number of points enabled for communication.	Check and correct the sent message of the external device and restart communication.
$7E42_H$	Data count error	The requested number of points exceeds the range of the command.	Check and correct the sent message of the external device and restart communication.
$7E43_H$	Device error	The device specified does not exist. The device specified cannot be specified by the corresponding command.	Check and correct the sent message of the external device and restart communication.
$7E47_H$	Continuous request error	The next request was received before the response message was returned.	Do not give continuous requests from the external device. Match the monitoring time of timer 1 with the time-out time of the external device.
$7F21_H$	Receive header section error	The command (frame) section specified is in error.	Check and correct the sent message of the external device and restart communication.
		The ASCII code received cannot be converted into binary.	
$7F22_H$	Command error	The command or device specified does not exist.	Check and correct the sent message of the external device and restart communication.
$7F23_H$	MC protocol message error	The data (such as ETX, CR+LF) specified after the character part does not exist or in error.	Check and correct the sent message of the external device and restart communication.
$7F24_H$	Sumcheck error	The calculated sumcheck does not match the received sumcheck.	Review the sumcheck of external device.

Error code (hexadecimal)	Error item	Description	Corrective action
$7F67_H$	Overrun error	The next data was received before the CPU module completed receive processing.	Reduce the communication speed and restart communication. Check the CPU module for momentary power failure. (For the CPU module, use the special register SD53 to check.) When an momentary power failure occurs, remove its cause.
$7F68_H$	Framing error	• The stop bit setting does not match. • Communication line became unstable by powering on/off the target device. • Noise is generated on the communication line.	• Match the setting of the CPU module with that of the external device. • Take noise reduction measures.
$7F69_H$	Parity error	• The parity bit setting does not match. • Communication line became unstable by powering on/off the target device. • Noise is generated on the communication line.	• Match the setting of the CPU module with that of the external device. • Take noise reduction measures.

3.24 Service Processing

3.24.1 Service processing setting

This function allows to set the time and the number of times of service processing performed at END processing by parameters.
Processing for requests from peripherals to the CPU module are performed with this function. The processing speed for the communication response to the requests varies depending on the scan time and communication load. Set the parameter of the service processing time as follows to achieve an optimal service processing environment for the system used.

- Setting a longer time can improve the processing speed for the communication response.
- Setting a shorter time can avoid a prolonged scan time caused by the service processing.

Communications from multiple peripherals to the CPU module may slow down the processing speed for the communication response. Adjust the settings for the system by specifying a longer service processing time or modifying the parameter settings of the peripherals to set longer timeout times taking the processing speed for the communication response and increases in scan times into consideration.

Point

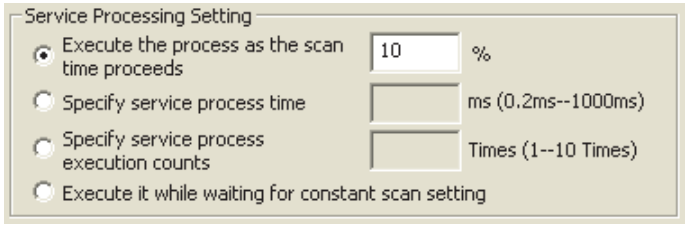
The service processing refers to the following:

- Communications via intelligent function modules (A link refresh from network modules is not included.)
- Communications via USB cables, RS-232 cables, and the built-in Ethernet port (Communications with programming tool and GOT)

Using the COM instruction enables service processing during program execution of same performance with service processing at END processing. Therefore, the high-speed service processing response can be performed even if the scan time is long.

(1) Parameter setting

Set the parameters in the PLC system tab of the PLC parameter dialog box.



To perform the service processing, select any of the parameter items in the following table. The setting value of deselected parameter cannot be entered. (Default: Execute the process as the scan time proceeds. = 10%)

Item	Description	Setting range	Remarks
Execute the process as the scan time proceeds.	Set the percentage of service processing for one scan.	• Range: 1 to 99% • Unit: 1%	Default when selected =10%
Specify service process time.	Set the time of service processing for one scan.	• Range: 0.2ms to 1000ms • Unit: 0.1ms	Default when selected =0.2ms
Specify service process execution counts.	Set the number of service processing for one scan.	• Range: 1 to 10 times • Unit: 1 time	Default when selected =1 time

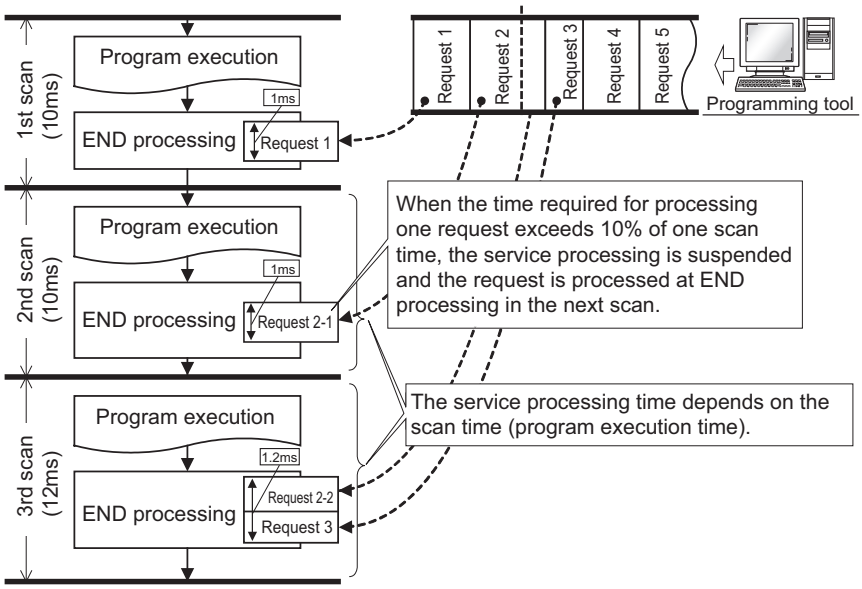
Item	Description	Setting range	Remarks
Execute it while waiting for constant scan setting.	Set whether to perform service processing during waiting time for constant scan setting.	-	Even when the waiting time is 0.2ms or less, the service processing time (0.2ms) will be added to the scan time at service processing execution.

(2) Operations for service processing setting

Operations for each service processing setting is described below.

(a) Operation when "Execute the process as the scan time proceeds." is selected

• Operation when 10% is set



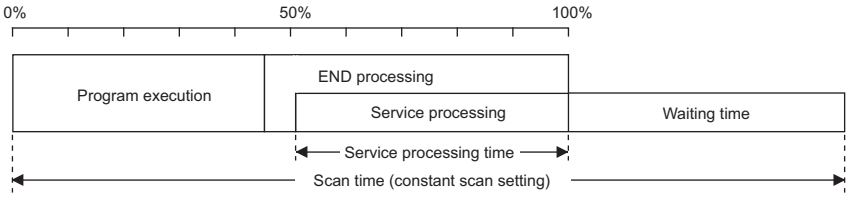
Point

If no request data for service processing exists, END processing speeds up by the request processing time. (The CPU module does not wait for requests.)

• Operation for constant scan setting

The calculation of the service processing time is a calculation of the percentage of the time excluding the waiting time of the constant scan from the scan time, not a calculation of the percentage of the scan time.

Ex. Operation when 50% is set

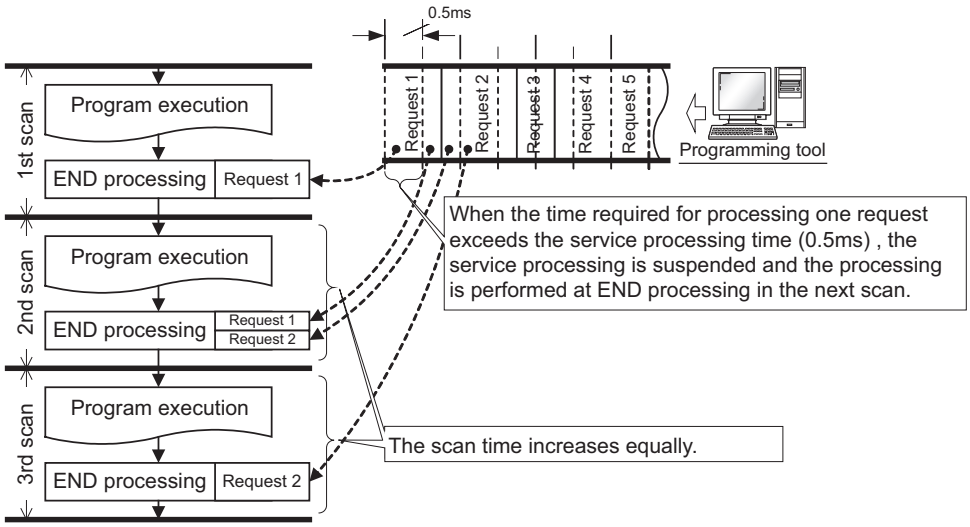


Point

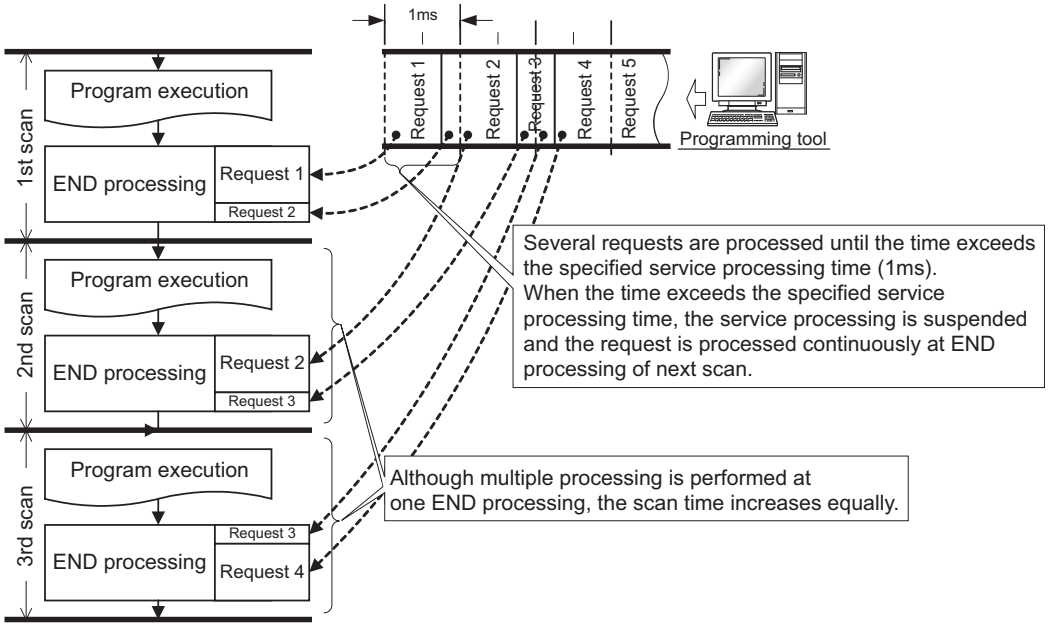
When setting the constant scan, selecting "Execute it while waiting for constant scan setting." can perform the service processing efficiently. (☞ Page 245, Section 3.24.1 (2) (d))

(b) Operation when "Specify service process time." is selected

• Operation when 0.5ms is set



• Operation when 1ms is set

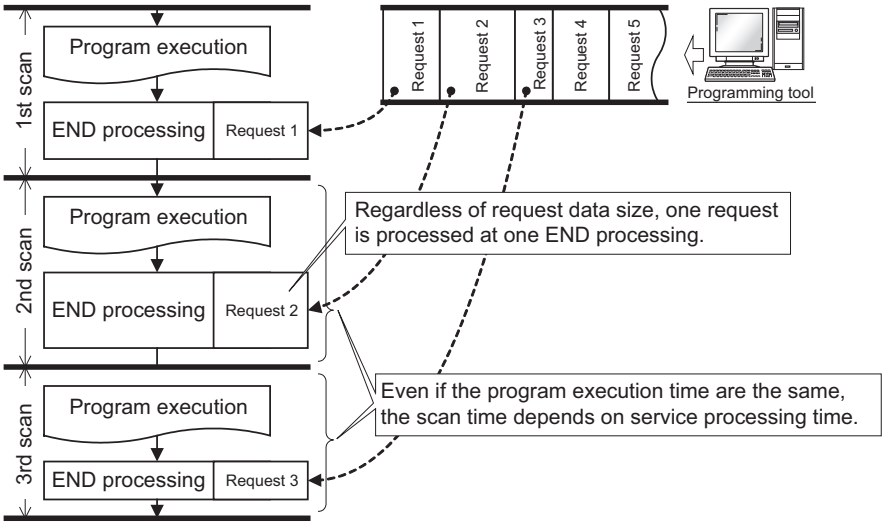


Point

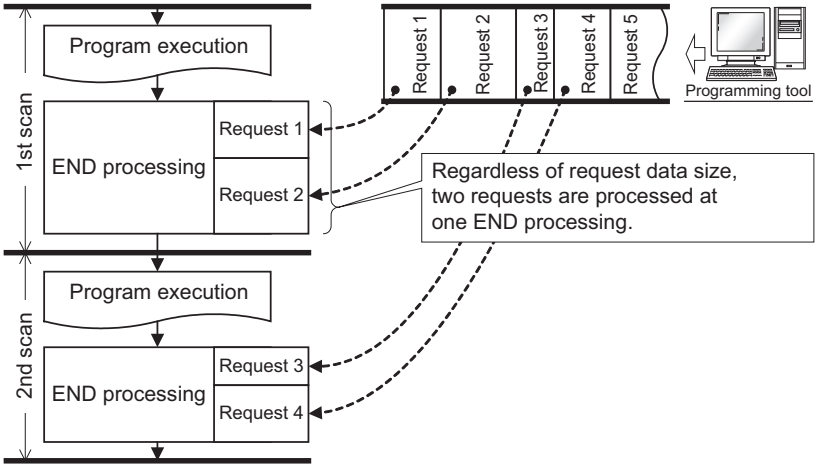
If no request data exists when setting the service processing time, END processing speeds up by the request processing time. (The CPU module does not wait for requests.)

(c) Operation when "Specify service process execution counts." is selected

- Operation when 1 time is set



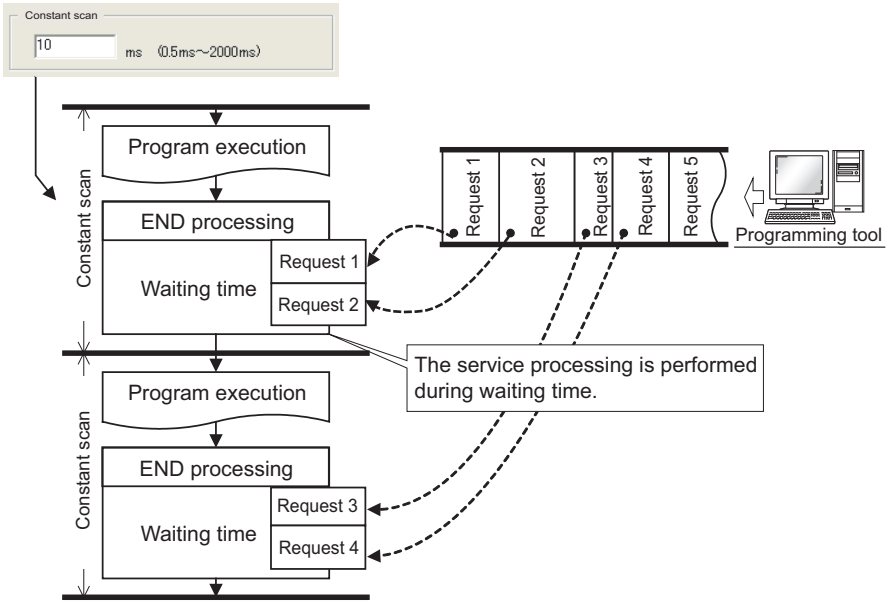
- Operation when 2 times is set



Point

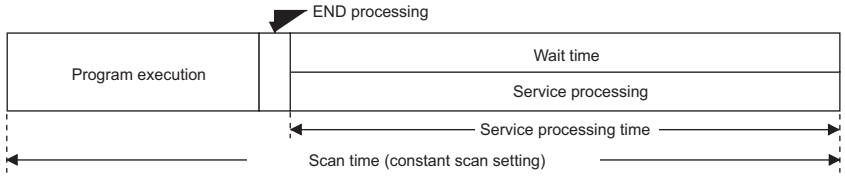
- When several devices are connected to one CPU module, each device requests service processing. When the CPU module receives requests from several devices simultaneously, a single END processing can accept several requests simultaneously if the service processing count is set to the number of connected devices. This improves response performance. (Note that the scan time increases by the service processing time.)
- If no request data exists when setting the service processing count, END processing speeds up by the request processing time. (The CPU module does not wait for requests.)

(d) Operation when "Execute it while waiting for constant scan setting." is selected

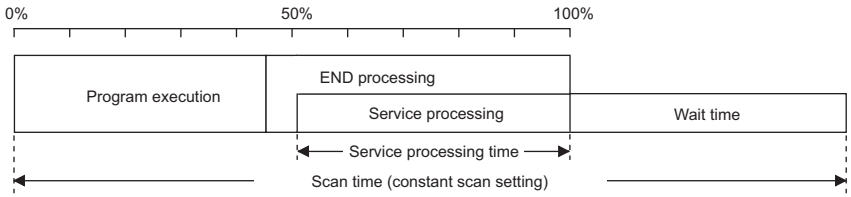


Point

- When setting the constant scan, selecting "Execute it while waiting for constant scan setting." can perform the service processing efficiently.
  - When "Execute it while waiting for constant scan setting." is selected



  - When "Execute the process as the scan time proceeds." is selected (50% is set.)



- Even when there is no waiting time, the service processing (0.2ms) is performed. Therefore, when the waiting time is less than 0.2ms, the constant scan time may be exceeded.

(3) Precautions

The following describes precautions when the service processing setting is configured.

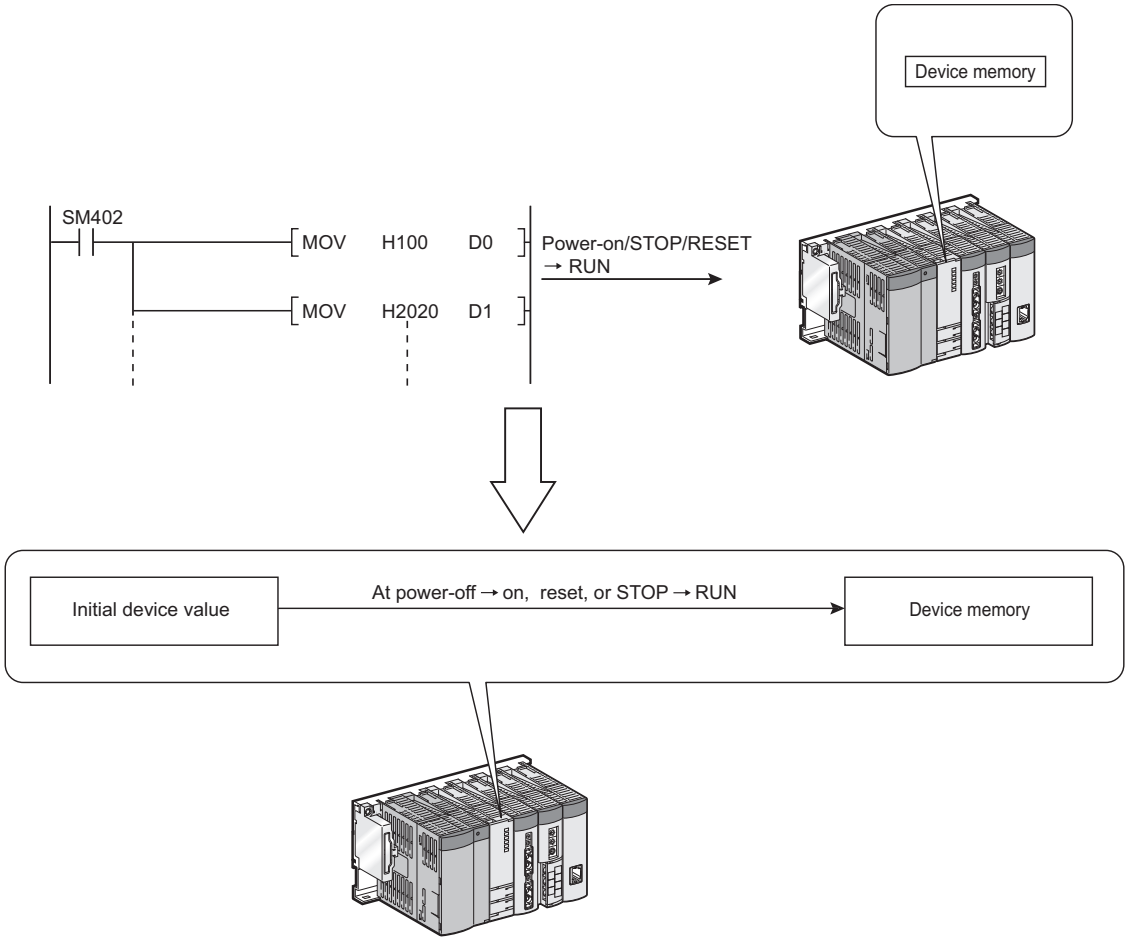
- For the following functions, scan time will be increased longer than the specified time during service processing even if the service processing time specification is set.
    - Online change
    - Change TC setting
    - Local device monitor
    - Program memory backup
    - Writing/reading data to/from a file register (The scan time will be increased when the write or read size is large.)
    - Writing to/reading from the buffer memory of the intelligent function module (The scan time will be increased when the write or read size is large.)
    - Access to a network module
        a) Diagnostic functions (CC IE Control diagnostics, CC IE Field diagnostics, MELSECNET diagnostics, Ethernet diagnostics, CC-Link/ CC-Link/LT diagnostics)
        b) Monitor function (Module access device, Link direct device)
- Note that the scan time will be increased much longer if the CPU module receives multiple requests simultaneously while the service processing count specification is set many.
- The response performance of service processing significantly decreases in the following cases. Set service processing time considering the time-out time of the peripheral.
    - Service processing time is set much shorter than the scan time.
    - Setting "Execute it while waiting for constant scan setting" results in increase in the scan time and decrease in the service processing time.
- An error of -20μs to +30μs occurs between the actual processing time and the set service processing time.
- When "Specify service process execution counts." is selected and Ethernet communication is performed, the scan time increases by the service processing time (approx. 500ms). To keep the time increase 500ms or less, select an item other than "Specify service process execution counts.". (Example: Select "Specify service process time." and set a desired time value.)
- If communications in the MC protocol are performed with an item other than "Specify service process execution counts" selected, data inconsistency may occur. To prevent data inconsistency, select "Specify service process execution counts".
- In the service processing, processing for the requests from each receive port (each intelligent function module, USB, the RS-232, and built-in Ethernet) is performed one by one. If multiple requests are sent from a single receive port, the service processing for those requests may not performed in the same scan regardless of whether or not the service processing time is left. Therefore, when communications are performed between multiple peripherals via built-in Ethernet port, response time to each peripheral cannot be shortened even though a longer service processing time is set. In this case, connect peripherals not only to the built-in Ethernet of the CPU module but also to the Ethernet module to shorten the response time.
- Since the processing for a request including accesses to files takes time, responding to the request necessarily takes time. The processing for request data is performed one by one in the service processing, and therefore responses to succeeding request data delays when the CPU module receives a request including accesses to files. For this reason, set a longer timeout time for peripherals when using a system that regularly send these requests.

3.25 Initial Device Value

This function registers data used in a program to the device of the CPU module or the buffer memory of the intelligent function module without a program.

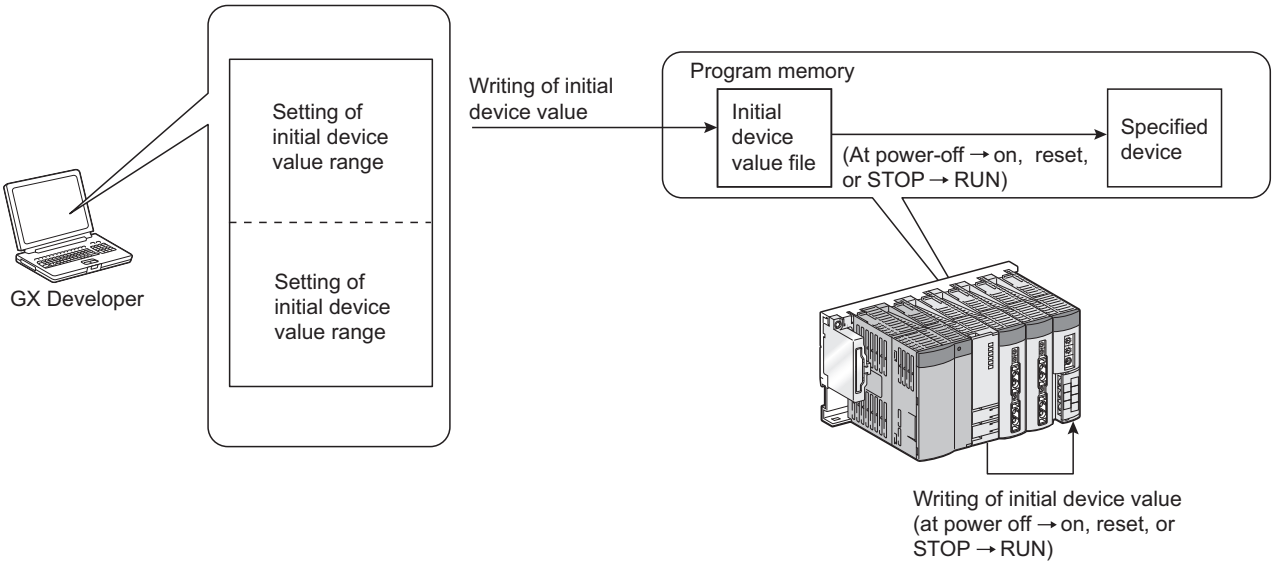
(1) Application

Use of this function can omit device data setting program by initial processing program.



(2) Timing when initial device values are written to the specified device

The CPU module writes data in the specified initial device value file to the specified device or the buffer memory of the intelligent function module when the CPU module is powered off and then on, is reset, or is set to the STOP status and then the RUN status.



(3) Applicable devices

For devices applicable for this function, refer to the following.

Operating manual for the programming tool used

(4) Setting procedures

For setting procedures of initial device values, refer to the following.

Operating manual for the programming tool used

(5) Precautions

(a) When initial device value and latch range are overlapped

In that case, the initial device value data takes priority. Therefore, the latch range data will be overwritten to the initial device value data after the CPU module is powered off and then on.

(b) Area disabling the initial device value setting when the CPU module is set from STOP to RUN

The initial device values are also reflected when the CPU module is set from STOP to RUN.

For an area where the initial device values should not be set when the CPU module is set from STOP to RUN (data that are set when the CPU module is powered off and then on and changed by a program), this function cannot be used.

Use an instruction such as the MOV instruction in the main routine program so that the initial device values will be set to the specified devices.

Use the TO instruction to write data to the buffer memory of the intelligent function module.

(c) Devices that require module synchronization setting

To set the following devices for the initial device value setting range, set "Module Synchronization" in the PLC system tab of the PLC parameter dialog box.

If the setting is not configured, the initial device values may not be set to the target module properly.

- Intelligent function module device (U□\G□)
- Link direct device (J□\W□, J□\SW□)

3.26 Battery Life-prolonging Function Note 3.17

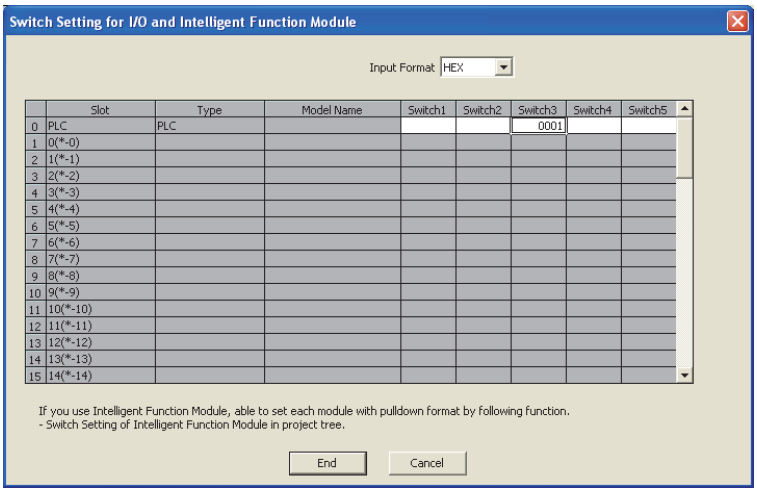
This function extends the life of battery installed in the CPU module by restricting data to be held by the battery to clock data only. This function initializes all data other than the clock data when the CPU module is powered off or is reset.

Data held by a battery		Description
Error history		The number of error history data is initialized to zero.
Latch device (L)		Cleared to zero.
Device in the latch range		Cleared to zero.
Standard RAM		Formatted (cleared to zero).
File register assigned to the standard RAM	Set "Use the same file name as the program."	A file is deleted.
	Set "Use the following file".	A file is deleted (A file is recreated at power-on or reset. (Data is cleared to zero.))

(1) Setting

Set the function in the I/O Assignment tab of the PLC parameter dialog box.

1. **Configure the I/O assignment setting.**
2. **Click the** Switch Setting **button.**
3. **Enter 0001_H to the switch 3 of the slot where the CPU module is mounted. (When entering the value to a slot in another station, the value is ignored.)**



(2) Battery life

For the life of battery installed in the CPU module when the battery life-prolonging function is used, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Note 3.17 Universal

The High-speed Universal model QCPU and Universal model Process CPU do not support this function.

3.27 Memory Check Function

This function checks whether data in the memories of the CPU module are not changed due to such as excessive electric noise.
Since the CPU module automatically checks a memory, setting for enabling this function is unnecessary.
This function does not require processing time.

(1) Data to be checked

(a) Program

The program during execution is compared with the user program written to the program memory.
If they do not match, a stop error, "RAM ERROR" (error code: 1160) is detected.

(b) Parameter

The parameters are compared with the ones written to the parameter-valid drive.

(c) Device memory

If the CPU module detects the change of data in the device memory, a stop error, "RAM ERROR" (error code: 1161) occurs. For the Universal model QCPU whose serial number (first five digits) is "13022" or later, the device information, which contains data change information, can be checked in SD927 and SD928. For details on the special register, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(2) Execution timing

- Program: At program execution
- Parameter:
  - When the CPU module is powered off and then on
  - When the CPU module is reset
  - When the CPU module is set from STOP to RUN after data are written to it
- Device memory: When device data are read

3.28 Program Cache Memory Auto Recovery Function Note 3.18

This function is to restore the error location automatically by using data in the program memory, which are stored in the flash ROM, when the memory check function (☞ Page 251, Section 3.27) detects an error in the program cache memory. This function enables the CPU module to continue its operation even if an error such as change of data in the program cache memory occurs due to noise.

(1) Execution condition

The function is executed when the following conditions are all met.

- The CPU module is in RUN status.
- Data in the program memory match those in the program cache memory.

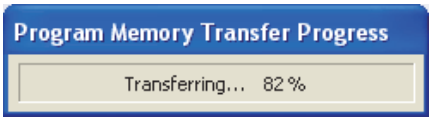
Note that the function is not executed in the following conditions even though the above conditions are met, and "RAM ERROR" (error code: 1160) occurs.

- The change of data in the program cache memory was detected while the following operations were being performed with the CPU module during RUN.

○: The auto recovery processing is performed and no error occurs. △: An error occurs depending on the condition. ×: An error occurs.

Operation item	Universal model QCPU with a serial number (first five digits) of "12121" or earlier	Universal model QCPU with a serial number (first five digits) of "12122" or later
Online change (ladder mode)	×	△*1
Password registration (program files)	×	○
Program memory batch download	×	△*1
Export to ROM format	×	○
Write to PLC (device comments)	×	△*1

*1 If the change of data is detected while data in the program cache memory are being transferred to the program memory (while the following window is being displayed on the programming tool), the auto recovery processing is not performed and an error occurs.



- The change of data in the program cache memory was detected by SFC program.*2
- The change of data in the program cache memory was detected by the dedicated instruction such as the S(P). instruction.*2
- The change of data in the program cache memory was detected by the rise instruction or the fall instruction.*2
- The data in the program cache memory was not restored due to the failure of the memory.

*2 The High-speed Universal model QCPU and Universal model Process CPU execute the function when the change of data in the program cache memory is detected by the corresponding factors above. (Data are automatically recovered and no error occurs.)

Note 3.18 Universal

Before executing the function, check the version of the CPU module used.

(☞ Page 466, Appendix 2)

Point

To match the data in the program memory and those in the program cache memory, configure the setting to transfer the data of the program cache memory to the program memory from "Options" screen.*2 (Page 178, Section 3.12.3 (4))

[Tool] ⇨ [Options]

*2 The transferring of the data in the program cache memory to the program memory is set by default.

Other than the above, you can also use "Program Memory Batch Download" to match the data in the program memory and those in the program cache memory.

(2) Execution timing

The execution timing of this function is described below.

- When the program is executed
- When data are verified or data are read from the programmable controller
- When any one of the write operations listed under (1) in this section is performed*3

*3 To perform the auto recovery processing when any one of the write operations listed under (1) in this section is performed, check the version of the CPU module and use the programming tool with the following version or later.
- GX Works2: 1.80J or later
- GX Developer: 8.102G or later

3.29 Latch Data Backup to Standard ROM

This function holds (backs up) latch data, such as device data and error history, to the standard ROM without using a battery when the system is stopped for a long period. This function helps to extend battery life.

Remark

When a CPU module other than the High-speed Universal model QCPU and Universal model Process CPU executes this function, the battery life-prolonging function is enabled regardless of its parameter setting. The function is disabled after backup data are restored.
The status of the function can be checked in SD119 (Battery life-prolonging factor).
For details of the battery life-prolonging function, refer to Page 250, Section 3.26.

(1) Backup target data and file size

The following table lists backup target data and sizes of files where data are stored.

<table>
<tr><th colspan="2">Backup target data</th><th>File size (byte)</th></tr>
<tr><td rowspan="2">Device data</td><td>• File register (R, ZR)*1
• Extended data register (D)*1
• Extended link register (W)*1</td><td>64 + 2 × Number of file register points</td></tr>
<tr><td>• Internal user device (M, L, B, F, V, T, ST, C, D, W)
• Index register (Z)/standard device register (Z)</td><td rowspan="8">File sizes differ depending on the CPU module used.*7
<table><tr><th>CPU module</th><th>Serial No. (first 5 digits)</th><th>File size (byte)</th></tr><tr><td rowspan="2">Q00UJCPU, Q00UCPU, Q01UCPU</td><td>11042 or earlier</td><td>87210</td></tr><tr><td>11043 or later</td><td>89790</td></tr><tr><td rowspan="2">Q02UCPU</td><td>11042 or earlier</td><td>87210</td></tr><tr><td>11043 or later</td><td>93630</td></tr><tr><td rowspan="2">QnUD(H)CPU, QnUDE(H)CPU</td><td>11042 or earlier</td><td>110890</td></tr><tr><td>11043 or later</td><td>117310</td></tr><tr><td rowspan="4">Q03UDVCPU</td><td>17011 or earlier</td><td>124952</td></tr><tr><td>17012 to 17102</td><td>132582</td></tr><tr><td>17103 or later</td><td>132608</td></tr><tr><td>18052 or later</td><td>132634</td></tr><tr><td rowspan="4">Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU</td><td>17011 or earlier</td><td>144152</td></tr><tr><td>17012 to 17102</td><td>151782</td></tr><tr><td>17103 or later</td><td>151808</td></tr><tr><td>18052 or later</td><td>151834</td></tr><tr><td rowspan="4">Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU</td><td>17011 or earlier</td><td>184728</td></tr><tr><td>17012 to 17102</td><td>192358</td></tr><tr><td>17103 or later</td><td>192384</td></tr><tr><td>18052 or later</td><td>192410</td></tr></table></td></tr>
<tr><td colspan="2">Error history</td></tr>
<tr><td colspan="2">Module error collection file (stored in the system memory)*3</td></tr>
<tr><td colspan="2">File transfer error history</td></tr>
<tr><td colspan="2">SFC program continuation start information</td></tr>
<tr><td colspan="2">Sampling trace information</td></tr>
<tr><td colspan="2">Scan time measurement information</td></tr>
<tr><td colspan="2">Standard RAM system data*4, *5</td><td>4</td></tr>
<tr><td colspan="2">Standard RAM directory data*4, *5</td><td>14212</td></tr>
<tr><td colspan="2">Trace setting (Sampling trace file)*6</td><td>16 + Sampling trace file size</td></tr>
<tr><td colspan="2">Module error collection file (stored in the standard RAM)</td><td>92 + 64 × Maximum number of module errors</td></tr>
</table>

Backup target data	File size (byte)
Parameter[*5]	16 + File size
Intelligent function module parameter[*5]	
Program[*5]	
Device comment[*5]	
Initial device value[*5]	
Drive heading[*5]	
Boot setting file[*5]	
Remote password[*5]	

*1 The data are backed up only when the file register in the standard RAM is used and the following parameter is set.
- CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU: Check the "Transfer to Standard ROM at Latch data backup operation" checkbox on the PLC file tab. (☞ Page 441, Appendix 1.2.3)
- High-speed Universal model QCPU and Universal model Process CPU: Check the "Backup all files in the internal of standard RAM" checkbox on the PLC system tab. (☞ Page 439, Appendix 1.2.2)

*2 The data are backed up only when the Universal model QCPU whose serial number (first five digits) is "10042" or later is used.

*3 The data are backed up regardless of the setting status of the module error collection function.

*4 The data are backed up regardless of the existence of files in the standard RAM if the "Backup all files in the internal of standard RAM" checkbox is checked on the PLC system tab.

*5 Only the High-speed Universal model QCPU and Universal model Process CPU back up these data.

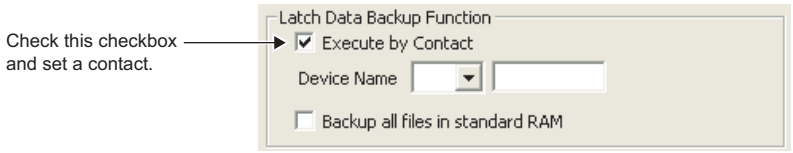
*6 A storage file is created only when the trace registration has been made. The data are not backed up in the following cases.
- Trace settings are stored in a memory card.
- Trace settings are not written to the CPU module.

*7 These are sizes when the device assignment is default. Sizes differ depending on parameter settings.

(2) Execution using a contact

(a) Setting method

Check the "Execute by contact" checkbox and set a contact for the latch data backup operation in the PLC system tab of the PLC parameter dialog box. (Devices X, M, or B can be selected.)



(b) Execution method

Backup starts at the rise of a contact (off → on).

After the backup operation is completed, the BAT.LED of the CPU module flashes (green). The module is in the standby status and is ready to be powered off.

(c) Precautions

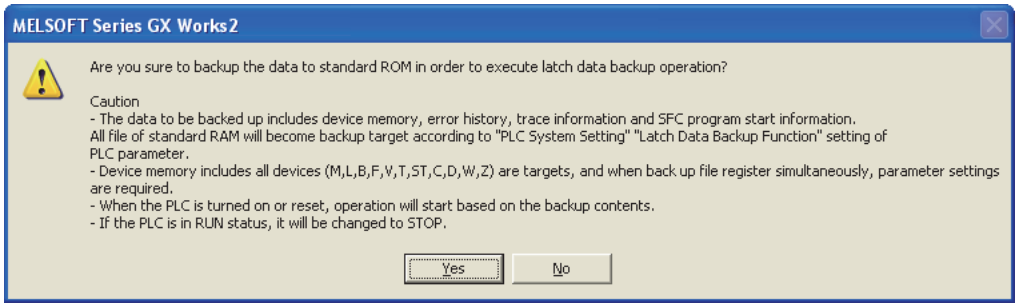
- Backup data are the data when the contact is on (END processing). To run the CPU module again after the backup operation, power on or reset the CPU module.
- Since the status of the contact is checked during execution of the END instruction, data are not backed up even if the contact is turned on → off → on, or off → on → off within one scan.
- In the following cases, data are not backed up unless the contact is turned off and then on.
  - The device X is set as a contact, the backup operation is performed by turning off and then on the contact, and the CPU module is powered off and then on or is reset without turning off the contact.
  - The device M or B is set as a contact and the backup operation is performed by turning off and then on the contact.

(3) Execution by remote operation

(a) Execution method

Open a dialog box to execute a remote operation.

[Online] ⇨ [Latch Data Backup] ⇨ [Backup]



After the backup operation is completed, the BAT. LED of the CPU module flashes (green). The module is in the standby status and is ready to be powered off. Backup data are the data at the execution of remote operation. For details, refer to the following.

(4) Restoring backup data

The backup data are automatically restored when:

- the CPU module is powered off and then on, or
- the CPU module is reset.

Whether to restore data once after backup or per above operation is set by SM676 (Specification of restore repeated execution).

Status of SM676 at backup operation	Restoration operation
SM676 is off.	Data are restored once when the CPU module is powered off and then on or is reset after backup.
SM676 is on.	Data are restored whenever the CPU module is powered off and then on or is reset after backup. Data are repeatedly restored until the backup data are deleted or the latch data are backed up next time.

After backup data are restored, the BAT.LED on the CPU module turns on (green) for five seconds.

Point

If the number of device points configured in the parameter setting and the number of device points at the time of backup are different, "RESTORE ERROR" (error code: 2220) is detected when the backup data are restored, and data restoration is not normally completed.
(Data will be restored again when the CPU module is powered off and then on or is reset in the next time.)
Perform any of the following operations to normally complete restoring data.

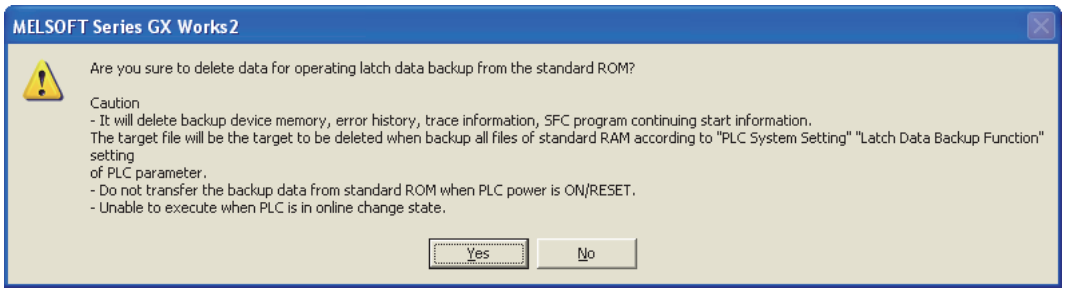
- Return the status of data when the parameters are backed up.
- Delete the backup data.
- Backup data again.

(5) Deleting backup data

The backup data can be deleted in the following screen. (Stop the CPU module before deleting the backup data. This operation cannot be performed when the CPU module is in the RUN status.)

[Online] ⇨ [Latch Data Backup] ⇨ [Delete Backup Data]

Also, information of special registers (SD671 to SD675) can be initialized (cleared to 0) by deleting the backup data.



(6) Checking the status in the special relay and special register

The execution status of the function and the restoration operation status can be checked in SM671, SM676, and SD671 to SD679. For details, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(7) Precautions

The following provides precautions for backing up latch data.

- Do not power off or reset the CPU module during backup of latch data. If performed, "RESTORE ERROR" (error code: 2221) occurs, and the backup data is not restored. (The backup data will be deleted.)
- Even if the backup data exists, the initial device value has a priority over the backup data when the initial device value is set. Therefore, the device where the initial device value setting is configured is overwritten by the device data of the initial device value after reflecting the backup data.
- Even if the latch device or latch range setting is used, backup data has a priority. Therefore, even when data of latch device and latch range setting are changed after backup, it is overwritten by data backed up when the CPU module is powered off and then on or is reset.
- Devices where local device range setting is configured are not backed up. They are initialized (cleared to 0) when the CPU module is powered off and then on or is reset.
- When the number of writes to the standard ROM exceeds 100,000 times ("FLASH ROM ERROR" (error code: 1610) is detected), data may not be normally backed up.
- Backup data cannot be deleted unless the data are deleted or the storage location memory (standard ROM) for the backup data is formatted.
- The following operations cannot be performed during latch data backup. Perform them after the backup operation. If performed, an error is displayed on the programming tool.
  - Format PLC memory (standard ROM only)
  - Latch data backup by remote operation
  - Online change (ladder mode, files, function block)
  - Program Memory Batch Download
  - Write to PLC (Flash ROM)
  - CPU module change function with memory card
  - CPU module data backup/restoration function
- When the High-speed Universal model QCPU and Universal model Process CPU are used and the "Backup all files in the internal of standard RAM" checkbox is checked in parameter, "RESTORE ERROR" (error code: 2228) occurs if the size of the standard RAM is smaller than the backup data.

3.30 Writing/Reading Device Data to/from Standard ROM

This function writes device data to the standard ROM. Writing the fixed values for operation and operation results to the standard ROM can prevent losing data due to low battery. Also, timing of writing to the standard ROM can be set by an instruction.

(1) Execution method

Device data are written to the standard ROM by the SP.DEVST instruction.
The device data written to the standard ROM is read to the specified device by the S(P).DEVLD instruction.

(2) Devices that can be written to the standard ROM

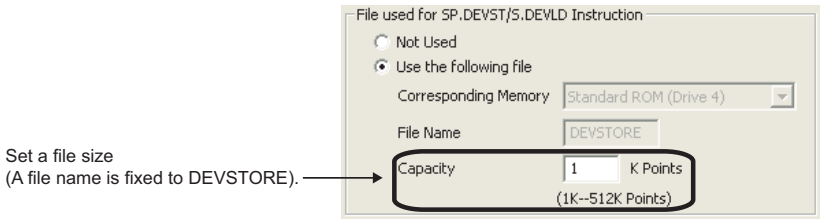
The following table lists the devices whose data can be written to the standard ROM.

Device	Storage location
X, Y, M, L, B, F, V, T, ST, C, D*1, W*2, SM, SD, SB, SW, R, ZR	Standard ROM (device data storage location)

*1 The extended data register (D) is included.
*2 The extended link register (W) is included.

(3) Setting method

The area for storing the device data is set in the standard ROM by the PLC file tab of the PLC parameter dialog box.



(a) File size setting

The capacity that can be set varies depending on the CPU module.

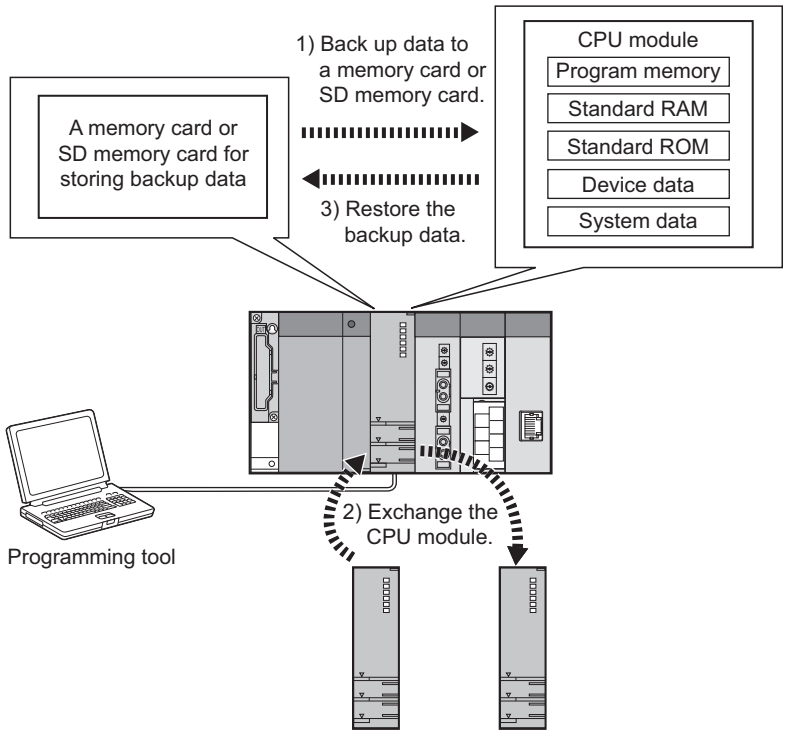
CPU module	Setting range
Q00UJCPU, Q00UCPU, Q01UCPU	1K Points (fixed)
Q02UCPU	1 to 16K Points
QnUD(H)CPU, Built-in Ethernet port QCPU	1 to 512K Points

Remark

For details of the instructions, refer to the following.
MELSEC-Q/L Programming Manual (Common Instruction)

3.31 CPU Module Change Function with Memory Card Note 3.19

This function backs up all the data (only the file register files and latch-target device data) in a CPU module to a memory card or SD memory card. The data backed up can be restored to a replaced CPU module.



This function consists of the following two functions.

- Data backup for the CPU module change function: Page 263, Section 3.31.1
- Restoration for the CPU module change function: Page 272, Section 3.31.2

Note 3.19 Universal

The Q00UJCPU, Q00UCPU, and Q01UCPU do not support this function.
Before executing the function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, or QnUDE(H)CPU, check the versions of the CPU module and programming tool used.

(Page 466, Appendix 2)

(1) Backup data file

After data are backed up, a backup data file "MEMBKUP0.QBP" is created in a memory card or SD memory card. Only one backup data file can be stored to each card. If a backup data file has existed in the card, the data in the file are overwritten whenever the backup operation is performed.[*1*2]

*1 If there is no parameter file in the CPU module where the backup data are restored, and there is a parameter file in the memory card or SD memory card, the CPU module operates using the parameter settings in the card.

*2 When data are backed up to a Flash card, only a backup data file is stored. (Other files cannot be stored.)

To delete a backup data file, perform the following operation using a programming tool.

[Online] ⇨ [Delete PLC Data]

(2) Backup data details

(a) Backup target data

The following table lists the backup target data.

Backup data (drive)	Description	Backup selection by the user
Program memory (drive 0)	All data in the program memory (drive 0)[*1]	Selectable
Standard RAM (drive 3)[*3]	All data in the standard RAM (drive 3)	
Standard ROM (drive 4)	All data in the standard ROM (drive 4)	
Device data[*2]	Internal user device (L, B, F, V, T, ST, C, D, W)	Not selectable (Data are automatically backed up by the system.)
System data	Data managed by the system (such as error history)	

*1 Data in the program cache memory are backed up.

*2 Only data in the latch relay (L) and devices to which the latch range can be set are backed up.

*3 When an extended SRAM cassette is used, data in the cassette are also backed up.

(b) Data size

The following table lists the maximum size of data to be backed up.

(Unit: K byte)

Backup target data (drive)	Q02UCPU	Q03UD/ Q03UDECPU	Q04UDH/ Q04UDEHCPU	Q06UDH/ Q06UDEHCPU	Q10UDH/ Q10UDEHCPU
Program memory (drive 0)	82	124	164	244	408
Standard RAM (drive 3)	130	194	258	770	1026
Standard ROM (drive 4)	516	1032	1032	1032	2056
Device data	128	128	128	128	128
System data	41	64	64	64	64
Total (maximum)	897	1542	1646	2238	3682

Backup target data (drive)	Q13UDH/ Q13UDEHCPU	Q20UDH/ Q20UDEHCPU	Q26UDH/ Q26UDEHCPU	Q50UDEHCPU	Q100UDEHCPU
Program memory (drive 0)	528	808	1048	2008	4008
Standard RAM (drive 3)	1026	1282	1282	1538	1794
Standard ROM (drive 4)	2056	4104	4104	8200	16392
Device data	128	128	128	128	128
System data	64	64	64	64	64
Total (maximum)	3802	6386	6626	11938	22386

Backup target data (drive)	Q03UDVCPU	Q04UDVCPU/ Q04UDPVCPU	Q06UDVCPU/ Q06UDPVCPU	Q13UDVCPU/ Q13UDPVCPU	Q26UDVCPU/ Q26UDPVCPU
Program memory (drive 0)	124	164	244	528	1048
Standard RAM (drive 3) (without an extended SRAM cassette)	204	268	780	1036	1292
With an extended SRAM cassette (1M)	1228	1292	1804	2060	2316
With an extended SRAM cassette (2M)	2252	2316	2828	3084	3340
With an extended SRAM cassette (4M)	4300	4364	4876	5132	5388
With an extended SRAM cassette (8M)	8396	8460	8972	9228	9484
Standard ROM (drive 4)	1034	1034	1034	2059	4110
Device data	130	170	170	250	250
System data	70	70	70	70	70
Total (maximum) (without an extended SRAM cassette)	1562	1706	2298	3943	6770

The backup data size can be checked by the following methods.

- On the window of a programming tool
- In SD698 and SD699[*1]

*1 The data size is checked after the backup operation starts.

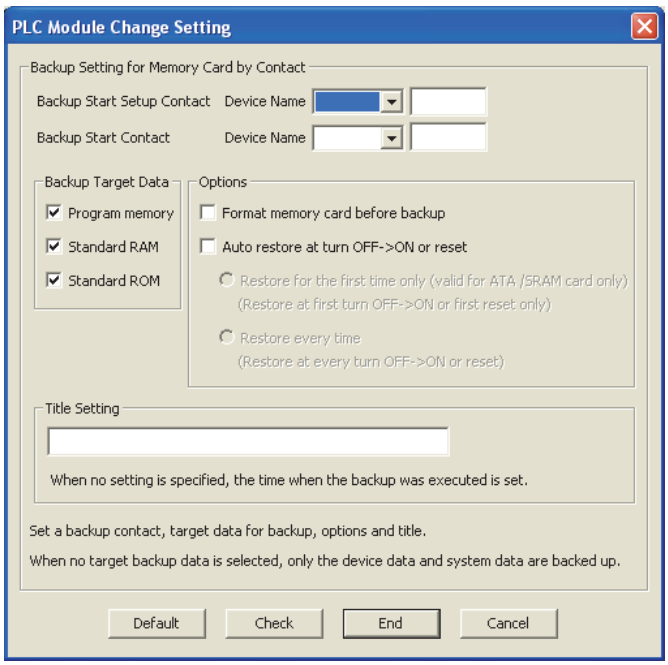
3.31.1 Data backup for the CPU module change function

This function backs up all the data (only the file register files and latch-target device data) in a CPU module to a memory card or SD memory card. If a memory card or SD memory card is used in a running system, users can stop its operation, change the card, and back up data to another card.

(1) Data backup using contacts

(a) Setting method

To back up data using contacts, turn on the devices set in the PLC Module Change Setting window, opened by clicking the button on the PLC system tab of the PLC parameter dialog box.



Item	Description	Setting range	Default
Backup Start Setup Contact*1	At the rise of the selected device among the items shown in the right column, backup is ready to start.	Available devices*2 • X (0 to 1FFF) • M (0 to 8191)*3 • B (0 to 1FFF)*3	-
Backup Start Contact	At the rise of the selected device among the items shown in the right column, backup enters execution status.		
Backup Target Data	Select data to be backed up stored in any of memories shown in right column.*4	• Program memory (drive 0) • Standard RAM (drive 3) • Standard ROM (drive 4)	All drives are backed up.
Format memory card before backup	Select whether to format a memory card before backup.	Formatted Selected/deselected	Deselected
Title Setting*5	Set a title for a backup data file stored in a memory card or SD memory card.	32 characters (16 in two-byte characters)	A title based on the time of backup is set. (Example) If data are backed up at 12 p.m. on October 1, 2008, "200810011200" is set.

*1 The CPU module enters the STOP status at the rise of the backup start setup contact, the backup start contact cannot be turned on in the sequence program.

*2 Set different devices for the backup start setup contact and backup start contact.

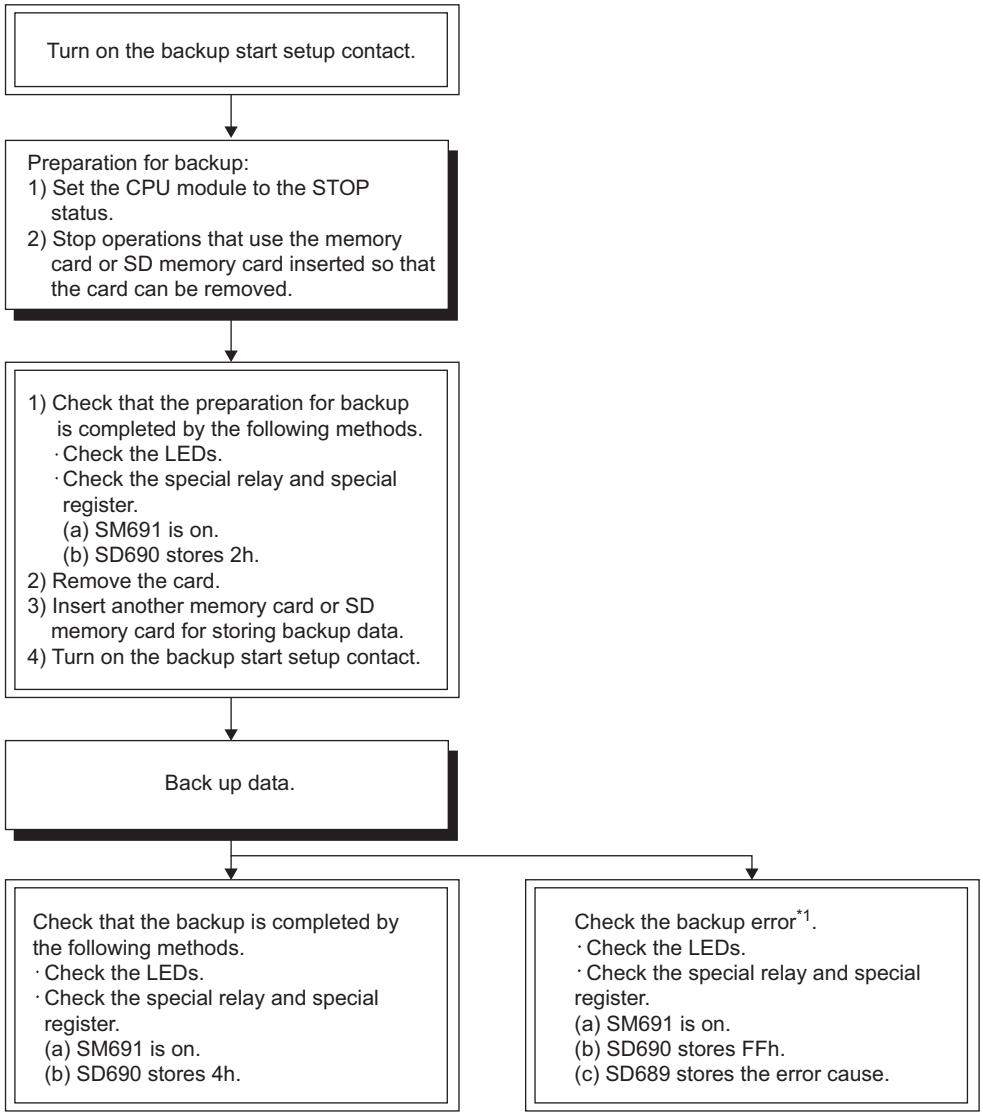
*3 These are the setting range for the default number of points. The setting range will be M (0 to 61439) and B (0 to 0EFFF) when the number of internal user device points is set to the maximum, 60K points.

*4 If no backup target data is selected, only device data and system data are backed up.

*5 A title is used to identify backup data. The title of the backup data stored in a memory card or SD memory card can be checked on the window opened by selecting [Delete PLC Data] from the menu using a programming tool.

(b) Operating procedure

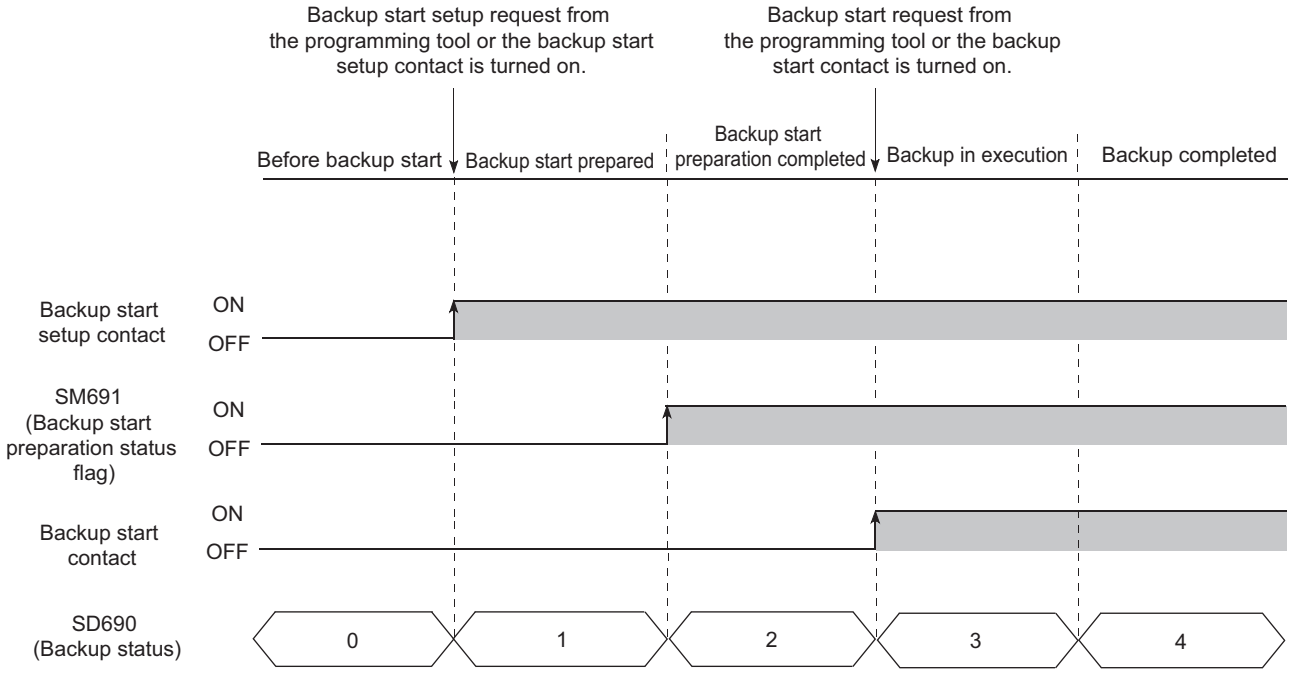
Turn on the backup start setup contact first, and then the backup start contact. Data are not backed up when only the backup start contact is on.



*1 Since the SM691 (Backup start preparation status flag) is on, data can be backed up again by turning off and then on the backup start contact.

(c) Operation of data backup using contacts

The following figure shows the operations of the backup start setup contact, backup start contact, SM691 (Backup start preparation status flag), and SD690 (Backup status).



If the backup start contact is turned on while the value in SD690 is 0_H (Before backup start) or 1_H (Backup start prepared), data are not backed up.

If the backup start contact is on before the value in SD690 becomes 2_H (Backup start preparation completed), turning off and then on the backup start contact again while the value in SD690 is 2_H (Backup start preparation completed) starts backup.

(2) Data backup using a programming tool

Data backup using a programming tool can be performed on the "Create Backup Data for PLC Module Change" window.

[Online] ⇨ [PLC Module Change] ⇨ [Create Backup Data]

For details, refer to the following.

Operating manual for the programming tool used

(3) Data backup operation details

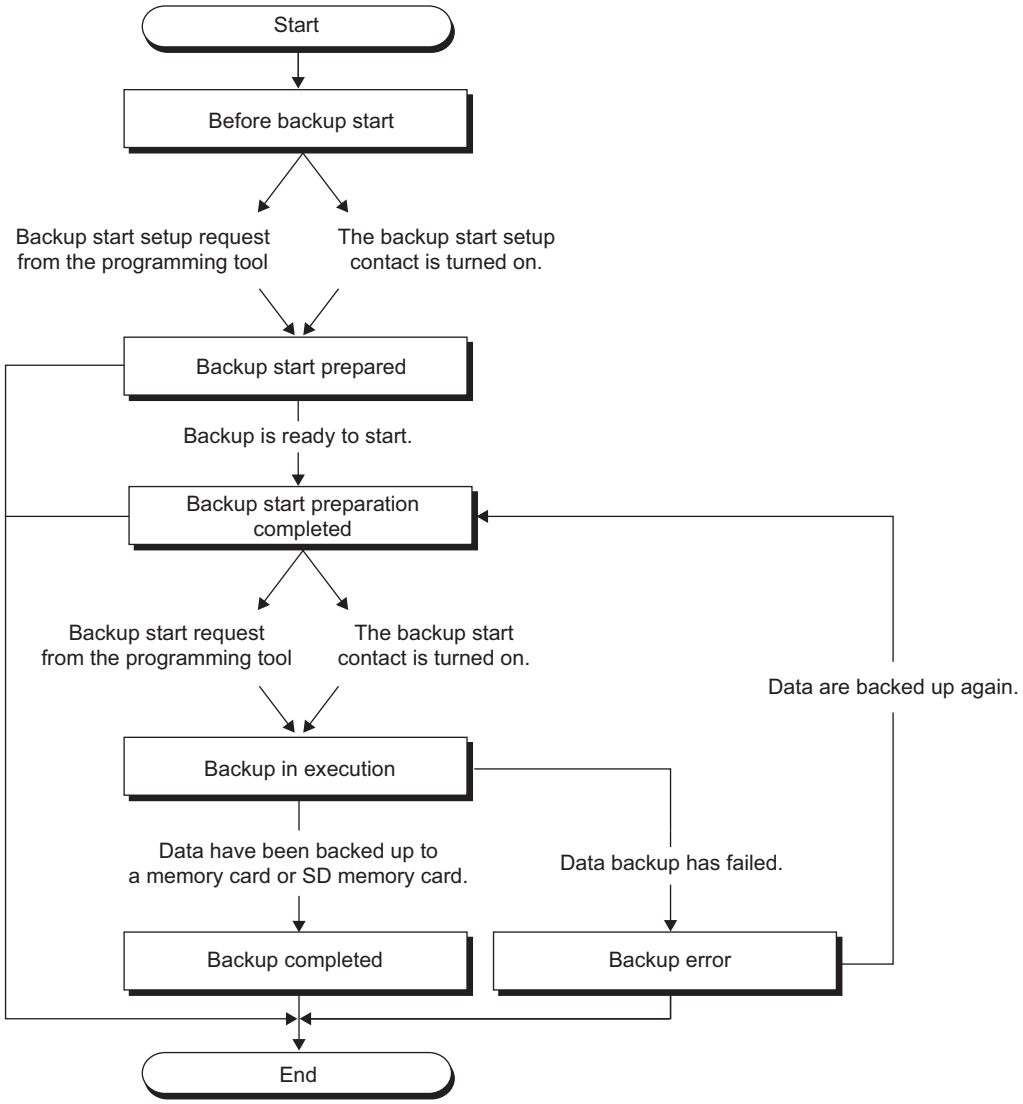
(a) Changing a memory card or SD memory card

If a memory card or SD memory card is being used in a running system, the card can be changed to another card after the status of the data backup function shifts into the ready status. (SM609 (Memory card remove/insert enable flag) does not need to be turned on.) Upon the status change, the CPU module turns off SM604 (Memory card in-use flag).

(b) Data backup operation status

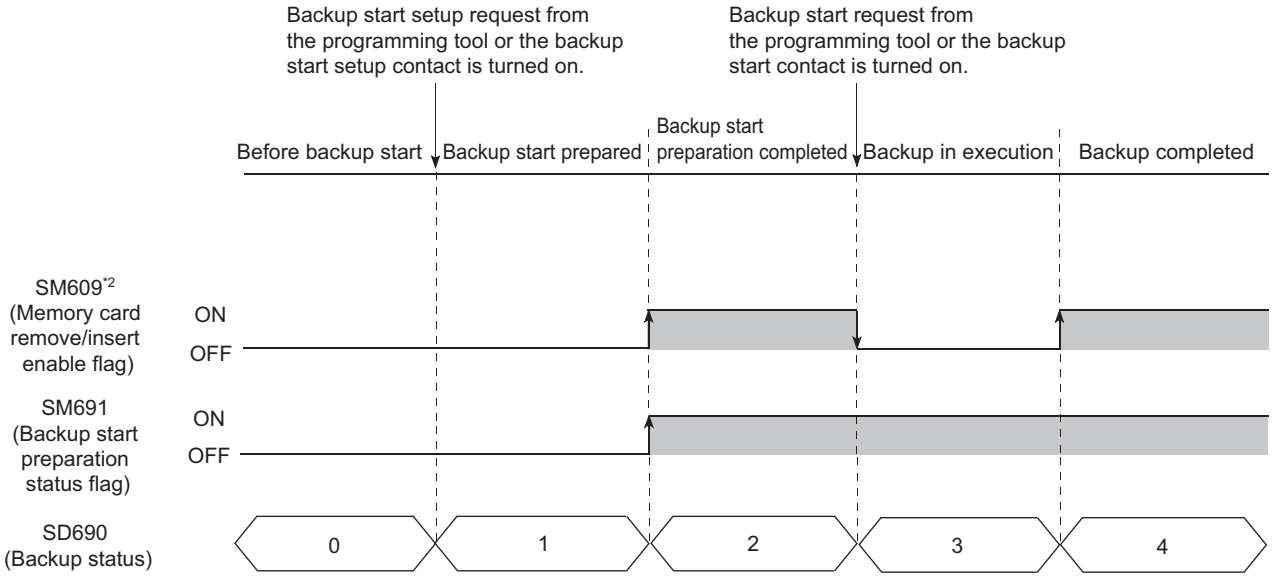
The following table lists the backup operation status.

Status	Description	Value in SD690
Before backup start	Data backup processing has not been started.	0_H
Backup start prepared	A memory card or SD memory card can be changed (removed and inserted).	1_H
Backup start preparation completed	Backup target data are set.	2_H
Backup in execution	Data are being backed up.	3_H
Backup completed	Data have been backed up normally	4_H
Backup error	Data backup has failed and an error has occurred.	FF_H



(c) Operations of the special relay and special register*1

The following figure shows the operations of SM609 (Memory card remove/insert enable flag), SM691 (Backup start preparation status flag), and SD690 (Backup status).



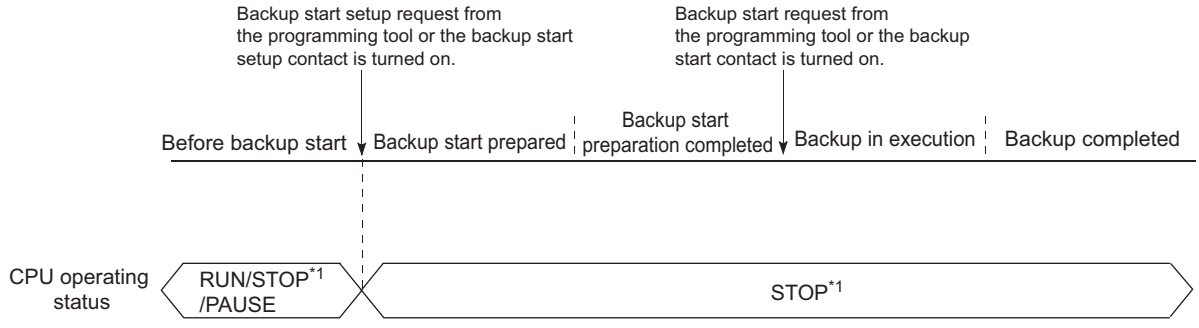
*1 For details of the special relay and special register used for this function, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

*2 This special relay is turned on/off by the system.

(d) Operating status of the CPU module

After END processing where the backup start setup request is accepted, the CPU module changes its operating status to STOP. Data can be backed up regardless of the operating status. After the data backup processing ends, power off and then on or reset the CPU module. (If only the RESET/STOP/RUN switch is moved to RUN without power-on or reset, the operating status of the CPU module remains in STOP.)



*1 The status includes a stop error.

(4) LED indication

The LEDs on the front of the CPU module indicate backup status.

Value stored in SD690	Backup status	LED indication
2_H	Backup start preparation completed	MODE: Flash (green), BAT.: Flash (green)
3_H	Backup in execution	Indication status changes as follows at 800ms intervals. 1) MODE: Flash (green), BAT.: On (green) ↓ 2) MODE: Flash (green), BAT.: On (green), USER: On (red) ↓ 3) MODE: Flash (green), USER: On (red)
4_H	Backup completed	MODE: Flash (green), BAT.: Flash (green), BOOT: Flash (green)
FF_H	Backup error	MODE: Flash (green), USER.: Flash (red), BAT.: Flash (green)

(5) Error causes

Even when backup is not completed successfully, a diagnostic error is not detected. In that case, the error cause is stored in SD689 (Backup error factor) or the error response is returned to the programming tool.

Value stored in SD689	Error response number	Error cause
100_H*1	$41FE_H$*2	• Backup started without a memory card or SD memory card being inserted. • Backup started while use of the SD memory card is disabled with the SD memory card lock switch. • Backup preparation or backup started while use of the SD memory card is disabled by SM606 (SD memory card forced disable instruction).
200_H	-	Size of backup target data exceeds the capacity of a memory card or SD memory card used. Free space of the standard RAM in a restoration-target CPU module is not enough.
300_H	-	Write protection has been set to a memory card or SD memory card used.
400_H	-	Writing data to a memory card or SD memory card did not complete normally.
500_H	-	Reading data from a target drive did not complete normally (a program memory read error).
503_H	-	Reading data from a target drive did not complete normally (a standard RAM read error).
504_H	-	Reading data from a target drive did not complete normally (a standard ROM read error).
510_H	-	Reading data from a target drive did not complete normally (a system data read error).
600_H*1	4335_H*2	Another backup preparation started while data were being backed up to the standard ROM.
601_H*1	$410A_H$*2	Backup started while data were being written to the running CPU module (during online change).
602_H*1	4336_H*2	Backup preparation started while an external device (FTP client) is communicating with the CPU module using the FTP.
-	4082_H*3, 4330_H*4	Another backup preparation or backup started while data were being backed up.
-	4333_H*2	Backup started while the CPU module was in "Before backup start" (SD690 = 0).
603_H*1	4276_H*2	Backup preparation started while the data logging function was being executed.
607_H*1	4800_H*2	Backup preparation started while the iQ Sensor Solution function (data backup/restoration) was being executed.
609_H*1	$4C1F_H$*2	Backup preparation started while the CPU module data backup/restoration function was being executed.
701_H*1	4426_H*2	Backup preparation started when a block password for which "Execution Program Protection Setting" was enabled was set.

*1 Only when data are backed up using contacts
*2 Only when data are backed up using a programming tool
*3 When data are backed up from another boot source
*4 When data are backed up from the same boot source

(6) Operations and functions that cannot be performed during backup

The following table lists the operations and functions that cannot be performed during backup.

Operation and function	
Operation using programming tool	Change TC setting
	Online change (ladder mode)
	Online change (inactive block) for SFC program
	Write to PLC (including writing data to the CPU module during RUN)
	Remote latch clear
	Password registration
	Format PLC memory
	Arrange PLC memory
	Delete PLC data
	Write/delete PLC user data
	Program Memory Batch Download
	Export to ROM Format
	Latch data backup to standard ROM
	Monitor condition setup
	Executional conditioned device test
	Sampling trace registration
Operation using CPU Module Logging Configuration Tool	Data logging function
Others	Latch clear by using the special relay and special register areas
	File transfer function (FTP) for the built-in Ethernet function
	Data backup/restoration function for the iQ Sensor Solution function
	CPU module data backup/restoration function
	Operation history function • Operation history display and data update (only during restoration) • Operation history clear (only during restoration)

(7) Precautions

- Do not perform the following operations during data backup.
  - Insertion/removal of a memory card or SD memory card
  - Power-off of the CPU module
  - Reset
- Even when parameters that are backed up using contacts are booted to the Universal model QCPU whose serial number (first five digits) is "10101" or earlier, parameters are ignored. In this case, even if the backup start setup contact or backup start contact set in parameter is turned on, the operating status of the CPU module does not change. (No diagnostic error is detected.)
- When the data backup operation is ready, the CPU module stops the operations of the following functions. The operations do not resume even after the data backup operation ends.
  - Refresh of network modules
  - Refresh of CC-Link IE Field Network Basic
  - Data link transfer
  - Auto refresh of intelligent function modules
  - Auto refresh of CPU shared memory
  - Simple PLC communication function
- Data cannot be backed up while data logging is being executed. Stop the data logging processing, and then start the data backup operation. For how to stop the data logging processing, refer to the following.

  QnUDVCPU/LCPU User's Manual (Data Logging Function)
- If the data backup processing is performed while the CPU module is locked with a security key, an error occurs.

3.31.2 Restoration for the CPU module change function

This function restores data backed up in a memory card or SD memory card to a replaced CPU module.

(1) Restoration using a programming tool

Data restoration using a programming tool can be performed on the "Restoration execution from backup data" window.

[Online] ⇨ [PLC Module Change] ⇨ [Restore]

Click "Execute" to start restoration, select "Yes" in the screen appears, and power off and on or reset the CPU module. The restored data becomes valid. For details, refer to the following.

Operating manual for the programming tool used

(2) Automatic restoration

Check the "Auto restore at turn OFF → ON or reset" checkbox in the PLC Module Change Setting window, opened by clicking the button in the PLC system tab of the PLC parameter dialog box. (Page 264, Section 3.31.1 (1) (b)) After data are backed up and the CPU module is powered off and then on or is reset, restoration automatically starts. Optional items listed in the following table can be set at the same time.

Option setting item	Powering off and then on or resetting the CPU module	
	First time	Second time and later
Restore for the first time only*1 *2	Restored	Not restored (The CPU module operates the memory card as usual.)*3
Restore every time	Restored	Restored

*1 This setting is valid only when backup data are stored in the ATA card or SRAM card.
Note if the write protect switch of the SRAM card is set to be valid (write protection), the setting does not become valid and the data are restored even after the first time. ("RESTORE ERROR" (error code: 2226) occurs.)

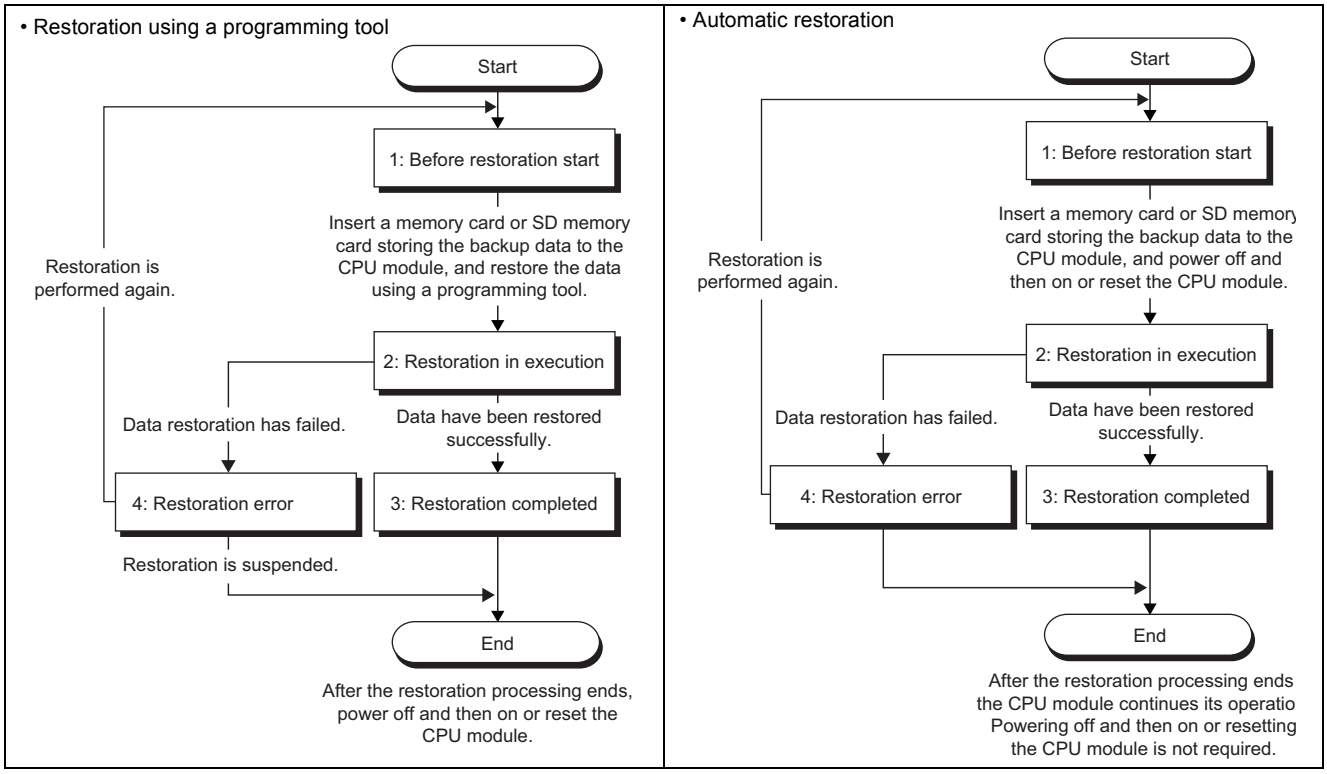
*2 When using the FLASH card, restoration can be performed even after the first time.

*3 Data are restored using a programming tool for the second time and later.

Data are restored during initial processing after the CPU module is powered off and then on or is reset. After the data are restored, the operating status of the CPU module shifts to the one set with the RUN/STOP/RESET switch. Therefore, powering off and then on or resetting the CPU module again is not required.
If a memory card or SD memory card needs to be changed after restoration, turn on SM609 (Memory card remove/insert enable flag), check that SM600 (Memory card usable flags) turns off, and then change the card.

(3) Restoration behavior of data backed up

The following figures show the restoration behavior of data backed up.

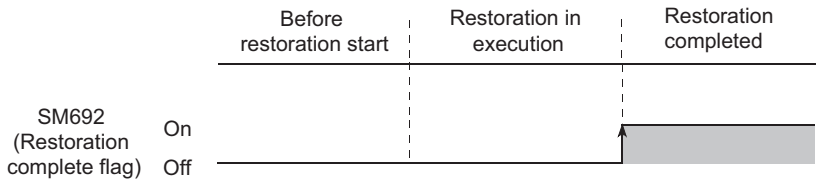


A value indicating restoration status is stored in SD693 (Restoration status).

Status	Description	Value stored in SD693
Before restoration start	Restoration processing has not been started.	0_H
Restoration in execution	Data are being restored.	1_H
Restoration completed	Data have restored normally.	2_H
Restoration error	Data restoration has failed and an error has occurred.	FF_H

(4) Operation of the special relay and special register*1

The following figure shows the operation of SM692 (Restoration complete flag).



*1 For details of the special relay and special register used for this function, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(5) LED indication

The LEDs on the front of the CPU module indicate restoration status.

Value stored in SD693	Restoration status	LED indication
0_H	Before restoration start	MODE: On (green)
1_H	Restoration in execution	Indication status changes as follows at 800ms intervals. 1) MODE: Flash (orange), BAT.: On (green) ↓ 2) MODE: Flash (orange), BAT.: On (green), USER: On (red) ↓ 3) MODE: Flash (orange), USER: On (red)
2_H	Restoration completed	• Restoration using a programming tool MODE: Flash (orange), BAT.: Flash (green), BOOT: Flash (green) • Automatic restoration MODE: On (green)
FF_H	Restoration error	• Restoration using a programming tool • MODE: Flash (orange), USER: Flash (red), BAT.: Flash (green) • Automatic restoration MODE: On (green), ERR.: Flash (red)

(6) Error causes

Even when restoration is not completed successfully, a diagnostic error is not detected. In that case, the error cause is stored in SD692 (Restoration error factor) or the error response is returned to the programming tool.

Value stored in SD692	Error response number	Error cause
800_H	-	The model of a restoration-target CPU module differs from that of a backup-source CPU module.
801_H	-	• Backup data files do not match. • Reading backup data from a memory card or SD memory card did not complete normally.
810_H	-	Writing backup data to the restoration-target drive did not complete normally.
-	4335_H*1	Restoration started while latch data were backed up to the standard ROM.
-	$410A_H$*1	Restoration started while data were being written to the running CPU module (during online change).
-	4336_H*1	Restoration started while an external device (FTP client) is communicating with the CPU module using the FTP.
-	4330_H*1	Restoration started while another restoration was being executed.
-	$41FE_H$*1	• Restoration started without a memory card or SD memory card being inserted. • Restoration started while use of the SD memory card is disabled with the SD memory card lock switch. • Restoration started while use of the SD memory card is disabled by SM606 (SD memory card forced disable instruction).
811_H	-	Free space of the standard RAM in a restoration-target CPU module is not enough.
-	4276_H*1	Restoration started while the data logging function was being executed.
-	4800_H*1	Restoration started while the iQ Sensor Solution function (data backup/restoration) was being executed.
-	$4C1F_H$*1	Restoration started while the CPU module data backup/restoration function was being executed.

*1 Only when data are restored using a programming tool

When automatic restoration did not complete normally, "RESTORE ERROR" (error code: 2228) occurs.

Error code	Error message	Error cause
2225	RESTORE ERROR	The model of a restoration-target CPU module differs from that of a backup-source CPU module.
2226		• Backup data file is corrupted. (The contents of backup data file do not match with the check code.) • Reading backup data from a memory card or SD memory card did not complete normally. • Since the write protect switch of an SRAM card or SD memory card is valid, the "Restore for the first time only" parameter cannot be set.
2227		Writing backup data to the restoration-target drive did not complete normally.
2228		Free space of the standard RAM in a restoration-target CPU module is not enough.

(7) Functions that cannot be executed during restoration

Functions that cannot be executed during restoration are the same as those cannot be executed during data backup. (☞ Page 270, Section 3.31.1 (6))

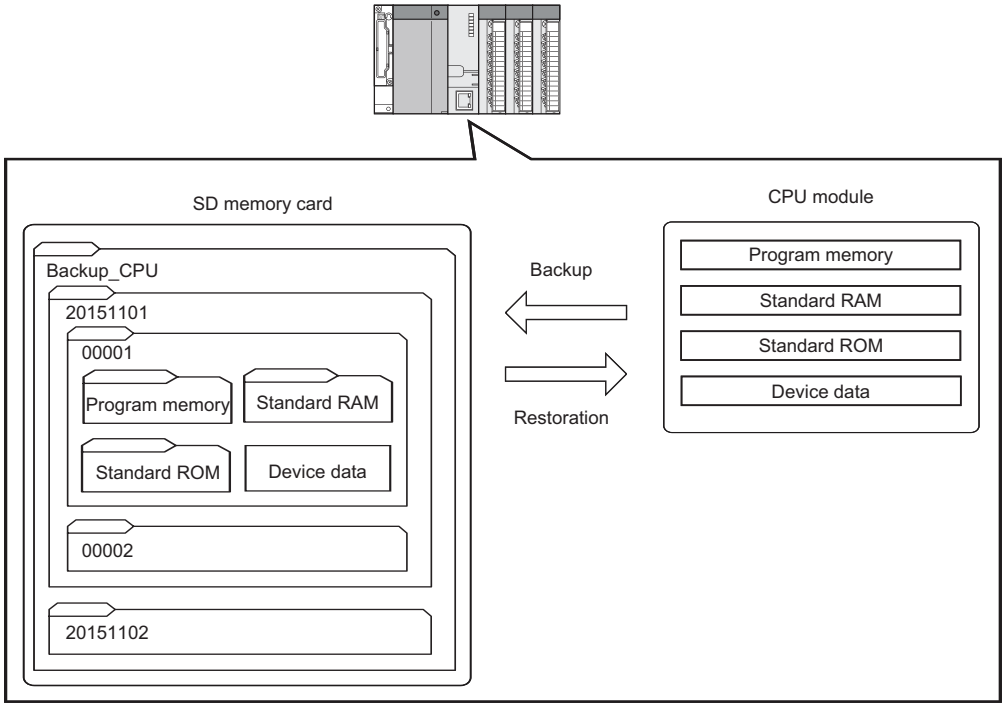
(8) Precautions

- Do not perform the following operations during restoration.
  - Insertion/removal of a memory card or SD memory card
  - Power-off of the CPU module
  - Reset
- Even when a memory card storing backup data files is inserted to the Universal model QCPU whose serial number (first five digits) is "10101" or earlier, and the CPU module is powered off and on or is reset, the system ignores the files. (No diagnostic error is detected.)
- Do not use a memory card or SD memory card where a parameter file with boot settings is stored. If used, data are overwritten according to the boot settings even data are restored.
- If restoration is started using a programming tool, the CPU module stops the operations of the following functions. The operations do not resume even after data are restored.
  - Refresh of network modules
  - Refresh of CC-Link IE Field Network Basic
  - Data link transfer
  - Auto refresh of intelligent function modules
  - Auto refresh of CPU shared memory
  - Simple PLC communication function
- If the model of a restoration-target CPU module differs from that of a backup-source CPU module, "RESTORE ERROR" (error code: 2225) occurs.
- Check that the free space of the standard RAM in a restoration-target CPU module is larger than backup data. If a backup-source CPU module uses an extended SRAM cassette, "RESTORE ERROR" (error code: 2228) occurs in the following cases.
  - Even though a backup-source CPU module uses an extended SRAM cassette, a restoration-target CPU module is not using it.
  - The size of the extended SRAM cassette used with a restoration-target CPU module is smaller than that of the extended SRAM cassette used with a backup-source CPU module.
- If the restoration processing is performed while the CPU module is locked with a security key, an error occurs.

3.32 CPU Module Data Backup/restoration Function Note 3.20

This function backs up data such as program files, a parameter file, and device data*1 including the file register in a CPU module to an SD memory card. The data backed up can be restored as necessary.

*1 Except for devices and buffer memory in the intelligent function module



The following table lists the backup and restoration methods of the CPU module data backup/restoration function.

