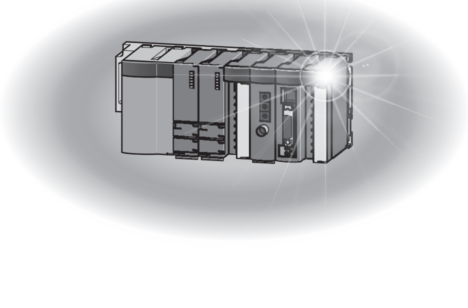
MITSUBISHI
ELECTRIC

Programmable Controller

MELSEC Q series

Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)

-Q00JCPU
-Q00CPU
-Q01CPU
-Q02CPU
-Q02HCPU
-Q06HCPU
-Q12HCPU
-Q25HCPU
-Q02PHCPU
-Q06PHCPU
-Q12PHCPU
-Q25PHCPU
-Q12PRHCPU
-Q25PRHCPU



●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.

Status	Q series model	AnS/A series model
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 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.

● 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.

[Design Precautions]

⚠ WARNING

- 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, in the case of a control from an external device to a remote programmable controller, immediate action cannot be taken for a problem on 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 located 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.

- Shut off the external power supply for the system in all phases before mounting or removing the 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.
  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

- Ground the FG and LG terminals to the protective ground conductor dedicated to the programmable controller.
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.
- 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.
Failure to do so may result in malfunction due to noise.
- 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.

[Wiring Precautions]

⚠ CAUTION

- 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.

- 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.
  Doing so will cause the battery to produce heat, explode, or ignite, 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.
  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.

[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.

- ● Shut off the external power supply for the system in all phases before mounting or removing the 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
  - Mounting/removing the terminal block to/from the module

- ● 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.

REVISIONS

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

Print date	Manual number	Revision
Dec., 2008	SH(NA)-080808ENG-A	First edition
Mar., 2009	SH(NA)-080808ENG-B	Partial correction SAFETY PRECAUTIONS, INTRODUCTION, MANUALS, MANUAL PAGE ORGANIZATION, Section 1.3, 1.6, 2.2.2, 2.2.3, 2.3, 2.3.4, 2.3.5, 3.3, 3.7, 3.8.1, 3.8.2, 4.1.1, 4.1.2, 4.2.2, 4.2.3, 5.1.1, 5.2.1, 5.2.5, 5.2.6, 5.2.8, 6.1, 6.3, 6.5, 6.6.1, 6.6.5, 6.11.3, 6.12.2, 6.13.3, 6.14, 6.15.2, 6.20, 6.22.2, 6.22.4, CHAPTER 7, Section 7.1.2, 7.2, 8.2, 9.1, 9.2, 9.2.11, 9.3.3, 9.5.2, 9.7.4, 9.10, 10.1.1, 10.1.2, Appendix 1, Appendix 3
Nov., 2009	SH(NA)-080808ENG-C	Partial correction SAFETY PRECAUTIONS, Section 4.2.2, 9.2.10, 9.7.4 Addition CONDITIONS OF USE FOR THE PRODUCT
May, 2010	SH(NA)-080808ENG-D	Partial correction SAFETY PRECAUTIONS, MANUALS, Section 1.5.1, 1.5.2, 2.1, 2.2.1, 2.2.3, 2.4, 3.7, 4.2.2, 5.1.1, 5.1.4, 5.2.9, 6.2, 6.5, 6.6.2, 6.6.3, 6.6.4, 6.11.2, 6.12.2, 6.13.1, 6.14, 6.17.1, 6.22, 6.24, 7.1.3, 8.1.1, 9.2.9, 9.2.10, 9.3.1, 9.5.1, 9.7, 9.8, 9.9, 9.10.1, 9.11.1, 9.11.2, 9.12.3, 9.13.2, 10.1.2, Appendix 2.1, Appendix 2.2, Appendix 2.3, Appendix 2.4
May, 2011	SH(NA)-080808ENG-E	Partial correction SAFETY PRECAUTIONS, GENERIC TERMS AND ABBREVIATIONS, Section 3.1, 3.4, 3.6, 3.8.1, 3.8.2, 5.1.1, 5.1.2, 5.1.5, 5.2.1, 5.2.8, 5.2.9, 6.5, 6.25, 7.2, CHAPTER 8, Section 9.2, 9.2.4, 9.2.10, 9.3.2, 9.3.3, 9.10, 10.1.3 Deletion CHAPTER 12
Sep., 2011	SH(NA)-080808ENG-F	Partial correction SAFETY PRECAUTIONS, INTRODUCTION, MANUAL PAGE ORGANIZATION, Section 2.4.5, 5.2.6, 6.24, 9.2, 9.2.10, 9.12.4
Oct., 2011	SH(NA)-080808ENG-G	Partial correction GENERIC TERMS AND ABBREVIATIONS, Section 1.5.2, 3.8, 4.2.2, 6.22.2, 6.22.4, 10.1.2
May, 2012	SH(NA)-080808ENG-H	Partial correction Section 5.2.8, 6.2, 6.12.3, 8.1.1, 8.1.2, 9.1, 9.2.13, 9.3.2, 9.3.3, 9.7, 9.7.4, Appendix 1
Jan., 2014	SH(NA)-080808ENG-I	Partial correction GENERIC TERMS AND ABBREVIATIONS, Section 4.2.1, 5.2.1, 5.2.5, 5.2.10, 6.12.3, 9.1, 9.7.2, 9.7.6, 10.1.3, Appendix 2.2, 2.3
Jul. 2014	SH(NA)-080808ENG-J	Partial correction GENERIC TERMS AND ABBREVIATIONS, Section 1.5.2, 3.8, 4.2.2, 6.22.2, 6.22.4, 10.1.2

Print date	Manual number	Revision
Dec., 2015	SH(NA)-080808ENG-K	Partial correction Section 1.2, 5.2.1, 5.2.7, 5.2.8, WARRANTY
Dec., 2017	SH(NA)-080808ENG-L	Partial correction SAFETY PRECAUTIONS, GENERIC TERMS AND ABBREVIATIONS, Section 6.2, 6.13.1, 6.13.3, 6.17, 6.24, 9.13.2
Apr., 2019	SH(NA)-080808ENG-M	Partial correction MANUALS, GENERIC TERMS AND ABBREVIATIONS, Section 5.2.1, 5.4.4

Japanese manual version SH-080803-O

This manual confers no industrial property rights or any rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

INTRODUCTION

This manual describes the memory maps, functions, programs, I/O number assignment, and devices of the Q series CPU module.

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
Basic model QCPU	Q00JCPU, Q00CPU, Q01CPU
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
Redundant CPU	Q12PRHCPU, Q25PRHCPU

Remark

This manual does not describe the specifications of the power supply modules, base units, extension cables, memory cards, and batteries.
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)

For redundant systems, refer to the following.
☞ QnPRHCPU User's Manual (Redundant System)

CONTENTS

CHAPTER6 FUNCTIONS 6-1 to 6-151

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.

Number (in the list below)	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU

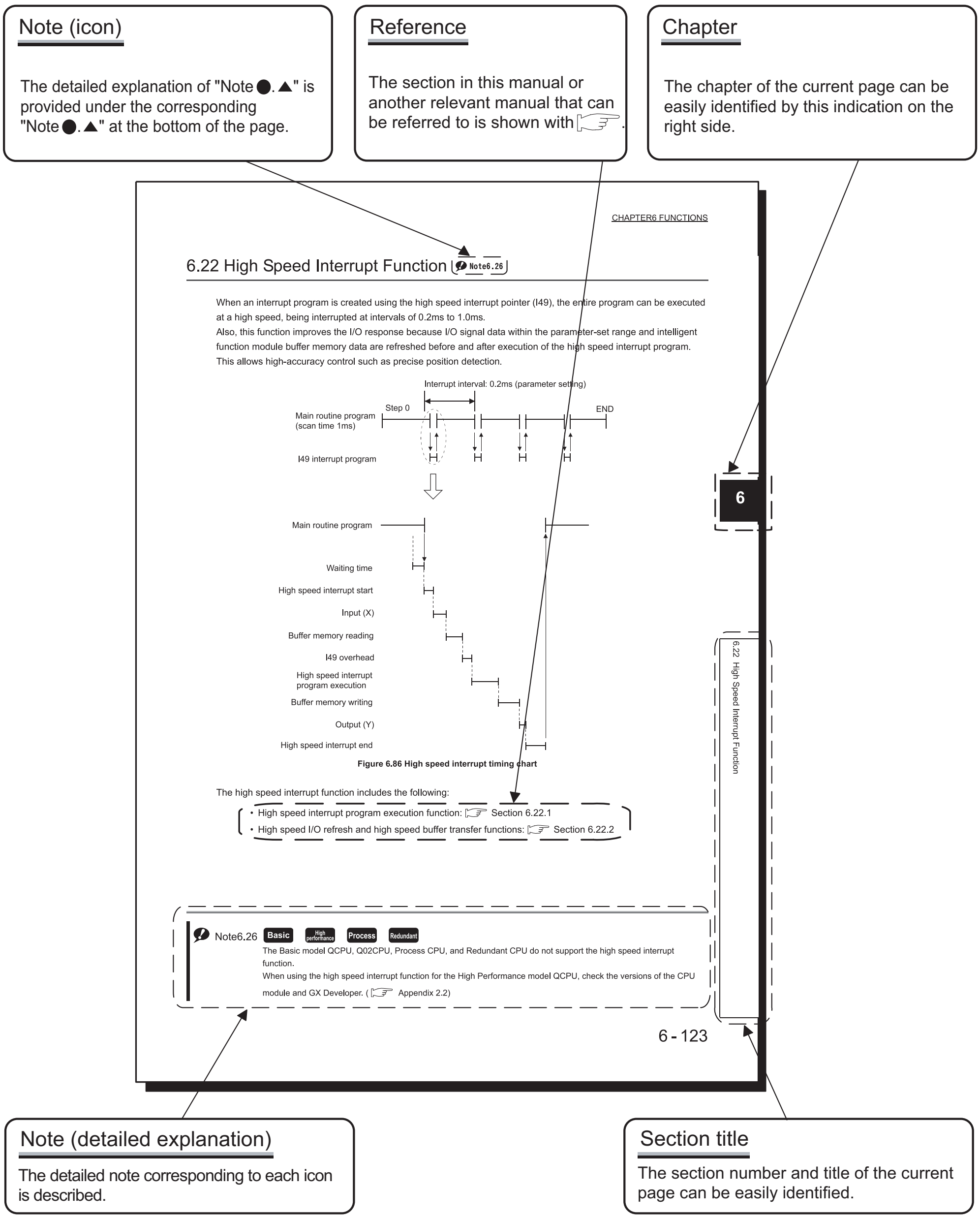
● : Basic manual, ○ : Other CPU module manuals

Manual name < Manual number (model code) >	Description	Manual type			
		1)	2)	3)	4)
● User's manual					
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, and memory cards), system maintenance and inspection, troubleshooting, and error codes	●	●	●	●
Qn(H)/QnPH/QnPRHCPUCPU Users Manual (Function Explanation, Program Fundamentals) < SH-080807ENG (13JZ27) >	Functions, methods, and devices for programming	●	●	●	●
QCPU User's Manual (Multiple CPU System) < SH-080485ENG (13JR75) >	Information for configuring a multiple CPU system (system configuration, I/O numbers, communication between CPU modules, and communication with the input/output modules and intelligent function modules)	○	○	○	
QnPRHCPU User's Manual (Redundant System) < SH-080486ENG (13JR768) >	Information on redundant system configuration (system configuration, functions, communication with external devices, and troubleshooting)				●
● Programming manual					
MELSEC-Q/L Programming Manual (Common Instruction) < SH-080809ENG (13JW10) >	How to use sequence instructions, basic instructions, and application instructions	●	●	●	●
MELSEC-Q/L/QnA Programming Manual (SFC) < SH-080041 (13JF60) >	System configuration, performance specifications, functions, programming, debugging, and error codes for SFC (MELSAP3) programs	○	○	○	○
MELSEC-Q/L Programming Manual (MELSAP-L) < SH-080076 (13JF61) >	Programming methods, specifications, and functions for SFC (MELSAP-L) programs	○	○	○	○
MELSEC-Q/L Programming Manual (Structured Text) < SH-080366E (13JF68) >	Programming methods using structured languages	○	○	○	○
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			○	○

Other relevant manuals

Manual name	Description
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
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 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
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
GX Developer Version 8 Operating Manual < SH-080373E (13JU41) >	Operating methods of GX Developer, such as programming and printout

MANUAL PAGE ORGANIZATION



Note (icon)

The detailed explanation of "Note●.▲" is provided under the corresponding "Note●.▲" at the bottom of the page.

Reference

The section in this manual or another relevant manual that can be referred to is shown with ☞.

Chapter

The chapter of the current page can be easily identified by this indication on the right side.

CHAPTER6 FUNCTIONS

6.22 High Speed Interrupt Function Note6.26

When an interrupt program is created using the high speed interrupt pointer (I49), the entire program can be executed at a high speed, being interrupted at intervals of 0.2ms to 1.0ms.
Also, this function improves the I/O response because I/O signal data within the parameter-set range and intelligent function module buffer memory data are refreshed before and after execution of the high speed interrupt program.
This allows high-accuracy control such as precise position detection.

Figure 6.86 High speed interrupt timing chart

The high speed interrupt function includes the following:

- High speed interrupt program execution function: ☞ Section 6.22.1
- High speed I/O refresh and high speed buffer transfer functions: ☞ Section 6.22.2

Note6.26 Basic High performance Process Redundant

The Basic model QCPU, Q02CPU, Process CPU, and Redundant CPU do not support the high speed interrupt function.
When using the high speed interrupt function for the High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.2)

6 - 123

6

6.22 High Speed Interrupt Function

Note (detailed explanation)

The detailed note corresponding to each icon is described.

Section title

The section number and title of the current page can be easily identified.

*The above page illustration is for explanation purpose only, and is different from the actual page.

Icons				Description
Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	
Basic	High performance	Process	Redundant	Icons indicate that specifications described on the page contain some precautions.

In addition, this manual uses the following types of explanations.

Point

In addition to description of the page, notes or functions that require special attention are described here.

Remark

The reference related to the page or useful information are described here.

GENERIC TERMS AND ABBREVIATIONS

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 Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU
Basic model QCPU	Generic term for the Q00JCPU, Q00CPU, and Q01CPU
High Performance model QCPU	Generic term for the Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
Process CPU	Generic term for the Q02PHCPU, Q06PHCPU, Q12PHCPU, and Q25PHCPU
Redundant CPU	Generic term for the Q12PRHCPU and Q25PRHCPU
Motion CPU	Generic term for 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	Generic term for MELSEC-Q series PC CPU modules, PPC-CPU852(MS)-512 manufactured by CONTEC Co., Ltd.
C Controller module	Generic term for the Q06CCPU-V, and Q06CCPU-V-B, Q12DCCPU-V, and Q24DHCCPUV C Controller modules
■ CPU module model	
QnHCPU	Generic term for the Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
Qn(H)CPU	Generic term for the Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
QnPHCPU	Generic term for the Q02PHCPU, Q06PHCPU, Q12PHCPU, and Q25PHCPU
QnPRHCPU	Generic term for the Q12PRHCPU and Q25PRHCPU
■ Redundant CPU	
Control system	Primary system for control and network communication in the redundant system
Standby system	Backup system in the redundant system
System A	System to which the system A connector of tracking cable is connected
System B	System to which the system B connector of tracking cable is connected
■ 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, Q6□WRB, 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
Redundant type extension base unit	Another name for the Q6□WRB
Multiple CPU high speed main base unit	Another name for the Q3□DB
Redundant base unit	Generic term for the Q3□RB, Q6□RB, and Q6□WRB
Redundant power supply base unit	Generic term for the Q3□RB and Q6□RB

Generic term/abbreviation	Description
■ 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
Q6□WRB	Another name for the Q65WRB 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 on which the QA6ADP is mounted
QA1S6ADP+A1S5□B/A1S6□B	Abbreviation for A small type extension base unit on which 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
AnS series power supply module	Generic term for the A1S61PN, A1S62PN, and A1S63P power supply modules
A series power supply module	Generic term for the A61P, A61PN, A62P, A63P, A68P, A61PEU, and A62PEU 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	
MELSECNET/H	Abbreviation for the MELSECNET/H network system
Ethernet	Abbreviation for the Ethernet network system
CC-Link	Abbreviation for the Control & Communication Link
■ Memory card	
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, and Q3MEM-4MBS SRAM cards
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
■ Others	
GX Developer	Product name for SW□D5C-GPPW-E GPP function software package compatible with the Q series
QA6ADP	Abbreviation for the QA6ADP QA conversion adapter module
QA1S6ADP	Abbreviation for the QA1S6ADP(-S1) Q-AnS base unit conversion adapter
Extension cable	Generic term for the QC05B, QC06B, QC12B, QC30B, QC50B, and QC100B extension cables
Tracking cable	Generic term for the QC10TR and QC30TR tracking cables for redundant systems
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, and GOT1000 series

Memo

CHAPTER1 OVERVIEW

The CPU module performs sequence control by executing programs.
This chapter describes the processing order in the CPU module, locations where the created programs are stored, and devices and instructions useful for programming.

1.1 Processing Order in the CPU Module

The CPU module performs processing in the following order.

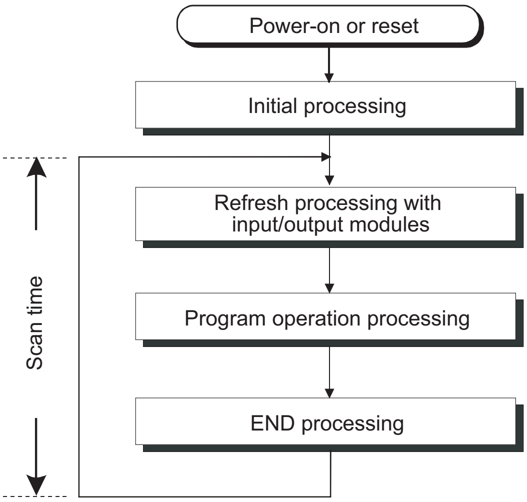

Power-on or reset
Initial processing
Refresh processing with input/output modules
Program operation processing
END processing
Scan time

Figure 1.1 Processing order in the CPU module

(1) Initial processing (☞ Section 3.1)

The CPU module performs preprocessing required for program operations.
The preprocessing is performed only once when the module is powered on or reset.

(2) Refresh processing with input and output modules (☞ Section 3.2)

The CPU module takes on/off data from the input module or intelligent function module and outputs on/off data to the output module or intelligent function module.

(3) Program operation processing (☞ Section 3.3)

The CPU module sequentially executes the program stored in the module from the step 0 to the END or FEND instruction.

(4) END processing (☞ Section 3.4)

The CPU module performs refresh processing with network modules or communicates with external devices.

1.2 Storing and Executing Programs

This section describes where to store and how to execute the programs in the CPU module.

(1) Programming

Programs are created with GX Developer.

For details of program configuration and execution conditions, refer to CHAPTER 2.

(2) Storing programs

Created programs and set parameters are stored in the following memories of the CPU module. (☞ Section 5.1, Section 5.2)

- Program memory
- Standard ROM
- Memory card

(3) Executing programs

The CPU module executes the programs stored in the program memory.

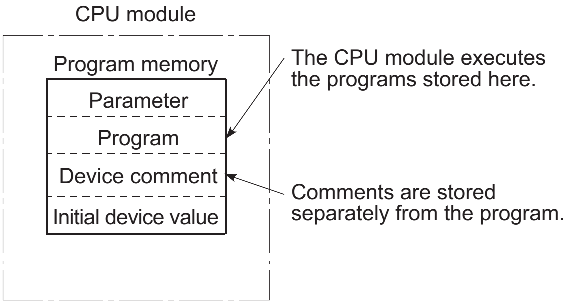

CPU module
Program memory
Parameter
Program
Device comment
Initial device value
The CPU module executes the programs stored here.
Comments are stored separately from the program.

Figure 1.2 Executing programs

To execute the programs stored in the standard ROM or a memory card, the programs need to be booted to the program memory (☞ Section 5.1.5) when the CPU module is powered off and then on or reset.

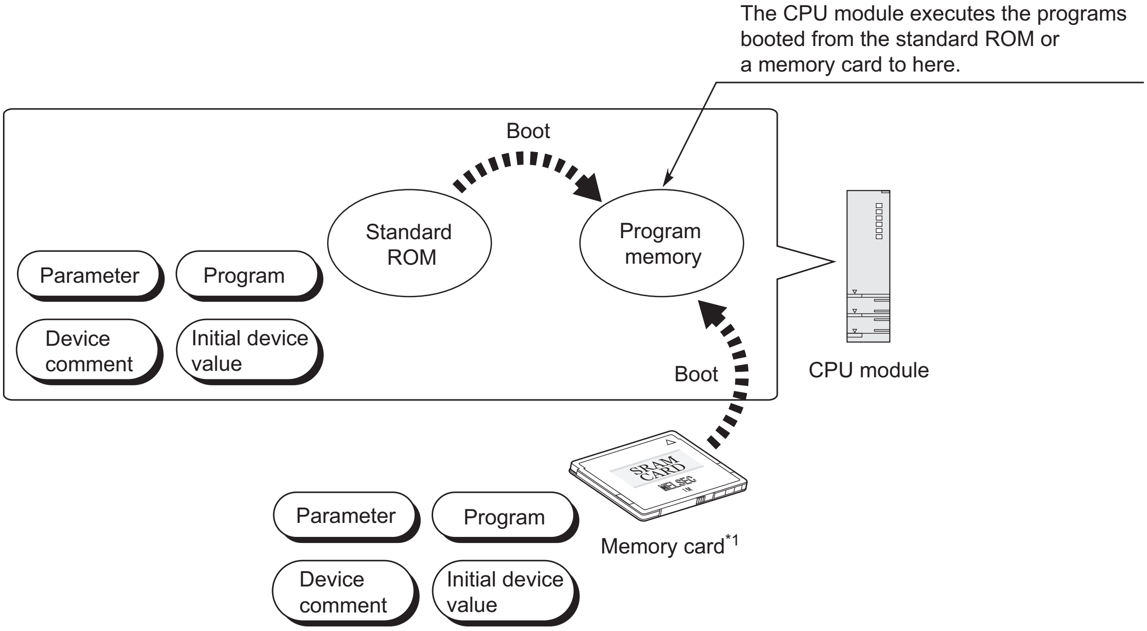


Figure 1.3 Executing programs by performing a boot operation

1.3 Structured Programming

The programs to be executed in the CPU module can be structured in the following two ways.

- In one program
- By dividing into multiple files Note1.1

(1) Structuring in one program

Structured programming is available by creating one program as a collection of three program sections: main routine program (Section 2.2.1), subroutine program (Section 2.2.2), and interrupt program (Section 2.2.3)

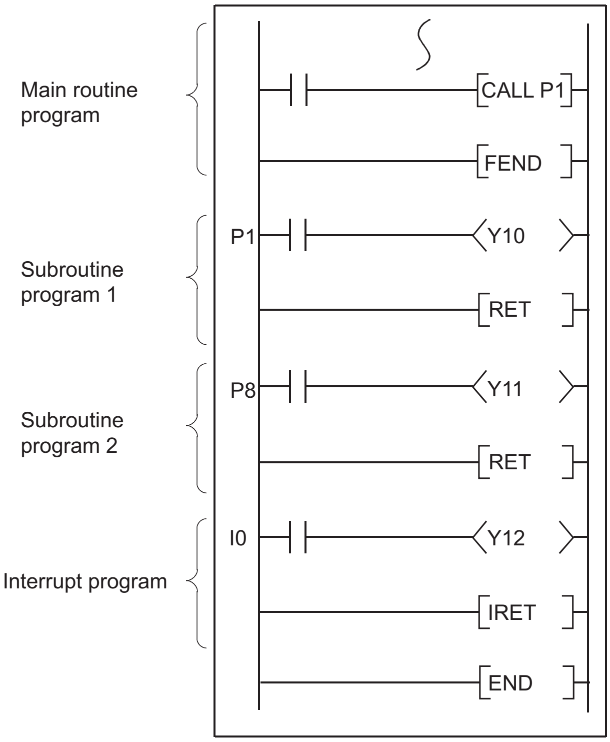


Figure 1.4 Structuring in one program

Note1.1 **Basic**

The Basic model QCPU cannot store multiple programs, structured programming by dividing into multiple files is not available.

(2) Structuring by dividing into multiple files

A program is stored in a file.

Changing the file name allows the CPU module to store multiple programs.

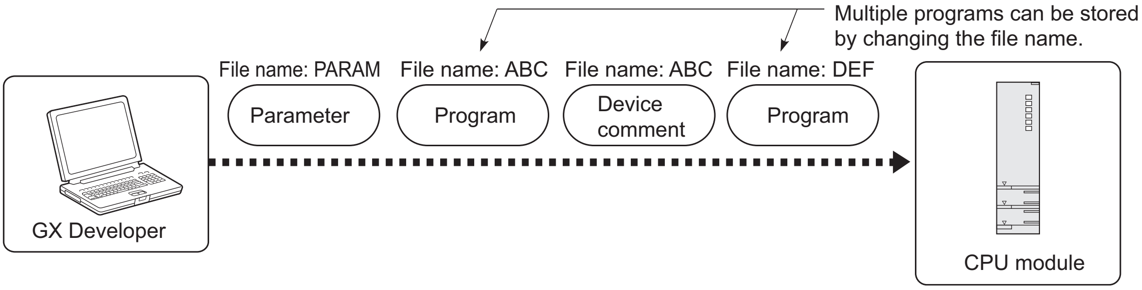


Figure 1.5 Structuring by dividing into multiple files

Dividing into multiple files according to the processes or functions enables simultaneous programming by two or more designers. Managing the files separately eases reuse and utilization to other programs.

Structured programming is efficient in this way because only the corresponding file needs to be modified or debugged in case of change in the specifications.

(a) Dividing into multiple files according to the processes [*1]

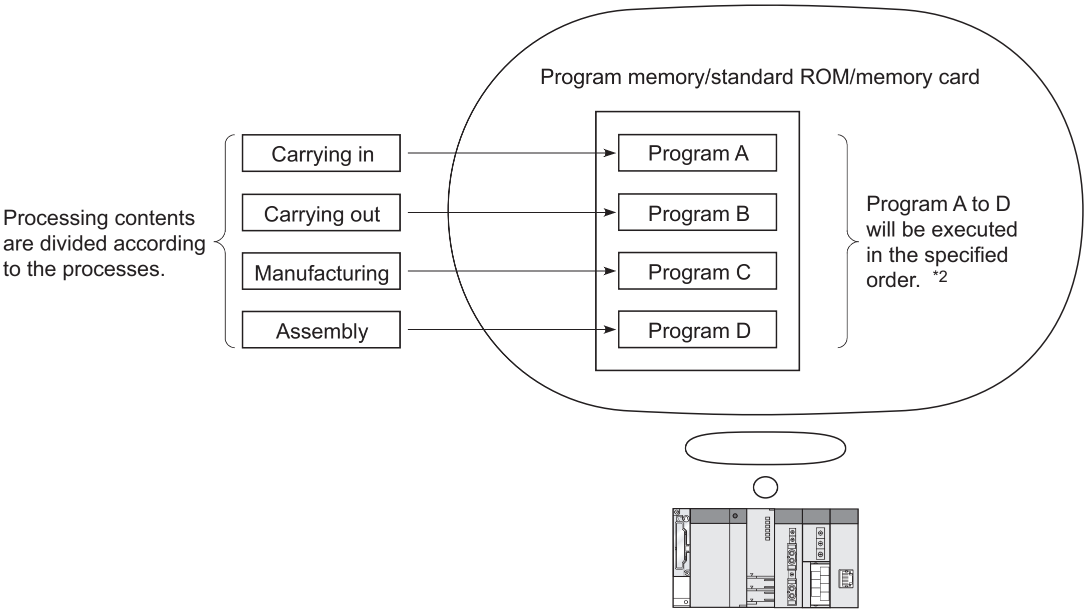

Program memory/standard ROM/memory card
Processing contents are divided according to the processes.
Carrying in
Carrying out
Manufacturing
Assembly
Program A
Program B
Program C
Program D
Program A to D will be executed in the specified order. *2

Figure 1.6 Dividing into multiple files according to the processes

*1: The processing contents divided according to the processes can further be divided and managed according to the functions.

*2: The execution order can be set in the Program tab of the PLC parameter dialog box. (☞ Section 2.3(2))

(b) Dividing into multiple files according to the functions

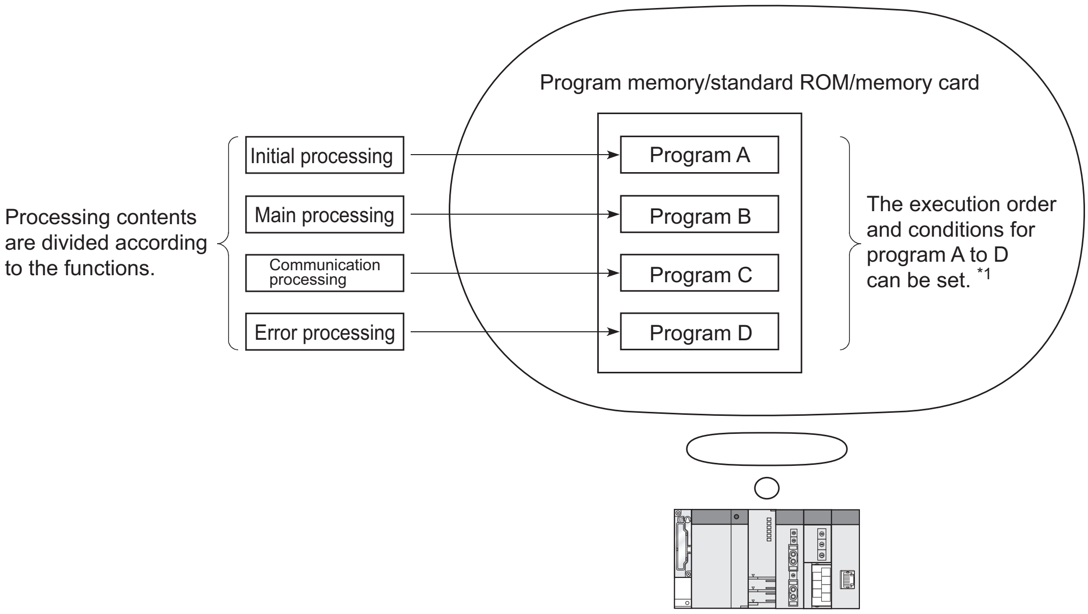

Program memory/standard ROM/memory card
Processing contents are divided according to the functions.
Initial processing
Main processing
Communication processing
Error processing
Program A
Program B
Program C
Program D
The execution order and conditions for program A to D can be set. *1

Figure 1.7 Dividing into multiple files according to the functions

*1: The execution order and conditions can be set in the Program tab of the PLC parameter dialog box (☞ Section 2.3(2))

1.4 Devices and Instructions Useful for Programming

The CPU module is provided with devices and instructions useful for programming.
This section describes the outline of these devices and instructions.

(1) Various ways of device specification

(a) Using each bit of a word device as a contact or coil

By specifying a bit of a word device, the bit can be used as a contact or coil.

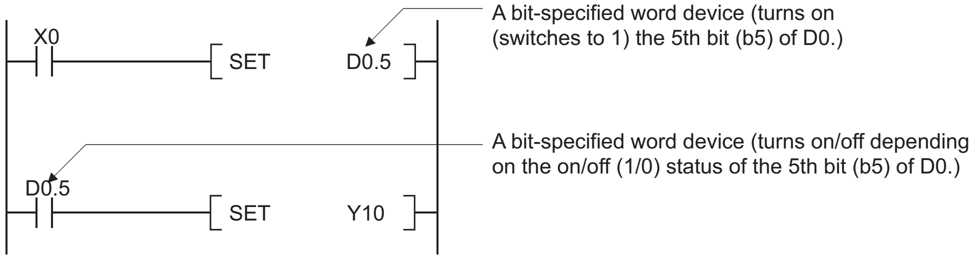

X0
SET
D0.5
A bit-specified word device (turns on (switches to 1) the 5th bit (b5) of D0.)
A bit-specified word device (turns on/off depending on the on/off (1/0) status of the 5th bit (b5) of D0.)
D0.5
SET
Y10

Figure 1.8 Specifying a bit of a word device

(b) Easy direct processing in units of one point

Use of the direct access input (DX □) and direct access output (DY □) enables easy direct processing (in units of one point) in the program. (☞ Section 3.8.2)

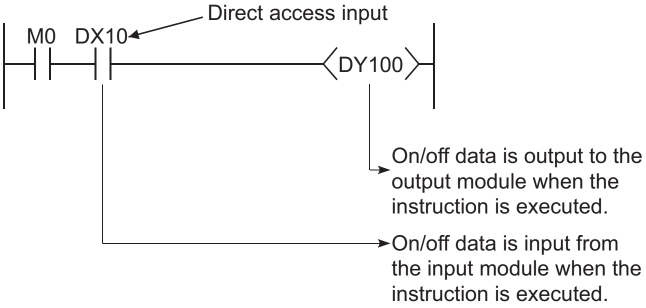

Direct access input
M0
DX10
DY100
On/off data is output to the output module when the instruction is executed.
On/off data is input from the input module when the instruction is executed.

Figure 1.9 Direct processing in units of one point

(c) No input pulse conversion required by using a differential contact

Pulse conversion processing for inputs is no longer required with the use of a differential contacts (┤↑├ and ┤↓├).

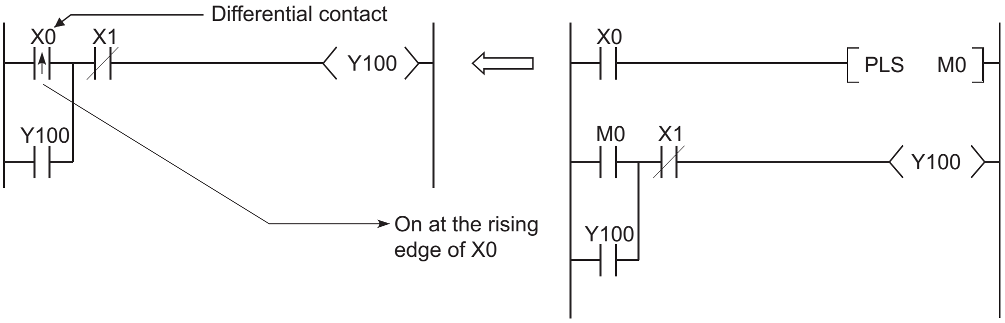


Figure 1.10 Use of a differential contact

(d) Direct access to the buffer memory of the intelligent function module

The buffer memory of the intelligent function module can be used as a device area in a program. (☞ Section 9.5.1)

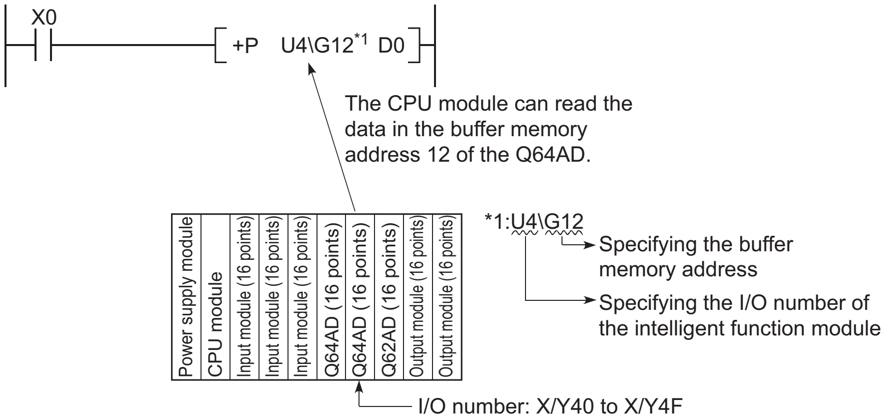

X0
+P U4\G12*1 D0
The CPU module can read the data in the buffer memory address 12 of the Q64AD.
Power supply module
CPU module
Input module (16 points)
Input module (16 points)
Input module (16 points)
Q64AD (16 points)
Q64AD (16 points)
Q62AD (16 points)
Output module (16 points)
Output module (16 points)
*1:U4\G12
Specifying the buffer memory address
Specifying the I/O number of the intelligent function module
I/O number: X/Y40 to X/Y4F

Figure 1.11 Direct access to the buffer memory of the intelligent function module

(e) Direct access to the link devices

The link devices (LX, LY, LB, LW, SB, or SW) in network modules can be directly accessed without the refresh setting. (☞ Section 9.4)

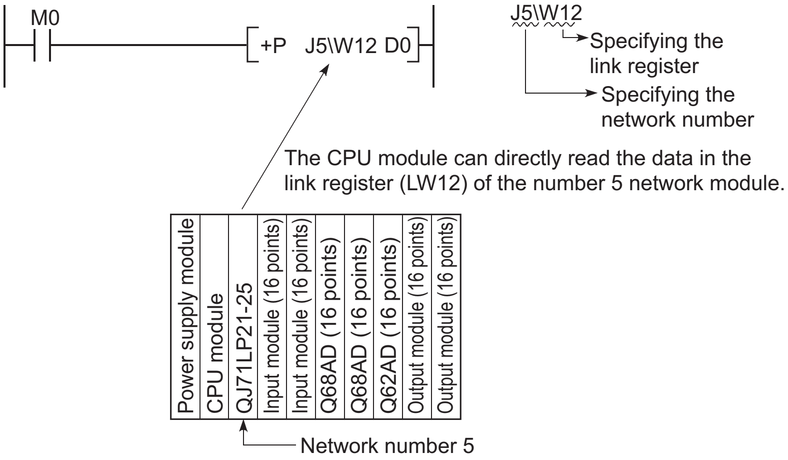

M0
+P J5\W12 D0
J5\W12
Specifying the link register
Specifying the network number
The CPU module can directly read the data in the link register (LW12) of the number 5 network module.
Power supply module
CPU module
QJ71LP21-25
Input module (16 points)
Input module (16 points)
Q68AD (16 points)
Q68AD (16 points)
Q62AD (16 points)
Output module (16 points)
Output module (16 points)
Network number 5

Figure 1.12 Direct access to the link devices

(2) Structural description of programs

Use of the index register and edge relay enables easy structured programming including the pulse conversion processing. (☞ Section 9.2.6)

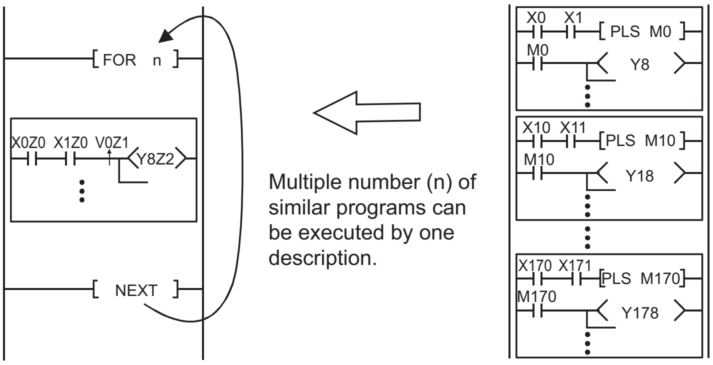


Figure 1.13 Structured programming including the pulse conversion processing

(3) Easy data processing

(a) Using real numbers and character string constants without conversion

Real numbers (floating-point data) and character string constants can be used without conversion for programming.

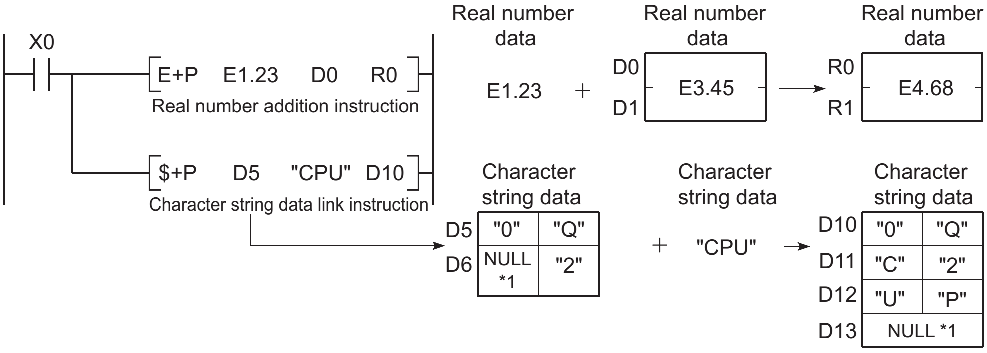


Figure 1.14 Structured programming including the pulse conversion processing

*1: The NULL character represents "00H" (end of character strings)

(b) High-speed processing of bulk data

Extension of the data table operation instructions, such as the data table processing instruction, allows high-speed processing of bulk data.

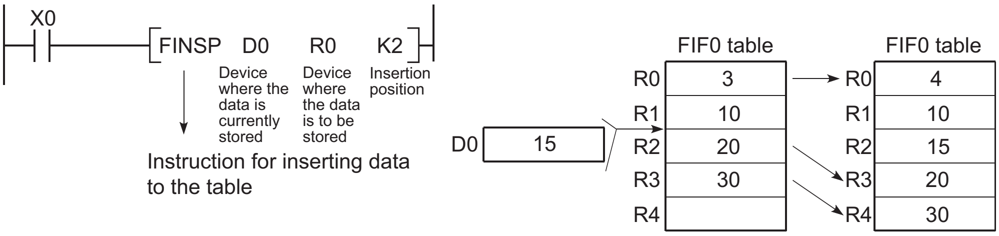

X0
FINSP D0 R0 K2
Device where the data is currently stored
Device where the data is to be stored
Insertion position
Instruction for inserting data to the table
D0
15
FIF0 table
R0 3
R1 10
R2 20
R3 30
R4
FIF0 table
R0 4
R1 10
R2 15
R3 20
R4 30

Figure 1.15 Data processing with the table processing instruction

(4) Flexible management of subroutine programs

(a) Subroutine program sharing

The number of steps in a program can be reduced by sharing subroutine programs.
In addition, creating and managing programs become easier.

Subroutine programs can be created within the same program and called. Subroutine programs in other programs can also be called by using the common pointer.

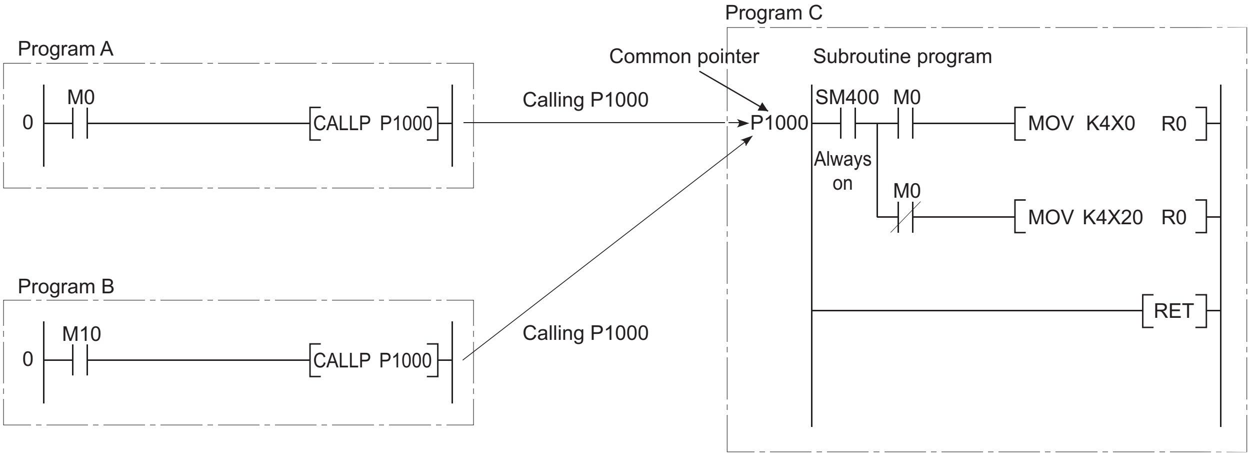


Figure 1.16 Subroutine program sharing

(b) Subroutine call instruction with argument passing

Subroutine program that is called more than one time can be created easily

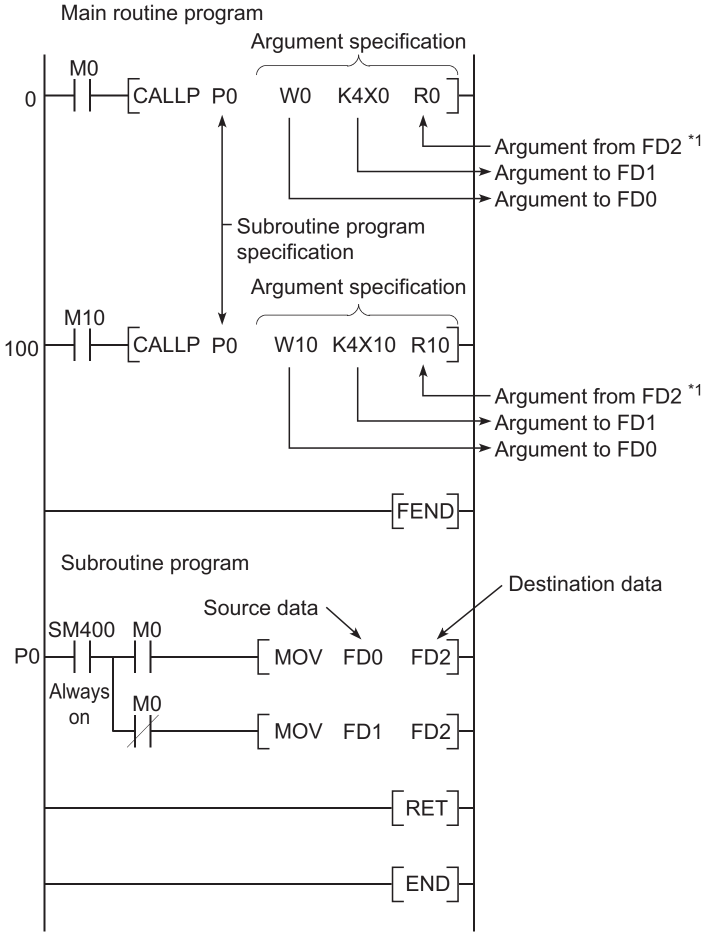


Figure 1.17 Calling subroutine program with argument passing

*1: For input and output conditions for an argument, refer to Section 9.3.1.

1.5 Features

This section describes the features of the CPU modules.

1.5.1 Features of the Basic model QCPU

The features specific to the Basic model QCPU are described below.

(1) High cost performance for small-scale system

The Basic model QCPU is suitable for controlling a small-scale system of simple and compact system. The CPU module provides high cost performance for a small-scale system.

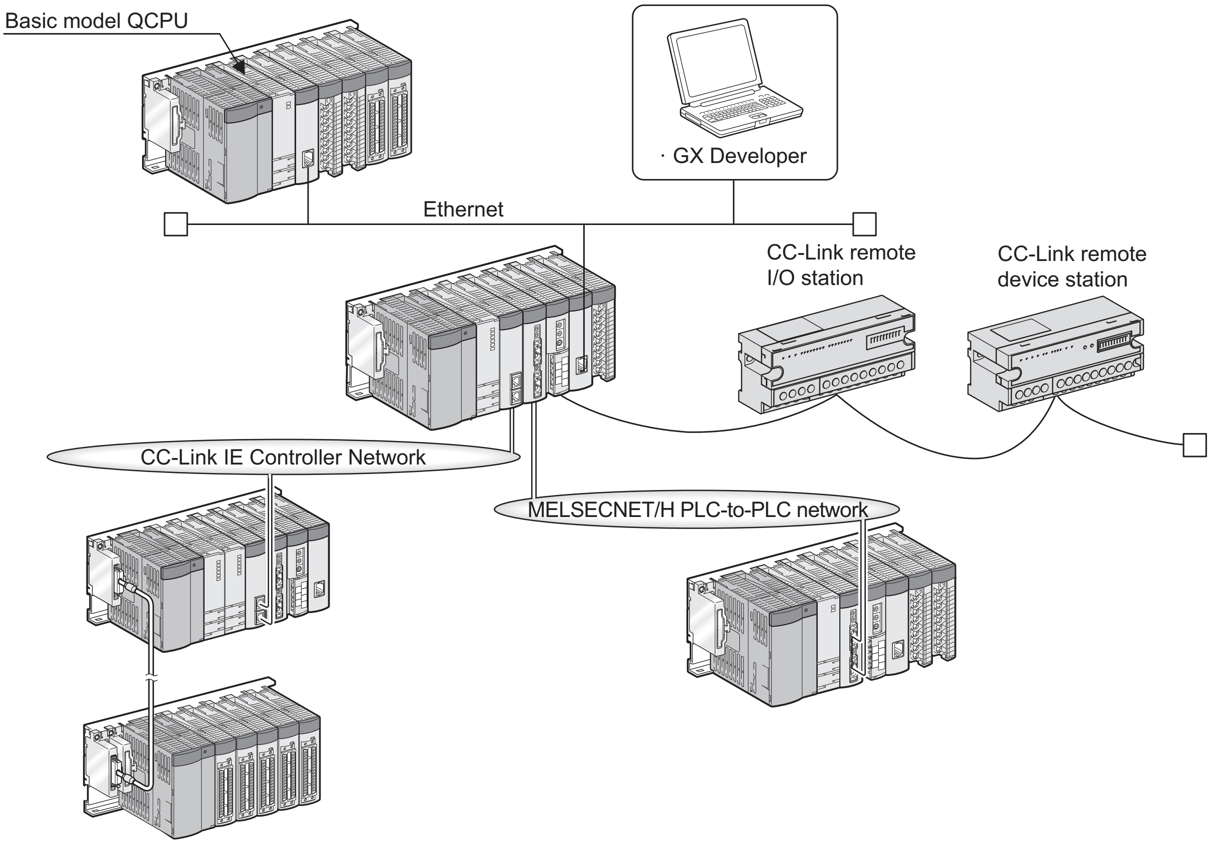

Basic model QCPU
· GX Developer
Ethernet
CC-Link remote I/O station
CC-Link remote device station
CC-Link IE Controller Network
MELSECNET/H PLC-to-PLC network

Figure 1.18 System example using the Basic model QCPU

(2) Communication with the personal computer and HMI by the serial communication function (☞ Section 6.24)

The Q00CPU and Q01CPU can communicate using the MELSEC communication protocol (hereafter, MC protocol) by connecting a RS-232 interface and personal computer or HMI.

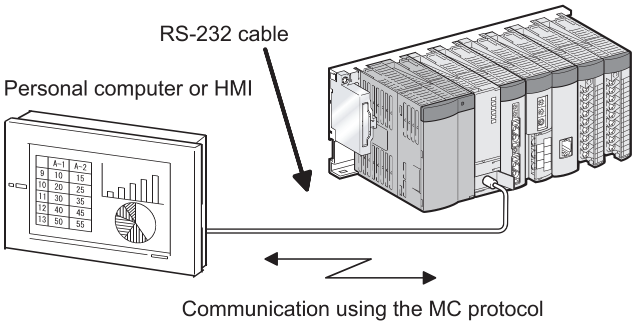


Figure 1.19 Communication with the personal computer or HMI

Point

The CPU module functions are added at the update of serial number of CPU module or GX Developer version.
For functions added by the update of the Basic model QCPU, refer to Appendix 2.1.

1.5.2 Features of the High Performance model QCPU

The features specific to the High Performance model QCPU are described below.

(1) High performance and large capacity

The High Performance model QCPU has large capacity and allows high-speed processing suitable for any scale system.

This allows configuration of optimum and high performance system.

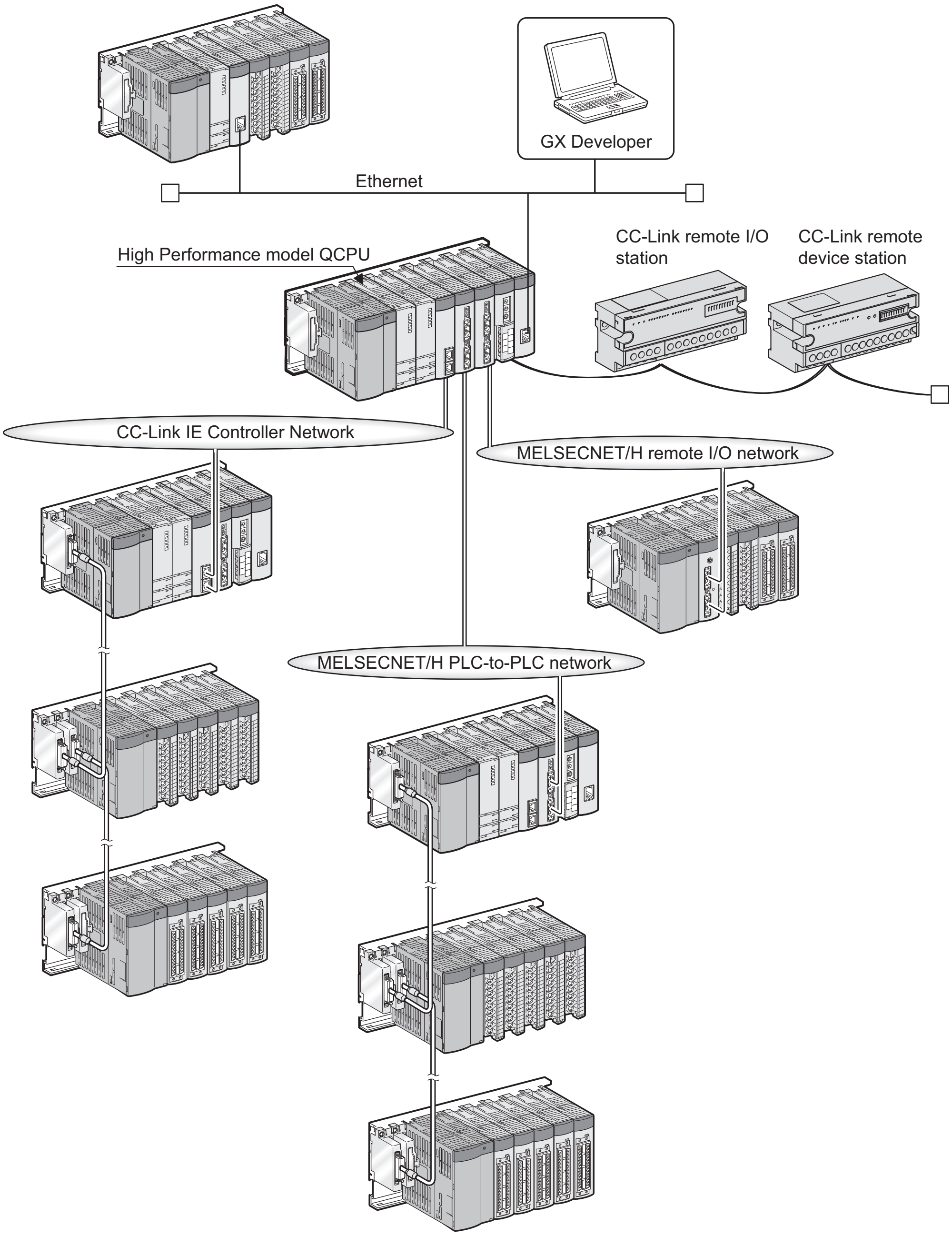

GX Developer
Ethernet
High Performance model QCPU
CC-Link remote I/O station
CC-Link remote device station
CC-Link IE Controller Network
MELSECNET/H remote I/O network
MELSECNET/H PLC-to-PLC network

Figure 1.20 System example using the High Performance model QCPU

(2) AnS/A series I/O modules and special function modules are available.

These modules can be used together with the High Performance model QCPU by mounting them on the AnS/A series-compatible extension base unit (QA1S5□B, QA1S6□B, QA1S6ADP+A1S5□B/A1S6□B, QA6□B, and QA6ADP+A5□B/A6□B).

Point

The CPU module functions are added at the update of serial number of CPU module or GX Developer version.
For functions added by the update of the High Performance model QCPU, refer to Appendix 2.2.

1.5.3 Features of the Process CPU

The features specific to the Process CPU are described below.

(1) Addition of 52 process control instructions

Based on the instructions for the High Performance model QCPU, 52 instructions for high-level process control have been added to the Process CPU.

For details of the added instructions, refer to the following.

☞ QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions)

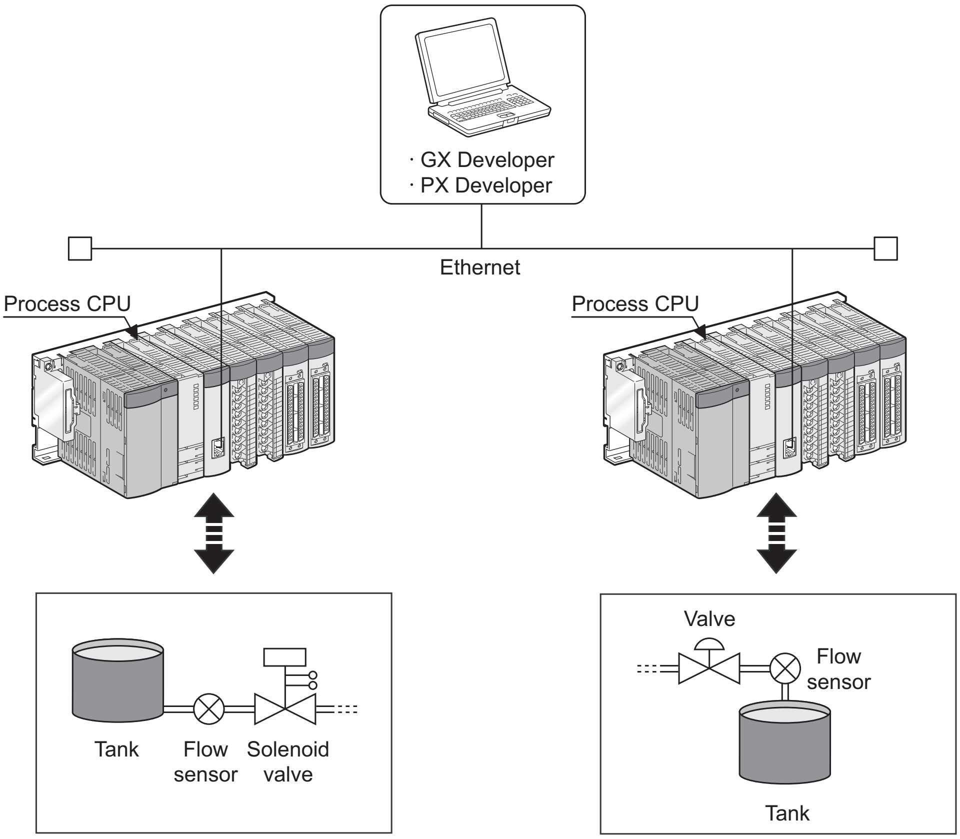


Figure 1.21 Operations of the Process CPU

(2) Support of the two-degree-of-freedom PID control system

This system allows optimum response independent of set value and disturbance changes.

(3) Auto tuning function (PID constant initial value setting)

This function enables automatic tuning with control parameters, reduction in tuning time, man-hour, and control engineering, and elimination of gaps caused by manual tuning.

For details of the auto tuning function, refer to the following.

☞ QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions)

(4) Module can be changed online. (Online module change)

When a module fails, it can be changed without the system being stopped.
For details of the online module change, refer to the following.

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

☞ Manual for the module supporting the online module change

(5) Configuration of MELSECNET/H multiplexed remote I/O system

Installing a remote master station in the MELSECNET/H network system can configure the MELSECNET/H multiplexed remote I/O system.

(6) Support of software package dedicated to process control (PX Developer)

Process control software package (PX Developer) easily creates PID control programs with function blocks.
The combination of the Process CPU and the process control software package (PX Developer) offers excellent engineering environment.

Point

- For the Q12PHCPU and Q25PHCPU, use GX Developer Version 7.10L or later.
  For the Q02PHCPU and Q06PHCPU, use GX Developer Version 8.68W or later.
- Use PX Developer together with GX Developer Version 7.12N or later.
  For the Q12PHCPU or Q25PHCPU, use PX Developer Version 1.00A or later.
  For the Q02PHCPU or Q06PHCPU, use PX Developer Version 1.18U or later.
  For details of PX Developer, refer to the manual for PX Developer.
- The CPU module functions are added at the update of serial number of CPU module or GX Developer version.
  For functions added by the update of the Process CPU, refer to Appendix 2.3.

1.5.4 Features of the Redundant CPU

The features specific to the Redundant CPUs are described below.

(1) Support of the redundant system in addition to the Process CPU functions

(a) Redundant system using the Redundant CPU

Using the Redundant CPU can configure a redundant system including base units, power supply modules, and CPU modules (Redundant CPUs).

In case of a failure of the control system, the standby system takes over the control in the redundant system, enabling high reliability of the system.

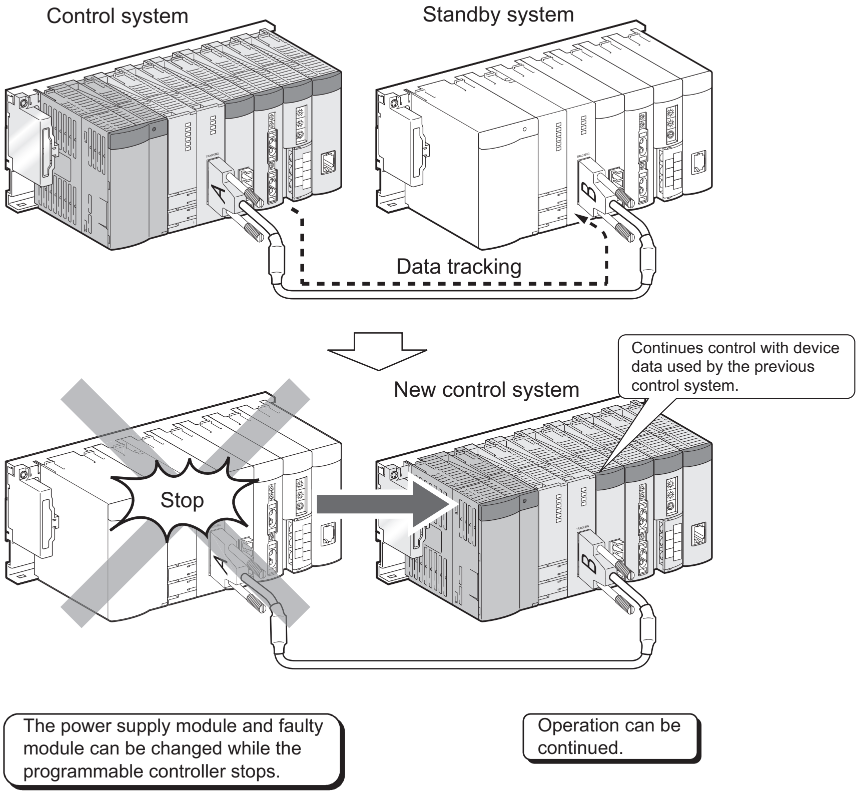

Control system
Standby system
A
B
Data tracking
Continues control with device data used by the previous control system.
New control system
Stop
A
B
The power supply module and faulty module can be changed while the programmable controller stops.
Operation can be continued.

Figure 1.22 Operation of the Redundant CPU

(b) Redundant power supply system

Using the redundant power main base unit (Q3□RB) and the redundant power supply module (Q63RP and Q64RP) on the remote I/O station enables use of two power supply modules on the remote I/O station side. This enables changing the power supply module without stopping the system even when the power supply module on the remote I/O station side fails.

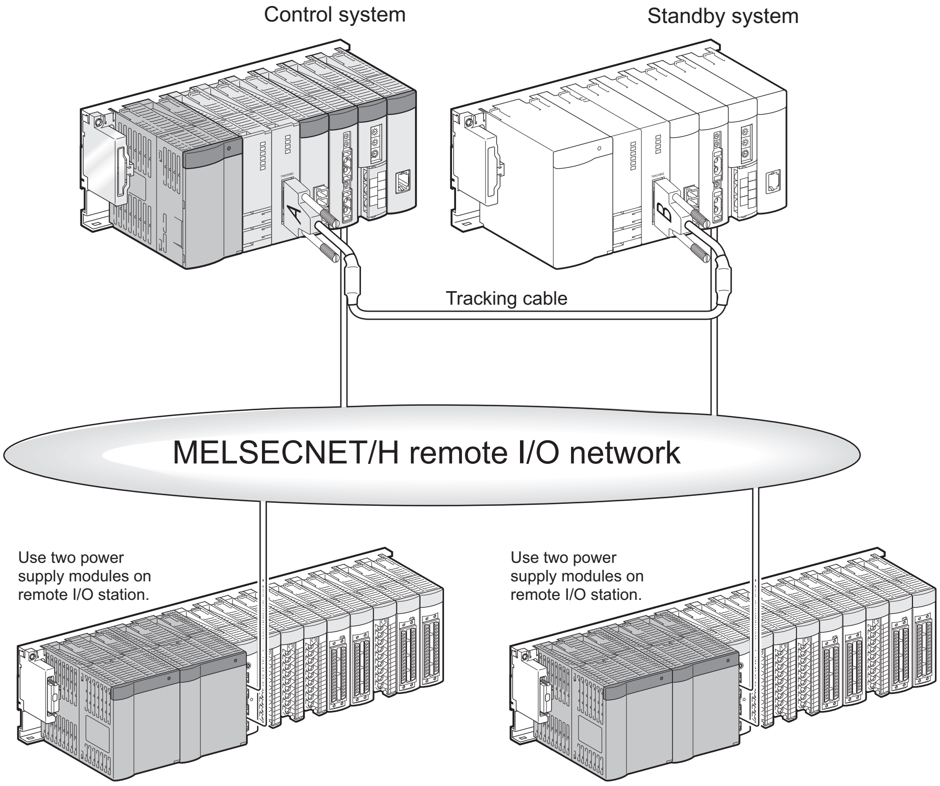

Control system
Standby system
A
B
Tracking cable
MELSECNET/H remote I/O network
Use two power supply modules on remote I/O station.
Use two power supply modules on remote I/O station.

Figure 1.23 Redundant power supply system

Point

- Use PX Developer and GX Developer together with the following versions.
  - GX Developer Version 8.17T or later
  - PX Developer Version 1.05F or later

- The CPU module functions are added at the update of serial number of CPU module or GX Developer version.
  For functions added by the update of the Redundant CPU, refer to Appendix 2.4.

Remark

For details of the features, functions of the Redundant CPU, refer to the following.

QnPRHCPU User's Manual (Redundant System)

1.6 Checking Serial Number and Function Version

The serial number and function version of the CPU module can be checked on the rating plate, on the front of the module, and on the System monitor screen in GX Developer.

(1) Checking on the rating plate

The rating plate is located on the side of the CPU module.

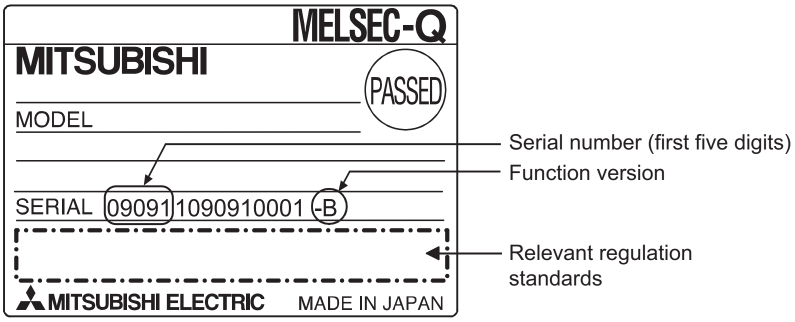


Figure 1.24 Rating plate

(2) Checking on the front of the module

The serial number on the rating plate is printed on the front (at the bottom) of the module. This applies only to the CPU module manufactured in mid-September, 2007 or later.

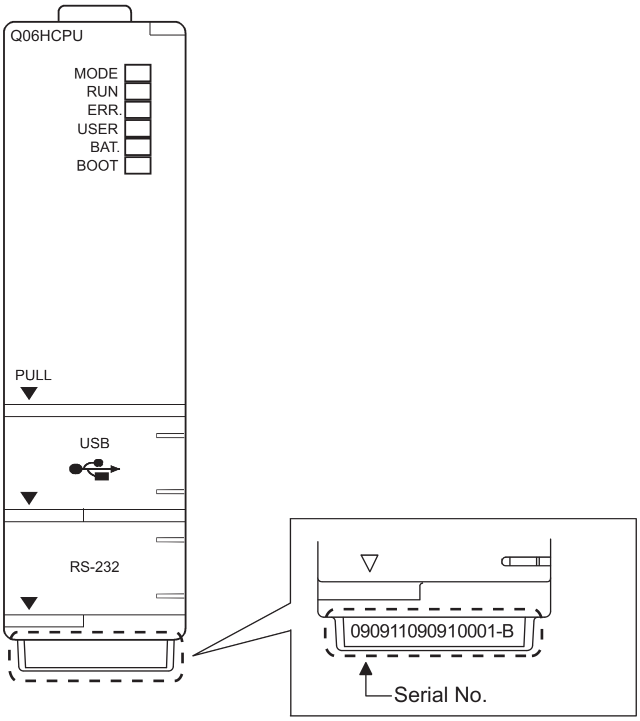
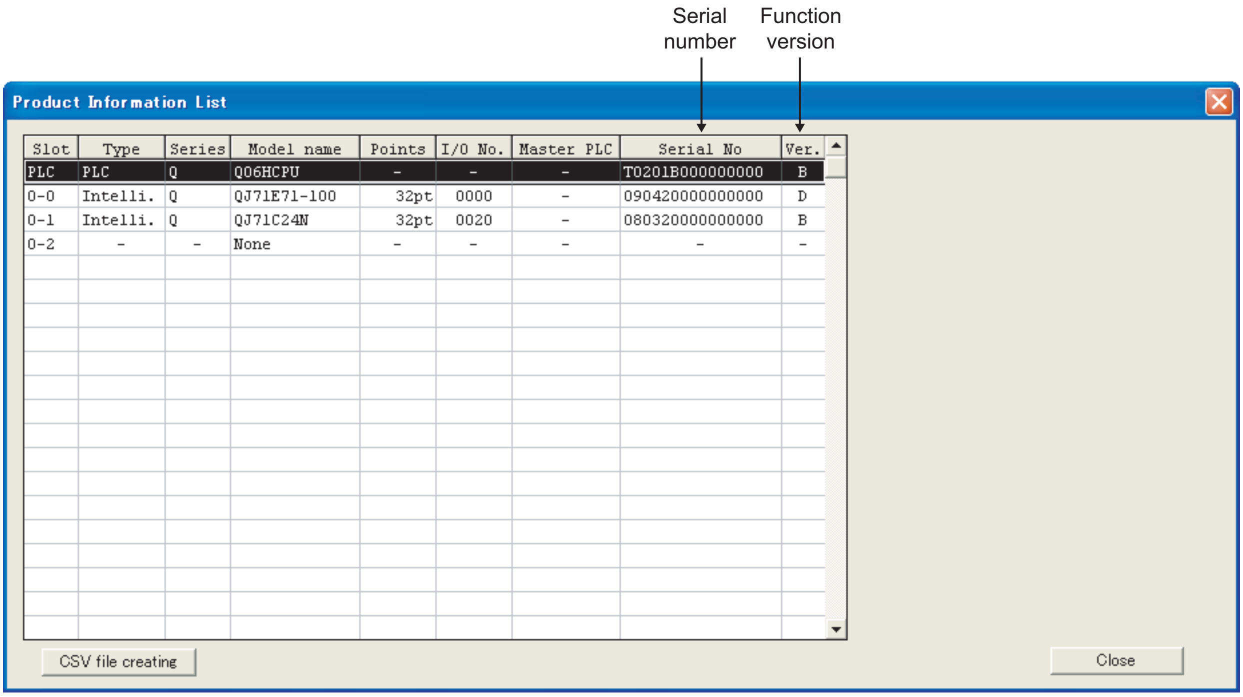


Figure 1.25 Front of the module

(3) Checking on the System monitor (Product Information List) screen

To open the screen for checking the serial number and function version, select [Diagnostics] → [System monitor] and click the Trace data setting button in GX Developer.

On the same screen, the serial number and function version of intelligent function modules can also be checked.



Slot	Type	Series	Model name	Points	I/O No.	Master PLC	Serial No	Ver.
PLC	PLC	Q	Q06HCPU	-	-	-	T0201B000000000	B
0-0	Intelli.	Q	QJ71E71-100	32pt	0000	-	090420000000000	D
0-1	Intelli.	Q	QJ71C24N	32pt	0020	-	080320000000000	B
0-2	-	-	None	-	-	-	-	-

Figure 1.26 System monitor (Product Information List) screen

[Serial number, function version, and product number]

- The serial number of the module is displayed in the "Serial No." column.
- The function version of the module is displayed in the "Ver." column.

Point

The serial number displayed on the Product Information List screen of GX Developer may differ from that on the rating plate and on the front of the module.

- The serial number on the rating plate and on the front of the module indicates the management information of the product.
- The serial number displayed on the Product Information List screen indicates the functional information of the product.
  The functional information of the product will be updated when a function is added.

CHAPTER2 SEQUENCE PROGRAMS

2.1 Sequence Program Overview

(1) Definition

Sequence program is one of the programs that can be executed in the CPU module.

A sequence program consists of instructions, such as sequence instructions, basic instruction, and application instruction.

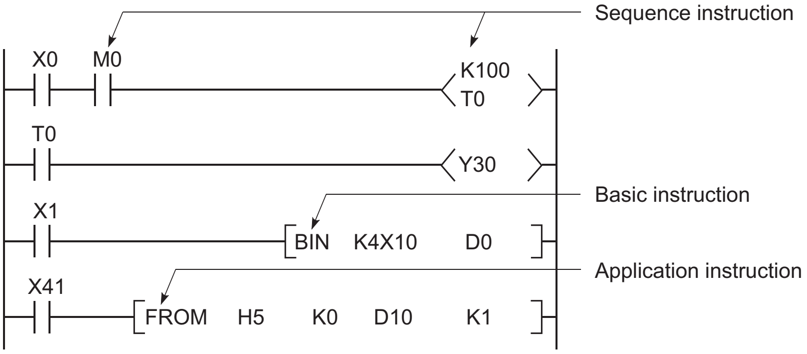


Figure 2.1 Sequence program

Point

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

Remark

For the instructions used in sequence programs, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

(2) Programming method

There are two programming modes for sequence programs.

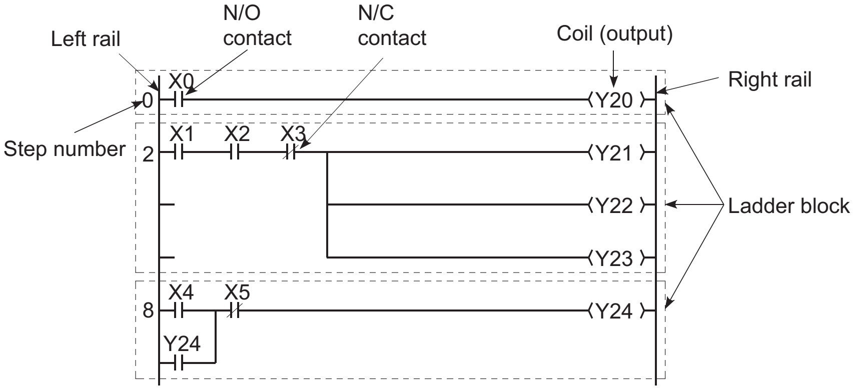
- Ladder mode
- List mode

(a) Ladder mode

Ladder mode is a mode based on the concept of sequential control by relay circuits.

A program in ladder mode is similar to a schematic for a set of relay circuits. Programming in units of ladder blocks is available.

A ladder block, which starts from the left rail and ends at the right rail, is the minimum unit for operating sequence programs.



* X0 to X5: Input, Y20 to Y24: Output

Figure 2.2 Ladder mode

(b) List mode

List mode uses dedicated instructions for programming. The symbols, such as contacts and coils, used in ladder mode are replaced with dedicated instructions.

The following instructions are used for normally open contacts, normally closed contacts, and coils.

- Normally open contact ・・・ LD, AND, OR
- Normally closed contact ・・・ LDI, ANI, ORI
- Coil ・・・ OUT

Remark

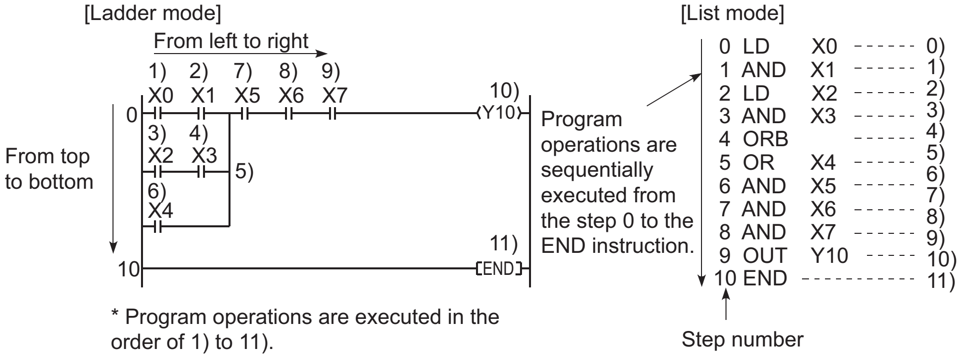
There are two other types of programs that can be executed in the CPU module: SFC programs and ST programs. For details, refer to the following.

☞ MELSEC-Q/L/QnA Programming Manual (SFC)
☞ MELSEC-Q/L Programming Manual (MELSAP-L)
☞ MELSEC-Q/L Programming Manual (Structured Text)

(3) Sequence program operation

A sequence program is sequentially operated from the step 0 to the END or FEND instruction.

In ladder mode, a sequence program is operated from left to right and top to bottom in a ladder block.



```
0  LD    X0  ------ 0)
1  AND   X1  ------ 1)
2  LD    X2  ------ 2)
3  AND   X3  ------ 3)
4  ORB       ------ 4)
5  OR    X4  ------ 5)
6  AND   X5  ------ 6)
7  AND   X6  ------ 7)
8  AND   X7  ------ 8)
9  OUT   Y10 -----  9)
10 END ----------- 10)
                   11)
```

* Program operations are executed in the order of 1) to 11).

Figure 2.3 Comparison between ladder mode and list mode

2.2 Sequence Program Configuration

Sequence programs are classified into the following three types.

- Main routine program
- Subroutine program
- Interrupt program

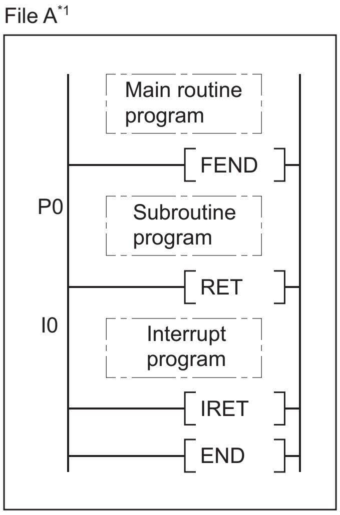


Figure 2.4 Sequence program classification

*1: Since the Basic model QCPU cannot execute multiple programs, the file name is fixed to "MAIN".

2.2.1 Main routine program

(1) Definition

Main routine program is an entire program from the step 0 to the END or FEND instruction.

(2) Program operation

A main routine program executes its operations from the step 0 to the END or FEND instruction and then performs END processing.

After the END processing, the program restarts its operations from the step 0.

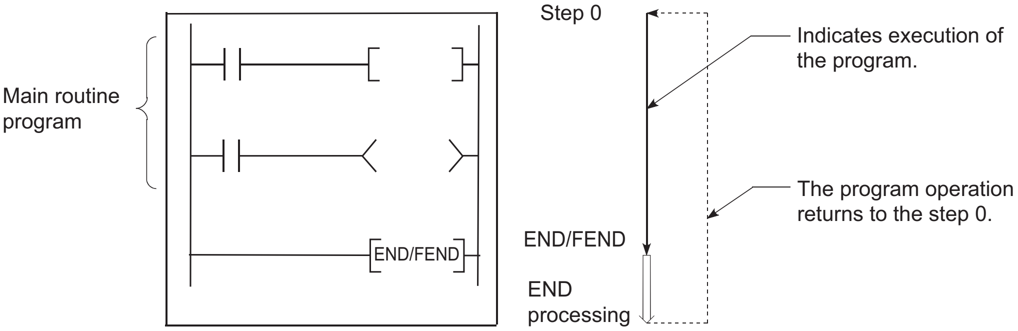

Main routine program
END/FEND
Step 0
Indicates execution of the program.
The program operation returns to the step 0.
END/FEND
END processing

Figure 2.5 Main routine program

Point

When multiple programs are executed, the main routine program operation after execution of the END or FEND instruction varies depending on the preset execution conditions. (☞ Section 2.3)

Remark

For details of the END and FEND instructions, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

2.2.2 Subroutine program

(1) Definition

Subroutine program is a program from a pointer (P□) to the RET instruction.
This program is executed only when it is called by a subroutine program call instruction (such as CALL(P), FCALL(P)) from a main routine program.

(2) Application

- Programming a program which is executed two or more times in one scan as a subroutine program can reduce the number of steps in the entire program.
- Programming a program which is executed only when a certain condition is satisfied as a subroutine program can shorten the scan time.

(3) Programming of subroutine programs

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

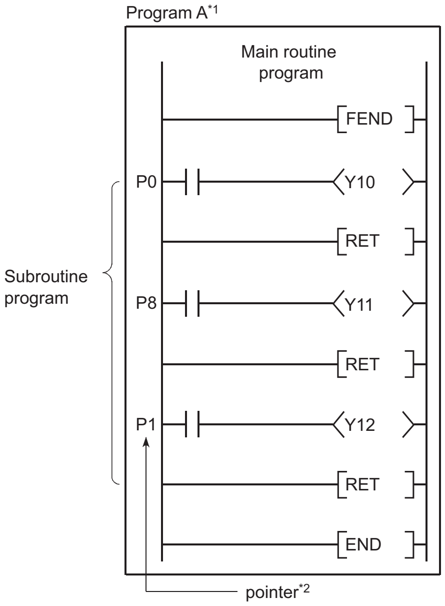


Figure 2.6 Programming location of subroutine programs

*1: Since the Basic model QCPU cannot execute multiple programs, the file name is fixed to "MAIN".
*2: The pointer numbers do not need to be specified in ascending order.

Point

Subroutine programs can be managed as one separate program (stand-by type program). (☞ Section 2.3.4)

Remark

Subroutine programs can be configured with the nesting. (☞ Section 9.8)

2.2.3 Interrupt program

(1) Definition

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

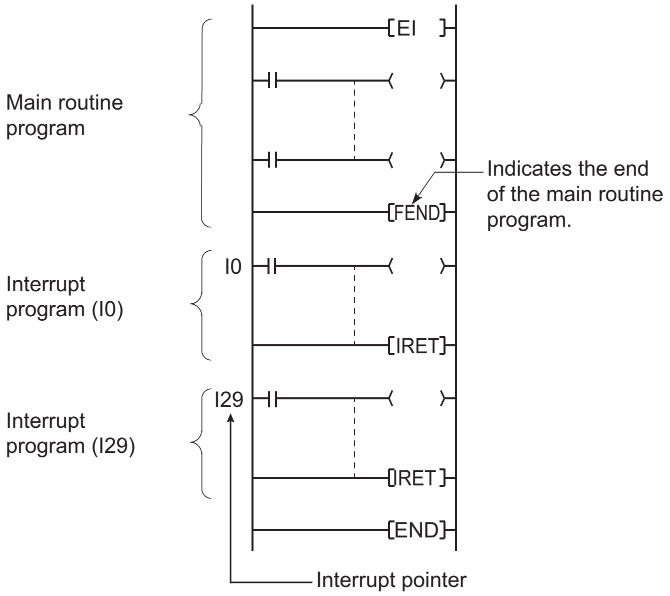


Figure 2.7 Interrupt program

The interrupt pointer (I□) number varies depending on the interrupt factor. (☞ Section 9.10)
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 corresponding interrupt factor occurs.)

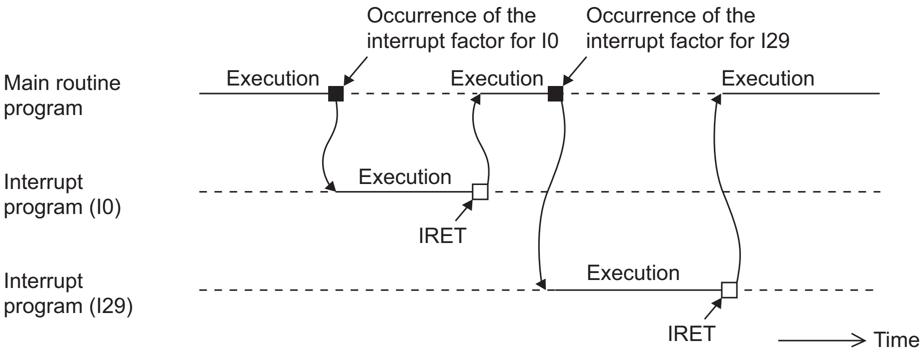


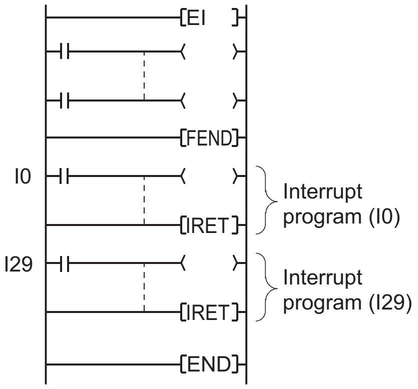
Figure 2.8 Interrupt program execution timing

Point

- A pointer dedicated to the high-speed interrupt function (I49) Note2.1 is available as an interrupt pointer. When using I49, do not execute the following:
  - Another interrupt pointer (interrupt pointer other than I49)
  - Interrupt program
  - Fixed scan execution type program

  If any of the above pointer or program is executed, the interrupt program of I49 cannot be executed in the preset interrupt cycles.

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



Remark

For details of the interrupt factors and interrupt pointers, refer to Section 9.10.

Note2.1 Basic High performance Process Redundant

The Basic model QCPU, Q02CPU, Process CPU, and Redundant CPU do not support the use of the pointer dedicated to the high-speed interrupt function (I49).

(2) Programming of interrupt programs

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

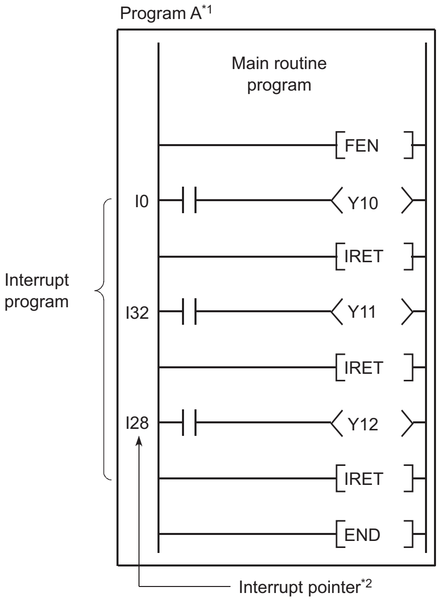


Figure 2.9 Programming location of interrupt programs

*1: Since the Basic model QCPU cannot execute multiple programs, the file name is fixed to "MAIN".

*2: 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). (☞ Section 2.3.4)

(a) Before executing an interrupt program

1) Basic model QCPU

Enable interrupts with the EI instruction.

2) High Performance model QCPU, Process CPU, or Redundant CPU

Enable interrupts with the IMASK or EI instruction for execution of the interrupt programs corresponding to the interrupt pointers I32 to I47.

Enable interrupts with the EI instruction for execution of the interrupt programs corresponding to the interrupt pointers I0 to I31 and I48 to I255.

Remark

For details of the IMASK and EI instructions, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

(b) Restrictions on programming

1) PLS and PLF instructions

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

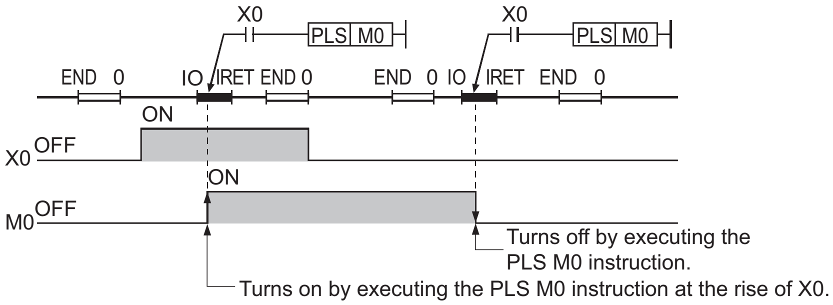

X0
PLS M0
X0
PLS M0
END 0
IO IRET END 0
END 0 IO IRET END 0
ON
X0 OFF
ON
M0 OFF
Turns off by executing the PLS M0 instruction.
Turns on by executing the PLS M0 instruction at the rise of X0.

Figure 2.10 Device turned on by the PLS instruction in the interrupt program

2) 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.

3) 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) not to measure the number of interrupts correctly.

4) Instructions not available in interrupt programs

Refer to sections corresponding to each instruction in the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

(3) 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.

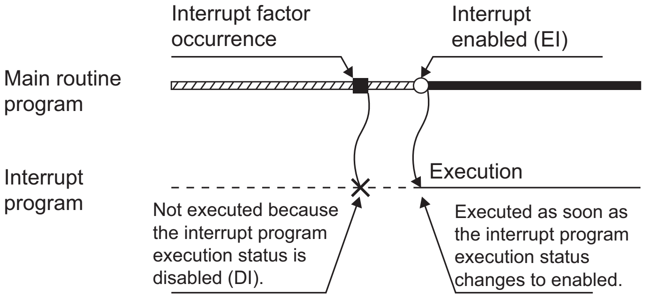

Interrupt factor occurrence
Interrupt enabled (EI)
Main routine program
Interrupt program
Execution
Not executed because the interrupt program execution status is disabled (DI).
Executed as soon as the interrupt program execution status changes to enabled.

Figure 2.11 When an interrupt factor occurs before interrupts are enabled

When the same interrupt factor occurs more than one time before the interrupt program execution status is enabled, the CPU module operates as follows:

1) Basic model QCPU

The CPU module stores the interrupt factors of I0 to I15, I28 to I31, and I50 to I127 only once.

2) High Performance model QCPU, Process CPU, or Redundant CPU

The CPU module stores the interrupt factors of I0 to I27, I28 to I31, I50 to I255, and fixed scan execution type programs only once.

As for I32 to I41 and I49, the CPU module discards the interrupt factors occurred while interrupts are disabled.

(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.

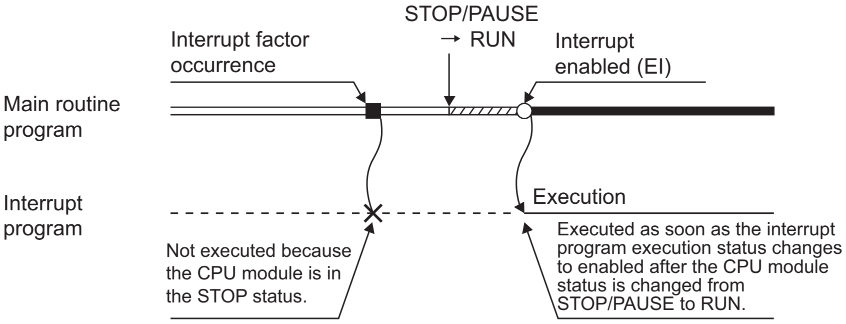

STOP/PAUSE → RUN
Interrupt factor occurrence
Interrupt enabled (EI)
Main routine program
Interrupt program
Execution
Not executed because the CPU module is in the STOP status.
Executed as soon as the interrupt program execution status changes to enabled after the CPU module status is changed from STOP/PAUSE to RUN.

Figure 2.12 When an interrupt factor occurs in the STOP or PAUSE status

(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. (☞ Section 9.10.1)

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

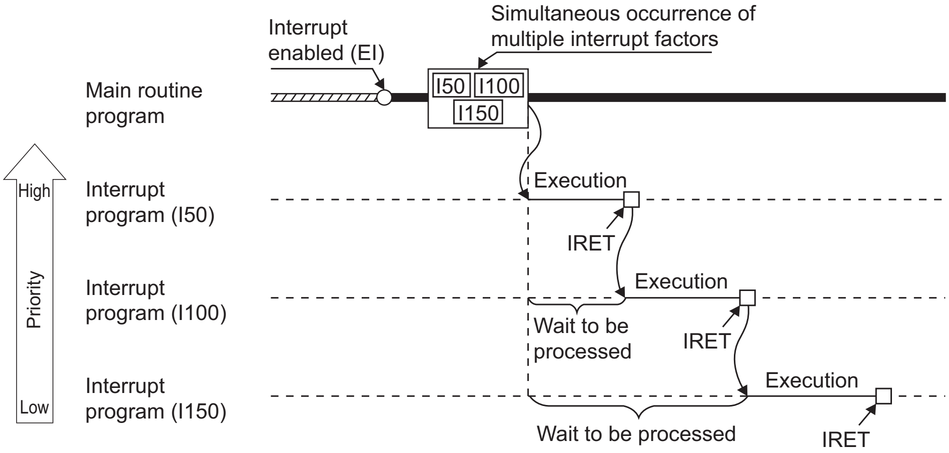


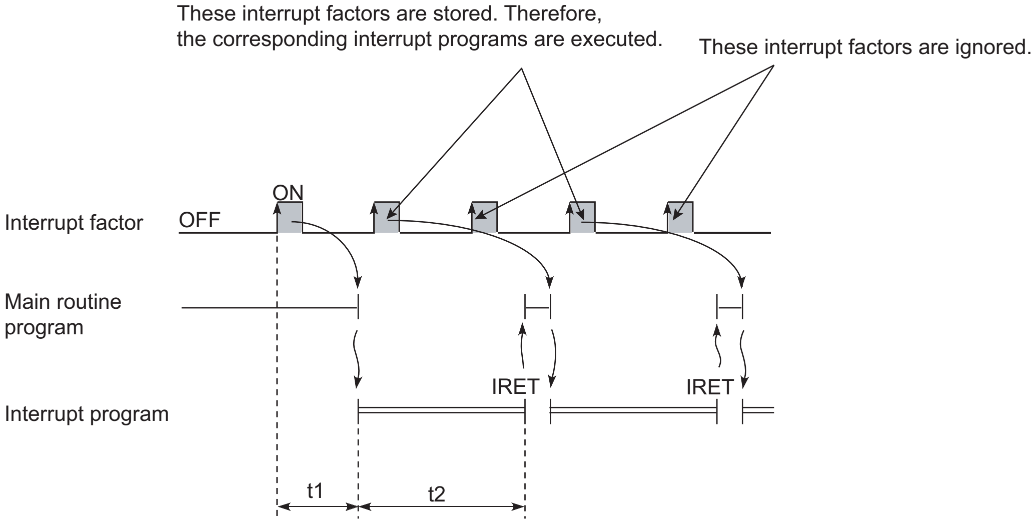
Figure 2.13 When multiple interrupt factors occur simultaneously

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

1) Basic model QCPU

The CPU module stores the interrupt factors of I0 to I15, I28 to I31, and I50 to I127 only once. Then, the CPU module executes the interrupt program corresponding to the stored interrupt factor after completion of current interrupt program execution.

Even if the same interrupt factor occurs more than one time, the CPU module stores only the first factor. The second interrupt factor and later are ignored.



These interrupt factors are stored. Therefore, the corresponding interrupt programs are executed.
These interrupt factors are ignored.
ON
Interrupt factor
OFF
Main routine program
IRET
IRET
Interrupt program
t1
t2

t1: Time from interrupt factor occurrence to interrupt program execution
t2: Interrupt program execution time

Figure 2.14 Operation when the same interrupt factor as that of the interrupt program being executed occurs

2) High Performance model QCPU, Process CPU, or Redundant CPU

- The CPU module stores the interrupt factors of I0 to I27 and I50 to I255 only once. Then, the CPU module executes the interrupt program corresponding to the stored interrupt factor after completion of current interrupt program execution.
  As for the I28 to I31 and fixed scan execution type programs, the CPU module stores all interrupt factors. Then, the CPU module executes the interrupt program corresponding to the stored interrupt factor after completion of current interrupt program execution.
- The interrupt factors of I32 to I41 and I49 are discarded.

(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 Controller Network 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.

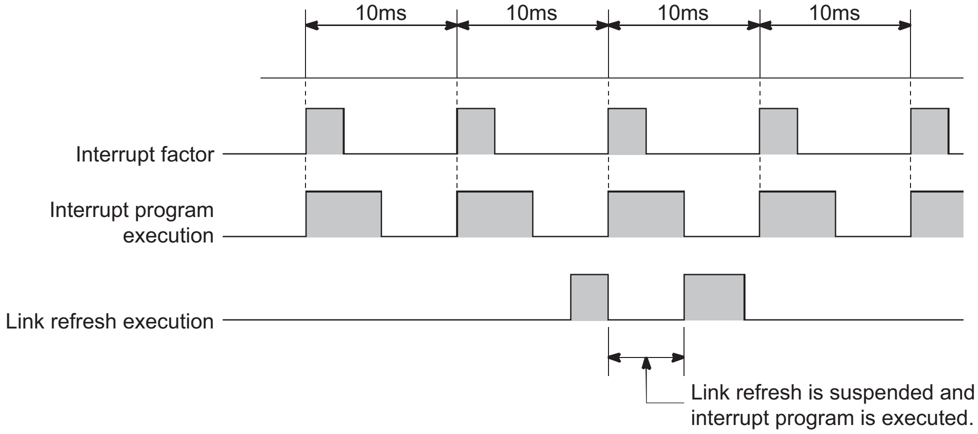

10ms
10ms
10ms
10ms
Interrupt factor
Interrupt program execution
Link refresh execution
Link refresh is suspended and interrupt program is executed.

Figure 2.15 When an interrupt factor occurs during link refresh

Remark

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.

(4) Processing at program execution type change

When the program execution type is changed from the scan execution type or low-speed execution type to the interrupt, the CPU module saves and restores the following data. (☞ Section 9.6.3)

- 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. (☞ Section 10.1.2)

(5) Restrictions

(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.

1) 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.

2) 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. Note2.2

Note2.2 Basic

For the Basic model QCPU, data inconsistency may occur in units of argument.

2.3 Settings When Program is Divided Note2.3

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.

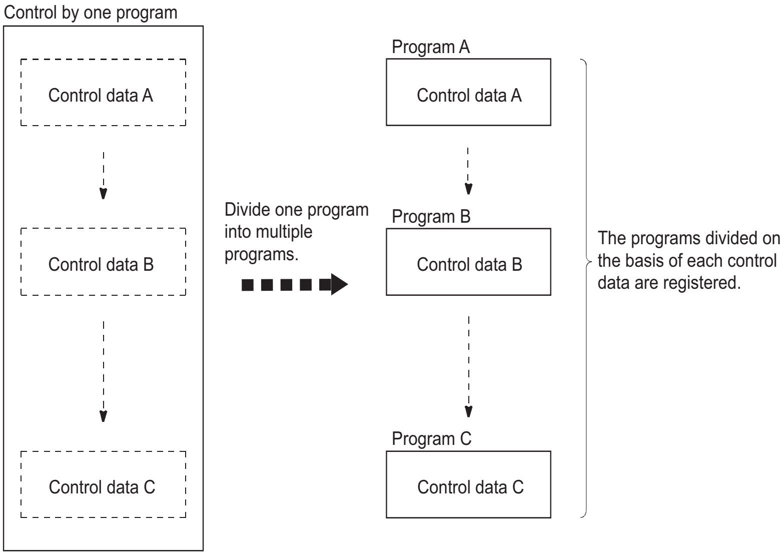


Figure 2.16 Control by multiple programs

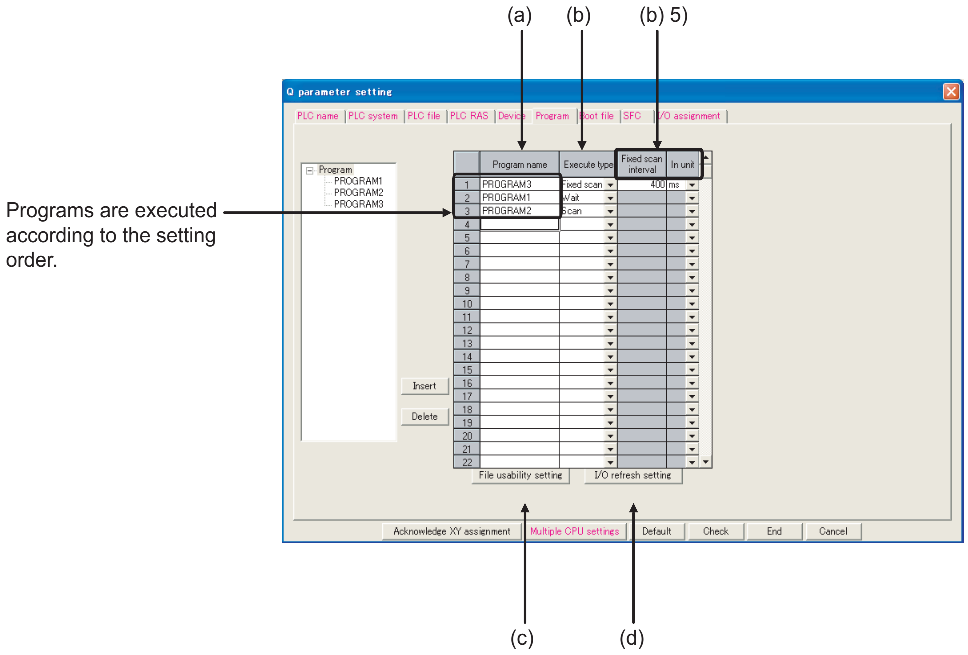
Note2.3 Basic

The Basic model QCPU cannot execute multiple programs.

(2) Settings required for execution of multiple programs

To execute multiple programs in the CPU module, names (file names) and execution conditions of the programs must be set.

Set them in the Program tab of the PLC parameter dialog box.



Programs are executed according to the setting order.

	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Fixed scan	400	ms
2	PROGRAM1	Wait		
3	PROGRAM2	Scan		
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				

Figure 2.17 Program setting

(a) Program name

Enter the name (file name) of the program to be executed in the CPU module.

(b) Execute type

Select an execution type of the program set under "Program name".

The CPU module executes programs whose execution type has been set here according to the setting order.

1) 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. (☞ Section 2.3.1)

2) Scan execution type ("Scan")

This program is executed once in every scan, starting in the next scan of which the initial execution type program is executed and later. (☞ Section 2.3.2)

3) Low-speed execution type program ("Low speed") Note2.4

This program is executed only when the constant scan function is used or a low-speed program execution time value is set. (☞ Section 2.3.3)

4) Stand-by type ("Wait")

This program is executed only when its execution is requested. (☞ Section 2.3.4)

Note2.4 Redundant

The Redundant CPU does not support the use of low-speed execution type programs. Therefore, "Low speed" cannot be selected.

5) Fixed scan execution type ("Fixed scan")

This program is executed at time intervals specified with fixed scan interval and unit. (☞ Section 2.3.5)

- Fixed scan interval ("Fixed scan interval")

  Enter the execution interval of fixed scan execution type program.

  The setting range varies depending on the setting unit.

  •When the unit is "ms" : 0.5 to 999.5ms (in increments of 0.5ms)

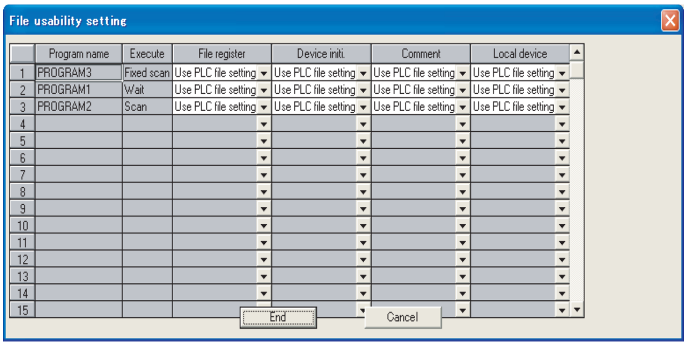
  •When the unit is "s" : 1 to 60s (in increments of 1s)

- Unit ("In unit")

  Select the unit ("ms" or "s") of the fixed scan interval.

(c) File usability setting

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



	Program name	Execute	File register	Device initi.	Comment	Local device
1	PROGRAM3	Fixed scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
2	PROGRAM1	Wait	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
3	PROGRAM2	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Figure 2.18 File usability setting

The default is set to "Use PLC file setting".

Table2.1 shows the processing when "Not used" is selected.

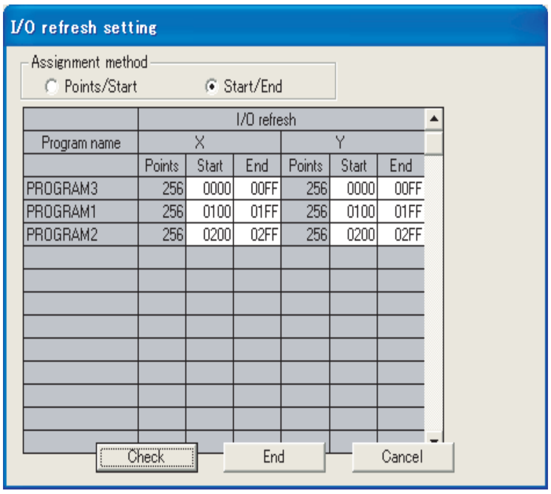
Table2.1 File usability setting item and processing when "Not used" is selected

Item	Processing
File register	The file register cannot be used in the program.
Initial device value	Initial device values are not set when the file name is same as the program file.
Device comment	Device comments cannot be used in the program.
Local device	Data in local devices cannot be saved or restored when program type is changed.

(d) I/O refresh setting

The CPU module updates all inputs and outputs of the input/output modules and intelligent function modules by I/O refresh. (☞ Section 3.8.1)

I/O refresh setting allows I/O refresh to be performed for each program (within the specified range).



Assignment method
○ Points/Start ● Start/End

Program name	I/O refresh					
	X			Y		
	Points	Start	End	Points	Start	End
PROGRAM3	256	0000	00FF	256	0000	00FF
PROGRAM1	256	0100	01FF	256	0100	01FF
PROGRAM2	256	0200	02FF	256	0200	02FF

Figure 2.19 I/O refresh setting

1) Application

This setting is useful when fixed scan execution type programs are used. The CPU module takes in inputs (X) before execution of each fixed scan execution type program and externally outputs the operation results (outputs (Y).