Function		Reference
Backup function	Backup by turning on SM1926	Page 284, Section 3.32.1 (2)
	Automatic backup using SD910	Page 285, Section 3.32.1 (3)
Restoration function	Restoration by turning on SM1929	Page 292, Section 3.32.2 (1)
	Automatic restoration using SD918	Page 293, Section 3.32.2 (2)

Point

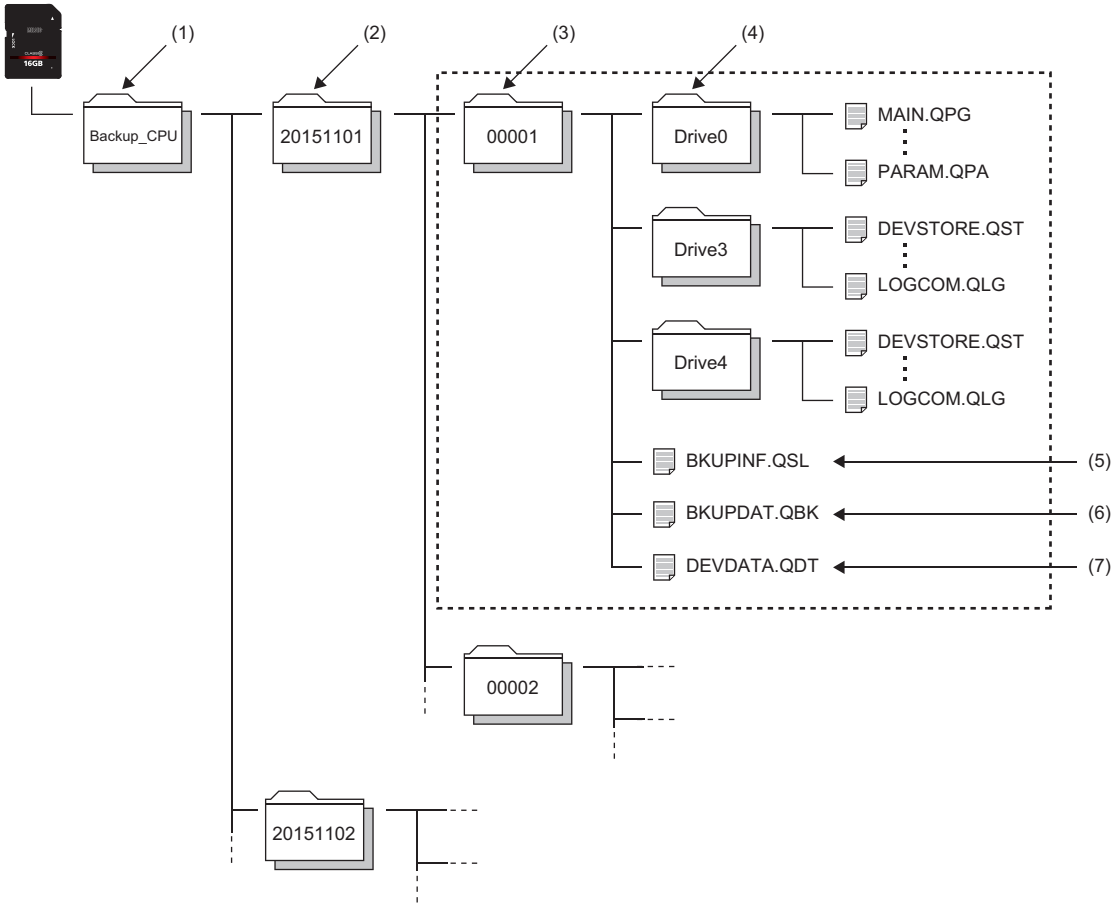
The restoration function modifies programs, parameters, or device data in the CPU module. After restoration, check the restored data carefully before an actual operation. (Check the data with an engineering tool or a device monitor.)

Note 3.20 Universal

Only the High-speed Universal model QCPU and Universal model Process CPU support this function. When using the function, check the version of CPU module used. (Page 466, Appendix 2)

(1) Backup data

The data backed up is saved in an SD memory card. The following shows the folder structure of the backup data.



No.	Folder type	Folder name	Storable number of folders	Description
(1)	Backup data folder	Backup CPU (fixed)	One	A folder which stores whole backup data.
(2)	Date stamp folder	Automatically determined[*1] Folder name format: YYYYMMDD YYYY: Year the data backed up (4 digits) MM: Month the data backed up (2 digits) DD: Day the data backed up (2 digits)	Depends on the capacity of the SD memory card used[*2], or 1 to 100 folders[*4]	Folders which store backup data by date. As for the upper limit value setting for the number of backup data, the number of backup data represents the number of date stamp folders.(☞ Page 282, Section 3.32.1 (1))
(3)	Numbered folder	Automatically determined[*1] Folder name: Sequentially numbered from 00001 to 32767 (5 digits)	Depends on the capacity of the SD memory card used[*2]	Folders which store backup data in units of 1 backup data. Each backup data made on the same date is stored by sequentially numbered folders.
(4)	Drive folder	Drive 0 (fixed), Drive 3 (fixed), Drive 4 (fixed)	One each in a numbered folder	Folders which store files stored in each drive of a backup source CPU module by each drive.
(5)	System information file for CPU module data backup/restoration	BKUPINF.QSL	One in a numbered folder	Files which store the information required at a time of restoration, such as a list of the data to be backed up and identification information of the CPU module.
(6)	System data file for CPU module data backup/restoration	BKUPDAT.QBK	One in a numbered folder	Files which store the following data. • Data regarding operations of sampling trace setting at a time of a backup • Data regarding operations of data logging setting • Data stored to a flash ROM with the IP address change function • Error history of a CPU module and intelligent function module.
(7)	Device data file for CPU module data backup/restoration	DEVDATA.QDT	One in a numbered folder	Files which store device data at a time of a backup.[*3]

*1 Folder names of date stamp folders and numbered folders are automatically determined by the CPU module.

*2 The number of storable folders is up to 32767.

*3 The file register (R, ZR), the extended data register (D), and the extended link register (W) are backed up as the file register file. Also, the local device is backed up as the local device file.

*4 For the modules whose serial number (first five digits) is “18052” or later, the number of folders can be set with SD1928 (Upper limit value setting for the number of backup data). (☞ Page 282, Section 3.32.1 (1))

(2) Target data for backup and restoration

The backup target data is all the target data in a CPU module. (☞ Page 279, Section 3.32 (2) (b))

The restoration target data is set with SD917 (Restoration target data setting). (☞ Page 290, Section 3.32.2)

(a) Target drives for backup and restoration

Target drives are Drive 0 (Program memory), Drive 3 (Standard RAM), and Drive 4 (Standard ROM).

Drive 2 (SD memory card) cannot be backed up or restored.

(b) Target files for backup and restoration

The following lists the target files for backup and restoration.

○: Available, ×: Not available

File type	File name and extension	Availability of backup/restoration
Parameter	PARAM.QPA	○
Intelligent function module parameter	IPARAM.QPA	○
Program*1	***.QPG	○
Device comment	***.QCD	○
Initial device value	***.QDI	○
File register	***.QDR	○
Local device	***.QDL	○
Sampling trace file	***.QTD	○
Device data storage file	DEVSTORE.QST	○
Module error collection file	IERRLOG.QIE	○
Boot setting file	AUTOEXEC.QBT	○
Remote password	00000000.QTM	○
Latch data backup file	LCHDAT00.QBP	○
Backup data file	MEMBKUP0.QBP	×
Data logging setting file	LOGCOM.QLG, LOG01.QLG to LOG10.QLG	○
Data logging file	***.CSV	×
PLC user data	***.CSV/BIN	○
Symbolic information	***.C32	○
Drive heading	QN.DAT	○
System file for the iQ Sensor Solution function (data backup/restoration)	SSBRINF.QSI	×
Backup data file for the iQ Sensor Solution function (data backup/restoration)	***.QBR	×
Predefined protocol setting file	ECPRTCL.QPT	○
Operation history file	OPERATE.QOL	○

*1 The backup function cannot be executed when a block password for which "Execution Program Protection Setting" is enabled has been set.

(c) Applicable number of backup and restoration

Applicable number of backup and restoration is 32767, which is the same as the maximum number of folders.

The number of files can be backed up or restored depends on the maximum number of files of each model and drive.

(d) Target device data for bakup and restoration

The following lists target device data.

○: Available, ×: Not available

Category	Device name	Backup	Restoration*3
Internal user device	Input (X)	○	○
	Output (Y)	○	○
	Internal relay (M)	○	○
	Latch relay (L)	○	○
	Annunciator (F)	○	○
	Edge relay (V)	○	○
	Step relay (S)	○	○
	Link relay (B)	○	○
	Link special relay (SB)	○	○
	Timer (T)	○	○
	Retentive timer (ST)	○	○
	Counter (C)	○	○
	Data register (D)	○	○
	Link register (W)	○	○
	Link special register (SW)	○	○
Internal system device	Function input (FX)	○	×
	Function output (FY)	○	×
	Special relay (SM)	○	○*1*2
	Function register (FD)	○	×
	Special register (SD)	○	○*1*2
Link direct device	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)	×	×
Module access device	Intelligent function module device (U□\G□)	×	×
	Cyclic transmission area device (U3E□\G□)	○	○*1
Index register or standard device register	Index register (Z) or standard device register (Z)	○	○
File register	File register (R, ZR)	○	○
Extended data register	Extended data register (D)	○	○
Extended link register	Extended link register (W)	○	○
Nesting	Nesting (N)	×	×
Pointer	Pointer (P)	×	×
	Interrupt pointer (I)	×	×
Others	SFC block device (BL)	○	×
	Network No. specification device (J)	×	×
	I/O No. specification device (U)	×	×
	Macro instruction argument device (VD)	×	×

*1 The area used by the system may be overwritten after a restoration.
*2 Whether to be restored the device can be set with SD918 (Restoration function setting).
*3 Device data restored may be overwritten by the I/O refresh according to modules mounted on and refresh settings.

(3) Progression status of backup and restoration

Progression status of backup and restoration can be checked by SD1925(Number of backup/restoration uncompleted files) or SD1926(Backup/restoration progression status). However, the progression status of automatic restoration cannot be checked by the special register. For the operating status of automatic restoration, check the LED on the front side of the CPU module. (☞ Page 293, Section 3.32.2 (2) (c))

Special register	Description
SD1925	Displays the number of remaining files of backup and restoration. • When a backup/restoration is started, the total number of backup/restoration files are stored. • When a backup/restoration is completed, 0 is stored.
SD1926	The progression status of backup/restoration is displayed with 0 to 100%.

3.32.1 Backup function

This function backs up data such as program files, a parameter file, and device data including the file register in a CPU module to an SD memory card.

Point

The backup function is performed even while the CPU module is in RUN state.

When executing the backup function during RUN, do not make the device data change during the execution. (☞ Page 287, Section 3.32.1 (5) (c))

(1) Upper limit value setting for the number of backup data

The upper limit value of the number of backup data and the operation when the number of backup data reaches the upper limit can be set[*1] when a backup function has not been executed yet. (No backup data folder (Backup_CPU) exists in the SD memory card used.) As for the upper limit value setting, the number of backup data represents the number of date stamp folders.

To enable the upper limit value of the backup data and the operation when the number of backup data reaches the upper limit, specify the value in SD1928 (Upper limit value setting for the number of backup data) and SM923 (Upper limit setting flag for the number of backup data), and then turn on bit5 (Upper limit status setting for the number of backup data) of SD910. The value specified can be checked in SD923 (Upper limit value status for the number backup data). Turning off bit5 of SD910 disables the upper limit value setting.

Special relay/special register	Description
SM923	Specify the operation when the number of backup data reaches the upper limit value with this relay. (This relay is valid only when bit5 of SD910 is on.) • Off: After the oldest date stamp folder is deleted, the backup processing is continued. • On: The backup processing is not continued if the number of backup data exceeds the upper limit value. (if the processing is continued, the backup will complete with an error.)
Bit5 of SD910	Set whether to enable or disable the upper limit value setting for the number of backup data with this bit. • Off: Disable (No upper limit (As many date stamp folders are created as the capacity of an SD memory card.)) • On: Enable
SD923	This register indicates the value (1 to 100) that is set to SD1928. If bit5 of SD910 is off, 0 is stored.
SD1928	Set the upper limit value (1 to 100) of the number of backup data with this register.

*1 For the modules whose serial number (first five digits) is “18052” or earlier, this setting cannot be used. (The upper limit value of the number of backup data is the maximum capacity of an SD memory card.)

Point

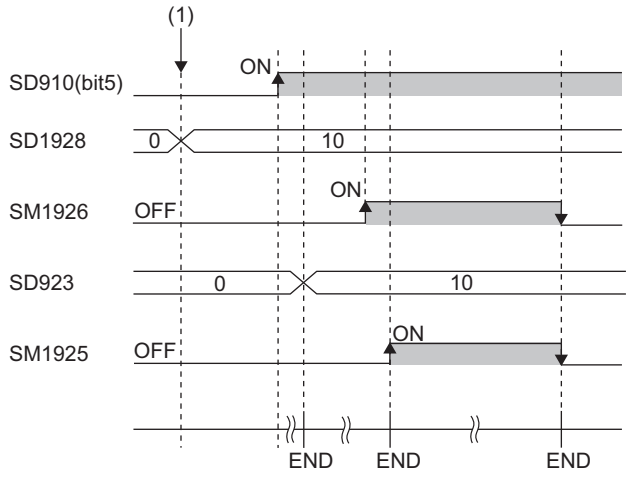
Regardless of the upper limit value set, the backup will complete with an error if the number of backup data exceeds the capacity of an SD memory card.

(a) Status of special relay and special register

The following figure shows the status of the special relay and special register when the upper limit value of the number of backup data is set.

The CPU module checks the following at the timing when bit5 (Upper limit status setting for the number of backup data) of SD910 is turned on, and enables the upper limit value of the number of bakup data.

- A backup function has not been executed yet. (No backup data folder (Backup_CPU) exists in the SD memory card used.)
- The value set to SD1928 (Upper limit value setting for the number of backup data) is within the specified range (1 to 100).



(1) The upper limit value of the number of backup data is set (0 to 10).

(2) Backup by turning on SM1926

This relay backs up data of a CPU module in a desired timing.

(a) Operating procedure

Back up data by turning on SM1926.

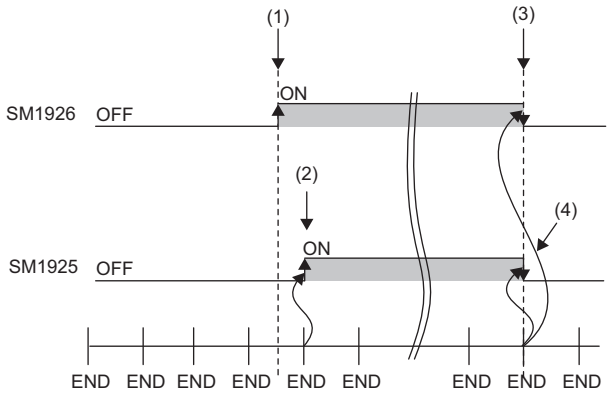
1. **To specify the upper limit value of the number of backup data, set the value with the following procedure**

- Set SD1928 (Upper limit value setting for the number of backup data).
- Set SM923 (Upper limit setting flag for the number of backup data).
- Turn on bit5 of SD910 (Backup function setting).

For the upper limit value of the number of backup data, refer to the following.

☞ Page 282, Section 3.32.1 (1)

2. **Turn off and on SM1926 (Backup execution request).**



(1) Backup execution requested
(2) Turning on SM1925 (Backup execution status flag) by the system
(3) Turning off SM1926 after completion of backup by the system
(4) Turning off SM1925 by the system

If a backup is completed with an error and SM916 (Backup error check flag) turns on, check SD916 (Cause of error occurred during backup), take corrective action, and then backup again as necessary.

Point

- Progression status of backup can be checked in SD1925 (Number of backup/restoration uncompleted files) and SD1926 (Backup/restoration progression status). (☞ Page 281, Section 3.32 (3))

(3) Automatic backup using SD910

The data of CPU module is automatically backed up by an execution timing which is set in advance.
The execution timing of the automatic backup is set with SD910 (Backup function setting). Multiple execution timing can be set simultaneously.

Bit pattern of SD910	Execution timing
Bit0: On	At the time set to SD913 on the day set to SD912
Bit1: On	At the time set to SD914 on the day of the week set to SD915
Bit15: On	When a CPU stop error has occurred

Point

Since the special register set for the automatic backup is in latch area, the set data are hold.

(a) Retry of the automatic backup

Set whether to retry the automatic backup in the case the automatic backup is executed while any of the exclusive functions (refer to Page 289, Section 3.32.1 (5) (i)) are being executed.*1
The retry is executed every three minutes and repeated 10 times.

Special relay/special register	Description
SM924	This relay turns on when the retry of the automatic backup is failed even after the specified number of retries are attempted. (This relay turns off at the start of the automatic backup. (This relay does not turn off when SM1926 (Backup execution request) is on.)) • Off: Retry not executed/Retry being executed • On: Retry failed
SM1931	This relay turns on while the retry is being executed. (This relay turns on at the start of the retry, and turns off when the automatic backup is triggered by the retry while the exclusive functions are not executed, or when the specified number of retries are attempted.) • Off: Retry not performed • On: Retry being performed
Bit10 of SD910	Set whether to retry the automatic backup with this bit. • Off: No retry (The automatic backup will be completed with an error without a retry.) • On: Retry

*1 Modules whose serial number (first five digits) is “18052” or earlier cannot use this setting. (The automatic backup cannot be executed while any of the exclusive functions are executed.)

To reflect the setting of the following special relay and special register areas, which are set before starting backup, in backup operation, turn off and on bit0, bit1, or bit15 of SD910.
When the following special relay and special register areas are changed while a backup is being executed, value in bit0 of SD910 is reflected next time a backup is executed.

- Bit0 of SD910 (When bit0, bit1, or bit15 of SD910 is turned off and on)
- SD912 (When bit0 of SD910 is turned off and on)
- SD913 (When bit0 of SD910 is turned off and on)
- SD914 (When bit1 of SD910 is turned off and on)
- SD915 (When bit1 of SD910 is turned off and on)

(b) Operating procedure (specifying day and time)

The data is automatically backed up at the specified day and time.

1. **To specify the upper limit value of the number of backup data, set the upper limit value setting. (The setting method and operating procedure are the same as the backup by turning on SM1926.)**
   ☞ **Page 284, Section 3.32.1 (2) (a))**
2. **Set the day and time with SD912 and SD913.**

Special register	Description
SD912	Set the day when the data is automatically backed up in BCD.
SD913	Set the time (hour and minute) when the data is automatically backed up in BCD.

3. **To retry the automatic backup, turn on bit10 of SD910 (Backup function setting). For the retry of the automatic backup, refer to the following.**
   ☞ **Page 285, Section 3.32.1 (3) (a)**
4. **Turn on bit0 of SD910.**

If a backup is completed with an error and SM916 (Backup error check flag) turns on, check SD916 (Cause of error occurred during backup), take corrective action, and then backup again as necessary.

Remark

In months that does not have the day specified with SD912 (Day and time setting for automatic backup [day]), no error occurs and automatic backup is not executed. For example, if SD912 is "31", the months when the automatic backup is executed shall be January, March, May, July, August, October, and December.

(c) Operating procedure (specifying time and days of the week)

The data is automatically backed up at the specified time and days of the week.

1. **To specify the upper limit value of the number of backup data, set the upper limit value setting. (The setting method and operating procedure are the same as the backup by turning on SM1926.)**
   ☞ **Page 284, Section 3.32.1 (2) (a))**
2. **Set the time and days of the week with SD914 and SD915.**

Special register	Description
SD914	Set the time (hour and minute) when the data is automatically backed up by BCD.
SD915	Set the days of the week when the data is automatically backed up. b0: Sunday, b1: Monday, b2: Tuesday, b3: Wednesday, b4: Thursday, b5: Friday, b6: Saturday

3. **To retry the automatic backup, turn on bit10 of SD910 (Backup function setting). For the retry of the automatic backup, refer to the following.**
   ☞ **Page 285, Section 3.32.1 (3) (a)**
4. **Turn on bit1 of SD910.**

If a backup is completed with an error and SM916 (Backup error check flag) turns on, check SD916 (Cause of error occurred during backup), take corrective action, and then backup again as necessary.

(d) Operating procedure (when a CPU stop error has occurred)

The data is automatically backed up when a CPU stop error has occurred.

1. **To specify the upper limit value of the number of backup data, set the upper limit value setting. (The setting method and operating procedure are the same as the backup by turning on SM1926.)**
   Page 284, Section 3.32.1 (2) (a))
2. **To retry the automatic backup, turn on bit10 of SD910 (Backup function setting). For the retry of the automatic backup, refer to the following.**
   Page 285, Section 3.32.1 (3) (a)
3. **Turn on bit15 of SD910.**

Remark

The automatic backup may not be able to be performed when a major error has occurred.

(4) Checking errors

Even when an error has occurred, a diagnostic error is not detected. In that case, the error code is stored in SD916 (Cause of error occurred during backup). (QCPU User's Manual (Hardware Design, Maintenance and Inspection))

(5) Precautions

The following describes precautions for the backup function.

(a) Removal/insertion of the SD memory card and power-off/reset of the CPU module during the backup

Do not perform the following operations during the backup operation.

- Removal and insertion of the SD memory card
- Power-off or reset of the CPU module

If performed, the backup data in the SD memory card is left in an incomplete state, which is middle of the backup processing. Do not use these data for a restoration. If these data are used, the restoration is completed with an error.

(b) Suspending processing of the backup

The following operations can suspend the processing of the backup.

- To disable the SD memory card operation forcibly
- To enable removal and insertion of the SD memory card

If the backup processing is suspended, the backup data in the SD memory card is left in an incomplete state, which is middle of the backup processing. Do not use these data for a restoration. If these data are used, the restoration is completed with an error.

(c) Changing device data

Do not make the device data change during the execution of backup. Since the device data is backed up separately into multiple scans, the data inconsistency is occurred if the device data is changed.

(d) Changing backup target data

Do not change the backup target data in the CPU module during backup. If the target data are changed, the change is not reflected.

(e) Required time for backup

According to the data size or number of files stored in a CPU module, a backup may take time.

(f) The Special relay and special register which request function operations

Before executing backup, turn off the special relay and special register which request operating of the functions such as SM801 (Trace start). If backup is executed while they are ON, the corresponding requests may be turned on and the functions are executed when the data of the special relay and special register are restored.

(g) Data protected by security functions

Data protected by the following security functions cannot be backup up.

- File password 32
  If a file with a file password is in the backup target data, the backup cannot be performed.
- File access control by security key
  If the CPU module is locked, the backup cannot be performed.

(h) When a backup is executed with the upper limit value setting for the number of backup data being valid

Check if there is enough free space in the SD memory card used before start of backup.

(i) Operations and functions which cannot be performed

The following lists the operations and functions which cannot be performed simultaneously during backup/restoration.

Operation and function	
Operation using programming tool	Change TC setting
	Online change (ladder mode)
	Online change (inactive block) for SFC program
	Write to PLC (including Write to PLC (during RUN))
	Write title
	Remote latch clear
	Password/Keyword • New (registration/change) • Delete • Disable
	Locking CPU module with security key (file access control by security key)
	Format PLC memory
	Clear PLC memory (Clear all file registers)
	Arrange PLC memory
	Delete PLC data
	Write/delete PLC user data
	Program Memory Batch Download
	Latch data backup to standard ROM
	CPU module change function with memory card
	Monitor condition setup
	Executional conditioned device test
	Sampling trace function • Start Trace • Register Trace • Write to PLC
	Clearing module error history
	Writing protocol setting data to the CPU module (predefined protocol support function)
Operation using CPU Module Logging Configuration Tool	Data logging function • Deleting/writing the data logging setting • Stopping data logging operation • Deleting data logging file(s)
Others	Latch clear by using the special relay and special register areas
	Writing or deleting files using FTP or MC protocol
	File transfer function (FTP) for the built-in Ethernet function
	IP address change function of built-in Ethernet function
	Data backup/restoration function for the iQ Sensor Solution function
	Operation history function • Operation history display and data update (only during restoration) • Operation history clear (only during restoration)

(j) Backup execution during backup processing

During a backup processing, the backup by turning on SM1926 or the automatic backup cannot be executed. (Those execution will be ignored.)

3.32.2 Restoration function

This function restores data backed up to an SD memory card as necessary.

(a) Restoration target folders

Set a folder to be restored from backup data in an SD memory card with SD919 to SD921. The latest backup data can be restored with bit13 of SD918.

Special register	Description
Bit13 of SD918	Sets the function setting of restoration with bit patterns. Off: Restores the specified data in the restoration target folders. On: Restores the latest data.*1
SD919, SD920	Specifies the date of the folder to be restored in BCD. SD919: Year, SD920: Month and day
SD921	Specifies the numbered folder (00001 to 32767) to be restored.

*1 The latest data indicates the backup data in the date stamp folder which is the closest to the present time, and has the rearmost serial number.

(b) Restoration target data

Set the restoration target data with SD917.

Value of SD917	Restoration target data setting
0	All target data
1	Device data only
2	All target data except for device data

(c) Restoration of the special relay and special register

Set whether to restore the special relay and special register with bit14 of SD918.

Bit14 of SD918	Description
On	Restores the special relay and special register.
Off	Does not restore the special relays and special registers.

Note that the following special relay areas and special register areas, which are used for the CPU module backup/restoration function, are not restored even if bit14 of SD918 is on.

- SM916, SM922, SM924, SM1925, SM1926, SM1928, SM1929, SM1931, SD916, SD922, SD1925, SD1926

(d) Operation setting after restoration

At a time of restoration, whether to operate a CPU module with the state at backup or to operate with the initial status can be set with bit15 of SD918 (Restoration function setting). If value of SD917 (Restoration target data setting) is set to 1 (restoration target data is device data only), this setting is invalid since the device initial value file or the module error collection file are not restored.

Item	Operation setting after restoration	
	Operate with the state at backup (b15 of SD918 = On)	Operate with the initial status (b15 of SD918 = Off)
Device initial value	The device initial value is not set after restoration.	The device initial value is set after restoration.*1 (The device data at backup is overwritten with the device initial value.)
Module error collection	The error history at backup is restored.	The error history at backup is not restored.

*1 The setting is valid for the automatic restoration. As for the restoration by turning SM1929 on, the device initial value is set when the CPU module is powered off and on or is switched STOP to RUN.

Point

In the operation setting after restoration, the operation at the completion of the restoration is specified. Therefore, when the CPU module is switched from STOP to RUN, the device value changes depending on the operating specifications of the device memory at operating status change of the CPU module. (Page 72, Section 2.5 (4))

(1) Restoration by turning on SM1929

The backup data is restored in an optional timing.

Point

Use the restoration by turning on SM1929 for checking the backup data and operation check before an actual operation. To operate the system using the backup data, use the automatic restoration using SD918.

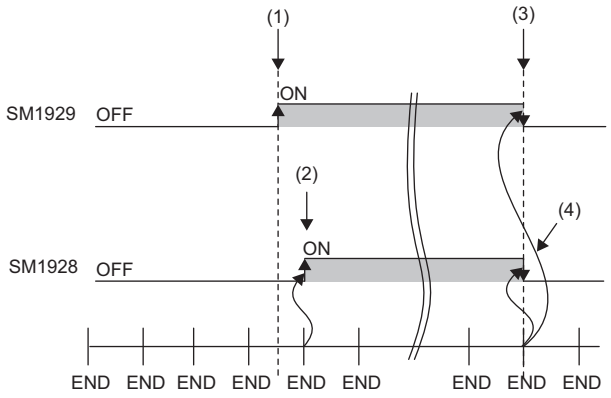
(Page 293, Section 3.32.2 (2))

Remark

The restoration by turning on SM1929 can be executed only when the operating status of the CPU module is STOP.

(a) Operating procedure

1. **Switch the operating status of the CPU module to STOP.**
2. **Set the data to be restored with SD917 (Restoration target data setting).**
3. **Set the restoration target folder with SD919, SD920, and SD921.**
   (Setting is not required when bit13 of SD918 = ON.)
4. **Set the each setting with bit13 to 15 of SD918 (Restoration function setting).**
5. **Turn off and then on SM1929 (Restoration execution request).**



(1) Restoration execution requested
(2) Turning on SM1928 (Restoration execution status flag) by the system
(3) Turning off SM1929 after completion of restoration by the system
(4) Turning off SM1928 by the system

If a restoration is completed with an error and SM922 (Restoration error check flag) turns on, check SD922 (Cause of error occurred at restoration), take corrective action, and then restore the data again as necessary.

Point

Progression status of restoration can be checked in SD1925 (Number of backup/restoration uncompleted files) and SD1926 (Backup/restoration progression status). (Page 281, Section 3.32 (3))

(2) Automatic restoration using SD918

The backup data is automatically restored when the CPU module is powered off and then on or is reset.

(a) Formatting at automatic restoration

At the execution of automatic restoration, set whether to format drives except for the SD memory card with bit1 of SD918 (Restoration function setting). This setting is valid only when the value of SD917 (Restoration target data setting) is 0 (all target data).

Bit1 of SD918	Description
Off	No format
On	Format

(b) Operating procedure

1. **Set the data to be restored with SD917 (Restoration target data setting).**
2. **Set the restoration target folder with SD919, SD920, and SD921.**
   (Setting is not required when bit13 of SD918 = ON.)
3. **Set each setting with bit1 and bit13 to 15 of SD918 (Restoration function setting).**
4. **Turn on bit0 of SD918 (Restoration function setting).**
5. **Power off and then on or reset the CPU module.**

If a restoration is completed with an error and SM922 (Restoration error check flag) turns on, check SD922 (Cause of error occurred at restoration), take corrective action, and then restore the data again as necessary.

Point

- Since the special register set for the automatic restoration is in the latch area, the setting data is hold.
- SD918 (Restoration function setting) holds the setting data when the CPU module is powered off and on or is reset. Therefore, if the CPU module is powered off and on or is reset while bit0 (Execution of automatic restoration) of SD918 is remaining in ON, the automatic restoration is executed again. If the automatic restoration is not required at the next time the CPU module is powered off and on or is reset, turn off bit0 of SD918 after the restoration is completed, and then power off and on or reset the CPU module.

(c) Operating status

Operating status of the automatic restoration can be checked by the LEDs on the front side of the CPU module.

Operating status	LED indication
Before automatic restoration start	MODE: On (green), SD CARD: On (green)
Automatic restoration in execution	MODE: Flash (green), SD CARD: Flash (green)
Automatic restoration completed	MODE: On (green), SD CARD: On (green)
Automatic restoration error	MODE: On (green), ERR: Flash (red), SD CARD: On (green)

(3) Checking error

- Even when an error has occurred at a restoration by turning on SM1929, a diagnostic error is not detected. In that case, the error code is stored in SD922 (Cause of error occurred during restoration).
- When an error has occurred at an automatic restoration using SD918, a diagnostic error is detected. Also, an error code is stored in SD922 (Cause of error occurred during restoration).

(QCPU User's Manual (Hardware Design, Maintenance and Inspection))

(4) Precautions

The following describes precautions for the restoration function.

(a) Removal/insertion of the SD memory card and power-off/reset of the CPU module during the restoration

Do not perform the following operations during the restoration operation.

- Removal and insertion of the SD memory card
- Power-off or reset of the CPU module

If performed, the data in the CPU module is left in an incomplete state, which is middle of the restoration processing. Do not run the CPU module with this incomplete state. Doing so may cause an unintended operation. Restore the data again or format each drive in the CPU module, and clear devices before writing programs or parameters to the programmable controller.

(b) Suspending processing of the restoration

Except for the automatic backup using SD918, the following operations can suspend the processing of the restoration.

- To disable the SD memory card operation forcibly
- To enable removal and insertion of the SD memory card

If the restoration processing is suspended, the data in the CPU module is left in an incomplete state, which is middle of the restoration processing. Do not run the CPU module with this incomplete state. Doing so may cause an unintended operation. Restore the data again or format each drive in the CPU module, and clear devices before writing programs or parameters to the programmable controller.

(c) Model name of the CPU module to be restored

Always restore the data to the CPU module whose model name is the same as the one of the backup source. If not, restoration cannot be performed.

(d) Combination of automatic restoration and other functions

Do not set automatic restoration using SD918, boot operation, automatic restoration for the CPU module change function, and restoration for the latch data backup to standard ROM simultaneously. If those functions are set simultaneously, the automatic restoration is not operated when the CPU module is powered off and on or is reset.

(e) State of the CPU module

If the states of the CPU module differ between the restoration destination and the backup source, set SD917 to 0 (all target data) and then restore the data.

The data cannot be restored when SD917 is 1 (device data only) or SD917 is 2 (all target data except for all device data).

(f) High speed monitor area from other station

If a restoration is executed without formatting Drive 0 (program memory) of the CPU module to be restored, the setting of high speed monitor area from other station follows to the setting of the CPU module to be restored.

(g) When the same name file exists in the restoration-target CPU module

If the same name file as the one in the backup data exists in the restoration-target CPU module, the file shall be overwritten by the one in the backup data.

(h) Changing operating status during restoration

During a restoration, the CPU module remains in STOP status even though the RUN/STOP/RESET switch is changed from STOP to RUN, or remote RUN or remote PAUSE is operated. If the operating status of a CPU module is changed, it shifts to the one changed after the restoration is completed.

(i) Required time for restoration

According to the number of data (folders) backed up, file size, and number of files, a restoration may take time. Due to the time, for an automatic restoration at a multiple CPU system configuration, an error occurs in other CPU modules, and an error occurs also in the CPU module restored automatically after the completion of the restoration. The restoration itself, however, is completed correctly. Therefore, change SD917 (Restoration target data setting) so that only the device data, which is cleared by the restart of the system, shall be restored, and execute the automatic restoration again.

(j) Reflection of restored data

Some of the parameters are reflected only when the CPU module is powered off and then on or is reset. If a restoration is executed while the CPU module is in STOP state and then is switched to RUN, the CPU module may not operate with the backed up data. In that case, power off and then on or reset the CPU module. For device data, since the device data except the latch-specified devices is initialized by powering off and then on or resetting the CPU module, restore the device data again as the need arises.

(k) Completed with an error

Since a restoration is completed with an error, do not execute a restoration in the following situation.

- Any files which have exactly the same name as the backup data exist in the restoration destination, and also, which the file password is set
- The data in the backup folder is deleted (Do not delete the data in the backup folder that is likely to be restored)
- An error exists in the backup data (the backup data which has been changed its contents or which the CPU module is powered on and then off during the backup)

(l) Parameter-valid drive at automatic restoration

When bit1 of SD918 (Restoration function setting) is turned off (No format), always check the data in the CPU module of restoration destination before restoration. When the drive formatting is invalid, if the parameter-valid drive in the CPU module of the restoration destination is different from the one at the backup, parameters in the data backed up cannot be valid as following example.

Ex. When the parameter-valid drive at backup is the standard ROM (Drive 4), and parameters exist in the program memory (Drive 0) in the CPU module of restoration destination
If the drive formatting at automatic restoration set to invalid, parameters are restored to Drive 4 while the parameter-valid drive is still Drive 0. Thus, parameters in the program memory shall be valid.

(m) The rise instruction at restoration

At a restoration of program files, the execution status of the instruction becomes non-execution. Therefore, if the restoration is executed while SD917 is 0 (all target data) or 2 (all target data except for device data) and the execution condition of the rise instruction (PLS and □P) is satisfied when the RUN/STOP/RESET switch from STOP to RUN, the rise instruction is executed.

(n) Restoration of SFC program and device data

When the SFC program and device data are restored, turn on SM326 (SFC device clear mode) and then switch to RUN after the restoration. If the CPU module is changed from STOP to RUN after the restoration while SM326 is off, the device is cleared.

(o) When using the IP address change function

If a backup is performed while the IP address is stored in IP address storage area (flash ROM), the value of the area is changed at a time of restoration. The IP address is changed at the following timing.

- Restoration by turning on SM1929: When the CPU module is powered off and then on or is reset
- Automatic restoration using SD918: When an automatic restoration is executed

(p) Operations and functions which cannot be performed

Operations and functions which cannot be performed are the same as the ones at backup. (☞ Page 289, Section 3.32.1 (5) (i))

(q) Operation behavior of data logging function and sampling trace function

If the data is backed up while the data logging function or sampling trace function is executed and each function is set as it is started automatically, when the CPU module is shift to RUN after restoration, the data logging function or sampling trace function is automatically started.
If the restart of each function after restoration is required without the above setting, each operation to start the data logging function or sampling trace function is required.

(r) Request source of the sampling trace function

If the data is backed up while the sampling trace function is executed and then start/suspend the trace or execute the trigger, the request is identified as the one from different request source. At that time, an alert window is displayed, this has no effect on the operation of the sampling trace function.

(s) Sampling trace function

If data are backed up while the sampling trace function is being executed, the restoration destination CPU module may misunderstand that a request is from a different request source when trace start/suspension or trigger execution is requested. An alarm message is displayed at this time. However, there is no impact on the operation of sampling trace function.

(t) Data protected by security functions

Data protected by the following security functions cannot be restored.

- File password 32
  If the name of a file in the restoration destination CPU module and the name of a file in backup data are identical, and a file password has been set to the file, the restoration cannot be performed.
- File access control by security key
  If the CPU module is locked, the backup cannot be performed.

(u) Continuation start of the SFC program after restoration

- Restored SFC program will not be executed with a continuation start, but with an initial start.

(v) When data is restored with the setting to restore the special relay and special register

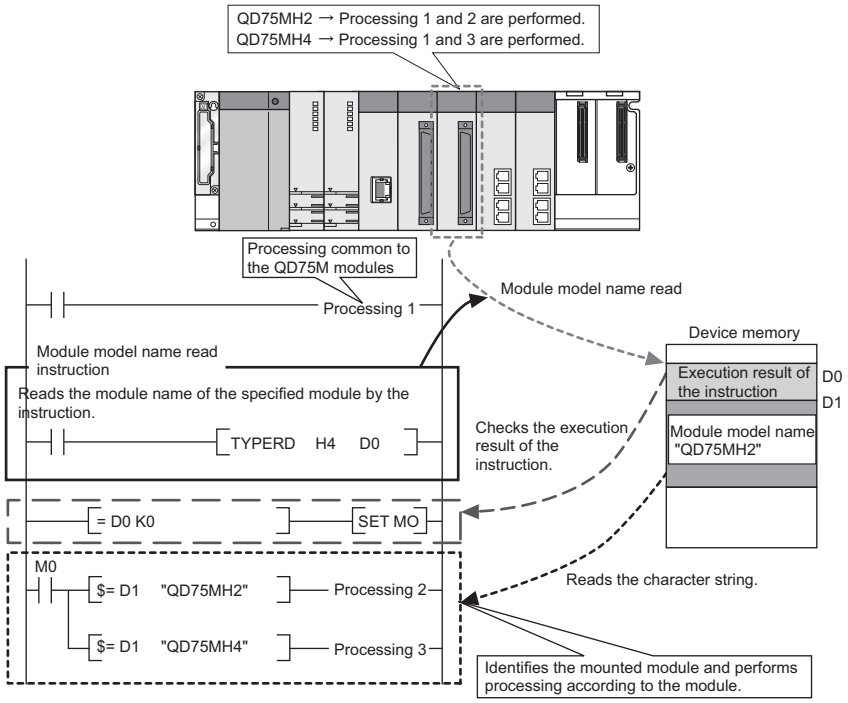
The CPU module is operated with the values of the backup function setting such as time/day of the week, day/time setting for the automatic backup, or the upper limit value of the number of backup data set before the restoration. (The special register except for the values of the backup function setting is overwritten.) To back up data with the values of the backup function setting set at the backup, set the values again.

(w) Restoration execution during restoration processing

During a restoration processing, the restoration by turning on SM1929 cannot be executed. (The execution will be ignored.)

3.33 Module Model Name Read Note 3.21

This function reads the model name of a module on a base unit. The mounted module is identified in a ladder program and processing according to the module can be performed.



(1) Execution method

Use the TYPERD instruction to read model names. For details of the instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

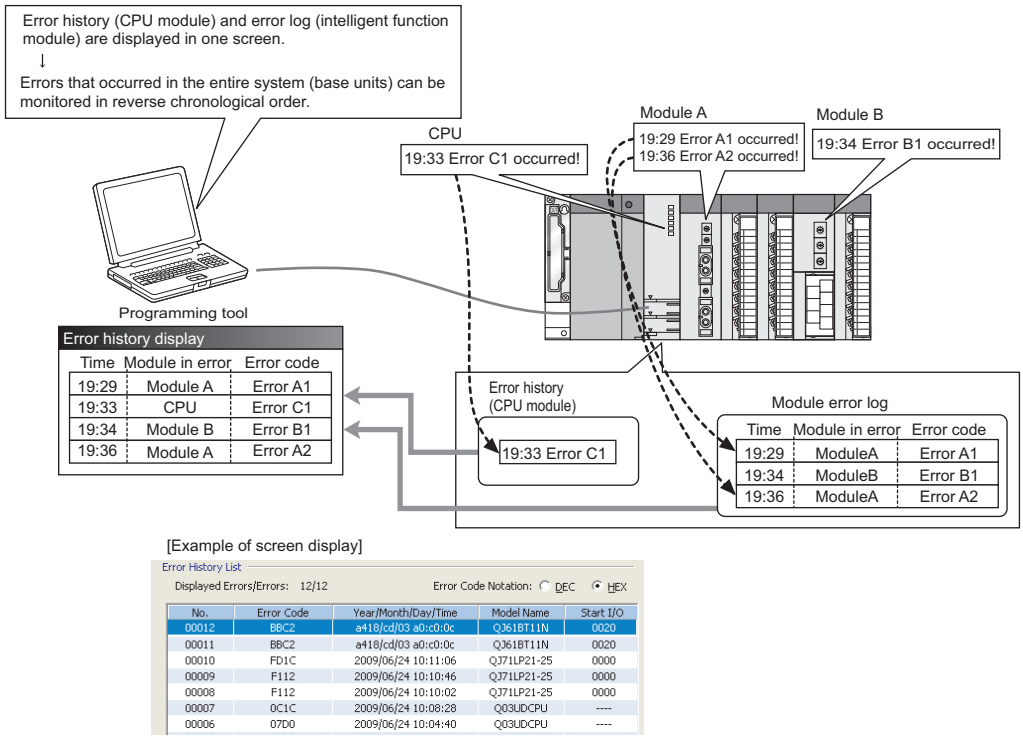
Note 3.21 Universal

Before executing the function, check the versions of the CPU module and programming tool used.

(Page 466, Appendix 2)

3.34 Module Error Collection Note 3.22

This function collects errors occurred in the connected intelligent function modules in the CPU module. By storing the errors in a memory that can hold data in the event of a power failure, the errors can be held even after power-off or reset.



[Example of screen display]

(1) Supported modules

The CPU module collects errors occurred in the connected intelligent function modules*1.
The CPU module does not collect the errors of intelligent function modules on other stations in the network.

*1 Indicates intelligent function modules that support this function. For supported module versions, refer to the manual for each module.

(2) Timing when module errors are collected

Module errors are not collected during execution of a program such as the COM instruction but collected in END processing.

Note 3.22 Universal

Before executing the function, check the versions of the CPU module and GX Works2 used.

(Page 466, Appendix 2) GX Developer cannot display module errors.

(3) Storing module errors

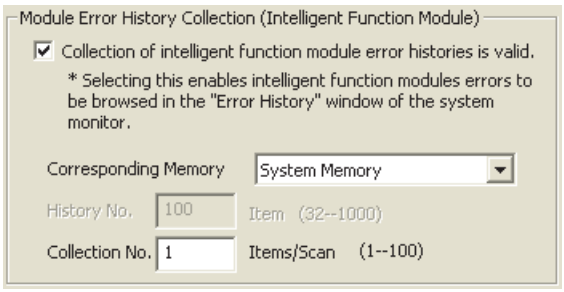
The module errors can be stored either in the system memory[*1] or the standard RAM.
The errors are stored separately from error history (CPU module) data.

CPU module	System memory	Standard RAM
Q00UJCPU	40 (Fixed)	-
Q00UCPU, Q01UCPU	40 (Fixed)	1000
Q02UCPU, QnUD(H)CPU, Built-in Ethernet port QCPU	100 (Fixed)	1000

*1 The memory is managed inside the system.

(4) Setting method

Select "Collect error histories of intelligent function modules" in "Module Error History Collection (Intelligent Function Module)" in the PLC RAS tab of the PLC parameter dialog box.



Item	Description	Setting range	Default
Corresponding Memory	Select a storage location.	• System Memory • Standard RAM[*1, *2]	System Memory
History No.	Set the number of collected errors only when the errors are stored in the standard RAM.	32 to 1000	40/100
Collection No.	Set the number of collected errors in one scan.[*3]	• Stored in system memory: 1 to 100 • Stored in standard RAM: 1 to 128	1

*1 With CPU modules other than the High-speed Universal model QCPU and Universal model Process CPU, if a sampling trace file is stored in the standard RAM, the file will be deleted when the CPU module is powered off and then on or is reset. However, with the High-speed Universal model QCPU and Universal model Process CPU, the file will not be deleted.

*2 The battery consumption may be increased.
QCPU User's Manual (Hardware Design, Maintenance and Inspection)

*3 If collected module errors are frequently lost, set a greater value to "Collection No.". The recommended value is the number of intelligent function modules that support this function.

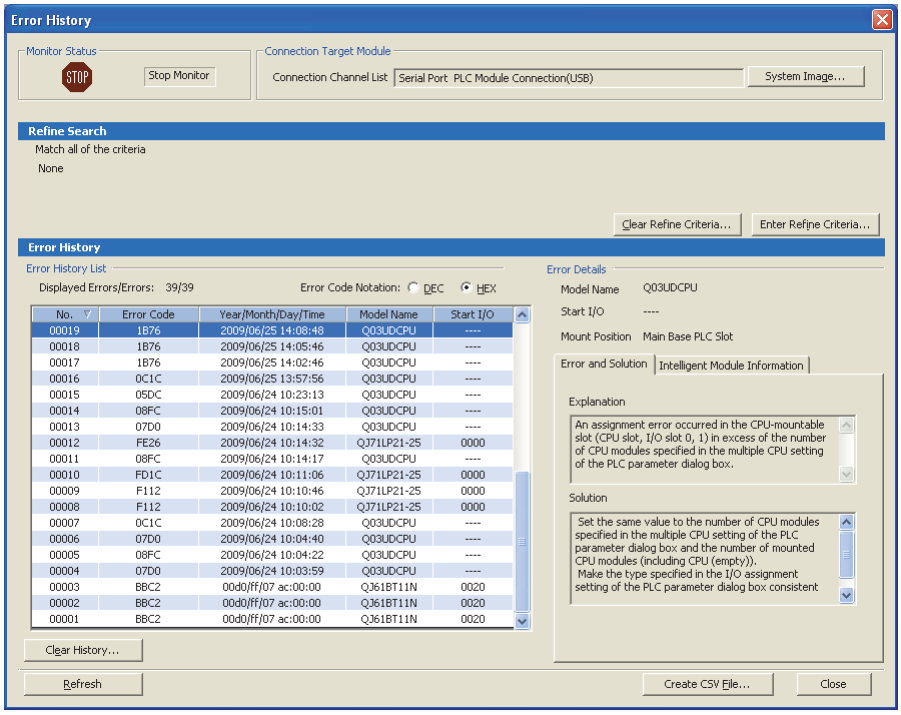
Parameter settings are enabled to the CPU module when:

- the CPU module is powered off and then on or
- the CPU module is reset.

(5) Monitoring module errors

Collected module error logs can be checked in the "Error History" screen of GX Works2.

[Diagnostics] ⇨ [System Monitor] ⇨ [System Error History] button



Item	Description	Remarks
Error Code*1	Displays error code numbers.	-
Year/Month/Day/Time*2	Displays the year, month, day, hour, minute, and second when an error occurred.	The year can be displayed within the range of 1980 to 2079.
Model Name	Displays a module model name.	-
Start I/O	Displays the start I/O number of a module in error.	-

*1 For details of error codes, refer to the manual for the intelligent function module.

*2 If an error occurred during initial processing, its occurrence time may be stored as "0000/00/00 00:00:00" in the module error collection file. In this case, the error is not displayed in correct order in Error History List.

Point

- The Error History screen can be displayed by selecting a module figure in the System Monitor screen and clicking the Error History Detail button. In this case, only the errors of the selected module are displayed.

  GX Works2 Version 1 Operating Manual (Common)
- Errors are not displayed for modules that do not support the module error collection function.
- Errors may not be displayed when they occur successively.

(6) Clearing module error history

Module error logs can be cleared by clicking the Clear History... button in the "Error History" screen. Note that error information on each intelligent function module displayed under "Error Details" is not cleared.

Point

The module error history is cleared when the standard RAM is formatted.
Note that a module error collection file cannot be deleted since it is automatically created after the CPU module is powered off and then on or is reset. To delete the file, clear the setting and then format the standard RAM.

(7) Precautions

(a) Using the CPU module change function with memory card

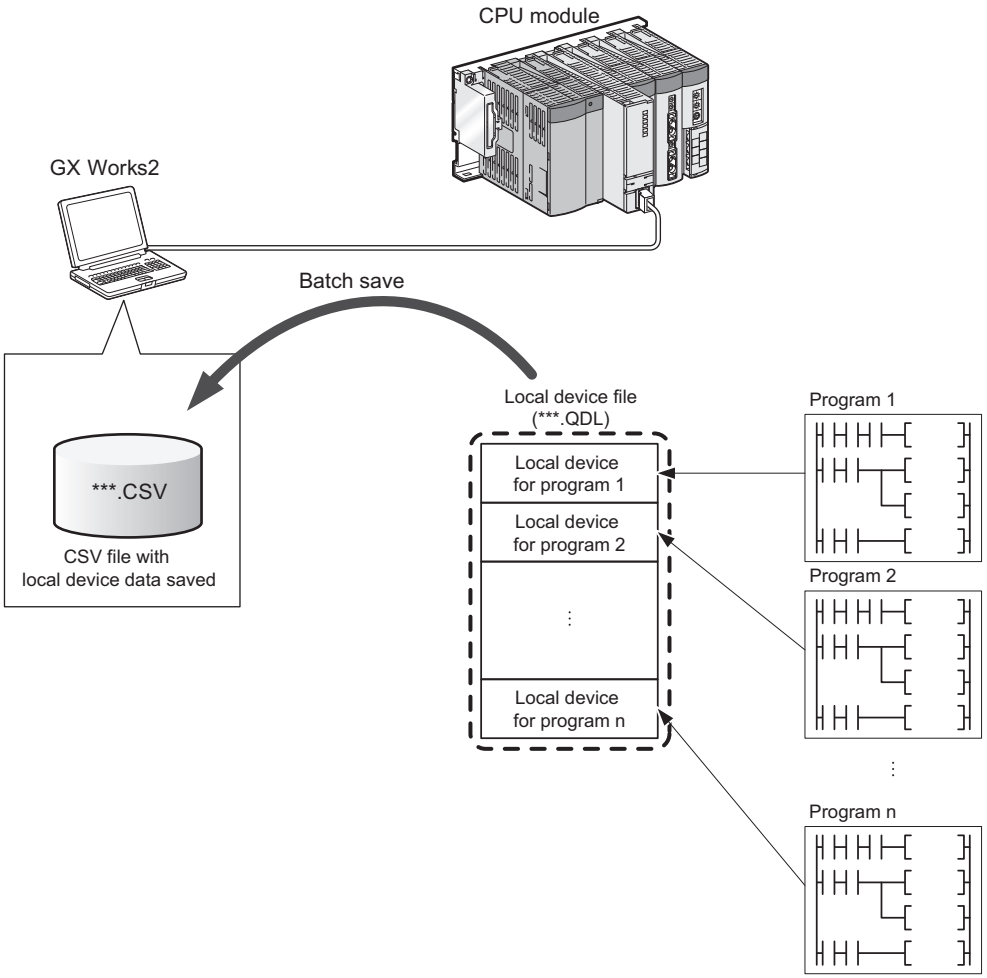
Backing up or restoring data will stop collecting module errors.

(b) Using the CPU module data backup/restoration function

Backing up or restoring data during the module error collection will stop collecting module errors.

3.35 Local Device Batch Read Function Note 3.23

This function batch-reads the contents of local devices in a CPU module and saves them in a CSV file. This function enables saving all the contents of local devices in one CSV file.



Point

Use GX Works2 to execute this function. (GX Developer does not support the function.)

Note 3.23 Universal

Before executing this function, check the versions of a CPU module and GX Works2 used.

(Page 466, Appendix 2) The Q00UJCPU does not support this function.

(1) Operating method

Open the "Local Device Batch Read + Save CSV" screen of GX Works2.

[Online] ⇨ [Local Device Batch Read + Save CSV]

For details, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

(2) CSV file contents and format

For the contents and format of CSV files, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

(3) Precautions

(a) When no local device file exists

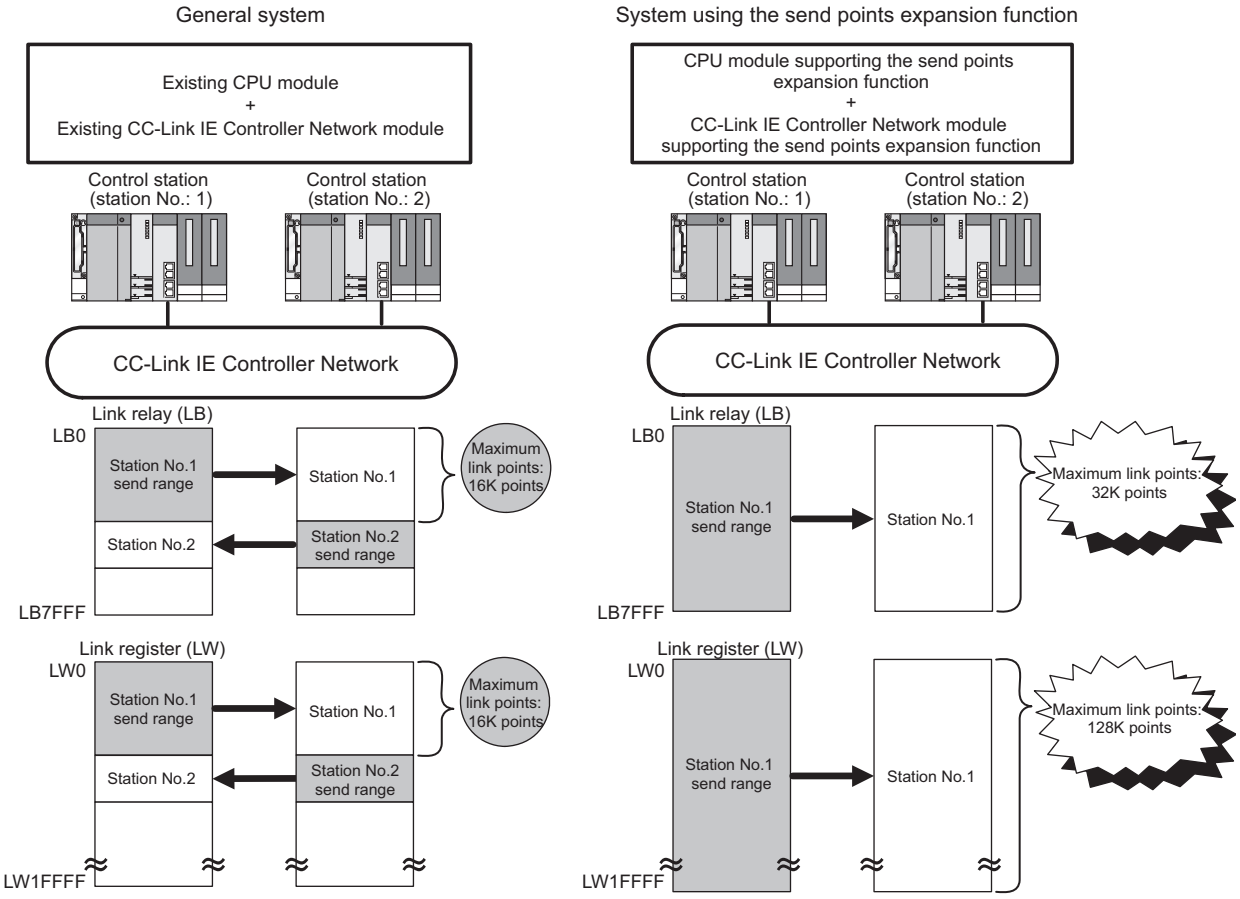
The local device batch read function cannot be executed.

(b) When the CPU module is in RUN status

If the local device batch read function is executed while the CPU module is running, the module performs read processing through multiple scans, and the read data are divided into the number of scans.
It is recommended to execute the function when the CPU module is in STOP or PAUSE status (when the device data do not change).

3.36 Send Points Extension Function (CC-Link IE Controller Network Module) Note 3.24

This function extends the maximum number of link points per CC-Link IE Controller Network module. Cyclic transfer can be performed up to 32k points for link relay (LB) and 128k points for link register (LW).



Point

Use GX Works2 to execute this function. (GX Developer does not support the function.)

Note 3.24 Universal

Before executing this function, check the versions of the CPU module and GX Works2 used.

(Page 466, Appendix 2)

(1) Settings

Set the following network parameters using GX Works2.

- Network type
- Network range assignment
- Refresh parameters

For details, refer to the following.

CC-Link IE Controller Network Reference Manual

(2) Precautions

(a) Boot operation

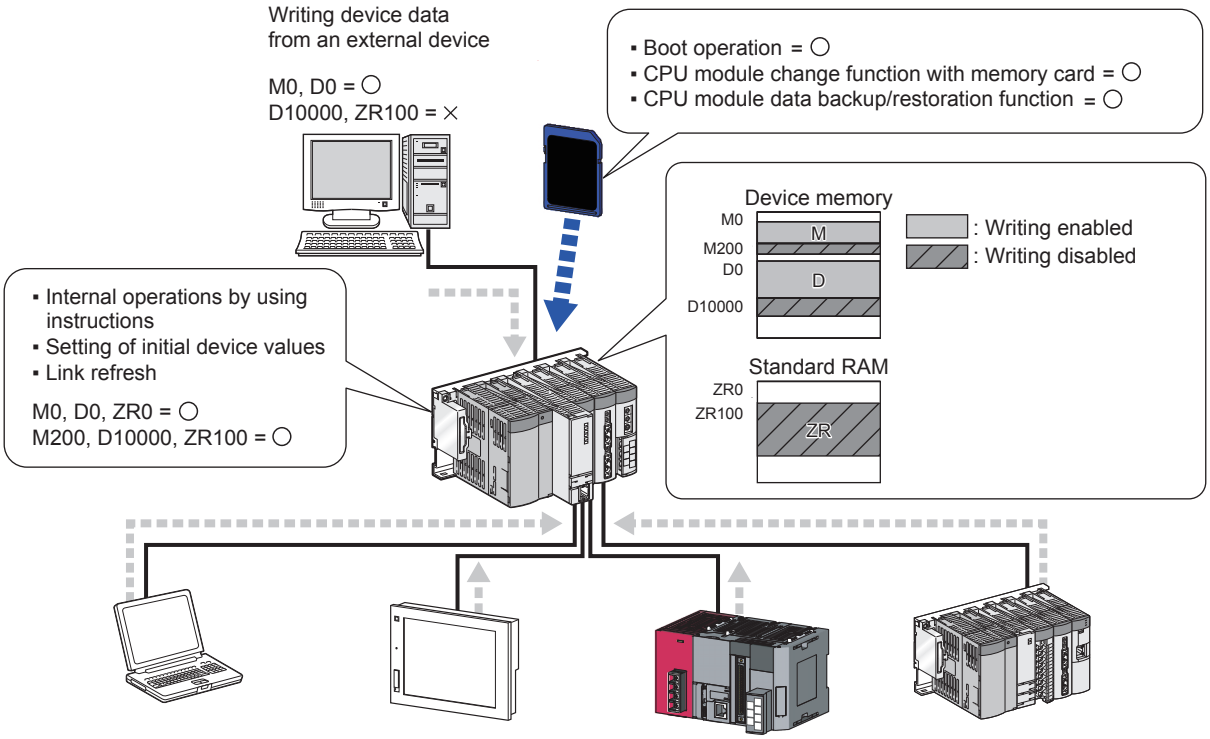
If the parameters for the send points extension function are stored in a memory card or SD memory card and the parameters are transferred to a CPU module that does not support this function, "LINK PARA. ERROR" (error code: 3102) occurs.

(b) Restoring backup data from a memory card, SD memory card, or GOT

If the parameters for the send points extension function backed up in a memory card, SD memory card, or GOT are restored to a CPU module that does not support this function, "LINK PARA. ERROR" (error code: 3102) occurs.

3.37 Write-Protect Function for Device Data (from Outside the CPU Module) Note 3.25

This function disables device data writing (including the file register) from outside the CPU module such as the programming tool, GOT, SLMP/MC protocol, and FTP to the write-protected range set in the parameter. Desired device data can be protected from tampering.
Even when the write-protected range is set, the operation of the CPU module which is set and device data writing by the execution of functions (device data writing by instructions and internal device data writing by initial device value setting, boot operation, and CPU module data backup/restoration function) are not inhibited.



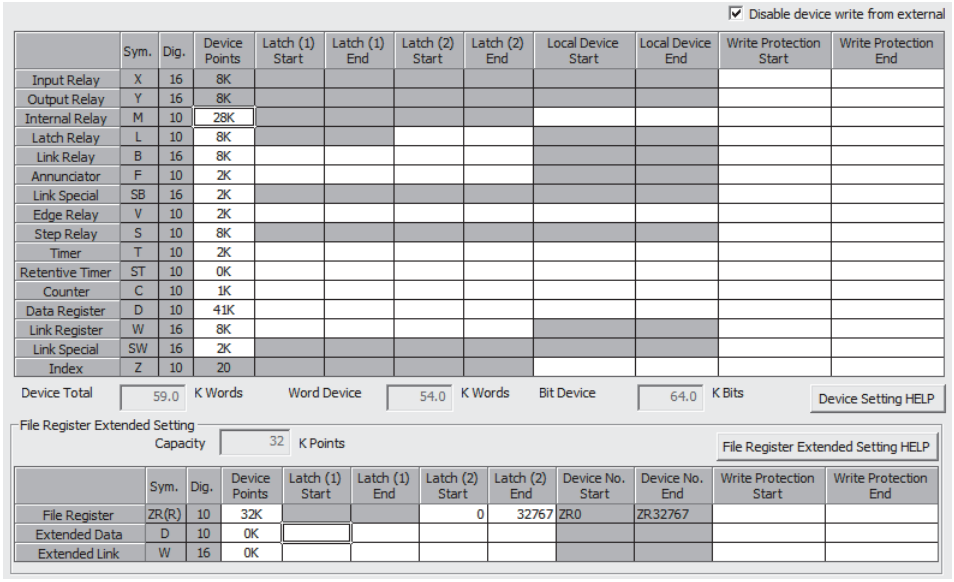
○: Can be written (function/instruction can be executed)
×: Cannot be written (function/instruction cannot be executed)

Note 3.25 Universal

The write-protect function for device data (from outside the CPU module) can be used only with the High-speed Universal model QCPU and Universal model Process CPU. When using the write-protect function for device data (from outside the CPU module), check the versions of the CPU module and programming tool used. (☞ Page 466, Appendix 2)

3.37.1 Setting method

Set the write-protected range in the device tab of the PLC parameter.



Disable device write from external

	Sym.	Dig.	Device Points	Latch (1) Start	Latch (1) End	Latch (2) Start	Latch (2) End	Local Device Start	Local Device End	Write Protection Start	Write Protection End
Input Relay	X	16	8K								
Output Relay	Y	16	8K								
Internal Relay	M	10	28K								
Latch Relay	L	10	8K								
Link Relay	B	16	8K								
Annunciator	F	10	2K								
Link Special	SB	16	2K								
Edge Relay	V	10	2K								
Step Relay	S	10	8K								
Timer	T	10	2K								
Retentive Timer	ST	10	0K								
Counter	C	10	1K								
Data Register	D	10	41K								
Link Register	W	16	8K								
Link Special	SW	16	2K								
Index	Z	10	20								

Device Total 59.0 K Words Word Device 54.0 K Words Bit Device 64.0 K Bits Device Setting HELP

File Register Extended Setting

Capacity 32 K Points File Register Extended Setting HELP

	Sym.	Dig.	Device Points	Latch (1) Start	Latch (1) End	Latch (2) Start	Latch (2) End	Device No. Start	Device No. End	Write Protection Start	Write Protection End
File Register	ZR(R)	10	32K			0	32767	ZR0	ZR32767		
Extended Data	D	10	0K								
Extended Link	W	16	0K								

Selecting the "Disable device write from external" checkbox allows input of the write-protected range (Write Protection Start/End). (Page 447, Appendix 1.2.8)

To input the write-protected range of the file register, extended data register, and extended link register, set the file register file to "Use the following file" in the PLC file tab of the PLC parameter. (Page 441, Appendix 1.2.3)

Point

- Setting a file password to the user parameter file protects the setting of the write-protect function for device data (from outside the CPU module).
- Whether the write-protect function for device data (from outside the CPU module) is enabled or not can be checked in SM214 (Write-protect status for device data (from outside the CPU module).

3.37.2 Target devices

This section describes the devices that can be write-protected from outside the CPU module. Device data writing by digit specification of bit data, bit specification of word, and index modification cannot be performed when it is specified in the write-protected range.
One write-protected range can be set per device.

Device type
X(DX), Y(DY), M, L, B, F, SB, V, S, T, ST, C, D[*1], W[*1], SW, Z, ZR(R)[*2]

*1 The extended data register and extended link register are included.
*2 When the file register in the block switching method (R) is specified from the SLMP/MC protocol, the write-protected range depends on the block number which is enabled.

Point

When the write-protect function for device data (from outside the CPU module) is enabled, device data writing by indirect specification cannot be executed regardless of the write-protected range specification.

Device data writing by the following files cannot be executed.

File type	Condition under which writing of files is inhibited
Initial device value file	The write-protect function for device data (from outside the CPU module) is enabled.[*1]
File register file	The write-protected range is set for the file register (ZR (R)), extended data register (D), and extended link register (W).[*2]

*1 The initial device value file cannot be written as long as this function is enabled, even if the write-protected range set does not include the device set as the initial device value range.
*2 When the write-protected range is set for the file register, the file register file cannot be written.

Even when the write-protect function for device data (from outside the CPU module) is enabled, files can be deleted (including format of the CPU memory).

The following table provides the availability of writing of files by the setting of the PLC file.

×: File cannot be written

Item	Initial device value file		File register file	
	Device in the write-protected range	Device out of the write-protected range	Device in the write-protected range	Device out of the write-protected range
Not used	×	×	×	×
Use the same file name as the program	×	×	×	×
Use the following file	×	×	×	×

3.37.3 Operations and functions that cannot be executed for devices in write-protected range

This section describes operations and functions that cannot be executed for devices in the write-protected range.

Operation and function		
Operation from the programming tool	Modify value[*1]	Watch window
		Device/buffer memory batch monitoring
		Local device monitoring
	Write to PLC	Device memory
		Initial device value
		File register
	Remote operation	Remote RUN when clearing is specified in clear mode
		Remote latch clear
	Clear PLC memory	Clear device's whole memory
		Clear all file registers
	External input/output forced on/off function • Registration	
	Executional conditioned device test • Registration[*2]	
Others	Device data writing and writing of files using the SLMP/MC protocol (☞ Page 310, Section 3.37.3 (1))	
	Device data writing and writing of files by instructions (☞ Page 311, Section 3.37.3 (2))	
	Device data writing and writing of files by FTP (☞ Page 311, Section 3.37.3 (3))	
	Device data writing from the programmable controller in another station or another CPU module	
	Simple PLC communication function • "Write" or "Transfer" from the communication destination	
	Latch clear by using the special relay and special register areas[*3]	
	Change TC setting value from GOT when the setting value has been specified by the device[*4]	

*1 If a label is specified, the operation cannot be performed when the device assigned to the label is in the write-protected range.

*2 Device data writing by index modification or indirect specification cannot be executed regardless of the write-protected range. When the write-protected range has been set for the file register (ZR (R)), an error occurs in the file register (R) at registration.

*3 SD339 (Latch clear operation setting) is not cleared.

*4 When index modification is specified to the setting value of the timer or counter, an error occurs.

(1) SLMP/MC protocol

An error occurs when the following operations are performed to devices in the write-protected range from an external device such as a personal computer via the SLMP/MC protocol: writing/clearing of device data, writing/changing of the initial device value file or file register file.
An error also occurs when the following commands are executed in the predefined protocol support function or the SLMP frame send instruction.

- Writing or clearing device data by using the SLMP/MC protocol

Classification	Operation	Command				
		SLMP and 4C/3C/4E/3E frame	2C frame	1C frame		1E frame
Writing to device	Batch write	1401	3	BW	JW	02H
			4	WW	QW	03H
	Random write (Test)	1402	6	BT	JT	04H
			7	WT	QT	05H
	Multiple blocks batch write	1406	-	-	-	-
Clearing device	Remote RUN when clearing is specified in the clear mode	1001	-	-	-	-
	Remote latch clear	1005	-	-	-	-

- Writing or changing the initial device value file or file register file by using the SLMP/MC protocol

Classification	Operation	Command
		SLMP and 4C/3C/4E/3E frame
Writing to file	New file creation	1820
		1202
	File copy	1824
		1206
File change	File open	1827*1
	File information modification	1204*2
	File write	1829
		1203

*1 The operation cannot be performed only when the open mode is for writing (0100H).
*2 The operation cannot be performed only when the extension is changed to QDI (initial device value) or QDR (file register). (Subcommand: 0001, 0002)

Point

For details on commands, refer to the following.

SLMP Reference Manual
MELSEC Communication Protocol Reference Manual

(2) Instruction

An error occurs when the following operations are performed to devices in the write-protected range from other CPU modules by using instructions: writing/clearing of device data, writing/transferring of the initial device value file or file register file.

• Writing or clearing device data by instructions

Classification	Operation	Instruction
Writing to device	Writing data to the programmable controller in another station	SP.WRITE
		JP/GP.WRITE
		JP/GP.SWRITE
		J(P).ZNWR
		GP.RIWT
	Writing device data from another CPU module	D(P).DDWR
	Writing to a notification device for reading/writing data of programmable controller in another station (with the read/write notification)	JP/GP.SWRITE
		JP/GP.SREAD
Clearing device	Remote RUN when clearing is specified in the clear mode	Z(P).RRUN
		J(P)/G(P).REQ

• Writing or transferring the initial device value file or file register file by an instruction

Classification	Instruction
Writing or transferring a file by FTP client	SP.FTPPUT

(3) File transfer function (FTP server)

An error occurs when the following operations are performed to the CPU module, in which the write-protect function for device data (from outside the CPU module) is enabled, by using the FTP command: writing/clearing of device data, writing (transferring)/changing of the initial device value file or file register file.

• Writing/clearing device data by the FTP command

Classification	Operation	Command (subcommand)
Clearing device	Remote RUN when clearing is specified in the clear mode	quote (run)

• Writing (transferring) or changing the initial device value file or file register file by the FTP command

Classification	Operation	Command
Writing to file	File transfer	put
		mput
File change	Extension change to QDI or QDR	rename

3.37.4 Precautions

The following describes the precautions for the write-protect function for device data (from outside the CPU module).

(1) Execution of the executional conditioned device test

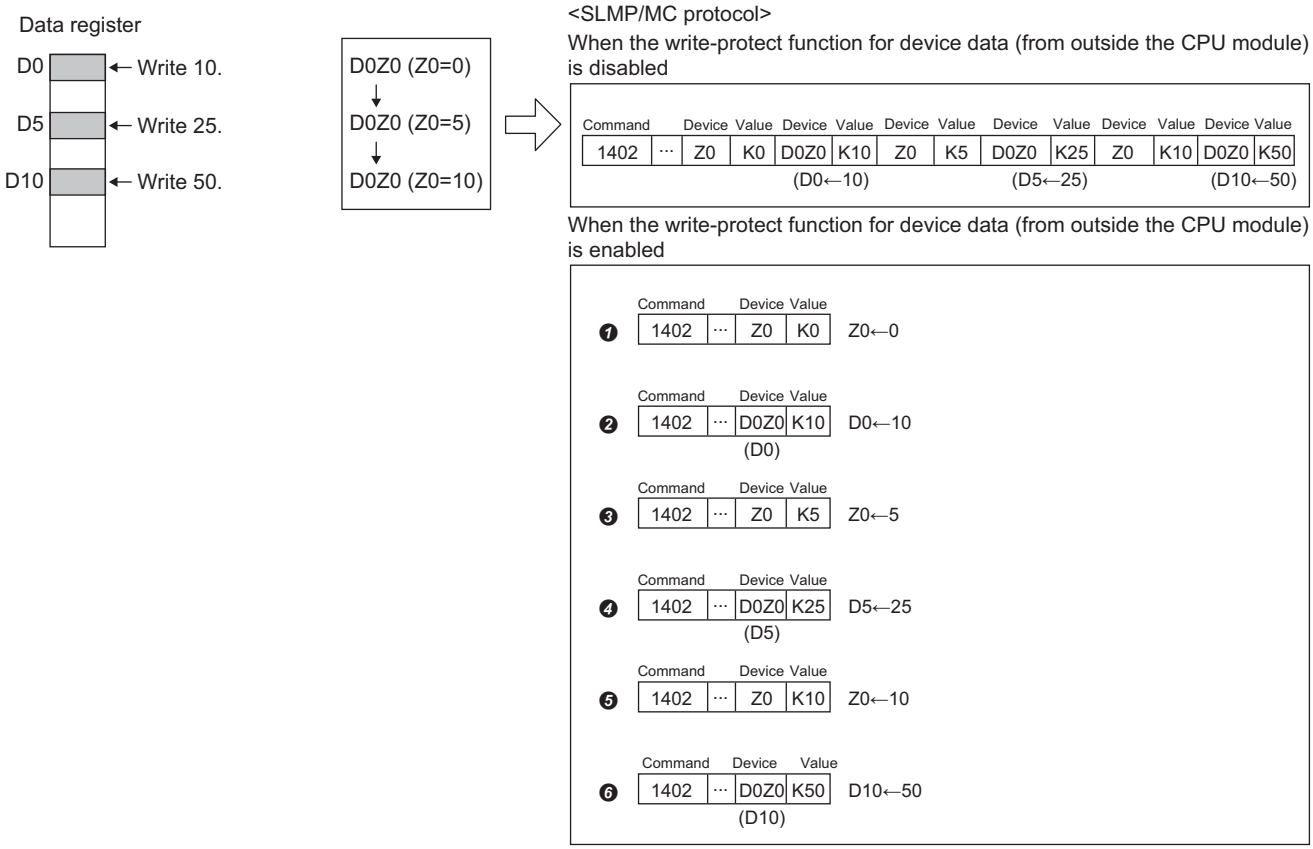
When specifying and registering the indirect specified/index-modified device or the file register (R) in the executional conditioned device test, disable the write-protect function for device data (from outside the CPU module) before execution.

If the function is enabled, an error occurs at registration when the indirect specified/index-modified device or the file register (R) is specified.

(2) Writing to the index register (Z) and index-modified device

When writing a value to the index register with the SLMP/MC protocol command 1402 (Write Random/Random write) and changing the access destination to write data by using the index register, disable the write-protect function for device data (from outside the CPU module) before execution. Alternatively, execute writing to the index register (Z) and writing to the index-modified device separately and with multiple commands. If they are specified simultaneously in a single command, an error occurs.

Ex. When the index register (Z0) is changed and device data are written to D0, D5, and D10



(3) Device memory or initial device value file, and writing parameter

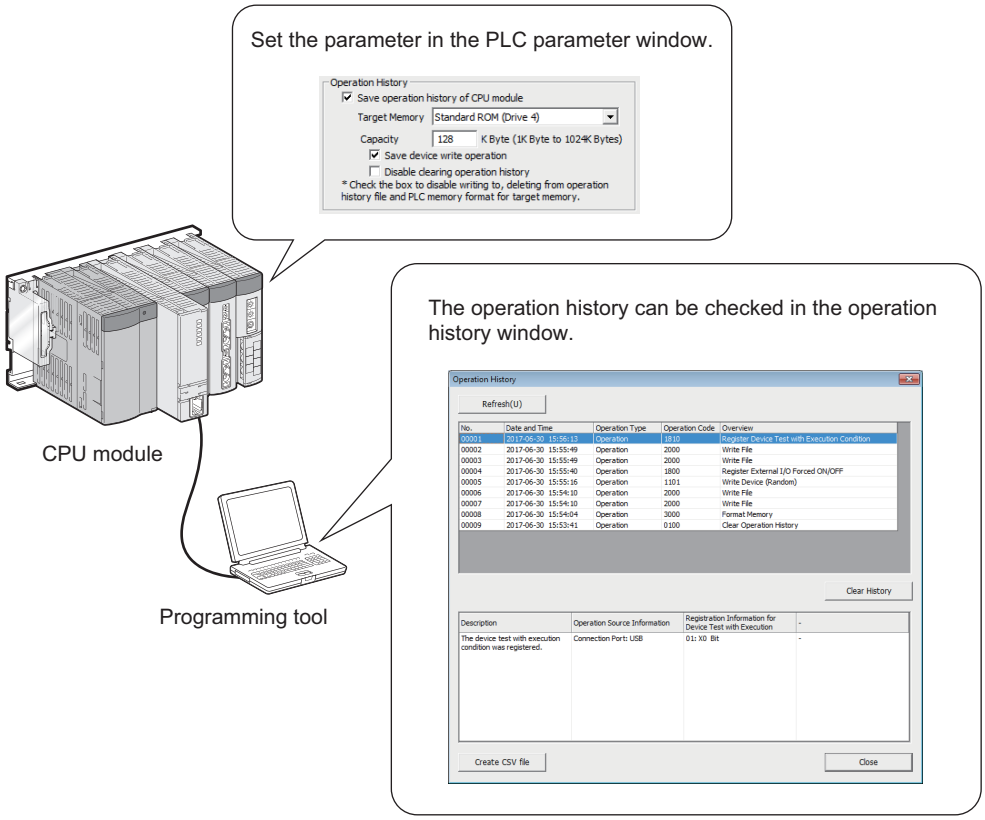
Settings for the write-protect function for device data (from outside the CPU module) are enabled at the following timing:

- The CPU module is powered off and on.
- The CPU module is reset.
- The operating status of the CPU module is switched from STOP to RUN.
- Data are written to the CPU module.

If the parameter in which the write-protect function for device data (from outside the CPU module) is enabled at the same time when the device memory or initial device value in the write-protected range are written, the parameter is written first and instantly enabled. Consequently, an error occurs at writing the device memory or initial device value.
Write the device memory or initial device value before writing the parameter.

3.38 Operation History Function Note 3.26

This function saves the operation information of device data writing and writing of files from outside the CPU module such as the programming tool, GOT, SLMP/MC protocol, and FTP into the CPU module as an operation history file, and displays it in the programming tool. Even if a problem occurs in the system due to incorrect operations, the history of operations to files and devices helps users to detect the incorrect operation and to investigate the cause of the problem.



The operation history function consists of the following functions.

Function	Reference
Operation history save function	Page 316, Section 3.38.1
Operation history display	Page 326, Section 3.38.2
Operation history clear function	Page 326, Section 3.38.3

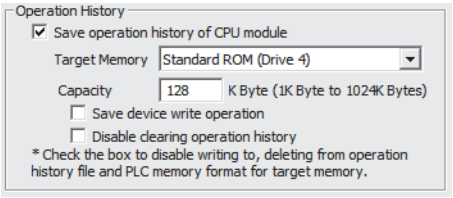
Operation history of device data writing and writing of files by operation of the CPU module itself or the execution of functions (writing device data or file internally) are not saved.

Note 3.26 Universal

The operation history function can be used only with the High-speed Universal model QCPU and Universal model Process CPU. When using the operation history function, check the versions of the CPU module and the programming tool used. (Page 466, Appendix 2)

(1) Setting method

To use the operation history function, select the "Save operation history of CPU module" checkbox in the PLC RAS tab of the PLC parameter. (☞ Page 442, Appendix 1.2.4)



Point

For a system where data is frequently written to devices and files, set a large capacity to save many histories. When setting a large capacity, specifying the SD memory card is highly recommended as the save destination memory.

3.38.1 Operation history save function

This function saves the memory operation history in the save destination set in the PLC parameter.

(1) Operation histories to be saved

The following table lists the operation histories to be saved.

Operation and function		
Writing to device	Operation from the programming tool	• Current value change using watch/monitor[*1] • Remote RUN when clearing is specified in the clear mode • Remote latch clear • Registration of external input/output forced on/off function[*1] • Registration of executional conditioned device test[*1]
	Others	• Device data writing using the SLMP/MC protocol (☞ Page 318, Section 3.38.1 (1) (a)) • Device data writing by instructions (☞ Page 319, Section 3.38.1 (1) (b)) • Device data writing by FTP (☞ Page 319, Section 3.38.1 (1) (c)) • Device data writing from the programmable controller in another station or another CPU module • Simple PLC communication function ("Write" or "Transfer" from the communication destination) • Latch clear by using the special relay and special register areas
Writing to file	Operation from the programming tool	• Write to PLC • Online Change • Clear all file registers • Writing protocol setting data of the predefined protocol support function[*2] • Writing the data logging setting • Change TC setting value
	Others	• Writing files using the SLMP/MC protocol (☞ Page 318, Section 3.38.1 (1) (a)) • Writing files by instruction (☞ Page 319, Section 3.38.1 (1) (b)) • Writing files by FTP (☞ Page 319, Section 3.38.1 (1) (c)) • Change TC setting value from GOT[*5]
Clear operation	Operation from the programming tool	• Clear device's whole memory • Format PLC memory[*3*4] • Delete PLC data
Clock setting	Operation from the programming tool	Clock setting with the programming tool (excluding the time adjustment using SNTP)
	Others	• Clock setting by the DATEWR/REQ/RTMWR instruction • Clock setting by the special relay (SM210, SM1025)
Operation to the operation history file	Operation from the programming tool	• Clearing the operation history[*3] • Creating a new operation history file

*1 This operation is saved when "Save device write operation" is enabled in the PLC RAS tab of the PLC parameter.
*2 This operation is saved only when data is written to the CPU module.
*3 Except when "Disable clearing operation history" is enabled in the PLC RAS tab of the PLC parameter.
*4 When this operation is performed to the drive specified as the target memory for saving operation history, the history of operation history file creation is saved.
*5 When the setting value is specified with the device, this operation is saved as device data writing. Since it is saved as a device, the operation is saved when "Save device write operation" is enabled in the PLC RAS tab of the PLC parameter.

Point

Clear operations are saved even when "Save device write operation" is not enabled in the PLC RAS tab of the PLC parameter.

(a) SLMP/MC protocol

When the following operations are performed, operation histories are saved.

• Writing to device

Classification	Operation	Command				
		SLMP and 4C/3C/4E/3E frame	2C frame	1C frame		1E frame
Writing to device[*1]	Batch write	1401	3	BW	JW	02H
			4	WW	QW	03H
	Random write (Test)	1402	6	BT	JT	04H
			7	WT	QT	05H
	Multiple blocks batch write	1406	-	-	-	-
Clearing device	Remote RUN when clearing is specified in the clear mode	1001				
	Remote latch clear	1005				

*1 This operation is saved when "Save device write operation" is enabled in the PLC RAS tab of the PLC parameter.

• Writing to file

Classification	Operation	Command
		SLMP and 4C/3C/4E/3E frame
Writing to file	File delete	1822
		1205
	File copy	1824
		1206
	File open	1827[*1]
File change	File information modification	1204
	File write	1203[*2]

*1 The operation is saved only when the open mode is for writing (0100H).

*2 The operation is saved only for the batch write (subcommand: 0000H).

(b) Instruction

When the following operations are performed, operation histories are saved.

• Writing to device

Classification		Instruction
Writing to device*1	Writing data to the programmable controller in another station	SP.WRITE
		JP/GP.WRITE
		JP/GP.SWRITE
		J(P).ZNWR
		GP.RIWT
	Writing device data from another CPU module	D(P).DDWR
	Writing to a notification device for reading/writing data of programmable controller in another station (with the read/write notification)	JP/GP.SWRITE
		JP/GP.SREAD
Clearing device	Remote RUN when clearing is specified in the clear mode	Z(P).RRUN
		J(P)/G(P).REQ

*1 This operation is saved when "Save device write operation" is enabled in the PLC RAS tab of the PLC parameter.

• Writing (transferring) of file

Classification	Instruction
Writing or transferring a file by FTP client	SP.FTPPUT

• Clock setting

Classification	Instruction
Clock data writing	DATEWR
Writing data to the clock of the programmable controller in another station	J(P)/G(P).REQ
	Z(P).RTMWR

(c) File transfer function (FTP server)

When the following operations are performed, operation histories are saved.

• Writing to device

Classification	Operation	Command
Clearing device	Remote RUN when clearing is specified in the clear mode	quote

• Writing to file

Classification	Operation	Command
Writing to file	Creating new file	put
		mput
	Deleting file	delete
		mdelete
File change	Modifying file information	rename

(2) Operation history file

The following describes the operation history file.

(a) Save destination memory

The operation history file can be saved in one of the following.

- Standard ROM
- SD memory card

Select the save destination memory in the RAS tab of the PLC parameter. (☞ Page 442, Appendix 1.2.4)

When the SD memory card is set as the save destination and the write protect switch of the SD memory card is enabled (write inhibited), the operation history is not saved. Therefore, if the write protect switch of the SD memory card is changed to be enabled during operation, an error occurs when the operation to be stored into operation history is performed after the write protect switch is enabled.

(b) File format

The file format of the operation history file is binary file. The operation history can be checked in the programming tool. (☞ Page 326, Section 3.38.2)

(c) File name

The file name and extension of the operation history file is fixed to OPERATE.QOL.

(d) File size (size for saving)

The size of the operation history file can be changed in the RAS tab of the PLC parameter. (☞ Page 442, Appendix 1.2.4)
The setting range is from 1 to 1024K bytes.
If the storage size exceeds the specified size, histories are deleted in order from the oldest one and the latest one is stored.

The following table lists each element to be stored in the operation history file and its size.

Element name	Size
File header size	64 bytes (fixed)
Operation history management information size	12 bytes (fixed)
Size per operation history	40 bytes minimum (variable)*1

*1 Since detailed information may differ depending on the operation history to be saved, the size per operation history is variable.

The number of the operations to be saved in the operation history file differs depending on the operation type to be saved. The following provides examples of the registrable number of operations when the size of the operation history file is the default of the function setting (128K bytes).

Ex. When programs (program name: 8 characters (12 characters including a period and extension)) are continued to be written to the CPU module
[Calculation formula]
128 [K bytes] = 128 [bytes] × 1024 = 131072 [bytes]
131072 [bytes] - (64 [bytes] + 12 [bytes]) = 130996 [bytes]
130996 [bytes] ÷ 72 [bytes] = 1819 [operations]

Up to 1819 operation histories (2000:Write File) can be registered.

Ex. When one point of M0 (bit) is continued to be written with SLMP Write command to the CPU module
[Calculation formula]
128 [K bytes] = 128 [bytes] × 1024 = 131072 [bytes]
131072 [bytes] - (64 [bytes] + 12 [bytes]) = 130996 [bytes]
130996 [bytes] ÷ 64 [bytes] = 2046 [operations]

Up to 2046 operation histories (1000: Write Device) can be registered.

The following shows an example of the operating procedure and the size of the operation history file.

Ex. When 100 programs (program name: 8 characters (12 characters including a period and extension)) are written to the CPU module by the following operating procedure.

[Operating procedure]

1. **Power on the CPU module with STOP state.**
2. **Write the parameter and 100 programs (program name: 8 characters (12 characters including a period and extension)) to the CPU module with the programming tool.**
3. **Switch the operating status of the CPU module to RUN.**

[Size of operation history file]

Element name		Size
File header		64 bytes
Operation history management information		12 bytes
Operation to be saved	Write File (PARAM.QPA)	72 bytes
	Write File (MAIN_001.QPG to MAIN_100.QPG)	7200 bytes
Total		7348 bytes

(e) File creation timing

The following table lists the timing when an operation history file is created.

Timing	Description
The CPU module is powered off and on.	• When no operation history file exists • When an operation history file exists and the file size is changed in the operation history setting*1
The CPU module is reset.	• When no operation history file exists • When an operation history file exists and the file size is changed in the operation history setting*1
The PLC memory is formatted.	When the PLC memory is formatted for the save destination memory of the operation history setting of the PLC parameter
An operation history file is deleted.	When an operation history file stored in the save destination memory set in the operation history setting is deleted
An operation history file name is changed.	When the name of an operation history file stored in the save destination memory set in the operation history setting is changed
An SD memory card is inserted.*2	• When no operation history file exists • When an operation history file exists and the file size is changed in the operation history setting*1
The CPU module data backup/restoration function is executed.	When a restoration, whose setting for operation after restoration is set to operate with the initial status (b15 of SD918 is off), is executed

*1 The existing operation history file is deleted and a new file is created.

*2 An operation history file is created when the parameter-valid drive is the program memory or standard ROM and the SD memory card is set as the save destination memory in the operation history setting of the PLC parameter.

Point

When a new operation history file is created, "Operation History File Creation Information" is saved.

The following table lists operations of the CPU module by the SD memory card status when the SD memory card is specified as the save destination memory.

- Operations when the CPU module is powered off and on or is reset

Status	Operation
An SD memory card is not inserted.	A stop error occurs when an operation history file is created.
The write protect switch of the SD memory card is enabled (write-protected).	If no operation history file exists, a stop error occurs when an operation history file is created. When an operation history file exists, the CPU module starts up successfully.
An SD memory card is inserted.	The CPU module starts up successfully.

- Operations when an operation whose operation history is saved in RUN/STOP (PAUSE) state is executed

Status	Operation
The SD memory card is removed or is forcibly disabled.	The operation history is temporarily saved in the internal memory. If the memory reaches the maximum number of operation histories it can store, all subsequent histories are lost.
The write protect switch of the SD memory card is enabled (write-protected).	A continuation error occurs at the execution of the operation to be saved. (The operation history can be displayed in the programming tool.)
The SD memory card is removed and other SD memory card in which the operation history file is stored is inserted.	• If the file size is same, operation histories are continued to be saved in the existing operation history file. • If the file size is different, the existing operation history file is deleted and a new file is created.
The write protect switch of the SD memory card is changed to be disabled or the SD memory card forced disable is cleared.	Operation histories are continued to be saved in the existing operation history file.
An SD memory card is inserted. (The write protect switch of the SD memory card is disabled and the SD memory card forced disable is cleared.)	Operation histories are saved in the operation history file.

(f) Timing when the setting is enabled

Any changed parameters take effect when:

- The CPU module is powered off and on.
- The CPU module is reset.

Any parameters in which the save destination memory and file size are changed in STOP state does not take effect when the CPU module operating status is changed from STOP to RUN. In this case, the changed parameters will take effect the next time when the CPU module is powered off and on or is reset.

(3) The number of displayed operation histories

The latest operation histories are displayed in the programming tool.

For the maximum number of displayed operation histories, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

(4) Loss of operation history

Some operation histories may be lost in the following cases.

- The CPU module is powered off and on or is reset immediately after the operation to be saved is performed (while the operation history is being written to the operation history file).
- Operations to be saved such as writing devices (writing to the operation history file) are frequently detected in the system.

When an operation history is lost, "*HST LOSS*" appears in the operation code in the operation history window. Whether any operation history has been lost or not can be checked in SM386 (Operation history information loss flag).

(5) Devices to be saved

If the following devices are written, "Write Device" is saved in the operation history. Writing by index modification or indirect specification is also saved. Although devices where the local device has been set are also saved, the local device and global device cannot be distinguished by the stored device name.

Device type
X(DX), Y(DY), M, L, B, F, SB, V, S, T(TS, TC, TN)*1, ST(STS, STC, STN)*1, C(CS, CC, CN)*1, D*2, W*2, SW, FX, FY, SM, FD, SD, Z, R, ZR, Jn\X, Jn\Y, Jn\B, Jn\SB, Jn\W, Jn\SW, Un\G, U3En\G

*1 Timer, retentive timer, and counter are displayed as TS, STS, and CS in contact, TC, STC, and CC in coil, and TN, STN, and CN in current value.

*2 The extended data register and extended link register are included.

3.38.2 Operation history display

Operation histories can be checked in the "Operation History" window.