Point

Scan time of the program being executed (except for fixed scan execution type programs) can be checked on the Program monitor list screen. (☞ Section 6.13.1)

(3) Program sequence in the CPU module

Figure 2.20 shows the program sequence after the CPU module is powered on or its operating status is changed from STOP to RUN.

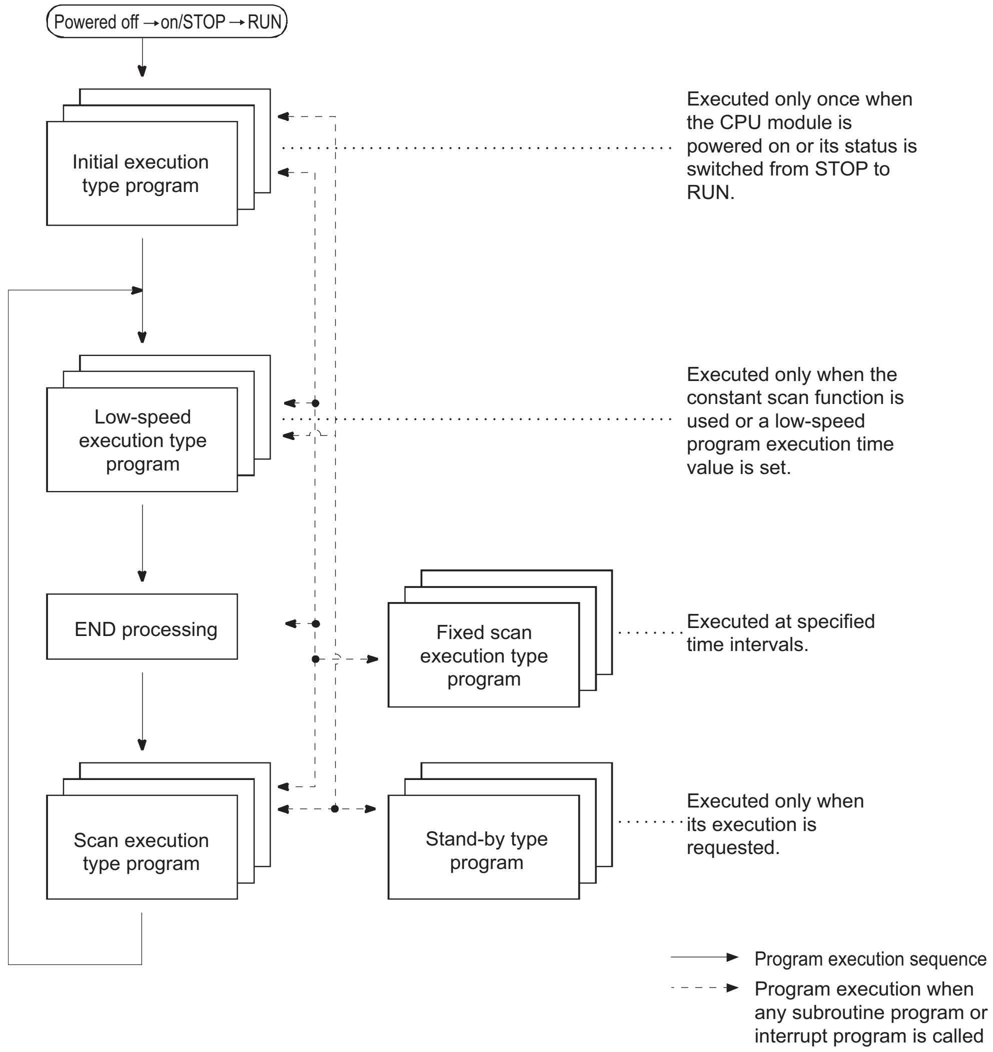

Powered off →on/STOP→RUN
Initial execution type program
Executed only once when the CPU module is powered on or its status is switched from STOP to RUN.
Low-speed execution type program
Executed only when the constant scan function is used or a low-speed program execution time value is set.
END processing
Fixed scan execution type program
Executed at specified time intervals.
Scan execution type program
Stand-by type program
Executed only when its execution is requested.
Program execution sequence
Program execution when any subroutine program or interrupt program is called

Figure 2.20 Program sequence

Point

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

2.3.1 Initial execution type program

(1) Definition

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.

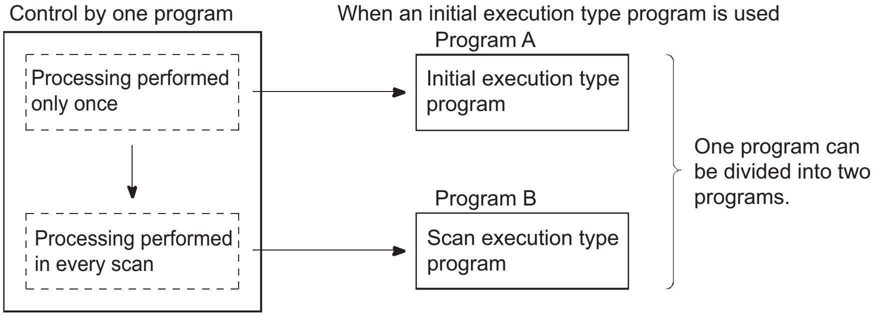

Control by one program
When an initial execution type program is used
Processing performed only once
Processing performed in every scan
Program A
Initial execution type program
Program B
Scan execution type program
One program can be divided into two programs.

Figure 2.21 When processing performed only once is separated as an initial execution type program

(2) 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.

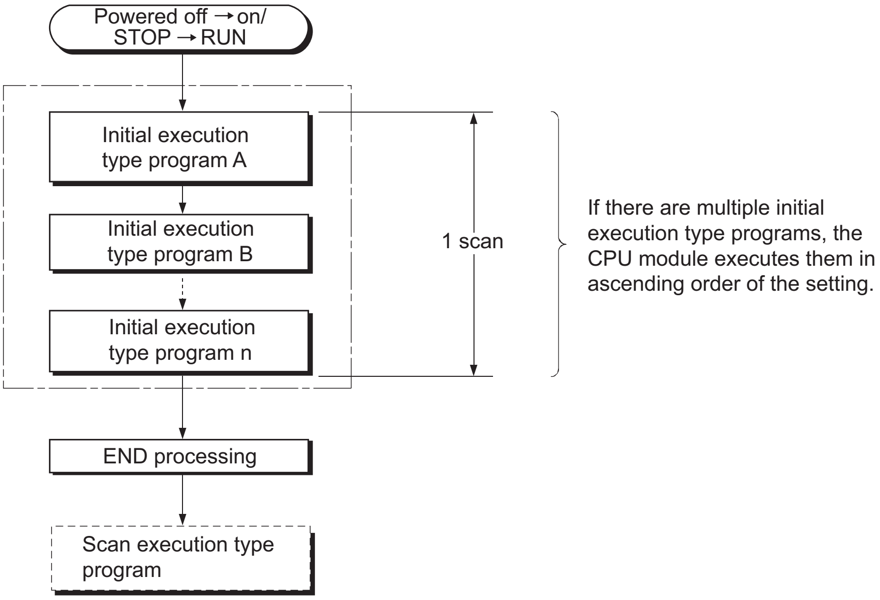

Powered off → on/
STOP → RUN
Initial execution type program A
Initial execution type program B
Initial execution type program n
1 scan
If there are multiple initial execution type programs, the CPU module executes them in ascending order of the setting.
END processing
Scan execution type program

Figure 2.22 Execution order of the initial execution type programs

(b) Initial scan time

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

1) 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.

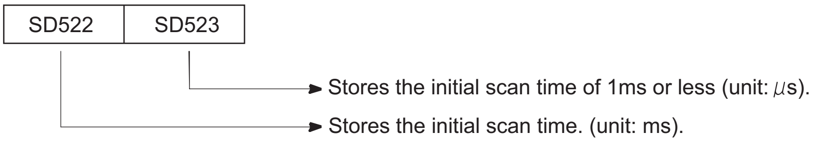

SD522
SD523
Stores the initial scan time of 1ms or less (unit: μs).
Stores the initial scan time. (unit: ms).

Figure 2.23 Initial scan time storage location

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

2) 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.

3) 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.

(3) Precautions on programming

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

Example SEND, RECV, and similar instructions

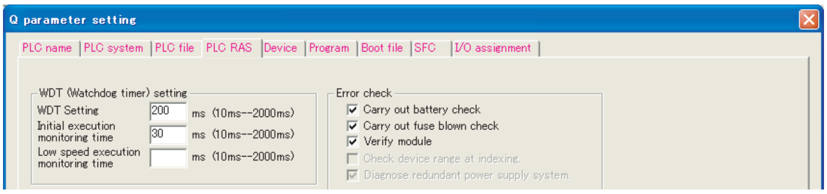
(4) Initial execution monitoring time setting

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.



WDT (Watchdog timer) setting

WDT Setting 200 ms (10ms--2000ms)

Initial execution monitoring time 30 ms (10ms--2000ms)

Low speed execution monitoring time ms (10ms--2000ms)

Error check

☑ Carry out battery check

☑ Carry out fuse blown check

☑ Verify module

□ Check device range at indexing.

☑ Diagnose redundant power supply system.

Figure 2.24 PLC RAS setting (Initial execution monitoring time)

(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

- When an initial execution type program and low-speed execution type program are to be executed, the low-speed execution type program (☞ Section 2.3.3) is executed after completion of the initial execution type program.
  Set a time value so that the initial execution monitoring time becomes longer than total execution time required for both programs.

- 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.

2.3.2 Scan execution type program

(1) Definition

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.

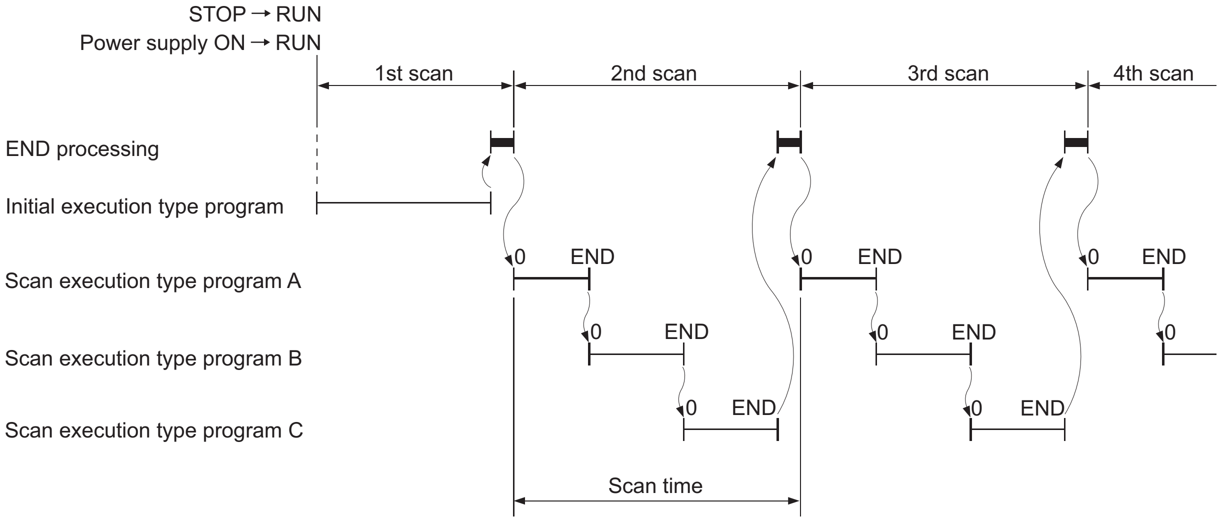

STOP → RUN
Power supply ON → RUN
1st scan
2nd scan
3rd scan
4th scan
END processing
Initial execution type program
Scan execution type program A
Scan execution type program B
Scan execution type program C
0
END
Scan time

Figure 2.25 Execution order of the scan execution type program

(2) Processing

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.

2.3.3 Low-speed execution type program Note2.5

(1) Definition

Low-speed execution type program is executed only when the constant scan function is used or a low-speed program execution time value is set.

This type of program can be used for programs that are not necessary to be executed in every scan (such as a program for printer output).

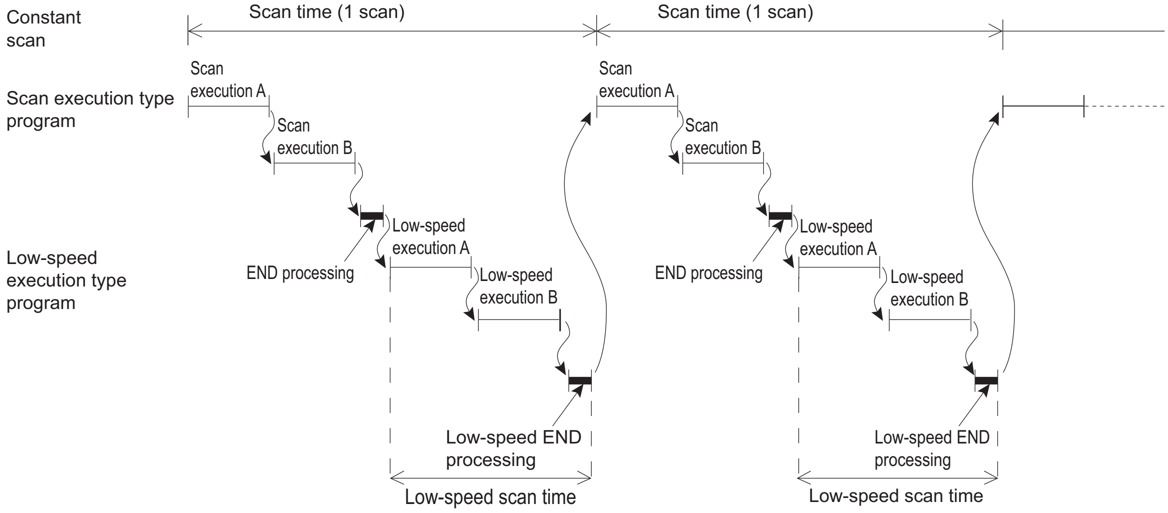


Figure 2.26 Execution of a low-speed execution type program (When executed within the remaining time in each constant scan)

(2) Processing

(a) Program operation

Program operation differs depending on the setting in the PLC RAS tab of the PLC parameter dialog box (☞ (4) in this section).

Set the parameter as necessary.

1) When keeping each scan time constant and giving priority to control accuracy

Set a constant scan time value.

2) When securing the time for execution of low-speed execution type programs

Set a low-speed program execution time value.

Note2.5 Redundant

The Redundant CPU does not support the use of low-speed execution type programs.

(b) When time is still remained in scan time after execution of all low-speed execution type programs

Processing performed after execution of all low-speed execution type programs differs depending on the on/off status of the special relay (SM330) and the execution condition of low-speed execution type programs.

1) Asynchronous mode (SM330 = off)

Execution of low-speed execution type programs is continued within the remaining time.

2) Synchronous mode (SM330 = on)

Execution of low-speed execution type programs is not continued within the remaining time. Program operations are started in the next scan.

Table2.2 Processing depending on the operation mode and execution condition

Low-speed execution type program operation mode	Status of SM330	Execution condition of low-speed execution type program	
		When a constant scan value is set	When a low-speed program execution time value is set
Asynchronous mode	Off	Low-speed execution type programs are re-executed.[*1]	Low-speed execution type programs are re-executed.[*2]
Synchronous mode	On	Waiting time remains in constant scan.[*3]	Execution of scan execution type program is started.[*4]

*1: Low-speed execution type programs are repeatedly executed during the remaining time in each constant scan.
For this reason, the execution time of the low-speed execution type programs varies in each scan. (☞ Figure 2.27)

*2: Low-speed execution type programs are repeatedly executed for the period of time set for low-speed program execution time.
For this reason, the scan time varies in each scan. (☞Figure 2.29)

*3: Time remained after completion of low-speed END processing is regarded as the waiting time.
At the specified constant scan time interval, scan execution type programs are executed.
For this reason, each scan time is constant. (☞ Figure 2.28)

*4: Time remained after after completion of low-speed END processing is ignored and scan execution type programs are executed.
For this reason, the scan time varies in each scan. (☞ Figure 2.30)

<<When a constant scan time value is set>>

The following timing charts show the CPU module operation when low-speed execution type programs are executed under the conditions given below.

- Constant scan time value : 8ms
- Total execution time of scan execution type programs : 4 to 5ms
- Execution time of the low-speed execution type program A : 1ms
- Execution time of the low-speed execution type program B : 3ms
- END processing/low-speed END processing : 0ms (assumed to be 0ms for easy understanding)

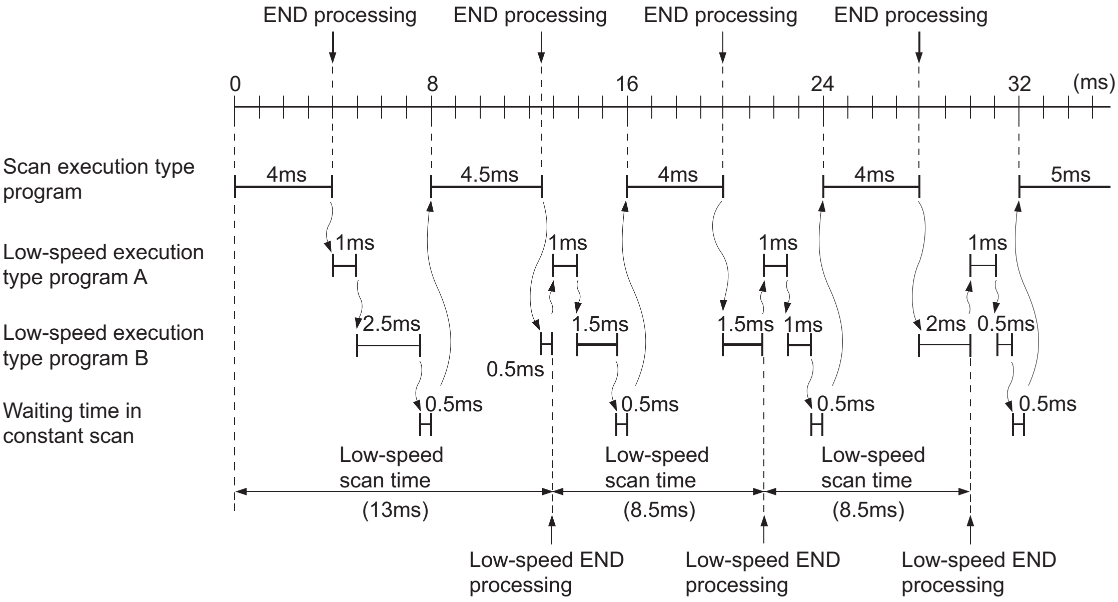


Figure 2.27 Asynchronous mode (SM330: off)

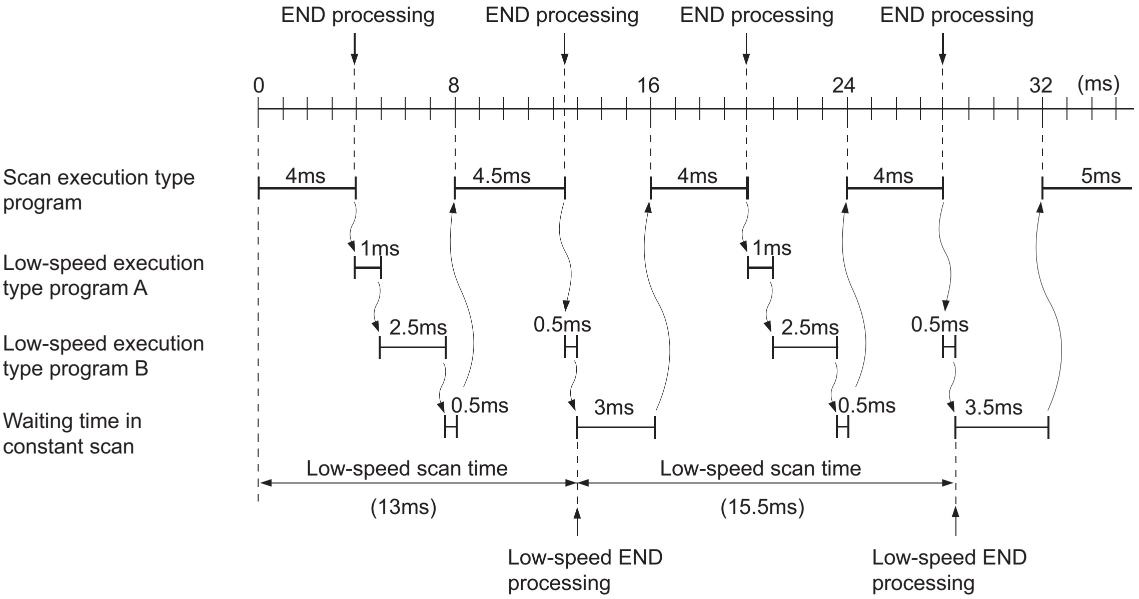


Figure 2.28 Synchronous mode (SM330: on)

<<When a low-speed program execution time value is set>>

The following timing charts show the CPU module operation when low-speed execution type programs are executed under the conditions given below.

- Low-speed program execution time value : 3ms
- Total execution time of scan execution type programs : 4 to 5ms
- Execution time of the low-speed execution type program A : 1ms
- Execution time of the low-speed execution type program B : 3ms
- END processing/low-speed END processing : 0ms (assumed to be 0ms for easy understanding)

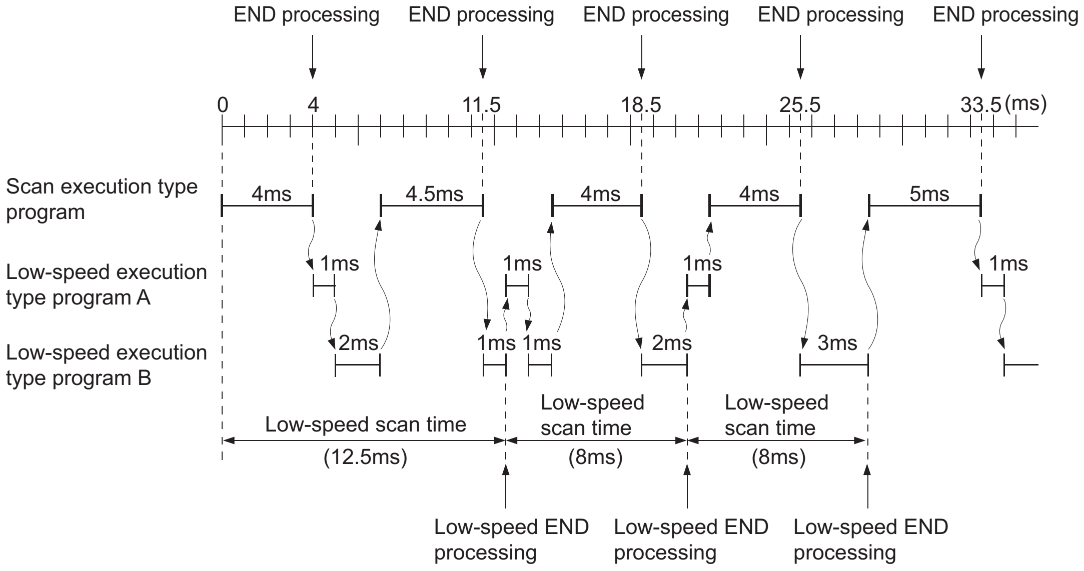


Figure 2.29 Asynchronous mode (SM330: off)

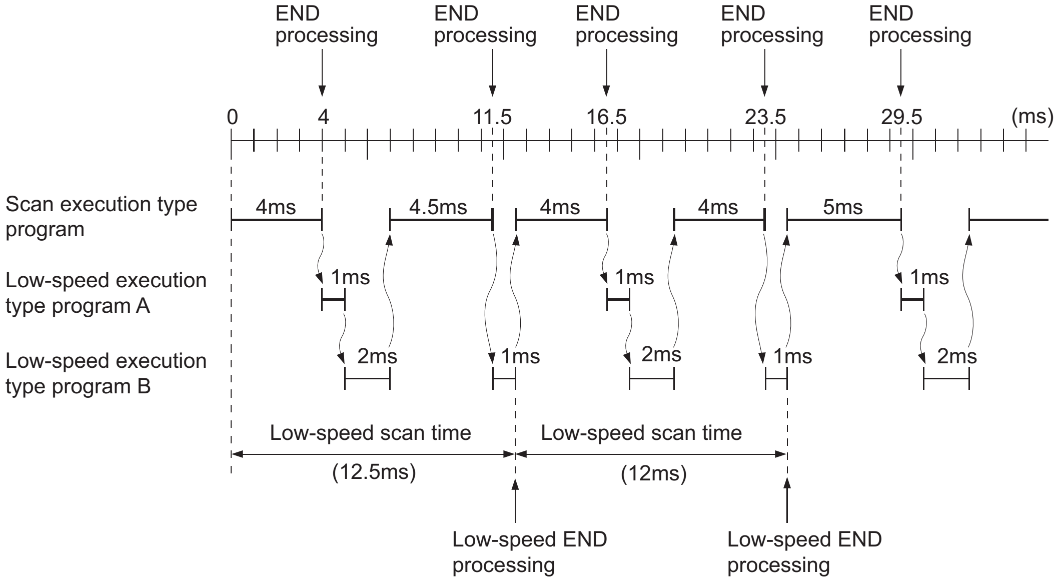


Figure 2.30 Synchronous mode (SM330: on)

(c) When low-speed execution type programs were not processed within the remaining time in each constant scan or the low-speed program execution time

Program execution is stopped and the rest of the program is executed in the next scan.

(d) Low-speed END processing

When all the low-speed execution type programs are executed, the CPU module performs low-speed END processing.

Low-speed END processing includes the following.

- Setting of the special relay and special register for low-speed execution type programs*1
- Online change of low-speed execution type programs
- Measurement of low-speed scan time
- Reset of a watchdog timer for low-speed execution type programs

If SM330 is on and there is still a remaining time in a constant scan, the CPU module reexecutes the first low-speed execution type program after the low-speed END processing.

*1: The special relay and special register for low-speed execution type programs are as follows.

- SM330, SM404, SM405, SM510
- SD430, SD510, SD528 to SD535, SD544 to SD547

Point

When low-speed execution type programs are executed, the constant scan may increase by the maximum processing time of the instructions being executed + the low-speed END processing time.

Remark

For differences between low-speed END processing and END processing, refer to Figure 2.26.

(e) Low-speed scan time

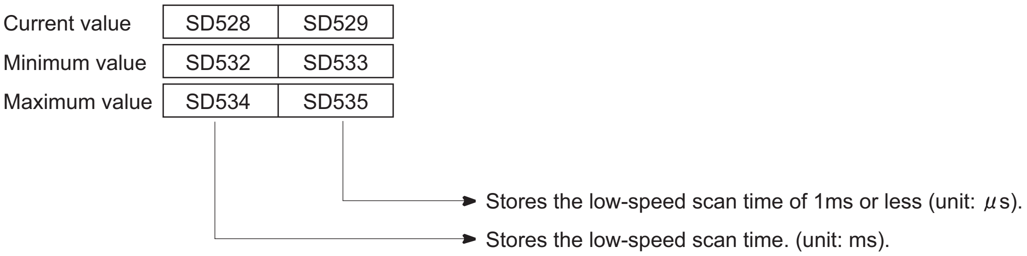
Low-speed scan time is a total of the execution time of all low-speed execution type programs and the low-speed END processing time.

For differences between the low-speed scan time and the scan time, refer to Figure 2.31.

1) Low-speed scan time storage location

The CPU module measures the low-speed scan time and stores it into the special register (SD528 to SD535).

The low-speed scan time can be checked by monitoring SD528 to SD535.



Current value	SD528	SD529
Minimum value	SD532	SD533
Maximum value	SD534	SD535

Stores the low-speed scan time of 1ms or less (unit: μs).
Stores the low-speed scan time. (unit: ms).

Figure 2.31 Low speed scan time storage location

Example If the stored values in SD528 and SD529 are 50 and 400 respectively, the low-speed scan time is 50.4ms.

Remark

The low-speed scan time and program execution time can be checked on the Program monitor list screen. (Section 6.13.1)

2) Accuracy and measurement of low-speed scan time

Accuracy of each low-speed scan time stored in the special register is ±0.1ms.

Even if the WDT instruction (instruction that resets the watchdog timer) is executed in a sequence program, the measurement of each low-speed scan time continues.

3) Execution of an interrupt program or fixed scan execution type program

The execution time of an interrupt program or fixed scan execution type program will be added to the low-speed scan time.

(3) Precautions on programming

(a) Low-speed program execution time value setting

Setting a low-speed program execution time value increases the scan time since the set period of time is secured.

Set a low-speed program execution time value so that the scan time becomes shorter than the watchdog timer or increase the WDT setting value.

(b) Unavailable instruction

The COM instruction is not available for low-speed execution type programs.

(c) Execution timing

Low-speed execution type programs are executed even in the same scan where the initial execution type program is executed.

To prevent low-speed execution type programs from being executed after the initial execution type program, provide an interlock with the special relay (SM402 or SM403).

(d) Setting range

Set either a constant scan time value or low-speed program execution time value.

If both values are set, "PRG. TIME OVER" (error code: 5010) occurs under the following condition: (remaining time in a constant scan) < (low-speed program execution time).

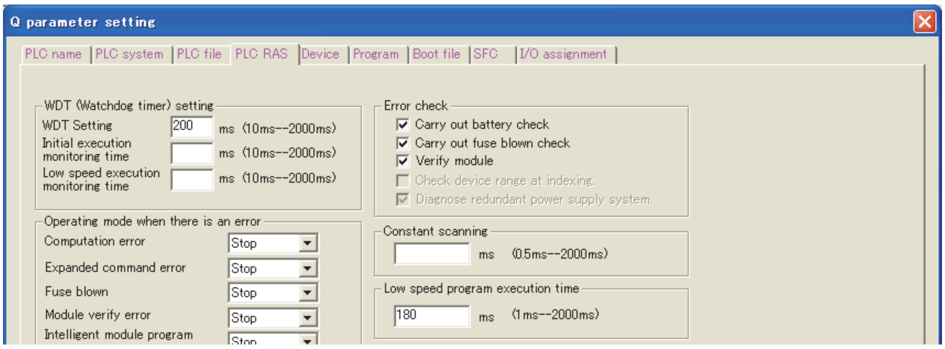
(e) Processing at program execution type change

For how to save and restore data in the index register when the program execution type is changed, refer to Section 9.6.3.

(4) Settings for low-speed execution type program execution

Set the following in the PLC RAS tab of the PLC parameter dialog box.

- Constant scan time value ("Constant scanning")· · · Setting range: 0.5 to 2000ms (in increments of 0.5ms)
- Low-speed program execution time value ("Low speed program execution time")· · · Setting range: 1 to 2000ms (in increments of 1ms)



WDT (Watchdog timer) setting
WDT Setting 200 ms (10ms--2000ms)
Initial execution monitoring time ms (10ms--2000ms)
Low speed execution monitoring time ms (10ms--2000ms)

Error check
☑ Carry out battery check
☑ Carry out fuse blown check
☑ Verify module
□ Check device range at indexing
☑ Diagnose redundant power supply system

Operating mode when there is an error
Computation error Stop
Expanded command error Stop
Fuse blown Stop
Module verify error Stop
Intelligent module program Stop

Constant scanning
ms (0.5ms--2000ms)

Low speed program execution time
180 ms (1ms--2000ms)

Figure 2.32 PLC RAS setting (Constant scan, low-speed program execution time, and low-speed execution monitoring time)

Point

To execute low-speed execution type programs, make sure to set either a constant scan time value or low-speed program execution time value.

To monitor the execution time of low-speed execution type programs, set a low-speed execution monitoring time value.

The setting range is 10 to 2000ms (in increments of 10ms).

No default value is set.

(a) When the low-speed scan time exceeds the preset low-speed execution monitoring time

"PRG TIME OVER" (error code: 5010) occurs.

Point

When executing low-speed execution type programs and initial execution type programs, the low-speed execution type programs will be executed after the completion of the initial execution type programs (☞ Section 2.3.1).
Set an initial execution monitoring time value which is longer than a total of the initial scan time and the execution time of low-speed execution type programs.

2.3.4 Stand-by type program

(1) Definition

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.

(2) 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.

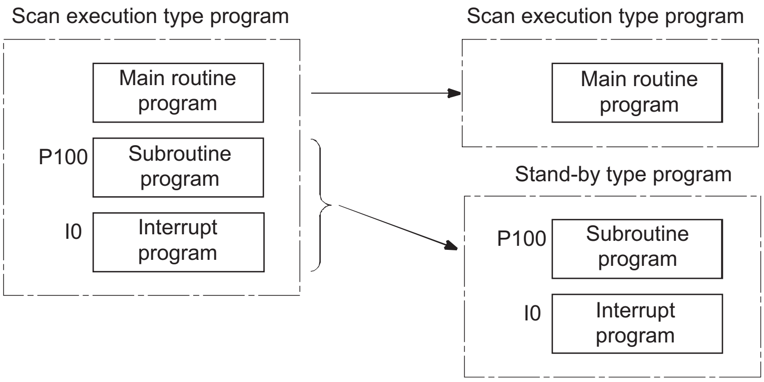


Figure 2.33 Program library using a stand-by type program

(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.

(3) 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.

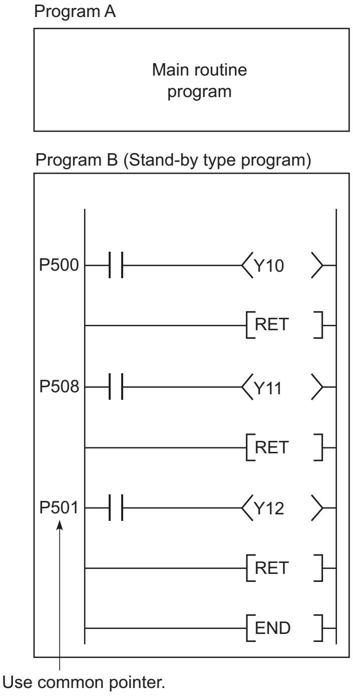


Figure 2.34 Creating subroutine programs in a single stand-by type program

1) Executing a subroutine program and interrupt program in a stand-by type program

After execution of the stand-by type program, the CPU module reexecutes the program that called a program in the stand-by type program.

Figure 2.35 shows the operation when the subroutine and interrupt programs in the stand-by type program are executed.

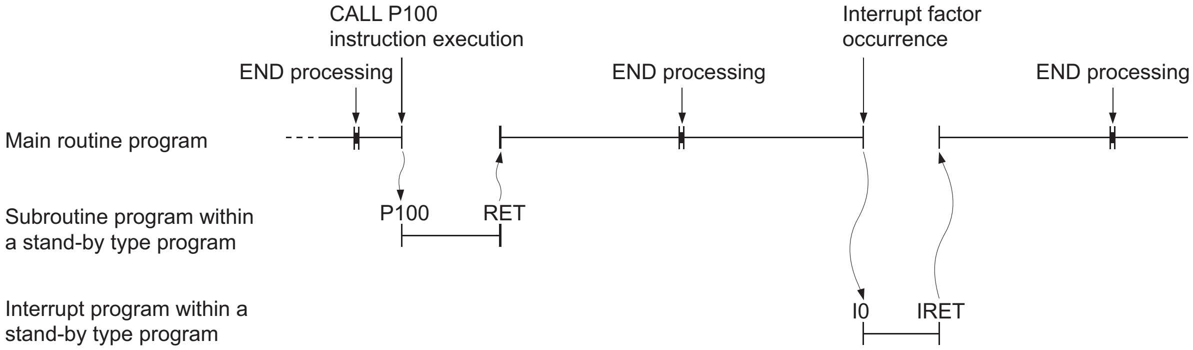


Figure 2.35 Operation when the subroutine and interrupt programs in the stand-by type program are executed

Point

- For restrictions on programming of subroutine and interrupt programs, refer to the following.
    - Subroutine program : ☞ Section 2.2.2
    - Interrupt program : ☞ Section 2.2.3

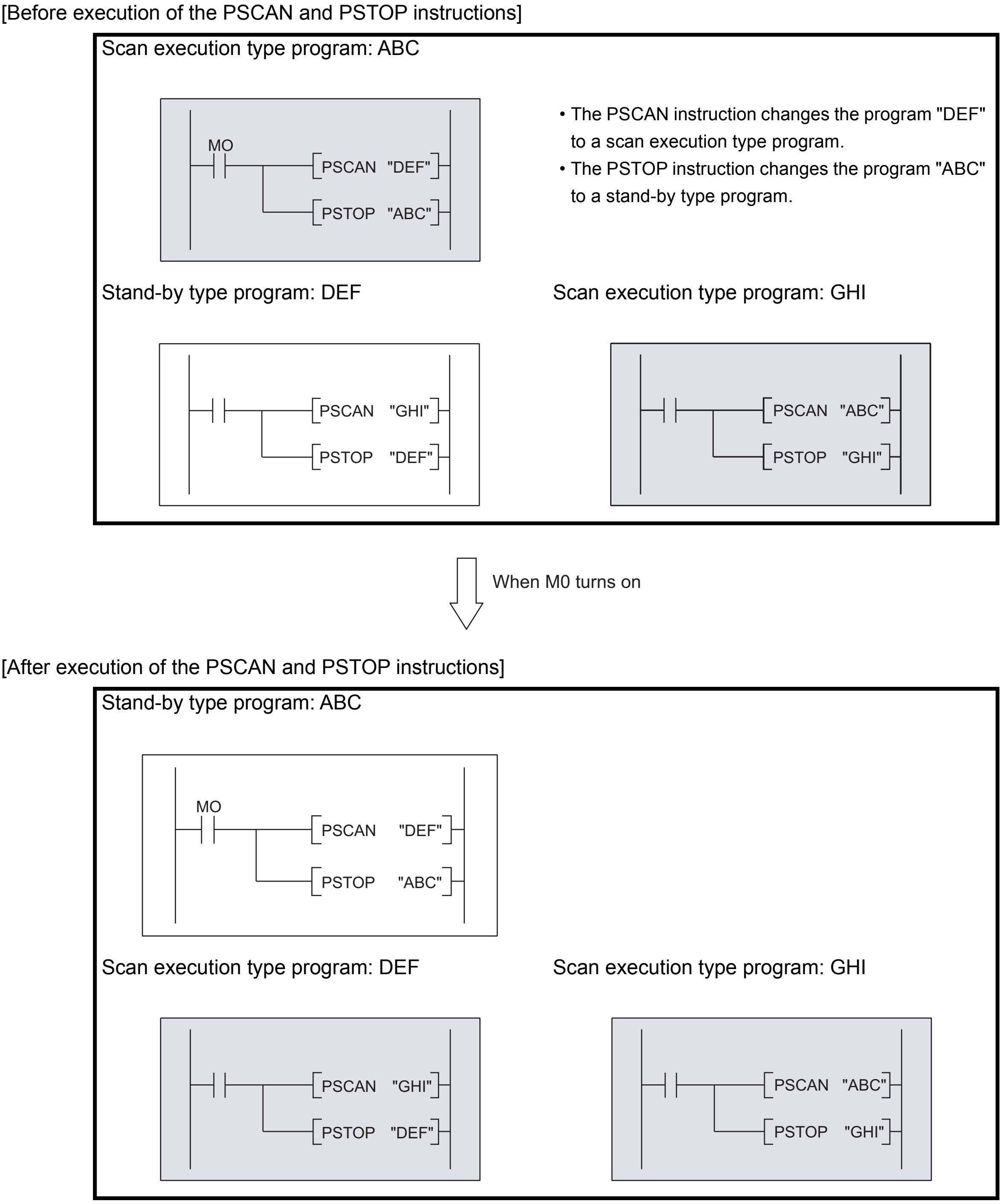
- Use common pointers. (☞ Section 9.9.2)
  If local pointers are used, subroutine programs in a stand-by type program cannot be executed from any other program.

(b) Changing the program execution type using instructions

Use the PSCAN, PSTOP, or POFF instruction to change a program execution type.

1) Changing the execution type (in the case of scan execution type program)

- Set the programs "ABC" and "GHI" as scan execution type programs and the program "DEF" as a stand-by type program.
- When the condition is established (the internal relay (M0) in Figure 2.36 turns on), the program "DEF" is changed to a scan execution type program and the program "ABC" to a stand-by type program.


[Before execution of the PSCAN and PSTOP instructions]
Scan execution type program: ABC
MO
PSCAN "DEF"
PSTOP "ABC"
Stand-by type program: DEF
PSCAN "GHI"
PSTOP "DEF"
Scan execution type program: GHI
PSCAN "ABC"
PSTOP "GHI"
When M0 turns on
[After execution of the PSCAN and PSTOP instructions]
Stand-by type program: ABC
MO
PSCAN "DEF"
PSTOP "ABC"
Scan execution type program: DEF
PSCAN "GHI"
PSTOP "DEF"
Scan execution type program: GHI
PSCAN "ABC"
PSTOP "GHI"

- The PSCAN instruction changes the program "DEF" to a scan execution type program.
- The PSTOP instruction changes the program "ABC" to a stand-by type program.

Figure 2.36 Example of changing the execution type (in the case of scan execution type program)

2) Execution type change timing

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.

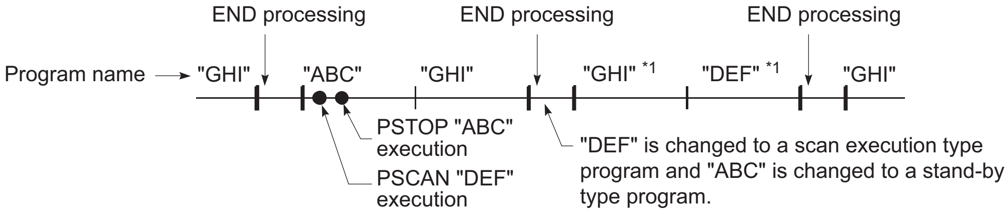


Figure 2.37 Execution type change timing

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

(4) 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 Section 9.13.2.

2.3.5 Fixed scan execution type program

(1) Definition

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 Section 2.2.3(2)(b).

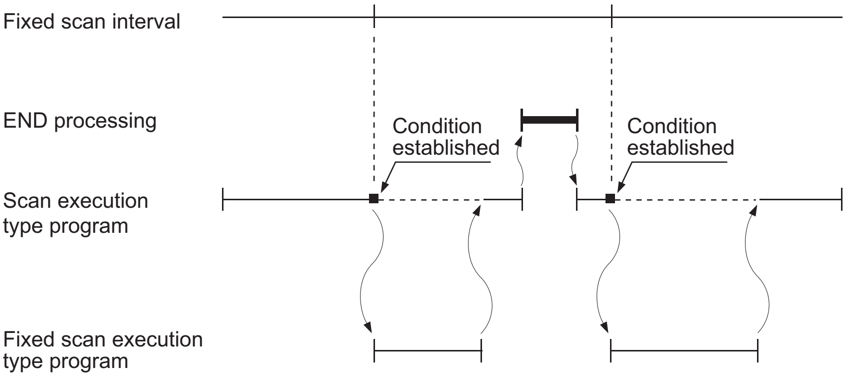


Figure 2.38 Execution of a fixed scan execution type program

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.

(2) 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 Controller Network 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.

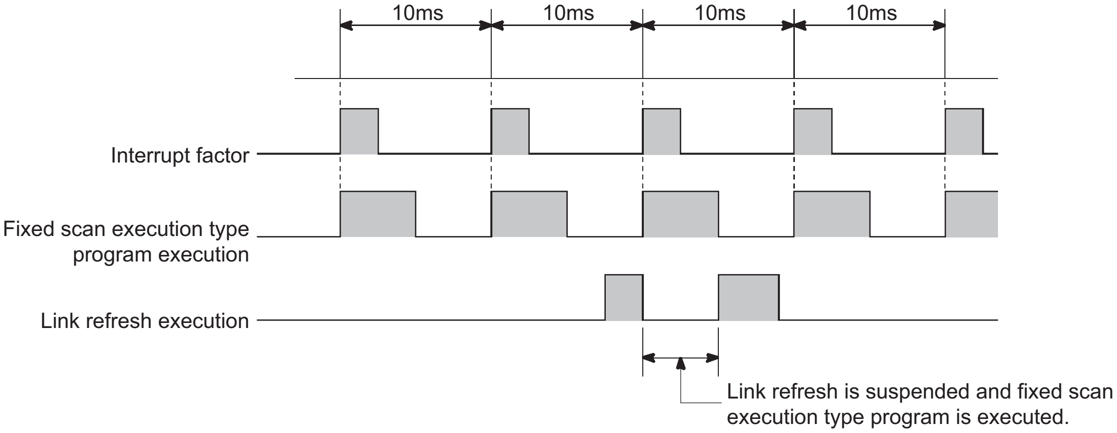

10ms
10ms
10ms
10ms
Interrupt factor
Fixed scan execution type program execution
Link refresh execution
Link refresh is suspended and fixed scan execution type program is executed.

Figure 2.39 When the execution condition is established during link refresh

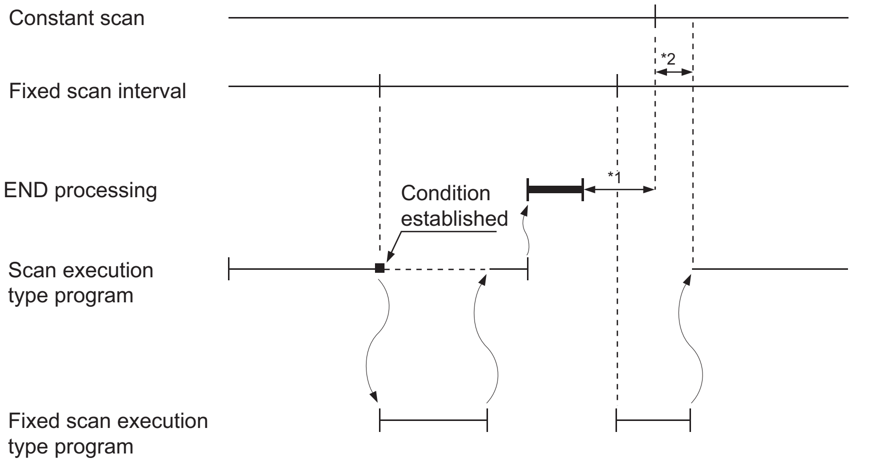
Remark

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 waiting time of the constant scan execution or 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.

Figure 2.40 When the execution condition is established during the waiting time

(3) 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 Section 2.2.3(4). (The method is the same as that for interrupt programs.)

(4) 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.

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

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

Example

- 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.

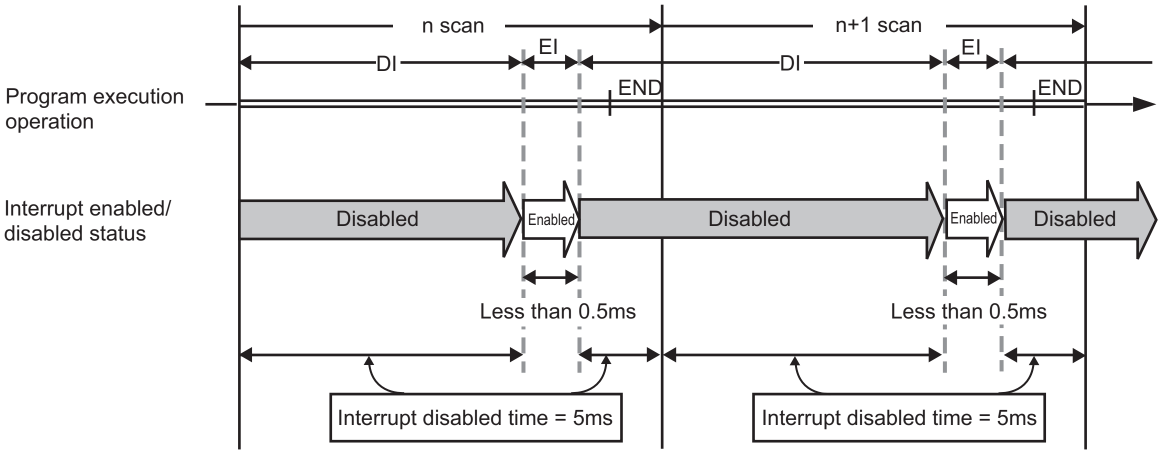

n scan
n+1 scan
DI
EI
DI
EI
END
END
Program execution operation
Interrupt enabled/ disabled status
Disabled
Enabled
Disabled
Enabled
Disabled
Less than 0.5ms
Less than 0.5ms
Interrupt disabled time = 5ms
Interrupt disabled time = 5ms

Figure 2.41 Program execution and interrupt enabled/disabled status

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.

2.3.6 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, PLOW, Note2.6, PSTOP, or POFF instruction to change the execution type.

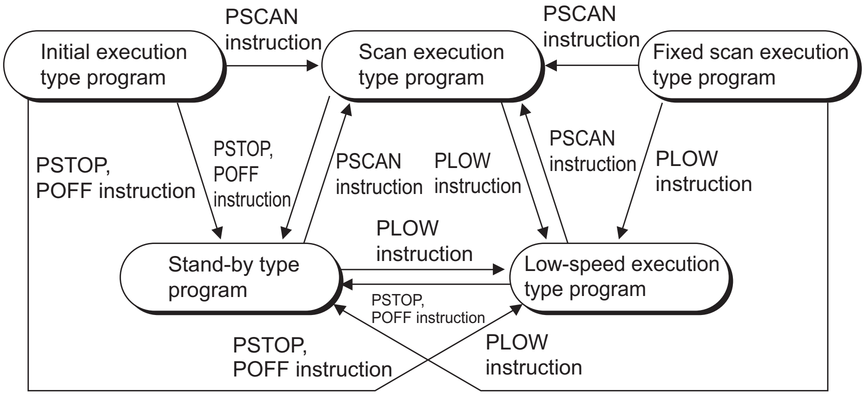


Figure 2.42 Pattern of execution type change

Table2.3 Timing of execution type change

Execution type before change	Instruction			
	PSCAN	PSTOP	POFF	PLOW
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.	Changes to the low-speed execution type.
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	
Low-speed execution type	Stops execution of the low-speed execution type program and changes to the scan execution type in the next scan. (Executed from the step 0.)	Stops execution of the low-speed execution type program and changes to the stand-by type.	Stops execution of the low-speed execution type program and turns off outputs in the next scan. Changes to the stand-by type in two scans later.	Remains unchanged.
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.	Changes to the low-speed execution type.

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.

Note2.6 Redundant

The Redundant CPU does not support the use of low-speed execution type programs.

(b) Execution type change example

In a control program, a stand-by 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 stand-by type program.

Figure 2.43 shows the case where the execution type of the stand-by type programs "ABC", "DEF", "GHI", and "JKL" are changed in the control program.

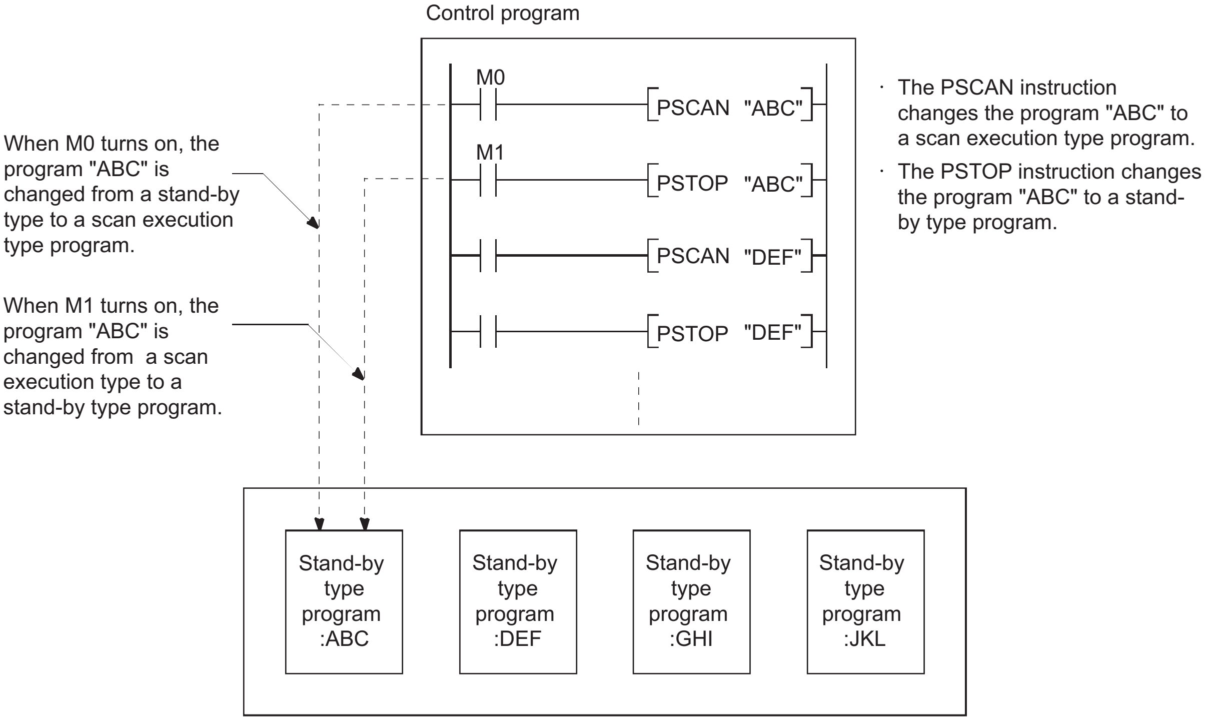

Control program
M0
PSCAN "ABC"
M1
PSTOP "ABC"
PSCAN "DEF"
PSTOP "DEF"
When M0 turns on, the program "ABC" is changed from a stand-by type to a scan execution type program.
When M1 turns on, the program "ABC" is changed from a scan execution type to a stand-by type program.
· The PSCAN instruction changes the program "ABC" to a scan execution type program.
· The PSTOP instruction changes the program "ABC" to a stand-by type program.
Stand-by type program :ABC
Stand-by type program :DEF
Stand-by type program :GHI
Stand-by type program :JKL

Figure 2.43 Execution type change example

(2) Changing the execution type from the Program monitor list screen

The execution type of programs can be changed on the screen opened by selecting [Online] → [Monitor] → [Program monitor list]. (☞ Section 6.13.1)

2.4 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.
Table2.4 shows the numeric representations of BIN, HEX, BCD, and DEC (decimal).

Table2.4 Numeric representations of BIN, HEX,BCD,and DEC

DEC (Decimal)	HEX (Hexadecimal)	BIN (Binary)				BCD (Binary-coded decimal)			
0	0				0				0
1	1				1				1
2	2				10				10
3	3				11				11
. . .	. . .				. . .				. . .
9	9				1001				1001
10	A				1010			1	0000
11	B				1011			1	0001
12	C				1100			1	0010
13	D				1101			1	0011
14	E				1110			1	0100
15	F				1111			1	0101
16	10			1	0000			1	0110
17	11			1	0001			1	0111
. . .	. . .				. . .				. . .
47	2F			10	1111			100	0111
.	.								
.	.								
.	.								
32766	7FFE	0111	1111	1111	1110	---			
32767	7FFF	0111	1111	1111	1111	---			
-32768	8000	1000	0000	0000	0000	1000	0000	0000	0000
-32767	8001	1000	0000	0000	0001	1000	0000	0000	0001
. . .	. . .								
-2	FFFE	1111	1111	1111	1110	---			
-1	FFFF	1111	1111	1111	1111	---			

(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 in (b).

(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. (☞ (1)(b) in this section)

(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.

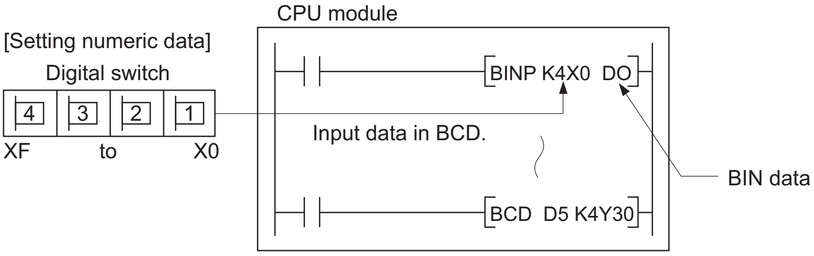


Figure 2.44 Inputting data from a digital switch to the CPU module

Remark

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.

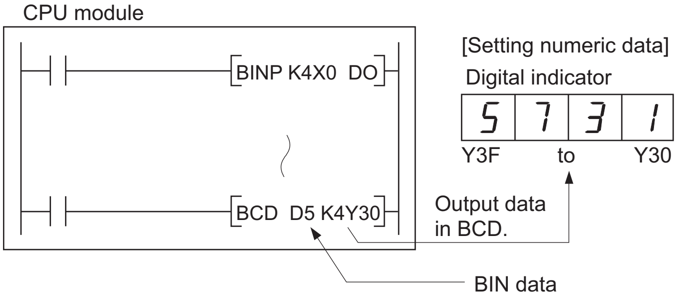


Figure 2.45 Display of operation results in the CPU module by a digital indicator

Remark

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

☞ MELSEC-Q/L Programming Manual (Common Instruction)

2.4.1 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.)
Table2.3 shows the numeric representations in BIN and DEC.

Table2.5 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.
Figure 2.46 shows the numeric representations for registers.

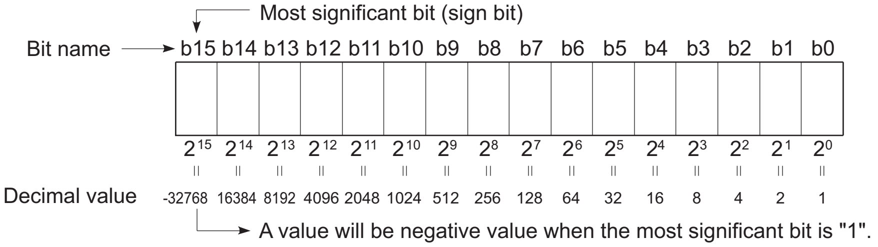

Most significant bit (sign bit)
Bit name
b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0
2^15 2^14 2^13 2^12 2^11 2^10 2^9 2^8 2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0
Decimal value
-32768 16384 8192 4096 2048 1024 512 256 128 64 32 16 8 4 2 1
A value will be negative value when the most significant bit is "1".

Figure 2.46 Numeric representations for registers in the CPU module

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

2.4.2 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 AH to FH to represent values ten to fifteen. After a digit reaches FH, the next-higher digit (to the left) is incremented.

Table2.6 shows the numeric representations in BIN, HEX, and DEC.

Table2.6 Numeric representations in BIN, HEX, and DEC

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 $\mathrm{FFFF_H}$ can be specified in hexadecimal.

2.4.3 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.
Table2.7 shows the numeric representations in BIN, BCD, and DEC.

Table2.7 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.

2.4.4 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] × $2^{[\text{Exponent}]}$

The bit configuration and the meaning of each bit are described below

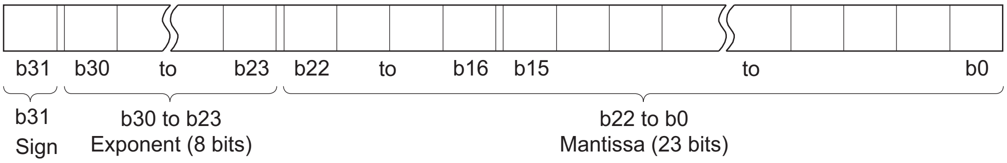


Figure 2.47 Bit configuration of real number data

1) Sign

The most significant bit, b31, is the sign bit.

he most significant bit, b31, is the sign bit.

0: Positive
1: Negative

2) 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	FFH	FEH	FDH			81H	80H	7FH	7EH			02H	01H	00H
n	Not used	127	126			2	1	0	-1			-125	-126	Not used

Figure 2.48 Relation between the exponent and excess n

3) 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) × indicates the numeral system used.)

1) Storing "10"

$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.010000..... \times 2^3)_2$

Sign: Positive → 0

Exponent: 3 → 82_H → $(10000010)_2$

Mantissa: $(010\ 00000\ 00000\ 00000\ 00000)_2$

In this case, the value will be encoded as 41200000_H.

Sign Exponent Mantissa

$$\underbrace{0\ 1\ 0\ 0}_{4}\ \underbrace{0\ 0\ 0\ 1}_{1}\ \underbrace{0\ 0\ 1\ 0}_{2}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}$$

2) Storing "0.75"

$(0.75)_{10} \rightarrow (0.11)_2 \rightarrow (1.100..... \times 2^{-1})_2$

Sign: Positive → 0

Exponent: −1 → $7E_H$ → $(01111110)_2$

Mantissa: $(100\ 00000\ 00000\ 00000\ 00000)_2$

In this case, the value will be encoded as $3F400000_H$.

Sign Exponent Mantissa

$$\underbrace{0\ 0\ 1\ 1}_{3}\ \underbrace{1\ 1\ 1\ 1}_{F}\ \underbrace{0\ 1\ 0\ 0}_{4}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}\ \underbrace{0\ 0\ 0\ 0}_{0}$$

Point

Values after the decimal point (in binary) is calculated as follows.

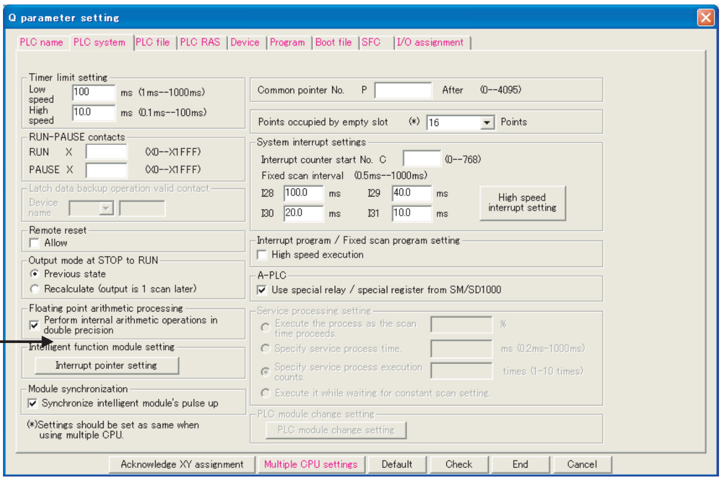
Example $(0.1101)_2$

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}

$(0.1101)_2=2^{-1}+2^{-2}+2^{-4}=0.5+0.25+0.0625=(0.8125)_{10}$

(c) Performing internal operations in double precision Note2.7

Select the checkbox for the Floating-point arithmetic processing parameter in the PLC system tab of the PLC parameter dialog box. (The checkbox is selected by default.)



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
□ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms

Interrupt program / Fixed scan program setting
□ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting

Deselect the checkbox here if internal operations are not necessary to be performed in double precision.

Figure 2.49 PLC system setting

Operation results are represented in single-precision floating-point data regardless of the parameter setting. When the parameter is selected, only internal operations are performed in double precision (64 bits).

Determine whether to select the "Perform internal arithmetic operations in double precision" checkbox according to the purpose as described below.

1) Selecting the checkbox

Select the checkbox when accuracy is required to ensure compatibility with the existing models.
When instructions that perform many real number operations internally, such as the SIN or COM instruction, are used, data accuracy increases by selecting this checkbox.

2) Not selecting the checkbox

Do not select the checkbox when the real number operation processing time needs to be reduced.
Since internal operations are performed in single precision (32 bits), the operation speed is improved.
However, data accuracy may decrease.

Point

- The real number data in the CPU module can be monitored by monitor operation from GX Developer.
  Note that if the data to be monitored cannot be represented in real number (for example, "FFFFH"), "-" will be displayed.
- To represent "0", all bits from b0 to b31 must be "0".

Note2.7 **Basic** **Process** **Redundant**

The "Perform internal arithmetic operation in double precision" checkbox cannot be selected for the Basic model QCPU, Process CPU, and Redundant CPU.

2.4.5 Character string data

(1) Definition

The CPU module uses shift JIS code character strings.

CHAPTER3 CPU MODULE OPERATION

This chapter describes operation of the CPU module.

3.1 Initial Processing

The CPU module performs preprocessing required for sequence program operations.
The preprocessing is executed only once when any of the operations described in Table3.1 is performed to the CPU module.
When initial processing is completed, the CPU module will be placed in the operation status set by the RUN/STOP switch (RUN/STOP/RESET switch for the Basic model QCPU). (☞ Section 3.5)

Table3.1 Initial processing list

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	○	○	×
PLC parameter check	○	○	○
Multiple CPU system parameter consistency check	○	○	○
Initialization of devices outside the latch range *2 (bit device: off, word device: 0)	○	○	×
Automatic I/O number assignment of mounted modules	○	○	○
CC-Link IE Controller Network 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	○	○	×

○:Performed,× :Not performed

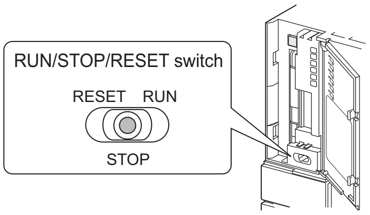
*1: The operation indicates that the status is changed back to RUN without resetting the module after any parameter or program was changed in the STOP status.
The RUN/STOP switch (the RUN/STOP/RESET switch for the Basic model QCPU) is set from STOP to RUN (the RUN LED will flash), then back to STOP and 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.

*2: If the start mode is set to the hot-start mode in the Redundant CPU, devices outside the latch range are not initialized.
(Except some devices, such as the step relay and the index register.)

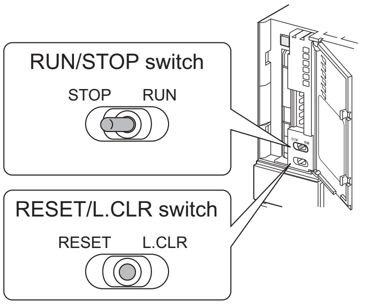
Point

- The switch for STOP, RUN, and RESET of the CPU module differs depending on the CPU module.

Basic model QCPU



High Performance model QCPU, Process CPU, and Redundant CPU



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

3.2 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.)

3.3 Program Operation

The CPU module sequentially executes the program stored in the module from the step 0 to the END or FEND instruction. (☞ CHAPTER 2)

3.4 END Processing

The CPU module performs refresh processing with network modules and communication with external devices.

END processing includes the following.

Table3.2 END processing list

Item	CPU module				Reference
	Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	
Refresh with network modules	○	○	○	○	CHAPTER 10
Auto refresh with intelligent function module	○	○	○	○	Section 7.1.1
Intelligent function module dedicated instruction processing	○	○	○	○	CHAPTER 10
Service processing	○	○	○	○	Section 6.25.1
Watchdog timer reset	○	○	○	○	Section 6.16
Auto refresh between multiple CPU modules	○	○	○	×	QCPU User's Manual (Multiple CPU System
Device data collection using the sampling trace function (only when trace point is set to every scan (after END instruction execution))	×	○	○	○	Section 6.14
Self-diagnostics processing	○	○	○	○	Section 6.17
Special relay/special register value setting (only for those that should be set during END processing)	○	○	○	○	QCPU User's Manual (Hardware Design, Maintenance and Inspection)

○: Executed,×: Not executed

Point

- When the constant scan function (☞ Section 6.2) 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.
- When the low-speed execution type program is executed, low-speed END processing is performed after all low-speed execution type programs are executed. (☞ Section 2.3.3)

3.5 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. (☞ Section 6.4)

- Output (Y) status saved immediately before entering the STOP status
- Result of operations performed for one scan after entering the RUN status

(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. (☞ Section 6.6.1)
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 (☞ Section 6.6.2) after operations are performed for one scan, holding the output and device memory status.

(4) Operation processing in the CPU module when switch operation is performed

Table3.3 Operation processing when switch operation is performed

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. (☞ Section 6.4)	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. (☞ Section 6.4)

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
- 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

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 GX Developer
- Read/write data from/to external devices using the MC protocol
- Communication with other stations using CC-Link IE Controller Network or MELSECNET/H
- Communication with CC-Link remote stations

3.6 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).

For example, 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

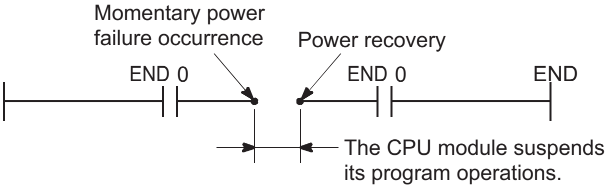

Momentary power failure occurrence
Power recovery
END 0
END 0
END
The CPU module suspends its program operations.

Figure 3.1 Operation processing during a momentary power failure

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

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 GX Developer (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.
  For details of SM53 and SD53, refer to the following.
  ☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

3.7 Data Clear Processing

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

(1) Clearing data in the CPU module

Data in the CPU module are cleared when the reset operation (by the RUN/STOP/RESET switch or by powering the module off and then on) is performed.
However, data in (a) below cannot be cleared by the reset operation.

(a) Data that cannot be cleared by the reset operation

- Data in the program memory
- Data in the standard ROM
- Data in a memory card
- Data in latch-specified devices (☞ (2) in this section)
- Data in the file register

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

1) Data in the program memory

Data can be cleared by:

- selecting the "Clear program memory" checkbox in the Boot file tab of the PLC parameter dialog box, or Note3.1
- selecting [Online] → [Delete PLC data] in GX Developer

2) Data in the standard ROM

Data can be cleared automatically when the data is written to the standard ROM.

3) Data in a memory card

Data can be cleared by selecting [Online] → [Delete PLC data] in GX Developer.

4) Data in latch-specified devices

Refer to (2) in this section.

5) Data in the file register

Data can be cleared by:

- resetting devices with the RST instruction,
- transferring K0 with the MOV or FMOV instruction, or
  ☞ MELSEC-Q/L Programming Manual (Common Instruction)
- executing "Clear all file registers" from the screen opened by selecting [Online] → [Clear PLC memory] in GX Developer.

Note3.1 Basic

The "Clear program memory" checkbox cannot be selected in the Basic model QCPU.

(2) Latch specification of devices

Set a latch range for each latch-target device in the Device tab of the PLC parameter dialog box. (☞ Section 6.3(5))

(a) Latch range setting

Two kinds of latch range can be set by GX Developer.

1) Latch clear operation enable range ("Latch (1) start/end")

Data in this latch range can be cleared by the RESET/L.CLR switch Note3.2 or the remote latch clear operation.

2) Latch clear operation disable range ("Latch (2) start/end")

Data in this latch range cannot be cleared by the RESET/L.CLR switch Note3.2 or the remote latch clear operation.

(b) Clearing device data set in the latch clear operation enable range

Clear data by the RESET/L.CLR switch Note3.2 or the remote latch clear operation (☞ Section 6.6.4).

(c) Clearing device data set in latch clear operation disable range

Clear data by:

- resetting devices with the RST instruction,
- transferring K0 with the MOV or FMOV instruction, or

  ☞ MELSEC-Q/L Programming Manual (Common Instruction)
- executing "Clear device's whole memory (including latch)" from the screen opened by selecting [Online] → [Clear PLC memory] in GX Developer.

Point

If the start mode is set to the hot-start mode in the Redundant CPU, device data outside the latch range are held as well. (Except some devices, such as the step relay and the index register.)

Remark

For the operation of GX Developer, refer to the following.
☞ GX Developer Version 8 Operating Manual

Note3.2 Basic

The Basic model QCPU does not support the latch clear operation by the switch.

3.8 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 (☞ Section 3.8.1)

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 (☞ Section 3.8.2)

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 input/output modules at execution of an instruction. Therefore, data input is faster than in refresh mode.
Processing time required for each instruction, however, is longer.

Table3.4 shows the availability of the refresh mode and the direct mode for each input and output.

Table3.4 Availability of modes

Item	Refresh mode	direct mode
Input/output modules	Available	Not available
Input/output of intelligent function modules		
Input/output of the MELSEC-I/OLINK remote I/O system master module (AJ51T64/A1SJ51T64)*1		
Remote input/output in CC-Link IE Controller Network, MELSECNET/H, or CC-Link	Available	Not available

*1: The module must be used with being mounted on the AnS/A series-compatible extension base unit (QA1S5□B, QA1S6□B, QA1S6ADP+A1S5□B/A1S6□B, QA6□B, or QA6ADP+A5□B/A6□B). (The High Performance model QCPU only)

3.8.1 Refresh mode

(1) Definition

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

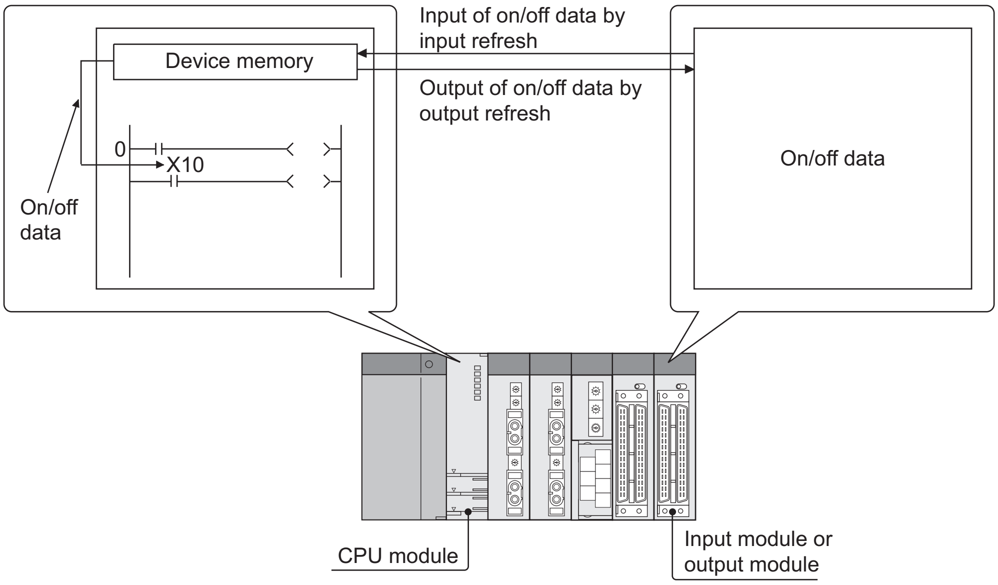

Input of on/off data by input refresh
Device memory
Output of on/off data by output refresh
0
X10
On/off data
On/off data
CPU module
Input module or output module

Figure 3.2 Refresh mode

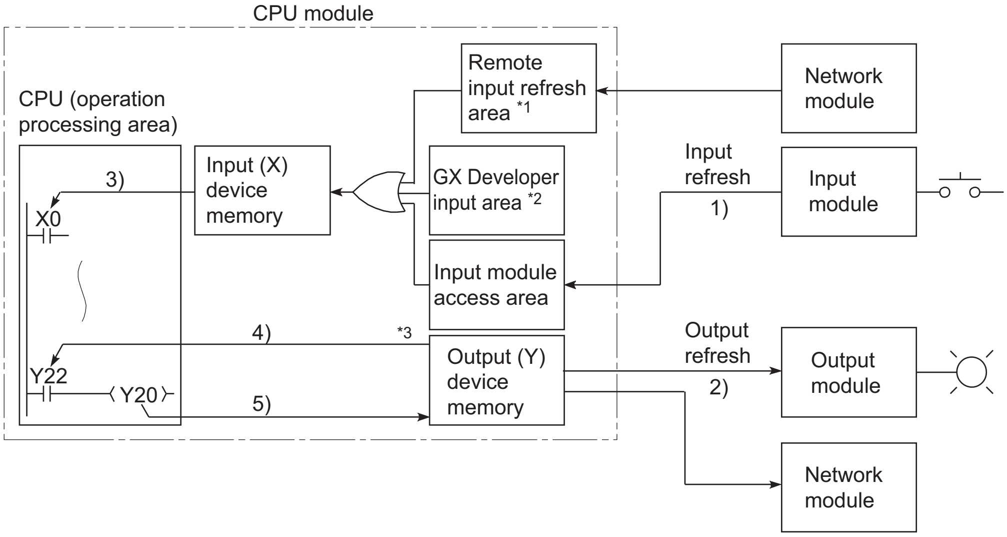
(2) Input

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.

(3) 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.



- **Input refresh:**
  Before the start of sequence program operations, the CPU module batch-reads input data 1) from an input module, performs a logical OR operation with data in the GX Developer input area or data in the remote input refresh area, and then stores the result in the input (X) device memory.
- **Output refresh:**
  Before the start of sequence program operations, the CPU module batch-outputs data in the output (Y) device memory 2) to the output module.
- **When a contact instruction for input is executed:**
  The CPU module reads input data 3) from the input (X) device memory and executes a sequence program.
- **When a contact instruction for output is executed:**
  The CPU module reads output data 4) from the output (Y) device memory and executes a sequence program.
- **When the OUT instruction for output is executed:**
  The CPU module stores the operation result of the sequence program 5) in the output (Y) device memory.

Figure 3.3 I/O data flow in refresh mode

*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 Controller Network, MELSECNET/H, or CC-Link.
Data in the remote input refresh area will be refreshed automatically during END processing.

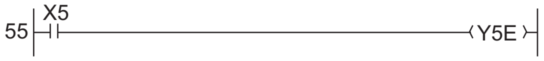
*2: Data in the GX Developer input area can be turned on/off by the following operation.
- Test operation by GX Developer
- Writing data from a network module
- Writing data from an external device using the MC protocol

*3: Data in the output (Y) device memory can be turned on/off by the following operation.
- Test operation by GX Developer
- Writing data from an external device using the MC protocol
- Writing data from a network module

(4) 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.

Examples



A program that turns on the output Y5E when the input X5 turns on.

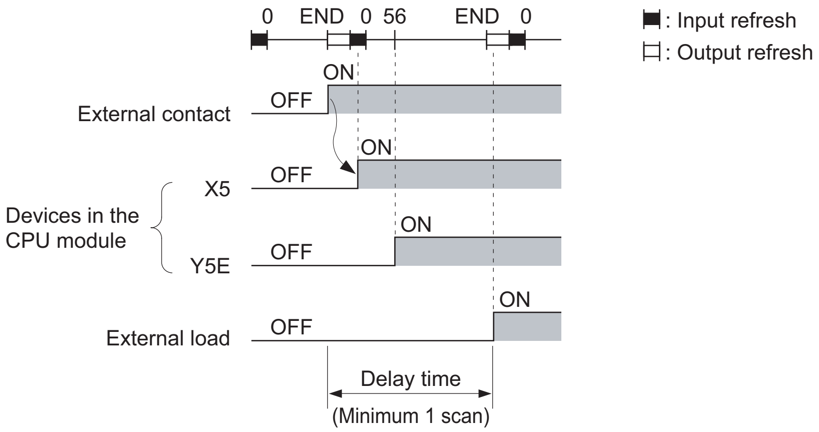


Figure 3.4 Y5E turns on the earliest

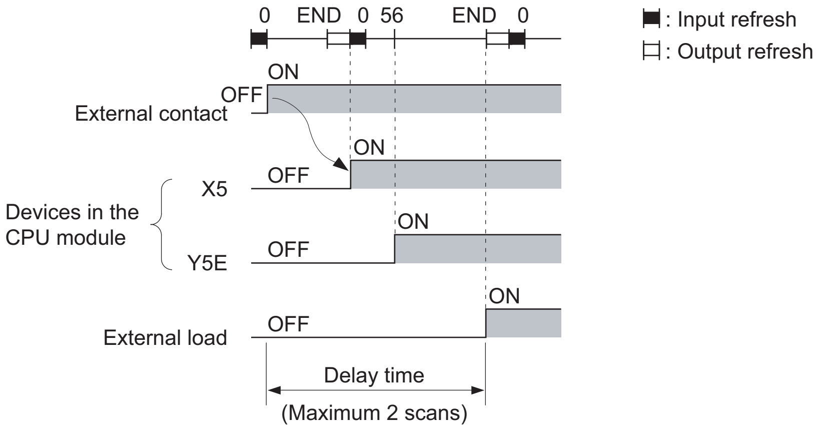
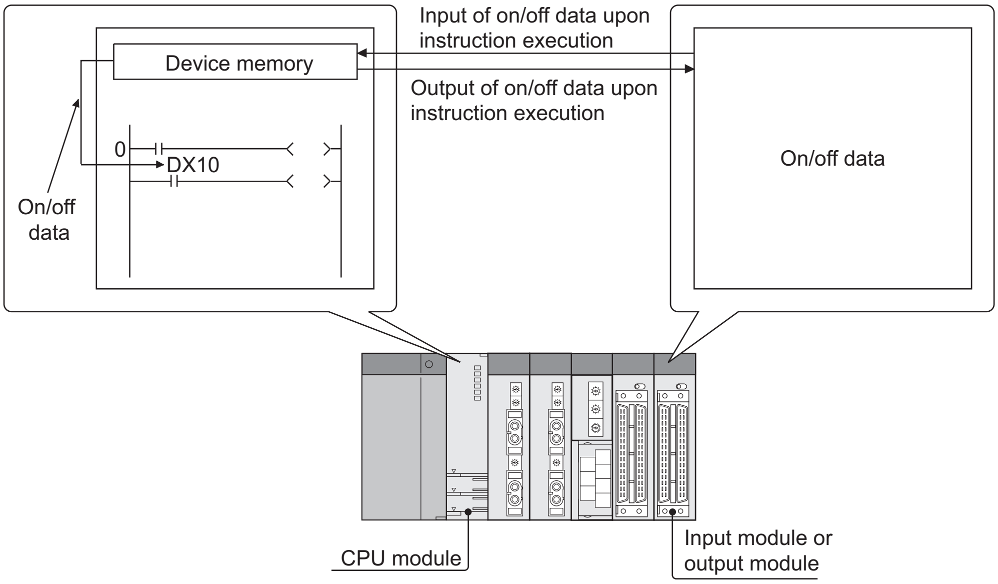


Figure 3.5 Y5E turns on the latest

3.8.2 Direct mode

(1) Definition

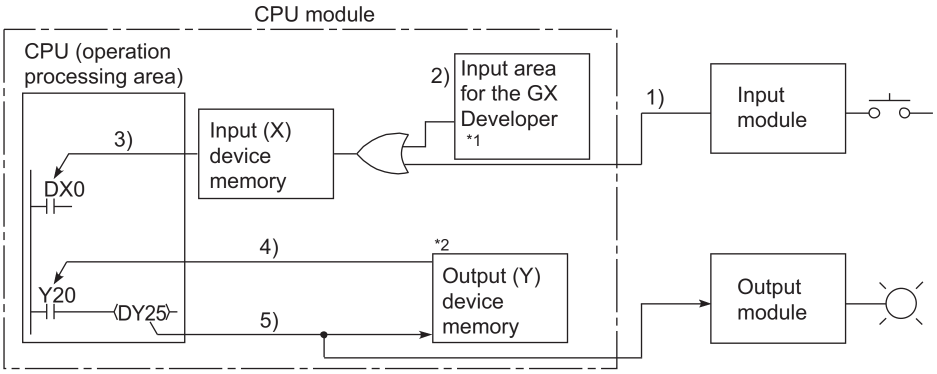
The direct mode is a mode for the CPU module to access input/output modules and performs I/O processing at the timing when each instruction is executed in a sequence program.



Input of on/off data upon instruction execution
Device memory
Output of on/off data upon instruction execution
0
DX10
On/off data
On/off data
CPU module
Input module or output module

Figure 3.6 Direct mode

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



- **When a contact instruction for input is executed:**
  The CPU module performs a logical OR operation between input data from the input module 1) and input data in the GX Developer input area 2) or data in the remote input refresh area.
  Then, the module stores the result in the input (X) device memory and executes a sequence program using the stored result as input data 3).
- **When a contact instruction for output is executed:**
  The CPU module reads output data 4) from the output (Y) device memory and executes a sequence program.
- **When the OUT instruction for output is executed:**
  The CPU module outputs the operation result of the sequence program 5) to the output module and also stores the result in the output (Y) device memory.

Figure 3.7 I/O data flow in direct mode

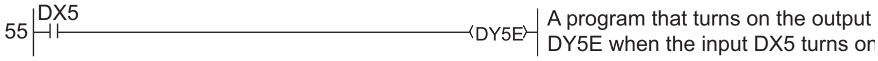
*1: Data in the GX Developer input area can be turned on/off by the following operation.
- Test operation by GX Developer
- Writing data from a network module
- Writing data from an external device using the MC protocol

*2: Data in the output (Y) device memory can be turned on/off by the following operation.
- Test operation by GX Developer
- Writing data from an external device using the MC protocol
- Writing data from a network module

(2) Response delay

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

Examples


55
DX5
DY5E
A program that turns on the output
DY5E when the input DX5 turns or

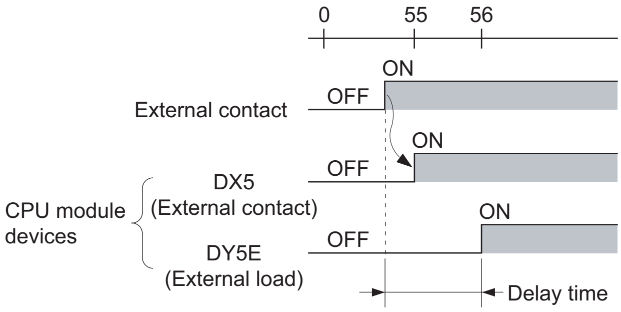


Figure 3.8 DY5E turns on the earliest

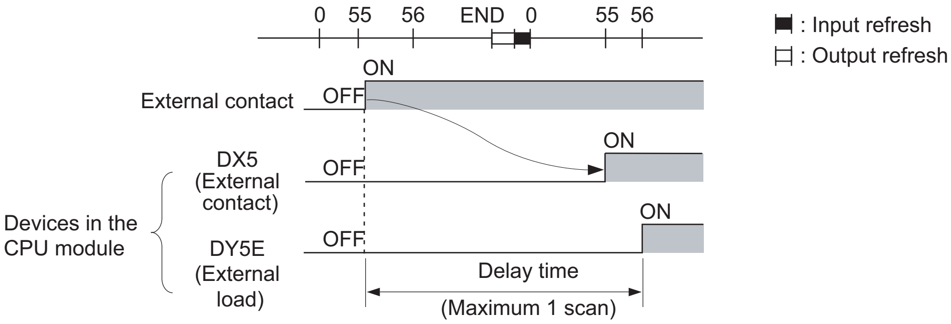


Figure 3.9 DY5E turns on the latest

CHAPTER4 ASSIGNMENT OF BASE UNIT AND I/O NUMBER

This chapter describes the base unit and I/O number assignment required for the CPU module to communicate data with I/O modules and/or intelligent function modules.

4.1 Base Unit Assignment

4.1.1 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.
For example, 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.

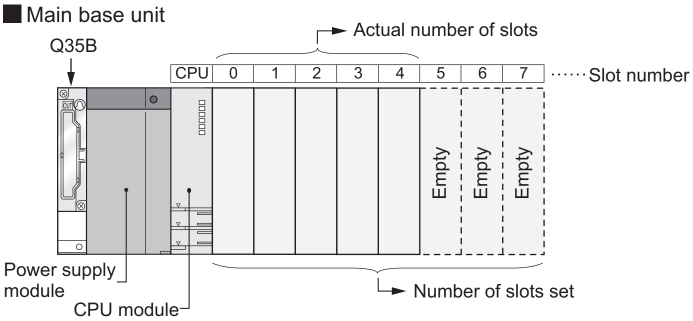


Figure 4.1 Setting the number of slots greater than the actual one

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.

For example, 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.".)

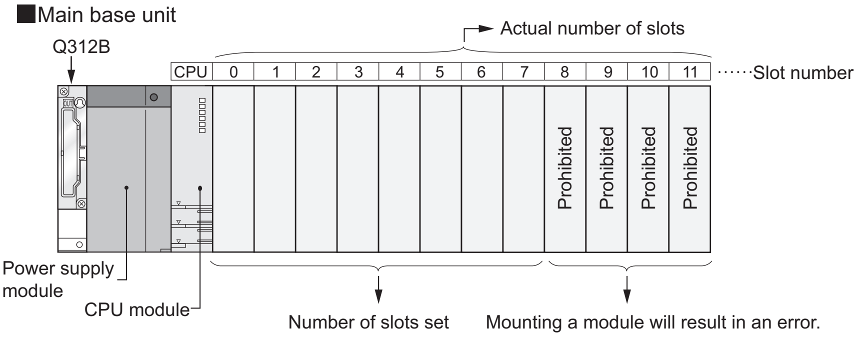
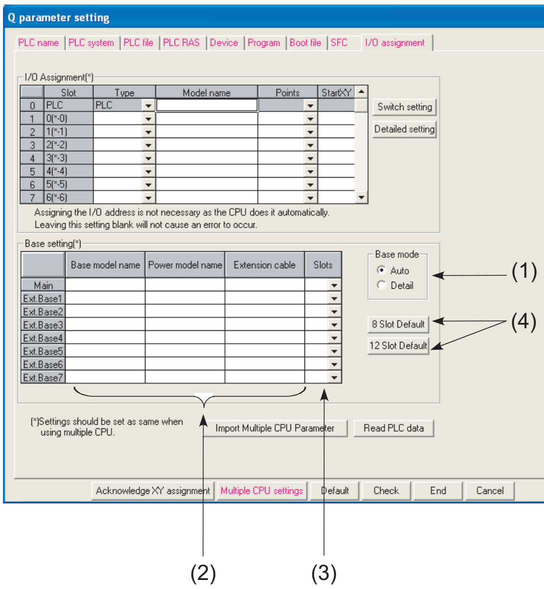


Figure 4.2 Setting the number of slots smaller than the actual one

4.1.2 Base unit assignment setting

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



I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode
● Auto
○ Detail

8 Slot Default
12 Slot Default

(*)Settings should be set as same when using multiple CPU.

Figure 4.3 I/O assignment setting

(1) Auto/Detail

Select the mode for the base unit assignment either from auto mode or detail mode.

(2) Base model name, Power model name, Extension cable

Enter the model names of mounted base units, power supply modules, and extension cables to be used within 16 characters for user reference or when printing out parameters.

CPU modules do not use the entered model names.

(3) Slots

When "Detail" is set, select the number of slots on the base unit to use from the following.

2 (2 slots), 3 (3 slots), 5 (5 slots), 8 (8 slots), 10 (10 slots), or 12 (12 slots)

(4) 8 Slot Default/12 Slot Default

When "Detail" is set, select either of these items for batch-setting the base units to the specified number of slots.

Point

- In auto mode, when any extension base number is skipped at the setting using the base number setting connector, an empty extension base cannot be reserved.
  To reserve empty extension bases 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.

4.2 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

I/O number representation

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 Figure 4.4.

"X" and "Y" is prefixed to the I/O number of input modules and the I/O number of output modules, respectively.

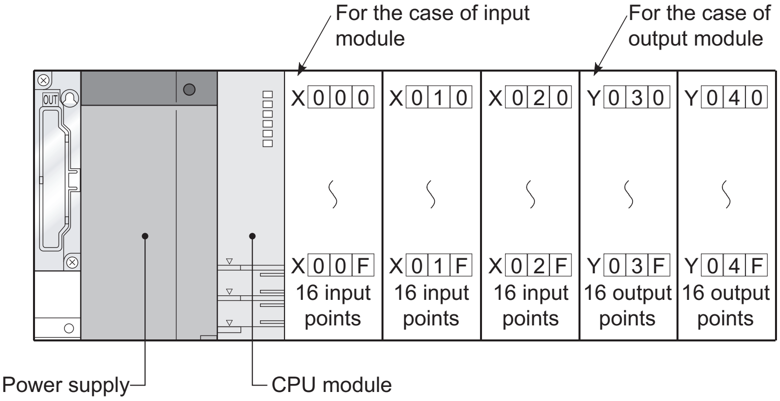


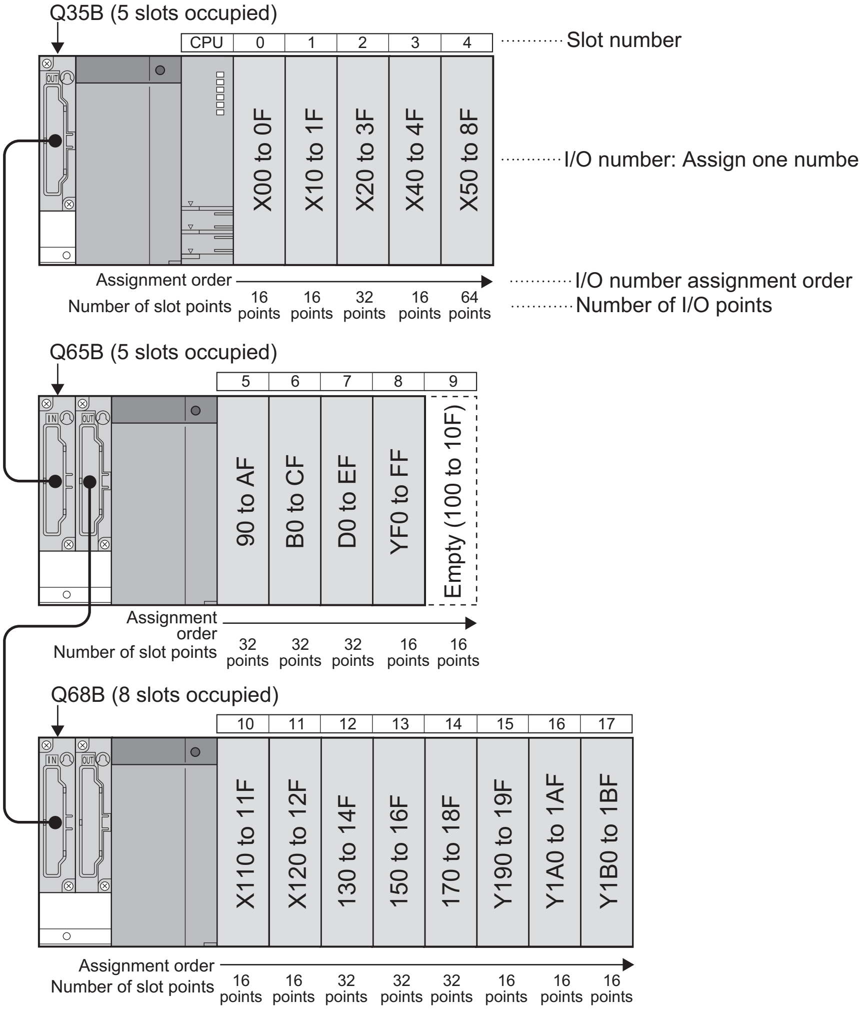
Figure 4.4 I/O numbers

4.2.1 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 Figure 4.5 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.



X□□0 to□□ F: Input module
Y□□0 to□□ F: Output module
□□0 to□□ F: Intelligent function module

Figure 4.5 I/O number assignment example

(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 0H 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 Note4.1

CPU module device input (X) and output (Y) 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 MELSECNET/H remote I/O network and CC-Link.

Also, inputs (X) and outputs (Y) can be used for the refresh target (devices on the CPU module side) of the MELSECNET/H module link I/O (LX and LY).

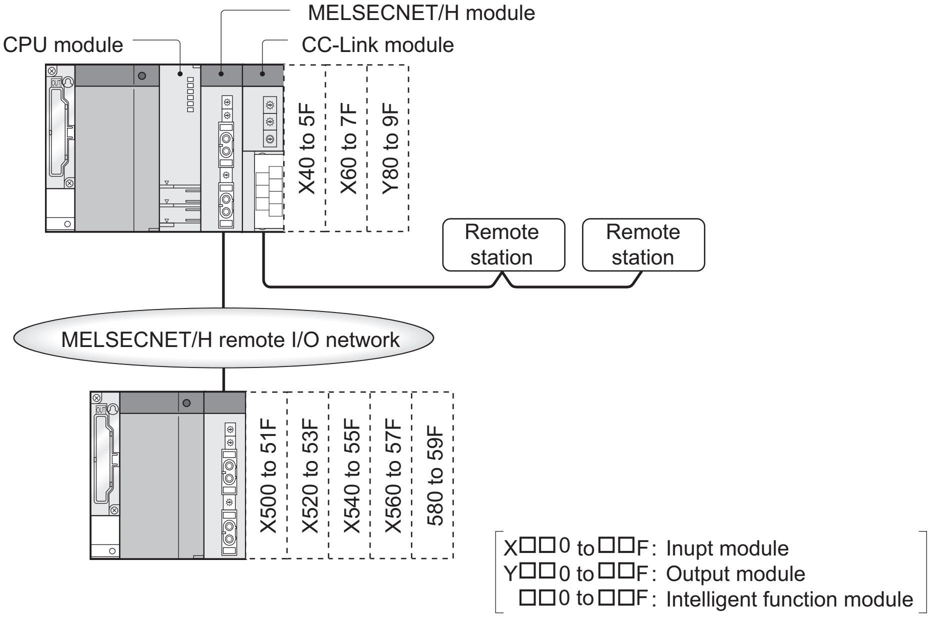


Figure 4.6 I/O number assignment on the remote station

(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.

Example 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.

Note4.1 Basic High performance

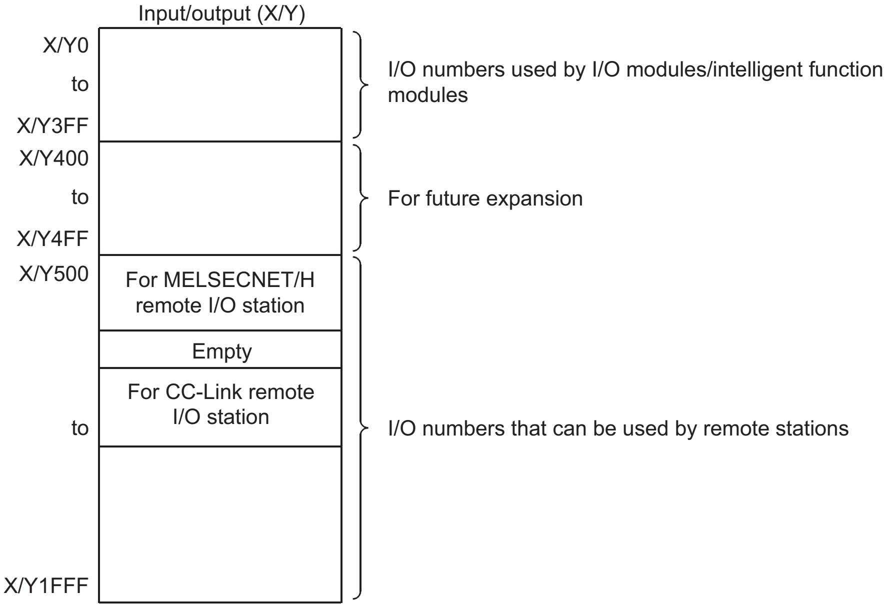
The basic model QCPU does not support the MELSECNET/H remote I/O network.

When using the MELSECNET/H remote I/O network for the High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.2)

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

1) 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.


Input/output (X/Y)
X/Y0
to
X/Y3FF
X/Y400
to
X/Y4FF
X/Y500
to
X/Y1FFF
For MELSECNET/H remote I/O station
Empty
For CC-Link remote I/O station
I/O numbers used by I/O modules/intelligent function modules
For future expansion
I/O numbers that can be used by remote stations

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

Figure 4.7 Remote station I/O number assignment

2) When MELSECNET/H and CC-Link are used

I/O numbers to the refresh target (CPU module side device) of MELSECNET/H and to the CC-Link remote I/O system must be unique.

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 MELSECNET/H remote I/O networks, CC-Link, or other networks.
- Space can be provided between the I/O area for MELSECNET/H remote I/O station and the I/O area for CC-Link remote station.

4.2.2 Setting I/O numbers

Set the I/O number 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.

For example, 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.
  - Input response time (I/O response time) (☞ Section 6.7)
  - Error time output mode (☞ Section 6.8)
  - CPU module operation during a hardware error of intelligent function modules (☞ Section 6.9)
  - Switch setting of intelligent function modules and interrupt modules (☞ Section 6.10)

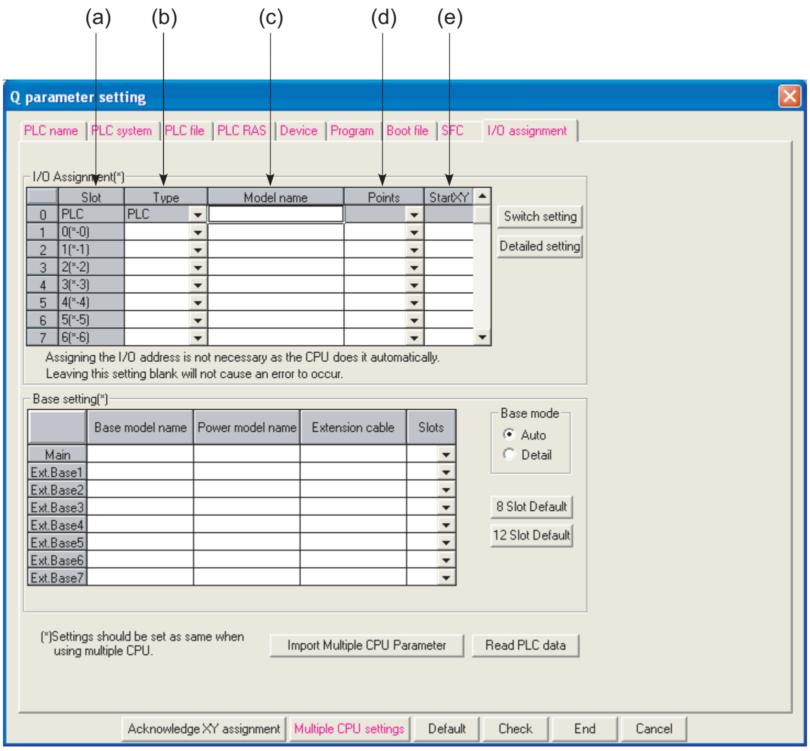
  The I/O assignment is required for the input response time and switch settings.

(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)

For example, 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.



I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

(*)Settings should be set as same when using multiple CPU.

Figure 4.8 I/O assignment

(a) Slot

The slot number and location of the slot are displayed.
When the base unit is set in Auto mode, the base unit number is indicated in "*", and the slot number is counted from slot 0 of the main base unit.

(b) 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)
- Intelli. (intelligent function module)
- Interrupt (interrupt module)

If the type is not specified, the type of the actually mounted module is used.

(c) Type

Enter the model names of mounted modules within 16 characters.
CPU modules do not use entered model names. (Use the entered model names for user reference.)

(d) Points

When changing the number of I/O points for each slot, the selections are as follows.
0 points, 16 points, 32 points, 48 points, 64 points, 128 points, 256 points, 512 points, 1024 points

If the number of points is not selected, the points of the actually mounted module is used.
For empty slots, the points that are set on the PLC system tab of the PLC parameter dialog box is assigned. (Default: 16 points)

(e) Start XY

When changing the I/O number for each slot, enter a new start I/O number.
If start XY is not specified for a slot, the I/O number continuing from the last number of the current setting is assigned.

(3) Precautions

(a) Type setting

The type set to the I/O assignment tab must be the same as that of the mounted module.
Setting a different type may cause incorrect operation.
For the intelligent function module, the I/O points must also be the same in addition to the I/O assignment setting.

Table4.1 shows the operations when the mounted module type differs from the one in the I/O assignment tab.

Table4.1 Incorrect operation when the module type differs

Mounted module	I/O assignment setting	Result
Input module, output module, I/O combined module	• Intelli. • Interrupt	Error (SP.UNIT.LAY.ERR.)
Intelligent function module	• Input • Hi. input • Output • I/O combined	Error (SP.UNIT.LAY.ERR.)
Empty slot	• Input • Hi. input • Output • I/O combined • Intelli. • Interrupt	Empty slot
All modules	• Empty	Empty slot
Other combinations	–	No error but the system will not operate normally.

(b) I/O points of slots

The number of I/O points for each slot selected in the I/O assignment tab is set in priority to those of mounted modules.

1) When the preset number of I/O points is less than those of mounted I/O modules

The available points for the mounted I/O module will be decreased.
For example, when the number of I/O points is set to 16 points in the I/O assignment setting of PLC parameter to the slot where a 32-point input module is mounted, the second half 16 points of the 32-point input module becomes unavailable.

2) When the preset number of I/O points exceeds those of mounted I/O modules

The exceeded number of points will not be used in I/O modules.

3) Last I/O number

Set the last I/O number within the I/O point range.
Failure to do so causes an error ("SP. UNIT LAY ERR."). ("***" is displayed as an I/O address on the System monitor screen of GX Developer.)

4) When setting 0 points in empty slots

Setting "Empty" and "0" to type and points respectively even occupies one slot.
To set any slots unoccupied after the specific slot number, set the number of slots in detail mode.
(☞ Section 4.1.1)

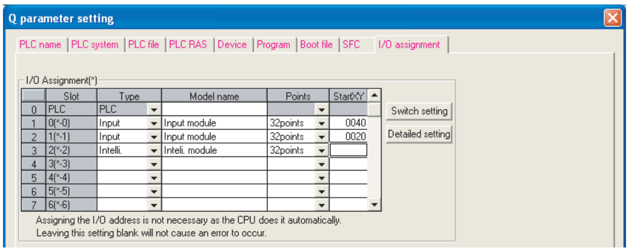
(c) Start XY setting

When the start XY has not been entered, the CPU module automatically assigns it. The CPU module automatically assigns the start XY if it is not set. 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

An example of start XY duplication is given in Figure 4.9 below.



Q parameter setting

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)	Input	Input module	32points	0040
2	1(*-1)	Input	Input module	32points	0020
3	2(*-2)	Intelli.	Inteli. module	32points	
4	3(*-3)				
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Figure 4.9 assignment setting with start XY duplication

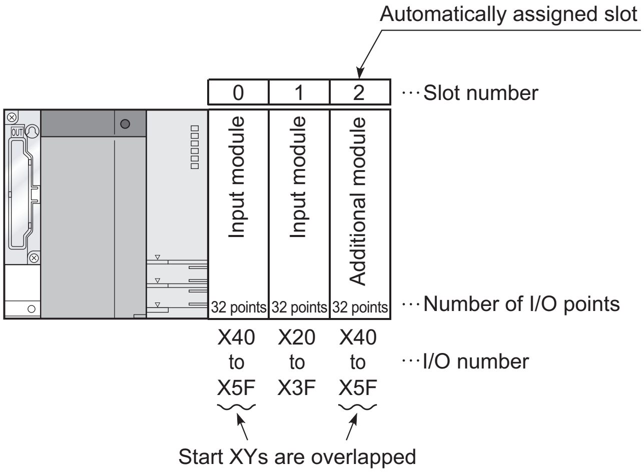


Figure 4.10 Start XY set by above (Figure 4.9) I/O assignment

Do not set duplicated start XY for each slot.
Specify start XY in the additional module to prevent the duplication of start XY.
(Example: Input "0060" to "start XY" in slot2.)

In the High Performance model QCPU and Process CPU, mounting the slot for duplication of start XY to modules will result in "SP. UNIT LAY ERR."
In the Basic model QCPU and Redundant CPU, duplication of start XY will result in "SP. UNIT LAY ERR." (An error occurs even the module is not mounted.)

(d) When using extension base units compatible with the AnS/A series Note4.2

Take the following precautions when using the AnS/A series-compatible extension base units (QA1S5□B, QA1S6□B, and QA6□B):

- They should be connected in the order of Q5□B/Q6□B→QA1S5□B/QA1S6□B→QA6□B where the Q5□B/Q6□B is the first extension base unit connected 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.
- The I/O numbers of modules mounted on the base unit should be assigned by separating them into the Q and A series.
  Failure to follow this precaution 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

Note4.2 Basic Process Redundant

The Basic model QCPU, Process CPU, and Redundant CPU do not support the AnS/A series extension base units.

4.2.3 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 using GX Developer

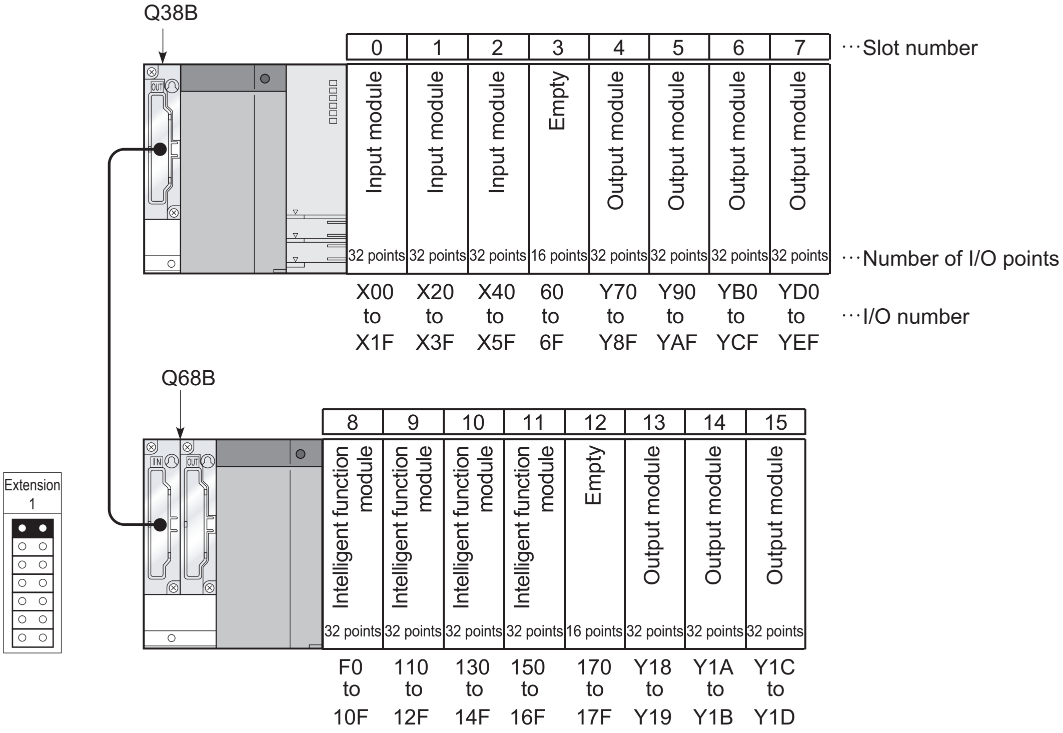
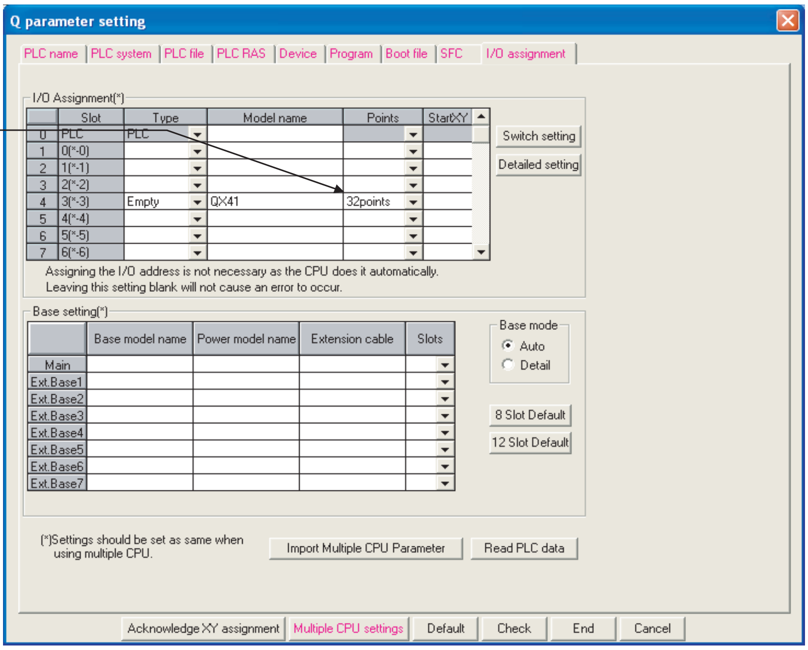


Figure 4.11 I/O number assignment (Before changing points of the empty slot)

(b) I/O assignment

Select "32 points" for the number of I/O points of Slot 3 in the I/O assignment setting of PLC parameter in GX Developer.

Select 32 points. (When the type is not selected, the type of the mounted module will be set.)



Q parameter setting

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)	Empty	QX41	32points	
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

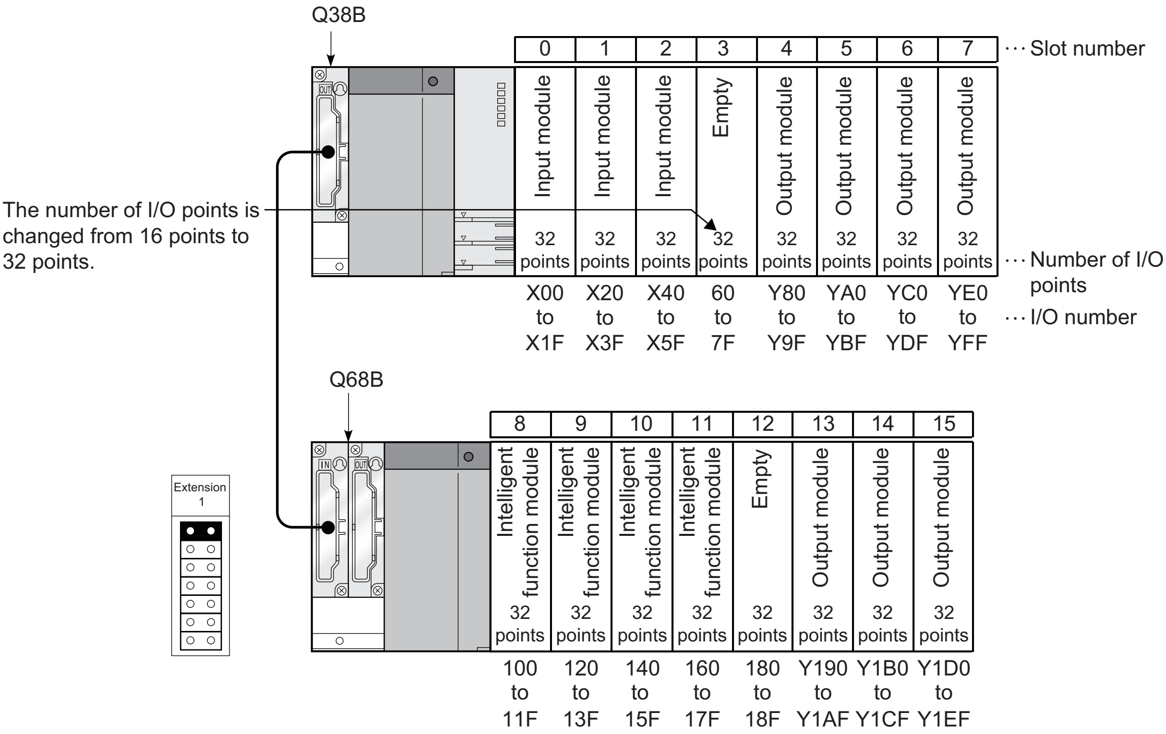
	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode
● Auto
○ Detail

(*)Settings should be set as same when using multiple CPU.

Figure 4.12 I/O assignment setting (When changing points of Slot 3)

(c) I/O number assignment after the I/O assignment using GX Developer



Slot number	0	1	2	3	4	5	6	7
Module	Input module	Input module	Input module	Empty	Output module	Output module	Output module	Output module
Number of I/O points	32 points	32 points	32 points	32 points	32 points	32 points	32 points	32 points
I/O number	X00 to X1F	X20 to X3F	X40 to X5F	60 to 7F	Y80 to Y9F	YA0 to YBF	YC0 to YDF	YE0 to YFF

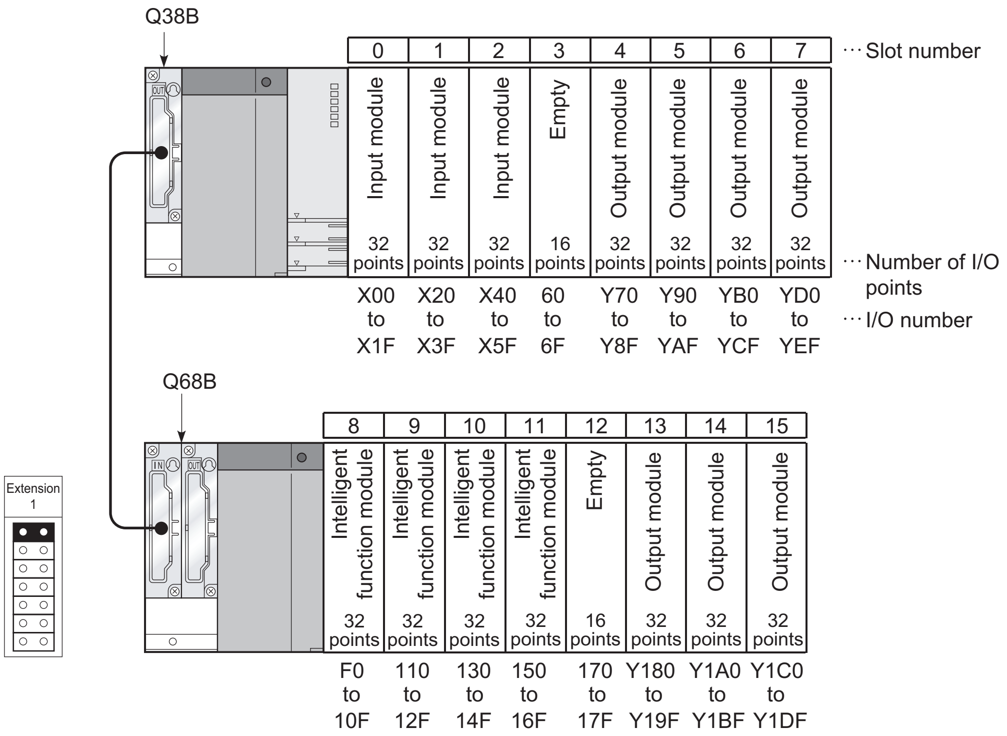
Slot number	8	9	10	11	12	13	14	15
Module	Intelligent function module	Intelligent function module	Intelligent function module	Intelligent function module	Empty	Output module	Output module	Output module
Number of I/O points	32 points	32 points	32 points	32 points	32 points	32 points	32 points	32 points
I/O number	100 to 11F	120 to 13F	140 to 15F	160 to 17F	180 to 18F	Y190 to Y1AF	Y1B0 to Y1CF	Y1D0 to Y1EF

Figure 4.13 I/O numbers after I/O assignment (After changing points of the empty slot)

(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 the I/O assignment using GX Developer



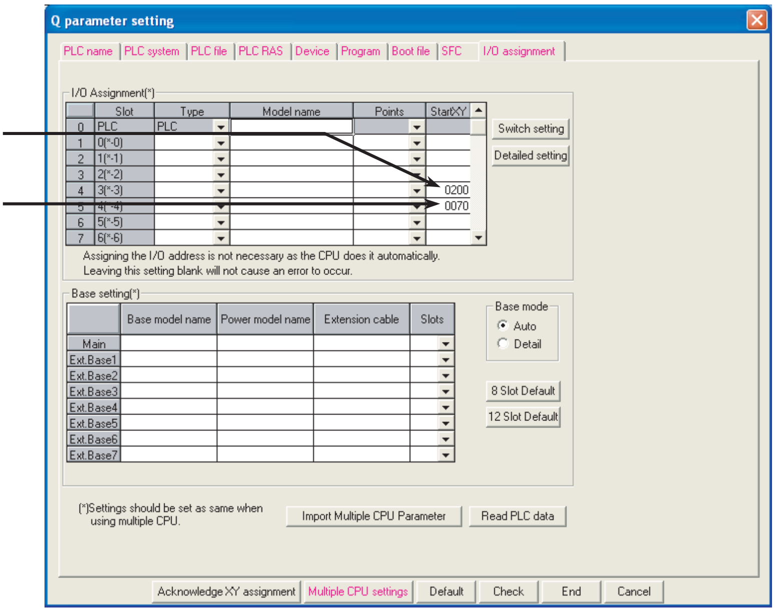
Slot number	0	1	2	3	4	5	6	7
	Input module	Input module	Input module	Empty	Output module	Output module	Output module	Output module
Number of I/O points	32 points	32 points	32 points	16 points	32 points	32 points	32 points	32 points
I/O number	X00 to X1F	X20 to X3F	X40 to X5F	60 to 6F	Y70 to Y8F	Y90 to YAF	YB0 to YCF	YD0 to YEF

Slot number	8	9	10	11	12	13	14	15
	Intelligent function module	Intelligent function module	Intelligent function module	Intelligent function module	Empty	Output module	Output module	Output module
Number of I/O points	32 points	32 points	32 points	32 points	16 points	32 points	32 points	32 points
I/O number	F0 to 10F	110 to 12F	130 to 14F	150 to 16F	170 to 17F	Y180 to Y19F	Y1A0 to Y1BF	Y1C0 to Y1DF

Figure 4.14 I/O number assignment (Before changing slot I/O numbers)

(b) I/O assignment

Set "200" for the start XY of Slot 3 and "70" to Slot 4 in the I/O assignment setting of PLC parameter in GX Developer.



Set "200" to start XY.

Set "70" to start XY.
(If not set, the I/O
number following the
slot 3 will be set.)

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				0200
5	4(*-4)				0070
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode
● Auto
○ Detail

(*)Settings should be set as same when
using multiple CPU.

Figure 4.15 I/O assignment setting (When changing I/O numbers of Slot 3)

(c) I/O number assignment after the I/O assignment using GX Developer

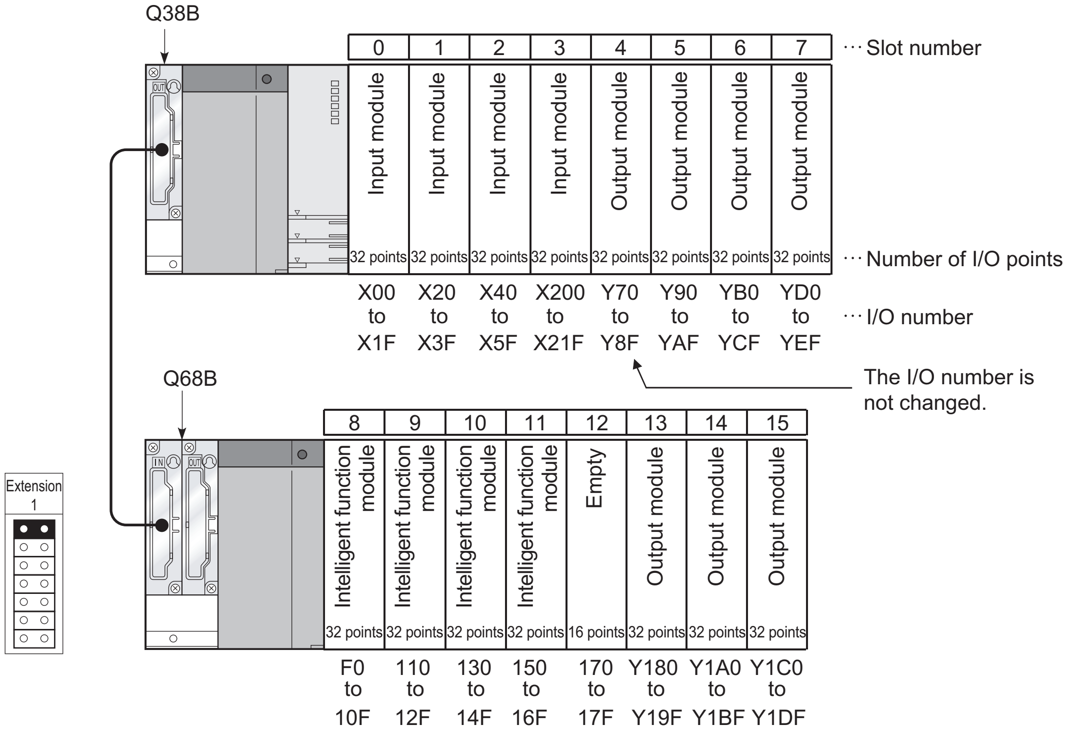


Figure 4.16 I/O number assignment (After changing slot I/O numbers)

4.2.4 Checking I/O numbers

Information of mounted modules and their I/O numbers can be checked on the System monitor screen of GX Developer. (☞ Section 6.20)

CHAPTER5 MEMORIES AND FILES USED FOR CPU MODULE

5.1 Memories Used for Basic Model QCPU

5.1.1 Memory composition and storable data

This section describes the memories used for the Basic model QCPU and data that can be stored in the memories.

(1) Memory composition

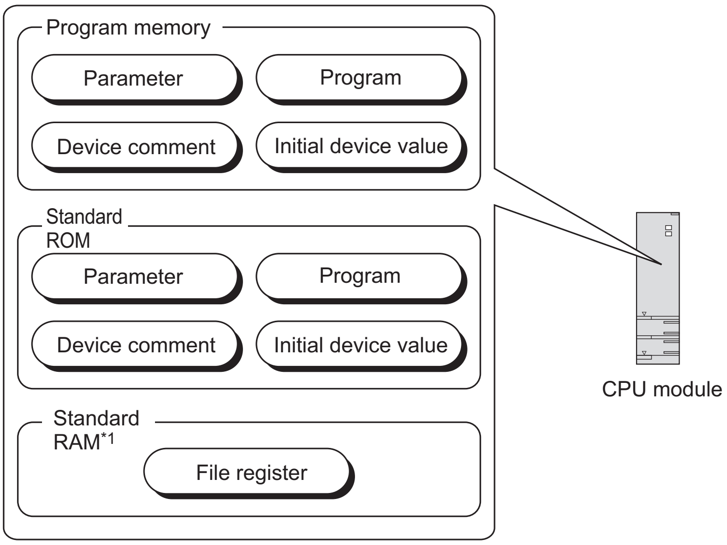


Figure 5.1 Memory composition of the Basic model QCPU

*1: The Q00JCPU does not have the standard RAM.

(a) Program memory (☞ Section 5.1.2)

This memory is for storing programs and parameters for CPU module operation.

(b) Standard ROM (☞ Section 5.1.3)

This memory is for storing data such as programs and parameters.
Programs and parameters can be stored without battery backup.

(c) Standard RAM (☞ Section 5.1.4)

This memory is for using file registers without a memory card.

(2) Data that can be stored in each memory

Table5.1 provides the data that can be stored in each memory.

Table5.1 Data that can be stored in each memory

Item	CPU module built-in memory			File name and extension
	Program memory	Standard RAM	Standard ROM	
	Drive 0 *1	Drive 3 *1	Drive 4 *1	
Parameter	◎	×	○	PARAM.QPA
Intelligent function module parameter*2	○	×	○	IPARAM.QPA
Program	◎ *2	×	○ *3	MAIN.QPG
SFC Program	◎ *2	×	○ *3	MAIN-SFC.QPG
File register	×	○ *4	×	MAIN.QDR
Device comment	○ *5	×	○ *5	MAIN.QCD
Initial device value	○	×	○	MAIN.QDI
Label program *11	○	×	○	PROJINFO.CAB
User setting system area *6	○	×	×	-

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

*1: A drive number is used to specify a memory to be written/read by the external device using a sequence program or MC protocol. Since the memory name is used to specify the target memory by GX Developer, the drive number needs not to be considered.

*2: Any of sequence program, ST program and SFC program data is necessary.

*3: To execute a program stored in the standard ROM, make the setting in the Boot file tab of the PLC parameter dialog box.

*4: Only one file register file can be stored in the standard RAM. (☞ Section 9.7)

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

*6: Set the area used by the system. (☞ Section 5.1.2(2)(b))

*7: Data that stores the label program configuration.
For the label program, refer to the following.
☞ GX Developer Version8 Operating Manual

(3) Memory capacities and necessity of formatting

Table5.2 provides the memory capacities and necessity of formatting of each memory.
Format a memory requiring formatting by GX Developer beforehand.

Table5.2 Memory capacities and necessity of formatting

	Q00JCPU	Q00CPU	Q01CPU	Formatting
Program memory	58k bytes	94k bytes	94k bytes	Necessary
Standard ROM	58k bytes	94k bytes	94k bytes	Unnecessary
Standard RAM	Unavailable	128k bytes*1		Necessary*2

*1: The standard RAM capacities of the Basic model QCPUs vary depending on their versions. (☞ Appendix 2.1)

*2: Formatting is unnecessary for the CPU module of function version B or later.
(Formatting the standard RAM from GX Developer to the CPU module of function version B or later displays the error code: 4150H on the GX Developer screen.)

Point

When data are written to each memory, the unit of stored file size depends on the target CPU module and memory area.
(☞ Section 5.4.4)

5.1.2 Program memory

(1) Definition

This memory is for storing programs and parameters for CPU module operation.

Point

If the total size of data to be stored exceeds the program memory capacity:

- reduce the user setting system area, or
- transfer data other than programs to the standard ROM.

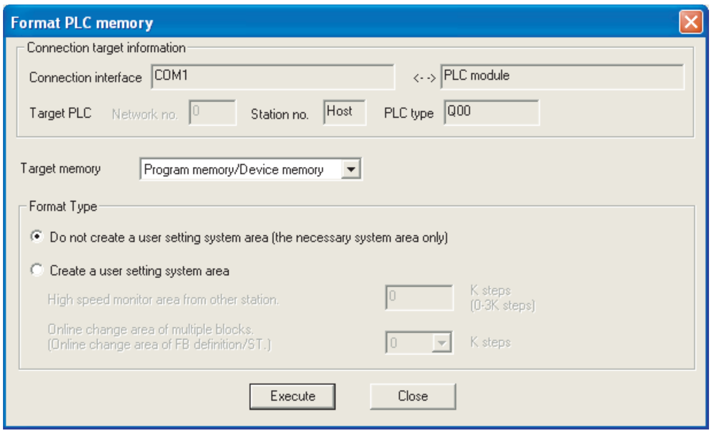
(2) Before using the program memory

Format the program memory by GX Developer.

(a) Formatting

Select [Online] → [Format PLC memory] in GX Developer.

Select "Program memory/Device memory" in "Target memory"



Format PLC memory

Connection target information

Connection interface: COM1 <--> PLC module

Target PLC: Network no. 0; Station no. Host; PLC type Q00

Target memory: Program memory/Device memory

Format Type

● Do not create a user setting system area (the necessary system area only)

○ Create a user setting system area

High speed monitor area from other station. 0 K steps (0-3K steps)

Online change area of multiple blocks. (Online change area of FB definition/ST.) 0 K steps

Figure 5.2 Formatting the program memory

(b) Creating a user setting system area

When formatting a program memory, set the capacity of user setting system area.

1) Do not create a user setting system area (the necessary system area only)

The user setting system area is not created during formatting.

2) Create a user setting system area

The user setting system area is created during formatting.

Table5.3 provides the type of user setting system area.

Table5.3 Type of user setting system area

System area	Description
High speed monitor area from other station.	Setting this area increases the monitoring speed in the GX Developer connected to such as a serial communication module. When using RS-232 and USB together with GX Developer, this area is used to register monitor data from GX Developer connected to such as a serial communication module.
Online change area of multiple blocks (Online change area of FB definition/ST.)	Setting this area allows writing of multiple blocks to the CPU module in the RUN status. For the number of blocks that can be written to the CPU module in the RUN status, refer to the following. ☞ GX Developer Version 8 Operating Manual

Point

When a user setting system area is created, the available area reduces by the number of steps created in the area.

(c) Checking the memory capacity after formatting

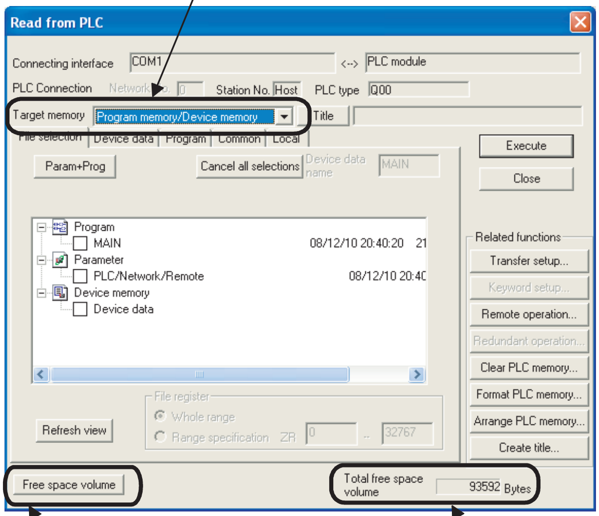
Select [Online] → [Read from PLC] in GX Developer.

1) Select "Program memory/Device memory" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".

1) Select the target memory.



Read from PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q00

Target memory: Program memory/Device memory

Device data name: MAIN

- Program
  - □ MAIN 08/12/10 20:40:20 21
- Parameter
  - □ PLC/Network/Remote 08/12/10 20:4C
- Device memory
  - □ Device data

File register: ● Whole range ○ Range specification ZR 0 -- 32767

Total free space volume 93592 Bytes

2) Click the Free space volume button.

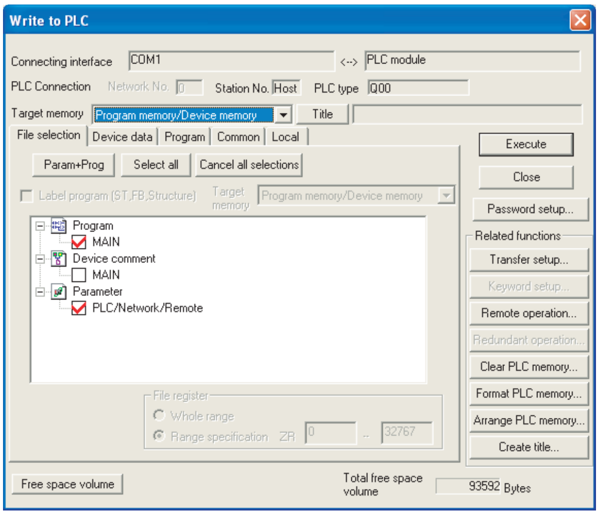
3) The memory capacity value is shown.

Figure 5.3 Procedure for checking the memory capacity

(3) Writing to the program memory

Select [Online] → [Write to PLC] in GX Developer.

Select "Program memory/Device memory" in "Target memory" on the Write to PLC screen.



Write to PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q00

Target memory: Program memory/Device memory

□ Label program (ST,FB,Structure) — Target memory: Program memory/Device memory

- Program
  - ☑ MAIN
- Device comment
  - □ MAIN
- Parameter
  - ☑ PLC/Network/Remote

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume: 93592 Bytes

Figure 5.4 Write to PLC screen

Point

The file size has its minimum unit. (☞ Section 5.4.4)

The occupied memory capacity may be greater than the actual file size.

Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

5.1.3 Standard ROM

(1) Definition

This memory is for storing data such as parameters and programs.

Programs and parameters can be stored without battery backup.

(2) Checking the memory capacity

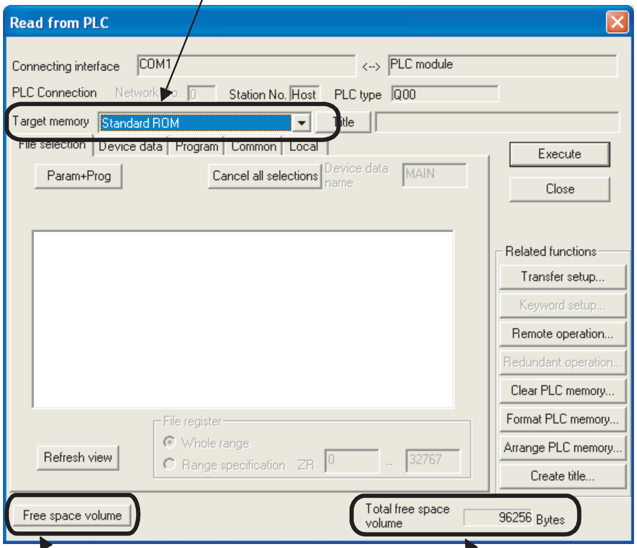
Select [Online] → [Read from PLC] in GX Developer.

1) Select "Standard ROM" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".

1) Select the target memory.



Read from PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type: Q00

Target memory: Standard ROM

Total free space volume: 96256 Bytes

2) Click the Free space volume button.

3) The memory capacity value is shown.

Figure 5.5 Procedure for checking the memory capacity

(3) Writing to the standard ROM

To write data, select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] in GX Developer. (☞ Section 5.1.5)

Point

The file size has its minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.
Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

5.1.4 Standard RAM Note5.1

(1) Definition

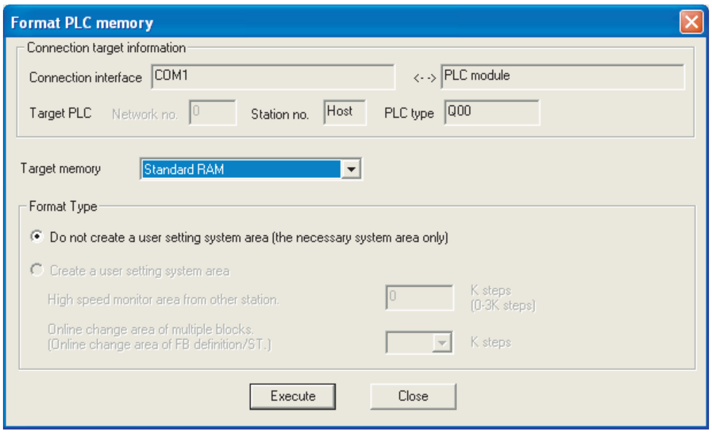
This memory is for file registers.

(2) Before using the standard RAM

Format the standard RAM by GX Developer for the Basic model QCPU of function version A.
For the Basic model QCPU of function version B or later, formatting the standard RAM is not required but clearing standard RAM data is required. For how to clear the data (file register), refer to Section 9.7(3).

(a) Formatting

Select [Online] → [Format PLC memory] in GX Developer.
Select "Standard RAM" in "Target memory".



Connection target information

Connection interface: COM1 <--> PLC module

Target PLC: Network no. 0 Station no. Host PLC type Q00

Target memory: Standard RAM

Format Type

● Do not create a user setting system area (the necessary system area only)

○ Create a user setting system area

High speed monitor area from other station: 0 K steps (0-3K steps)

Online change area of multiple blocks. (Online change area of FB definition/ST.): K steps

Figure 5.6 Formatting the standard RAM

Note5.1 **Basic**

The Q00JCPU does not have the standard RAM.

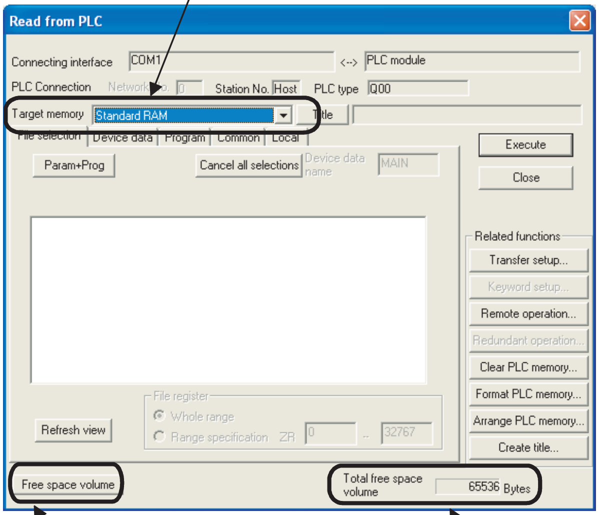
(b) Checking the memory capacity after formatting

Select [Online] → [Read from PLC] in GX Developer.

1) Select "Standard RAM" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".



Read from PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q00

Target memory: Standard RAM

File register: ● Whole range ○ Range specification ZR 0 -- 32767

Total free space volume: 65536 Bytes

1) Select the target memory.

2) Click the Free space volume button.

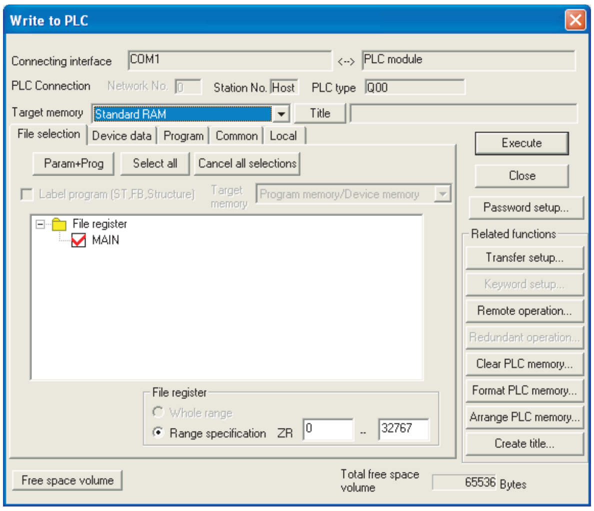
3) The memory capacity value is shown.

Figure 5.7 Procedure for checking the memory capacity

(3) Writing to the standard RAM

Select [Online] → [Write to PLC] in GX Developer.

Select "Standard RAM" in "Target memory" on the Write to PLC screen.



Write to PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q00

Target memory: Standard RAM

□ Label program (ST,FB,Structure) Target memory: Program memory/Device memory

File register
☑ MAIN

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume: 65536 Bytes

Figure 5.8 Write to PLC screen

Point

The file size has its minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.
Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

5.1.5 Operating and writing programs in the standard ROM (boot operation)

Since a program stored in the standard ROM cannot be operated, boot the program to the program memory.

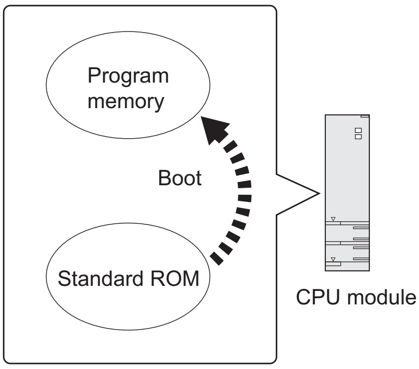


Figure 5.9 Boot operation

(1) Program execution (boot operation)

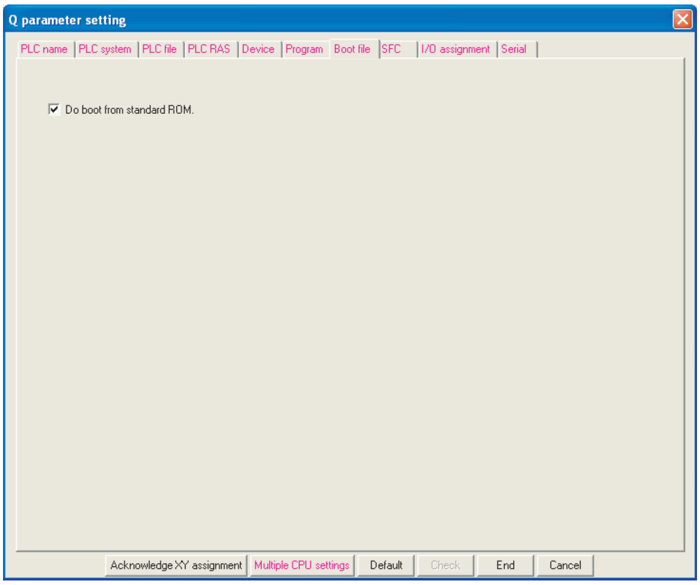
Boot a program by the following procedure.

(a) Creating a program by GX Developer

Create a program to be booted.

(b) Setting for boot operation

Select "Do boot from Standard ROM." in the Boot file tab of the PLC parameter dialog box.



☑ Do boot from standard ROM.

Figure 5.10 Boot file tab

(c) Writing to the standard ROM

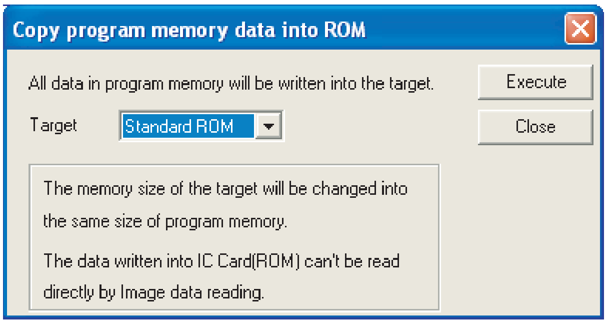
The following describes the operations before writing and the methods for writing.

1) Before writing

Writing a file to the standard ROM automatically deletes all files stored in the standard ROM.
Therefore, store the stored files again by “Read from PLC” before the writing.
(☞ Section 5.1.3)

2) Writing procedure

To write a file to the program memory, select [Online] → [Write to PLC] in GX Developer.
To write the file written to the program memory to the standard ROM, select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM].



Copy program memory data into ROM

All data in program memory will be written into the target.

Target: Standard ROM

The memory size of the target will be changed into the same size of program memory.

The data written into IC Card(ROM) can't be read directly by Image data reading.

Figure 5.11 Copy program memory data into ROM screen

(d) Program execution

Resetting the CPU module starts boot operation from the standard ROM.
Whether the boot operation is normally completed or not can be checked by the special relay (SM660).
For details of the special relay (SM660), refer to the following.

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

(2) Adding or changing a file in the standard ROM

Add or change the file by the following method (stored files cannot be added or changed directly).

1) To read all files in the standard ROM, select [Online] → [Read from PLC].
2) Add or change the read files.
3) Write the added or changed files to the program memory.
4) Select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] and write the files to the standard ROM.

(3) Operation for stopping boot operation

To stop boot operation and operate parameters and programs written to the program memory, perform the following operations.

1) Format the program memory.
2) Select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] in GX Developer (programs and parameters in the standard ROM are deleted).
3) Write programs and parameters to the program memory.

(4) Precautions

(a) Files stored to the standard ROM

Before boot operation, store the following files to the standard ROM.

- Parameter
- Intelligent function module parameter
- Program[*1]
- Device comment
- Initial device value

[*1] Any of the sequence program, ST program, and SFC program is required.

(b) Online change in boot operation

When a program in the program memory is written in the RUN status during boot operation from the standard ROM, the change cannot be updated to the program in the boot source standard ROM.
Set the CPU module to STOP and then write the program to the standard ROM.

(c) 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. Refer to (3) in this section and stop the boot operation.

(d) Setting the communication time check period in GX Developer

If less than 180 seconds is set to "Check at communication time" in the Host station Detailed setting screen when a file is written to the standard ROM, set it to 180 seconds.
If an error occurs, extend the communication time check period by the setting in the Transfer Setup screen.

(e) Writing a file from GX Developer in another station via CC-Link

Since writing a file to the standard ROM takes time, make the CPU monitoring time setting (SW0A) of CC-Link to 180 seconds or longer.

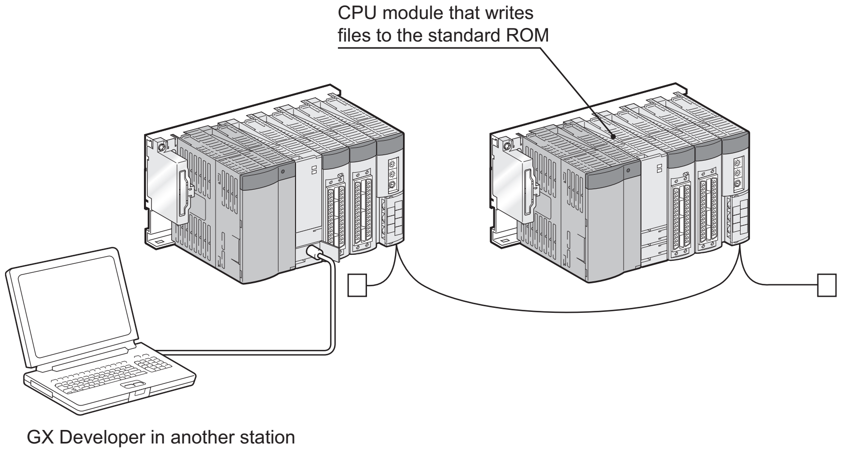


Figure 5.12 Writing a file from GX Developer in another station (via CC-Link)

Remark

For settings of the communication time check period and CPU monitoring time, refer to the following.

☞ GX Developer Version8 Operating Manual

5.2 Memories Used for High Performance model QCPU, Process CPU, and Redundant CPU

5.2.1 Memory composition and storable data

This section describes the memories used for the High Performance model QCPU, Process CPU, and Redundant CPU and data that can be stored in the memories.

(1) Memory composition

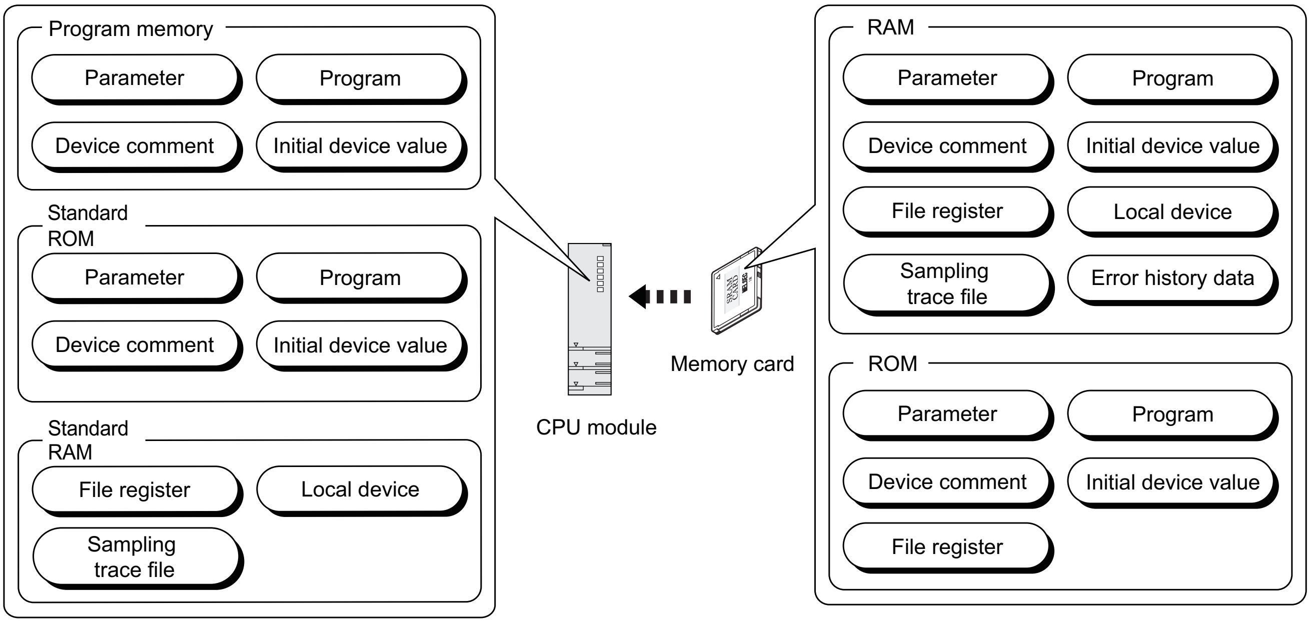


Figure 5.13 Memory composition of the High Performance model QCPU, Process CPU, and Redundant CPU

(a) Program memory (☞ Section 5.2.2)

This memory is for storing programs and parameters for CPU module operation.

(b) Standard ROM (☞ Section 5.2.3)

This memory is for storing data such as parameters and programs.
Programs and parameters can be stored without battery backup.

(c) Standard RAM (☞ Section 5.2.4)

This memory is for using file registers, local devices, and sampling trace files without a memory card.

(d) Memory card (☞ Section 5.2.5)

This memory is for expansion of built-in memory in the CPU module.
Three types of memory cards are available: SRAM card, Flash card, and ATA card.

(2) Data that can be stored in each memory

Table5.4 provides the data that can be stored in each memory.

Table5.4 Data that can be stored in each memory

Item	CPU module built-in memory			Memory card (RAM)	Memory card (ROM)		File name and extension	Remarks
	Program memory	Standard RAM	Standard ROM	SRAM card	Flash card	ATA card		
	Drive 0 *1	Drive 3 *1	Drive 4 *1	Drive 1 *1	Drive 2 *1			
Parameter	○	×	○	○	○	○	PARAM.QPA	1 data/drive
Intelligent function module parameter*2	○	×	○	○	○	○	IPARAM.QPA	1 data/drive
Program	◎	×	○*3	○*3	○*3	○*3	***.QPG	-
Device comment	○*4	×	○*5	○*5	○*5	○*5	***.QCD	-
Initial device value	○	×	○	○	○	○	***.QDI	-
Device data	×	×	×	×	×	×	***.QST	-
File register	×	○*6*7	×	○	○*8	×	***.QDR	-
Local device	×	○*6	×	○	×	×	***.QDL	1 data/CPU module
Sampling trace file	×	○*6	×	○	×	×	***.QTD	-
Boot setting file	○	×	○	○	○	○	AUTOEXEC.QBT	-
Remote password	○	×	○	○	○	○	00000000.QTM	-
Error history data	×	×	×	○	×	×	***.QFD	-
Programmable controller user data	×	×	×	×	×	○*9	***.CSV/BIN	-
Label program*11	○	×	○	×	×	○	PROJINFO.CAB	-
User setting system area *10	○	×	×	×	×	×	-	-

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

*1: A drive number is used to specify a memory to be written/read by the external device using a sequence program or MC protocol. Since the memory name is used to specify the target memory by GX Developer, the drive number needs not to be considered.
*2: Store the intelligent function module parameters in the same drive with the parameters. When they are stored in different drives, the intelligent function module parameters do not become valid.
*3: A program stored in the standard ROM cannot be executed. Store the program to the program memory before execution.
*4: The device comments cannot be read by instructions in a sequence program.
*5: Reading from a sequence program requires several scans.
*6: Only each one of file register, one local device, and/or sampling trace file can be stored in the standard RAM.
*7: For the number of storable file register points, refer to Section 9.7.
*8: A sequence program allows reading only. No data can be written from the sequence program.
*9: 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.)

*10: Set an area used by the system. (☞ Section 5.2.2(2)(b))
*11: Data that stores the label program configuration. For the label program, refer to the following.
☞ GX Developer Version8 Operating Manual

(3) Memory capacities and necessity of formatting

Table5.5 provides the memory capacities and necessity of formatting of each memory.
Format a memory requiring formatting by GX Developer beforehand.

Table5.5 Memory capacities and necessity of formatting

		Q02CPU*2	Q02HCPU*2	Q06HCPU*2	Q12HCPU*2	Q25HCPU*2	Formatting
Program memory		112k bytes (28k steps)	112k bytes (28k steps)	240k bytes (60k steps)	496k bytes (124k steps)	1008k bytes (252k steps)	Necessary*1
Standard ROM		112k bytes	112k bytes	240k bytes	496k bytes	1008k bytes	Unnecessary
Standard RAM		64k bytes	128k bytes		256k bytes		Necessary*1
Memory card	SRAM card	Q2MEM-1MBSN, Q2MEM-1MBS: 1M byte Q2MEM-2MBSN, Q2MEM-2MBS: 2M bytes Q3MEM-4MBS: 4M bytes					Necessary (Use GX Developer.)
	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 (Use GX Developer.)

		Q02PHCPU	Q06PHCPU	Q12PHCPU	Q25PHCPU	Q12PRHCPU	Q25PRHCPU	Formatting
Program memory		112k bytes (28k steps)	240k bytes (60k steps)	496k bytes (124k steps)	1008k bytes (252k steps)	496k bytes (124k steps)	1008k bytes (252k steps)	Necessary*1
Standard ROM		112k bytes	240 bytes	496k bytes	1008k bytes	496k bytes	1008k bytes	Unnecessary
Standard RAM		128k bytes		256k bytes				Necessary*1
Memory card	SRAM card	Q2MEM-1MBSN, Q2MEM-1MBS: 1M byte Q2MEM-2MBSN, Q2MEM-2MBS: 2M bytes Q3MEM-4MBS: 4M bytes						Necessary (Use GX Developer.)
	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 (Use GX Developer.)

*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: The standard RAM capacity of the High Performance model QCPU varies depending on its version. (☞ Appendix 2.2)

Point

- When files are written to each memory, the unit of stored file size depends on the target CPU module and memory area. (☞ Section 5.4.4)
- In memory capacity calculation, 1 step is equal to 4 bytes.

5.2.2 Program memory

(1) Definition

This memory is for storing programs and parameters for CPU module operation.

Point

If the total size of data to be stored exceeds the program memory capacity:

- reduce the user setting system area, or
- transfer data other than programs to the standard ROM or memory card.

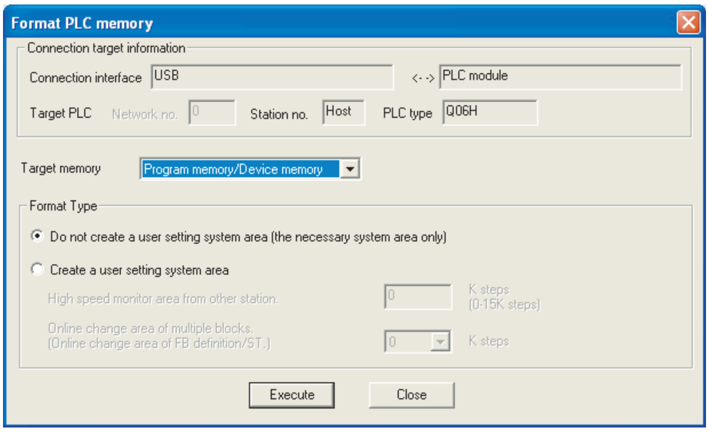
(2) Before using the program memory

Format the program memory by GX Developer.

(a) Formatting

Select [Online] → [Format PLC memory] in GX Developer.

Select "Program memory/Device memory" in "Target memory".



Format PLC memory

Connection target information

Connection interface: USB <--> PLC module

Target PLC: Network no. 0, Station no. Host, PLC type Q06H

Target memory: Program memory/Device memory

Format Type

● Do not create a user setting system area (the necessary system area only)

○ Create a user setting system area

High speed monitor area from other station: 0 K steps (0-15K steps)

Online change area of multiple blocks. (Online change area of FB definition/ST.): 0 K steps

Figure 5.14 Formatting the program memory

(b) Creating a user setting system area

When formatting a program memory, set the capacity of user setting system area.

1) Do not create a user setting system area (the necessary system area only)

The user setting system area is not created during formatting.

2) Create a user setting system area

The user setting system area is created during formatting.

Table5.6 provides the type of user setting system area.

Table5.6 Type of user setting system area

System area	Description
High speed monitor area from other station.	Setting this area increases the monitoring speed in the GX Developer connected to such as a serial communication module. When using RS-232 and USB together with GX Developer, this area is used to register monitor data from GX Developer connected to such as a serial communication module.
Online change area of multiple blocks (Online change area of FB definition/ST.)	Setting this area allows writing of multiple blocks to the CPU module in the RUN status. For the number of blocks that can be written to the CPU module in the RUN status, refer to the following. GX Developer Version 8 Operating Manual

Point

When a user setting system area is created, the available area reduces by the number of steps created in the area.

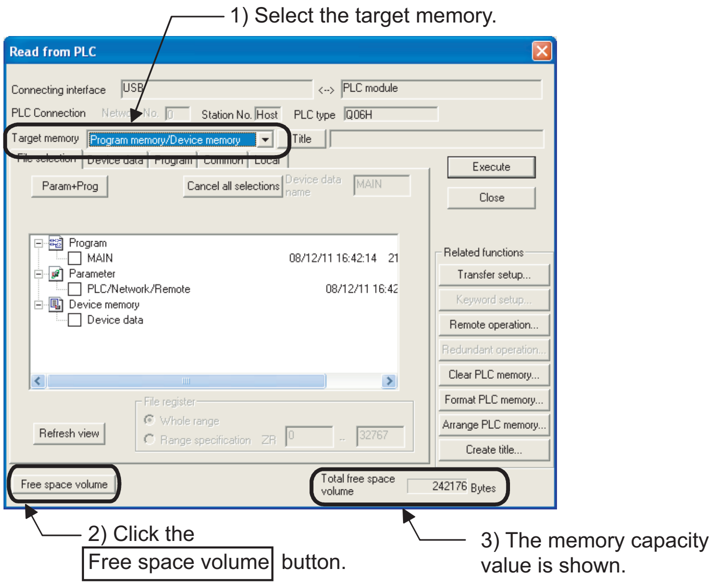
(c) Checking the memory capacity after formatting

Select [Online] → [Read from PLC] in GX Developer.

1) Select "Program memory/Device memory" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".



1) Select the target memory.

Read from PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type: Q06H

Target memory: Program memory/Device memory

Device data name: MAIN

Program: MAIN 08/12/11 16:42:14 21

Parameter: PLC/Network/Remote 08/12/11 16:42

Device memory: Device data

File register: ● Whole range ○ Range specification ZR 0 -- 32767

Total free space volume: 242176 Bytes

2) Click the Free space volume button.

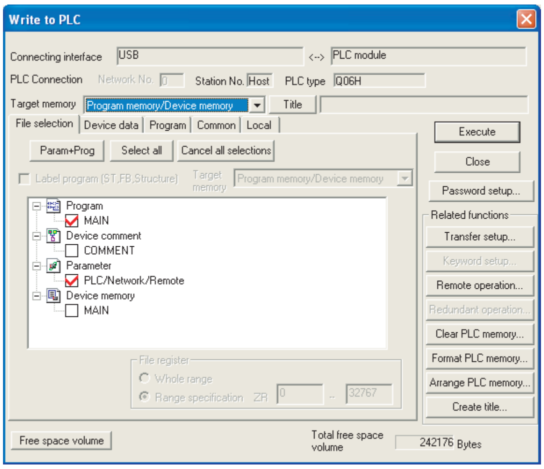
3) The memory capacity value is shown.

Figure 5.15 Procedure for checking the memory capacity

(3) Writing to the program memory

Select [Online] → [Write to PLC] in GX Developer.

Select "Program memory/Device memory" in "Target memory" on the Write to PLC screen.



Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q06H

Target memory: Program memory/Device memory

- Program
  - ☑ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - ☑ PLC/Network/Remote
- Device memory
  - □ MAIN

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume 242176 Bytes

Figure 5.16 Write to PLC screen

Point

The file size has its minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.
Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

5.2.3 Standard ROM

(1) Definition

This memory is for storing data such as parameters and programs.
Programs and parameters can be stored without battery backup.

(2) Checking the memory capacity

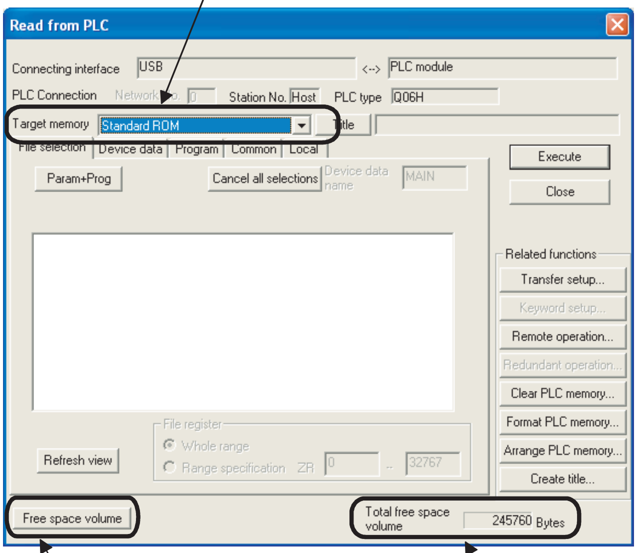
Select [Online] → [Read from PLC] in GX Developer.

1) Select "Standard ROM" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".

1) Select the target memory.



Read from PLC

Connecting interface USB <--> PLC module

PLC Connection Network No. 0 Station No. Host PLC type Q06H

Target memory Standard ROM Title

Device data name MAIN

File register ○ Whole range ○ Range specification ZR 0 ~ 32767

Total free space volume 245760 Bytes

2) Click the Free space volume button.

3) The memory capacity value is shown.

Figure 5.17 Procedure for checking the memory capacity

(3) Writing to the standard ROM

The following three methods are available.

- Batch-writing program memory data to the standard ROM by selecting [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] in GX Developer (☞ Section 5.2.6)
- Writing data by selecting [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] in GX Developer (☞ Section 5.2.6)
- Writing data with the automatic all data write from memory card to standard ROM function (☞ Section 5.2.7)

Point

The file size has its minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.
Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

5.2.4 Standard RAM

(1) Definition

This memory is for using file registers, local devices, and sampling trace files without a memory card.
Storing the file registers in the standard RAM allows fast access as data registers do.

Point

- If the size of files to be stored exceeds the standard RAM capacity:
  - store the files in the memory card, or
  - reduce the number of points of the file register, local device, or sampling trace.

  Note when file registers are stored in the memory card, the access speed will be slower than when they are stored in the standard RAM.

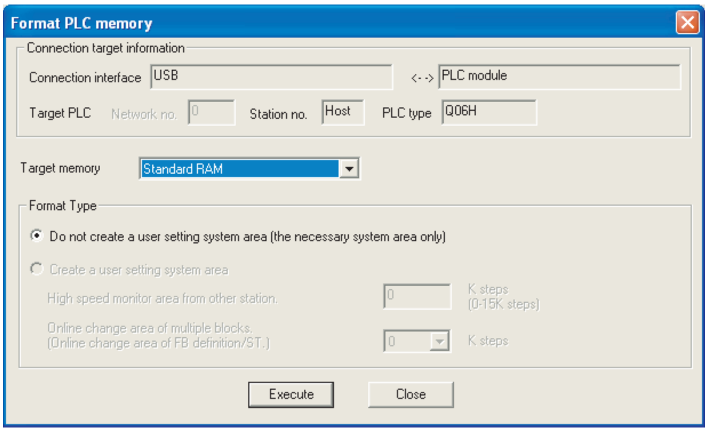
- Before storing a sampling trace file to the standard RAM, check the versions of the CPU module and GX Developer.
  (Appendix 2)

(2) Before using the standard RAM

Format the standard RAM by GX Developer.

(a) Formatting

Select [Online] → [Format PLC memory] in GX Developer.
Select "Standard RAM" in "Target memory".



Format PLC memory

Connection target information
Connection interface: USB <--> PLC module
Target PLC: Network no. 0 Station no. Host PLC type Q06H

Target memory: Standard RAM

Format Type
● Do not create a user setting system area (the necessary system area only)
○ Create a user setting system area
High speed monitor area from other station. 0 K steps (0-15K steps)
Online change area of multiple blocks. (Online change area of FB definition/ST.) 0 K steps

Figure 5.18 Formatting the standard RAM

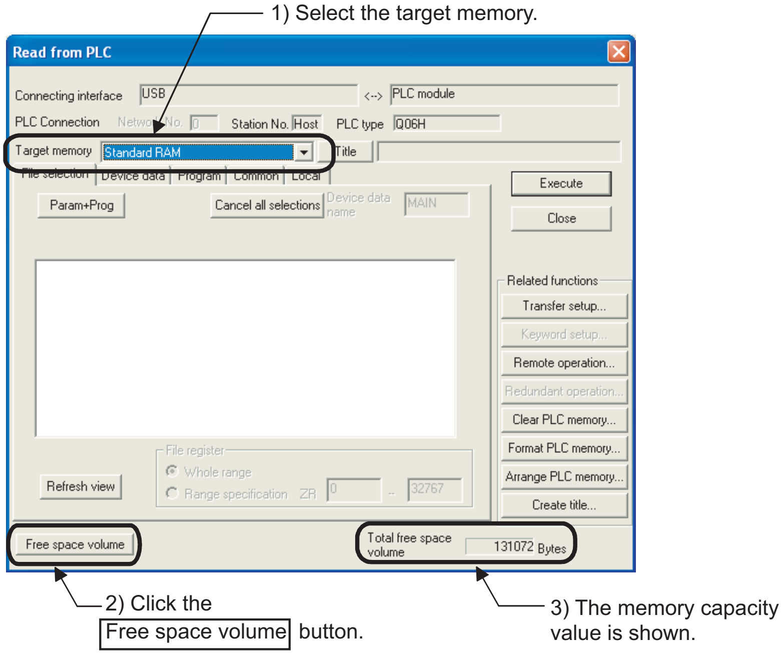
(b) Checking the memory capacity after formatting

Select [Online] → [Read from PLC] in GX Developer.

1) Select "Standard RAM" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".



Read from PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q06H

Target memory: Standard RAM

Total free space volume: 131072 Bytes

1) Select the target memory.

2) Click the Free space volume button.

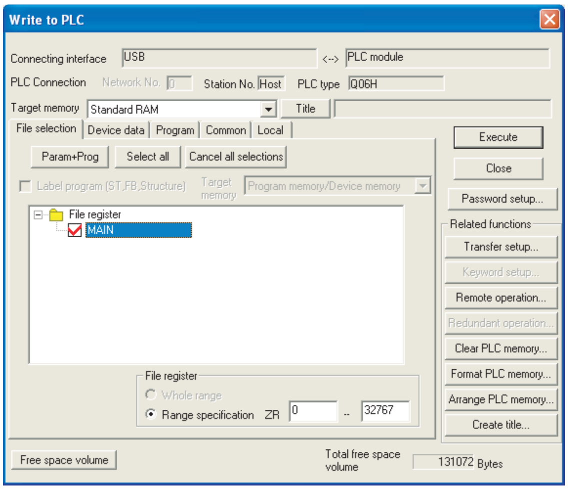
3) The memory capacity value is shown.

Figure 5.19 Procedure for checking the memory capacity

(3) Writing to the standard RAM

Select [Online] → [Write to PLC] in GX Developer.

Select "Standard RAM" in "Target memory" on the Write to PLC screen.



Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q06H

Target memory: Standard RAM

File register: ☑ MAIN

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume: 131072 Bytes

Figure 5.20 Write to PLC screen

Point

The file size has its minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.
Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

5.2.5 Memory card

(1) Definition

This memory is for expansion of a memory in the CPU module.

The following three types are available:

- SRAM card
- Flash card
- ATA card

(a) SRAM card

File registers in the SRAM card can be written or read by the sequence program.

The SRAM card is used when:

- the number of file register points is greater than the standard RAM capacity,
- the sampling trace function is used, or (☞ Section 6.14)
- 17 or more error history data are saved. (☞ Section 6.18.2)

When storing file registers to the SRAM card, the file registers can be written or read by the sequence program up to 1018K points.

(b) Flash card

Write data by GX Developer and read it by the sequence program. (Data can be read only by the sequence program.)

Use the Flash card when data are not changed.

File registers can be stored up to 1018K points.

(c) ATA card

This card is used for programmable controller user data (general-purpose data).

With the file access instruction (such as the SP. FWRITE instruction) in the sequence program, access the programmable controller user data in the ATA card in CSV format/binary format.

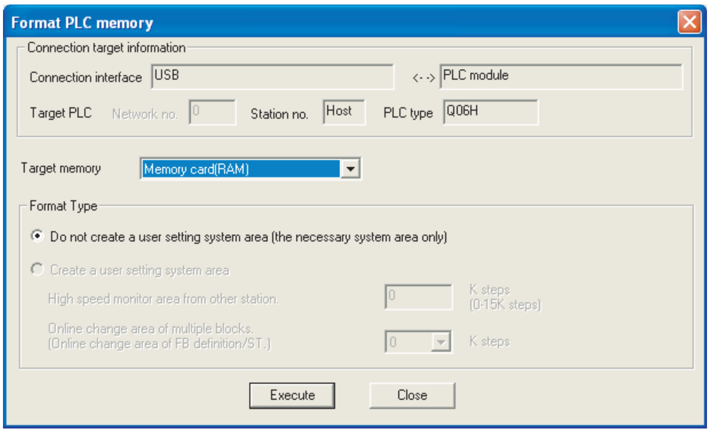
(2) Before using the SRAM card or ATA card

Format the SRAM card or ATA card by GX Developer.

(a) Formatting

Select [Online] → [Format PLC memory] in GX Developer

- When formatting the SRAM card, select "Memory card (RAM)" in "Target memory".
- When formatting the ATA card, select "Memory card (ROM)" in "Target memory".



Format PLC memory

Connection target information

Connection interface USB <-> PLC module

Target PLC Network no. 0 Station no. Host PLC type Q06H

Target memory Memory card(RAM)

Format Type

● Do not create a user setting system area (the necessary system area only)

○ Create a user setting system area

High speed monitor area from other station. 0 K steps (0-15K steps)

Online change area of multiple blocks. (Online change area of FB definition/ST.) 0 K steps

Figure 5.21 Formatting the SRAM card or ATA card

Point

- ● Use only GX Developer to format the ATA card.

  If formatting the ATA card by such as the formatting function of Microsoft® , Windows® , the card may not be used with the CPU module.

- ● When formatting the SRAM card or ATA card, the memory card information area is automatically secured. Therefore, the card capacity reduces by the area size.

Remark

Formatting is not required for the Flash card.

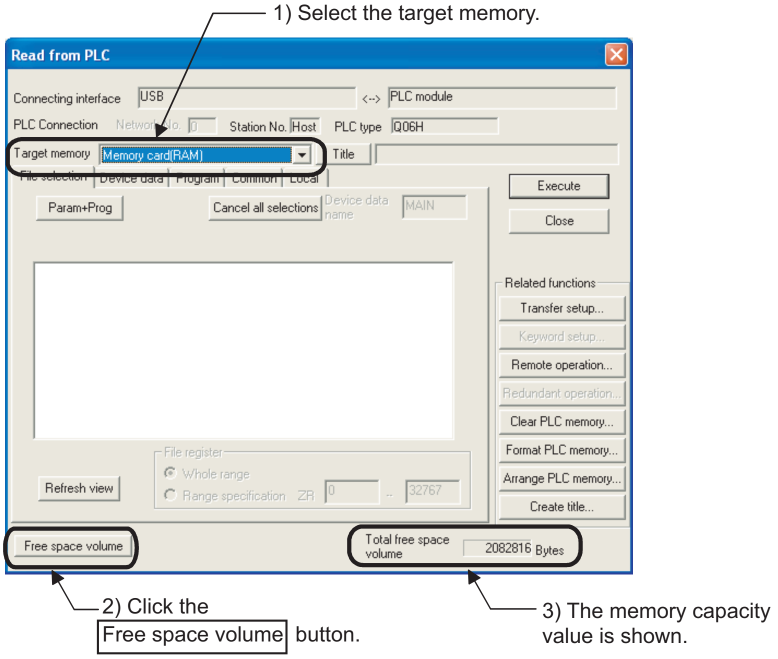
(b) Checking the memory capacity after formatting

Select [Online] → [Read from PLC] in GX Developer.

1) Select "Memory card (RAM)" or "Memory card (ROM)" in "Target memory" on the Read from PLC screen.

2) Click the Free space volume button.

3) The memory capacity appears in "Total free space volume".



Read from PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type: Q06H

Target memory: Memory card(RAM)

Total free space volume: 2082816 Bytes

Figure 5.22 Procedure for checking the memory capacity

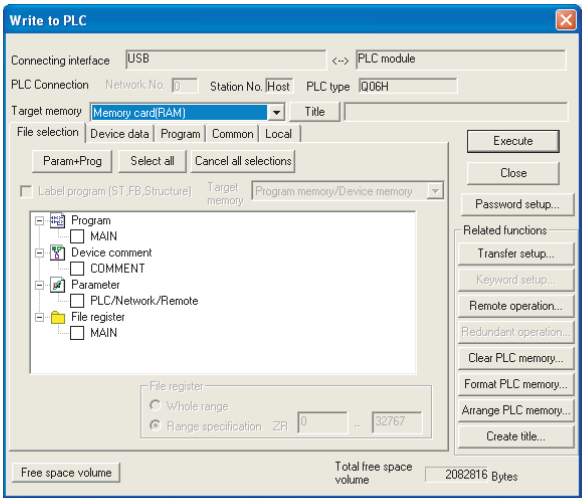
(3) Writing to the memory card

The following describes the operations before writing and the methods for writing.

(a) Writing to the SRAM card or the ATA card

Select [Online] → [Write to PLC] in GX Developer

- When writing data to the SRAM card, select "Memory card (RAM)" in "Target memory" on the Write to PLC screen.
- When writing data to the ATA card, select "Memory card (ROM)" in "Target memory" on the Write to PLC screen.



Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type: Q06H

Target memory: Memory card(RAM)

Total free space volume: 2082816 Bytes

Figure 5.23 Write to PLC screen

(b) Writing to the Flash card

The following two methods are available.

- Writing by "Write the program memory to ROM" (☞Section 5.2.6(1)(a))
- Writing by "Write to PLC (Flash ROM)" (☞ Section 5.2.6(1)(b))

Point

The file size has its minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.
Note that as the number of files increases, the difference between the occupied memory capacity and the actual file size increases.

(4) How to use the program stored in the memory card

Boot the program stored in the memory card to the program memory before its execution. (☞ Section 5.2.7)

5.2.6 Writing to the standard ROM and Flash card by GX Developer

(1) Methods for writing data to the standard ROM and Flash card and applications

Figure 5.24 provides the methods for writing data to the standard ROM and Flash card.

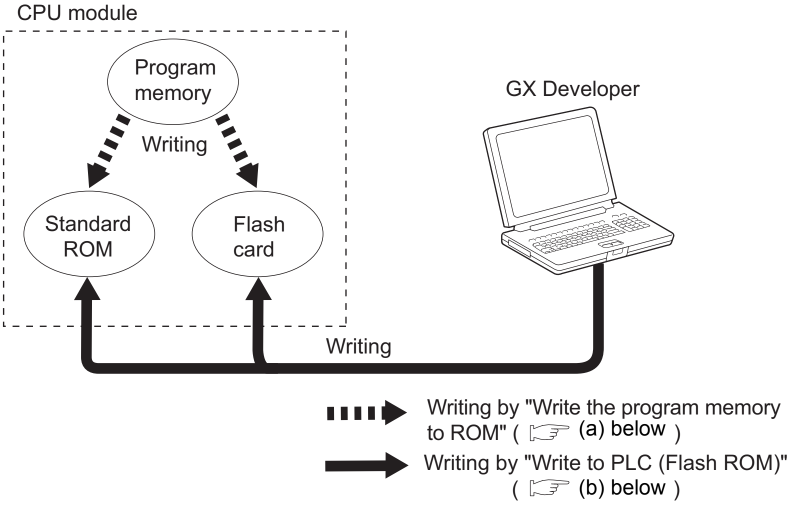

CPU module
Program memory
Writing
Standard ROM
Flash card
GX Developer
Writing
Writing by "Write the program memory to ROM" (☞ (a) below)
Writing by "Write to PLC (Flash ROM)" (☞ (b) below)

Figure 5.24 Methods for writing data to the standard ROM and Flash card

(a) Writing by "Write the program memory to ROM"

Data in the program memory are batch-written to the Flash card.
To batch-write the data, select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] in GX Developer.

Use this method when:

- debugging the program in the program memory and writing the debugged program to the Flash card without change for boot operation (☞Section 5.2.8), or
- storing the data in the program memory to the Flash card without battery backup.

(b) Writing by "Write to PLC (Flash ROM)"

Files specified by GX Developer are batch-written to the Flash card.
To batch-write the files, select [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] in GX Developer.

Use this method when:

- storing parameters, device initial values, or device comments whose size exceeds the program memory capacity to the Flash card, or
- using file registers storied in the Flash card.

(2) Writing to the standard ROM and Flash card

The following describes the operations before writing and the methods for writing.

(a) Before writing

Check the following.

1) Preparing files to be written

Writing a file to the standard ROM or Flash card automatically deletes all files stored in the standard ROM or Flash card.

Also write all files same as the stored files together.

2) Boot operation

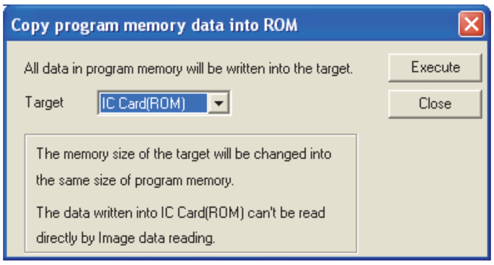
When storing parameters to the standard ROM or Flash card at boot operation, make the boot file setting described in Section 5.2.8.

(b) Writing procedure

The following describes a procedure for writing a file to the standard ROM and Flash card.

1) Procedure for [Write the program memory to ROM] in GX Developer

- Select [Online] → [Write to PLC (Flash ROM)] → [[Write the program memory to ROM].
- The program memory data into ROM screen appears.



Copy program memory data into ROM

All data in program memory will be written into the target.

Target: IC Card(ROM)

The memory size of the target will be changed into the same size of program memory.

The data written into IC Card(ROM) can't be read directly by Image data reading.

Figure 5.25 Copy program memory data into ROM screen

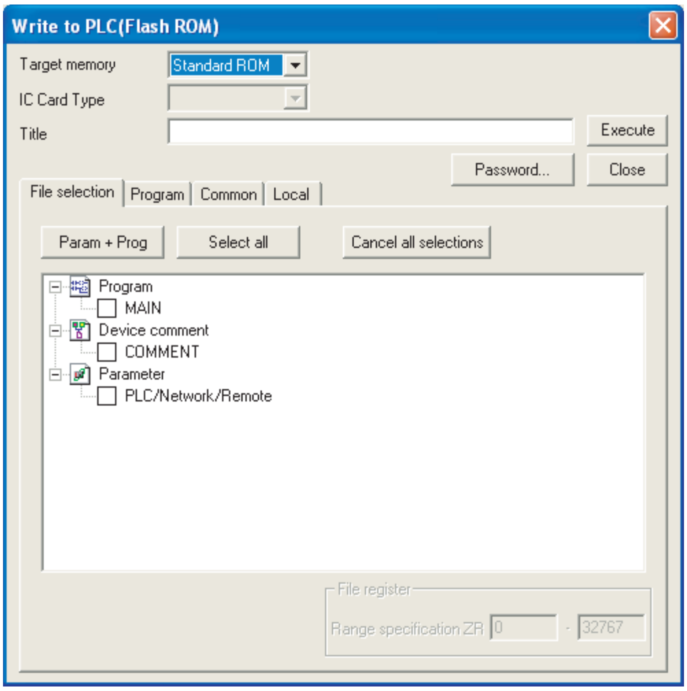
- Select the memory to be written in "Target" to write the program memory file to the Flash card.

Point

- When files are written by "Write the program memory to ROM", the capacity of used target memory is equal to that of used program memory.
  To fully use the capacity of the target memory, write the files by "Write to PLC (Flash ROM)".
- When writing data that cannot be stored to the program memory (file register) to the Flash card, write it by "Write to PLC (Flash ROM)".

2) Procedure using [Write to PLC (Flash ROM)] in GX Developer

- Select [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)].
- The Write to PLC (Flash ROM) screen appears.



Write to PLC(Flash ROM)

Target memory: Standard ROM

IC Card Type:

Title:

File selection

- □ MAIN
- □ COMMENT
- □ PLC/Network/Remote

File register

Range specification ZR 0 - 32767

Figure 5.26 Write to PLC (Flash ROM) screen

- Select the target memory.
- Select a file to be written and write it to the Flash card.

(3) Adding or changing a file in the standard ROM and Flash card

Add or change the file by either of the following methods (stored files cannot be added or changed directly).

(a) When writing files using [Write the program memory to ROM] in GX Developer

- To read all files in the program memory, select [Online] → [Read from PLC].
- Add or change the read files.
- Write the added or changed files to the program memory.
- Select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] and write the files to the standard ROM or Flash card.

(b) When writing files using [Write to PLC (Flash ROM)] in GX Developer

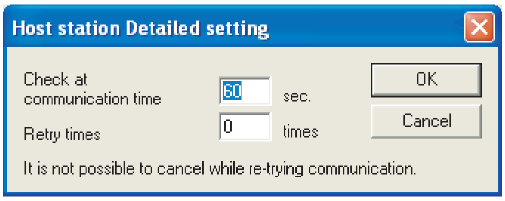
- To read all files in the Flash card, select [Online] → [Read from PLC].
- Add or change the read files.
- Select [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] and write the files to the standard ROM or Flash card.

(4) Precautions

(a) Setting the communication time check period in GX Developer

Since writing a file to the standard ROM or Flash card takes time, set "Check at communication time" in GX Developer to 60 seconds or longer.

If the set time is short, GX Developer may time out.



Host station Detailed setting

Check at communication time 60 sec.

Retry times 0 times

It is not possible to cancel while re-trying communication.

Figure 5.27 Check at communication time setting

(b) Writing a file from GX Developer in another station via CC-Link

Since writing a file to the standard ROM or Flash card takes time, make the CPU monitoring time setting (SW000A) of CC-Link to 60 seconds or longer.

(The default value of 90 seconds can be used.)

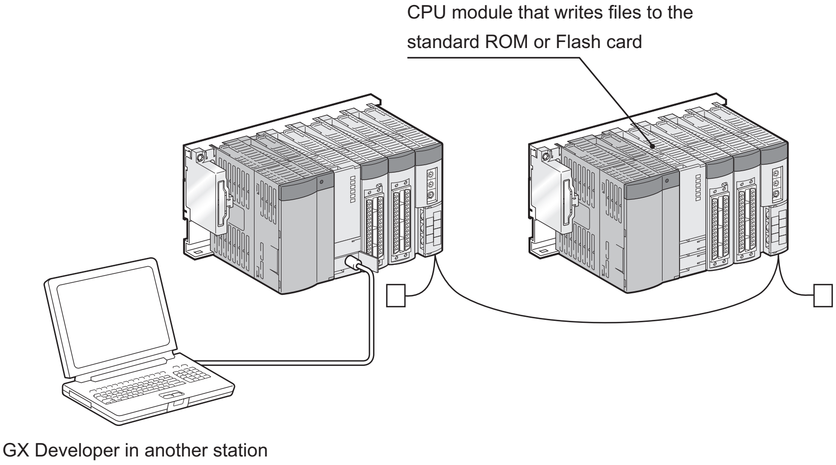

CPU module that writes files to the standard ROM or Flash card
GX Developer in another station

Figure 5.28 Writing a file from GX Developer in another station(via CC-Link)

Remark

For settings of the communication time check period and CPU monitoring time, refer to the following.

☞ GX Developer Version8 Operating Manual

(c) Time required for “Write to PLC (Flash ROM)”

Using "Write to PLC (Flash ROM)” writes data to the entire space in the Flash card.

Therefore, even if a program having the small number of steps is written to the Flash card, the processing takes time.

(Writing using the Q2MEM-4MBF at a communication speed of 115.2Kbps with an RS-232 interface requires 14 minutes.)

When writing data to the Flash card, increase the transmission speed or use an USB.

Communication time takes time when “Write to PLC (Flash ROM)” is executed from another station.

(d) Online change

If "Write to PLC (Flash ROM)" is executed during the RUN status in the situations described below, an error may occur and the CPU module may stop. To avoid this, execute "Write to PLC (Flash ROM)" after setting the RUN/STOP/RESET switch to STOP.

1) File registers stored in the Flash card are used in the sequence program, or

2) although "Not used" is set to the file registers by the PLC parameter dialog box, they are used in the sequence program.

(e) Writing/reading data during execution of “Write to PLC (Flash ROM)”

While “Write to PLC (Flash ROM)” is executed, data cannot be read from/written to other modules.

Therefore, time-out may occur in other modules.

(f) When “Write to PLC (Flash ROM)” is executed during the STOP status

Do not set the RUN/STOP/RESET switch to RUN during the writing.

The CPU module may not enter the RUN status normally during execution of "Write to PLC (Flash ROM)".

Set the RUN/STOP/RESET switch to RUN after the writing.

5.2.7 Automatic all data write from memory card to standard ROM Note5.2

(1) Definition

The automatic all data write from memory card to standard ROM function (hereafter, automatic write to standard ROM) automatically writes parameters and programs written to a memory card to the standard ROM.
As shown in Figure 5.29, this function boots programs and parameters from a memory card to the program memory and writes the booted programs and parameters from the program memory to the standard ROM.

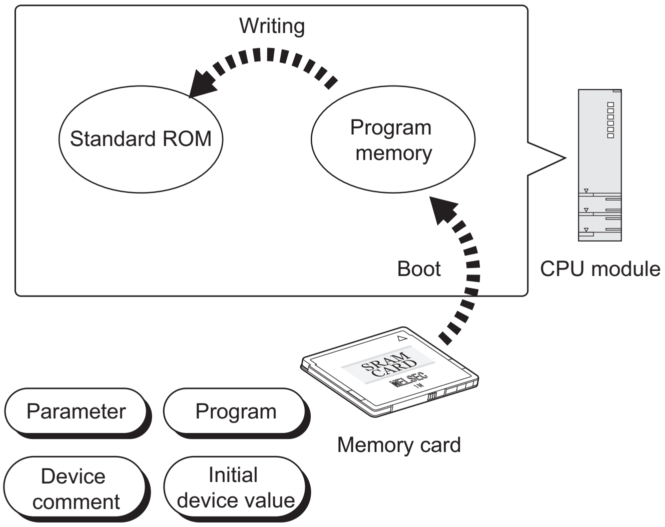


Figure 5.29 Automatic write to standard ROM

(2) Applications of automatic write to standard ROM

Using this function can write programs and parameters in a memory card to the standard ROM without GX Developer (personal computer).
This function is useful when:

- the same parameters and programs are written to multiple CPU modules, or
- the same environment is configured at a remote site.

Note5.2 High performance

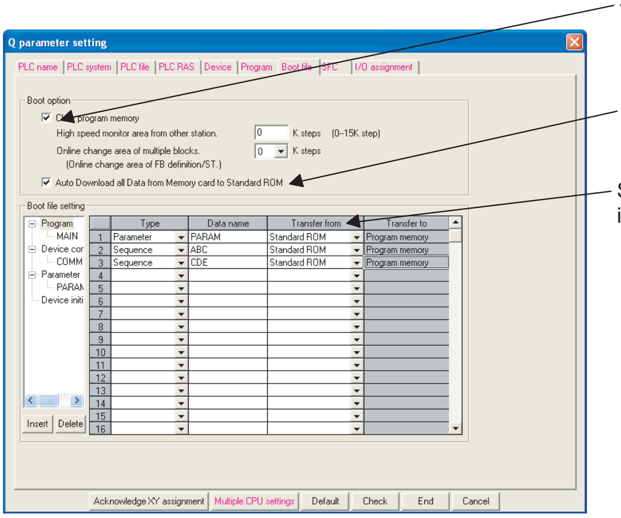
When using the automatic write to standard ROM function for the High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.2)
When set memory card is mounted to the High Performance model QCPU that does not support the automatic write to standard ROM function, boot operation is performed from the standard ROM.

(3) Execution procedure for automatic write to standard ROM

Perform this function by the following procedure.

(a) Setting with GX Developer

1) Select "Clear program memory" and "Auto Download all Data from Memory card to Standard ROM" in the Boot file tab of the PLC parameter dialog box.
Set the programs and parameters to be booted in the Boot file tab.
(Select "Standard ROM" in "Transfer from".)



Q parameter setting

Boot option

☑ Clear program memory

High speed monitor area from other station. 0 K steps (0--15K step)

Online change area of multiple blocks. 0 K steps
(Online change area of FB definition/ST.)

☑ Auto Download all Data from Memory card to Standard ROM

Boot file setting

	Type	Data name	Transfer from	Transfer to
1	Parameter	PARAM	Standard ROM	Program memory
2	Sequence	ABC	Standard ROM	Program memory
3	Sequence	CDE	Standard ROM	Program memory
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

Select "Clear program memory".

Select "Auto Download all Data from Memory card to Standard ROM".

Select "Standard ROM" in "Transfer from".

Figure 5.30 Auto Download all Data from Memory card to Standard ROM setting

2) Store the set parameters and programs to be booted to the memory card.

(b) Operation on the CPU module side

1) Power off the CPU module.

2) Mount the memory card storing the parameters and programs to be booted to the CPU module.

3) Set the parameter-valid drive with the DIP switches on the CPU module to the mounted memory card.

- SRAM card •••••• SW2: on, SW3: off
- Flash card and ATA card •••••• SW2: off, SW3: on

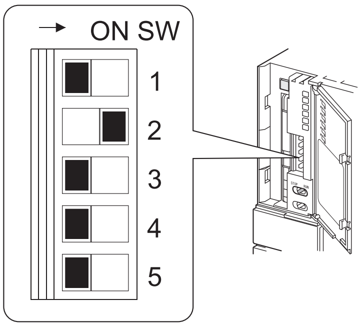


Figure 5.31 DIP switch setting when the SRAM card is mounted

4) Power on the CPU module.
Boot the files in the memory card to the program memory.
After boot operation, write the program memory data to the standard ROM.

5) After automatic write to standard ROM, the BOOT LED and ERR. LED start flashing and the CPU module enters stop error status.

6) Power off the CPU module.

7) Remove the memory card and set the parameter-valid drive with the DIP switches on the CPU module to the standard ROM.

- Standard ROM •••••• SW2: on, SW3: on

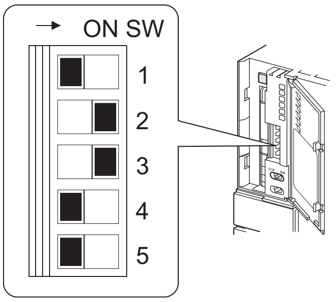


Figure 5.32 When setting the parameter-valid drive to the standard ROM with the DIP switches

8) Powering on the CPU module starts boot operation from the standard ROM to the program memory and can start actual operation.

(4) Precautions

The following describes precautions for automatic write to standard ROM.

(a) When a file with the same name exists in the program memory

When the program memory has a file whose name is the same as that of the one to be booted from the memory card, the file is overwritten by the data in the memory card.
When the program memory does not have the same file name, the file to be booted from the memory card is added to the program memory.
If the program memory capacity is exceeded then, "FILE SET ERROR” (error code: 2401) is detected.

(b) Program memory clear at boot operation

Whether files are booted after program memory clear or booted without program memory clear can be selected.
For automatic write to standard ROM, setting boot operation after program memory clear can prevent the program memory capacity from being exceeded at boot operation.

(c) Parameter-valid drive setting with the DIP switches when using this function

The automatic write to standard ROM setting becomes valid only when the parameter-valid drive are set to the memory card with the DIP switches.
Therefore, when starting actual operation after the automatic write to standard ROM, deselecting "Auto Download all Data from Memory card to Standard ROM” in the Boot file tab is unnecessary.

5.2.8 Operating the program in the standard ROM and memory card (boot operation)

This section describes methods for operating the program stored in the standard ROM and memory card.

(1) Operating the program in the standard ROM and memory card

To execute the boot operation, set the names of files to be booted in the Boot file tab of the PLC parameter dialog box. (☞ (3), (4) in this section)

The program stored in the standard ROM and memory card, whose file name is specified in the Boot file tab, is booted with the program memory after the CPU module is powered off and then on or is reset.

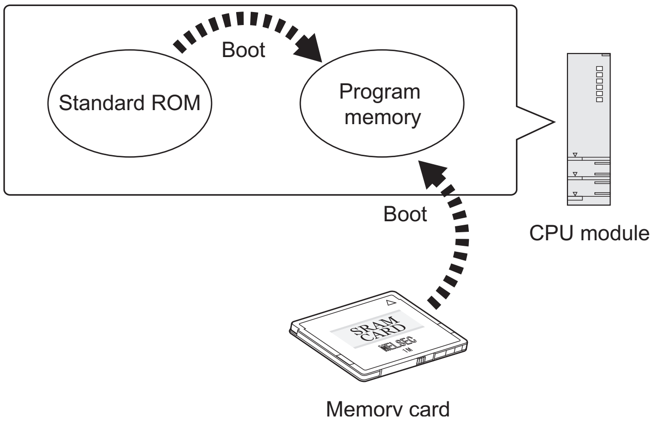


Figure 5.33 Boot operation

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

Table5.7 provides the combinations of bootable file, transfer source, and transfer destination.

Table5.7 Combinations of bootable file, transfer source, and transfer destination

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

○: Bootable, ×: Cannot be booted

*1: The intelligent function module parameter is included.

(3) Procedure before boot operation

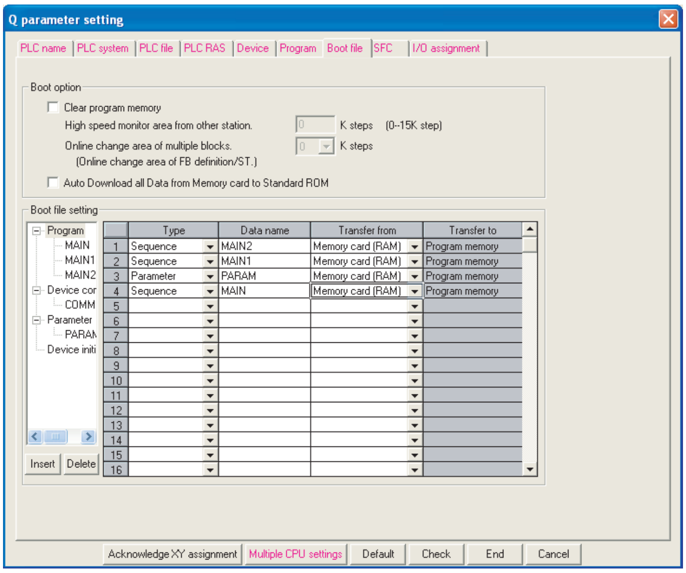
The following explains the procedures to store the files to be booted in the standard ROM or memory card and start boot operation.

(a) Creating a program

Create a program.

(b) Boot file setting

Set the files ("Type", "Data name", and "Transfer from") to be booted to the program memory in the Boot file tab of the PLC parameter dialog box.



Boot option

□ Clear program memory

High speed monitor area from other station. 0 K steps (0--15K step)

Online change area of multiple blocks. 0 K steps
(Online change area of FB definition/ST.)

□ Auto Download all Data from Memory card to Standard ROM

Boot file setting

	Type	Data name	Transfer from	Transfer to
1	Sequence	MAIN2	Memory card (RAM)	Program memory
2	Sequence	MAIN1	Memory card (RAM)	Program memory
3	Parameter	PARAM	Memory card (RAM)	Program memory
4	Sequence	MAIN	Memory card (RAM)	Program memory
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

Figure 5.34 Boot file tab

(c) Hardware setting for boot operation

Set the parameter-valid drive to a memory where parameters are to be stored with the DIP switches.

For boot operation, set the storage location of the parameters to the standard ROM or memory card.

(☞ (6)(a) in this section)

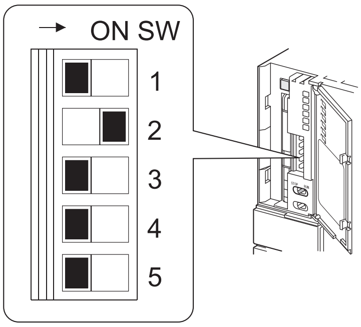


Figure 5.35 When setting the parameter-valid drive to the SRAM card with the DIP switches

(d) Installing a memory card (when the boot source is a memory card)

Mount the memory card to the CPU module.

(e) Writing parameters to the standard ROM or memory card

Write the parameters to a memory set as the parameter-valid drive with the DIP switches.

Also, write the files set in the Boot file tab in (b) to the transfer source memory.

(f) Executing the program

Set the RESET/L.CLR switch to RESET.

The BOOT LED turns on after a boot from the specified memory is completed.

(g) Checking whether a boot is normally completed

The following status indicates normal completion of boot operation.

- The BOOT LED turns on.
- The special relay (SM660) turns on.
- The data written to the transfer source memory and the data in the program memory are found the same by verification made by selecting [Online] → [Verify with PLC] in GX Developer.

(4) Operation for stopping 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) Write the parameters for which the boot file setting is not configured to the program memory.
   Set the parameter-valid drive to the program memory (SW2: off, SW3: off) with the DIP switches on the CPU module.

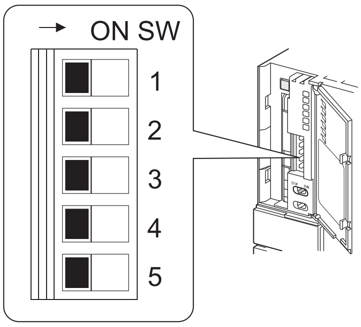


Figure 5.36 When setting the parameter-valid drive to the program memory with the DIP switches

2) Power on again or reset the CPU module.

(5) Changing a program file in the RUN status Note5.3

(a) Methods

Use the following instructions.

- PLOADP instruction (program transfer from memory card to program memory)
- PUNLOADP instruction (program deletion from program memory)
- PSWAPP instruction (program deletion from program memory and program transfer from memory card to program memory)

For details of each instruction, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

(b) Program setting when a program file is changed

Even if a program file has been changed in the RUN status, the settings configured in the Program tab of the PLC parameter dialog box are not changed.

When setting the CPU module to STOP, correct the settings in the Program tab of the PLC parameter dialog box to the settings configured in the RUN status (addition/change/deletion of program name).

Otherwise, an error will occur when the CPU module is set from STOP to RUN.

(6) Boot operation precautions

(a) Storage location of parameters

Store the parameters set in the Boot file tab of the PLC parameter dialog box to the standard ROM or memory card.

If the parameters are stored in the program memory, the CPU module ignores the parameter settings even if the parameter-valid drive is set to the program memory with the DIP switches. (☞ Section 5.2.10)

Note5.3 Redundant

The Redundant CPU does not support the program file change in the RUN status.

(b) Online change in boot operation

1) SRAM card, ATA card

When a program in the program memory is written in the RUN status (☞ Section 6.12), the change can be updated to the program in the boot source SRAM card or ATA card.

2) Standard ROM, Flash card

Even if data are written to a program in the program memory in the RUN status, the change is not updated to the program in the boot source standard ROM or flash card.

Therefore, write the same file whose program was written to the program memory to the standard ROM or flash card by Write to PLC (Flash ROM). (☞ Section 5.2.5, Section 5.2.6)

(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 reduces by 1 when:

- a heading is set, or
- the parameters set in the Boot file tab of the PLC parameter dialog box and stored in the standard ROM or memory card is booted.

(d) Boot operation when the ATA card is used

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

- Boot from the ATA card
- Boot from the standard ROM with the ATA card mounted

(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 (4) in this section and stop the boot operation.

(f) Size after a boot from the memory card

The size unit of a file stored in each memory differs between the memory card and the program memory. (☞ Section 5.4.4)

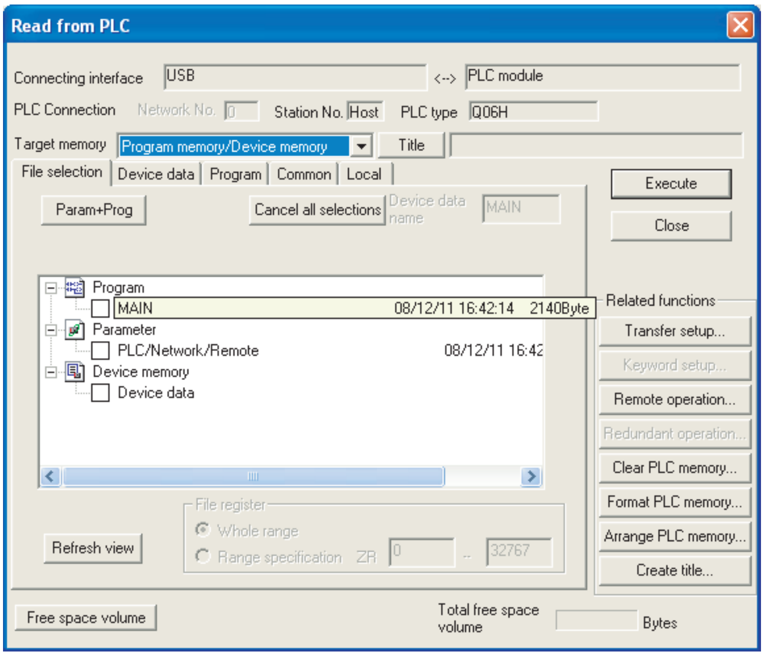
Therefore, note that files transferred from the memory card to the program memory differ in memory capacity between before and after the transfer.

(g) Program written to the 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.

5.2.9 Details of written files

For each file written to the CPU module, its name, size, and created date and time set at the file creation are appended.
The file is displayed on the Read from PLC screen, opened by selecting [Online] → [Read from PLC] in GX Developer, as shown below.



Read from PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q06H

Target memory: Program memory/Device memory Title

File selection | Device data | Program | Common | Local

Param+Prog | Cancel all selections | Device data name: MAIN

Program
□ MAIN 08/12/11 16:42:14 2140Byte
Parameter
□ PLC/Network/Remote 08/12/11 16:42
Device memory
□ Device data

Refresh view

File register: ● Whole range ○ Range specification ZR 0 - 32767

Free space volume | Total free space volume Bytes

Figure 5.37 Displaying the details of a file

(a) File name

1) 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)*1 and an extension (3 characters).
Create a file name with upper-case characters only.
An extension is automatically appended according to the type set when the file was created.

*1: Only ASCII characters can be used for a name of the file stored in a memory card.
Also, characters other than ASCII may not be used for a name of the file stored in memories other than a memory card.
For ASCII characters that can be used, refer to the following.
☞ Operating manual for the programming tool used

2) Characters that cannot be used for a file name

he following reserved words for Microsoft® , Windows® cannot be used as a file name.
COM1 to COM9, PRN, LPT1 to LPT9, NULL, AUX, CLOCK$, CON

3) How to specify a file name in the sequence program

In single-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.
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.

(b) Date and time

The date and time when a file was written to the CPU module is shown.
The date and time are appended according to the clock set on the GX Developer (personal computer) side.

(c) Size

The file size when the file was written from GX Developer to the CPU module is shown in units of bytes (To display the latest data, select [Online] → [Read from PLC] and click the Refresh view button). At least 64 bytes (136 bytes for a program) are added to the file created by an user except a file register file. (☞ Section 5.4.3)

5.2.10 Specifying valid parameters (parameter-valid drive setting)

The drive (memory) storing parameters to be valid can be selected with the DIP switches on the CPU module.

(1) Selection method of parameter-valid drive

Select the drive with the DIP switches (SW2 and SW3).

Table5.8 Drive specification with SW2 and SW3

SW2	SW3	Parameter-valid drive
Off	Off	Drive 0 (program memory)
On	Off	Drive 1 (memory card RAM)
Off	On	Drive 2 (memory card ROM)
On	On	Drive 4 (standard ROM)

(2) When to determine valid parameters

The CPU module determines whether to validate the parameters when:

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

The CPU module automatically enables parameters stored in the drive set with SW2 and SW3 at the above timing and operates by the parameter settings.

5.3 Program File Structure

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

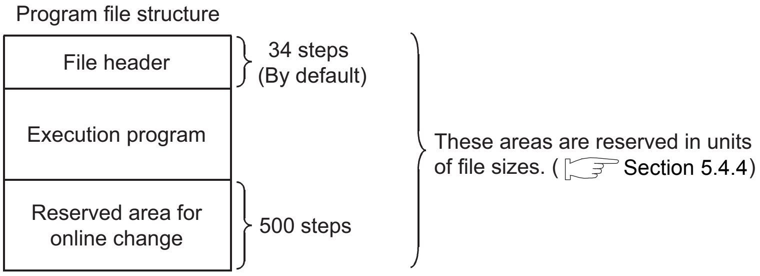

Program file structure
File header
Execution program
Reserved area for online change
34 steps (By default)
500 steps
These areas are reserved in units of file sizes. (Section 5.4.4)

Figure 5.38 Program file structure

(1) Details of each structure

The capacity of the programs stored in the CPU module program memory is the total of above three areas.

(a) File header

This area stores the name, size, and created date of files.

The file header size ranges from 25 to 35 steps (100 to 140 bytes) depending on the setting made in the Device tab of the PLC parameter dialog box (34 steps is set by default).

(b) Execution program

This area stores the created program.

(c) Reserved area for online change

This area is used when the number of steps is increased after writing data in the RUN status from GX Developer.

When such operation is performed, the remaining reserved area for online change is shown.

1) Default

500 steps (2000 bytes) is set by default.

2) Setting change

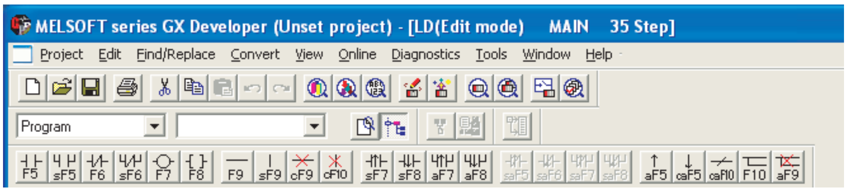
To change the number of steps, select [Online] → [Write to PLC] → <<Program>> tab in GX Developer.

This setting can be made when data are written in the RUN status. (Section 6.12.1)

(2) Displaying the program capacity on the GX Developer screen

During programming by GX Developer, the program size (total of the file header size and the number of steps in the created program) is displayed by the number of steps as shown in Figure 5.39.

The program size is displayed.



MELSOFT series GX Developer (Unset project) - [LD(Edit mode) MAIN 35 Step]

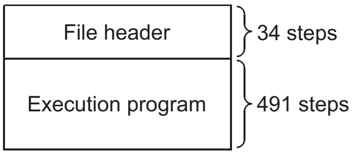
Project Edit Find/Replace Convert View Online Diagnostics Tools Window Help
Program
F5 sF5 F6 sF6 F7 F8 F9 sF9 cF9 cF10 sF7 sF8 aF7 aF8 saF5 saF6 saF7 saF8 aF5 caF5 caF10 F10 aF9

Figure 5.39 Displaying the program size

Point

- The program size displayed during programming by GX Developer is the total of the file header and execution program, and the capacity of the reserved area for online change (500 steps) is not included.

  Example The size of program having the execution program area of 491 steps is displayed on the GX Developer screen as shown below (The file header is fixed to 34 steps).



Display on the GX Developer screen:
34 steps + 491 steps = 525 steps

Figure 5.40 File status displayed on the GX Developer screen

- Since files are stored in units of file sizes in the program memory, the program size displayed during programming by GX Developer may differ from the program file size in the CPU module. (☞ Section 5.4.3)

5.4 File Operations by GX Developer and Handling Precautions

5.4.1 File operations

Table5.9 shows the functions can be performed to files stored in the program memory, standard ROM, and memory card by the online functions of GX Developer.

However, the executable operations depend on the password registration setting by GX Developer, status of the system protect switch of the CPU module, and the CPU module status.

Table5.9 File operations executable from GX Developer

File operation	Operation detail	Operability[*1]														
		Basic model QCPU			High Performance model QCPU				Process CPU				Redundant CPU			
		A	B	C	A	B	C	D	A	B	C	D	A	B	C	D
Read from PLC	Reads a file from the target memory.	○	△	○	○	△	○	○	○	△	○	○	○	△	○	○
Write to PLC	Writes a file to the program memory or SRAM card.	△	△	×	△	△	○	×	△	△	○	×	△	△	○	×
	Writes a file to the standard ROM.	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Verify with PLC	Verifies the file in the target memory and the file of GX Developer.	△	△	○	△	△	○	○	△	△	○	○	△	△	○	○
Write the program memory to ROM	Batch-writes files stored in the program memory to the Flash card.	○	○	×	○	○	○	×	○	○	○	×	○	○	○	×
	Batch-writes the files stored in the program memory to the standard ROM.	○	○	×	○	○	○	×	○	○	○	×	○	○	○	×
Write to PLC (Flash ROM)	Batch-writes specified files from GX Developer to the Flash card.	×	×	×	△	△	○	×	△	△	○	×	△	△	○	×
	Batch-writes the specified files to the standard ROM by GX Developer.	×	×	×	△	△	○	×	△	△	○	×	△	△	○	×
Delete PLC data	Deletes the files stored on the memory.	△	△	×	△	△	×	×	△	△	×	×	△	△	×	×
Format PLC memory	Formats a memory.	○	○	×	○	○	×	×	○	○	×	×	○	○	×	×
Arrange PLC memory	Rearranges files stored in a memory at random.	○	○	×	○	○	×	×	○	○	×	×	○	○	×	×
Online change (ladder mode)	Writes data changed in the ladder mode to the program memory.	△	△	○	△	△	○	×	△	△	○	×	△	△	○	×

○: Executable, △: Executable when entered password matches, ×: Cannot be executed.

[*1] The following shows definitions of the alphabets in the Operability field.
A: A password for write protection is set to the file.
B: A password for read and write protections are set to the file.
C: The CPU module is in the RUN status.
D: The system protect switch of the CPU module is on.

5.4.2 Precautions for handling files

(1) Power-off or reset at file operation

When the CPU module is powered off or reset during file operation, files in each memory will be the status as shown in Table5.10.

Table5.10 File status when the CPU module is powered off or reset during file operation

CPU module	Memory status
Basic model QCPU	Files in the memories will corrupt or the values in the files will be inconsistent.
High Performance model QCPU Process CPU Redundant CPU	Files in the memories will not corrupt (For use of a memory card, the memory will be in the status only when the CPU module is powered on without the memory card being removed).

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.

(2) Concurrent writing from multiple GX Developers to one file

GX Developers other than the one writing data to a file cannot access the file until the writing ends.
Also, GX Developers other than the one accessing a file cannot write data to the file until the access ends.
Therefore, write data to one file from multiple GX Developers one by one.

(3) Concurrent access from multiple GX Developers to different files

Except the currently viewed GX Developer, maximum 10 GX Developers can access to different files in one CPU module concurrently.