[Diagnostics] ⇨ [PLC Diagnostics] ⇨ [Operation History] button

For details on the operation method and screen items, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

3.38.3 Operation history clear function

Clicking the [Clear History] button in the "Operation History" window clears all operation histories in the memory specified as the save destination memory.

[Diagnostics] ⇨ [PLC Diagnostics] ⇨ [Operation History] button

For details on the operation method, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

In the following cases, operation histories are not cleared and a message is displayed in the programming tool.

- "Disable clearing operation history" is enabled in the operation history setting in the PLC RAS tab of the PLC parameter (Page 326, Section 3.38.3 (1)).
- The operation history file is being accessed from other programming tools.
- The operation history file cannot be accessed.

(1) Disabling the operation history clear

Enabling "Disable clearing operation history" in the operation history setting in the PLC RAS tab of the PLC parameter disables the operation history clear. (Page 442, Appendix 1.2.4)

The following table lists the operations which cannot be executed when "Disable clearing operation history" is enabled.

Operation	
Operation history file	Writing the file
	Copying the file
	Changing the file name
	Deleting the file
Clearing the operation history	
Formatting the PLC memory*1	

*1 The operation cannot be executed only when the PLC memory is formatted for the save destination memory of the operation history file.

3.38.4 Precautions

This section describes the precautions for the operation history function.

(1) Operation history display and data update during execution of another function

The operation history cannot be displayed and data cannot be updated[*1] during execution of the following functions. Check that the following functions are not being executed before displaying the operation history or updating data.

- Restoration by the CPU module change function with memory card
- Restoration by the CPU module data backup/restoration function

*1 Data update indicates the operation of clicking the [Refresh] button in the "Operation History" window of the programming tool. For details, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

(2) Operating history clear during execution of anther function

No operation history can be cleared during execution of the following functions. Check that the following functions are not being executed before clearing the operation history.

- Backup and restoration by the CPU module change function with memory card
- Backup and restoration by the CPU module data backup/restoration function

(3) Combination with the data logging function

When the operation history function is used, the collecting performance of the data logging is decreased. Therefore, missing may occur in the data logging with the setting where no missing has been occurred.

(4) Operation history display and data update during an operation to be saved such as device data writing

It is recommended to display the operation history or update data when the operation to be saved as the operation history is not being performed.

If device data writing is executed frequently with the setting for saving device data writing, it may takes several tens of seconds to display operation history or update data due to frequent save processing of the operation history. In addition, "*HST LOSS*" may appear when the operation history is displayed or data is updated. (☞ Page 325, Section 3.38.1 (4))

3.38.5 List of operation codes

This section lists the operation codes displayed in the operation history window of the operation history function.

(1) How to read the list

The list contains the following information.

Item	Description
Operation Type	Information used to identify the operation type
Operation Code (decimal)	The ID number assigned to operation information
Overview	The overview of the detected operation
Description	The description of the detected operation
Detailed information 1 to 3	Details of the detected operation

(a) Detailed information

The following table lists the details of information displayed in the detailed information 1 to 3.

Detailed information	Item	Description
Detailed information 1	Operation Source Information	Information on the operation source • Connection port (connection information such as Ethernet and USB) • I/O No. • CPU number (a number assigned to CPU modules in a multiple CPU system) • Network number • Station number • IP address
	Operation History File Creation Information	Information on the operation history file
Detailed information 2	Clock Information (Before Change)	Clock information before change
	Drive/File Information	Information on the drive and file
	Copy Source (SRC) Drive/File Information	Information on the drive and file of the copy source
	Device Write Information	Information on the specified device (device name, data type, and number of points)
	Device Write Information (User Specification)	
	ON/OFF Registration Information	Information on the registered device (device name and on/off)
	Registration Information for Device Test with Execution Condition	Information on the device registered for the executional conditioned device test function
	Information for Clearing Device	Information on the cleared device
Detailed information 3	Clock Information (After Change)	Clock information after change
	Copy Destination (DST) Drive/File Information	Information on the drive and file of the copy destination
	Device Write Information (Access Destination)	Information on the access destination device (device name and memory address)

(2) List of operation codes

The following table lists the operation codes.

Operation Type	Operation Code (decimal)	Overview	Description	Detailed information		
				Detailed information 1	Detailed information 2	Detailed information 3
System	0001	Create Operation History File	The operation history file was created.	Operation History File Creation Information	-	-
Operation	0100	Clear Operation History	The operation history was cleared.	Operation Source Information	-	-
	0200	Set Clock	The clock was set.		Clock Information (Before Change)	Clock Information (After Change)
	1000	Write Device	It was written to the device.		Device Write Information	-
	1101	Write Device (Random)	It was written to the device. (Write Random 1st Point to 10th Point)		Device Write Information (User Specification)	Device Write Information (Access Destination)
	1102	Write Device (Random 11th Point or later)	It was written to the device. (Write Random 11th Point to 20th Point)			
	1103	Write Device (Random 21st Point or later)	It was written to the device. (Write Random 21st Point to 30th Point)			
	1104	Write Device (Random 31st Point or later)	It was written to the device. (Write Random 31st Point to 40th Point)			
	1105	Write Device (Random 41st Point or later)	It was written to the device. (Write Random 41st Point to 50th Point)			
	1106	Write Device (Random 51st Point or later)	It was written to the device. (Write Random 51st Point to 60th Point)			
	1107	Write Device (Random 61st Point or later)	It was written to the device. (Write Random 61st Point to 70th Point)			
	1108	Write Device (Random 71st Point or later)	It was written to the device. (Write Random 71st Point to 80th Point)			
	1109	Write Device (Random 81st Point or later)	It was written to the device. (Write Random 81st Point to 90th Point)			
	1110	Write Device (Random 91st Point or later)	It was written to the device. (Write Random 91st Point to 100th Point)			

Operation Type	Operation Code (decimal)	Overview	Description	Detailed information		
				Detailed information 1	Detailed information 2	Detailed information 3
Operation	1111	Write Device (Random 101st Point or later)	It was written to the device. (Write Random 101st Point to 110th Point)	Operation Source Information	Device Write Information (User Specification)	Device Write Information (Access Destination)
	1112	Write Device (Random 111th Point or later)	It was written to the device. (Write Random 111th Point to 120th Point)			
	1113	Write Device (Random 121st Point or later)	It was written to the device. (Write Random 121st Point to 130th Point)			
	1114	Write Device (Random 131st Point or later)	It was written to the device. (Write Random 131st Point to 140th Point)			
	1115	Write Device (Random 141st Point or later)	It was written to the device. (Write Random 141st Point to 150th Point)			
	1116	Write Device (Random 151st Point or later)	It was written to the device. (Write Random 151st Point to 160th Point)			
	1117	Write Device (Random 161st Point or later)	It was written to the device. (Write Random 161st Point to 170th Point)			
	1118	Write Device (Random 171st Point or later)	It was written to the device. (Write Random 171st Point to 180th Point)			
	1119	Write Device (Random 181st Point or later)	It was written to the device. (Write Random 181st Point to 188th Point)			
	1201	Write Device (Block)	It was written to the device. (Write Block 1st Point to 10th Point)		Device Write Information	-
	1202	Write Device (Block 11th Point or later)	It was written to the device. (Write Block 11th Point to 20th Point)			
	1203	Write Device (Block 21st Point or later)	It was written to the device. (Write Block 21st Point to 30th Point)			
	1204	Write Device (Block 31st Point or later)	It was written to the device. (Write Block 31st Point to 40th Point)			
	1205	Write Device (Block 41st Point or later)	It was written to the device. (Write Block 41st Point to 50th Point)			
	1206	Write Device (Block 51st Point or later)	It was written to the device. (Write Block 51st Point to 60th Point)			
	1207	Write Device (Block 61st Point or later)	It was written to the device. (Write Block 61st Point to 70th Point)			

Operation Type	Operation Code (decimal)	Overview	Description	Detailed information		
				Detailed information 1	Detailed information 2	Detailed information 3
Operation	1208	Write Device (Block 71st Point or later)	It was written to the device. (Write Block 71st Point to 80th Point)	Operation Source Information	Device Write Information	-
	1209	Write Device (Block 81st Point or later)	It was written to the device. (Write Block 81st Point to 90th Point)			
	1210	Write Device (Block 91st Point or later)	It was written to the device. (Write Block 91st Point to 100th Point)			
	1211	Write Device (Block 101st Point or later)	It was written to the device. (Write Block 101st Point to 110th Point)			
	1212	Write Device (Block 111th Point or later)	It was written to the device. (Write Block 111th Point to 120th Point)			
	1213	Write Device (Block 121st Point or later)	It was written to the device. (Write Block 121st Point to 130th Point)			
	1214	Write Device (Block 131st Point or later)	It was written to the device. (Write Block 131st Point to 140th Point)			
	1215	Write Device (Block 141st Point or later)	It was written to the device. (Write Block 141st Point to 150th Point)			
	1216	Write Device (Block 151st Point or later)	It was written to the device. (Write Block 151st Point to 160th Point)			
	1217	Write Device (Block 161st Point or later)	It was written to the device. (Write Block 161st Point to 170th Point)			
	1218	Write Device (Block 171st Point or later)	It was written to the device. (Write Block 171st Point to 180th Point)			
	1219	Write Device (Block 181st Point or later)	It was written to the device. (Write Block 181st Point to 190th Point)			
	1220	Write Device (Block 191st Point or later)	It was written to the device. (Write Block 191st Point to 192nd Point)			

Operation Type	Operation Code (decimal)	Overview	Description	Detailed information		
				Detailed information 1	Detailed information 2	Detailed information 3
Operation	1800	Register External I/O Forced ON/OFF	The external I/O forced ON/OFF was registered.	Operation Source Information	ON/OFF Registration Information	-
	1810	Register Device Test with Execution Condition	The device test with execution condition was registered.		Registration Information for Device Test with Execution Condition	-
	2000	Write File	A new file was created, or the file was written.		Drive/File Information	-
	2001	Copy File	The file was copied.		Copy Source (SRC) Drive/File Information	Copy Destination (DST) Drive/File Information
	2002	Rename File	The file was renamed.			
	3000	Format Memory	The PLC memory was formatted.		Drive/File Information	-
	3001	Clear Device Memory, and File Register	The device memory, and file register was cleared.		Information for Clearing Device	-
	3002	Delete File	The file was deleted.		Drive/File Information	-

3.39 iQ Sensor Solution Function Note 3.27 Note 3.28

The iQ Sensor Solution function performs the following operation.

Function	Description
Automatic detection of connected device	Detects devices supporting iQ Sensor Solution connected to the CPU module, and automatically displays them on "List of devices" and "Device map area" using a programming tool.
System configuration check	Compares the system configuration information displayed on a programming tool with the actual system configuration, and checks if they match.
Communication setting reflection	Reflects the communication settings (such as IP addresses) of devices supporting iQ Sensor Solution on "Device map area" to the devices connected over Ethernet in the system.
Sensor parameter read/write	Reads/writes parameters from/to devices supporting iQ Sensor Solution.
Monitoring	Monitors the current values (such as measurement values and input/output values), status (error existence), and error information of devices supporting iQ Sensor Solution graphically using a programming tool.
Data backup/restoration	Backs up the setting data (such as parameters) of devices supporting iQ Sensor Solution to an SD memory card. The data backed up can be restored as necessary.

Note 3.27 Universal

For details of the function, refer to the following.

iQ Sensor Solution Reference Manual

Note 3.28 Universal

The iQ Sensor Solution function can be used only with the High-speed Universal model QCPU and Universal model Process CPU. Before using the function, check the version of the CPU module and the programming tool used. (Page 466, Appendix 2)

Memo

PART 3 DEVICES, CONSTANTS

In this part, the devices and constants used in the CPU module are described.

CHAPTER 4 DEVICES

This chapter describes the devices that can be used in the CPU module.

4.1 Device List

The following table shows the devices used in the CPU module and applicable ranges.

(1) Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU

Classification	Type	Name	Default			Parameter-set range	Reference
			Point	Range			
Internal user device	Bit device	Input	8192	X0 to X1FFF	Hexadecimal	Can be changed within 29K words.*3	Page 348, Section 4.2.1
		Output	8192	Y0 to Y1FFF	Hexadecimal		Page 350, Section 4.2.2
		Internal relay	8192	M0 to M8191	Decimal		Page 351, Section 4.2.3
		Latch relay	8192	L0 to L8191	Decimal		Page 352, Section 4.2.4
		Annunciator	2048	F0 to F2047	Decimal		Page 353, Section 4.2.5
		Edge relay	2048	V0 to V2047	Decimal		Page 357, Section 4.2.6
		Step relay	8192	S0 to S8191	Decimal		Page 360, Section 4.2.9
		Link relay	8192	B0 to B1FFF	Hexadecimal		Page 358, Section 4.2.7
		Link special relay	2048	SB0 to SB7FF	Hexadecimal		Page 359, Section 4.2.8
	Word device	Timer*1	2048	T0 to T2047	Decimal		Page 360, Section 4.2.10
		Retentive timer*1	0	(ST0 to ST2047)	Decimal		
		Counter*1	1024	C0 to C1023	Decimal		Page 369, Section 4.2.11
		Data register	12288	D0 to D12287	Decimal		Page 373, Section 4.2.12
		Link register	8192	W0 to W1FFF	Hexadecimal		Page 374, Section 4.2.13
		Link special register	2048	SW0 to SW7FF	Hexadecimal		Page 376, Section 4.2.14

Classification	Type	Name	Default			Parameter-set range	Reference
			Point	Range			
Internal system device	Bit device	Function input	16	FX0 to FXF	Hexadecimal	Cannot be changed.	Page 377, Section 4.3.1
		Function output	16	FY0 to FYF	Hexadecimal		Page 377, Section 4.3.1
		Special relay	2048	SM0 to SM2047	Decimal		Page 379, Section 4.3.2
	Word device	Function register	5	FD0 to FD4	Decimal		Page 377, Section 4.3.1
		Special register	2048	SD0 to SD2047	Decimal		Page 379, Section 4.3.3
Link direct device	Bit device	Link input	8192	Jn\X0 to Jn\X1FFF	Hexadecimal	Cannot be changed.	Page 380, Section 4.4
		Link output	8192	Jn\Y0 to Jn\Y1FFF	Hexadecimal		
		Link relay	32768	Jn\B0 to Jn\B7FFF	Hexadecimal		
		Link special relay	512	Jn\SB0 to Jn\S1FF	Hexadecimal		
	Word device	Link register	131072	Jn\W0 to Jn\W1FFFF	Hexadecimal		
		Link special register	512	Jn\SW0 to Jn\SW1FF	Hexadecimal		
Module access device	Word device	Intelligent function module device	65536	Un\G0 to Un\G65535[*2]	Decimal	Cannot be changed.	Page 384, Section 4.5
		Cyclic transmission area device[*4]	4096	U3En\G0 to U3En\G4095	Decimal	Cannot be changed.	
Index register or standard device register	Word device	Index register or standard device register	20	Z0 to Z19	Decimal	Cannot be changed.	Page 387, Section 4.6
File register[*7]	Word device	File register	0	-	-	0 to 4086K points[*6]	Page 392, Section 4.7
Extended data register[*7]	Word device	Extended data register	0	-	-		Page 402, Section 4.8
Extended link register[*7]	Word device	Extended link register	0	-	-		Page 402, Section 4.8
Nesting	-	Nesting	15	N0 to N14	Decimal	Cannot be changed.	Page 407, Section 4.9
Pointer	-	Pointer	4096[*11]	P0 to P4095[*12]	Decimal	Cannot be changed.	Page 408, Section 4.10
		Interrupt pointer	256[*8]	I0 to I255[*9]	Decimal		Page 412, Section 4.11
Others	Bit device	SFC block device	128	BL0 to BL127	Decimal	Cannot be changed.	Page 415, Section 4.12.1
	-	Network No. specification device	255	J1 to J255	Decimal		Page 415, Section 4.12.2
		I/O No. specification device	-	U0 to U7F, U3E0 to U3E2[*10]	Hexadecimal		Page 416, Section 4.12.3
		Macro instruction argument device	10	VD0 to VD9	Decimal		Page 416, Section 4.12.4

Classification	Type	Name	Default		Parameter-set range	Reference
			Point	Range		
Constant	-	Decimal constant	K-2147483648 to K2147483647			Page 417, Section 5.1
		Hexadecimal constant	H0 to HFFFFFFFF			Page 417, Section 5.2
		Real number constant	Single-precision floating-point data: E ± 1.17549435 — 38 to E ± 3.40282347 + 38			Page 418, Section 5.3
			Double-precision floating-point data*5: E ± 2.2250738585072014 — 308 to E ± 1.7976931348623157 + 308			Page 418, Section 5.3
		Character string constant	Up to 32 characters, such as "ABC" and "123"			Page 419, Section 5.4

*1 These devices are used as a bit device for contacts and coils, and as a word device for controlling the present value.
*2 The number of points that can be actually used varies depending on intelligent function modules. For the number of buffer memory points, refer to the manual for the intelligent function module used.
*3 When changing device points, refer to ☞ Page 345, Section 4.2 (1), the parts describing the precaution.
*4 Available only in multiple CPU systems.
*5 Up to 15 digits can be entered in a programming tool.
*6 This is the total number of points for the file register, extended data register (D), and extended link register (W).
*7 Not available for the Q00UJCPU.
*8 The number of points for the Q00UJCPU, Q00UCPU, and Q01UCPU is 128.
*9 The range for the Q00UJCPU, Q00UCPU, and Q01UCPU is I0 to I127.
*10 The range for the Q00UJCPU is U0 to UF, and for the Q00UCPU and Q01UCPU is U0 to U3F and U3E0 to U3E2.
*11 The number of points for the Q00UJCPU, Q00UCPU, and Q01UCPU is 512.
*12 The range for the Q00UJCPU, Q00UCPU, and Q01UCPU is P0 to P511.

(2) QnUD(H)CPU, QnUDE(H)CPU

Classification	Type	Name	Default			Parameter-set range	Reference
			Point	Range			
Internal user device	Bit device	Input	8192	X0 to X1FFF	Hexadecimal	Can be changed within 29K words.*3	Page 348, Section 4.2.1
		Output	8192	Y0 to Y1FFF	Hexadecimal		Page 350, Section 4.2.2
		Internal relay	8192	M0 to M8191	Decimal		Page 351, Section 4.2.3
		Latch relay	8192	L0 to L8191	Decimal		Page 352, Section 4.2.4
		Annunciator	2048	F0 to F2047	Decimal		Page 353, Section 4.2.5
		Edge relay	2048	V0 to V2047	Decimal		Page 357, Section 4.2.6
		Step relay	8192	S0 to S8191	Decimal		Page 360, Section 4.2.9
		Link relay	8192	B0 to B1FFF	Hexadecimal		Page 358, Section 4.2.7
		Link special relay	2048	SB0 to SB7FF	Hexadecimal		Page 359, Section 4.2.8
	Word device	Timer*1	2048	T0 to T2047	Decimal		Page 360, Section 4.2.10
		Retentive timer*1	0	(ST0 to ST2047)	Decimal		
		Counter*1	1024	C0 to C1023	Decimal		Page 369, Section 4.2.11
		Data register	12288	D0 to D12287	Decimal		Page 373, Section 4.2.12
		Link register	8192	W0 to W1FFF	Hexadecimal		Page 374, Section 4.2.13
		Link special register	2048	SW0 to SW7FF	Hexadecimal		Page 376, Section 4.2.14
Internal system device	Bit device	Function input	16	FX0 to FXF	Hexadecimal	Cannot be changed.	Page 377, Section 4.3.1
		Function output	16	FY0 to FYF	Hexadecimal		Page 377, Section 4.3.1
		Special relay	2048	SM0 to SM2047	Decimal		Page 379, Section 4.3.2
	Word device	Function register	5	FD0 to FD4	Decimal		Page 377, Section 4.3.1
		Special register	2048	SD0 to SD2047	Decimal		Page 379, Section 4.3.3
Link direct device	Bit device	Link input	16384*7	Jn\X0 to Jn\X3FFF*8	Hexadecimal	Cannot be changed.	Page 380, Section 4.4
		Link output	16384*7	Jn\Y0 to Jn\Y3FFF*8	Hexadecimal		
		Link relay	32768	Jn\B0 to Jn\B7FFF	Hexadecimal		
		Link special relay	512	Jn\SB0 to Jn\S1FF	Hexadecimal		
	Word device	Link register	131072	Jn\W0 to Jn\W1FFFF	Hexadecimal		
		Link special register	512	Jn\SW0 to Jn\SW1FF	Hexadecimal		

Classification	Type	Name	Default			Parameter-set range	Reference
			Point	Range			
Module access device	Word device	Intelligent function module device	65536	Un\G0 to Un\G65535[*2]	Decimal	Cannot be changed.	Page 384, Section 4.5
		Cyclic transmission area device[*4]	4096	U3En\G0 to U3En\G4095	Decimal	Cannot be changed.	
			14336	U3En\G10000 to U3En\G24335	Decimal	Can be changed.	
Index register or standard device register	Word device	Index register or standard device register	20	Z0 to Z19	Decimal	Cannot be changed.	Page 387, Section 4.6
File register	Word device	File register	0	-	-	0 to 4086K points[*6]	Page 392, Section 4.7
Extended data register	Word device	Extended data register	0[*9]	-	-		Page 402, Section 4.8
Extended link register	Word device	Extended link register	0	-	-		Page 402, Section 4.8
Nesting	-	Nesting	15	N0 to N14	Decimal	Cannot be changed.	Page 407, Section 4.9
Pointer	-	Pointer	4096[*10]	P0 to P4095[*11]	Decimal	Cannot be changed.	Page 408, Section 4.10
		Interrupt pointer	256	I0 to I255	Decimal		Page 412, Section 4.11
Others	Bit device	SFC block device	320	BL0 to BL319	Decimal	Cannot be changed.	Page 415, Section 4.12.1
	-	Network No. specification device	255	J1 to J255	Decimal		Page 415, Section 4.12.2
		I/O No. specification device	516	U0 to U1FF, U3E0 to U3E3	Hexadecimal		Page 416, Section 4.12.3
		Macro instruction argument device	10	VD0 to VD9	Decimal		Page 416, Section 4.12.4
Constant	-	Decimal constant	K-2147483648 to K2147483647				Page 417, Section 5.1
		Hexadecimal constant	H0 to HFFFFFFFF				Page 417, Section 5.2
		Real number constant	Single-precision floating-point data: E ± 1.17549435 — 38 to E ± 3.40282347 + 38				Page 418, Section 5.3
			Double-precision floating-point data[*5]: E ± 2.2250738585072014 — 308 to E ± 1.7976931348623157 + 308				Page 418, Section 5.3
		Character string constant	Up to 32 characters, such as "ABC" and "123"				Page 419, Section 5.4

*1 These devices are used as a bit device for contacts and coils, and as a word device for controlling the present value.
*2 The number of points that can be actually used varies depending on intelligent function modules. For the number of buffer memory points, refer to the manual for the intelligent function module used.
*3 When changing device points, refer to ☞ Page 345, Section 4.2 (1), the parts describing the precaution.
*4 Available only in multiple CPU systems.
*5 Up to 15 digits can be entered in a programming tool.
*6 This is the total number of points for the file register, extended data register (D), and extended link register (W).
*7 The number of points for the Universal model QCPU whose serial number (first five digits) is "12011" or earlier is 8192.
*8 The range for the Universal model QCPU whose serial number (first five digits) is "12011" or earlier is Jn\X/Y0 to Jn\X/Y1FFF.
*9 The number of points for the Q50UDEHCPU and Q100UDEHCPU is 128K.
*10 The number of points for the Q50UDEHCPU and Q100UDEHCPU is 8192.
*11 The range for the Q50UDEHCPU and Q100UDEHCPU is P0 to P8191.

(3) QnUDVCPU, QnUDPVCPU

Classification	Type	Name	Default			Parameter-set range	Reference
			Point	Range			
Internal user device	Bit device	Input	8192	X0 to X1FFF	Hexadecimal	Can be changed.[*3]	Page 348, Section 4.2.1
		Output	8192	Y0 to Y1FFF	Hexadecimal		Page 350, Section 4.2.2
		Internal relay	28672[*7]	M0 to M28672[*8]	Decimal		Page 351, Section 4.2.3
		Latch relay	8192	L0 to L8191	Decimal		Page 352, Section 4.2.4
		Annunciator	2048	F0 to F2047	Decimal		Page 353, Section 4.2.5
		Edge relay	2048	V0 to V2047	Decimal		Page 357, Section 4.2.6
		Step relay	8192	S0 to S8191	Decimal		Page 360, Section 4.2.9
		Link relay	8192	B0 to B1FFF	Hexadecimal		Page 358, Section 4.2.7
		Link special relay	2048	SB0 to SB7FF	Hexadecimal		Page 359, Section 4.2.8
	Word device	Timer[*1]	2048	T0 to T2047	Decimal		Page 360, Section 4.2.10
		Retentive timer[*1]	0	(ST0 to ST2047)	Decimal		
		Counter[*1]	1024	C0 to C1023	Decimal		Page 369, Section 4.2.11
		Data register	41984[*9]	D0 to D41983[*10]	Decimal		Page 373, Section 4.2.12
		Link register	8192	W0 to W1FFF	Hexadecimal		Page 374, Section 4.2.13
		Link special register	2048	SW0 to SW7FF	Hexadecimal		Page 376, Section 4.2.14
Internal system device	Bit device	Function input	16	FX0 to FXF	Hexadecimal	Cannot be changed.	Page 377, Section 4.3.1
		Function output	16	FY0 to FYF	Hexadecimal		Page 377, Section 4.3.1
		Special relay	2048	SM0 to SM2047	Decimal		Page 379, Section 4.3.2
	Word device	Function register	5	FD0 to FD4	Decimal		Page 377, Section 4.3.1
		Special register	2048	SD0 to SD2047	Decimal		Page 379, Section 4.3.3
Link direct device	Bit device	Link input	16384	Jn\X0 to Jn\X3FFF	Hexadecimal	Cannot be changed.	Page 380, Section 4.4
		Link output	16384	Jn\Y0 to Jn\Y3FFF	Hexadecimal		
		Link relay	32768	Jn\B0 to Jn\B7FFF	Hexadecimal		
		Link special relay	512	Jn\SB0 to Jn\S1FF	Hexadecimal		
	Word device	Link register	131072	Jn\W0 to Jn\W1FFFF	Hexadecimal		
		Link special register	512	Jn\SW0 to Jn\SW1FF	Hexadecimal		

Classification	Type	Name	Default			Parameter-set range	Reference
			Point	Range			
Module access device	Word device	Intelligent function module device	65536	Un\G0 to Un\G65535[*2]	Decimal	Cannot be changed.	Page 384, Section 4.5
		Cyclic transmission area device[*4]	4096	U3En\G0 to U3En\G4095	Decimal	Cannot be changed.	
			14336	U3En\G10000 to U3En\G24335	Decimal	Can be changed.	
Index register or standard device register	Word device	Index register or standard device register	20	Z0 to Z19	Decimal	Cannot be changed.	Page 387, Section 4.6
File register	Word device	File register	0	-	-	Can be changed. (in increments of 1K)[*6]	Page 392, Section 4.7
Extended data register	Word device	Extended data register	0	-	-		Page 402, Section 4.8
Extended link register	Word device	Extended link register	0	-	-		Page 402, Section 4.8
Nesting	-	Nesting	15	N0 to N14	Decimal	Cannot be changed.	Page 407, Section 4.9
Pointer	-	Pointer	4096	P0 to P4095	Decimal	Cannot be changed.	Page 408, Section 4.10
		Interrupt pointer	256	I0 to I255	Decimal		Page 412, Section 4.11
Others	Bit device	SFC block device	320	BL0 to BL319	Decimal	Cannot be changed.	Page 415, Section 4.12.1
	-	Network No. specification device	255	J1 to J255	Decimal		Page 415, Section 4.12.2
		I/O No. specification device	516	U0 to U1FF, U3E0 to U3E3	Hexadecimal		Page 416, Section 4.12.3
		Macro instruction argument device	10	VD0 to VD9	Decimal		Page 416, Section 4.12.4
Constant	-	Decimal constant	K-2147483648 to K2147483647				Page 417, Section 5.1
		Hexadecimal constant	H0 to HFFFFFFFF				Page 417, Section 5.2
		Real number constant	Single-precision floating-point data: E ± 1.17549435 — 38 to E ± 3.40282347 + 38				Page 418, Section 5.3
			Double-precision floating-point data[*5]: E ± 2.2250738585072014 — 308 to E ± 1.7976931348623157 + 308				Page 418, Section 5.3
		Character string constant	Up to 32 characters, such as "ABC" and "123"				Page 419, Section 5.4

*1 These devices are used as a bit device for contacts and coils, and as a word device for controlling the present value.

*2 The number of points that can be actually used varies depending on intelligent function modules. For the number of buffer memory points, refer to the manual for the intelligent function module used.

*3 The setting range differs depending on the CPU module.
- Q03UDVCPU: 30K words
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: 40K words
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: 60K words

When changing device points, refer to Page 345, Section 4.2 (1), the parts describing the precaution.

*4 Available only in multiple CPU systems.

*5 Up to 15 digits can be entered in a programming tool.

*6 This is the total number of points for the file register, extended data register (D), and extended link register (W).
The number of points differs depending on the CPU module. (The following is the number of points when an extended SRAM cassette (8M bytes) is used.)
- Q03UDVCPU: 4192K points
- Q04UDVCPU, Q04UDPVCPU: 4224K points
- Q06UDVCPU, Q06UDPVCPU: 4480K points
- Q13UDVCPU, Q13UDPVCPU: 4608K points
- Q26UDVCPU, Q26UDPVCPU: 4736K points

*7 The number of points differs depending on the CPU module.
- Q03UDVCPU: 9216K points
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: 15360K points
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: 28672K points

*8 The setting range differs depending on the CPU module.
- Q03UDVCPU: M0 to M9215
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: M0 to M15359
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: M0 to M28671

*9 The number of points differs depending on the CPU module.
- Q03UDVCPU: 13312 points
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: 22528 points
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: 41984 points

*10 The setting range differs depending on the CPU module.
- Q03UDVCPU: D0 to D13311
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: D0 to D22527
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: D0 to D41983

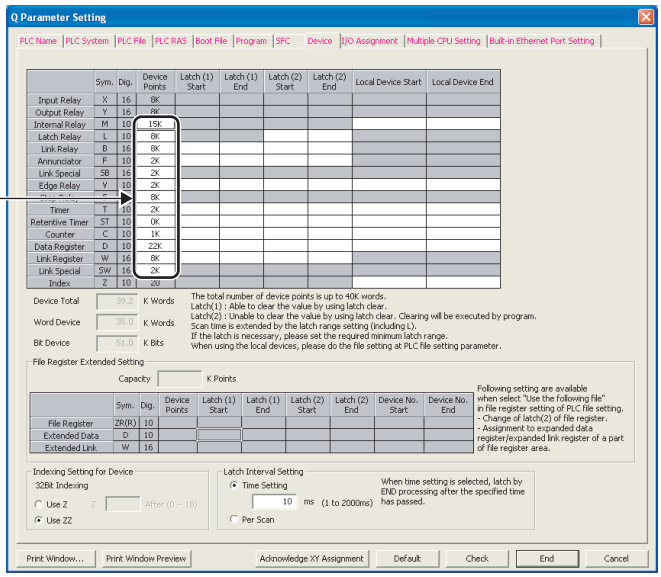
4.2 Internal User Devices

Internal user devices can be used for various user applications.

(1) Points for internal user devices

The device points can be changed in the Device tab of the PLC parameter dialog box.

Most of the default device points can be changed.



When changing device points, note the following.

- The number of points for the input (X), output (Y), and step relay (S) Note 4.1 Note 4.3 cannot be changed.
- Set points for each device in increments of 16.
- The total number of internal user device points differs depending on the CPU module.
  (Page 336, Section 4.1)
- The maximum number of points for bit devices is 32K.
  For the internal relay and link relay, the number of points can be set up to 60K. Note 4.2
- For the timer (T), retentive timer (ST), or counter (C), one point includes one point of a word device and two points of a bit device. (Page 346, Section 4.2 (2))

Note 4.1 Universal

For the Universal model QCPU whose serial number (first five digits) is "10042" or later, the step relay (S) points can be changed to 0K. (Page 466, Appendix 2)

Note 4.2 Universal

This applies to the Universal model QCPU whose serial number (first five digits) is "10042" or later.
(Page 466, Appendix 2)

Note 4.3 Universal

For the Universal model QCPU whose serial number (first five digits) is "12052" or later, the points for the step relay (S) can be set up to the following points in increments of 1K. (Page 466, Appendix 2)

•Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 8192 points
•Universal model QCPU other than the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 16384 points

Point

- When changing device points, the following refresh ranges must not exceed the corresponding device ranges.
  - Refresh range of network module
  - Refresh range of CC-Link IE Field Network Basic
  - Auto refresh range of intelligent function module

  If device points are set exceeding the corresponding device range, data may be written to any other device or an error may occur.
- The total number of points for the internal relay, latch relay, annunciator, edge relay, link relay, link special relay, step relay, timer, retentive timer, and counter must be set within the following range.
  - Serial number (first five digits) of the Universal model QCPU is "10041" or earlier: Up to 64K points
  - Serial number (first five digits) of the Universal model QCPU is "10042" or later: Not limited
- If the device points of the internal user devices are changed and the parameters are written from the "Write to PLC" screen, the device address may be shifted and does not correspond to the original stored value. Because the shifted value might be used for the operation, the following files, which are created by using the parameters before the device point change, cannot be used under existing condition.
  - Sequence program files
  - SFC program files
  - Structured text program files

  When change the device points of the internal user devices, perform the following operations from a programming tool.
  [Before changing the device points of the internal user devices]
  Read devices to be used and each program from the CPU module.
  [After the device points of the internal user devices are changed]
  Write the devices and each program, which were read before the device point change, to the CPU module.

  For the read/write of devices and programs, refer to the following.
  Operating manual for the programming tool used

(2) Memory size

Set the internal user devices so that the following condition is satisfied.

(Bit device size) + (Timer, retentive timer, and counter sizes) + (Word device size) ≤ Total number of internal user device points

(a) Bit device

For bit devices, 16 points are calculated as one word.

$$\text{(Bit device size)} = \frac{(X+Y+M+L+B+F+SB+V+S)}{16} \text{ words}$$

(b) Timer (T), retentive timer (ST), and counter (C)

For the timer (T), retentive timer (ST), and counter (C), 16 points are calculated as 18 words.

$$\text{(Timer, retentive timer, or counter size)} = \frac{(T+ST+C)}{16} \times 18 \text{ words}$$

(c) Word device

For the data register (D), link register (W), and link special register (SW), 16 points are calculated as 16 words.

$$\text{(Word device size)} = \frac{(D+W+SW)}{16} \times 16 \text{ words}$$

(3) Device point assignment example

The following table shows device point assignment examples based on the device point assignment sheet in Page 574, Appendix 10.

Device name	Symbol	Numeric notation	Number of device point[*2]		Restriction check			
			Points	Range	Size (words)[*3]		Points (bits)[*2]	
Input relay[*1]	X	Hexadecimal	8K (8192)	X0000 to X1FFF	/16	512	×1	8192
Output relay[*1]	Y	Hexadecimal	8K (8192)	Y0000 to Y1FFF	/16	512	×1	8192
Internal relay	M	Decimal	16K (16384)	M0 to M16383	/16	1024	×1	16384
Latch relay	L	Decimal	4K (4096)	L0 to L4095	/16	256	×1	4096
Link relay	B	Hexadecimal	4K (4096)	B0000 to B0FFF	/16	256	×1	4096
Annunciator	F	Decimal	1K (1024)	F0 to F1023	/16	64	×1	1024
Link special relay	SB	Hexadecimal	2K (2048)	SB0000 to SB07FF	/16	128	×1	2048
Edge relay	V	Decimal	1K (1024)	V0 to V1023	/16	64	×1	1024
Step relay[*1]	S	Decimal	8K (8192)	S0 to S8191	/16	512	×1	8192
Timer	T	Decimal	2K (2048)	T0 to T2047	$\times\frac{18}{16}$	2304	×2	4096
Retentive timer	ST	Decimal	2K (2048)	ST0 to ST2047	$\times\frac{18}{16}$	2304	×2	4096
Counter	C	Decimal	1K (1024)	C0 to C1023	$\times\frac{18}{16}$	1152	×2	2048
Data register	D	Decimal	14K (14336)	D0 to D14335	×1	14336	-	
Link register	W	Hexadecimal	4K (4096)	W0000 to W4095	×1	4096	-	
Link special register	SW	Hexadecimal	2K (2048)	SW0000 to SW07FF	×1	2048	-	
Total						29568 (29696 or less)		63488 (65536 or less)

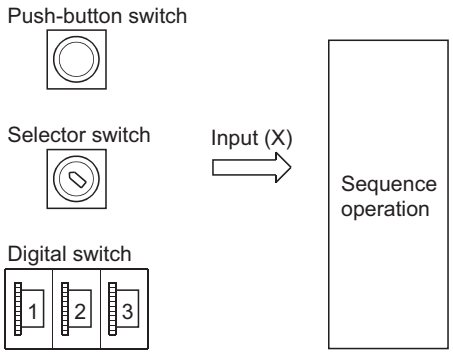
*1 The points are fixed for the system. (Cannot be changed)
The points for the step relay can be changed to 0 if the Universal model QCPU whose serial number (first five digits) is "10042" or later.
For the Universal model QCPU whose serial number (first five digits) is "12052" or later, a step relay can be set in increments of 1k point and up to the following points. (Page 466, Appendix 2)
- Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 8192 points
- Universal model QCPU other than the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 16384 points

*2 Up to 32K points can be set for each device.
However, up to 60K points can be set for each device of the internal relay and link relay if the Universal model QCPU whose serial number (first five digits) is "10042" or later,

*3 Enter the values multiplied (or divided) by the number shown in the Size (words) column.

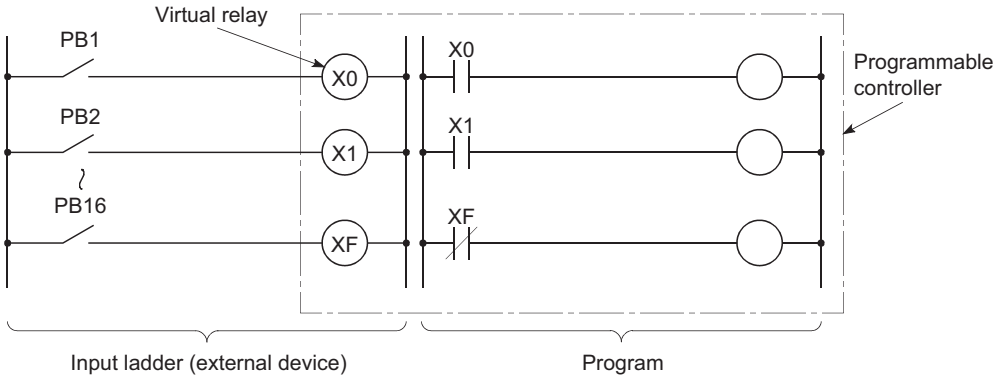
4.2.1 Input (X)

The input (X) is used to send commands or data to the CPU module from external devices such as push-button switches, selector switches, limit switches, and digital switches.



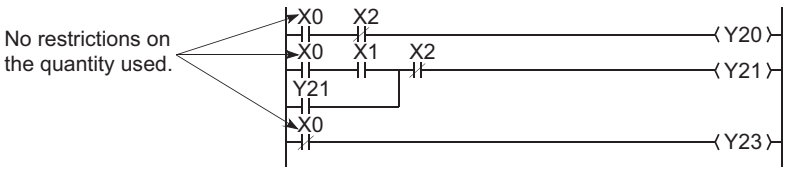
(1) Concept of input (X)

One input point is assumed to be a virtual relay Xn in the CPU module.
Programs use the normally open or closed contact of Xn.



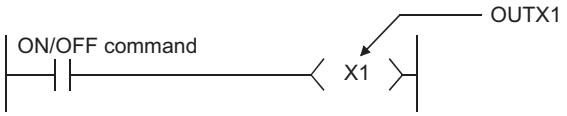
(2) Allowable number of normally open or closed contacts

There are no restrictions on the number of normally open or closed contacts of Xn used in a program, as long as the program capacity is not exceeded.



Point

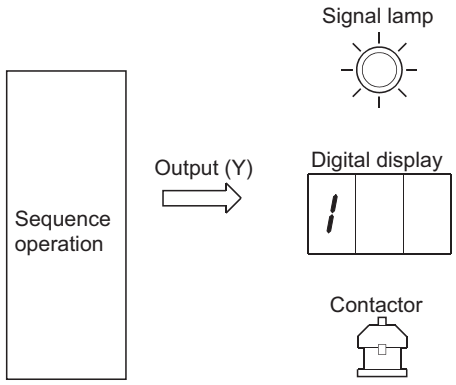
- When debugging a program, the input (X) can be set to on or off by the following:
  - Device test using a programming tool
  - OUT Xn instruction



- The input (X) can also be used for the following.
  - Refresh target device (CPU module side) of RX in CC-Link IE Field Network, CC-Link IE Field Network Basic, or CC-Link
  - Refresh target device (CPU module side) of CC-Link IE Controller Network or MELSECNET/H

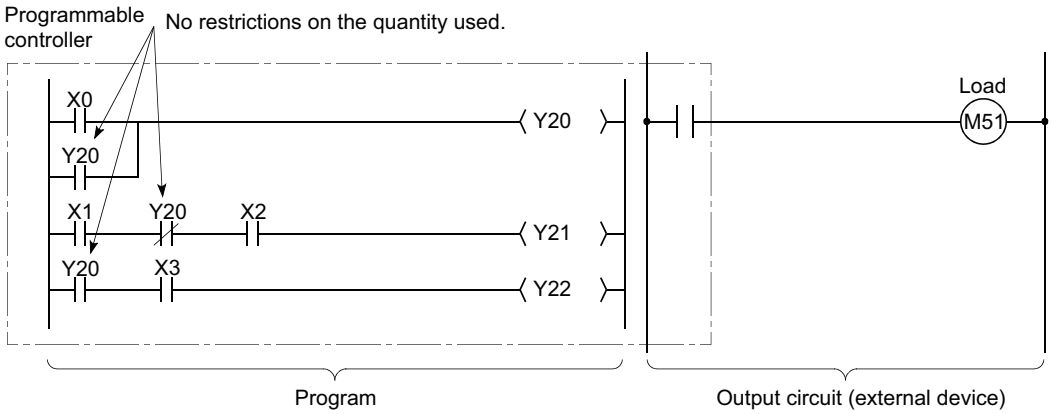
4.2.2 Output (Y)

The output (Y) is used to output control results on programs to external devices such as signal lamps, digital displays, electromagnetic switches (contactors), or solenoids.
Data can be output to the outside like using a normally open contact.



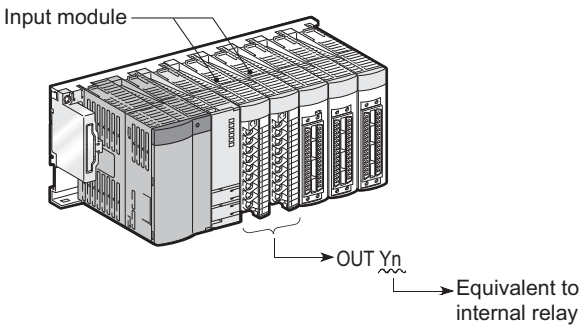
(1) Allowable number of normally open or closed contacts

There are no restrictions on the number of normally open or closed contacts of Yn used in a program, as long as the program capacity is not exceeded.



(2) Using the output (Y) as the internal relay (M)

The output (Y) corresponding to the slots for input modules or empty slots can be utilized as the internal relay (M).



4.2.3 Internal relay (M)

The internal relay (M) is a device for auxiliary relays used in the CPU module.
All of the internal relays are set to off in the following cases:

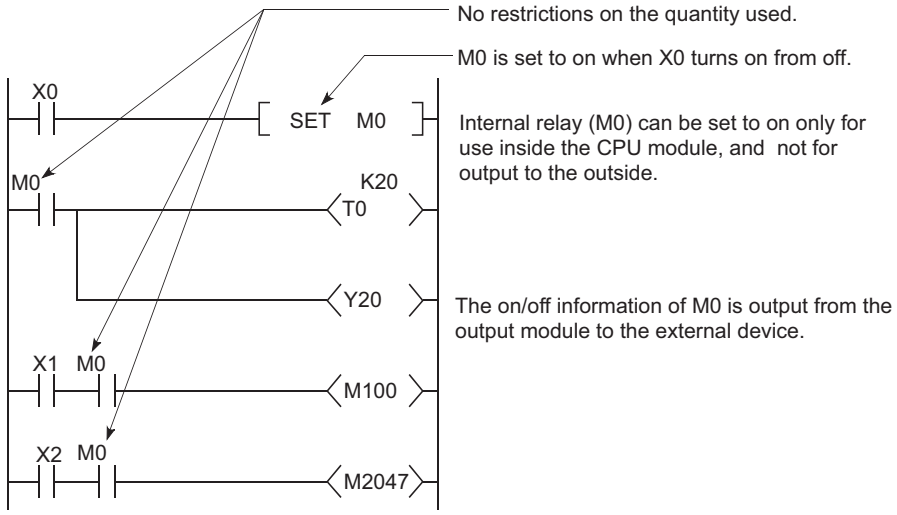
- When the CPU module is powered on from off
- When the CPU module is reset
- When latch clear is executed (☞ Page 124, Section 3.3 (7))

(1) Latch (data retention during power failure)

The internal relay cannot be latched.

(2) Allowable number of normally open or closed contacts

There are no restrictions on the number of normally open or closed contacts used in a program, as long as the program capacity is not exceeded.



(3) Method for external output

The output (Y) is used to output sequence program operation results to external devices.

Point

Use the latch relay (L) when latch (data retention during power failure) is required. (☞ Page 352, Section 4.2.4)

4.2.4 Latch relay (L)

The latch relay (L) is a device for auxiliary relays that can be latched inside the CPU module.
Latch relay data are retained by batteries in the CPU module during power failure.
Operation results (on/off information) immediately before the following will be also retained.

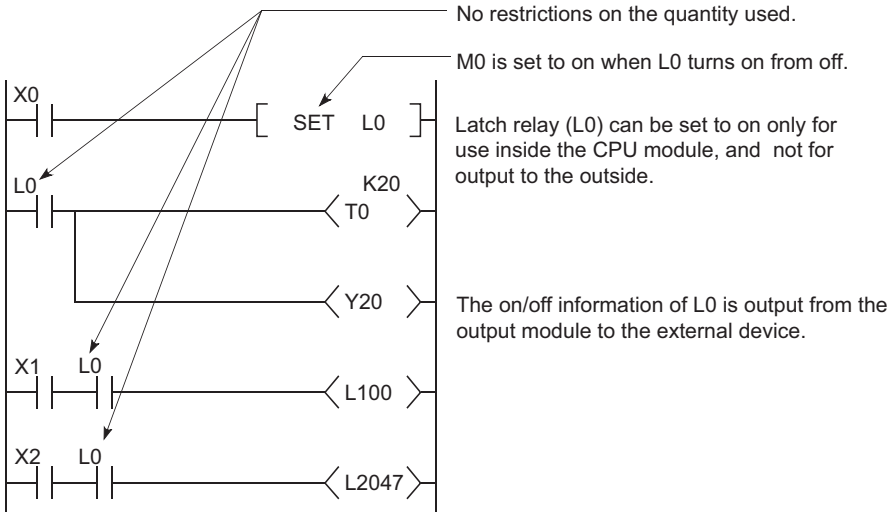
- Powering off and then on the CPU module
- Resetting the CPU module

(1) Latch relay clear

The latch relay is turned off by the latch clear operation. (☞ Page 124, Section 3.3 (7))
However, the latch relay set in "Latch (2) Start/End" in the Device tab of the PLC parameter dialog box cannot be turned off even by a latch clear operation.

(2) Allowable number of normally open or closed contacts

There are no restrictions on the number of normally open or closed contacts used in a program, as long as the program capacity is not exceeded.



Point

Scan time is prolonged when the latch relay (L) is used. Reducing the points of latch relay (L) can reduce the prolonging scan time. (☞ Page 478, Appendix 3.2 (6))

(3) Method for external output

The output (Y) is used to output sequence program operation results to external devices.

Point

- If latch is not required, use the internal relay (M). (☞ Page 351, Section 4.2.3)
- The latch clear invalid area is set in the Device setting of PLC parameter. (☞ Page 123, Section 3.3 (4))

4.2.5 Annunciator (F)

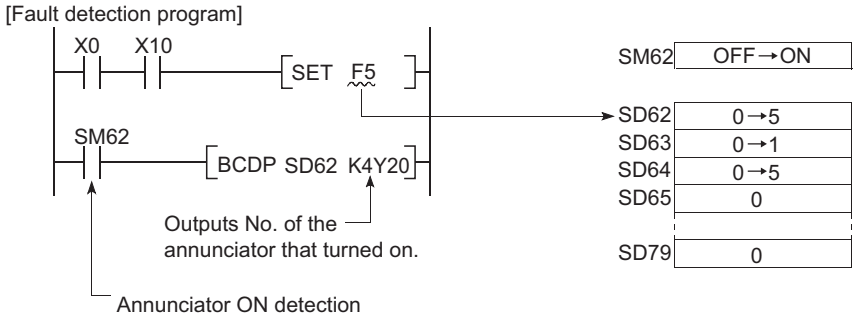
The annunciator (F) is an internal relay that can be effectively used in fault detection programs for a user-created system. When any annunciator turns on, SM62 turns on, and the number of annunciators turned on and the corresponding numbers are stored in SD62 to SD79. The annunciator number stored in SD62 is also registered to the error history area.

Special relay/special register	Description
SM62	Turns on even if only one of the annunciator number areas is turned on.
SD62	Stores the number of the annunciator that was turned on first.
SD63	Stores the quantity of the annunciator number areas that are on.
SD64 to SD79	Stores annunciator numbers in the order of turning on. (The same annunciator number is stored in SD62 and SD64.)

(1) Applications of the annunciator

Using the annunciator in a fault detection program allows check for a system fault and identification of the fault (annunciator number) by monitoring the special register (SD62 to SD79) when the special relay (SM62) turns on.

Ex. In this program, when annunciator (F5) is turned on, the corresponding annunciator number is output to the outside.



(2) Allowable number of normally open or closed contacts

There are no restrictions on the number of normally open or closed contacts used in a program, as long as the program capacity is not exceeded.

(3) Turning on the annunciator and processing

(a) Turning on the annunciator

The following instructions can be used.

- SET F□ instruction

  The SET F□ instruction can be used to turn on the annunciator only on the leading edge (off to on) of an input condition. Even if the input condition turns off, the annunciator is held on. Using many annunciator numbers can shorten scan time more than using the OUT F□ instruction.

- OUT F□ instruction

  The OUT F□ instruction can be also used to turn on or off the annunciator. However, since the processing is performed for every scan, the scan time is longer than the case of using the SET F□ instruction. In addition, execution of the RST F, LEDR, or BKRST instruction is required after the annunciator is turned off with the OUT F□ instruction. Therefore, use of the SET F□ instruction is recommended.

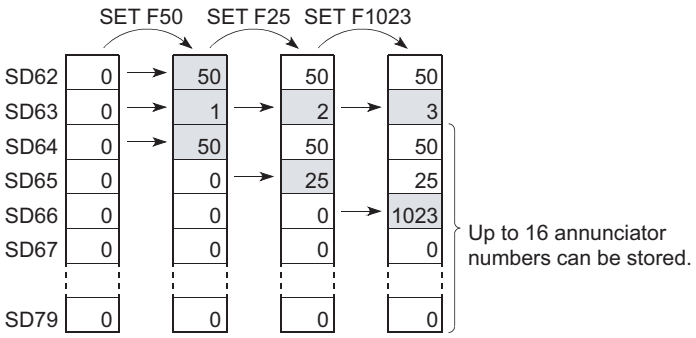
Point

If the annunciator is turned on with any instruction other than SET F□ and OUT F□ (for example, the MOV instruction), the same operation as the internal relay (M) is performed.
The ON information is not stored in SM62, and annunciator numbers are not stored in SD62 and SD64 to SD79.

(b) Processing after annunciator on

1. Data stored in the special register (SD62 to SD79)

- Turned-on annunciator numbers are stored in SD64 to SD79 in order.
- The annunciator number in SD64 is stored in SD62.
- SD63 value is incremented by "1".



2. Processing on the CPU

The USER LED on the front side turns on (red).

3. On/off setting for the LED

Whether to turn on the USER.LED or not when the annunciator is turned on can be set in the LED indication setting. (☞ Page 223, Section 3.20.2).

(4) Turning off the annunciator and processing

(a) Turning off the annunciator

The following instructions can be used.

- RST F□ instruction

  This is used to turn off the annunciator number that was turned on with the SET F□ instruction.
- LEDR instruction

  This is used to turn off the annunciator number stored in SD62 and SD64.
- BKRST instruction

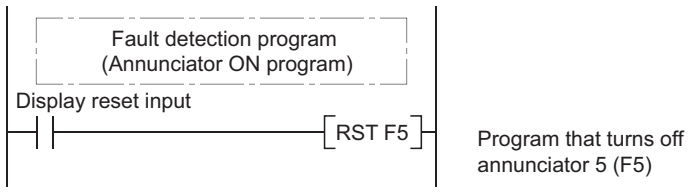
  This is used to turn off all of the annunciator numbers within the specified range.
- OUT F□ instruction

  One annunciator number can be turned on or off with the same instruction.

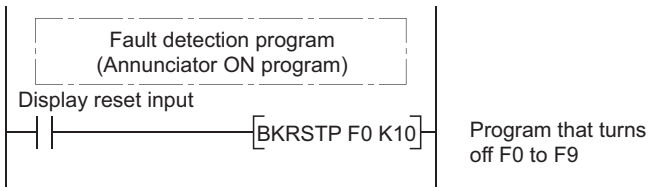
  However, even if an annunciator number is turned off with the OUT F□ instruction, the off processing described in Page 356, Section 4.2.5 (4) (b) is not performed.

  If the annunciator is turned off with the OUT F□ instruction, execution of the RST F□, LEDR, or BKRST instruction is required.

Ex. Turning off annunciator 5 (F5)



Ex. Turning off all of the turned-on annunciator numbers



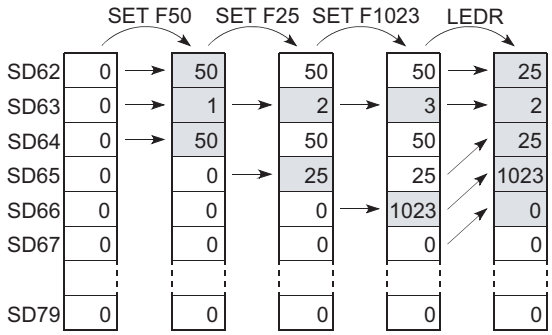
For details of each instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

(b) Processing after annunciator off

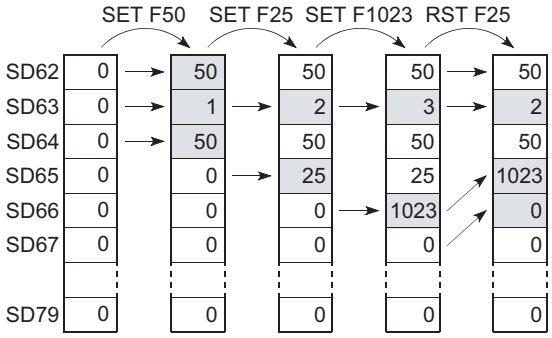
1. Data stored in the special register (SD62 to SD79) after execution of the LEDR instruction

- The annunciator number in SD64 is deleted, and the other annunciator numbers in the register addressed SD65 and after are shifted accordingly.
- The annunciator number in SD64 is stored into SD62.
- SD63 value is decremented by "1".
- If the SD63 value is changed to "0", SM62 is turned off.



2. Data stored in the special register (SD62 to SD79) when the annunciator is turned off with the RST F□ or BKRST instruction

- The annunciator number specified with the RST or BKRST instruction is deleted, and the other annunciator numbers in the register addressed SD65 and after are shifted accordingly.
- If the existing annunciator number in SD64 is turned off, a new annunciator number stored in SD64 will be stored in SD62.
- SD63 value is decremented by "1".
- If the SD63 value is changed to "0", SM62 is turned off.



3. LED indication

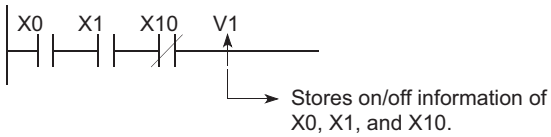
When all of the annunciator numbers in SD64 to SD79 turn off, the LED that was turned on by turn-on of the annunciator will turn off. (☞ Page 354, Section 4.2.5 (3) (b))

Point

If the LEDR instruction is executed while the annunciator is on and at the same time the operation continuation error that has higher priority (☞ Page 223, Section 3.20.2) than the annunciator has occurred, the LEDR instruction clears the higher priority error. Because of this, the annunciator is not turned off by execution of the LEDR instruction.
To turn off the annunciator with the LEDR instruction, remove the error whose priority is higher than that of the annunciator.

4.2.6 Edge relay (V)

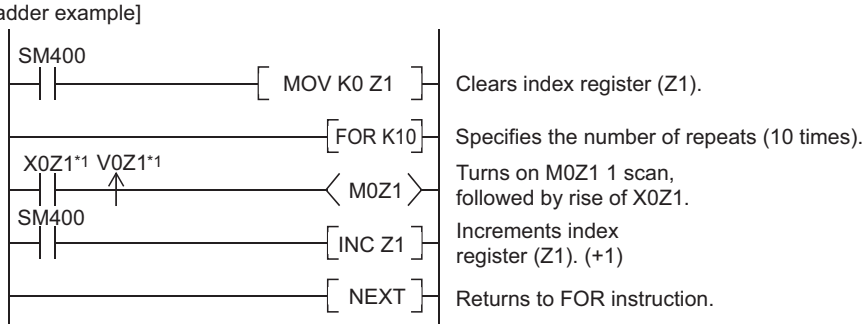
The edge relay (V) is a device in which the on/off information of contacts from the beginning of the ladder block is memorized. The device can be used only as contacts (cannot be used as coils).



(1) Applications of the edge relay

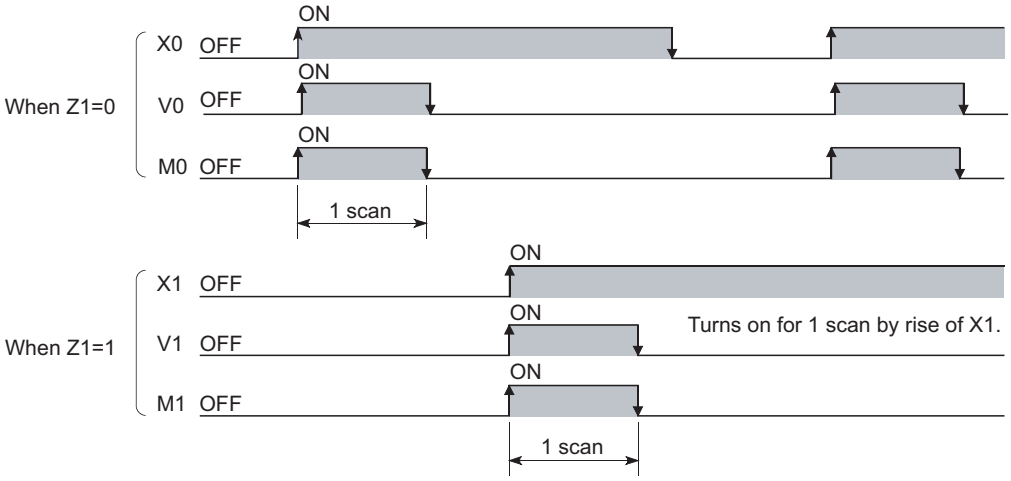
The edge relay can be utilized to detect the leading edge (off to on) in programs configured using index modification.

[Ladder example]



*1 The on/off information for X0Z1 is stored in the V0Z1 edge relay.
For example, the on/off information of X0 is stored in V0, and that of X1 is stored in V1.

[Timing chart]

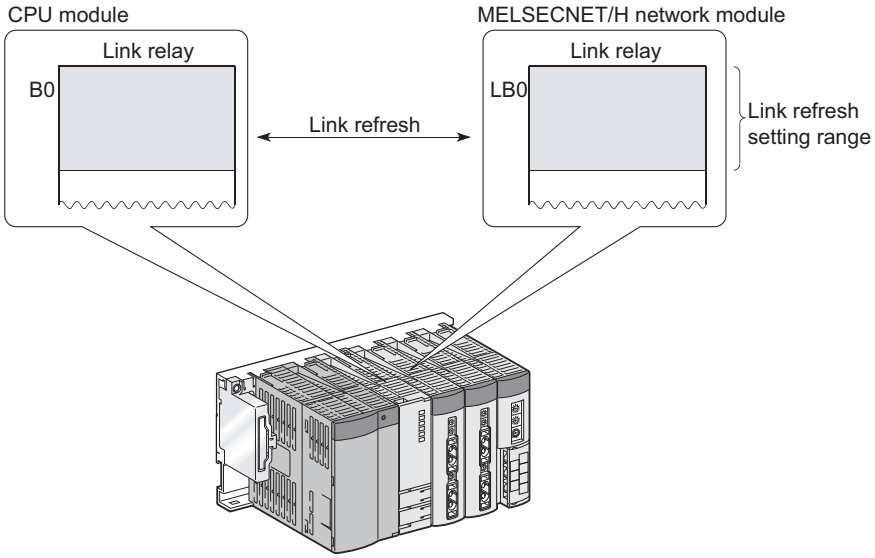


(2) Precautions

The edge relay of the same number cannot be used more than one time in a program.

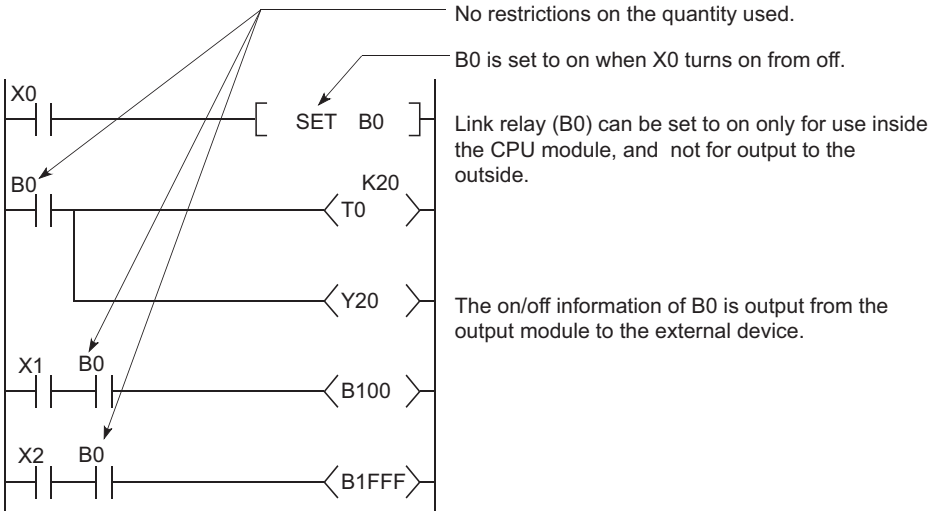
4.2.7 Link relay (B)

The link relay (B) is a relay on the CPU module side, and it is used for refreshing the link relay (LB) data of another module such as a MELSECNET/H network module to the CPU module or refreshing the CPU module data to the link relay (LB) of the MELSECNET/H network module.



(1) Allowable number of normally open or closed contacts

There are no restrictions on the number of normally open or closed contacts used in a program, as long as the program capacity is not exceeded.



(2) Using the link relay in the network system

Network parameters must be set.

The link relay range areas that are not set by network parameters (not used for a network system such as a MELSECNET/H network) can be used as the internal relay or latch relay.

- Link relay range where no latch is performed••••••Internal relay
- Link relay range where latch is performed••••••••••Latch relay

Point

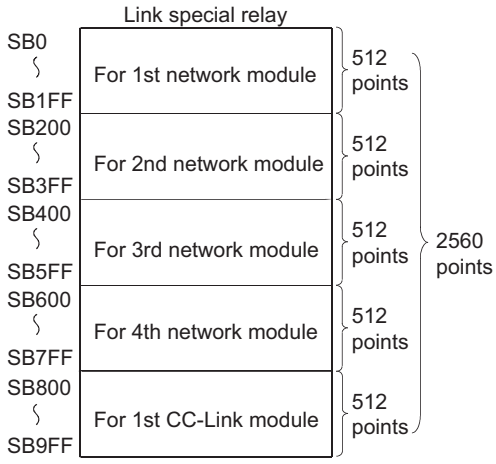
To use the link device in the network module exceeding link relay points of the CPU module (default: 8192 points), change the link relay points in the Device tab of the PLC Parameter dialog box.

4.2.8 Link special relay (SB)

The Link special relay (SB) is a relay that indicates various communication status and detected errors of intelligent function modules such as CC-Link IE module or MELSECNET/H module.
Each of this device area is turned on or off according to a factor occurred during data link.
The communication status and errors on the network can be confirmed by monitoring the link special relay (SB).

(1) Number of link special relay points

The points for the link special relay in the CPU module is 2048. (SB0 to SB7FF).
However, the points can be changed in the Device tab of the PLC parameter dialog box. (☞ Page 447, Appendix 1.2.8) To an intelligent function module that has a link special relay, such as a CC-Link IE module or MELSECNET/H module, 512 points are assigned. Assigning the link special relay as shown in below allows refresh of the CC-Link link special relay (SB) to the link special relay (SB) of the CPU module.



For details of the link special relay, refer to the manual for an intelligent function module that has the link special relay.

4.2.9 Step relay (S)

This device is provided for SFC programs. For how to use the step relay, refer to the following manual.

MELSEC-Q/L/QnA Programming Manual (SFC)

Point

Because the step relay is a device exclusively used for SFC programs, it cannot be used as an internal relay in the sequence program.
If used, an SFC error will occur, and the system may go down.

4.2.10 Timer (T)

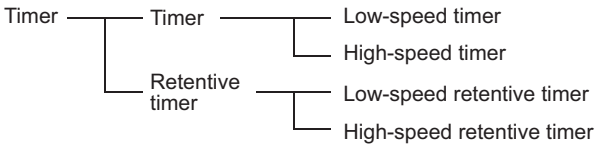
Time counting starts when a coil is turned on, and it times out and the contact turns on when the current value reaches the set value.
The timer is of an incremental type.

(1) Timer types

Timers are mainly classified into the following two types.

1) Timer of which value is set to 0 when the coil is turned off.
2) Retentive timer that holds the current value even if the coil is turned off.

Also, low-speed and high-speed timers are included in timer 1), and low-speed and high-speed retentive timers are included in timer 2).



(2) Specification of the timer

- The same device is used for the low- and high-speed timers, and the type is determined according to the instruction used.

Ex. For the OUT T0 instruction, the low-speed timer is specified, and for the OUTH T0 instruction, the high-speed timer is specified.

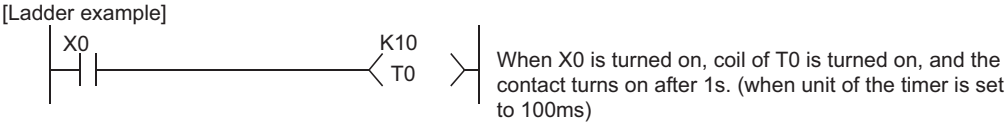
- The same device is used for the low- and high-speed retentive timers, and the type is determined according to the instruction used.

Ex. For the OUT ST0 instruction, the low-speed retentive timer is specified, and for the OUTH ST0 instruction, the high-speed retentive timer is specified.

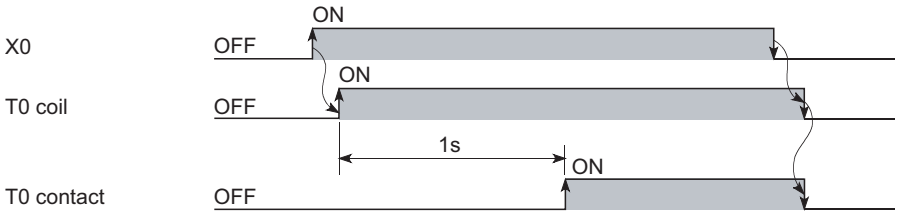
(3) Low-speed timer

This type of timer measures time in increments of 1 to 1000ms.
The timer starts time measurement when its coil is turned on, and when it times out, the contact is turned on. If the timer's coil is turned off, the current value is changed to "0" and the contact is turned off.

[Ladder example]

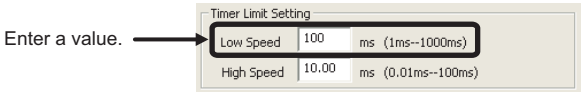


[Timing chart]



(a) time increment setting

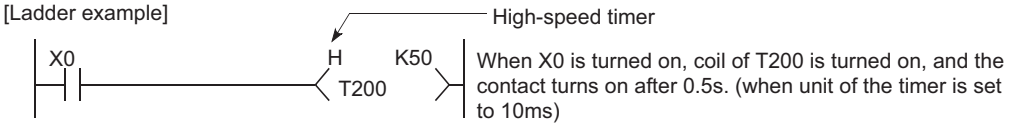
The time increment is set in the PLC system tab of the PLC parameter dialog box.
The default is 100ms, and it can be changed in increments of 1ms.



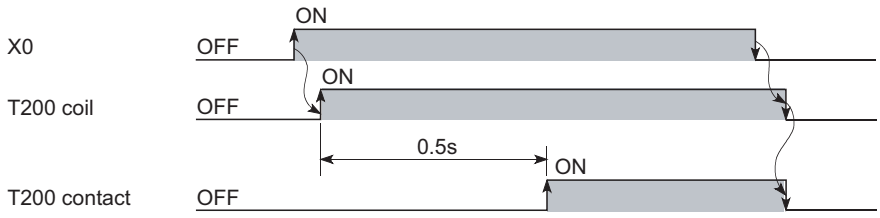
(4) High-speed timer

This type of timer measures time in increments of 0.01 to 100ms.
The timer starts time measurement when its coil is turned on, and when it times out, the contact is turned on. If the timer's coil is turned off, the current value is changed to "0" and the contact is turned off.

[Ladder example]

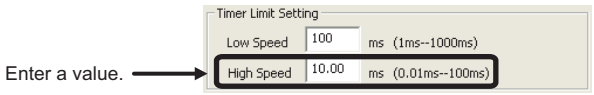


[Timing chart]



(a) Time increment setting

The time increment is set in the PLC system tab of the PLC parameter dialog box.
The default is 10.0ms, and it can be changed in increments of 0.01ms.



(5) Retentive timer

This timer measures the period of time during which the coil is on.

The timer starts time measurement when its coil is turned on, and when it times out, the contact is turned on.

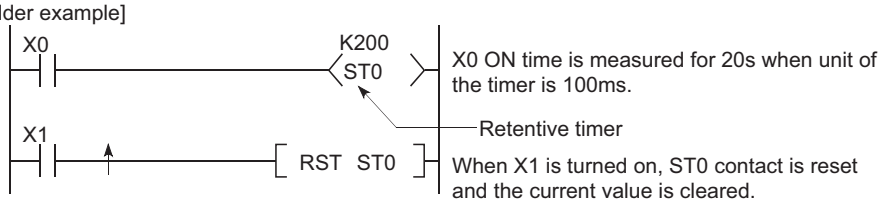
Even if the timer's coil is turned off, the current value and the on/off status of the contact are retained.

When the coil is turned on again, the measurement restarts from the retained current value.

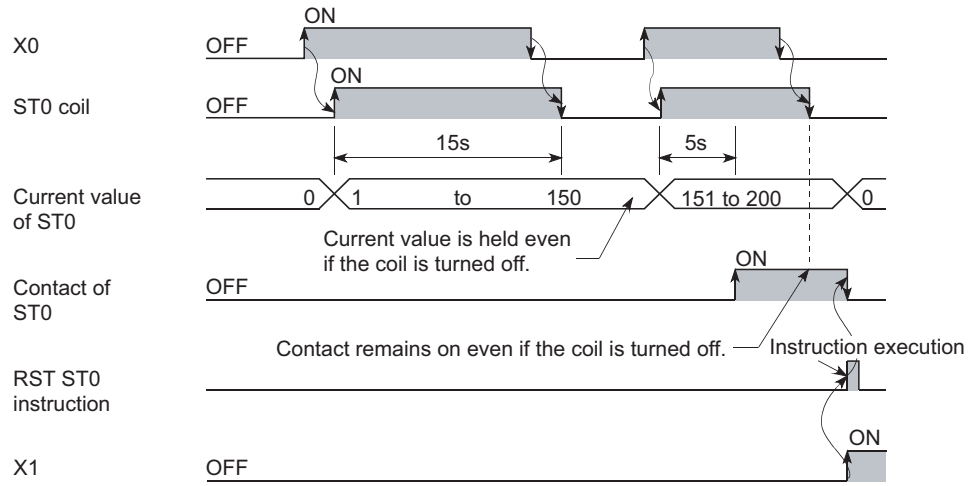
(a) Retentive timer clear

The current value and the contact off status can be cleared with the RST ST□ instruction.

[Ladder example]



[Timing chart]



(b) Time increment setting

The time increment is set in the same manner as the corresponding low- or high-speed timer.

- Low-speed retentive timer: Low-speed timer
- High-speed retentive timer: High-speed timer

Point

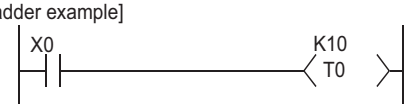
To use a retentive timer, set the points for it in the Device tab of the PLC parameter dialog box.

(6) Timer processing and accuracy

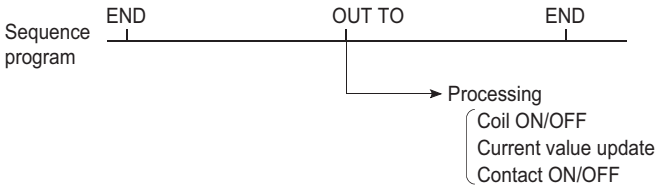
(a) Processing

When the OUT T□ or OUT ST□ instruction is executed, the on/off switching of the timer coil, current value update, and on/off switching of the contact are performed.
In the END processing, the current timer value is not updated and the contact is not turned on/off.

[Ladder example]



[Processing at execution of OUT T0 instruction]

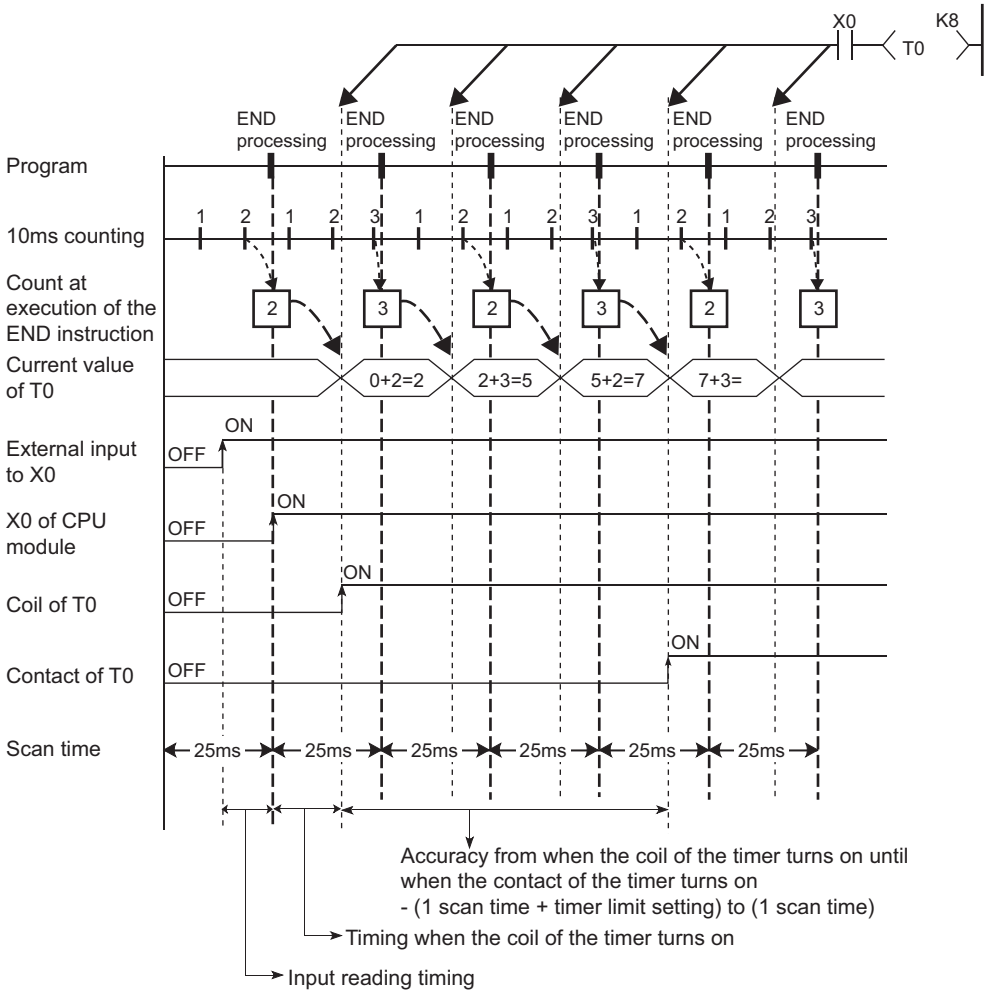


(b) Accuracy

The value obtained by the END instruction is added to the current value when the OUT T□ or OUT ST□ instruction is executed.

The current value is not updated while the timer coil is off even if the OUT T□ or OUT ST□ instruction is executed.

Timer limit setting=10ms, Setting value of T0=8 (10ms×8=80ms), Scan time=25ms



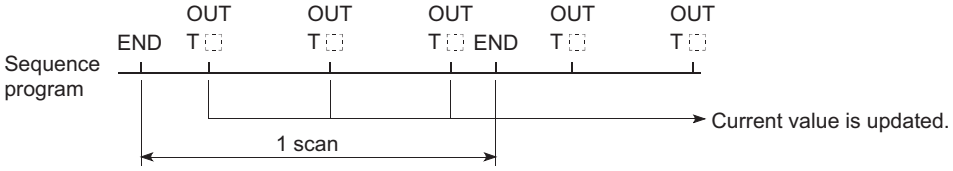
Accuracy of the timer response that is from reading input (X) to output the data is up to "2-scan time + timer limit setting".

(7) Precautions for using timers

(a) Use of the same timer

Do not use the OUT T□ instruction that describes the same timer more than once within one scan.

If this occurs, the current timer value will be updated by each OUT T□ instruction execution, resulting in incorrect time measurement.



(b) When the timer is not executed in every scan

While a coil of a timer (for example. T1) is on, do not make the OUT T1 instruction jumped to any other part with another instruction such as CJ.

If jump of the OUT T□ instruction has occurred, the current timer value is not updated.

Also, if a timer exists in a subroutine program, execute a subroutine call including the OUT T1 instruction once in each scan while the coil of the timer (for example, T1) is on.

Failure to do so will not update the current timer value.

(c) Programs that cannot use timers

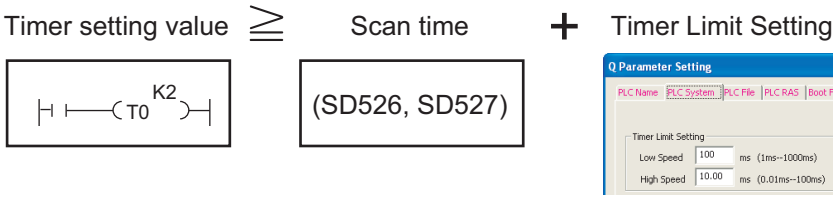
Timers cannot be used in interrupt programs and fixed scan execution programs.

(d) When the set value is 0

The contact turns on when the OUT T□ instruction is executed.

(e) Timer setting value and timer limit setting

Set the timer to meet the following condition:

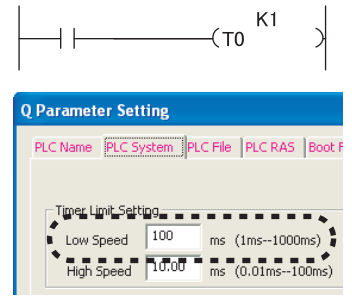


If the values are set to become "Timer setting value < Scan time + Timer Limit Setting", the coil and the contact might be simultaneously turned on depending on the timing on which the coil is turned on.

If the setting does not meet the above condition, make the value of the timer limit setting smaller to meet the condition.

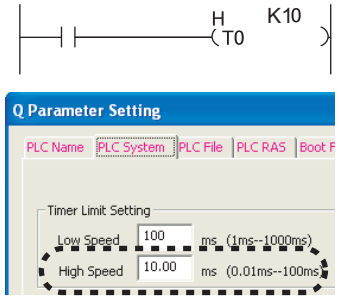
Ex. Make the value of the timer limit setting smaller by changing from low speed timer to high speed timer. (Assume that the scan time is 20ms.)

Before change (low-speed timer)



Timer setting value (100ms × 1=100ms) < Scan time (20ms) + Timer Limit Setting (100ms)

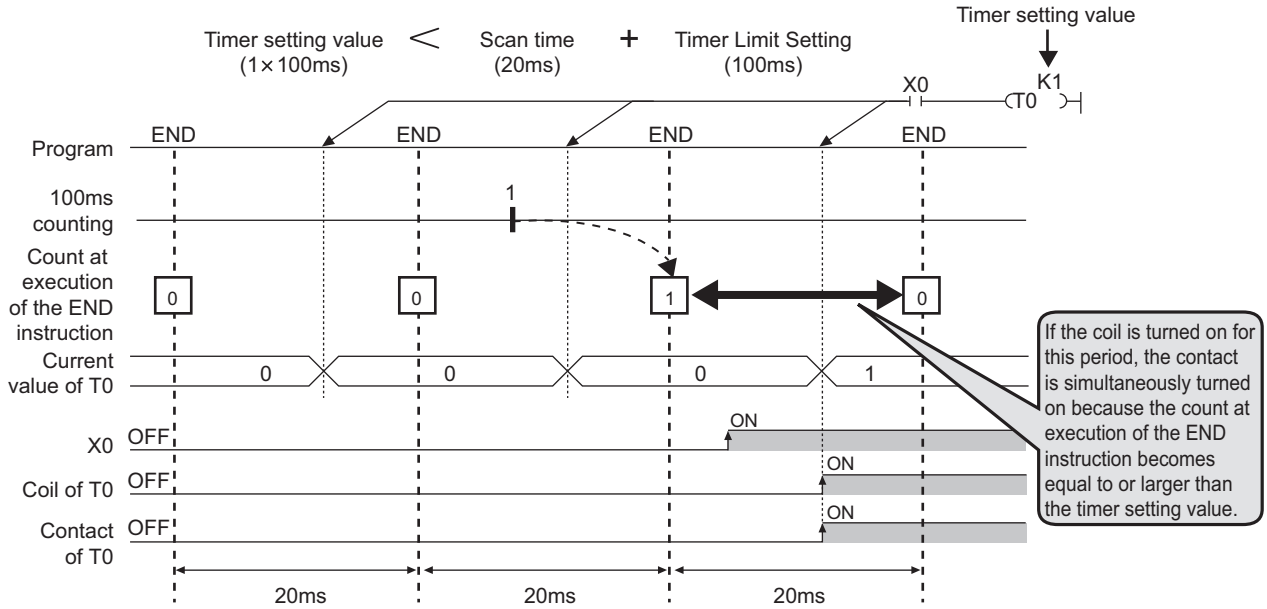
After change (high-speed timer)



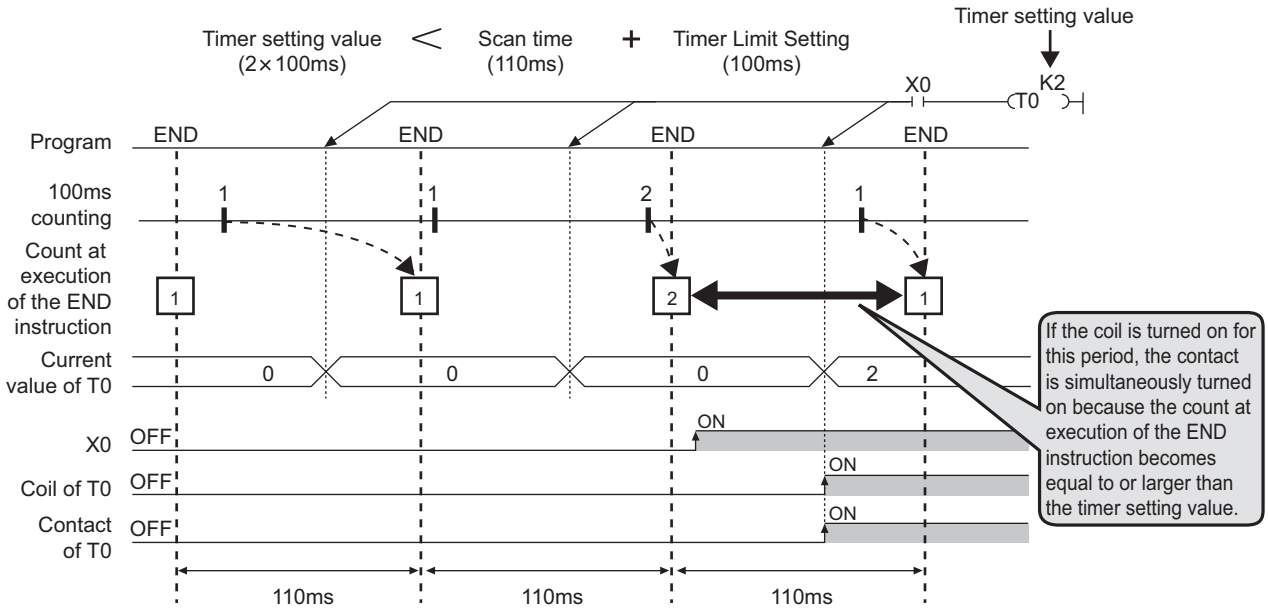
Timer setting value (10.00ms × 10=100ms) ≧ Scan time (20ms) + Timer Limit Setting (10ms)

The following show the examples of the coil and the contact being simultaneously turned on if the values are set to become "Timer setting value < Scan time + Timer Limit Setting":

Ex. When the timer setting value is 1 (1 × 100ms), the scan time is 20ms, and the timer limit setting is 100ms If the coil of the timer (T0) is turned on at the next scan after the values satisfy "Count at execution of the END instruction ≥ Timer setting value", the coil and the contact are simultaneously turned on because the values satisfy "Timer current value = Timer setting value" at the start of the timer.



Ex. When the timer setting value is 2 (2 × 100ms), the scan time is 110ms, and the timer limit setting is 100ms If the coil of the timer (T0) is turned on at the next scan after the values satisfy "Count at execution of the END instruction ≥ Timer setting value", the coil and the contact are simultaneously turned on because the values satisfy "Timer current value = Timer setting value" at the start of the timer.



(f) When the set value is changed after time-out

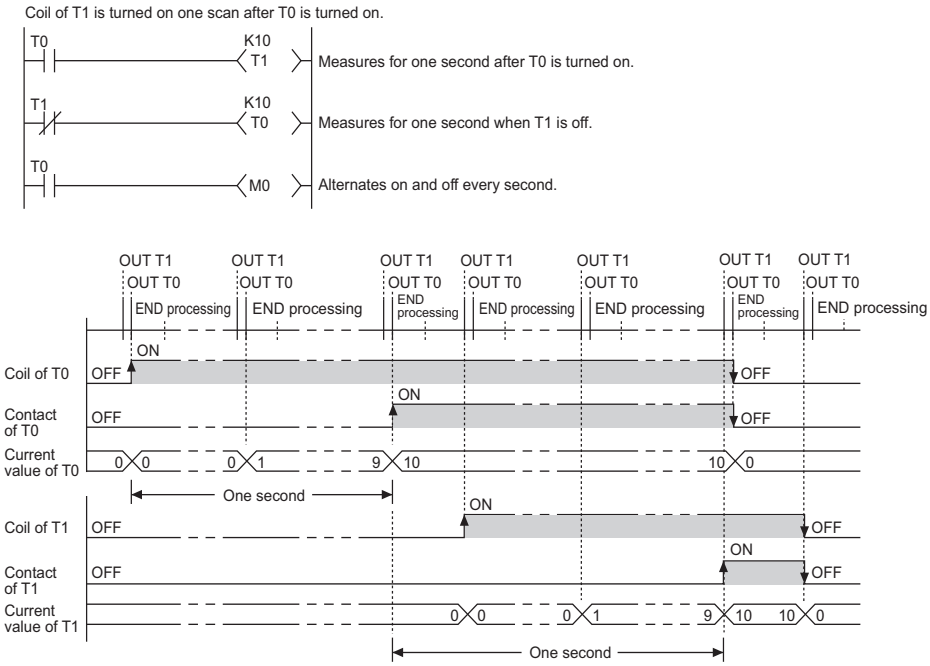
Even if the set value is changed to a larger value after time-out of the timer, the timer remains timed-out and does not start the operation.

(g) When using multiple timers

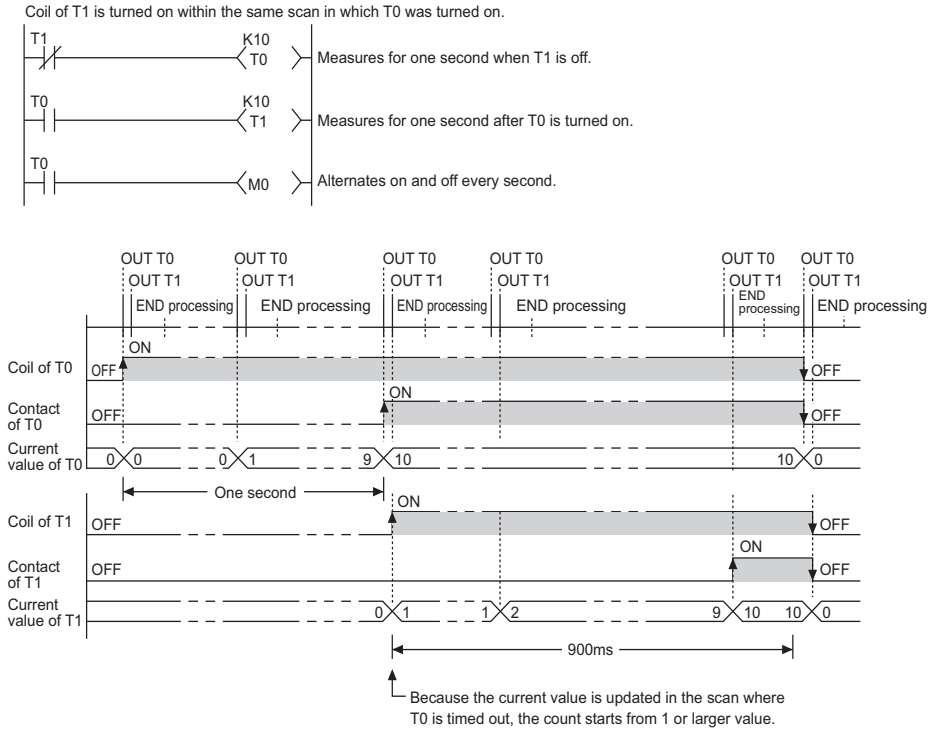
When using multiple timers to update the respective current values at execution of each OUT T□ instruction, pay attention to the ladder sequence.