5.4.3 File size

The size of a file used for the CPU module depends on the file type.
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. (☞ Section 5.4.4)

(1) Basic model QCPU

When using the program memory, standard RAM, or standard ROM, calculate the rough size of each file with reference to Table5.11.

Table5.11 Calculation of file size (Basic model QCPU)

<table>
<tr><th>Function</th><th>Rough file size (unit: Byte)</th></tr>
<tr><td>Drive heading</td><td>64</td></tr>
<tr><td>Parameter</td><td>Default:522 (can be increased by parameter setting.)
Reference
Boot setting → 94
• With CC-Link IE Controller Network setting → Increase up to 7214
• With the MELSECNET/H setting → Increase up to 6180
• With the Ethernet setting → Increase up to 922
• With the CC-Link setting → Increase up to the values in the following table (The values indicate an increment of each module).
<table>
<tr><th rowspan="2">CC-Link setting</th><th colspan="3">Mode setting</th></tr>
<tr><th>Ver.1 mode</th><th>Ver.2 mode</th><th>Ver.2 additional mode</th></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>
• With the remote password setting → 64 + 20 + (number of target modules × 10), increase up to 164</td></tr>
<tr><td>Sequence program</td><td>136*1+ (4× ((number of steps) + (number of steps of reserved area for online change)))</td></tr>
<tr><td>Device comment</td><td>74 + (total comment data size of each device)
Comment data size per device = 10 + 10250 × a + 40 × b
• a: Quotient of ((number of device points)/256)
• b: remainder of ((number of device points)/256)</td></tr>
<tr><td>File register</td><td>2 × (number of file register points)</td></tr>
<tr><td>Initial device value</td><td>66 + 44 × n + 2 × (total number of device points set to the initial device value)
• n: number of settings of the initial device value</td></tr>
<tr><td>Intelligent parameter</td><td>68 + (24 × number of set modules) + parameter size of each utility</td></tr>
<tr><td>User setting area</td><td>Setting value at formatting (0 to 3K)</td></tr>
<tr><td>Online change (multiple blocks) setting</td><td>Setting value at formatting (0/1.25K/2.5K)</td></tr>
</table>

*1: 136 is set by default (This value depends on the parameter setting).

Remark

For calculation example of memory capacity, refer to Section 5.4.4.

(2) High Performance model QCPU, Process CPU, and Redundant CPU

When using the program memory, standard RAM, standard ROM, or memory card, calculate the rough size of each file with reference to Table5.12.

Table5.12 Calculation of file size (High Performance model QCPU, Process CPU, and Redundant CPU)

<table>
<tr><th>Function</th><th>Rough file size (unit: Byte)</th></tr>
<tr><td>Drive heading</td><td>64</td></tr>
<tr><td>Parameter</td><td>Default (can be increased by parameter setting.)
<table>
<tr><th>CPU module</th><th>File size</th></tr>
<tr><td>High Performance model QCPU</td><td><b>564</b></td></tr>
<tr><td>Process CPU</td><td><b>558</b></td></tr>
<tr><td>Redundant CPU</td><td><b>764</b></td></tr>
</table>
Reference
Boot setting → 70 + (18 × (number of files))
• With CC-Link IE Controller Network setting → Increase up to 7214/module
• With the MELSECNET/H setting → Increase up to 6180/module
• With the Ethernet setting → Increase up to 922/module
• With the CC-Link setting → Increase up to the values in the following table (The values indicate an increment of each module.)
<table>
<tr><th rowspan="2">CC-Link setting</th><th colspan="3">Mode setting</th></tr>
<tr><th>Ver.1 mode</th><th>Ver.2 mode</th><th>Ver.2 additional mode</th></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>
• With the remote password setting → 64 + 20 + (number of target modules × 10), increase up to 164</td></tr>
<tr><td>Sequence program</td><td>136[*1] + (4 × ((number of steps) + (number of steps of reserved area for online change)))</td></tr>
<tr><td>Device comment</td><td>74 + (total comment data size of each device)
Comment data size per device = 10 + 10250 × a + 40 × b
• a: Quotient of ((number of device points)/256)
• b: remainder of ((number of device points/256)</td></tr>
<tr><td>Initial device value</td><td>66 + 44 × n + 2 × (total number of device points set to the initial device value)
• n: number of settings of the initial device value</td></tr>
<tr><td>User setting area</td><td>Setting value at formatting (0 to 15K)</td></tr>
<tr><td>Online change (multiple blocks) setting</td><td>Setting value at formatting (0/2K/4K)</td></tr>
<tr><td>File register</td><td>2 × (number of file register points)</td></tr>
<tr><td>Sampling trace file[*6]</td><td>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)[*2]
• Apply the following values to N1 to N3 according to the items set in "Trace additional information" of the Trace condition settings screen. (☞ Section 6.14(4)(b))
N1: When "Time" is set, apply "4".
N2: When "Step no." is set, apply "10".
N3: When "Program name" is set, apply "8".</td></tr>
<tr><td>Error history data</td><td>72 + 54 × (number of failures stored)</td></tr>
</table>

(To the next page)

Table5.12 Calculation of file size (High Performance model QCPU, Process CPU, and Redundant CPU) (continued)

Function	Rough file capacity (unit: Byte)
Local device	72 + 6 × (set device type) + (2 × ((total number of M and V points)/16 + (number of D points) + 18 × (total number of T, ST, and C points)/16)) × (number of programs) • M, V, D, T, ST, and C indicate the following set devices. M: internal relay V: edge relay D: data register T: timer ST: retentive timer C: counter

*1: 136 is set by default (This value depends on the parameter setting).

*2: After the decimal point of a value found by the number of bit device points/16 is rounded up.

Remark

For calculation example of memory capacity, refer to Section 5.4.4.

5.4.4 Units of file sizes

(1) Definition

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 shows the file size unit depending on the CPU module and memory area to be written.

Table5.13 File size unit for each CPU module and memory area

CPU module model	Memory area	
	Program memory, standard ROM, Flash card*1	Standard RAM
Q00JCPU	1 step/4 bytes	-
Q00CPU, Q01CPU		4 bytes
Q02CPU, Q02HCPU, Q06HCPU	128 steps/512 bytes*2	512 bytes
Q12HCPU	256 steps/1024 bytes*2	1024 bytes*3
Q25HCPU	512 steps/2048 bytes*2	
Q02PHCPU, Q06PHCPU	128 steps/512 bytes	512 bytes
Q12PHCPU	256 steps/1024 bytes*4	1024 bytes
Q25PHCPU	512 steps/2048 bytes*4	
Q12PRHCPU	256 steps/1024 bytes	
Q25PRHCPU	512 steps/2048 bytes	

*1: The file size unit of the Flash card is applied when a file is written to the Flash card by [Online] → [Write the program memory to ROM] in GX Developer. (☞ Section 5.2.6)

*2: For the High Performance model QCPU, if the serial number (first five digits) is "04121" or earlier, the file size unit will be 1024 steps/4096 bytes.

*3: For the Q12HCPU or Q25HCPU, if the serial number (first five digits) is "02091" or earlier, the file size unit will be 512 bytes.

*4: For the Process CPU, if the serial number (first five digits) is "07031" or earlier, the file size unit will be 1024 steps/4096 bytes.

(b) File size unit for each memory card

Table5.14 File size unit for each memory card

Type	Memory card model	File size unit (cluster size)
SRAM card	Q2MEM-1MBSN, Q2MEM-1MBS	512 bytes
	Q2MEM-2MBSN, Q2MEM-2MBS	1024 bytes
Flash card*1	Q2MEM-2MBF	1024 bytes
	Q2MEM-4MBF	1024 bytes
ATA card	Q2MEM-8MBA	4096 bytes
	Q2MEM-16MBA	4096 bytes
	Q2MEM-32MBA	2048 bytes

*1: The file size unit of the Flash card is applied when:
- a file is written to the Flash card by [Online] → [Write to PLC (Flash ROM)] in GX Developer, or (☞ Section 5.2.6(1)(b))
- a file is written to the Flash card by GX Developer without accessing the CPU module. (☞ Section 5.2.6)

(2) Calculation example of memory capacity

The following shows an calculation example of memory capacity when the parameters and sequence program are written to the program memory.

(a) Conditions

1) CPU module to be written: Q25HCPU

2) Writing file

Table5.15 File sizes

File name	File size[*1]
PARAM.QPA (parameter file)	564 bytes
MAIN.QPG (sequence program)	525 steps/2100 bytes[*2]

*1: For the file size, refer to Section 5.4.3.

*2: This indicates the program size (file header + execution program) displayed on the GX Developer screen. (☞ Section 5.3)

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

(b) 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. (☞ (1) in this section)

1) Calculation of parameter file size

Since files are stored in units of file sizes to the program memory, the parameter file size is 564 bytes. However, the file occupies 512 steps/2048 bytes.

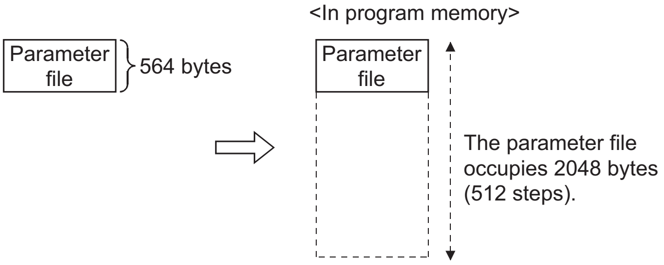


Figure 5.41 Occupation in units of the file size (parameter file)

2) Calculation of program size

The program size is found by the formula: sequence program size + reserved area for online change. Since files are stored in units of file sizes to the program memory, the parameter file size is 525 steps + 500 steps = 1025 steps. However, the file occupies 1536 steps/6144 bytes.

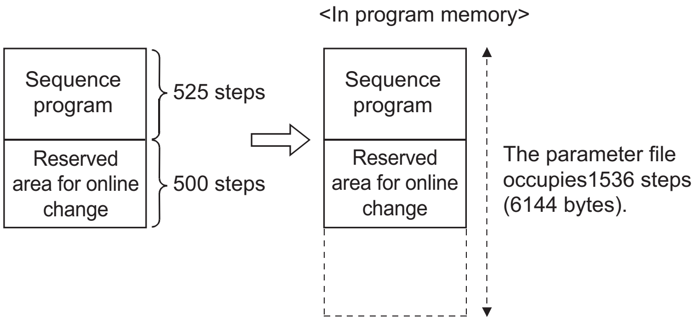

<In program memory>
Sequence program
Reserved area for online change
525 steps
500 steps
Sequence program
Reserved area for online change
The parameter file occupies1536 steps (6144 bytes).

Figure 5.42 Occupation in units of the file size (program file)

3) Result

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

Table5.16 Calculation results of memory capacities

File name	File size		Memory capacity
PARAM.QPA	564 bytes		512 steps (2048 bytes)
MAIN.QPG	Sequence program size	525 steps	1536 steps (6144 bytes)
	Reserved area for online change	500 steps	
	Total	1025 steps	
Total memory capacity			2048 steps (8192 bytes)

Point

The file size unit of the following CPU modules has been changed.

- High Performance model QCPU having the serial number (first five digits) "04122" or later
- Process CPU having the serial number (first five digits) "07032" or later

Note the following.

1) Due to difference of file capacity, a file operated by the CPU module having the serial number equal to or later than the one mentioned above may not be stored to the CPU module having the serial number earlier than the one mentioned above.

2) The following table shows restrictions depending on combination of the CPU module and GX Developer versions when a file read from the CPU module to GX Developer is written to another CPU module.

Table5.17 High Performance model QCPU

CPU module to be written	GX Developer that writes a file			
	GX Developer Version 8		GX Developer Version 7	
	File operated by the CPU module having the serial number “04122” or later	File operated by the CPU module having the serial number “04121” or earlier	File operated by the CPU module having the serial number “04122” or later	File operated by the CPU module having the serial number “04121” or earlier
Serial number "04122" or later	○	○	△ *2	△ *2
Serial number "04121" or earlier	△ *1	○	△ *1 *2	△ *2

○: Can be performed. △: Can be performed with restrictions.

*1: Since the file size unit differs, a file may not be stored to the CPU module depending on the file size.
*2: Unless the number of steps of the reserved area for online change is reduced, a file may not be stored to the CPU module depending on the file size.

Table5.18 Process CPU

CPU module to be written	GX Developer that writes a file			
	GX Developer Version 8		GX Developer Version 7	
	File operated by the CPU module having the serial number “07032” or later	File operated by the CPU module having the serial number “07031” or earlier	File operated by the CPU module having the serial number “07032” or later	File operated by the CPU module having the serial number “07031” or earlier
Serial number "07032" or later	○	○	△ *2	△ *2
Serial number "07031" or earlier	△ *1	○	△ *1 *2	△ *2

○: Can be performed. △: Can be performed with restrictions.

*1: Since the file size unit differs, a file may not be stored to the CPU module depending on the file size.
*2: Unless the number of steps of the reserved area for online change is reduced, a file may not be stored to the CPU module depending on the file size.

CHAPTER6 FUNCTIONS

This chapter describes the functions of the CPU module.

6.1 Function List

Table6.2 lists the functions of the CPU module.
Each number in the "CPU module" column corresponds to the CPU module listed in Table6.1.

Table6.1 Number in the column and the corresponding CPU module

Number	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU

Table6.2 Function list

Item	Description	CPU module				Reference
		1)	2)	3)	4)	
Constant scan	Executes a program in a set time interval regardless of its scan time.	○	○	○	○	Section 6.2
Latch function	Holds the device data even at power OFF or reset.	○	○	○	○	Section 6.3
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 CPU module status is switched from STOP to RUN.	○	○	○	○	Section 6.4
Clock function	Reads the internal clock data of the CPU module to use it for time management.	○	○	○	○	Section 6.5
Remote RUN/STOP	Runs or stops the program operations in the CPU module externally.	○	○	○	○	Section 6.6.1
Remote PAUSE	Stops the program operations in the CPU module externally, holding the status of outputs (Y).	○	○	○	○	Section 6.6.2
Remote RESET	Resets the CPU module externally when the CPU module is in a STOP status.	○	○	○	○	Section 6.6.3
Remote latch clear	Clears the latch data in the CPU module when the CPU module is in a STOP status.	○	○	○	○	Section 6.6.4
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.	○	○	○	○	Section 6.7
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.	○	○	○	○	Section 6.8
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.	○	○	○	○	Section 6.9

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

(To the next page)

Table6.2 Function list (continued)

Item	Description	CPU module				Reference
		1)	2)	3)	4)	
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.)	○	○	○	○	Section 6.10
Monitor function	Reads the status of programs and devices in the CPU module by GX Developer.	○	○	○	○	Section 6.11
Monitor condition setting	Specifies the monitoring timing of the CPU module with device condition or step number.	×	○	○	○	Section 6.11.1
Local device monitor/test	Monitors and/or tests the local devices of the specified program by GX Developer.	×	○	○	○	Section 6.11.2
External input/output forced on/off	Forcibly turns on/off the external input/output of the CPU module by GX Developer.	×	△*4	○	○	Section 6.11.3
Online change	Changes a device value within the specified step of a sequence program.	△*5	○	○	○	Section 6.12
Program monitor list	Displays the scan time and execution status of the program being executed.	○	○	○	○	Section 6.13.1
Interrupt program monitor list	Displays the number of executions of interrupt programs.	○	○	○	○	Section 6.13.2
Scan time measurement	Measures the execution time of the area specified by the steps in a program.	×	○	○	○	Section 6.13.3
Sampling trace function	Continuously samples the specified device data at a preset timing.	×	○	○	○	Section 6.14
Debug function from multiple GX Developers	Enables simultaneous debugging by multiple GX Developers.	○	○	○	○	Section 6.15
Watchdog timer	Monitors operational delays caused by hardware failure or program error of the CPU module.	○	○	○	○	Section 6.16
Self-diagnostic function	Self-diagnoses the CPU module to see whether an error exists or not.	○	○	○	○	Section 6.17
Error history	Stores the result of self-diagnostics to the memory as error history data.	○	○	○	○	Section 6.18
System protection	Prevents the programs from being modified from GX Developer, serial communication module, and Ethernet module.	△*1	○	○	○	Section 6.19
Password registration	Prohibits writing/reading data to/from each file in the CPU module using GX Developer.	○	○	○	○	Section 6.19.1
Remote password	Prevents unauthorized access from external devices such as serial communication module and Ethernet module.	△*4	△*4	○	○	Section 6.19.2
System display	Monitors the system configuration using GX Developer.	○	○	○	○	Section 6.20
LED indication	Displays the operating status of the CPU module with LEDs on the front of the module.	○	○	○	○	Section 6.21
LED indication priority	Sets the priority of error messages stored in the LED display data (SD220 to SD227) in case of an error. LED indication can be set to non-display.	○	○	○	○	Section 6.21.2
High-speed interrupt function	Executes an interrupt program at fixed intervals of 0.2ms to 1.0ms using the interrupt pointer (I49).	×	△*3	×	×	Section 6.22
Interrupt from intelligent function module	Executes an interrupt program at the time of interrupt request from the intelligent function module.	△*4	○	○	○	Section 6.23

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

(To the next page)

Table6.2 Function list (continued)

Item	Description	CPU module				Reference
		1)	2)	3)	4)	
Serial communication function	Connects the RS-232 interface of the CPU module and the personnel computer or HMI with RS-232 cable and communicates in the MC protocol.	△[2]	×	×	×	Section 6.24
Module service interval time read	Monitors the service interval time (time from service acceptance to next service acceptance) of the intelligent function module, network module, or GX Developer.	○	○	○	○	Section 6.25.1
Initial device value	Registers data used in programs with devices and the buffer memories of the intelligent function modules and special function modules without programs.	△[4]	○	○	○	Section 6.26
Memory check function	Checks whether the data in memories of the CPU module are changed or not due to such as excessive electric noise.	×	×	△[4]	○	Section 6.27
Online module change	Changes the Q series input/output modules or intelligent function modules of function version C mounted on the main base unit, extension base unit, or MELSECNET/H remote I/O station. In a redundant power supply system, the power supply module can also be changed online.	×	×	○	○	QCPU User's Manual (Hardware Design, Maintenance and Inspection)
Auto tuning function	Makes initial settings of PID constants. This function can be used for the PID control that uses the S.PID or S.2PID instruction (for example, a process with relatively slow response such as temperature control).	×	×	○	○	QnPHCPU/ QnPRHCPU Programming Manual (Process Control Instructions)
Redundant system function	Configures a redundant system with CPU modules, power supply modules, network modules, and main base units.	×	×	×	○	QnPRHCPU User's Manual (Redundant System)
System switching function (switching between the control system and standby system)	Switches the systems between the control system and standby system. The systems can be switched either by the system or by the user.	×	×	×	○	
Operation mode change	Switches the operation mode between the separate mode and backup mode.	×	×	×	○	
Tracking function	Has the standby system CPU module share the same data with the control system CPU module (transfers data in the control system CPU module to the standby system CPU module). Control can be continued with the same data even after the systems are switched due to a failure or error in the control system.	×	×	×	○	
Online change for redundancy	Transfers data written to the control system CPU module by the write to PLC or online change operations to the standby system CPU module.	×	×	×	○	

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

[1] The Basic model QCPU does not support the system protection with DIP switches.
[2] The Q00JCPU does not support this function.
[3] The Q02CPU does not support this function.
[4] Availability depends on the version of the CPU module. (☞ Appendix 2)
[5] Online change (files) is not supported.

6.2 Constant Scan

(1) Definition

Scan time of the CPU module is not constant because the processing time varies depending on the execution status of instructions used in a sequence program.

This function allows sequence programs to be executed repeatedly, maintaining its scan time constant.

(2) 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.

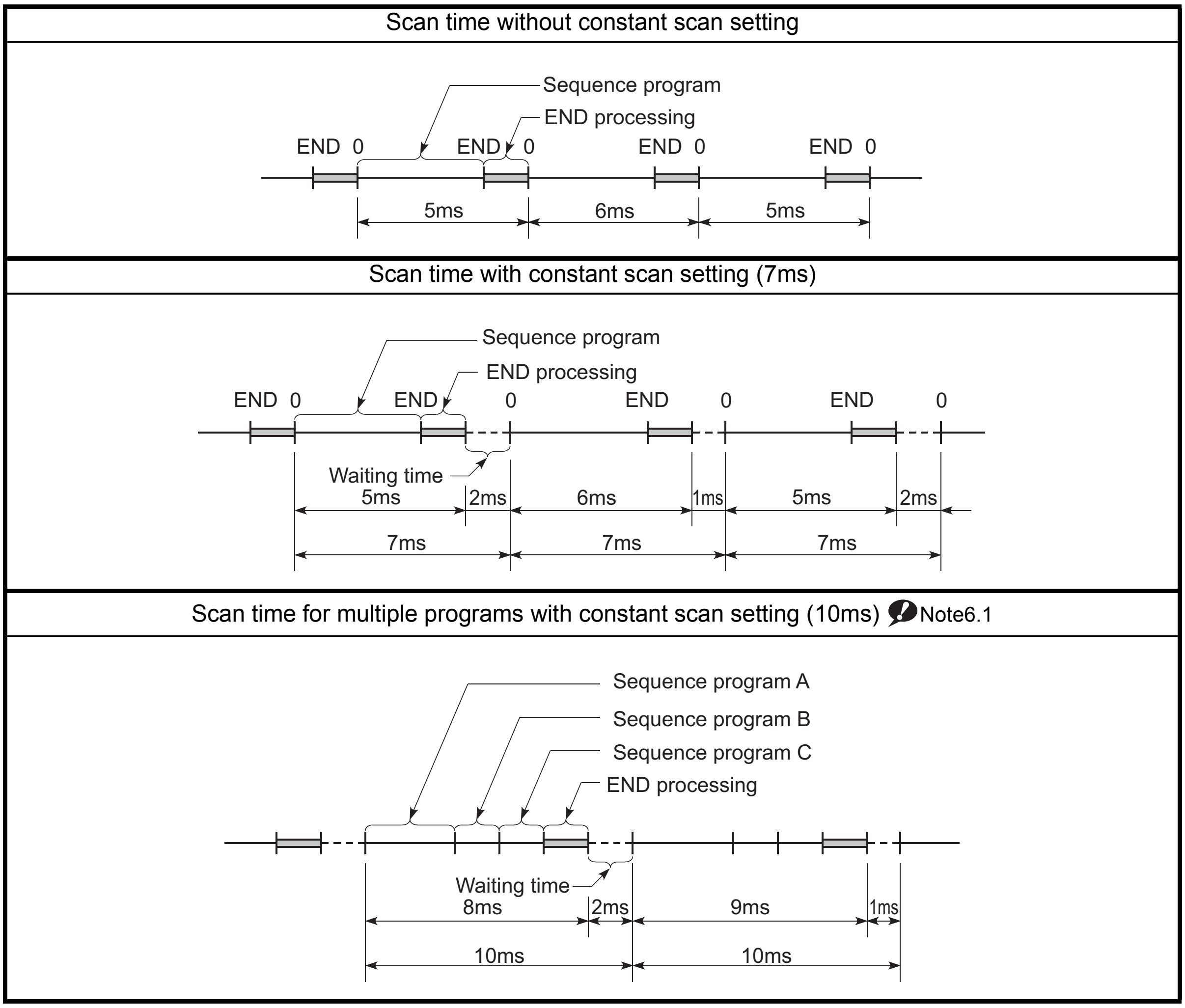


Figure 6.1 Constant scan operation

Note6.1 **Basic**

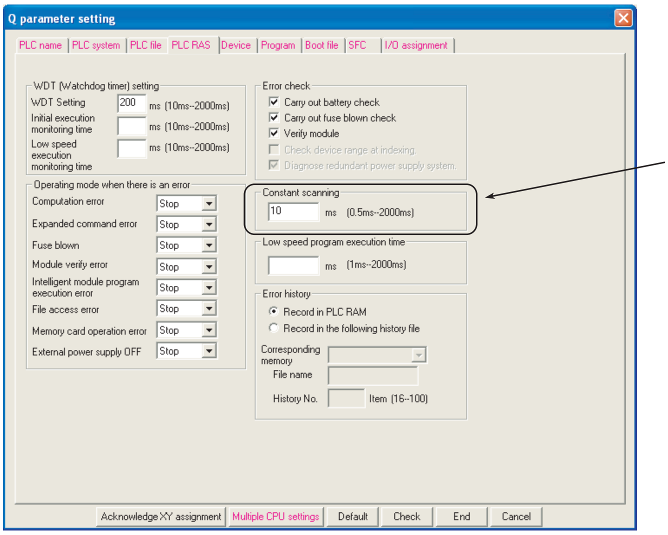
Since the Basic model QCPU cannot execute multiple programs, it is not necessary to be conscious of the scan time taken for execution of multiple programs.

(3) Constant scan time setting

Set a constant scan time value in the PLC RAS tab of the PLC parameter dialog box.
The setting range varies depending on the CPU module.

- Basic model QCPU: 1 to 2000ms (in increments of 1ms)
- High Performance model QCPU, Process CPU, or Redundant CPU:
  0.5 to 2000ms (in increments of 0.5ms)

When not executing the constant scan function, leave the constant scan time setting box blank.



WDT (Watchdog timer) setting
WDT Setting 200 ms (10ms--2000ms)
Initial execution monitoring time ms (10ms--2000ms)
Low speed execution monitoring time ms (10ms--2000ms)

Operating mode when there is an error
Computation error Stop
Expanded command error Stop
Fuse blown Stop
Module verify error Stop
Intelligent module program execution error Stop
File access error Stop
Memory card operation error Stop
External power supply OFF Stop

Error check
☑ Carry out battery check
☑ Carry out fuse blown check
☑ Verify module
□ Check device range at indexing
☑ Diagnose redundant power supply system

Constant scanning
10 ms (0.5ms--2000ms)

Low speed program execution time
ms (1ms--2000ms)

Error history
● Record in PLC RAM
○ Record in the following history file
Corresponding memory
File name
History No. Item (16--100)

Enter a value.

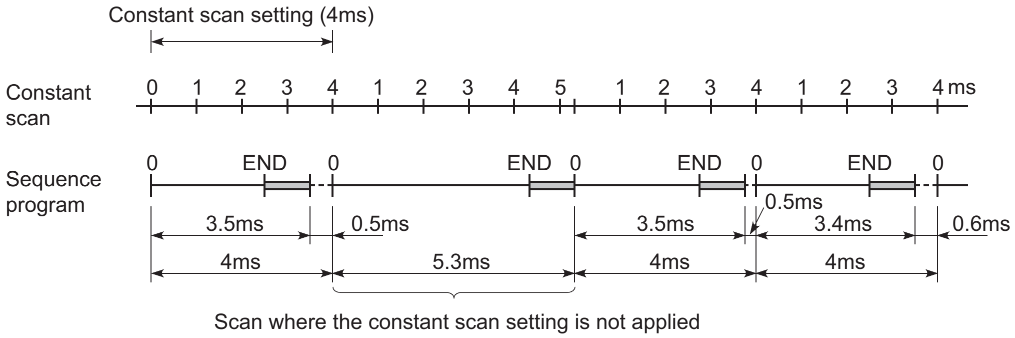
Figure 6.2 When the constant scan time is set to 10ms

(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.



Constant scan setting (4ms)
Constant scan
0 1 2 3 4 1 2 3 4 5 1 2 3 4 1 2 3 4 ms
Sequence program
0
END
0
END
0
END
0
END
0
3.5ms
0.5ms
3.5ms
0.5ms
3.4ms
0.6ms
4ms
5.3ms
4ms
4ms
Scan where the constant scan setting is not applied

Figure 6.3 Operation when the scan time is longer than the constant scan setting time

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.

(4) 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 a low-speed execution type program is executed

Execution of the low-speed execution type program will be stopped when the time reaches the following point.

(Constant scan) - 0.5ms

(b) When an interrupt factor occurs during waiting time

Either of the following programs is executed.

- Interrupt program
- Fixed scan execution type program

(5) Constant scan accuracy

Table6.3 shows the constant scan accuracy.

Table6.3 Constant scan accuracy

CPU module	Without monitor, without user interrupt	With monitor, without user interrupt	Without monitor, with user interrupt	With monitor, with user interrupt
Q00JCPU	0.20ms	0.90ms	Interrupt program execution time (refer to Section 10.1.2(4)(a))	Total of the following: 1) Time given under the "With monitor, without user interrupt" column 2)Total of interrupt program execution times
Q00CPU	0.12ms	0.60ms		
Q01CPU	0.10ms	0.50ms		
Q02CPU	0.02ms			
High Performance model QCPU, Process CPU, Redundant CPU	0.01ms			

- With monitor: Indicates the status where monitor operation is performed by GX Developer connected or communication with an external device is performed using the serial communication function.
- Without monitor: Indicates the status where no operation by GX Developer or no communication using the serial communication function is performed.

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

(a) When low-speed execution type program is executed

There is a waiting time of 0.5ms.

- When maximum processing time of one instruction is within 0.5ms
  A constant scan error will be the same as the constant scan accuracy shown in Table6.3.
- When maximum processing time of one instruction exceeds 0.5ms
  The constant scan time may increase by the time exceeding 0.5ms.
  Also, the CPU module detects "PRG, TIME OVER" (error code: 5010).

For the maximum processing time of one instruction, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

(b) 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.

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

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

6.3 Latch Function

(1) Definition

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 each device of the CPU module is cleared and set back to its default (bit device: off, word device: 0) without using the latch function.

(2) 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.

(3) Program operation when the latch function is used

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

(4) Devices that can be latched

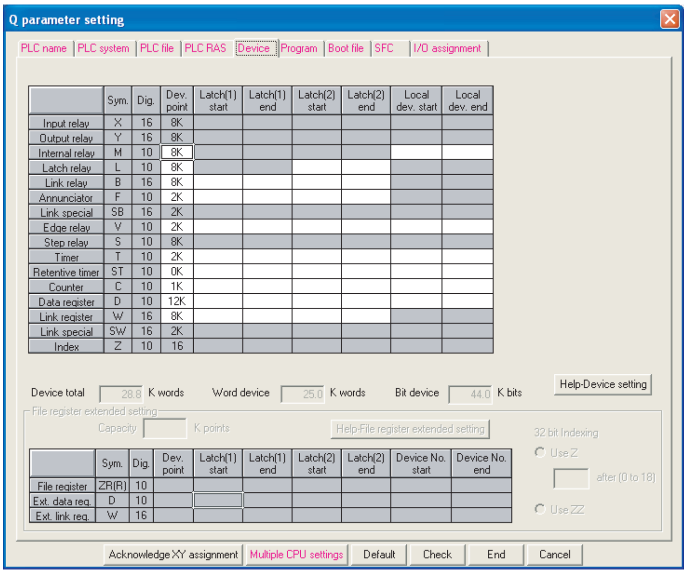
The following devices can be latched.
(By default, only the latch relay is enabled for latch.)

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

(5) Latch range setting

Set a latch range in the Device tab of the PLC parameter dialog box.

There are two types of latch range settings: the latch clear operation enable range setting (Latch (1)) and the latch clear operation disable range setting (Latch (2)).



	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Local dev. start	Local dev. end
Input relay	X	16	8K						
Output relay	Y	16	8K						
Internal relay	M	10	8K						
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	12K						
Link register	W	16	8K						
Link special	SW	16	2K						
Index	Z	10	16						

Device total 28.8 K words   Word device 25.0 K words   Bit device 44.0 K bits

	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Device No. start	Device No. end
File register	ZR(R)	10							
Ext. data reg.	D	10							
Ext. link reg.	W	16							

Figure 6.4 Latch range setting

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

Data will be latched at the same time when the data is written to a device in the latch range.

There is no influence on the scan time since no latch processing is performed.

(7) Device data latch clear

Table6.4 shows the status of device data when the latch clear operation is performed.

Table6.4 Status 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 Section 3.7.

Point

Data in the file register (R or ZR) will not be cleared by the latch clear operation.

To clear data in the file register (R or ZR), perform data clear operation by a sequence program or GX Developer. (☞ Section 9.7.6(3))

6

6.3 Latch Function

(8) Precautions

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

Device data cannot be latched even if the device has been latch-specified.

(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.

(c) When the start mode is set to the hot-mode in the Redundant CPU

Device data outside the latch range are held as well. (Except some devices, such as the step relay and the index register.)

To clear these data, perform latch clear operation. (☞ Section 3.7(2)(b))

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

(1) Definition

When the operating status is changed from RUN to STOP, the CPU module internally stores the outputs (Y) in the RUN status and then turns off all the outputs (Y).

The status of the outputs (Y) when the operating status of the CPU module is changed back from STOP to RUN can be selected from the following two options in the parameter setting in GX Developer.

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

(2) 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.

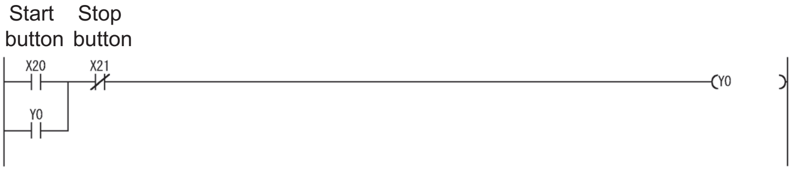


Figure 6.5 Holding circuit

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

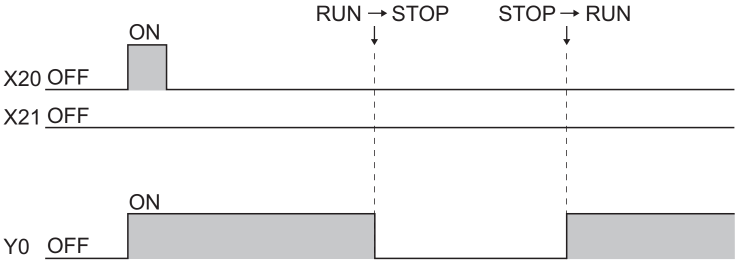


Figure 6.6 Timing chart when the parameter is set to "Previous state"

- When clearing the output (Y) status

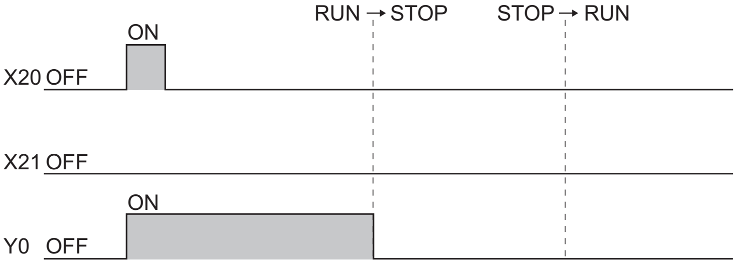


Figure 6.7 Timing chart when the parameter is set to "Recalculate (output is 1 scan later)"

(3) 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 changing 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 (5) in this section.

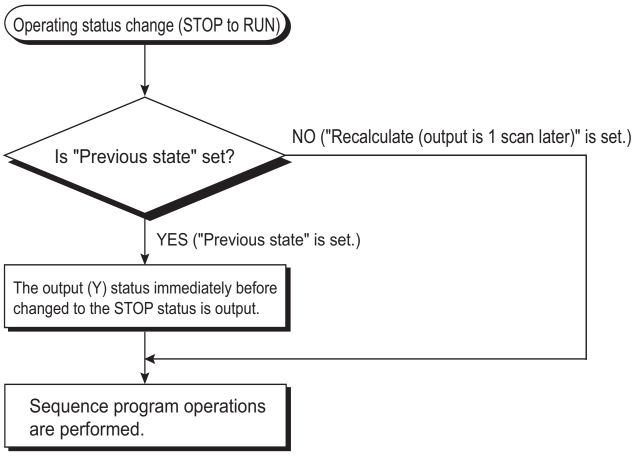
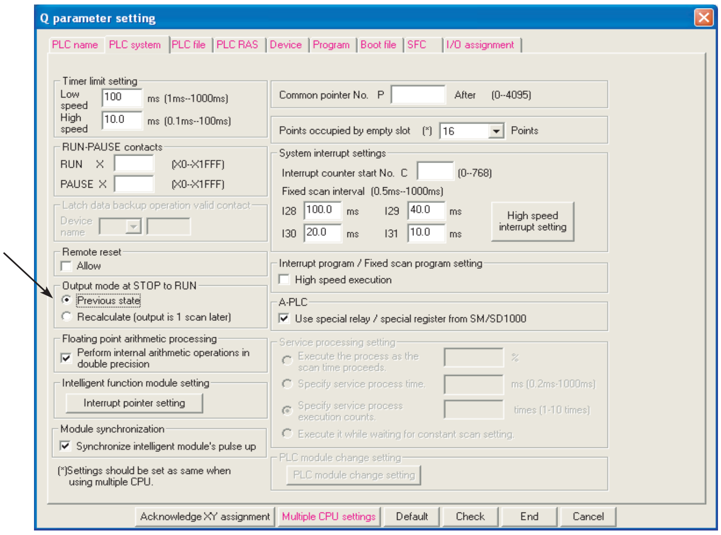

Operating status change (STOP to RUN)
Is "Previous state" set?
NO ("Recalculate (output is 1 scan later)" is set.)
YES ("Previous state" is set.)
The output (Y) status immediately before changed to the STOP status is output.
Sequence program operations are performed.

Figure 6.8 Operation when the operating status is changed from STOP to RUN

(4) 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.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
□ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms

Interrupt program / Fixed scan program setting
□ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting

Set the output mode here.

Figure 6.9 PLC system setting

(5) Precautions

Table6.5 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.

Table6.5 Output status when the operating status is changed from STOP to RUN after the output forced on operation is performed

Output mode ("Output mode at STOP to RUN") selected	Output status
Previous state	The output status prior to STOP is output. Even if the outputs are forcibly turned on, 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.

6.5 Clock Function

(1) Definition

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.

(2) 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.

(3) Clock data

Table6.6 shows the details of clock data, which is used internally in the CPU module.

Table6.6 Clock data details

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: Use only the expansion clock data read (S(P).DATERD) to read.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

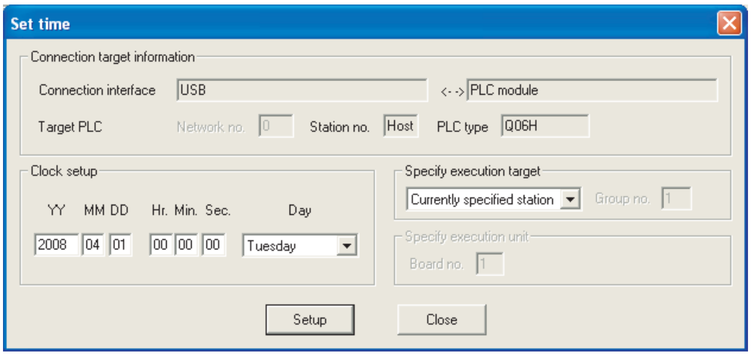
(4) Changing and reading clock data

(a) Changing clock data

Clock data can be changed either by GX Developer or a program.

1) Changing clock data by GX Developer

Select [Online] → [Set time] to open the Set time screen and change the clock data.



Set time

Connection target information

Connection interface: USB <-> PLC module

Target PLC: Network no. 0 Station no. Host PLC type Q06H

Clock setup

YY	MM	DD	Hr.	Min.	Sec.	Day
2008	04	01	00	00	00	Tuesday

Specify execution target: Currently specified station Group no. 1

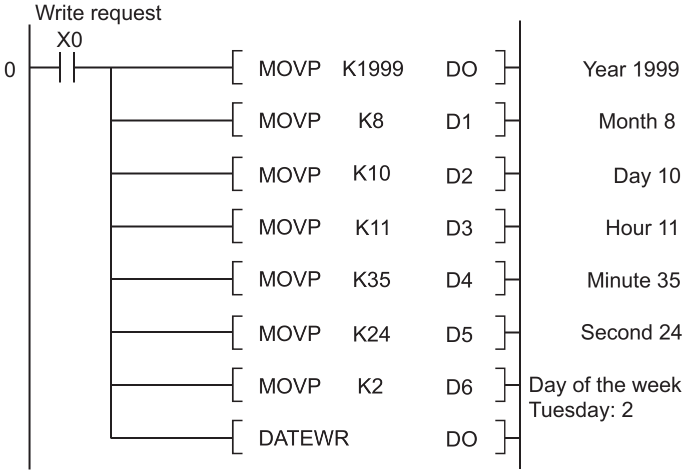
Specify execution unit: Board no. 1

Figure 6.10 Set time screen

2) Changing clock data by a program

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

Figure 6.11 shows a program for writing the set clock data to D0 to D6.



```
Write request
X0
0  ─┤├─┬─[ MOVP  K1999  DO ]   Year 1999
       ├─[ MOVP  K8     D1 ]   Month 8
       ├─[ MOVP  K10    D2 ]   Day 10
       ├─[ MOVP  K11    D3 ]   Hour 11
       ├─[ MOVP  K35    D4 ]   Minute 35
       ├─[ MOVP  K24    D5 ]   Second 24
       ├─[ MOVP  K2     D6 ]   Day of the week
       │                       Tuesday: 2
       └─[ DATEWR       DO ]
```

Figure 6.11 Program example for writing clock data

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

☞ MELSEC Communication Protocol Reference Manual

Point

- When clock data is changed, the clock of 1/1000 second is reset to 0.
- Year data settable by GX Developer 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)

Figure 6.12 shows a program for storing the clock data read with the DATERD instruction to D10 to D16.

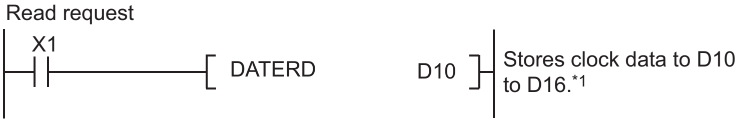


Figure 6.12 Program example for storing clock data

*1: Figure 6.13 shows the clock data stored in D10 to D16.

D10	2004	Year (four digits)
D11	4	Month
D12	1	Date
D13	11	Hour
D14	35	Minute
D15	24	Second
D16	2	Day of the week

(☞ (3) in this section)

Figure 6.13 Clock data stored

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)

(5) 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 storage, or for intelligent function modules.

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

(b) Clock data correction

If a part of the clock data is corrected, rewrite the entire clock data to the CPU module.

(c) Clock data setting range

When changing clock data, write data within the range given in (3) in this section.

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 out of the range.

Table6.7 Example of clock data

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

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

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

Only the following instructions can use the clock data of 1/1000 sec.

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

Other instructions cannot use the clock data of 1/1000 sec.

(Such as for reading data by SM/SD, storing the time of error occurrence as error history data, reading data by GX Developer, or reading data by dedicated instructions of other modules.)

2) When clock data is changed

When clock data is changed by GX Developer or instructions (including dedicated instruction of other modules), the clock of 1/1000 sec. is reset to 0.

(6) Clock data accuracy

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

Table6.8 Clock data accuracy

Ambient temperature (℃)	Accuracy (Day difference, S)		
	Basic model QCPU	High Performance model QCPU, Process CPU	Redundant CPU
0	-3.2 to +5.27 (TYP.+1.98)	-3.18 to +5.25 (TYP.+2.12)	-3.2 to +5.27 (TYP.+2.07)
+25	-2.57 to +5.27 (TYP.+2.22)	-3.93 to +5.25 (TYP.+1.9)	2.77 to +5.27 (TYP.+2.22)
+55	-11.68 +3.65 (TYP.-2.64)	-14.69 to +3.53 (TYP.-3.67)	-12.14 to +3.65 (TYP.-2.89)

(7) 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.

6.6 Remote Operation

Remote operation allows to change the operating status of the CPU module externally (by GX Developer or external devices using the MC protocol, with link dedicated instructions of the CC-Link IE Controller Network module or MELSECNET/H module, or using remote contacts).

There are four types of remote operations:

• Remote RUN/STOP	:	☞ Section 6.6.1
• Remote PAUSE	:	☞ Section 6.6.2
• Remote RESET	:	☞ Section 6.6.3
• Remote latch clear	:	☞ Section 6.6.4

6.6.1 Remote RUN/STOP

(1) Definition

This operation changes the operating status of the CPU module externally to RUN or STOP, keeping the RUN/STOP switch of the CPU module (the RUN/STOP/RESET switch for the Basic model QCPU) in the RUN position.

(2) 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.

(3) 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. (The remote RUN operation must be performed to the CPU module whose operating status has been changed to STOP by the remote STOP operation.)

(4) Executing method

There are three methods for performing the remote RUN/STOP operation.

- Using a RUN contact
- By GX Developer or an external device using the MC protocol
- With link dedicated instructions of the CC-Link IE Controller Network module or MELSECNET/H module

(a) Using a RUN contact

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

The settable device range differs depending on the CPU module.

- Basic model QCPU: Input X0 to X7FF
- High Performance model QCPU, Process CPU, or Redundant CPU: Input X0 to X1FFF

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.

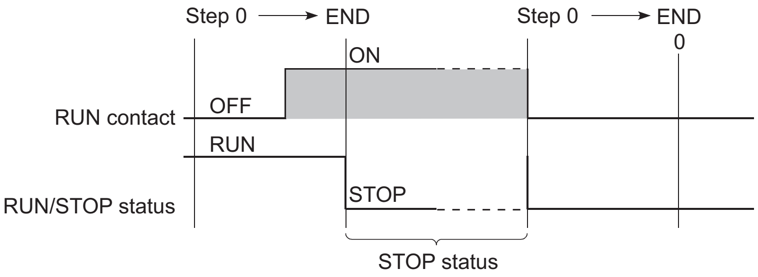


Figure 6.14 Remote RUN/STOP using a RUN contact

(b) By GX Developer or an external device using the MC protocol

Select [Online] → [Remote operation] in GX Developer.

To perform the remote RUN/STOP operation from an external device, use the MC protocol command.

☞ MELSEC Communication Protocol Reference Manual

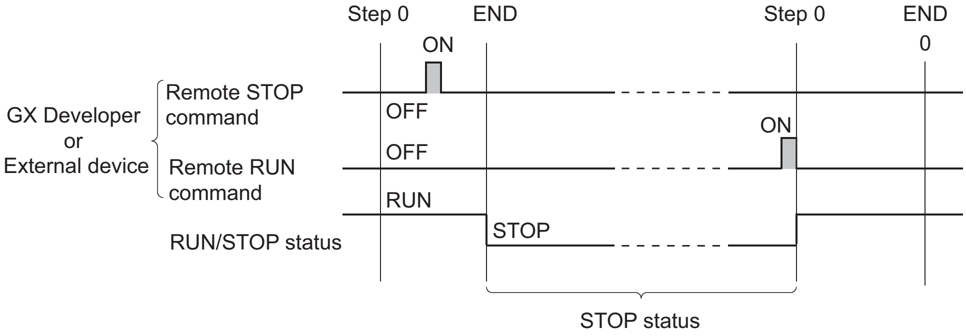


Figure 6.15 Remote RUN/STOP by GX Developer or an external device

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

The remote RUN/STOP operation by link dedicated instructions of the CC-Link IE Controller Network 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

(5) Precautions

Pay attention to the following since the STOP status is given priority over the RUN 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, GX Developer, 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.

(c) In a redundant system where the Redundant CPU is used

1) When the "Both systems" item is not selected on the Remote operation screen

Remote RUN or STOP operation is performed for only the CPU module in the system specified in the connection target settings.

2) When the "Both systems" item is selected on the Remote operation screen

Remote RUN or STOP operation is performed for both CPU modules in the control system and standby system.
Make sure to set the Redundant CPUs in both systems in backup mode.

Point

- The definitions of the RUN/STOP status are described below.
  - RUN status•••A status where sequence program operations are repeatedly performed in a loop between the step 0 and the END or FEND instruction.
  - STOP status•••A status where sequence program operations are stopped. All outputs (Y) are turned off.
- After being reset, the operating status of the CPU module becomes the one set with the RUN/STOP switch (the RUN/STOP/RESET switch for the Basic model QCPU).

6.6.2 Remote PAUSE

(1) Definition

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

(2) 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.

(3) Executing method

There are two methods for performing the remote PAUSE operation.

- Using a PAUSE contact
- By GX Developer or an external device using the MC protocol

(a) Using a PAUSE contact

Set a PAUSE contact in the PLC system tab of the PLC parameter dialog box.
The settable device range differs depending on the CPU module.

- Basic model QCPU: Input X0 to X7FF
- High Performance model QCPU, Process CPU, or Redundant CPU: Input X0 to X1FFF

1) 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.

2) 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.

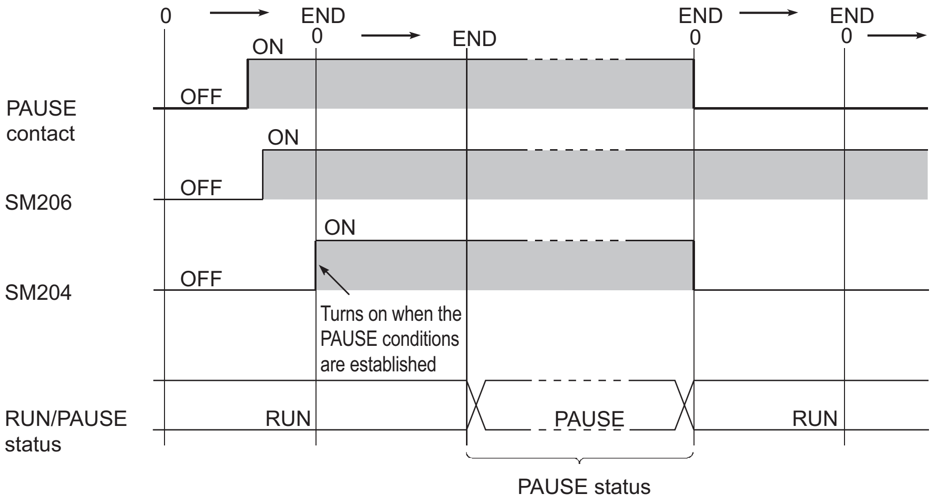


Figure 6.16 Remote PAUSE using a PAUSE contact

Point

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

(b) By GX Developer or an external device using the MC protocol

Select [Online] → [Remote operation] in GX Developer.

To perform the remote PAUSE operation from an external device, use the MC protocol command.

MELSEC Communication Protocol Reference Manual

1) 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.

2) Upon execution of the remote RUN command, the CPU module will restart the sequence program operations from the step 0.

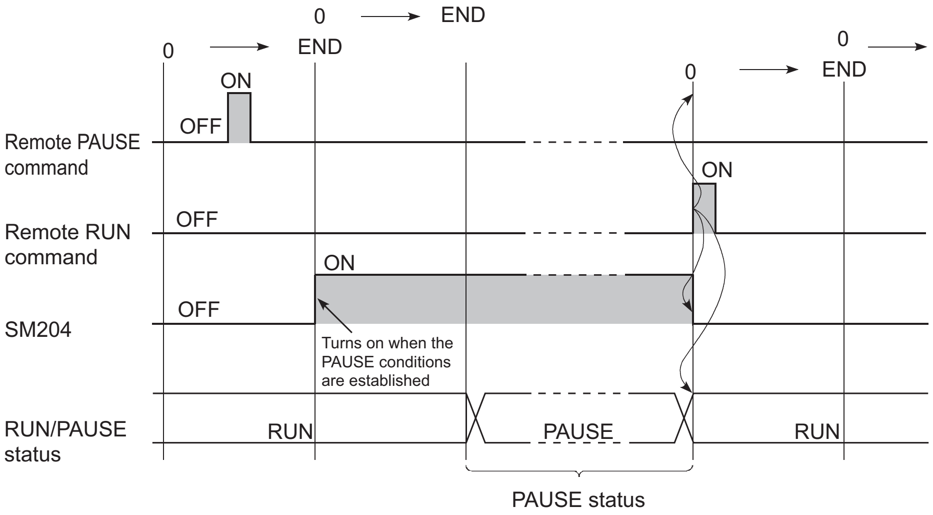


Figure 6.17 Remote PAUSE by GX Developer or an external device

(4) Precautions

(a) When forcibly keeping output status

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

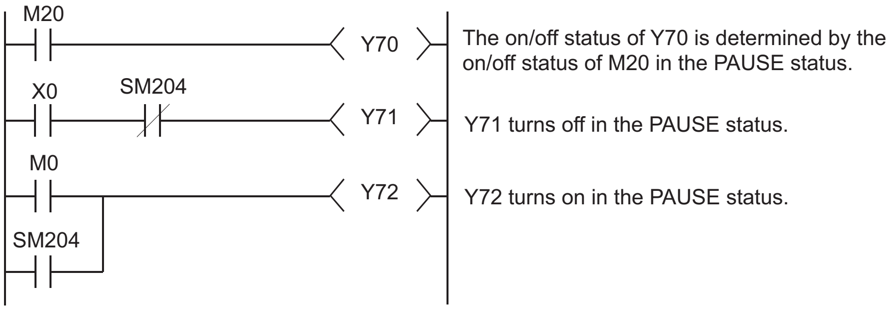

M20
Y70
The on/off status of Y70 is determined by the on/off status of M20 in the PAUSE status.
X0
SM204
Y71
Y71 turns off in the PAUSE status.
M0
Y72
Y72 turns on in the PAUSE status.
SM204

Figure 6.18 Program example for forcibly keeping output status in the PAUSE status

(b) In a redundant system where the Redundant CPU is used

1) When the "Both systems" item is not selected on the Remote operation screen

Remote PAUSE operation is performed for only the CPU module in the system specified in the connection target settings.

2) When the "Both systems" item is selected on the Remote operation screen

Remote PAUSE operation is performed for both CPU modules in the control system and standby system. Make sure to set the Redundant CPUs in both systems in backup mode.

6.6.3 Remote RESET

(1) Definition

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

(2) Application

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

(3) Executing method

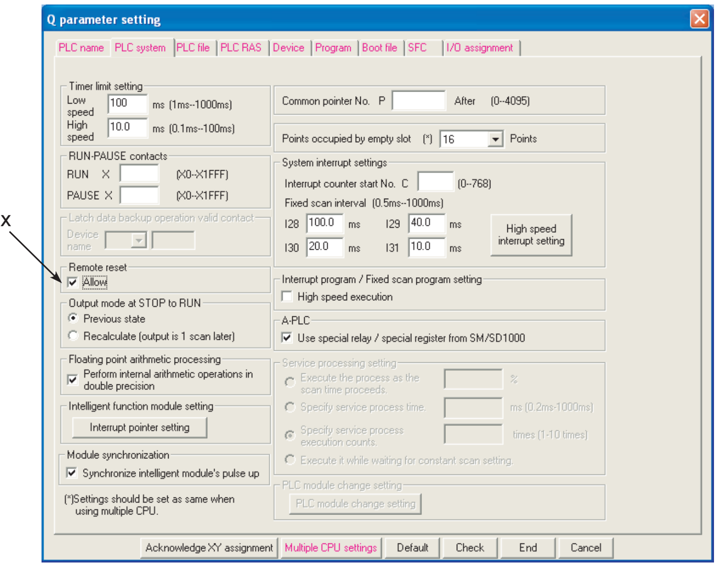
This operation can only be performed by GX Developer or an external device using the MC protocol.

- Select [Online] → [Remote operation] in GX Developer.
- To perform the remote RESET operation from an external device, use the MC protocol command.

☞ 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 then write the parameter setting to the CPU module.
Without the preset parameter setting, the operation cannot be performed.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
☑ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms

Interrupt program / Fixed scan program setting
□ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting

Select the checkbox here to enable the remote RESET operation.

Figure 6.19 Parameter setting required for the remote RESET operation

(4) 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 or similar operation.

(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 switch (the RUN/STOP/RESET switch for the Basic model QCPU).

If the 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 GX Developer, check that the "Allow" checkbox for the remote RESET operation in the PLC system tab of the PLC parameter dialog box is selected.
  If the checkbox is not selected, the CPU module cannot be reset even after the remote RESET processing from GX Developer is completed.

(c) When an error occurs due to noise

Note that the CPU module may not be reset by the remote RESET operation from GX Developer.
In this case, reset the CPU module using the RUN/STOP switch (the RUN/STOP/RESET switch for the Basic model QCPU) or power off and then on the CPU module.

(d) In a redundant system where the Redundant CPU is used

Remote RESET operation is performed for the control system CPU module in backup mode. As a result, the CPU modules in both systems are reset.

When the CPU module is in separate mode or debug mode, the operation is performed for only the CPU module in the system specified in the connection target settings.

1) When the Redundant CPUs are in backup mode

Perform the remote RESET operation for the control system CPU module. (The CPU modules in both systems are reset.)

If the operation is performed for the standby system CPU module, an error (error code: 4240H) occurs.

2) When RUN/STOP status differs between the CPU modules (in backup mode) in two systems

When performing the remote RESET operation, if the control system CPU module is in the STOP status, set the standby system CPU module to the same status.

If the operation is performed with the control system CPU module in the STOP status and the standby system CPU module in the RUN status, the systems will be switched.

In the case with the control system CPU module in the RUN status and the standby system CPU module in the STOP status, only the standby system CPU module is reset.

3) When a WDT error has occurred in the standby system CPU module (in backup mode)

The standby system CPU module cannot be reset even when the remote RESET operation is performed for the control system CPU module.

In this case, perform the operation using the following path (communication path where the tracking cable is not relayed).

- Directly connect a personal computer (GX Developer) to the standby system CPU module.
- Perform the operation via a module in the standby system (such as use of the MC protocol).

4) When remote operation is being performed for the CPU module (in backup mode) using a different path

No remote operation can be performed from another GX Developer for the CPU module where remote operation is being performed.

In the case where remote operation is being performed using different paths for each CPU module in the control system and the standby system as shown in Figure 6.20, the standby system CPU module may not be reset, even when the remote RESET operation is performed for the control system CPU module.

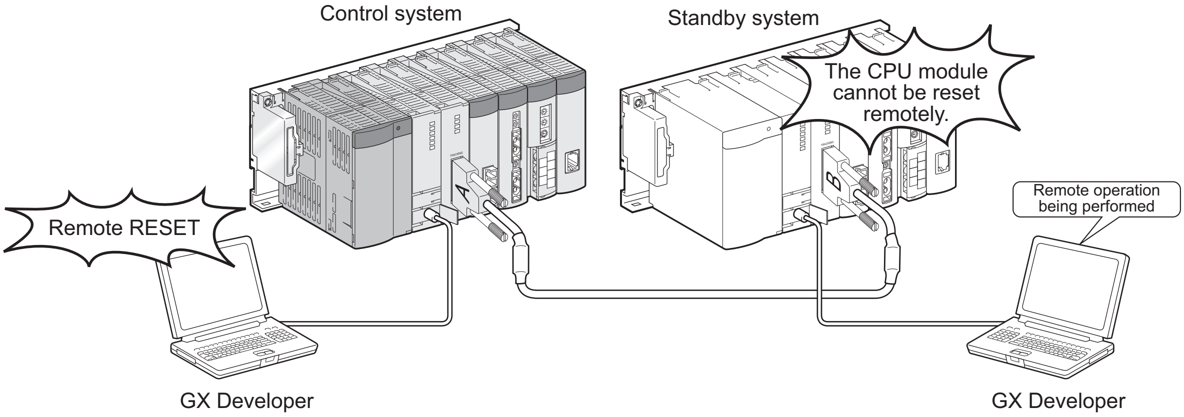

Control system
Standby system
The CPU module cannot be reset remotely.
Remote RESET
Remote operation being performed
A
B
GX Developer
GX Developer

Figure 6.20 Remote operation using different paths

If remote operation is being performed for the standby system CPU module from different GX Developer, cancel the remote operation for the standby system CPU module and then perform the remote RESET operation for the control system CPU module.

6.6.4 Remote latch clear

(1) Definition

This function resets the latched device data from GX Developer or an external device when the CPU module is in the STOP status.

(2) 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

(3) Executing method

This operation can only be performed by GX Developer or an external device using the MC protocol.

- Select [Online] → [Remote operation] in GX Developer.
- To execute the remote latch clear operation from an external device, use the MC protocol command.

☞ 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.

(4) 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 Section 3.7(2)(c).

(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.

(d) In a redundant system where the Redundant CPU is used

1) When the "Both systems" item is not selected on the Remote operation screen

Remote latch clear operation is performed for only the CPU module in the system specified in the connection target settings.

2) When the "Both systems" item is selected on the Remote operation screen

Remote latch clear operation is performed for both CPU modules in the control system and standby system.

Make sure to set the CPU modules in both systems in backup mode.

6.6.5 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

Table6.9 shows the operating status of the CPU module according to the combination of remote operation and RUN/STOP status.

Table6.9 Operating 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 STOP status includes a case where the CPU module is stopped due to an error.

(2) Remote operations from a single GX Developer

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

(3) Remote operations from multiple GX Developers

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

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

(1) Definition

This function changes the input response time for each Q series-compatible module.

Table6.10 shows the modules available for input response time change and selectable time settings.

Table6.10 Modules available for input response time change

Module name	Type	Settable time setting
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 time here.

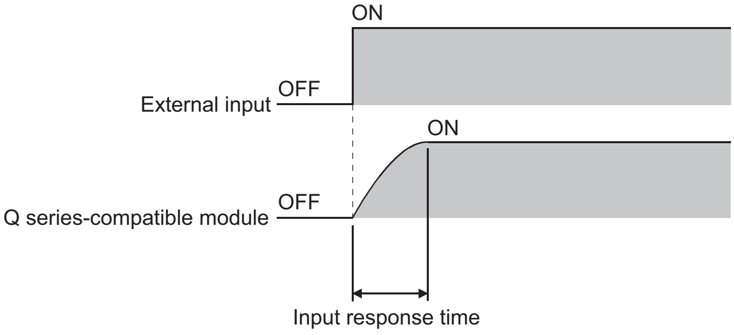


Figure 6.21 Input response time

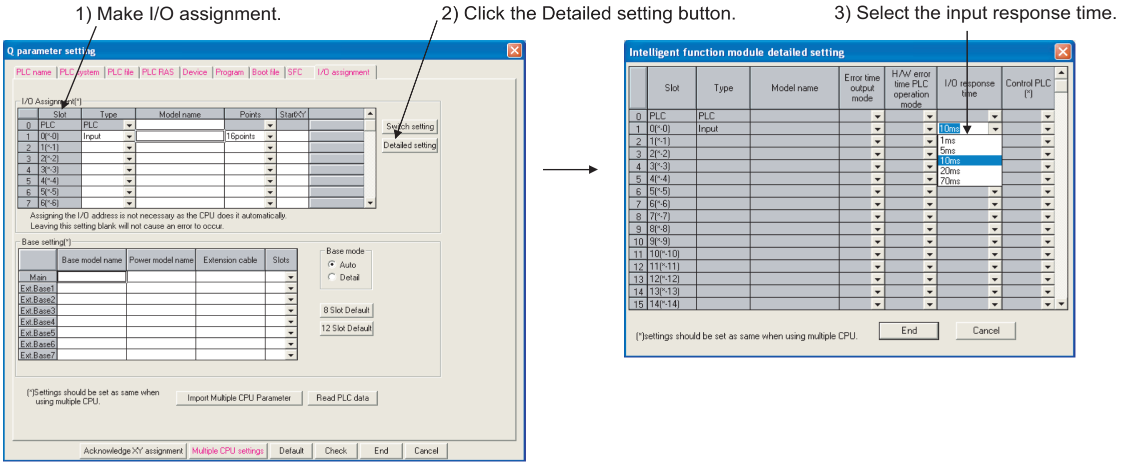
(2) Input response time setting

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

1) Make I/O assignment for the target module.

2) Click the Detailed setting button.

3) On the screen opened, select an input response time value ("I/O response time").


1) Make I/O assignment.

2) Click the Detailed setting button.

3) Select the input response time.

Q parameter setting — I/O assignment

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)	Input		16points	

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base mode: ● Auto ○ Detail

(*)Settings should be set as same when using multiple CPU.

Intelligent function module detailed setting

	Slot	Type	Model name	Error time output mode	H/W error time PLC operation mode	I/O response time	Control PLC (*)
0	PLC	PLC					
1	0(*-0)	Input				10ms	

I/O response time: 1ms, 5ms, 10ms, 20ms, 70ms

(*)settings should be set as same when using multiple CPU.

Figure 6.22 Input response time setting

(3) Precautions

(a) Restrictions on GX Developer version

When changing the input response time of the following modules, use GX Developer shown below.

- High-speed input module: GX Developer Version 5 (SW5D5C-GPPW-E) or later
- Interrupt module: GX Developer Version 6 (SW6D5C-GPPW-E) or later

When GX Developer version earlier than the version above is used, a default time value is used.

(b) 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.

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

An input response time value cannot be changed.
If any input response time value is set for the slot where an AnS/A series-compatible input module or interrupt module is mounted, the setting is ignored.

(d) 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.

6.8 Error Time Output Mode Setting

(1) Definition

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.

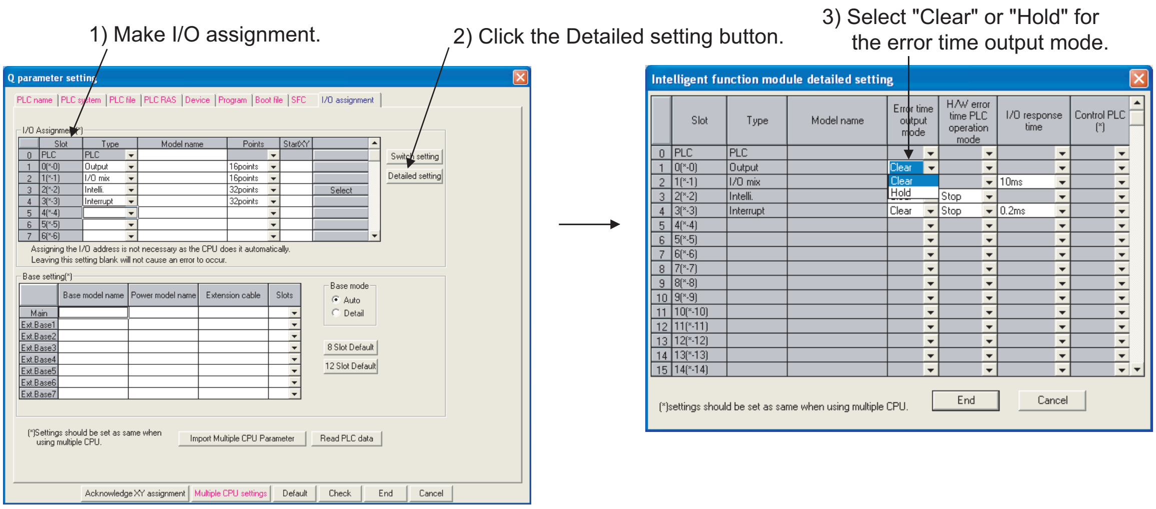
(2) Error time output mode setting

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

1) Make I/O assignment for the target module.

2) Click the Detailed setting button.

3) On the screen opened, select "Clear" or "Hold". (Default: "Clear")



1) Make I/O assignment.

2) Click the Detailed setting button.

3) Select "Clear" or "Hold" for the error time output mode.

Q parameter setting — I/O assignment

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)	Output		16points	
2	1(*-1)	I/O mix		16points	
3	2(*-2)	Intelli.		32points	
4	3(*-3)	Interrupt		32points	
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode: ● Auto ○ Detail

(*)Settings should be set as same when using multiple CPU.

Intelligent function module detailed setting

	Slot	Type	Model name	Error time output mode	H/W error time PLC operation mode	I/O response time	Control PLC (*)
0	PLC	PLC					
1	0(*-0)	Output		Clear			
2	1(*-1)	I/O mix				10ms	
3	2(*-2)	Intelli.			Stop		
4	3(*-3)	Interrupt		Clear	Stop	0.2ms	
5	4(*-4)						
6	5(*-5)						
7	6(*-6)						
8	7(*-7)						
9	8(*-8)						
10	9(*-9)						
11	10(*-10)						
12	11(*-11)						
13	12(*-12)						
14	13(*-13)						
15	14(*-14)						

(*)settings should be set as same when using multiple CPU.

Figure 6.23 Error time output mode setting

(3) 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.

Failure to perform either of the operations above after changing the error time output mode setting will result in "PARAMETER ERROR" (error code: 3000).

6.9 H/W Error Time PLC Operation Mode Setting

(1) Definition

This function determines the program operation mode (stop or continue) of the CPU module when a hardware error occurs in an intelligent function module or interrupt module.

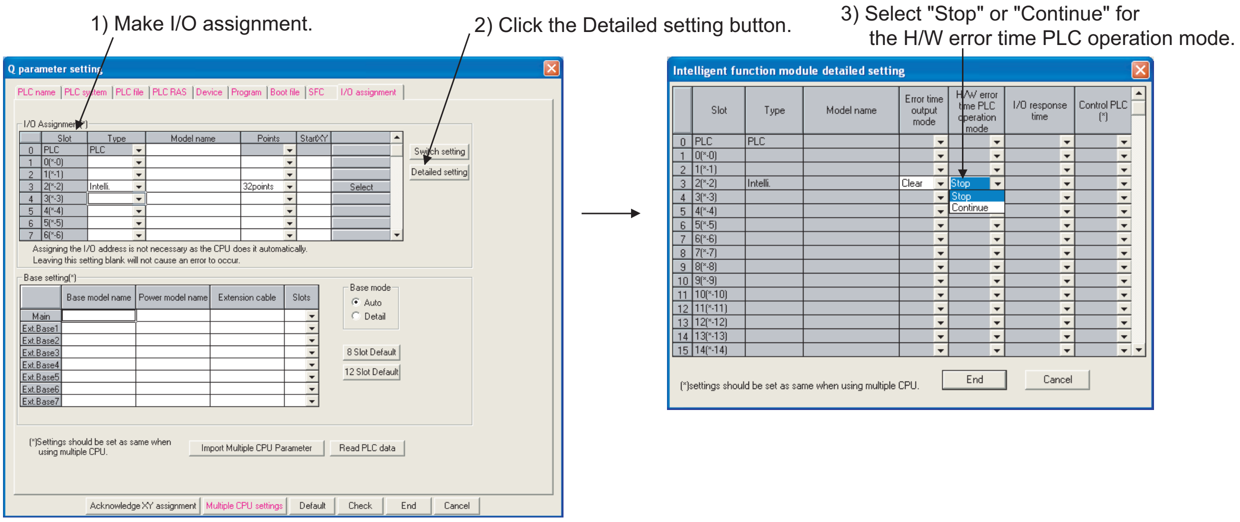
(2) H/W error time PLC operation mode setting

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

1) Make I/O assignment for the target module.

2) Click the Detailed setting button.

3) On the screen opened, select "Stop" or "Continue". (Default: "Stop")



1) Make I/O assignment.

2) Click the Detailed setting button.

3) Select "Stop" or "Continue" for the H/W error time PLC operation mode.

Q parameter setting

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY	
0	PLC	PLC				
1	0(*-0)					
2	1(*-1)					
3	2(*-2)	Intelli.		32points		Select
4	3(*-3)					
5	4(*-4)					
6	5(*-5)					
7	6(*-6)					

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode
● Auto
○ Detail

(*)Settings should be set as same when using multiple CPU.

Intelligent function module detailed setting

	Slot	Type	Model name	Error time output mode	H/W error time PLC operation mode	I/O response time	Control PLC (*)
0	PLC	PLC					
1	0(*-0)						
2	1(*-1)						
3	2(*-2)	Intelli.		Clear	Stop Stop Continue		
4	3(*-3)						
5	4(*-4)						
6	5(*-5)						
7	6(*-6)						
8	7(*-7)						
9	8(*-8)						
10	9(*-9)						
11	10(*-10)						
12	11(*-11)						
13	12(*-12)						
14	13(*-13)						
15	14(*-14)						

(*)settings should be set as same when using multiple CPU.

Figure 6.24 H/W error time PLC operation setting

(3) 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.

6.10 Intelligent Function Module Switch Setting

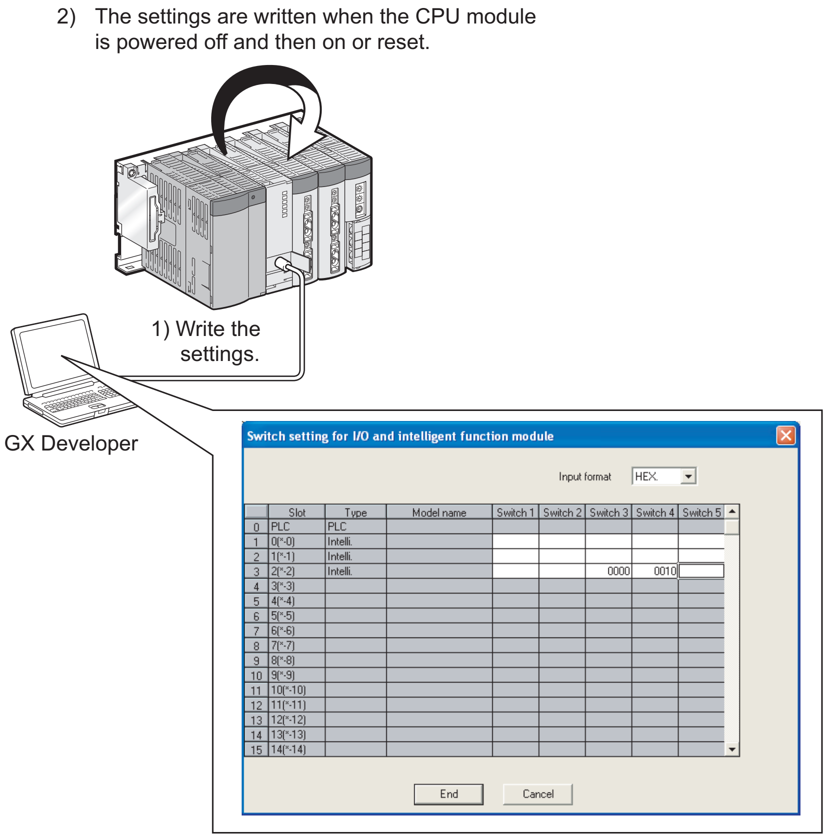
(1) Definition

This function sets the switches of each Q series-compatible intelligent function module and interrupt module in GX Developer.

(2) 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) The settings are written when the CPU module is powered off and then on or reset.
1) Write the settings.
GX Developer
End
Cancel

Switch setting for I/O and intelligent function module

Input format HEX.

	Slot	Type	Model name	Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
0	PLC	PLC						
1	0(*-0)	Intelli.						
2	1(*-1)	Intelli.						
3	2(*-2)	Intelli.				0000	0010	
4	3(*-3)							
5	4(*-4)							
6	5(*-5)							
7	6(*-6)							
8	7(*-7)							
9	8(*-8)							
10	9(*-9)							
11	10(*-10)							
12	11(*-11)							
13	12(*-12)							
14	13(*-13)							
15	14(*-14)							

Figure 6.25 Writing the switch settings to each intelligent function module

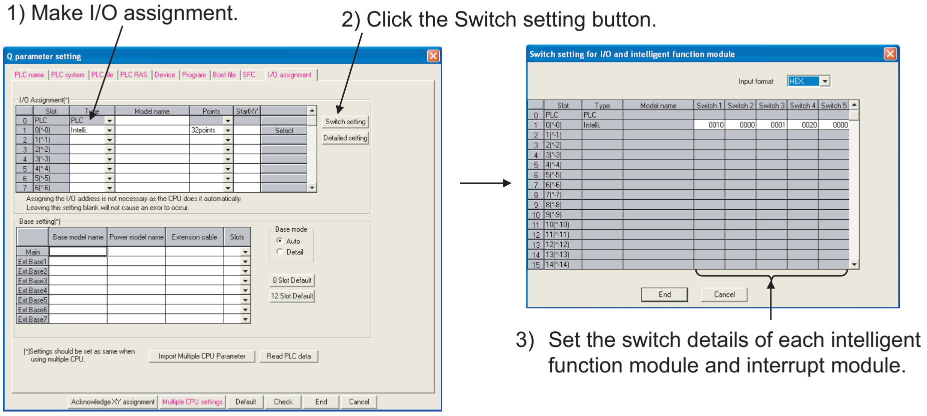
(3) Switch settings

Set the switch details for each intelligent function module and interrupt module in the I/O assignment tab of the PLC parameter dialog box.

1) Make I/O assignment for the target module.

2) Click the Switch setting button.

3) Set the switch details for each module.



1) Make I/O assignment.

2) Click the Switch setting button.

3) Set the switch details of each intelligent function module and interrupt module.

Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY	
0	PLC	PLC				
1	0(*-0)	Intelli.		32points		Select
2	1(*-1)					
3	2(*-2)					
4	3(*-3)					
5	4(*-4)					
6	5(*-5)					
7	6(*-6)					

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode
● Auto
○ Detail

(*)Settings should be set as same when using multiple CPU.

Switch setting for I/O and intelligent function module

Input format HEX.

	Slot	Type	Model name	Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
0	PLC	PLC						
1	0(*-0)	Intelli.		0010	0000	0001	0020	0000
2	1(*-1)							
3	2(*-2)							
4	3(*-3)							
5	4(*-4)							
6	5(*-5)							
7	6(*-6)							
8	7(*-7)							
9	8(*-8)							
10	9(*-9)							
11	10(*-10)							
12	11(*-11)							
13	12(*-12)							
14	13(*-13)							
15	14(*-14)							

Figure 6.26 Switch settings

(4) Precautions

(a) When an AnS/A series-compatible module is used Note6.2

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) Setting differences between GX Developer versions

Set the following type for interrupt modules depending on the version of GX Developer.

- GX Developer Version 6 (SW6D5C-GPPW-E) or later: "Interrupt"
- GX Developer Version 5 (SW5D5C-GPPW-E) or earlier: "Intelli."

(d) 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.

Note6.2 Basic Process Redundant

The Basic model QCPU, Process CPU and Redundant CPU do not support the use of AnS/A series-compatible input modules or interrupt modules.

6.11 Monitor Function

(1) Definition

This function reads program and device data in the CPU module, and intelligent function module status using GX Developer.

Table6.11 List of monitor functions and availability

Monitor function	Availability				Reference
	Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	
Ladder monitor	○	○	○	○	GX Developer Version 8 Operating Manual
Device/buffer memory batch monitor	○	○	○	○	
Entry data monitor	○	○	○	○	
Device test	○	○	○	○	
Entry ladder monitor	○	○	○	○	
Monitor condition setting	×	○	○	○	Section 6.11.1
Local device monitor/test	×	○	○	○	Section 6.11.2
External input/output forced on/off	△*1	○	○	○	Section 6.11.3

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

*1: External input/output can forcibly be turned on/off only by device test.

(2) Monitor request timing and displayed data

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

(3) Monitor with monitor condition settings

By setting the monitor condition in GX Developer during debugging, the program operation status in the CPU module can be monitored under the specified condition.
Besides, the monitoring status under the specified condition can be held by setting the monitoring stop condition.

(4) Local device monitor

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

6.11.1 Monitor condition setting Note6.3

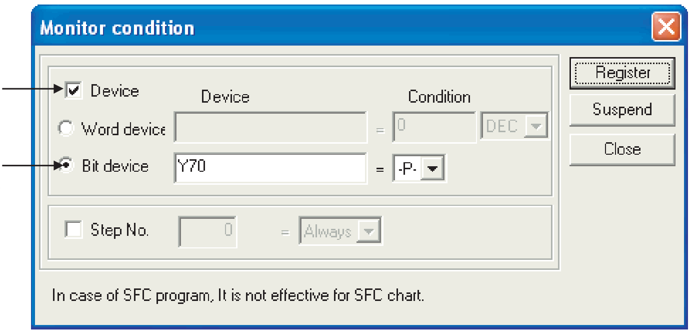
This function is used to monitor data in the CPU module under the specified condition.

(1) Monitor condition setting for ladder monitor

Switch GX Developer into monitor mode.

Select [Online] → [Monitor] → [Monitor condition setup] to open the Monitor condition screen.

Set the condition as shown below to monitor data on the rising edge of Y70.



Monitor condition

☑ Device

○ Word device: Device (blank) = Condition 0, DEC

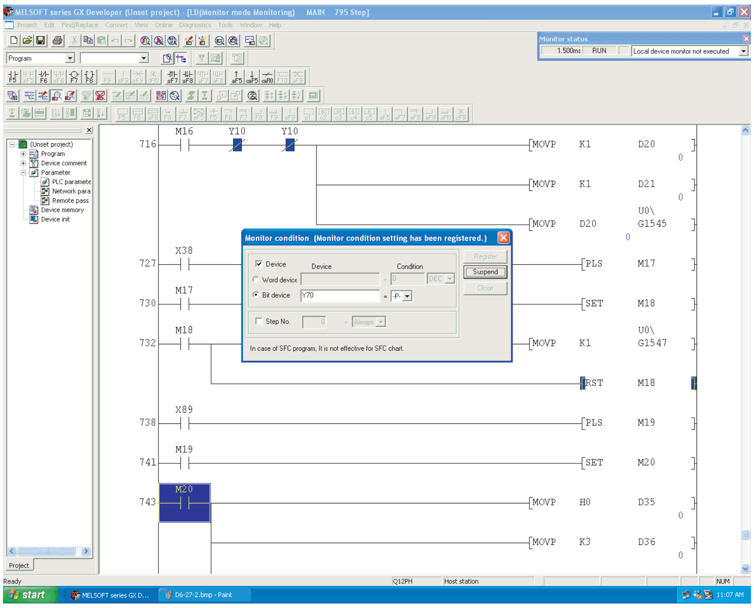
● Bit device: Y70 = -P-

□ Step No. 0 = Always

In case of SFC program, It is not effective for SFC chart.

Select to use a device as a monitor condition.

Select to use a step number as a monitor condition.



Monitor condition (Monitor condition setting has been registered.)

☑ Device

● Bit device: Y70 = -P-

□ Step No. 0 = Always

In case of SFC program, It is not effective for SFC chart.

Monitoring starts when the condition is established.
(An operation status when the condition is established is displayed.)

Figure 6.27 Monitor condition setting screen

Note6.3 **Basic**

The Basic model QCPU cannot perform monitor operation under the specified condition.

(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.

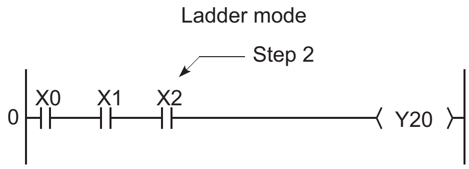
The following status can be specified.

- When the operation of the specified step changes from the non-execution status to the execution status: <-P->
- When the operation of the specified step changes from the execution status to the non-execution status: <-F->
- 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 step specified as a monitor condition.
The following shows examples of monitoring when the step 2 is on (Step No. [2] = <ON>).

- When the step 2 is connected by the AND instruction:
  In Figure 6.28, the monitor execution condition is established when both X0 and X1 are on.

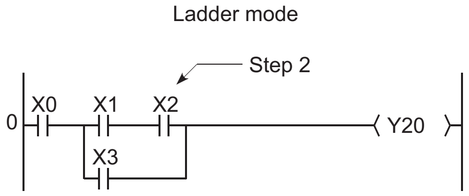


List mode

```
0  LD   XO
1  AND  X1
2  AND  X2
3  OUT  Y2
```

Figure 6.28 When the step 2 is connected by the AND instruction

- When the step 2 is connected in the middle of the AND/OR block:
  In Figure 6.29, 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.)

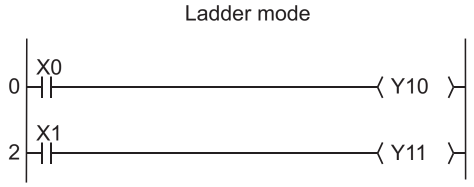


List mode

```
0  LD   XO
1  LD   X1
2  AND  X2
3  OR   X3
4  ANB
5  OUT  Y20
```

Figure 6.29 When the step 2 is connected in the middle of the AND/OR block

- If 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 immediately 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.



List mode

```
0  LD   XO
1  OUT  Y10
2  LD   X1
3  OUT  Y11
```

Figure 6.30 When the start of a ladder block other than the step 0 is specified for the step number

● Be sure to set the condition of the step set as step No.0 to "Always".

(b) When only a device is specified

Either word device or bit device can be specified.

1) 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).

2) 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]=<-P->, 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.

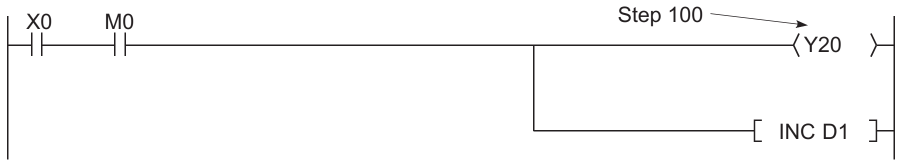


Figure 6.31 When the rising edge of the step 100 and D1=5 are specified

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

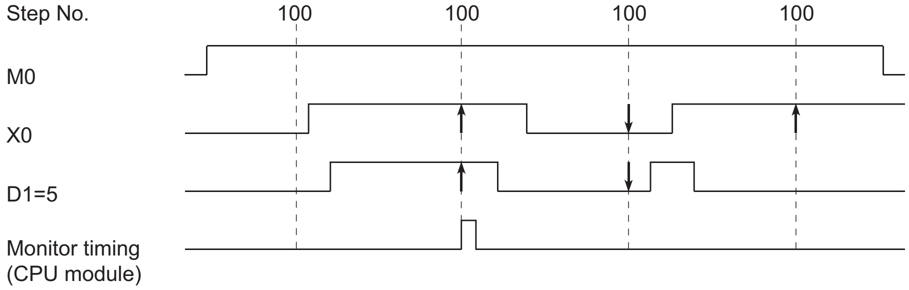
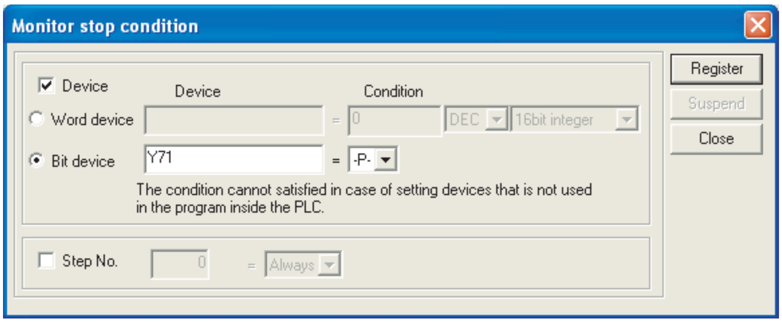


Figure 6.32 Monitor timing of GX Developer

(2) Monitor stop condition setting

Set a monitor stop condition on the screen opened by selecting [Online] → [Monitor] → [Monitor stop condition setup].

Set the condition as shown in Figure 6.33 to stop a monitor operation on the rising edged of Y71.



Monitor stop condition

☑ Device

Device | Condition

○ Word device = 0 DEC 16bit integer

● Bit device Y71 = -P-

The condition cannot satisfied in case of setting devices that is not used in the program inside the PLC.

☐ Step No. 0 = Always

Figure 6.33 Monitor stop condition screen

(a) When a device is specified

Either word device or bit device can be specified.

1) When a word device is specified

A monitor operation is stopped when the current value of the specified word device becomes the specified value.

Enter the current value (in decimal, hexadecimal, 16-bit integer, 32-bit integer, or real number).

2) When a bit device is specified

A monitor operation is stopped when the execution status of the specified bit device becomes the specified status.

Select the execution condition (on the rising edge or falling edge).

(b) When a step number is specified

A monitor operation is stopped when the execution status of the step specified as a monitor condition 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: <-P->
- When the operation of the specified step changes from the execution status to the non-execution status: <-F->
- 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>

If a step number is not specified, a monitor operation is stopped after END processing of the CPU module.

(3) Precautions

(a) Files to be monitored

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

(b) No file register setting

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

(c) Device assignment

For monitor operation, device assignment in the CPU module and GX Developer 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.

- GX Developer
- 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.

6.11.2 Local device monitor/test Note6.4

This operation is useful for debugging a program, monitoring local devices (☞ Section 9.13.2) in the program monitored by GX Developer.

(1) Monitoring a local device

Table6.12 shows 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....)

Table6.12 Data displayed when three programs are executed

Setting	Monitored device	
	D0 (Local device)	**D100 (Global device)**
Local device monitor is set	The D0 value (local device) in the specified 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" has been set for "Local device" in "File usability setting" of the Program tab of the PLC parameter dialog box, the D0 value after execution of the specified program is monitored.

Note6.4 **Basic**

For the Basic model QCPU, there is no conceptual distinction between the global and local devices. Monitor settings for local devices are not necessary.

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.

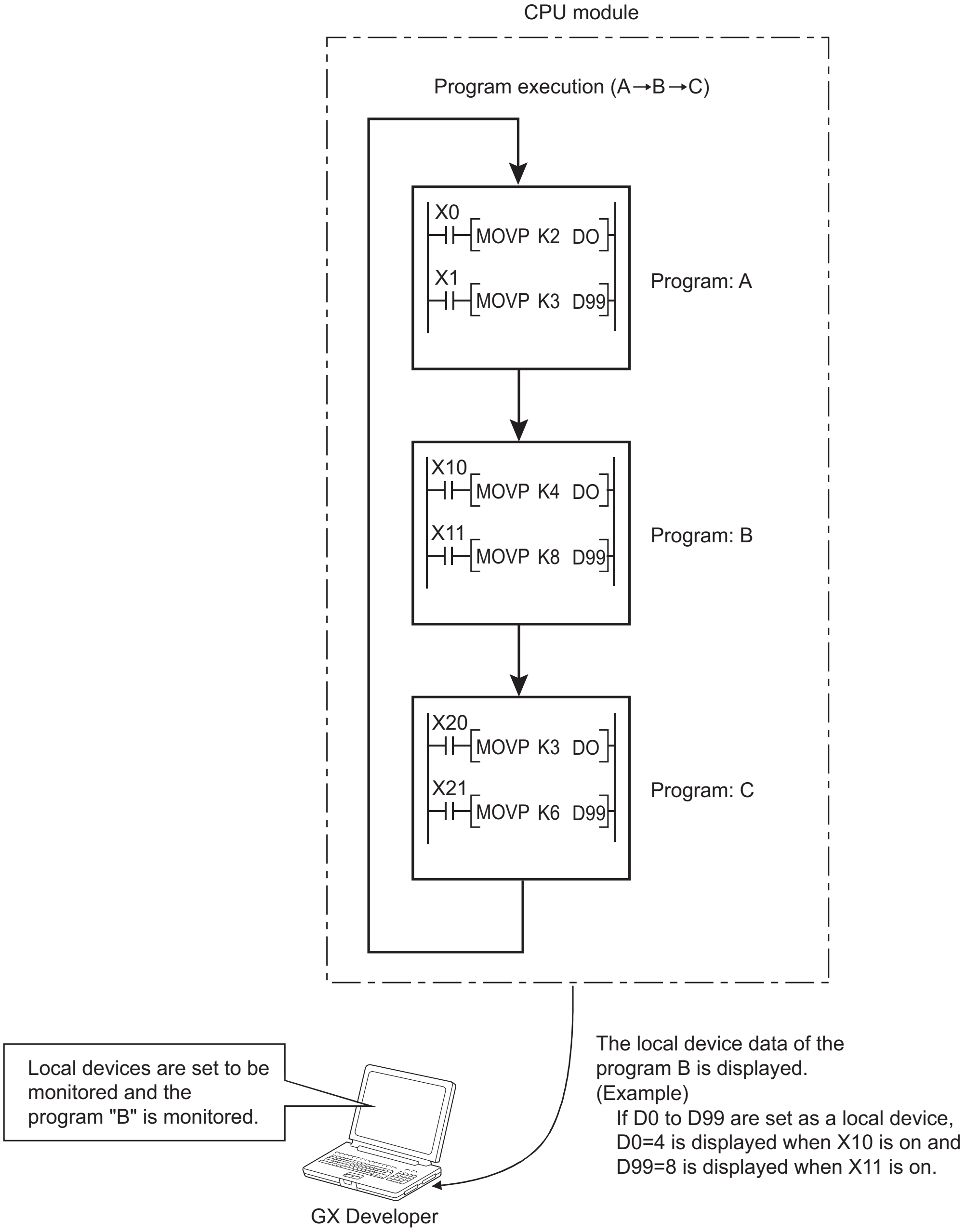

CPU module
Program execution (A→B→C)
X0
MOVP K2 DO
X1
MOVP K3 D99
Program: A
X10
MOVP K4 DO
X11
MOVP K8 D99
Program: B
X20
MOVP K3 DO
X21
MOVP K6 D99
Program: C
Local devices are set to be monitored and the program "B" is monitored.
The local device data of the program B is displayed.
(Example)
If D0 to D99 are set as a local device, D0=4 is displayed when X10 is on and D99=8 is displayed when X11 is on.
GX Developer

Figure 6.34 Local device monitor example

(2) Monitoring procedure

The following shows the local device monitoring procedure.

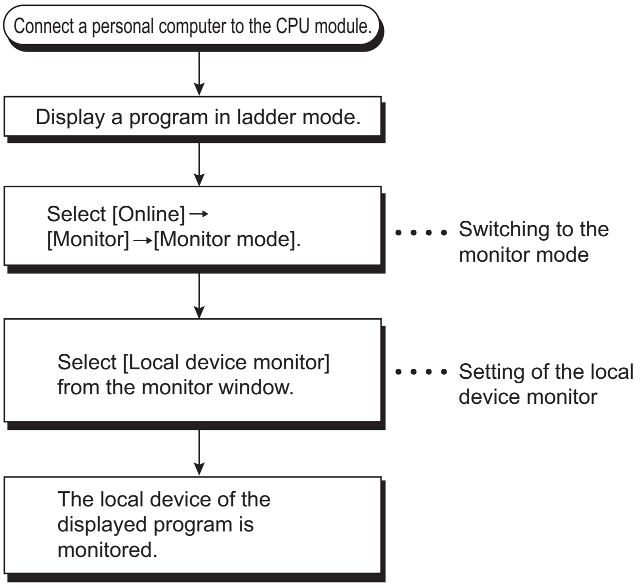

Connect a personal computer to the CPU module.
Display a program in ladder mode.
Select [Online]→ [Monitor]→[Monitor mode].
Switching to the monitor mode
Select [Local device monitor] from the monitor window.
Setting of the local device monitor
The local device of the displayed program is monitored.

Figure 6.35 Local device monitoring procedure

(3) Precautions

(a) Local devices that can be monitored/tested by a single GX Developer

A single GX Developer 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 GX Developers 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. (☞ Section 9.13.2)

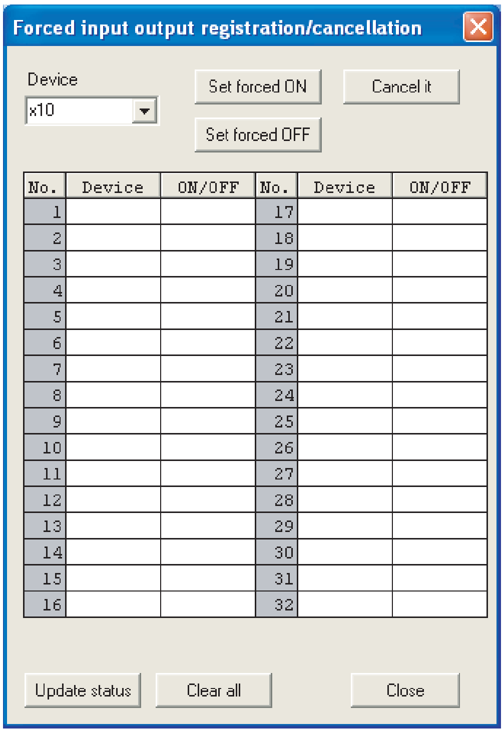
(d) Monitoring local devices in a fixed scan execution type program

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

6.11.3 External input/output forced on/off Note6.5

The external input/output can forcibly be turned on/off on the screen opened by selecting [Online] → [Debug] → [Forced input output registration/cancellation] in GX Developer.

The information registered for forced on/off can be cancelled by an operation from GX Developer.



Device: x10

No.	Device	ON/OFF	No.	Device	ON/OFF
1			17		
2			18		
3			19		
4			20		
5			21		
6			22		
7			23		
8			24		
9			25		
10			26		
11			27		
12			28		
13			29		
14			30		
15			31		
16			32		

Figure 6.36 Forced input output registration/cancellation screen

(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").

Table6.13 shows the CPU module operation when a forced on/off operation is performed.

Table6.13 Input/output operation 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.

*1: When the Redundant CPU modules in redundant systems are in backup mode, output (Y) of the standby system CPU module cannot forcibly be turned on/off.

Note6.5 Universal

When performing an external input/output forced on/off operation for the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, or Q26UDHCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.2)

Figure 6.37 shows the input/output operation when a forced on/off operation is performed.

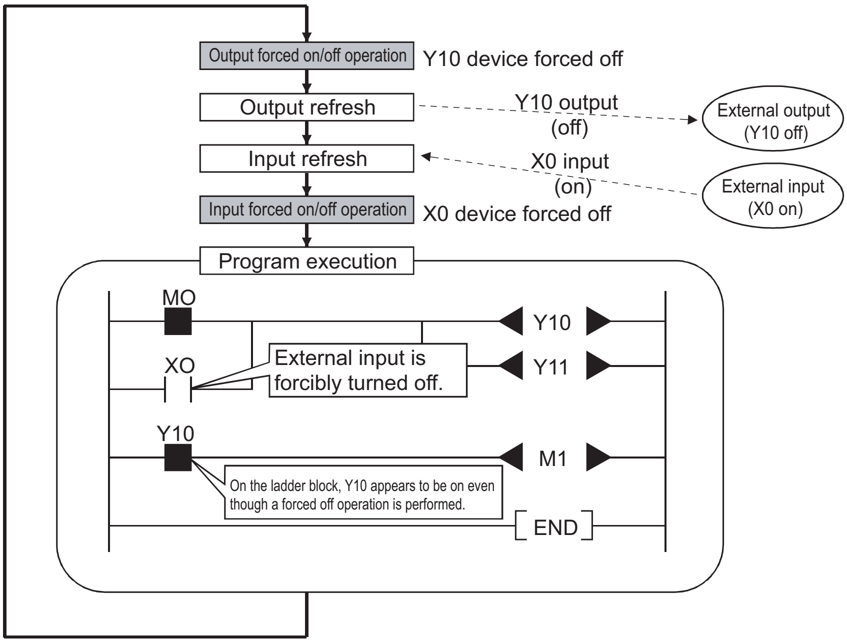

Output forced on/off operation
Y10 device forced off
Output refresh
Y10 output (off)
External output (Y10 off)
Input refresh
X0 input (on)
External input (X0 on)
Input forced on/off operation
X0 device forced off
Program execution
MO
XO
Y10
External input is forcibly turned off.
Y10
Y11
M1
On the ladder block, Y10 appears to be on even though a forced off operation is performed.
END

Figure 6.37 Input/output operation when a forced on/off operation is performed

(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.

Point

When the systems are switched in the redundant system, the new control system CPU module continues the forced on/off operation using the data registered in the old control system CPU module.

(b) Registerable devices

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 or MELSECNET/H module
- Input (X) and output (Y) of the CPU module to be refreshed from RX/RY of a CC-Link module

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 host CPU module can forcibly turned on/off.)

(d) Cancelling on/off registration data

The registered forced ON/OFF data can be canceled by GX Developer.

Once the registered data is canceled, the status of the forced on/off registered devices will be as follows.

Table6.14 Status of devices after forced on/off registration data is canceled

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 or MELSECNET/H module	Used the on/off status refreshed via CC-Link IE Controller Network or MELSECNET/H.	
	Input of the CPU module to be refreshed from RX of a CC-Link module	Uses the on/off status refreshed via CC-Link.	
	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 or MELSECNET/H module		
	Output of the CPU module to be refreshed from RY of a CC-Link module		
	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.

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.

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

Table6.15 shows the external input/output forced on/off timing.

Table6.15 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 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 module	• During END processing (auto refresh) • At execution of the COM instruction (auto refresh) • At execution of the ZCOM instruction (auto refresh)	

(f) Number of registerable devices

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 registration status

Forced on/off execution status can be checked by:

- reading the forced on/off registration status by GX Developer,
- 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).

(i) Forcibly turning input or output on/off from multiple GX Developers

Forced on/off can be registered to a single CPU module from multiple GX Developers connected via network. In this case, the last registration will be effective.

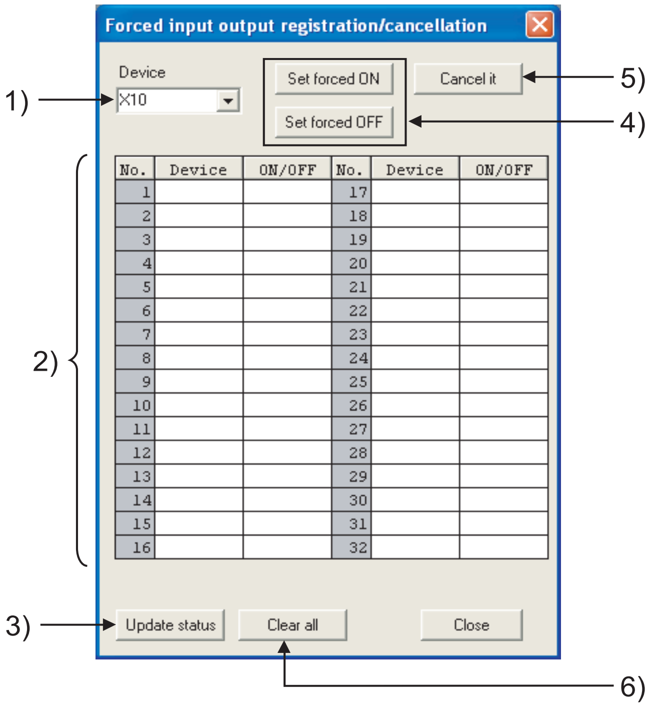
For this reason, the forced on/off status which is different from the status actually registered in the CPU module may be displayed on the screen of GX Developer that registered forced on/off earlier.

When the forced on/off registration is performed from multiple GX Developers, click the "Update status" button to update the registered data and execute the function.

(3) Operating procedure

Operating procedure is described below.

- To register forced on/off for a device, select [Online] → [Debug] → [Forced input output registration/cancellation] in GX Developer.
- On the screen opened, specify a device and click the "Set forced ON" or "Set forced OFF" button.



No.	Device	ON/OFF	No.	Device	ON/OFF
1			17		
2			18		
3			19		
4			20		
5			21		
6			22		
7			23		
8			24		
9			25		
10			26		
11			27		
12			28		
13			29		
14			30		
15			31		
16			32		

Figure 6.38 Forced input output registration/cancellation screen

Table6.16 Items on the Forced input output registration/cancellation screen

No.	Item	Description
1)	Device	Select the I/O number for which forced on/off is to be registered or cancelled.
2)	Registration status display area	Displays the forced on/off registration status.
3)	Update status	Reads the forced on/off registration status from the CPU module.
4)	Set forced ON/OFF	Registers forced on/off for a device specified.
5)	Cancel it	Cancels forced on/off registered for the device specified.
6)	Clear all	Cancels all forced on/off registration.

(4) Precautions in a redundant system where the Redundant CPU is used

(a) CPU module for which forced on/off is registered or cancelled

In redundant systems, register or cancel forced on/off for the control system CPU module. (Forced on/off cannot be registered or cancelled for the CPU modules in both systems individually.)

After the systems are switched, register or cancel forced on/off for the CPU module in the new control system (the system that was switched from the standby system to the control system).

If forced on/off is registered or cancelled for the standby system CPU module, an error (error code: 4240H) occurs.

(b) When the CPU modules are in separate mode

When the control system CPU module is powered off and then on or reset, the registered forced on/off is cancelled.

Input and output operations of the CPU module after cancellation will be the same as those described for "Forced on/off cancellation (no operation)" in Table6.13.

Note that the output operation of the module on the MELSECNET/H remote I/O station will be as follows.

1) While the CPU module is powered off or reset

The CPU module holds the output status in the control system.

2) When the CPU module is powered off and then on or reset

The CPU module outputs the results of sequence program operations. (Same operation for "Forced on/off cancellation (no operation).)

6.12 Writing Programs While CPU Module is in RUN Status

There are two ways of writing programs in the RUN status.

- Online change (ladder mode): ☞ Section 6.12.1
- Online change (files): ☞ Section 6.12.2

Data can also be written in the RUN status using a pointer. (☞ Section 6.15.2)

Point

For online change in the Redundant CPU system (order of writing to the control system and standby system and enable/disable setting of execution of tracking transfer during online change), refer to the following.

☞ QnPRHCPU User's Manual (Redundant System)

6.12.1 Online change (ladder mode)

(1) Definition

This function writes programs to the CPU module in the RUN status.

This function enables the program to be changed without stopping the program operation in the CPU module.

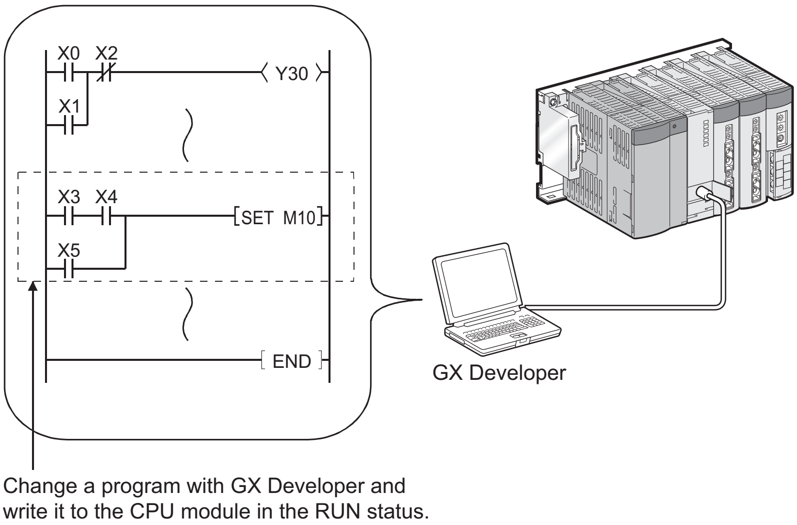

X0
X2
Y30
X1
X3
X4
SET M10
X5
END
GX Developer
Change a program with GX Developer and write it to the CPU module in the RUN status.

Figure 6.39 Outline of online change (ladder mode)

This function also can write programs by GX Developer connected to another station on the network.