Ex. Creating an on/off ladder using two timers

[Correct ladder example]



[Incorrect ladder example]



4.2.11 Counter (C)

The counter (C) is a device that counts the number of rises for input conditions in sequence programs.
When the count value matches the set value, the counting stops and its contact is turned on.
The counter is of an incremental type.

(1) Counter type

The following counter is available.

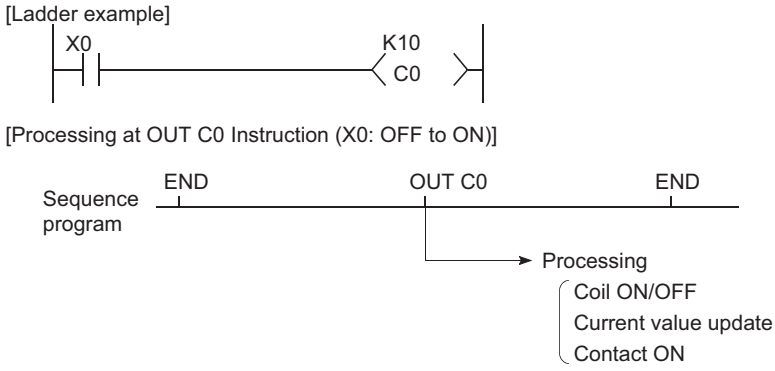
- Counter that counts the number of rises for input conditions in sequence programs

(2) Counting

(a) When the OUT C□ instruction is executed

The coil of the counter is turned on/off, the current value is updated (the count value + 1), and the contact is turned on.

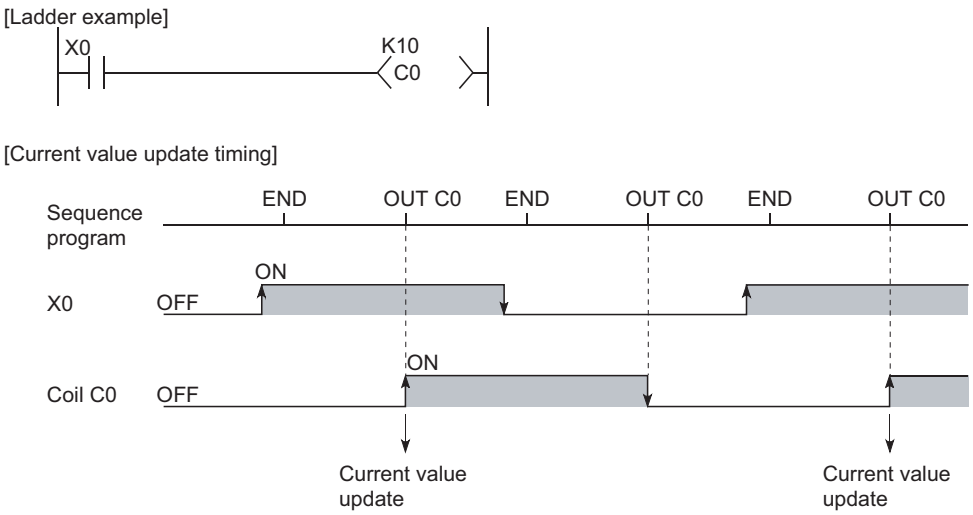
In the END processing, the current counter value is not updated and the contact is not turned on.



(b) Current value update (count value + 1)

The current value is updated (count value + 1) at the leading edge (OFF → ON) of the OUT C□ instruction.

The current value is not updated while the coil is off, or when it remains on or turns off from on by the OUT C□ instruction.



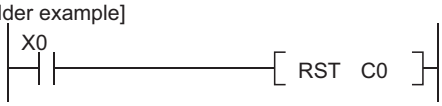
(c) Resetting the counter

The current counter value is not cleared even if the OUT C□ instruction is turned off.

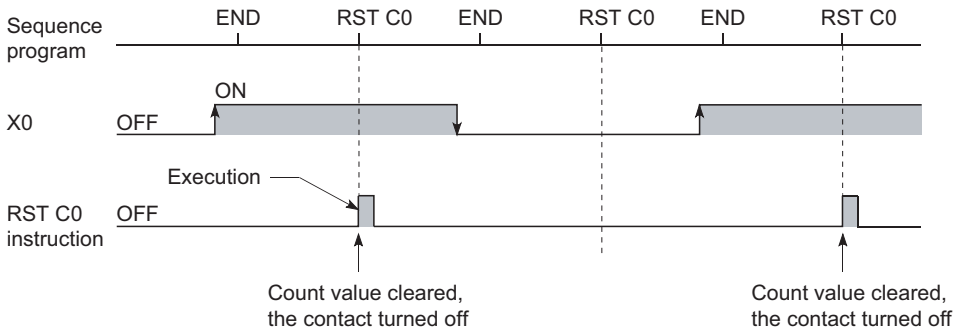
To clear the current value and to turn off the contact of the counter, use the RST C□ instruction.

At the time of execution of the RST C□ instruction, the counter value is cleared, and the contact is also turned off.

[Ladder example]



[Counter reset timing]

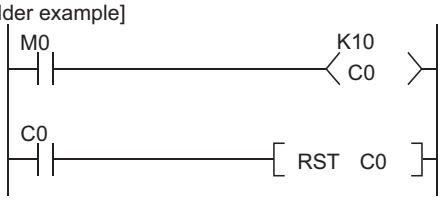


[Precautions for resetting the counter]

Execution of the RST C□ instruction also turns off the coil of C□.

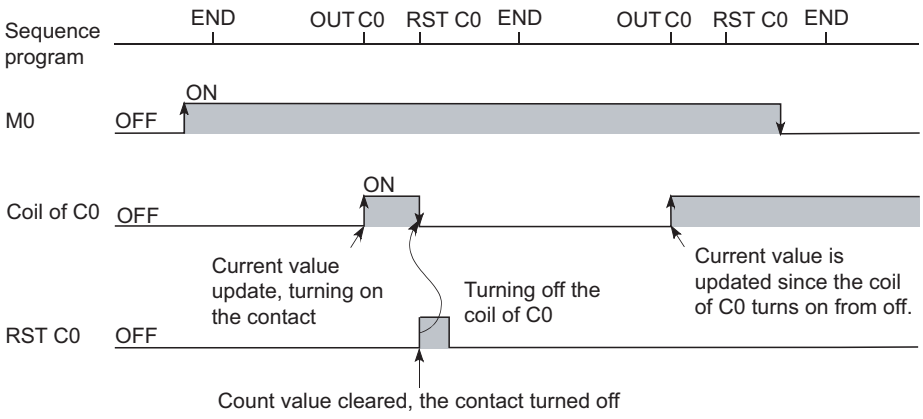
If the execution condition for the OUT C□ instruction is still ON after execution of the RST C□ instruction, turn on the coil of C□ at execution of the OUT C□ instruction and update the current value (count value + 1).

[Ladder example]



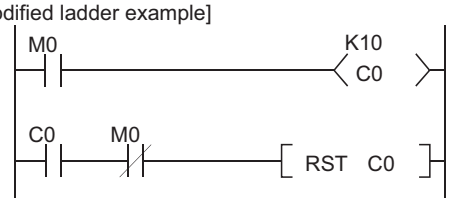
In the above ladder example, when M0 turns on from off, the coil of C0 turns on, updating the current value. When C0 reaches the preset value finally, the contact of C0 turns on, and execution of the RST C0 instruction clears the current value of C0. At this time, the coil of C0 also turns off.

If M0 is still on in the next scan, the current value is updated since the coil of C0 turns on from off at execution of the OUT C0 instruction. (The current value is changed to 1.)



To prevent the above, it is recommended to add a normally closed contact of the OUT C0 instruction execution to the condition for the RST C0 instruction execution so that the coil of C0 does not turn off while the execution condition (M0) of the OUT C0 instruction is on.

[Modified ladder example]



(d) Maximum counting speed

The counter can count only when the on/off time of the input condition is longer than the execution interval of the corresponding OUT C□ instruction.

The maximum counting speed is calculated by the following expression:

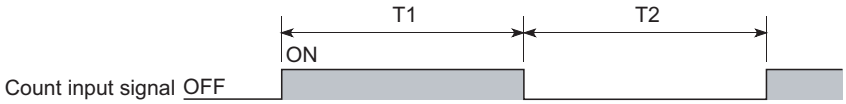
$$\text{Maximum counting speed (Cmax)} = \frac{n}{100} \times \frac{1}{T} \text{[times/s]}$$

- n: Duty (%)*1
- T: Execution interval of the OUT C□ instruction (sec)

*1 Duty (n) is the ON-OFF time ratio of count input signal, and is expressed as a percentage value

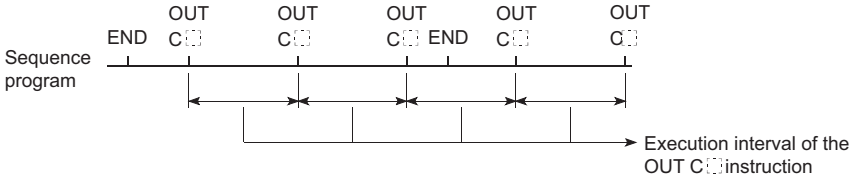
$$\text{When } T1 \geqq T2,\ n = \frac{T2}{T1+T2} \times 100\%$$

$$\text{When } T1 < T2,\ n = \frac{T1}{T1+T2} \times 100\%$$



Point

The maximum counting speed can be increased by placing multiple counters within one scan. At this time, use the direct access input (DX□) (Page 80, Section 2.8.2) for the counter input signal.



(3) Precautions

(a) When counting processing is suspended

If an interrupt occurs during execution of the processing shown below, counting is suspended until the execution of each processing is completed.

- Each instruction on the sequence program
- Interrupt program
- Fixed scan execution type program

Upon completion of the processing, the counting restarts.

However, if the same interrupt occurs again during each processing, these interrupts are counted as once.

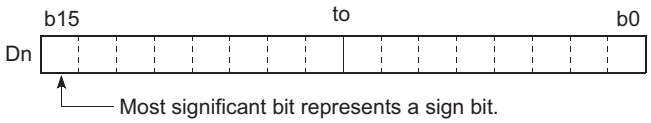
4.2.12 Data register (D)

The data register (D) is a memory in which numeric data (-32768 to 32767, or 0000_H to $FFFF_H$) can be stored.

(1) Bit structure of the data register

(a) Bit structure and read/write unit

One point of the data register consists of 16 bits, and data can be read or written in units of 16 bits.



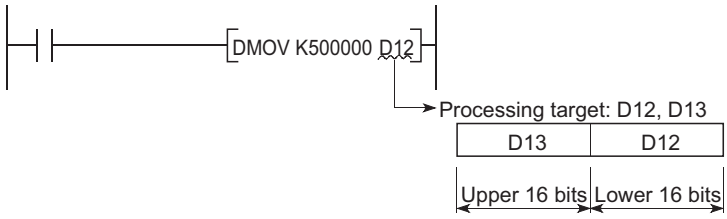
Point

Data register data are handled as signed data.
In the case of the hexadecimal notation, 0000_H to $FFFF_H$ can be stored. However, because the most significant bit represents a sign bit, decimal values that can be specified are -32768 to 32767.

(b) When using a 32-bit instruction for the data register

For a 32-bit instruction, two consecutive points of the data register (D_n and D_{n+1}) are the target of the processing.
The lower 16 bits correspond to the data register number (Dn) specified in the sequence program, and the higher 16 bits correspond to the specified data register number + 1.

Ex. When D12 is specified in the DMOV instruction, D12 represents the lower 16 bits and D13 represents the higher 16 bits.



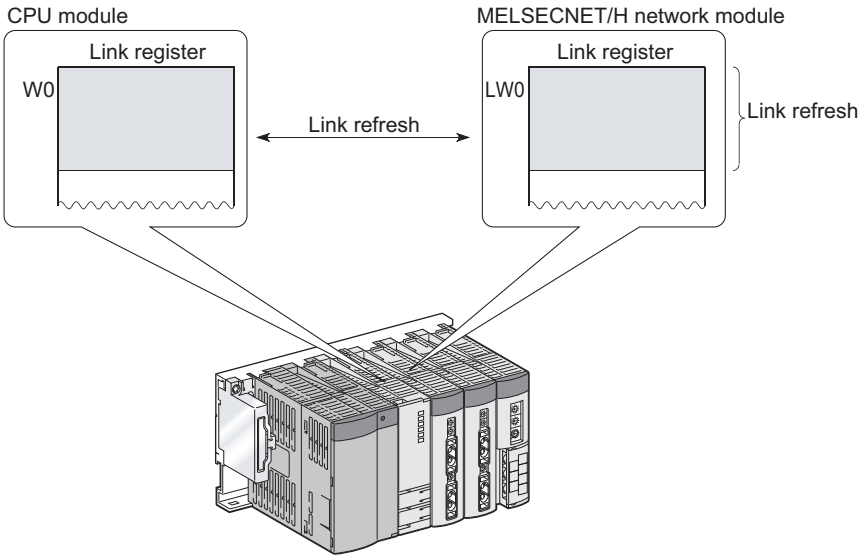
Data of -2147483648 to 2147483647 or 00000000_H to $FFFFFFFF_H$ can be stored in a two-point area of the data register. (The most significant bit in a 32-bit structure is a sign bit.)

(2) Retention of stored data

The data stored in the data register are held until other different data are stored.
Note that the stored data are initialized when the CPU module is powered off or reset.

4.2.13 Link register (W)

The link register (W) is a memory in the CPU module, which is refreshed with link register (LW) data of an intelligent function module such as a MELSECNET/H network module.

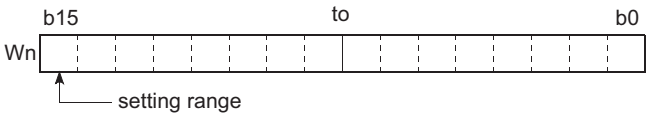


In the link register, numeric data (-32768 to 32767, or 0000_H to $FFFF_H$) are stored.

(1) Bit structure of the link register

(a) Bit structure and read/write unit

One point of the link register consists of 16 bits, and data can be read or written in units of 16 bits.



Point

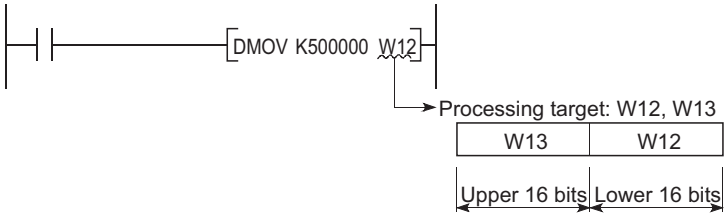
Link register data are handled as signed data.
In the case of the hexadecimal notation, 0000_H to $FFFF_H$ can be stored. However, because the most significant bit represents a sign bit, decimal values that can be specified are -32768 to 32767.

(b) When using a 32-bit instruction for the link register

For a 32-bit instruction, two consecutive points of the data register (Wn and Wn+1) are the target of the processing.

The lower 16 bits correspond to the link register number (Wn) specified in the sequence program, and the higher 16 bits correspond to the specified link register number + 1.

Ex. When W12 is specified in the DMOV instruction, W12 represents the lower 16 bits and D13 represents the higher 16 bits.



Data of -2147483648 to 2147483647 or 00000000_H to $FFFFFFFF_H$ can be stored in a two-point area of the link register. (The most significant bit in a 32-bit structure is a sign bit.)

(2) Retention of stored data

The data stored in the link register are held until other different data are stored.

Note that the stored data are initialized when the CPU module is powered off or reset.

Point

To use the link device in the network module exceeding link register points of the CPU module (default: 8192 points), change the link register points in the Device tab of the PLC Parameter dialog box.

(3) Using the link register in a network system

Network parameters must be set.

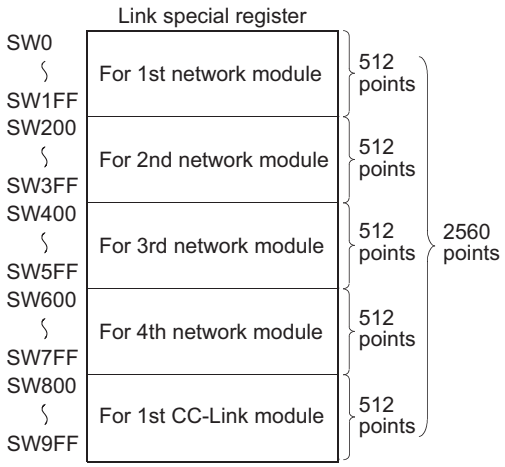
The area range that is not set by network parameters can be used as a data register.

4.2.14 Link special register (SW)

The link special register (SW) is used to store communication status data and error data of intelligent function modules, such as CC-Link IE module and MELSECNET/H module.
Because the data link information is stored as numeric data, error locations and causes can be checked by monitoring the link special register.

(1) Number of link special register points

The points for the link special register in the CPU module are 2048 (SW0 to SW7FF). However, the number of points can be changed in the Device tab of the PLC parameter dialog box. (☞ Page 447, Appendix 1.2.8) The link special register points for intelligent function modules (such as CC-Link IE modules and MELSECNET/H modules) are 512. Assign the link special register points as shown below. This enables data in the link special register (SW) of the CC-Link module to be refreshed to the link special relay (SW) of the CPU module.



For details of the link special register, refer to the manual for each intelligent function module that has the link special register.

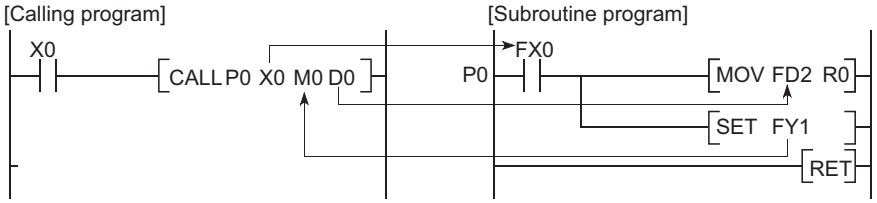
4.3 Internal System Devices

Internal system devices are provided for system operations.
The allocations and sizes of internal system devices are fixed, and cannot be changed by the user.

4.3.1 Function devices (FX, FY, FD)

Function devices are used in subroutine programs with argument passing.
Data are read or written between such subroutine programs and calling programs, using function devices.

Ex. When FX0, FY1, and FD2 are used in a subroutine program, and if X0, M0, and D0 are specified with a subroutine program call instruction, on/off data of X0 and FY1 are passed to FX0 and M0 respectively, and D0 data are passed to FD2.



(1) Applications of function devices

Because a device in each calling program can be determined by using a function device for subroutine programs, the same subroutine program can be used without considering other calling programs.

(2) Types of function devices

The following three types of function devices are available.

- Function input (FX)
- Function output (FY)
- Function register (FD)

(a) Function input (FX)

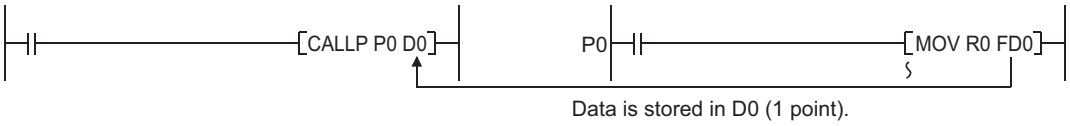
- The function input is used to pass on/off data to a subroutine program.
- Bit data specified by a subroutine call instruction with argument passing are fetched into a subroutine program and they are used for operations.
- All bit devices for the CPU module can be used.

(b) Function output (FY)

- The function output is used for passing an operation result (on/off data) in a subroutine program to a calling program.
- An operation result is stored in the device specified in the subroutine program with argument passing.
- All bit devices except for input devices of the CPU module (X and DX) can be used.

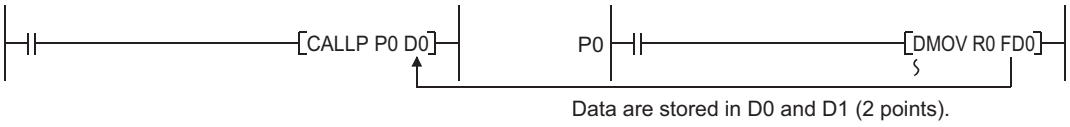
(c) Function register (FD)

- The function register is used for data writing or reading between a subroutine program and a calling program.
- The CPU module auto-detects the input or output conditions of the function register.
  Source data are input data of the subroutine program.
  Destination data are output data from the subroutine program.
- The function register of one point can occupy up to four words.
  Note that, however, the number of words used differs depending on the instruction in the subroutine program.
  1) A one-word instruction uses one word only.



Data is stored in D0 (1 point).

2) Two-word instruction uses two words.



Data are stored in D0 and D1 (2 points).

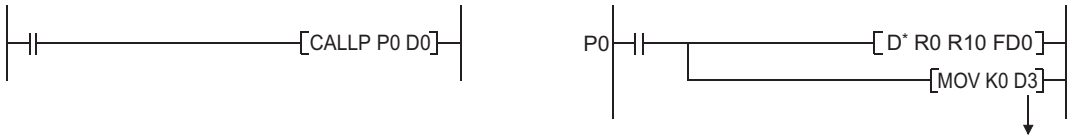
3) At a destination using 32-bit multiplication or division, four words are used.

Data are stored in D0 to D3 (4 points).

- Word devices of the CPU module can be used.

Point

In subroutine programs with argument passing, do not use any devices that are used by the function register.
If this occurs, function register values will not be normally passed to the calling program.



Since D0 to D3 are used for FD0, D3 cannot be used in the subroutine program.

For use of function devices, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

4.3.2 Special relay (SM)

The special relay (SM) is an internal relay whose specifications are determined by the programmable controller. This device stores the CPU module status data. For details, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

4.3.3 Special register (SD)

The special register (SD) is an internal relay whose specifications are determined by the programmable controller. This device stores the CPU module status data (such as error diagnostics and system information). For details, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

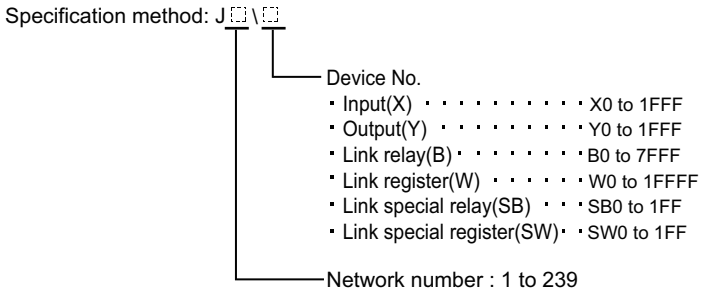
4.4 Link Direct Device

The link direct device is a device for direct access to the link device in a CC-Link IE Controller Network module, CC-Link IE Field Network master/local module or MELSECNET/H module. This CPU module can directly write data to or read data from the link device in each network module using sequence programs, regardless of the link refresh of the CPU module.

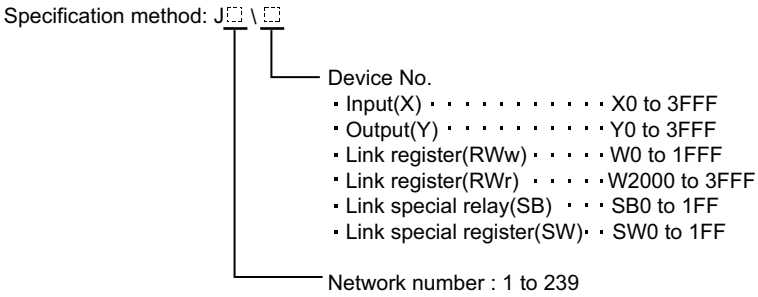
(1) Specification method and application example

(a) Specification method

- For a CC-Link IE Controller Network module and MELSECNET/H module, specify the device with a network number and a device number as shown below.

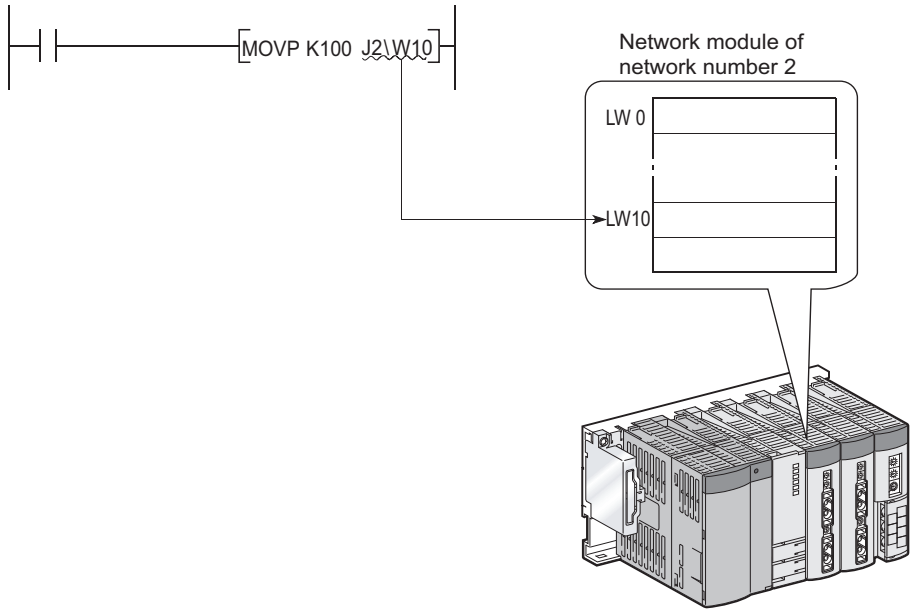


- For a CC-Link IE Field Network master/local module, specify the device with a network number and a device number as shown below.



(b) Application example

For link register 10 (W10) of network number 2, "J2\W10" must be used.



For a bit device (X, Y, B, or SB), the digit must be specified.

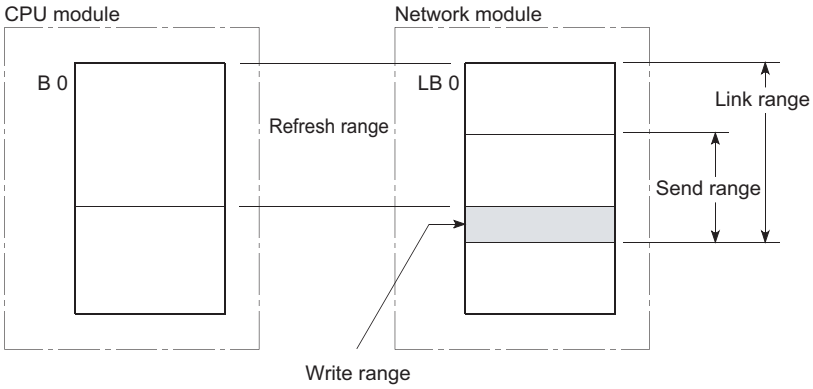
Ex. J1\K1X0, J10\K4B0

(2) Specification range

A link device that is not set in the Network parameter dialog box can be specified.

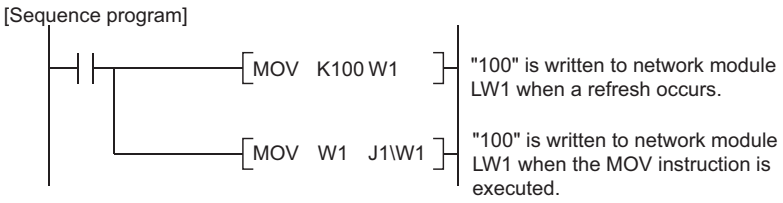
(a) Writing

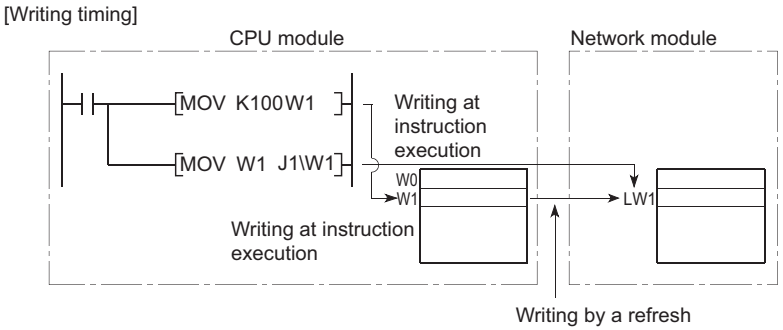
- The write range must be within the link device send range that is set by common parameters on Network parameter setting dialog box, and it must be outside the refresh range set by network refresh parameters.



- Although writing can be done to a refresh range portion (specified by refresh parameters) within the link device range, the link module's link device data will be overwritten when a refresh occurs.
  When writing data by using a link direct device, write the same data to the relevant devices on the CPU module side, which are set by refresh parameters.

  [Refresh parameter settings]
  1) Network number: 1
  2) CPU module (W0 to W3F) ↔ Network module (LW0 to LW3F)





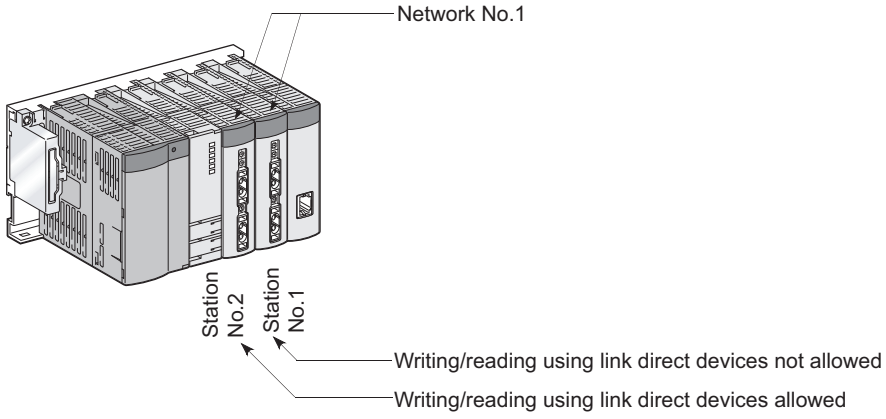
- If data are written to another station's write range using a link direct device, the data will be overwritten with other data that are received from the corresponding station.

(b) Reading

The link device ranges of network modules can be read.

Point

Writing or reading data by using a link direct device is allowed for only one network module that is on the same network. If two or more network modules are mounted on the same network, a network module with the lowest slot number is the target of writing or reading by the link direct device.
For example, if network modules set as station numbers 1 and 2 are mounted on network number 1 as shown in below, station number 2 is the target of the link direct device.



(3) Differences between link direct devices and link refresh

Item		Link direct device	Link refresh
Description on programs	Link relay	J□\K4B0 or higher	B0 or higher
	Link register	J□\W0 or higher	W0 or higher
	Link special relay	J□\K4SB0 or higher	SB0 or higher
	Link special register	J□\SW0 or higher	SW0 or higher
Number of steps		2 steps	1 step
Range of network module access		J□\□0 to J□\□3FFF	Range specified by refresh parameters
Guaranteed access data integrity		2 word (32-bit) units	

For network parameters, common parameters, and network refresh parameters, refer to the following.

- Details: Manual for each network module
- Setting method: Operating manual for the programming tool used

4.5 Module Access Devices

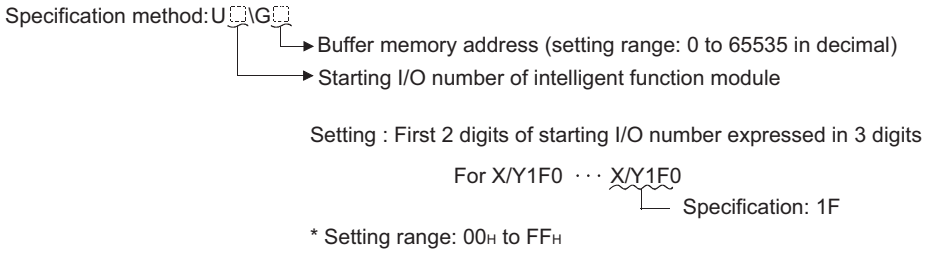
4.5.1 Intelligent function module device

The intelligent function module device allows direct access from the CPU module to the buffer memory of the intelligent function modules which are mounted on the main and extension base units.

(1) Specification method and application example

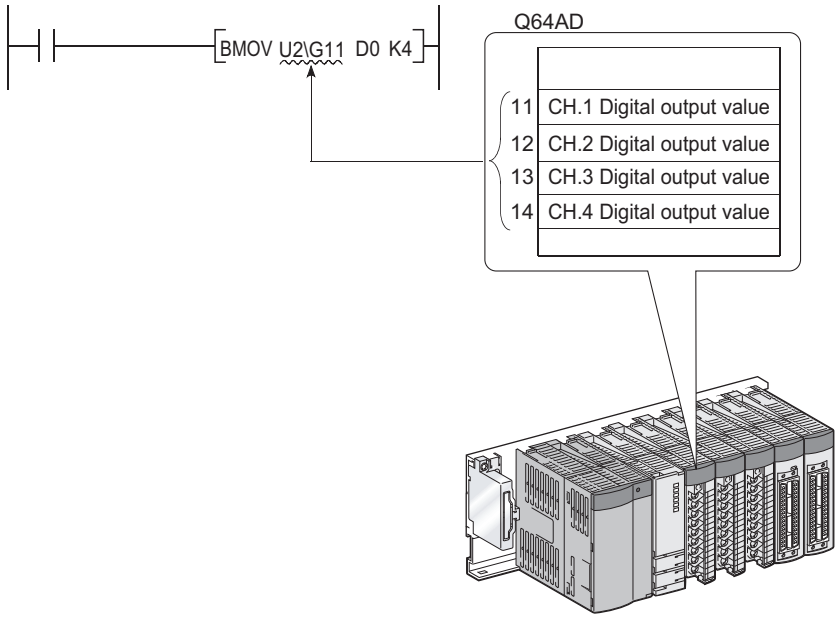
(a) Specification method

Specify the I/O number and buffer memory address of the intelligent function module.



(b) Application example

Specify the device as shown below to store CH.1 to CH.4 digital output values of the Q64AD analog-digital converter module into D0 to D3 of the CPU module when the Q64AD is mounted in the position of I/O number 020 (X/Y020 to X/Y02F).



If the intelligent function module device is used, device comments can be attached to the buffer memory areas.

Operating manual for the programming tool used

(2) Processing speed

The processing speed of the intelligent function module device is as follows:

- The processing speed of writing or reading using the intelligent function module device is slightly higher compared with the case of using the FROM or TO instruction.

  Ex. "MOV U2\G11 D0"

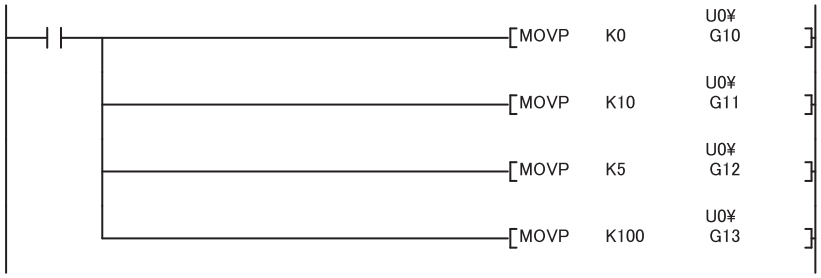
- When reading from the buffer memory of an intelligent function module and another processing with one instruction, totalize the processing speed of the FROM or TO instruction and the other instruction.

  Ex. "+ U2\G11 D0 D10"

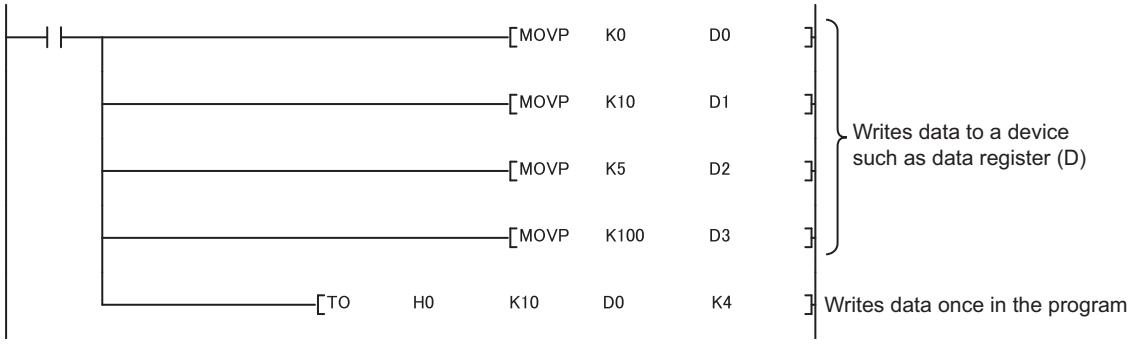
Point

Instead of using the intelligent function module device in the sequence program twice or more to write or read buffer memory data, using the FROM or TO instruction once in one place can increase the processing speed.

[Writing data using the intelligent function module device multiple times]



[Writing data using the TO instruction once in the program]



For buffer memory addresses and applications, refer to the manual for each intelligent function module used. For the FROM and TO instructions, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

4.5.2 Cyclic transmission area device

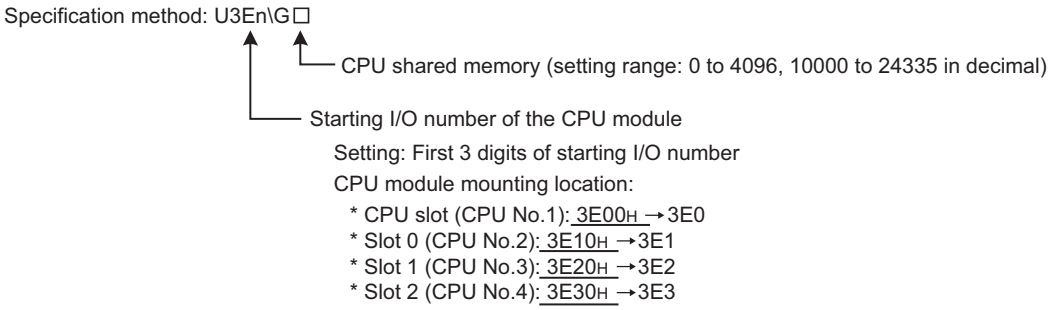
The cyclic transmission area device is used to access the CPU shared memory of each CPU module in a multiple CPU system.

(1) Features

- The transfer speed is higher than the case of using the write (S.TO or TO) or read (FROM) instruction to the CPU shared memory, resulting in reduced programming steps.
- Using the cyclic transmission area device allows bit manipulation.
- By setting device comments for the cyclic transmission area device, program readability is increased.
- Because information on the CPU shared memory can be directly specified as an argument of the instruction, no interlock device is required.

(2) Specification method

Specify the I/O number of the CPU module and the CPU shared memory address.



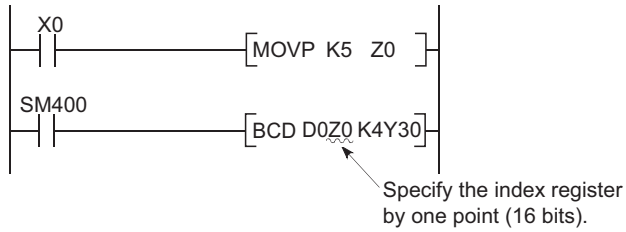
For details of the cyclic transmission area device, refer to the following.

QCPU User's Manual (Multiple CPU System)

4.6 Index Register (Z)/Standard Device Resister (Z)

4.6.1 Index register (Z)

The index register is used for indirect specification (index modification) in sequence programs.
Index modification uses one point of the index register.

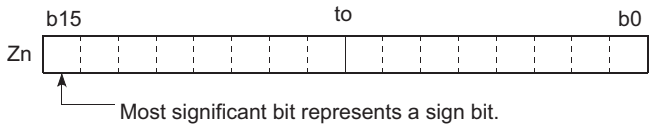


The index register has 20 points (Z0 to Z19).

(1) Bit structure of the index register

(a) Bit structure and read/write unit

One point of the index register consists of 16 bits, and data can be read or written in units of 16 bits.



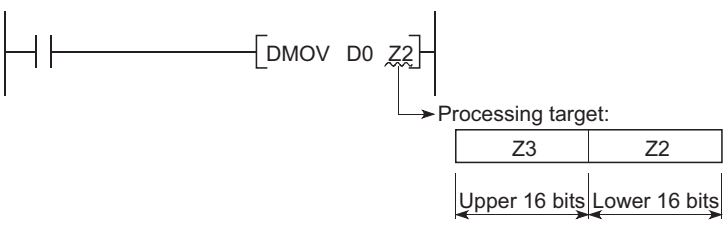
Point

Index register data are handled as signed data.
In the case of the hexadecimal notation, 0000_H to $FFFF_H$ can be stored. However, because the most significant bit represents a sign bit, decimal values that can be specified are -32768 to 32767. (When using T, TS, or C device, specify the values within the range of -16384 to 16383.)

(b) When using the index register for a 32-bit instruction

The processing target is Zn and Zn+1.
The lower 16 bits correspond to the specified index register number (Zn), and the higher 16 bits correspond to the specified index register number + 1.

Ex. When Z2 is specified in the DMOV instruction, Z2 represents the lower 16 bits and Z3 represents the higher 16 bits. (The most significant bit in a 32-bit structure is a sign bit.)



(2) 32-bit index modification

For 32-bit index modification, use two points of the index register. The index register areas to be used for 32-bit index modification is set in two ways:.

- by specifying the index register range used, or
- by using "ZZ". Note 4.4

Remark

For details and precautions of index modification using the index register, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

Note 4.4 Universal

When specifying the 32-bit index modification using "ZZ" with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

4.6.2 Standard device register (Z)

By using the index register between register operations, operations can be executed at a higher speed.
The index register used in this case is called the standard device resister.

(1) Device number

Since the standard device register is the same device as the index register, pay attention not to use the same device number when using the index modification.

Remark

For operation processing and processing time of the standard device register, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

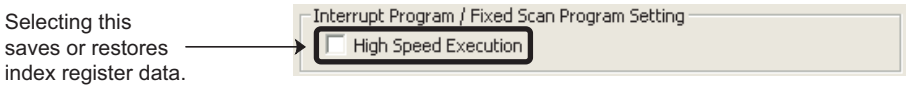
4.6.3 Switching from the scan execution type to the interrupt/fixed scan execution type program

The CPU module performs the following when switching from the scan execution type program to the interrupt/fixed scan execution type program.

- Saving and restoring the index register data
- Saving and restoring block numbers of the file register

(1) Setting for saving and restoration

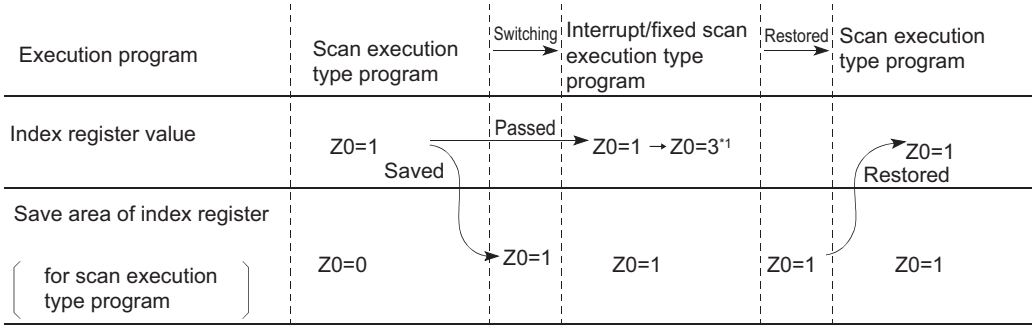
Saving and restoration setting can be enabled in the PLC system tab of the PLC parameter dialog box.
To disable writing to the index register in the interrupt/fixed scan execution type program, select "High Speed Execution" in the Interrupt Program/Fixed Scan Program Setting area.
If this setting is enabled, the program will switch faster than before.



(2) Processing of the index register

(a) When "High-speed execution" is not selected

- When switching from the scan execution type program to the interrupt/fixed scan execution type program
  The CPU module saves index register values in the scan execution type program, and passes them to the interrupt/fixed scan execution type program.
- When switching from the interrupt/fixed scan execution type program to the scan execution type program
  The CPU module restores the saved index register values.



*1 The Z0 value is changed to 3 in the interrupt program.

Point

To pass index register values from the interrupt/fixed scan execution type program to the scan execution type program, use word devices.

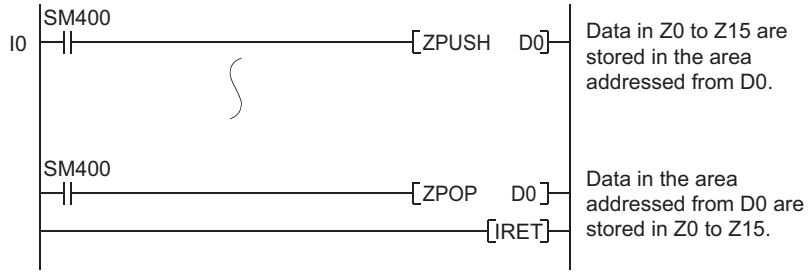
(b) When "High-speed execution" is selected

- When switching from the scan execution type program to the interrupt/fixed scan execution type program
  The CPU module does not save/restore any index register values.
- When switching from the interrupt/fixed scan execution type program to the scan execution type program
  If data are written to the index register by the interrupt/fixed scan execution type program, the values of the index register used in the scan execution type program will be corrupted.

Execution program	Scan execution type program	Switching	Interrupt/fixed scan execution type program	Restored	Scan execution type program
Index register value	Z0=1	Passed	Z0=1 → Z0=3*1	Passed	Z0=3
Save area of index register for scan execution type program	Z0=0	Z0=0	Z0=0	Z0=0	Z0=0

*1 The Z0 value is changed to 3 in the interrupt program.

When writing data to the index register by the interrupt/fixed scan execution type program, use the ZPUSH or ZPOP instruction to save and restore the data.



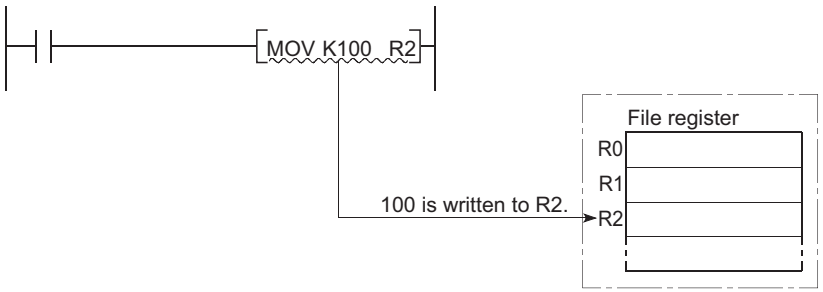
(3) Processing of file register's block numbers

- When switching from the scan execution type program to the interrupt/fixed scan execution type program
  The CPU module saves the file register block numbers in the scan execution type program, and passes them to the interrupt/fixed scan execution type program.
- When switching from the interrupt/fixed scan execution type program to the scan execution type program
  The CPU module restores the saved block numbers of the file register.

Execution program	Scan execution type program	Switching	Interrupt/fixed scan execution type program	Restored	Scan execution type program
Block No. of file register	Block 1 (Saved)	Passed	[RSET K0] Block1 → 0		Block 1 (Restored)
Save area	Block 0	Block 1	Block 1	Block 1	Block 1

4.7 File Register (R) Note 4.5

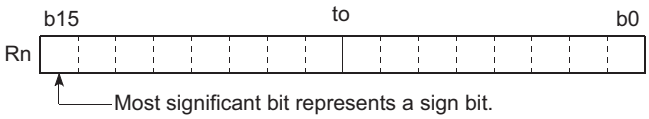
The file register (R) is a device provided for extending the data register.
The file register can be used at the same processing speed as the data register.



(1) Bit structure of the file register

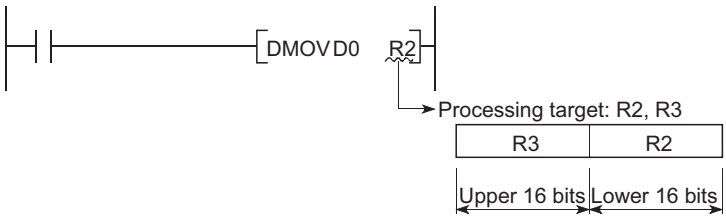
(a) Bit structure and read/write unit

One point of the file register consists of 16 bits, and data can be read or written in units of 16 bits.



(b) When using a 32-bit instruction for the file register

The processing target is R_n and R_{n+1}.
The lower 16 bits correspond to the file register number (Rn) specified in the sequence program, and the higher 16 bits correspond to the specified file register number + 1.
For example, when R2 is specified in the DMOV instruction, R2 represents the lower 16 bits and R3 represents the higher 16 bits.



Data of -2147483648 to 2147483647 or 00000000_H to $FFFFFFFF_H$ can be stored in a two-point area of the file register. (The most significant bit in a 32-bit structure is a sign bit.)

Note 4.5 Universal

The Q00UJCPU does not have the file register.

(2) Clearing the file register

If the Latch (2) is set in the Device tab of the PLC parameter dialog box, the data in the file register are not cleared even if the CPU module is powered off or reset. (Data cannot be initialized by performing latch clear operation.*1)
For how to clear the data, refer to the "Data Clear Processing" section. (☞ Page 75, Section 2.7 (4))

*1 The latch range of the file register can be set in the Device tab of the PLC parameter dialog box. (☞ Page 397, Section 4.7.4 (1) (c))

4.7.1 Storage location

File register data are stored in the following memory.

CPU module	Memory
Q00UCPU, Q01UCPU, QnUDVCPU, QnUDPVCPU	Standard RAM
Q02UCPU, QnUD(H)CPU, QnUDE(H)CPU	Standard RAM, SRAM card, Flash card

4.7.2 File register size

The size of file register is the total number of points of the file register (ZR), extended data register (D), and extended link register (W). Set the size so that the total number of points will be less than the free space of memory specified as a storage location. The free space can be checked in the Confirm Memory Size window using a programming tool.

[Tool]⇨[Confirm Memory Size]

(1) Storing data in the standard RAM

The following table lists the points available for the file register data to be stored in the standard RAM.
Note that if data other than the file register data are stored in the standard RAM, the points will decrease.
(☞ Page 36, Section 2.1.1 (2))

CPU module		Point
Q00UCPU, Q01UCPU, Q02UCPU		64K
Q03UD(E)CPU		96K
Q03UDVCPU		96K
	With an extended SRAM cassette (1M)	608K
	With an extended SRAM cassette (2M)	1120K
	With an extended SRAM cassette (4M)	2144K
	With an extended SRAM cassette (8M)	4192K
Q04UD(E)HCPU		128K
Q04UDVCPU, Q04UDPVCPU		128K
	With an extended SRAM cassette (1M)	640K
	With an extended SRAM cassette (2M)	1152K
	With an extended SRAM cassette (4M)	2176K
	With an extended SRAM cassette (8M)	4224K
Q06UD(E)HCPU		384K

CPU module		Point
Q06UDVCPU, Q06UDPVCPU		384K
	With an extended SRAM cassette (1M)	896K
	With an extended SRAM cassette (2M)	1408K
	With an extended SRAM cassette (4M)	2432K
	With an extended SRAM cassette (8M)	4480K
Q10UD(E)HCPU, Q13UD(E)HCPU		512K
Q13UDVCPU, Q13UDPVCPU		512K
	With an extended SRAM cassette (1M)	1024K
	With an extended SRAM cassette (2M)	1536K
	With an extended SRAM cassette (4M)	2560K
	With an extended SRAM cassette (8M)	4608K
Q20UD(E)HCPU, Q26UD(E)HCPU		640K
Q26UDVCPU, Q26UDPVCPU		640K
	With an extended SRAM cassette (1M)	1152K
	With an extended SRAM cassette (2M)	1664K
	With an extended SRAM cassette (4M)	2688K
	With an extended SRAM cassette (8M)	4736K
Q50UDEHCPU		768K
Q100UDEHCPU		896K

(2) Storing data in an SRAM card

Up to 4086K points can be stored in one file.
Since one block consists of 32K words, up to 128 blocks can be stored.

Note that the number of points or blocks that can be added depends on the size of the programs and device comments stored in the memory card.

(3) Storing data in a Flash card

Up to 2039K points can be stored in one file.
Since one block consists of 32K words, up to 64 blocks can be stored.
Note that the number of points or blocks that can be added depends on the memory card capacity and the size of the programs and device comments stored in the memory card.

Remark

For the memory cards available for the CPU module, refer to Page 37, Section 2.1.1 (4).

4.7.3 Differences in available accesses by storage memory

Accesses available for the file register vary for each memory.

Access		Standard RAM	SRAM card	Flash card
Program writing		○	○	×
Program reading		○	○	○
Writing device memory to programmable controller		○	○	×
Reading device memory from programmable controller		○	○	○
Data modification	Online test operation using a programming tool	○	○	×
	Writing data to a CPU module using a programming tool	○	○	×
	Writing data to a CPU module (flash ROM) using a programming tool	×	×	○
	Batch write by a serial communication module	○	○	×
	Device data writing from GOT1000 series	○	○	×
	Random write command from GOT1000 series	○	○	×

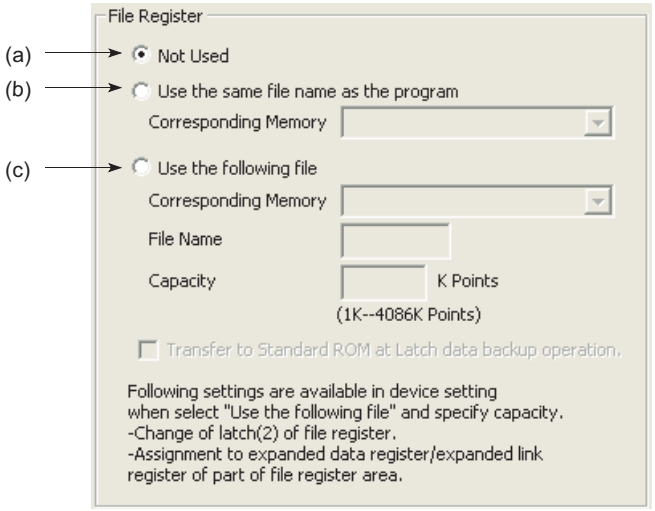
4.7.4 Registration procedure for the file register

To use the file register, register files to the CPU module by the following procedure.

1. **Set a file register file. (☞ Page 395, Section 4.7.4 (1))**
2. **Write the file register file. (☞ Page 398, Section 4.7.4 (2))**

(1) Setting a file register file

Set whether to use a file register in the PLC file tab of the PLC parameter dialog box.



(a) Not Used

Select this in the following cases.

- When not using any file register
- When specifying a file register used in the sequence program (the QDRSET instruction is used for specification.)

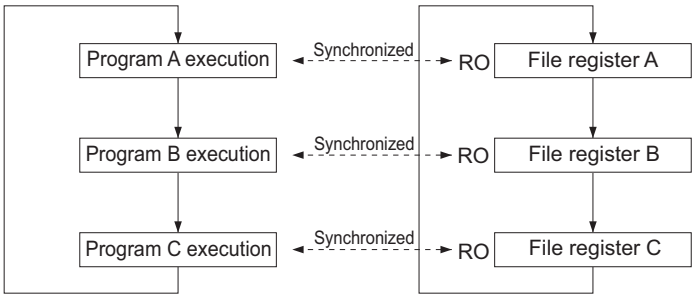
(b) Use the same file name as the program.

Select this when executing the file register with the same file name as the sequence program. Select the memory to be used for the file register from a standard RAM or a memory card.

- When the program is changed

  The file name of the file register is automatically changed to the same name as the program. This feature is useful if the file register is exclusively used for one program as a local device.

Ex. When each of file registers A to C has the same name with the corresponding one of the program A to C, the operation is as described below.



- Point setting for file registers

  Set the number of file register points in the "File Register Detail Setting" screen when writing data to the programmable controller.

  [Online] ⇨ [Write to PLC] ⇨ the Detail button of "File Register"

Point

- Only one file register can be created in the standard RAM. (This applies to CPU modules except the High-speed Universal model QCPU and Universal model Process CPU.)
  To create more than one, use a SRAM or Flash card.
- With some instructions, file registers set for respective programs cannot be specified.
  For details, refer to the pages describing devices available for each instruction in the following manual.
  MELSEC-Q/L Programming Manual (Common Instruction)

(c) Use the following file.

Select this when one file register is to be shared by all execution programs.

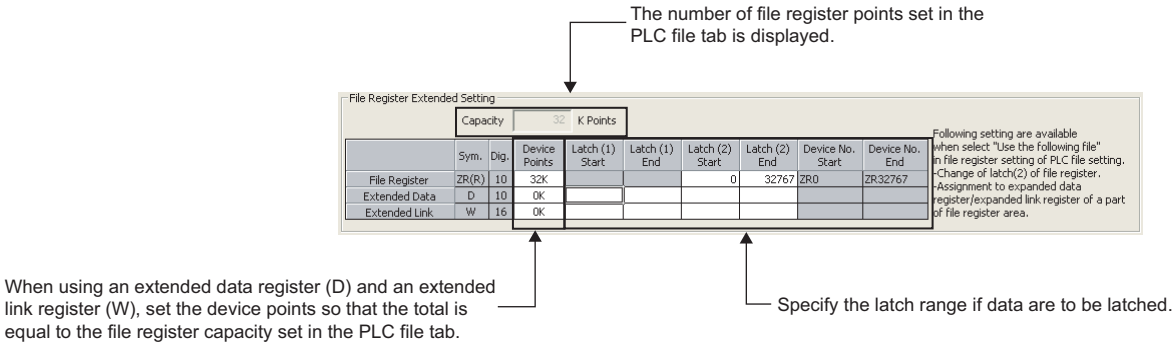
Specify "Corresponding Memory", "File Name", and "Capacity" and write these parameters to the CPU module to create a file for the file register.

If the capacity is not specified, note the following.

- When the specified file register file is stored in the specified drive, the file is used. (The capacity is the same as that of the stored file register file.)
- If the file register file with the specified file name is not found on the specified drive, "PARAMETER ERROR" (error code: 3002) will occur.
- For use of an ATA card, "Memory card (ROM) cannot be selected for "Corresponding Memory".
  (File register data cannot be stored in ATA cards.)
  Selecting "Memory card (ROM)"for "Corresponding Memory" and writing the settings to the CPU module will result in "PARAMETER ERROR" (error code: 3002).

Point

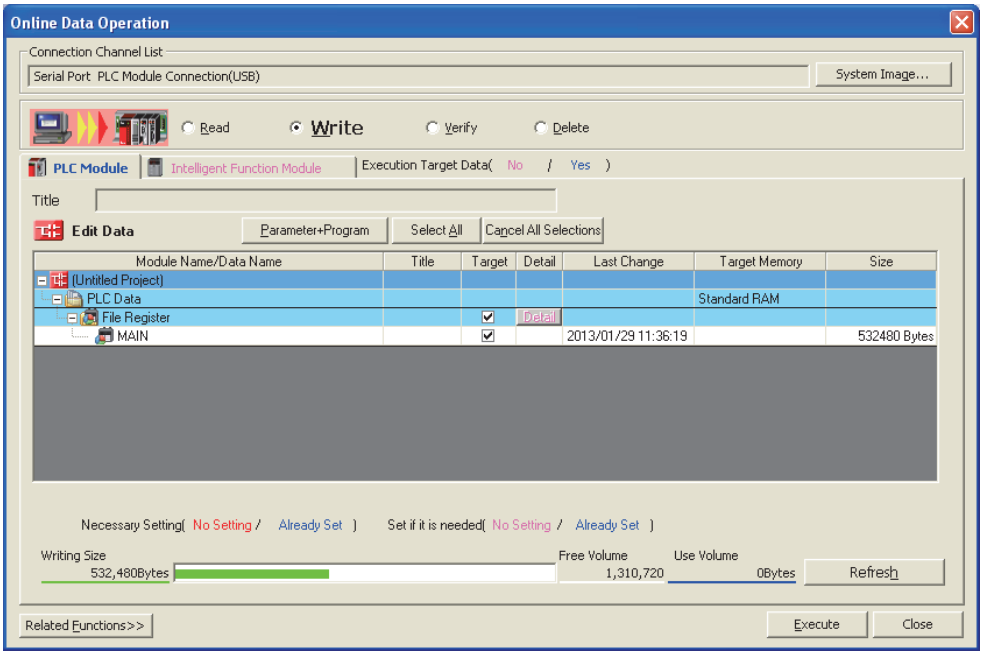
If necessary, the latch range of a file register can be changed in the "Use the following file" setting.



(2) Registering a file register file to the CPU module

Register a file to the CPU module by executing the write to PLC function.

[Online] ⇨ [Write to PLC]



(a) Registration memory

Select a memory where the file register file is to be registered from the following.

- Standard RAM
- Memory card (SRAM)
- Memory card (Flash)

To use the same file name that is used in the program, register the file register file to the memory specified in the PLC File tab of the PLC parameter dialog box.

(b) File register size of the CPU module

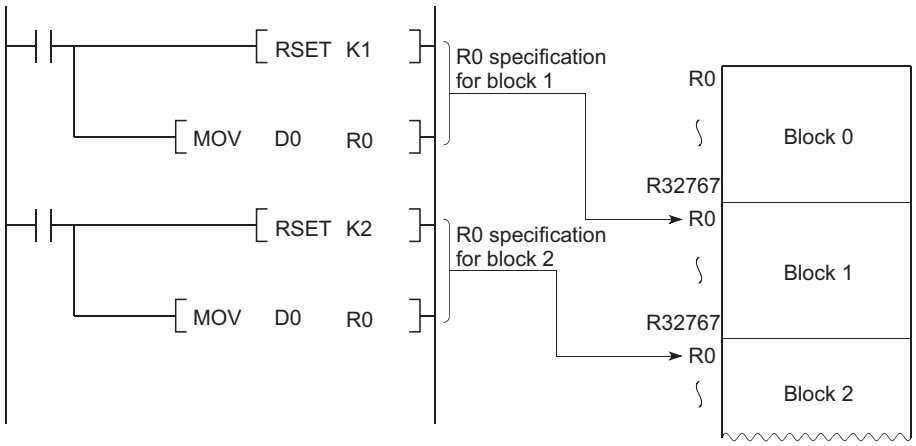
The file register size can be set in increments of one point. Note that each file size is ensured in units of 256 points. Even if a file register is not specified from ZR0, the created file will have an assignment from ZR0 to the last number.

Ex. If the write range of a file register is specified to be ZR1000 to ZR1791, the created file register file will have an assignment from ZR0 to ZR1791. However, because the data in ZR0 to ZR999 are unreliable, specify the file register from ZR0. The size of the file register is checked in the units of 1K points. Therefore, the file register size must be specified from R0 in the units of 1K points.

4.7.5 Specification methods of the file register

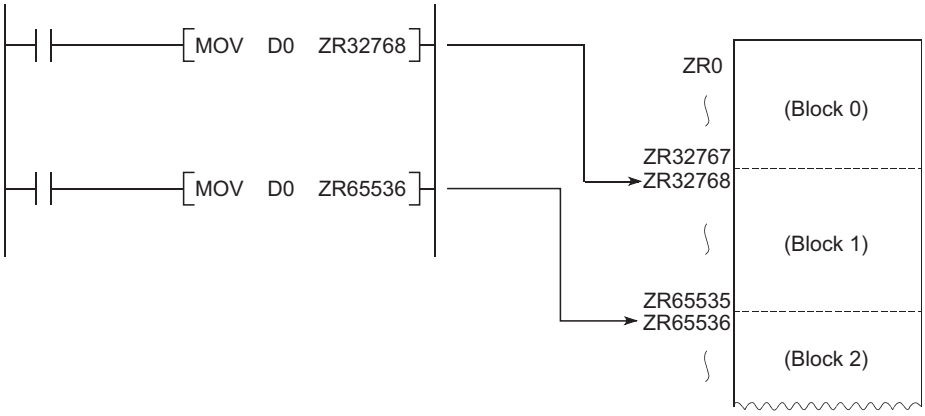
(1) Block switching method

The file register points used are divided and specified in units of 32K points (R0 to R32767).
If multiple blocks are used, the desired block is specified with the block number in the RSET instruction.
Each block has a specification range of R0 to R32767.



(2) Serial number access method

A file register whose size is exceeding 32K points can be specified using consecutive device numbers.
Multiple blocks of a file register can be used as a continuous file register.
This kind of device is expressed as "ZR".



Point

The block numbers and ZR device points that can be specified vary depending on the following.

- Storage location of the file register (Page 393, Section 4.7.1)
- File register size (Page 393, Section 4.7.2)

4.7.6 Precautions for using the file register

(1) No registration or use of an invalid file register number

(a) When the file of the file register has not been registered

Writing to or reading from the file register will result in "OPERATION ERROR" (error code: 4101).

(b) When writing to or reading from the file register exceeding the registered size (points)

"OPERATION ERROR" (error code: 4101) will occur.

(2) File register size check

When writing to or reading from the file register, check the file register size so that data can be written or read within the size (points) set for the CPU module.

(a) Checking the file register size

The file register size can be checked in the File register capacity area (SD647).*1

The file register size data in units of 1K points is stored in this SD647.

*1 If a file register file is switched to another, the size of the currently selected file register file is stored in SD647.

Point

The remainder after dividing the file register size by 1K points is discarded.
To ensure an accurate "range of use" check, specify the file register setting in units of 1K points (1024 points).

(b) Checking timing

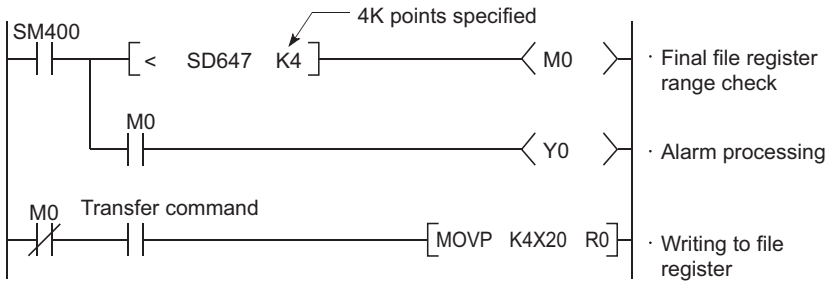
- In a program using any file register, check the file register size at step 0.
- After execution of the file register file switching instruction (QDRSET), check the file register size.
- Before executing the file register block switching instruction (RSET), check that space of 1K points or more is ensured in a block after switching. The space can be calculated using the following formula.
  (File register size) > [32K points × (Switching block No.) + 1K points]

(c) File register size checking procedure

- Check the file register size used for each sequence program.
- Check the total file register size set in SD647 on the sequence program to see if there are sufficient number of points to be used or not.

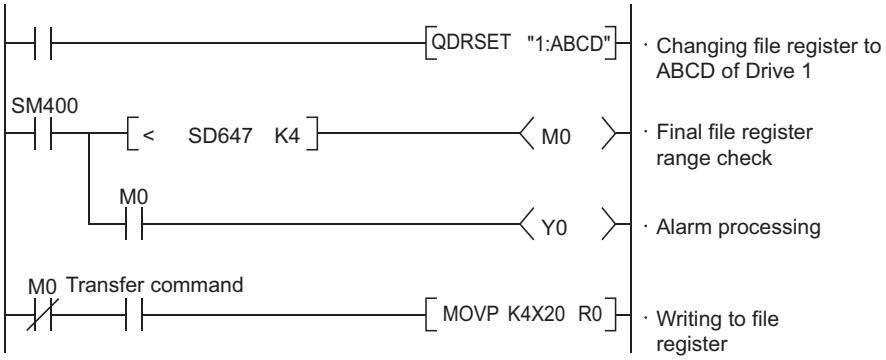
[Program example 1]

The file register range of use is checked at the beginning of each program.



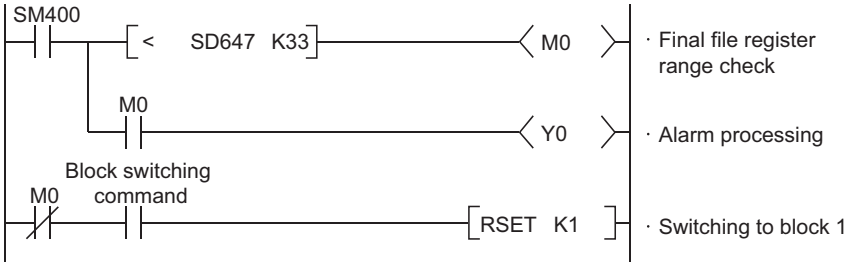
[Program example 2]

The file register range of use is checked after execution of the QDRSET instruction.



[Program example 3]

When a block is switched to another:



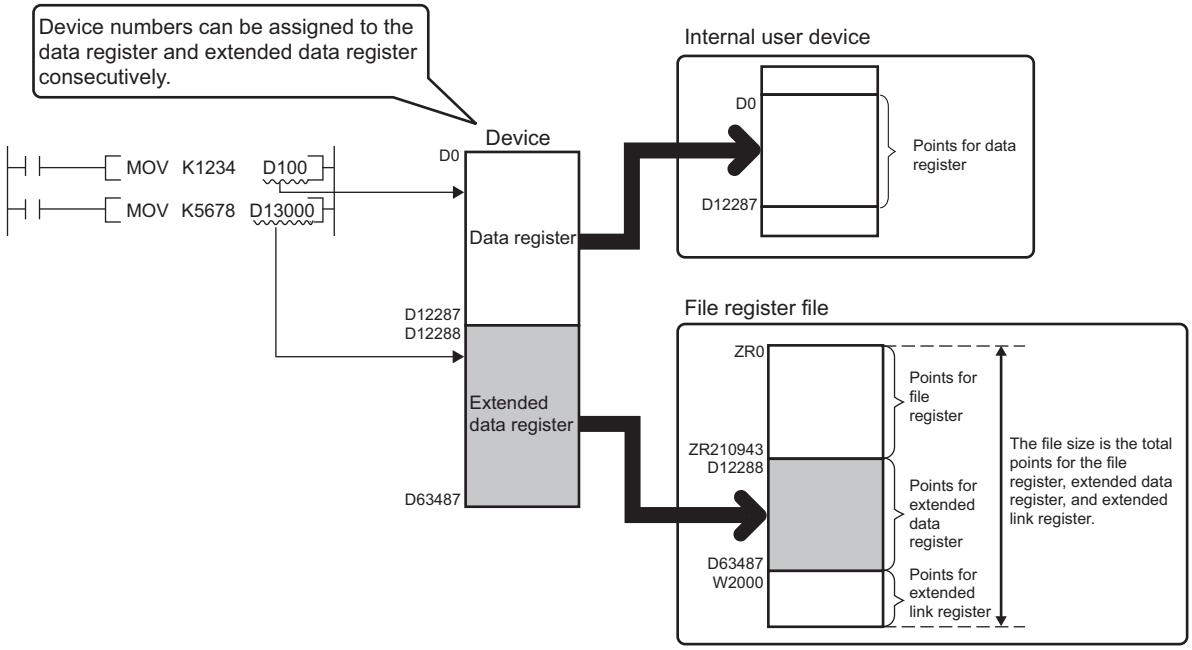
(3) Deleting a file register file

Delete an unnecessary file register file from "Delete PLC Data" of programming tool.

[Online] ⇨ [Delete PLC Data]

4.8 Extended Data Register (D) and Extended Link Register (W) Note 4.6

The extended data register (D) and extended link register (W) are devices for using the large-capacity file register (ZR) area as an extended area of the data register (D) and link register (W). These devices can be programmed as the data register (D) and link register (W) together with the file register (ZR) area.



(1) Device numbers

Device numbers for the extended data register (D) and extended link register (W) can be assigned consecutively after those for the internal user devices, data register (D) and link register (W).

Point

- Even though device numbers are consecutively assigned, there is no physical area contiguity between the data register (D) (internal user device) and the extended data register (D), and between the link register (W) (internal user device) and the extended link register (W). To use them as one contiguous area, set the points for the data register (D) and link register (W) (internal user device) to "0" in the Device tab of the PLC parameter dialog box, and use only the extended data register (D) and extended link register (W).
- When the file register (ZR), extended data register (D), and extended link register (W) are used for auto refresh, set the points so that they should not exceed those set in the "File Register Extended Setting" in the Device tab of the PLC parameter dialog box.

Note 4.6 Universal

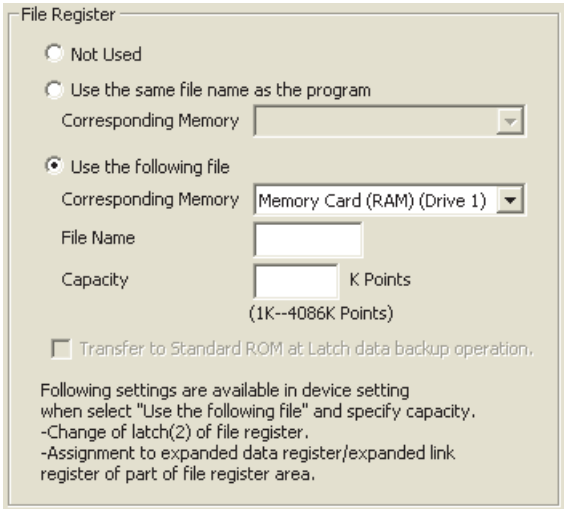
The Q00UJCPU does not support the use of these devices.
When using these devices with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, or QnUDE(H)CPU, check the version of the programming tool used. (Page 466, Appendix 2)

(2) Setting method

Since the extended data register (D) and extended link register (W) use the file register area, data must be set for both the file register setting and the device setting.

(a) File register setting

Select "Use the following file." in the PLC file tab of the PLC parameter dialog box, and do the setting as shown below. The "Use the same file name as the program." cannot be selected.



Item	Corresponding Memory		File Name	Capacity*1	Remarks
"Use the following file"	Memory card (RAM)*4		Any	1 to 4086K points*2	-
	Memory card (ROM)*4		Any	1 to 2039K points*3	Read only
	Standard RAM	Q00UCPU, Q01UCPU, Q02UCPU	Any	1 to 64K points	-
		Q03UD(E)CPU, Q03UDVCPU*5		1 to 96K points	
		Q04UD(E)HCPU, Q04UDVCPU*5, Q04UDPVCPU*5		1 to 128K points	
		Q06UD(E)HCPU, Q06UDVCPU*5, Q06UDPVCPU*5		1 to 384K points	
		Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU*5, Q13UDPVCPU*5		1 to 512K points	
		Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU*5, Q26UDPVCPU*5		1 to 640K points	
		Q50UDEHCPU		1 to 768K points	
		Q100UDEHCPU		1 to 896K points	

*1 The size of file register is the total number of points for the file register (ZR), extended data register (D), and extended link register (W). Set the size so that the total number of points will be less than the free space of memory specified as a storage location. The free space can be checked in the Confirm Memory Size window using a programming tool.

[Tool]⇨[Confirm Memory Size]

*2 This is the maximum number of points when an SRAM card (8M bytes) is used.

*3 This is the maximum number of points when a Flash card (4M bytes) is used.

*4 The Q00UCPU, Q01UCPU, High-speed Universal model QCPU, and Universal model Process CPU do not support the use of memory card.

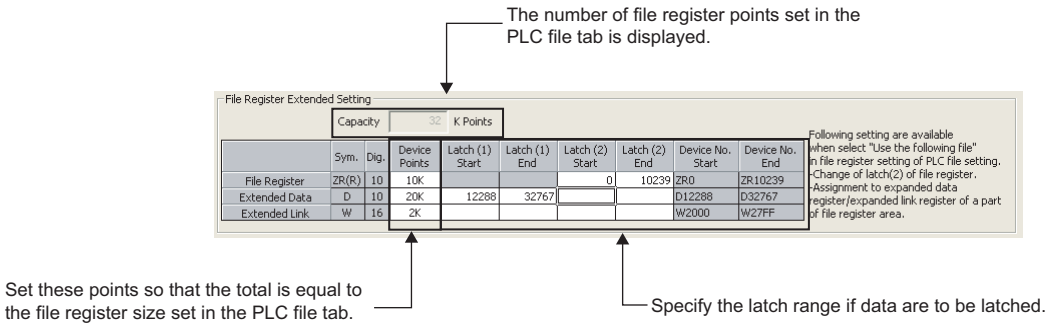
*5 For the memory size when an extended SRAM cassette is used, refer to Page 393, Section 4.7.2.

(b) Device setting

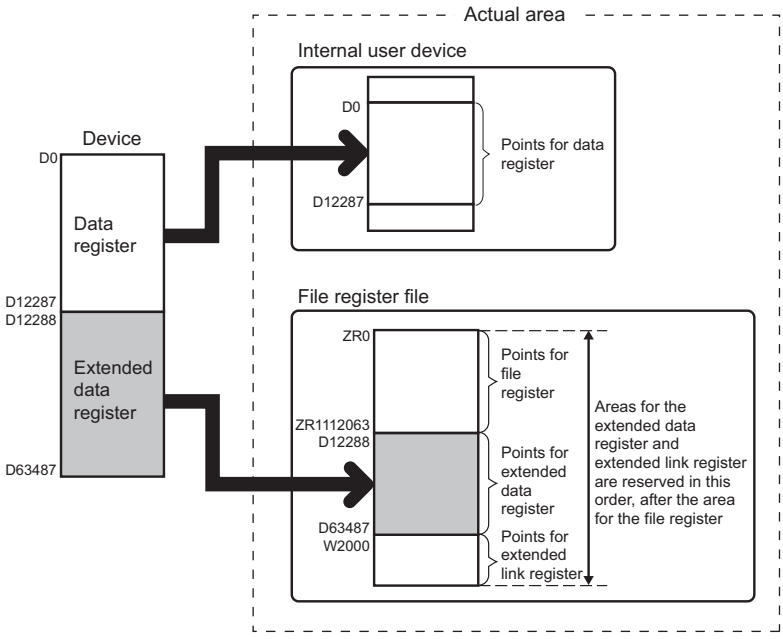
Set each number of points for the extended data register (D) and extended link register (W) in the File Register Extended Setting in the Device tab of the PLC parameter dialog box. Assign a part of the points set for the file register (ZR) in the PLC file tab to the extended data register (D) and extended link register (W). The latch range can be changed if necessary.

If data are to be latched, specify the latch range.

- Latch (1) and (2) of the extended data register (D)
- Latch (1) and (2) of the extended link register (W)



Once the points for the extended data register (D) and extended link register (W) is set, areas for these devices are reserved in the file register file.



(3) Checking the points by the special register

The points for each of the file register (ZR), extended data register (D), and extended link register (W) can be checked in the following special register areas.

- SD306, SD307: File register (ZR)
- SD308, SD309: Extended data register (D)
- SD310, SD311: Extended link register (W)

(4) Precautions

For use of the extended data register (D) and extended link register (W), pay attention to the following.

- Since the file register (ZR) area is used, the values of the following items will be the same as those for the file register (ZR) when the extended data register (D) and extended link register (W) are specified.

  - Number of program steps
  - Instruction processing time
  - Processing time of auto refresh with network modules
  - Processing time of auto refresh with CC-Link IE Field Network Basic
  - Processing time of auto refresh with intelligent function modules
  - Processing time of auto refresh between CPU shared memories

- The file register size cannot be changed while the CPU module is in the RUN status.
- The file register cannot be switched to another by using the QDRSET instruction. ("OPERATION ERROR" (error code: 4100))
- Set the refresh ranges for the following auto refresh properly so that each refresh range does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).

  - Auto refresh with network modules
  - Auto refresh with CC-Link IE Field Network Basic
  - Auto refresh with intelligent function modules
  - Auto refresh between CPU shared memories

- Set the following properly so that each specification does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).

  - Index modification
  - Indirect specification
  - Specification for instructions that use block data*1

*1 Block data means the following:
- Data used in instructions, such as FMOV, BMOV, and BK+, which treat more than one word for operation.
- Control data, composed of two or more words, which are specified in instructions, such as SP.FWRITE and SP.FREAD.
- Data in a 32-bit or greater format (binary 32 bits, real number, indirect address of a device)

Remark

For details on the index modification and indirect specification with the extended data register (D) and extended link register (W), refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

- To access the extended data register (D) or extended link register (W) from a module that does not support the use of these devices, device numbers need to be specified with those of the file register (ZR). Calculation formulas for obtaining device numbers of the file register (ZR) to be specified to access the extended data register (D) and extended link register (W) and calculation examples are described below

Item	Calculation formula
Device number of the file register (ZR) used to access the extended data register (D)	$ED_{ZN} = ZR_C + (ED_N - D_C)$
Device number of the file register (ZR) used to access the extended link register (W)	$EW_{ZN} = ZR_C + ED_C + (EW_N - W_C)$

*1 Variables in the table indicate the following:

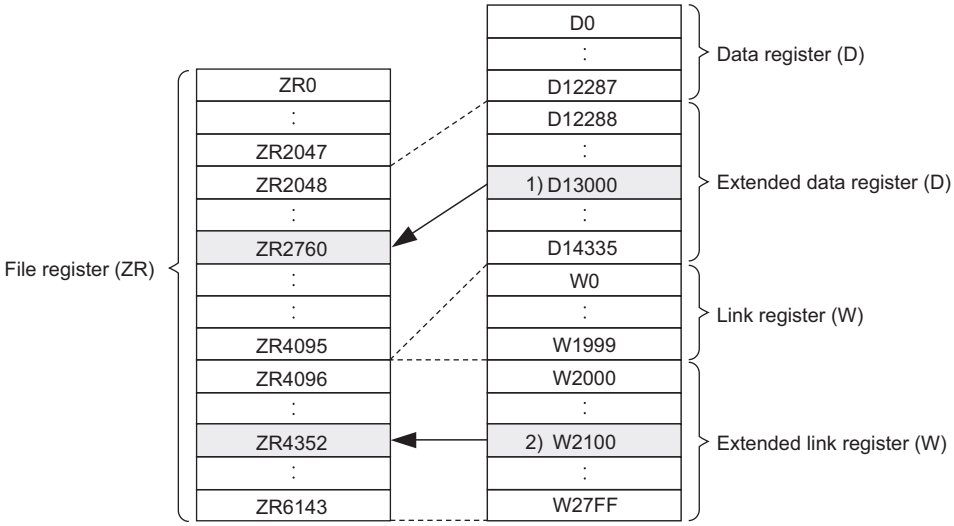
- ZR_C: Points of the file register (ZR)
- ED_{ZN}: Device number of the file register (ZR) used to access the extended data register (D)
- ED_N: Access target device number of the extended data register (D)
- D_C: Points of the data register (D)
- ED_C: Points of the extended data register (D)
- EW_{ZN}: Device number of the file register (ZR) used to access the extended link register (W)
- EW_N: Access target device number of the extended link register (W) (hexadecimal)
- W_C: Points of the link register (W)

[Calculation example]

- D_C: Points of the data register (D) ••• 12288 points
- W_C: Points of the link register (W) ••• 8192 points
- ZR_C: Points of the file register (ZR) ••• 2048 points
- ED_C: Points of the extended data register (D) ••• 2048 points

1) Device number of the file register (ZR) used to access D13000
$ED_{ZN} = 2048 + (13000-12288) = 2760$

2) Device number of the file register (ZR) used to access W2100
$EW_{ZN} = 2048 + 2048 + (2100_H - 8192) = 2048 + 2048 + (8448 - 8192) = 4352$



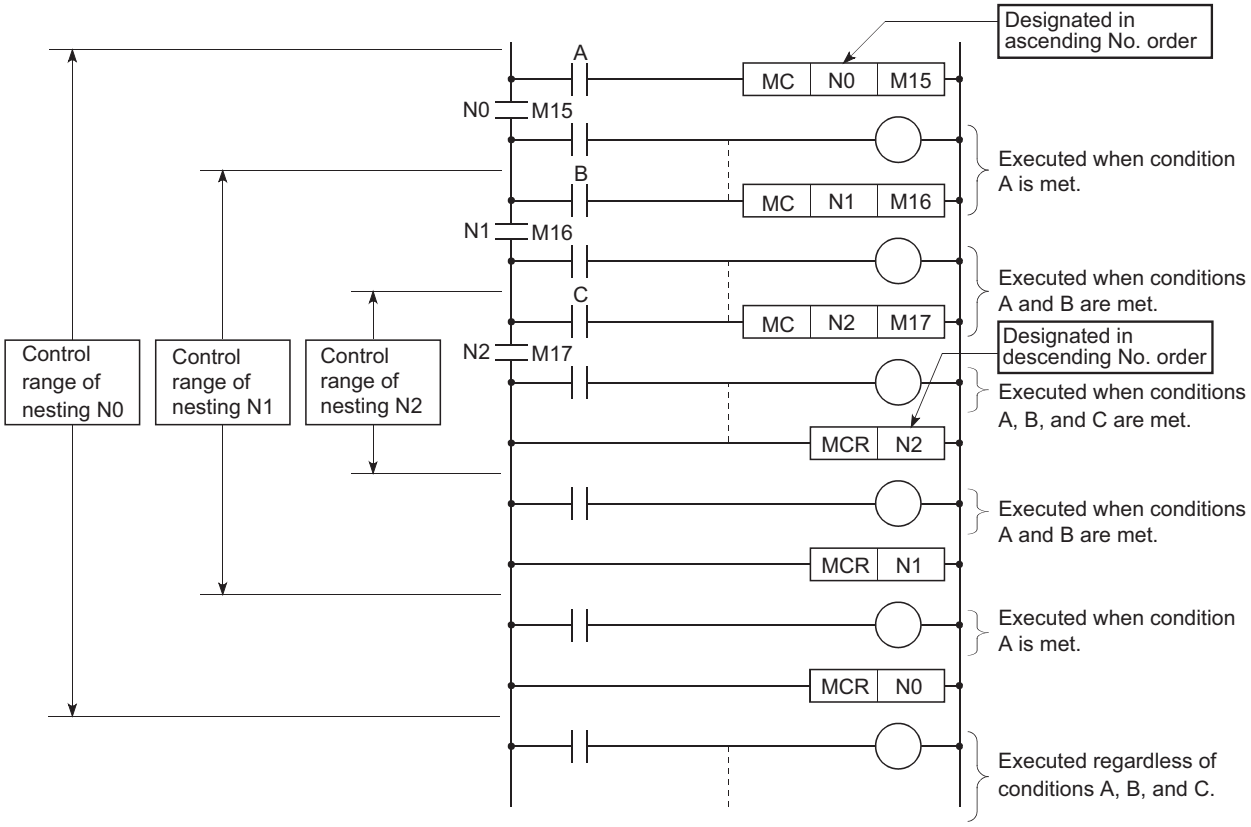
4.9 Nesting (N)

Nesting (N) is a device used in the master control instructions (MC and MCR instructions) to program operation conditions in a nesting structure.

(1) Specification method using master control instructions

The master control instruction opens or closes a common ladder gate to switch the ladder of a sequence program efficiently.

Specify the nesting (N) in ascending order (in order of N0 to N14), starting from the outside of the nesting structure.



For use of the nesting, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

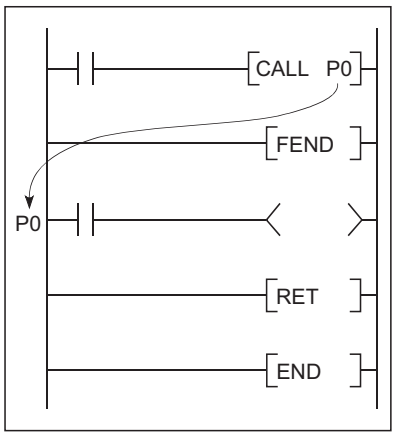
4.10 Pointer (P)

The pointer (P) is a device used in jump instructions (CJ, SCJ, or JMP) or subroutine call instructions (such as CALL).

(1) Applications

Pointers can be used in the following applications.

- Specification of the jump destination in a jump instruction (CJ, SCJ, or JMP) and a label (start address of the jump destination)
- Specification of the call destination of a subroutine call instruction (CALL or CALLP) and a label (start address of the subroutine program)



(2) Pointer types

There are the following two different pointer types.

- Local pointer (Page 409, Section 4.10.1):
  The pointer used independently in each program
- Common pointer (Page 411, Section 4.10.2):
  The pointer that can be called in all running programs by the subroutine call instruction.

(3) Number of available pointer points

The following shows the available number of points.

CPU module	Points
Q00UJCPU, Q00UCPU, Q01UCPU	512 points
Q02UCPU, Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU	4096 points
Q50UDEHCPU, Q100UDEHCPU	8192 points

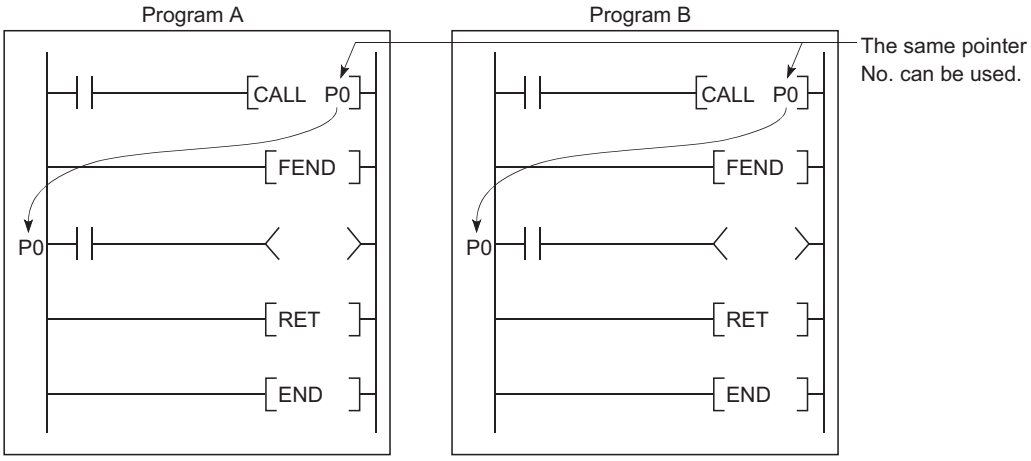
Remark

For the jump instructions and subroutine call instructions, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

4.10.1 Local pointer

The local pointer is a pointer that can be used independently in jump instructions and subroutine call instructions in each program.
The same pointer number can be used in respective programs.

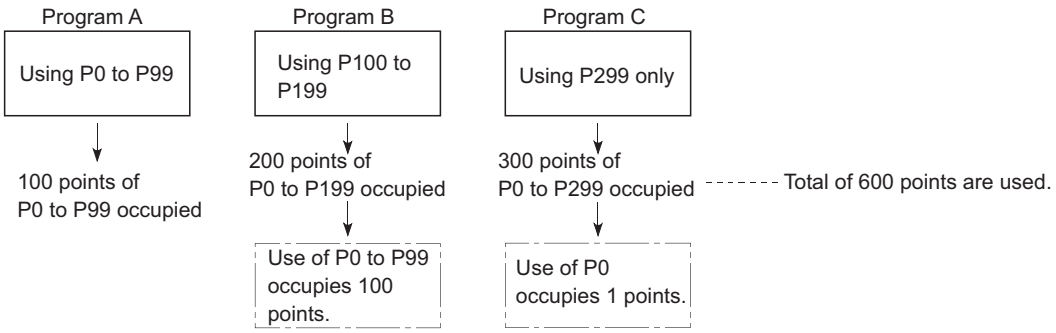


(1) Number of local pointer points

The local pointer can be divided for use of all the programs stored in the program memory.
The local pointer number ranges from P0 to the highest number of the local pointer in use. (The CPU module's system computes the number of points used.)

Even if only P99 is used in a program, for example, the number of points used will be 100, which is from P0 to P99.
For using the local pointer for several programs, use the pointers in ascending order starting from P0 in each program.

Ex. The total is 600 points when the pointer is used as shown below.



(2) Precautions for using the local pointer

(a) Program where the local pointer is described

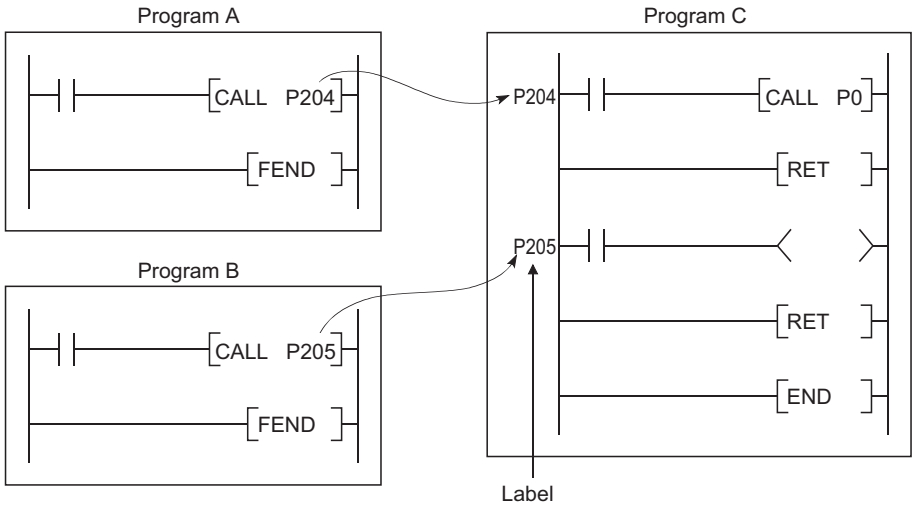
A jump from another program is not allowed.
Use the ECALL instruction from another program when calling a subroutine program in a program file that contains any local pointer.

(b) Total number of local pointer points

If the total number of pointers (in all programs) exceeds the number of points available for each CPU module, a "Pointer configuration error" (error code: 4020) occurs. For the number of available pointer points of each CPU module, refer to Page 408, Section 4.10 (3).

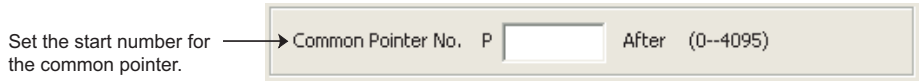
4.10.2 Common pointer

The common pointer is used to call subroutine programs from all programs that are being executed.



(1) Common pointer range

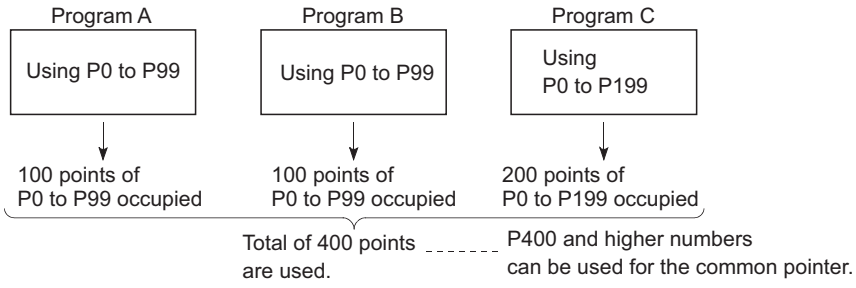
In the PLC system tab of the PLC parameter dialog box, set the start number for the common pointer.
The common pointer range is from the specified pointer number to P4095.
However, the pointer number that can be entered here is a number higher than the total points used for the local pointer.



Ex. If a total of 400 points are used in three programs (100 points in each of Program A and Program B, and 200 points in Program C), for example, P400 and higher numbers can be set for the common pointer.

(2) Precautions

- The same pointer number cannot be used as a label. Doing so will result in a "Pointer configuration error" (error code: 4021).
- If the total number of the local pointer points used in several programs exceeds the start number of the common pointer, a "Pointer configuration error (error code: 4020) will occur.

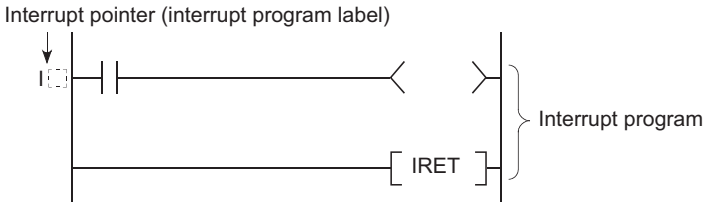


Point

The jump instructions are not capable of executing a jump to the common pointer in other programs.
Use the common pointer with subroutine call instructions only.

4.11 Interrupt Pointer(I)

The interrupt pointer (I) is used as a label at the start of an interrupt program, and can be used in any programs.



(1) Number of available points

The number of points and the range available for the interrupt pointer are shown below.

CPU module	Point	Range
Q00UJCPU, Q00UCPU, Q01UCPU	128 points	I0 to I127
Q02UCPU, Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU, Q50UDEHCPU, Q100UDEHCPU	256 points	I0 to I255

(2) Interrupt factors

The interrupt factors for the available interrupt pointers are shown below.

Interrupt factor	Interrupt pointer No.	Description
Interrupt by an interrupt module*1	I0 to I15	Interrupt input from an interrupt module
Interrupt by a sequence-started module	I16 to I27	Interrupt from an AnS/A series special function module*2 that is capable of starting an interrupt in the CPU module
Interrupt by the internal timer	I28 to I31, I49*5	Fixed scan interrupt by the internal timer of the CPU module
Multiple CPU synchronous interrupt*3	I45	Fixed scan interrupt to execute synchronized control with the operation cycle of a motion controller
Intelligent function module interrupt	I50 to I255	Interrupt from an intelligent function module*4

*1 For available interrupt modules, refer to the following.
QCPU User's Manual (Hardware Design, Maintenance and Inspection)

*2 The corresponding module is an intelligent communication module. For details, refer to the manual for each module.

*3 Applicable when using the Universal model QCPU and motion controller that support the multiple CPU high speed transmission.

*4 This module can be a serial communication module, MELSECNET/H module, Ethernet module, or high-speed counter module. For details, refer to the manual for each module.

*5 Only the High-speed Universal model QCPU and Universal model Process CPU support the use of I49.

Point

To use the intelligent function module interrupt (Page 232, Section 3.22), the intelligent function module setting (interrupt pointer setting) is required in the "PLC system" tab of the PLC parameter dialog box. (Page 439, Appendix 1.2.2)

4.11.1 List of interrupt pointer numbers and interrupt factors

The list of interrupt pointer numbers and interrupt factors are shown below.

(1) When a Q series interrupt module is mounted

I No.	Interrupt factor		Priority	I No.	Interrupt factor		Priority
I0	Interrupt by an interrupt module (QI60)	1st point	6	I32 to I44	-	N/A	-
I1		2nd point	7				
I2		3rd point	8				
I3		4th point	9				
I4		5th point	10				
I5		6th point	11				
I6		7th point	12				
I7		8th point	13				
I8		9th point	14				
I9		10th point	15				
I10		11th point	16				
I11		12th point	17				
I12		13th point	18				
I13		14th point	19				
I14		15th point	20	I45*2 *5	Multiple CPU synchronous interrupt	0.88ms	1
I15		16th point	21				
I16	Interrupt by a sequence-started module	1st module	228	I46 to I48	-	N/A	-
I17		2nd module	229				
I18		3rd module	230				
I19		4th module	231				
I20		5th module	232	I49	High-speed interrupt	0.1 to 1.0ms	*6
I21		6th module	233	I50 to I255	Intelligent function module interrupt*3*4 /Interrupt by an interrupt module (QI60)	Specify an intelligent function module or interrupt module (QI60) in parameter.	22 to 227
I22		7th module	234				
I23		8th module	235				
I24		9th module	236				
I25		10th module	237				
I26		11th module	238				
I27		12th module	239				
I28*5	Interrupt by the internal timer*1	100ms	5				
I29*5		40ms	4				
I30*5		20ms	3				
I31*5		10ms	2				

*1 The time-limit value of the internal timer is set by default. In the PLC system tab of the PLC parameter dialog box, the value can be changed within the range of 0.5ms to 1000ms in increments of 0.5ms.

*2 This is available for multiple CPU system configuration.

*3 To use the intelligent function module interrupt, the intelligent function module setting (interrupt pointer setting) is required in the PLC system tab of the PLC parameter dialog box. (For interrupt from an intelligent function module, refer to Page 232, Section 3.22.

*4 I50 has the highest priority (priority 22), and I255 has the lowest priority (priority 227).

*5 When an interrupt occurs, even if no interrupt pointer exists on the program, CAN'T EXECUTE(I) (error code: 4220) does not occur.

*6 To use I49, do not execute any other interrupt programs (I0 to I48, I50 to I255) nor fixed scan execution type programs. If executed, the interrupt program (I49) will not be executed at preset intervals.

(2) When an A series interrupt module is mounted

I No.	Interrupt factor		Priority	I No.	Interrupt factor		Priority
I0	Interrupt by an interrupt module (A1SI61)	1st point	224	I32 to I44	-	N/A	-
I1		2nd point	225				
I2		3rd point	226				
I3		4th point	227				
I4		5th point	228				
I5		6th point	229				
I6		7th point	230				
I7		8th point	231				
I8		9th point	232				
I9		10th point	233				
I10		11th point	234				
I11		12th point	235				
I12		13th point	236				
I13		14th point	237				
I14		15th point	238	I45*2 *5	Multiple CPU synchronous interrupt	0.88ms	1
I15		16th point	239				
I16	Interrupt by a sequence-started module	1st module	212	I46 to I48	-	N/A	-
I17		2nd module	213				
I18		3rd module	214				
I19		4th module	215	I49	High-speed interrupt	0.1 to 1.0ms	*6
I20		5th module	216	I50 to I255	Intelligent function module interrupt*3*4 /Interrupt by an interrupt module (A1SI61)	Specify an intelligent function module or interrupt module (A1SI61) in parameter.	6 to 211
I21		6th module	217				
I22		7th module	218				
I23		8th module	219				
I24		9th module	220				
I25		10th module	221				
I26		11th module	222				
I27		12th module	223				
I28*5	Interrupt by the internal timer*1	100ms	5				
I29*5		40ms	4				
I30*5		20ms	3				
I31*5		10ms	2				

*1 The time-limit value of the internal timer is set by default. In the PLC system tab of the PLC parameter dialog box, the value can be changed within the range of 0.5ms to 1000ms in increments of 0.5ms.

*2 This is available for multiple CPU system configuration.

*3 To use the intelligent function module interrupt, the intelligent function module setting (interrupt pointer setting) is required in the PLC system tab of the PLC parameter dialog box. (For interrupt from an intelligent function module, refer to Page 232, Section 3.22.)

*4 I50 has the highest priority (priority 6), and I255 has the lowest priority (priority 211).

*5 When an interrupt occurs, even if no interrupt pointer exists on the program, CAN'T EXECUTE(I) (error code: 4220) does not occur.

*6 To use I49, do not execute any other interrupt programs (I0 to I48, I50 to I255) nor fixed scan execution type programs. If executed, the interrupt program (I49) will not be executed at preset intervals.

4.12 Other Devices

4.12.1 SFC block device (BL)

The SFC block is used to check that the specified block in the SFC program is activated.

Remark

For use of the SFC block device, refer to the following.

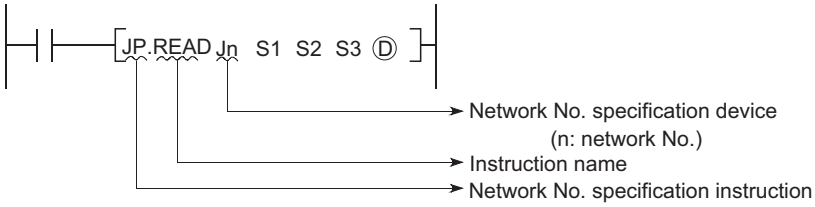
MELSEC-Q/L/QnA Programming Manual (SFC)

4.12.2 Network No. specification device (J)

The network No. specification device is used to specify the network number in the link dedicated instructions.

(1) Specification method

Specify as shown below by using the link dedicated instructions.



Remark

For details of the link dedicated instructions, refer to the following.

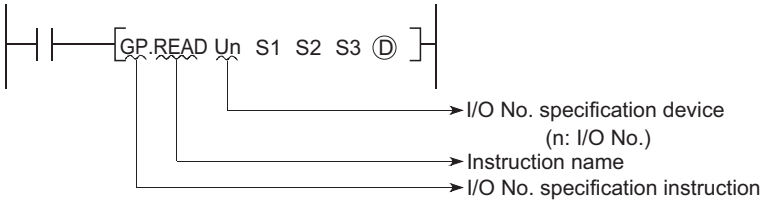
Manual for each network module

4.12.3 I/O No. specification device (U)

The I/O No. specification device is used to specify I/O numbers in the intelligent function module dedicated instructions.

(1) Specification method

Specify as shown below by using the intelligent function module dedicated instructions.



Remark

For details of the intelligent function module dedicated instructions, refer to the following.

Manual for the intelligent function module used

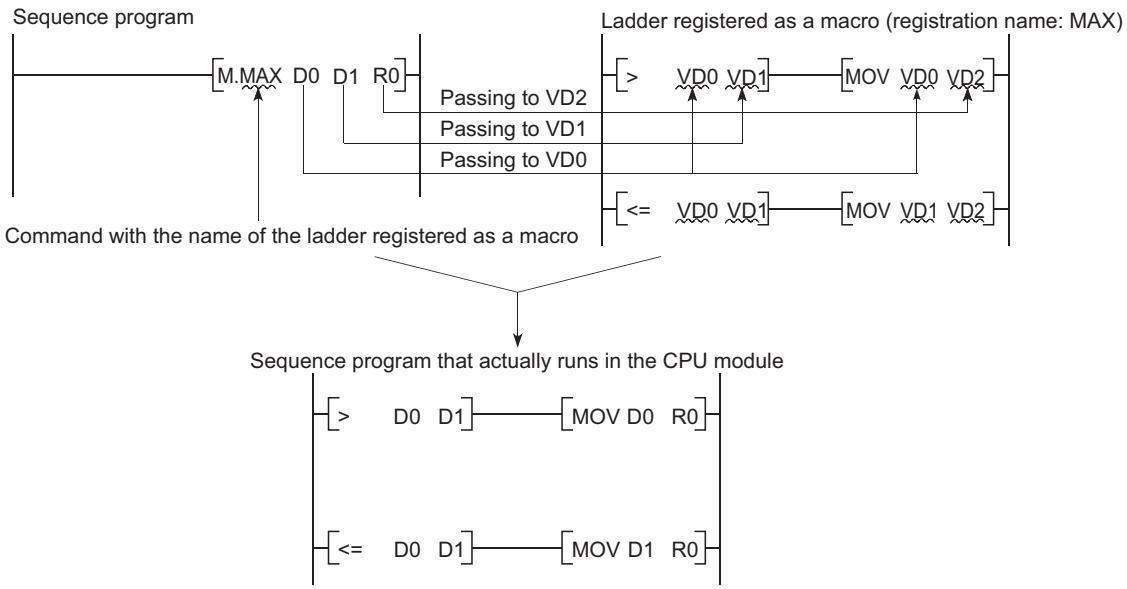
4.12.4 Macro instruction argument device (VD)

The macro instruction argument device (VD) is used with ladders registered as macros.

When a VD□ setting is specified, the value is converted to the specified device when the macro instruction is executed.

(1) Specification method

Among the devices used in the ladders registered as macros, specify a device used for VD. When using macro instructions in the sequence program, specify devices that correspond to the macro instruction argument devices used in the macro registration ladders in ascending order.



CHAPTER 5 CONSTANTS

5.1 Decimal Constant (K)

The decimal constant (K) is used to specify decimal data in sequence programs.

Specify it as K⬚ (example: K1234) in sequence programs.

In the CPU module, data are stored in binary (BIN). (☞ Page 495, Appendix 4.1)

(1) Specification range

The specification ranges for decimal constants are as follows:

- When using word data (16-bit data) ••• K-32768 to K32767
- When using 2-word data (32-bit data) ••• K-2147483648 to K2147483647

Point

The most significant bit represents a sign bit.

5.2 Hexadecimal Constant (H)

The hexadecimal constant (H) is a device for specifying hexadecimal or BCD data in sequence programs.
(For BCD data, each digit of a hexadecimal number is specified with 0 to 9.)

In sequence programs, specify it as H⬚ (example: H1234). (☞ Page 496, Appendix 4.2)

(1) Specification range

The specification ranges for hexadecimal constants are as follows:

- When using word data (16-bit data) ••• H0 to HFFFF (For BCD data, H0 to H9999)
- When using 2-word data (32-bit data) ••• H0 to HFFFFFFFF (For BCD data, H0 to H99999999)

5.3 Real Number (E)

The real number (E) is a device used to specify real numbers in sequence programs.

In sequence programs, specify it as E□ (example: E1.234). (☞ Page 498, Appendix 4.4)

(1) Specification range

(a) Real number setting range

- For single-precision floating-point data

  $-2^{128} < \text{Device} \leqq -2^{-126}, 0, 2^{-126} \leqq \text{Device} < 2^{128}$

- For double-precision floating-point data

  $-2^{1024} < \text{Device} \leqq -2^{-1022}, 0, 2^{-1022} \leqq \text{Device} < 2^{1024}$

(b) When an overflow or underflow has occurred

The following table shows the operation of the CPU module when an overflow or underflow has occurred during arithmetic operation.

Overflow	Underflow
OPERATION ERROR (error code: 4141)	Turned to 0 without any error

(c) When a special value*1 is input

If operation is performed with input data that contains a special value, "OPERATION ERROR" (error code: 4140) occurs.

*1 The special values are -0, unnormalized numbers, nonnumeric characters, and ±∞.

(2) Specification method

Real numbers can be specified in sequence programs by the following expressions.

- Normal expression ••• A numeric value can be specified as it is.

  Ex. 10.2345 can be specified as E10.2345.

- Exponential expression ••• A numeric value is specified by (Value) × 10^n.

  Ex. 1234 is specified as E1.234 + 3.*1

*1 + 3 represents 10^3 in E1.234 + 3.

5.4 Character String (" ")

The character string is a device used to specify a character string in sequence program.
Characters enclosed in quotation marks (example: "ABCD1234") are specified.

(1) Available characters

The shift JIS code can be used for character strings.
The CPU module distinguishes between upper and lower case characters.

(2) Number of specified characters

A string from the specified character to the NUL code (00_H) is one unit.

Note that, however, up to 32 characters can be specified for an instruction using a character string, such as $MOV.

CHAPTER 6 CONVENIENT USAGE OF DEVICES

When multiple programs are executed in the CPU module, each program can be executed independently by specifying an internal user device as a local device.

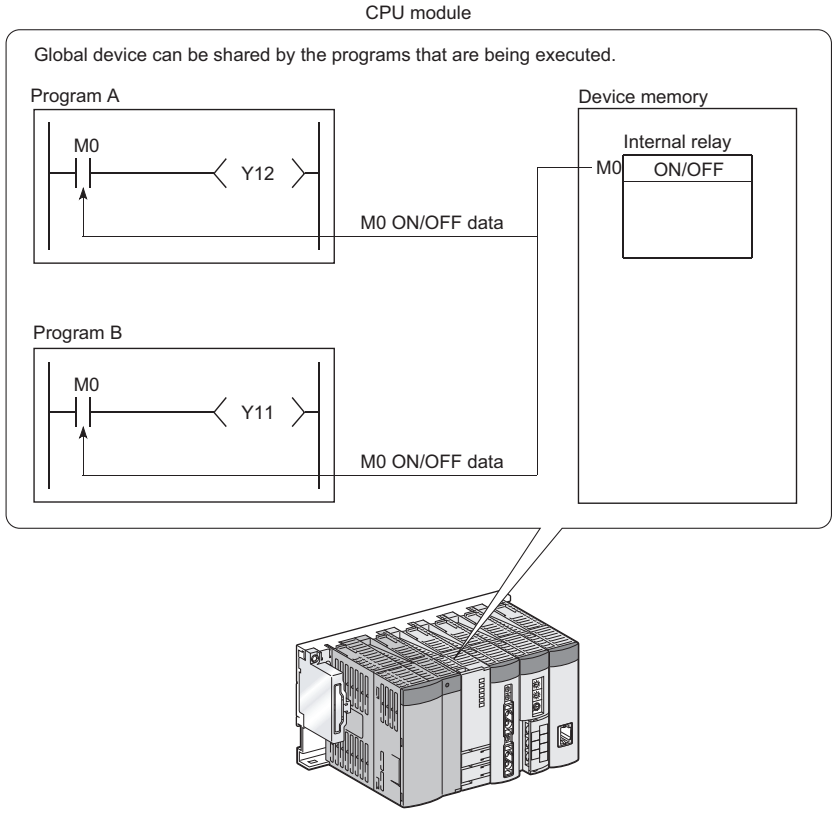
Devices of the CPU module are classified into the following two types:

- Global device that can be shared by multiple programs that are being executed.
- Local device that is used independently for each program.

6.1 Global Device

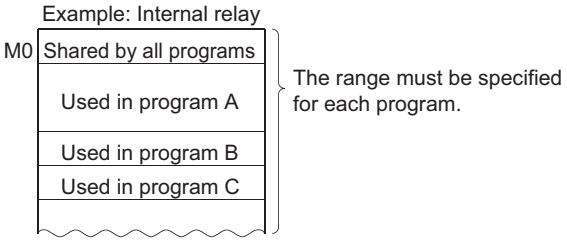
Programs being executed in the CPU module can share the global device.

Global device data are stored in the device memory of the CPU module, and can be shared by all programs.



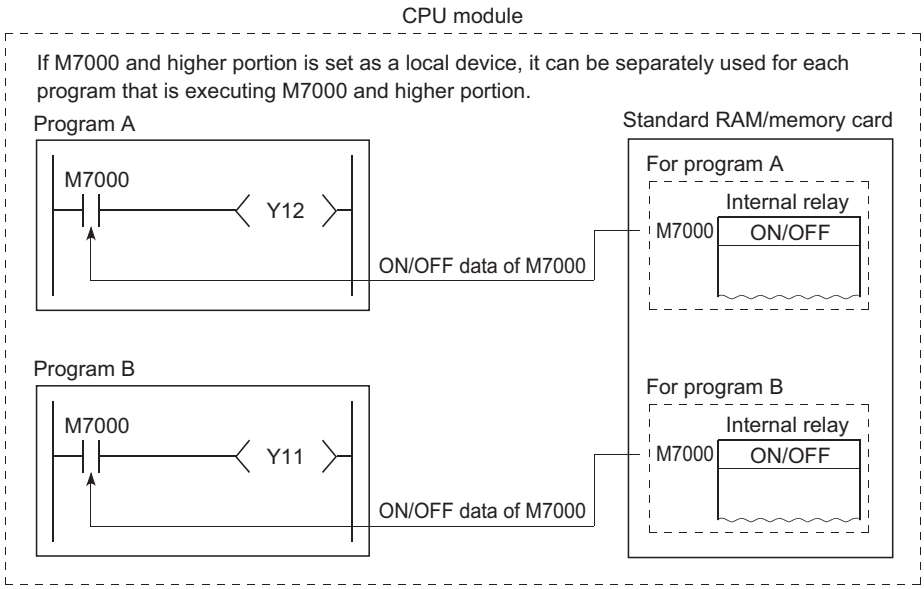
Point

- All of the devices that have not been set as local devices (☞ Page 422, Section 6.2) are global devices.
- For execution of multiple programs, the range to be shared by all programs and the range to be used independently by each program must be specified in advance.



6.2 Local Device Note 6.1

The local device is a device that can be used independently for each program.
Using local devices allows programming of multiple independently-executed programs without considering other programs.
Note that local device data can be stored in the standard RAM and a memory card (SRAM) only.



(1) Devices that can be used as local devices

The following devices can be used as local devices.

- Internal relay (M)
- Edge relay (V)
- Timer (T, ST)
- Counter (C)
- Data register (D)
- Index register (Z) Note 6.2

Note 6.1 Universal

The Q00UJCPU does not support the use of local devices.

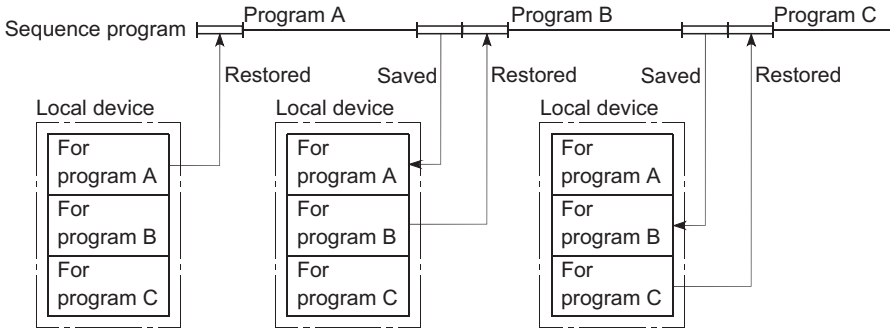
Note 6.2 Universal

When using the index register as a local device with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, or QnUDE(H)CPU, check the versions of the CPU module and programming tool used.

(Page 466, Appendix 2)

(2) Saving and restoring a local device file

When some programs use a local device, respective local device file data in the standard RAM or a memory card (SRAM) are exchanged with the device memory data of the CPU module after execution of each program.
For this reason, the scan time increases by the time spent for data exchange.



Remark

- There are some instructions for which a local device cannot be specified.
  For details, refer to the pages describing devices available for each instruction in the following manual.
  MELSEC-Q/L Programming Manual (Common Instruction)
- For the concept of the number of words used for the local devices, refer to Page 345, Section 4.2.

(3) Local device setting

(a) Setting the local device range

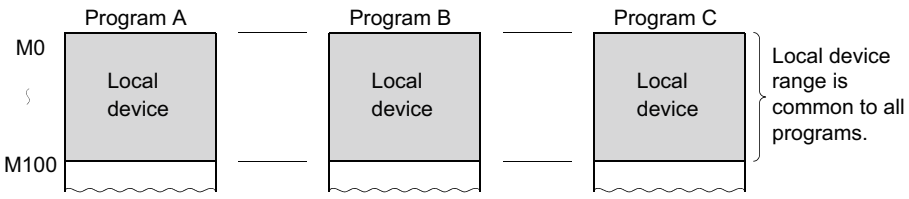
In the Device tab of the PLC parameter dialog box, set the range that is used as a local device.

	Sym.	Dig.	Device Points	Latch (1) Start	Latch (1) End	Latch (2) Start	Latch (2) End	Local Device Start	Local Device End
Input Relay	X	16	8K						
Output Relay	Y	16	8K						
Internal Relay	M	10	15K						
Latch Relay	L	10	8K						
Link Relay	B	16	8K						
Annunciator	F	10	2K						
Link Special	SB	16	2K						
Edge Relay	V	10	2K						
Step Relay	S	10	8K						
Timer	T	10	2K						
Retentive Timer	ST	10	0K						
Counter	C	10	1K						
Data Register	D	10	22K						
Link Register	W	16	8K						
Link Special	SW	16	2K						
Index	Z	10	20						

Device Total 39.2 K Words
Word Device 35.0 K Words
Bit Device 51.0 K Bits

The total number of device points is up to 40K words.
Latch(1) : Able to clear the value by using latch clear.
Latch(2) : Unable to clear the value by using latch clear. Clearing will be executed by program.
Scan time is extended by the latch range setting (including L).
If the latch is necessary, please set the required minimum latch range.
When using the local devices, please do the file setting at PLC file setting parameter.

Note that the local device range is common to all programs, and cannot be changed for each program.
For example, if a local device range is specified as M0 to M100, this range setting applies to all programs that use the local device.

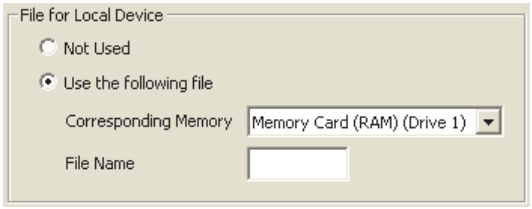


Point

- The 32-bit index modification range must not overlap with the local device setting range of the index register. If overlapped, 32-bit index modification values will be written over the local device values.
- When CPU module parameters which contain local device setting of the index register are read from a programming tool that does not support the setting, all of the index register data will be read out as global device data.

(b) Setting the drive and file name

After setting the local device range, set a memory for storing the local device file and a file name in the PLC file tab of the PLC parameter dialog box.



(c) Writing the setting data

Write the data set in (a) and (b) to the CPU module.

[Online] ⇨ [Write to PLC]

Point

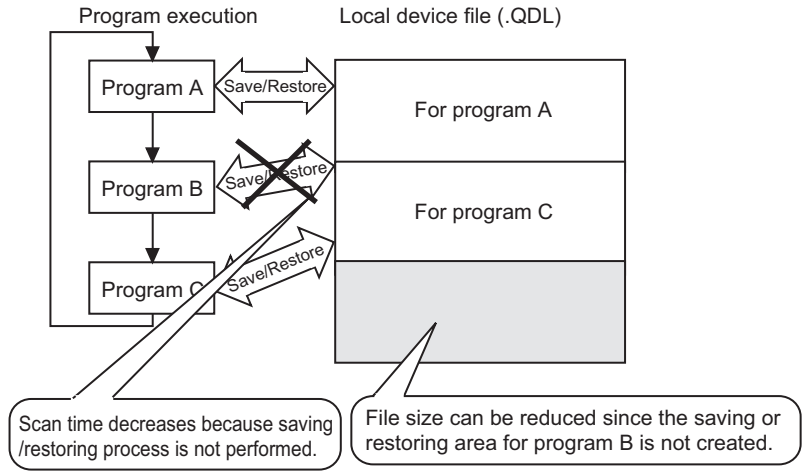
- If the size setting of the local device in the standard RAM is changed with a sampling trace file stored in the standard RAM, the sampling trace file is cleared. To save the trace results in your personal computer, perform the following operations.

  [Debug] ⇨ [Sampling Trace] ⇨ [Read from PLC]

- All devices that are not set as local devices are global devices.

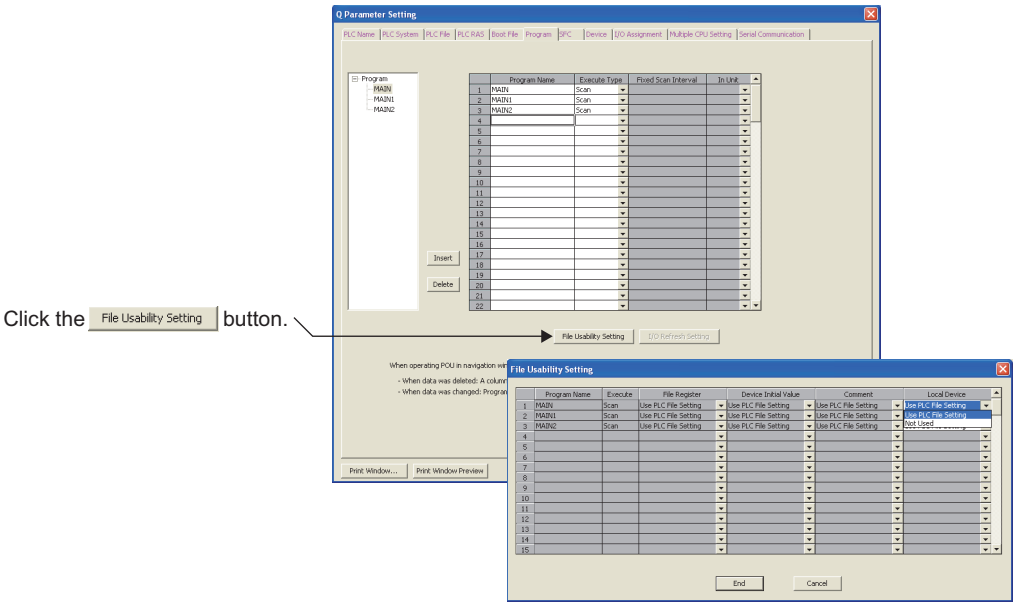
(4) Setting of whether to use a local device (for each program) Note 6.3

Use of the local device can be set for each program, and this function can reduce the scan time.
Also, since the area for saving and restoring data is not required for the programs not using a local device, the local device file size can be reduced.



(a) Setting method

In addition to the setting in (3) in this section, set the following.
Select the File Usability Setting button in the Program tab of the PLC parameter dialog box, and specify the programs that use the local device.



Note 6.3 Universal

When setting local devices for each program with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

(b) Precautions

- Change of the local device
  Do not change or refer to the local device in a program for which the local device is set to "Not Used". Even if the local device is changed in such a program, the changed data will not be held.
- Conditions for creating a local device file
  Creation of a local device file depends on the PLC parameter settings. Creation of a local device file depends on the PLC parameter settings. The following table shows the conditions to create a local device file.

○ : Created, × : Not created

PLC parameter setting			File creation	Error detection
PLC file setting	Device setting*1	File Usability Setting		
Set	Set	Use PLC File Setting	○	-
		Not Used	○	-
	Not set	Use PLC File Setting	×	-
		Not Used	×	-
Not set	Set	Use PLC File Setting	×	PARAMETER ERROR (error code: 3000)
		Not Used	×	-
	Not set	Use PLC File Setting	×	-
		Not Used	×	-

*1 Indicates the local device range setting in the Device tab.

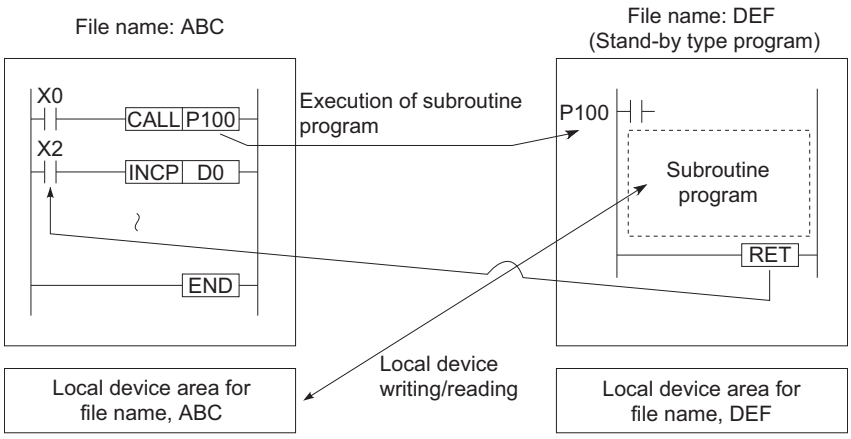
(5) Using the local device corresponding to the file where a subroutine program is stored

When a subroutine program is executed, the local device corresponding to the file where the subroutine program is stored can be utilized.

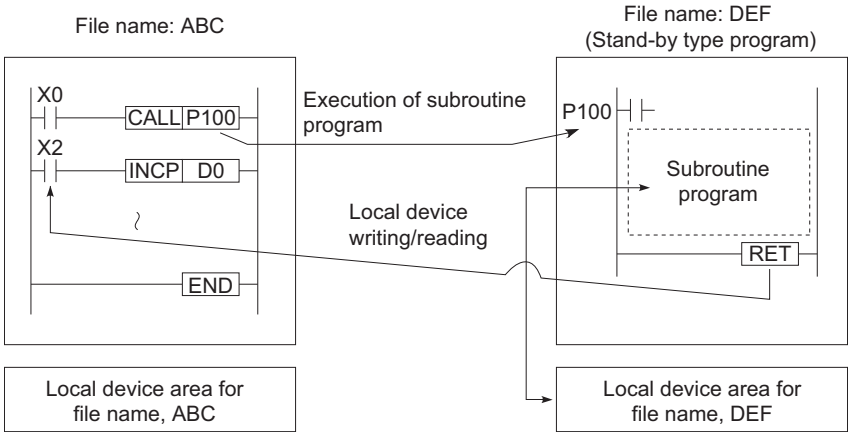
Use of the relevant local device is set by ON/OFF of SM776.

SM776	Operation
OFF	Perform operations with the local device that corresponds to the source file of the subroutine program.
ON	Perform operations with the local device that corresponds to the file where the subroutine program is stored.

(a) When SM776 is off



(b) When SM776 is on



(c) Precautions

- When SM776 is on, local device data are read out when a subroutine program is called, and the data are saved after execution of the RET instruction.
  Because of this, the scan time is increased if one subroutine program is executed with SM776 set to on.
- The on/off status of SM776 is set for each CPU module.
  It cannot be set for each file.
- If the on/off status of SM776 is changed during sequence program execution, control is implemented according to the information after the change.

Remark

For details of SM776, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

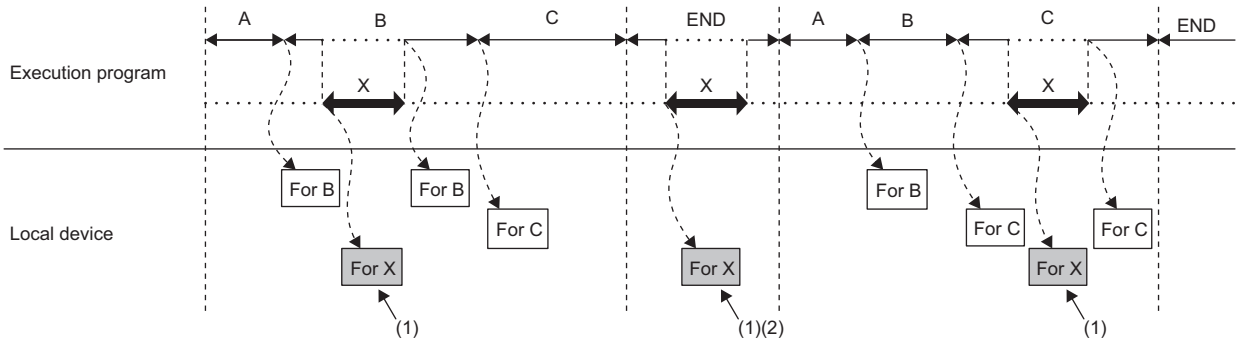
(6) Usage of the local device when an interrupt/fixed scan execution type program is executed

When the local device is used for an interrupt/fixed scan execution type program, turn on SM777 (Enable/disable local device in interrupt program). The programs will not function properly if SM777 is turned off.

*1 The index register set as the local device uses the local device area for the program executed before the interrupt/fixed scan execution type program, regardless of the on/off status of SM777.

Ex. Operation example when SM777 is turned on with the following setting

Program name	Execution type	Local device
A	Scan	Not used
B	Scan	Used
C	Scan	Used
X	Fixed scan	Used



(1) Uses the program X local device.

(2) When an interrupt/fixed scan execution type program is executed during the END processing, the local device for Program C, which was executed before the END processing, is read out and saved. Thus, the END processing time increases by the time required for the read and save.

(a) Precautions

- When SM777 is on, local device data are read out before execution of an interrupt/fixed scan execution type program, and the data are saved after execution of the IRET instruction.
  Because of this, the scan time is increased if one interrupt/fixed scan execution type program is executed with SM777 set to on.
- The on/off status of SM777 is set for each CPU module.
  It cannot be set for each file.
- For the local device monitoring, the target local device is monitored by switching the selection of corresponding program manually. When SM777 is off, when an interrupt occurred immediate after the switch processing, monitoring target local device is monitored. (The local device for the program executed prior to the interrupt (the program immediate before END) is not monitored.)

Remark

For details of SM777, refer to the following.

QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(7) Clearing local device data

Local device data is cleared by either of the following:

- When the CPU module is powered off and then on or is reset
- When the CPU module status is changed from STOP to RUN

Local device data cannot be cleared using a programming tool.

APPENDICES

Appendix 1 Parameters

This chapter describes parameters set for programmable controller systems.

(1) Parameter types

The following parameters are provided for CPU modules.

- PLC parameters (☞ Page 438, Appendix 1.2)
  These parameters are set when a CPU module is used stand alone in a system.
- Network parameters (☞ Page 458, Appendix 1.3)
  These parameters are set when a CPU module is used in combination with network modules, such as CC-Link IE Controller Network modules, MELSECNET/H modules, Ethernet modules, and CC-Link modules.
- Remote password (☞ Page 464, Appendix 1.4)
  This parameter is set when the remote password function of Built-in Ethernet port QCPUs, Ethernet modules, or serial communication modules is used.

(2) Parameter setting method

Use a programming tool.

For the setting, refer to the following.

📖 Operating manual for the programming tool used

Point

The setting cannot be done in the grayed-out area (not selectable) of the screen because the function of the area is not available.

Remark

Each parameter number shown in the tables in this chapter is stored in the special register (SD16 to SD26) when an error occurs in parameter setting.
Identify the parameter error location from the parameter number.

Appendix 1.1 List of parameter numbers

Each parameter number will be stored in SD16 to SD26 when an error occurs in the parameter settings. The following table lists the parameter items and corresponding parameter numbers. For explanation of M and N shown in the "Parameter No." column, refer to Appendix 1.3.

Parameter No.	Item		Set in:	Reference
0000_H	Label		PLC Name	Page 438, Appendix 1.2.1
0001_H	Comment			
0400_H	I/O Assignment	Type	I/O Assignment	Page 59, Section 2.3.2, Page 450, Appendix 1.2.9
		Model Name		
		Points		
		Start XY (Start I/O number)		
0401_H	Base Setting	Base Model Name		Page 54, Section 2.2.2, Page 450, Appendix 1.2.9
		Power Model Name		
		Extension Cable		
		Slots		
0403_H	Detailed Setting	Error Time Output Mode		Page 141, Section 3.8, Page 450, Appendix 1.2.9
0405_H		I/O Response Time		Page 139, Section 3.7, Page 450, Appendix 1.2.9
0406_H		Control PLC		QCPU User's Manual (Multiple CPU System)
0409_H	Switch Setting			Page 450, Appendix 1.2.9
$0E00_H$	No. of PLC		Multiple CPU Setting	Page 452, Appendix 1.2.10, QCPU User's Manual (Multiple CPU System)
$0E01_H$	Operation Mode			
$0E04_H$	I/O Sharing When Using Multiple CPUs	All CPUs Can Read All Inputs		
		All CPUs Can Read All Outputs		

Parameter No.	Item		Set in:	Reference
1000_H	Timer Limit Setting	Low Speed	PLC System	Page 439, Appendix 1.2.2
		High Speed		
1001_H	RUN-PAUSE Contacts	RUN		
		PAUSE		
1002_H	Remote Reset			Page 136, Section 3.6.3, Page 439, Appendix 1.2.2
1003_H	Output Mode at STOP to RUN			Page 125, Section 3.4, Page 439, Appendix 1.2.2
1005_H	Common Pointer No.			Page 439, Appendix 1.2.2
1007_H	Points Occupied by Empty Slot			
1008_H	Interrupt Program/Fixed Scan Program Setting			Page 82, Section 2.9, Page 98, Section 2.10.4, Page 439, Appendix 1.2.2
	System Interrupt Settings	Fixed Scan Interval (n : 28 to 31)		Page 439, Appendix 1.2.2
$100F_H$	High Speed I/O Refresh Setting	X Input		
1010_H		Y Output		
1011_H	High Speed Buffer Transfer Setting	Buffer Read		
1012_H		Buffer Write		
$100A_H$	Intelligent Function Module Setting (Interrupt Pointer Setting)			
$100C_H$	Module Synchronization			
$100E_H$	Use Serial Communication		Serial Communication	Page 233, Section 3.23, Page 456, Appendix 1.2.12
	Transmission Speed			
	Sum Check			
	Transmission Wait Time			
	Online Change			
1013_H	Service Processing Setting		PLC System	Page 241, Section 3.24.1, Page 439, Appendix 1.2.2
1014_H	Latch Data Backup Operation Valid Contact			Page 254, Section 3.29, Page 439, Appendix 1.2.2
1016_H	Ethernet port setting		Built-in Ethernet port setting	Page 454, Appendix 1.2.11
1017_H	PLC Module Change Setting		PLC System	Page 439, Appendix 1.2.2
1019_H	Simple PLC communication function		Built-in Ethernet port setting	Page 454, Appendix 1.2.11
$101F_H$	MELSOFT Connection Extended Setting		Built-in Ethernet port setting	Page 454, Appendix 1.2.11
1030_H	CC-Link IEF Basic Setting	Network Configuration Setting		
1031_H		Refresh Setting		

Parameter No.	Item		Set in:	Reference
1100_H	File Register		PLC File	Page 392, Section 4.7, Page 441, Appendix 1.2.3
1101_H	Comment File Used in a Command			Page 441, Appendix 1.2.3
1102_H	Device Initial Value			Page 247, Section 3.25, Page 441, Appendix 1.2.3
1103_H	File for Local Device			Page 422, Section 6.2, Page 441, Appendix 1.2.3
1104_H	Transfer to Standard ROM at Latch data backup operation.			Page 254, Section 3.29, Page 441, Appendix 1.2.3
1105_H	File used for SP.DEVST/S.DEVLD Instruction			Page 259, Section 3.30, Page 441, Appendix 1.2.3
2000_H	Device Points		Device	Page 336, Section 4.1, Page 447, Appendix 1.2.8
2001_H	Latch (1) Start/End			Page 122, Section 3.3, Page 447, Appendix 1.2.8
2002_H	Latch (2) Start/End			
2003_H	Local Device Start/End			Page 447, Appendix 1.2.8
2000_H	File Register Extended Setting	Device Points		Page 402, Section 4.8, Page 447, Appendix 1.2.8
2004_H		Latch (1) Start/End		
2005_H		Latch (2) Start/End		
2000_H	Indexing Setting for ZR Device			Page 387, Section 4.6, Page 447, Appendix 1.2.8
2006_H	Latch Interval Setting			Page 122, Section 3.3, Page 447, Appendix 1.2.8
2007_H	Disable device write from external		Device	Page 306, Section 3.37, Page 447, Appendix 1.2.8
	Write Protection Start/End			
3000_H	WDT (Watchdog Timer) Setting	WDT Setting	PLC RAS	Page 193, Section 3.16, Page 442, Appendix 1.2.4
		Initial Execution Monitoring Time		Page 92, Section 2.10.1, Page 442, Appendix 1.2.4

Parameter No.	Item		Set in:	Reference
3001_H	Error Check	Carry Out Battery Check	PLC RAS	Page 195, Section 3.17, Page 442, Appendix 1.2.4
		Carry Out Fuse Blown Check		
		Verify Module		
		Check Device Range at Indexing		
		Diagnose Redundant Power Supply System		
3002_H	Operating Mode When There is an Error	Computation Error		
		Expanded Command Error		
		Fuse Blown		
		Module Verify Error		
		Intelligent Module Program Execution Error		
		File Access Error		
		Memory Card Operation Error		
		External Power Supply OFF		
3003_H	Constant Scanning			Page 119, Section 3.2, Page 442, Appendix 1.2.4
$300A_H$	Module Error History Collection			Page 298, Section 3.34, Page 442, Appendix 1.2.4
$300B_H$	Operation History			Page 314, Section 3.38 Page 442, Appendix 1.2.4
4004_H	Detailed Setting	PLC Operation Mode at H/W Error	I/O Assignment	Page 142, Section 3.9, Page 450, Appendix 1.2.9
5000_H	Number of modules on MELSECNET/H		MELSECNET/H	Page 461, Appendix 1.3.3
5001_H	Valid Module During Other Station Access			
5002_H	Interlink Transmission Parameters			
5003_H	Routing Parameters			
$5NM0_H$	Start I/O No.			
	Network No.			
	Total Stations			
$5NM0_H$	Mode			
$5NM1_H$	Refresh Parameters			
$5NM2_H$	Common Parameters			
$5NM3_H$	Station Inherent Parameters			
$5NM5_H$	Sub-master parameters			
$5NMA_H$	Common Parameters 2			
$5NMB_H$	Station Inherent Parameters 2			
	Interrupt Settings			
7000_H	Program		Program	Page 88, Section 2.10, Page 445, Appendix 1.2.6
7000_H	Boot Option	Clear Program Memory	Boot File	Page 104, Section 2.11, Page 444, Appendix 1.2.5
		Auto Download All Data from Memory Card to Standard ROM		
	Boot File Setting			
8002_H	SFC Program Start Mode		SFC	Page 446, Appendix 1.2.7
8003_H	Start Conditions			
8006_H	Output Mode When the Block is Stopped			

Parameter No.	Item	Set in:	Reference
9000_H	Number of modules on Ethernet	Ethernet	Page 462, Appendix 1.3.4
$9N00_H$	Start I/O No.		
	Network No.		
	Group No.		
	Station No.		
	Operation Setting		
$9N01_H$	Initial Setting		
$9N02_H$	Open Setting		
$9N03_H$	Router Relay Parameter		
$9N05_H$	Station No.<->IP Information		
$9N06_H$	FTP Parameters		
$9N07_H$	E-mail Setting		
$9N08_H$	News Setting		
$9N09_H$	Interrupt Settings		
$9N04_H$	Routing Parameters		
$A000_H$	Number of modules on CC-Link IE Controller Network	CC-Link IE Controller Network, CC-Link IE Field Network	Page 459, Appendix 1.3.1, Page 460, Appendix 1.3.2
$A002_H$	Interlink Transmission Parameters		
$A003_H$	Routing Parameters		
$A080_H$	Network Type		
$A082_H$	Interlink Transmission Parameters		
$ANM0_H$	Start I/O No.		
	Network No.		
	Total Stations		
	Station No.		
$ANM0_H$	Mode		
$ANM1_H$	Refresh Parameters		
$ANM2_H$	Common Parameters		
	Network Configuration Settings		
$ANM3_H$	Station Inherent Parameters		
	Network Operation Settings		
	IP Address Setting		
	IP Address		
	Interrupt Settings		

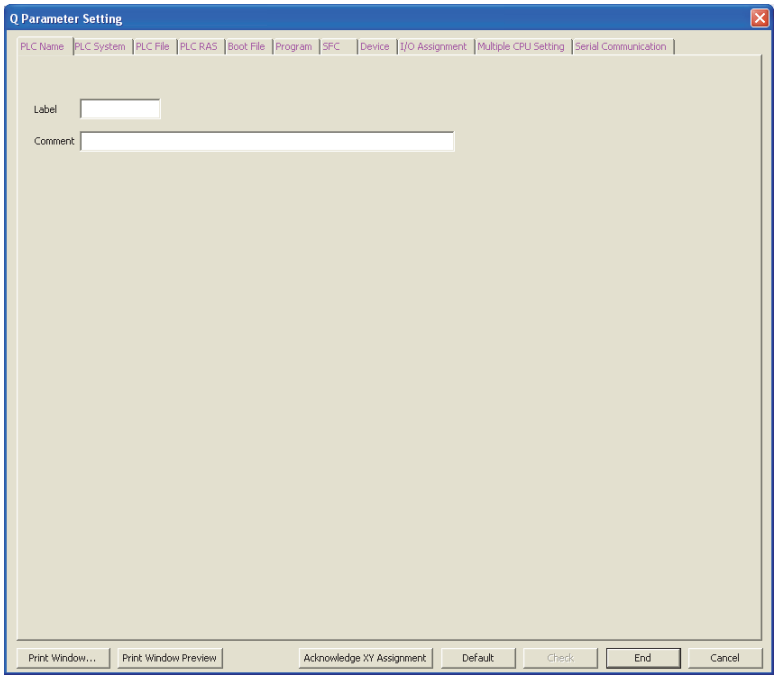
Parameter No.	Item		Set in:	Reference
$C000_H$	Number of Modules		CC-Link	Page 463, Appendix 1.3.5
$CNM1_H$	Remote Input (RX)			
	Remote Output (RY)			
	Remote Register (RWr)			
	Remote Register (RWw)			
	Ver.2 Remote Input (RX)			
	Ver.2 Remote Output (RY)			
	Ver.2 Remote Register (RWr)			
	Ver.2 Remote Register (RWw)			
	Special Relay (SB)			
	Special Register (SW)			
$CNM2_H$	Start I/O No.			
	Operation Setting			
	Total Module Connected			
	Retry Count			
	Automatic Reconnection Station Count			
	Standby Master Station No.			
	PLC Down Select			
	Scan Mode Setting			
	Delay Time Setting			
	Station Information Setting			
	Remote Device Station Initial Setting			
	Interrupt Settings			
$E002_H$ $E003_H$	Communication Area Setting (Refresh Setting)		Multiple CPU	Page 452, Appendix 1.2.10, QCPU User's Manual (Multiple CPU System)
$E006_H$	Online Module Change			
$E007_H$	Refresh parameter detailed device specification			
$E008_H$	Multiple CPU High Speed Transmission Area Setting	CPU Specific Send Range		
$E009_H$		Auto Refresh		
$E00B_H$	Multiple CPU Synchronous Startup Setting			
$E00C_H$	Host Station			
$EF00_H$	Programming tool parameters		-	-

Appendix 1.2 PLC parameters

This section describes PLC parameter details with setting windows.

Appendix 1.2.1 PLC name

A label and a comment for the CPU module are set. The settings will be displayed in the list for the find CPU function. Note Appx.1



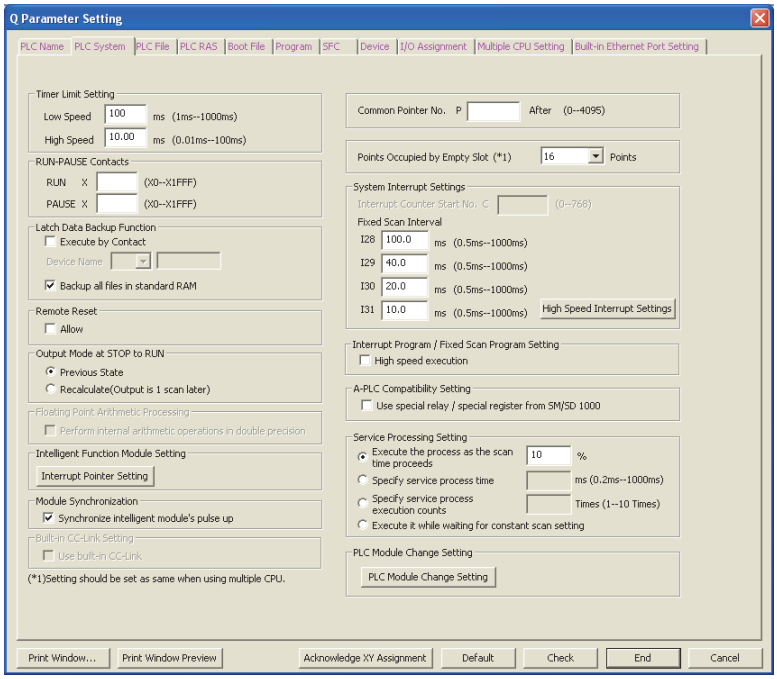
Item	Parameter No.	Description	Setting range	Default	Reference
Label	0000_H	Set a label (name, application) for the CPU module.	Up to 10 characters	Blank	-
Comment	0001_H	Set a comment for the CPU module label.	Up to 64 characters	Blank	-

Note Appx.1 Universal

The Universal model QCPUs other than the Built-in Ethernet port QCPU do not support the find CPU function.

Appendix 1.2.2 PLC system

Parameters required for use of the CPU module are set.

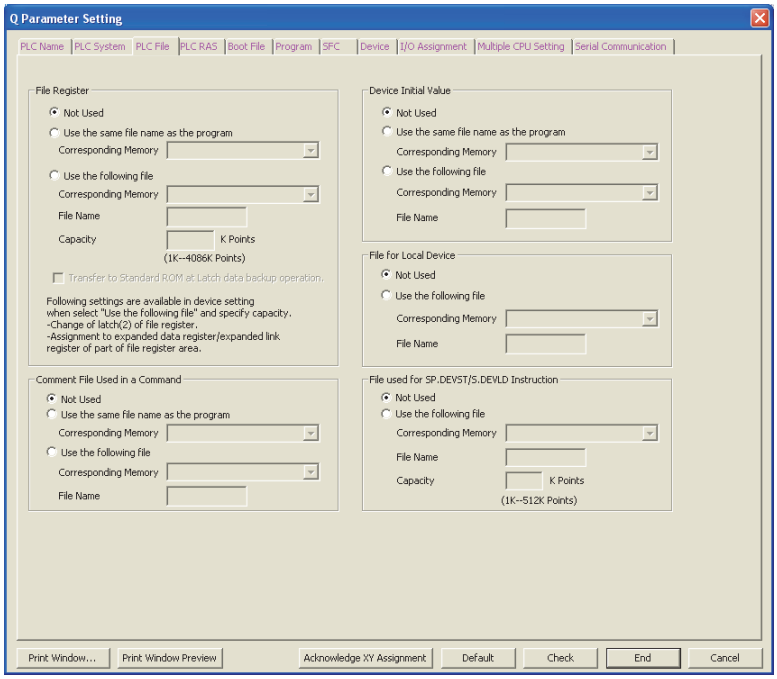


Item		Parameter No.	Description	Setting range	Default	Reference
Timer Limit Setting	Low Speed	1000_H	Set the time limit for the low speed timer or high speed timer.	1ms to 1000ms (in increments of 1ms)	100ms	Page 360, Section 4.2.10
	High Speed			0.01ms to 100.0ms (in increments of 0.01ms)	10.0ms	
RUN-PAUSE Contacts	RUN	1001_H	Set the contacts that control RUN/PAUSE of the CPU module. Setting of only the PAUSE contact is not allowed.	X0 to 1FFF	Blank	Page 131, Section 3.6.1
	PAUSE					Page 134, Section 3.6.2
Latch Data Backup Operation	Execute by contact	1014_H	Select to back up data using a contact. When selected, specify the device number used as a contact.	X, M, B	Blank	Page 254, Section 3.29
	Backup all files in the internal of standard RAM		Select to back up all files in the standard RAM.	-	Selected	
Remote Reset		1002_H	Select whether to allow the remote reset from programming tool.	Selected/deselected	Deselected	Page 136, Section 3.6.3
Output Mode at STOP to RUN		1003_H	Set the status of the outputs (Y) when the operating status is switched from STOP to RUN.	Previous State, Recalculate (Output is 1 scan later)	Previous state	Page 125, Section 3.4
Intelligent Function Module Setting (Interrupt Pointer Setting)		$100A_H$	Assign the interrupt pointers (I50 to I255) and set the start I/O number and start SI number of each intelligent function module.	• Start I/O No. • Start SI No. • I50 to 255	Blank	Page 412, Section 4.11
Module Synchronization		$100C_H$	Select whether to synchronize CPU module startup with intelligent function module startup.	Selected/deselected	Selected	-

Item		Parameter No.	Description	Setting range	Default	Reference
Common Pointer No.		1005_H	Set the start number of common pointers.	P0 to 4095	Blank	Page 411, Section 4.10.2
Points Occupied by Empty Slot		1007_H	Set the number of points for empty slots on the main/extension base units.	0, 16, 32, 64, 128, 256, 512, or 1024 Points	16 Points	Page 54, Section 2.2.2
System Interrupt Settings	Fixed Scan Interval (n: 28 to 31)	1008_H	Set each execution interval for the interrupt pointers (I28 to I31).	0.5ms to 1000ms (in increments of 0.5ms)	• I28: 100.0ms • I29: 40.0ms • I30: 20.0ms • I31: 10.0ms	Page 412, Section 4.11
	High Speed Interrupt Setting	$100F_H$ to 1012_H	Set this parameter when a high-speed interrupt pointer (I49) is used.	-	-	Page 225, Section 3.21
Interrupt Program/Fixed Scan Program Setting		1008_H	Enable or disable high speed execution of interrupt programs or fixed scan programs.	Selected/deselected	Deselected	Page 82, Section 2.9, Page 98, Section 2.10.4
Service Processing Setting		1013_H	Select any of the following options. • Execute the process as the scan time proceeds. • Specify service process time. • Specify service process execution counts. • Execute it while waiting for constant scan setting.	• 1 to 99% (in increments of 1%) • 1 to 10 (in increments of 1 time) • 0.2 to 1000ms (in increments of 0.1ms) • Blank	Execute the process as the scan time proceeds.: 10%	Page 241, Section 3.24.1
PLC Module Change Setting (PLC Module Change Setting)		1017_H	Set items required when performing the CPU module change with memory card function.	• Backup Start Setup Contact • Backup Start Contact • Backup Target Data • Title Setting	Blank	Page 260, Section 3.31

Appendix 1.2.3 PLC file

Parameters required for the files used in the CPU module are set.

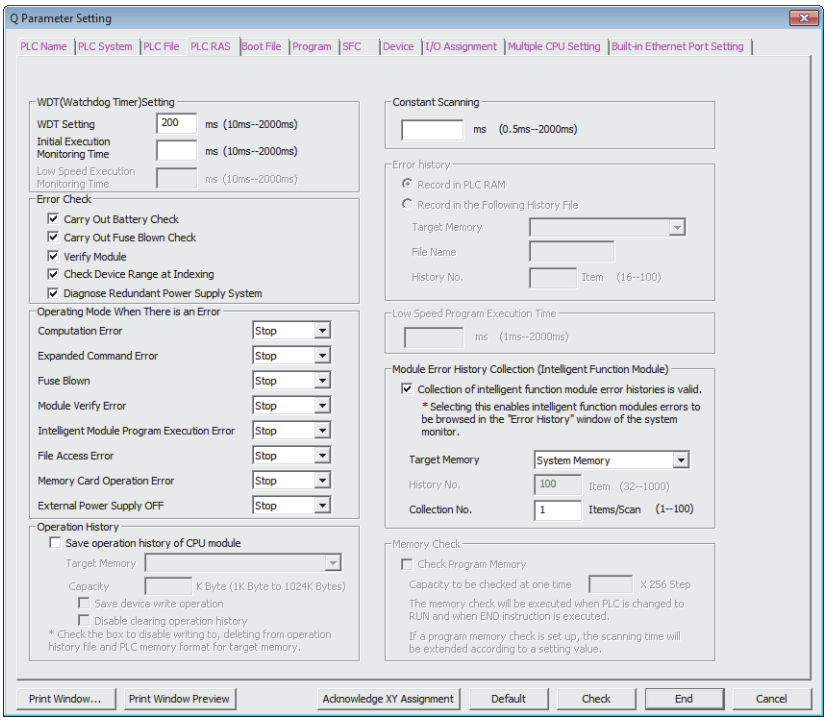


Item	Parameter No.	Description	Setting range	Default	Reference
File Register*1	1100_H	Set a file for the file register used in the program.	• Not Used • Use the same file name as the program. • Use the following file.	Not Used	Page 392, Section 4.7
Transfer to Standard ROM at Latch data backup operation*1	1104_H	Select whether to batch-transfer the data in the file register at the time of latch data backup to the standard ROM.	Checked/unchecked	Unchecked	Page 254, Section 3.29
Comment File Used in a Command	1101_H	Set a file for device comments used in the program.	• Not Used • Use the same file name as the program. • Use the following file.	Not Used	-
Device Initial Value	1102_H	Set a file for initial values of the devices used for the CPU module.	• Not Used • Use the same file name as the program. • Use the following file.	Not Used	Page 247, Section 3.25
File for Local Device*1	1103_H	Set a file for local devices used in the program.	• Not Used • Use the following file.	Not Used	Page 422, Section 6.2
File used for SP.DEVST/S.DEVLD Instruction	1105_H	Set a device data file used for writing to or reading from the standard ROM.	• Not Used • Use the following file.	Not Used	Page 259, Section 3.30

*1 Not available for the Q00UJCPU.

Appendix 1.2.4 PLC RAS

Parameters required for performing the RAS functions are set.



Item		Parameter No.	Description	Setting range	Default	Reference
WDT (Watchdog Timer) Setting	WDT Setting	3000_H	Set a watchdog timer value for the CPU module.	10ms to 2000ms (in increments of 10ms)	200ms	Page 193, Section 3.16
	Initial Execution Monitoring Time		Set a watchdog timer value in the case of using an initial execution type program.	10ms to 2000ms (in increments of 10ms)	Blank	Page 92, Section 2.10.1
Operating Mode When There is an Error	Computation Error	3002_H	Set the operation mode of the CPU module when an error is detected.	Stop/Continue	Stop	Page 195, Section 3.17
	Expanded Command Error*1					
	Fuse Blown					
	Module Verify Error					
	Intelligent Module Program Execution Error					
	File Access Error					
	Memory Card Operation Error*3					
	External Power Supply OFF*1					
Error Check	Carry Out Battery Check	3001_H	Enable or disable detection of the specified error.	Selected/deselected	Deselected	Page 195, Section 3.17
	Carry Out Fuse Blown Check					
	Verify Module					
	Check Device Range at Indexing					
	Diagnose Redundant Power Supply System*2					
Constant Scanning		3003_H	Set a constant scan time value.	0.5ms to 2000ms*4	Blank	Page 119, Section 3.2

Item		Parameter No.	Description	Setting range	Default	Reference
Module Error History Collection (Intelligent Function Module)	Collection of intelligent function module error histories is valid.	$300A_H$	Set whether to collect module errors.	Selected/deselected	Selected	Page 298, Section 3.34
	Corresponding Memory		Select a storage location.	• System Memory • Standard RAM	System Memory	
	History No.		Set the number of collected errors only when the errors are stored in the standard RAM.	32 to 1000	40/100	
	Collection No.		Set the number of collected errors in one scan.	• Stored in system memory: 1 to 100 • Stored in standard RAM: 1 to 128	1	
Operation History*5	Save operation history of CPU module	$300B_H$	Set whether to use the operation history function.	-	Unchecked	Page 314, Section 3.38
	Target Memory		Set the destination memory to save the operation history file.	• Standard ROM (Drive 4) • Memory Card (SD) (Drive 2)	Standard ROM (Drive 4)	
	Capacity		Set the file size of the operation history file. (☞ Page 321, Section 3.38.1 (2) (d))	1K to 1024K bytes	128K bytes	
	Save device write operation		Set whether to save the operation of device data writing from outside the CPU module.	-	Unchecked	
	Disable clearing operation history		Set whether to disable the clear operation of the operation history. (☞ Page 326, Section 3.38.3)	-	Unchecked	

*1 These items are provided for future expansion.

*2 When selecting this item for the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool. (☞ Page 466, Appendix 2)

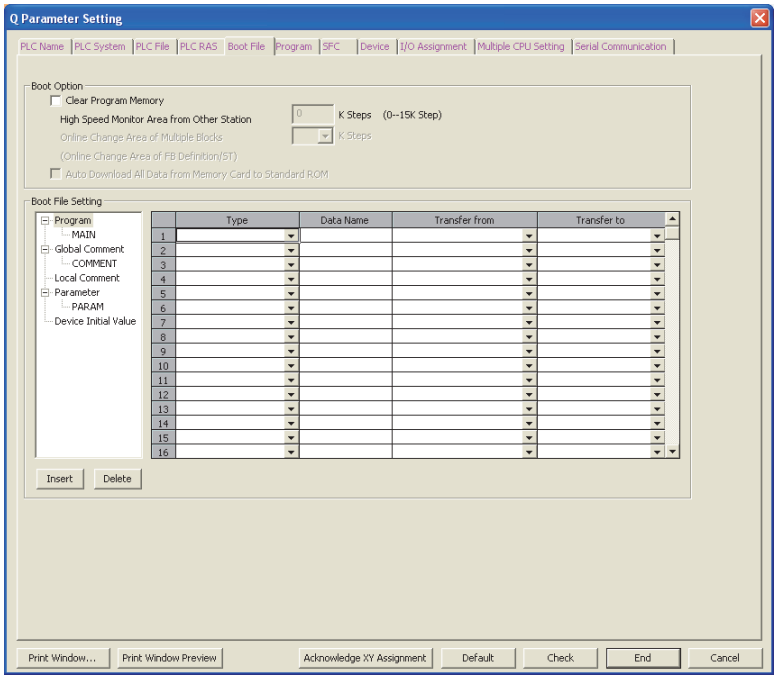
*3 Not available for the Q00UJCPU, Q00UCPU, and Q01UCPU.

*4 The setting value differs depending on the CPU module used.
CPU modules other than High-speed Universal model QCPU and Universal model Process CPU: in increments of 0.5ms
High-speed Universal model QCPU and Universal model Process CPU: in increments of 0.1ms

*5 This setting item is available to the High-speed Universal model QCPU and Universal model Process CPU. Before setting this item, check the versions of the CPU module and GX Works2 used. (☞ Page 466, Appendix 2)

Appendix 1.2.5 Boot file

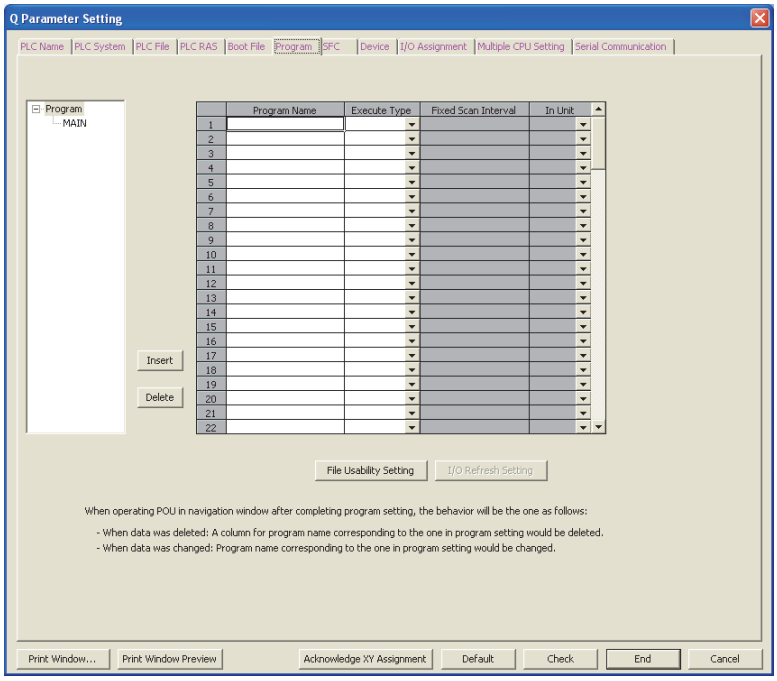
Parameters required for a boot from a memory card are set.



Item		Parameter No.	Description	Setting range	Default	Reference
Boot Option	Clear Program Memory	7000_H	Select whether to clear the program memory at the time of boot.	Selected/deselected	Deselected	Page 104, Section 2.11
Boot File Setting			Set the type and data name of the boot file, and transfer source drive for boot operation.	Type, Data Name, and Transfer from (The transfer target drive (Transfer to) is automatically set in the program memory.)	Blank	

Appendix 1.2.6 Program

File names and execution types (execution conditions) are set for each program when two or more programs are written to the CPU module.

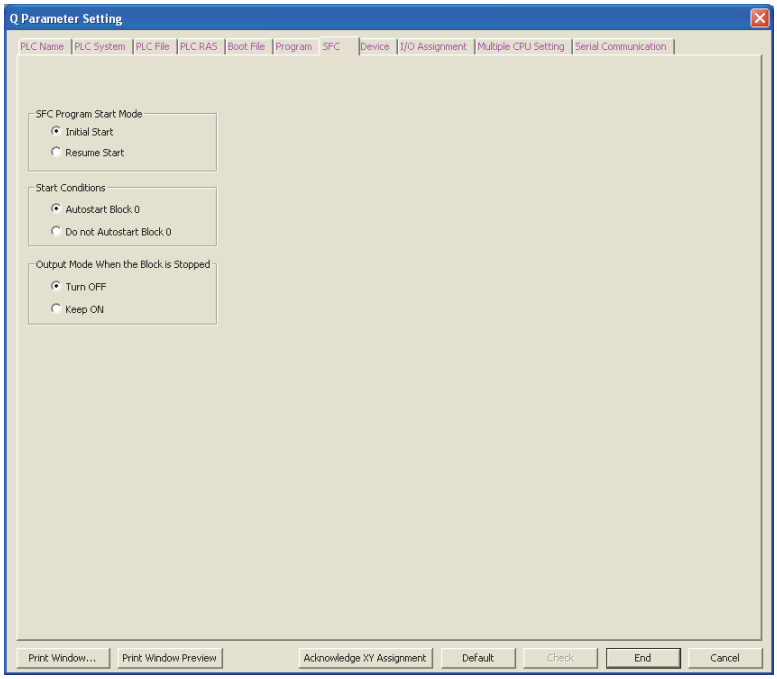


Item	Parameter No.	Description	Setting range	Default	Reference
Program setting	7000_H	When writing two or more programs to the CPU module, set a file name and execution type (execution condition) of each program. Also, set a fixed scan interval (execution interval of the fixed scan execution type program).	• Program Name • Execute Type (fixed scan interval when the fixed scan execution is selected) • File Usability Setting*1	Blank	Page 88, Section 2.10

*1 Available for local devices only. When using the file usability setting, check the versions of the CPU module and programming tool. (☞ Page 466, Appendix 2) The setting is not available for the Q00UJCPU.

Appendix 1.2.7 SFC

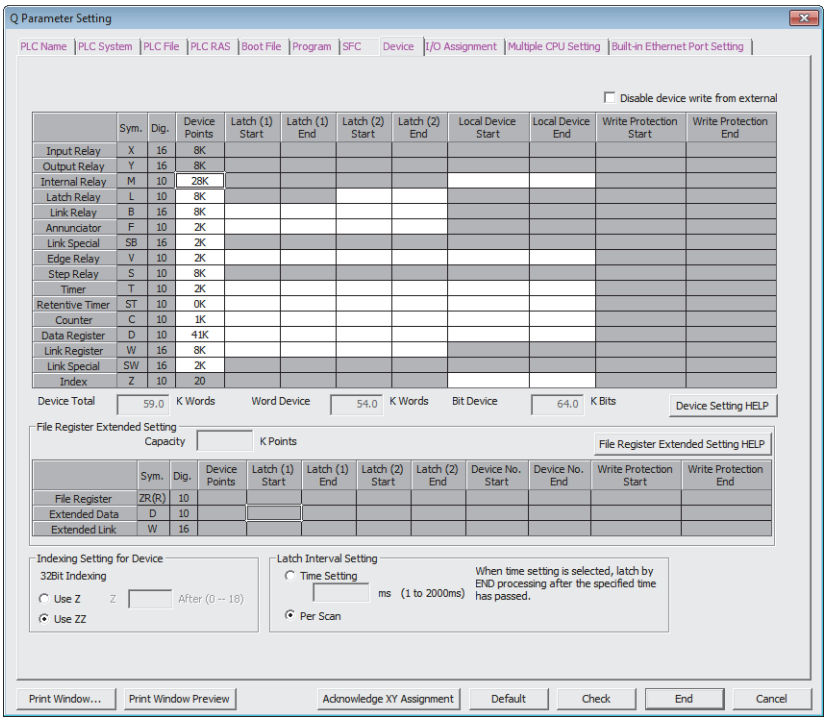
The mode and conditions for starting an SFC program, and the output mode in the case of a block stop are set.



Item	Parameter No.	Description	Setting range	Default	Reference
SFC Program Start Mode	8002_H	Set the mode and conditions for starring an SFC program, and also set the output mode in case a program block is stopped.	MELSEC-Q/L/QnA Programming Manual (SFC)	Initial Start	MELSEC-Q/L/QnA Programming Manual (SFC)
Start Conditions	8003_H			Autostart Block 0	
Output Mode When the Block is Stopped	8006_H			Turn OFF	

Appendix 1.2.8 Device

Number of points, latch range, and local device range are set for each device.



Item	Parameter No.	Description	Setting range	Default	Reference
Device Points[*1]	2000_H	Set the number of device points that is appropriate to the system.	X (8K), Y (8K), and S (8K)[*3] are fixed. Setting is available within the range of 29K words[*8] in total, including the above fixed points (1.5K words). One device: Up to 32K points[*7]	• X: 8K • Y: 8K • M: 8K[*9] • L: 8K[*2] • B: 8K • F: 2K • SB: 2K • V: 2K • S: 8K[*3] • T: 2K • ST: 0K • C: 1K • D: 12K[*10] • W: 8K • SW: 2K	Page 336, Section 4.1
Latch (1) Start/End (Latch clear valid)[*2]	2001_H	Set a latch range (start and end device numbers), which can be cleared by a latch clear operation.	Setting is available for only one range for each of B, F, V, T, ST, C, D, and W devices.	Blank	Page 122, Section 3.3
Latch (2) Start/End (Latch clear invalid)[*2]	2002_H	Set a latch range (start and end device numbers), which cannot be cleared by a latch clear operation.	Setting is available for only one range for each of L, B, F, V, T, ST, C, D, and W devices.	Blank	Page 122, Section 3.3
Local Device Start/End[*6]	2003_H	Set a range (start and end device numbers), which is used for a local device.	Setting is available for only one range for each of M, V, T, ST, C, D, and Z devices.[*5]	Blank	Page 422, Section 6.2

Item		Parameter No.	Description	Setting range	Default	Reference
File Register Extended Setting*6	Device Points	2000_H	Set points for the file register (ZR), extended data register (D), and extended link register (W).	Point assignment to the file register (ZR), extended data register (D), or extended link register (W). Assign part of the points of the file register to the extended data register and extended link register.	Blank	Page 336, Section 4.1, Page 402, Section 4.8
	Latch (1) Start/End (Latch clear valid)	2004_H	Set a latch range (start and end device numbers), which can be cleared by a latch clear operation.	Each latch range for the file register (ZR), extended data register (D), or extended link register (W).	Blank	Page 122, Section 3.3
	Latch (2) Start/End (Latch clear invalid)	2005_H	Set a latch range (start and end device numbers), which cannot be cleared by a latch clear operation.	Each latch range for the file register (ZR), extended data register (D), or extended link register (W).	Blank	Page 122, Section 3.3
Indexing Setting for ZR Device	32Bit Indexing*4	2000_H	Select Z or ZZ device for 32-bit indexing.	Z0 to Z18 (when using Z device)	Use Z	MELSEC-Q/L Programming Manual (Common Instruction)
Latch Interval Setting*6		2006_H	Set a latch interval.	1 to 2000ms	Each Scan	Page 122, Section 3.3
Disable device write from external		2007_H	Set whether to use the write-protect function for device data (from outside the CPU module).	-	Unchecked	Page 306, Section 3.37
Write Protection Start/End			Set the write-protected range (set the start device number and the end device number). To set this item, select the checkbox of "Disable device write from external".	One range per device (available to X, Y, M, L, B, F, SB, V, S, T, ST, C, D, W, SW, Z, ZR (R), extended data register, extended link register)	Blank	

*1 When changing the device points, new setting must not exceed the refresh ranges of network modules or the auto refresh ranges of intelligent function modules. If a new device point setting exceeds the corresponding device range, the data may be written to another device or an error may occur.

*2 When a device latch range is set, the scan time increases. When latching a device, consider the increase in the scan time. (☞ Page 478, Appendix 3.2 (6))

*3 For the Universal model QCPU whose serial number (first five digits) is "10042" or later, the points for step relay (S) can be changed to 0K. For the Universal model QCPU whose serial number (first five digits) is "12052" or later, the points for step relay (S) can be set up to the following in increments of 1K points. (☞ Page 466, Appendix 2)

- Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 8192 points
- Universal model QCPU other than the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 16384 points

*4 Available only when the serial number (first five digits) of the Universal model QCPU is "10042" or later.

*5 When using the index register as a local device with the Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, or QnUDE(H)CPU, check the versions of the CPU module and programming tool used. (☞ Page 466, Appendix 2)

*6 Not available for the Q00UJCPU.

*7 For the Universal model QCPU whose serial number (first five digits) is "10042" or later, the maximum number of points for the internal relay (M) and link relay (B) is 60K. (☞ Page 466, Appendix 2)

*8 The setting range differs depending on the CPU module.

- Q03UDVCPU: 30K words
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: 40K words
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: 60K words

*9 The default value differs depending on the CPU module.

- Q03UDVCPU: 9K points
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: 15K points
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: 29K points

*10 The default value differs depending on the CPU module.

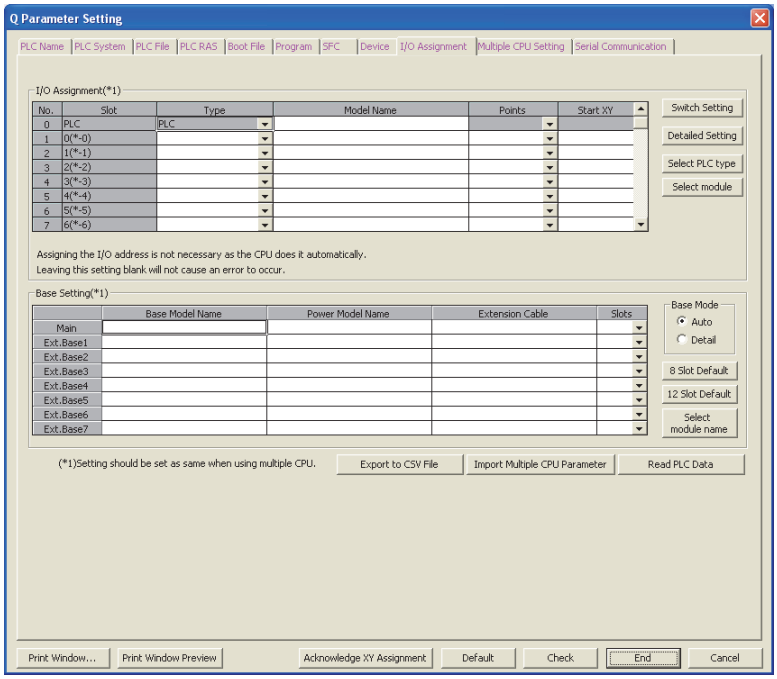
- Q03UDVCPU: 13K points
- Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU: 22K points
- Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU: 41K points

*11 Only the High-speed Universal model QCPU and Universal model Process CPU can select "Time Setting". The latch interval setting is fixed to "Each Scan" for other CPU modules.

*12 This setting item is available to the High-speed Universal model QCPU and Universal model Process CPU. Before setting this item, check the versions of the CPU module and GX Works2 used. (☞ Page 466, Appendix 2)

Appendix 1.2.9 I/O assignment

The mounting status of each module in the system is set.

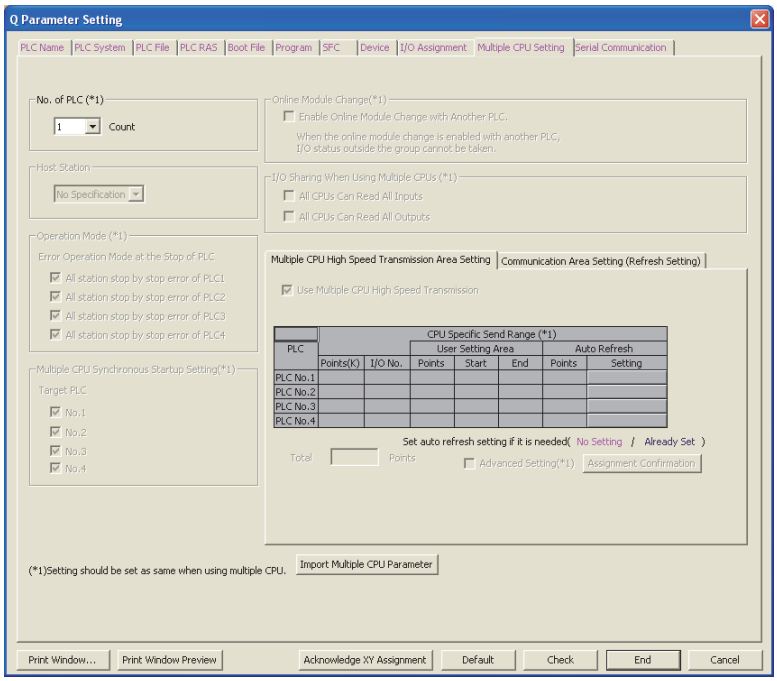


Item		Parameter No.	Description	Setting range	Default	Reference
I/O Assignment	Type	0400_H	Set the type of the mounted module.	• CPU No.2 to No.4: No.n/Empty (Set "CPU (Empty)" for the slot where no CPU module is mounted.) • Empty, Input, Hi. Input, Output, I/O Mix, Intelligent, or Interrupt	Blank	Page 59, Section 2.3.2
	Model Name		Set the model name of the mounted module. (Entered at user's discretion. Do not use the one for the CPU module.)	Up to 16 characters		
	Points		Set the number of points assigned to each slot.	0, 16, 32, 48, 64, 128, 256, 512, or 1024 points		
	Start XY		Set the start I/O number of each slot.	0_H to $FF0_H$		
Base Setting	Base Model Name	0401_H	Set the model name of the main base unit or extension base unit. (Entered at user's discretion. Do not use the one for the CPU module.)	Up to 16 characters	Blank	Page 54, Section 2.2.2
	Power Model Name		Set the model name of the power supply module on the main base unit or extension base unit. (Entered at user's discretion. Do not use the one for the CPU module.)	Up to 16 characters		
	Extension Cable		Set the extension cable name. (Entered at user's discretion. Do not use the one for the CPU module.)	Up to 16 characters		
	Slots		Set the number of slots of the main base unit or extension base unit. This setting must be done for all base units.	2, 3, 5, 8, 10, or 12		
Switch Setting		0407_H	Set various switches of an intelligent function module.	Refer to the manual for the intelligent function module used.	Blank	Page 143, Section 3.10

Item		Parameter No.	Description	Setting range	Default	Reference
Detailed Setting	Error Time Output Mode	0403_H	Set whether to clear or hold the output in case of a stop error in the control CPU.	Clear/Hold	Clear	Page 141, Section 3.8
	PLC Operation Mode at H/W Error	4004_H	Set whether to stop or continue the operation of the control CPU in case of a hardware failure of the intelligent function module.	Stop/Continue	Stop	Page 142, Section 3.9
	I/O Response Time	0405_H	Set a response time for the input module, high-speed input module, I/O combined module, or interrupt module.	• Input or I/O Mix: 1ms, 5ms, 10ms, 20ms, or 70ms • Hi. Input or Interrupt: 0.1ms, 0.2ms, 0.4ms, 0.6ms, or 1ms	• Input or I/O Mix: 10ms • Hi. Input or Interrupt: 0.2ms	Page 139, Section 3.7
	Control PLC	0406_H	Set the control CPU for the input/output modules and intelligent function module.	PLC No.1, No.2, No.3, or No.4	PLC No.1	QCPU User's Manual (Multiple CPU System)
Select PLC type		-	Sets the model of the connected CPU module automatically in the "I/O Assignment" area by selecting the module from the pull-down menu.	-	Blank	Page 54, Section 2.2.2
Select module		-	Sets the model name, points, and start I/O number of the selected module automatically in the "I/O Assignment" area by selecting the module from the pull-down menu.	-	Blank	

Appendix 1.2.10 Multiple CPU setting Note Appx.2

Parameters required for configuring a multiple CPU system are set.



Note Appx.2 Universal

The Q00UJCPU cannot be used in multiple CPU systems.

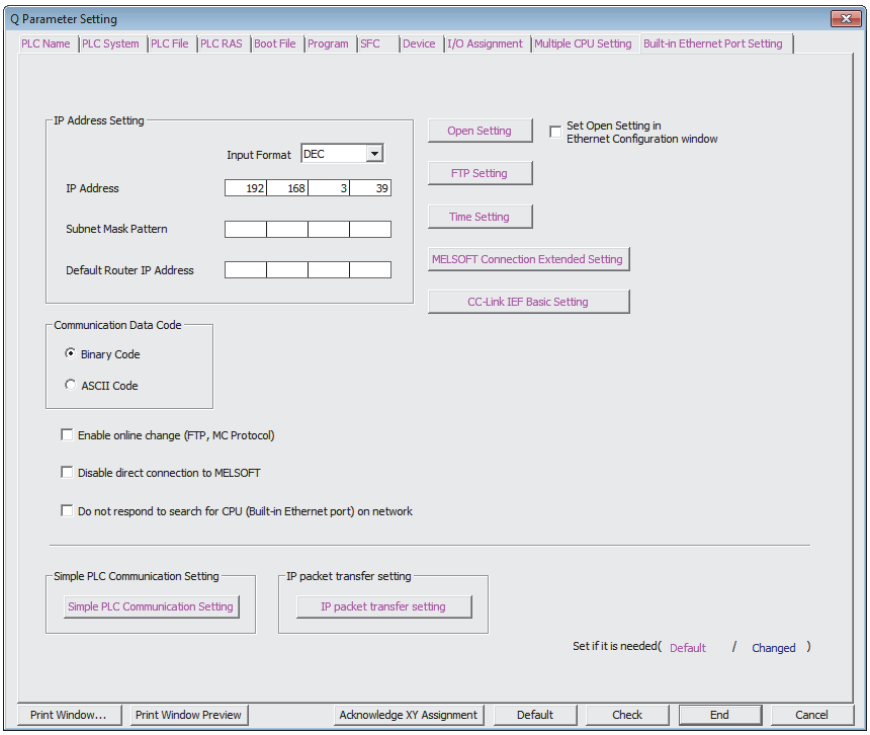
Item			Parameter No.	Description	Setting range	Default	Reference
No. of PLC			$0E00_H$	Set the number of CPU modules used in a multiple CPU system.	1 to 4	1	QCPU User's Manual (Multiple CPU System)
Host Station*1			$E00C_H$	Set a CPU number for which the multiple CPU setting parameters are set. (Set the number of the connected CPU module.)	PLC No.1 to No.4	Blank	
Operation Mode			$0E01_H$	Select the multiple CPU system operation to be performed in case a stop error occurs in any of CPU No.2 to No.4. When CPU No.1 results in a stop error, the multiple CPU system stops. (Fixed)	Selected/deselected	All items selected	
Multiple CPU Synchronous Startup Setting*1			$E00B_H$	Enable or disable synchronous startup of the CPU modules on the multiple CPU system.	No.1 to No.4	All items selected	
Online Module Change*1			$E006_H$	Enable or disable the online module change in the multiple CPU system. (When enabled, the CPU module cannot read the I/O data outside the specified group.)	Selected/deselected	Deselected	
I/O Sharing When Using Multiple CPUs		All CPUs Can Read All Inputs	$0E04_H$	Select whether to read the input data of the input modules or intelligent function modules controlled by another CPU.	Selected/deselected	Deselected	
		All CPUs Can Read All Outputs		Select whether to read the output data of the output modules controlled by another CPU.	Selected/deselected	Deselected	
Multiple CPU High Speed Transmission Area Setting*1	CPU Specific Send Range		$E008_H$	Set the size of the multiple CPU high-speed transmission area that is assigned to each CPU module of the multiple CPU system.	Simple setting:0 to 12K (in increments of 1K points) Advanced setting:0 to 16K (in increments of 0.5K points)	3K points	
	Auto Refresh	Refresh setting	$E009_H$ $E00A_H$	Set an area range for data communication performed with the auto refresh function in the user area of the multiple CPU high-speed transmission area.	Available devices*2:X, Y, M, L, B, D, W, R, ZR, SM, SD, SB, and SW	-	
Communication Area Setting (Refresh Setting)			$E002_H$ $E003_H$	In the multiple CPU system, data are transferred by auto refresh among respective CPU modules. Set the devices to be written or read and their points.	[Set Starting Devices for each PLC] Selected/deselected	Deselected	
					[CPU Specific Send Range] 0 to 2048 points (in increments of 2 points) per CPU Up to 8K points (8192 points) per system	Blank	
					[PLC Side Device] B, M, Y, D, W, R, or ZR Occupies the device of the points set for the send range and starting from the specified device number. • One point in the send range equals 16 points in B, M, or Y. • One point in the send range equals one point in D, W, R, or ZR.		