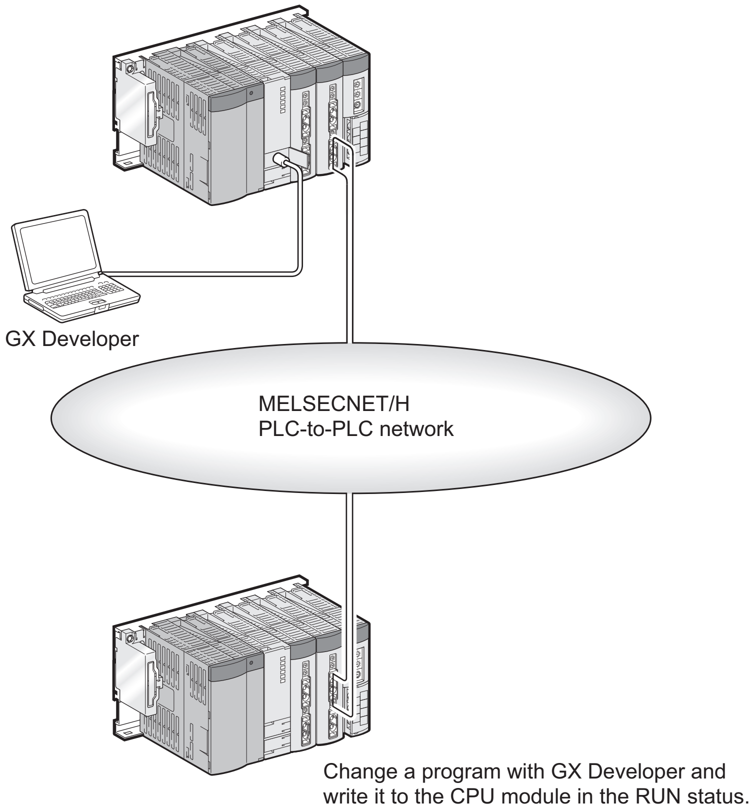

GX Developer
MELSECNET/H
PLC-to-PLC network
Change a program with GX Developer and write it to the CPU module in the RUN status.

Figure 6.40 Outline of online change via network

(2) Memory for online change

Online change can be performed to the program memory only.

(3) Number of steps that can be batch-written by online change

Up to 512 steps can be batch-written.

(4) Execution timing in low-speed execution type program Note6.6

In the low-speed execution type program, data are written in the RUN status at END processing in the scan following the scan where execution of all the programs are completed.
Note that the execution of the programs is suspended during online change.

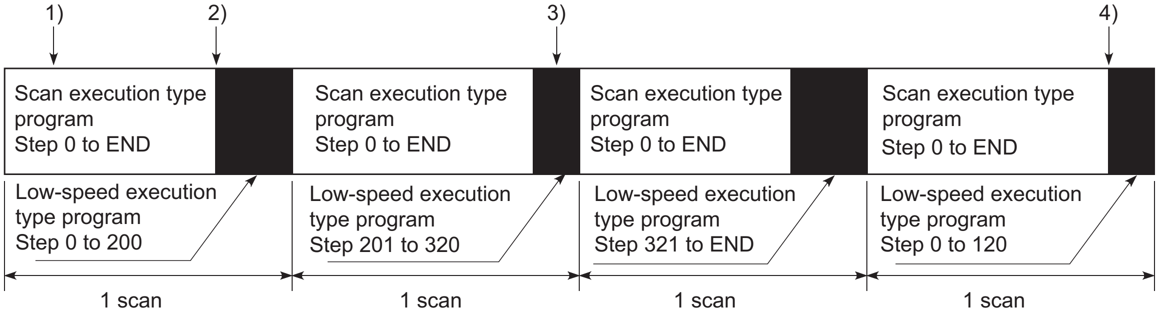

1)
2)
3)
4)
Scan execution type program Step 0 to END
Scan execution type program Step 0 to END
Scan execution type program Step 0 to END
Scan execution type program Step 0 to END
Low-speed execution type program Step 0 to 200
Low-speed execution type program Step 201 to 320
Low-speed execution type program Step 321 to END
Low-speed execution type program Step 0 to 120
1 scan
1 scan
1 scan
1 scan

Figure 6.41 Online change at execution of low-speed execution type program

1): Online change command of scan execution type program
2): Online change execution of scan execution type program
3): Online change command of low-speed execution type program
4): Online change execution of low-speed execution type program

(5) Online change during execution of the PLOADP, PUNLOADP, or PSWAPP instruction Note6.7

Online change is suspended until execution of these instructions are completed.
If any of the instructions is executed during online change, the processing does not start until the online change is completed.

Note6.6 Basic Redundant

For the Basic model QCPU and Redundant CPU, the low-speed execution type program cannot be used.

Note6.7 Basic Redundant

The Basic model QCPU and Redundant CPU cannot change a program file in the RUN status with the PLOAP, PUNLOADP, or PSWAPP instruction.

(6) 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 changing the size of reserved area for online change.

(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 re-set 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 re-set when programs are written in the RUN status.

For increase in the scan time, refer to Section 10.1.3.

(7) Instructions that do not operate normally when programs are written to the CPU module in the RUN status

Refer to Section 6.12.3(2).

6.12.2Online change (files) Note6.8

(1) Definition

This function batch-writes files shown in Table6.17 to the CPU module in the RUN status by online operation from GX Developer.

Table6.17 Files that can be written to the CPU module in the RUN status

File type	CPU module built-in memory			Memory card (RAM)	Memory card (ROM)	
	Program memory	Standard RAM	Standard ROM	SRAM card	Flash card	ATA card
Parameter	×	×	×	×	×	×
Intelligent function module parameter	×	×	×	×	×	×
Program	○	×	○	○	×	○
Device comment	○	×	△*1	△*1	×	△*1
Initial device value	×	×	×	×	×	×
File register	×	△*1	×	△*1	×	×
Local device	×	×	×	×	×	×
Sampling trace file	×	○	×	○	×	×
Programmable controller user data	×	×	○	×	×	○

○: Can be written. △: Partially restricted. × : Cannot be written.

*1: The file can be written if not being accessed by a sequence program.

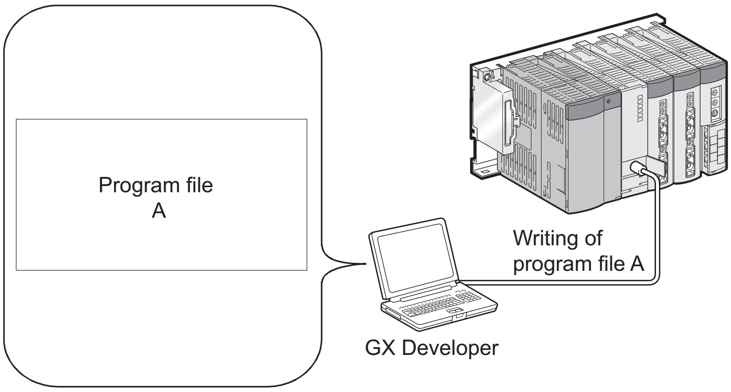


Figure 6.42 Outline of online change (files)

Note6.8 Basic

The Basic model QCPU does not support the online change (files) function.

(2) Availability

Table6.18 shows whether online change (files) can be performed or not depending on memory area.

Table6.18 Execution of online change (files) depending on memory status

Free area equal to or larger than a program file to be written		Online change (files)
Program memory	**Memory card**	
Available	Not available/available	Executable
Not available	Available	Executable*1
Not available	Not available	Not executable

*1: When "Trailing edge instructions are not executed" is selected in "Instruction operational settings for online change/file online change" in the Options tab of GX Developer, if the program file to be written contains the fall instruction, it cannot be written in the RUN status.
Note when a program to be written is a SFC program, even if the program file to be written contains the fall instruction, it can be written in the RUN status.
(For operation when "Trailing edge instructions are not executed" is selected, refer to Section 6.12.3.)

(3) 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 Section 10.1.3.

(4) When a file is accessed by a sequence program instruction

An instruction in a program cannot access to the file being written to the CPU module in the RUN status.

If doing so, the instruction will not be executed.

The relevant instructions are as follows:

- PLOADP instruction
- PUNLOADP instruction
- PSWAPP instruction

(5) 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.

(6) Online change (files) of SFC programs Note6.9

SFC programs cannot be written in units of files to the CPU module in the RUN status.

(7) Instructions that do not operate normally when files are written to the CPU module in the RUN status

Refer to Section 6.12.3(2).

Note6.9 High performance Process

When writing a SFC program file in the RUN status to the High Performance model QCPU or Process CPU, check the versions of the CPU module and GX Developer. (Appendix 2.2, Appendix 2.3)

6.12.3Precautions for online change

The following shows precautions for online change.

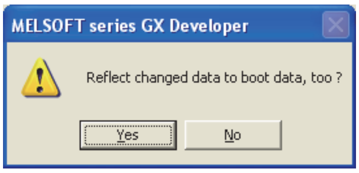
(1) Online change during boot operation

The status of a boot source program when data are written in the RUN status during boot operation depends on the CPU module and boot source memory used.

Table6.19 Status of boot source program when data are written in the RUN status

Boot source memory		Status of boot source program			
		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
Standard ROM		Not changed*2	Not changed*2	Not changed*2	Changed*1 *3
Memory card	• SRAM card • ATA card	-	Changed*1	Changed*1	Changed*1
	Flash card	-	Not changed*2	Not changed*2	Changed*1 *3

*1: Clicking "Yes" in the following message box changes a boot source program.
In this case, the online change processing takes time.



MELSOFT series GX Developer

Reflect changed data to boot data, too ?

Clicking "No" does not change a boot source program.
Before powering off or resetting the CPU module after online change, write data in the program memory to the standard ROM or memory card.

*2: Before powering off or resetting the CPU module after online change, write data in the program memory to the standard ROM or memory card.

*3: When a boot source memory is the standard ROM or Flash card, files in the boot source are deleted and replaced by the ones in the program memory.
Configure setting in the Boot file tab of the PLC parameter dialog box so that the files in the boot source memory are to be transferred to the program memory.

(2) Instructions do not operate normally during online change

If the following instructions are executed during online change, they do not operate normally.

- Fall instruction
- Rise instruction
- SCJ instruction
- STMR instruction

(a) Fall instruction

The fall instruction within the program targeted for online change will be executed even though the execution condition of the instruction (off → on) is not met at the completion of online change.

To prevent the fall instruction from being executed during online change, refer to POINT in this section.

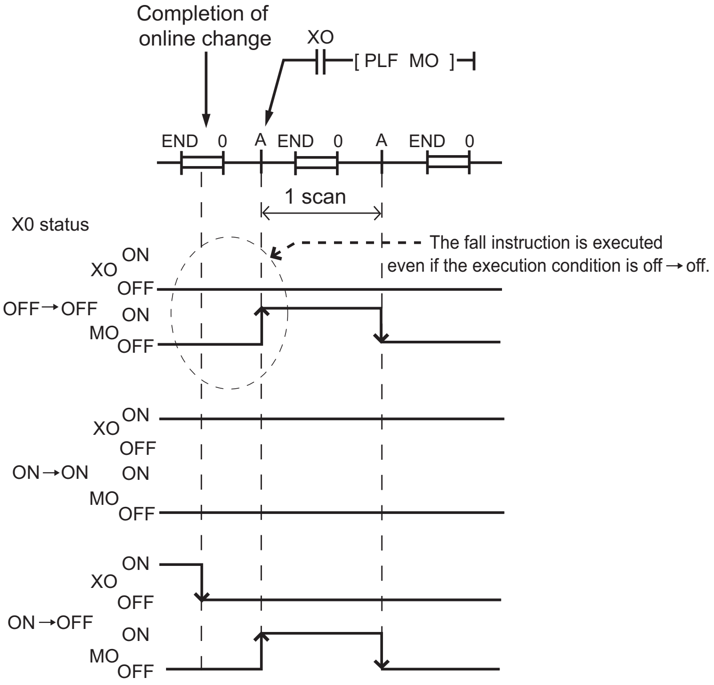


Figure 6.43 Operation of the fall instruction

The corresponding fall instructions are LDF, ANDF, ORF, MEF, PLF, FCALLP, and EFCALLP.

(b) Rise instruction

The rise instruction within the program targeted for online change will not be executed even though the execution condition of the instruction (off → on) is met at the completion of online change.

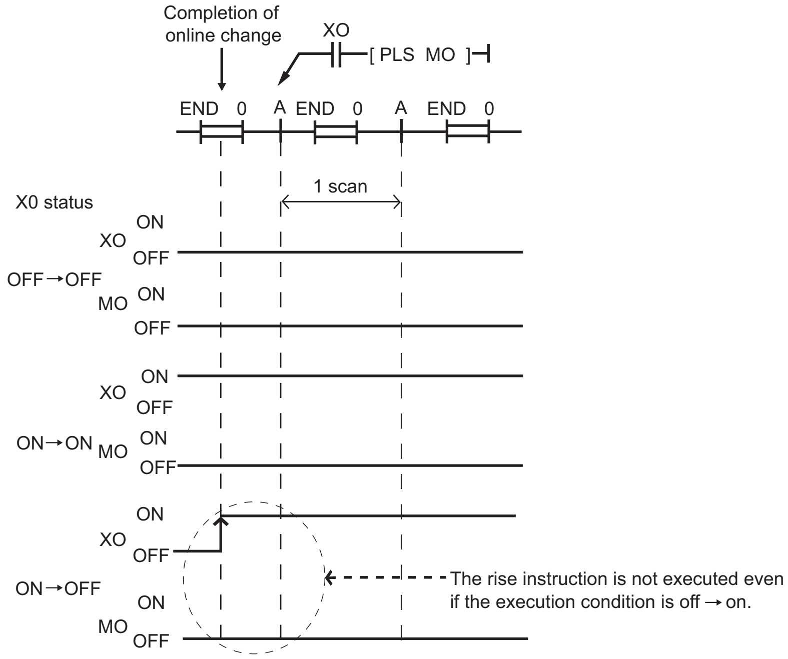


Figure 6.44 Operation of the rise instruction

The corresponding rise instructions are PLS and □P.

(c) SCJ instruction

When the SCJ instruction is used within the program targeted for online change and the execution condition of the instruction is on at the completion of online change, the program jumps to the specified pointer without waiting for one scan.

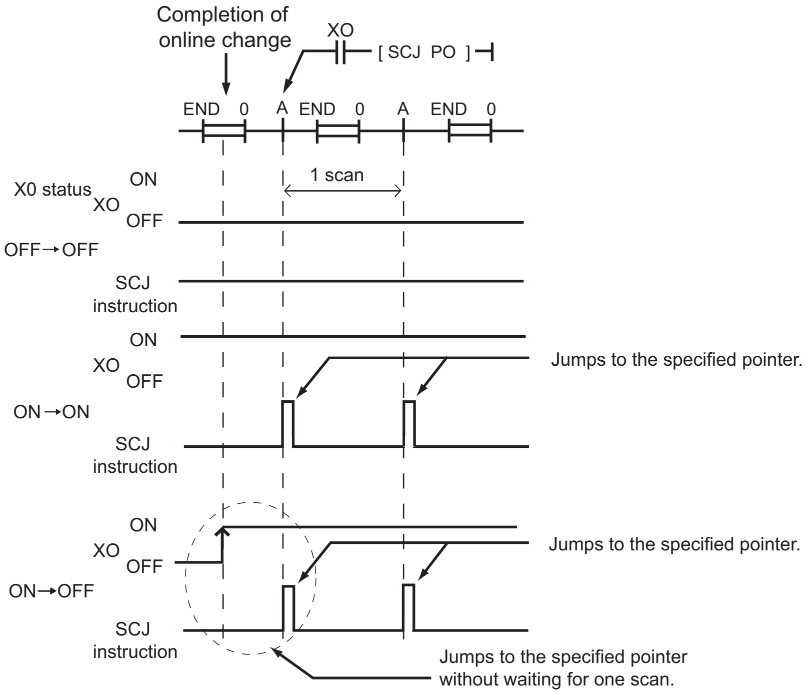
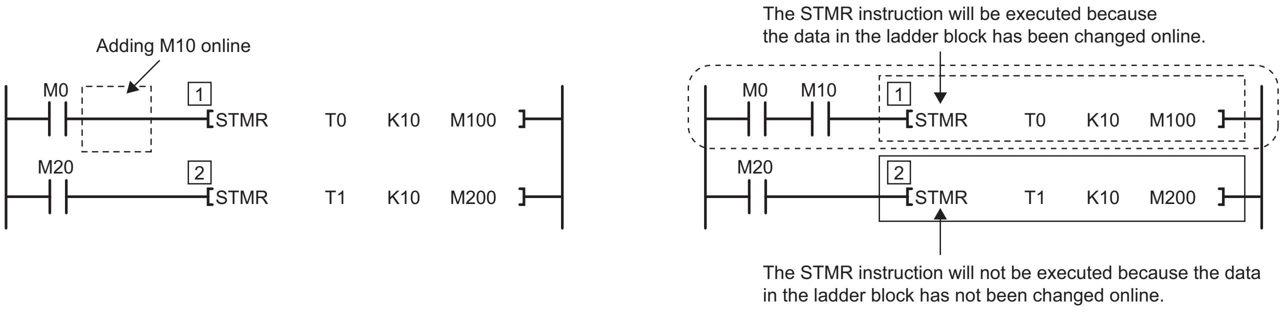

Completion of online change
XO
[SCJ PO]
END 0 A END 0 A END 0
1 scan
X0 status
XO ON OFF
OFF→OFF
SCJ instruction
XO ON OFF
ON→ON
SCJ instruction
Jumps to the specified pointer.
XO ON OFF
ON→OFF
SCJ instruction
Jumps to the specified pointer.
Jumps to the specified pointer without waiting for one scan.

Figure 6.45 Operation of the SCJ instruction

(d) STMR instruction

Note that when the STMR instruction is used in the program targeted for online change, the instruction is executed.


Adding M10 online
M0
1
STMR T0 K10 M100
M20
2
STMR T1 K10 M200
The STMR instruction will be executed because the data in the ladder block has been changed online.
M0 M10
1
STMR T0 K10 M100
M20
2
STMR T1 K10 M200
The STMR instruction will not be executed because the data in the ladder block has not been changed online.

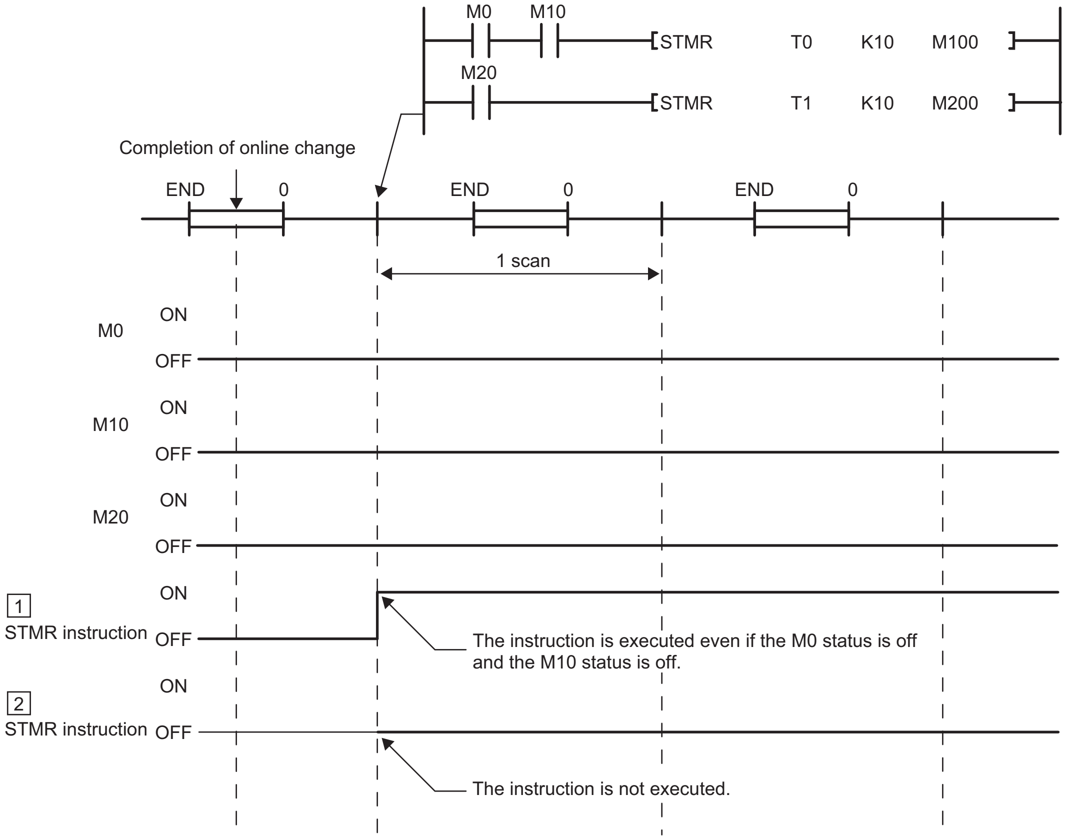

M0 M10
STMR T0 K10 M100
M20
STMR T1 K10 M200
Completion of online change
END 0
END 0
END 0
1 scan
M0 ON OFF
M10 ON OFF
M20 ON OFF
1 STMR instruction ON OFF
2 STMR instruction ON OFF
The instruction is executed even if the M0 status is off and the M10 status is off.
The instruction is not executed.

Figure 6.46 Operation of the STMR instruction

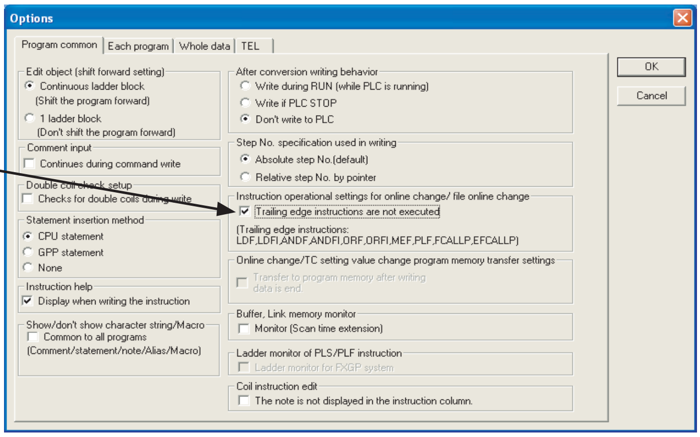
Point

To avoid execution of the fall instruction even when the execution condition (on → off) is not met after data are written to the CPU module in the RUN status, select "Trailing edge instructions are not executed" in the Options screen in GX Developer.

Note6.10

The fall instruction whose execution condition is off is executed by default.

Selecting this option avoids execution of the fall instruction whose execution condition is "off".



Options

Edit object (shift forward setting)
● Continuous ladder block (Shift the program forward)
○ 1 ladder block (Don't shift the program forward)

Comment input
□ Continues during command write

Double coil check setup
□ Checks for double coils during write

Statement insertion method
● CPU statement
○ GPP statement
○ None

Instruction help
☑ Display when writing the instruction

Show/don't show character string/Macro
□ Common to all programs (Comment/statement/note/Alias/Macro)

After conversion writing behavior
○ Write during RUN (while PLC is running)
○ Write if PLC STOP
● Don't write to PLC

Step No. specification used in writing
● Absolute step No.(default)
○ Relative step No. by pointer

Instruction operational settings for online change/ file online change
☑ Trailing edge instructions are not executed
(Trailing edge instructions: LDF,LDFI,ANDF,ANDFI,ORF,ORFI,MEF,PLF,FCALLP,EFCALLP)

Online change/TC setting value change program memory transfer settings
□ Transfer to program memory after writing data is end.

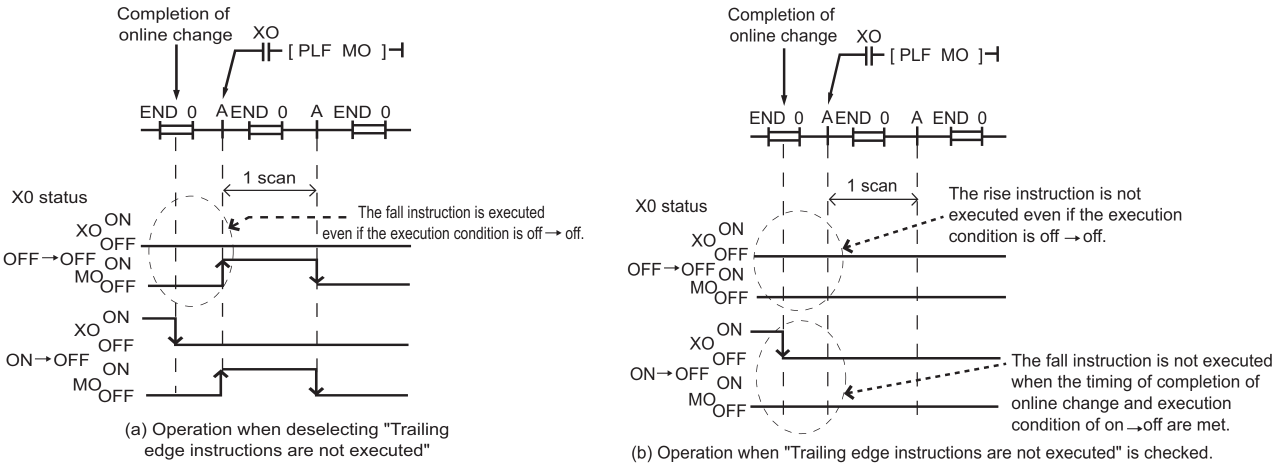
Buffer, Link memory monitor
□ Monitor (Scan time extension)

Ladder monitor of PLS/PLF instruction
□ Ladder monitor for FXGP system

Coil instruction edit
□ The note is not displayed in the instruction column.

Figure 6.47 Options screen

Figure 6.48 shows operations of the fall instruction depending on the setting of "Trailing edge instructions are not executed" in GX Developer.



The fall instruction is executed even if the execution condition is off → off.

(a) Operation when deselecting "Trailing edge instructions are not executed"

The rise instruction is not executed even if the execution condition is off → off.

The fall instruction is not executed when the timing of completion of online change and execution condition of on → off are met.

(b) Operation when "Trailing edge instructions are not executed" is checked.

Figure 6.48 Operation comparison of the fall instruction

Note6.10 Basic High performance Process Redundant

The Basic model QCPU does not support the selection function whether to execute the fall instruction during online change.

To use the selection function for the High Performance model QCPU, Process CPU, or Redundant CPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2)

6.13 Execution Time Measurement

(1) Definition

This function displays the processing time of the program being executed.

(2) Application 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: Section 6.13.1
- Interrupt program list monitor: Section 6.13.2
- Scan time measurement: Section 6.13.3

6.13.1 Program monitor list

(1) Definition

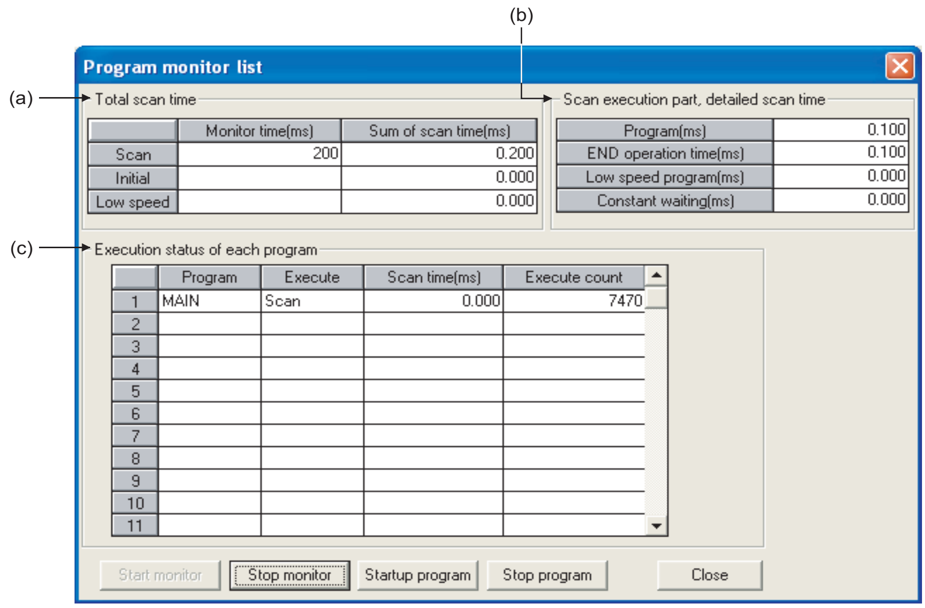
This function displays the processing time of the program being executed.

The scan time, number of execution times, and processing time by item can be displayed for each program.

(2) Execution

Selecting [Online] → [Monitor] → [Program monitor list] displays the Program monitor list screen.[*1]

Figure 6.49 shows an example of executing the program monitor list.



Total scan time

	Monitor time(ms)	Sum of scan time(ms)
Scan	200	0.200
Initial		0.000
Low speed		0.000

Scan execution part, detailed scan time

Program(ms)	0.100
END operation time(ms)	0.100
Low speed program(ms)	0.000
Constant waiting(ms)	0.000

Execution status of each program

	Program	Execute	Scan time(ms)	Execute count
1	MAIN	Scan	0.000	7470
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				

Figure 6.49 Program monitor list screen

[*1] During execution of a fixed scan execution type program, the scan time of the fixed scan execution type program is not displayed.
"-" is displayed in the Scan time column.

(a) Total Scan Time

The monitoring time set in "WDT (Watchdog timer) setting" of the PLC RAS tab of the PLC parameter dialog box and total scan time for each program type during execution by the CPU module are displayed.

1) Monitor time

The monitoring time of each program is displayed.
If the scan time exceeds this time, the CPU module detects "WDT ERROR".

2) Sum of scan time

The total time of each item in "Scan execution part, detailed scan time" is displayed.
When constant scan time is set, the constant scan time is displayed.

(b) Scan execution part, detailed scan time

The details of the scan time are displayed.

1) Program

The total execution time of the scan execution type program is displayed.

2) END operation time

The END processing time is displayed.

3) Low speed program

- High Performance model QCPU and Process CPU
  The total execution time of a low-speed execution type program, or when the constant scan time is set, the total execution time of a low-speed execution type program and constant scan is displayed.
- Basic model QCPU and Redundant CPU
  Since a low-speed execution type program is not used "0.000" is displayed.

4) Constant waiting

The constant scan waiting time is displayed when the constant scan time is set.

(c) Execution status of each program

The execution status of a program selected at the program tab of the PLC parameter dialog box is displayed.

1) Program

The program name is displayed in the order set in the PLC parameter dialog box.

2) Execution

The program type set in the PLC parameter dialog box is displayed.

3) Scan time

The actual scan time (current value) is displayed.
When a program is in stop (standby) status, the scan time is displayed as 0.000 ms.

4) Execute count

The number of execution times of programs before monitoring is displayed, setting the measurement start as "0". The number of execution times is displayed up to 65535 and returns to 0 when the 65536 is measured. The execution times is held even when the program is stopped.

Remark

When the POFF instruction is executed, a non-execution processing is performed for one scan. The number of execution times displayed is the addition of the execution times of the non-execution processing.

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

☞ MELSEC-Q/L Programming Manual (Common Instruction)

(3) Program start Note6.11

Clicking the Startup button on the screen shown in Figure 6.49 (☞ (2) in this section) opens the Startup program screen.

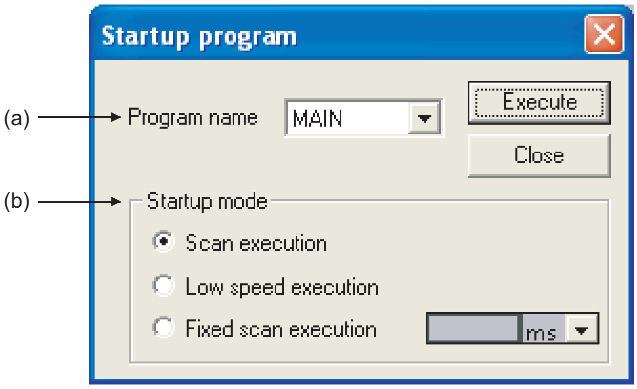


Figure 6.50 Startup program screen

(a) Program name

Select a program set in the Program tab of the PLC parameter dialog box.
A program name cannot be entered as desired.

(b) Startup mode

Set any of the following programs as a start-up program for a stand-by type program.

- "Scan execution": Scan execution type program
- "Low speed execution": Low-speed execution type program Note6.12
- "Fixed scan execution": Fixed scan execution type program

The value set in the Program tab of the PLC parameter dialog box is displayed as the default value of fixed scan execution type program.
The unit can be selected from ms or s.

Note6.11 Basic

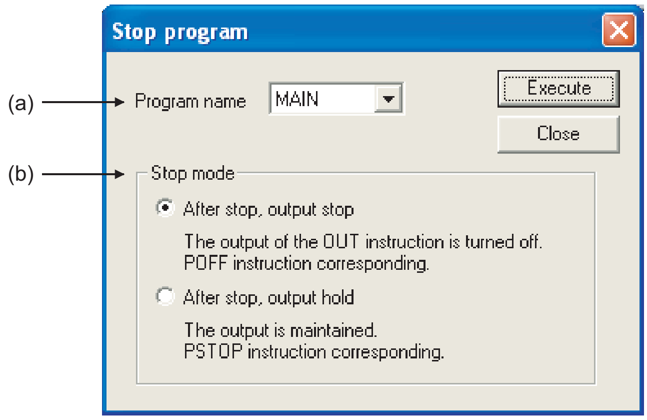
The Basic model QCPU does not support program start from the Program monitor list screen.

Note6.12 Redundant

Since the Redundant CPU does not support the use of low-speed execution type programs, "Low speed execution" cannot be selected in “Startup mode”.

(4) Program stop Note6.13

Clicking the Stop program button on the screen shown in Figure 6.49 (☞ (2) in this section) opens the Stop program screen.



Program name: MAIN

Stop mode

● After stop, output stop

The output of the OUT instruction is turned off.
POFF instruction corresponding.

○ After stop, output hold

The output is maintained.
PSTOP instruction corresponding.

Figure 6.51 Stop program screen

(a) Program name

Select a program set in the Program tab of the PLC parameter dialog box.

A program name cannot be entered as desired.

(b) Stop mode

Executing "After stop, output stop" to the following programs operates as follows.

- Scan execution type program

  Outputs are turned off (non-execution) in the next scan.

  The program enters the standby status after the next scan (This operation is the same when the POFF instruction is executed).

- Low-speed execution type program

  The low-speed execution type program is suspended and outputs are turned off in the next scan.

  The program enters the standby status after the next scan (This operation is the same when the POFF instruction is executed).

- Stand-by type program

  The program is not executed (This operation is the same when the POFF instruction is executed).

  Therefore, "Execute count" is not increased by 1.

Point

Even if "After stop, output stop" is executed, the output may not turn off depending on an instruction.
For details, refer to the section for the POFF instruction in the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

Note6.13 **Basic**

The Basic model QCPU does not support program stop from the program monitor list screen.

6.13.2Interrupt program monitor list

(1) Definition

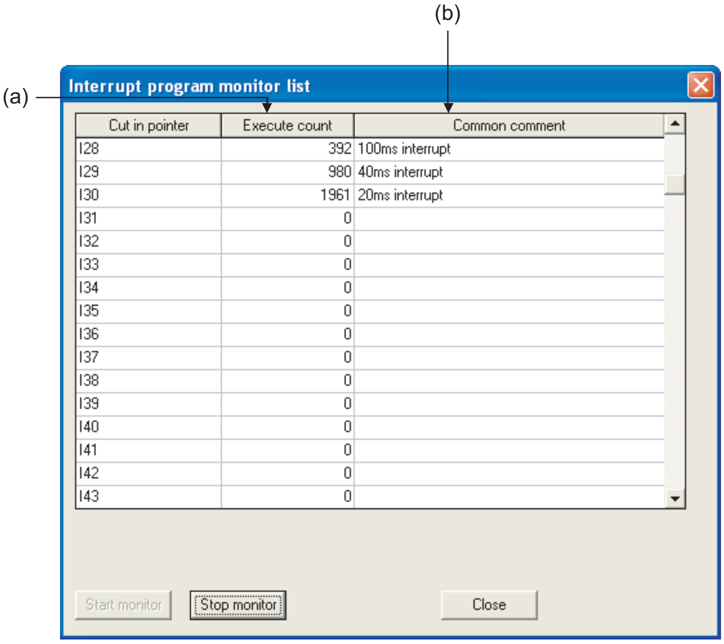
This function displays the number of executions of an interrupt program.

This function is used to check the execution status of the interrupt program.

(2) Execution

Selecting [Online] → [Monitor] → [Interrupt program monitor list] of GX Developer opens the Interrupt program monitor list screen.

Figure 6.52 shows an execution example of the interrupt program monitor list.



Cut in pointer	Execute count	Common comment
I28	392	100ms interrupt
I29	980	40ms interrupt
I30	1961	20ms interrupt
I31	0	
I32	0	
I33	0	
I34	0	
I35	0	
I36	0	
I37	0	
I38	0	
I39	0	
I40	0	
I41	0	
I42	0	
I43	0	

Figure 6.52 Interrupt program monitor list screen

(a) Execute count

The number of executions of an interrupt program is displayed.

This function starts counting after the CPU module is in the RUN status.

When the counting reaches 65536 times, it is reset to 0.

(b) Common comment

Device comments created to an interrupt pointer are displayed.

6.13.3Scan time measurement Note6.14

(1) Definition

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.

(2) 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

(3) 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.

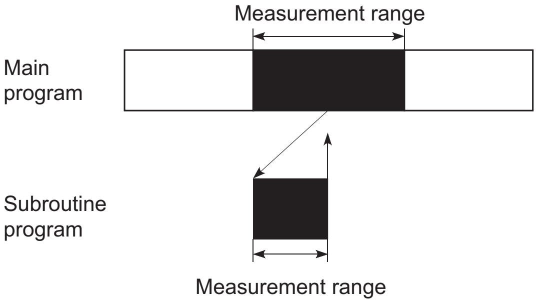


Figure 6.53 When subroutine program is in measurement range

(4) 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.

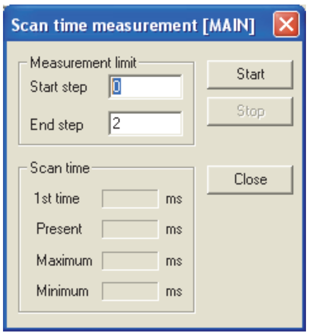
Note6.14 Basic

The Basic model QCPU does not support the scan time measurement.

(5) Execution

Measure the scan time by the following procedure.

- Display the start of the ladder program where scan time is measured in GX Developer and set the monitor mode.
- Select [Online] → [Monitor] → [Scan time measurement] to open the Scan time measurement screen.



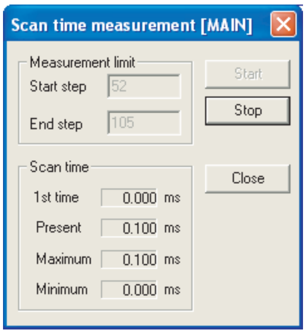
Scan time measurement [MAIN]

Measurement limit
Start step 0
End step 2

Scan time
1st time ms
Present ms
Maximum ms
Minimum ms

- Enter the start and end steps and click the Start button.

Example When the start step is 52 and the end step is 105



Scan time measurement [MAIN]

Measurement limit
Start step 52
End step 105

Scan time
1st time 0.000 ms
Present 0.100 ms
Maximum 0.100 ms
Minimum 0.000 ms

Figure 6.54 Scan time measurement screen

Remark

When displaying the Scan time measurement screen after specifying the scan time measurement rage in the monitor mode, the start and end steps are set in the specified range.
To specify the range, press the "Shift" key and click the mouse. (The specified part is inverted).

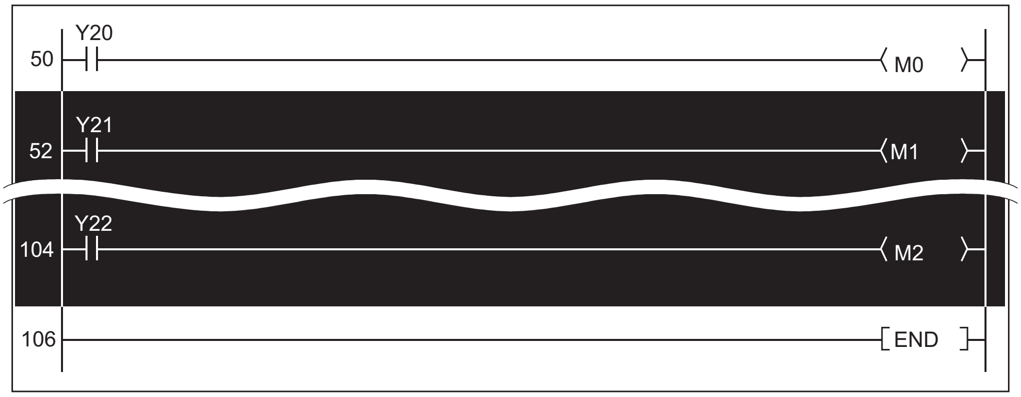


Figure 6.55 Measurement range specification

(6) 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 0.100ms.

If the measurement time is less than 0.100ms, 0.000ms is displayed.

(c) When between the FOR and NEXT instructions is specified

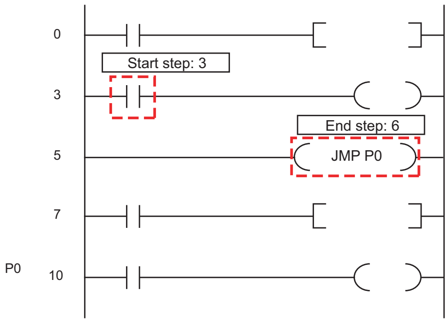
The execution time of one scan between the specified steps is displayed.

(d) When scan time cannot be measured

Scan time cannot be updated on the Scan time measurement screen in the following cases.

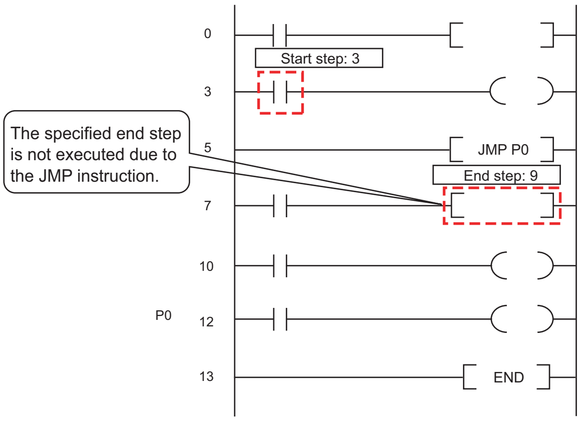
- When the branch instruction is specified the to end step

Example The JMP instruction is specified to the end step.



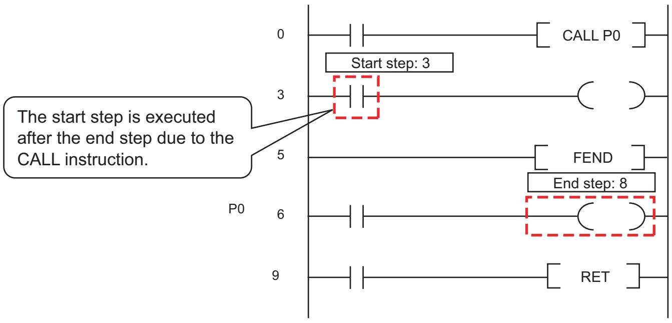
- When only the start step is executed

Example The specified end step is not executed by the JMP instruction.


0
Start step: 3
3
The specified end step is not executed due to the JMP instruction.
5
JMP P0
End step: 9
7
10
P0
12
13
END

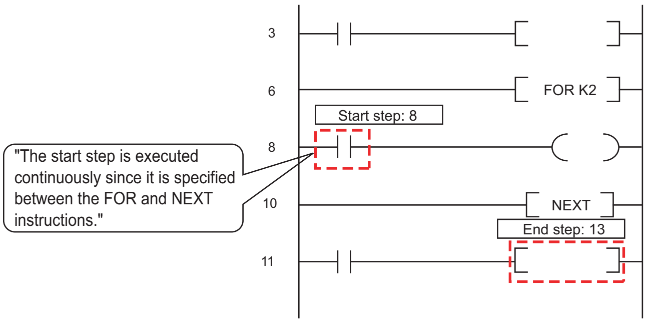
- When the end step is executed before the start step

Example 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.


0
CALL P0
Start step: 3
3
The start step is executed after the end step due to the CALL instruction.
5
FEND
End step: 8
P0
6
9
RET

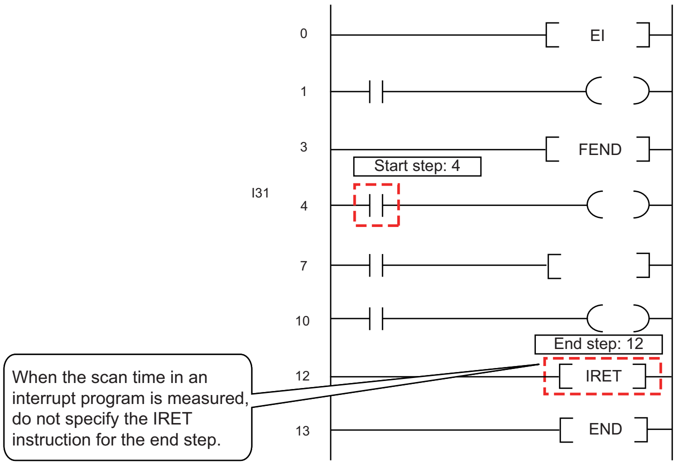
- When the start step is executed continuously

Example Only the start step is specified between the FOR and NEXT instructions.


3
6
FOR K2
Start step: 8
8
"The start step is executed continuously since it is specified between the FOR and NEXT instructions."
10
NEXT
End step: 13
11

- When the end step is specified the IRET instruction, the FEND instruction, the BREAK instruction, or the RET instruction

Example When the end step is specified the IRET instruction on an interrupt program by I31.


0
EI
1
3
FEND
Start step: 4
I31
4
7
10
End step: 12
12
IRET
When the scan time in an interrupt program is measured, do not specify the IRET instruction for the end step.
13
END

6.14 Sampling Trace Function Note6.15

(1) Definition

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.

(2) Application

This function is useful to check the change of the device data used in the program during debugging at a preset timing.
In addition, this function is used to read the device data at trigger condition establishment.

(3) Sampling trace file

This file stores the trace setting necessary for executing the function and trace results.

Point

- Sampling trace file can be stored only in the Standard RAM or SRAM card. (☞ Section 5.2.1(2))
- When storing a sampling trace file in the standard RAM, check the versions of the CPU module and GX Developer. (☞ Appendix 2)

Note6.15 Basic

The Basic model QCPU does not support the sampling trace function.

(4) Sampling trace operation

(a) Operation of the CPU module

When a sampling trace trigger is issued by GX Developer, the CPU module executes traces for the preset number of times.

The sampling trace area can store data up to 60K bytes.

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".

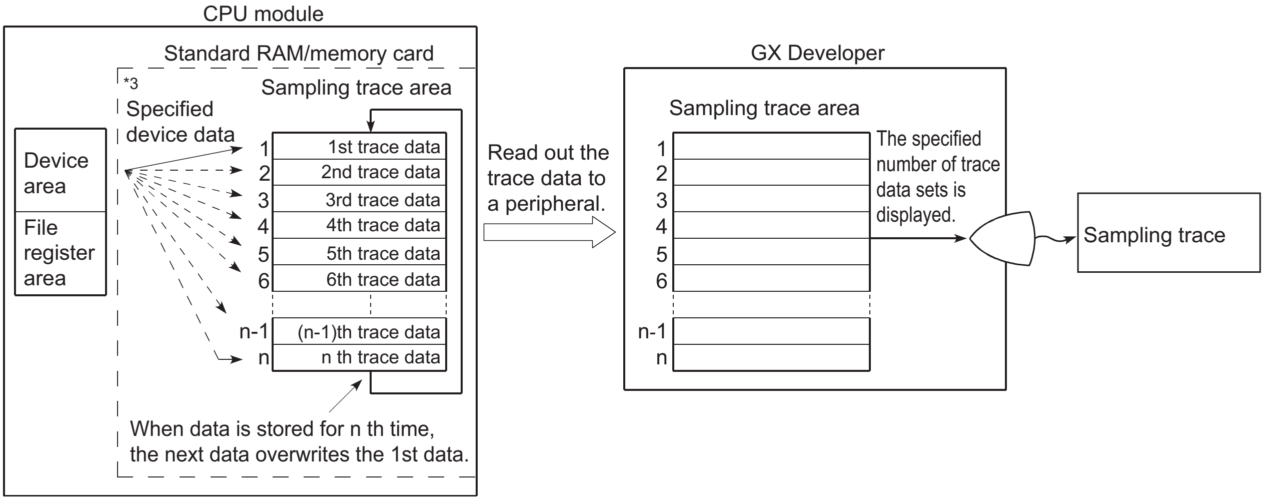


Figure 6.56 Sampling trace operation

*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.

(b) Operation of the special relay

1) When the sampling trace is executed normally

The execution status of the sampling trace can be checked in the special relay listed in Table6.20.

Table6.20 Execution status of the sampling trace

Number	Name	Description
SM800	Trace preparation	Turns on when the trace setting in GX Developer is written to the CPU module. The relay is used to check the sampling trace enable status.
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 condition is established. • A trigger is issued by GX Developer. • The TRACE instruction is executed. • SM803 turns on. • Detailed setting (Device and Step No.) The relay is used to check the trigger condition establishment status.
SM805	Trace completed	Turns on when the sampling trace is completed.
SM826	Trace error	Turns on when an error occurs during sampling trace execution.

Figure 6.57 shows the operation flow chart of the special relay for sampling trace execution.

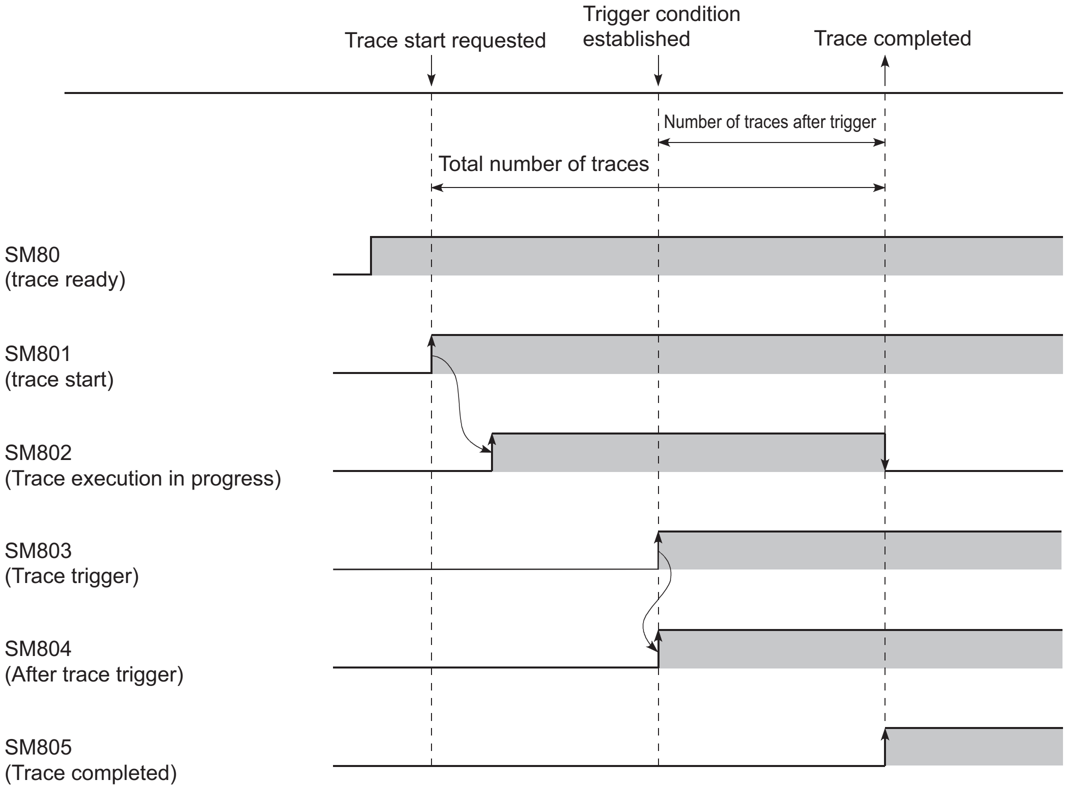


Figure 6.57 Operation flow chart of the special relay (for sampling trace execution)

2) When the sampling trace is interrupted

If SM801 (Trace start) is turned off during sampling trace, execution of the sampling trace will be interrupted.

When the sampling trace is interrupted, the trace count is cleared.

The sampling trace restarts by turning on SM801.

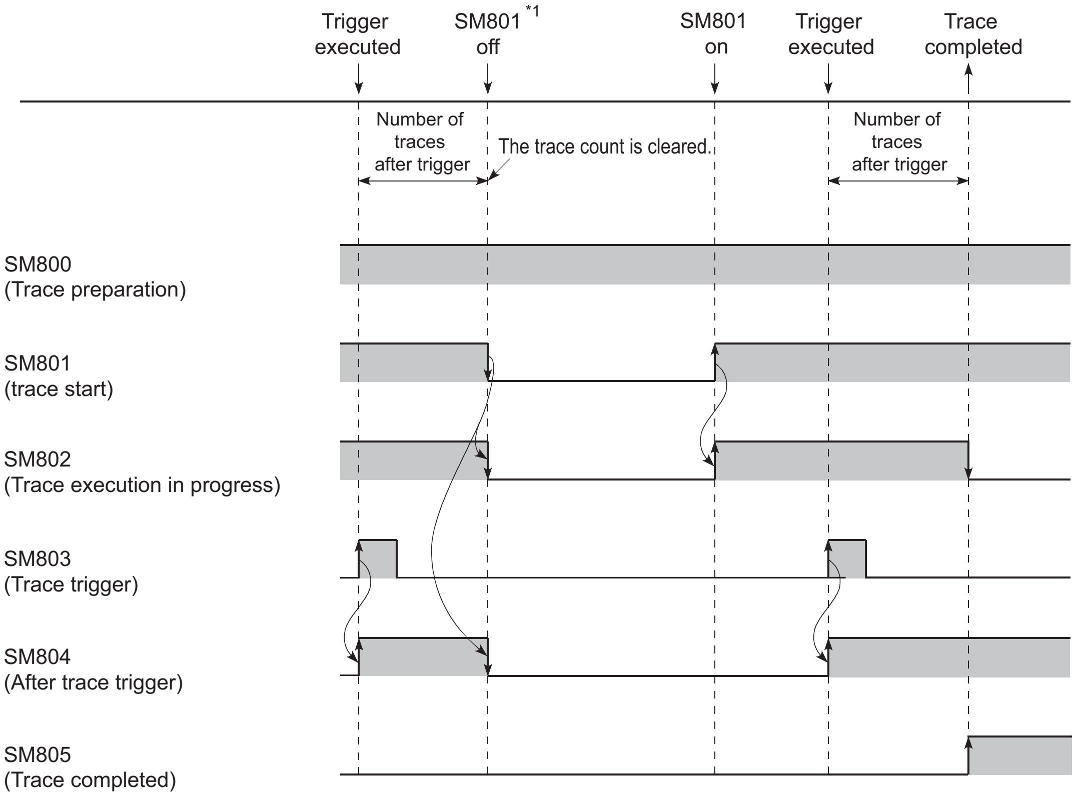

Trigger executed
SM801 *1 off
SM801 on
Trigger executed
Trace completed
Number of traces after trigger
The trace count is cleared.
Number of traces after trigger
SM800 (Trace preparation)
SM801 (trace start)
SM802 (Trace execution in progress)
SM803 (Trace trigger)
SM804 (After trace trigger)
SM805 (Trace completed)

Figure 6.58 Operation flow chart of the special relay (for sampling trace interruption)

*1: SM801 also turns off when the sampling trace is interrupted by GX Developer.

(5) Operating procedure

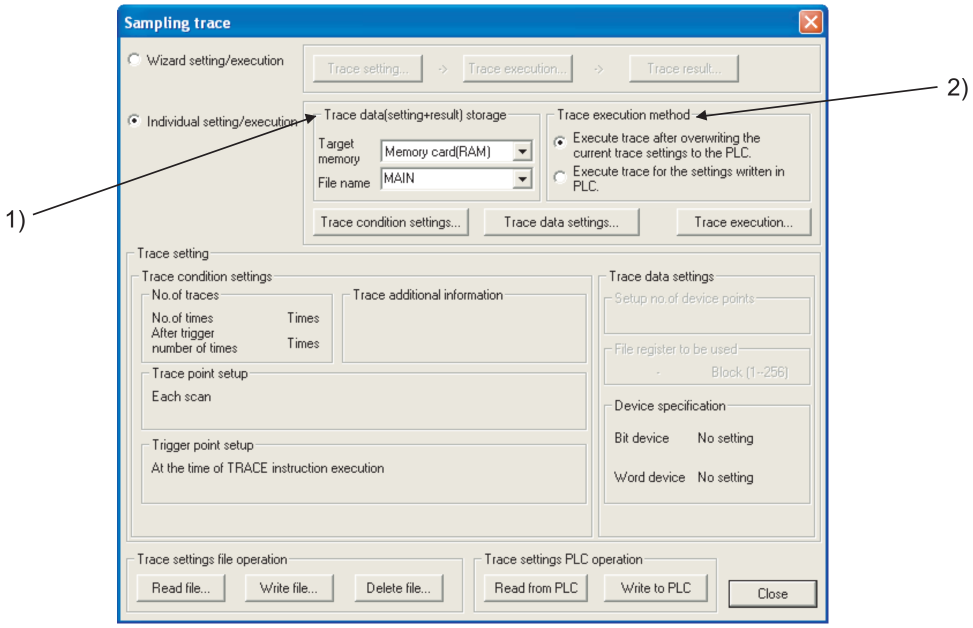
Select [Online] → [Trace] → [Sampling trace...] in GX Developer.

On the screen opened, select the method for operating the sampling trace.

- "Wizard setting/execution"
  (☞ GX Developer Version 8 Operating Manual)
- "Individual setting/execution"
  (☞ (5)(a) in this section)

(a) Setting "Trace data (setting + result) storage" and "Trace execution method"

On the screen opened, set the trace data storage location and trace execution method.



Sampling trace

○ Wizard setting/execution

● Individual setting/execution

Trace data(setting+result) storage

Target memory: Memory card(RAM)

File name: MAIN

Trace execution method

● Execute trace after overwriting the current trace settings to the PLC.

○ Execute trace for the settings written in PLC.

Trace setting

Trace condition settings

No.of traces

No.of times — Times

After trigger number of times — Times

Trace additional information

Trace point setup: Each scan

Trigger point setup: At the time of TRACE instruction execution

Trace data settings

Setup no.of device points

File register to be used: - Block (1~256)

Device specification

Bit device: No setting

Word device: No setting

Figure 6.59 Sampling trace screen

1) Trace data (setting + result) storage

Select the memory for storing the trace data and the file for writing the trace conditions.

Select either "Standard RAM" or "Memory card (RAM)" for the target memory setting.

Trace results will be stored in the memory set here under the selected file name.

2) Trace execution method

Select either of the following trace execution method.

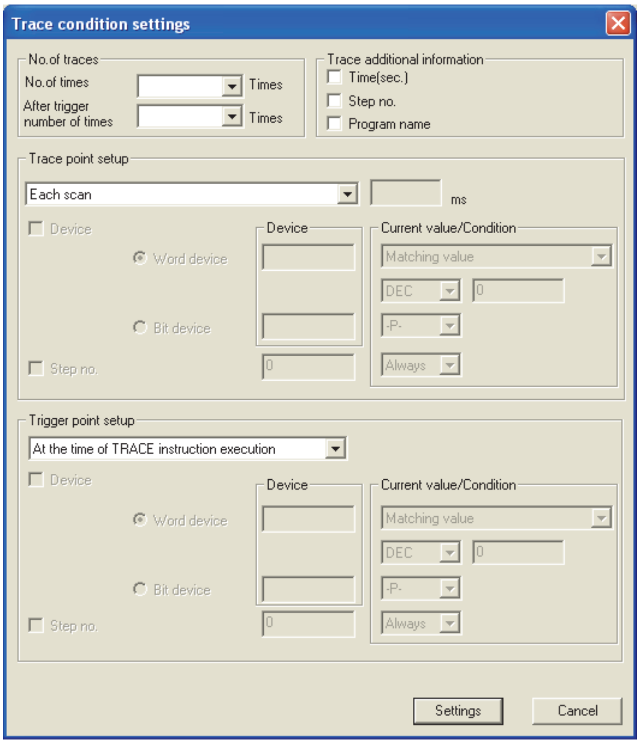
- "Execute trace after overwriting the current trace setting to the PLC":
  The CPU module executes the sampling trace after the trace settings are overwritten to the existing sampling trace file.
- "Execute trace for the settings written in PLC":
  The CPU module executes the sampling trace with the trace settings in the sampling trace file selected for "Trace data (setting + result) storage".

(b) Setting trace conditions

Set trace conditions on the screen opened by clicking the Trace condition setting button on the screen shown in Figure 6.59.

On the Trace condition settings screen, set the following items.

- Number of traces ("No. of traces", "After trigger number of times")
- Trace point setting ("Trace point setup")
- Trigger point setting ("Trigger point setup")
- Additional information ("Trace additional information")
- Auto start setting ("Auto start trace")



Trace condition settings

No.of traces

No.of times [ ] Times

After trigger number of times [ ] Times

Trace additional information

□ Time(sec.)

□ Step no.

□ Program name

Trace point setup

Each scan [ ] ms

□ Device

Device

Current value/Condition

● Word device [ ]

○ Bit device [ ]

Matching value

DEC 0

-P-

□ Step no. 0

Always

Trigger point setup

At the time of TRACE instruction execution

□ Device

Device

Current value/Condition

● Word device [ ]

○ Bit device [ ]

Matching value

DEC 0

-P-

□ Step no. 0

Always

Settings Cancel

Figure 6.60 Trace condition settings screen

1) No. of traces

There are two items need to be set: "No. of times" and "After trigger number of times".

- No. of times: Select the number of executions from the start to the end of sampling trace.
- After trigger number of times: Select the number of executions from the trigger point to the end of sampling trace.

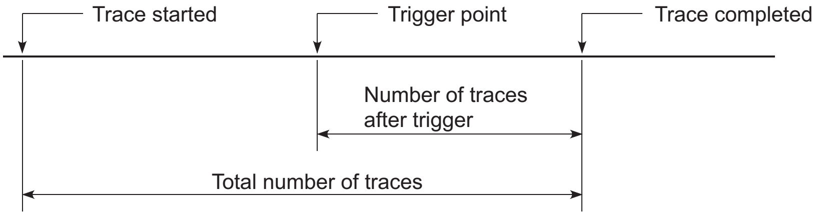


Figure 6.61 Relationship between two setting items

Set the numbers for each items within the following setting range.

("After trigger number of times")≦("No. of times")≦(8192)

2) Trace point setup

Select the timing for collecting trace data from the items listed in Table6.21.

Table6.21 Trace point setup item

<table>
<tr><th>Item</th><th>Description</th></tr>
<tr><td>Each scan</td><td>Collects trace data during END processing of each scan.</td></tr>
<tr><td>Interval</td><td>Collects trace data at specified time intervals.</td></tr>
<tr><td>Detail</td><td>A trace point (device and/or step number) needs to be set.
The following devices can be set as a trace point.
• Bit device: X(DX), Y(DY), M, L, F, SM, V, B, SB, T(contact), ST(contact), C(contact), FX, FY, J□\X, J□\Y, J□\B, J□\SB, BL□\S
• Word device: T(current value), ST(current value), C(current value), D, SD, W, SW, R, Z, ZR, FD, U□\G, J□\W, J□\SW

The following modifications are available for the above devices.
• Digit specification of bit device
• Bit specification of word device
• Indirect specification of word device
• Index modification

When the set conditions are met, data collection is performed. The following shows the conditions of setting items.
<table>
<tr><th>Device</th><th>Conditional formula</th><th>Description</th></tr>
<tr><td rowspan="2">Bit device</td><td>↑</td><td>Data are collected on the rising edge of the specified device/label.</td></tr>
<tr><td>↓</td><td>Data are collected on the falling edge of the specified device/label.</td></tr>
<tr><td rowspan="2">Word device</td><td>When values match</td><td>Data are collected when the current value of the specified device/label is equal to the condition value.</td></tr>
<tr><td>When values are changed</td><td>Data are collected when the current value of the specified device/label is changed.</td></tr>
</table>
The collection timing for trace data when "Step No." is selected is the same as when setting the monitor conditions. (☞ Section 6.11.1)</td></tr>
</table>

3) Trace additional information

Set the information added for each trace.

Select one or more items from the following. (If not necessary, do not select any item.)

- Time(sec.): Stores the time when the trace was executed.
- Step no.: Stores the step number where the trace was executed.
- Program name: Stores the program name where the trace was executed.

4) Trigger point setup

Select the trigger point from the items listed in Table6.22.

Table6.22 Trigger point setup item

<table>
<tr><th>Item</th><th>Description</th></tr>
<tr><td>At the time of TRACE instruction execution</td><td>The time of execution of the TRACE instruction is set as a trigger.</td></tr>
<tr><td>At the time of trigger operation
from GX Developer</td><td>The time when a trigger is issued by GX Developer is set as a trigger.</td></tr>
<tr><td>Detail</td><td>A trace point (device and/or step number) needs to be set.
The following devices can be set as a trace point.
• Bit device: X(DX), Y(DY), M, L, F, SM, V, B, SB, T(contact), ST(contact), C(contact), FX, FY
• Word device: T(current value), ST(current value), C(current value), D, SD, W, SW, R, ZR

The following modification is available for the above devices.
• Bit specification of word device

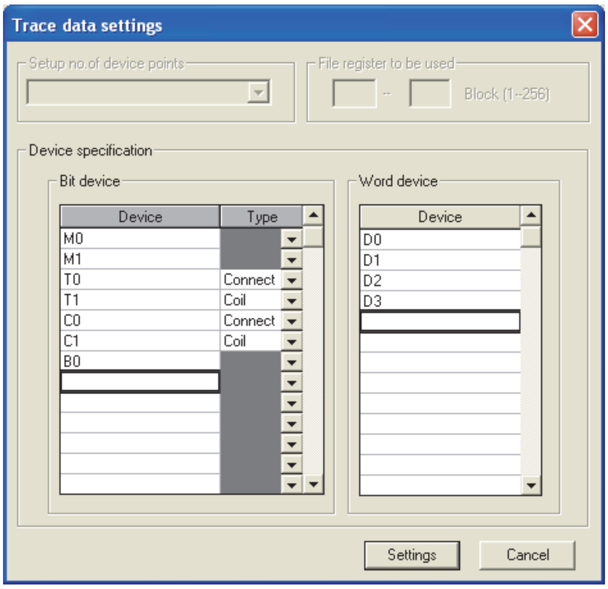
Indirectly-specified devices cannot be set.

A trigger point is set as the timing of when the set conditions are met. The following shows the conditions of setting items.
<table>
<tr><th>Device</th><th>Conditional formula</th><th>Description</th></tr>
<tr><td rowspan="2">Bit device</td><td>↑</td><td>A trigger occurs at the rising edge of the specified device/label.</td></tr>
<tr><td>↓</td><td>A trigger occurs at the falling edge of the specified device/label.</td></tr>
<tr><td rowspan="2">Word device</td><td>When values match</td><td>A trigger occurs when the current value of the specified device/ label is equal to the condition value.</td></tr>
<tr><td>When writing into devices</td><td>A trigger occurs when a value is written into the specified device/ label.</td></tr>
</table>
The collection timing for trace data when "Step No." is selected is the same as when setting the monitor conditions.
(☞ Section 6.11.1)</td></tr>
</table>

(c) Setting trace data

Set trace data on the screen opened by clicking the Trace data setting button on the screen shown in Figure 6.59.

Table6.23 shows the devices can be set as trace data.



Device	Type
M0	
M1	
T0	Connect
T1	Coil
C0	Connect
C1	Coil
B0	

Device
D0
D1
D2
D3

Figure 6.62 Trace data settings screen

Table6.23 Devices can be set as trace data

Item	Description
Bit device	The following bit devices can be set up to 50 points. X, DX, Y, DY, M, L, F, SM, V, B, SB, T(contact), T(coil), ST(contact), ST(coil), C(contact), C(coil), J□X, J□\Y, J□\B, J□\SB, BL□\S
Word device	The following word devices can be set up to 50 points. T (current value), ST (current value), C (current value), D*1, SD, W*2,SW, R, Z, ZR, U□\G, J□\W, J□\SW The following modifications are available for the above devices. • Digit specification of bit device • Bit specification of word device • Index modification Indirectly-specified devices cannot be set.

(d) Writing the trace condition settings and trace data settings

Write the set trace conditions and trace data to the memory selected as a sampling trace file for "Trace data (setting + result) storage".

Click the Write to PLC button on the screen shown in Figure 6.59 to write the settings.

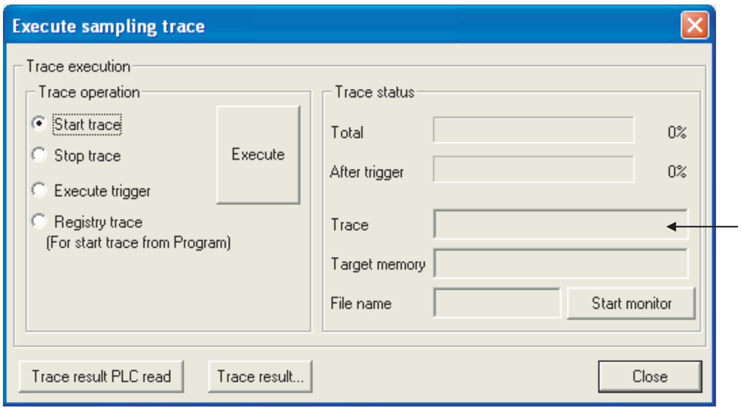
Point

When storing the sampling trace file into a memory card (SRAM card), more than one sampling trace files can be stored by changing the file name.
For the standard RAM, only one sampling trace file can be stored.
When multiple sampling trace files are used, use the memory card (SRAM card).

(e) Executing the sampling trace

Click the Trace execution button on the screen shown in Figure 6.59 to open the Execute sampling trace screen.

Select an item shown in Table6.24 and click the Execute button.



Execute sampling trace

Trace execution

Trace operation

● Start trace
○ Stop trace
○ Execute trigger
○ Registry trace
(For start trace from Program)

Trace status

Total: 0%
After trigger: 0%
Trace:
Target memory:
File name:

Displayed only when status display is selected

Figure 6.63 Execute sampling trace screen

Table6.24 Trace point setting

Item	Description
Start trace	Starts the function, and starts counting the number of sampling trace executions.
Stop trace	Stops the function, and clears the total sampling trace execution count and the execution count after trigger. (To restart the function, select "Start trace" again.)
Execute trigger	Executes a trigger, and starts counting the number of sampling trace executions after trigger. The function will be ended when the trace execution count after trigger reaches the preset count.
Registry trace (For start trace from Program)	Registers trace data when a program is executed.

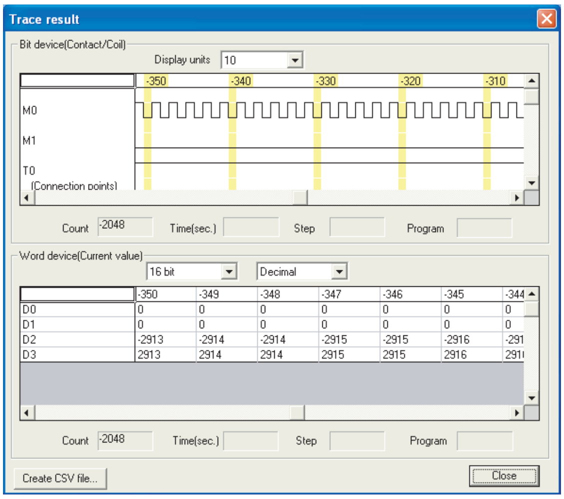
(f) Displaying trace results

Read trace results form the CPU module and display the data.

1) Click the [Trace result PLC read] button on the screen shown in Figure 6.63 to read trace results.

2) Click the [Trace result] button on the same screen to display the trace results read.

The trace results shows the on/off status of each bit device for every sampling cycle and the current value of each word device.



Trace result

Bit device(Contact/Coil)

Display units: 10

Count: -2048 Time(sec.) Step Program

Word device(Current value)

16 bit Decimal

	-350	-349	-348	-347	-346	-345	-344
D0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0
D2	-2913	-2914	-2914	-2915	-2915	-2916	-291
D3	2913	2914	2914	2915	2915	2916	291

Count: -2048 Time(sec.) Step Program

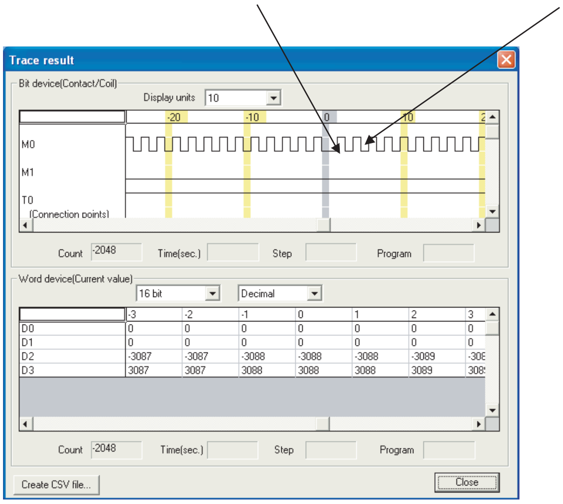
Figure 6.64 Trace result screen

Point

Specified devices are read when the condition selected in "Trigger point setup" (trigger condition) is established. Therefore, when devices are sampled in every scan and the sampling is finished by trigger operation from a peripheral, the data is sampled twice because timing of the sampling is the same with that of the establishment of trigger condition.

Data when the trigger conditions are satisfied.

Data when trigger condition is met



Trace result

Bit device(Contact/Coil)

Display units: 10

Count: -2048 Time(sec.) Step Program

Word device(Current value)

16 bit Decimal

	-3	-2	-1	0	1	2	3
D0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0
D2	-3087	-3087	-3088	-3088	-3088	-3089	-308
D3	3087	3087	3088	3088	3088	3089	308

Count: -2048 Time(sec.) Step Program

Figure 6.65 Trace result

(6) Method for clearing trace execution status

The trace execution status can be cleared by latch clear using the RESET/L.CLR switch or the remote latch clear operation. (☞ Section 6.6.4)
To perform the sampling trace again after latch clear, select "Start trace" or "Registry trace".

(7) 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, 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 with GX Developer.

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.

Point

To keep the trace result to the personal computer even after configuring the setting that changes the local device size, perform the following operations.

- Click the Trace result PLC read button on the screen shown in Figure 6.63 to read the trace result to the personal computer.
- Click the Trace result button on the screen shown in Figure 6.63 to display the trace result.
- Click the Create CSV file button on the screen shown in Figure 6.64 to save the trace result in CSV format.

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) Reading trace result in the STOP status

The trace result cannot be read while the CPU module is in the STOP status.
When reading the trace result, read it while the CPU module is in the RUN status.

(d) Sampling trace registration while the trigger condition is established

When registering the sampling trace setting, the trigger condition set for the trigger point must not to be established.
If the condition is established, the setting cannot be registered.

(e) When a file register is selected as a specified device by the detail setting of trace conditions

When a file register is selected as a specified device by the detail setting of trace point setting and trigger point setting, do not change the block numbers of file register file and file register after trace registration.
Trace data may not be normally sampled.

(f) Trace point setting

When setting the trace point setting per each time, pay attention to the sampling interval and sampling processing time for one sampling since the sampling trace is performed as interrupt processing.
If the sampling processing time for one sampling is long, "WDT ERROR" may be detected.

(g) Performing sampling trace during execution of another sampling trace

The first sampling trace is performed normally.
The second sampling trace cannot be performed.

(h) Executing online change

When sampling trace and online change are performed simultaneously, they operate as follows.

1) Performing sampling trace during online change

- 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.
- The trace point and trigger point are specified by except the step number:
  Both the online change and sampling trace can be performed.

2) Performing online change during execution of sampling trace

- The trace point or trigger point is specified by the step number:
  The sampling trace is suspended but the online change is normally performed.
- The trace point and trigger point are specified by except the step number:
  Both the online change and sampling trace can be performed.

6.15 Debug Function from Multiple GX Developers

(1) Definition

This function allows debugs from multiple GX Developers connected to such as a CPU module or serial communication module.

When files are divided according to the processes or functions, this function can be used when multiple GX Developers debug different files.

(2) Description

Table6.25 shows combinations of the debug functions executable from multiple GX Developers.

Table6.25 Combinations of the debug functions

Function in execution	Function executed later			
	Monitor	Online change	Execution time measurement	Sampling trace
Monitor	○*1	○*2	○	○
Online change	○*2	×*3	×	×
Scan measurement	○	○	×	○
Sampling trace	○	×	○	×

○: Can be executed simultaneously. × : Can be executed from one GX Developer.

*1: Since only one GX Developer can set the monitor conditions (☞ Section 6.11.1), other GX Developers cannot set them.

*2: The monitoring with monitor conditions and online change cannot be performed simultaneously.

*3: For how to perform online change to a file from multiple GX Developers, refer to Section 6.15.2.

6.15.1 Simultaneous monitoring from multiple GX Developers function

(1) Definition

This function allows simultaneous monitoring from multiple GX Developers connected to such as a CPU module or serial communication module.

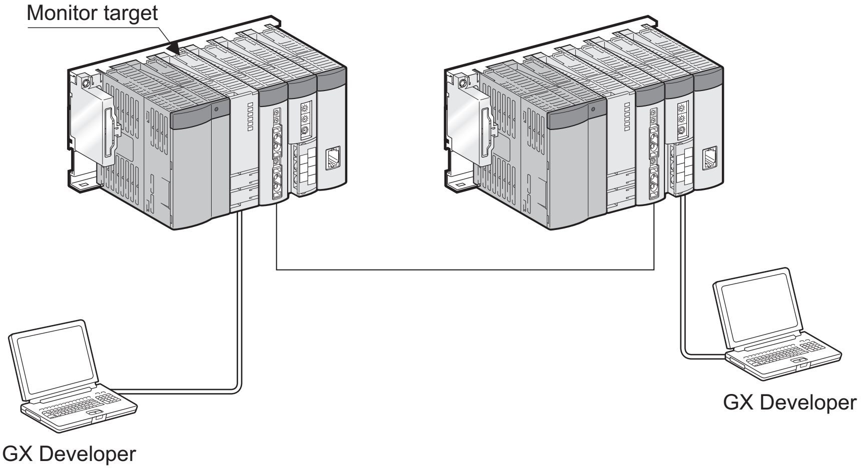


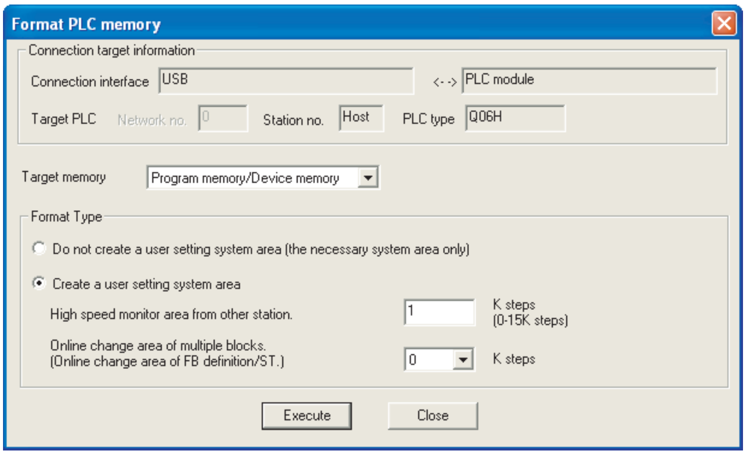
Figure 6.66 Simultaneous monitoring

Creating a user setting system area allows high-speed monitoring from multiple GX Developers (Setting a monitoring file for the host station is unnecessary).

(2) Setting for simultaneous monitoring from multiple GX Developers

Create a user setting system area in the following procedure.

- Select [Online] → [Format PLC memory] in GX Developer to open the screen shown in Figure 6.67.
- Select "Program memory/Device memory" in "Target Memory".
- Select "Create a user setting system area" in "Format Type".
- Set the number of steps for the system area (in increments of 1K step).



Format PLC memory

Connection target information

Connection interface: USB <-> PLC module

Target PLC   Network no. 0   Station no. Host   PLC type Q06H

Target memory: Program memory/Device memory

Format Type

○ Do not create a user setting system area (the necessary system area only)

● Create a user setting system area

High speed monitor area from other station. 1 K steps (0-15K steps)

Online change area of multiple blocks.
(Online change area of FB definition/ST.) 0 K steps

Figure 6.67 System area setting (when 1K step is set)

Table6.26 shows the maximum number of steps settable in the system area.
1K step is available for a monitoring file from another station.

Table6.26 Maximum size of steps settable in the system area

CPU module	Maximum size of settable step	System area for monitoring from another station
Basic model QCPU	Maximum 3K steps	Maximum 3
High Performance model QCPU, Process CPU, Redundant CPU	Maximum 15K steps	Maximum 15

(3) Precautions

(a) Monitor condition setting

The monitor conditions can be set from one GX Developer. Note6.16

(b) Necessity of system area setting

Although multiple GX Developers in other stations can simultaneously monitor a CPU module without the user setting system area, the monitor speed will be slow.
Since the system area is set in the program memory, the area for storing programs reduces by the size of set system area.

(c) The number of GX Developers for which high-speed monitoring can be set

The number of GX Developers that can simultaneously monitor a CPU module at high-speed is "the number of user setting system areas (the number of K steps) + 1".
For example, when user setting system area of 15K steps is created, maximum 16 GX Developers can simultaneously monitor a CPU module at high-speed.

6.15.2 Online change function from multiple GX Developers Note6.17

(1) Definition

This function allows multiple GX Developers to perform online change to one file or different files

- Online change to one file:
  Select "Relative step No. by pointer".
- Online change to different files:
  The writing can be executed without selecting "Relative step No. by pointer".

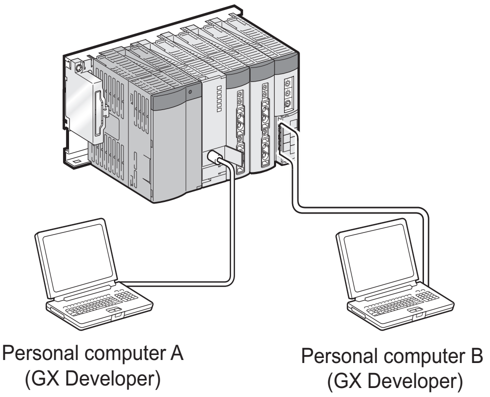


Figure 6.68 Simultaneous online change from multiple GX Developers

Note6.16 Basic

The Basic model QCPU does not support monitoring with monitor condition setting.

Note6.17 Basic

When multiple GX Developers write to one file in the Basic model QCPU in the RUN status, check the versions of the CPU module and GX Developers. (☞ Appendix 2.1)

(2) Operating procedure for performing online change to one file

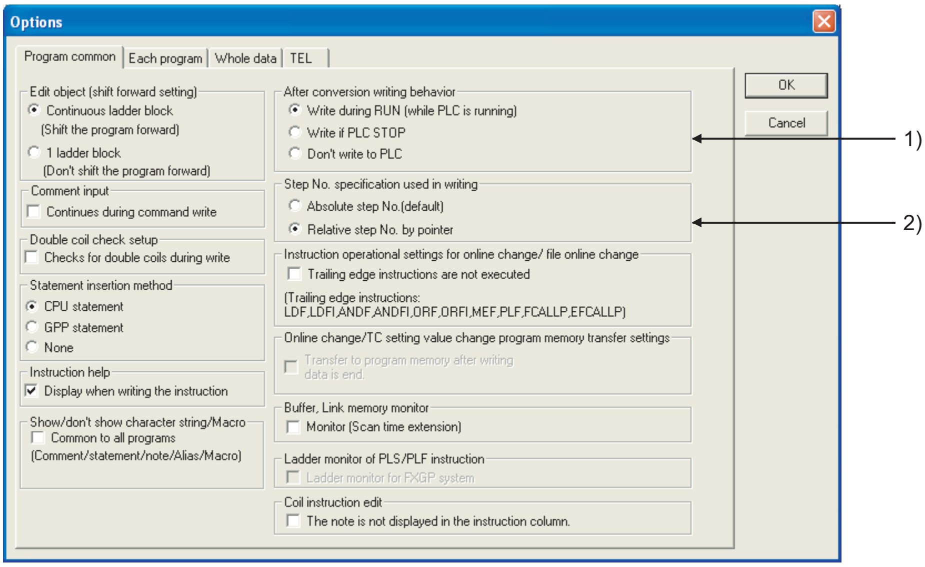
Select [Tools] → [Options] → <Program common> tab in GX Developer.
Set a pointer for Write during RUN beforehand.

(a) Setting "After conversion writing behavior" and "Step No. specification used in writing"

Set them as follows:

1) Select "Write during RUN (while PLC is running)" in "After conversion writing behavior".

2) Select "Absolute step No. (default)" or "Relative step No. by pointer" in "Step No. specification used in writing".



Edit object (shift forward setting)
● Continuous ladder block
(Shift the program forward)
○ 1 ladder block
(Don't shift the program forward)

Comment input
□ Continues during command write

Double coil check setup
□ Checks for double coils during write

Statement insertion method
● CPU statement
○ GPP statement
○ None

Instruction help
☑ Display when writing the instruction

Show/don't show character string/Macro
□ Common to all programs
(Comment/statement/note/Alias/Macro)

After conversion writing behavior
● Write during RUN (while PLC is running)
○ Write if PLC STOP
○ Don't write to PLC

Step No. specification used in writing
○ Absolute step No.(default)
● Relative step No. by pointer

Instruction operational settings for online change/ file online change
□ Trailing edge instructions are not executed
(Trailing edge instructions:
LDF,LDFI,ANDF,ANDFI,ORF,ORFI,MEF,PLF,FCALLP,EFCALLP)

Online change/TC setting value change program memory transfer settings
□ Transfer to program memory after writing data is end.

Buffer, Link memory monitor
□ Monitor (Scan time extension)

Ladder monitor of PLS/PLF instruction
□ Ladder monitor for FXGP system

Coil instruction edit
□ The note is not displayed in the instruction column.

Figure 6.69 Options screen

(b) Performing online change

Display the ladder including the specified pointer and write the changed ladder during RUN.

(3) Precautions

Precautions for online change from multiple GX Developers are the same as those for usual online change.
(☞ Section 6.12.3)

6.16 Watchdog Timer (WDT)

(1) Definition

This function serves as an CPU module internal timer to detect errors of CPU module hardware and sequence programs.

(2) 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 200 ms.
The setting range is 10 to 2000 ms (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.

(3) When the watchdog timer times up

"WDT ERROR" is detected and the following status occurs:

1) The CPU module turns off all outputs.

2) The RUN LED on the front of the CPU module turns off and the ERR. LED starts flashing.

3) SM1 turns on and the error codes 5000 and 5001 are stored in SD0.

(4) 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.

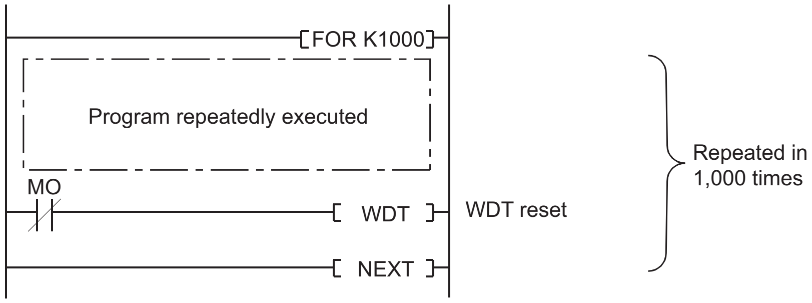


Figure 6.70 Resetting a watchdog timer when the program is executed between the FOR and NEXT instructions

(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 value is measured up to the END instruction.

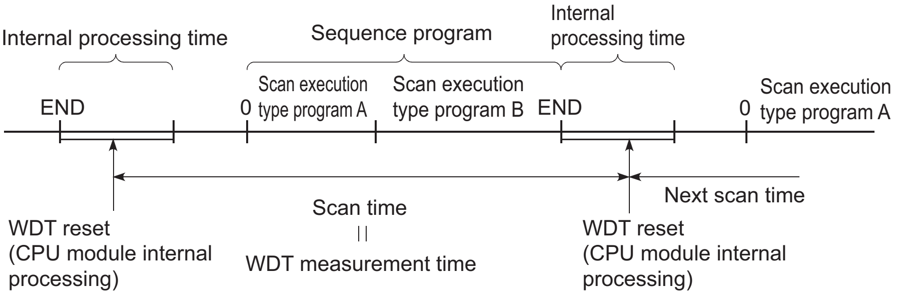

Internal processing time
Sequence program
Internal processing time
END
0
Scan execution type program A
Scan execution type program B
END
0
Scan execution type program A
Scan time
||
WDT measurement time
Next scan time
WDT reset (CPU module internal processing)
WDT reset (CPU module internal processing)

Figure 6.71 Watchdog timer reset

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 keep the same scan time in every scan, use the constant scan function. (☞ Section 6.2)

6.17 Self-diagnostic Function

(1) Definition

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.

(2) 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.

(3) 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) are 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.

Point

● When the Redundant CPU is used, the details of an error occurred in the other system are stored in the special relays (SM1610 to SM1626) and special registers (SD1610 to SD1636) except when:
  - the other system is during off, reset, or hardware failure,
  - "WDT ERROR" (error code: 5000, 5001) is detected, or
  - tracking cable has an error (not connected, disconnection, or failure).

● The details of the following errors that indicate the CPU module status are not stored in the special relays (SM0 and SM1) and special registers (SD0 to SD26), and the ERR. LED does not turn on.
The error details are stored in the error history (☞ Section 6.18).
  - System switching from standby system to control system... "CONTROL EXE." (error code: 6200)
  - System switching from control system to standby system... "STANDBY" (error code: 6210)

(4) Checking error history

To check the latest error code, select [Diagnostics] → [PLC diagnostics] → "Error log" in GX Developer.
The error history data are backed up using a battery even after the programmable controller is powered off.

(5) 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.

1) 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).

2) 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.

1) 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 Note6.18
- Memory card operation error Note6.18
- External power supply OFF (setting for future extension) Note6.18

For example, 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". (☞ (7) in this section)

2) 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

Note6.18 **Basic**

The file access error, memory card operation error, and turn-off of external power supply cannot be selected for the Basic model QCPU.

(6) 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).

1) Carry out battery check

2) Carry out fuse blown check

3) Verify module

(7) Self-diagnostics list

The following table shows the self-diagnostics performed by the CPU module.

To check the error messages in the "Error message" column of Table6.27, select [Diagnostics] → [PLC diagnostics] of GX Developer.

Table6.27 Self-diagnostics list

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
					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	○	○	○	○
	Tracking memory and tracking hardware error	TRK. CIR. ERROR	• Power-on/reset • During operation	Stop	Off	Flashing	×	×	×	○
	Operation circuit check	OPE.CIR-CUIT 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	○	○	○	○
	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	○	○	○	○
	Voltage drop of power supply for redundant base unit	SINGLE PS. DOWN	• Always	Continue	On	On	×	○ *3	○ *3	○

○: Self-diagnostics is performed, ×: Self-diagnostics is not performed

(To the next page)

Table6.27 Self-diagnostics list (continued)

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
					RUN	ERR.				
Hardware failure	Redundant power supply module failure	SINGLE PS. ERROR	• Always	Continue	On	On	×	○ *3	○ *3	○
	Battery low*2	BATTERY ERROR	• Always	Continue	On	BAT.ALM LED 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	○	○ *4	○	○
	Intelligent function module version error	SP.UNIT VER.ERR	• Power-on/reset	Stop	Off	Flashing	○ *4	×	○	○
	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	○	○	○	○
	Memory card operation error*1	ICM.OPE. ERROR	• Mounting/removal of the memory card	Stop/ Continue	Off/ On	Flashing/ 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	○	○	○	○
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	○	○	○	○

○: Self-diagnostics is performed, ×: Self-diagnostics is not performed

(To the next page)

Table6.27 Self-diagnostics list (continued)

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
					RUN	ERR.				
Parame-ter error	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 RUN	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	○	○	○	○
Program error	Operation error*1 *3	OPERA-TION 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	○	○	○	○
	SFC program structure error	SFCP.CODE ERROR	• Switching from STOP to RUN	Stop	Off	Flashing	×	○	○	○
	SFC block configuration error	CAN'T SET(BL)	• Switching from STOP to RUN	Stop	Off	Flashing	○	○	○	○

○: Self-diagnostics is performed, ×: Self-diagnostics is not performed

(To the next page)

Table6.27 Self-diagnostics list (continued)

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
					RUN	ERR.				
Program error	SFC step configuration error	CAN'T 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 operation check error[*1]	SFCP.OPE. ERROR	• Execution of an instruction	Stop/ Continue	Off/ On	Flashing/ On	×	○	○	○
	SFC program execution error	SFCP.EXE. ERROR	• Switching from STOP to RUN	Continue	On	On	×	○	○	○
	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	○	○	○	○
Redundant system error	Program, parameter, or DIP switch mismatch	FILE DIFF.	• Always • Power-on/reset • Mounting of tracking cable • Backup mode change • Online change • System switching • Switching the both CPU modules to the RUN status	Stop	Off	Flashing	×	×	×	○
	Operating status or key switch mismatch	OPE.MODE DIFF.	• Power-on/reset • Always	Continue /stop	Off/ on	Flashing /on	×	×	×	○
	Module mounting configuration mismatch	UNIT LAY. DIFF.	• Always • Power-on/reset • Mounting of tracking cable • Operation mode change	Stop	Off	Flashing	×	×	×	○
	Memory card mounting status mismatch	CARD TYPE DIFF.	• Power-on/reset	Stop	Off	Flashing	×	×	×	○
	Function disable during current mode	CAN'T EXE. MODE	• Always	Continue	On	On	×	×	×	○

○: Self-diagnostics is performed, ×: Self-diagnostics is not performed

(To the next page)

Table6.27 Self-diagnostics list (continued)

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
					RUN	ERR.				
Redun-dant sys-tem error	Both systems file consistency error Parameter-valid drive consistency error	CPU MODE DIFF.	• Power-on/reset • Execution of the END instruction • Mounting of tracking cable	Stop	Off	Flashing	×	×	×	○
	Tracking data communication error	TRK.TRANS. ERR.	• Always	Continue	On	On	×	×	×	○
	Tracking capacity excess error	TRK.SIZE ERROR	• Execution of the END instruction	Continue	On	On	×	×	×	○
	Tracking cable error or tracking transfer hardware failure	TRK.CABLE ERR.	• Power-on/reset	Stop	Off	Flashing	×	×	×	○
	Tracking cable not connected, failure, or tracking transfer hardware failure	TRK. DISCON-NECT	• Always	Continue	On	On	×	×	×	○
	Tracking transfer initial error	TRK.INIT. ERROR	• Power-on/reset	Stop	Off	Flashing	×	×	×	○
	System switching from standby system to control system*5	CONTROL EXE.	• Always	Continue	On	Off	×	×	×	○
	System switching from control system to standby system*5	STANDBY	• Always	Continue	On	Off	×	×	×	○
	System switching error	CAN'T SWITCH	• System switching	Continue	On	On	×	×	×	○
	Standby system not started or stop error	STANDBY SYS.DOWN	• Always	Continue	On	On	×	×	×	○
	Control system not started or stop error	CONTROL SYS.DOWN	• Always	Stop	Off	Flashing	×	×	×	○
	Program memory clear	PRG.MEM. CLEAR	• Execution of the program memory copy function	Stop	Off	Flashing	×	×	×	○

○: Self-diagnostics is performed, ×: Self-diagnostics is not performed

(To the next page)

Table6.27 Self-diagnostics list (continued)

Diagnostics		Error message	Diagnostic timing	CPU module status	LED status		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
					RUN	ERR.				
Redundant system error	Memory copy function execution	MEM.COPY EXE.	• Execution of the memory copy function	Continue	On	On	×	×	×	○
	Tracking setting parameter error	TRK.PARA. ERROR	• Power-on/reset	Stop	Off	Flashing	×	×	×	○
	Multiple CPU system not configurable	CPU LAY ERROR	• Power-on/reset	Stop	Off	Flashing	×	×	×	○
Multiple CPU system 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	○	○	○	×
Annunciator check		F****	• Execution of an instruction	Continue	On	USER LED turns on.	○	○	○	○
CHK instruction check		<CHK>ERR ***-***	• Execution of an instruction	Continue	On	USER LED turns on.	×	○	○	○
Boot OK		BOOT OK	• Power-on/reset	Stop	Off	Flashing	×	○	○	○

○: Self-diagnostics is performed, ×: Self-diagnostics is not performed

*1: The operation status can be changed to "Continue" with the parameter setting in GX Developer. (The default is set to "Stop".)

*2: This option can be set "not checked" (set "checked" by default) with the parameter setting of GX Developer.

*3: In multiple CPU systems, when the serial numbers of all the CPU modules (first five digits) are "07032" or later, this error is detected from the CPU No.1 only.

*4: The CPU module function version B or later supports the detection of this error.

*5: Since this diagnostics indicates CPU module status, the error message is not displayed in the "Current Error" field on the PLC diagnostics screen of GX Developer.
This error is displayed in the Error history field only.

Point

When the Redundant CPU is used, the details of an error occurred in the other system are stored in the special relays (SM1610 to SM1626) and special registers (SD1610 to SD1636) except when:

- the other system is during off, reset, or hardware failure,
- "WDT ERROR" (error code: 5000, 5001) is detected, or
- tracking cable has an error (not connected, disconnection, or failure).

6.17.1 Interrupt caused by an error Note6.19

The CPU module can execute an interrupt program for the interrupt pointer of the error occurred.

(1) Interrupt by an error that can be set to "Continue" or "Stop" in the PLC RAS tab

For an error that is set to "Continue" in the PLC RAS tab of the PLC parameter dialog box, an interrupt can be executed only for the error.
For an error that is set to "Stop", an interrupt program for all of stop errors (I32) is executed.

(2) Errors corresponding to interrupt pointers Note6.20

Figure 6.72 shows the errors corresponding to respective interrupt pointers.

Interrupt pointer	Error message
I32	All of stop errors[*1]
I33	SINGLE PS.DOWN[*2]
I34	UNI VERIF ER FUS BREA OFF SP.UNI ERRO MULTI CPU ERROR
I35	OPERATI ERRO SFC OPE.ERR SFC EXE.ERRO EX.POWER
I36	ICM.OPE.ERR FIL OPE.ERR
I37	N/A
I38	PRG.TI OVE
I39	CHK instruction Annunciator detection
I40	CAN' SWIT
I41	STAND

Errors for which the operation mode after the error is set to "Continue", or errors that are set to "Continue" from selection of "Stop/Continue"

Figure 6.72 List of the error-interrupt pointers

*1: If any of the following serious errors occurs, the interrupt program of I32 is not executed.
- MAIN CPU DOWN
- END NOT EXECUTE
- RAM ERROR
- OPE CIRCUIT ERR.

*2: In the case of a multiple CPU system that includes any High Performance model QCPU or Process CPU, this is applicable to CPU No.1 only.

Note6.19 **Basic**

Since no interrupt pointers for errors are provided for the Basic model QCPU, even if an error due to a fault of the CPU module itself occurs, no interrupt can be executed.

Note6.20 **High performance** **Process**

For the High Performance model QCPU and Process CPU, interrupt pointers I40 and I41 cannot be used.
Also, I33 is applicable to the module whose serial number (first five digits) is "07032" or later.

Point

Execution of interrupt pointers, I32 to I48, are disabled when the system is powered on or when the CPU module is reset.
To use any of I32 to I41, enable the interrupt with the IMASK and EI instructions.
For details of the IMASK and EI instructions, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

(3) Precautions when using the Redundant CPU

(a) When using the interrupt program of interrupt pointer I41

I41 is an interrupt pointer that is used when the control system is switched to the standby system.
Note that the interrupt program of I41 is executed in the new standby system (the former control system) after the system switching. Therefore, pay attention to the following.

1) Control system judgment flag and Standby system judgment flag in the special relay (SM1515 and SM1516)

The Control system judgment flag and Standby system judgment flag indicate the standby system (SM1515: OFF, SM1516: ON).

2) When changing tracking target devices

When change of tracking target devices has been programmed for the interrupt program of I41, devices of the new standby system are overwritten with device data of the new control system (which is the system switched from the standby system) by tracking transfer.

To change the tracking transfer devices in the interrupt program of I41, perform the following.

- In the Tracking settings tab of the Redundant parameter dialog box (☞ Section 8.2(2)), remove the devices from the tracking transfer target.
- Create a program so that the tracking target device data will be transferred to other devices for the change.

3) Outputs to the modules on the main base unit and network modules

Because the interrupt program of I41 is executed in the new standby system, the following outputs are not performed.

- Output (Y) to the modules mounted on the main base unit
- Output (Y) to the MELSECNET/H remote I/O network
- Transmission from the link relay (B) and link register (W) to other stations on the MELSECNET/H remote I/O network and MELSECNET/H PLC to PLC network

6.17.2LEDs indicating errors

When an error occurs, the LEDs on the front of the CPU module turns on/flashes. (☞ Section 6.21)

6.17.3 Error clear

The CPU module can clear an error by a program if the error does not stop program operation.

(1) Procedures for error clear

Clear an error by the following procedures.

- Resolve the error cause.
- Store the code of the error to be cleared in the special register SD50.
- Turn off and then on the special relay SM50.
- The error is cleared.

(a) Procedures for error clear in case of multiple errors

When the latest error (error stored in the special register SD0) is cleared, error information stored in special relays and special registers (SM0, SM1, SM5, SM16, SD0 to SD26) are cleared and therefore information on errors that have not been cleared cannot be obtained from the special relays and special registers.
For the errors that have not been cleared, obtain the past errors from the error history.
(☞ Section 6.18) and clear the errors.

(2) Status after error clear

When the CPU module is recovered by clearing the error, the special relay, special register, and LEDs affected by the error return to the status before the error.
If the same error occurs after clearing the error, it is logged in the error history again.

(3) Clear of annunciator

When multiple annunciators are detected, only the first detected "F" is cleared. (☞ Section 9.2.5)

Point

● When an error is cleared by storing the code of the error to be cleared in SD50, the last digit of the code number is ignored.

Example When the error code 2410, 2411, or 2412 occurs, clearing the error by storing 2412 in SD50 also clears the error codes 2410 and 2411.

● If the CPU module is not an error cause, the error cannot be resolved by using the special relay (SM50) and special register (SD50).

Example Since "SP. UNIT DOWN" indicates an error occurred to the Q bus, the error cannot be resolved by using the special relay (SM50) and special register (SD50).

To resolve the error cause, refer to the following.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspecting)

6.18 Error History

This function stores an error detected by the self-diagnostic function and the detection time as an error history in a memory.
Select [Diagnostics] → [PLC diagnostics] of GX Developer to check the history.

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.
(☞ Section 6.5)

6.18.1Basic model QCPU

(1) Storage area

16 latest error logs are stored in the latched error history storage memory of the Basic model QCPU.

(2) Storage data

If the same error occurs several times while the CPU module is on, the error logs are stored in the error history storage memory once.

(3) How to clear error history

To clear the storage memory for error history, select [Diagnostics] → [PLC diagnostics] in GX Developer and click the [Clear log] button.
This method clears all data stored in the storage memory for error history of the Basic model QCPU.

6.18.2 High Performance model QCPU, Process CPU, and Redundant CPU

(1) Storage area

16 latest error logs are stored in the latched error history storage memory of the CPU module.
When more than 16 error logs are stored, the logs can be stored in a file in a memory card by the setting in the PLC RAS tab of the PLC parameter dialog box.

Table6.28 Storage area for error history file

Storage area	Number of storable logs
Built-in memory and a file in set memory card	Up to 100 (can be changed.)[*1]

*1: When the number of storable logs are exceeded, the latest error log is stored by deletion of the oldest error log.

(2) Storage data

Clear the history file in a memory card and transfer the 16 logs in the error history storage memory of the CPU module to the history file when:

- The number of logs in the parameter history file is changed during operation or
- The number of logs in mounted memory card differs from the ones set in the parameter.

Point

Even if a memory card does not contain an error history file set in the parameter, the CPU module will not cause an error.
In this case, the CPU module stores the latest error in the error history storage memory of the CPU module only.

(3) How to clear error history

To clear the storage memory for error history and error history file, select [Diagnostics] → [PLC diagnostics] in GX Developer and click the Clear log button.
This method clears all data stored in the storage memory for error history of the CPU module and error history file in a memory card.

6.19 System Protection Note6.21

The CPU module has protection functions (system protection) to prevent programs being modified by a third party other than the designer with GX Developer or serial communication module.

Table6.29 System protection types

Protection target	File that can be protected	Description	Method	Valid timing	Reference
Entire CPU module [*1]	All files (devices are also protected.)	Prohibits all remote operation directions from such as GX Developer to the CPU module.	Turn on the dip switch of the CPU module (SW1).	Always	-
In units of memory cards[*1]	All files	Prohibits writing to a memory card.	Turn on the write protect switch of a memory card.	Always	-
In units of files	• Program • Device comment • Initial device value	• Changes the attribute for each file to either of the following. • Read/write (reading/writing to programs) prohibition • Write prohibition (operations regarding writing such as writing to programs and tests)	Make setting in the Password registration screen.	Always	Section 6.19.1

[*1] The Basic model QCPU does not support the system protection in units of memory cards and for entire CPU module since it cannot use a memory card.

Point

The following functions set in the PLC parameter dialog box or with the dip switches of the CPU module are performed even when the dip switch of the CPU module (SW1) is turned on and the system protection is activated. Note6.21

- Boot from the standard ROM and memory card
- Auto writing to the standard ROM

Note6.21 Basic

The Basic model QCPU does not support the system protection with the dip switch.

6.19.1Password registration

This function prohibits reading and writing data such as a programs and device comments in the CPU module with GX Developer.