*1 Not available for the Q00UCPU, Q01UCPU, and Q02UCPU.

*2 SM, SD, SB, and SW are valid only when they are selected as send devices.

Appendix 1.2.11 Built-in Ethernet port setting Note Appx.3

Parameters required for using the built-in Ethernet port are set.



Note Appx.3 Universal

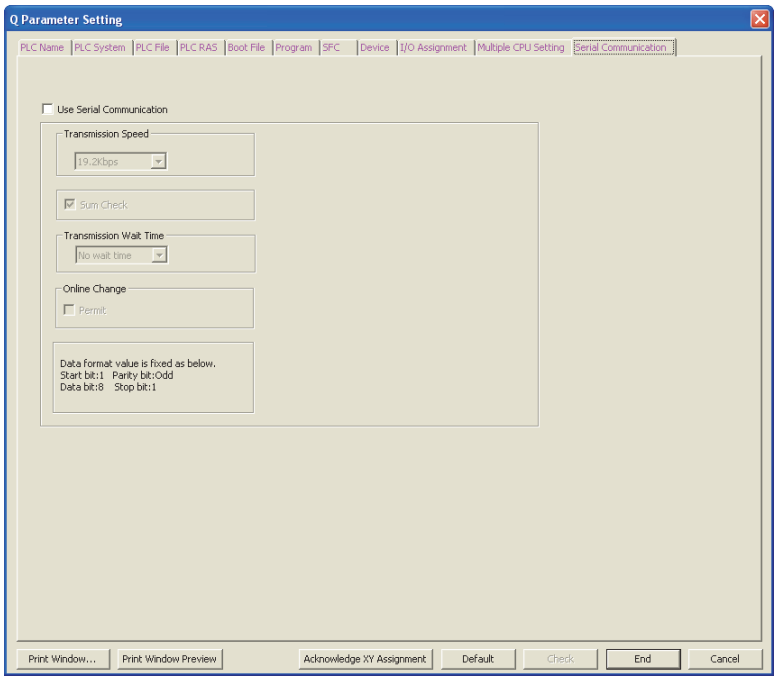
The Universal model QCPUs other than the Built-in Ethernet port QCPU do not have any Ethernet port.

Item		Parameter No.	Description	Setting range	Default	Reference
IP Address Setting		1016_H	• IP Address: Enter the IP address of the CPU module. • Subnet Mask Pattern: Enter the subnet mask pattern when using a router. • Default Router IP Address: Enter the IP address of the router.	• IP Address: 0.0.0.1 to 223.255.255.254 (00000001_H to $0DFFFFFFE_H$) • Subnet Mask Pattern: Blank or 192.0.0.0 to 255.255.255.252 ($0C0000000_H$ to $0FFFFFFFC_H$) • Default Router IP Address: Blank or 0.0.0.1 to 223.255.255.254 (00000001_H to $0DFFFFFFE_H$)	• IP Address: 192.168.3.39 • Subnet Mask Pattern: Blank • Default Router IP Address: Blank	QnUCPU User's Manual (Communication via Built-in Ethernet Port)
Communication Data Code			Select the code for MC protocol communication.	Binary Code/ASCII Code	Binary Code	
Ethernet Conf.*1/Open Setting			Set when using the following functions. • MC protocol • MELSOFT connection • Socket communication • Predefined protocol	-	Blank	
FTP Setting			Set data when using the file transfer function (FTP).	-	Blank	
Time Setting			Set data when using the time setting function.	-	Blank	
MELSOFT Connection Extended Setting*1		$101F_H$	Set data when using the MELSOFT connection extended setting.	-	Blank	
CC-Link IEF Basic Setting*1	Network Configuration Setting	1030_H	Set the network configuration when using CC-Link IE Field Network Basic.	-	Blank	CC-Link IE Field Network Basic Reference Manual
	Refresh Setting	1031_H	Set this parameter to refresh link device data to the internal device or file register automatically.	-	Blank	
Enable online change (FTP, MC Protocol)		1016_H	Enable or disable writing data in devices or files to the running CPU module when MC protocol or FTP is used.	Selected/deselected	Deselected	QnUCPU User's Manual (Communication via Built-in Ethernet Port)
Disable direct connection to MELSOFT			Enable or disable direct connection to MELSOFT. To enhance the security with the remote password setting, check it to disable it.	Selected/deselected	Deselected	
Do not respond to search for CPU (Built-in Ethernet port) on network			Checking this box disables response to the find CPU function of the MELSOFT connection. To enhance the security, check it to disable this.	Selected/deselected	Deselected	
Simple PLC communication function		1019_H	Set parameters when using the simple PLC communication function.	-	-	
IP packet transfer setting		1016_H	Set data when using IP packet transfer function.	Use/Not used	Not used	Manual for the CC-Link IE Field Network module used

*1 This setting item is available to the High-speed Universal model QCPU and Universal model Process CPU. Before setting this item, check the versions of the CPU module and GX Works2 used. (☞ Page 466, Appendix 2)

Appendix 1.2.12 Serial communication Note Appx.4

The transmission speed, sum check, transmission wait time, and RUN write setting for using the serial communication function of the CPU module are set.



Item	Parameter No.	Description	Setting range	Default	Reference
Use Serial Communication	$100E_H$	Select the item when using the serial communication function.	Selected/deselected	Deselected	Page 233, Section 3.23
Transmission Speed		Set a transmission speed for data communication with the external device.	9.6Kbps, 19.2Kbps, 38.4Kbps, 57.6Kbps, 115.2Kbps	19.2Kbps	
Sum Check		Set whether to add a sum check code to a message sent or received when using the serial communication function, according to the specifications of the external device.	Selected/deselected	Selected	
Transmission Wait Time		Set a period of waiting time on the CPU module side in case the CPU module cannot receive data immediately after the external device sends data.	No wait time/10ms to 150ms (in increments of 10ms)	No wait time	
Online Change		Enable or disable writing of data from the external device to the running CPU module.	Selected/deselected	Deselected	

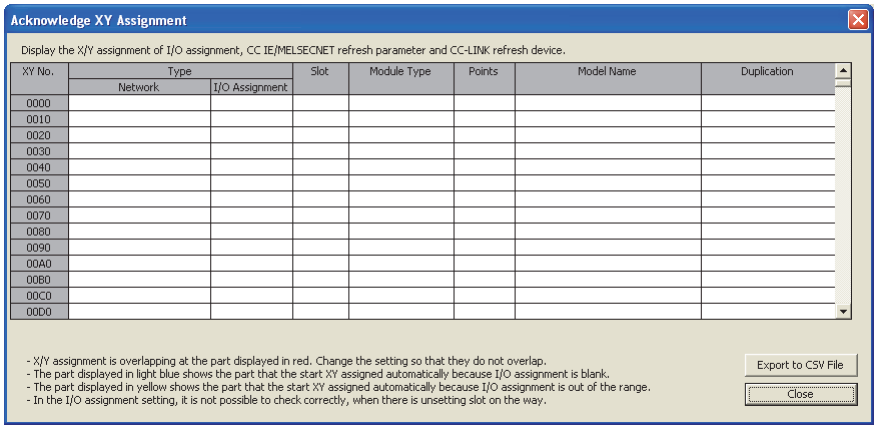
Note Appx.4 Universal

Before using the serial communication function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, or Q26UDHCPU, check the versions of the CPU module and programming tool used. (Page 466, Appendix 2)

The Built-in Ethernet port QCPU does not support the serial communication function.

Appendix 1.2.13 Acknowledge XY assignment

The parameters set in the I/O Assignment, Ethernet/CC IE/MELSECNET setting, and CC-Link setting can be checked.



Item	Parameter No.	Description	Setting range	Default	Reference
Acknowledge XY Assignment	-	The data set in the I/O Assignment, Ethernet/CC IE/MELSECNET setting, and CC-Link setting can be checked.	-	-	-
Export to CSV File	-	Writes parameters set in this screen to a CSV file.	-	-	-

Appendix 1.3 Network Parameters

This section describes network parameter details with setting windows.

■Symbols, M and N, used in the "Parameter No." column

M and N in "Parameter No." in this section denote the following:

- N: Indicates the module number.
- M: Indicates the network type.

(1) For CC-Link IE, MELSECNET/H

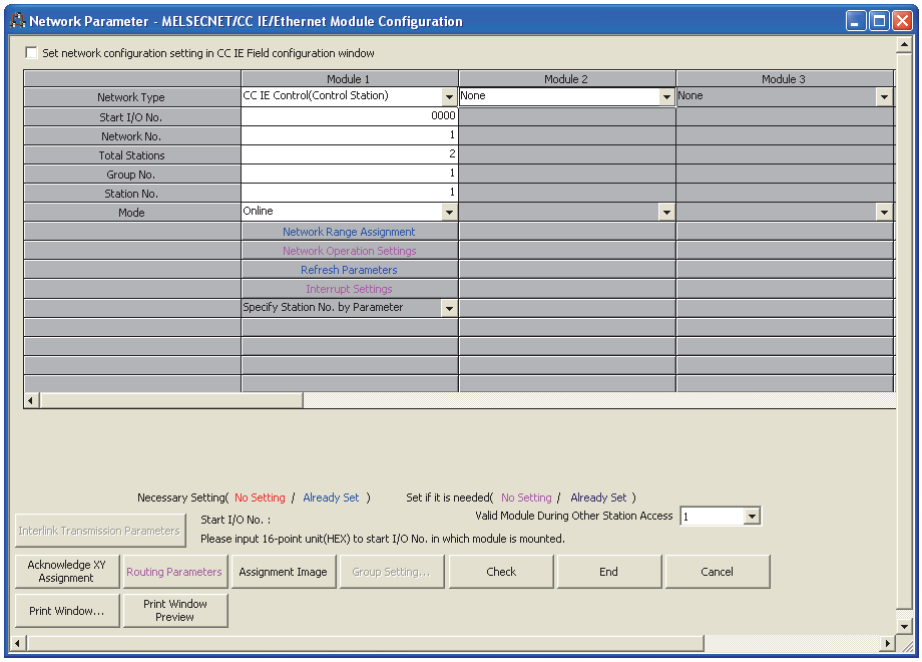
M	Network type
1_H	CC IE Control (Control station), MELSECNET/H mode (Control station), MELSECNET/H Extended mode (Control station), MELSECNET/10 mode (Control station)
2_H	CC IE Control (Normal station), MELSECNET/H mode (Normal station), MELSECNET/H Extended mode (Normal station), MELSECNET/10 mode (Normal station)
5_H	MELSECNET/H (Remote master station)
8_H	CC IE Field (Master station), CC IE Field (Submaster station) (when parameters are set)
9_H	CC IE Field (Local station), CC IE Field (Submaster station) (when no parameter is set)
A_H	MELSECNET/H (Standby station)
B_H	MELSECNET/H mode multiplexed remote I/O network master station
D_H	MELSECNET/H mode multiplexed remote I/O network sub-master station (when no parameter is set)
E_H	MELSECNET/H mode multiplexed remote I/O network sub-master station (when parameters are set)

(2) For CC-Link

M	Network type
0_H	Master station
1_H	Local station
2_H	Standby master station

Appendix 1.3.1 CC-Link IE Controller Network setting

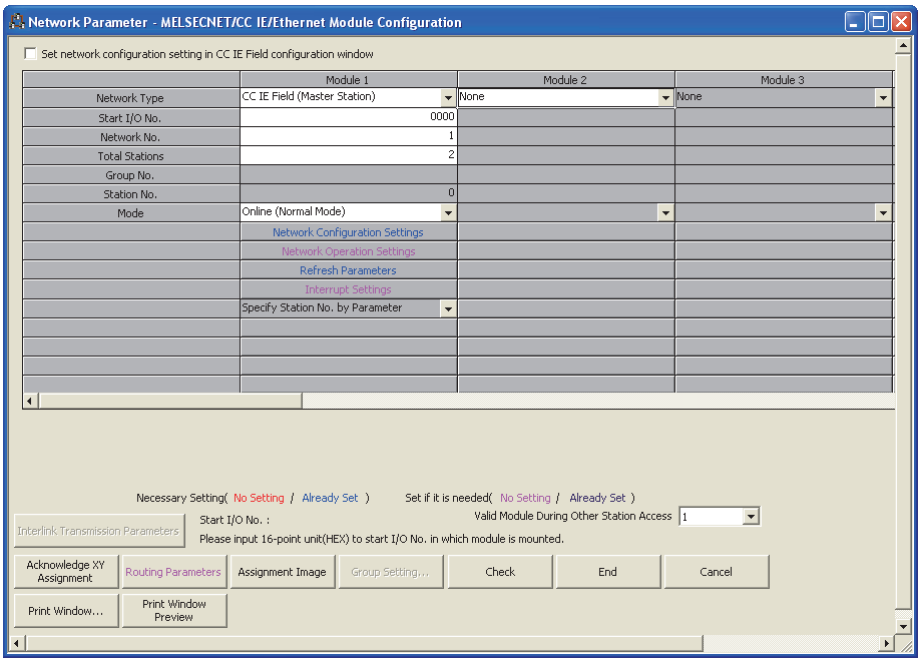
Network parameters for the CC-Link IE Controller Network are set.



Item	Parameter No.	Description	Setting range	Default	Reference
Network Type	$A000_H$	Set network parameters for the CC-Link IE Controller Network.	Refer to the manual for the CC-Link IE Controller Network.	-	-
Station number setting method					
Start I/O No.	$ANM0_H$				
Network No.					
Total Stations					
Station No.					
Group No.	$0Amn_H$				
Mode	$ANM0_H$				
Refresh Parameters	$ANM1_H$				
Common Parameters	$ANM2_H$				
Station Inherent Parameters	$ANM3_H$				
Interlink Transmission Parameters	$A002_H$				
Routing Parameters	5003_H				

Appendix 1.3.2 CC-Link IE Field Network setting

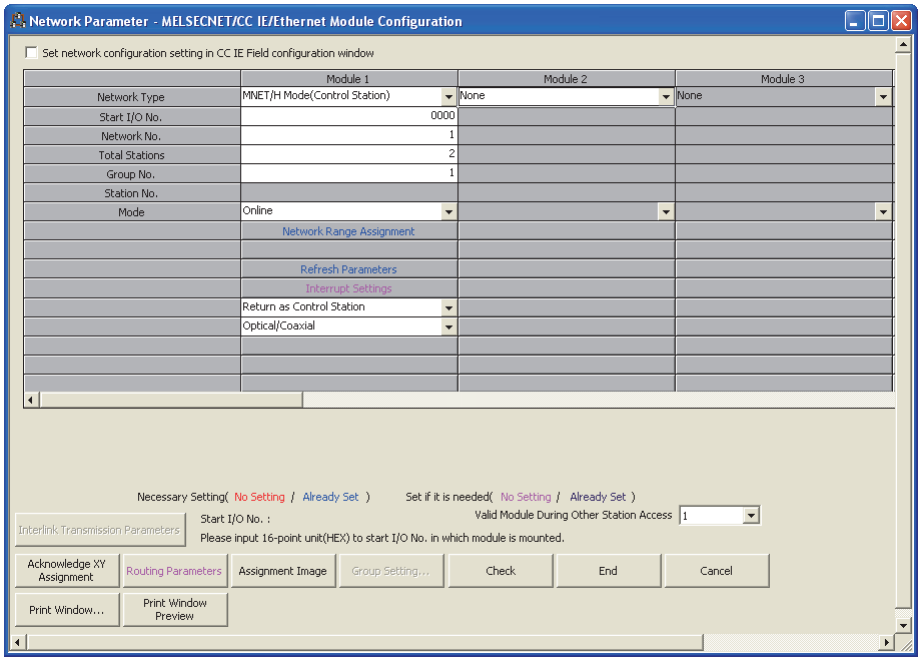
Network parameters for the CC-Link IE Field Network are set.



Item	Parameter No.	Description	Setting range	Default	Reference
Network Type	$A080_H$	Set network parameters for the CC-Link IE Field Network.	Refer to the manual for the CC-Link IE Field Network.	-	-
Station number setting method					
Start I/O No.	$ANM0_H$				
Network No.					
Total Stations					
Station No.					
Mode					
Refresh Parameters	$ANM1_H$				
Network Configuration Settings	$ANM2_H$				
Network Operation Settings	$ANM3_H$				
Interrupt Settings					
Interlink Transmission Parameters	$A082_H$				
Routing Parameters	5003_H				

Appendix 1.3.3 MELSECNET/H setting

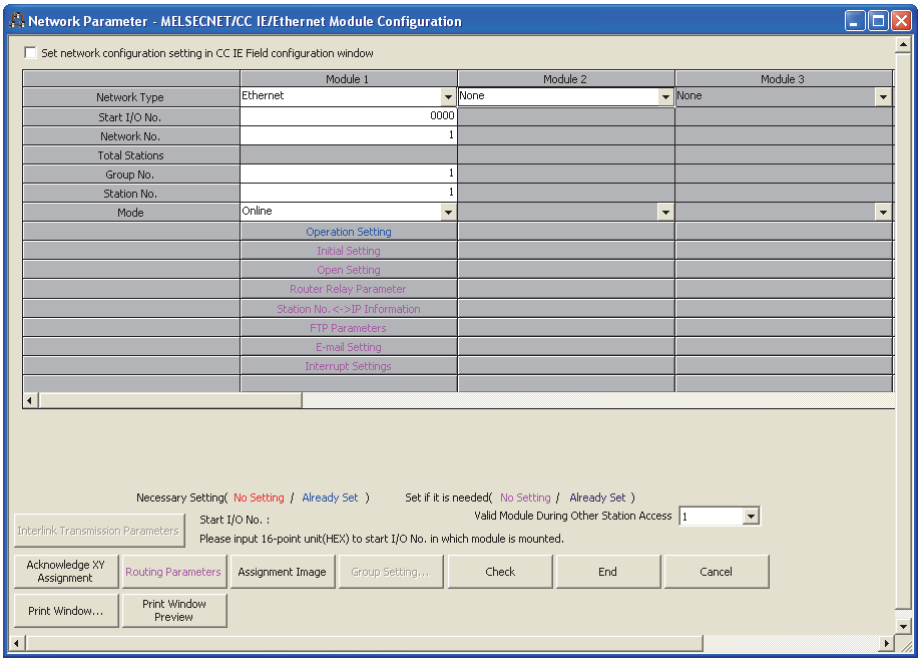
Network parameters for MELSECNET/H are set.



Item	Parameter No.	Description	Setting range	Default	Reference
Number of modules on MELSECNET/H	5000_H	Set MELSECNET/H network parameters.	Refer to the manual for the Q series-compatible MELSECNET/H.	-	-
Start I/O No.	$5NM0_H$				
Network No.					
Total Stations					
Group No.	$05mn_H$				
Mode	$5NM0_H$				
Refresh Parameters	$5NM1_H$				
Common Parameters	$5NM2_H$				
Station Inherent Parameters	$5NM3_H$				
Common Parameters 2	$5NMA_H$				
Station Inherent Parameters 2	$5NMB_H$				
Interrupt Settings					
Valid Module During Other Station Access	5001_H				
Interlink Transmission Parameters	5002_H				
Routing Parameters	5003_H				

Appendix 1.3.4 Ethernet setting

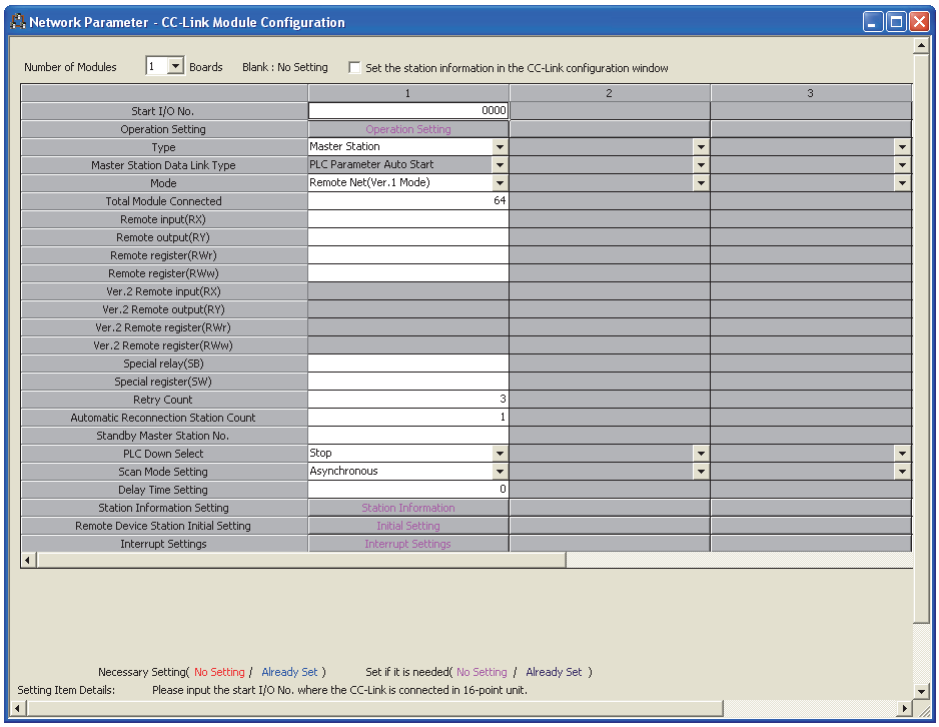
Network parameters for Ethernet are set.



Item	Parameter No.	Description	Setting range	Default	Reference
Number of modules on Ethernet	9000_H	Set Ethernet network parameters.	Refer to the manual for the Q series-compatible Ethernet.	-	-
Start I/O No.	$9N00_H$				
Network No.					
Group No.					
Station No.					
Operation Setting					
Initial Setting	$9N01_H$				
Open Setting	$9N02_H$				
Router Relay Parameter	$9N03_H$				
Station No.<->IP Information	$9N05_H$				
FTP Parameters	$9N06_H$				
E-mail Setting	$9N07_H$				
News Setting	$9N08_H$				
Interrupt Settings	$9N09_H$				
Valid Module During Other Station Access	5001_H				
Routing Parameters	$9N04_H$				

Appendix 1.3.5 CC-Link setting

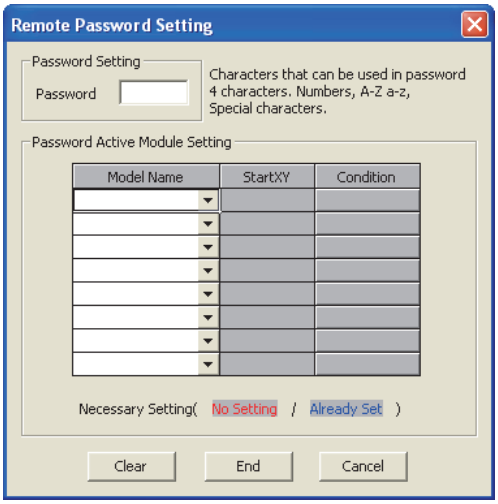
Parameters for CC-Link are set.

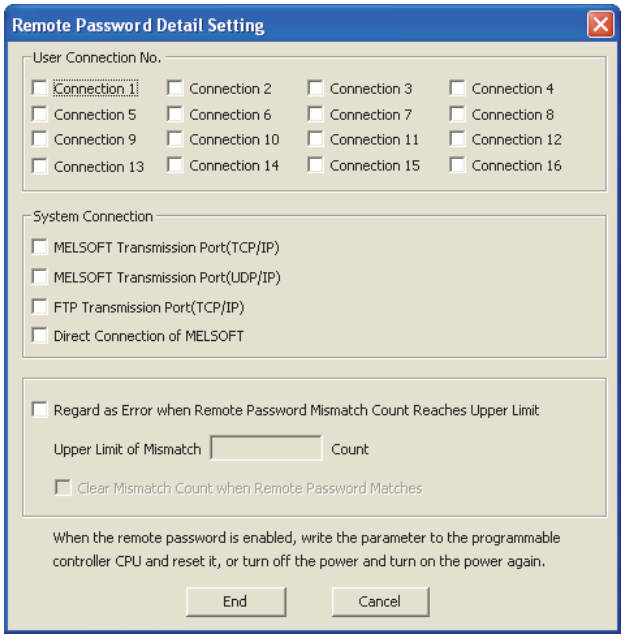


Item	Parameter No.	Description	Setting range	Default	Reference
Number of Modules	$C000_H$	Set CC-Link parameters.	Refer to the manual for CC-Link.	-	-
Type					
Start I/O No.	$CNM2_H$				
Operation Setting					
Total Module Connected					
Remote Input(RX)	$CNM1_H$				
Remote Output(RY)					
Remote Register(RWr)					
Remote Register(RWw)					
Ver.2 Remote Input(RX)					
Ver.2 Remote Output(RY)					
Ver.2 Remote Register(RWr)					
Ver.2 Remote Register(RWw)					
Special Relay(SB)	$CNM2_H$				
Special Register(SW)					
Retry Count					
Automatic Reconnection Station Count					
Standby Master Station No.					
PLC Down Select					
Scan Mode Setting					
Delay Time Setting					
Station Information Setting					
Remote Device Station Initial Setting					
Interrupt Settings					

Appendix 1.4 Remote Password

This section provides the list of parameters for remote password and describes parameter details.





Item		Parameter No.	Description	Setting range	Default	Reference
Password Setting		-	Enter a remote password.	Four characters or less (alphanumeric characters, special symbols)	-	• Built-in Ethernet port QCPU: QnUCPU User's Manual (Communication via Built-in Ethernet Port) • QJ71E71: Ethernet module manual • QJ71C24: Serial communication module manual
Password Active Module Setting	Model Name	-	Select a model name of the module for which the remote password set to the CPU module is checked.	• Built-in Ethernet port QCPU • QJ71E71 • QJ71C24/CMO	-	
	Start XY	-	Set the start address of the module for which the remote password is checked.	0000_H to $0FE0_H$	-	
Detail		-	Set details of the remote password for the QJ71E71.	-	-	
User Connection No.*1		-	Select user connection No.	Connection 1 to Connection 16	-	
System Connection*2		-	Select a valid port of the remote password for system connection.	• Built-in Ethernet port QCPU • MELSOFT Transmission Port (TCP/IP) • MELSOFT Transmission Port (UDP/IP) • FTP Transmission Port (TCP/IP) • Direct Connection of MELSOFT • QJ71E71 • Auto Open UDP Port • FTP Transmission Port (TCP/IP) • MELSOFT Application Transmission Port (TCP/IP) • MELSOFT Application Transmission Port (UDP/IP), Dedicated Instruction, CC-Link IE, NET/10(H) Relay Transmission Port • HTTP Port, HTTP Protocol	-	
Regard as Error when Remote Password Mismatch Count Reaches Upper Limit		-	Set upper limit of mismatch count.	Set whether to regard it as an error when the number of mismatches on remote password reaches the upper limit.	Unchecked	
Upper Limit of Mismatch		-	Set upper limit of remote password mismatch count.	1 to 65535	10	

*1 This is a connection used by users for communications using the MC protocol or fixed buffer.

*2 This is a connection used by a system for FTP or MELSOFT (TCP/IP, UDP/IP) communications.

Appendix 2 Functions Added or Changed by Version Upgrade

The Universal model QCPU is upgraded when some functions are added or specifications are changed. Therefore, the functions and specifications differ depending on the function version and serial number.

× : Not supported, – : Not related to the programming tool

Function	Function version	Serial number (first 5 digits)	Programming tool version	
			GX Works2	GX Developer
Use of the PC CPU module*1 (QCPU User's Manual (Multiple CPU System))	B	"09072" or later	Version 1.15R or later	–
Setting of whether to use local device for each program (Page 422, Section 6.2)		*6		Version 8.62Q or later
Program memory batch transfer execution status check (SM165) (Page 36, Section 2.1.1 (1) (d))				–
Multiple CPU high-speed transmission dedicated instruction*1 (MELSEC-Q/L Programming Manual (Common Instruction))				
Display of the amount of battery consumption (QCPU User's Manual (Hardware Design, Maintenance and Inspection))				
Bit device extension (Page 345, Section 4.2)		"10042" or later		Version 8.68W or later
Executional conditioned device test (Page 159, Section 3.11.4)				
Sampling trace auto start function*1 (Page 184, Section 3.14)				
CC-Link IE group cyclic transmission function (CC-Link IE Controller Network Reference Manual)				
Scan time measurement (Page 181, Section 3.13.3)				
External input/output forced on/off (Page 154, Section 3.11.3)				
Monitor condition setting*1 (Page 146, Section 3.11.1)			Version 1.73B or later	
Redundant power supply system*1 (Page 442, Appendix 1.2.4)			Version 1.15R or later	
32-bit indexing with "ZZ" specification (MELSEC-Q/L Programming Manual (Common Instruction))				
Extended data register (D) and extended link register (W)*1 *2 (Page 402, Section 4.8)		"09042" or later*1		Version 8.70Y or later
Serial communication function (Q02UCPU) (Page 233, Section 3.23)		"10102" or later		Version 8.76E or later
CPU module change function with memory card*1 (Page 260, Section 3.31)				
Local device setting of the index register*1 (Page 447, Appendix 1.2.8)				
Communication using the A-compatible 1C/1E frame (MC protocol)*3 *4 (MELSEC Communication Protocol Reference Manual)				
A → QnA converted special relay/special register (SM1000 to SM1255, SD1000 to SD1255) (QCPU User's Manual (Hardware Design, Maintenance and Inspection))		"10102" or later		Version 8.78G or later
Socket communication function*1 (QnUCPU User's Manual (Communication via Built-in Ethernet Port))		"11012" or later		

Function	Function version	Serial number (first 5 digits)	Programming tool version	
			GX Works2	GX Developer
Module model name read (Page 297, Section 3.33)	B	"11043" or later	Version 1.15R or later	Version 8.82L or later
Module error collection function*1 *5 (Page 298, Section 3.34)			Version 1.12N or later	×
IP address change function*1 (QnUCPU User's Manual (Communication via Built-in Ethernet Port))		"11082" or later	–	–
Local device batch read function*1 (Page 302, Section 3.35)		"12012" or later	Version 1.31H or later	×
CC-Link IE Field Network (MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual)				
Send points extension function*5 (CC-Link IE Controller Network module) (Page 304, Section 3.36)		"12052" or later	Version 1.40S or later	
Online change of inactive blocks (SFC)*1 (MELSEC-Q/L/QnA Programming Manual (SFC))				
Expansion of SFC step relay points (Page 345, Section 4.2)				
Operation mode setting at SFC double block START*1 (MELSEC-Q/L/QnA Programming Manual (SFC))				
SFC comment readout instruction*1 (MELSEC-Q/L/QnA Programming Manual (SFC))			–	
Data up to 10238 bytes can be exchanged with the SP.SOCSND/S(P).SOCRCV(S)/S(P).SOCRDATA instructions*1 (QnUCPU User's Manual (Communication via Built-in Ethernet Port))				
Parameter-valid drive information (Page 42, Section 2.1.2)		–	Version 1.40S or later	×
Program cache memory auto recovery function (Page 252, Section 3.28)		"12122" or later	–	–
Extension of available index register range (Z0 to Z19) when Jn and Un are used in the dedicated instruction (Manuals for the network module and the intelligent function module used)		"13022" or later		
Storage of device memory error information (Memory check function) (QCPU User's Manual (Hardware Design, Maintenance and Inspection))				
Storage of program error location (Memory check function) (QCPU User's Manual (Hardware Design, Maintenance and Inspection))		"13042" or later		
Serial communication function (Q03UD/Q04UDH/Q06UDH/Q10UDH/Q13UDH/Q20UDH/Q26UDHCPU) (Page 233, Section 3.23)		"13062" or later	Version 1.62Q or later	×
SFC control target block switching (MELSEC-Q/L/QnA Programming Manual (SFC))		"13102" or later	Version 1.73B or later	×
AnS/A series compatible extension base unit			–	–
Communication using the A-compatible 1E frame (MC protocol) through built-in Ethernet port (MELSEC Communication Protocol Reference Manual)				
IP packet transfer function (CC-Link IE Field Network) (QnUCPU User's Manual (Communication via Built-in Ethernet Port))*1 *5		"14022" or later	Version 1.77F or later	×
Own station number setting function for CC-Link IE Field Network (MELSEC-Q CC-Link IE Field Network Master/Local Module User's Manual)		"14042" or later	Version 1.87R or later	×
Writing/reading data to/from refresh devices with the specified station number*1 (MELSEC-Q/L Programming Manual (Common Instruction))		*10	–	–

Function	Function version	Serial number (first 5 digits)	Programming tool version	
			GX Works2	GX Developer
High-speed interrupt function (☞ Page 225, Section 3.21)	B	–	Version 1.98C or later	×
Data logging function[*7] (QnUDVCPU/LCPU User's Manual (Data Logging Function))		–	Version 1.98C or later	×
IP packet transfer function (CC-Link IE Controller Network) (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1] [*5]		"14022" or later	Version 1.98C or later	×
Use of the file register in communication using the A-compatible 1E frame (MC protocol) through built-in Ethernet port (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]		*8	–	–
Expansion of routing parameter settings (MELSEC-Q/L Programming Manual (Common Instruction))[*1]		*9		
Latch clear by using the special relay and special register areas (☞ Page 75, Section 2.7 (4) (a))[*1]		*10		
Predefined protocol function (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]		"15103" or later	Version 1.501X or later	×
Reading/writing device data from/to the CPU module on another station by specifying an IP address (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]		"16043" or later	Version 1.513K or later	×
Support of the iQ Sensor Solution function (data backup/restoration only) for AnyWireASLINK and CC-Link (iQ Sensor Solution Reference Manual)[*1]		"17012" or later	1.530C or later	–
Support of the iQ Sensor Solution function (data backup/restoration only) for CC-Link IE Field Network (iQ Sensor Solution Reference Manual)[*1]		"17052" or later	×	–
MELSOFT connection extended setting (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]			1.535H or later	×
CPU module data backup/restoration function (☞ Page 276, Section 3.32)[*1] [*7]		"17103" or later	–	–
Upper limit value setting for the number of backup data (CPU module data backup/restoration function) (☞ Page 282, Section 3.32.1 (1))[*1]		"18052" or later	–	–
Retrying the automatic backup (CPU module data backup/restoration function) (☞ Page 285, Section 3.32.1 (3) (a))[*1]				
Support of the iQ Sensor Solution function (automatic detection of connected device, system configuration check, communication setting reflection, sensor parameter read/write, monitoring, and data backup/restoration) for built-in Ethernet (iQ Sensor Solution Reference Manual)[*1]		"18072" or later	1.550Y or later	–
CC-Link IE Field Network Basic[*7]		"18112" or later	1.555D or later	×
SLMP frame send instruction (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]			–	–
Write-protect function for device data (from outside the CPU module) (☞ Page 306, Section 3.37)[*1]		"19062" or later	1.566Q or later	×
Operation history function (☞ Page 314, Section 3.38)[*1]				
Simple PLC communication function (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]		"20042" or later	1.575Z or later	×
Simple PLC communication function (for MELSEC iQ-F series) (QnUCPU User's Manual (Communication via Built-in Ethernet Port))[*1]		"20102" or later	1.580E or later	×

*1 Some models do not support the function. For details, refer to the corresponding reference.

*2 Use the Universal model QCPU whose serial number (first five digits) is "10042" or later to store data of the extended data register (D) and extended link register (W) in the standard ROM using the latch data backup function.
(☞Page 254, Section 3.29).

*3 Communication using the A-compatible 1E frame is available only via any Ethernet module.
If the module is connected to the built-in Ethernet port of the CPU module, this function is not available.

*4 Communication using the A-compatible 1C frame is available only via any serial communication module.
If the module is connected to the built-in RS-232 interface of the CPU module, this function is not available.

*5 For the versions of the intelligent function modules that support the function, refer to the manual for the intelligent function module used.

*6 The serial number (first five digits) differs depending on the CPU module.
- Q13UDHCPU, Q26UDHCPU: "10011" or later
- CPU modules other than above: "10012" or later

*7 Only the QnUDVCPU and QnUDPVCPU support this function.

*8 The serial number (first five digits) differs depending on the CPU module.
- QnUDE(H)CPU: "14112" or later
- QnUDVCPU: "15043" or later
- QnUDPVCPU: "15072" or later

*9 The serial number (first five digits) differs depending on the CPU module.
- QnU(D)(H)CPU, QnUDE(H)CPU: "14112" or later
- QnUDVCPU: "15043" or later
- QnUDPVCPU: "15072" or later

*10 The serial number (first five digits) differs depending on the CPU module.
- QnUDVCPU: "15043" or later
- QnUDPVCPU: "15072" or later

*11 The serial number (first five digits) differs depending on the CPU module.
- QnUD(E)(H)CPU: "14072" or later
- QnUD(P)VCPU: "16043" or later

Appendix 3 CPU Module Processing Time

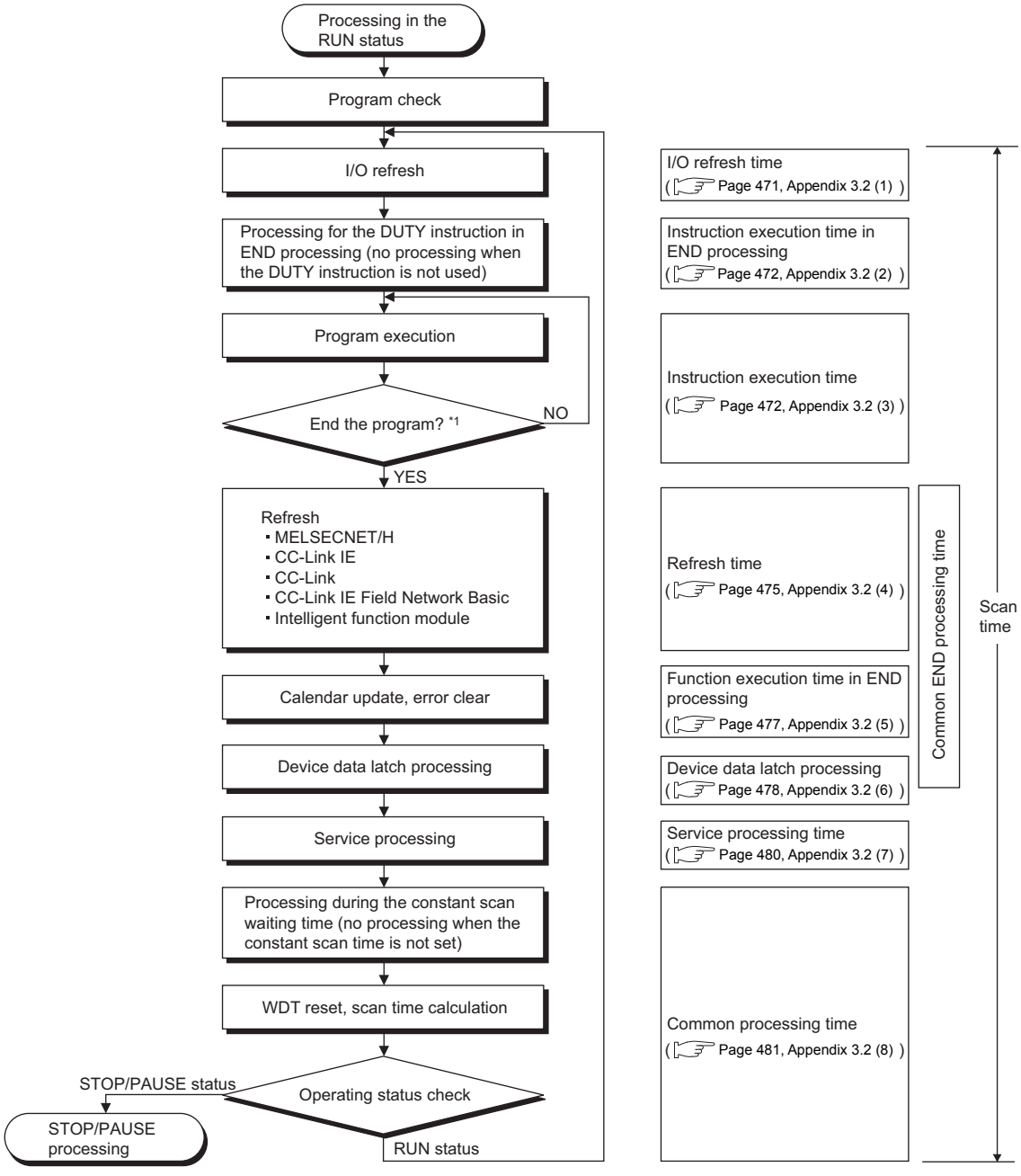
This chapter describes the CPU module processing time.

This section describes the scan time structures and CPU module processing time.

Appendix 3.1 Scan time structure

A CPU module sequentially performs the following processing in the RUN status.

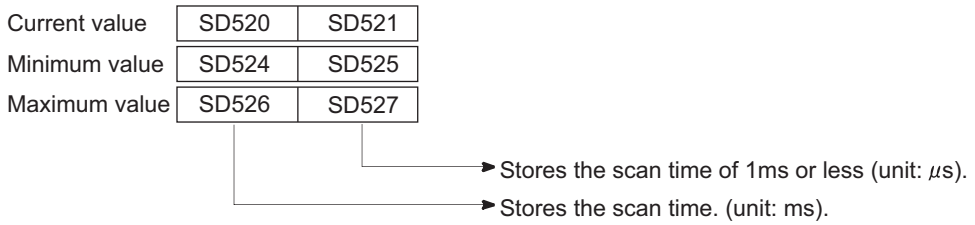
Scan time is the time required for all processing and executions to be performed.



*1 End of a program indicates the timing when the END, GOEND, FEND, or STOP instruction is executed.

(1) How to check scan time

The CPU module measures current, minimum, and maximum values of the scan time.
The scan time can be checked by monitoring the special register (SD520, SD521, and SD524 to SD527).
Accuracy of each stored scan time is ±0.1ms.



Ex. If the stored values in SD520 and SD521 are 3 and 400 respectively, the scan time is 3.4ms.

Appendix 3.2 Time required for each processing included in scan time

This section describes how to calculate the processing time and execution time described in Page 470, Appendix 3.1.

(1) I/O refresh time

I/O refresh time is the time required for refreshing I/O data to/from the following modules mounted on the main base unit and extension base units.

- Input module
- Output module
- Intelligent function module

■ Calculation method

Use the following expression to calculate the I/O refresh time. For N1 and N2, refer to the following table.

(I/O refresh time) = (number of input points/16) × N1 + (number of output points/16) × N2

CPU module	Q3□B, Q3□SB, Q3□RB, Q3□DB		Q5□B, Q6□B, Q6□RB		QA1S5□B, QA1S6□B, QA1S6ADP+A1S5□B, QA1S6ADP+A1S6□B*1		QA6□B, QA6ADP+A5□B, QA6ADP+A6□B*1	
	N1	N2	N1	N2	N1	N2	N1	N2
Q00UJCPU, Q00UCPU, Q01UCPU	1.8μs	1.1μs	2.6μs	1.9μs	4.9μs	4.0μs	5.7μs	4.9μs
Q02UCPU	1.5μs	1.1μs	2.4μs	1.9μs	4.5μs	4.0μs	5.3μs	4.9μs
Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q50UDEHCPU, Q100UDEHCPU	1.5μs	1.0μs	2.3μs	1.8μs	4.3μs	3.9μs	5.0μs	4.8μs
Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, Q26UDPVCPU	1.5μs	1.0μs	2.3μs	1.8μs	Not applicable	Not applicable	Not applicable	Not applicable

*1 Applicable only when the CPU module whose serial number (first five digits) is "13102" or later is used.

(2) Instruction execution time in END processing

This is the processing time of the DUTY instruction in END processing.
The user timing clock (SM420 to SM424 and SM430 to SM434) specified with the DUTY instruction is turned on/off during the END processing.

CPU module	Processing time in END processing	
	When set to 1	When set to 5
Q00UJCPU, Q00UCPU, Q01UCPU	0.0120ms	0.0140ms
Q02UCPU	0.0050ms	0.0055ms
Q03UD(E)CPU	0.0043ms	0.0046ms
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.0041ms	0.0045ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.0040ms	0.0042ms

(3) Instruction execution time

Instruction execution time is the time required for all instructions used in the program to be executed.

For the processing time required for each instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

When calculating instruction execution time, add the overhead time given in the following tables. Two kinds of overhead time (pre-start and program-end) need to be added to interrupt programs.

(a) Pre-start overhead time for interrupt programs

CPU module	Fixed scan interrupt (I28 to I31)		Multiple CPU synchronous interrupt (I45)		High-speed interrupt (I49)	Interrupt[*1] (I0 to I15) from QI60 or interrupt (I50 to I127) from the intelligent function module	
	Without high-speed start	With high-speed start	Without high-speed start	With high-speed start	With high-speed start	Without high-speed start	With high-speed start
Q00UJCPU, Q00UCPU, Q01UCPU	55μs	35μs	---	---	---	76μs	55μs
Q02UCPU	48μs	17μs	---	---	---	60μs	31μs
Q03UD(E)CPU	47μs	17μs	46μs	16μs	---	54μs	22μs
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	46μs	16μs	44μs	14μs	---	52μs	22μs
Q50UDEHCPU, Q100UDEHCPU	73μs	16μs	71μs	14μs	---	79μs	22μs

CPU module	Fixed scan interrupt (I28 to I31)		Multiple CPU synchronous interrupt (I45)		High-speed interrupt (I49)	Interrupt*1 (I0 to I15) from QI60 or interrupt (I50 to I127) from the intelligent function module	
	Without high-speed start	With high-speed start	Without high-speed start	With high-speed start	With high-speed start	Without high-speed start	With high-speed start
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	11μs	8μs	11μs	8μs	7.0μs	20.5μs	17.5μs

*1 Indicates the value when the QI60 is mounted on the slot 0 of the main base unit.

(b) Program-end overhead time for interrupt programs

CPU module	Without high-speed start	With high-speed start
Q00UJCPU, Q00UCPU, Q01UCPU	28μs	15μs
Q02UCPU	26μs	7μs
Q03UD(E)CPU	26μs	7μs
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	26μs	7μs
Q50UDEHCPU, Q100UDEHCPU	44μs	7μs
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	5.4μs	5.2μs

(c) Overhead time for fixed scan execution type programs

CPU module	Without high-speed start	With high-speed start
Q00UJCPU, Q00UCPU, Q01UCPU	92μs	60μs
Q02UCPU	73μs	25μs
Q03UD(E)CPU	73μs	24μs
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	72μs	23μs
Q50UDEHCPU, Q100UDEHCPU	117μs	23μs
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	18.5μs	15.5μs

(d) Overhead time when local devices in the interrupt program are enabled

When SM777 (Enable/disable local device in interrupt program) turns on, the time given in the following tables will be added to the overhead time given in Page 472, Appendix 3.2 (3) (a). Each n, N1, N2, and N3 in the table indicates the following.

- n: Number of local device points (unit: K words)
- N1: Number of devices that specified a local device
- N2: Number of word device points that specified a local device
- N3: Number of bit device points that specified a local device

CPU module	When a local device file in the standard RAM is used	
	Additional time to the pre-start overhead time for interrupt programs (Page 472, Appendix 3.2 (3) (a))	Additional time to the program-end overhead time for interrupt programs (Page 473, Appendix 3.2 (3) (b))
Q00UCPU, Q01UCPU	(13.2 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 210μs	(8 × N1) + (0.23 × (N2 + (N3 ÷ 16))) + 30μs
Q02UCPU	(13.2 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 210μs	(8 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 30μs
Q03UD(E)CPU	(8 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 80μs	(8 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 20μs
Q03UDVCPU	(4.1 × N1) + (0.165 × (N2 + (N3 ÷ 16))) + 22.0μs	(4.1 × N1) + (0.165 × (N2 + (N3 ÷ 16))) + 5.7μs
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	(8 × N1) + (0.10 × (N2 + (N3 ÷ 16))) + 80μs	(8 × N1) + (0.10 × (N2 + (N3 ÷ 16))) + 20μs
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	(4.0 × N1) + (0.085 × (N2 + (N3 ÷ 16))) + 22.0μs	(4.0 × N1) + (0.085 × (N2 + (N3 ÷ 16))) + 5.7μs

CPU module	When a local device file in the standard RAM is used (with an extended SRAM cassette)	
	Additional time to the pre-start overhead time for interrupt programs (Page 472, Appendix 3.2 (3) (a))	Additional time to the program-end overhead time for interrupt programs (Page 473, Appendix 3.2 (3) (b))
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	(4.2 × N1) + (0.220 × (N2 + (N3 ÷ 16))) + 22.0μs	(4.2 × N1) + (0.220 × (N2 + (N3 ÷ 16))) + 5.7μs

CPU module	When a local device file in the SRAM card is used	
	Additional time to the pre-start overhead time for interrupt programs (Page 472, Appendix 3.2 (3) (a))	Additional time to the program-end overhead time for interrupt programs (Page 473, Appendix 3.2 (3) (b))
Q02UCPU	(16 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + 260μs	(16 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + 60μs
Q03UD(E)CPU	(12 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + 100μs	(12 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + 20μs
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	(12 × N1) + (0.40 × (N2 + (N3 ÷ 16))) + 100μs	(12 × N1) + (0.40 × (N2 + (N3 ÷ 16))) + 20μs

(4) Refresh time

Refresh time is the total time required for the CPU module to refresh data with the network such as CC-Link IE, MELSECNET/H, and CC-Link.

(a) Refresh with CC-Link IE

This is the time required for refreshing data between link devices in a CC-Link IE module and devices in the CPU module.

(b) Refresh with MELSECNET/H

This is the time required for refreshing data between link devices in a MELSECNET/H module and devices in the CPU module.

(c) Auto refresh with CC-Link

This is the time required for refreshing data between a CC-Link system master/local module and the CPU module.

Remark

For each refresh time, refer to the following.

Manual for each network module

(d) Refresh with CC-Link IE Field Network Basic

This is the time required for refreshing data between link devices of CC-Link IE Field Network Basic and user devices in the CPU module.

■ Calculation method

Calculate using the following formulas. Use the values in the following table for KN1 to KN4.

$\alpha T = KM1 + KM2 \times (((RX + RY) \div 16) + RWw + RWr) + \alpha E$ [μs]

$\alpha E = KM3 + KM4 \times (((RX + RY) \div 16) + RWw + RWr)$ [μs]

- αT: Link refresh time
- αE: Link refresh time when the file register (R, ZR) is used*1
- RX: Number of points of remote input (RX) refreshed by the master station*2
- RY: Number of points of remote output (RY) refreshed by the master station*2
- RWw: Number of points of remote register (RWw) refreshed by the master station*2
- RWr: Number of points of remote register (RWr) refreshed by the master station*2
- KM1 to KM4: Constant

CPU module	KM1	KM2	KM3	KM4	
				Without extended SRAM cassette	With extended SRAM cassette
Q03UDVCPU	20.0	0.02	3.0	0.05	0.11
Q04UDVCPU, Q04UDPVCPU	20.0	0.02	3.0	0.05	0.11
Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	20.0	0.02	3.0	0.05	0.11

*1 This time is added when the file register (R, ZR) is used.

*2 This value is determined according to the number of slave stations to be connected and the number of occupied stations.

(e) Auto refresh with an intelligent function module

This is the time required for refreshing data between the buffer memory of an intelligent function module and devices in the CPU module.

■ Calculation method

Calculate using the following formulas. Use the values in the following tables for KN1 and KN2.

(Refresh time) = KN1 + KN2 × (number of refresh points)

CPU module	When an intelligent function module is mounted on the main base unit		When an intelligent function module is mounted on the extension base unit	
	KN1	KN2	KN1	KN2
Q00UJCPU	96.3μs	6.7μs	79.7μs	8.9μs
Q00UCPU, Q01UCPU	96.3μs	6.7μs	79.7μs	8.1μs
Q02UCPU	23μs	6.0μs	45μs	7.0μs
Q03UD(E)CPU	6.0μs	5.0μs	7.0μs	6.0μs
Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU, Q50UDEHCPU, Q100UDEHCPU	4.0μs	5.0μs	5.0μs	6.0μs

Ex. When the number of auto refresh points for the analog-digital converter module (Q64AD) is 4 points (when the module is mounted on the Q26UDHCPU main base unit)

0.024 (ms) = 0.004 + 0.005 × 4

(5) Function execution time in END processing

This is the time required for updating calendar or clearing error in END processing.

(a) Calendar update

The following processing time is required to change or read the clock data when the clock data set request (SM210 changes from off to on) or the clock data read request (SM213 turns on) is issued.

CPU module	Processing time in END processing	
	When the clock data set request is issued	When the clock data read request is issued
Q00UJCPU, Q00UCPU, Q01UCPU	0.028ms	0.017ms
Q02UCPU	0.027ms	0.013ms
Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.011ms	0.004ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU,Q26UDVCPU, Q26UDPVCPU	0.011ms	0.007ms

(b) Error clear

The following processing time is required to clear continuation errors stored in SD50 on the rising edge of SM50 (Error reset).

CPU module	Processing time in END processing	
	When the error is cleared (the one detected by the annunciator)	When the error is cleared
Q00UJCPU, Q00UCPU, Q01UCPU	0.185ms	0.180ms
Q02UCPU	0.180ms	0.175ms
Q03UD(E)CPU	0.068ms	0.062ms
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.065ms	0.062ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU,Q26UDVCPU, Q26UDPVCPU	0.03ms	0.026ms

(c) Error clear by types

The following processing time is required to clear continuation errors by types.

CPU module	Processing time in END processing	
	When the error is cleared (the one detected by the annunciator)	When the error is cleared
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.04ms	0.036ms

(6) Device data latch processing time

When the latch range is set in the Device tab of the PLC parameter dialog box[*1] [*2] [*3], the processing time listed in the following tables is required. Each N1, N2, and N3 in the table indicates the following.

- N1: Number of devices specified to be latched (Count the latch range (1) and the latch range (2) as different devices.)
- N2: Number of bit device points specified to be latched
- N3: Number of word device points specified to be latched

(a) When the latch interval is set to "Each Scan"

The processing time listed in the following table is required.

CPU module	Processing time
Q00UJCPU, Q00UCPU, Q01UCPU	(4.4 × N1) + (0.12 × (N2 ÷ 16 + N3))μs
Q02UCPU	(4.0 × N1) + (0.12 × (N2 ÷ 16 + N3))μs
Q03UD(E)CPU	(3.0 × N1) + (0.12 × (N2 ÷ 16 + N3))μs
Q03UDVCPU	(1.0 × N1) + (0.085 × (N2 ÷ 16 + N3)) + 1.2μs
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	(3.0 × N1) + (0.05 × (N2 ÷ 16 + N3))μs
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	(1.0 × N1) + (0.045 × (N2 ÷ 16 + N3)) + 1.2μs

*1 When setting the latch range of the timer (T), retentive timer (ST), and counter (C), one point for word device and two points for bit device are occupied per point.

*2 The case where the points are set for the latch relay (L) is included.

*3 The scan time will not increase if the latch range is set for the file register (ZR), extended data register (D), or extended link register (W).

(b) When the latch interval is set to "Time Setting"

The processing time listed in the following table is required. The scan time including the first END processing after a preset time has elapsed increases.

CPU module	Processing time
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	(1.0 × N1) + (0.004 × (N2 ÷ 16 + N3)) + 17.5μs

Point

To reduce the scan time increase due to latch*1, minimize the number of latch points (latch (1) setting, latch (2) setting, and latch relay) as much as possible by performing the following.

- Move data to be latched to the file register.
- Store device data that is less frequently updated in the standard ROM with the SP.DEVST instruction. (The device data stored in the standard ROM can be read with the S(P).DEVLD instruction. (Page 259, Section 3.30)
- Set the latch interval to "Time Setting". (Page 124, Section 3.3 (5) (b))

*1 For file registers (including an extended data register (D) and an extended link register (W)), the scan time is not increased due to latch.

(7) Service processing time

Service processing is the communication processing with a programming tool and external devices. When monitoring device data, reading programs, and setting monitor conditions in a programming tool, the processing time listed in the following table is required.

(a) Processing time to monitor device data and read programs

CPU module	Processing time*1	
	Monitoring device data (Data register: 32 points)	Reading programs (10K step)
Q00UJCPU, Q00UCPU, Q01UCPU	1.60ms	3.70ms
Q02UCPU	1.00ms	1.55ms
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU	0.35ms	0.95ms
Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU	0.90ms	1.10ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.29ms	0.95ms

*1 The time in the table is for the case where the service processing count is set to one.

(b) Processing time to set monitor conditions

CPU module	Processing time	
	Specified step match	Specified device match
Q02UCPU	0.03ms	0.04ms
Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU	0.01ms	0.03ms

(8) Common processing time

The CPU module performs common processing by the system. The common processing time shown below is required.

CPU module	Processing time
Q00UJCPU, Q00UCPU, Q01UCPU	0.28ms
Q02UCPU	0.20ms
Q03UDCPU	0.13ms
Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU	0.10ms
Q03UDECPU	0.22ms
Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU	0.18ms
Q03UDVCPU	0.140ms*1
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.125ms*1

*1 At default setting of parameters, and including processing time for latching 8K points of latch relay (L).

(9) Multiple CPU high speed transmission processing time

This is the processing time required for data transmission between the CPU modules when the multiple CPU high speed transmission function is used.

For the multiple CPU high speed transmission processing time, refer to the following.

QCPU User's Manual (Multiple CPU System)

Appendix 3.3 Factors that increase the scan time

When executing any of the functions or operations described in this section, add the given processing time in this section to the time value calculated in Page 470, Appendix 3.1.

(1) Sampling trace

When the sampling trace function (☞Page 184, Section 3.14) is executed, the processing time listed in the following table is required.

[Conditions] Processing time (when 50 points of the internal relay (for bit device) and 50 points of the data register (for word device) are set as sampling trace data)

CPU module		Processing time
Standard RAM	Q00UCPU, Q01UCPU	0.12ms
	Q02UCPU	0.09ms
	Q03UD(E)CPU	0.07ms*1
	Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	0.06ms*1
	Q50UDEHCPU, Q100UDEHCPU	0.06ms
	Q03UDVCPU	0.045ms*1
	Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.045ms*1
Standard RAM (with an extended SRAM cassette)	Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.05ms*1
SRAM card	Q02UCPU	0.09ms
	Q03UD(E)CPU	0.08ms*1
	Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU	0.06ms*1
	Q50UDEHCPU, Q100UDEHCPU	0.06ms

*1 When specifying devices in the trigger point setting, the scan time may be increased by up to 0.163μs per instruction.

(2) Use of local devices

When local devices are used, the processing time listed in the following table is required. Each n, N1, N2, N3, and N4 in the table indicates the following.

- n: Number of programs using a local device*1
- N1: Number of devices that specified a local device
- N2: Number of word device points that specified a local device (except index register)
- N3: Number of bit device points that specified a local device
- N4: Number of index register points that were specified as a local device

CPU module		Processing time
Standard RAM	Q00UCPU, Q01UCPU	((16.0 × N1) + (0.23 × (N2 + (N3 ÷ 16))) + (1.49 × N4) + 98.3) × n +92.0μs
	Q02UCPU	((24.0 × N1) + (0.23 × (N2 + (N3 ÷ 16))) + (1.57 × N4) + 108) × n +59.0μs
	Q03UD(E)CPU	((8.0 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + (0.65 × N4) + 38.0) × n + 14.2μs
	Q03UDVCPU	((4.1 × N1) + (0.165 × (N2 + (N3 ÷ 16))) + (0.15 × N4) + 31.0) × n + 10.0μs
	Q04UD(E)HCPU, Q06UD(E)HCPU	((8.0 × N1) + (0.10 × (N2 + (N3 ÷ 16))) + (0.47 × N4) + 35.5) × n + 12.7μs
	Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	((8.0 × N1) + (0.10 × (N2 + (N3 ÷ 16))) + (0.68 × N4) + 35.5) × n + 17.3μs
	Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	((4.0 × N1) + (0.085 × (N2 + (N3 ÷ 16))) + (0.15 × N4) + 31.0) × n + 10.0μs
Standard RAM (with an extended SRAM cassette)	Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	((4.2 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + (0.20 × N4) + 31.0) × n + 10.0μs
SRAM card	Q02UCPU	((24.0 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + (1.40 × N4) + 66.0) × n + 83.0μs
	Q03UD(E)CPU	((12.0 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + (0.68 × N4) + 41.0) × n + 17.0μs
	Q04UD(E)HCPU, Q06UD(E)HCPU	((12.0 × N1) + (0.40 × (N2 + (N3 ÷ 16))) + (0.59 × N4) + 38.5) × n + 17.0μs
	Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	((12.0 × N1) + (0.40 × (N2 + (N3 ÷ 16))) + (0.79 × N4) + 44.7) × n + 12.1μs

*1 When the serial number (first five digits) of the Q02UCPU, Q03UDCPU, Q04UDHCPU, or Q06UDHCPU is "10011" or earlier, "n" indicates the number of executed programs.

(a) When local devices in a subroutine program are enabled

When SM776 (Enable/disable local device at CALL) is turned on, the processing time listed in the following table is required for each subroutine call. Each n, N1, N2, N3, and N4 in the table indicates the following.

- n: Number of local device points (unit: K words)
- N1: Number of devices that specified a local device
- N2: Number of word device points that specified a local device (except index register)
- N3: Number of bit device points that specified a local device
- N4: Number of index register points that were specified as a local device

CPU module	When a local device file in the standard RAM is used	
	Processing time when a subroutine program in the same file is called	Processing time when a subroutine program in a different file is called
Q00UCPU, Q01UCPU	0.00μs	(20.3 × N1) + (0.760 × (N2 + (N3 ÷ 16))) + (4.47 × N4) + 257.0μs
Q02UCPU	0.00μs	(20.3 × N1) + (0.760 × (N2 + (N3 ÷ 16))) + (4.71 × N4) + 257.0μs
Q03UD(E)CPU	0.00μs	(16 × N1) + (0.44 × (N2 + (N3 ÷ 16))) + (1.30 × N4) + 100μs
Q03UDVCPU	0.00μs	(6.1 × N1) + (0.330 × (N2 + (N3 ÷ 16))) + (0.30 × N4) + 66μs
Q04UD(E)HCPU, Q06UD(E)HCPU	0.00μs	(16 × N1) + (0.20 × (N2 + (N3 ÷ 16))) + (0.94 × N4) + 100μs
Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.00μs	(16 × N1) + (0.20 × (N2 + (N3 ÷ 16))) + (1.36 × N4) + 100μs
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.00μs	(6 × N1) + (0.162 × (N2 + (N3 ÷ 16))) + (0.30 × N4) + 66μs

CPU module	When a local device file in the standard RAM is used (with an extended SRAM cassette)	
	Processing time when a subroutine program in the same file is called	Processing time when a subroutine program in a different file is called
Q03UDVCPU	0.00μs	(6.2 × N1) + (0.440 × (N2 + (N3 ÷ 16))) + (0.40 × N4) + 66μs
Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.00μs	(6.2 × N1) + (0.432 × (N2 + (N3 ÷ 16))) + (0.40 × N4) + 66μs

CPU module	When a local device file in the SRAM card is used	
	Processing time when a subroutine program in the same file is called	Processing time when a subroutine program in a different file is called
Q02UCPU	0.00μs	(20.3 × N1) + (0.760 × (N2 + (N3 ÷ 16))) + (2.80 × N4) + 257.0μs
Q03UD(E)CPU	0.00μs	(24 × N1) + (0.86 × (N2 + (N3 ÷ 16))) + (1.36 × N4) + 120μs
Q04UD(E)HCPU, Q06UD(E)HCPU	0.00μs	(24 × N1) + (0.80 × (N2 + (N3 ÷ 16))) + (1.18 × N4) + 100μs
Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.00μs	(24 × N1) + (0.80 × (N2 + (N3 ÷ 16))) + (1.58 × N4) + 120μs

(3) Execution of multiple programs

When multiple programs are executed, the processing time listed in the following table is required for each program.

CPU module	Processing time
Q00UJCPU, Q00UCPU, Q01UCPU	0.053 × n*1 ms
Q02UCPU	0.04 × n*1 ms
Q03UD(E)CPU	0.02 × n*1 ms
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.02 × n*1 ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.010 × n*1 ms

*1 "n" indicates the number of program files.

(4) Removal and insertion of a memory card

When a memory card is removed or inserted, the processing time listed in the following table is required only for one scan where a memory card is removed or inserted.

CPU module	Processing time	
	When a memory card is inserted	When a memory card is removed
Q02UCPU	0.7ms	0.2ms
Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.6ms	0.1ms

(5) Removal and insertion of an SD memory card

When an SD memory card is removed or inserted, the processing time listed in the following table is required only for one scan where a memory card is removed or inserted.

CPU module	Processing time	
	When an SD memory card is inserted	When an SD memory card is removed
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.59ms	0.32ms

(6) Use of the file register

When "Use the same file name as the program." is selected in the PLC file tab of the PLC parameter dialog box, the processing time listed in the following table is required. When "Use the following file." is selected, the scan time will not be increased.

CPU module		Processing time
Standard RAM	Q00UCPU, Q01UCPU	0.135ms
	Q02UCPU	0.082ms
	Q03UD(E)CPU	0.043ms
	Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	0.041ms
	Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	$0.016 \times n^{*1}$ ms
Standard RAM (with an extended SRAM cassette)	Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	$0.016 \times n^{*1}$ ms
SRAM card	Q02UCPU	$0.11 \times n^{*1}$ ms
	Q03UD(E)CPU	$0.06 \times n^{*1}$ ms
	Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	$0.06 \times n^{*1}$ ms

*1 "n" indicates the number of program files.

(7) Online change

When data is written to the running CPU module, the processing time described below is required.

(a) Online change (ladder mode)

When a program in the running CPU module is changed in ladder mode, the processing time listed in the following table is required.*1

*1 The time in the table is for the case where the service processing count is set to one.

CPU module	Reserved area for online change	
	The reserved area for online change is not changed.	The reserved area for online change is re-set.
Q00UJCPU, Q00UCPU, Q01UCPU	Up to 2.1ms	Up to 2.1ms
Q02UCPU	Up to 1.3ms	Up to 1.3ms
Q03UD(E)CPU	Up to 1.0ms	Up to 1.0ms
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	Up to 0.7ms	Up to 0.7ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	Up to 0.6ms	Up to 0.6ms

(b) Online change (files)

When a file is written to the running CPU module, the processing time listed in the following table is required.*1

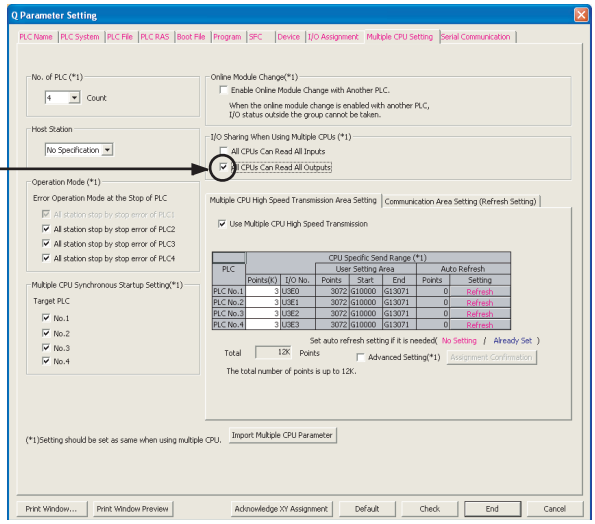
*1 The time in the table is for the case where the service processing count is set to one.

CPU module	Processing time	
	Scan time = 2ms	Scan time = 20ms
Q00UJCPU	Up to 4.00ms	Up to 6.20ms
Q00UCPU	Up to 3.50ms	Up to 5.80ms
Q01UCPU	Up to 3.50ms	Up to 5.60ms
Q02UCPU	Up to 4.80ms	Up to 4.80ms
Q03UD(E)CPU	Up to 3.75ms	Up to 3.75ms
Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	Up to 3.70ms	Up to 3.70ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	Up to 0.8ms	Up to 0.8ms

(8) Non-group output status read

In multiple CPU systems, the scan time increases when "All CPUs Can Read All Outputs" is selected in the Multiple CPU settings screen of the PLC parameter dialog box.

The scan time increases when this parameter is set.



(9) Scan time measurement

When the scan time is measured (Page 181, Section 3.13.3), the processing time listed in the following table is required.

CPU module	Processing time
Q00UJCPU	180.3 + 5.9 × number of branch instructions μs*1
Q00UCPU, Q01UCPU	179.5 + 5.8 × number of branch instructions μs*1
Q02UCPU	40.0 + 3.0 × number of branch instructions μs*1
Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	40.0 + 1.5 × number of branch instructions μs*1
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	30.0 + 0.40 × number of branch instructions μs*1

*1 The number of the branch instructions is a total of the following instructions, which are executed during the scan time measurement.
- Pointer branch instruction: CJ, SCJ, JMP
- Subroutine program call instruction: CALL(P), FCALL(P), ECALL(P), EFCALL(P), XCALL(P), RET

(10)Monitor condition setting

When the monitor condition is set (Page 146, Section 3.11.1), the processing time listed in the following table is required.

CPU module	Processing time	
	Specified step matches the execution condition	Specified device matches the execution condition (Device D matches the condition)
Q02UCPU	30μs	40μs
Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU	10μs	30μs

(11)Time taken to collect module errors

When using the module error collection, the scan time increases by the time found by the following calculation formula. Each N1 and N2 in the calculation formula indicates the following.

■ Calculation formula

Collection time = N1 + N2 × (Number of module errors collected in one scan)

CPU module	Main base unit		Extension base unit	
	N1	N2	N1	N2
Q00UJCPU	175μs	---	190μs	---
Q00UCPU, Q01UCPU	145μs	120μs	190μs	140μs
Q02UCPU	145μs	90μs	185μs	105μs
Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	15μs	70μs	15μs	100μs

CPU module	Main base unit		Extension base unit	
	N1	N2	N1	N2
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	6μs	45μs	6μs	70μs

(12)Batch transfer of data to the program memory

When data in the program cache memory is batch-transferred to the program memory, the processing time listed in the following table is required.*1

*1 The time in the table is for the case where the service processing count is set to one.

CPU module	Processing time	
	Scan time = 2ms	Scan time = 20ms
Q00UJCPU, Q00UCPU, Q01UCPU	2.35ms	5.10ms
Q02UCPU	2.35ms	4.50ms
Q03UDCPU	1.10ms	3.65ms
Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU	1.05ms	3.65ms
Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU, Q50UDEHCPU, Q100UDEHCPU	1.40ms	4.00ms
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.60ms	0.60ms

(13) Diagnostics of the redundant power supply system

When the "Diagnose Redundant Power Supply System" is selected on the PLC RAS tab of the PLC parameter dialog box, the processing time listed in the following table is required.

CPU module	Processing time	
	With a power supply module failure*1	Without a power supply module failure*1
Q00UCPU	125μs	165μs
Q01UCPU	125μs	135μs
Q02UCPU	90μs	90μs
Q03UD(E)CPU, Q03UDVCPU, Q04UD(E)HCPU, Q04UDVCPU, Q04UDPVCPU, Q06UD(E)HCPU, Q06UDVCPU, Q06UDPVCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q13UDVCPU, Q13UDPVCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q26UDVCPU, Q26UDPVCPU, Q50UDEHCPU, Q100UDEHCPU	43μs	52μs

*1 A power supply module failure indicates any of the following.
- The redundant power supply module has failed.
- Power for the redundant power supply module is turned off.
- The redundant power supply module is not mounted.

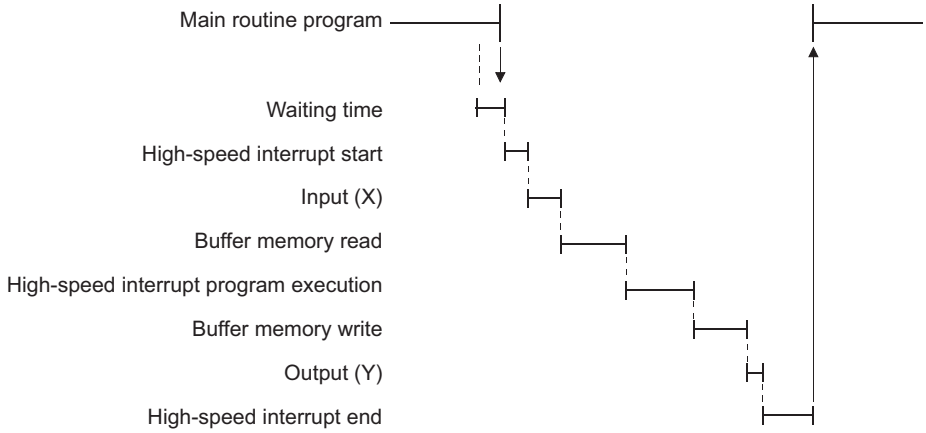
(14)A-PLC compatibility setting

When "A-PLC Compatibility Setting" is enabled in the PLC System tab of the PLC Parameter dialog box, the processing time listed in the following table is required.

CPU module	Processing time
Q00UJCPU, Q00UCPU, Q01UCPU	95μs
Q02UCPU	90μs
Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU, Q50UDEHCPU, Q100UDEHCPU	34μs
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	13μs

(15)High-speed interrupt function

The high-speed interrupt function performs the following operations.



The processing time of each operation is as follows.

(a) Waiting time

For the waiting time before a high-speed interrupt starts, refer to Page 229, Section 3.21.3 (1).

(b) High-speed interrupt start

For the overhead time at the startup of a high-speed interrupt, refer to Page 472, Appendix 3.2 (3) (a).

(c) Input (X)

The processing time listed in the following table is required.

Processing time = (KM1 × total number of X points) + (KM2 × number of setting points) + KM3 [μs]

CPU module	Main base unit			Extension base unit		
	KM1	KM2	KM3	KM1	KM2	KM3
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.09	0.20	4.50	0.14	0.20	6.00

(d) Buffer memory read

The processing time listed in the following table is required.

Processing time = (KM1 × total number of words transferred) + (KM2 × number of setting points) + KM3 [μs]

CPU module	Read data size: 16 words or less						Read data size: more than 16 words					
	Main base unit			Extension base unit			Main base unit			Extension base unit		
	KM1	KM2	KM3	KM1	KM2	KM3	KM1	KM2	KM3	KM1	KM2	KM3
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	1.25	0.55	10.00	2.75	0.56	27.00	0.36	0.55	10.00	0.90	0.56	27.00

(e) Buffer memory write

The processing time listed in the following table is required.

Processing time = (KM1 × total number of words transferred) + (KM2 × number of setting points) + KM3 [μs]

CPU module	Write data size: 16 words or less						Write data size: more than 16 words					
	Main base unit			Extension base unit			Main base unit			Extension base unit		
	KM1	KM2	KM3	KM1	KM2	KM3	KM1	KM2	KM3	KM1	KM2	KM3
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	1.17	1.30	15.00	2.57	0.50	23.00	0.35	1.30	15.00	0.90	0.50	23.00

(f) Output (Y)

The processing time listed in the following table is required.

Processing time = (KM1 × total number of Y points) + (KM2 × number of setting points) + KM3 [μs]

CPU module	Main base unit			Extension base unit		
	KM1	KM2	KM3	KM1	KM2	KM3
Q03UDVCPU, Q04UDVCPU, Q04UDPVCPU, Q06UDVCPU, Q06UDPVCPU, Q13UDVCPU, Q13UDPVCPU, Q26UDVCPU, Q26UDPVCPU	0.06	0.20	2.80	0.11	0.20	2.60

(g) High-speed interrupt end

For the overhead time at the end of a high-speed interrupt, refer to Page 473, Appendix 3.2 (3) (b).

Appendix 4 Data Used in Sequence Programs

The CPU module represents numeric and alphabetic data using two symbols (states): 0 (off) and 1 (on).
Data represented using these two symbols is called binary number (BIN).
The CPU module can also use hexadecimal (HEX) (each hexadecimal digit represents four binary bits), binary-coded decimal (BCD), or real numbers.
The following table lists the numeric representations of BIN, HEX, BCD, and DEC (decimal).

BIN (Binary)				DEC (Decimal)	HEX (Hexadecimal)	BCD (Binary-coded decimal)			
			0	0	0				0
			1	1	1				1
			10	2	2				10
			11	3	3				11
			•	4	4				•
			•	5	5				•
			•	6	6				•
			•	7	7				•
			•	8	8				•
			1001	9	9				1001
			1010	10	A			1	0000
			1011	11	B			1	0001
			1100	12	C			1	0010
			1101	13	D			1	0011
			1110	14	E			1	0100
			1111	15	F			1	0101
		1	0000	16	10			1	0110
		1	0001	17	11			1	0111
		•	•	•	•			•	•
		•	•	•	•			•	•
		•	•	•	•			•	•
		10	1111	47	2F			100	0111
		•	•	•	•				
		•	•	•	•				
		•	•	•	•				
0111	1111	1111	1110	32766	7FFE	-			
0111	1111	1111	1111	32767	7FFF	-			
1000	0000	0000	0000	-32768	8000	1000	0000	0000	0000
1000	0000	0000	0001	-32767	8001	1000	0000	0000	0001
•	•	•	•	•	•				
•	•	•	•	•	•				
•	•	•	•	•	•				
1111	1111	1111	1110	-2	FFFE	-			
1111	1111	1111	1111	-1	FFFF	-			

(1) Inputting numeric values externally to the CPU module

When setting a numeric value to the CPU module externally using a digital switch, BCD (binary-coded decimal) can be used as DEC (decimal) by the method given below.

(a) Numeric values used inside the CPU module

The CPU module performs program operations in binary.

If the value set in binary-coded decimal is used without conversion, the CPU module performs program operations regarding the set value as binary.

Therefore, the program operations are not performed correctly.

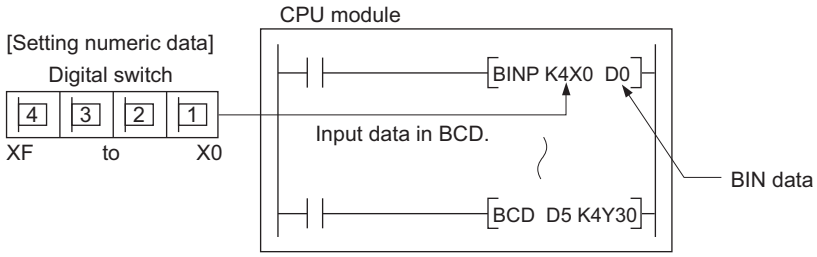
(b) Using any numeric data regardless of the data type

To convert the data set in binary-coded decimal into binary, which can be used in the CPU module, use the BIN instruction.

The BIN instruction allows the CPU module to use any external numeric data regardless of the data type.

For details of the BIN instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)



(2) Outputting numeric values externally from the CPU module

When externally displaying numeric values operated in the CPU module, a digital indicator can be used.

(a) Outputting numeric values

The CPU module performs program operations in binary.

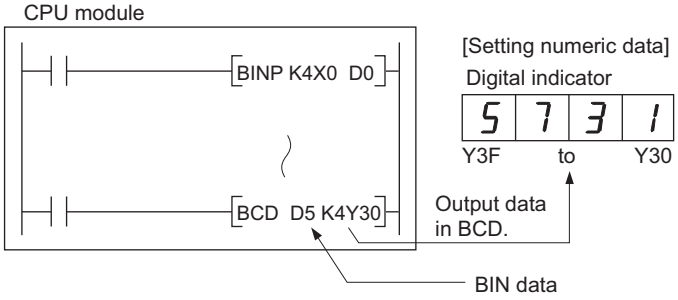
If the binary values used in the CPU module are output to a digital indicator, the indicator does not show the values correctly.

To convert the data set in binary into binary-coded decimal, which can be used in the external indicator, use the BCD instruction.

The BCD instruction allows the external indicator to display values in decimal.

For details of the BCD instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)



Appendix 4.1 BIN (Binary Code)

(1) Definition

Binary is a numeral system that represents numeric values using two symbols, 0 (off) and 1 (on).
Decimal notation uses the symbols 0 through 9. When the symbols for the first digit are exhausted (a digit reaches 9), the next-higher digit (to the left) is incremented, and counting starts over at 0.
In binary notation, only the symbols 0 and 1 are used. After a digit reaches 1, an increment resets it to 0 and the next digit (to the left) is incremented. (The numeric value becomes 10, which is equal to 2 in decimal.)
The following table lists the numeric representations in BIN and DEC.

DEC (Decimal)	BIN (Binary)	
0	0000	
1	0001	
2	0010	Carry
3	0011	
4	0100	Carry
5	0101	
6	0110	
7	0111	
8	1000	Carry
9	1001	
10	1010	
11	1011	

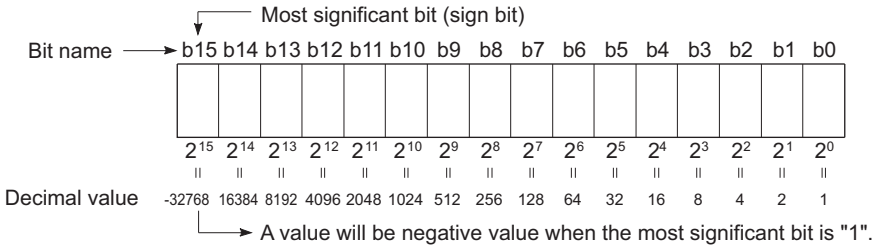
(2) Numeric representation in BIN

(a) Bit configuration of BIN used in the CPU module

Each register (such as the data register, link register) in the CPU module consists of 16 bits.

(b) Numeric data available in the CPU module

Each register in the CPU module can store numeric values in the range of -32768 to 32767.
The following figure shows the numeric representations for registers.



Point

A numeric value of 2^n is assigned for each bit of registers.
Note that an unsigned binary number (0 to 65535) cannot be used in the most significant bit position since the most significant bit is a sign bit.

- The most significant bit is "0"...Positive
- The most significant bit is "1"...Negative

Appendix 4.2 HEX (Hexadecimal)

(1) Definition

Hexadecimal (HEX) is a numeral system that represents four binary bits as one digit.
With four binary bits, sixteen different numeric values, 0 to 15, can be represented.
Hexadecimal notation uses 16 symbols to represent numeric values 0 to 15 in one digit, the symbols 0 to 9 to represent values zero to nine, and A_H to F_H to represent values ten to fifteen. After a digit reaches F_H, the next-higher digit (to the left) is incremented.
The following table lists the numeric representations in BIN, HEX, and DEC (decimal).

DEC (Decimal)	HEX (Hexadecimal)	BIN (Binary)	
0	0		0
1	1		1
2	2		10
3	3		11
•	•		•
•	•		•
•	•		•
9	9		1001
10	A		1010
11	B		1011
12	C		1100
13	D		1101
14	E		1110
15	F		1111
16	10	1	0000
17	11	1	0001
•	•		•
•	•		•
•	•		•
47	2F	10	1111

Carry

(2) Numeric representation in HEX

Each register (such as the data register, link register) in the CPU module consists of 16 bits.
In the 16-bit configuration register, 0 to $FFFF_H$ can be specified in hexadecimal.

Appendix 4.3 BCD (Binary-coded Decimal)

(1) Definition

BCD is a numeral system that uses four binary bits to represent the decimal digits 0 through 9.
The difference from hexadecimal is that BCD does not use letters A to F.
The following table lists the numeric representations in BIN, BCD, and DEC.

DEC (Decimal)	BIN (Binary)	BCD (Binary-coded Decimal)	
0	0000		0
1	0001		1
2	0010		10
3	0011		11
4	0100		100
5	0101		101
6	0110		110
7	0111		111
8	1000		1000
9	1001		1001
10	1010	1	0000
11	1011	1	0001
12	1100	1	0010

Carry

(2) Numeric representation in BCD

Each register (such as the data register, link register) in the CPU module consists of 16 bits.
Therefore, the numeric values can be stored in each register are those in the range between 0 to 9999 in BCD.

Appendix 4.4 Real number (Floating-point data)

There are two types of real number data: single-precision floating-point data and double-precision floating-point data.

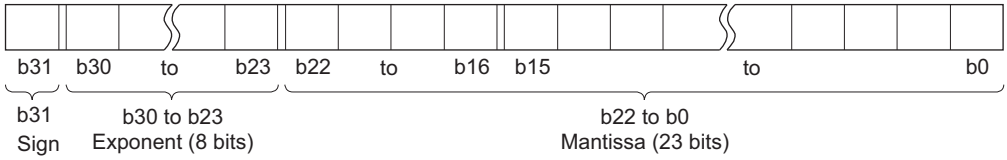
(1) Single-precision floating-point data

(a) Internal representation

Internal representation of real numbers used in the CPU module is given below.
Real number data can be represented as follows, using two word devices.

[Sign] 1. [Mantissa] × $2^{[Exponent]}$

The bit configuration and the meaning of each bit are described below.



- Sign
  The most significant bit, b31, is the sign bit.
  0: Positive
  1: Negative
- Exponent
  The 8 bits, b23 to b30, represent the excess n of 2^n.
  The following shows the excess n according to the binary values in b23 to b30.

b23 to b30	FF$_H$	FE$_H$	FD$_H$		81$_H$	80$_H$	7F$_H$	7E$_H$		02$_H$	01$_H$	00$_H$
n	Not used	127	126		2	1	0	-1		-125	-126	Not used

- Mantissa
  Each of the 23 bits, b0 to b22, represents the "XXXXXX..." portion when the data is represented in binary, "1.XXXXXX...".

(b) Calculation example

Calculation examples are shown below. (The "X" in (nnnnnn) x indicates the numeral system used.)

- Storing "10"

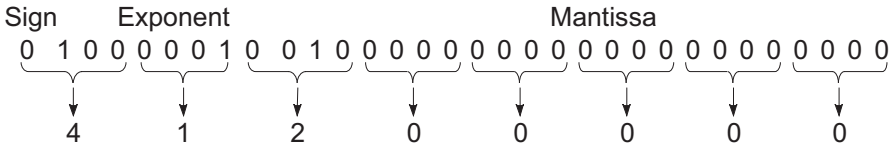
$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.010000..... \times 2^3)_2$

Sign: Positive → 0

Exponent: $3 \rightarrow 82_H \rightarrow (10000010)_2$

Mantissa: $(010\ 00000\ 00000\ 00000\ 00000)_2$

In this case, the value will be encoded as 41200000_H.



- Storing "0.75"

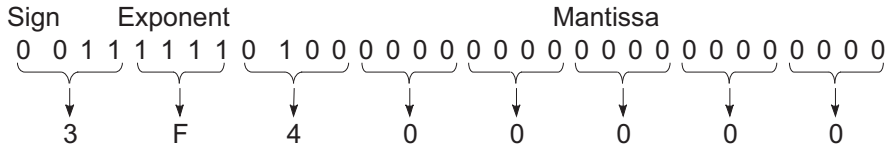
$(0.75)_{10} \rightarrow (0.11)_2 \rightarrow (1.100..... \times 2^{-1})_2$

Sign: Positive → 0

Exponent: $-1 \rightarrow 7E_H \rightarrow (01111110)_2$

Mantissa: $(100\ 00000\ 00000\ 00000\ 00000)_2$

In this case, the value will be encoded as $3F400000_H$.



Point

Values after the decimal point (in binary) is calculated as follows.

Ex. $(0.1101)_2$

0.	1	1	0	1
	↑	↑	↑	↑
	The bit represents 2^{-1}.	The bit represents 2^{-2}.	The bit represents 2^{-3}.	The bit represents 2^{-4}.

$(0.1101)_2 = 2^{-1} + 2^{-2} + 2^{-4} = 0.5 + 0.25 + 0.0625 = (0.8125)_{10}$

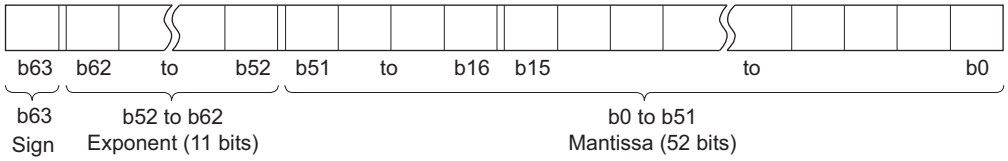
(2) Double-precision floating-point data

(a) Internal representation

Real number data used in the CPU module is internally represented as follows, using four word devices.

[Sign] 1. [Mantissa] × $2^{[Exponent]}$

The bit configuration and the meaning of each bit are described below.



- Sign

  The most significant bit, b63, is the sign bit.

  0: Positive

  1: Negative

- Exponent

  The 11 bits, b52 to b62, represent the excess n of 2^n.

  The following shows the excess n according to the binary values in b52 to b62.

b52 to b62	7FF$_H$	7FE$_H$	7FD$_H$			400$_H$	3FF$_H$	3FE$_H$	3FD$_H$	3FC$_H$			02$_H$	01$_H$	00$_H$
n	Not used	1023	1022			1	0	-1	-2	-3			-1021	-1022	Not used

- Mantissa

  Each of the 52 bits, b0 to b51, represents the "XXXXXX..." portion when the data is represented in binary, "1.XXXXXX...".

(b) Calculation example

Calculation examples are shown below. (The "X" in (nnnnnn) x indicates the numeral system used.)

- Storing "10"

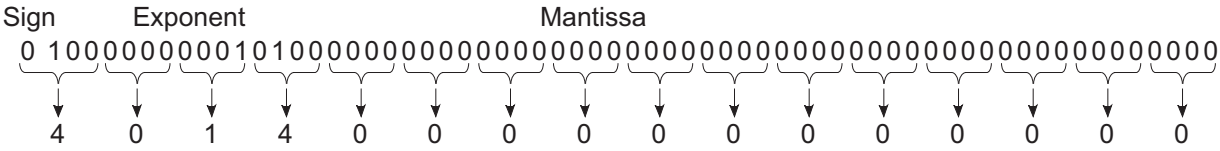
$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.010000..... \times 2^3)_2$

Sign: Positive → 0

Exponent: $3 \rightarrow 401_H \rightarrow (100\ 0000\ 0001)_2$

Mantissa: $(0100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000)_2$

In this case, the value will be encoded as 4014000000000000_H.



- Storing "0.75"

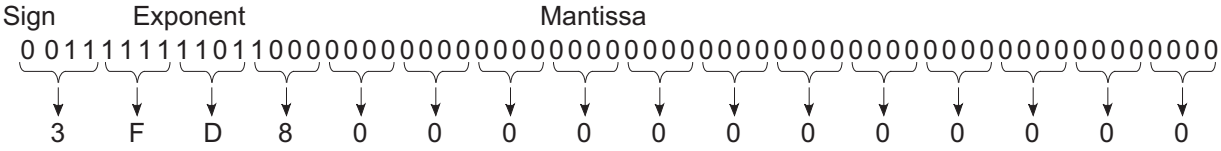
$(0.75)_{10} \rightarrow (0.11)_2 \rightarrow (1.100..... \times 2^{-1})_2$

Sign: Positive → 0

Exponent: $-1 \rightarrow 3FD_H \rightarrow (011\ 1111\ 1101)_2$

Mantissa: $(1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000)_2$

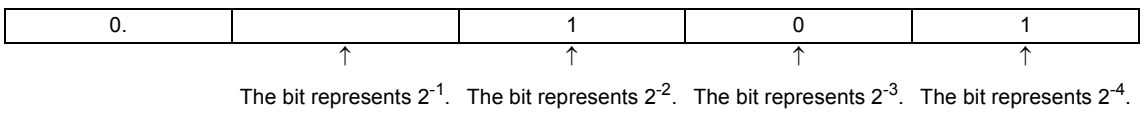
In this case, the value will be encoded as $3FD8000000000000_H$.



Point

Values after the decimal point (in binary) is calculated as follows.

Ex. $(0.1101)_2$



$(0.1101)_2 = 2^{-1} + 2^{-2} + 2^{-4} = 0.5 + 0.25 + 0.0625 = (0.8125)_{10}$

Appendix 4.5 Character string data

(1) Definition

The CPU module uses shift JIS code character strings.

Appendix 5 Replacing Basic Model QCPU or Qn(H)CPU with QnUCPU

Appendix 5.1 Replacement precautions

This section describes precautions for replacing the Basic model QCPU or High Performance model QCPU with the Universal model QCPU and the replacement methods.

Appendix 5.1.1 Replacing Basic model QCPU with Universal model QCPU

(1) System configuration

Item	Precautions	Replacement method	Reference
GOT	GOT900 series cannot be connected.	Use GOT1000 or GOT2000 series.	---
Applicable products and software	Products and software compatible with the Universal model QCPU must be used.	Products need to be replaced for the compatibility with the Universal model QCPU and software need to be upgraded for the communication with the Universal model QCPU are described in Page 514, Appendix 5.2.	Page 514, Appendix 5.2
Multiple CPU system	To configure a multiple CPU system, CPU modules compatible with the Universal model QCPU must be used.	CPU modules compatible with the Universal model QCPU are described in Page 514, Appendix 5.2.	Page 514, Appendix 5.2

(2) Program

Item	Precautions	Replacement method	Reference
Language and instruction	Some instructions are not supported.	Replace the instructions not supported in the Universal model QCPU are described in Page 519, Appendix 5.3.	Page 519, Appendix 5.3
Floating-point operation	When using the floating-point data comparison instructions, LDE□, ANDE□, ORE□, LDED□, ANDED□, and ORED□, if the comparison source data are -0, nonnumeric, unnormalized number, or ±∞ , "OPERATION ERROR" (error code: 4101) is detected.*1 (□ indicates one of the following; =,<>,<=,>=,<,>.)	When the floating-point data comparison instructions are used, modify the program as described in Page 543, Appendix 5.4.2.	Page 498, Appendix 4.4, Page 543, Appendix 5.4.2
Device range check at index modification	When a device number exceeds a setting range due to index modification, "OPERATION ERROR" (error code: 4101) is detected.	Deselect the "Check Device Range at Indexing" checkbox in the PLC RAS tab of the PLC parameter dialog box so that checking is not performed.	Page 547, Appendix 5.4.3
Latch setting	If latch ranges of internal user devices are specified, the processing time is added in proportion to the device points set to be latched.	The latch function of the Universal model QCPU is enhanced. (1) Large-capacity file register (R, ZR) (2) Writing/reading device data to the standard ROM (SP.DEVST and S(P).DEVLD instructions) (3) Latch range specification of internal devices (4) "Time Setting" specification in the latch interval setting parameter*2 Change the latch method to the one described above according to the application.	Page 122, Section 3.3, Page 124, Section 3.3 (5) (b), Page 551, Appendix 5.4.4

Item	Precautions	Replacement method	Reference
Interrupt counter	Interrupt counter is not supported.	Check the numbers of executions for interrupt programs on the Interrupt program monitor list screen.	Page 180, Section 3.13.2
SCJ instruction	When the SCJ instruction is used in the Universal model QCPU, the AND SM400 (or NOP instruction) needs to be inserted immediately before the SCJ instruction.	Insert the AND SM400 (or NOP instruction) immediately before the SCJ instruction when the SCJ instruction is used.	Section 6.5 in the MELSEC-Q/L Programming Manual (Common Instruction)
JP/GP.SREAD and JP/GP.SWRITE instructions	Use of the completion notification device of the target station specified in D3 when the SREAD and SWRITE instructions are used is available. (The Basic model QCPU ignores the device specified in D3.)	To get the same operation as the Basic model QCPU, omit D3 or use the READ instruction instead of the SREAD instruction.	Section 8.3.2, 8.3.4 in the QnACPU Programming Manual (Common Instructions)
ZPUSH instruction	The number of index registers is increased to 20 for the Universal model QCPU. The area for saving the data in the index register with the ZPUSH instruction is increased as well.	Increase the save area used for the ZPUSH instruction as needed.	Section 7.19 in the MELSEC-Q/L Programming Manual (Common Instruction)
Use of the annunciator (SET F□ and OUT F□ instructions)	When the annunciator is turned on by the SET F□ or OUT F□ instruction, the USER LED turns on. (The ERR.LED does not turn on.)	---	---
I/O refresh between programs	I/O refresh between programs cannot be executed.	Execute I/O refresh at the start or end of each program with the RFS or COM instruction. (When the COM instruction is used, I/O refresh to be executed can be specified in SD778 by turning on SM774.)	---
SM/SD	Usage of a part of the special relay and special register is different.	Replace the corresponding special relay and special register as described in Page 563, Appendix 5.5.	Page 563, Appendix 5.5
Multiple CPU system	The start address for the user setting area (auto refresh) in the CPU shared memory is changed.	If the user setting area in the CPU shared memory is specified in the program, change the address for the user setting area by performing an operation for replacing a device in a programming tool. (Example: "MOV D0 U3E0\G192" → "MOV D0U3E0\G2048")	---
File register	To use the file register, capacity setting is required.	Set the capacity of the file register used in the PLC file tab of the PLC parameter.	Page 447, Appendix 1.2.8
SFC program	The following settings are required for using SFC programs. Program setting (when both sequence programs and SFC programs exist) Common pointer No. setting (to execute the CALL instruction from SFC programs)	Set program details in the Program tab of the PLC parameter dialog box. Enter a common pointer number in the PLC system of the PLC parameter dialog box.	
Number of steps	The number of steps increases by one*3 when: • Index modification is performed. • A leading or trailing edge instruction is used. • Bit devices are used as word data by specifying digits using K1, K2, K3, K5, K6, or K7, or by specifying a device number of other than multiples of 16.	If index modifications mentioned on the left are frequently used in the program, the program size may exceed the storage capacity of the replaced CPU module. After the program controller type is changed, check the program size using the confirm memory size function. If the program size exceeds the storage capacity, take the following actions or change the CPU module to that with larger program memory. • Move parameters and device comments to the standard ROM. • Reduce the reserved area for online change. • Use the file register, extended data register, and extended link register within 64K words because the number of steps decreases by one when used in that way.	MELSEC-Q/L Programming Manual (Common Instruction)

*1 This will not apply when the Basic model QCPU is replaced with the High-speed Universal model QCPU and Universal model Process CPU.

*2 Only the High-speed Universal model QCPU and Universal model Process CPU support this setting.

*3 This will apply only when the Basic model QCPU is replaced with the High-speed Universal model QCPU and Universal model Process CPU.

(3) Drive and file

Item	Precautions	Replacement method	Reference
Boot file setting	The boot file setting is not supported.	Since the Universal model QCPU holds the data in the program memory even when the battery voltage drops, the boot file setting is not necessary. Move files with the boot setting (from the standard ROM to the program memory) to the program memory.	Page 104, Section 2.11

(4) External communication (Service processing)

Item	Precautions	Replacement method	Reference
Time reserved for communication processing	The time reserved for communication processing (SM315/SD315) is not supported.	Set service processing time in the PLC system tab of the PLC parameter dialog box.	Page 241, Section 3.24.1
MC protocol	The following commands cannot specify monitoring conditions. Randomly reading data in units of word (Command: 0403) Device memory monitoring (Command: 0801) The applicable frame types are as follows: QnA-compatible 3C/4C frame QnA-compatible 3E frame 4E frame	---	MELSEC Communication Protocol Reference Manual

(5) Battery installation position

Item	Precautions	Replacement method	Reference
Battery installation position	The battery replacement method is different. The battery installation position varies depending on the model. • Q00JCPU, Q00CPU, Q01CPU ...On the front of the module. • Q00UJCPU, Q00UCPU, Q01UCPU ...At the bottom of the module.	For the battery replacement method, refer to the in the Reference column.	QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(6) Program size

Item	Precautions	Replacement method	Reference
Program size	Data in the program memory of the Basic model QCPU may exceed the size of the program memory of the Universal model QCPU.	Store parameter and device comment files in the standard ROM.	---

Appendix 5.1.2 Replacing High Performance model QCPU with Universal model QCPU

(1) System configuration

Item	Precautions	Replacement method	Reference
Use of AnS/A series module	The Universal model QCPU whose serial number (first 5 digits) is "13102" or later must be used.	Use Q series modules when using the Universal model QCPU whose serial number (first 5 digits) is "13101" or earlier.	---
GOT	GOT900 series cannot be connected.	Use GOT1000 or GOT2000 series.	---
Programming tool connection	Applicable USB cables are different. • High Performance model QCPU ... A-B type • Universal model QCPU ... A-miniB type	Use USB cables of A-miniB type. Or, use USB conversion adapters of B-miniB type.	---
Applicable products and software	Products and software compatible with the Universal model QCPU must be used.	Products need to be replaced for the compatibility with the Universal model QCPU and software need to be upgraded for the communication with the Universal model QCPU are described in Page 514, Appendix 5.2.	Page 514, Appendix 5.2
Multiple CPU system	To configure a multiple CPU system, CPU modules compatible with the Universal model QCPU must be used.	CPU modules compatible with the Universal model QCPU are described in Page 514, Appendix 5.2.	Page 514, Appendix 5.2
	In a multiple CPU system using the Motion CPU, an existing auto refresh area and user setting area cannot be used for data communication with the Motion CPU.	For data communication with the Motion CPU, use an auto refresh area and user setting area in the multiple CPU high-speed transmission area.	Chapter 4 in the QCPU User's Manual (Multiple CPU System)
Redundant power supply system	To check the status of the power supply module in a redundant power supply system by using SM1780 to SM1783/SD1780 to SD1783 or with the system monitor screen, the Universal model QCPU whose serial number (first 5 digits) is "10042" or later must be used.	Check the status of the power supply module with the LED on the front of the module when using the Universal model QCPU whose serial number (first 5 digits) is "10041" or earlier. (In a redundant power supply system, the status of the power supply module cannot be stored in SM1780 to SM1783/SD1780 to SD1783. The status cannot be displayed on the system monitor screen either.)	Section 7.1 in the QCPU User's Manual (Hardware Design, Maintenance and Inspection)
MELSECNET/H	The simple dual-structured network function is not supported.	---	Section 7.7 in the Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
MELSECNET/H, CC-Link IE Controller Network	Interlink transmission timing differs.	Add a handshake program to the send side and receive side so that the module does not receive data while sending data.	Section 6.2 in the Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network), Section 4.1 in the MELSEC-Q CC-Link IE Controller Network Reference Manual

(2) Program

Item	Precautions	Replacement method	Reference
Language and instruction	Some instructions are not supported.	Replace the instructions not supported in the Universal model QCPU are described in Page 519, Appendix 5.3.	Page 519, Appendix 5.3
Floating-point operation	The Universal model QCPU performs program operations of floating-point data in single precision.	Instructions for floating-point double-precision operations are added for the Universal model QCPU. Replace the instructions if floating-point double-precision operations are required, as described in Page 536, Appendix 5.4.1.	Page 498, Appendix 4.4, Page 536, Appendix 5.4.1, Page 543, Appendix 5.4.2
	When using the floating-point data comparison instructions, LDE□, ANDE□, ORE□, LDED□, ANDED□, and ORED□, if the comparison source data are -0, nonnumeric, unnormalized number, or ±∞ , "OPERATION ERROR" (error code: 4101) is detected.*2 (□ indicates one of the following; =,<>,<=,>=,<,>.)	When the floating-point data comparison instructions are used, modify the program as described in Page 543, Appendix 5.4.2.	
Device range check at index modification	When a device number exceeds a setting range due to index modification, "OPERATION ERROR" (error code: 4101) is detected.	Deselect the "Check Device Range at Indexing" checkbox in the PLC RAS tab of the PLC parameter dialog box so that checking is not performed.	Page 195, Section 3.17 Page 547, Appendix 5.4.3
Program execution type	Low-speed execution type programs are not supported.	Use scan execution type programs or fixed scan execution type programs.	Page 88, Section 2.10
	A program execution type cannot be changed by remote operation. For the QnUDVCPU and QnUDPVCPU whose serial number (first five digits) is "18112" or later, however, a program execution type can be changed by remote operation when it is the scan execution type or stand-by type.	Use instructions for switching program execution types, such as PSTOP, POFF, and PSCAN.	Page 102, Section 2.10.5
Latch setting	If latch ranges of internal user devices are specified, the processing time is added in proportion to the device points set to be latched. (For example, if 8K points are latched for the latch relay (L), the processing time of 28.6μs is required.)	The latch function of the Universal model QCPU is enhanced. (1) Large-capacity file register (R, ZR) (2) Writing/reading device data to the standard ROM (SP.DEVST and S(P).DEVLD instructions) (3) Latch range specification of internal devices (4) "Time Setting" specification in the latch interval setting parameter*3 Change the latch method to the one described above according to the application.	Page 122, Section 3.3, Page 124, Section 3.3 (5) (b), Page 551, Appendix 5.4.4
Interrupt program	The interrupt pointer (I49) for the high-speed interrupt function is not supported.*2	Consider the use of interrupt pointers for fixed scan interrupt (I28 to I31).	Page 180, Section 3.13.2
	Interrupt counter is not supported.	Check the numbers of executions for interrupt programs on the Interrupt program monitor list screen.	
	The interrupt pointers (I32 to I40) due to an error are not supported.	---	Page 412, Section 4.11
SCJ instruction	When the SCJ instruction is used in the Universal model QCPU, the AND SM400 (or NOP instruction) needs to be inserted immediately before the SCJ instruction.	Insert the AND SM400 (or NOP instruction) immediately before the SCJ instruction when the SCJ instruction is used.	Section 6.5 in the MELSEC-Q/L Programming Manual (Common Instruction)
ZPUSH instruction	The number of index registers is increased to 20 for the Universal model QCPU. The area for saving the data in the index register with the ZPUSH instruction is increased as well.	Increase the save areas used for the ZPUSH instruction as needed.	Section 7.19 in the MELSEC-Q/L Programming Manual (Common Instruction)

Item	Precautions	Replacement method	Reference
File usability setting for each program	The following file usability setting for each program is not available.*1 • File register • Initial device value • Comment	When file usability is set, modify the program as described in Page 553, Appendix 5.4.5.	Page 88, Section 2.10, Page 553, Appendix 5.4.5
I/O refresh setting for each program	I/O refresh setting for each program is not available.	Use the RFS instruction if I/O refresh setting for each program is required.	MELSEC-Q/L Programming Manual (Common Instruction)
SM/SD	Usage of a part of the special relay and special register is different.	Replace the corresponding special relay and special register as described in Page 563, Appendix 5.5.	Page 563, Appendix 5.5
	A series-compatible special relay and special register are not supported. (SM1000 to SM1255/SD1000 to SD1255)	By using a programming tool, A series-compatible special relay and special register can be replaced with the Universal model QCPU-compatible special relay and special register. Note, however, that the ones which are not compatible with the Universal model QCPU are replaced with SM1255 and SD1255. Modify programs as needed.	QCPU User's Manual (Hardware Design, Maintenance and Inspection)
Processing time	Scan time and each processing time are different.	Modify programs as needed, checking the processing timing.	---
Number of steps	The number of steps increases by one*4 when: • Index modification is performed. • A leading or trailing edge instruction is used. • Bit devices are used as word data by specifying digits using K1, K2, K3, K5, K6, or K7, or by specifying a device number of other than multiples of 16.	If index modifications mentioned on the left are frequently used in the program, the program size may exceed the storage capacity of the replaced CPU module. After the program controller type is changed, check the program size using the confirm memory size function. If the program size exceeds the storage capacity, take the following actions or change the CPU module to that with larger program memory. • Move parameters and device comments to the standard ROM. • Reduce the reserved area for online change. • Use the file register, extended data register, and extended link register within 64K words because the number of steps decreases by one when used in that way.	MELSEC-Q/L Programming Manual (Common Instruction)

*1 Even the local device file usability setting is not available for the Q02UCPU, Q03UDCPU, Q04UDHCPU, and Q06UDHCPU if the serial number (first five digits) is "10011" or earlier.

*2 This will not apply when the High Performance model QCPU is replaced with the High-speed Universal model QCPU and Universal model Process CPU.

*3 Only the High-speed Universal model QCPU and Universal model Process CPU support this setting.

*4 This will apply only when the High Performance model QCPU is replaced with the High-speed Universal model QCPU and Universal model Process CPU.

(3) Drive and file

Item	Precautions	Replacement method	Reference
Boot file setting	Files in the standard ROM cannot be booted to the program memory.	Since the Universal model QCPU holds the data in the program memory even when the battery voltage drops, the boot file setting is not necessary. Move files with the boot setting (from the standard ROM to the program memory) to the program memory.	Page 104, Section 2.11, Page 556, Appendix 5.4.6
	Booting operation is different.	Replacement method when the parameter-valid drive and the boot file setting are set in the High Performance model QCPU is described in Page 556, Appendix 5.4.6.	
	A memory card (SRAM card, ATA card, or Flash card) cannot be specified as a transfer source.*1	Specify an SD memory card as a transfer source.	

Item	Precautions	Replacement method	Reference
Automatic all data write from memory card to standard ROM	The setting method of this function is different.	In the Boot file tab of the PLC parameter dialog box, select "standard ROM" for the transfer destination. Note, however, that the transfer destination of "program" is fixed to "program memory". (Setting by DIP switches is not necessary.)	Page 104, Section 2.11
Device comment	A device comment file cannot be stored in an SRAM card.*1	Store the file in the standard RAM.	---
	A device comment file cannot be stored in an ATA card nor Flash card.*1	Store the file in an SD memory card.	---
Initial device value	An initial device value file cannot be stored in an SRAM card.*1	Store the file in the standard RAM or standard ROM.	Page 247, Section 3.25
	An initial device value file cannot be stored in an ATA card nor Flash card.*1	Store the file in an SD memory card.	
Local device	A local device file cannot be stored in an SRAM card.*1	• Store the file in the standard RAM. • If the size of the local device file exceeds the standard RAM capacity, consider the use of an extended SRAM cassette.	Page 422, Section 6.2
File register	A file register file cannot be stored in an SRAM card.*1	• Store the file in the standard RAM. • If the size of the file register file exceeds the standard RAM capacity, consider the use of an extended SRAM cassette.	Page 393, Section 4.7.1
	A file register file cannot be stored in a Flash card. (Sequence programs only can read file register data in a Flash card.)*1	Use the initial device value file in an SD memory card or the FREAD/FWRITE instructions.	Page 247, Section 3.25, MELSEC-Q/L Programming Manual (Common Instruction)
Sampling trace	A sampling trace file cannot be stored in an SRAM card.*1	• Store the file in the standard RAM. • If the size of the sampling trace file exceeds the standard RAM capacity, consider the use of an extended SRAM cassette.	Page 184, Section 3.14 (2)
CPU module change function with memory card	A memory card cannot be specified as a backup destination or restoration source.	Specify an SD memory card as a backup destination or restoration source.	Page 260, Section 3.31

*1 This applies when the High Performance model QCPU is replaced with the High-speed Universal model QCPU and Universal model Process CPU.

(4) External communication

Item	Precautions	Replacement method	Reference
Module service interval time read	The module service interval time cannot be read.	---	Page 241, Section 3.24.1
MC protocol	To access CPU modules by using A-compatible 1C frame and A-compatible 1E frame, the Q10UDHCPU, Q20UDHCPU, Built-in Ethernet port QCPU, or the modules (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU) whose serial numbers (first 5 digits) are "10102" or later must be used.	Use the frame types below when using the modules (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU) whose serial numbers (first 5 digits) are "10101" or earlier. • QnA-compatible 2C/3C/4C frame • QnA-compatible 3E frame • 4E frame	MELSEC Communication Protocol Reference Manual
	The following commands cannot specify monitoring conditions. • Randomly reading data in units of word (Command: 0403) • Device memory monitoring (Command: 0801) The applicable frame types are as follows: • QnA-compatible 3C/4C frame • QnA-compatible 3E frame • 4E frame	---	

(5) Diagnostic function

Item	Precautions	Replacement method	Reference
Error history	Error history data cannot be stored in the memory card.	The Universal model QCPU can store history data by the number of storable history data in a memory card (100) to the system memory.	Page 206, Section 3.18
LED indication priority setting	LED indication priority cannot be set. Only LED indication setting at error occurrence is supported.	---	Page 223, Section 3.20.2

(6) Debugging

Item	Precautions	Replacement method	Reference
Monitor condition setting	To use the monitor condition setting, the Q10UDHCPU, Q20UDHCPU, Built-in Ethernet port QCPU, or the modules (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU) whose serial numbers (first 5 digits) are "10042" or later must be used.	Check device data under the specified monitoring condition by using the sampling trace function when using the Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU. With this function, changes of the specified device data can be recorded at the following timings: • at execution of the specified step • on the rising/falling edge of bit devices • when the value of word devices coincide with the setting value • at every specified time (settable range: 1 to 5000ms)	Page 146, Section 3.11.1, Page 184, Section 3.14
Scan time measurement	Time required for executing a part of the program cannot be measured using the scan time measurement function.*1	Calculate the time using instruction processing time described in the manual.	• Page 181, Section 3.13.3 • Appendix 1 in the MELSEC-Q/L Programming Manual (Common Instruction)
External input/output forced on/off	To use the external input/output forced on/off function, the Q10UDHCPU, Q20UDHCPU, Built-in Ethernet port QCPU, or the modules (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU) whose serial numbers (first 5 digits) are "10042" or later must be used.*2	Replace the function by using the programs described in Page 559, Appendix 5.4.7 when using the modules (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU) whose serial numbers (first 5 digits) are "10041" or earlier. Note, however, that replacement method described does not apply in the following cases: • Input and output targeted for forced on/off are referred to or changed using the direct input device (DX) and direct output device (DY). • Input and output targeted for forced on/off are referred to or changed within an interrupt program.	Page 154, Section 3.11.3, Page 559, Appendix 5.4.7

* 1 Scan time of each program can be checked on the Program monitor list screen.

* 2 Device test can be performed when the modules (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU) whose serial numbers (first five digits) are "10041" or earlier are used.

(7) Switch on the front of the CPU module

Item	Precautions	Replacement method	Reference
Switch on the front of the CPU module	The operation method with the RESET/RUN/STOP switch is modified.	The RESET/STOP/RUN switch of the Universal model QCPU can be used for the reset operation of the CPU module and switching between the STOP and RUN status.	Section 6.1.3 in the QCPU User's Manual (Hardware Design, Maintenance and Inspection)
	Latch data cannot be cleared by the switch.	Perform either of the following operations to clear latch data. • Remote latch clear using a programming tool • Latch clear by using the special relay and special register areas*1	Page 137, Section 3.6.4
	The system protect cannot be set by the switch.	Data in the files can be protected by setting a password for each file. Password for each file can be registered with a programming tool.	Page 207, Section 3.19
	The parameter-valid drive setting is not necessary.	The Universal model QCPU automatically determines the parameter-valid drive. Change the setting as described in Page 556, Appendix 5.4.6 when the parameter-valid drive is set to other than the program memory in the High Performance model QCPU.	Page 42, Section 2.1.2, Page 556, Appendix 5.4.6

*1 Only the High-speed Universal model QCPU whose serial number (first five digits) is "15043" or later and the Universal model Process CPU whose serial number (first five digits) is "15072" or later support this type of latch clear operation.

(8) SFC

Item	Precautions	Replacement method	Reference
Step transition monitoring timer	The step transition monitoring timer is not supported.	Change the program as described in Appendix 3.1 in the manual in the Reference column.	Section 4.6 and Appendix 3.1 in the MELSEC-Q/L/QnA Programming Manual (SFC)
SFC operation mode setting	The periodic execution block setting is not supported.	Change the program as described in Appendix 3.2 in the manual in the Reference column.	Section 4.7.4 and Appendix 3.2 in the MELSEC-Q/L/QnA Programming Manual (SFC)
	To select an operation mode at double block START, the Universal model QCPU whose serial number (first 5 digits) is "12052" or later must be used.	An operation mode at double block START is fixed to "WAIT" when the Universal model QCPU whose serial number (first 5 digits) is "12051" or earlier is used.	Section 4.7.5 in the MELSEC-Q/L/QnA Programming Manual (SFC)
	An operation mode at transition to active step cannot be selected. (Fixed to "TRANSFER".)	---	Section 4.7.6 in the MELSEC-Q/L/QnA Programming Manual (SFC)
SFC program for program execution management	SFC programs for program execution management are not supported.	---	Section 5.2.3 in the MELSEC-Q/L/QnA Programming Manual (SFC)
SFC control instruction	Some SFC control instructions are not supported.	SFC control instructions not supported in the Universal model QCPU and replacing methods are described in Page 519, Appendix 5.3.	• Page 519, Appendix 5.3 • Section 4.4 in the MELSEC-Q/L/QnA Programming Manual (SFC)
SFC comment readout instruction	To use the following SFC comment readout instructions, the Universal model QCPU whose serial number (first 5 digits) is "12052" or later must be used. • S(P).SFCSCOMR (SFC step comment readout instruction) • S(P).SFCTCOMR (SFC transition condition comment readout instruction)	---	Section 4.8 in the MELSEC-Q/L/QnA Programming Manual (SFC)
Method of SFC program change	SFC program files cannot be written to the running CPU module. (Programs in SFC Figure can be changed online.)	• Write program data to the CPU module after changing the Universal model QCPU status to STOP. • An inactive block in an SFC program can be changed by online change of inactive block.*1	Section 6.6 in the MELSEC-Q/L/QnA Programming Manual (SFC)

*1 This operation is available for the Universal model QCPU other than the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU and whose serial number (first five digits) is "12052" or later.

Appendix 5.2 Applicable devices and software

(1) Products need to be replaced for the compatibility with the Universal model QCPU

The following tables show products need to be replaced for the compatibility with the Universal model QCPU. (As for products not listed in the tables below, replacement is not required.)

(a) Communication module

Product	Model	Serial number (first five digits) of the product compatible with the Universal model QCPU*2				
		Used with Q02U/Q03UD/ Q04UDH/ Q06UDHCPU	Used with Q13UDH/ Q26UDHCPU	Used with Q00UJ/Q00U/ Q01U/Q10UDH/ Q20UDHCPU, or QnUDE(H)CPU	Used with High-speed Universal model QCPU	Used with Universal model Process CPU
Web server module*1	• QJ71WS96	"09042" or later	"10011" or later	"10012" or later	"14122" or later	"14122" or later
MES interface module	• QJ71MES96	"09042" or later	"10011" or later	"10012" or later	"14122" or later	"14122" or later
High speed data logger module	• QD81DL96	No restrictions	No restrictions	No restrictions	"14122" or later	"14122" or later

*1 The Universal model QCPU does not operate normally when the Web server module on which GX RemoteService-I is installed is used.

*2 The Universal model QCPU does not operate normally when a product not compatible with the Universal model QCPU is used.

(b) PC interface board

Product		Model	Dedicated software package version compatible with the Universal model QCPU*1				
			Used with Q02U/Q03UD / Q04UDH/ Q06UDHCPU	Used with Q13UDH/ Q26UDHCPU	Used with Q00UJ/Q00U/ Q01U/Q10UDH/ Q20UDHCPU, or QnUDE(H)CPU	Used with High-speed Universal model QCPU	Used with Universal model Process CPU
CC-Link IE Field Network interface board		• Q81BD-J71GF11-T2	No restrictions	No restrictions	No restrictions	1.03D or later	1.12N or later
CC-Link IE Controller Network interface board		• Q81BD-J71GP21-SX • Q81BD-J71GP21S-SX • Q80BD-J71GP21-SX • Q80BD-J71GP21S-SX	No restrictions	1.03D or later	1.06G or later	1.15R or later	1.15R or later
MELSECNET/H interface board	SI/QSI/H-PCF optical cable	• Q80BD-J71LP21-25 • Q80BD-J71LP21S-25	15R or later	18U or later	20W or later	25B or later	1.15R or later
		• Q81BD-J71LP21-25	19V or later	19V or later			
	GI optical cable	• Q80BD-J71LP21G	15R or later	18U or later			
	Coaxial cable	• Q80BD-J71BR11					
CC-Link system master/local interface board		• Q80BD-J61BT11N	1.02C or later	1.05F or later	1.07H or later	1.12N or later	1.12N or later
		• Q81BD-J61BT11	1.06G or later	1.06G or later			

*1 No restrictions on the board itself.

(c) GOT

Product	Model	GT Designer2 OS version compatible with the Universal model QCPU*1				
		Used with Q00UJ/Q00U/ Q01U/Q10UDH/ Q20UDHCPU	Used with Q02U/Q03UD/ Q04UDH/ Q06UDHCPU	Used with Q13UDH/ Q26UDHCPU	Used with Q03UDE/ Q04UDEH/ Q06UDEH/ Q13UDEH/ Q26UDEHCPU	Used with Q10UDEH/ Q20UDEHCPU
GOT1000	GT16□-□	---	---	---	---	---
	GT15□-□	2.91V or later	2.60N or later	2.76E or later	2.81K or later	2.91V or later
	GT14□-□	---	---	---	---	---
	GT11□-□	2.91V or later	2.60N or later	2.76E or later	2.81K or later	2.91V or later
	GT10□-□	2.91V or later	2.76E or later	2.76E or later	---	---

Product	Model	GT Works3 OS version compatible with the Universal model QCPU*1	
		High-speed Universal model QCPU	Universal model Process CPU
GOT1000	GT16□-□	1.64S or later	1.73B or later
	GT15□-□	1.64S or later	1.73B or later
	GT14□-□	1.64S or later	1.73B or later
	GT11□-□	1.64S or later	1.73B or later
	GT10□-□	1.64S or later	1.73B or later

*1 No restrictions on GOT itself.

(d) Network module and serial communication module

Product	Model	Module version compatible with the Universal model QCPU		
		Used with Q00UJ/Q00U/Q01U/Q02U/ Q03UD/Q04UDH/Q06UDH/ Q10UDH/Q13UDH/Q20UDH/ Q26UDHCPU	Used with QnUDE(H)CPU	Used with QnUDV/QnUDPVCPU
MELSECNET/H model	• QJ71LP21-25 • QJ71LP21S-25 • QJ71LP21G • QJ71LP21GE • QJ71BR11	No restrictions	Some restrictions depending on use conditions*1	
Serial communication module	• QJ71C24N • QJ71C24N-R2 • QJ71C24N-R4		Serial number (first five digits) "10042" or later	No restrictions

*1 The serial number (first five digits) of the MELSECNET/H module must be "10042" or later if all conditions 1) to 4) described below are satisfied.

1) A multiple CPU system including Built-in Ethernet port QCPU is configured.
2) A programming tool or GOT is connected to an Ethernet port of the Built-in Ethernet port QCPU.
3) A programming tool or GOT accesses the CPU module on another station via the MELSECNET/H module controlled by another CPU.
4) The access target on another station is A/QnA series CPU module.

(2) CPU modules that can configure a multiple CPU system with the Universal model QCPU

CPU modules that can configure a multiple CPU system with the Universal model QCPU are shown below.

(a) For the QnUD(H)CPU or Built-in Ethernet port QCPU

<table>
<tr><th rowspan="2">CPU module</th><th rowspan="2">Model</th><th colspan="5">Applicable version</th><th rowspan="2">Restrictions</th></tr>
<tr><th>Configured with Q03UD/ Q04UDH/ Q06UDHCPU</th><th>Configured with Q13UDH/ Q26UDH, Q03UDE/ Q04UDEH, Q06UDEH/ Q13UDEH/ Q26UDEHCPU</th><th>Configured with Q10UDH/ Q20UDH/ Q10UDEH/ Q20UDEHCPU</th><th>Configured with High-speed Universal model QCPU</th><th>Configured with Universal model Process CPU</th></tr>
<tr><td>Motion CPU</td><td>• Q172DCPU
• Q173DCPU
• Q172DCPU-S1
• Q173DCPU-S1
• Q172DSCPU
• Q173DSCPU</td><td colspan="5">No restrictions</td><td>Use only the multiple CPU high-speed main base unit (Q3□DB) as a main base unit.</td></tr>
<tr><td>PC CPU module</td><td>• PPC-CPU852(MS)</td><td>Driver S/W (PPC-DRV-02) version 1.01 or later</td><td>Driver S/W (PPC-DRV-02) version 1.02 or later</td><td>Driver S/W (PPC-DRV-02) version 1.03 or later</td><td colspan="2">N/A</td><td>---</td></tr>
<tr><td rowspan="4">C Controller module</td><td>• Q06CCPU-V
• Q06CCPU-V-B</td><td>No restrictions</td><td>Serial number (first five digits) "10012" or later</td><td>Serial number (first five digits) "10102" or later</td><td colspan="2">N/A</td><td>---</td></tr>
<tr><td>• Q12DCCPU-V</td><td colspan="3">No restrictions</td><td>Serial number (first five digits) "14122" or later</td><td>N/A</td><td>---</td></tr>
<tr><td>• Q24DHCCPU-V</td><td colspan="3">No restrictions</td><td>Serial number (first five digits) "14122" or later</td><td>Serial number (first five digits) "15051" or later</td><td>---</td></tr>
<tr><td>• Q24DHCCPU-LS</td><td colspan="5">No restrictions</td><td>---</td></tr>
<tr><td>High Performance model QCPU</td><td>• Q02CPU
• Q02HCPU
• Q06HCPU
• Q12HCPU
• Q25HCPU</td><td colspan="5">Function version B or later</td><td>---</td></tr>
<tr><td>Process CPU</td><td>• Q02PHCPU
• Q06PHCPU
• Q12PHCPU
• Q25PHCPU</td><td colspan="5">No restrictions</td><td>---</td></tr>
</table>

(b) For the Q00UCPU, Q01UCPU, or Q02UCPU

CPU module	Model	Applicable version		Restrictions
		Configured with Q00U/Q01UCPU	Configured with Q02UCPU	
Motion CPU	• Q172CPUN(-T) • Q173CPUN(-T) • Q172HCPU(-T) • Q173HCPU(-T)	No restrictions		The multiple CPU high-speed main base unit (Q3□DB) cannot be used as a main base unit.
PC CPU module	• PPC-CPU852(MS)	Driver S/W (PPC-DRV-02) version 1.03 or later	Driver S/W (PPC-DRV-02) version 1.01 or later	---
C Controller module	• Q06CCPU-V • Q06CCPU-V-B	Serial number (first five digits) "10102" or later	No restrictions	---
	• Q12DCCPU-V • Q24DHCCPU-V • Q24DHCCPU-LS	No restrictions		---

(3) Software need to be upgraded for the compatibility with the Universal model QCPU

The following table shows software need to be upgraded for the communication with the Universal model QCPU. (As for software not listed in the table below, version upgrade is not required.)

Product	Model	Version compatible with the Universal model QCPU			
		Used with Q02U/ Q03UD/Q04UDH/ Q06UDHCPU	Used with Q13UDH/ Q26UDHCPU	Used with Q03UDE/Q04UDEH/ Q06UDEH/Q13UDEH /Q26UDEHCPU	Used with Q00UJ/ Q00U/Q01U/Q10UDH /Q20UDH/Q10UDEH/ Q20UDEHCPU
GX Developer	SW8D5C-GPPW-E	Version 8.48A or later	Version 8.62Q or later	Version 8.68W or later	Version 8.78G or later
GX Configurator-AD	SW2D5C-QADU-E	Version 2.05F or later*1	Version 2.05F or later*2	Version 2.05F or later*3	Version 2.05F or later*4
GX Configurator-DA	SW2D5C-QDAU-E	Version 2.06G or later*1	Version 2.06G or later*2	Version 2.06G or later*3	Version 2.06G or later*4
GX Configurator-SC	SW2D5C-QSCU-E	Version 2.12N or later*1	Version 2.12N or later*2	Version 2.17T or later*3	Version 2.17T or later*4
GX Configurator-CT	SW0D5C-QCTU-E	Version 1.25AB or later*1	Version 1.25AB or later*2	Version 1.25AB or later*3	Version 1.25AB or later*4
GX Configurator-TI	SW1D5C-QTIU-E	Version 1.24AA or later*1	Version 1.24AA or later*2	Version 1.24AA or later*3	Version 1.24AA or later*4
GX Configurator-TC	SW0D5C-QTCU-E	Version 1.23Z or later*1	Version 1.23Z or later*2	Version 1.23Z or later*3	Version 1.23Z or later*4
GX Configurator-FL	SW0D5C-QFLU-E	Version 1.23Z or later*1	Version 1.23Z or later*2	Version 1.23Z or later*3	Version 1.23Z or later*4
GX Configurator-QP	SW2D5C-QD75P-E	Version 2.25B or later	Version 2.29F or later	Version 2.30G or later*5	Version 2.32J or later
GX Configurator-PT	SW1D5C-QPTU-E	Version 1.23Z or later*1	Version 1.23Z or later*2	Version 1.23Z or later*3	Version 1.23Z or later*4
GX Configurator-AS	SW1D5C-QASU-E	Version 1.21X or later*1	Version 1.21X or later*2	Version 1.21X or later*3	Version 1.21X or later*4
GX Configurator-MB	SW1D5C-QMBU-E	Version 1.08J or later*1	Version 1.08J or later*2	Version 1.08J or later*3	Version 1.08J or later*4
GX Configurator-DN	SW1D5C-QDNU-E	Version 1.23Z or later*1	Version 1.23Z or later*2	Version 1.24AA or later*3	Version 1.24AA or later*4
GX Configurator-DP*6	SW7D5C-PROFID-E	Version 7.02C or later*7	Version 7.03D or later	Version 7.03D or later	Version 7.04E or later
MX Component	SW3D5C-ACT-E	Version 3.09K or later	Version 3.10L or later	Version 3.11M or later	Version 3.12N or later
GX Simulator	SW7D5C-LLT-E	Version 7.23Z or later*4	Version 7.23Z or later*4	Version 7.23Z or later*4	Version 7.23Z or later*4

*1 The software can be used by installing GX Developer Version 8.48A or later.
*2 The software can be used by installing GX Developer Version 8.62Q or later.
*3 The software can be used by installing GX Developer Version 8.68W or later.
*4 The software can be used by installing GX Developer Version 8.78G or later.
*5 GX Configurator-QP Version 2.29F can be used when connected via USB.
*6 When using the GX Configurator with the Q50UDEH/Q100UDEHCPU, use the Version 7.07H or later.
*7 When using the GX Configurator with the Q02UCPU, use the Version 7.03D or later.

(4) Software not supported in the Universal model QCPU

The following table shows software not supported in the Universal model QCPU.

Product	Model
GX Explorer	SW□D5C-EXP-E
GX Converter	SW□D5C-CNVW-E

Appendix 5.3 Instructions

Appendix 5.3.1 Instructions not supported in the Universal model QCPU and replacing methods

The Universal model QCPU does not support instructions listed in the following tables. Instructions need to be replaced using replacing methods described in the tables. (If no instruction in the list is used, replacement is not required.)

Symbol	Instruction	Replacing method	Reference
IX	Index modification of entire ladder	Instructions can be replaced using a replacement program.	Page 522, Appendix 5.3.3 (1)
IXEND			
IXDEV	Modification value specification in index modification of entire ladder	Change the program so that the device offset values specified by the IXSET instruction are directly set to the index modification table using the MOV instruction.	Page 524, Appendix 5.3.3 (2)
IXSET			
PR	Print ASCII code instruction	• It is recommended to use GOT as an ASCII code display device. ASCII codes stored in devices are directly displayed as characters on GOT. • Instructions can be replaced using a replacement program.	Page 526, Appendix 5.3.3 (3)
PRC	Print comment instruction	• It is recommended to use GOT as an ASCII code display device. Device comments can be displayed on GOT. • Comment data can be output to a display device in the replacement program of the PR instruction after reading data using the reading device comment data instruction (COMRD(P)).	
CHKST	Specific format failure check instruction	Instructions can be replaced using a replacement program.	Page 530, Appendix 5.3.3 (4)
CHK			
CHKCIR	Format change instruction for CHK instruction	Failure detection ladder patterns can be changed in a replacement program.	
CHKEND			
PLOW	Program low-speed execution registration instruction	• Use the PSCAN instruction instead of this instruction when low-speed execution type programs are replaced with scan execution type programs. • No instruction can be used if low-speed execution type programs are replaced with fixed scan execution type programs.	---
PCHK	Program execution status check instruction	Check a program execution status on the Program monitor list screen of programming tool. For details, refer to Page 180, Section 3.13.1 in this manual.	---
KEY	Numerical key input instruction	• It is recommended to use GOT as a numeral input device. • Instructions can be replaced using a replacement program.	Page 533, Appendix 5.3.3 (5)
PLOADP	Load program from memory card	Store all programs to be executed in the program memory. The Universal model QCPU can neither add programs to the program memory nor change them with other programs during RUN. If the capacity of the program memory is not enough, store parameters, device comments, and initial device values in the program memory into the standard ROM or a memory card instead.	---
PUNLOADP	Unload program from memory card		
PSWAPP	Load + Unload		

Symbol	Instruction	Replacing method
LD TRn	Forced transition check instruction	When the programmable controller type is changed, these instructions are converted into SM1255. Modify programs as needed.
AND TRn		
OR TRn		
LDI TRn		
ANDI TRn		
ORI TRn		
LD BLm\TRn		
AND BLm\TRn		
OR BLm\TRn		
LDI BLm\TRn		
ANDI BLm\TRn		
ORI BLm\TRn		
SCHG(D)	Active step change instruction	Refer to Appendix 3 "Restrictions on Basic Model QCPU, Universal Model QCPU, and LCPU and Alternative Methods" in the MELSEC-Q/L/QnA Programming Manual (SFC).
SET TRn	Transition control instruction	Refer to Appendix 3 "Restrictions on Basic Model QCPU, Universal Model QCPU, and LCPU and Alternative Methods" in the MELSEC-Q/L/QnA Programming Manual (SFC).
SET BLm\TRn		
RST TRn		
RST BLm\TRn		
BRSET(S)*1	Block switching instruction	When the programmable controller type is changed, these instructions are converted into SM1255. Modify programs as needed.

*1 Usable for the Universal model CPU whose serial number (first five digits) is "13102" or later.

Appendix 5.3.2 Replacing programs using multiple CPU transmission dedicated instructions

(1) Replacing the module with the QnUD(H)CPU or Built-in Ethernet port QCPU

The following table shows instructions need to be replaced and corresponding alternative instructions. For the specifications of each instruction, refer to the manuals for the Motion CPU.

Symbol	Instruction description	Symbol of alternative instruction
S(P).DDWR	Write other CPU device data into host CPU	D(P).DDWR
S(P).DDRD	Read other CPU device data into host CPU	D(P).DDRD
S(P).SFCS	Request of motion SFC program startup	D(P).SFCS
S(P).SVST	Request of servo program startup	D(P).SVST
S(P).CHGA	Current value change of halted axis/synchronized encoder/cam axis	D(P).CHGA
S(P).CHGV	Axis speed change during positioning and JOG operation	D(P).CHGV
S(P).CHGT	Torque control value change during operation and suspension in real mode	D(P).CHGT
S(P).GINT	Request of other CPU interrupt program startup	D(P).GINT

(2) Replacing the module with the Q00UCPU, Q01UCPU, or Q02UCPU

The Q00UCPU, Q01UCPU, and Q02UCPU support the same multiple CPU transmission dedicated instructions used in the Basic model QCPU.

The alternative instructions in the table in (1) are not available for Q00UCPU, Q01UCPU, and Q02UCPU.

Appendix 5.3.3 Program replacement examples

This section shows program replacement examples for the instructions of which replacement programs are available in Page 519, Appendix 5.3.1. (Skip this section if instructions listed in Page 519, Appendix 5.3.1 are not used.)

(1) Replacement example of the IX and IXEND instructions

Since index registers are saved using the ZPUSH instruction, a 23-word index register save area is required.

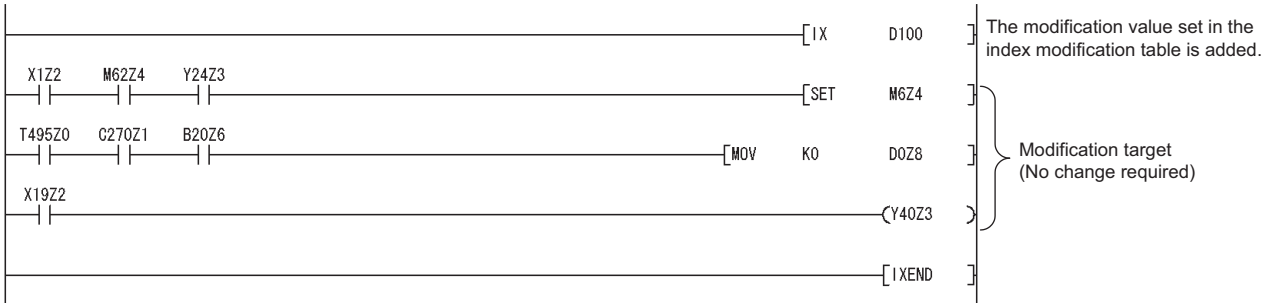
(a) Example of device assignment

Before replacement	
Application	**Device**
Index modification table	D100 to D115

After replacement	
Application	**Device**
Index modification table	D100 to D115
Index register save area	D200 to D222

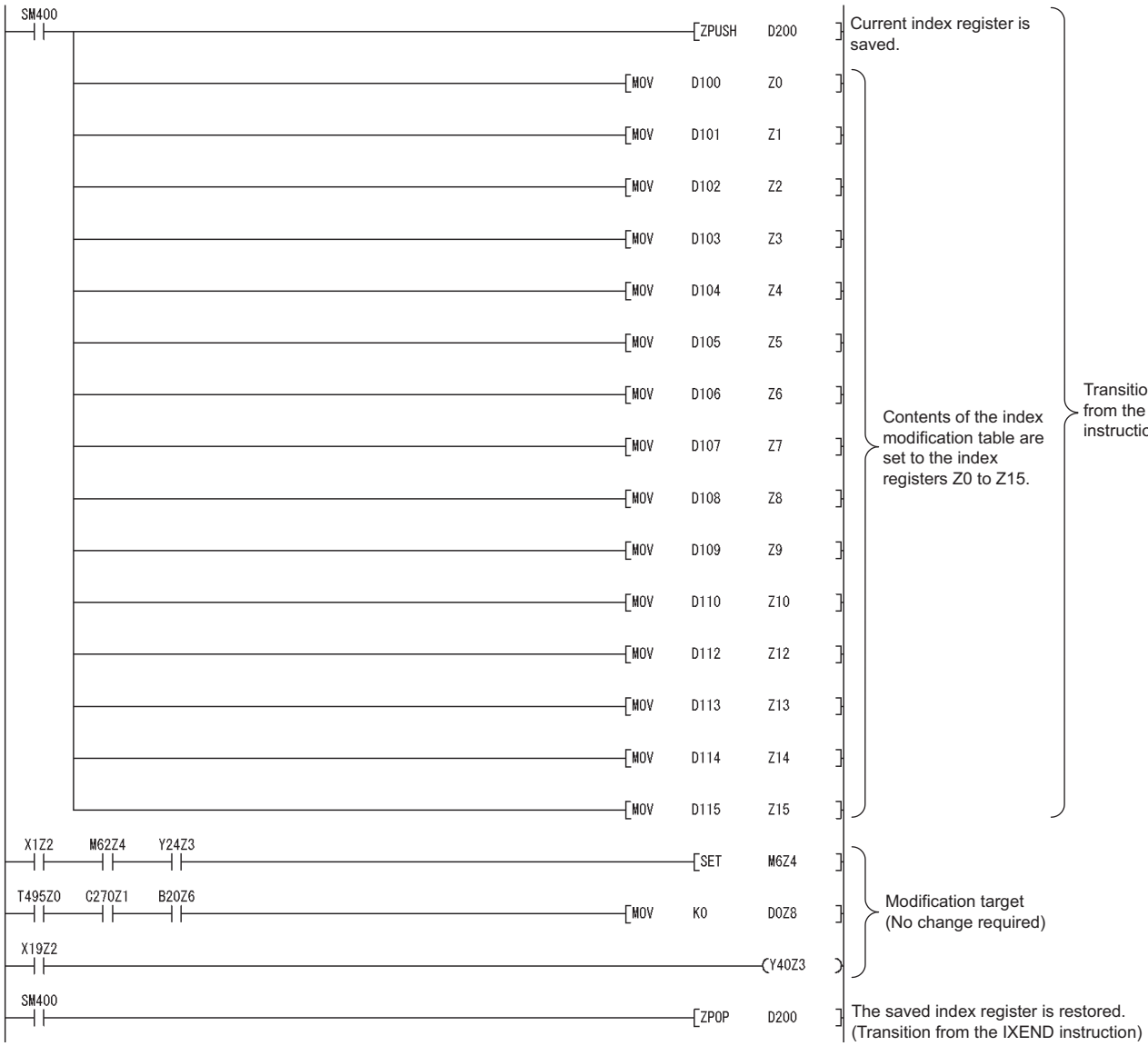
If the device numbers in the example above are used for other applications, assign unused device numbers instead.

(b) Program before replacement



(c) Program after replacement

- Replace the IX instruction with the ZPUSH instruction and the processing for setting the contents of index modification table to index registers.
- Replace the IXEND instruction with the ZPOP instruction.

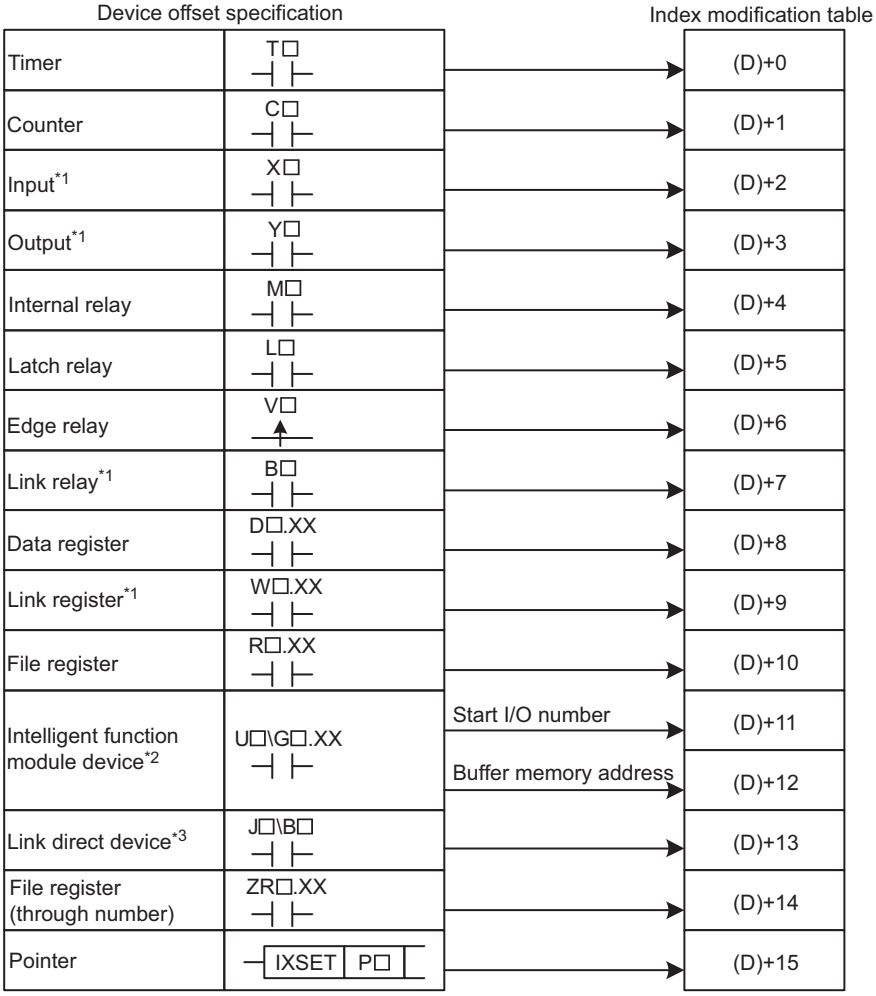


(2) Replacement example of the IXDEV and IXSET instructions

Change the program so that the device offset value specified by the contacts between the IXDEV and the IXSET instructions are directly set to the index modification table using the MOV instruction.

For the devices whose device offset value is not specified by the IXDEV and IXSET instructions, set the device offset value to 0 in the program after replacement.

The following figure shows how the device offset value is set in the program before and after replacement by the IXDEV and IXSET instructions.

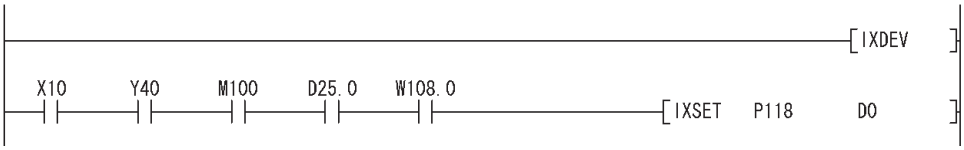


*1 Device numbers are represented in hexadecimal. Use hexadecimal constants (H□) when setting values in the index modification table.

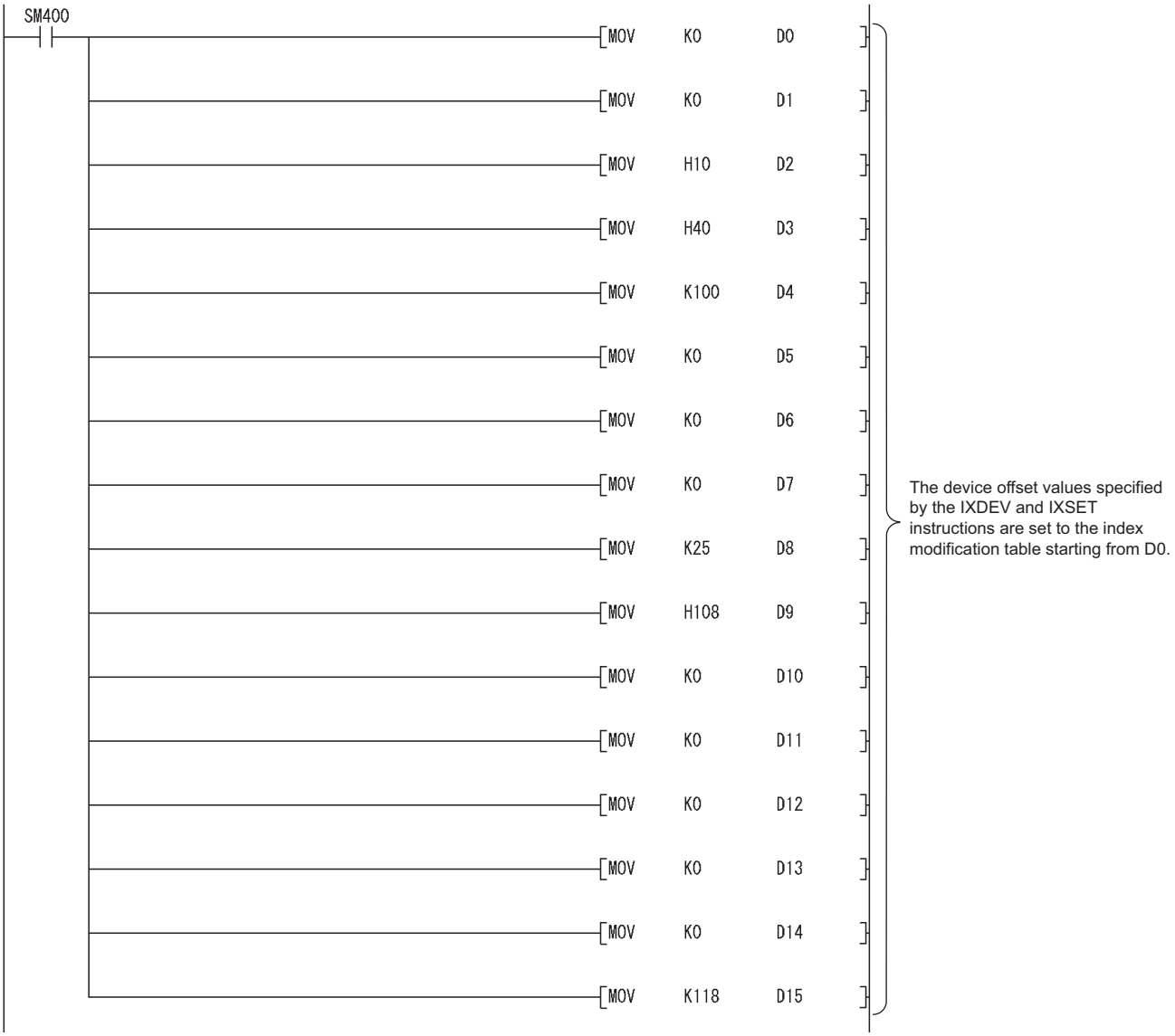
*2 Start I/O numbers (U□) are represented in hexadecimal. Use hexadecimal constants (H□) when setting values in the index modification table.

*3 Devices B, W, X, or Y can be specified following J□\. Set device numbers for B, W, X, and Y as device offset values of each device in the index modification table. For example, if "J10\Y220" is specified by the IXDEV and IXSET instructions, set "K10" in (D)+13 and "H220" in (D)+3 in the replacement program. ((D) indicates the start device in the index modification table.)

(a) Program before replacement



(b) Program after replacement



(3) Replacement example of the PR instruction

The number of output characters can be switched by the on/off status of SM701.

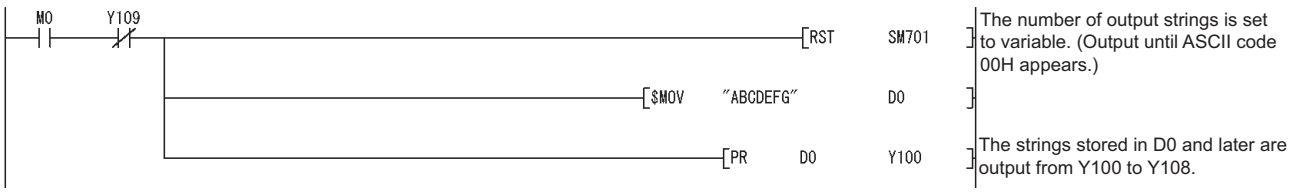
(a) Example of device assignment

Before replacement	
Application	**Device**
Output string	D0 to D3
ASCII code output signal	Y100 to Y107
Strobe signal	Y108
In-execution flag	Y109

After replacement	
Application	**Device**
Output string	D0 to D3
ASCII code output signal	Y100 to Y107
Strobe signal	Y108
In-execution flag	Y109
Output string storage address (BIN32)	D20 to D21
Output string storage address (BIN32) (Used for sub-routine programs and interrupt programs)	D200 to D201
Number of output characters	D202
Output module start Y number	D203
Character extraction position	D204
Number of extracted characters	D205
String output status value	D206
Result of string extraction by the MIDR instruction	D207
String output in-execution flag	M200
For index modification	Z0

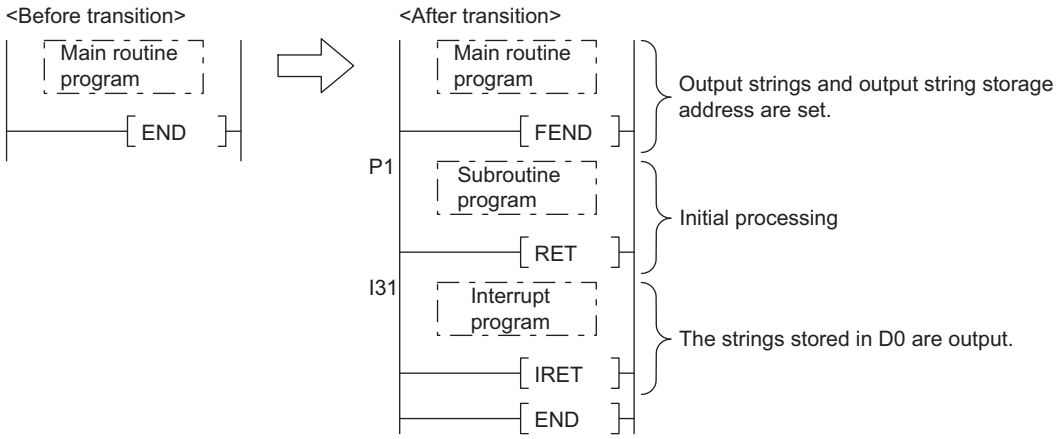
If the device numbers in the example above are used for other applications, assign unused device numbers instead.

(b) Program before replacement



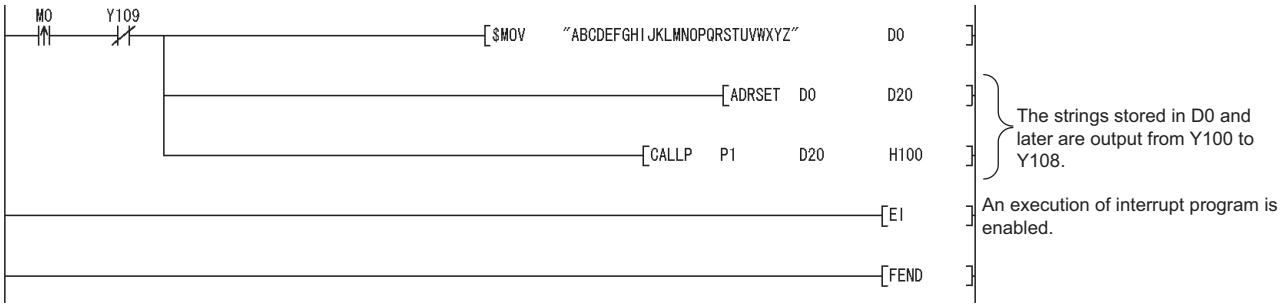
(c) Program after replacement

In the sequence program after replacement, three programs are required as shown below.



1. Main routine program

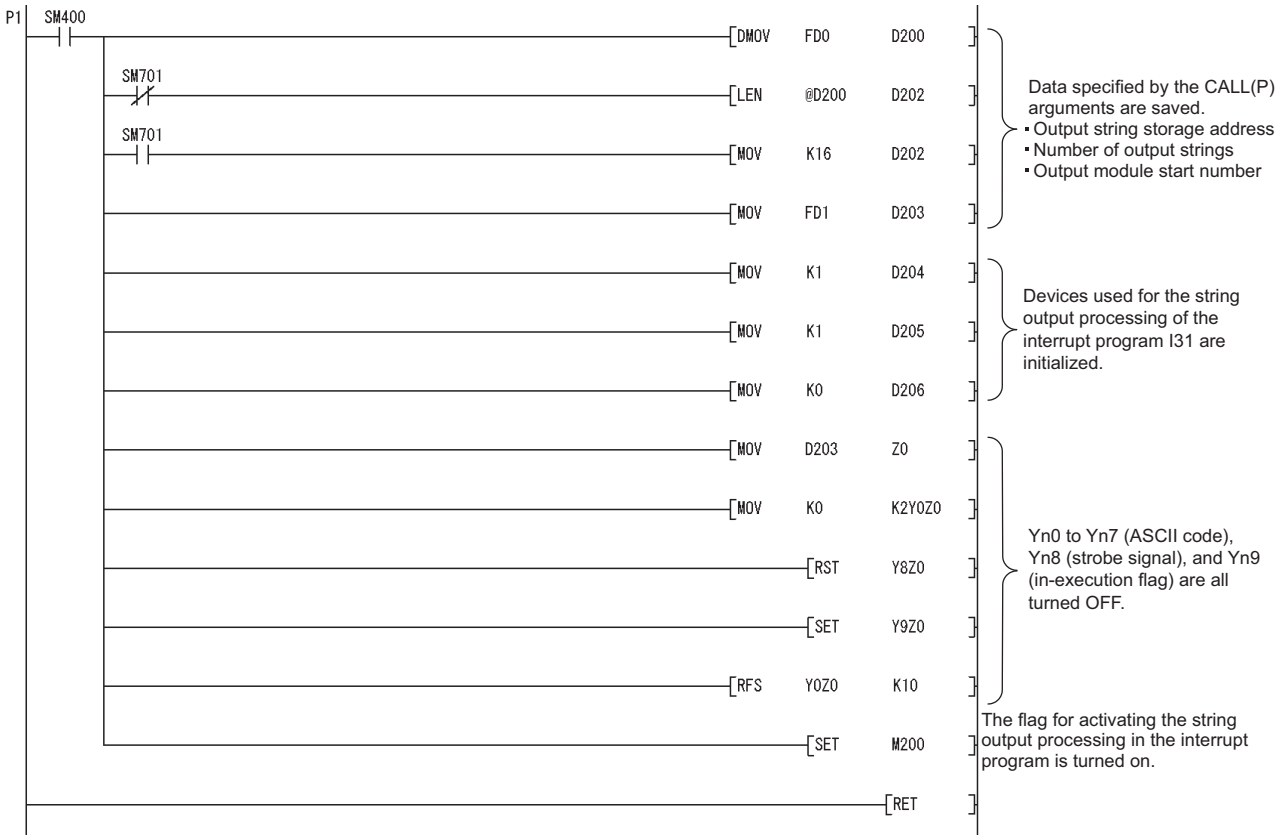
- Replace the PR instruction with the CALL instruction so that a subroutine program is called.
- Output string storage device ("D0" in the program below) cannot be specified directly with the CALL instruction. Use the ADRSET instruction to acquire the indirect address for the CALL instruction.
- Y device ("Y100" in the program before replacement shown in (b)) cannot be specified directly as output Y number with the CALL instruction. Specify the output Y number in integer.
- An interrupt program is used to output character codes via the output module. Enable the execution of interrupt programs using the EI instruction.



2. Subroutine program

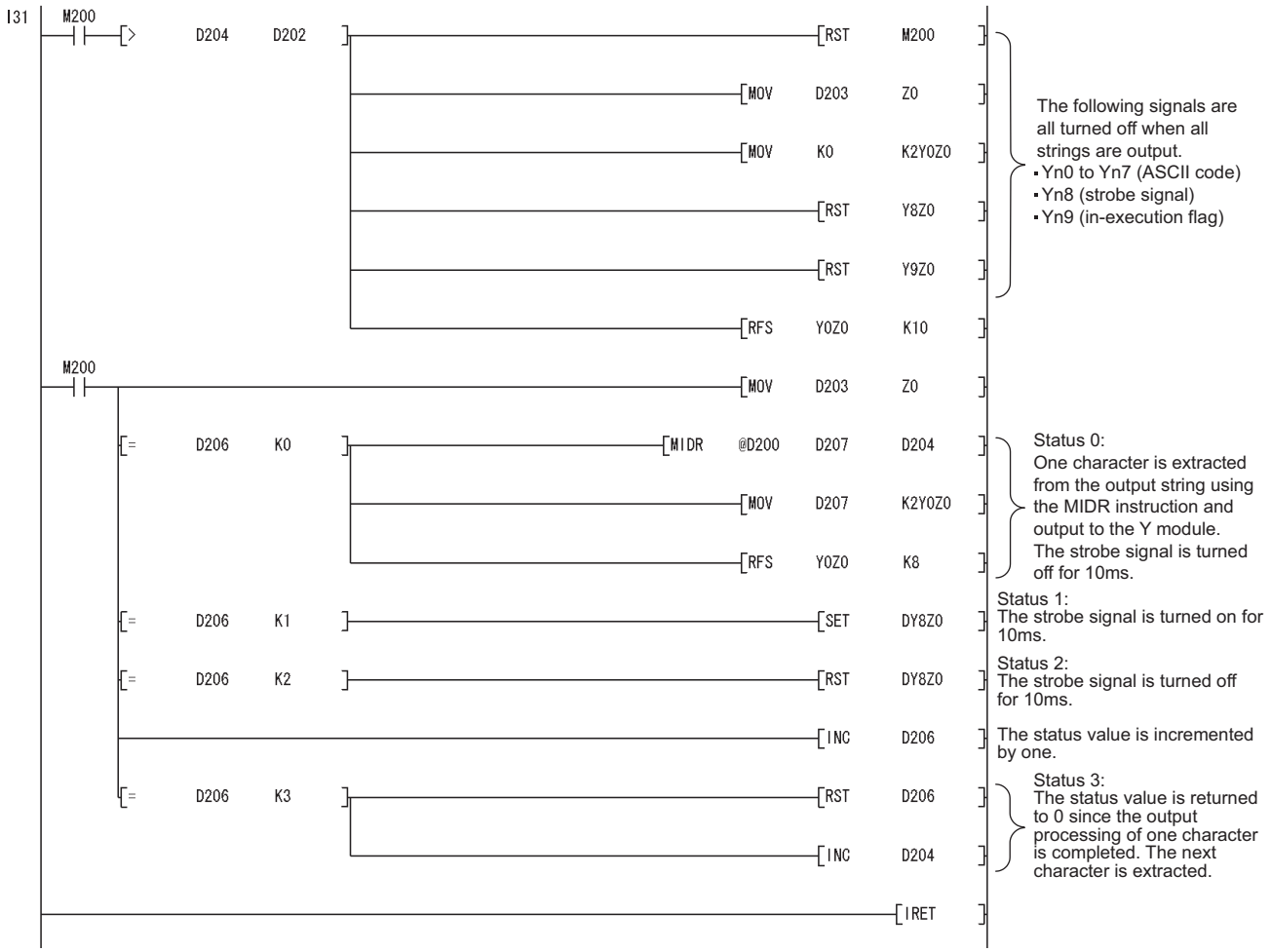
- In the subroutine program, the data for outputting ASCII codes using a fixed scan interrupt program (10ms) are set to work devices. Also, the flag for activating the processing in the fixed scan interrupt program is turned on.
- Specify the following arguments for the subroutine program.

First argument	Output string storage address	(Input)
Second argument	Output module start Y number	(Input)



3. Interrupt program

The following processing is added to a fixed scan interrupt program (10ms).
The fixed scan interrupt program outputs ASCII codes from the output module and controls the strobe signal.



(4) Replacement example of the CHKST and CHK instructions

In the example below, if the replacement program for the CHKST and CHK instructions detects a failure, a failure number (contact number + coil number) is stored in D200 and the annunciator F200 is turned on.

(a) Example of device assignment

Before replacement	
Application	**Device**
Advance end detection sensor input 1	X100
Retract end detection sensor input 1	X101
Advance end detection sensor input 2	X102
Retract end detection sensor input 2	X103
Advance end detection sensor input 3	X104
Retract end detection sensor input 3	X105
Advance end detection sensor input 4	X106
Retract end detection sensor input 4	X107
Failure detection output 1	Y100
Failure detection output 2	Y102
Failure detection output 3	Y104
Failure detection output 4	Y106

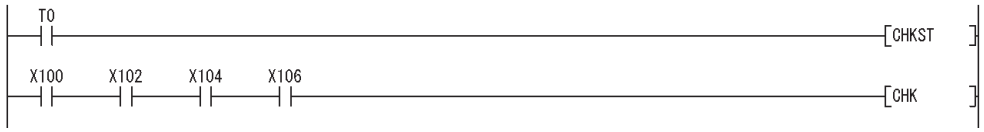
After replacement	
Application	**Device**
Advance end detection sensor input 1	X100
Retract end detection sensor input 1	X101
Advance end detection sensor input 2	X102
Retract end detection sensor input 2	X103
Advance end detection sensor input 3	X104
Retract end detection sensor input 3	X105
Advance end detection sensor input 4	X106
Retract end detection sensor input 4	X107
Failure detection output 1	Y100
Failure detection output 2	Y102
Failure detection output 3	Y104
Failure detection output 4	Y106
Coil number (failure type detected)	D100
Contact number	D101
Failure number	D200
Failure detection display	F200
For index modification	Z0

If the device numbers in the example above are used for other applications, assign unused device numbers instead.

When the advance end detection sensor input performs a failure detection of Xn, assign device numbers for the retract end detection sensor input and the failure detection output as described below.

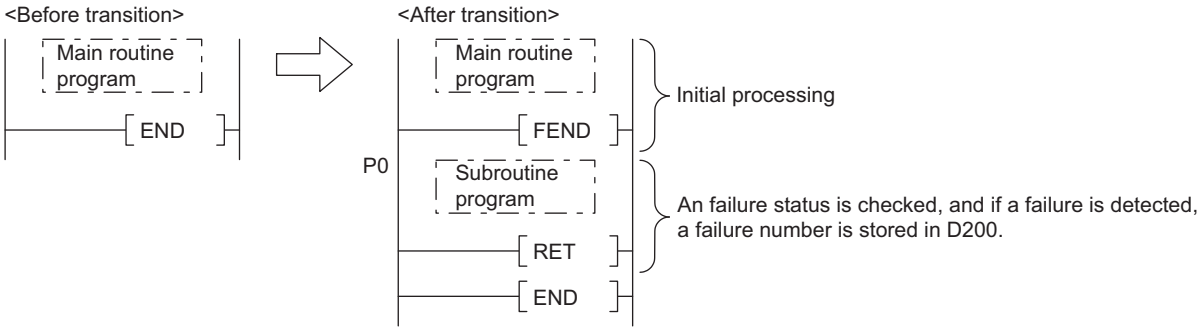
Advance end detection sensor input	Xn
Retract end detection sensor input	Xn+1
Failure detection output	Yn

(b) Program before replacement



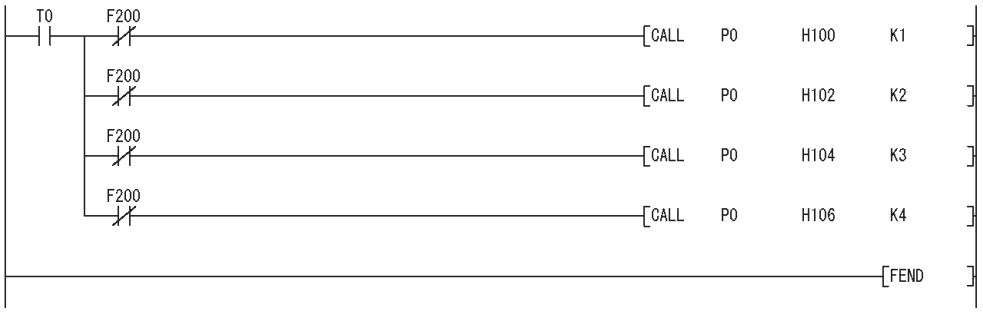
(c) Program after replacement

In the sequence program after replacement, two programs are required as shown below.



1. Main routine program

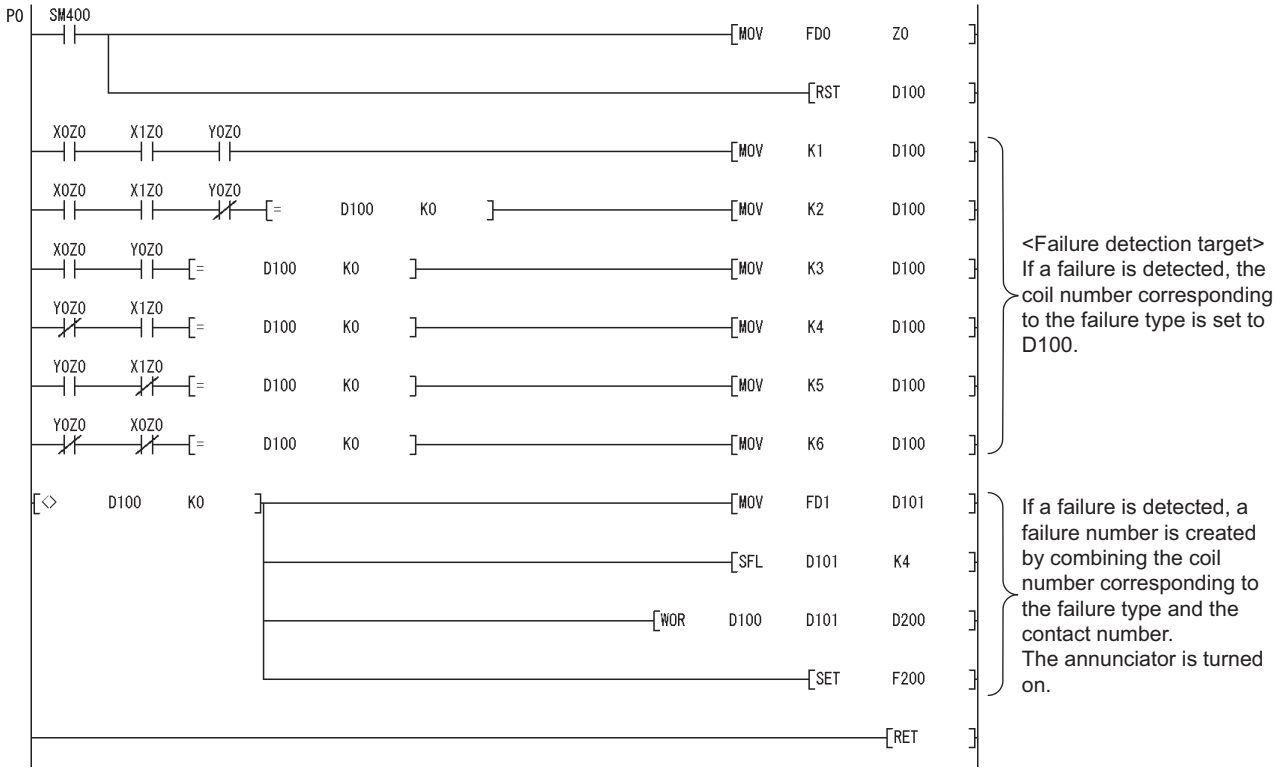
- Replace the CHKST and CHK instructions with the CALL instructions so that a subroutine program is called.
- One CALL instruction is required for each device specified as check condition in front of the CHK instruction. (In the program before replacement shown in (b), four CALL instructions need to be added since there are four check conditions in front of the CHK instruction.)
- Device number and contact number of X devices (check condition) are specified in each CALL instruction.
- Contact number is used to display failure number when a failure is detected.



2. Subroutine program

- In the subroutine program, a failure status is checked using a failure detection ladder pattern.
- If a failure is detected, a failure number is stored in D200 and the annunciator F200 is turned on.
- Specify the following arguments for the subroutine program.

First argument	Device number of X device targeted for failure check	(Input)
Second argument	Contact number of X device targeted for failure check	(Input)



(d) Replacement method when failure detection ladder patterns are changed by the CHKCIR and CHKEND instructions

Failure detection ladder patterns can be changed in the subroutine program described in (c).

(5) Replacement example of the KEY instruction

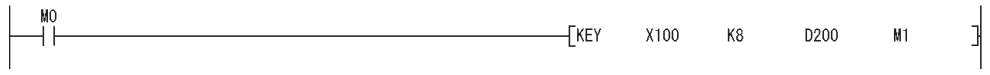
(a) Example of device assignment

Before replacement	
Application	**Device**
Numeric input execution instruction	M0
Input complete flag	M1
Input data area	D200 to D203
ASCII code input signal	X100 to X107
Strobe signal	X108

After replacement	
Application	**Device**
Numeric input execution instruction	M0
Input complete flag	M1
Input data area	D200 to D203
ASCII code input signal	X100 to X107
Strobe signal	X108
Input data area address (BIN32)	D210 to D211
(Input data area + 0) address (BIN32)	D212 to D213
(Input data area + 1) address (BIN32)	D214 to D215
(Input data area + 2) address (BIN32)	D216 to D217
For shifting input data	D218
For converting input data	D219 to D220

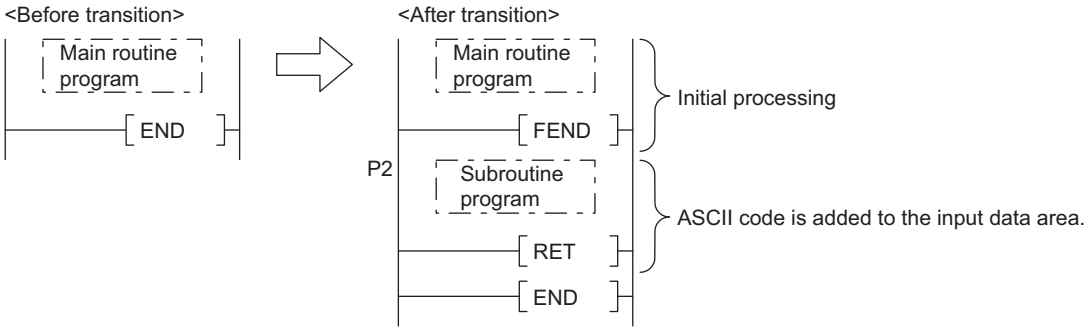
If the device numbers in the example above are used for other applications, assign unused device numbers instead.

(b) Program before replacement



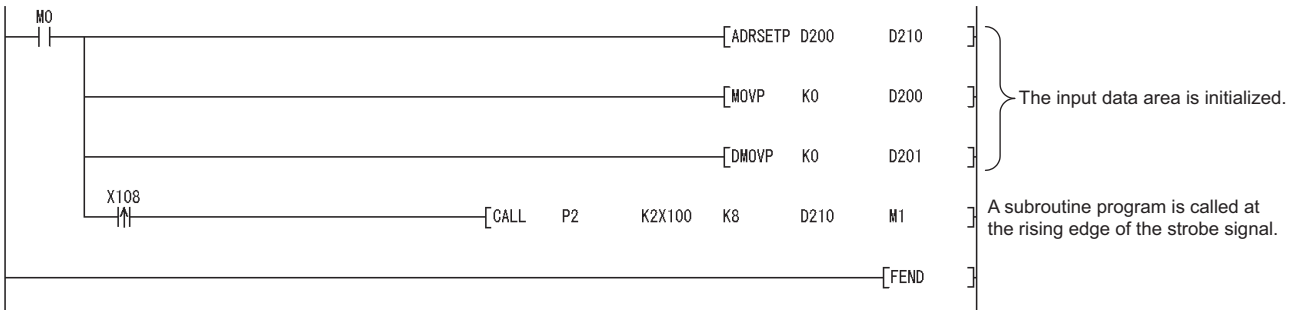
(c) Program after replacement

In the sequence program after replacement, two programs are required as shown below.



1. Main routing program

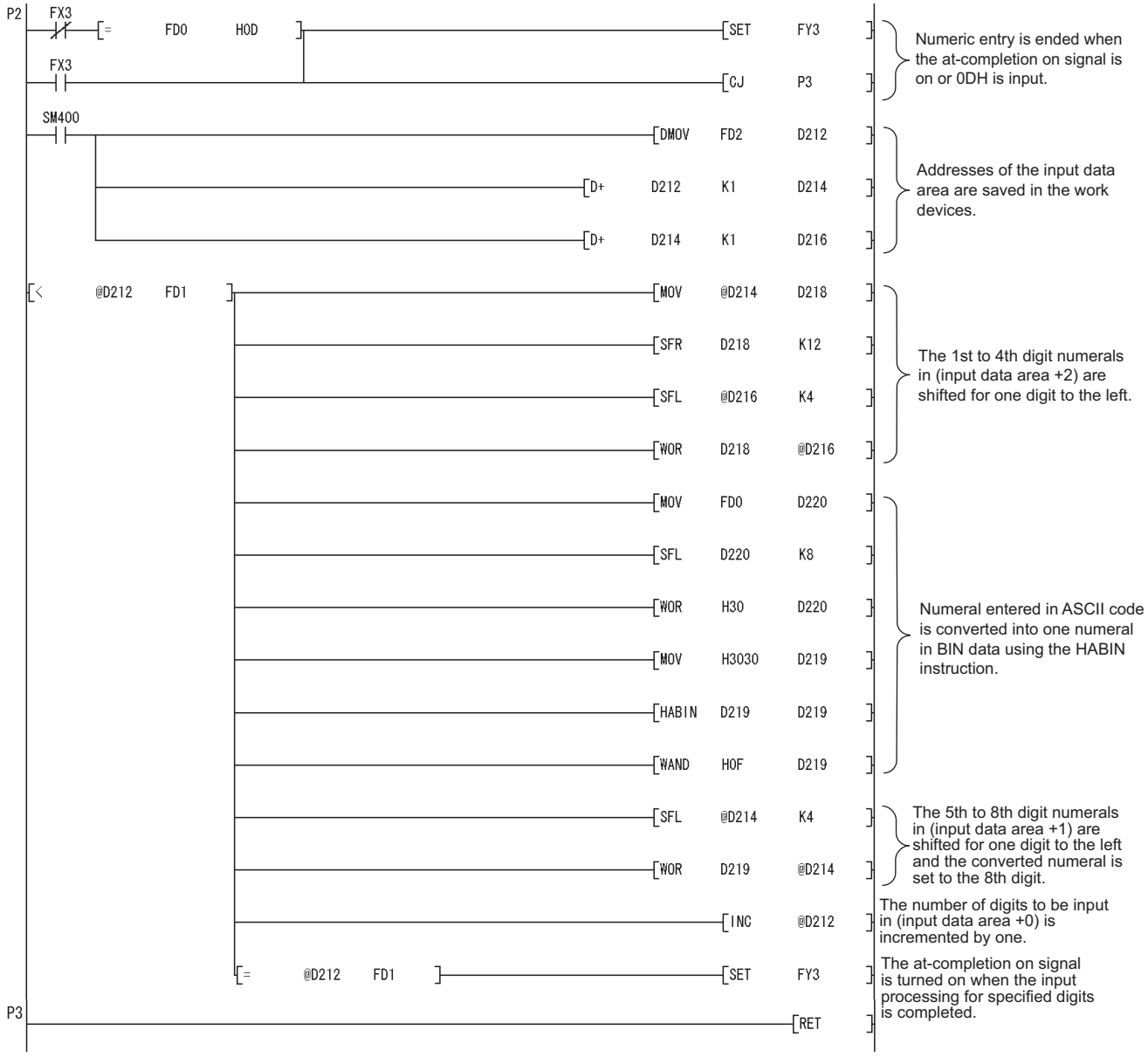
- Set "0" in the input data area on the rising edge of the execution instruction ("M0" in the program below) and initialize the program.
- Execute the CALL instruction on every rising edge of the strobe signal ("X108" in the program below) so that a subroutine program is called.
- In the subroutine program, input codes are added to the input data area and the completion status is checked.
- Pass the following data to the subroutine program at execution of the CALL instruction.
  1) ASCII code input values from the input module (Xn0 to Xn7)
  2) Number of digits to be input
  3) Indirect address of the input data area (Use the ADRSET instruction to acquire the indirect address for the input data area.)
  4) Bit devices to be turned on when input is completed



2. Subroutine program

- In the subroutine program, ASCII codes specified by an argument are added to the input data area and the completion status is checked.
- Specify the following arguments for the subroutine program.

First argument	ASCII code input from the input module (K2Xn)	(Input)
Second argument	Number of digits to be input	(Input)
Third argument	Indirect address of the input data area	(Input)
Fourth argument	Bit device to be turned on when input is completed	(Output)



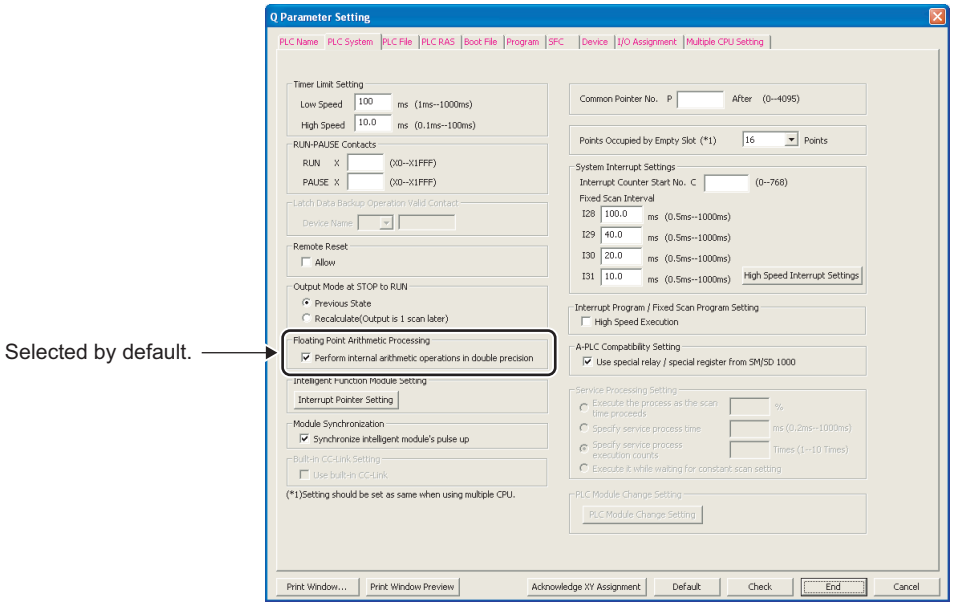
Appendix 5.4 Functions

Appendix 5.4.1 Floating-point operation instructions

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

The High Performance model QCPU can perform only the single-precision floating-point operation instructions. Note, however, that internal operation processing can be performed in double precision by selecting the item shown below (default: selected).



(b) Universal model QCPU

The Universal model QCPU supports the double-precision floating-point operation instructions.
The operation can be performed either in single precision or double precision depending on the data.
Therefore, the "Perform internal arithmetic operations in double precision" item in the PLC system tab of the PLC parameter dialog box cannot be selected.
Because of this new function, operation results (both in single precision and double precision) slightly differ between the High Performance model QCPU and the Universal model QCPU if "Perform internal arithmetic operations in double precision" is selected in the High Performance model QCPU.
If higher accuracy is required in floating-point operations, replace the floating-point operation instructions as described in Page 539, Appendix 5.4.1 (4). However, if six or less digits are used as significant digits for the floating-point operation instructions, replacement is not necessary. The single-precision floating-point operation results in the Universal model QCPU can be used as they are in the system. When not replacing the instructions, ensure that it does not cause any problems in the system.

(2) Floating-point operation instructions for the Universal model QCPU

The following table lists floating-point operation instructions for the Universal model QCPU.
Specifications of the single-precision floating-point operation instructions are compatible with those for the High Performance model QCPU.