(1) Valid password range

The password can be registered with program, device comment, and initial device values files in the specified memory (program memory, standard ROM, and memory card). Note6.22

(2) Operations that can be prohibited

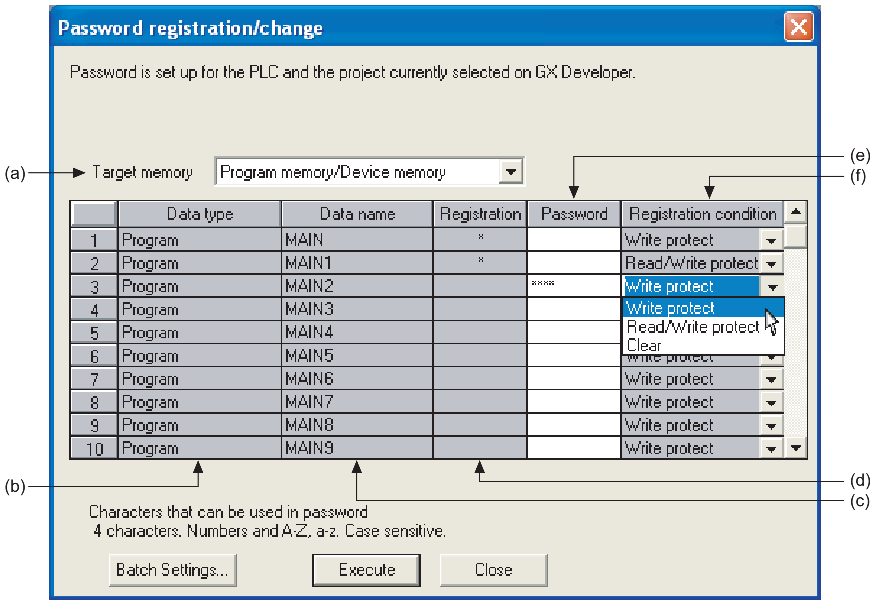
The following two operations can be prohibited.

- Reading and writing a file
- Writing a file

When a password has been registered with a file, GX Developer cannot operate the file unless entered password does not match with the registered password.

(3) Setting method

Select [Online] → [Password setup] or click the Password setup button on the Write to PLC screen in GX Developer.



Password is set up for the PLC and the project currently selected on GX Developer.

Target memory: Program memory/Device memory

	Data type	Data name	Registration	Password	Registration condition
1	Program	MAIN	*		Write protect
2	Program	MAIN1	*		Read/Write protect
3	Program	MAIN2		****	Write protect
4	Program	MAIN3			Write protect Read/Write protect Clear
5	Program	MAIN4			
6	Program	MAIN5			Write protect
7	Program	MAIN6			Write protect
8	Program	MAIN7			Write protect
9	Program	MAIN8			Write protect
10	Program	MAIN9			Write protect

Characters that can be used in password
4 characters. Numbers and A-Z, a-z. Case sensitive.

Figure 6.73 Password registration/change screen

Note6.22 **Basic**

The valid password range of the Basic model QCPU can be set for program files, device comment files, and initial device values in the program memory.
The range cannot be set for the standard ROM and memory card.

(a) Target memory

Select a memory storing a file where a password is to be registered.

(b) Data type

Displays the type of a file stored in the target memory.

(c) Data name

Displays the name of a file stored in the target memory.

(d) Registration

Displays "*" when a password has been set to the target file.

(e) Password

Enter current password or a password to be registered.
The Registration Condition column can be set after setting a password.

(f) Registration Condition

1) Write protect

Writing to a password-protected file is prohibited (reading is allowed).

2) Read/Write protect

Reading and writing to a password-protected file is prohibited.

3) Clear

The set password is cleared (entering the current password is necessary).

(4) Precautions

A password registered with a file cannot be read from the file.
Forgetting the registered password disables the following operations.

(a) Basic model QCPU

- Program memory: Format PLC memory
- Standard ROM: Write the program memory to ROM

(b) High Performance model QCPU, Process CPU, and Redundant CPU

- Program memory and memory card: Format PLC memory
- Standard ROM: Batch-writing

Make sure to record the registered password and store the recording paper.

6.19.2 Remote password Note6.23

(1) Definition

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.

(2) Settable modules and the number of settable modules

Table6.30 shows the modules for which the remote password can be set and the number of settable modules.

Table6.30 Settable modules and the number of settable modules

Settable module	Number of settable modules
Ethernet module	4
Serial communication module	8 *1
Modem interface module	

*1: When GX Developer Version 6 or 7 is used, the number of settable modules is 4.

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 using the 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 of each module, refer to the following.
  ☞ Manual for each module

Note6.23 Basic High performance

When using a remote password for the Basic model QCPU or High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.1, Appendix 2.2)

(3) Flow from remote password setting to reflection of the password

Set a remote password (☞ (5) in this section) and then write it to the CPU module.

The remote password is transferred to the target module when the CPU module is powered off and then on or is reset. (☞ (2) in this section)

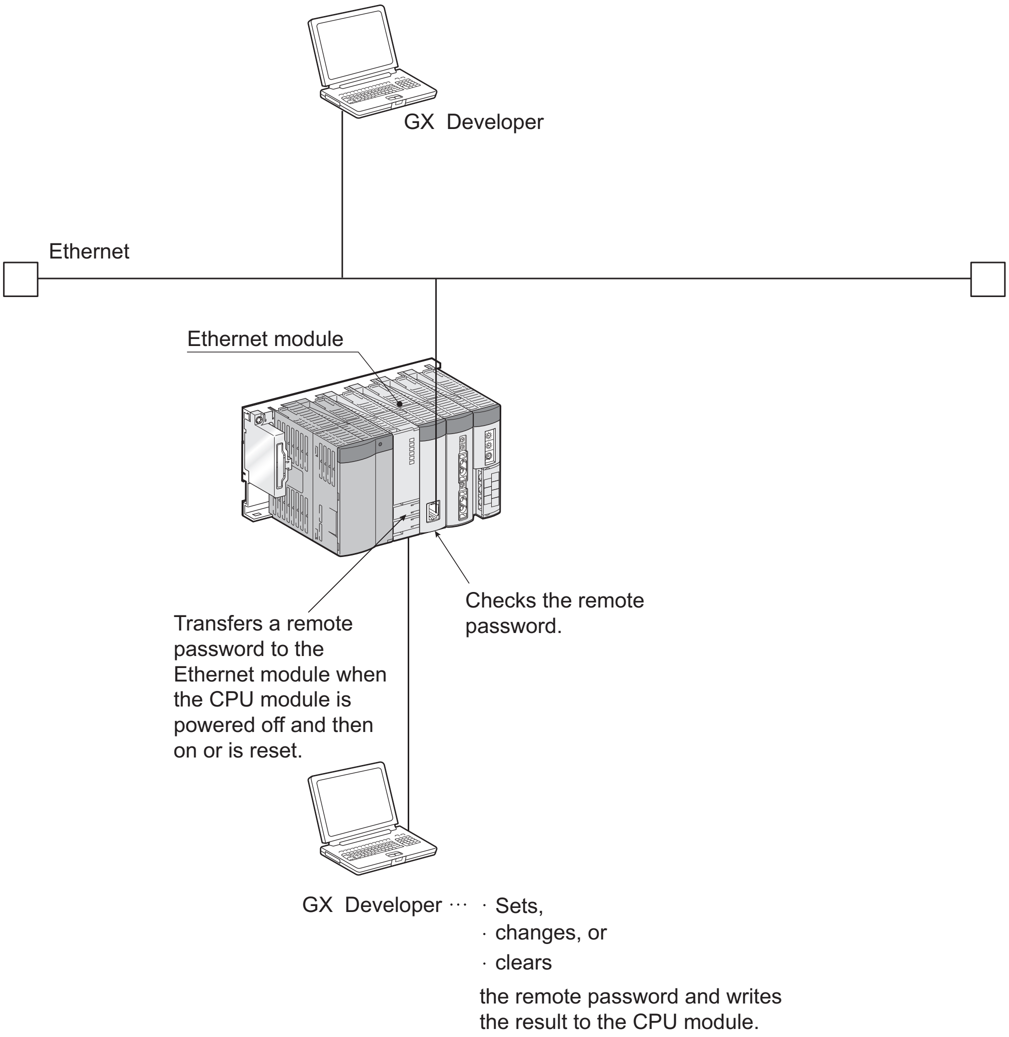

GX Developer
Ethernet
Ethernet module
Transfers a remote password to the Ethernet module when the CPU module is powered off and then on or is reset.
Checks the remote password.
GX Developer ··· · Sets,
· changes, or
· clears
the remote password and writes the result to the CPU module.

Figure 6.74 Outline of a remote password

(4) Remote password lock/unlock

Unlock the remote password of a serial communication module via a modem or the Ethernet module via Ethernet. When entered remote password matches with the registered password, the module can access the CPU module.

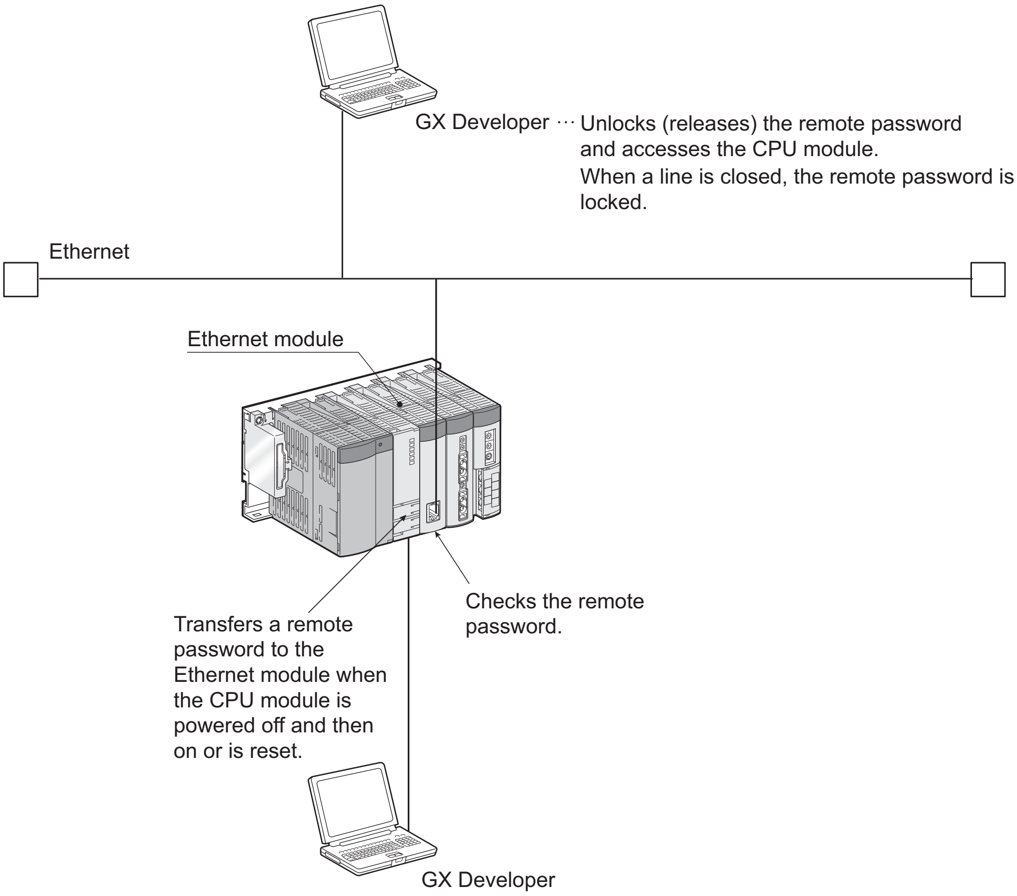

GX Developer … Unlocks (releases) the remote password and accesses the CPU module.
When a line is closed, the remote password is locked.
Ethernet
Ethernet module
Transfers a remote password to the Ethernet module when the CPU module is powered off and then on or is reset.
Checks the remote password.
GX Developer

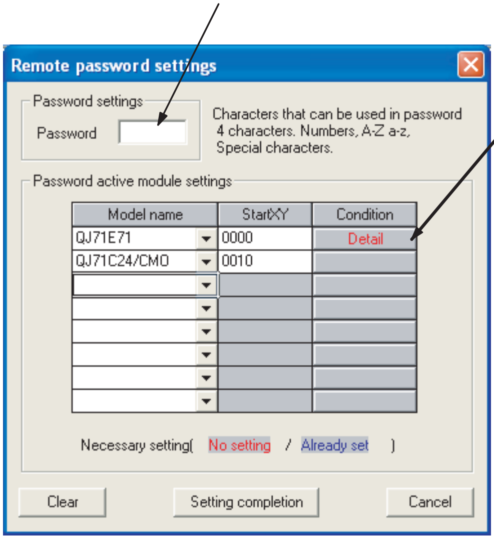
Figure 6.75 Outline of locking/unlocking a remote password with an Ethernet module

(5) Procedures for setting/changing/clearing a remote password

(a) Setting a remote password

- In the project data list of GX Developer, select [Parameter] → [Remote pass]

Remote password setting



Remote password settings

Password settings

Password

Characters that can be used in password
4 characters. Numbers, A-Z a-z,
Special characters.

Password active module settings

Model name	StartXY	Condition
QJ71E71	0000	Detail
QJ71C24/CMO	0010	

Necessary setting(No setting / Already set)

For the QJ71E71, configure setting in "Detail".

Figure 6.76 Remote password settings screen

Table6.31 Setting items on the Remote password settings screen

Item			Description	Setting range/option
Password settings			Enter a remote password.	Four characters or less (alphanumeric characters, special symbols)
Password active module settings	Model name		Select a model name.	• QJ71E71 • QJ71C24/CMO
	Start X/Y		Set the start address of the module.	• Basic model QCPU: 0000_H to $03E0_H$ • High Performance model QCPU, Process CPU, and Redundant CPU: 0000_H to $0FE0_H$
Detail			-	-
	User connection No.		Select user connection No.	Connection 1 to Connection 16
	System connection	Auto open UDP port	Select a valid port of the remote password.	-
		FTP transmission port (TCP/IP)		
		GX Developer transmission port (TCP/IP)		
		GX Developer transmission port (UDP/IP)		
		HTTP port		

- Connect GX Developer to the CPU module.
  Write a set remote password to the CPU module.
  In multiple CPU systems, write a remote password to the control CPU of the module to which the remote password is to be set.

Point

- After setting a remote password, store the parameters to the program memory (drive 0). Note6.24
  If not, the remote password function does not work properly.
- For boot operation, store the parameter file to the standard ROM or memory card and make setting in the Boot file tab of GX Developer so that the parameter file is transferred to the program memory. Note6.24
  Then, set the parameter-valid drive of the dip switch to the standard ROM or memory card storing the parameter file (If setting the parameter-valid drive to the program memory, the boot operation does not normally operate).

(b) Changing a remote password

Change set password in the Remote password settings screen and write a new password to the CPU module.

(c) Clearing a remote password

- To delete set remote password, click the Clear button in the Remote password settings screen.
- Write a remote password with GX Developer.

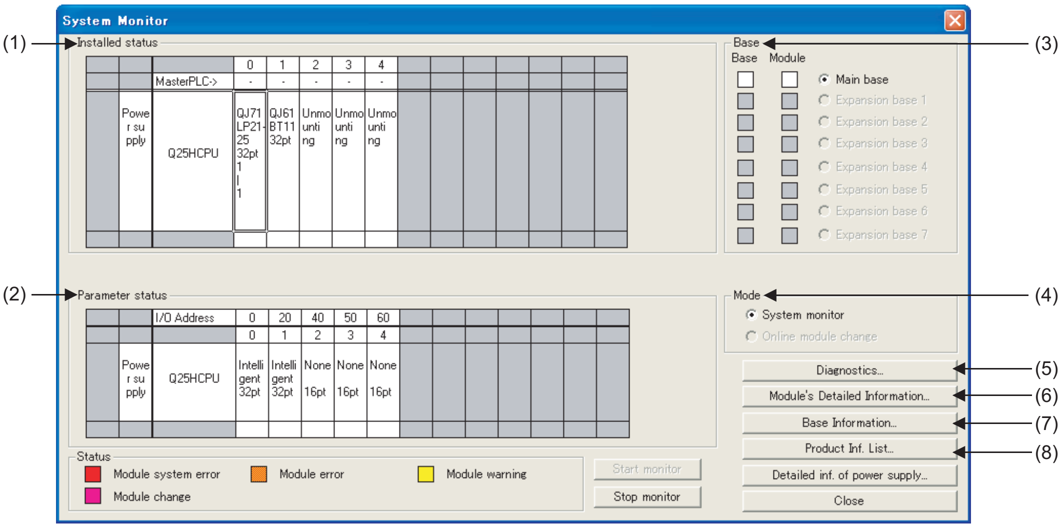
Note6.24 Basic

The Basic model QCPU always stores parameters in the program memory.

6.20 System Display of CPU Module with GX Developer

When the CPU module is connected to GX Developer, this function can check the following items of the modules on the base unit in the System Monitor screen.

- Installed status
- Parameter status
- Module's Detailed Information
- Product information



System Monitor

Installed status

			0	1	2	3	4
		MasterPLC->	-	-	-	-	-
	Powe r su pply	Q25HCPU	QJ71 LP21- 25 32pt 1 | 1	QJ61 BT11 32pt	Unmo unti ng	Unmo unti ng	Unmo unti ng

Base

Base Module

● Main base
○ Expansion base 1
○ Expansion base 2
○ Expansion base 3
○ Expansion base 4
○ Expansion base 5
○ Expansion base 6
○ Expansion base 7

Parameter status

		I/O Address	0	20	40	50	60
			0	1	2	3	4
	Powe r su pply	Q25HCPU	Intelli gent 32pt	Intelli gent 32pt	None 16pt	None 16pt	None 16pt

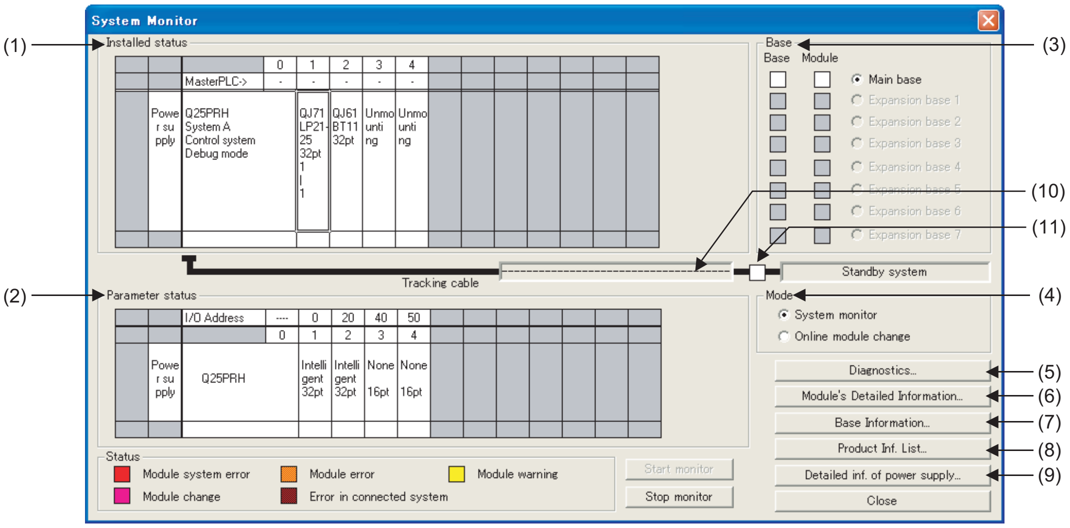
Mode

● System monitor
○ Online module change

Status

🟥 Module system error 🟧 Module error 🟨 Module warning
🟪 Module change

Figure 6.77 System Monitor screen



System Monitor

Installed status

			0	1	2	3	4
		MasterPLC->	-	-	-	-	-
	Powe r su pply	Q25PRH System A Control system Debug mode	QJ71 LP21- 25 32pt 1 | 1	QJ61 BT11 32pt	Unmo unti ng	Unmo unti ng	

Tracking cable

Standby system

Base

Base Module

● Main base
○ Expansion base 1
○ Expansion base 2
○ Expansion base 3
○ Expansion base 4
○ Expansion base 5
○ Expansion base 6
○ Expansion base 7

Parameter status

		I/O Address	----	0	20	40	50
			0	1	2	3	4
	Powe r su pply	Q25PRH		Intelli gent 32pt	Intelli gent 32pt	None 16pt	None 16pt

Mode

● System monitor
○ Online module change

Status

🟥 Module system error 🟧 Module error 🟨 Module warning
🟪 Module change 🟫 Error in connected system

Figure 6.78 System Monitor screen (Redundant CPU)

(1) Installed status

The following information of the module mounted on the selected base unit can be checked.

- Control CPU (except the Basic model QCPU)
- Model name[*1]
- Number of points

"Unmounting" is displayed in the field of a number of slot where a module is not mounted.
When using a redundant base unit, mounted status of the power supply module is also displayed (not displayed for the Basic model QCPU).

[*1] The model name of a module on a slot to which "Empty" is set in the I/O assignment tab of the PLC parameter dialog box is not displayed.

(2) Parameter status

The following information on the module on each slot of the selected base unit can be displayed.

- I/O number
- Module type
- Number of points

If setting configured in the I/O assignment tab of the PLC parameter dialog box and the mounted status differ, “None 0 pt” or an assignment error is displayed.
Therefore, change the I/O assignment setting so that it may match with the mounted status.
When using a redundant base unit, operating status of the power supply module is also displayed. (not displayed for the Basic model QCPU).

(3) Base

The status of the base unit and modules on the base unit can be checked.
When there is even one faulty module, the "Module" field color changes according to the status described at the bottom of the screen.

(4) Mode Note6.25

Use this option for online module change.
For online module change, refer to the following.

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

☞ Manual for the module supporting the online module change

(5) Diagnostics

Click this button to check an error and status of the selected module.

Note6.25 Basic High performance

Since the Basic model QCPU and High Performance model QCPU cannot be changed online, "Mode" cannot be

(6) Module's Detailed Information

Click this button to check the details of the selected module.

For details of the intelligent function module, refer to the manual for the intelligent function module used.

(7) Base Information

The "Overall Information" and "Base Information" can be checked.

(a) Overall Information

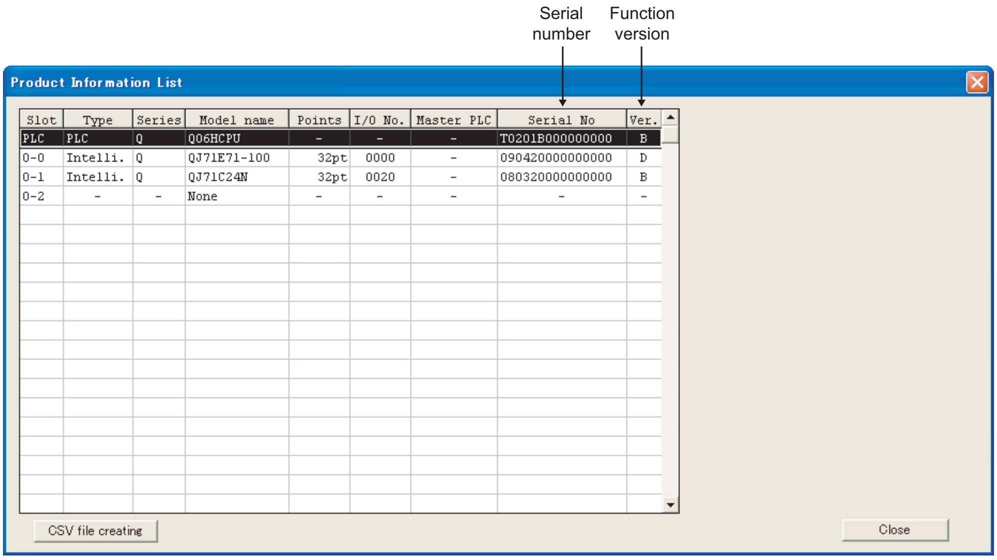
The number of base units and the number of modules on the base units can be checked.

(b) Base Information

The name, the number of slots, type, and the number of modules of the selected base unit can be checked.

(8) Product Inf. List

Individual information (Type, Series, Model name, Points, I/O No., Master PLC, Serial No., and Ver.) of the mounted CPU module, I/O modules, and intelligent function module can be checked.



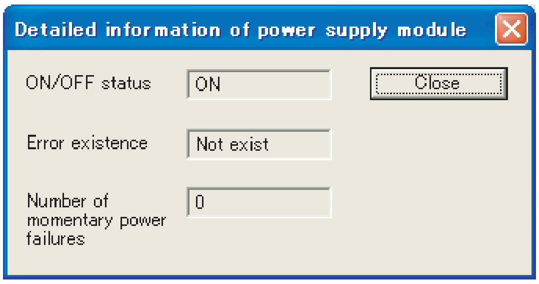
Slot	Type	Series	Model name	Points	I/O No.	Master PLC	Serial No	Ver.
PLC	PLC	Q	Q06HCPU	-	-	-	T0201B000000000	B
0-0	Intelli.	Q	QJ71E71-100	32pt	0000	-	090420000000000	D
0-1	Intelli.	Q	QJ71C24N	32pt	0020	-	080320000000000	B
0-2	-	-	None	-	-	-	-	-

Figure 6.79 Product Information List screen

(9) Detailed information of power supply module

This screen displays "ON/OFF status", "Error existence", and "Number of momentary power failures" of the power supply module.

This screen can be displayed when using the power supply module supporting a redundant base unit and this screen.



Detailed information of power supply module

ON/OFF status: ON

Error existence: Not exist

Number of momentary power failures: 0

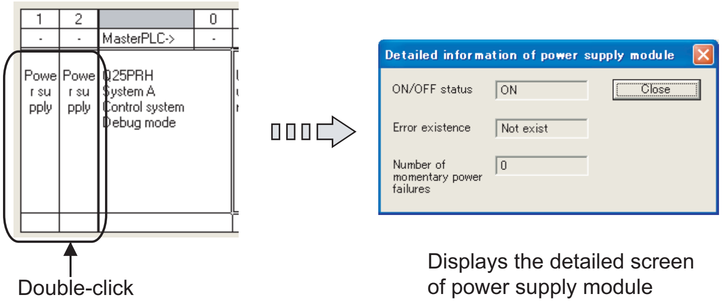
Figure 6.80 Detailed information of power supply module screen

Table6.32 Description of the Detailed information of power supply module screen

Item	Description
ON/OFF status	Displays the status of an input power supply to the redundant power supply module.
Error existence	Displays whether a failure (error) occurs in the redundant power supply module.
Number of momentary power failures	• When the redundant power supply module of redundant main base unit is selected Displays the number of momentary power failures of the redundant power supply module on the redundant main base unit (display range: 0 to 65535). • When the redundant power supply module of redundant extension base unit is selected Displays "-" and the number of momentary power failures is not counted.

Point

- In multiple CPU systems, the Detailed information of power supply module screen can be displayed only:
  - when GX Developer is connected to CPU No. 1, or
  - when serial numbers of all CPU modules (first five digits) are "07032" or later.
- Double-clicking the area of power supply module in "Installed status" can also display the screen.



1	2		0
-	-	MasterPLC->	-
Power supply	Power supply	Q25PRH System A Control system Debug mode	

Detailed information of power supply module

ON/OFF status: ON

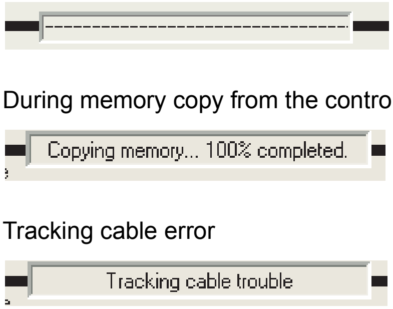
Error existence: Not exist

Number of momentary power failures: 0

(10)Memory copy status

This item indicates the execution status of memory copy from the control system CPU module to standby system CPU module.

- During normal operation



- During memory copy from the control system CPU module to standby system CPU module

Copying memory... 100% completed.

- Tracking cable error

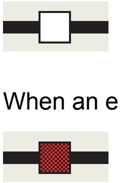
Tracking cable trouble

Figure 6.81 Memory copy status

(11)Status of the other system

This item indicates the status of the other system.

- During normal operation



- When an error occurs

Figure 6.82 Status of the other system

When the Redundant CPU is in the debug mode, however, this item indicates normal operation status even if an error has occurred in the other system (error status is not indicated).

Remark

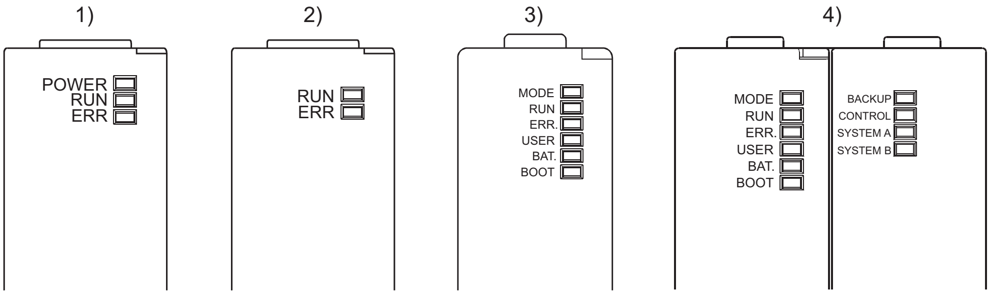
For details of the system monitor of GX Developer, refer to the following.

☞ GX Developer Version 8 Operating Manual

6.21 LED Display

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)



1): Basic model QCPU (Q00JCPU)
2): Basic model QCPU (Q00CPU, Q01CPU)
3): High Performance model QCPU and Process CPU
4): Redundant CPU

Figure 6.83 LEDs on the front of the CPU module

6.21.1 Methods for turning off the LEDs

(1) Methods

(a) Basic model QCPU

To turn off the ERR.LED, remove the error cause and then operate the special relay SM50 and special register SD50 to clear the error (except for reset operation).

(b) High Performance model QCPU, Process CPU, and Redundant CPU

The LEDs can be turned off by the following operations (except for reset operation.)

Table6.33 Methods for turing off the LEDs

Method for turning off the LED	Applicable 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]	×	○	×	○

○ : Valid, × : Invalid

*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 an code of an 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.

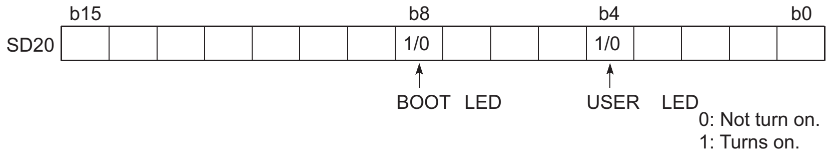


Figure 6.84 Bit structure of the special register SD202

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$_H$

(2) Methods for not turning on the ERR. LED, USER LED, and BAT. LED

There is a priority in indications of the ERR.LED, USER LED, and BAT.LED. (Section 6.21.2)
When an 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.

6.21.2LED 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. The priority can be changed (set it to the 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.

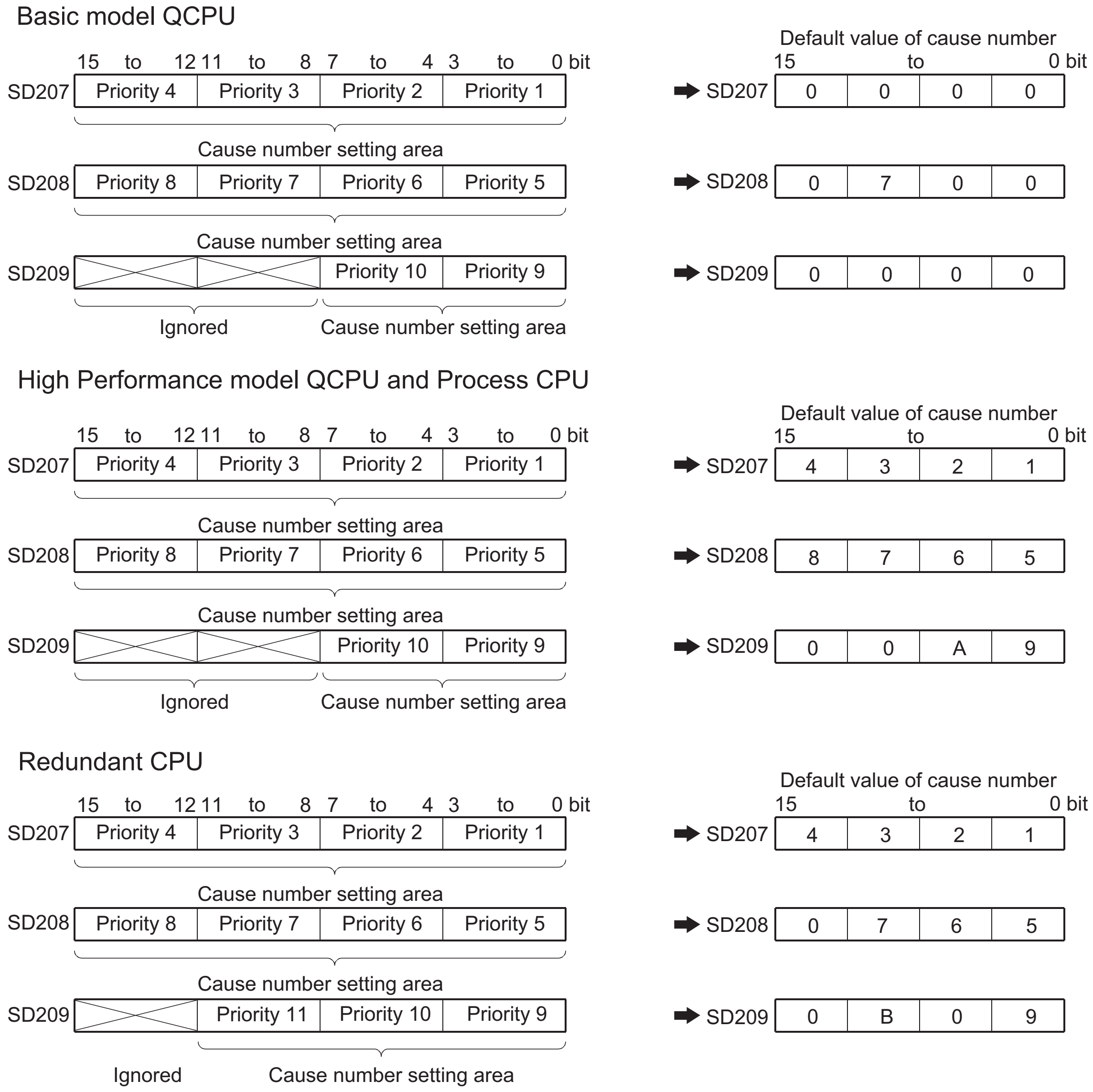

Basic model QCPU
15 to 12 11 to 8 7 to 4 3 to 0 bit
SD207 Priority 4 Priority 3 Priority 2 Priority 1
Cause number setting area
SD208 Priority 8 Priority 7 Priority 6 Priority 5
Cause number setting area
SD209 Priority 10 Priority 9
Ignored
Cause number setting area
Default value of cause number
15 to 0 bit
SD207 0 0 0 0
SD208 0 7 0 0
SD209 0 0 0 0
High Performance model QCPU and Process CPU
15 to 12 11 to 8 7 to 4 3 to 0 bit
SD207 Priority 4 Priority 3 Priority 2 Priority 1
Cause number setting area
SD208 Priority 8 Priority 7 Priority 6 Priority 5
Cause number setting area
SD209 Priority 10 Priority 9
Ignored
Cause number setting area
Default value of cause number
15 to 0 bit
SD207 4 3 2 1
SD208 8 7 6 5
SD209 0 0 A 9
Redundant CPU
15 to 12 11 to 8 7 to 4 3 to 0 bit
SD207 Priority 4 Priority 3 Priority 2 Priority 1
Cause number setting area
SD208 Priority 8 Priority 7 Priority 6 Priority 5
Cause number setting area
SD209 Priority 11 Priority 10 Priority 9
Ignored
Cause number setting area
Default value of cause number
15 to 0 bit
SD207 4 3 2 1
SD208 0 7 6 5
SD209 0 B 0 9

Figure 6.85 Special registers and bit structure regarding a priority

(2) Priorities and cause numbers

The following table shows the description and priority of the cause numbers set to the special registers SD207 to SD209.

Table6.34 List of cause numbers and priorities

Priority	Cause number[*1] (Hexadecimal)	Displayed error message	Remarks
1	1	• AC/DC DOWN • SINGLE PS.DOWN[*2] • SINGLE PS.ERROR[*2]	• Power-off • Redundant base unit power supply voltage drop • Redundant power supply module fault
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 • FILE OPE.ERROR • OPE. MODE DIFF.[*3] • CAN'T EXE.MODE[*3] • TRK.TRANS.ERR.[*3] • TRK.SIZE ERROR[*3] • TRK.DISCONNECT[*3]	• Memory card operation error • File access error • Operating status or key switch mismatch • Function disable during current mode • Tracking data communication error • Tracking capacity excess error • Tracking cable not connected, failure, or tracking transfer hardware failure
5	5	• PRG.TIME OVER • MULTI CPU ERROR	• Constant scan setting time-out error • Low-speed execution monitoring time-out error • Another CPU error in multiple CPU systems
6	6	• CHK instruction[*4]	-
7	7	• Annunciator	-
8	8	-	-
9	9	• BATTERY ERROR	-
10	A	-	-
11	B	• CAN'T SWITCH[*3] • STANDBY SYS.DOWN[*3] • MEM.COPY EXE[*3]	• System switching error • Standby system not started or stop error • Memory copy function execution

[*1] The Basic model QCPU can set the cause number 7 (annunciator) only.

[*2] For the High Performance model QCPU and Process CPU, check the version of the CPU module. (☞ Appendix 2)

[*3] The Basic model QCPU, High Performance model QCPU, and Process CPU cannot display the error message.

[*4] The Basic model QCPU cannot use the CHK instruction.

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".

  [Example]

  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".

SD209				SD208				SD207			
0	0	A	9	8	7	6	5	4	3	0	1

Figure 6.86 Cause numbers stored in SD207 to SD209

  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).

6.22 High Speed Interrupt Function Note6.26

When an interrupt program is created using the high speed interrupt pointer (I49), the entire program can be executed at a high speed, being interrupted at intervals of 0.2ms to 1.0ms.
Also, this function improves the I/O response because I/O signal data within the parameter-set range and intelligent function module buffer memory data are refreshed before and after execution of the high speed interrupt program.
This allows high-accuracy control such as precise position detection.

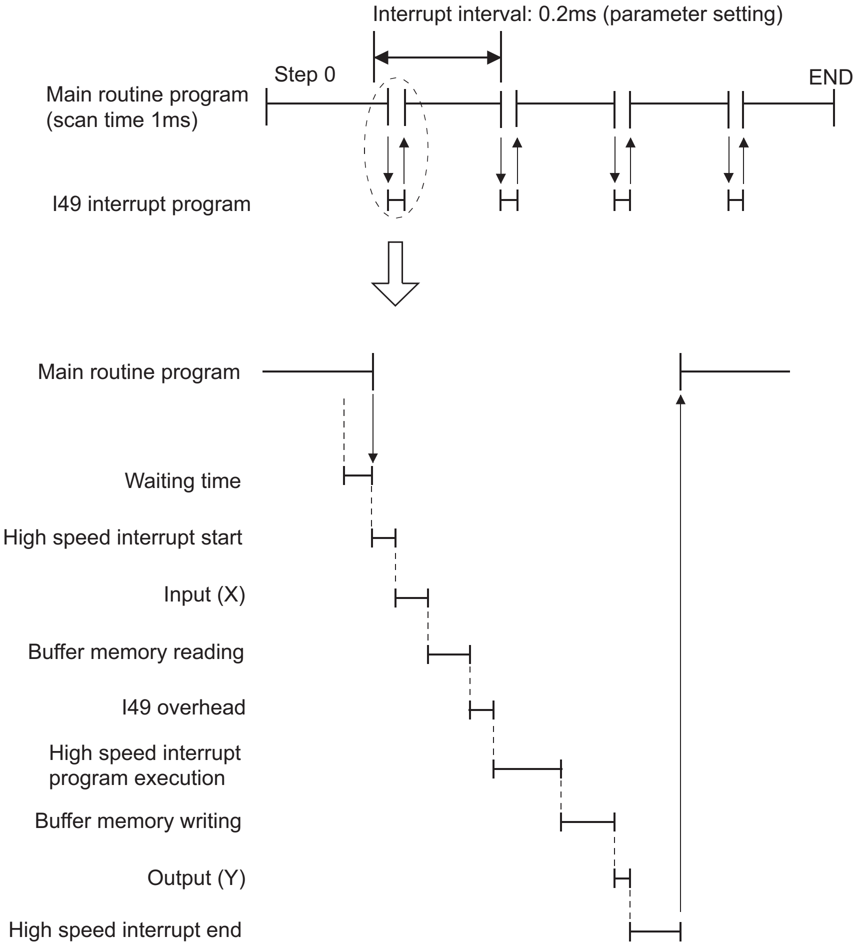


Figure 6.87 High speed interrupt timing chart

The high speed interrupt function includes the following:

- High speed interrupt program execution function: Section 6.22.1
- High speed I/O refresh and high speed buffer transfer functions: Section 6.22.2

Note6.26 Basic High performance Process Redundant

The Basic model QCPU, Q02CPU, Process CPU, and Redundant CPU do not support the high speed interrupt function.
When using the high speed interrupt function for the High Performance model QCPU, check the versions of the CPU module and GX Developer. (Appendix 2.2)

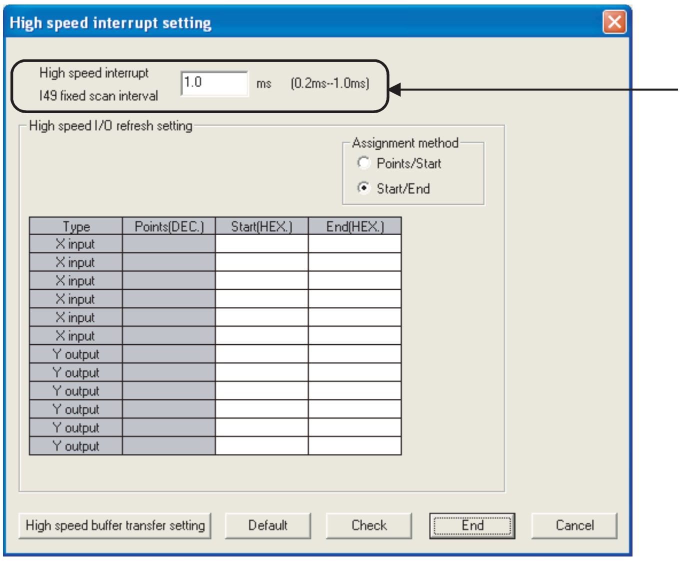
6.22.1 High speed interrupt program execution function

This function executes an interrupt program according to the high speed interrupt pointer (I49).

(1) Setting method

In the PLC system tab of the PLC parameter dialog box, click the High speed interrupt setting button in the System interrupt settings area.

Enter a value in the range of 0.2 to 1.0 (ms) for "High speed interrupt I49 fixed scan interval".



High speed interrupt setting

High speed interrupt I49 fixed scan interval: 1.0 ms (0.2ms--1.0ms)

High speed I/O refresh setting

Assignment method
○ Points/Start
● Start/End

Type	Points(DEC.)	Start(HEX.)	End(HEX.)
X input			
X input			
X input			
X input			
X input			
X input			
Y output			
Y output			
Y output			
Y output			
Y output			
Y output			

Set a value within the range of 0.2 to 1.0ms.

Figure 6.88 High speed interrupt setting

(2) Precautions

(a) High speed interrupt while interrupt is disabled

The high speed interrupt program is not executed while interrupt is disabled.

It becomes executable when interrupt is enabled.

For the items that cannot start the high speed interrupt due to disabled interrupt, refer to Section 6.22.4(3).

(b) When a high speed interrupt is ignored

If the interrupt-disabled period continues longer than the set interrupt interval, a high speed interrupt may be ignored.

High speed interrupt is ignored once when it occurs twice during interrupt-disabled period.

(c) High speed interrupt program execution

This function is executed when all of the following conditions are satisfied.

- The EI instruction is being executed.
- The CPU module is in the RUN status.
- The high speed interrupt pointer (I49) is not masked by the IMASK instruction. (By default, it is not masked.)

For the IMASK and EI instructions, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

6.22.2 High speed I/O refresh and high speed buffer transfer functions

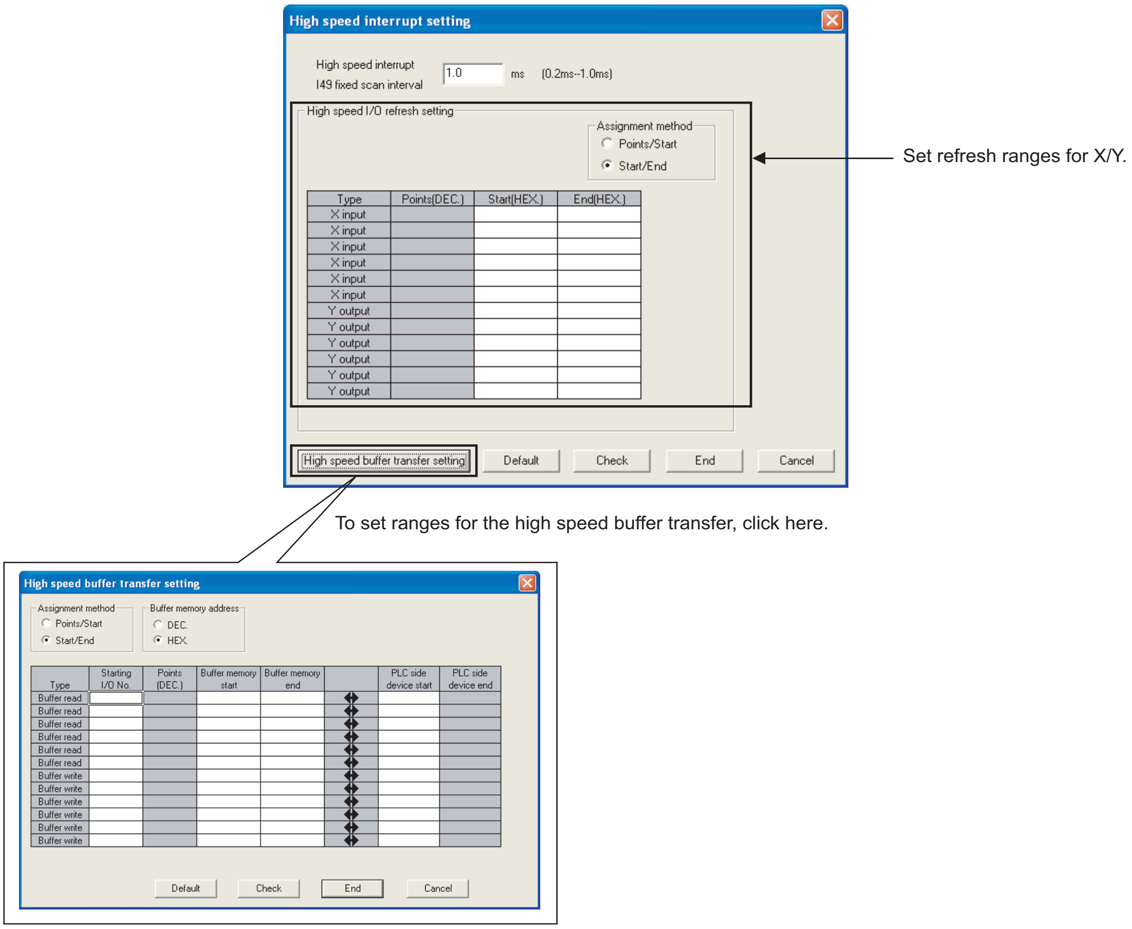
High speed I/O refresh is a function that updates I/O signal data between I/O modules and intelligent function modules and the CPU module at the specified interrupt intervals.

High speed buffer transfer is a function that updates data between intelligent function module buffer memories and CPU module devices at the specified interrupt intervals.

(1) Setting method

Set the following in the High speed interrupt setting dialog box.

- "High speed interrupt I49 fixed scan interval"
- "High speed I/O refresh setting"
- "High speed buffer transfer setting"



High speed interrupt setting

High speed interrupt I49 fixed scan interval: 1.0 ms (0.2ms--1.0ms)

High speed I/O refresh setting

Assignment method: ○ Points/Start ● Start/End

Type	Points(DEC.)	Start(HEX.)	End(HEX.)
X input			
X input			
X input			
X input			
X input			
X input			
Y output			
Y output			
Y output			
Y output			
Y output			
Y output			

Set refresh ranges for X/Y.

To set ranges for the high speed buffer transfer, click here.

High speed buffer transfer setting

Assignment method: ○ Points/Start ● Start/End

Buffer memory address: ○ DEC. ● HEX.

Type	Starting I/O No.	Points (DEC.)	Buffer memory start	Buffer memory end		PLC side device start	PLC side device end
Buffer read					↔		
Buffer read					↔		
Buffer read					↔		
Buffer read					↔		
Buffer read					↔		
Buffer read					↔		
Buffer write					↔		
Buffer write					↔		
Buffer write					↔		
Buffer write					↔		
Buffer write					↔		
Buffer write					↔		

Figure 6.89 High speed interrupt setting dialog box

Table6.35 High speed I/O refresh setting and high speed buffer transfer setting

Item	Setting item	Description	Restriction	Number of settings
High speed I/O refresh setting	Points (DEC.)	Number of transferred bits (16 to I4096)	• I/O modules and intelligent function modules only • Specify multiples of 16 only [*1]	Up to six settings for X input and Y output, respectively
	Start (HEX.)	Start device number (X0 to X0FF0/Y0 to Y0FF0)		
	End (HEX.)	End device number (X00E to X0FFF/Y00E to Y0FFF)		
High speed buffer transfer setting	Starting I/O No. (HEX.)	Starting I/ONo.÷10H (0 to FF$_H$)	Intelligent function modules only [*2]	Up to six settings for read and write, respectively
	Points (DEC.)	Number of transferred words	• Intelligent function modules only • Specify even addresses and even words only [*3]	
	Buffer memory start	Start address (0 to FFFF$_H$)		
	Buffer memory end	End address (FFFD to FFFF$_H$)		
	PLC side device start	Start device number	D, W, R, and ZR only	
	PLC side device end	End device number		

*1: Only multiples of 16 can be set for both the start device number and the number of transferred bits.

*2: The AnS/A series special function modules are not included because the AnS/A series extension base unit (QA1S5□B, QA1S6□B, QA1S6ADP+A1S5□B/A1S6□B, QA6□B, and QA6ADP+A5□B/A6□B) cannot be connected.
If connected, "PARAMETER ERROR" (error code: 3006) occurs.
This "PARAMETER ERROR" (error code: 3006) also occurs when an error is detected during the intelligent function module mounting status check or buffer memory size check.

*3: An odd address is allowed only when the specified number of transferred words is 1.

Point

Use of a main base unit is recommended for these functions.
(Access time to modules mounted on a main base unit is shorter than access time to modules on an extension base unit.)

(2) Execution of the high speed I/O refresh and high speed buffer transfer functions

These functions are executed when all of the following conditions are satisfied.

- The EI instruction is being executed.
- The CPU module is in the RUN status.
- The high speed interrupt pointer (I49) is not masked by the IMASK instruction.
(By default, it is not masked.)

For the IMASK and EI instructions, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

6.22.3 Processing time

The following shows each processing time during the period from the start to the end of the high speed interrupt function.

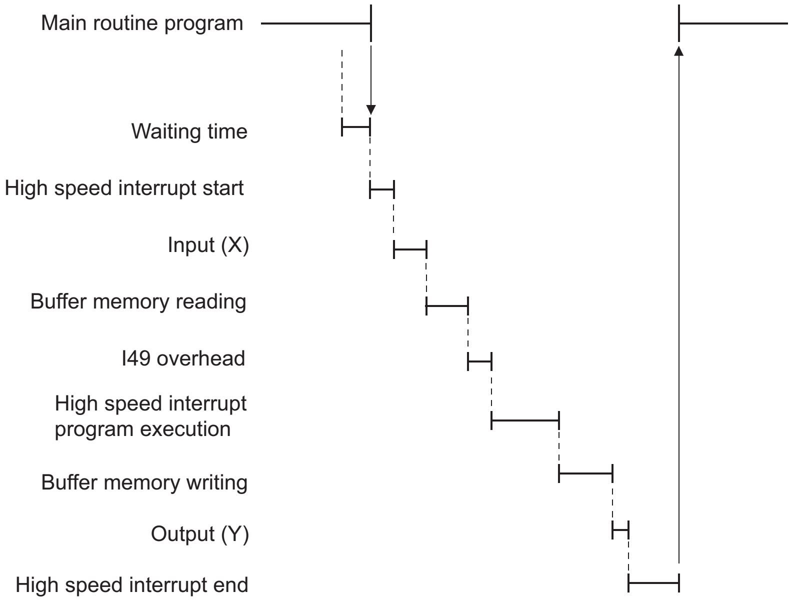


Figure 6.90 The start-to-end high speed interrupt function processing

Table6.36 Processing times related to the high speed I/O refresh and high speed buffer transfer

Item	Processing time
Waiting time	Up to 37.5μs (More than 37.5μs may be required to process some of instructions.) Up to 40μs when any of CC-Link IE Controller Network modules, MELSECNET/H modules, CC-Link modules, and intelligent function modules are mounted on an extension base unit.
High speed interrupt start+High speed interrupt end	22μs
X input	Main base unit: Time=0.14×(total of × points)+0.65×(number of settings)+0.85 Extension base unit: Time=0.21×(total of points)+0.65×(number of settings)+0.85 (Calculation example) When a main base unit is used, the number of settings is 1, and 16 points are set for X, the time is 3.74μs.
Buffer memory reading	Main base unit: • 16 words or less ••• Time=0.47×(total number of transferred words)+2.85×(number of settings)+0.95 • More than 16 words••• Time=0.5×(total number of transferred words)+0.95 Extension base unit: • 16 words or less••• Time=1.07×(total number of transferred words)+2.85×(number of settings)+0.95 • More than 16 words ••• Time=1.1×(total number of transferred words)+0.95 (Calculation example) When a main base unit is used, the number of settings is 1, and two words are to be transferred, the time is 4.74μs.
I49 overhead	41μs
High speed interrupt program execution	Depends on the user-created interrupt program.

(To the next page)

Table6.36 Processing times related to the high speed I/O refresh and high speed buffer transfer (continued)

Item	Processing time
Buffer memory writing	Main base unit: • 16 words or less ••• Time=0.47×(total number of transferred words)+2.65×(number of settings)+0.95 • More than 16 words••• Time=0.55×(total number of transferred words)+0.95 Extension base unit: • 16 words or less••• Time=1.07×(total number of transferred words)+2.65×(number of settings)+0.95 • More than 16 words ••• Time=1.15×(total number of transferred words)+0.95 (Calculation example) When a main base unit is used, the number of settings is 1, and two words are to be transferred, the time is 4.54μs.
Y output	Main base unit: Time=0.13×(total of Y points)+1.55 Extension base unit: Time=0.2×(total of Y points)+1.55 (Calculation example) When a main base unit is used, the number of settings is 1, and 16 points are set for Y, the time is 3.63μs.

6.22.4 Restrictions

This section describes restrictions on execution of the high speed interrupt function.
If the function is executed incorrectly, "WDT ERROR" may occur or the high speed interrupt may not be executed at the specified intervals.

There are the following four kinds of restriction items

(1) Restrictions that apply to all the high speed interrupt setting
(2) Restrictions that apply to the high speed interrupt only
(3) Items that delay the high speed interrupt start due to disabled interrupt
(4) Items other than the above ((1) to (3))

The time required for running interrupt program once must not exceed the specified interrupt interval.
(If exceeded, the operation of the high speed interrupt cannot be guaranteed.)

(1) Restrictions that apply to all the high speed interrupt setting

Table6.37 Restrictions that apply to all the high speed interrupt setting

Item	Restriction	When used
Base unit	QA1S5□B, QA1S6□B, QA1S6ADP+A1S5□B/A1S6□B, QA6□B and QA6ADP+A5□B/A6□B cannot be used.	"PARAMETER ERROR" is detected.
Multiple CPU system	Not available for multiple CPU systems.	Availability is checked when parameters are set in GX Developer.
Instruction	The PR/PRC, UDCNT1/2, PLSY, PWM, SPD, and PLOADP/PUNLOADP/PSWAPP instructions are not executable.	An error is detected without executing the instruction.
Instruction	Any instruction of which processing time is longer than the high speed interrupt interval cannot be used.	Since interrupt is disabled during instruction execution, high speed interrupt is not executed at the specified intervals.
Programming unit	No programming unit can be connected.	Response to instruction search slows down. Or, a communication error may be detected on the programming unit.
SFC	The following two SFC functions are not executable. • SFC transition monitoring check using SM90 to SM99 and SD90 to SD99 • Periodic execution block function	These functions are ignored without being executed.
Sampling trace	Periodic sampling trace cannot be used. (Available when each scan or detailed conditions are set.)	Sampling trace is ignored without being executed. (Trace results may not be read out when reading trace data.)
Interrupt program (I0 to I48, I50 to I255), fixed scan execution type program	Interrupt programs (I0 to I48 and I50 to I255) and the fixed scan execution type program are not executable.	The high speed interrupt cannot be executed at the specified intervals during execution of an interrupt program or the fixed scan execution type program because multiple interrupts are disabled.
Online change (ladder mode and files)	Online change is not executable.	Since interrupts are disabled while writing a file in the RUN status, the high speed interrupt cannot be executed at the specified intervals due to delay of the start, and requires the following time. • Up to 102μs for online change (ladder mode) • Up to 300ms for online change (files)

(To the next page)

Table6.37 Restrictions that apply to all the high speed interrupt setting (continued)

Item	Restriction	When used
File register that is the same name as a program name	File register that is the same name as a program name cannot be used.	The high interrupt cannot be executed at the specified intervals since interrupts are disabled when switching to the file register that is the same name as a program name. Because of this, the following time is required. • 410μs for the standard RAM • 400μs+100μs×Number of program files for an SRAM card
Local device	Local devices cannot be used.	The high interrupt cannot be executed at the specified intervals since interrupts are disabled when switching to a local device. Because of this, the following time is required. • 390μs+170μs×n for the standard RAM • 390μs+950μs×n for an SRAM card (n: number of program files)
Command of the intelligent function module that accesses the CPU module	CPU access commands cannot be issued from the intelligent function module that accesses the CPU module such as the QJ71C24 or QJ71E71.	Since interrupts are disabled when issuing a CPU access command, the high speed interrupt cannot be executed at the specified intervals due to delay of the start. • Reading or writing of N points: (0.07×N+34)μs • Random reading or writing of N points: (0.07×N+101)μs
Monitoring via other stations	Monitoring via any of CC-Link IE Controller Network modules, MELSECNET/H modules, or intelligent function modules such as the QJ71C24 cannot be executed during monitoring its own station .	If a self-monitoring request and a request for monitoring via any intelligent function module are overlapped, interrupt-disabled time is increased. The high speed interrupt cannot be executed at the specified intervals due delay of the start (102μs).
Interrupt counter	The interrupt counter corresponding to interrupt pointer I49 cannot be used.	Even if the interrupt counter is set, the setting for I49 is ignored, and the high speed interrupt I49 is executed normally. (Interrupt programs of other interrupt pointers are not executed, and the interrupt counter is executed.)

(2) Restrictions that apply to the high speed interrupt only

Table6.38 Restrictions that apply to the high speed interrupt only

Item	Restriction	When used
Device comment	In high speed interrupt programs, any device comment that has the same name as the program name is not saved and restored.	The written device comment of the high speed interrupt program is changed.
Index register	In high speed interrupt sequences, the index register is not saved and restored.	The index register of the high speed interrupt program is changed.
Access execution flag (SM390)	In high speed interrupt programs, the access execution flag SM390 is not saved and restored.	The SM390 value of the high speed interrupt program is changed.
Forced on/off	The high speed X/Y refresh area cannot be forced to turn on or off.	The forced on/off is not executed in the high speed interrupt, being ignored. (No timeout error will occur.)
Monitoring condition setting	No setting is allowed for the high speed interrupt program.	The setting is not executed normally. (No timeout error will occur.)
Execution time measurement	No setting is allowed for the high speed interrupt program.	The setting is not executed, being ignored. (No timeout error will occur.)

(3) Items that delay the high speed interrupt start due to disabled interrupt

Table6.39 Items that delay the high speed interrupt start due to disabled interrupt

Item	Precaution
Instruction	During instruction execution, any interrupt is disabled.
Link refresh	During a refresh (bus access), any interrupt is disabled. For a refresh of CC-Link IE Controller Network modules, MELSECNET/H modules, CC-Link modules, and intelligent function modules, waiting time is up to 37.5μs when mounting these modules on a main base unit, and up to 40μs when using an extension base unit.
Execution of multiple programs	During execution of multiple programs or during program switching, any interrupt is disabled. The waiting time is 30μs. For the high speed interrupt function setting, use of a single program is recommended.
Monitoring	The following waiting time is required for the ladder monitor, the device batch monitor, and the entry data monitor. (0.096× number of device points+20)μs
AC DOWN	The high speed interrupt start delays up to 20ms.

(4) Items other than the above ((1) to (3))

(a) Interrupt program/Fixed scan program setting in the PLC parameter dialog box

The "High speed execution" setting is disabled for the high speed interrupt function.

(b) High speed buffer transfer

If the file register area outside the setting range (the range exceeding the maximum points) is used, no error will occur and data will not be transferred to the out-of-range area. (Data in other devices will not be destroyed.)

(c) Programming precautions

There are the same kinds of programming precautions as other interrupt programs. (☞ Section 2.2.3)

6.23 Interrupt from Intelligent Function Module Note6.27

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 the communication by nonprocedural protocol
- Data reception during the communication by bidirectional protocol

Using an interrupt program enables a CPU module to receive data quickly.

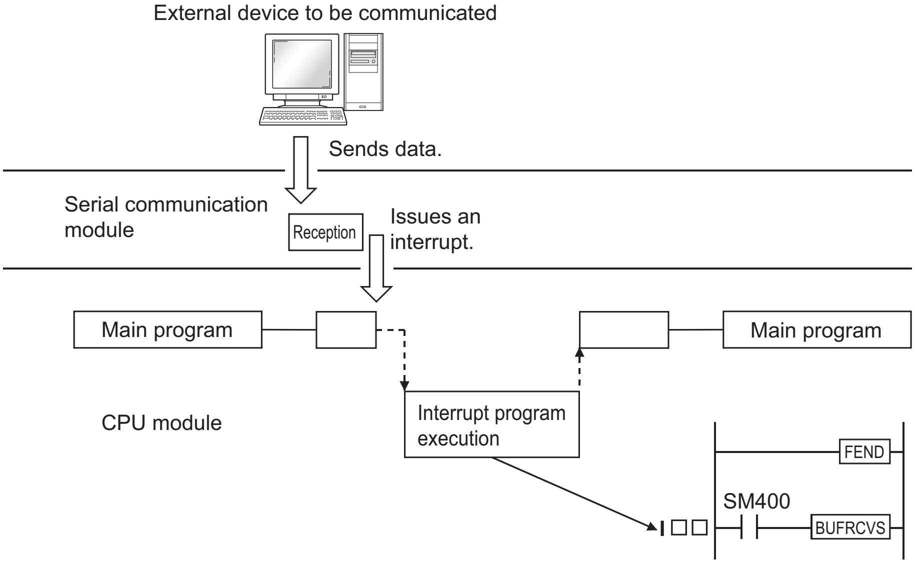


Figure 6.91 Interrupt from a serial communication module

(1) Setting an interrupt from the intelligent function module

To execute an interrupt program by an interrupt from the intelligent function module, select "Intelligent function module interrupt pointer setting" in "Interrupt pointer setting" at "Intelligent function module setting" of the PLC system tab in the PLC parameter dialog box.
To configure intelligent function module parameters at 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 of the intelligent function module used

Remark

For the numbers of interrupt pointers available for an interrupt from the intelligent function module, refer to Section 9.10.

Note6.27 **Basic**

When using an interrupt from the intelligent function module for the Basic model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.1)

6.24 Serial Communication Function Note6.28

(1) Definition

This function communicates in the MC protocol[*1] by connecting the RS-232 interface of the CPU module, personal computer, and HMI by RS-232 cable.

This section describes the specifications, functions, and various settings of the function.

*1: The MC protocol is an 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).
For the MC protocol, refer to the following.
☞ MELSEC Communication Protocol Reference Manual

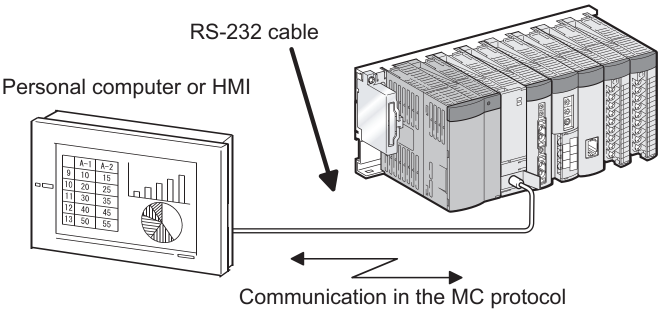


Figure 6.92 Communication with personal computer or HMI

Point

- A personal computer or HMI can communicate with a CPU module by the serial communication function only when the CPU module is connected to it.
The CC-Link IE Controller Network, MELSECNET/H, Ethernet, or CC-Link cannot be communicated with another station.
- The serial communication function is not used for connection of GX Developer or GX Configurator with the CPU module.

Note6.28 Basic High performance Process Redundant

When using the Q00JCPU, High Performance model QCPU, Process CPU, or Redundant CPU, the serial communication function cannot be used.

(2) Specifications

(a) Transmission specifications

Table6.40 shows the transmission specifications of RS-232 for the serial communication function of the CPU module.

Check that the specifications of the personal computer and HMI match those of Table6.40 before using the function.

Table6.40 Transmission specifications of the serial communication function

Item	Default	Setting Range
Communication method	Full-duplex communication	-
Synchronization method	Asynchronous method	-
Transmission speed*1	19.2kbps	9.6kbps, 19.2kbps, 38.4kbps, 57.6kbps, 115.2kbps
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	Checked/not checked
Transmission wait time*1	No waiting time	No waiting time, 10ms to 150ms (in increments of 10ms)
RUN write setting*1	Prohibited (deselected)	Permit (checked), prohibited (deselected)
Overall cable distance	15m	-

*1: Can be set in the PLC parameter dialog box.

*2: Table6.41 shows the relationship between the MC protocol formats and frames.

Table6.41 Relationship between the MC protocol formats and frames

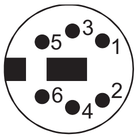
Function		Format 4	Format 5
Communication in ASCII code	QnA-compatible 3C frame	○	×
	QnA-compatible 4C frame	○	×
Communication in binary code	QnA-compatible 4C frame	×	○

○ : Available, × : Unavailable

(b) RS-232 connector specifications

Table6.42 shows the specifications of the RS-232 connector for the CPU module.

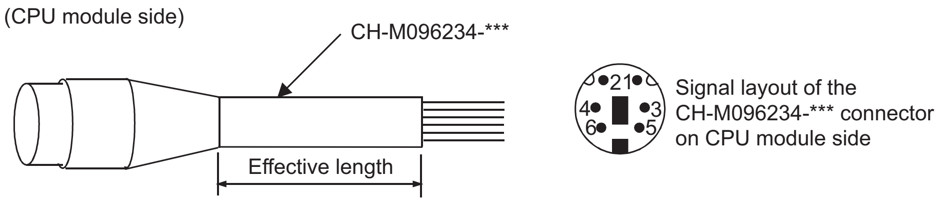
Table6.42 RS-232 connector specifications

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 set ready
	6	ER(DTR)	Data terminal ready

(c) RS-232 cable

The following RS-232 cable can be used for connection of the CPU module to the personal computer or HMI.

- 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 units of 0.1m.


(CPU module side)
CH-M096234-***
Effective length
2
1
4
3
6
5
Signal layout of the CH-M096234-*** connector on CPU module side

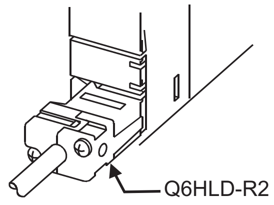
Pin number	1	2	3	4	5	6	Metal shell
Signal	**RD**	**SD**	**SG**	**-**	**DR**	**ER**	
Wire core	Red	Black	Green/ White	-	Yellow	Brown	Shield

Figure 6.93 Effective length and signal layout of RS-232 cable

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



(3) Functions

Table6.43 shows the MC protocol commands that can be executed by the serial communication function.

Table6.43 MC protocol commands supported by the serial communication function

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.	4480 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 specified monitor condition cannot be used.

*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.

(4) Accessible devices

Table6.44 shows accessible devices by the serial communication function.

Table6.44 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

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.

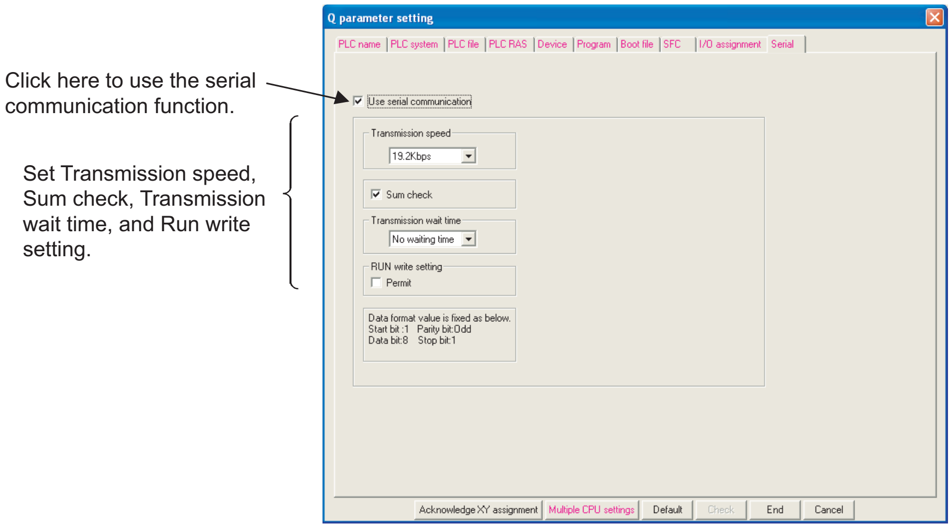
*2: Devices of DX/DY1000 or later are not available. Use X/Y devices to access devices of X/Y1000 or later.

*3: The Q00UJCPU does not support these devices.

(5) Setting of transmission specifications

Set Transmission speed, Sum check, Transmission wait time, and Run write setting of the serial communication function in the Serial tab of the PLC parameter dialog box.

- Select "Use serial communication" in communication with the personal computer or HMI.
- Set Transmission speed, Sum check, Transmission wait time, and Run write setting in the tab.



Click here to use the serial communication function.

Set Transmission speed, Sum check, Transmission wait time, and Run write setting.

☑ Use serial communication

Transmission speed: 19.2Kbps

☑ Sum check

Transmission wait time: No waiting time

RUN write setting: □ Permit

Data format value is fixed as below.
Start bit :1 Parity bit:Odd
Data bit:8 Stop bit:1

Figure 6.94 Serial tab

(6) Precautions

(a) Switching connection to GX Developer during communication with such as HMI

The CPU module can switch connection to GX Developer during communication with the personal computer or HMI with the serial communication function.

However, the personal computer or HMI in communication by the serial communication function results in a communication error.

For startup methods of the personal computer and HMI when they are reconnected to the CPU module, refer to the manual for used device.

(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 GX Developer 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 "Sumcheck enable" is set, 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.

(7) Error codes during communication with the serial communication function

Table6.45 shows the error codes, error description, and corrective actions sent from the CPU module to the external device when an error occurs during communication with the serial communication function.

Table6.45 Error codes sent from the CPU module to external device

Error code (Hexadeci-mal)	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 QCPU User's Manual (Hardware Design, Maintenance and Inspection), and take corrective action.
7153_H	Frame length error	The length of received message is outside the permissible range.	Review the sent message. Modify the message so that the number of access points may be within the permissible range.
7155_H	Unregistered monitor error	A monitor request was given before monitor registration.	Give a monitor request after registering a device to be monitored..
7164_H	Request data error	The requested data or device specification method is wrong.	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.
7168_H		The selected command cannot be executed in the RUN status.	Set the CPU module to STOP and restart communication.
$716D_H$	Monitor registration error	The QnA-compatible 3C/4C frame was not used for monitor registration.	Perform the monitor registration again.
$7E40_H$	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.
$7E4F_H$	Number of device points error	The number of access points is incorrect.	Check and correct the sent message of the external device and restart communication.
$7E5F_H$	Request destination module I/O number error	The I/O number of the request destination module is error.	Correct the I/O number of the module to which data are sent.
$7E64_H$	Number of registration points range error	The number of registration points (word/bit) is outside the range.	Correct the setting value of the number of registration points (word/bit).
$7F01_H$	Buffer full error	The next data was received before processing received data.	Increase the transmission intervals by such as handshake with the external device.
$7F21_H$	Receive header section error	The command (frame) section specified is in error. The ASCII code received cannot be converted into binary.	Check and correct the sent message of the external device and restart communication.
$7F22_H$	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.

(To the next page)

Table6.45 Error codes sent from the CPU module to external device (continued)

Error code (Hexadecimal)	Error item	Description	Corrective action
7F24H	Sumcheck error	The calculated sumcheck does not match the received sumcheck.	Review the sumcheck of external device..
7F67H	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.
7F69H	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.
7F6AH	Buffer full error	The receive buffer of the OS overflew and the received data was skipped.	Perform DTR control and make communication to prevent a buffer full error.
F□□□H	-	Error detected by the MELSECNET/H network system	Check and correct the sent message of the external device and restart communication. (The station number may not be set. Communication with other stations via MELSECNET/H and Ethernet is not made.)

6.25 Service Processing

Service processing is the communication processing with GX Developer and external devices.
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 and RS-232 cables (Communications with GX Developer or GOT.)

6.25.1 Module service interval time read

The module service interval designates the intervals of transient requests such as monitoring, test, and program writing and reading.
The CPU module can monitor the service interval time (time from service acceptance to next service acceptance) of the intelligent function module, network module, or GX Developer.
This indicates the frequency of access from external devices to the CPU module.

(1) Reading method

Operate the special relay and special registers shown inTable6.46.

Table6.46 Special relay and special registers that read the module service interval time

Number	Name	Description
SM551	Module service interval time read	Turning off and then on this relay reads the module service interval time of the intelligent function module specified at the special register SD550 to SD551 and SD552 (on: Reading, off: Non-processing).
SD550	Service interval measurement module	Set the I/O number of the module whose module service interval time is to be measured. Set the I/O number of the peripheral connected to RS-232 or USB interface of the CPU module to FFFFH.
SD551, SD552	Service interval read	Stores the service interval time read from the module specified at SD550 when SM551 is turned on. • SD551: In increments of 1ms (within the range of 0 to 65535) • SD552: In increments of 100 μs (within the range of 0 to 900, stored in increments of 100 μs) (Example) When the module service interval time is 123.4ms, SD551 is 123 and SD552 is 400.

(2) Program example

The program example of Figure 6.95 reads the module service interval time of the intelligent function module at X/Y160.

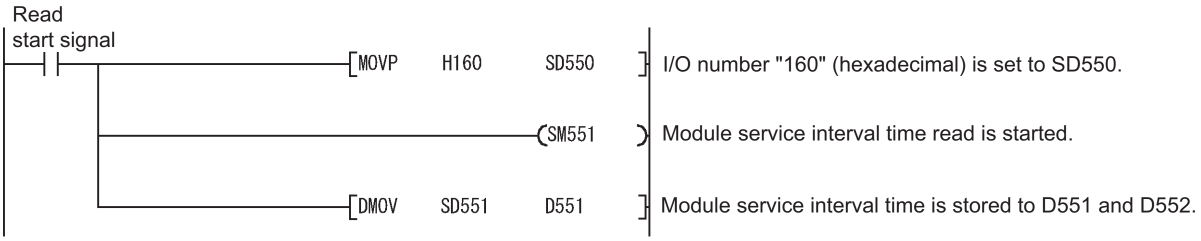

Read
start signal
MOVP H160 SD550
I/O number "160" (hexadecimal) is set to SD550.
SM551
Module service interval time read is started.
DMOV SD551 D551
Module service interval time is stored to D551 and D552.

Figure 6.95 Program example of module service interval time read

Point

- ● The access interval in cyclic communication from a network module is not stored.
- ● To read the service interval time when access is made from GX Developer in another station on the network, set the I/O number of the network module.

6.26 Initial Device Value Note6.29

(1) Definition

This function registers data used in a program to the device or the buffer memory of the intelligent function module without a program.

(2) Application

Using an initial device value can omit device data setting program by initial processing program.

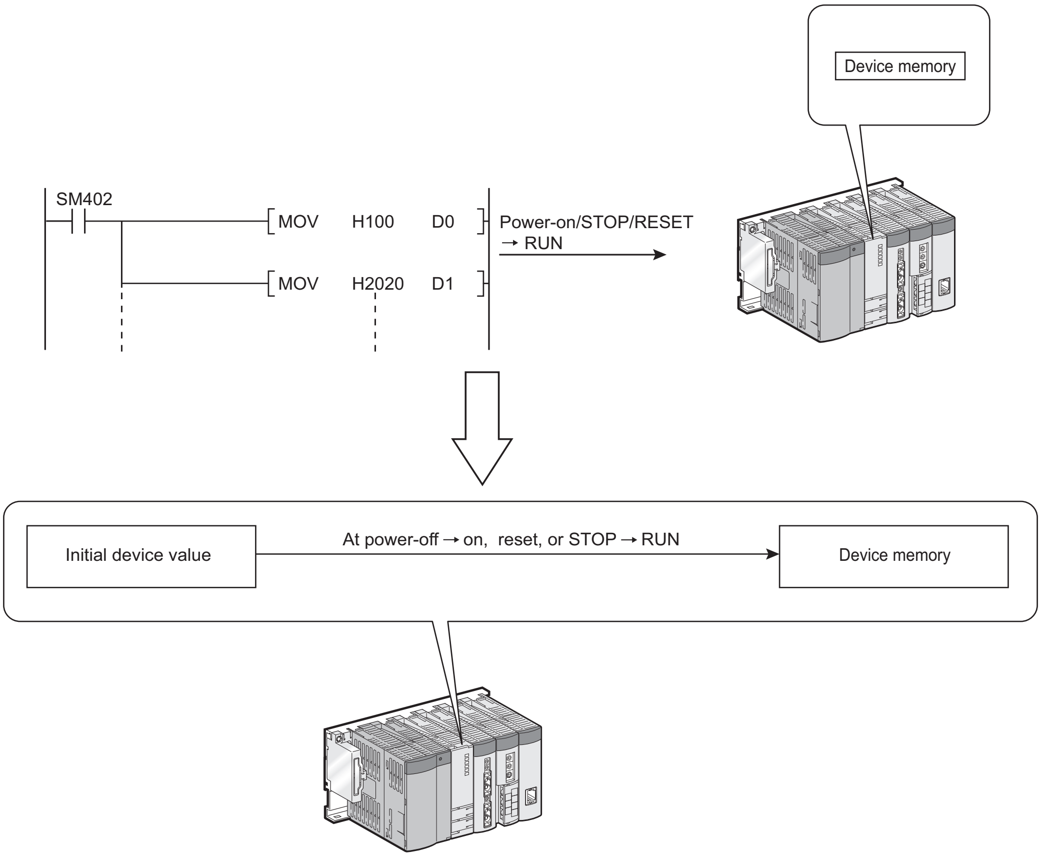


Figure 6.96 Data setting by initial program

Note6.29 **Basic**

When using the initial device value for the Basic model QCPU, check the versions of the CPU module and GX Developer. (Appendix 2.1)

(3) 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.

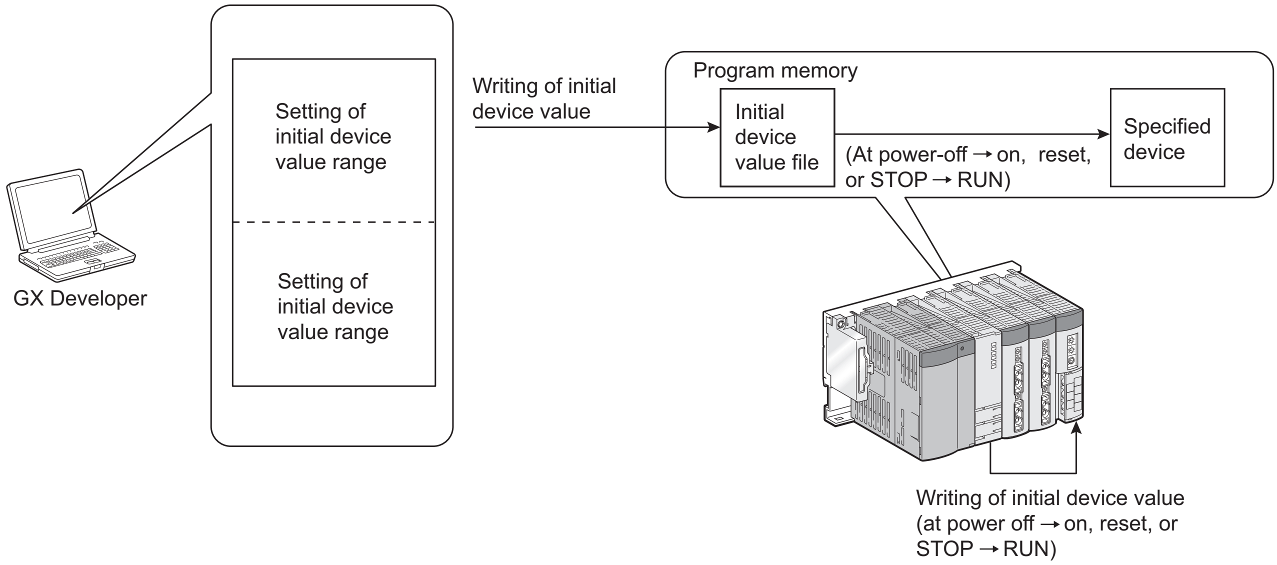

GX Developer
Setting of initial device value range
Setting of initial device value range
Writing of initial device value
Program memory
Initial device value file
(At power-off → on, reset, or STOP → RUN)
Specified device
Writing of initial device value (at power off → on, reset, or STOP → RUN)

Figure 6.97 Flow of writing initial device value

(4) Devices that can be used[*1]

The following shows devices that can be used for initial device value.

- Current timer value (T)
- Current retentive timer value (ST)
- Current counter value (C)
- Data register (D)
- Special register (SD)
- Link register (W)
- Link special register (SW)
- File register (R)
- File register (ZR)
- Intelligent function module device (U□\G□)
- Link direct device (J□\W□, J□\SW□)

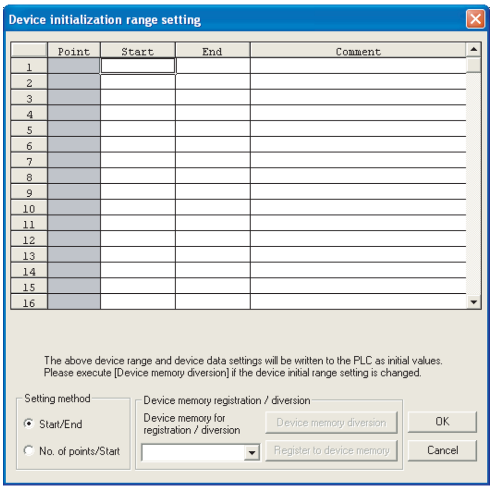
*1: For available ranges, refer to Section 9.1.

(5) Procedures and settings for using initial device values

To use initial device values, create initial device value data with GX Developer beforehand, and store the data as a initial device value file in the program memory, standard ROM, or memory card of the CPU module.

- Add an initial device value data to the project data list of GX Developer.
  The Device initialization range setting screen appears. Set the initial device value range.
  The number of settable points is up to 8000 points per range setting.

- Add device memory data in the project data list of GX Developer.
  The device memory screen appears. Set the initial device value data within the initial device value range set above.



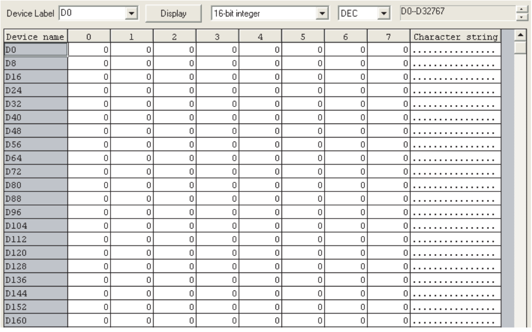
Device initialization range setting

	Point	Start	End	Comment
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

The above device range and device data settings will be written to the PLC as initial values.
Please execute [Device memory diversion] if the device initial range setting is changed.

Setting method
● Start/End
○ No. of points/Start

Device memory registration / diversion
Device memory for registration / diversion



Device Label D0 | 16-bit integer | DEC | D0--D32767

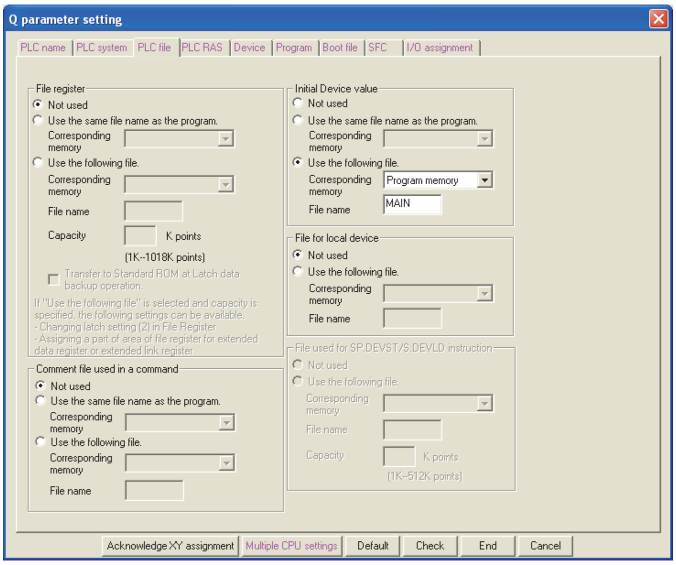
Device name	0	1	2	3	4	5	6	7	Character string
D0	0	0	0	0	0	0	0	0	....
D8	0	0	0	0	0	0	0	0	....
D16	0	0	0	0	0	0	0	0	....
D24	0	0	0	0	0	0	0	0	....
D32	0	0	0	0	0	0	0	0	....
D40	0	0	0	0	0	0	0	0	....
D48	0	0	0	0	0	0	0	0	....
D56	0	0	0	0	0	0	0	0	....
D64	0	0	0	0	0	0	0	0	....
D72	0	0	0	0	0	0	0	0	....
D80	0	0	0	0	0	0	0	0	....
D88	0	0	0	0	0	0	0	0	....
D96	0	0	0	0	0	0	0	0	....
D104	0	0	0	0	0	0	0	0	....
D112	0	0	0	0	0	0	0	0	....
D120	0	0	0	0	0	0	0	0	....
D128	0	0	0	0	0	0	0	0	....
D136	0	0	0	0	0	0	0	0	....
D144	0	0	0	0	0	0	0	0	....
D152	0	0	0	0	0	0	0	0	....
D160	0	0	0	0	0	0	0	0	....

Figure 6.98 Device initialization range setting screen and device memory screen

Point

When changing the setting on the Device initialization range setting screen, always execute "Device memory registration/diversion".
For details of the "Device memory registration/diversion", refer to the following.
☞ GX Developer Version 8 Operating Manual

- Set “Initial Device value” in the PLC file tab of the PLC parameter dialog box.
    1) For the Basic model QCPU
    Select "Use." for "Initial Device value".
    2) For the High Performance model QCPU, Process CPU, and Redundant CPU
    Select the target memory that stores initial device value file and enter the file name.



Q parameter setting

File register
● Not used
○ Use the same file name as the program.
Corresponding memory
○ Use the following file.
Corresponding memory
File name
Capacity K points
(1K--1018K points)
□ Transfer to Standard ROM at Latch data backup operation.
If "Use the following file" is selected and capacity is specified, the following settings can be available.
- Changing latch setting (2) in File Register
- Assigning a part of area of file register for extended data register or extended link register

Comment file used in a command
● Not used
○ Use the same file name as the program.
Corresponding memory
○ Use the following file.
Corresponding memory
File name

Initial Device value
○ Not used
○ Use the same file name as the program.
Corresponding memory
● Use the following file.
Corresponding memory: Program memory
File name: MAIN

File for local device
● Not used
○ Use the following file.
Corresponding memory
File name

File used for SP.DEVST/S.DEVLD instruction
○ Not used
○ Use the following file.
Corresponding memory
File name
Capacity K points
(1K--512K points)

Figure 6.99 PLC file tab (High Performance model QCPU)

- Write the set initial device value and parameters to the CPU module.

(6) Precautions

(a) When initial device value and latch range are overlapped

In that case, initial device value 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 value are also reflected when the CPU module is set from STOP to RUN.
For an area where an initial device value is not to 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), the initial device value 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

When setting the following devices in the Device initialization range setting screen, 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□)

Remark

For details of the initial device value range setting, setting of the initial device value data, and writing of the initial device values to the CPU module, refer to the following.

☞ GX Developer Version 8 Operating Manual

6.27 Memory Check Function Note6.30

This function checks whether data in the memories of the CPU module are changed or not due to such as excessive electric noise.

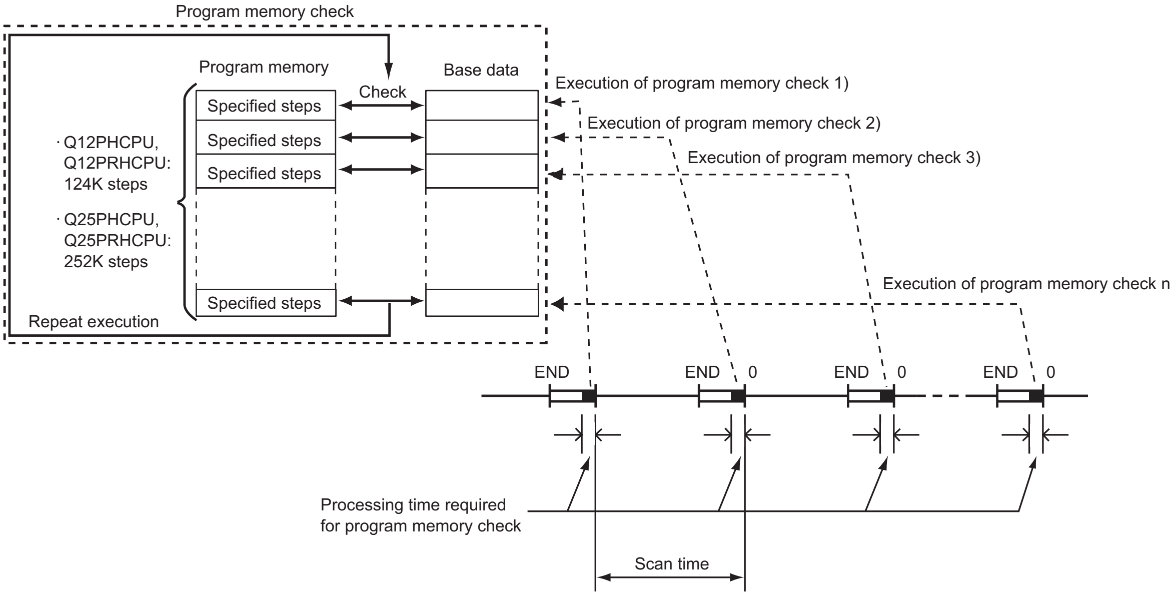

Program memory check
Program memory
Base data
Check
Specified steps
Specified steps
Specified steps
Specified steps
· Q12PHCPU, Q12PRHCPU: 124K steps
· Q25PHCPU, Q25PRHCPU: 252K steps
Repeat execution
Execution of program memory check 1)
Execution of program memory check 2)
Execution of program memory check 3)
Execution of program memory check n
END
END 0
END 0
END 0
Processing time required for program memory check
Scan time

Figure 6.100 Overview of memory check function

(1) Data to be checked

When the CPU module is set from STOP to RUN or END processing is performed, the data in execution in the program memory are compared with data, such as programs and parameters, when they were written to the program memory of the CPU module.

If they do not match, the CPU module judges that the data in the program memory have been changed and detects a stop error, "RAM ERROR" (error code: 1106).

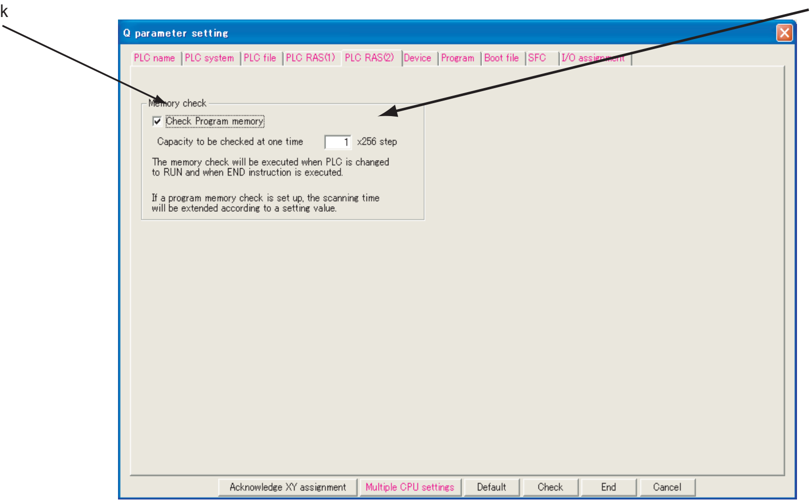
Note6.30 Basic High performance Process

The Basic model QCPU and High Performance model QCPU do not support the memory check function.

When using the function for the Process CPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.3)

(2) Setting for memory check

- Select the "Check Program memory" checkbox in the PLC RAS(2) tab of the PLC parameter dialog box.
- Enter a value for "Capacity to be checked at one time".



Memory check

☑ Check Program memory

Capacity to be checked at one time 1 x256 step

The memory check will be executed when PLC is changed to RUN and when END instruction is executed.

If a program memory check is set up, the scanning time will be extended according to a setting value.

Setting for program memory check

Setting range
· Q12PHCPU, Q12PRHCPU:
1 to 496(256 steps to 124K steps)
· Q25PHCPU, Q25PRHCPU:
1 to 1008(256 steps to 252K steps)

Figure 6.101 PLC RAS(2) tab

Data in the program memories of the CPU modules in the control system and standby system are checked.
If the standby system CPU module is in the backup mode, the data amount to be checked at a time is "1" (256 steps) regardless of setting for “Capacity to be checked at one time”.

(3) Execution timing

Data in a memory is checked at the following timing.

- When the CPU module is set from STOP to RUN: All areas in the program memory are checked.
- When END processing is performed: Data are checked at a time by the amount of Capacity to be checked at one time × 256 steps.

(4) Processing time

(a) When the CPU module is set from STOP to RUN

The scan time increases by the processing time shown below.

- Q12PHCPU and Q12PRHCPU: 434ms
- Q25PHCPU and Q25PRHCPU: 882ms

(b) When END processing is performed while the CPU module is in the RUN status

The scan time increases by the processing time shown below.

$$\text{(Extended scan time)} = 3.5 \times \frac{\text{(Capacity to be checked at one time)} \times 256}{1024} \text{ (ms)}$$

Consider the increase in the scan time when entering a value for "Capacity to be checked at one time" in the PLC RAS(2) tab of the PLC parameter dialog box.
If the “Capacity to be checked at one time” is set to "4", the processing time is shown below.
3.5 × 4 × 256/1024 = 3.5 (ms)

(5) Precautions

(a) Maximum delay time of error detection

The following shows the maximum delay time from rewrite of program memory data till detection of the rewritten data.

$$\text{Maximum delay time of error detection} = \frac{(\text{Maximum program memory capacity})^{*1}}{(\text{Capacity to be checked at one time}) \times 256} \times (\text{Scan time})\ (\text{ms})$$

*1: The maximum program memory capacity for each CPU module is shown below.

- 12PHCPU, Q12PRHCPU: 124K steps (124×1024 steps)
- Q25PHCPU, Q25PRHCPU: 252K steps (252×1024 steps)

The sequence program is executed until "RAM ERROR" is detected. Therefore, other errors may be detected before "RAM ERROR" is detected or an unexpected operation may be performed due to corrupt data in the program memory.

If “Capacity to be checked at one time” is set to "4" for the Q12PHCPU and the scan time is 10ms, the maximum delay time is as shown below.
(124×1024)/(4×256)×10 = 1240 (ms)

If an user setting system area is set while the program memory is formatted, the program memory capacity will reduce by the capacity set for the user setting system area.
For check methods of the user setting system area and program memory capacity, refer to Section 5.2.2.

(b) Instructions that cannot be used during memory check

If any of the following instructions is executed, "OPERATION ERROR" (error code: 4105) occurs.

- PLOADP instruction
- PUNLOADP instruction
- PSWAPP instruction

(c) Conditions to validate PLC parameters

After parameters set in the PLC parameter dialog box are written to the CPU module and either of the following operations is performed, whether the memory check is performed or not can be selected.

- Powering off and then on the CPU module
- Resetting the CPU module

(d) Memory check during execution of the COM instruction

If the COM instruction is executed, the memory check will not be performed.

CHAPTER7 COMMUNICATIONS WITH INTELLIGENT FUNCTION MODULE

(1) Intelligent function module

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.

For example, the analog-digital conversion module, one of the intelligent function modules, uses analog quantity by converting it into a digital value.

(2) Communication with intelligent function 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.

7.1 Communications between CPU Module and Intelligent Function Module

The following table shows the communication methods between the CPU module and intelligent function modules and the communication timing.

Table7.1 Communication methods with intelligent function module and the communication timing

Communication methods with intelligent function module		Communication timing					Reference
		Power off → on	CPU module is reset	STOP → RUN*2	Instruction execution	END processing	
GX Configurator	Initial setting	○	○	○	-	-	Section 7.1.1
	Auto refresh setting	-	-	-	-	○	
Initial device value		○	○	○	-	-	Section 7.1.2
FROM or TO instruction*1		-	-	-	○	-	Section 7.1.3
Intelligent function module device*1		-	-	-	○	-	Section 7.1.4
Intelligent function module dedicated instruction*1		-	-	-	○	-	Section 7.1.5

○: Executed, -: Not executed

*1: Indicates the program that uses the intelligent function module device, FROM or TO instruction, or the intelligent function module dedicated instruction.

*2: The RUN/STOP switch (RUN/STOP/RESET switch for the Basic model QCPU) is set from STOP → RUN (RUN LED flashes.) → STOP → RUN.

Point

- Data, such as initial settings used for communications with intelligent function module, are stored in the CPU module. For the storage location, refer to Section 5.1.1 and Section 5.1.2.
- Power off and then on or reset the CPU module to make the initial setting configured in GX Configurator taken effect.

7.1.1 Initial setting and auto refresh setting by GX Configurator

The initial setting and auto refresh setting can be made by adding in GX Configurator that is supported by the intelligent function module to GX Developer.
After the initial and auto refresh settings, data can be read or written without creating a program for communications with intelligent function modules.

(1) Starting GX Configurator

Select [Tools] → [Intelligent function module utility] → [Start] in GX Developer.

(2) Setting in GX Configurator

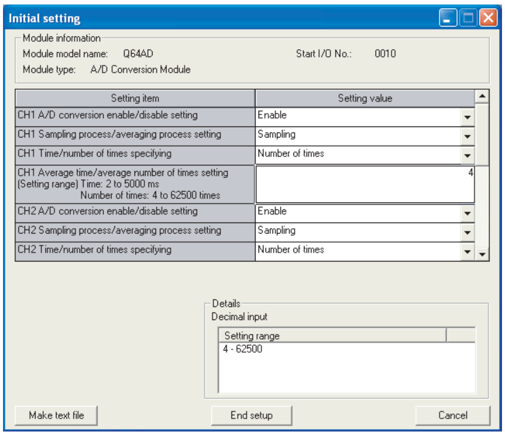
An example of initial setting and auto refresh setting for the A/D conversion module Q64AD is used in this section.

(a) Initial setting

The initial setting for the Q64AD offers the following four types.

- A/D conversion enable/disable setting
- Sampling process/averaging process setting
- Time/number of times specifying
- Average time/average number of times setting

Make the initial settings of the Q64AD in the Initial setting screen in GX Configurator as shown in Figure 7.1.



Module information
Module model name: Q64AD Start I/O No.: 0010
Module type: A/D Conversion Module

Setting item	Setting value
CH1 A/D conversion enable/disable setting	Enable
CH1 Sampling process/averaging process setting	Sampling
CH1 Time/number of times specifying	Number of times
CH1 Average time/average number of times setting (Setting range) Time: 2 to 5000 ms Number of times: 4 to 62500 times	4
CH2 A/D conversion enable/disable setting	Enable
CH2 Sampling process/averaging process setting	Sampling
CH2 Time/number of times specifying	Number of times

Details
Decimal input

Setting range
4 - 62500

Figure 7.1 Initial setting screen

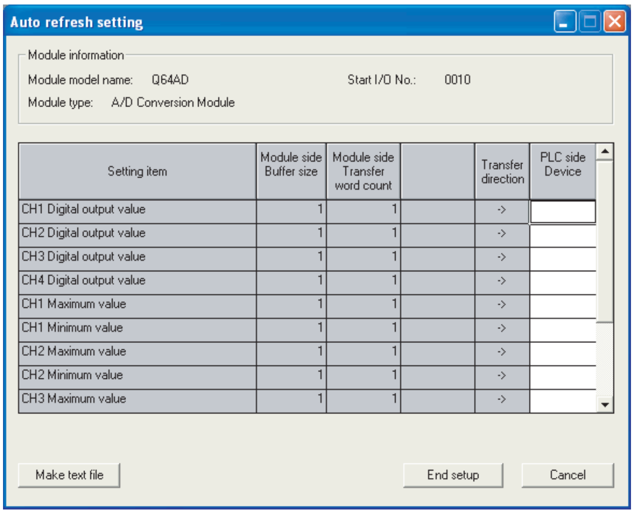
The initial setting data set in this screen are stored into the intelligent function module parameters of the CPU module.

(b) Auto refresh setting

The CPU module devices for storing the following data can be set in the Auto refresh setting screen.

- Digital output of the Q64AD
- Maximum and minimum values of the Q64AD
- Error codes

Make auto refresh settings of the Q64AD in the Auto refresh setting screen in GX Configurator as shown in Figure 7.2.



Auto refresh setting

Module information

Module model name: Q64AD

Start I/O No.: 0010

Module type: A/D Conversion Module

Setting item	Module side Buffer size	Module side Transfer word count		Transfer direction	PLC side Device
CH1 Digital output value	1	1		->	
CH2 Digital output value	1	1		->	
CH3 Digital output value	1	1		->	
CH4 Digital output value	1	1		->	
CH1 Maximum value	1	1		->	
CH1 Minimum value	1	1		->	
CH2 Maximum value	1	1		->	
CH2 Minimum value	1	1		->	
CH3 Maximum value	1	1		->	

Figure 7.2 Auto refresh setting screen

The auto refresh setting data set in this screen are stored into the intelligent function module parameters of the CPU module.

(3) Limitation on the number of parameter settings

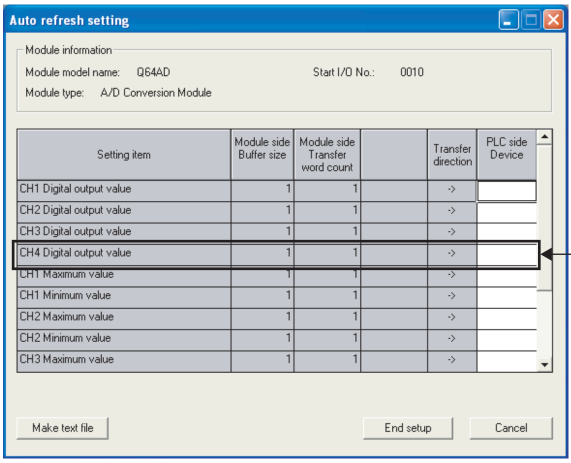
Limitations are placed on the number of parameters (initial setting and auto refresh setting) set in GX Configurator.

When multiple intelligent function modules are mounted, make setting in GX Configurator so that the number of parameter settings for all intelligent function modules may not exceed the limitation shown in Table7.2.

Table7.2 Number of parameters set in GX Configurator

CPU module	Number of parameter settings	
	Initial setting	Auto refresh setting
Basic model CPU, High Performance model QCPU, Process CPU, Redundant CPU, MELSECNET/H remote I/O station	512	256

One line in the Auto refresh screen is counted as one parameter setting.



Auto refresh setting

Module information

Module model name: Q64AD

Start I/O No.: 0010

Module type: A/D Conversion Module

Setting item	Module side Buffer size	Module side Transfer word count		Transfer direction	PLC side Device
CH1 Digital output value	1	1		->	
CH2 Digital output value	1	1		->	
CH3 Digital output value	1	1		->	
CH4 Digital output value	1	1		->	
CH1 Maximum value	1	1		->	
CH1 Minimum value	1	1		->	
CH2 Maximum value	1	1		->	
CH2 Minimum value	1	1		->	
CH3 Maximum value	1	1		->	

This one line is counted as one parameter setting.
Blanks are not counted.
Add the number of lines on this screen to those of other intelligent function modules.

Figure 7.3 Counting the number of parameter settings

(4) Precautions

For the AnS/A series special function modules, parameters cannot be set with GX Configurator.

Remark

For details of GX Configurator, refer to the manual for the intelligent function module used.

7.1.2 Initial setting by initial device value

(1) Initial device value

Using an initial device value (☞ Section 6.26) allows the initial setting of the intelligent function module without a program.
The set initial device values are written from the CPU module to the intelligent function module when the CPU module is powered off and then on, reset, or set from STOP to RUN.

(2) Setting initial device values

Use GX Developer to set the following.

- Set the device data of the intelligent function module (☞ Section 9.5.1) used as the initial device value to the device memory.
- In the initial device value setting, specify the device range of the intelligent function module used as the initial device value.

7.1.3 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.

Remark

- For details of the FROM and TO instructions, refer to the following.
  ☞ MELSEC-Q/L Programming Manual (Common Instruction)

- For details of the buffer memory of the intelligent function module, refer to the following.
  ☞ Manual for the intelligent function module used

7.1.4 Communications using the intelligent function module device

(1) Intelligent function module device

The intelligent function module device (☞ Section 9.5.1) represents the buffer memory of the intelligent function module as one of the CPU module devices.

The data stored in the buffer memory of the intelligent function module can be treated by the sequence instruction as well as the device memory.

A programming example is given below for the case where "100" is written to the buffer memory of an address 0 of the intelligent function module whose I/O number is X/Y20 to X/Y2F.

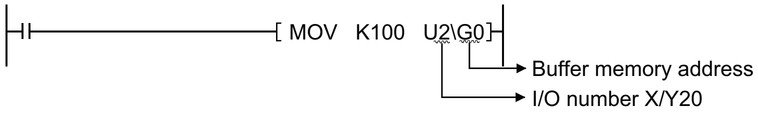


Figure 7.4 Use of the intelligent function module device

(2) Difference from the FROM and TO instructions

The intelligent function module device processes data read from the intelligent function module with one instruction as the device can be treated as one of the CPU module devices.

A program example is given below for the case where data read from the intelligent function module are added and then the result is stored in D2.

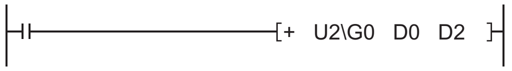


Figure 7.5 Application example of the intelligent function module device

This reduces the number of steps in whole program.

The instruction processing time indicates the time required for instruction execution and accessing the intelligent function module.

Point

The intelligent function module device accesses the intelligent function module every time when an instruction is executed. When writing or reading buffer memory data using multiple intelligent function module devices in a sequence program, write or read the data with the FROM or TO instruction in one location of the program.

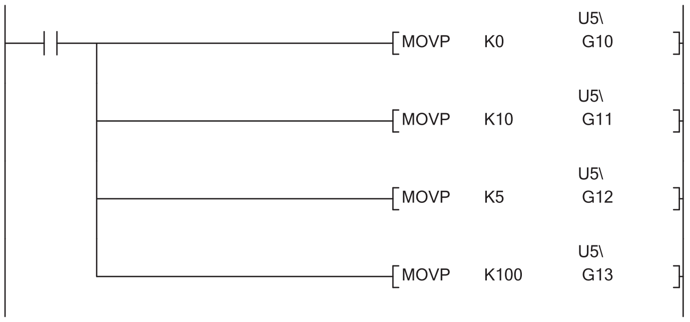
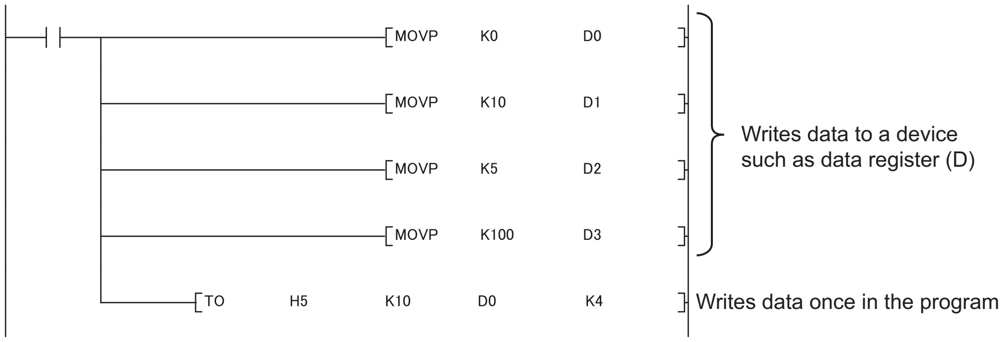


Figure 7.6 Writing using multiple intelligent function module devices



Writes data to a device such as data register (D)

Writes data once in the program

Figure 7.7 Writing using the TO instruction

7.1.5 Communications using the intelligent function module dedicated instruction

(1) Intelligent function module dedicated instruction

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

(a) Example with the 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.

[Transmission by the OUTPUT instruction]

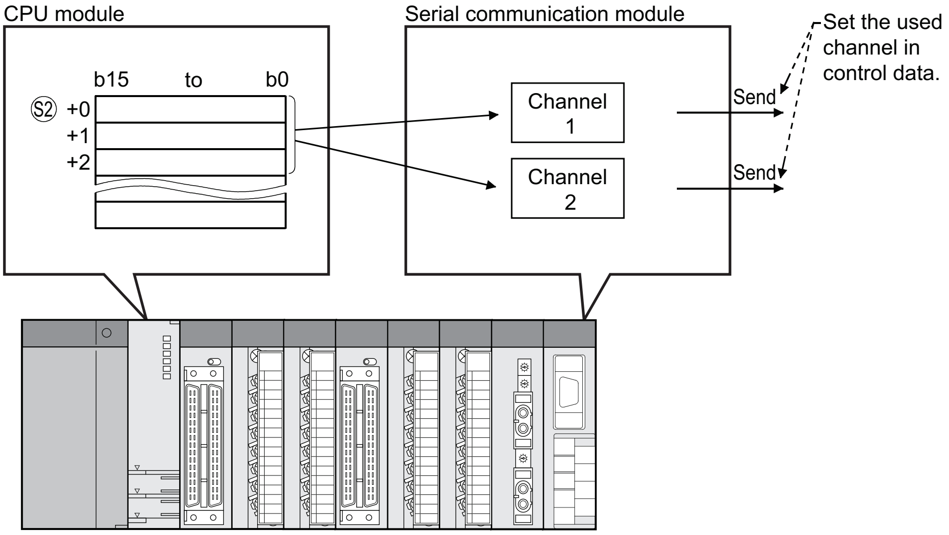

CPU module
b15
to
b0
S2
+0
+1
+2
Serial communication module
Channel 1
Channel 2
Send
Send
Set the used channel in control data.

Figure 7.8 Communications with external devices using the OUTPUT instruction

(2) Processing of the intelligent function module dedicated instruction

Some of intelligent function module dedicated instructions can specify the completion device.
This completion device turns on for one scan when an instruction execution is completed.
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.

(3) Precautions

(a) When the CPU module is set from RUN to STOP before the completion device turns on

When a intelligent function module dedicated instruction is executed and the CPU module is set from RUN to STOP before the completion device turns on, the completion device will not turn on until the CPU module is set to RUN and then finishes one scan.

(b) Available range

The intelligent function module dedicated instructions are not applicable to the intelligent function modules mounted on remote I/O stations of MELSECNET/H.
The instructions are applicable to the intelligent function modules mounted on the base unit or extension base units.

Remark

For intelligent function module dedicated instructions and completion devices, refer to the following.
☞ Manual for the intelligent function module used

7.2 Access to the AnS/A Series Special Function Modules Note7.1

(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

1) 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.
2) 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".

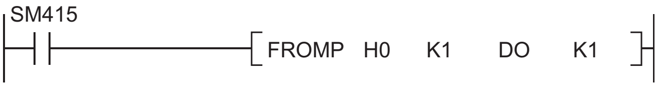


Figure 7.9 Program example

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)

Note7.1 **Basic** **Process** **Redundant**

The Basic model QCPU, Process CPU, and Redundant CPU do not support the AnS/A series special function modules.

CHAPTER8 PARAMETERS

This chapter describes the parameters required to be set for configuring a programmable controller system.

(1) Parameter types

The following parameters are provided for CPU module setting.

- PLC parameters (☞ Section 8.1)
  These parameters are set when a programmable controller is used.

- Redundant parameters (☞ Section 8.2)
  These parameters are set when a redundant system is configured using the Redundant CPU.

- Network parameters (☞ Section 8.3)
  These parameters are set when a programmable controller module is used in combination with any of CC-Link IE Controller Network modules, MELSECNET/H modules, Ethernet modules, and CC-Link modules.
- Remote password (☞ Section 8.4)
  Parameters are set when the remote password function is used for an Ethernet module, serial communication module, and modem interface module.

(2) Parameter setting method

Use GX Developer.

For the setting details, refer to the following.

☞ GX Developer Version 8 Operating Manual

Point

Grayed-out (unselectable) parameter settings in GX Developer are not available since the corresponding function is not supported.

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.

For the list of the special register (SD16 to SD26), refer to the following.

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

For the parameter updating procedure, refer to CHAPTER 11.

8.1 PLC Parameters

This section provides the list of PLC parameters and describes parameter details.

8.1.1 Basic model QCPU

(1) PLC name

A label and a comment for the CPU module are set.

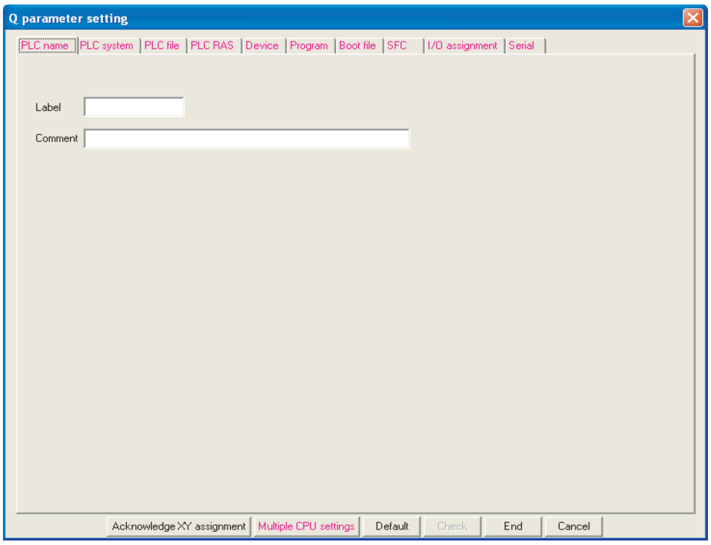


Figure 8.1 PLC name

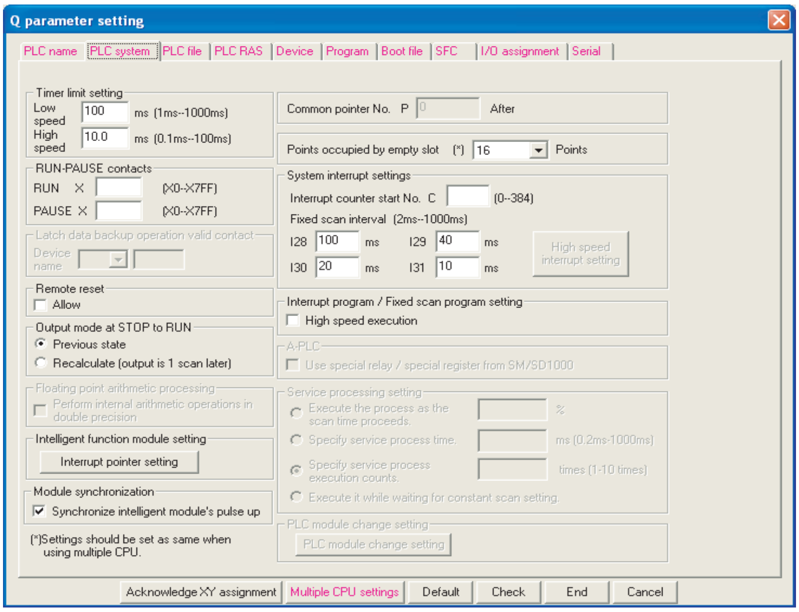
Table8.1 PLC name setting list

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	-

(2) PLC system

Parameters required for use of the CPU module are set.

The system can be controlled with default values.



Timer limit setting

Low speed 100 ms (1ms--1000ms)

High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts

RUN X (X0--X7FF)

PAUSE X (X0--X7FF)

Latch data backup operation valid contact

Device name

Remote reset

□ Allow

Output mode at STOP to RUN

● Previous state

○ Recalculate (output is 1 scan later)

Floating point arithmetic processing

□ Perform internal arithmetic operations in double precision

Intelligent function module setting

Interrupt pointer setting

Module synchronization

☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P 0 After

Points occupied by empty slot (*) 16 Points

System interrupt settings

Interrupt counter start No. C (0--384)

Fixed scan interval (2ms--1000ms)

I28 100 ms I29 40 ms

I30 20 ms I31 10 ms

High speed interrupt setting

Interrupt program / Fixed scan program setting

□ High speed execution

A-PLC

□ Use special relay / special register from SM/SD1000

Service processing setting

○ Execute the process as the scan time proceeds. %

○ Specify service process time. ms (0.2ms-1000ms)

● Specify service process execution counts. times (1-10 times)

○ Execute it while waiting for constant scan setting.

PLC module change setting

PLC module change setting

Figure 8.2 PLC system

Table8.2 PLC system setting list

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	Section 9.2.10
	High speed			0.1ms to 100.0ms (in increments of 0.1ms)	10.0ms	Section 9.2.10
RUN-PAUSE contact	RUN	1001_H	Set the contacts that control RUN/PAUSE of the CPU module. Setting of only the PAUSE contact is not allowed. (Setting of only the RUN contact or the RUN + PAUSE contacts is available.)	X0 to X7FF	Blank	Section 6.6.1
	PAUSE					Section 6.6.2
Remote reset		1002_H	Select whether to allow the remote reset from GX Developer.	Selected/deselected	Deselected	Section 6.6.3
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	Section 6.4
Intelligent function module setting (Interrupt pointer setting)		$100A_H$	Assign the interrupt pointers (I50 to I127) 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 I127	Blank	Section 6.23
Module synchronization		$100C_H$	Select whether to synchronize CPU module startup with intelligent function module startup.	Selected/deselected	Deselected	-

(To the next page)

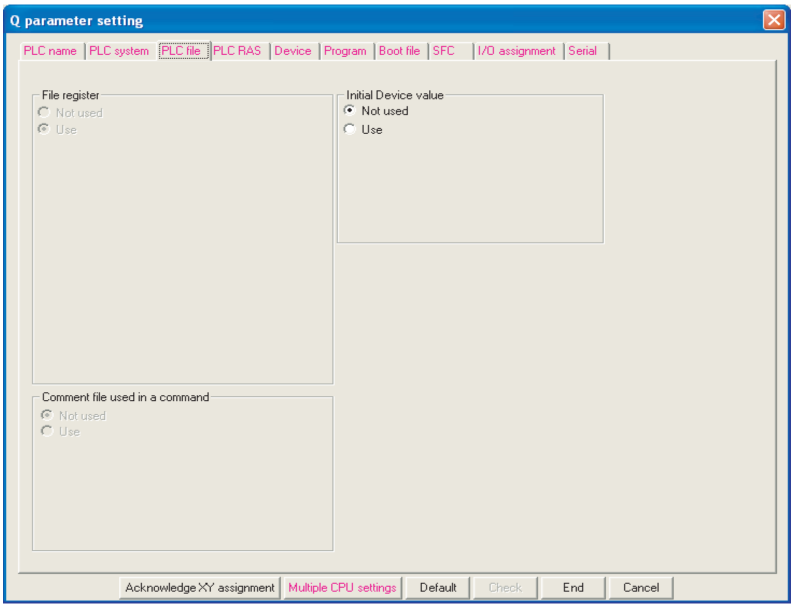
Table8.2 PLC system setting list (continued)

Item		Parameter No.	Description	Setting range	Default	Reference
Points occupied by empty slot		1007_H	Set the number of points for empty slots on the main/extension base units.	[Q00JCPU] 0, 16, 32, 64, 128, or 256 points [Q00CPU and Q01CPU] 0, 16, 32, 64, 128, 256, 512, or 1024 points	16 points	Section 4.2.2
System interrupt settings	Interrupt counter start No.	1008_H	Set the start number of interrupt counters.	C0 to C13184 (Up to "The number of counter setting points - 128")	Blank	Section 9.2.11(4)
	Fixed scan interval (n: 28 to 31)		Set each execution interval for the interrupt pointers (I28 to I31).	2ms to 1000ms (in increments of 1ms)	• I28: 100ms • I29: 40ms • I30: 20ms • I31: 10ms	Section 9.10
Interrupt program/Fixed scan program setting*1		1008_H	Enable or disable high speed execution of interrupt programs.	Selected/deselected	Deselected	Section 2.2.3

*1: The Basic model QCPU does not support the use of fixed scan execution type programs.

(3) PLC file

Parameters required for the files used in the CPU module are set.



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment | Serial

File register
○ Not used
● Use

Initial Device value
● Not used
○ Use

Comment file used in a command
● Not used
○ Use

Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

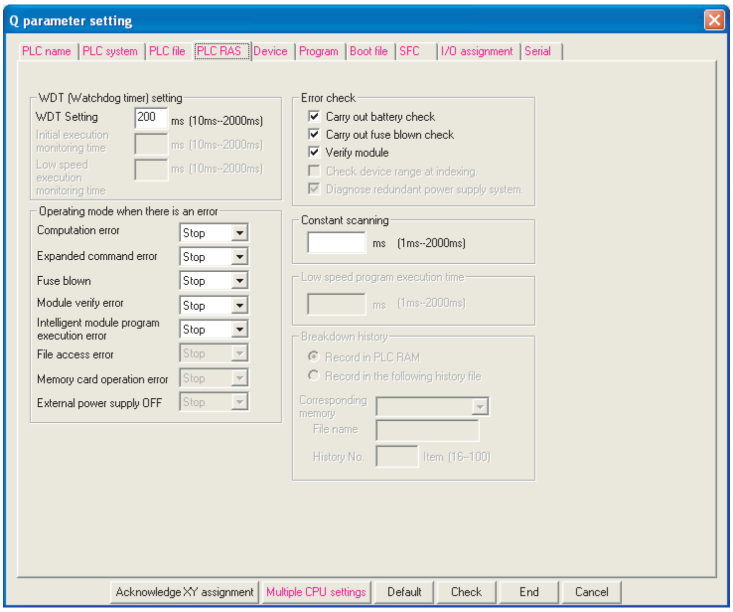
Figure 8.3 PLC file

Table8.3 PLC file setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Initial Device value	1102_H	Set a file for initial values of the devices used for the CPU module.	Not used/Use	Not used	Section 6.26

(4) PLC RAS

Parameters required for performing the RAS functions are set.



WDT (Watchdog timer) setting
WDT Setting 200 ms (10ms--2000ms)
Initial execution monitoring time ms (10ms--2000ms)
Low speed execution monitoring time ms (10ms--2000ms)

Operating mode when there is an error
Computation error Stop
Expanded command error Stop
Fuse blown Stop
Module verify error Stop
Intelligent module program execution error Stop
File access error Stop
Memory card operation error Stop
External power supply OFF Stop

Error check
☑ Carry out battery check
☑ Carry out fuse blown check
☑ Verify module
□ Check device range at indexing.
☑ Diagnose redundant power supply system.

Constant scanning
ms (1ms--2000ms)

Low speed program execution time
ms (1ms--2000ms)

Breakdown history
● Record in PLC RAM
○ Record in the following history file
Corresponding memory
File name
History No. Item (16--100)

Figure 8.4 PLC RAS

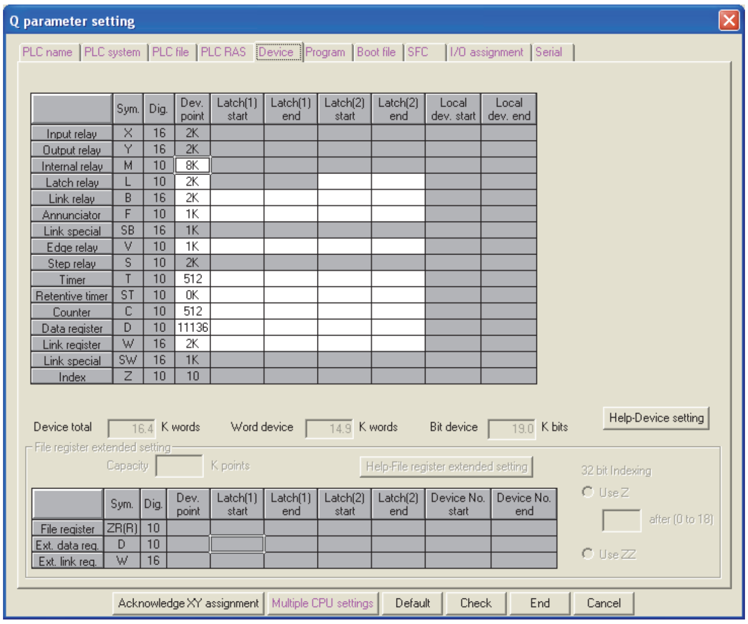
Table8.4 PLC RAS setting list

Item		Parameter No.	Description	Setting range	Default	Reference
WDT (Watchdog timer) setting	WDT setting	3000H	Set a watchdog timer value for the CPU module.	10ms to 2000ms (in increments of 10ms)	200ms	Section 6.16
Operating mode when there is an error	Computation error	3002H	Set the operation mode of the CPU module when an error is detected.	Stop/Continue	Stop	Section 6.17
	Expanded command error[*1]					
	Fuse blown					
	Module verify error					
	Intelligent module program execution error					
Error check	Carry out battery check	3001H	Enable or disable detection of the specified error.	Selected/deselected	Deselected	Section 6.17
	Carry out fuse blown check					
	Verify module					
Constant scanning		3003H	Set a constant scan time value.	1ms to 2000ms (in increments of 1ms)	Blank	Section 6.2

[*1] The item is provided for future extension.

(5) Device

Number of points and latch range are set for each device.



	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Local dev. start	Local dev. end
Input relay	X	16	2K						
Output relay	Y	16	2K						
Internal relay	M	10	8K						
Latch relay	L	10	2K						
Link relay	B	16	2K						
Annunciator	F	10	1K						
Link special	SB	16	1K						
Edge relay	V	10	1K						
Step relay	S	10	2K						
Timer	T	10	512						
Retentive timer	ST	10	0K						
Counter	C	10	512						
Data register	D	10	11136						
Link register	W	16	2K						
Link special	SW	16	1K						
Index	Z	10	10						

Device total 16.4 K words Word device 14.9 K words Bit device 19.0 K bits

File register extended setting

Capacity K points

	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Device No. start	Device No. end
File register	ZR(R)	10							
Ext. data reg.	D	10							
Ext. link reg.	W	16							

32 bit Indexing

○ Use Z

after (0 to 18)

○ Use ZZ

Figure 8.5 Device

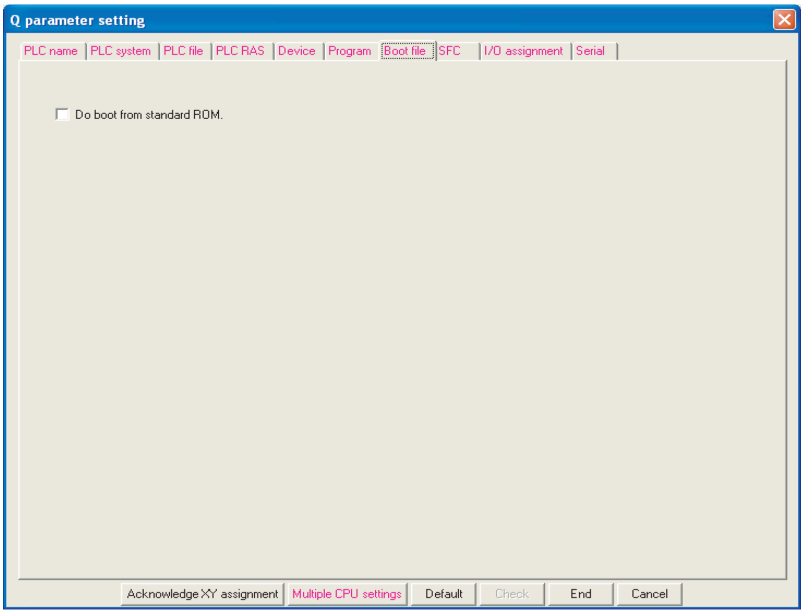
Table8.5 Device setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Device points[*1]	2000H	Set the number of device points that is appropriate to the system.	X(2K), Y(2K), S(2K), SB(1K), and SW(1K) are fixed. Setting is available within the range of 16.4K words in total, including the above fixed points (1.5K words). One device: Up to 32K points (No restrictions on the total points for bit devices.)	• X: 2K • Y: 2K • M: 8K • L: 2K • B: 2K • F: 1K • SB: 1K • V: 1K • S: 2K • T: 512 • ST: 0K • C: 512 • D: 11136 • W: 2K • SW: 1K	Section 9.1, Section 9.2
Latch (1) start/end (Latch clear valid)	2001H	Set a latch range (start and end device numbers), which can be cleared by remote latch clear operation.	Setting is available for only one range for each of B, F, V, T, ST, C, D, and W devices.	Blank	Section 3.7, Section 6.3
Latch (2) start/end (Latch clear invalid)	2002H	Set a latch range (start and end device numbers), which cannot be cleared by remote 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	Section 3.7, Section 6.3

*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.

(6) Boot file

Whether to perform a boot from the standard ROM is set.



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment | Serial

□ Do boot from standard ROM.

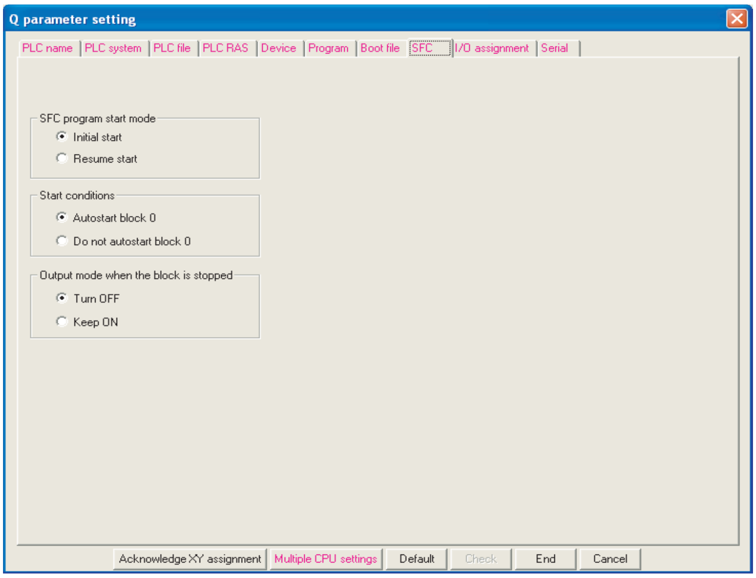
Figure 8.6 Boot file

Table8.6 Boot file setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Do boot from standard ROM.	-	Select whether to perform a boot from the standard ROM.	Selected/deselected	Deselected	Section 5.1.5

(7) SFC

The mode and conditions for starting an SFC program, and the output mode in the case of a block stop are set.



SFC program start mode
● Initial start
○ Resume start

Start conditions
● Autostart block 0
○ Do not autostart block 0

Output mode when the block is stopped
● Turn OFF
○ Keep ON

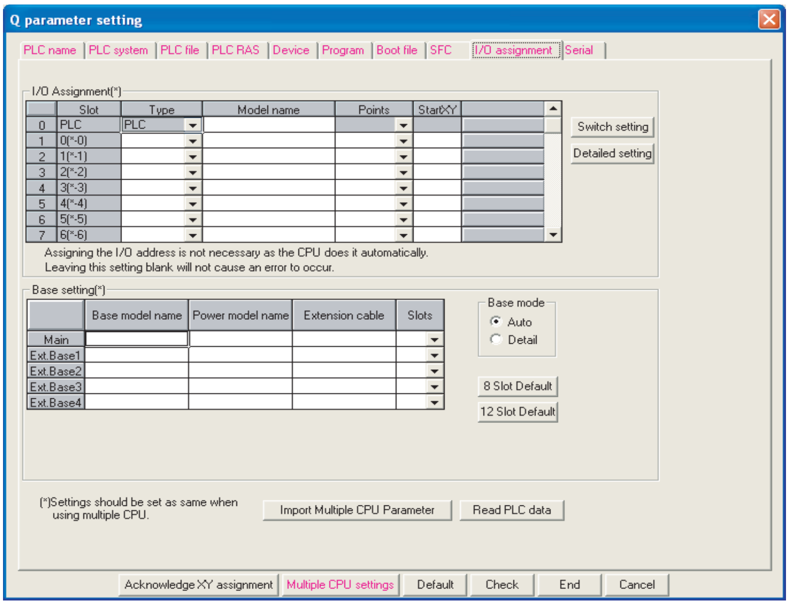
Figure 8.7 SFC

Table8.7 SFC setting list

Item	Parameter No.	Description	Setting range	Default	Reference
SFC program start mode	8002H	Set the mode and conditions for starring an SFC program, and also set the output mode in case a program block is stopped.	Refer to the MELSEC-Q/L/QnA Programming Manual (SFC).	Initial start	-
Start conditions	8003H			Autostart block 0	
Output mode when the block is stopped	8006H			Turn OFF	

(8) I/O assignment

The mounting status of each module in the system is set.



I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				

(*)Settings should be set as same when using multiple CPU.

Figure 8.8 I/O assignment

Table8.8 I/O assignment setting list

Item		Parameter No.	Description	Setting range	Default	Reference
I/O assignment	Type	0400H	Set the type of the mounted module.	• CPU No.2 and No.3: No.n/Empty (Set "CPU (Empty)" for the slot where no CPU module is mounted.) • Empty, Input, Hi input, Output, Intelli., I/O mix, or Interrupt	Blank	Section 4.2
	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.	[Q00JCPU] 0, 16, 32, 48, 64, 128, or 256 points [Q00CPU and Q01CPU] 0, 16, 32, 48, 64, 128, 256, 512, or 1024 points		
	Start XY		Set the start I/O number of each slot.	[Q00JCPU] 0H to F0H [Q00CPU and Q01CPU] 0H to 3F0H		

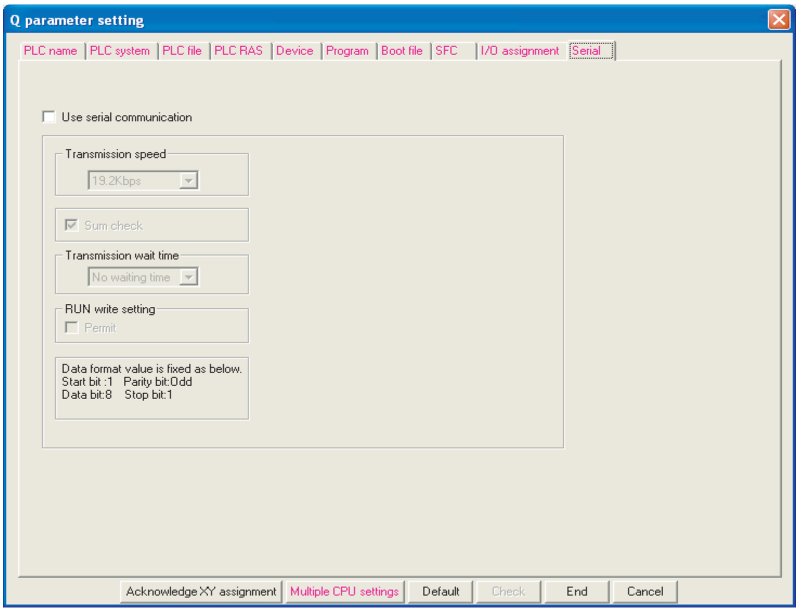
(To the next page)

Table8.8 I/O assignment setting list (continued)

Item		Parameter No.	Description	Setting range	Default	Reference
Base setting	Base model name	0401H	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	Section 4.1.2
	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		0407H	Set various switches of an intelligent function module.	Refer to the manual for the intelligent function module used.	Blank	Section 6.10
Detailed setting	Error time output mode	0403H	Set whether to clear or hold the output in case of a stop error in the control CPU.	Clear/Hold	Clear	Section 6.8
	H/W error time PLC operation mode	4004H	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	Section 6.9
	I/O response time	0405H	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	Section 6.7
	Control PLC	0406H	Set the control CPU for the input/ output modules and intelligent function module.	No.1, No.2, No.3, or No.4	No.1	QCPU User's Manual (Multiple CPU System)

(9) Serial Note8.1

The transmission speed, sum check, transmission wait time, and RUN write setting for using the serial communication function of the CPU module are set.



□ Use serial communication

Transmission speed: 19.2Kbps

☑ Sum check

Transmission wait time: No waiting time

RUN write setting: □ Permit

Data format value is fixed as below.
Start bit :1 Parity bit:Odd
Data bit:8 Stop bit:1

Figure 8.9 Serial

Table8.9 Serial setting list

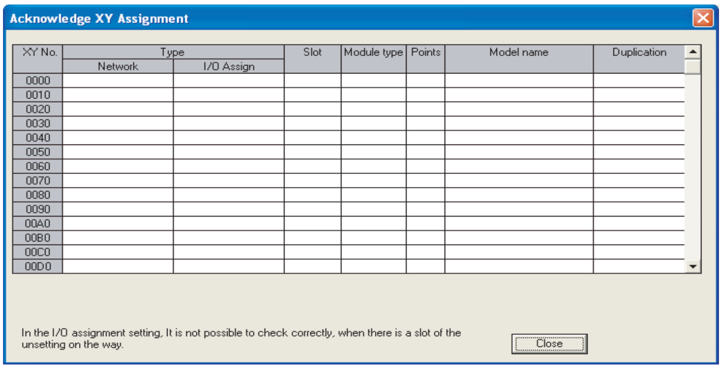
Item	Parameter No.	Description	Setting range	Default	Reference
Use serial communication	100E$_H$	Select the item when using the serial communication function.	Selected/deselected	Deselected	Section 6.24
Transmission speed		Set a transmission speed for data communication with the external device.	9.6kbps, 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 Basic model QCPU side in case the CPU module cannot receive data immediately after the external device sends data.	No waiting time/10ms to 150ms (in increments of 10ms)	No waiting time	
RUN write setting		Enable or disable writing of data from the external device to the running CPU module.	Selected/deselected	Deselected	

Note8.1 **Basic**

The Q00JCPU does not support the serial communication function.

(10)Acknowledge XY assignment

The parameters set in the I/O assignment, Ethernet/CC IE/MELSECNET setting, and CC-Link setting can be confirmed.



Acknowledge XY Assignment

XY No.	Type		Slot	Module type	Points	Model name	Duplication
	Network	I/O Assign					
0000							
0010							
0020							
0030							
0040							
0050							
0060							
0070							
0080							
0090							
00A0							
00B0							
00C0							
00D0							

In the I/O assignment setting, It is not possible to check correctly, when there is a slot of the unsetting on the way.

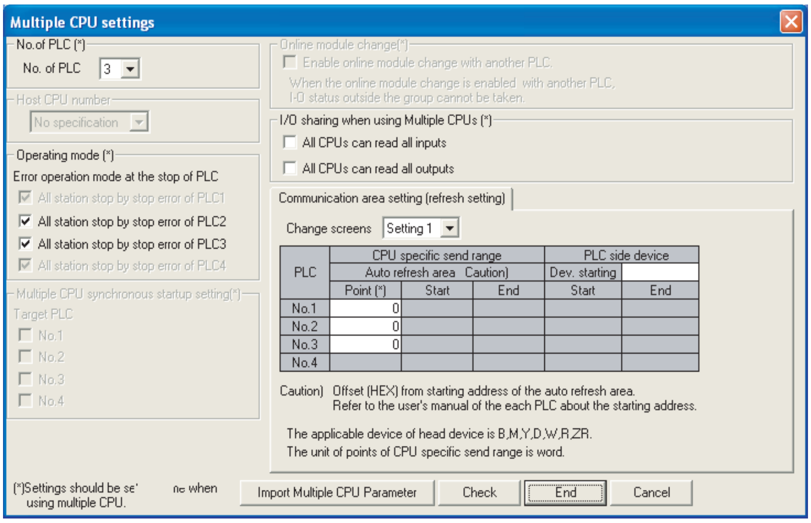
Figure 8.10 Acknowledge XY assignment

Table8.10 Acknowledge X/Y assignment list

Item	Parameter No.	Description	Setting range	Default	Reference
X/Y assignment	-	The data set in the I/O assignment, Ethernet/CC IE/ MELSECNET setting, and CC-Link setting can be checked.	-	-	-

(11)Multiple CPU settings

Parameters required for configuring a multiple CPU system are set.



Multiple CPU settings

No.of PLC (*)
No. of PLC: 3

Host CPU number
No specification

Operating mode (*)
Error operation mode at the stop of PLC
☑ All station stop by stop error of PLC1
☑ All station stop by stop error of PLC2
☑ All station stop by stop error of PLC3
☑ All station stop by stop error of PLC4

Multiple CPU synchronous startup setting(*)
Target PLC
□ No.1
□ No.2
□ No.3
□ No.4

Online module change(*)
□ Enable online module change with another PLC.
When the online module change is enabled with another PLC, I-O status outside the group cannot be taken.

I/O sharing when using Multiple CPUs (*)
□ All CPUs can read all inputs
□ All CPUs can read all outputs

Communication area setting (refresh setting)

Change screens: Setting 1

PLC	CPU specific send range			PLC side device	
	Auto refresh area Caution)			Dev. starting	
	Point (*)	Start	End	Start	End
No.1	0				
No.2	0				
No.3	0				
No.4					

Caution) Offset (HEX) from starting address of the auto refresh area.
Refer to the user's manual of the each PLC about the starting address.

The applicable device of head device is B,M,Y,D,W,R,ZR.
The unit of points of CPU specific send range is word.

(*)Settings should be se' ne when using multiple CPU.

Figure 8.11 Multiple CPU settings

Table8.11 Multiple CPU setting list

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 3	1	QCPU User's Manual (Multiple CPU system)
Operating mode		0E01$_{H}$	Select the multiple CPU system operation to be performed in case a stop error occurs in a CPU No.2 or No.3. When CPU No.1 results in a stop error, the multiple CPU system stops. (Fixed)	Selected/deselected	All items selected	
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	
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.	[CPU specific send range] CPU No.1: 0 to 320 points (in increments of 2 points), CPU No.2 and 3: 0 to 2048 points (in increments of 2 points) per CPU Up to 4416 points per system [PLC side device] B, M, Y, D, 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.	Blank	

8.1.2 High Performance model QCPU, Process CPU, and Redundant CPU

(1) PLC name

A label and a comment for the CPU module are set.

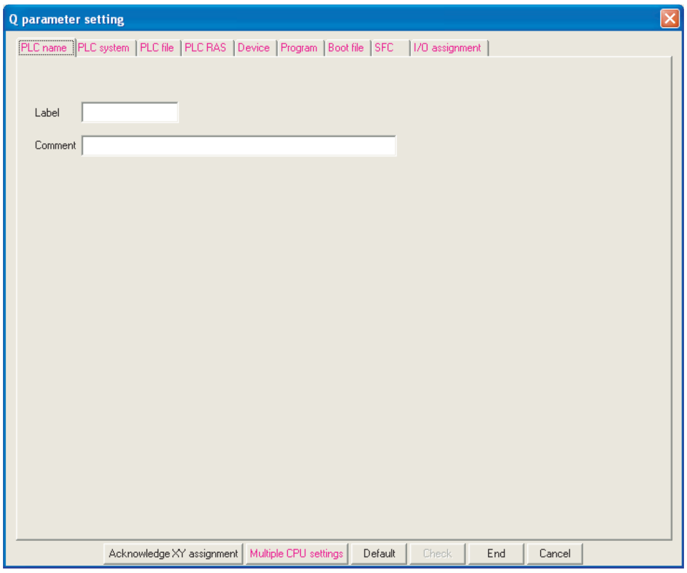


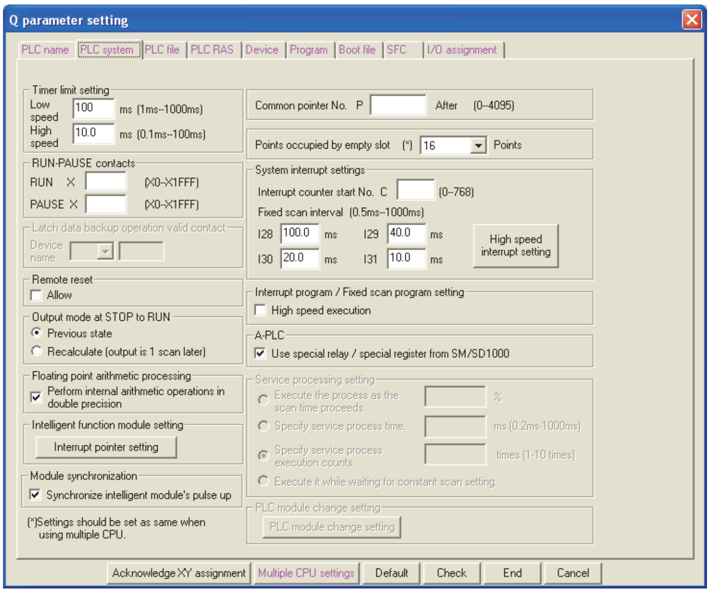
Figure 8.12 PLC name

Table8.12 PLC name setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Label	0000H	Set a label (name, application) for the CPU module.	Up to 10 characters	Blank	-
Comment	0001H	Set a comment for the CPU module label.	Up to 64 characters	Blank	-

(2) PLC system

Parameters required for use of the CPU module are set.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
□ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms

Interrupt program / Fixed scan program setting
□ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting

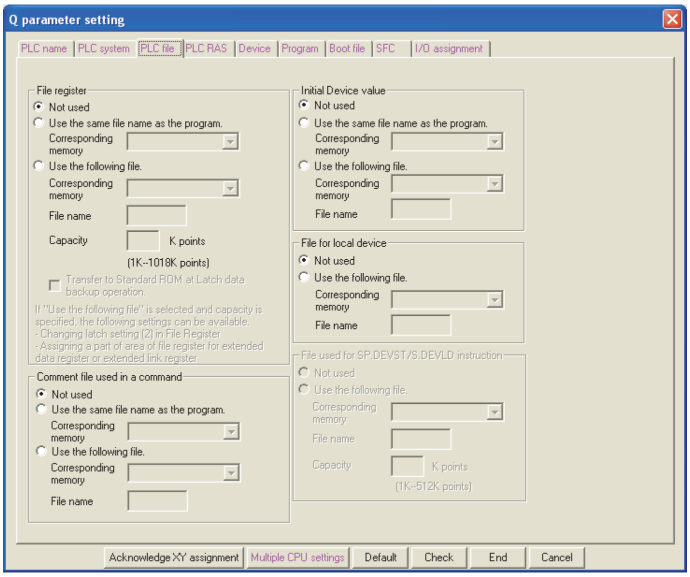
Figure 8.13 PLC system

Table8.13 PLC system setting list

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	Section 9.2.10
	High speed			0.01ms to 100.0ms (in increments of 0.01ms)	10.0ms	Section 9.2.10
RUN-PAUSE contcts	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 X1FFF	Blank	Section 6.6.1
	PAUSE					Section 6.6.2
Remote reset		1002_H	Select whether to allow the remote reset from GX Developer.	Selected/deselected	Deselected	Section 6.6.3
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	Section 6.4
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 I255	Blank	Section 6.23
Module synchronization		$100C_H$	Select whether to synchronize CPU module startup with intelligent function module startup.	Selected/deselected	Deselected	-
Common pointer No.		1005_H	Set the start number of common pointers.	P0 to P4095	Blank	Section 9.9.2
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	Section 4.2.2
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	Section 9.10
Interrupt program/ Fixed scan program setting		1008_H	Enable or disable high speed execution of interrupt programs or fixed scan execution type programs.	Selected/deselected	Deselected	Section 2.2.3, Section 2.3.5

(3) PLC file

Parameters required for the files used in the CPU module are set.



Q parameter setting

File register
- ● Not used
- ○ Use the same file name as the program.
  - Corresponding memory
- ○ Use the following file.
  - Corresponding memory
  - File name
  - Capacity K points (1K--1018K points)
  - □ Transfer to Standard ROM at Latch data backup operation.

If "Use the following file" is selected and capacity is specified, the following settings can be available.
- Changing latch setting (2) in File Register
- Assigning a part of area of file register for extended data register or extended link register

Comment file used in a command
- ● Not used
- ○ Use the same file name as the program.
  - Corresponding memory
- ○ Use the following file.
  - Corresponding memory
  - File name

Initial Device value
- ● Not used
- ○ Use the same file name as the program.
  - Corresponding memory
- ○ Use the following file.
  - Corresponding memory
  - File name

File for local device
- ● Not used
- ○ Use the following file.
  - Corresponding memory
  - File name

File used for SP.DEVST/S.DEVLD instruction
- ○ Not used
- ○ Use the following file.
  - Corresponding memory
  - File name
  - Capacity K points (1K--512K points)

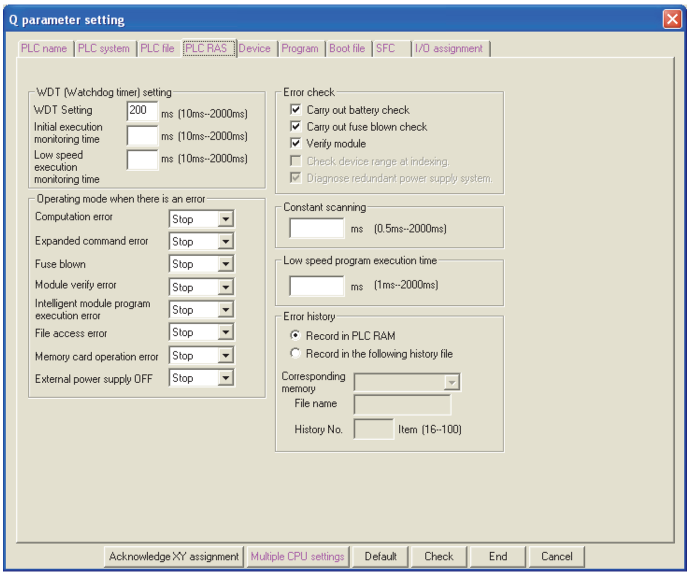
Figure 8.14 PLC file

Table8.14 PLC file setting list

Item	Parameter No.	Description	Setting range	Default	Reference
File register	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	Section 9.7
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	-
Initial Device 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	Section 6.26
File for local device	1103_{H}	Set a file for local devices used in the program.	• Not used • Use the following file.	Not used	Section 9.13.2

(4) PLC RAS (PLC RAS(1)[*1])

Parameters required for performing the RAS functions are set.



WDT (Watchdog timer) setting

WDT Setting 200 ms (10ms--2000ms)

Initial execution monitoring time ms (10ms--2000ms)

Low speed execution monitoring time ms (10ms--2000ms)

Operating mode when there is an error

Computation error Stop

Expanded command error Stop

Fuse blown Stop

Module verify error Stop

Intelligent module program execution error Stop

File access error Stop

Memory card operation error Stop

External power supply OFF Stop

Error check

☑ Carry out battery check

☑ Carry out fuse blown check

☑ Verify module

□ Check device range at indexing.

☑ Diagnose redundant power supply system.

Constant scanning

ms (0.5ms--2000ms)

Low speed program execution time

ms (1ms--2000ms)

Error history

● Record in PLC RAM

○ Record in the following history file

Corresponding memory

File name

History No. Item (16--100)

Figure 8.15 PLC RAS

Table8.15 PLC RAS setting list

Item		Parameter No.	Description	Setting range	Default	Reference
WDT (Watchdog timer) setting	WDT Setting	3000H	Set a watchdog timer value for the CPU module.	10ms to 2000ms (in increments of 10ms)	200ms	Section 6.16
	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	Section 2.3.1
Operating mode when there is an error	Computation error	3002H	Set the operation mode of the CPU module when an error is detected.	Stop/Continue	Stop	Section 6.17
	Expanded command error[*2]					
	Fuse blown					
	Module verify error					
	Intelligent module program execution error					
	File access error					
	Memory card operation error					
	External power supply OFF[*2]					
Error check	Carry out battery check	3001H	Enable or disable detection of the specified error.	Selected/deselected	Deselected	Section 6.17
	Carry out fuse blown check					
	Verify module					
Constant scanning		3003H	Set a constant scan time value.	0.5ms to 2000ms (in increments of 0.5ms)	Blank	Section 6.2
Low speed program execution time[*3]		3006H	Set execution time of the low-speed execution type program in each scan.	1ms to 2000ms	Blank	Section 2.3.3
Error history		3005H	Set the storage location of the error history data in the CPU module.	Record in PLC RAM/ Record in the following history file	Record in PLC RAM	Section 6.18

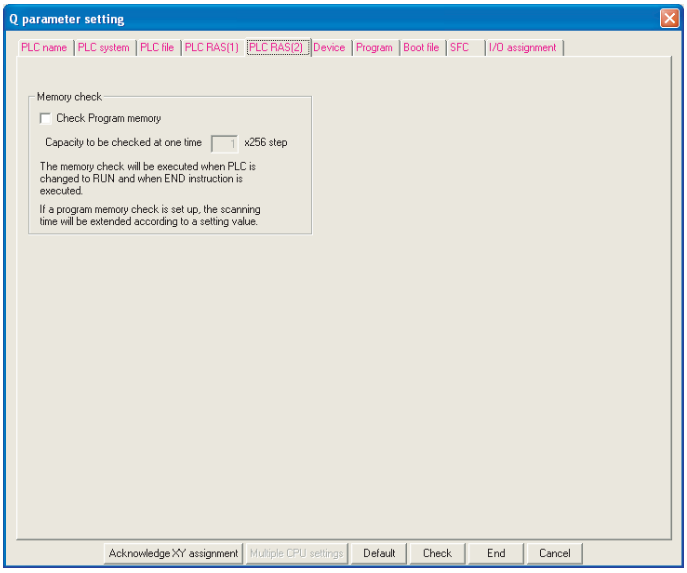
[*1] For the Process CPU and Redundant CPU, the tab name is "PLC RAS(1)".

[*2] These items are provided for future expansion.

[*3] Not available for the Redundant CPU.

(5) PLC RAS(2) Note8.2

Parameters required for performing the RAS functions in the Process CPU and Redundant CPU are set.



Q parameter setting

Memory check

□ Check Program memory

Capacity to be checked at one time 1 x256 step

The memory check will be executed when PLC is changed to RUN and when END instruction is executed.

If a program memory check is set up, the scanning time will be extended according to a setting value.

Figure 8.16 PLC RAS(2)

Table8.16 PLC RAS(2) setting list

Item		Parameter No.	Description	Setting range	Default	Reference
Memory check	Check Program memory	3008H	Set whether to check the user area in the program memory (excluding system areas) for data corruption.	Selected/deselected	Deselected	Section 6.27
			Enter the number of steps to be checked when "Check Program memory" is selected.	Q12PHCPU and Q12PRHCPU: 1 to 496 points (256 to 124K steps) Q25PHCPU and Q25PRHCPU: 1 to 1008 points (256 to 252K steps)	1 × 256 step	

Point

When checking all areas in the program memory, the number of steps to be checked can be calculated by the following procedure.

- On the Read from PLC screen, check the size of total free area in the program memory.
  Before checking the size, confirm that the program memory is formatted and no file is written in the memory.
  (Section 5.2.2)
- Calculate the number of steps with the following expression.

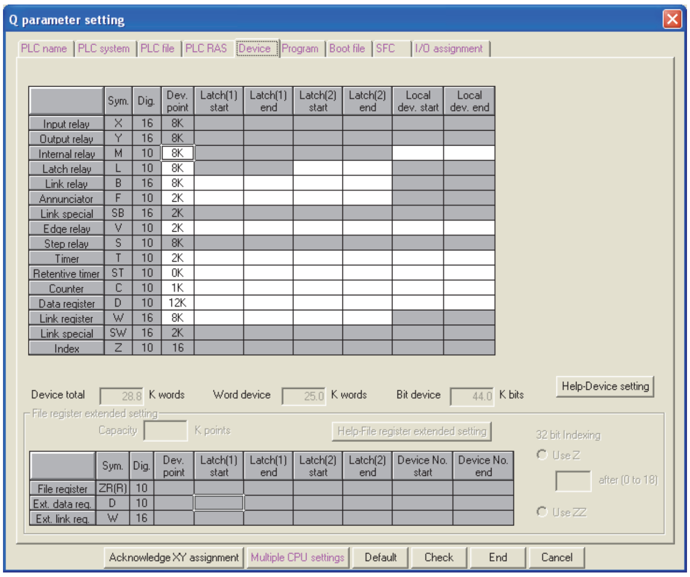
$$\text{Number of steps to be checked} = \frac{\text{Total free area size (bytes)}}{4}$$

Note8.2 High performance Process

The High Performance model QCPU does not support the memory check function.
When using the memory check function in the Process CPU, check the versions of the CPU module and GX Developer. (Appendix 2.3)

(6) Device

Number of points, latch range, and local device range are set for each device.



	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Local dev. start	Local dev. end
Input relay	X	16	8K						
Output relay	Y	16	8K						
Internal relay	M	10	8K						
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	12K						
Link register	W	16	8K						
Link special	SW	16	2K						
Index	Z	10	16						

Device total 28.8 K words   Word device 25.0 K words   Bit device 44.0 K bits

File register extended setting

Capacity K points

	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Device No. start	Device No. end
File register	ZR(R)	10							
Ext. data req.	D	10							
Ext. link req.	W	16							

32 bit Indexing

○ Use Z

after (0 to 18)

○ Use ZZ

Figure 8.17 Device

Table8.17 Device setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Device points[*1]	2000H	Set the number of device points that is appropriate to the system.	X(8K), Y(8K), S(8K), SB(2K), and SW(2K) are fixed. Setting is available within the range of 29K words in total, including the above fixed points (3.7K words). • One device: Up to 32K points • Total bit device points: Up to 64K points	• X: 8K • Y: 8K • M: 8K • L: 8K • B: 8K • F: 2K • SB: 2K • V: 2K • S: 8K • T: 2K • ST: 0K • C: 1K • D: 12K • W: 8K • SW: 2K	Section 9.1
Latch (1) start/end (Latch clear valid)	2001H	Set a latch range (start and end device numbers), which can be cleared by the RESET/L.CLR switch or remote latch clear operation.	Setting is available for only one range for each of B, F, V, T, ST, C, D, and W devices.	Blank	Section 3.4, Section 6.3
Latch (2) start/end (Latch clear invalid)	2002H	Set a latch range (start and end device numbers), which cannot be cleared by the RESET/L.CLR switch or remote 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	Section 3.4, Section 6.3
Local device start/end	2003H	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 and D devices.	Blank	Section 9.13.2

*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.

(7) Program

File names and execution types (execution conditions) are set for each program when two or more programs are written to the CPU module.

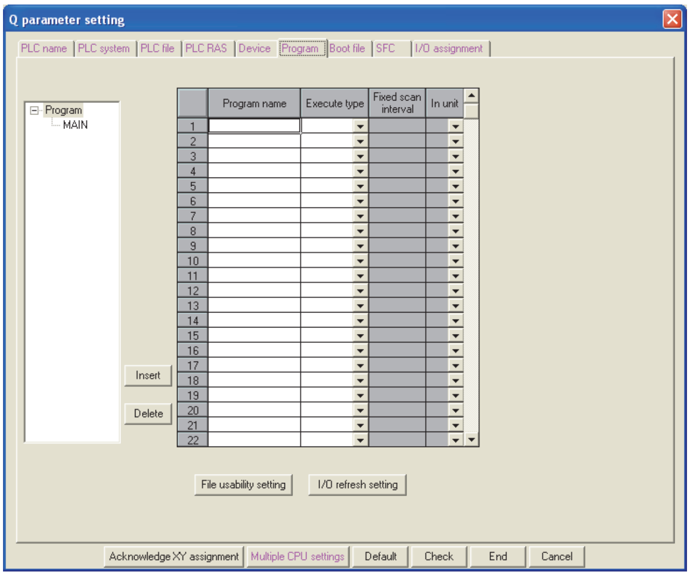


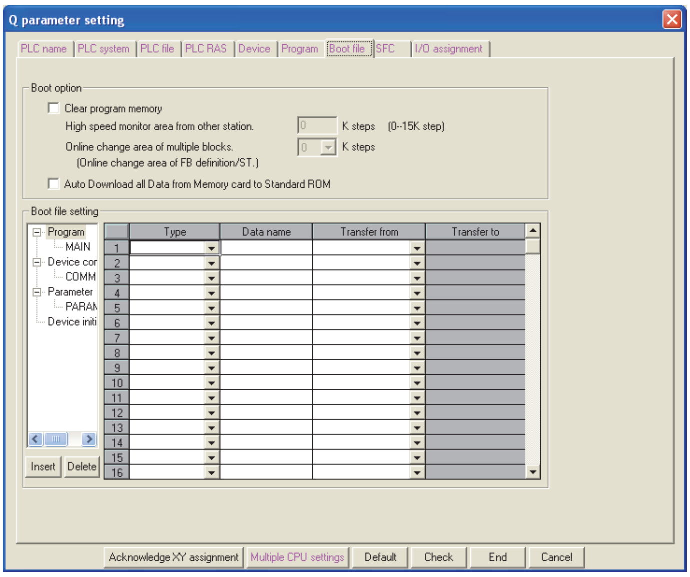
Figure 8.18 Program

Table8.18 Program setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Program setting	7000H	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 I/O refresh setting	Blank	Section 2.3

(8) Boot file

Parameters required for boot operation and writing data automatically to the standard ROM are set.



Q parameter setting

Boot option

☐ Clear program memory

High speed monitor area from other station. 0 K steps (0--15K step)

Online change area of multiple blocks. 0 K steps
(Online change area of FB definition/ST.)

☐ Auto Download all Data from Memory card to Standard ROM

Boot file setting

	Type	Data name	Transfer from	Transfer to
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				

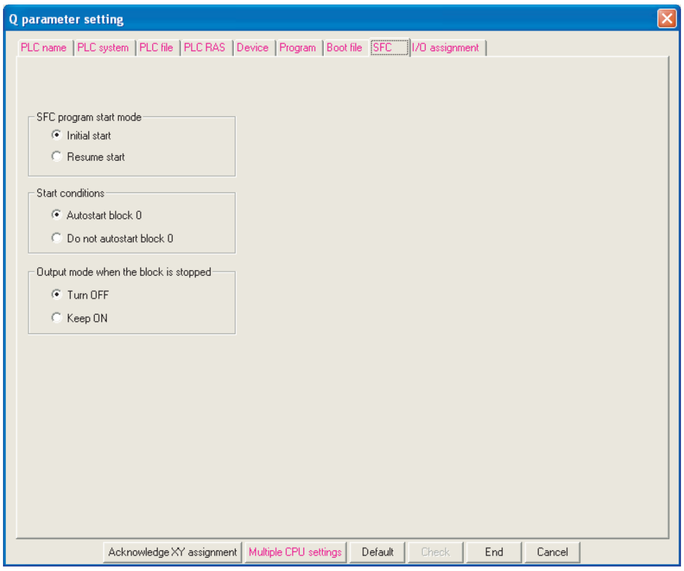
Figure 8.19 Boot file

Table8.19 Boot file setting list

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	Section 5.2.7, Section 5.2.8
	Auto Download all Data from Memory card to Standard ROM		Select whether to write data of the memory card automatically to the standard ROM when booting.	Selected/deselected	Deselected	
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	

(9) SFC

The mode and conditions for starting an SFC program, and the output mode in the case of a block stop are set.


Q parameter setting
PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment
Acknowledge XY assignment
Multiple CPU settings
Default
Check
End
Cancel

SFC program start mode
● Initial start
○ Resume start

Start conditions
● Autostart block 0
○ Do not autostart block 0

Output mode when the block is stopped
● Turn OFF
○ Keep ON

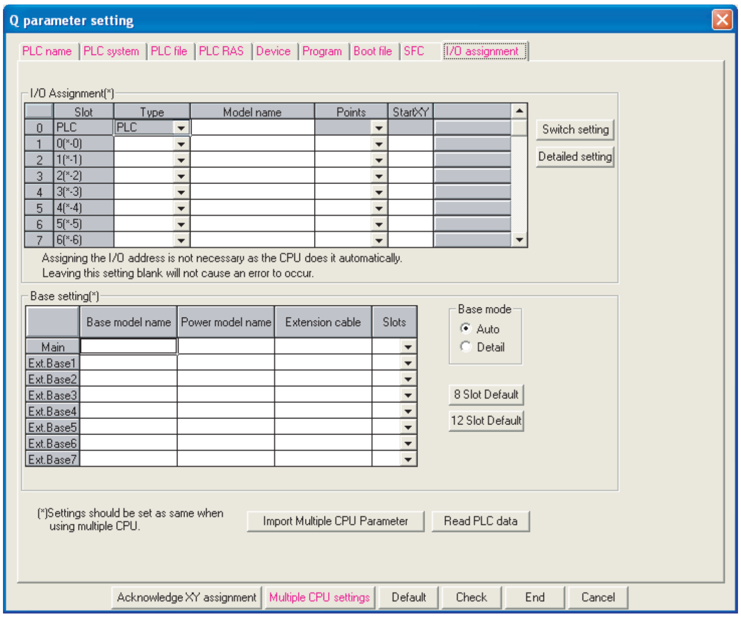
Figure 8.20 SFC

Table8.20 SFC setting list

Item	Parameter No.	Description	Setting range	Default	Reference
SFC program start mode	8002H	Set the mode and conditions for starring an SFC program, and also set the output mode in case a program block is stopped.	Refer to the MELSEC-Q/L/QnA Programming Manual (SFC).	Initial start	-
Start conditions	8003H			Autostart block 0	
Output mode when the block is stopped	8006H			Turn OFF	

(10)I/O assignment

The mounting status of each module in the system is set.



I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Assigning the I/O address is not necessary as the CPU does it automatically.
Leaving this setting blank will not cause an error to occur.

Base setting(*)

	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode: ● Auto ○ Detail

(*)Settings should be set as same when using multiple CPU.

Figure 8.21 I/O assignment

Table8.21 I/O assignment setting list

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, Intelli., I/O mix, or Interrupt	Blank	Section 4.2.2
	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	Section 4.1.2
	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		

(To the next page)

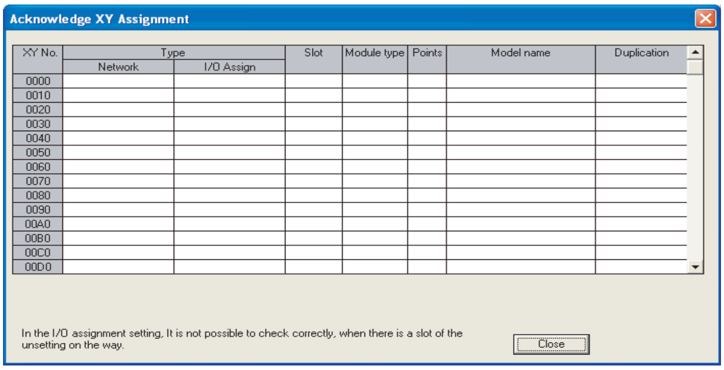
Table8.9 I/O assignment setting list (continued)

Item		Parameter No.	Description	Setting range	Default	Reference
Switch setting		0407H	Set various switches of an intelligent function module.	Refer to the manual for the intelligent function module used.	Blank	Section 6.10
Detailed setting	Error time output mode	0403H	Set whether to clear or hold the output in case of a stop error in the control CPU.	Clear/Hold	Clear	Section 6.8
	H/W error time PLC operation mode	4004H	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	Section 6.9
	I/O response time	0405H	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	Section 6.7
	Control PLC[*1]	0406H	Set the control CPU for the input/output modules and intelligent function module.	No.1, No.2, No.3, or No.4	No.1	QCPU User's Manual (Multiple CPU System)

*1: Not available for the Redundant CPU.

(11)Acknowledge XY assignment

The parameters set in the I/O assignment, Ethernet/CC IE/MELSECNET setting, and CC-Link setting can be confirmed.



Acknowledge XY Assignment

XY No.	Type		Slot	Module type	Points	Model name	Duplication
	Network	I/O Assign					
0000							
0010							
0020							
0030							
0040							
0050							
0060							
0070							
0080							
0090							
00A0							
00B0							
00C0							
00D0							

In the I/O assignment setting, It is not possible to check correctly, when there is a slot of the unsetting on the way.

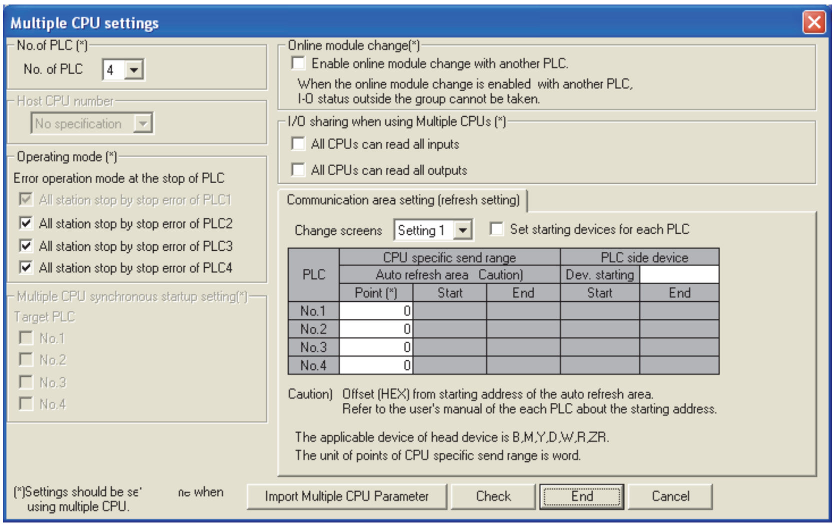
Figure 8.22 Acknowledge XY assignment

Table8.22 Acknowledge X/Y assignment list

Item	Parameter No.	Description	Setting range	Default	Reference
X/Y assignment	-	The data set in the I/O assignment, Ethernet/CC IE/MELSECNET setting, and CC-Link setting can be checked.	-	-	-

(12)Multiple CPU settings Note8.3

Parameters required for configuring a multiple CPU system are set.



Multiple CPU settings

No.of PLC (*)
No. of PLC 4

Host CPU number
No specification

Operating mode (*)
Error operation mode at the stop of PLC
☑ All station stop by stop error of PLC1
☑ All station stop by stop error of PLC2
☑ All station stop by stop error of PLC3
☑ All station stop by stop error of PLC4

Multiple CPU synchronous startup setting(*)
Target PLC
□ No.1
□ No.2
□ No.3
□ No.4

Online module change(*)
□ Enable online module change with another PLC.
When the online module change is enabled with another PLC,
I-O status outside the group cannot be taken.

I/O sharing when using Multiple CPUs (*)
□ All CPUs can read all inputs
□ All CPUs can read all outputs

Communication area setting (refresh setting)

Change screens Setting 1 □ Set starting devices for each PLC

PLC	CPU specific send range			PLC side device	
	Auto refresh area Caution)			Dev. starting	
	Point (*)	Start	End	Start	End
No.1	0				
No.2	0				
No.3	0				
No.4	0				

Caution) Offset (HEX) from starting address of the auto refresh area.
Refer to the user's manual of the each PLC about the starting address.

The applicable device of head device is B,M,Y,D,W,R,ZR.
The unit of points of CPU specific send range is word.

(*)Settings should be se' ne when using multiple CPU.

Figure 8.23 Multiple CPU settings

Table8.23 Multiple CPU setting list

Item	Parameter No.	Description	Setting range	Default	Reference
No. of PLC	0E00H	Set the number of CPU modules used in a multiple CPU system.	1 to 4	1	QCPU User's Manual (Multiple CPU System)
Host CPU number	E00CH	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	
Operating mode	0E01H	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	E00BH	Enable or disable synchronous startup of the CPU modules on the multiple CPU system.	Mp.1 to No.4	All items selected	
Online module change	E006H	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	• High Performance model QCPU: Deselected • Process CPU: Selected	

(To the next page)

Note8.3 Redundant

The Redundant CPU cannot be used in multiple CPU systems.

Table8.23 Multiple CPU setting list (continued)

Item	Parameter No.	Descrip-tion	Setting range	Default	Reference	Parameter No.
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	QCPU User's Manual (Multiple CPU System)
	All CPUs can read all outputs		Select whether to read the output data of the output modules controlled by another CPU.	Selected/deselected	Deselected	
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 CPU] 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.		

8.2 Redundant Parameters Note8.4

This section provides the list of redundant parameters and describes parameter details.

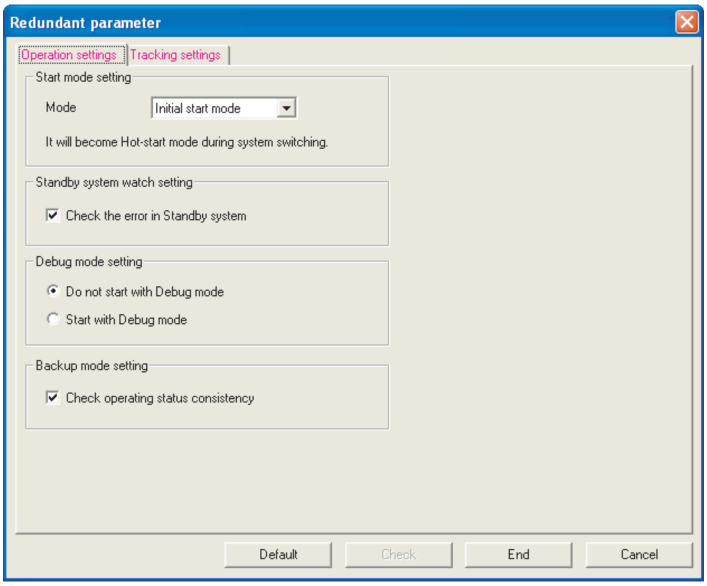
Table8.24 Redundant parameter

Item	Parameter No.	Description	Setting range	Default	Reference
Redundant parameter	0D00$_H$	Set the operation mode and tracking transfer settings of the Redundant CPU.	-	-	QnPRHCPU User's Manual (Redundant System)

☞ QnPRHCPU User's Manual (Redundant System)

(1) Operation settings

The operation mode of the Redundant CPU at power-on is set.



Start mode setting

Mode: Initial start mode

It will become Hot-start mode during system switching.

Standby system watch setting

☑ Check the error in Standby system

Debug mode setting

● Do not start with Debug mode

○ Start with Debug mode

Backup mode setting

☑ Check operating status consistency

Figure 8.24 Operation settings

Table8.25 Operation setting list

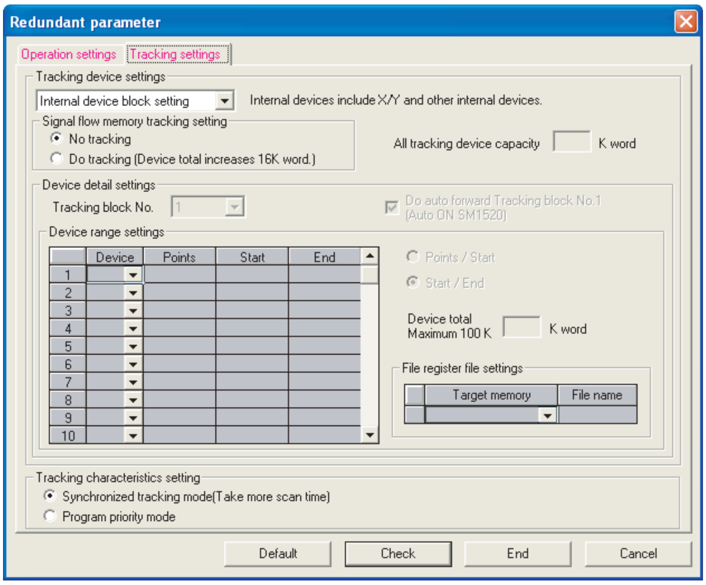
Item	Parameter No.	Description	Setting range	Default	Reference
Start mode setting	D001$_H$	Select the start mode when the Redundant CPU is powered off and then on.	Initial start mode/ Hot-start mode	Initial start mode	QnPRHCPU User's Manual (Redundant System)
Standby system watch setting		Set whether to monitor errors in the standby system.	Selected/deselected	Selected	
Debug mode setting		Select whether to start the Redundant CPU in debug mode.	Do not start with Debug mode/ Start with Debug mode	Do not start with Debug mode	
Backup mode setting		Select whether to check the operating status consistency between the control and standby systems while the Redundant CPU is operating in backup mode.	Selected/deselected	Selected	

Note8.4 Basic High performance Process

Redundant parameters are not available for the Basic model QCPU, High Performance model QCPU, and Process CPU.

(2) Tracking settings

Parameters required for the tracking function in a redundant system are set.



Redundant parameter

Tracking device settings: Internal device block setting

Internal devices include X/Y and other internal devices.

Signal flow memory tracking setting

● No tracking

○ Do tracking (Device total increases 16K word.)

All tracking device capacity □ K word

Device detail settings

Tracking block No. 1

☑ Do auto forward Tracking block No.1 (Auto ON SM1520)

Device range settings

	Device	Points	Start	End
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

○ Points / Start

● Start / End

Device total Maximum 100 K □ K word

File register file settings

	Target memory	File name

Tracking characteristics setting

● Synchronized tracking mode(Take more scan time)

○ Program priority mode

Figure 8.25 Tracking settings

Table8.26 Tracking setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Tracking device settings	D003$_H$	Select whether to set the range for tracking device data to be transferred.	Internal device block setting/ Device detail settings	Internal device block setting	QnPRHCPU User's Manual (Redundant System)
Signal flow memory tracking setting		Select whether to track the signal flow memory.	No tracking/Do tracking	No tracking	
Device detail settings		Set details when the tracking device to be transferred is set by user.	-	-	
Tracking block No.		Select the block number of the tracking device to be transferred. Then, set details in the Device range settings.	1 to 64	1	
Do auto forward Tracking block No.1 (Auto ON SM1520)		Select whether to automatically transfer the device data set to the tracking block No. 1.	Selected/deselected	Deselected	
Device range settings		Set the device to be tracked and its range.	Refer to the QnPRHCPU User's Manual	-	
File register file settings		When the file register is set in the Device range settings, select the memory where the target file register is stored and enter the file name.		-	
Tracking characteristics setting	D002$_H$	Select the operation mode of tracking.	Synchronized tracking mode/ Program priority mode	Synchronized tracking mode	

8.3 Network Parameters

This section provides the list of network parameters and describes parameter details.

■ Symbols, mn, **, M, and N used in the "Parameter No."

mn, **, M, and N in "Parameter No." in this section denote the following:

- mn: Indicates the value that is calculated by "start I/O No. divided by 16".
- **: Indicates any given value.
- N: Indicates the module number.
- M: Indicates the network type.

Table8.27 For CC-Link IE Controller Network and MELSECNET/H (☞ (1), (2) in this section)

M	Network type
1H	CC IE Control (Control station)*1, MELSECNET/H mode (Control station), MELSECNET/H Extended mode (Control station), MELSECNET/10 mode (Control station)
2H	CC IE Control (Normal station), MELSECNET/H mode (Normal station), MELSECNET/H Extended mode (Normal station), MELSECNET/10 mode (Normal station)
5H	MELSECNET/H (Remote master)
AH	MELSECNET/H (Standby station)
BH	MELSECNET/H mode multiplexed remote I/O network master station
DH	MELSECNET/H mode multiplexed remote I/O network sub-master station (when no parameter is set)
EH	MELSECNET/H mode multiplexed remote I/O network sub-master station (when parameters are set)

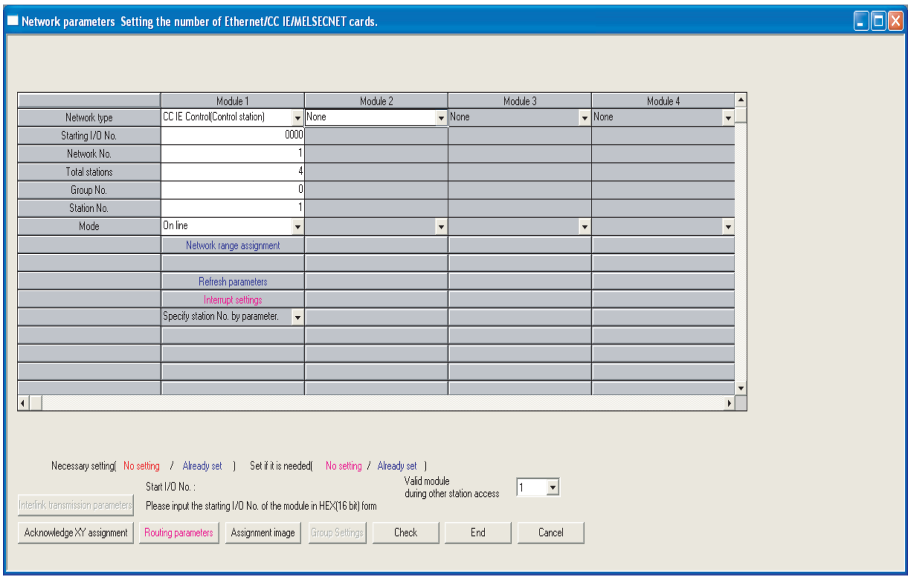
*1: For the Basic model QCPU, the "CC IE Control (Control station)" cannot be set.

Table8.28 For CC-Link setting (☞ (4) in this section)

M	Network type
0H	Master station
1H	Local station
2H	Standby master station

(1) CC-Link IE Controller Network setting

Network parameters for the CC-Link IE Controller Network are set.



Network parameters Setting the number of Ethernet/CC IE/MELSECNET cards.

	Module 1	Module 2	Module 3	Module 4
Network type	CC IE Control(Control station)	None	None	None
Starting I/O No.	0000			
Network No.	1			
Total stations	4			
Group No.	0			
Station No.	1			
Mode	On line			
	Network range assignment			
	Refresh parameters			
	Interrupt settings			
	Specify station No. by parameter.			

Necessary setting(No setting / Already set) Set if it is needed(No setting / Already set)

Start I/O No. :

Please input the starting I/O No. of the module in HEX(16 bit) form

Valid module during other station access: 1

Figure 8.26 Setting the number of Ethernet/CC IE/MELSECNET cards (CC-Link IE Controller Network setting)

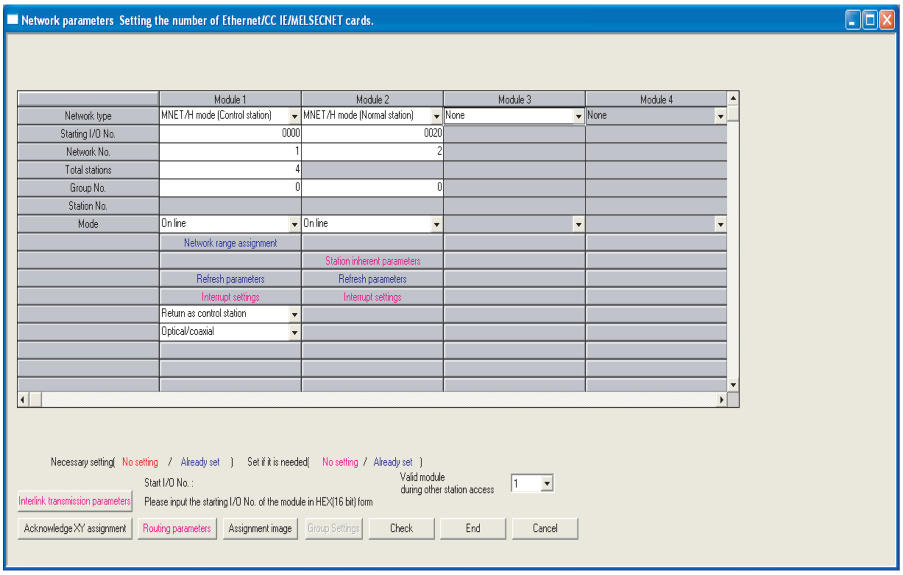
Table8.29 CC-Link IE Controller Network setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Number of modules on CC-Link IE Controller Network	A000$_H$	Set network parameters for the CC-Link IE Controller Network.	Refer to the manual for the CC-Link IE Controller Network.	-	-
Starting 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$_H$				
Group settings[*1]	D004$_H$				
Redundant settings[*1]	DA**$_H$				
Interlink transmission parameters	A002$_H$				
Routing parameters	5003$_H$				

[*1] Available only for the Redundant CPU.

(2) MELSECNET/H setting

Network parameters for MELSECNET/H are set.



Network parameters Setting the number of Ethernet/CC IE/MELSECNET cards.

	Module 1	Module 2	Module 3	Module 4
Network type	MNET/H mode (Control station)	MNET/H mode (Normal station)	None	None
Starting I/O No.	0000	0020		
Network No.	1	2		
Total stations	4			
Group No.	0	0		
Station No.				
Mode	On line	On line		
	Network range assignment			
		Station inherent parameters		
	Refresh parameters	Refresh parameters		
	Interrupt settings	Interrupt settings		
	Return as control station			
	Optical/coaxial			

Necessary setting(No setting / Already set) Set if it is needed(No setting / Already set)

Start I/O No. :

Please input the starting I/O No. of the module in HEX(16 bit) form

Valid module during other station access: 1

Figure 8.27 Setting the number of Ethernet/CC IE/MELSECNET cards (MELSECNET/H setting)

Table8.30 MELSECNET/H setting list

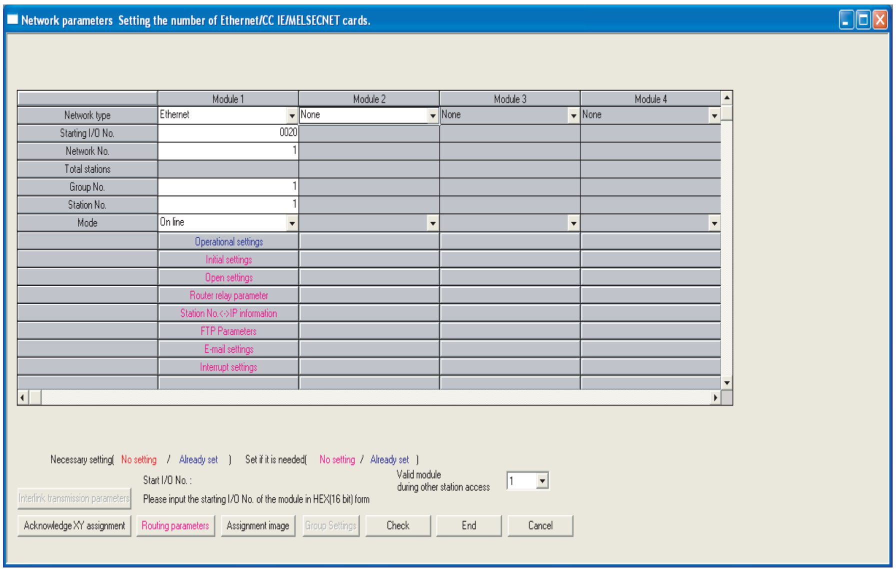
Item	Parameter No.	Description	Setting range	Default	Reference
Number of modules on MELSECNET/H	5000_H	Set MELSECNET/H network parameters.	Refer to the manual for the Q series-compatible MELSECNET/H.	-	-
Starting I/O No.	$5NM0_H$				
Network No.					
Total stations					
Group No.	$05mn_H$				
Mode	$5NM0_H$				
Refresh parameters	$5NM1_H$				
Common parameters	$5NM2_H$				
Station inherent parameters	$5NM3_H$				
Sub-master parameters*2	$5NM5_H$				
Common parameters 2	$5NMA_H$				
Station inherent parameters 2	$5NMB_H$				
Interrupt settings					
Group settings*1	$D004_H$				
Redundant settings*1	$D5{*}{*}_H$				
Valid module during other station access	5001_H				
Interlink transmission parameters	5002_H				
Routing parameters	5003_H				

*1: Available only for the Redundant CPU.

*2: Available only for the Process CPU or Redundant CPU.

(3) Ethernet setting

Network parameters for Ethernet are set.



Network parameters Setting the number of Ethernet/CC IE/MELSECNET cards.

	Module 1	Module 2	Module 3	Module 4
Network type	Ethernet	None	None	None
Starting I/O No.	0020			
Network No.	1			
Total stations				
Group No.	1			
Station No.	1			
Mode	On line			
	Operational settings			
	Initial settings			
	Open settings			
	Router relay parameter			
	Station No.<->IP information			
	FTP Parameters			
	E-mail settings			
	Interrupt settings			

Necessary setting(No setting / Already set) Set if it is needed(No setting / Already set)

Start I/O No. :

Please input the starting I/O No. of the module in HEX(16 bit) form

Valid module during other station access: 1

Figure 8.28 Setting the number of Ethernet/CC IE/MELSECNET cards (Ethernet setting)

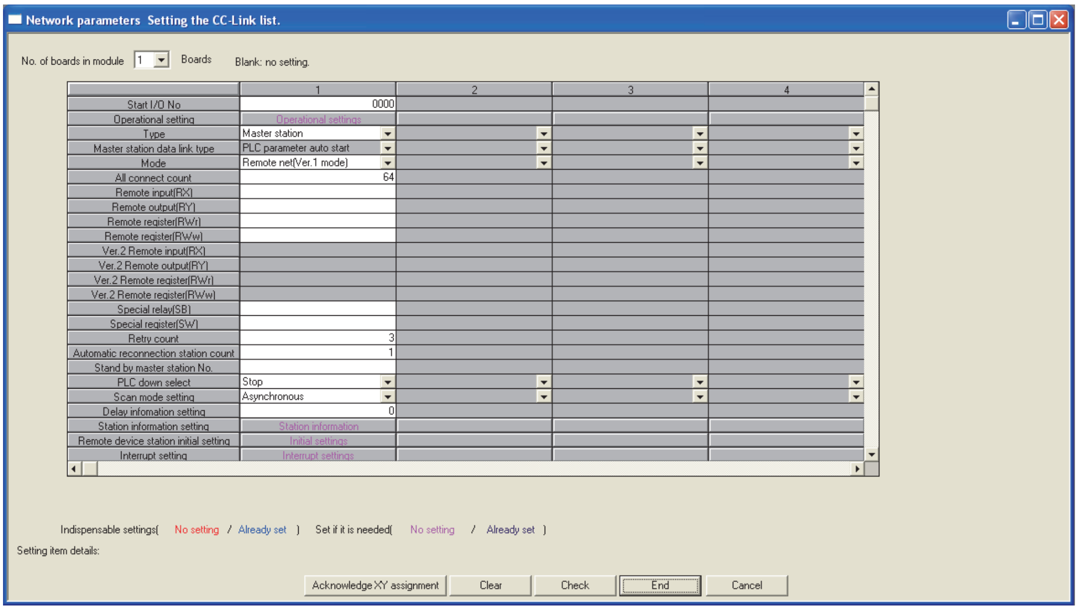
Table8.31 Ethernet setting list

Item	Parameter No.	Description	Setting range	Default	Reference
Number of modules on Ethernet	9000H	Set Ethernet network parameters.	Refer to the manual for the Q series-compatible Ethernet.	-	-
Starting I/O No.	9N00H				
Network No.					
Group No.					
Station No.					
Operational settings					
Initial settings	9N01H				
Open settings	9N02H				
Router relay parameter	9N03H				
Station No.<->IP information	9N05H				
FTP Parameters	9N06H				
E-mail settings	9N07H				
News setting	9N08H				
Interrupt settings	9N09H				
Redundant settings[*1]	D9**H				
Valid module during other station access	5001H				
Routing parameter	9N04H				
Group settings[*1]	D004H				

*1: Available only for the Redundant CPU.

(4) CC-Link setting

Parameters for CC-Link are set.



Network parameters Setting the CC-Link list.

No. of boards in module 1 Boards Blank: no setting.

	1	2	3	4
Start I/O No	0000			
Operational setting	Operational settings			
Type	Master station			
Master station data link type	PLC parameter auto start			
Mode	Remote net(Ver.1 mode)			
All connect count	64			
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)				
Retry count	3			
Automatic reconnection station count	1			
Stand by master station No.				
PLC down select	Stop			
Scan mode setting	Asynchronous			
Delay infomation setting	0			
Station information setting	Station information			
Remote device station initial setting	Initial settings			
Interrupt setting	Interrupt settings			

Indispensable settings(No setting / Already set) Set if it is needed(No setting / Already set)

Setting item details:

Figure 8.29 Setting the CC-Link list

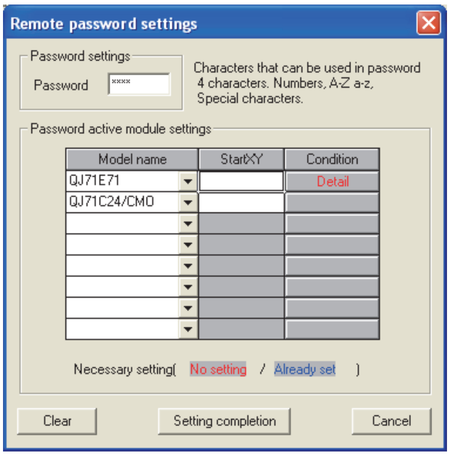
Table8.32 CC-Link setting list

Item	Parameter No.	Description	Setting range	Default	Reference
No. of boards in module	C000H	Set CC-Link parameters.	Refer to the manual for CC-Link.	-	-
Type					
Start I/O No.	CNM2H				
Operational setting					
All connect count					
Remote input (RX)	CNM1H				
Remote output (RY)					
Remote register (RWr)					
Remote register (RWw)					
Ver.2 Remote input (RX)*1					
Ver.2 Remote output (RY)*1					
Ver.2 Remote register (RWr)*1					
Ver.2 Remote register (RWw)*1					
Special relay (SB)	CNM2H				
Special register (SW)					
Retry count					
Automatic reconnection station count					
Standby master station No.					
PLC down select					
Scan mode setting					
Delay information setting					
Station information setting					
Remote device station initial setting					
Interrupt setting					

*1: Available only for the High Performance model QCPU or Process CPU.

8.4 Remote Password

This section provides the list of parameters for use of remote password and describes parameter details.



Remote password settings

Password settings

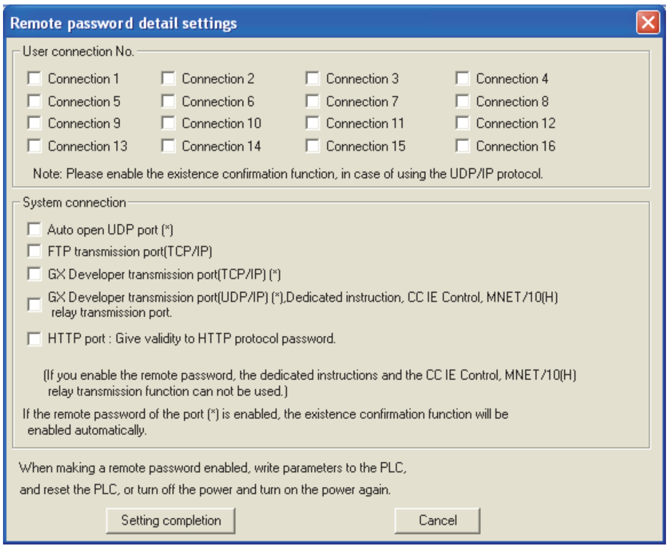
Password ****

Characters that can be used in password 4 characters. Numbers, A-Z a-z, Special characters.

Password active module settings

Model name	StartXY	Condition
QJ71E71		Detail
QJ71C24/CMO		

Necessary setting(No setting / Already set)



Remote password detail settings

User connection No.

□ Connection 1 □ Connection 2 □ Connection 3 □ Connection 4
□ Connection 5 □ Connection 6 □ Connection 7 □ Connection 8
□ Connection 9 □ Connection 10 □ Connection 11 □ Connection 12
□ Connection 13 □ Connection 14 □ Connection 15 □ Connection 16

Note: Please enable the existence confirmation function, in case of using the UDP/IP protocol.

System connection

□ Auto open UDP port (*)
□ FTP transmission port(TCP/IP)
□ GX Developer transmission port(TCP/IP) (*)
□ GX Developer transmission port(UDP/IP) (*),Dedicated instruction, CC IE Control, MNET/10(H) relay transmission port.
□ HTTP port : Give validity to HTTP protocol password.

(If you enable the remote password, the dedicated instructions and the CC IE Control, MNET/10(H) relay transmission function can not be used.)

If the remote password of the port (*) is enabled, the existence confirmation function will be enabled automatically.

When making a remote password enabled, write parameters to the PLC, and reset the PLC, or turn off the power and turn on the power again.

Figure 8.30 Remote password settings dialog box

A remote password is set for an Ethernet module, serial communication module, or modem interface module.

Table8.33 Remote password setting list

Item		Parameter No.	Description	Setting range	Default	Reference
Password settings		-	Enter a remote password.	Four characters or less (alphanumeric characters, special symbols)	-	• QJ71E71: Ethernet module manual • QJ71C24: Serial communication module manual • QJ71CMO: Modem interface module manual
Password active module settings	Model name	-	Select a model name of the module for which the remote password set to the CPU module is checked.	QJ71E71 QJ71C24/CMO	-	
	Start XY	-	Set the start address of the module for which the remote password is checked.	• Basic model QCPU: 0000$_H$ to 03E0$_H$ • High Performance model QCPU, Process CPU, and Redundant CPU: 0000$_H$ to 0FE0$_H$	-	
Detail		-	Set details of the remote password for the QJ71E71.	-	-	
	User connection No.	-	Select user connection No.	Connection 1 to Connection 16	-	
	System connection	-	Select a valid port of the remote password for system connection.	Specify a valid port of the remote password. Auto open UDP port FTP transmission port (TCP/IP) GX Developer transmission port (TCP/IP) GX Developer transmission port (UDP/IP), Dedicated instruction, CC IE Control, MNET/10(H) relay transmission port. HTTP port	-	

CHAPTER9 DEVICES

This chapter describes the devices that can be used in the CPU module.

9.1 Device List

Table9.1 and Table9.2 list the names and data ranges of the devices that can be used in the CPU module.

(1) Basic model QCPU

Table9.1 Device list

Classification	Type	Device name	Default			Parameter-set range	Reference
			Points	Range			
Internal user device	Bit device	Input	2048	X0 to 7FF	Hexadecimal	Can be changed within 16.4K words [*3]	Section 9.2.1
		Output	2048	Y0 to 7FF	Hexadecimal		Section 9.2.2
		Internal relay	8192	M0 to 8191	Decimal		Section 9.2.3
		Latch relay	2048	L0 to 2047	Decimal		Section 9.2.4
		Annunciator	1024	F0 to 1023	Decimal		Section 9.2.5
		Edge relay	1024	V0 to 1023	Decimal		Section 9.2.6
		Step relay	2048	S0 to 127/block	Decimal		Section 9.2.9
		Link relay	2048	B0 to 7FF	Hexadecimal		Section 9.2.7
		Link special relay	1024	SB0 to 3FF	Hexadecimal		Section 9.2.8
	Word device	Timer[*1]	512	T0 to 511	Decimal		Section 9.2.10
		Retentive timer[*1]	0	(ST0 to 511)	Decimal		
		Counter[*1]	512	C0 to 511	Decimal		Section 9.2.11
		Data register	11136	D0 to 11135	Decimal		Section 9.2.12
		Link register	2048	W0 to 7FF	Hexadecimal		Section 9.2.13
		Link special register	1024	SW0 to 3FF	Hexadecimal		Section 9.2.14
Internal system device	Bit device	Function input	16	FX0 to F	Hexadecimal	Cannot be changed	Section 9.3.1
		Function output	16	FY0 to F	Decimal		Section 9.3.1
		Special relay	1024	SM0 to 1023	Decimal		Section 9.3.2
	Word device	Function register	5	FD0 to 4	Decimal		Section 9.3.1
		Special register	1024	SD0 to 1023	Decimal		Section 9.3.3
Link direct device	Bit device	Link input	8192	Jn\X0 to 1FFF	Hexadecimal		Section 9.4
		Link output	8192	Jn\Y0 to 1FFF	Hexadecimal		
		Link relay	16384	Jn\B0 to 3FFF	Hexadecimal		
		Link special relay	512	Jn\SB0 to 1FF	Hexadecimal		
	Word device	Link register	16384	Jn\W0 to 3FFF	Hexadecimal		
		Link special register	512	Jn\SW0 to 1FF	Decimal		
Module access device	Word device	Intelligent function module device	65536	Un\G0 to 65535[*2]	Decimal		Section 9.5
Index register	Word device	Index register	10	Z0 to 9	Decimal		Section 9.6
File register[*5]	Word device	File register	64k	• R0 to 32767 • ZR0 to 65535	Decimal		Section 9.7
Nesting	-	Nesting	15	N0 to 14	Decimal		Section 9.8

(To the next page)

Table9.1 Device list (continued)

Classification	Type	Device name		Default			Parameter-set range	Reference
				Points	Range			
Pointer	-	Pointer		300	P0 to 299	Decimal	Cannot be changed	Section 9.9
		Interrupt pointer		128	I0 to 127	Decimal		Section 9.10
Other	Bit device	SFC block device		128	BL0 to 127	Decimal		Section 9.11.1
	-	Network No. specification device		239	J1 to 239	Decimal		Section 9.11.3
		I/O No. specification device	Q00JCPU	-	U0 to F	Hexadecimal		Section 9.11.4
			Q00CPU, Q01CPU	-	U0 to 3F	Hexadecimal		Section 9.11.4
	-	Macro instruction argument device		-	VD0 to □	Hexadecimal		Section 9.11.5
Constant	-	Decimal constant		K-2147483648 to 2147483647				Section 9.12.1
		Hexadecimal constant		H0 to FFFFFFFF				Section 9.12.2
		Real number constant		E±1.17550−38 to E±3.40282+38				Section 9.12.3
		Character string constant		"ABC", "123"[*4]				Section 9.12.4

*1: For the timer, retentive timer, and counter, a bit device is used for contacts and coils, and a word device is used for a present value.

*2: The number of points that can be actually used varies depending on the intelligent function module.
For the number of buffer memory points, refer to the manual for the intelligent function module used.

*3: Can be changed in the PLC parameter dialog box of GX Developer.
(Except the input, output, step relay, link special relay, and link special register) (☞ Section 9.2)

*4: Character strings can be used for $MOV, STR, DSTR, VAL, DVAL, ESTR, and EVAL instructions only.
Character strings cannot be used for other instructions.

*5: Since the Q00JCPU does not have the standard RAM, the file register cannot be used.

(2) High Performance model QCPU, Process CPU, and Redundant CPU

Table9.2 Device list

Classification	Type	Device name	Default			Parameter-set range	Reference
			Points	Range			
Internal user device	Bit device	Input	8192	X0 to 1FFF	Hexadecimal	Can be changed within 29K words *3	Section 9.2.1
		Output	8192	Y0 to 1FFF	Hexadecimal		Section 9.2.2
		Internal relay	8192	M0 to 8191	Decimal		Section 9.2.3
		Latch relay	8192	L0 to 8191	Decimal		Section 9.2.4
		Annunciator	2048	F0 to 2047	Decimal		Section 9.2.5
		Edge relay	2048	V0 to 2047	Decimal		Section 9.2.6
		Step relay	8192	S0 to 511/block	Decimal		Section 9.2.9
		Link relay	8192	B0 to 1FFF	Hexadecimal		Section 9.2.7
		Link special relay	2048	SB0 to 7FF	Hexadecimal		Section 9.2.8
	Word device	Timer*1	2048	T0 to 2047	Decimal		Section 9.2.10
		Retentive timer*1	0	(ST0 to 2047)	Decimal		
		Counter*1	1024	C0 to 1023	Decimal		Section 9.2.11
		Data register	12288	D0 to 12287	Decimal		Section 9.2.12
		Link register	8192	W0 to 1FFF	Hexadecimal		Section 9.2.13
		Link special register	2048	SW0 to 7FF	Hexadecimal		Section 9.2.14
Internal system device	Bit device	Function input	16	FX0 to F	Hexadecimal	Cannot be changed	Section 9.3.1
		Function output	16	FY0 to F	Hexadecimal		Section 9.3.1
		Special relay	2048	SM0 to 2047	Decimal		Section 9.3.2
	Word device	Function register	5	FD0 to 4	Decimal		Section 9.3.1
		Special register	2048	SD0 to 2047	Decimal		Section 9.3.3
Link direct device	Bit device	Timer*1	8192	Jn\X0 to 1FFF	Hexadecimal	Cannot be changed	Section 9.4
		Retentive timer*1	8192	Jn\Y0 to 1FFF	Hexadecimal		
		Counter*1	16384	Jn\B0 to 3FFF	Hexadecimal		
		Data register	512	Jn\SB0 to 1FF	Hexadecimal		
	Word device	Link register	16384	Jn\W0 to 3FFF	Hexadecimal		
		Link special register	512	Jn\SW0 to 1FF	Hexadecimal		
Module access device	Word device	Intelligent function module device	65536	Un\G0 to 65535*2	Decimal	Cannot be changed	Section 9.5
		Cyclic transmission area device *4	4096	U3En\G0 to 4095	Decimal	Can be changed	
Index register	Word device	Index register	16	Z0 to 15	Decimal	Cannot be changed	Section 9.6
File register	Word device	File register	0	-	-	0 to 1018k	Section 9.7
Nesting	-	Nesting	15	N0 to 14	Decimal	Cannot be changed	Section 9.8
Pointer	-	Pointer	4096	P0 to 4095	Decimal	Cannot be changed	Section 9.9
		Interrupt pointer	256	I0 to 255	Decimal		Section 9.10

(To the next page)

Table9.2 Device list (continued)

Classification	Type	Device name	Default			Parameter-set range	Reference
			Points	Range			
Other	Bit device	SFC block device	320	BL0 to 319	Decimal	Cannot be changed	Section 9.11.1
		SFC transition device	512	TR0 to 511	Decimal		Section 9.11.2
	-	Network No. specification device	255	J1 to 255	Hexadecimal		Section 9.11.3
		I/O No. specification device	-	U0 to FF	Hexadecimal		Section 9.11.4
		Macro instruction argument device	-	VD0 to □	Hexadecimal		Section 9.11.5
Constant	-	Decimal constant	K-2147483648 to 2147483647				Section 9.12.1
		Hexadecimal constant	H0 to FFFFFFFF				Section 9.12.2
		Real number constant	Single-precision floating-point data: E±1.17549435−38 to E±3.40282347+38				Section 9.12.3
			Double-precision floating-point data: E±2.2250738585072014−308 to E±1.7976931348623157+308				Section 9.12.3
		Character string constant	"ABC", "123"				Section 9.12.4

*1: For the timer, retentive timer, and counter, a bit device is used for contacts and coils, and a word device is used for a present value.

*2: The number of points that can be actually used varies depending on the intelligent function module.
For the number of buffer memory points, refer to the manual for the intelligent function module used.

*3: Can be changed in the PLC parameter dialog box of GX Developer.
(Except the input, output, step relay, link special relay, and link special register) (☞ Section 9.2)

*4: Can be only used in multiple CPU systems.

9.2 Internal User Devices

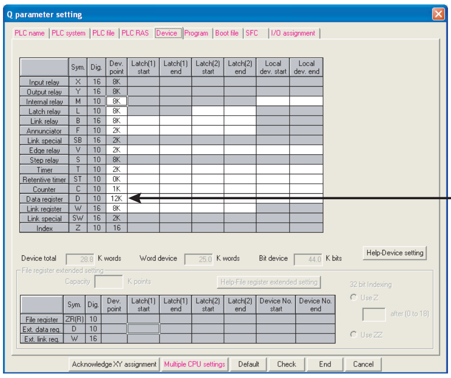
(1) Definition

Internal user devices can be used for various user applications.

(2) Points for internal user devices

The default values can be changed in the Device tab of the PLC parameter dialog box.

However, the points for the input (X), output (Y), step relay (S), link special relay (SB), and link special register (SW) cannot be changed.



	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Local dev. start	Local dev. end
Input relay	X	16	8K						
Output relay	Y	16	8K						
Internal relay	M	10	8K						
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	12K						
Link register	W	16	8K						
Link special	SW	16	2K						
Index	Z	10	16						

Device total 28.8 K words Word device 25.0 K words Bit device 44.0 K bits

File register extended setting

Capacity K points

	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Device No. start	Device No. end
File register	ZR(R)	10							
Ext. data reg.	D	10							
Ext. link reg.	W	16							

Default
Most of the default device
points can be changed.

Figure 9.1 Device tab of the PLC parameter dialog box

When changing device points, note the following.

- Set each device in increments of 16 points.
- The maximum number of points for a bit device is 32K.
- The maximum number of total points for all internal user devices varies depending on the CPU module.
  - Basic model QCPU: 16.4K words
  - High Performance model QCPU, Process CPU, and Redundant CPU: 29K words
- One point set for a timer, a retentive timer, or a counter is regarded as one point for a word device plus two points for a bit device.

Point

- When changing device points, the following refresh ranges must not exceed the corresponding device ranges.
  - Refresh range of network module
  - 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 is up to 64K points.

- 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
  - ST program files

- When change the device points of the internal user devices, perform the following operations from GX Developer.

  [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.

  ☞ GX Developer Version 8 Operating Manual

(3) 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) ≦ 29K words

(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.

$$\text{(Word device size)} = \frac{(D+W+SW)}{16} \times 16 \text{ Words}$$

(4) Device point assignment example

Table9.3 shows a device point assignment example.

Table9.3 uses the same format as the device point assignment sheet shown in Appendix 3.

Table9.3 Device point assignment example

Device name	Sym-bol	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 1FFF	/16 512	× 1 8192
Output relay*1	Y	Hexadecimal	8k(8192)	Y0000 to 1FFF	/16 512	× 1 8192
Internal relay	M	Decimal	16k(16384)	M0 to 16383	/16 1024	× 1 16384
Latch relay	L	Decimal	4k(4096)	L0 to 4095	/16 256	× 1 4096
Link relay	B	Hexadecimal	4k(4096)	B0000 to 0FFF	/16 256	× 1 4096
Annunciator	F	Decimal	1k(1024)	F0 to 1023	/16 64	× 1 1024
Link special relay*1	SB	Hexadecimal	2k(2048)	SB0000 to 07FF	/16 128	× 1 2048
Edge relay	V	Decimal	1k(1024)	V0 to 1023	/16 64	× 1 1024
Step relay *1	S	Decimal	8k(8192)	S0 to 8191	/16 512	× 1 8192
Timer	T	Decimal	2k(2048)	T0 to 2047	$\times \frac{18}{16}$ 2304	× 2 4096
Retentive timer	ST	Decimal	2k(2048)	ST0 to 2047	$\times \frac{18}{16}$ 2304	× 2 4096
Counter	C	Decimal	1k(1024)	C0 to 1023	$\times \frac{18}{16}$ 1152	× 2 2048
Data register	D	Decimal	14k(14336)	D0 to 14335	× 1 14336	-
Link register	W	Hexadecimal	4k(4096)	W0000 to 4095	× 1 4096	-
Link special register*1	SW	Hexadecimal	2k(2048)	SW0000 to 07FF	× 1 2048	-
Total					29568 (29696 or less)	63488 (65536 or less)

*1: The points are fixed for the system. (Cannot be changed)

*2: Up to 32K points can be set for each device.

*3: Enter the values multiplied (or divided) by the number shown in the Size (words) column.

9.2.1 Input (X)

(1) Definition

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.