Instruction name		Instruction symbol		Remarks
		Single-precision floating-point data	Double-precision floating-point data	
Comparison	Floating-point data comparison	LDE□	LDED□	□ indicates one of the following; <>,=,<,>,<=,>=.
		ANDE□	ANDED□	
		ORE□	ORED□	
Data transfer	Floating-point data transfer	EMOV(P)	EDMOV(P)	---
Four arithmetic operation	Floating-point data addition	E+(P)	ED+(P)	---
	Floating-point data subtraction	E-(P)	ED-(P)	
	Floating-point data multiplication	E*(P)	ED*(P)	
	Floating-point data division	E/(P)	ED/(P)	
Data conversion	Conversion from BIN 16-bit data to floating-point data	FLT(P)	FLTD(P)	---
	Conversion from BIN 32-bit data to floating-point data	DFLT(P)	DFLTD(P)	
	Conversion from floating-point data to BIN 16-bit data	INT(P)	INTD(P)	
	Conversion from floating-point data to BIN 32-bit data	DINT(P)	DINTD(P)	
	Floating-point sign inversion	ENEG(P)	EDNEG(P)	
Special function	SIN operation	SIN(P)	SIND(P)	---
	COS operation	COS(P)	COSD(P)	
	TAN operation	TAN(P)	TAND(P)	
	SIN-1 operation	ASIN(P)	ASIND(P)	
	COS-1 operation	ACOS(P)	ACOSD(P)	
	TAN-1 operation	ATAN(P)	ATAND(P)	
	Conversion from angle to radian	RAD(P)	RADD(P)	
	Conversion from radian to angle	DEG(P)	DEGD(P)	
	Square root	SQR(P)	SQRD(P)	
	Exponential operation	EXP(P)	EXPD(P)	
	Natural logarithm operation	LOG(P)	LOGD(P)	

Floating-point data can be converted mutually between single precision and double precision using instructions in the following table.

Instruction name	Instruction symbol
Single precision to double precision conversion	ECON(P)
Double precision to single precision conversion	EDCON(P)

(3) Advantages and disadvantages when using the double-precision floating-point data of the Universal model QCPU

The following table shows the advantages and disadvantages when executing the double-precision floating-point operation instructions in the Universal model QCPU.
If higher accuracy is required in floating-point operations, it is recommended to replace the instructions with the double-precision floating-point operation instructions.

Advantage	Disadvantage
The results are more accurate than those of the single-precision floating-point operation instructions.	The instruction processing speed is slower than that of the single-precision floating-point operation instructions.*1 Double-precision floating-operation data use twice as many word device points as single-precision floating-operation data.

*1 The processing speed of the double-precision floating-point operation instructions in the Universal model QCPU is higher than that of floating-point operation instructions using internal double-precision operations in the High Performance model QCPU.

The following table lists the comparison between single-precision and double precision floating-point data.

Item		Single-precision floating-point data	Double-precision floating-point data
Word point required for data retention		2 words	4 words
Setting range		$-2^{128} < N \leq -2^{-126}$, 0, $2^{-126} \leq N < 2^{128}$	$-2^{1024} < N \leq -2^{-1022}$, 0, $2^{-1022} \leq N < 2^{1024}$
Precision (number of bits)	Mantissa	23 bits	52 bits
	Exponent	8 bits	11 bits
	Sign	1 bit	1 bit
Instruction processing speed (Q04UDHCPU/ Q06UDHCPU) (minimum)	Data comparison (Conductive status) (LDE>= / LDED>=)	0.0285μs	3.6μs
	Data transfer (EMOV/EDMOV)	0.019μs	1.7μs
	Addition (3 devices) (E+ / ED+)	0.0665μs	4.8μs
	SIN operation (SIN/SIND)	4.1μs	8.5μs
Instruction processing speed (High-speed Universal model QCPU) (minimum)	Data comparison (Conductive status) (LDE>= / LDED>=)	0.0098μs	1.8μs
	Data transfer (EMOV/EDMOV)	0.0039μs	0.0078μs
	Addition (3 devices) (E+ / ED+)	0.015μs	1.9μs
	SIN operation (SIN/SIND)	1.6μs	2.6μs

(4) Replacing the High Performance model QCPU with the Universal model QCPU

(a) Replacing all single-precision floating-point operation instructions with double-precision floating-point operation instructions

Single-precision floating-point data occupy two points of word device per data.
On the other hand, four points are required per double-precision floating-point data.
Therefore, all device numbers for storing floating-point data need to be reassigned.

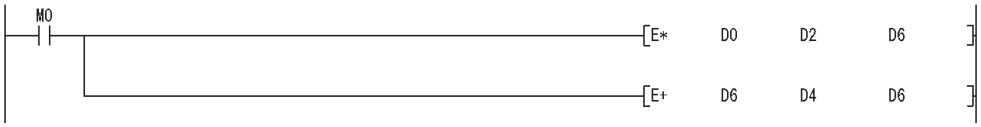
Ex. Replacing the floating-point operation [A × B + C] (Changing all floating-point data into double precision.)

• Device assignment

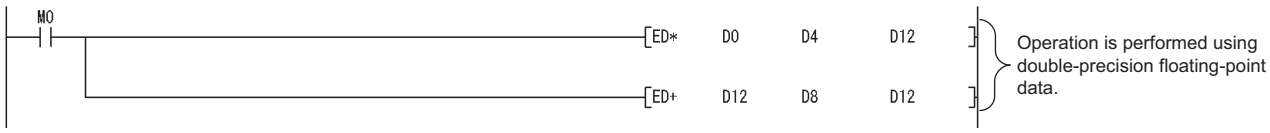
Before replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Result	D6 to D7	

After replacement		
Application	**Device**	**Data type**
Data A	D0 to D3	Floating-point data (double precision)
Data B	D4 to D7	
Data C	D8 to D11	
Result	D12 to D15	

• Program before replacement



• Program after replacement



(b) Replacing a part of floating-point operation instructions with double-precision floating-point operation instructions

Only operations that require high accuracy are replaced with double-precision floating-point operation instructions.

Using the ECON and EDCON instructions, convert floating-point data mutually between single precision and double-precision. The flow of a replacement program is as follows:

- Data required for operations are converted from single precision to double precision using the ECON instruction.
- Operations are performed in double precision using the double-precision floating-point operation instructions.
- Operation results are converted from double precision to single precision using the EDCON instruction.

A program example that floating-point data are converted mutually between single precision and double precision before and after operations is shown below.

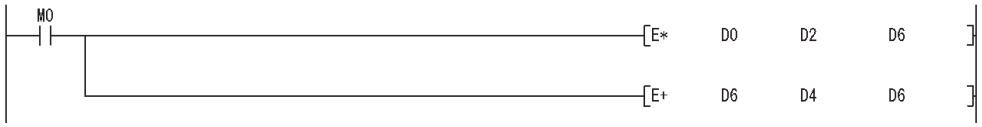
Ex. Replacing the floating-point operation [A × B + C] (Using the ECON and EDCON instructions)

- Device assignment

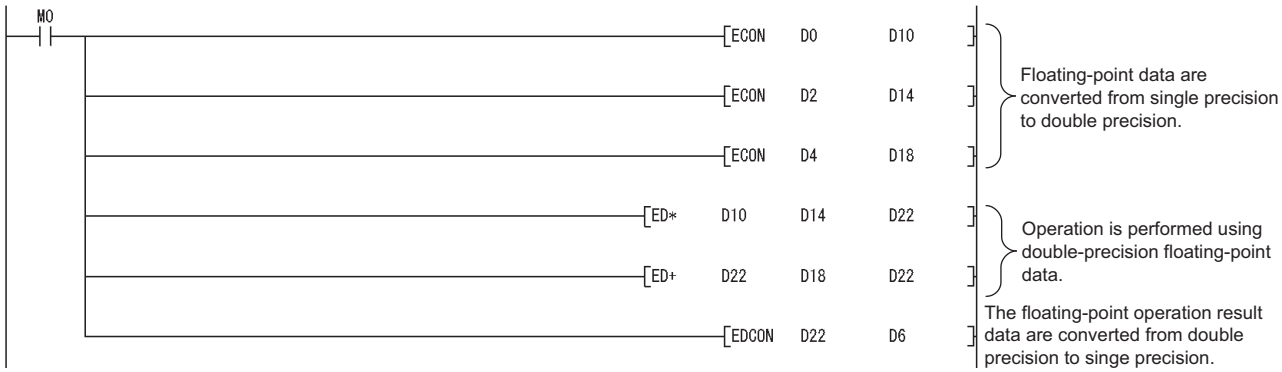
Before replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Result	D6 to D7	

After replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Result	D6 to D7	
Data A	D10 to D13	Floating-point data (double precision)
Data B	D14 to D17	
Data C	D18 to D21	
Result	D22 to D25	

- Program before replacement



- Program after replacement



(c) Replacing a part of floating-point operation instructions with double-precision floating-point operation instructions using subroutine programs

The flow of a replacement program described in (b) can be regarded as one subroutine program.
Create a subroutine program for each floating-point operation instruction and then replace the original floating-point operation instructions with the CALL(P) instruction so that the corresponding subroutine program is called.
With this method, changes in the program are minimized, but the processing for calling subroutine programs increases the scan time.
In addition, since conversions from double precision to single precision are performed for each instruction, rounding-off errors generated during operations are larger than those in the replacement program described in (b).

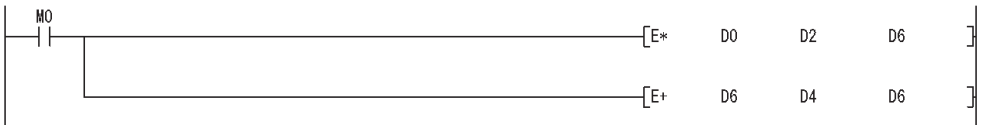
Ex. Replacing the floating-point operation[A × B + C] (Using a subroutine program)

• Device assignment

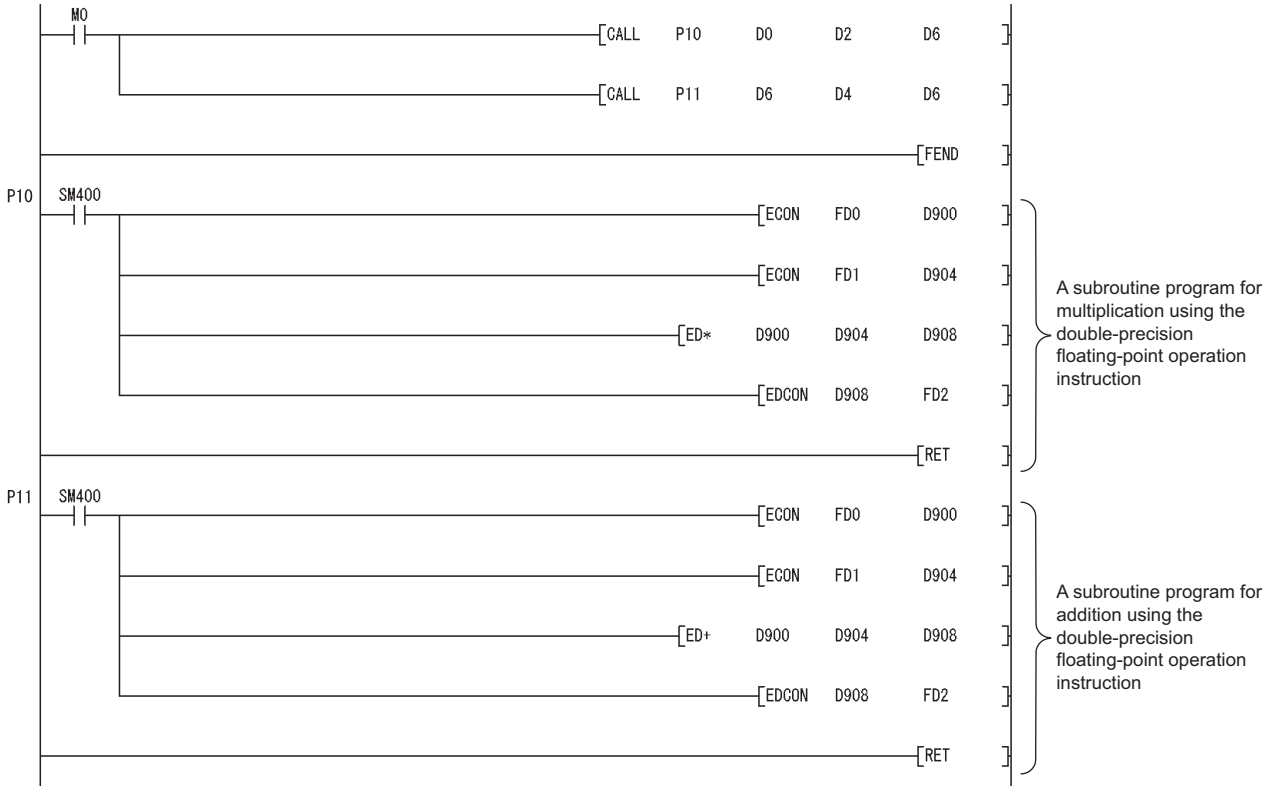
Before replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Result	D6 to D7	

After replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Result	D6 to D7	
Subroutine input data 1	D900 to D903	Floating-point data (double precision)
Subroutine input data 2	D904 to D907	
Subroutine operation result	D908 to D911	

• Program before replacement



• Program after replacement



Appendix 5.4.2 Error check processing for floating-point data comparison instructions (excluding High-speed Universal model QCPU)

(1) Input data check

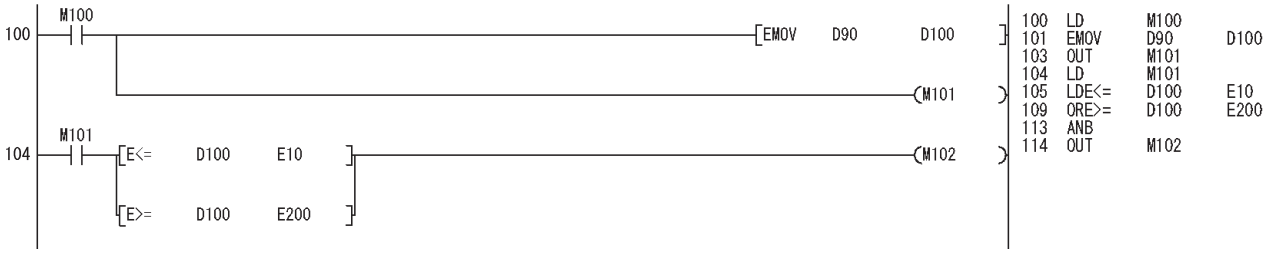
Error check processing for floating-point data comparison instructions performed in the Universal model QCPU are enhanced. Input of a "special value" (-0, nonnumeric, unnormalized number, or ±∞) is checked, and if those special values are input, the CPU module detects "OPERATION ERROR" (error code: 4140).
When the LDE□, ANDE□, ORE□, LDED□, ANDED□, and/or ORED□ instructions (□ indicates one of the following; =,<>,<,>,<=,>=) are used in the program, "OPERATION ERROR" (error code: 4140) may be detected if invalid floating-point data exist. This occurs even when interlocks are provided using the valid data flags (the signal which shows the floating-point validity).
Invalid floating-point data are not stored in the result of operations performed in the Universal model QCPU.
Those invalid data are considered to be stored in the following cases:

- The same device is used for storing floating-point data and other data, such as binary values, BCD values, and strings.
  → Use different devices for storing floating-point data and data other than floating-point data.
- Floating-point data externally written are invalid.
  → Take measures on the external-source side so that valid data are written.

If an error occurs in the floating-point data comparison instructions, take appropriate measures to remove error causes described above.

Ex.1) Detecting "OPERATION ERROR" (error code: 4140) in the LDE□ instruction

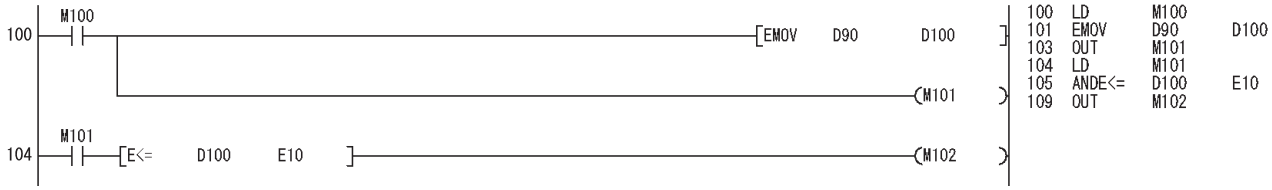
[Ladder mode] [List mode]



In the ladder block starting from the step 104, the floating-point data comparison instructions of the step 105 and 109 are not executed when the M101 (valid data flag) is off.
However, the LDE<= instruction of the step 105 and the ORE>= instruction of the step 109 are executed regardless of the execution result of the LD instruction of the step 104 in the program above.
Therefore, even when the M101 is off, "OPERATION ERROR" (error code: 4140) will be detected in the LDE<= instruction of the step 105 if a "special value" is stored in D100.
For the method of avoiding "OPERATION ERROR", refer to Page 545, Appendix 5.4.2 (2).

Ex.2) Not detecting "OPERATION ERROR" (error code: 4140) in the ANDE□ instruction

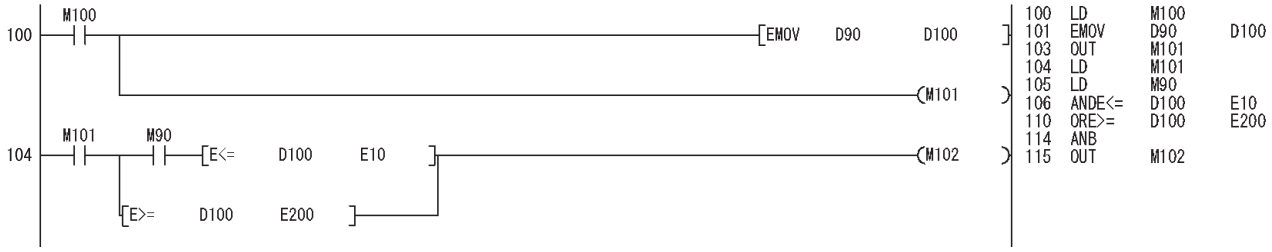
[Ladder mode] [List mode]



In the ladder block starting from the step 104, the ANDE<= instruction of the step 105 is not executed when the M101 (valid data flag) is off.
The ANDE<= instruction of the step 105 is not executed when the M101 is off in the LD instruction of the step 104 in the program above. Therefore, when the M101 is off, "OPERATION ERROR" (error code: 4140) will not be detected even if a "special value" is stored in D100.

Ex.3) Detecting "OPERATION ERROR" (error code: 4140) in the ANDE□ instruction

[Ladder mode] [List mode]

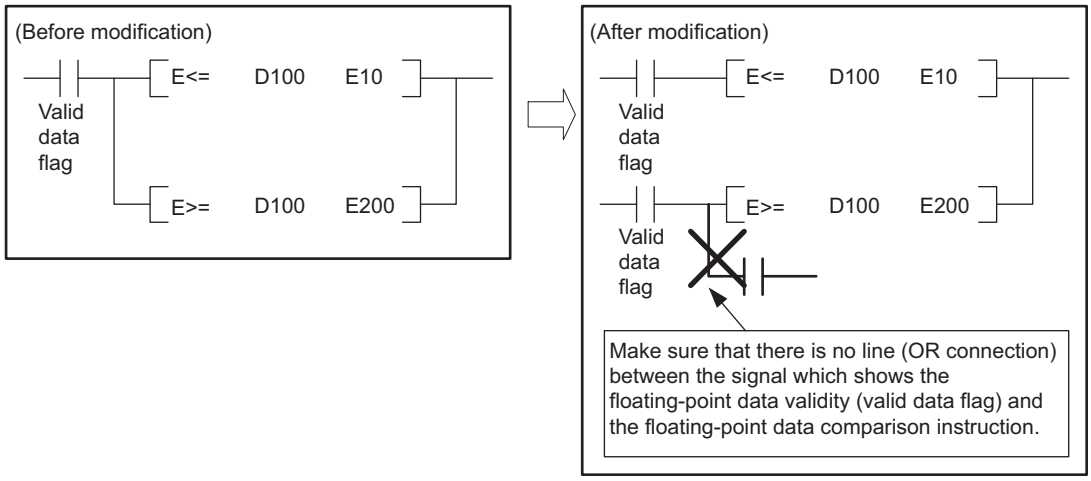


In the ladder block starting from the step 104, the ANDE<= instruction of the step 106 and the ORE>= instruction of the step 110 are not executed when the M101 (valid data flag) is off.
However, if the M90 is on in the LD instruction of the step 105, the ANDE<= instruction of the step 106 is executed. Therefore, even when the M101 is off, "OPERATION ERROR" (error code: 4140) will be detected in the ANDE<= instruction of the step 106 if the M90 is on and a "special value" is stored in D100.
For the method of avoiding "OPERATION ERROR", refer to Page 545, Appendix 5.4.2 (2).

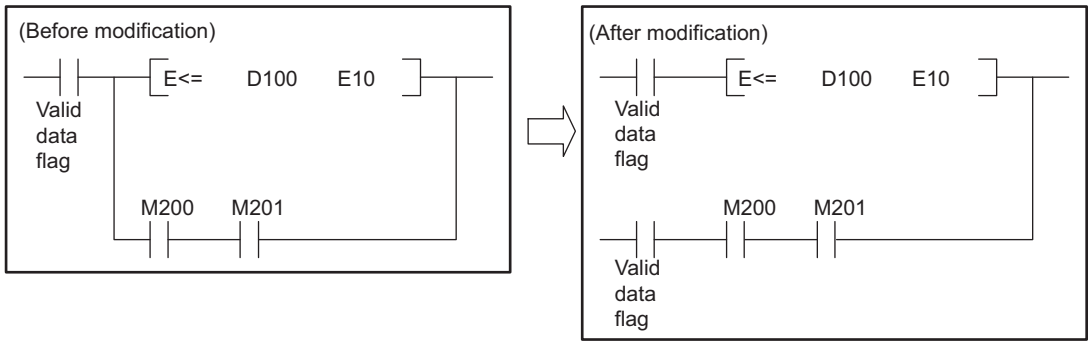
(2) Method of avoiding "OPERATION ERROR" (error code: 4140) in the floating-point data comparison instructions

As shown in the modification examples below, connect the contacts of valid data flag in series for each floating-point data comparison instruction. (Use AND connection for connecting the contact of the valid data flag and the floating-point data comparison instruction.)
Ensure that there is no line (OR connection) between the valid data flag and the floating-point data comparison instruction.

<Modification example 1>



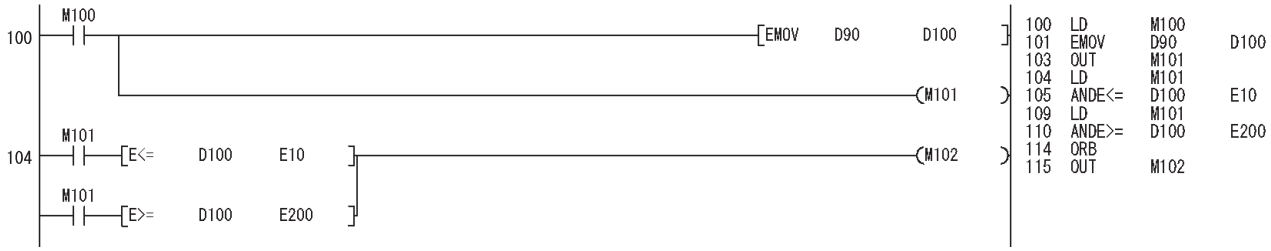
<Modification example 2>



Program examples after modification for Example 1) and 3) in (1) are shown below.

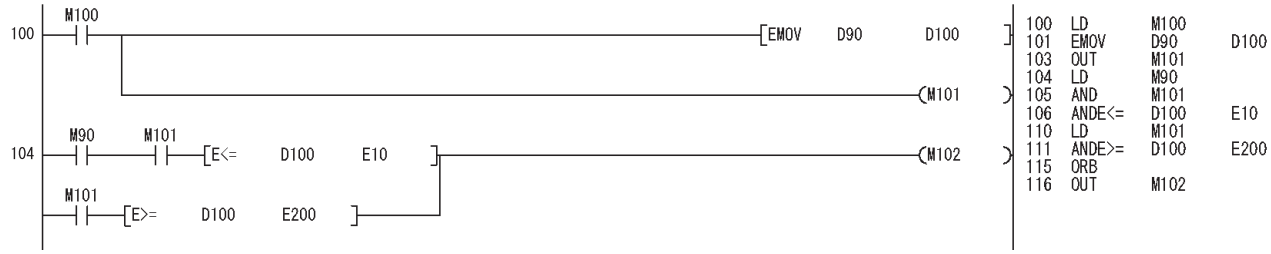
Ex.4) Program after modification for Example 1) ("OPERATION ERROR" (error code: 4140) is no longer detected.)

[Ladder mode] [List mode]



Ex.5) Program after modification for Example 3) ("OPERATION ERROR" (error code: 4140) is no longer detected.)

[Ladder mode] [List mode]



Appendix 5.4.3 Range check processing for index-modified devices

(1) Device range check

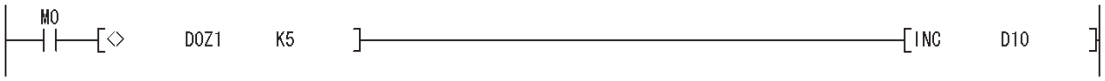
Error check processing at index modification of devices has been enhanced for the Universal model QCPU. Each index-modified device range is checked, and if the check target device is outside the device range before index modification, the CPU module detects "OPERATION ERROR" (error code: 4101).

Point

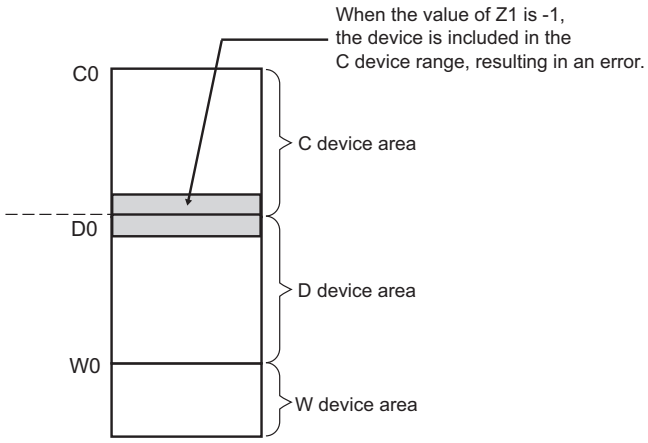
For details on the index-modified device range check, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

Ex.1) Detecting "OPERATION ERROR" (error code: 4101) by error check processing at index modification of devices (QnU(D)(H)CPU, QnUDE(H)CPU, LCPU)



In Example 1), when the contact (M0) is on and the value -1 or less is specified in Z1, the device D0Z1 is included in the C device range, exceeding the D device range, as shown in the following figure. As a result, "OPERATION ERROR" (error code: 4101) will be detected.

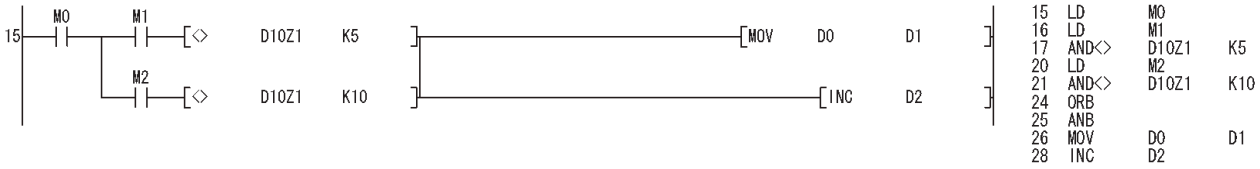


When an error is detected, check the index modification value (value of Z1 in the above example) and remove the error cause.

Examples of the cases where an error is detected and not detected are shown below.

Ex.2) Detecting "OPERATION ERROR" (error code: 4101) (QnU(D)(H)CPU, QnUDE(H)CPU, LCPU)

[Ladder mode] [List mode]



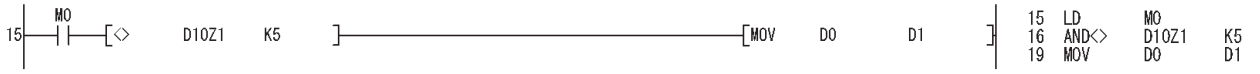
In Example 2, in the ladder block starting from the step 15, the AND<> instruction of the step 17 or 21 is supposed to be not executed when M0 (valid data flag) is off.
However, since the LD instruction which is always executed is used in the step 16 and 20, the AND<> instruction of the step 17 or 21 is executed regardless of the execution status of the LD instruction in the step 15 when M1 or M2 is on.
For this reason, even when M0 is off, if the D10Z1 value is outside the D device range, "OPERATION ERROR" (error code: 4101) will be detected in the AND<> instruction of the step 17.
Note that the step 26 (MOV D0 D1) and the step 28 (INC D2) are not executed. For the actions to be taken to avoid "OPERATION ERROR" (error code: 4101), refer to Page 549, Appendix 5.4.3 (2) 1) to 4).

Ex.3) Detecting "OPERATION ERROR" (error code: 4101) (QnU(D)(H)CPU, QnUDE(H)CPU, LCPU)

In Example 3, even when M0 (valid data flag) in the step 15 is off, the AND instruction in the next step (step 16) will be executed. For this reason, if the X10Z1 value is outside the X device range, "OPERATION ERROR" (error code: 4101) will be detected in the AND instruction of the step 16.
For the actions to be taken to avoid "OPERATION ERROR" (error code: 4101), refer to Page 549, Appendix 5.4.3 (2) 1), 3), and 4).

Ex.4) Not detecting "OPERATION ERROR" (error code: 4101)

[Ladder mode] [List mode]



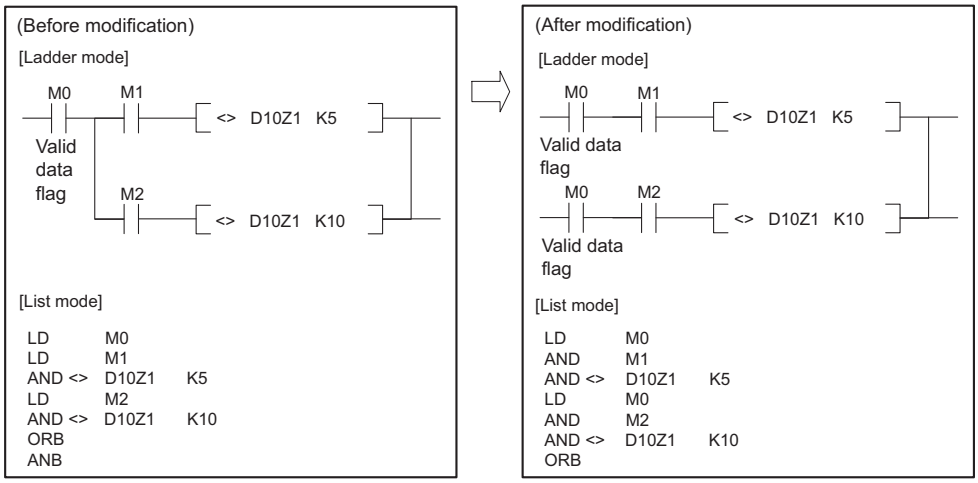
In Example 4, the AND<> instruction of the step 16 is not executed when M0 (valid data flag) of the step 15 is off.
For this reason, "OPERATION ERROR" (error code: 4101) will not be detected no matter what the D10Z1 value is.

(2) Actions taken to avoid "OPERATION ERROR" (error code: 4101)

If the index-modified device range does not need to be checked, set the parameter as described in 1).
If the index-modified device range needs to be checked, but the detection of errors shown in Examples 2 and 3 in Page 547, Appendix 5.4.3 (1) should be avoided, take actions described in 2) to 4).

1) Deselect the "Check Device Range at Indexing" item in the PLC RAS tab of the PLC parameter dialog box so that the index-modified device range will not be checked.
2) As shown in the following modification example, connect the valid data flag contact in series for each instruction that performs the index-modified device range check processing (except when a High-speed Universal model QCPU and Universal model Process CPU are used).

<Modification example>



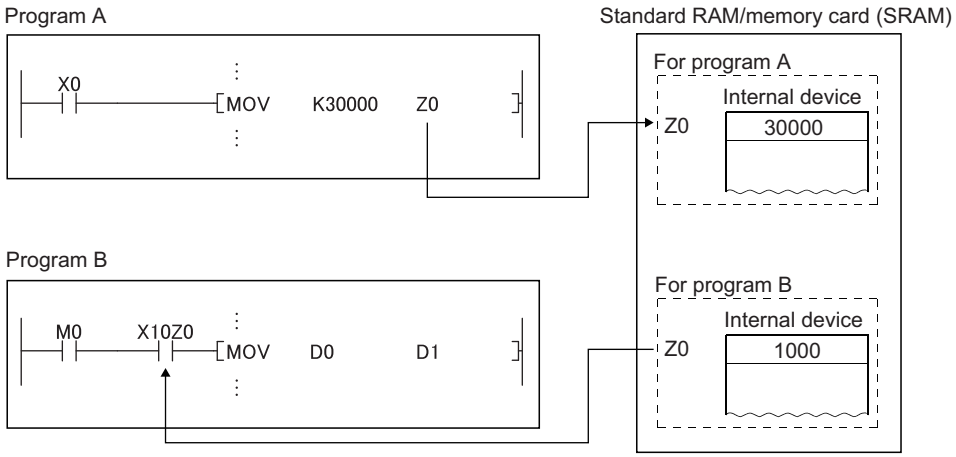
In the program before modification (on the left), the instruction immediately before the AND<> instruction is regarded as the LD instruction. However, in the program after modification (on the right), the same instruction will be regarded as the AND instruction.
In the program after modification, only when both contacts of M0 and M1 (or M2) turn on, the AND<> instruction is executed. As a result, no error will be detected during index-modified device range check processing.

3) Use the index register as a local device.

With a project where multiple programs are executed, to avoid the detection of "OPERATION ERROR" (error code: 4101) both when multiple programs are executed and a particular program alone is executed, use the index register as a local device.
Using the index register as a local device provides an independent index register for each program. Even if another program overwrites the index register with a "value that causes the index-modified device to be outside the device range," it will not affect the value of the index register used in the program where the error occurs with the overwritten value. As a result, "OPERATION ERROR" (error code: 4101) will not be detected.
Note that the scan time increases because the time for saving and restoring the file register file increases. For the local device settings, refer to Page 424, Section 6.2 (3).

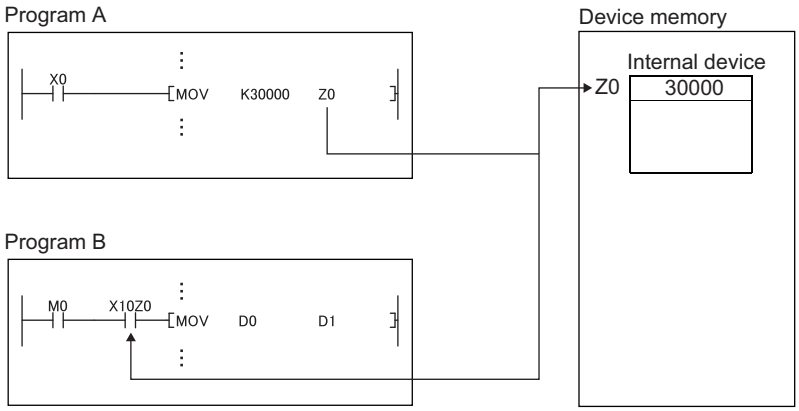
Ex. When the index register is used as a local device

Even when program A overwrites the index register Z0 with a value of 30000, no change is made to the index register Z0 used by program B. No error occurs as long as X10Z0 does not exceed the X device range.



Ex. When the index register is not used as a local device

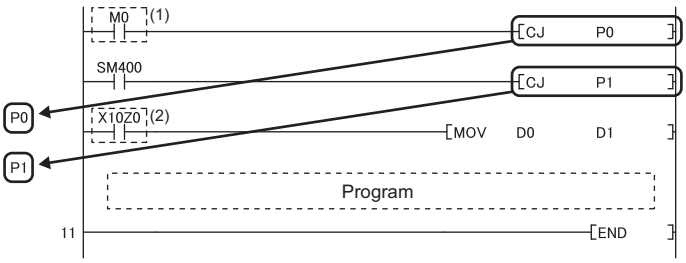
When program A overwrites the index register Z0 with a value of 30000, the value of the index register Z0 used by program B is also changed. An error occurs when X10Z0 exceeds the X device range.



4) Use the CJ instruction.

When the CJ instruction is used as shown below and the previous condition (denoted as "(1) LD M0" in the figure below) is off, avoid the execution of a contact instruction that uses the index register (denoted as "(2) LD X10Z0" in the figure below). When condition (1) is off, instruction (2) is not executed and the value of the device used as a contact is not read. Thus, the device range check processing does not detect "OPERATION ERROR" (error code: 4101).

Note that the use of the CJ instruction increases the scan time.



Appendix 5.4.4 Device latch function

(1) Overview

The device latch function[*1] of the Universal model QCPU is more enhanced than than of the Basic model QCPU or High Performance model QCPU.
This section describes the enhanced device latch function of the Universal model QCPU.

*1 The latch function is used to hold device data even when the CPU module is powered off or reset.

(2) Device data latch methods

Device data of the Universal model QCPU can be latched by:

- using the large-capacity file register[*1],
- writing/reading device data to/from the standard ROM (with the SP.DEVST and S(P).DEVLD instructions),
- specifying a latch range of internal user devices, or
- setting intervals ("Time Setting") in the latch interval setting parameter.[*2]

*1 The extended data register (D) and extended link register (W) are included.
*2 Only the High-speed Universal model QCPU and Universal model Process CPU support the setting.

(3) Details of each latch method

(a) Large-capacity file register

Data in the file register can be latched by batteries.[*1]
File register size is larger and processing speed is higher in the Universal model QCPU, compared to the Basic model QCPU and High Performance model QCPU.
To latch a lot of data (many device points), use of the file register is effective. For the file register size in each CPU module, refer to Page 393, Section 4.7.2 (1).

*1 The latch range can be changed in the Device tab of the PLC parameter dialog box. (Setting of a file register: Page 397, Section 4.7.4 (1) (c), Settings of an extended data register (D) and an extended link register (W): Page 404, Section 4.8 (2) (b))

(b) Writing/reading device data to the standard ROM (SP.DEVST and S(P).DEVLD instructions)

Device data of the Universal model QCPU can be latched using the SP.DEVST and S (P).DEVLD instructions (instructions for writing/reading data to/from the standard ROM).
Utilizing the standard ROM allows data backup without batteries.
This method is effective for latching data that will be updated less frequently.

(c) Specifying the latch range of internal user devices

Device data of the Universal model QCPU can be latched by specifying a latch range of internal user devices in the same way as for the Basic model QCPU and High Performance model QCPU. The ranges can be set in the Device tab of the PLC parameter dialog box. Internal user devices that can be latched are as follows:

- Latch relay (L)
- Link relay (B)
- Annunciator (F)
- Edge relay (V)
- Timer (T)
- Retentive timer (ST)
- Counter (C)
- Data register (D)
- Link register (W)

The following devices also can be set when a file register is set to be used in the PLC file.

- File register (R, ZR)
- Extended data register (D)
- Extended link register (W)

Point

If latch ranges of internal user devices are specified in the Universal model QCPU, the processing time will be added to the scan time in proportion to the device points set to be latched.*1 (For example, if 8K points are latched for the latch relay (L), the scan time will be 28.6μs.) To shorten the scan time, remove unnecessary latch device points to minimize the latch range.

*1 For file registers (including an extended data register (D) and an extended link register (W)), the scan time is not increased due to latch.

(4) How to shorten the scan time

When data to be latched are stored in the file register the processing time is shorter than that for latching internal user device.

Ex. Reducing the latch points of the data register (D) from 8K points to 2K points, and using the file register (ZR) instead (when the Q06UDHCPU is used)

Item		Before	After
Latch points of the data register (D)		8192 (8K) points	2048 (2K) points (6K points are moved to the file register.)
Number of devices in the program	Data register (D) (Latch range)	400	100
	File register (ZR) (Standard RAM)	0	300
Additional scan time		0.41ms	0.13ms*1
Number of steps increased		---	300 steps

*1 Time indicates the time required additionally when the file register is stored in the standard RAM.

Point

The High-speed Universal model QCPU and Universal model Process CPU can choose a latch interval setting between "Each Scan" and "Time Setting" in parameter. When "Time Setting" is selected, latch data processing starts during the first END processing after a preset time has elapsed. Since the latch data processing is performed asynchronous to the sequence program, an increase in scan time is reduced.

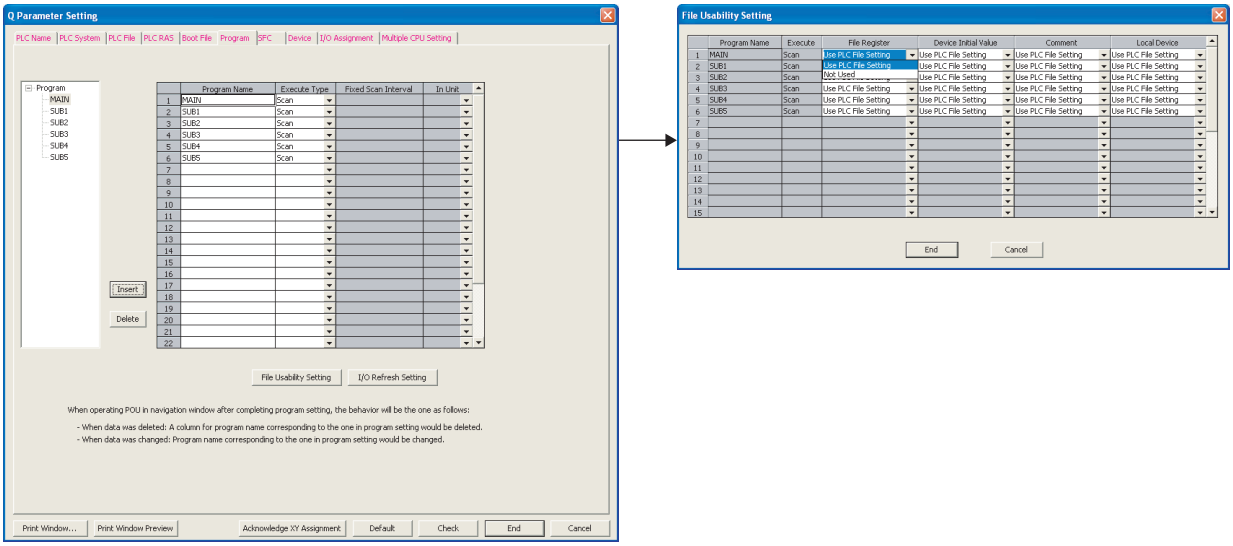
Appendix 5.4.5 File usability setting

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

In the High Performance model QCPU, file usability ("Use PLC File Setting" or "Not Used") of the following files can be set for each program on the screen opened by clicking the "File Usability Setting" button on the Program tab of the PLC parameter dialog box.

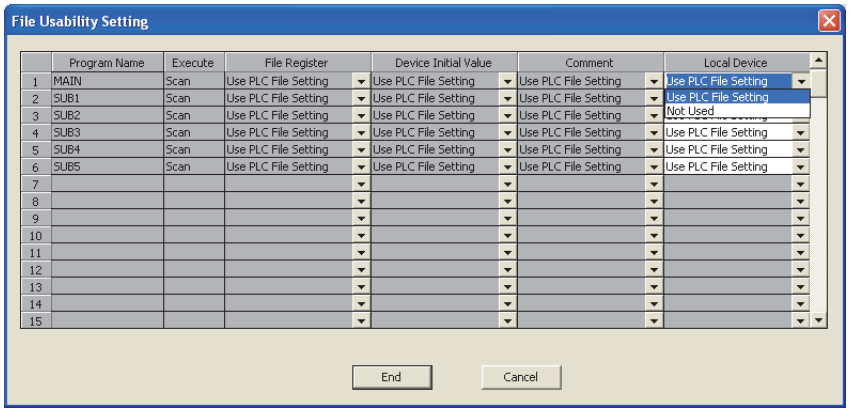
- File register
- Initial device value
- Comment
- Local device



(b) Universal model QCPU

In the Universal model QCPU, file usability of the following files[*1] cannot be set for each program on the screen opened by clicking the "File Usability Setting" button on the Program tab of the PLC parameter dialog box.

- File register
- Initial device value
- Comment

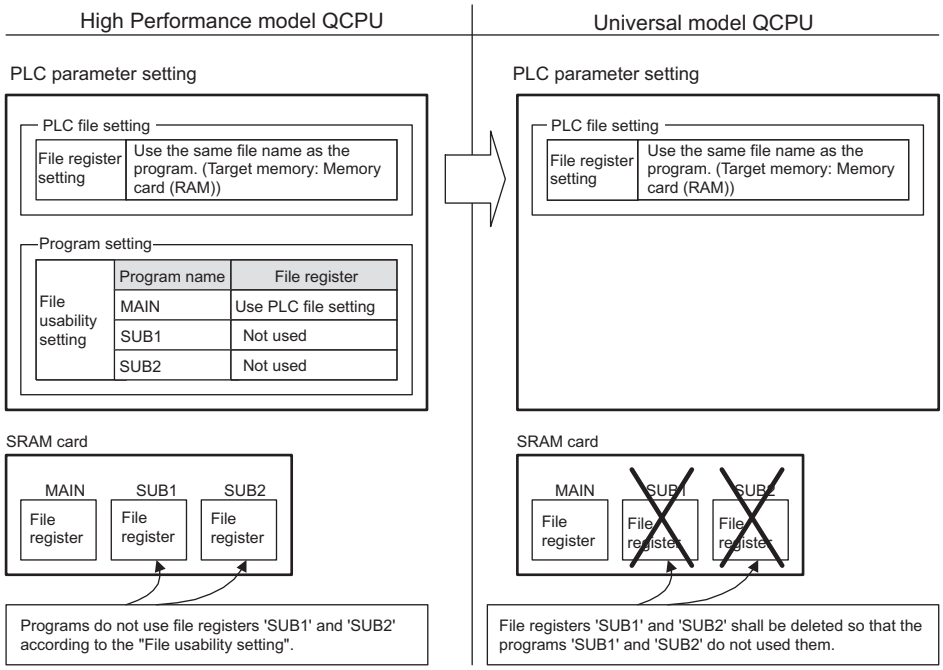


*1 Even file usability of local device file cannot be set if the serial number (first five digits) of the Q02UCPU, Q03UDCPU, Q04UDHCPU, or Q06UDHCPU is "10011" or earlier. If the local device is set to be used in the PLC file tab of the PLC parameter dialog box in the High Performance model QCPU, all the programs use the local device in the Universal model QCPU after replacement.

When the file usability setting is set in the High Performance model QCPU, change the setting as described below.

(2) Method of replacing High Performance model QCPU with Universal model QCPU

Replacement method varies depending on the settings in the PLC file tab of the PLC parameter dialog box.

Setting in the PLC file tab	Setting in Universal model QCPU
"Not Used" is selected.	No change in parameter settings is required. Operation of the Universal model QCPU is the same regardless of the file usability setting in the High Performance model QCPU.
"Use the same file name as the program" is selected.	When file usability is set to "Not Used" in the High Performance model QCPU, delete the corresponding program file (file register, initial device value, or comment), which uses the same name as the program, from th e target memory. The Universal model QCPU executes a program without using a program file if no program file that uses the same name as the program exists in the target memory. 
"Use the following file" is selected.	No change in parameter settings is required. Operation of the Universal model QCPU is the same regardless of the file usability setting in the High Performance model QCPU.

Appendix 5.4.6 Parameter-valid drive and boot file setting

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

The parameter-valid drive is specified by the switches on the front panel of the High Performance model QCPU.

(b) Universal model QCPU

The Universal model QCPU automatically determines the parameter-valid drive, depending on the existence of parameters in the drive (program memory, memory card, SD memory card, or standard ROM).
Therefore, when replacing the High Performance model QCPU with the Universal model QCPU, changing the boot file setting for parameter and/or moving files to another drive may be required.

When replacing the High Performance model QCPU with the Universal model QCPU, change the setting as described below.

(2) Replacing High Performance model QCPU with Universal model QCPU

(a) When the parameter-valid drive is set to the standard ROM in the High Performance model QCPU

<table>
<tr><th colspan="2">Setting in High Performance model QCPU
Setting in the Boot file tab of the PLC parameter dialog box</th><th>Setting in Universal model QCPU</th></tr>
<tr><td colspan="2">No boot file setting</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the standard ROM.
• Changes in parameter settings are not required.
• Delete parameters that exist in the program memory, memory card, and/or SD memory card.*2</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that programs are stored in the program memory in the first place, instead of booting from the standard ROM.
• Delete all settings for parameter in the Boot file tab of the PLC parameter dialog box.
• Delete parameters that exist in the program memory, memory card, and/or SD memory card.*2
• Move the programs with boot setting into the program memory from the standard ROM.*1</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Standard ROM</td><td>Program memory</td></tr></table></td><td>Change the setting so that programs and parameters are stored in the program memory in the first place, instead of booting from the standard ROM.
• Move the programs and parameters with boot setting into the program memory from the standard ROM.*1
• Delete all settings for parameter in the Boot file tab of the PLC parameter dialog box.</td></tr>
</table>

<table>
<tr><th>Setting in High Performance model QCPU
Setting in the Boot file tab
of the PLC parameter dialog box</th><th>Setting in Universal model QCPU</th></tr>
<tr><td>Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card or SD memory card, and programs are booted from the card to the program memory.
• Move the parameters in the standard ROM into the memory card or SD memory card.
• Make setting so that programs are booted from the memory card or SD memory card to the program memory in the Boot file tab of the PLC parameter dialog box.*3</td></tr>
<tr><td>Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card or SD memory card, and programs and parameters are booted from the card to the program memory.
• Move the parameters in the standard ROM into the memory card or SD memory card.
• Make setting so that programs and parameters are booted from the memory card or SD memory card to the program memory in the Boot file tab of the PLC parameter dialog box.*3</td></tr>
<tr><td>Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Data other than program and parameter</td><td>Memory card</td><td>Program memory</td></tr></table>Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Data other than program and parameter</td><td>Standard ROM</td><td>Program memory</td></tr></table>(Data other than program and parameter indicate initial device value, device comment, and label program.)</td><td>Delete all settings for data other than programs and parameters in the boot file setting.
Since these data can be used even not stored in the program memory, it is not necessary to transfer them to the program memory. Or, change the setting so that they are stored in the program memory in the first place.
• Delete all settings for data other than programs and parameters in the Boot file tab of the PLC parameter dialog box.
• Move the data other than programs and parameters into the program memory as needed.</td></tr>
</table>

*1 Since the Universal model QCPU holds the data in the program memory even when the battery voltage drops, the boot file setting is not necessary.

*2 The Universal model QCPU searches for parameters in order of in the program memory, in the memory card or SD memory card, and in the standard ROM. Then, the module uses the parameters found first. If parameters exist in the program memory or the card, the Universal model QCPU does not use the parameters in the standard ROM.

*3 The Universal model QCPU ignores the boot file setting for parameters in the standard ROM.

(b) When the parameter-valid drive is set to the memory card (RAM) or memory card (ROM) in the High Performance model QCPU

<table>
<tr><th colspan="2">Setting in High Performance model QCPU
Setting in the Boot file tab of the PLC parameter dialog box</th><th>Setting in Universal model QCPU</th></tr>
<tr><td colspan="2">No boot file setting</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card or SD memory card.
• Changes in parameter settings are not required.
• Delete parameters that exist in the program memory.*2</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card or SD memory card.
• Changes in parameter settings are not required.
• Delete parameters that exist in the program memory.*2</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td>No changes are required.</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that programs are stored in the program memory in the first place, instead of booting from the standard ROM.
• Move the programs targeted for booting from the standard ROM into the program memory.*1
• Delete all settings for program in the Boot file tab of the PLC parameter dialog box.
• Delete parameters that exist in the program memory.*2</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td>Change the setting so that programs are stored in the program memory in the first place, instead of booting from the standard ROM.
• Move the programs targeted for booting from the standard ROM into the program memory.*1
• Delete all settings for program in the Boot file tab of the PLC parameter dialog box.</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Data other than program and parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td rowspan="2">Delete all settings for data other than programs and parameters in the boot file setting.
Since these data can be used even not stored in the program memory, it is not necessary to transfer them to the program memory. Or, change the setting so that they are stored in the program memory in the first place.
• Delete all settings for data other than programs and parameters in the Boot file tab of the PLC parameter dialog box.
• Move the data other than programs and parameters into the program memory as needed.</td></tr>
<tr><td colspan="2">Settings in the Boot file tab
<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Data other than program and parameter</td><td>Standard ROM</td><td>Program memory</td></tr></table>(Data other than program and parameter indicate initial device value, device comment, and label program.)</td></tr>
</table>

*1 Since the Universal model QCPU holds the data in the program memory even when the battery voltage drops, the boot file setting is not necessary.

*2 The Universal model QCPU searches for parameters in order of in the program memory, in the memory card or SD memory card, and in the standard ROM. Then, the module uses the parameters found first. If parameters exist in the program memory or the card, the Universal model QCPU does not use the parameters in the standard ROM.

Appendix 5.4.7 External input/output forced on/off function

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

External input/output can be forcibly turned on/off on the screen opened by selecting [Online] → [Debug] → [Forced Input Output Registration/Cancellation] in programming tool.

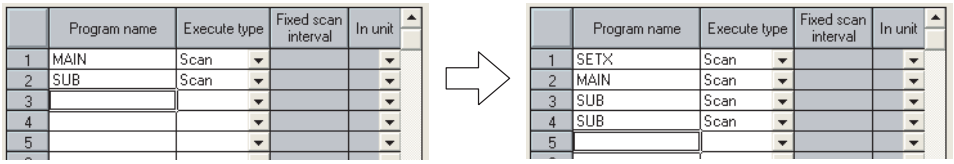
(b) Universal model QCPU

The external input/output forced on/off function is not supported in the Universal model QCPU. Note Appx.5

External input/output can be forcibly turned on/off by using the replacement program described below.

(2) Method of replacing High Performance model QCPU with Universal model QCPU

As shown in below, add programs, "SETX" and "SETY", in the Program tab of the PLC parameter dialog box.



The following table shows the program setting of the "SETX" and "SETY".

Program name	Execution type	Position where program is added
SETX	Scan	Start of Program setting (No.1)
SETY	Scan	End of Program setting

Note Appx.5 Universal

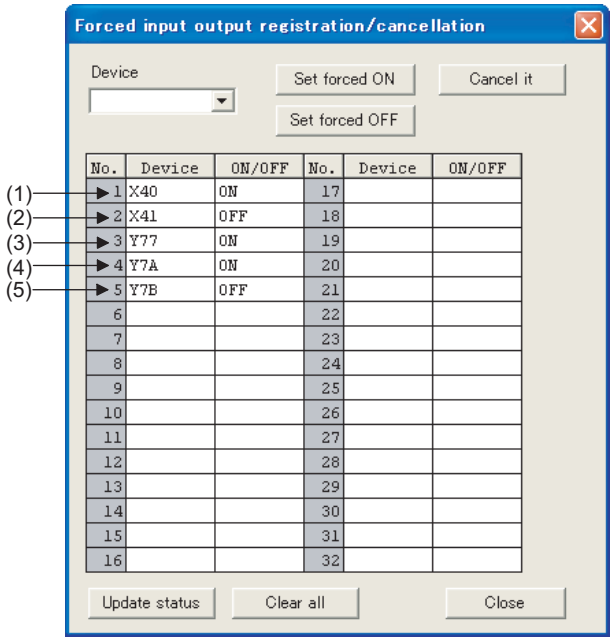
Before executing this function with the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and GX Developer used.

(Page 466, Appendix 2)

Ex. Forcibly turning X40, X77, and X7A on, and X41 and Y7B off

The programs, "SETX" and "SETY", turns on or off the X and Y devices, which have been registered for forced on/off using the external input/output forced on/off function, at each scan using the SET and RST instructions.

High Performance model QCPU



Universal model QCPU

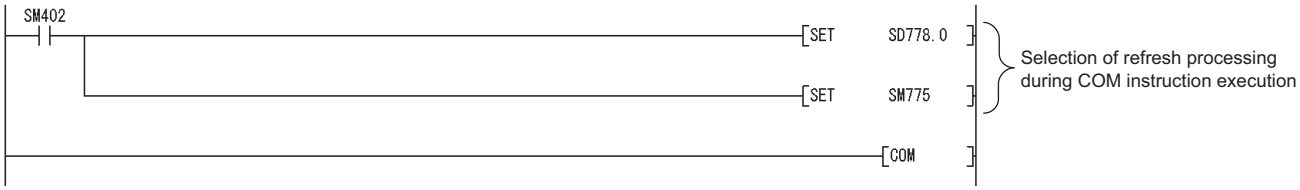
• Program example of "SETX"

• Program example of "SETY"

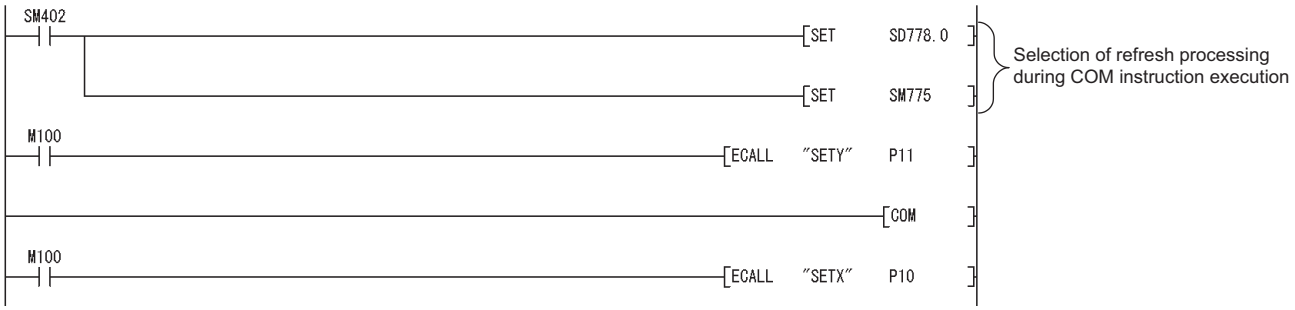
(3) Replacing the COM instruction

If the COM instruction is used, add subroutine calls for P10 and P11 before and after the COM instruction. (P10 and P11 are pointers shown in the program examples in (2).) When SM775 is on (Executes refresh set by SD778) and also the 0 bit of SD778 is off (Do not execute I/O refresh), replacement of the instruction is not necessary.

(a) Program before replacement



(b) Program after replacement



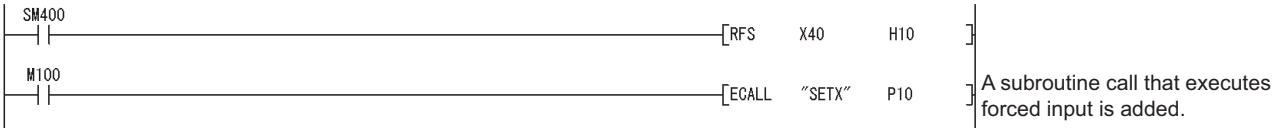
(4) Replacing the RFS instruction

If any I/O numbers targeted for forced on/off are included in the partial refresh range specified by the RFS instruction, add subroutine calls for P10 and P11 before and after the RFS instruction. (P10 and P11 are pointers shown in the program examples in (2).)

If no I/O number targeted for forced on/off is included, addition of subroutine calls for P10 and P11 is not necessary.

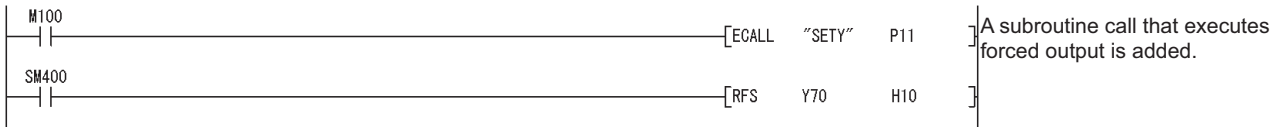
(a) When partial refresh for input (X) is executed by the RFS instruction

Add a subroutine call that executes forced input after the RFS instruction.



(b) When partial refresh for output (Y) is executed by the RFS instruction

Add a subroutine call that executes forced output before the RFS instruction.



(5) Restrictions

Replacements described in (2) to (4) do not apply in the following cases.

- Input and output targeted for forced on/off are referred to or changed using the direct input device (DX) and direct output device (DY).
- Input and output targeted for forced on/off are referred to or changed within an interrupt program.

Appendix 5.5 Special Relay and Special Register

The Universal model QCPU does not support the use of the special relay and special register described in Page 563, Appendix 5.5.1 and Page 566, Appendix 5.5.2.
Replace them using the method described in the table or delete the corresponding sections.

Appendix 5.5.1 Special relay list

The following table lists the special relay not supported by the Universal model QCPU and measures to be taken.

Number	Name/Description		Measures
SM80	CHK detection		The Universal model QCPU does not support the CHK instruction. For the replacing method of the CHK instruction, refer to Page 530, Appendix 5.3.3 (4).
SM91	Step transition monitoring timer start		The Universal model QCPU does not support the step transition monitoring timer function. For the replacing method of this function, refer to Appendix 3 "Restrictions on Basic Model QCPU, Universal Model QCPU, and LCPU and Alternative Methods" in the MELSEC-Q/L/QnA Programming Manual (SFC).
SM92			
SM93			
SM94			
SM95			
SM96			
SM97			
SM98			
SM99			
SM250	Largest mounted I/O number read		Operation of SD250 is not necessary. The Universal model QCPU always stores the largest mounted I/O number in SD250. Delete the corresponding sections.
SM255	MELSECNET/H module 1 information	Indicates regular network or standby network.	These are special relays for the simple dual-structured network function. Since the Universal model QCPU does not support this function, there is no application for these special relays. Delete the corresponding sections.
SM256		At refresh from link module to CPU, selects whether to read data from the link module.	
SM257		At refresh from CPU to link module, selects whether to write data to the link module.	
SM260	MELSECNET/H module 2 information	Indicates regular network or standby network.	
SM261		At refresh from link module to CPU, selects whether to read data from the link module.	
SM262		At refresh from CPU to link module, selects whether to write data to the link module.	
SM265	MELSECNET/H module 3 information	Indicates regular network or standby network.	
SM266		At refresh from link module to CPU, selects whether to read data from the link module.	
SM267		At refresh from CPU to link module, selects whether to write data to the link module.	
SM270	MELSECNET/H module 4 information	Indicates regular network or standby network.	
SM271		At refresh from link module to CPU, selects whether to read data from the link module.	
SM272		At refresh from CPU to link module, selects whether to write data to the link module.	
SM280	CC-Link error		Replace the relay with the I/O signals (Xn0, Xn1, and XnF) of the mounted CC-Link module.
SM315	Communication reserved time delay enable/disable flag		Set a service processing time value in the PLC system tab of the PLC parameter dialog box.

Number	Name/Description	Measures
SM330	Operation mode for low-speed execution type program	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections.
SM331	Normal SFC program execution status	The Universal model QCPU supports only normal SFC programs. Delete SM331 and SM332, which are used as interlocks, or replace them with SM321.
SM332	Program execution management SFC program execution status	
SM390	Access execution flag	Modify the program that Module ready signal (Xn) is used as an interlock, according to sample programs described in the manual for each module.
SM404	ON for only 1 scan after RUN of low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections or replace them with the special relays for scan execution type programs (SM402 and SM403).
SM405	Off for only 1 scan after RUN of low-speed execution type programs	
SM430	User timing clock No.5 (for low-speed execution type programs)	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections or replace them with the special relays for scan execution type programs (SM420 and SM424).
SM431	User timing clock No.6 (for low-speed execution type programs)	
SM432	User timing clock No.7 (for low-speed execution type programs)	
SM433	User timing clock No.8 (for low-speed execution type programs)	
SM434	User timing clock No.9 (for low-speed execution type programs)	
SM510	Low-speed execution type program executing flag	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections.
SM551	Module service interval time read	The Universal model QCPU does not support the service interval measurement function. Delete the corresponding sections.
SM580	Program to program I/O refresh	Perform I/O refresh at the start or end of each program with the RFS or COM instruction.
SM660	Boot operation	Move files with a boot setting (from the standard ROM or a memory card to the program memory) to the program memory.
SM672	Memory card file register access range flag	When outside the range of the file register in the memory card is accessed, the Universal model QCPU detects "OPERATION ERROR" (error code: 4101). Programming for detecting errors using this special relay is not necessary. Delete the corresponding sections.
SM710	CHK instruction priority flag	The Universal model QCPU does not support the CHK instruction. For the replacing method of the CHK instruction, refer to Page 530, Appendix 5.3.3 (4).
SM734	XCALL instruction execution condition specification	The Universal model QCPU executes the XCALL instruction on the rising edge of execution condition as well. There is no application for this special relay. Delete the corresponding sections.
SM735	SFC comment readout instruction in-execution flag	The Universal model QCPU does not support the following instructions: • SFC step comment readout instruction (S(P).SFCSCOMR) • SFC transition condition comment readout instruction (S(P).SFCTCOMR) Delete the corresponding sections.

Number	Name/Description	Measures
SM1780*1	Power supply off detection flag	The Universal model QCPU does not store redundant power supply system information in SM1780 to SM1783. Delete the corresponding sections. (SM1780 to SM1783 are always off.)
SM1781*1	Power supply failure detection flag	
SM1782*1	Momentary power failure detection flag for power supply 1	
SM1783*1	Momentary power failure detection flag for power supply 2	

*1 The special relay can be used if the serial number (first five digits) of the Universal model QCPU is "10042" or later.

Appendix 5.5.2 Special register list

The following table lists the special register not supported by the Universal model QCPU and measures to be taken.

Number	Name/Description	Measures
SD80	CHK number	The Universal model QCPU does not support the CHK instruction. For the replacing method of the CHK instruction, refer to Page 530, Appendix 5.3.3 (4).
SD90	Step transition monitoring timer setting value	The Universal model QCPU does not support the step transition monitoring timer function. For the replacing method of this function, refer to Appendix 3 "Restrictions on Basic Model QCPU, Universal Model QCPU, and LCPU and Alternative Methods" in the MELSEC-Q/L/QnA Programming Manual (SFC).
SD91		
SD92		
SD93		
SD94		
SD95		
SD96		
SD97		
SD98		
SD99		
SD130 to SD137	Fuse blown module	Replace SD130 to SD137 with SD1300 to SD1307.
SD150 to SD157	I/O module verify error	Replace SD150 to SD157 with SD1400 to SD1407.
SD245	No. of base slots (Mounting status)	Replace SD245 and SD246 with SD243 and SD244, respectively.
SD246		
SD280	CC-Link error	Replace these registers with the I/O signals (Xn0, Xn1, and XnF) of the mounted CC-Link module.
SD281		
SD315	Time reserved for communication processing	Service processing setting is available for the Universal model QCPU on the PLC system setting tab of the PLC parameter dialog box. Select "Specify service process time." for the service processing setting parameter and set the service processing time. Other setting methods can be selected as well.
SD394	CPU mounting information	• Check the type and model of other CPU modules mounted on the System monitor screen of GX Developer. • Check the mounting status of other CPU modules in SD396 to SD398.
SD430	Low-speed scan counter	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections or replace them with the special register for scan execution type programs (SD420).
SD510	Low-speed execution type program number	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections or replace them with the special register for scan execution type programs (SD500).
SD528	Current scan time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections or replace them with the special register for scan execution type programs (SD520 and SD521).
SD529		
SD532	Minimum scan time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections or replace them with the special registers for scan execution type programs (SD524 to SD527).
SD533		
SD534	Maximum scan time for low-speed execution type programs	
SD535		
SD544	Cumulative execution time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections.
SD545		
SD546	Execution time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs. Delete the corresponding sections.
SD547		

Number	Name/Description	Measures
SD550	Service interval measurement module	The Universal model QCPU does not support the service interval measurement function. Delete the corresponding sections.
SD551	Service interval time	
SD552		
SD720	Program No. specification for PLAODP instruction	The Universal model QCPU does not support the PLAODP instruction. Delete the corresponding sections.
SD1780*1	Power supply off detection status	The Universal model QCPU does not store redundant power supply system information in SD1780 to SD1783. Delete the corresponding sections. (SD1780 to SD1783 are always off.)
SD1781*1	Power supply failure detection status	
SD1782*1	Momentary power failure detection counter for power supply 1	
SD1783*1	Momentary power failure detection counter for power supply 1	

*1 The special register can be used if the serial number (first five digits)of the Universal model QCPU is "10042" or later.

Appendix 6 Precautions for Replacing QnUD(E)(H)CPU with QnUDVCPU/QnUDPVCPU

Appendix 6.1 Precautions

(1) System configuration

Item	Precautions	Replacement method	Reference
RS-232 port	There is no RS-232 port.*1	Use a USB or Ethernet port. To communicate with an RS-232 interface, use the QJ71C24N(-R2) in the system.	---
Applicable products and software	• A programming tool that was available for the QnUD(E)(H)CPU can no longer be used or needs to be upgraded. (The use of GX Developer is not supported in the system after replacement.) • Some GOTs and intelligent function modules that were available for the QnUD(E)(H)CPU can no longer be used or need to be upgraded.	• Upgrade the version of GX Works2 to the one compatible with the QnUDVCPU and QnUDPVCPU. • Replace the GOT and intelligent function modules with those compatible with the QnUDVCPU and QnUDPVCPU.	Page 514, Appendix 5.2
Multiple CPU system	Scan time is shortened in the High-speed Universal model QCPU and Universal model Process CPU because operations are performed at higher speed. When used in a multiple CPU system, the High-speed Universal model QCPU and Universal model Process CPU access to the other modules frequently. As a result, the processing time in other CPU modules may increase.	Check the processing timings of other CPU modules and adjust the access frequency of the High-speed Universal model QCPU and Universal model Process CPU using timers or the constant scan function.	QCPU User's Manual (Multiple CPU System)
Current consumption	The current consumption increases.	Select a power supply module according to the total current consumption in the system.	QCPU User's Manual (Hardware Design, Maintenance and Inspection)

*1 This applies when the QnUD(H)CPU is replaced with the High-speed Universal model QCPU.

(2) Program

Item	Precautions	Replacement method	Reference
Number of steps	The number of basic steps differs in some instructions.	If index modifications mentioned on the left are frequently used in the program, the program size may exceed the storage capacity of the replaced CPU module. After the program controller type is changed, check the program size using the confirm memory size function. If the program size exceeds the storage capacity, take the following actions or change the CPU module to that with larger program memory. • Move parameters and device comments to the standard ROM. • Reduce the reserved area for online change. • When a file register, extended data register, or extended link register with device numbers smaller then 64K words is used, there is no increase in the number of steps. Make adjustments so that device numbers smaller then 64K words are used.	---
	The number of steps increases by one when: • Index modification is performed.*1*2*3 • A leading or trailing edge instruction is used. • Device numbers equal to or greater than 64K words are used in a file register, extended data register, or extended link register.*4*5 • Bit devices are used as word data by specifying digits using K1, K2, K3, K5, K6, or K7, or by specifying a device number of other than multiples of 16.		

*1 When device numbers equal to or greater than 64K words are used and index modification is performed, there will be no increase in the number of steps.

*2 When the OUT instruction is used with a timer and counter and index modification is performed, the number of steps will increase by three.

*3 When the following instructions are used and index modification is performed on the destination, the number of steps will increase by two.

Instruction name
+, - (2 devices)
D+, D (2 devices)
E+, E- (2 devices)
INC, DEC
DINC, DDDEC
NEG
DNEG
WAND, WOR, WXOR, WXNR (2 devices)
DAND, DOR, DXOR, DXNR (2 devices)
BSET, BRST

*4 Even when index modification is specified and a device number equal to or greater than 64K words is specified, there will be no increase in the number of steps.

*5 When a device number equal to or greater than 64K words is used for the destination, the number of steps will increase by two.

(3) Parameter size

Item	Precautions	Replacement method	Reference
Parameter size	The parameter size increases because the built-in Ethernet port setting parameters are added.*1	• Delete unnecessary files and free some space. • Move the parameter file to another memory area.	---

*1 This applies when the QnUD(H)CPU is replaced with the High-speed Universal model QCPU.

(4) Drive and file

Item	Precautions	Replacement method	Reference
Boot file setting	A memory card (SRAM card, ATA card, or Flash card) cannot be specified as a transfer source.	Specify an SD memory card as a transfer source.	Page 104, Section 2.11
Device comment	A device comment file cannot be stored in an SRAM card.	Store the file in the standard RAM.	---
	A device comment file cannot be stored in an ATA card nor Flash card.	Store the file in an SD memory card.	---
Initial device value	An initial device value file cannot be stored in an SRAM card.	Store the file in the standard RAM or standard ROM.	Page 247, Section 3.25
	An initial device value file cannot be stored in an ATA card nor Flash card.	Store the file in an SD memory card.	
Local device	A local device file cannot be stored in an SRAM card.	• Store the file in the standard RAM. • If the size of the local device file exceeds the standard RAM capacity, consider the use of an extended SRAM cassette.	Page 422, Section 6.2
File register	A file register file cannot be stored in an SRAM card.	• Store the file in the standard RAM. • If the size of the file register file exceeds the standard RAM capacity, consider the use of an extended SRAM cassette.	Page 393, Section 4.7.1
	A file register file cannot be stored in a Flash card. (Sequence programs only can read file register data in a Flash card.)[*1]	Use the initial device value file in an SD memory card or the FREAD/FWRITE instructions.	Page 247, Section 3.25, MELSEC-Q/L Programming Manual (Common Instruction)
Sampling trace	A sampling trace file cannot be stored in an SRAM card.	• Store the file in the standard RAM. • If the size of the sampling trace file exceeds the standard RAM capacity, consider the use of an extended SRAM cassette.	Page 184, Section 3.14 (2)
CPU module change function with memory card	A memory card cannot be specified as a backup destination or restoration source.	Specify an SD memory card as a backup destination or restoration source.	Page 260, Section 3.31

(5) Built-in Ethernet port communications*1

Item	Precautions	Replacement method	Reference
File transfer function (FTP server)	The security function has been enhanced from the password registration function to the file password 32 function. For this reason, the keyword-set subcommand, that sets/displays/clears the file access password, is no longer supported.	Use the FTP commands, passwd-rd and passwd-wr, that set/display/clear the read/write passwords of the file password 32 function.	QnUCPU User's Manual (Communication via Built-in Ethernet Port)

*1 This applies when the QnUDE(H)CPU is replaced with the High-speed Universal model QCPU.

(6) Functions

Item	Precautions	Replacement method	Reference
Security function	The security function, that limits accesses to the files in the CPU module, has been enhanced from the password registration function to the file password 32 function.	Use the file password 32 function instead of the password registration function.	Page 207, Section 3.19
Latch data backup to standard ROM	If an extended SRAM cassette is used and the memory capacity of the standard RAM (drive 3) is larger than that of the standard ROM, data cannot be backed up using this function.	Deselect the "Backup all files in the internal of standard RAM" checkbox in the PLC System tab of the PLC parameter dialog box.	Page 254, Section 3.29 (1)
Battery life-prolonging function	The battery life-prolonging function is no longer supported. Without the use of the function, the battery life is as same as that of the QnUD(E)(H)CPU.	The switch setting parameters are ignored and the following operations are performed. • Data held by the battery are not cleared nor deleted. • The bits, b0 and b1, of SD119 (Battery life-prolonging factor) are fixed to 0.	Page 250, Section 3.26, QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Appendix 7 Precautions for Replacing QnPHCPU with QnUDPVCPU

For the precautions for replacing the QnPHCPU with the QnUDPVCPU, refer to the following.

Technical bulletin: No. FA-A-0155 (Method of replacing Process CPU with Universal model Process CPU)

Appendix 8 Precautions for Using GX Works2 and Differences with GX Developer

For the precautions for using GX Works2 and differences with GX Developer, refer to the following.

GX Works2 Version 1 Operating Manual (Common)

Appendix 9 Ways to Use Different Types of the Backup/restoration Function

The following table lists the backup/restoration functions. GOT also has the backup/restoration function. Application of each function, executable operating status, and target data are described as follows.

Item	Function	Application	Executable operating status				Target data
			Backup		Restoration		
			RUN	STOP	RUN	STOP	
CPU module	CPU module change function with memory card	This function is used when a CPU module is replaced due to a failure or other reasons. The function backs up the data in the CPU module at a failure, and after another CPU module is mounted on, restores the data to the replaced CPU module. The operation of this function is performed while the CPU module is stopped. For the device data, recovery target is only file register files or devices that the latch range is specified. One backup data can be stored.	×	○	×*1	○	Data in the CPU module • Drive 0 (such as programs and parameters) • Drive 3 (such as file register files) • Drive 4 (such as files for storing device data) • Device data (only the latch relay (L) and devices that the latch range is set)
	CPU module data backup/restoration function	This function is used when the CPU module is created without using a personal computer. The data in the CPU module is backed up periodically, and the data can be restored depending on the circumstance such as recovering from an error state of programs or device data. Note that the data protected by the security functions cannot be backed up or restored. All the device data in the CPU module including file register files that the latch range is specified. Only the device data can be restored. Multiple backup data can be stored.	○	○	×*1	○	Data in the CPU module • Drive 0 (such as programs and parameters) • Drive 3 (such as file register files) • Drive 4 (such as files for storing device data) • Device data (All devices)
	Data backup/restoration (iQ Sensor Solution)	This function is used when the same device supporting iQ Sensor Solution is created without using a personal computer. The function backs up the setting data such as parameters of the device supporting iQ Sensor Solution, and after a changeover, restores the data to the device supporting iQ Sensor Solution. Therefore, the process of setting change can be simplified.	○	○	○	○	Setting data of the device supporting iQ Sensor Solution on the following networks. • AnyWireASLINK • CC-Link • CC-Link IE Field Network • Ethernet (only when connected by built-in Ethernet)
GOT*2	Backup/restoration function	This function is used to when the same system is created without using a personal computer. The function backs up setting data, including programs, parameters, and setting values, for a controller connected to the GOT to a Compact Flash card or USB memory in the GOT periodically. The data backed up can be restored the controller. The security function using passwords can be added to the setting information of controllers. The target device data is only file register files. Multiple backup data can be stored.	○	○	○	○	Data in the CPU module connected to GOT • Drive 0 (such as programs and parameters) • Drive 1 (Programmable controller • user data) • Drive 3 (such as file register files) • Drive 4 (such as files for storing device data)

*1 Automatic restoration is operated when the CPU module is powered off and then on or reset.

*2 For details on the backup/restoration function for GOT, refer to the corresponding manuals of GOT.

Appendix 10 Device Point Assignment Sheet

Device name	Symbol	Numeric notation	Number of device points[*2]		Restriction check	
			Points	Range	Size (words)[*3]	Points (bits)[*2]
Input relay[*1]	X	Hexadecimal	8K (8192)	X0000 to X1FFF	/16 512	×1 8192
Output relay[*1]	Y	Hexadecimal	8K (8192)	Y0000 to Y1FFF	/16 512	×1 8192
Internal relay	M	Decimal	K ( )	M0 to	/16	×1
Latch relay	L	Decimal	K ( )	L0 to	/16	×1
Link relay	B	Hexadecimal	K ( )	B0000 to	/16	×1
Annunciator	F	Decimal	K ( )	F0 to	/16	×1
Link special relay	SB	Hexadecimal	K ( )	SB0000 to	/16	×1
Edge relay	V	Decimal	K ( )	V0 to	/16	×1
Step relay[*1]	S	Decimal	8K (8192)	S0 to S8191	/16 512	×1 8192
Timer	T	Decimal	K ( )	T0 to	$\times\frac{18}{16}$	×2
Retentive timer	ST	Decimal	K ( )	ST0 to	$\times\frac{18}{16}$	×2
Counter	C	Decimal	K ( )	C0 to	$\times\frac{18}{16}$	×2
Data register	D	Decimal	K ( )	D0 to	×1	
Link register	W	Hexadecimal	K ( )	W0000 to	×1	
Link special register	SW	Hexadecimal	K ( )	SW0000 to	×1	
Total					(29696 or less)	(65536 or less)

*1 The points are determined by the system (cannot be changed). The points for the step relay can be changed to 0K for the Universal model QCPU whose serial number (first five digits) is "10042" or later. For the Universal model QCPU whose serial number (first five digits) is "12052" or later, a step relay can be set in increments of 1K point and up to the following points. (Page 466, Appendix 2)

- Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 8192 points
- Universal model QCPU other than the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU: 16384 points

*2 Up to 32K points can be set for each device. However, up to 60K points can be set for each device of the internal relay and link relay for the Universal model QCPU whose serial number (first five digits) is "10042" or later.

*3 Enter the values multiplied (or divided) by the number shown in the Size (words) column.

INDEX

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

REVISIONS

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

Print date	*Manual number	Revision
Dec., 2008	SH(NA)-080807ENG-A	First edition
Mar., 2009	SH(NA)-080807ENG-B	Revision on the new functions of the Universal model QCPU whose serial number (first 5 digits) is "11012" or later Partial correction SAFETY PRECAUTIONS, INTRODUCTION, MANUALS, MANUAL PAGE ORGANIZATION, GENERIC TERMS AND ABBREVIATIONS, Section 1.3, 1.6, 2.2.2, 2.2.3, 2.3, 2.3.3, 2.3.4, 2.4, CHAPTER 3, Section 3.3, CHAPTER 4, Section 4.1.2, 4.2.2, 4.2.3, 5.1.1, 5.1.3, 5.1.6, 5.1.7, 5.1.8, 5.1.10, 6.1, 6.3, 6.4, 6.5, 6.6.1, 6.6.5, 6.11.1, 6.11.3, 6.11.4, 6.12.1, 6.13.3, 6.14, 6.15, 6.15.1, 6.15.2, 6.16, 6.17, 6.18, 6.20, 6.28, 6.30, 7.1.2, 8.2, 8.3, 9.2, 9.2.5, 9.2.11, 9.7.4, 9.11, 9.14.2, 11.5, Appendix 1, Appendix 2, Appendix 3.1, Appendix 3.3.2, Appendix 4 Addition Appendix 3.4.2
Jul., 2009	SH(NA)-080807ENG-C	Revision on the new functions of the Universal model QCPU whose serial number (first 5 digits) is "11043" or later Partial correction Section 5.1.1, 5.1.5, 5.3.3, 6.1, 6.18, 6.21.2, 6.28, 8.1, 9.2.10, 9.6.1, 10.1.3, 12.2, Appendix 1, Appendix 2, Appendix 3.1.2, Appendix 3.2 Addition Section 6.31, 6.32
Nov., 2009	SH(NA)-080807ENG-D	Partial correction SAFETY PRECAUTIONS, Section 4.2.2, 9.6.1, 9.7.4, Appendix 3.1.1, Appendix 3.1.2 Addition CONDITIONS OF USE FOR THE PRODUCT
Apr., 2010	SH(NA)-080807ENG-E	Revision on the new models and new functions of the Universal model QCPU whose serial number (first 5 digits) is "12012" or later Model addition Q50UDEHCPU, Q100UDEHCPU Partial correction INTRODUCTION, MANUALS, GENERIC TERMS AND ABBREVIATIONS, Section 1.5, 2.1, 2.2.1, 2.2.3, 2.3.4, 2.4, 3.1, 3.5, 3.7, 3.8, 3.8.1, 3.8.2, 4.2.1, 5.1.1, 5.1.3, 5.3.3, 5.3.4, 6.3, 6.5, 6.6, 6.6.1, 6.6.2, 6.6.3, 6.6.4, 6.9, 6.11.2, 6.11.3, 6.12.2, 6.12.3, 6.13.1, 6.14, 6.15, 6.23, 6.24, 6.26, 6.29, 6.30, 6.30.1, 6.31, 7.1.4, 7.1.5, 8.1, 8.2, 9.1, 9.2, 9.2.1, 9.2.5, 9.2.7, 9.2.8, 9.2.12, 9.2.13, 9.2.14, 9.3.1, 9.4, 9.5.1, 9.6.1, 9.6.2, 9.7, 9.7.2, 9.7.4, 9.8, 9.9, 9.10, 9.12.1, 9.14.2, 10.1.1, 10.1.2, 10.1.3, 11.1, 11.3, 11.4, 11.5, Appendix 1, Appendix 2, Appendix 3.1.1, Appendix 3.1.2, Appendix 3.3.1, Appendix 3.5.1, Appendix 3.5.2, Appendix 4 Addition Section 6.33
Aug., 2010	SH(NA)-080807ENG-F	Revision on the new functions of the Universal model QCPU whose serial number (first 5 digits) is "12052" or later Partial correction SAFETY PRECAUTIONS, Section 2.1, 2.4.4, 5.1.8, 6.1, 6.30.1, 6.30.2, 8.1, 9.1, 9.2, 9.2.1, 10.1.2, 10.1.3, Appendix 2, Appendix 4 Addition Section 6.34

Print date	*Manual number	Revision
Jan., 2011	SH(NA)-080807ENG-G	Partial correction SAFETY PRECAUTIONS, Section 5.3.3, 6.24.1, 12.1, 12.2, Appendix 2
May, 2011	SH(NA)-080807ENG-H	Partial correction GENERIC TERMS AND ABBREVIATIONS, Section 3.8.1, 3.8.2, 5.1.1, 5.2, 6.1, 6.24.1, 6.27, 9.2.10, 9.11, Appendix 2 Addition Section 6.28 Deletion Chapter 12
Jul., 2011	SH(NA)-080807ENG-I	Revision due to the layout change of the manual
Oct., 2011	SH(NA)-080807ENG-J	Revision on the new functions of the Universal model QCPU whose serial number (first 5 digits) is "13102" or later Partial correction TERMS, Section 1.7,2.3.2, 2.7, 2.8, 3.1, 3.3, 3.4, 3.7, 3.10, 3.12.3, 4.2, 4.2.10, 4.6.1, 4.7, 4.7.4, 4.8, 4.11, 5.4, Appendix 2, 3.2, 4.5, 5.1.1, 5.2, 5.3.1, 5.4.4 Addition Section 2.14
Feb., 2012	SH(NA)-080807ENG-K	Revision on the new functions of the Universal model QCPU whose serial number (first 5 digits) is "14022" or later Partial correction Section 2.1.1, 3.1, 3.11.1, 3.12.3, 3.15.2, 3.27, 3.29, 4.1, 4.2, Appendix 1.1, Appendix 1.2.8, Appendix 1.3.2, Appendix 1.2.11, Appendix 2, Appendix 5.1.2
May, 2012	SH(NA)-080807ENG-L	Motion CPU model addition Revision on the new functions of the Universal model QCPU whose serial number (first 5 digits) is "14042" or later Model addition Q172DCPU-S1, Q173DCPU-S1, Q172DSCPU, Q173DSCPU Partial correction TERMS, Section 2.11, 3.27, Appendix 1.1, 1.3, 1.3.1, 1.3.2, 2, 5.2
Aug., 2012	SH(NA)-080807ENG-M	Revision on the new function of the Universal model QCPU whose serial number (first 5 digits) is "14072" or later Partial correction Section 3.1, Appendix 2, 5.1.2, 5.2
Feb., 2013	SH(NA)-080807ENG-N	Revision on the new models of the Universal model QCPU Model addition Q03UDVCPU, Q04UDVCPU, Q06UDVCPU, Q13UDVCPU, Q26UDVCPU
Sep., 2013	SH(NA)-080807ENG-O	Revision on the new function of the Universal model QCPU whose serial number (first 5 digits) is "15043" or later Partial correction Section 2.7, 3.12.3, Appendix 2, 5.1.2
Jan., 2014	SH(NA)-080807ENG-P	Revision on the new function of the Universal model QCPU whose serial number (first 5 digits) is "15103" or later Partial correction Section 2.1.1, 2.1.3, 3.1, 3.7, 3.8, 3.9, 3.10, Appendix 1.2, 2
Jul., 2014	SH(NA)-080807ENG-Q	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "16043" or later Partial correction TERMS, Section 2.3.2, 2.8, 3.1, Appendix 2, 3.2, 5.4.3, 6.1

Print date	*Manual number	Revision
Mar., 2015	SH(NA)-080807ENG-R	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "17012" or later Model addition NZ1MEM-2GBSD, NZ1MEM-4GBSD, NZ1MEM-8GBSD, NZ1MEM-16GBSD Partial correction MANUALS, TERMS, Section 2.1.1, 2.1.3, 2.3.2, 2.11, 3.1, 3.20.2, 3.29, 3.31.1, 3.31.2, Appendix 2, 3.2, 3.3, 5.1, 5.2, 7.1 Addition Section 3.36
Jun., 2015	SH(NA)-080807ENG-S	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "17052" or later Partial correction TERMS, Section 2.1.1, 3.29, Appendix 1.1, 1.2.11, 2, 5.1.1, 5.1.2
Dec., 2015	SH(NA)-080807ENG-T	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "17103" or later Partial correction Section 2.1.1, 2.1.3, 3.1, 3.31, 3.34, Appendix 2 Addition Section 3.32, Appendix 8
Jul., 2016	SH(NA)-080807ENG-U	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "18052" or later Partial correction MANUALS, TERMS, Section 2.5, 3.1, 3.13.3, 3.29, 3.32, 3.32.1, 3.32.2, 4.6.3, 6.2, Appendix 2, 3.2, 5, 5.2
Oct., 2016	SH(NA)-080807ENG-V	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "18072" or later Partial correction Section 2.1.1, 3.1, 3.2, 3.32.2, 3.37, Appendix 2
Jan., 2017	SH(NA)-080807ENG-W	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "18112" or later Partial correction MANUALS, Section 2.4.4, 2.5, 2.8, 2.8.1, 3.1, 3.11.3, 3.19, 3.31.1, 4.2.1, 4.8, Appendix 1.1, 1.2, 2, 3.2
Apr., 2017	SH(NA)-080807ENG-X	Partial correction Section 3.31.1, 3.32
Aug., 2017	SH(NA)-080807ENG-Y	Revision on the new function of the High-speed Universal model QCPU whose serial number (first 5 digits) is "19062" or later Partial correction Section 2.1.1, 2.1.3, 3.1, 3.23, 3.32, Appendix 1.1, 1.2.4, 1.2.8, 2 Addition Section 3.37, 3.38
Dec., 2017	SH(NA)-080807ENG-Z	Partial correction Section 3.6.1, 3.6.2, 3.6.3, 3.6.4, 3.19.2, 3.23, 3.24.1, 3.37.2, Appendix 2

Print date	*Manual number	Revision
Sep., 2018	SH(NA)-080807ENG-AA	Revision on the new function of the High-speed Universal model QCPU and Universal model Process CPU whose serial number (first 5 digits) is "19062" or later Model addition Q04UDPVCPU, Q06UDPVCPU, Q13UDPVCPU, Q26UDPVCPU Partial correction Section 2.8.1, 2.8.2, 3.1, 3.31.1, 3.31.2, Appendix 1.1, 1.2.11, 2
Dec., 2018	SH(NA)-080807ENG-AB	Partial correction TERMS, Section 2.1.1, 2.1.3, Appendix 2
Apr., 2019	SH(NA)-080807ENG-AC	Partial correction TERMS

Japanese manual version SH-080802-AF

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.

SH(NA)-080807ENG-AC(1904)MEE
MODEL: QNUCPU-U-KP-E
MODEL CODE: 13JZ27

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