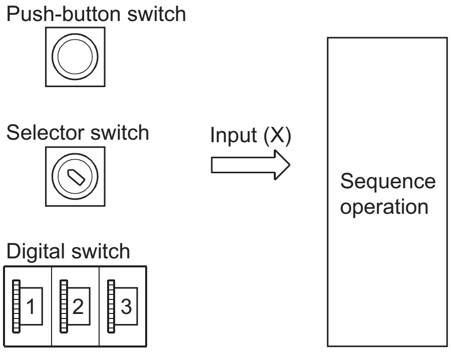


Figure 9.2 Commands from external devices to a CPU module

(2) 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.

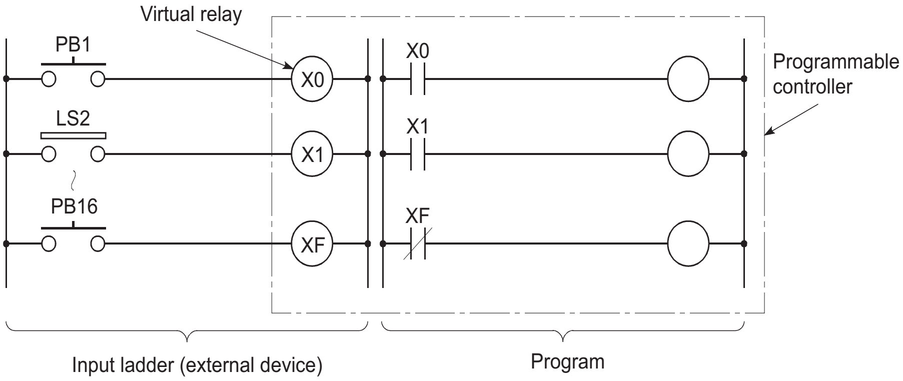


Figure 9.3 Concept of input (X)

(3) 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.

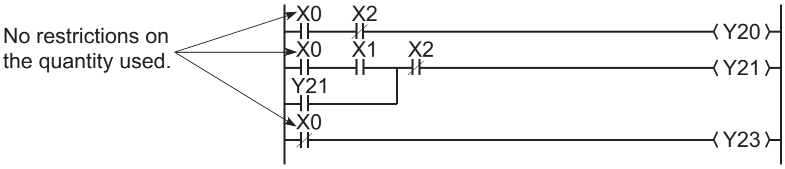


Figure 9.4 Input (X) used in a program

Point

- When debugging a program, the input (X) can be set to on or off by the following:
  - Device test in GX Developer
  - OUT Xn instruction

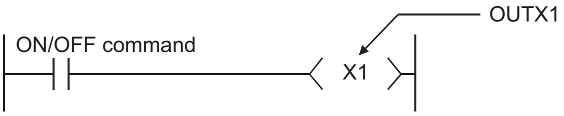


Figure 9.5 Input (X) on/off with the OUT Xn instruction

- The input (X) can also be used for the following.
  - Refresh target device (CPU module side) of RX in CC-Link
  - Refresh target device (CPU module side) of CC-Link IE Controller Network or MELSECNET/H

9.2.2 Output (Y)

(1) Definition

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.

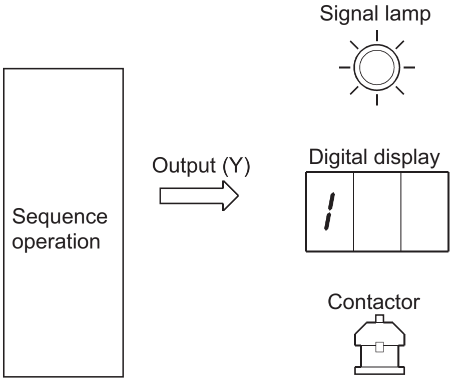


Figure 9.6 Output from a CPU module to external devices

(2) 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.

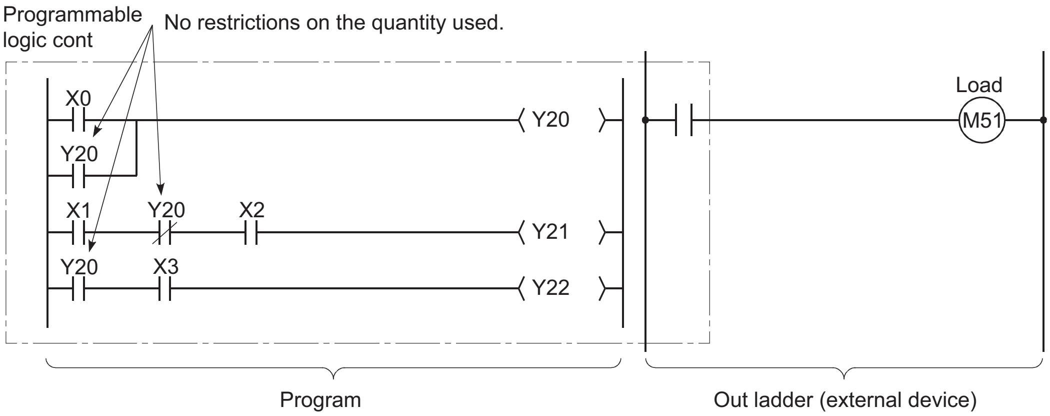

Programmable logic cont
No restrictions on the quantity used.
X0
Y20
Y20
X1
Y20
X2
Y21
Y20
X3
Y22
Load
M51
Program
Out ladder (external device)

Figure 9.7 Output (Y) used in a program

(3) 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).

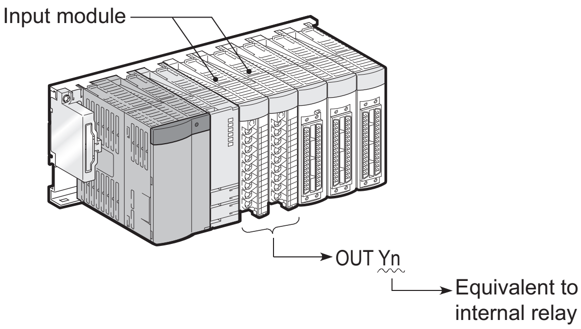


Figure 9.8 Using as the internal relay

9.2.3 Internal relay (M)

(1) Definition

The internal relay (M) is a device for auxiliary relays used in the CPU module.

All of the internal relay are set to off in the following cases:

- When the CPU module is powered off and then on
- When the CPU module is reset
- When latch clear is executed (☞ Section 6.3)

(2) Latch (data retention during power failure)

The internal relay cannot be latched.

(3) 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.

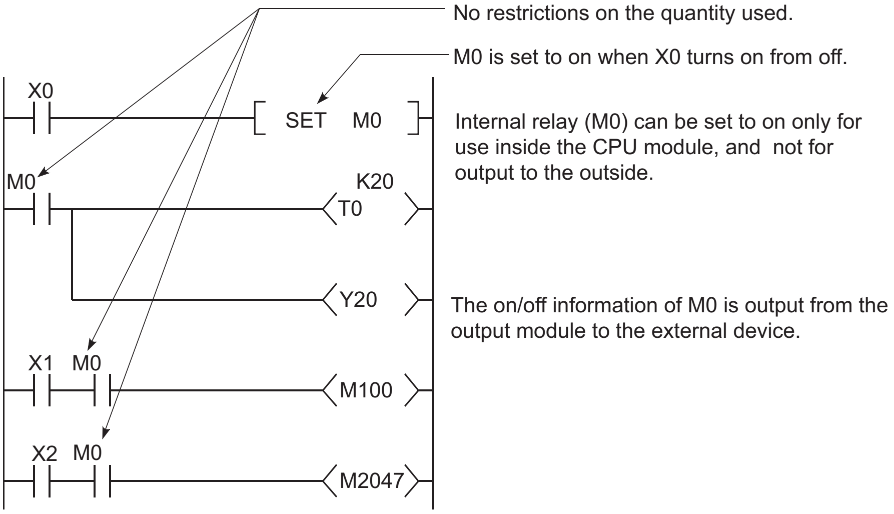

No restrictions on the quantity used.
M0 is set to on when X0 turns on from off.
X0
SET M0
Internal relay (M0) can be set to on only for use inside the CPU module, and not for output to the outside.
M0
K20
T0
Y20
The on/off information of M0 is output from the output module to the external device.
X1 M0
M100
X2 M0
M2047

Figure 9.9 Internal relay (M) used in a program

(4) 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 (☞ Section 9.2.4)

9.2.4 Latch relay (L)

(1) Definition

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

(2) Latch relay clear

The latch relay is turned off by the latch clear operation. (☞ Section 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 if the RESET/L.CLR switch Note9.1 or the remote latch clear function is used.

(3) 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.

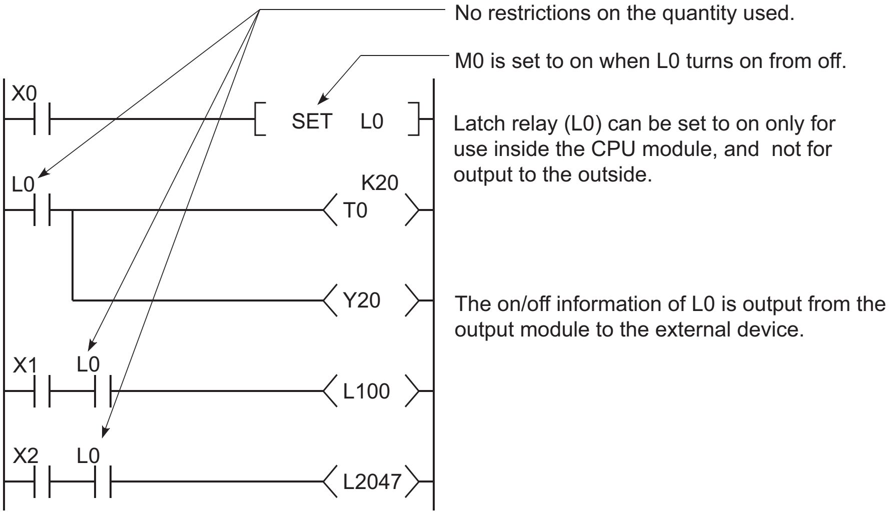

No restrictions on the quantity used.
M0 is set to on when L0 turns on from off.
X0
SET L0
Latch relay (L0) can be set to on only for use inside the CPU module, and not for output to the outside.
L0
K20
T0
Y20
The on/off information of L0 is output from the output module to the external device.
X1
L0
L100
X2
L0
L2047

Figure 9.10 Latch relay

Note9.1 **Basic**

For the Basic model QCPU, switch operation is not allowed to clear the latch.

(4) 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). (☞ Section 9.2.3)
- The latch clear invalid area is set in the Device setting of PCL parameter. (☞ Section 6.3)

9.2.5 Annunciator (F)

(1) Definition

The annunciator (F) is an internal relay which can be effectively used in fault detection programs for user-created system.

(2) Special relay and special register after annunciator ON

When the annunciator is turned on, the special relay (SM62) is set to on, and the numbers and quantity of the annunciator numbers are stored in the special register (SD62 to SD79).

• Special relay	:	SM62	· · ·	Turns on even if only one of the annunciator number areas is turned on.
• Special register	:	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.)

The annunciator number stored in SD62 is also registered in the error history area.

Point

For the Basic model QCPU, one annunciator number only is stored in the error history area while the programmable controller is powered on.

(3) 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.

Example In this program, when annunciator (F5) is turned on, the corresponding annunciator number is output to the outside.

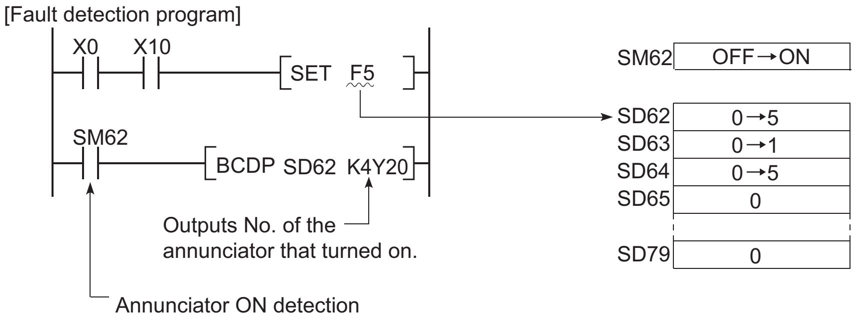


Figure 9.11 Detecting and saving the annunciator ON

(4) 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.

(5) Turning on the annunciator and processing

(a) Turning on the annunciator

The following instructions can be used.

1) 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.

2) 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 Note9.2, 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.

- If annunciator data are sent from the control system to the standby system by tracking of the redundant system, the annunciator in the standby system does not turn on.
  For processing of annunciator data tracking, refer to the following.
  ☞ QnPRHCPU User's Manual (Redundant System)

Note9.2 **Basic**

For the Basic model QCPU, the LEDR instruction cannot be used.

(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".

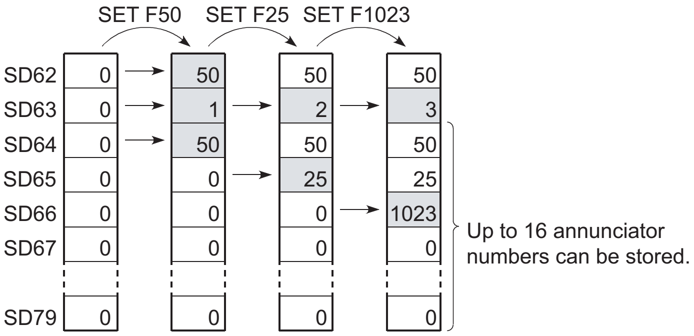


Figure 9.12 Processing after annunciator ON

2) Processing on the CPU

- When using the Basic model QCPU:
  The ERR LED on the front side turns on.
- When using the High Performance model QCPU, Process CPU, or Redundant CPU:
  The USER LED on the front side turns on (red).

3) On/off setting for the LED

By setting indication priority for an error in SD207 to SD209, whether to turn on the USER. LED (the ERR. LED for the Basic model QCPU) or not when the annunciator is turned on can be set.

(☞ Section 6.21.2)

(6) Turning off the annunciator and processing

(a) Turning off the annunciator

The following instructions can be used.

1) RST F□ instruction

This is used to turn off the annunciator number that was turned on with the SET F□ instruction.

2) LEDR instruction Note9.3

This is used to turn off the annunciator number stored in SD62 and SD64.

3) BKRST instruction

This is used to turn off all of the annunciator numbers within the specified range.

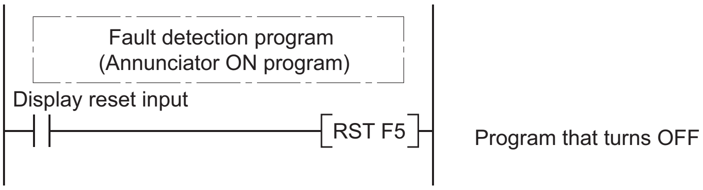
4) OOUT F instruction

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 (6)(b) in this section 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.

- Turning off annunciator 5 (F5)



- Turning off all of the turned-on annunciator numbers

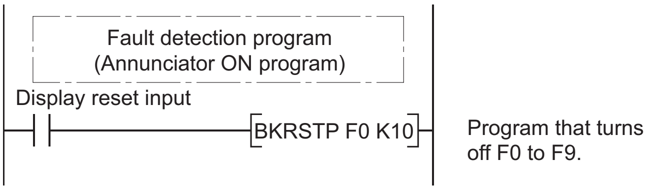


Figure 9.13 Example of turning off the annunciator

Remark

For details of each instruction, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

Note9.3 **Basic**

For the Basic model QCPU, the LEDR instruction cannot be used.

(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.

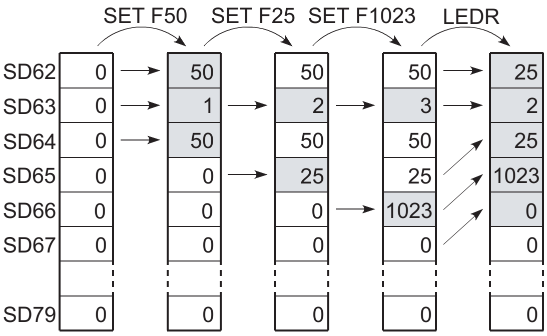


Figure 9.14 Processing after annunciator OFF (when the LEDR instruction executed)

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.

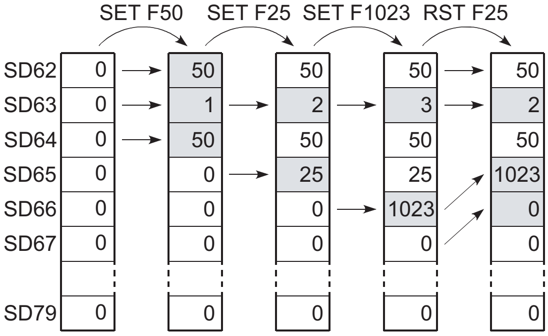


Figure 9.15 Processing after annunciator OFF (when the REST F□ instruction is executed)

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. (☞ (5)(b) in this section)

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 (☞ Section 6.21.2) 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.

9.2.6 Edge relay (V)

(1) Definition

The edge relay (V) is a device in which the on/off information from the beginning of the ladder block. Contacts only can be used. (Coils cannot be used).

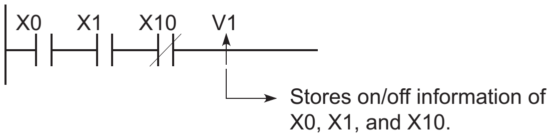
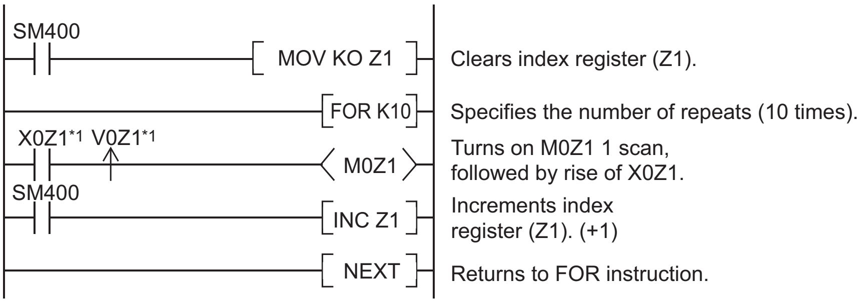


Figure 9.16 Edge relay

(2) 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]


SM400
MOV KO Z1
Clears index register (Z1).
FOR K10
Specifies the number of repeats (10 times).
X0Z1*1 V0Z1*1
M0Z1
Turns on M0Z1 1 scan, followed by rise of X0Z1.
SM400
INC Z1
Increments index register (Z1). (+1)
NEXT
Returns to FOR instruction.

*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]

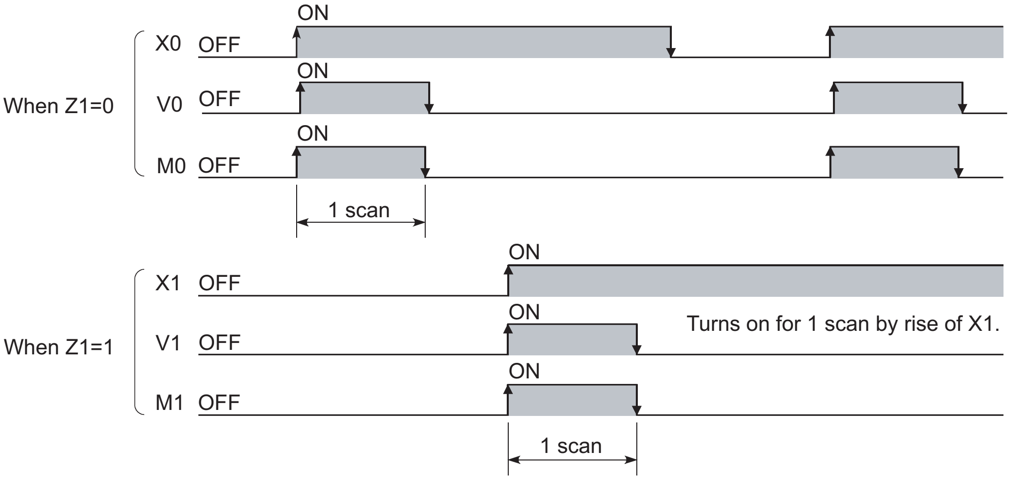


Figure 9.17 Application example and timing chart of the edge relay

(3) Precautions

The edge relay of the same number cannot be set only once in a program.

9.2.7 Link relay (B)

(1) Definition

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 MELECNET/H network module to the CPU module or refreshing the CPU module data to the link relay (LB) of the MELECNET/H network module.

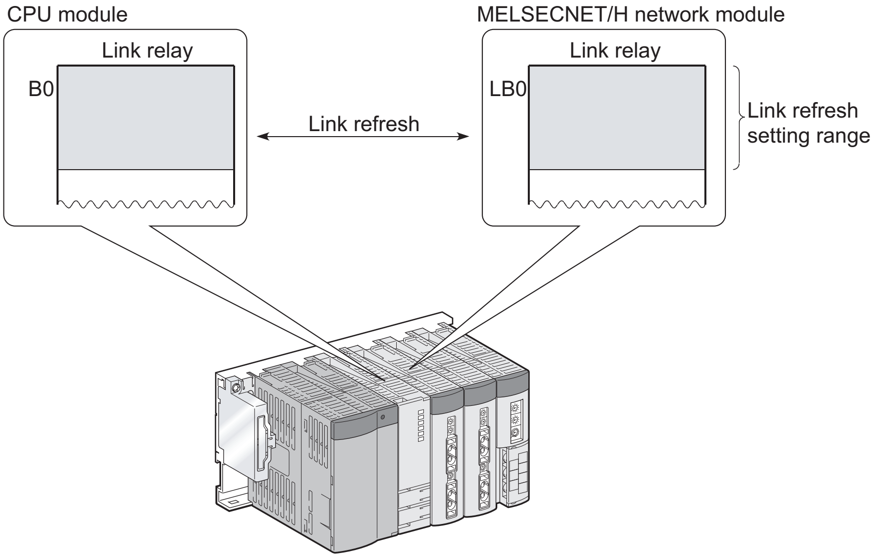


Figure 9.18 Link refresh

(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.

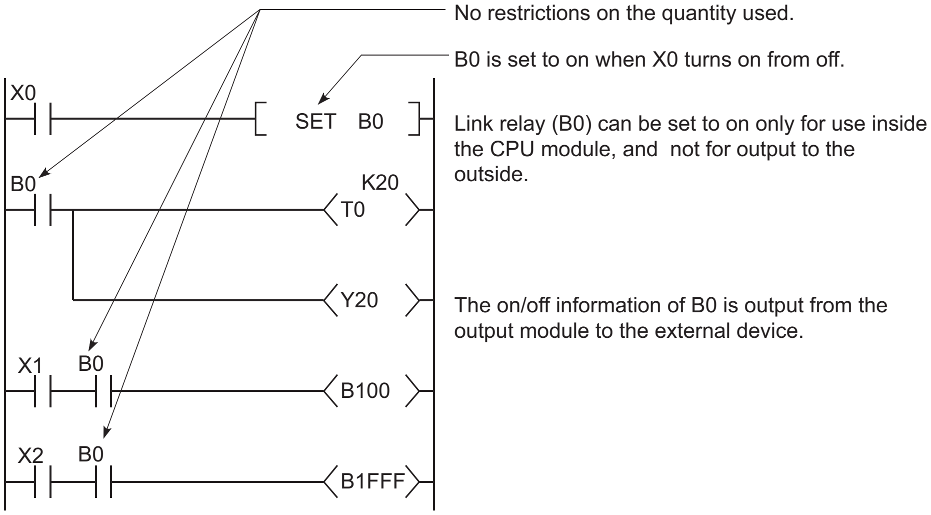

No restrictions on the quantity used.
B0 is set to on when X0 turns on from off.
X0
SET B0
Link relay (B0) can be set to on only for use inside the CPU module, and not for output to the outside.
B0
K20
T0
Y20
The on/off information of B0 is output from the output module to the external device.
X1
B0
B100
X2
B0
B1FFF

Figure 9.19 Link relay

(3) Using the link relay in the network system

Network parameters must be set.

The link relay range areas that is 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

- Although the points for the link relay in a CC-Link IE Controller Network module is 32768, the default for the link relay in the CPU module is 8192 points.
- Although the points for the link relay in a MELSECNET/H network module is 16384, the default for the link relay in the CPU module is 8192 points.

To use the link relay exceeding the above points, change the link relay points in the Device tab of the PLC parameter dialog box.

Remark

For network parameters, refer to the following.

☞ Manual for each network module

9.2.8 Link special relay (SB)

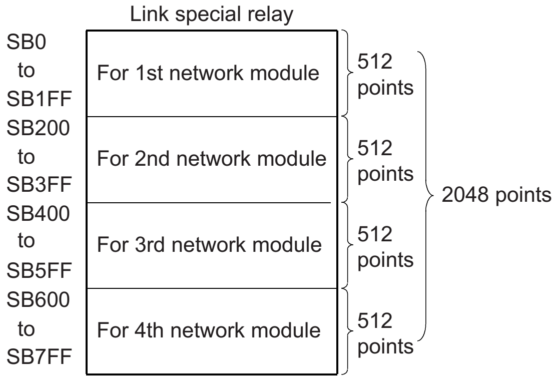
(1) Definition

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 Controller Network or MELSECNET/H network modules.
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).

(2) Number of link special relay points

The points for the link special relay depends on the CPU module.

Table9.4 Link special relay points for each CPU module type

CPU module	Link special relay points
Basic model QCPU	The points for the link special relay in the CPU module is 1024 (SB0 to SB3FF). To an intelligent function module that has a link special relay, such as a CC-Link IE Controller Network module or MELSECNET/H network module, 512 points are assigned.
High Performance model QCPU, Process CPU, and Redundant CPU	The points for the link special relay in the CPU module is 2048 (SB0 to SB7FF). To an intelligent function module that has a link special relay, such as a CC-Link IE Controller Network module or MELSECNET/H network module, 512 points are assigned. The link special relay can be allocated as shown below.  Link special relay SB0 to SB1FF For 1st network module 512 points SB200 to SB3FF For 2nd network module 512 points SB400 to SB5FF For 3rd network module 512 points SB600 to SB7FF For 4th network module 512 points 2048 points

Remark

For details of the link special relay, refer to the manual for an intelligent function module that has the link special relay.

9.2.9 Step relay (S)

This device is provided for SFC programs.

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.

Remark

For use of the step relay, refer to the following.
☞ MELSEC-Q/L/QnA Programming Manual (SFC)

9.2.10Timer (T)

(1) Definition

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.

(2) 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).

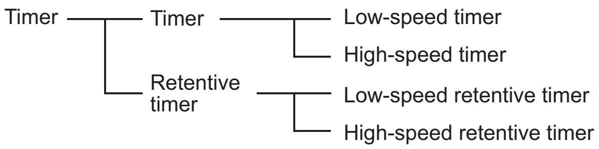


Figure 9.20 Timer types

(3) 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.

Example 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.

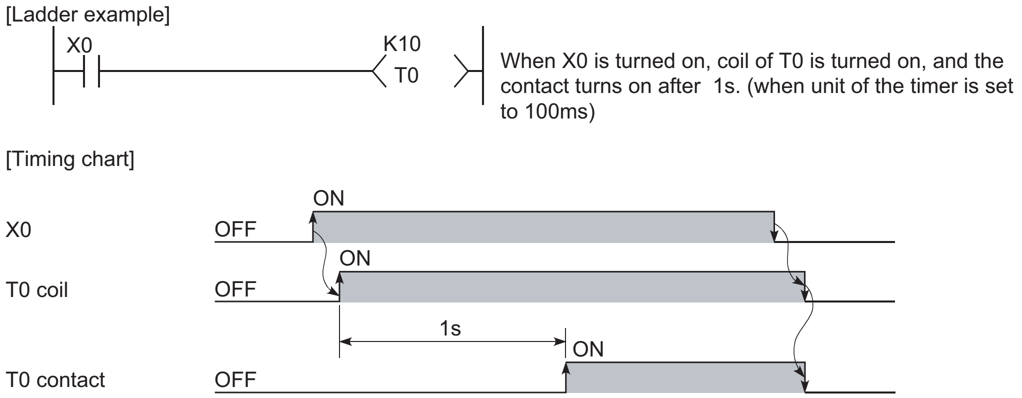
Example For the OUT ST0 instruction, the low-speed retentive timer is specified, and for the OUTH ST0 instruction, the high-speed retentive timer is specified.

(4) Low-speed timer

(a) Definition

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]

When X0 is turned on, coil of T0 is turned on, and the contact turns on after 1s. (when unit of the timer is set to 100ms)

[Timing chart]

Figure 9.21 Ladder example and timing chart of a low-speed timer

(b) Time increment setting

The time increment is set in the PLC system tab of the PLC parameter dialog box.

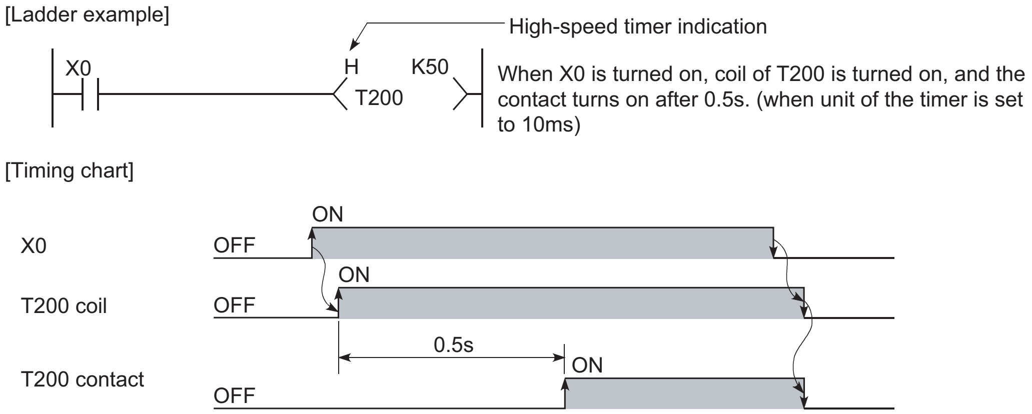
The default is 100ms, and it can be changed in increments of 1ms.

(5) High-speed timer

(a) Definition

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]

High-speed timer indication

When X0 is turned on, coil of T200 is turned on, and the contact turns on after 0.5s. (when unit of the timer is set to 10ms)

[Timing chart]

Figure 9.22 Ladder example and timing chart of a high-speed timer

(b) 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.

(6) Retentive timer

(a) Definition

This timer measures the period of time during which the coil is on.

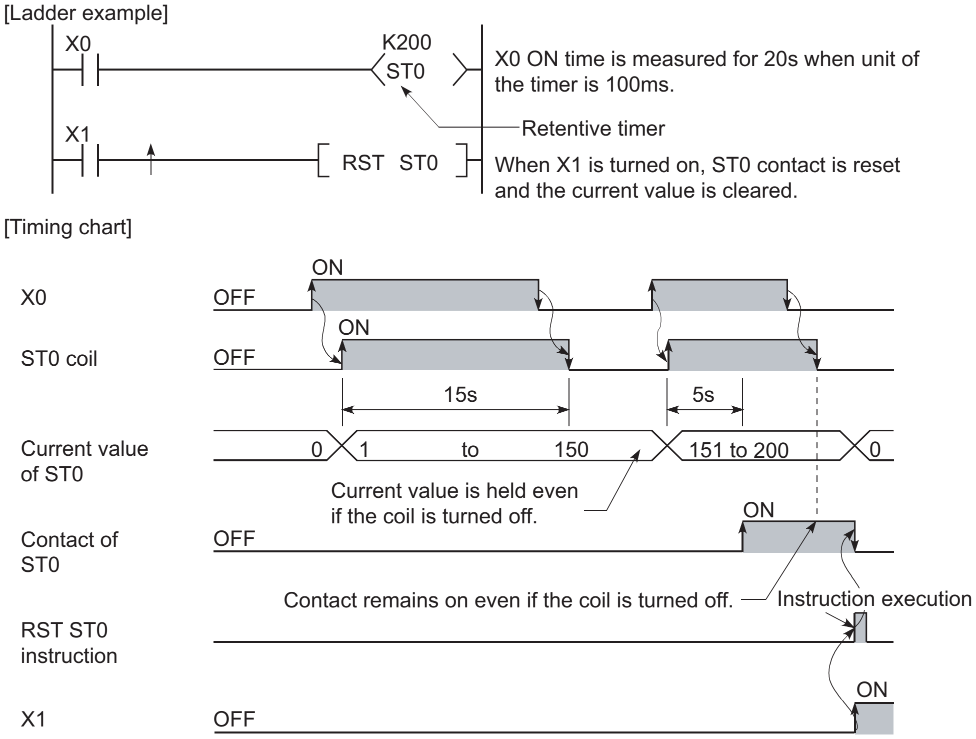
The timer starts time measurement when its coil is turned on, and when it times out, the contact is turned on.

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.

(b) Retentive timer clear

The current value and the contact off status can be cleared with the RST ST□ instruction.



[Ladder example]
X0
K200
ST0
X0 ON time is measured for 20s when unit of the timer is 100ms.
Retentive timer
X1
RST ST0
When X1 is turned on, ST0 contact is reset and the current value is cleared.
[Timing chart]
X0
ON
OFF
ST0 coil
ON
OFF
15s
5s
Current value of ST0
0
1 to 150
151 to 200
0
Current value is held even if the coil is turned off.
Contact of ST0
ON
OFF
Contact remains on even if the coil is turned off.
Instruction execution
RST ST0 instruction
X1
ON
OFF

Figure 9.23 Ladder example and timing chart of a retentive timer

(c) 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

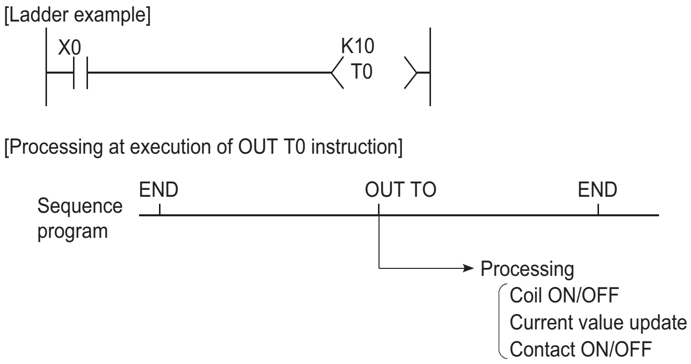
To use a retentive timer, set the points for it in the Device tab of the PLC parameter dialog box.

(7) 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]

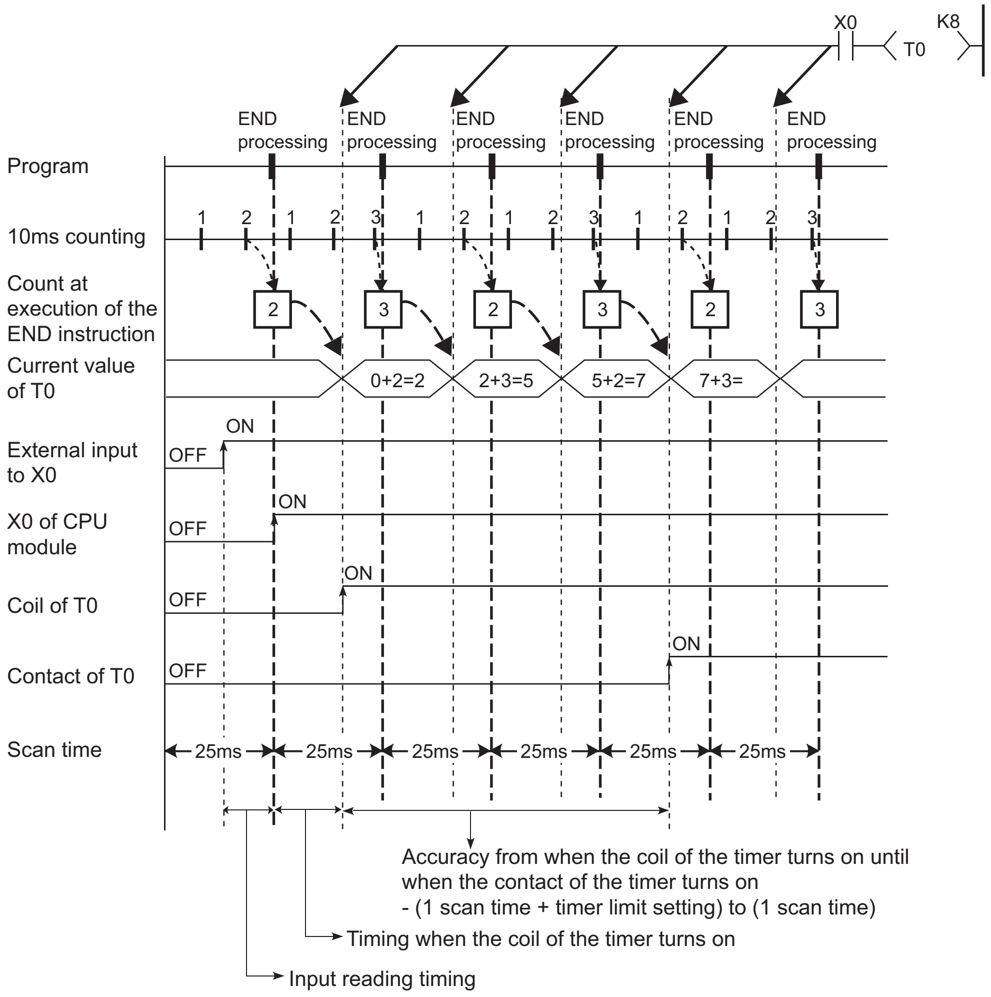
[Processing at execution of OUT T0 instruction]

Figure 9.24 Processing at execution of the 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

Figure 9.25 Timer accuracy (in the case of 10ms)

Accuracy of the timer response that is from reading input (X) to output the data is up to "2-scan time + timer limit setting".

(8) 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.

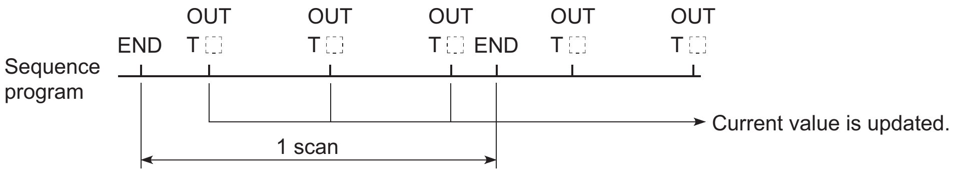


Figure 9.26 When the same timer is used

(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

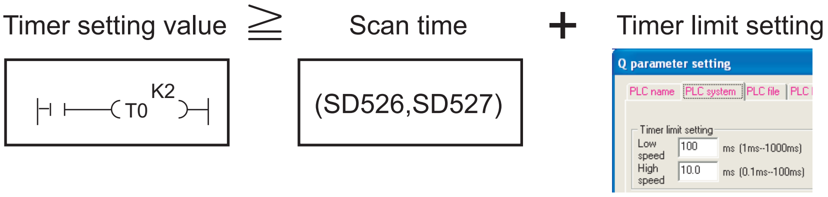
Timers cannot be used in interrupt programs and fixed scan execution programs.

(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:

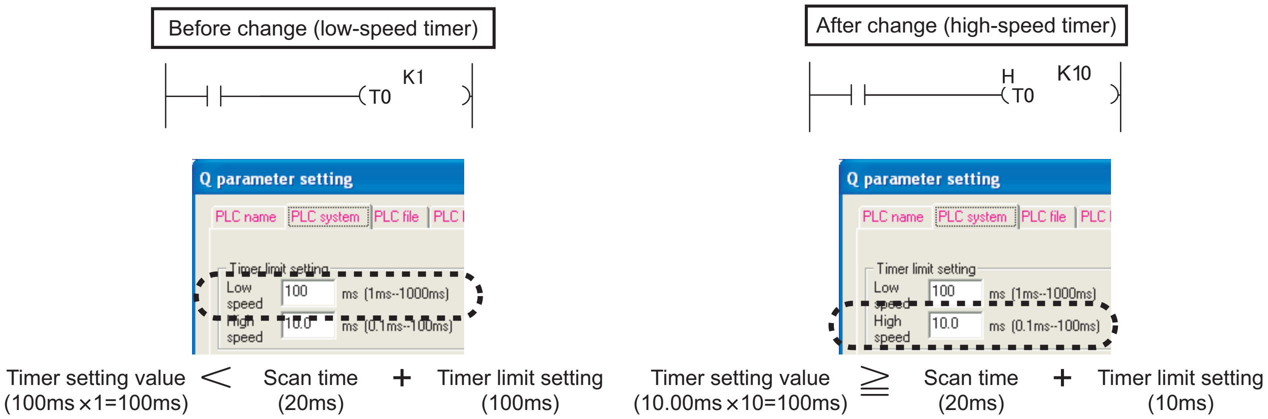


Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

Figure 9.27 Relationship between timer setting value and timer limit setting

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.

Example 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)
K1
T0
After change (high-speed timer)
H
K10
T0
Q parameter setting
PLC name
PLC system
PLC file
PLC
Q parameter setting
PLC name
PLC system
PLC file
PLC
Timer setting value < Scan time + Timer limit setting
(100ms×1=100ms) (20ms) (100ms)
Timer setting value ≧ Scan time + Timer limit setting
(10.00ms×10=100ms) (20ms) (10ms)

Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

Figure 9.28 Making the value of the timer limit setting smaller

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":

Example 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.

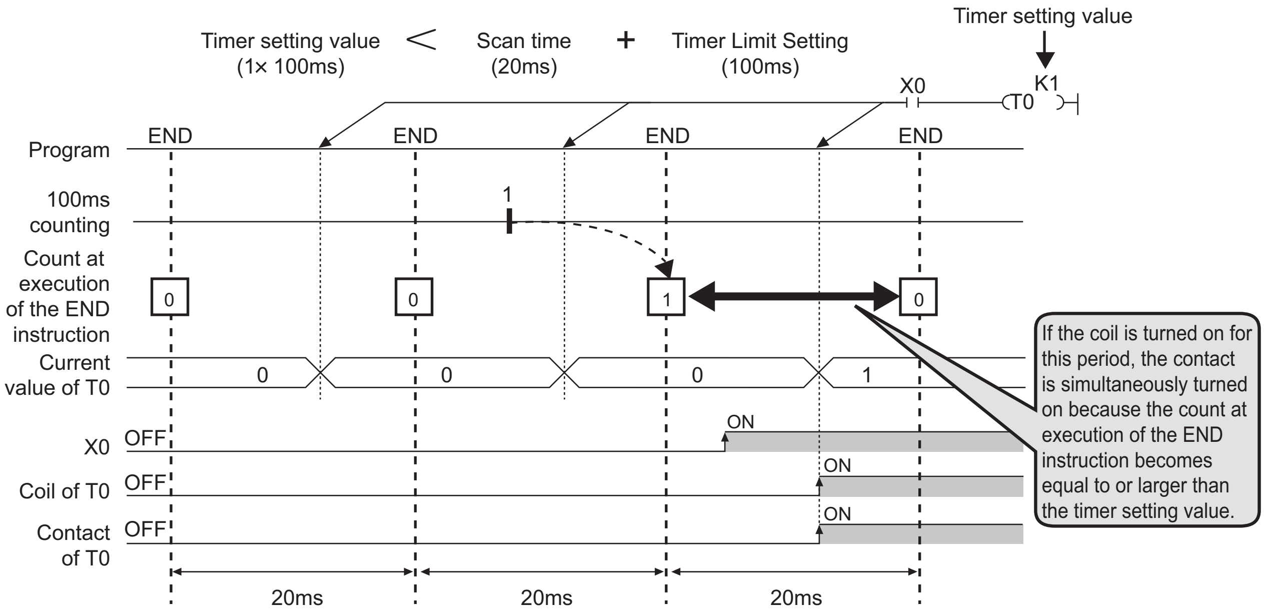

Timer setting value (1× 100ms)
<
Scan time (20ms)
+
Timer Limit Setting (100ms)
Timer setting value
X0
K1
T0
Program
END
END
END
END
100ms counting
1
Count at execution of the END instruction
0
0
1
0
Current value of T0
0
0
0
1
X0
OFF
ON
Coil of T0
OFF
ON
Contact of T0
OFF
ON
20ms
20ms
20ms
If the coil is turned on for this period, the contact is simultaneously turned on because the count at execution of the END instruction becomes equal to or larger than the timer setting value.

Figure 9.29 The coil and the contact are simultaneously turned on with the timer setting value being 1

Example 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.

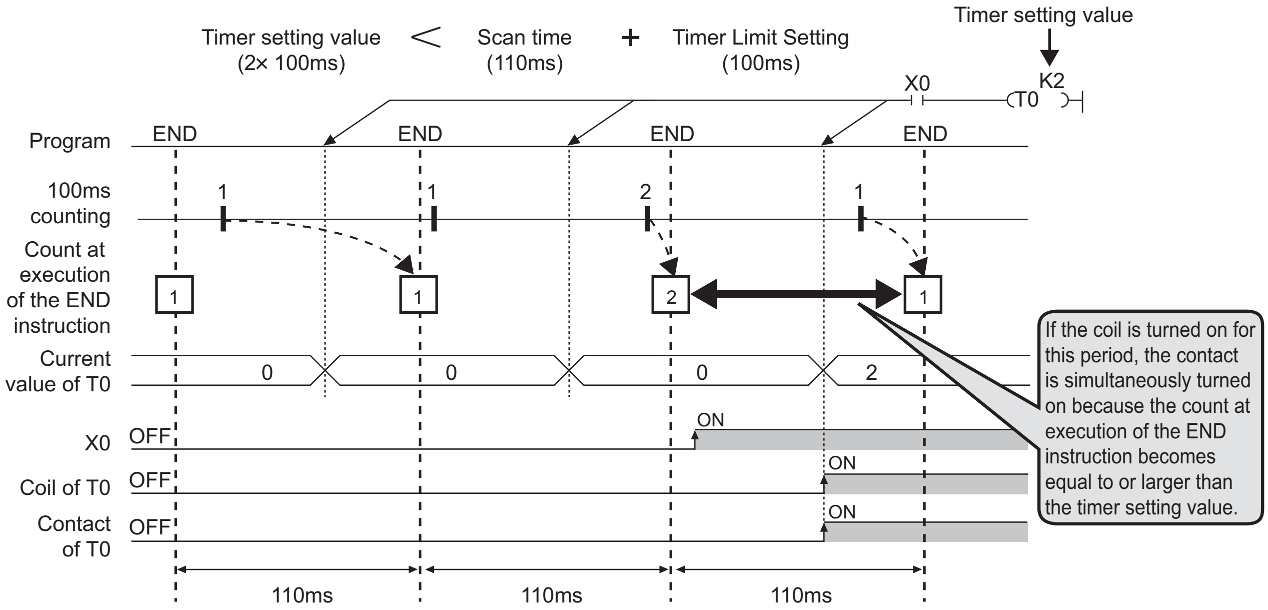

Timer setting value (2× 100ms)
<
Scan time (110ms)
+
Timer Limit Setting (100ms)
Timer setting value
X0
K2
T0
Program
END
END
END
END
100ms counting
1
1
2
1
Count at execution of the END instruction
1
1
2
1
Current value of T0
0
0
0
2
X0
OFF
ON
Coil of T0
OFF
ON
Contact of T0
OFF
ON
110ms
110ms
110ms
If the coil is turned on for this period, the contact is simultaneously turned on because the count at execution of the END instruction becomes equal to or larger than the timer setting value.

Figure 9.30 The coil and the contact are simultaneously turned on with the timer setting value being 2

(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.

For example, to create an on/off ladder using two timers, refer to examples shown in Figure 9.31.

[Correct ladder example]

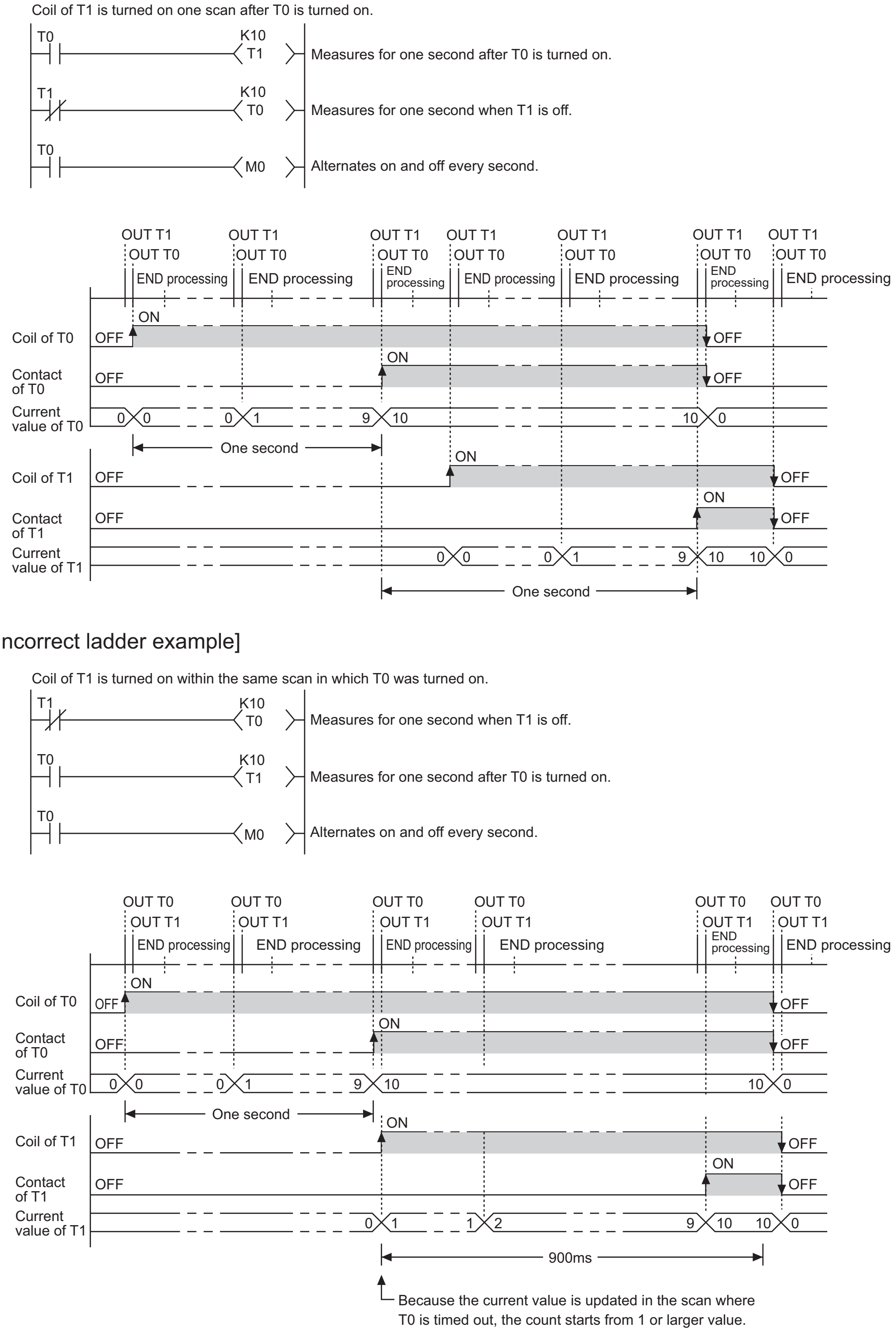

Coil of T1 is turned on one scan after T0 is turned on.
T0
K10
T1
Measures for one second after T0 is turned on.
T1
K10
T0
Measures for one second when T1 is off.
T0
M0
Alternates on and off every second.
OUT T1
OUT T0
END processing
Coil of T0
Contact of T0
Current value of T0
ON
OFF
0 0 0 1 9 10 10 0
One second
Coil of T1
Contact of T1
Current value of T1
0 0 0 1 9 10 10 0
One second
[Incorrect ladder example]
Coil of T1 is turned on within the same scan in which T0 was turned on.
T1
K10
T0
Measures for one second when T1 is off.
T0
K10
T1
Measures for one second after T0 is turned on.
T0
M0
Alternates on and off every second.
OUT T0
OUT T1
END processing
Coil of T0
Contact of T0
Current value of T0
ON
OFF
0 0 0 1 9 10 10 0
One second
Coil of T1
Contact of T1
Current value of T1
0 1 1 2 9 10 10 0
900ms
Because the current value is updated in the scan where T0 is timed out, the count starts from 1 or larger value.

Figure 9.31 On/off ladder using two timers

9.2.11 Counter (C)

(1) Definition

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.

(2) Counter types

Counters are mainly classified into the following two types.

- Counter that counts the number of rises for input conditions in sequence programs
- Interrupt counter that counts the number of interrupts occurred

(3) 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.

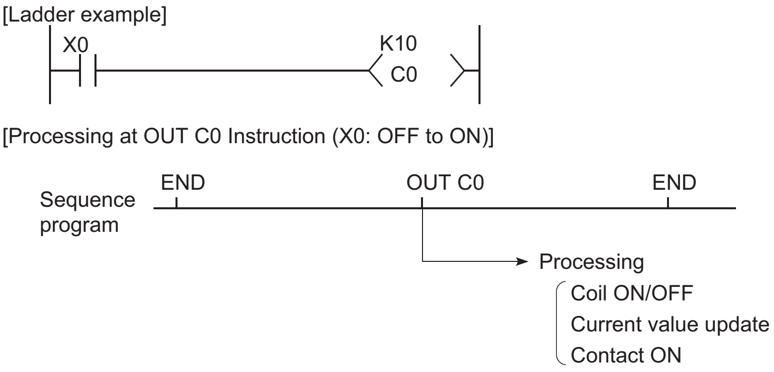
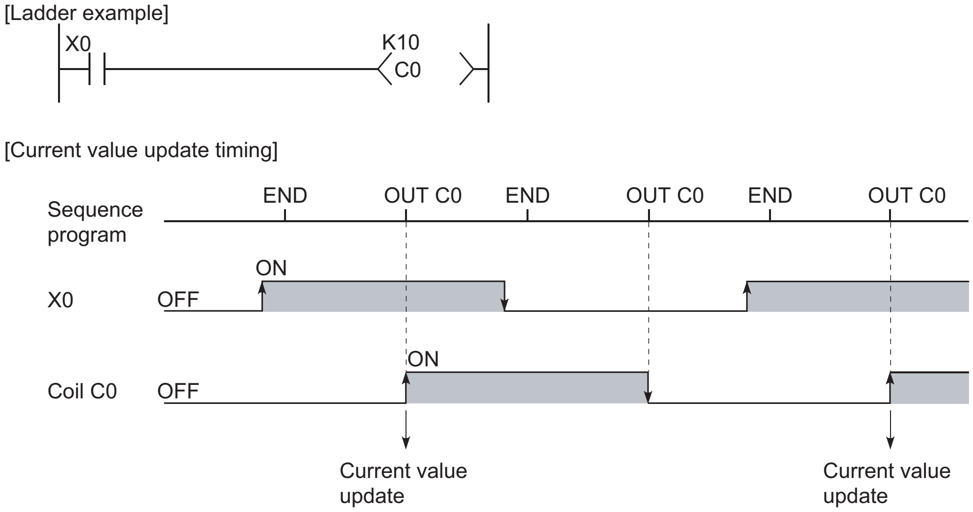


Figure 9.32 Execution and processing of the OUT C□ instruction

(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.



[Ladder example]

[Current value update timing]

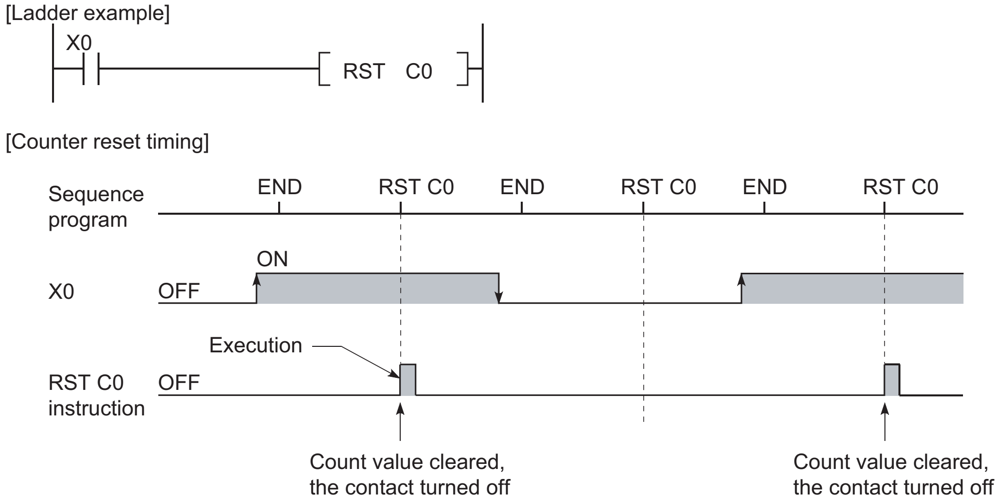
Figure 9.33 Current value update timing

(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]

Figure 9.34 Resetting the counter

1) 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).

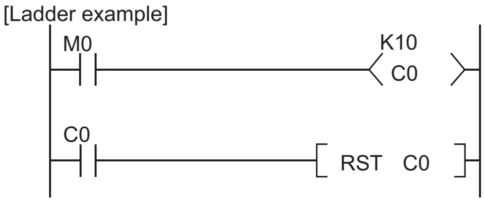


Figure 9.35 Counter resetting ladder example

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.)

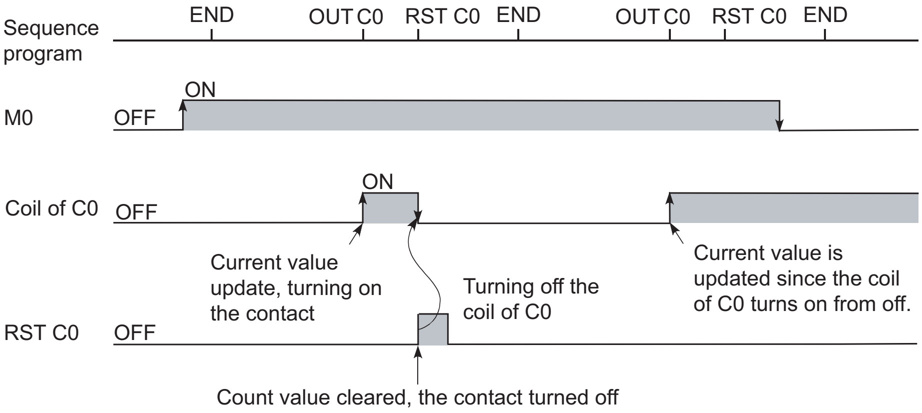


Figure 9.36 Current value update timing

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.

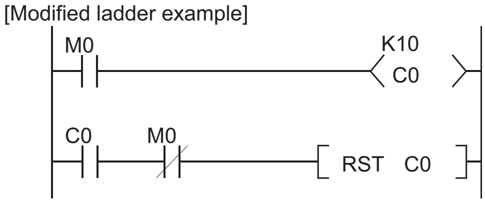


Figure 9.37 Counter resetting ladder example (recommended)

(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:

$$\boxed{\text{Maximum counting speed (Cmax)} = \frac{n}{100} \times \frac{1}{T} \text{ [times/s]}}$$

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.

When T1 ≧ T2, $n = \frac{T2}{T1+T2} \times 100\%$

When T1 < T2, $n = \frac{T1}{T1+T2} \times 100\%$

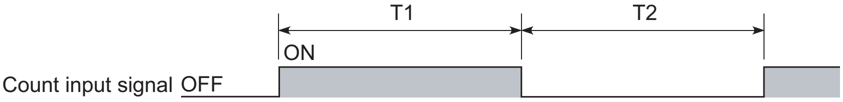


Figure 9.38 Duty ratio

Point

The maximum counting speed can be increased by placing multiple counters within one scan.

At this time, use the direct access input (DX□)(☞ Section 3.8.2) for the counter input signal.

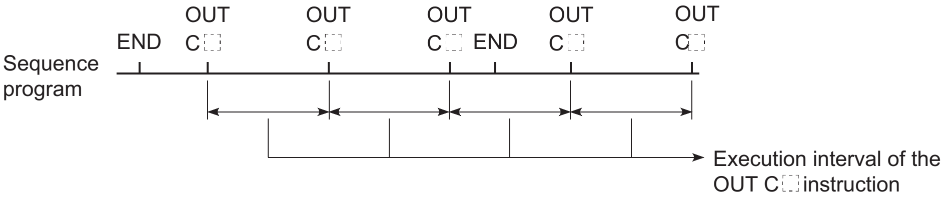


Figure 9.39 Ladder example for increasing the maximum counting speed

(4) Interrupt counter

(a) Definition

The interrupt counter counts how many times an interrupt factor has occurred.

(b) Count processing

1) When an interrupt occurs

The interrupt counter updates its current value when an interrupt occurs.
To utilize the interrupt counter, any program that includes the interrupt counter is not required.

2) Counting of the interrupt counter

The interrupt counter does not stop its counting even if a set value is specified.
To use the interrupt counter for control purposes, use comparison instructions (=, <=, etc.) for comparison with the set value to turn on or off a device such as the internal relay (M).
For example, to turn on M0 when interrupt input to I0 turned on ten times, create the program as shown in Figure 9.40. (In this case, "C300" is the interrupt counter corresponding to I0.)

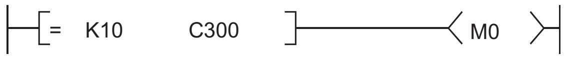
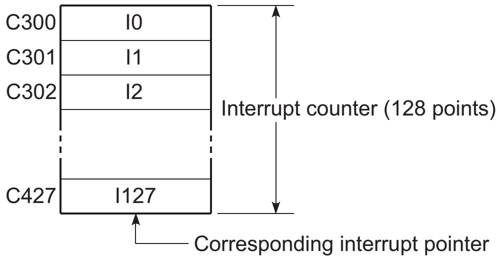
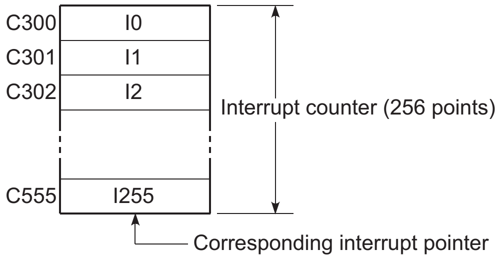


Figure 9.40 Ladder example of using the interrupt counter for control

(c) Setting the interrupt counter

Set the "Interrupt counter start No." in the PLC system tab of the PLC parameter dialog box.
The number of points indicated in Table9.5 starting from the set counter number is used as the interrupt counter.

Table9.5 Interrupt counter points for each CPU module

Basic model QCPU	High Performance model QCPU, Process CPU, Redundant CPU
128 points, starting from the set counter number, are used as the interrupt counter. When the Interrupt counter start No. is set to C300, the range of C300 to C427 is used as the interrupt counter.  C300 I0 C301 I1 C302 I2 C427 I127 Interrupt counter (128 points) Corresponding interrupt pointer	256 points, starting from the set counter number, are used as the interrupt counter. When the Interrupt counter start No. is set to C300, the range of C300 to C555 is used as the interrupt counter.  C300 I0 C301 I1 C302 I2 C555 I255 Interrupt counter (256 points) Corresponding interrupt pointer

To use the interrupt counter, enable the interrupt by the EI instruction in the main routine program.

(5) Precautions

(a) Interrupt counter and interrupt program execution

One interrupt pointer cannot be used for both interrupt counting and interrupt program execution.
Once the interrupt counter is set in the PLC system tab of the PLC parameter dialog box, interrupt programs are not executable.

(b) 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.

(c) Maximum counting speed of the interrupt counter

The maximum counting speed is determined by one of the longest processing time among the following:

- Processing time for the instruction that needs the longest time in the program
- Processing time for an interrupt program
- Processing time for a fixed scan execution type program

(d) When too many points are used for the interrupt counter

The processing time of the sequence program will increase, and "WDT ERROR" may occur.
If this occurs, reduce the number of interrupt counter points or slow down the input pulse signal counting speed.

(e) Resetting the interrupt counter

Use the RST C□ instruction in the main routine program.

(f) Reading the count value

The interrupt counter's count value can be read out with the MOV instruction in the sequence program.

9.2.12 Data register (D)

(1) Definition

The data register (D) is a memory in which numeric data (-32768 to 32767, or 0000H to FFFFH) can be stored.

(2) 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.

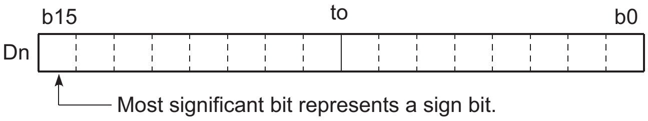


Figure 9.41 Bit structure of the data register

Point

Data register data are handled as signed data.
In the case of the hexadecimal notation, 0000H to FFFFH 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 (Dn and Dn+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.

Example When D12 is specified in the DMOV instruction, D12 represents the lower 16 bits and D13 represents the higher 16 bits.

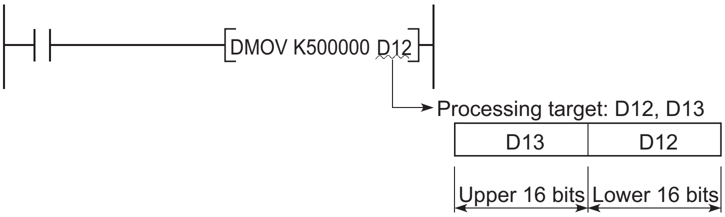


Figure 9.42 Data transfer with a 32-bit instruction and storage location

Data of -2147483648 to 2147483647 or 0H to FFFFFFFFH 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.)

(3) 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.

9.2.13 Link register (W)

(1) Definition

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.

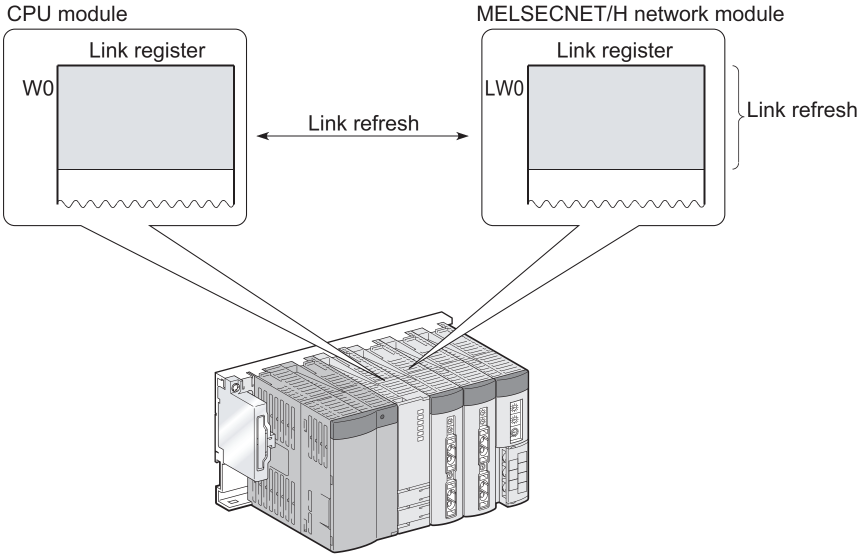


Figure 9.43 Link refresh

In the link register, numeric data (-32768 to 32767, or 0000H to FFFFH) are stored.

(2) 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.

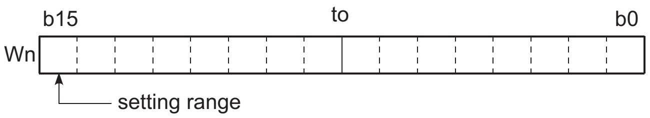


Figure 9.44 Bit structure of the link register

Point

Link register data are handled as signed data. In the case of the hexadecimal notation, 0000H to FFFFH 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.

Example When W12 is specified in the DMOV instruction, W12 represents the lower 16 bits and D13 represents the higher 16 bits.

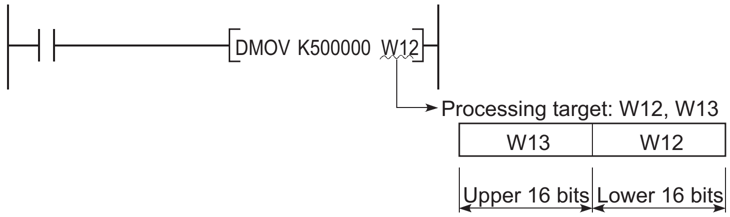


Figure 9.45 Data transfer with a 32-bit instruction and storage location

Data of -2147483648 to 2147483647 or 0H to FFFFFFFFH 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.)

(3) 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

- Although the number of points for the link register in a CC-Link IE Controller Network module is 131072 points, the default value for the link register in the CPU module is 8192 points.
- Although the number of points for the link register of a MELSECNET/H network module is 16384 points, the default value for the link register in the CPU module is 8192 points.

To use the link register exceeding the above points, change the link register points in the Device tab of the PLC parameter dialog box in GX Developer or use a file register.

(4) 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.

Remark

For the network parameters, refer to the following.

☞ Manual for each network module

9.2.14 Link special register (SW)

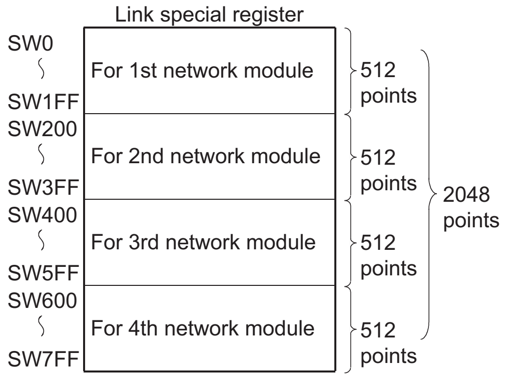
(1) Definition

The link special register (SW) is used to store communication status data and error data of intelligent function modules, such as CC-Link IE Controller Network modules and MELSECNET/H network modules.
Because the data link information is stored as numeric data, error locations and causes can be checked by monitoring the link special register.

(2) Number of link special register points

The points for the link special register depends on the CPU module.

Table9.6 Link special register points for each CPU module

CPU module	Link special register points
Basic model QCPU	The points for the link special register in the CPU module is 1024 (SW0 to SW3FF). To an intelligent function module that has a link special register, such as a CC-Link IE Controller Network module or MELSECNET/H network module, 512 points are assigned.
High Performance model QCPU, Process CPU, and Redundant CPU	The points for the link special register in the CPU module is 2048 (SW0 to SW7FF). To an intelligent function module that has a link special register, such as a CC-Link IE Controller Network module or MELSECNET/H network module, 512 points are assigned. The link special register can be allocated as shown below.  Link special register SW0 SW1FF For 1st network module 512 points SW200 SW3FF For 2nd network module 512 points SW400 SW5FF For 3rd network module 512 points SW600 SW7FF For 4th network module 512 points 2048 points

Remark

For details of the link special register, refer to the manual for each intelligent function module that has the link special register.

9.3 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.

9.3.1 Function devices (FX, FY, FD)

(1) Definition

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.

Example 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.

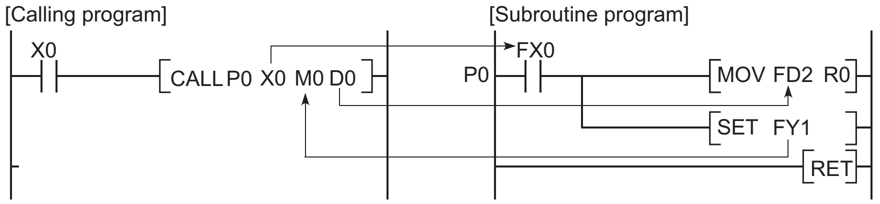


Figure 9.46 Application example of function devices

(2) 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.

(3) 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.

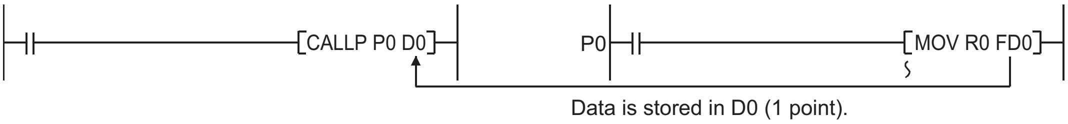


Figure 9.47 When the function register of one point occupies one word

2) A two-word instruction uses two words.

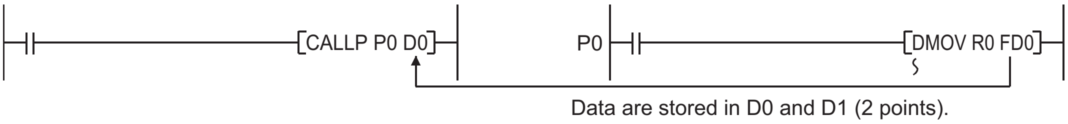


Figure 9.48 When the function register of one point occupies two words

3) At a destination using 32-bit multiplication or division, four words are used.

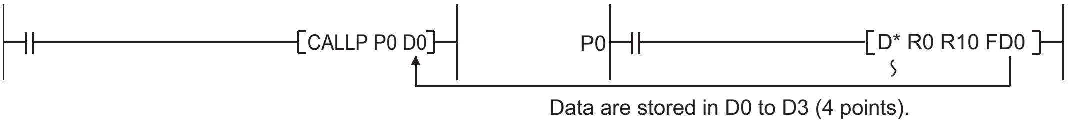
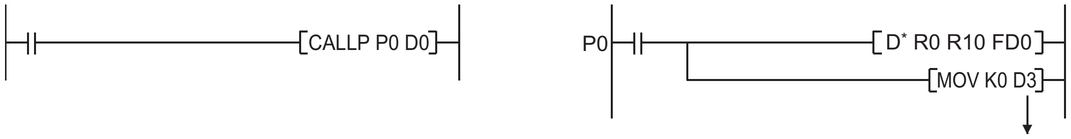


Figure 9.49 When the function register of one point occupies four words

- 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.

Figure 9.50 Ladder example in which use of devices is not allowed in a subroutine program with argument passing

Remark

For use of function devices, refer to the following.

MELSEC-Q/L Programming Manual (Common Instruction)

9.3.2 Special relay (SM)

(1) Definition

The special relay (SM) is an internal relay of which details are specified inside the programmable controller, and the CPU module status data are stored in this special relay.

(2) Special relay classifications

Table9.7 shows special relay classifications.

Table9.7 Special relay classification list

Classification	Special relay	CPU module			
		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
Diagnostics information	SM0 to SM99	○	○	○	○
	SM100 to SM199	×	○	○	○
Serial communication function	SM100 to SM129	○	×	×	×
System information	SM200 to SM399	○	○	○	○
System clocks and counters	SM400 to SM499	○	○	○	○
Scan information, I/O refresh	SM500 to SM599	○	○	○	○
Drive information	SM600 to SM699	○	○	○	○
Instruction related	SM700 to SM799	○	○	○	○
Debugging	SM800 to SM899	×	○	○	○
Latch area	SM900 to SM999	×	○	○	○
A → QnA conversion compatibility	SM1000 to SM1255[*1]	×	○	○	×
Process control instruction	SM1500 to SM1509	×	×	○	○
Redundant system (host system CPU information)	SM1510 to SM1599	×	×	×	○
Redundant system (other system CPU information)	SM1600 to SM1699	×	×	×	○
Redundant system (tracking information)	SM1700 to SM1779	×	×	×	○
Redundant power supply module information	SM1780 to SM1799	×	○	○	○

○:Usable special relay exists,×:Usable special relay does not exist

[*1] Valid only when "Use special relay/special register from SM/SD1000" is selected in the PLC system tab of the PLC parameter dialog box.

Remark

For details of the special relay, refer to the following.

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

9.3.3 Special register (SD)

(1) Definition

The special register (SD) is an internal relay of which details are specified inside the programmable controller, and the CPU module status data (such as error diagnostics or system information) are stored in this special register.

(2) Special register classifications

Table9.8 shows special register classifications.

Table9.8 Special register classification list

Classification	Special register	CPU module			
		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU
Diagnostics information	SD0 to SD99	○	○	○	○
	SD100 to SD199	×	○	○	○
Serial communication function	SD100 to SD129	○	×	×	×
Fuse blown module	SD130 to SD149	○	×	×	×
	SD1300 to SD1399	×	○	○	○
Check of I/O modules	SD150 to SD199	○	×	×	×
	SD1400 to SD1499	×	○	○	○
System information	SD200 to SD399	○	○	○	○
System clocks and counters	SD400 to SD499	○	○	○	○
Scan information	SD500 to SD599	○	○	○	○
Drive information	SD600 to SD699	○	○	○	○
Instruction related	SD700 to SD799	○	○	○	○
Debugging	SD800 to SD899	×	○	×	×
Latch area	SD900 to SD929	×	○	×	×
Redundant CPU information (power backup information)	SD952	×	×	×	○
A → QnA conversion compatibility	SD1000 to SD1255[1]	×	○	○	×
Process control instruction	SD1500 to SD1509	×	×	○	○
Redundant system (host system CPU information)	SD1510 to SD1599	×	×	×	○
Redundant system (other system CPU information)	SD1600 to SD1699	×	×	×	○
Redundant system (tracking information)	SD1700 to SD1779	×	×	×	○
Redundant power supply module information	SD1780 to SD1799	×	○	○	○

○:Usable special register exists,×:Usable special register does not exist

[1] Valid only when "Use special relay/special register from SM/SD1000" is selected in the PLC system tab of the PLC parameter dialog box.

Remark

For details of the special register, refer to the following.

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

9.3 Internal System Devices
9.3.3 Special register (SD)

9.4 Link Direct Device

(1) Definition

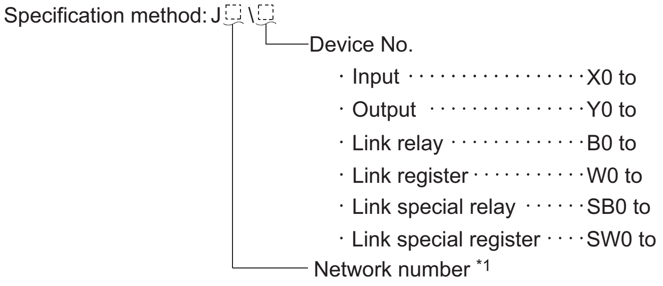
The link direct device is a device for direct access to the link device in a CC-Link IE Controller Network module or MELSECNET/H network module.

The CPU module can directly write data to or read data from the link device in a CC-Link IE Controller Network module or MELSECNET/H network module using sequence programs regardless of link refresh.

(2) Specification method and application example

(a) Specification method

Specify a network number and a device number.



Specification method: J□\□
Device No.
· Input ·················· X0 to
· Output ················ Y0 to
· Link relay ············· B0 to
· Link register ··········· W0 to
· Link special relay ······ SB0 to
· Link special register ···· SW0 to
Network number *1

Figure 9.51 Specification method

*1: Basic model QCPU ••• No.1 to No.239
High Performance model QCPU, Process CPU, Redundant CPU ••• No.1 to No.255

(b) Application example

For link register 10 (W10) of network number 2, "J2\W10" must be used.

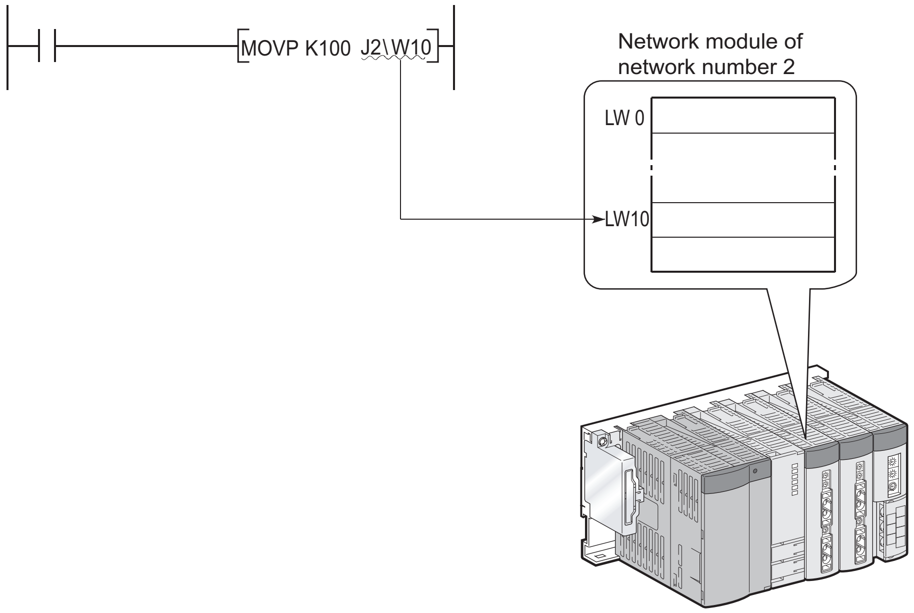


Figure 9.52 Application example

For a bit device (X, Y, B, or SB), the digit must be specified.

Example J1\K1X0,J10\K4B0

(3) 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.

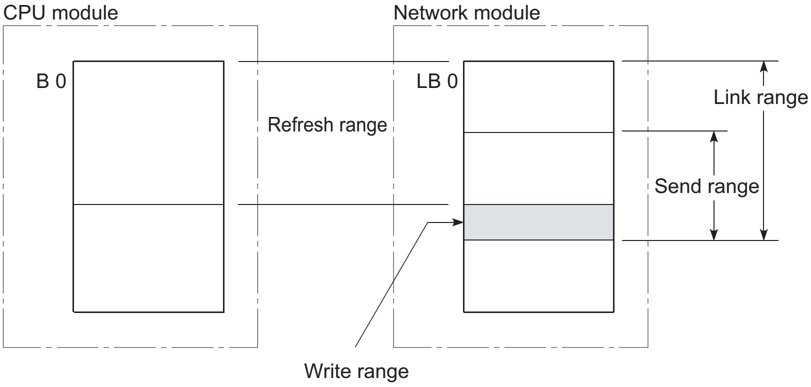


Figure 9.53 Write range of a link direct device

- 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)

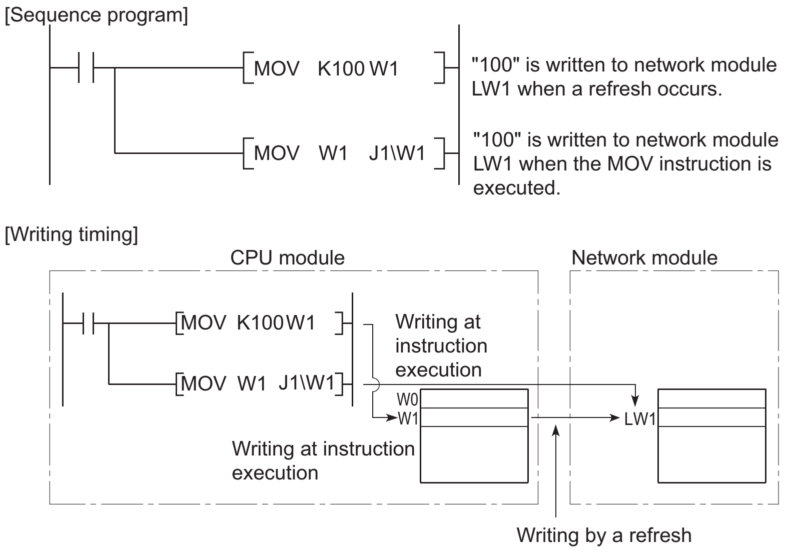

[Sequence program]
MOV K100 W1
"100" is written to network module LW1 when a refresh occurs.
MOV W1 J1\W1
"100" is written to network module LW1 when the MOV instruction is executed.
[Writing timing]
CPU module
Network module
MOV K100 W1
MOV W1 J1\W1
Writing at instruction execution
W0
W1
LW1
Writing at instruction execution
Writing by a refresh

Figure 9.54 Writing to the link device set within the refresh range

- 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.

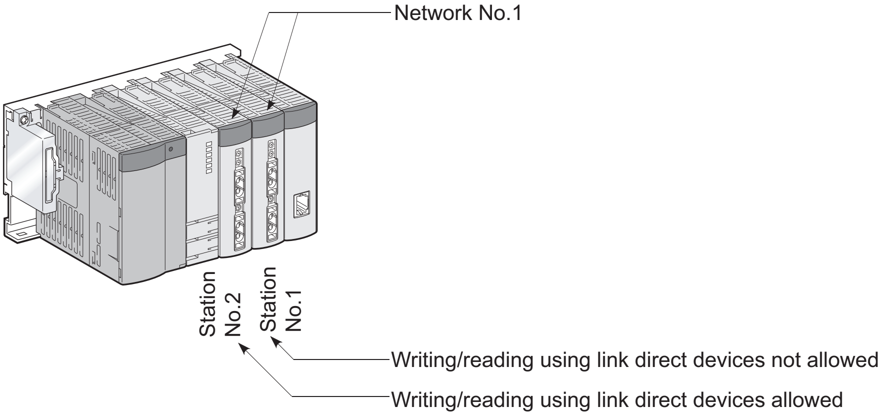
(b) Reading

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. Note9.4

For example, if network modules set as station numbers 1 and 2 are mounted on network number 1 as shown in Figure 9.55, station number 2 is the target of the link direct device.



Network No.1
Station No.2
Station No.1
Writing/reading using link direct devices not allowed
Writing/reading using link direct devices allowed

Figure 9.55 When two or more network module are mounted on the same network

Note9.4 **Basic**

A Basic model QCPU and more than one network module cannot be mounted with the same network number.

(4) Differences between link direct devices and link refresh

Table9.9 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		Word (16-bit) units	

Remark

For network parameters, common parameters, and network refresh parameters, refer to the following.

- Details:

  ☞ Network manual for each network module
- Setting method:

  ☞ GX Developer Version 8 Operating Manual

9.5 Module Access Devices

9.5.1 Intelligent function module device

(1) Definition

The intelligent function module device allows direct access from the CPU module to the buffer memories of the intelligent function modules which are mounted on the main and extension base units.

(2) Specification method and application example

(a) Specification method

Specify the I/O number and buffer memory address of the intelligent function module.

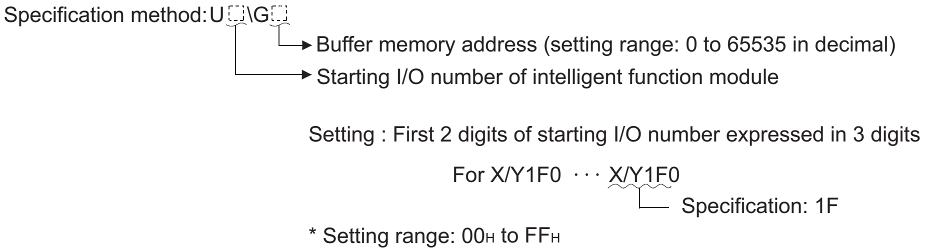

Specification method:U□\G□
Buffer memory address (setting range: 0 to 65535 in decimal)
Starting I/O number of intelligent function module
Setting : First 2 digits of starting I/O number expressed in 3 digits
For X/Y1F0 · · · X/Y1F0
Specification: 1F
* Setting range: 00H to FFH

Figure 9.56 Specification method

(b) Application example

When the Q64AD analog-digital converter module is mounted in the position of I/O number 020 (X/Y020 to X/Y02F), to store digital output values of CH.1 to CH.4 into D0 to D3 accordingly, specify the device as shown in Figure 9.57.

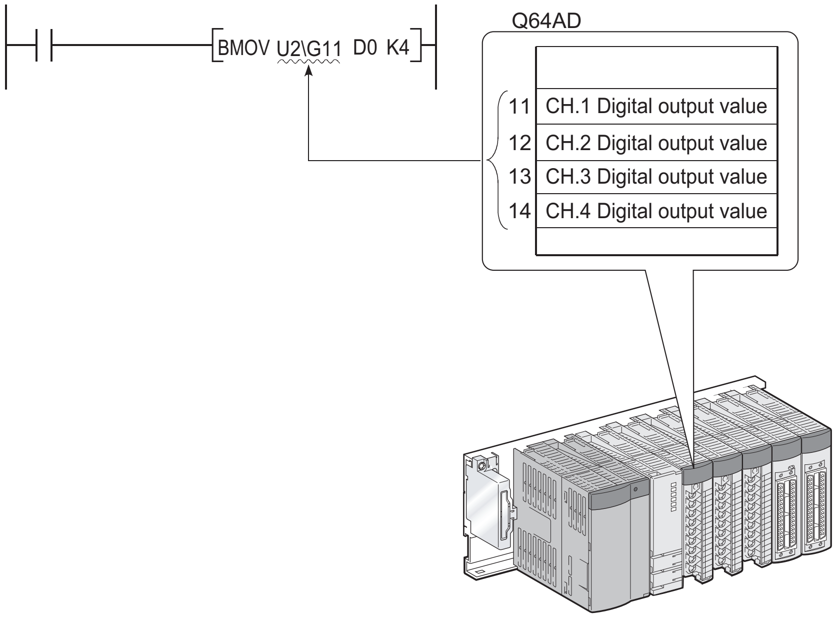

BMOV U2\G11 D0 K4
Q64AD
11 CH.1 Digital output value
12 CH.2 Digital output value
13 CH.3 Digital output value
14 CH.4 Digital output value

Figure 9.57 Application example

Point

If the intelligent function module device is used, device comments can be attached to the buffer memory.

☞ GX Developer Version 8 Operating Manual

(3) 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.

  Example "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.

  Note9.5

  Example "+ U2/G11 D0 D10"

Note9.5 Basic Process Redundant

For the Basic model QCPU, Process CPU, and Redundant CPU, AnS and A series special function modules cannot be used.

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.

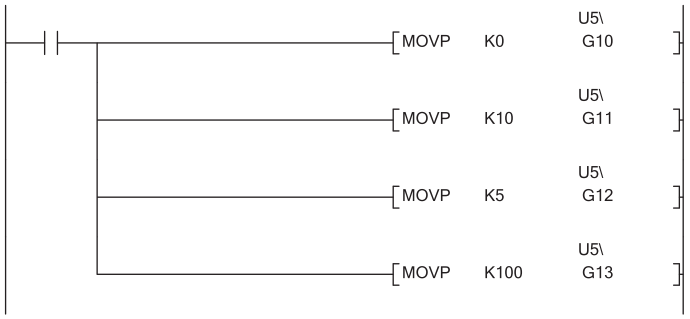


Figure 9.58 Writing data using the intelligent function module device multiple times

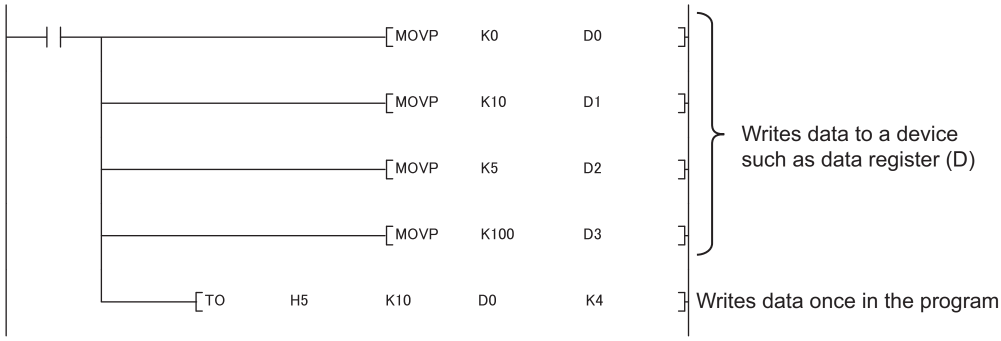

MOVP K0 D0
MOVP K10 D1
MOVP K5 D2
MOVP K100 D3
Writes data to a device such as data register (D)
TO H5 K10 D0 K4
Writes data once in the program

Figure 9.59 Writing data using the TO instruction once in the program

Remark

1) For buffer memory addresses and applications, refer to the manual for each intelligent function module/special function module.

2) For the FROM and TO instructions, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

9.5.2 Cyclic transmission area device Note9.6

(1) Definition

The cyclic transmission area device is used to access the CPU shared memory of each CPU module in a multiple CPU system.

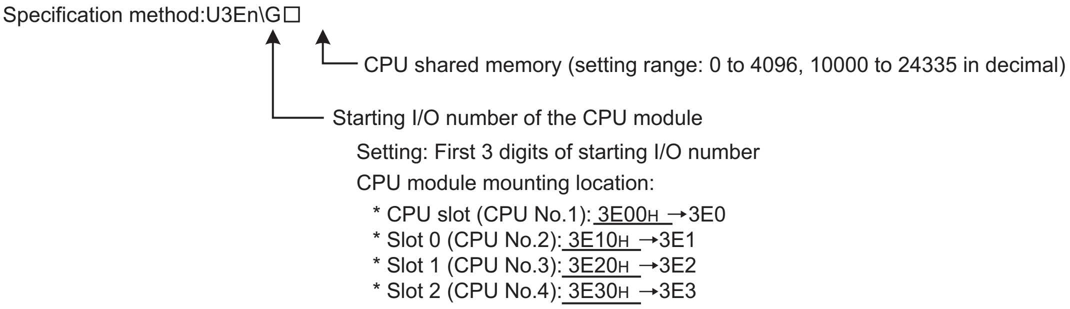
(2) Features

- 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 programing steps. *1
- 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.

*1:In the High Performance model QCPU and Process CPU, buffer memory data cannot be written to the CPU shared memory in host CPU using the cyclic transmission area device (U3En\G□).

(3) Specification method

Specify the I/O number of the CPU module and the CPU shared memory address.



Specification method:U3En\G□

CPU shared memory (setting range: 0 to 4096, 10000 to 24335 in decimal)

Starting I/O number of the CPU module

Setting: First 3 digits of starting I/O number

CPU module mounting location:

* CPU slot (CPU No.1): 3E00H →3E0
* Slot 0 (CPU No.2): 3E10H →3E1
* Slot 1 (CPU No.3): 3E20H →3E2
* Slot 2 (CPU No.4): 3E30H →3E3

Figure 9.60 Specification method

Remark

For details of the cyclic transmission area device, refer to the following.

☞ QCPU User's Manual (Multiple CPU System)

Note9.6 Redundant

For Redundant CPUs, the cyclic transmission area device cannot be used.

9.6 Index Register (Z)

9.6.1 Index register (Z)

(1) Definition

The index register is used for indirect specification (index modification) in sequence programs.
Index modification uses one point of the index register.

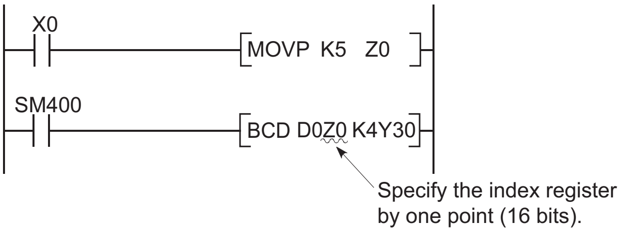


Figure 9.61 Index register

The points available for the index register are as follows:

- Basic model QCPU: Z0 to Z9 (10 points)
- High Performance model QCPU, Process CPU, Redundant CPU: Z0 to Z15 (16 points)

(2) 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.

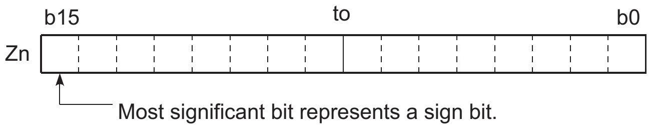


Figure 9.62 Bit structure of the index register

Point

Index register data are handled as signed data.
In the case of the hexadecimal notation, 0000H to FFFFH 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 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.

Example 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.)

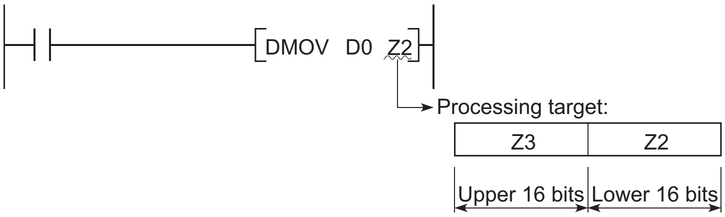


Figure 9.63 Data transfer with a 32-bit instruction and storage location

Remark

For details and precautions of index modification using the index register, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

9.6.2 Switching between the scan execution type and low-speed execution type programs Note9.7

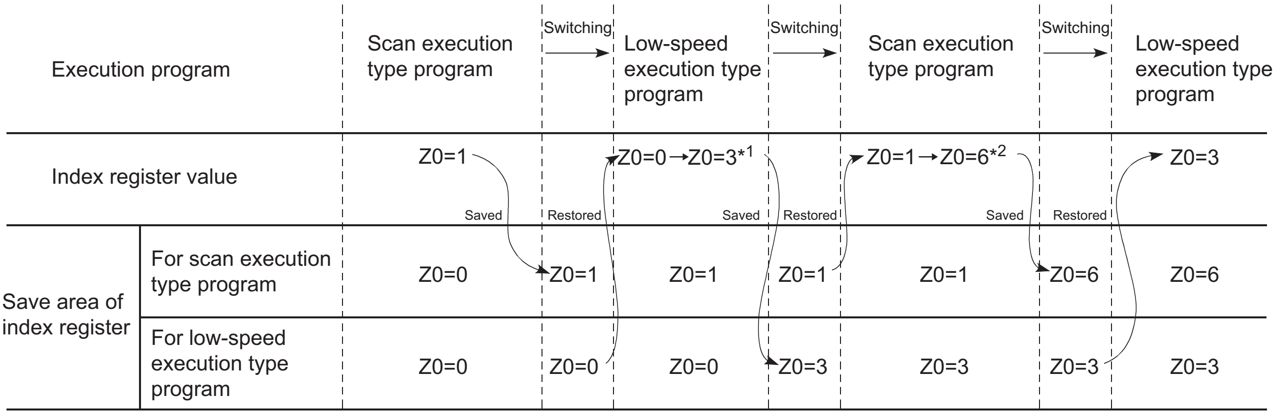
The CPU module saves and restores the index register data when switching between the scan execution type program and the low-speed execution type program.

(1) When switching from the scan execution type program to the low-speed execution type program

The CPU module saves index register values in the scan execution type program, and restores index register values in the low-speed execution type program.

(2) When switching from the low-speed execution type program to the scan execution type program

The CPU module saves index register values in the low-speed execution type program, and restores index register values in the scan execution type program.



Execution program		Scan execution type program	Switching →	Low-speed execution type program	Switching →	Scan execution type program	Switching →	Low-speed execution type program
Index register value		Z0=1		Z0=0 → Z0=3*1		Z0=1 → Z0=6*2		Z0=3
Save area of index register	For scan execution type program	Z0=0	Z0=1	Z0=1	Z0=1	Z0=1	Z0=6	Z0=6
	For low-speed execution type program	Z0=0	Z0=0	Z0=0	Z0=3	Z0=3	Z0=3	Z0=3

Figure 9.64 Saving and restoring index register data when switching the scan/low-speed execution type program

*1: The Z0 value is changed to 3 in the low-speed execution type program.
*2: The Z0 value is changed to 6 in the scan execution type program.

(3) Passing of index register data

To pass index register data between the scan execution type program and the low-speed execution type program, use word devices.

Note9.7 Basic Redundant

Since the Basic model QCPU and Redundant CPU cannot use the low-speed execution type program, switching to the low-speed execution type program is impossible.

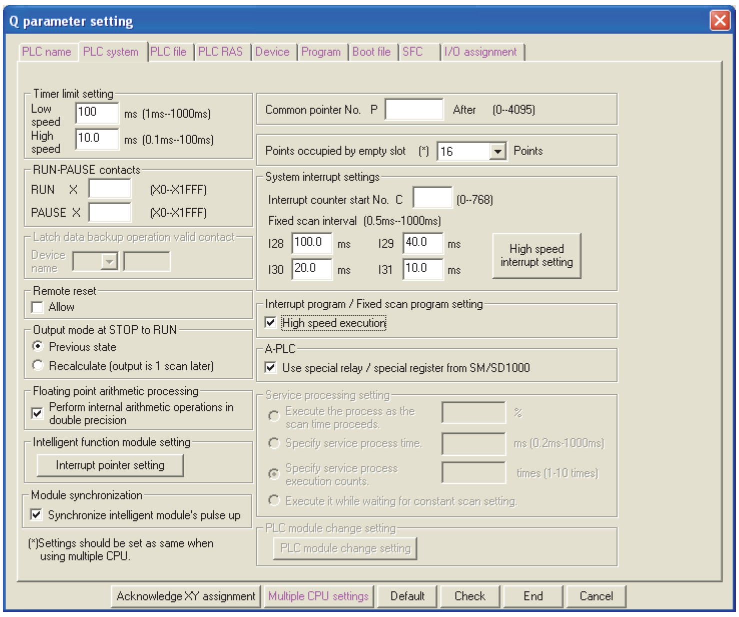
9.6.3 Switching from the scan execution type/low-speed execution type program to the interrupt/fixed scan execution type program Note9.8

The CPU module performs the following when switching from the scan execution type program or low-speed execution type program to the interrupt or 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.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
□ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms

Interrupt program / Fixed scan program setting
☑ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting

Figure 9.65 Interrupt/fixed scan program setting

Note9.8 Basic Redundant

For the Basic model QCPU, the low-speed execution type and fixed scan execution type programs cannot be used.
Therefore, interpret "scan execution type/low-speed execution type program" as "main/subroutine program", and "interrupt/fixed scan execution type program" as "interrupt program".
For the Redundant CPU, the low-speed execution type program cannot be used.
Interpret "scan execution type/low-speed execution type program" as "scan execution type program".

(2) Processing of the index register

(a) When "High-speed execution" is not selected

1) 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.

2) 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.

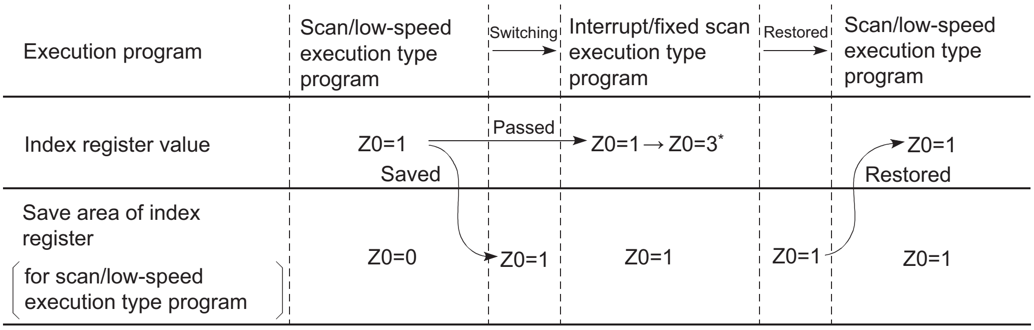

Execution program
Scan/low-speed execution type program
Switching
Interrupt/fixed scan execution type program
Restored
Scan/low-speed execution type program
Index register value
Z0=1
Passed
Z0=1 → Z0=3*
Z0=1
Saved
Restored
Save area of index register
(for scan/low-speed execution type program)
Z0=0
Z0=1
Z0=1
Z0=1
Z0=1

Figure 9.66 Saving and restoring index register data (when "High speed execution" is not selected)

*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.

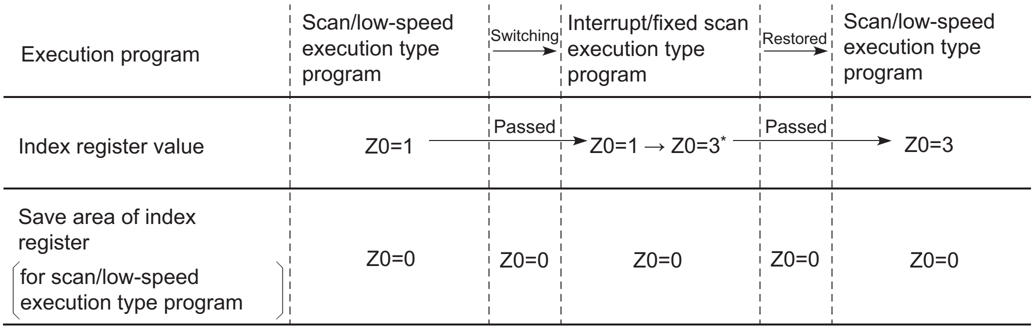
(b) When "High-speed execution" is selected

1) When switching from the scan execution type/low-speed execution type program to the interrupt/fixed scan execution type program

The CPU module does not save/restore any index register values.

2) When switching from the interrupt/fixed scan execution type program to the scan execution type/low-speed 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/low-speed execution type program will be corrupted.

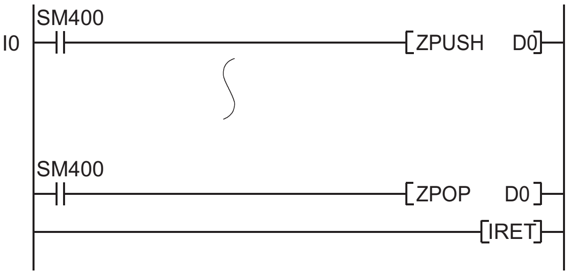


Execution program	Scan/low-speed execution type program	Switching →	Interrupt/fixed scan execution type program	Restored →	Scan/low-speed execution type program
Index register value	Z0=1	Passed →	Z0=1 → Z0=3*	Passed →	Z0=3
Save area of index register (for scan/low-speed execution type program)	Z0=0	Z0=0	Z0=0	Z0=0	Z0=0

Figure 9.67 Saving and restoring index register data (when "High speed execution" is selected)

*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.



Data in Z0 to Z15 are stored in the area addressed from D0.

Data in the area addressed from D0 are stored in Z0 to Z15.

Figure 9.68 Writing data to the index register in the interrupt/fixed scan execution type program

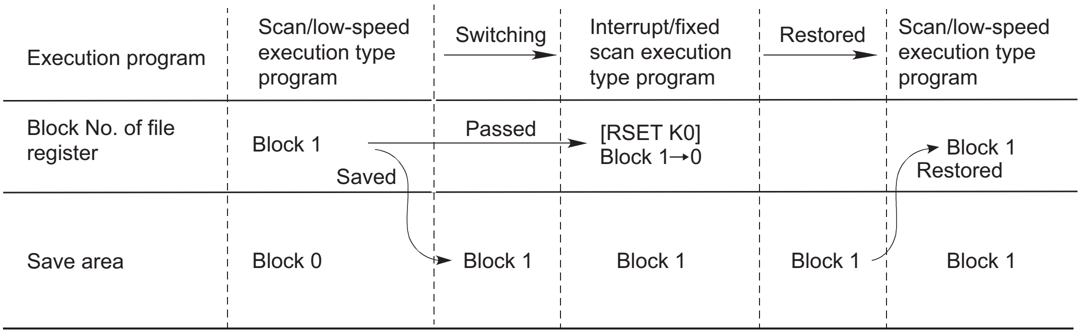
(3) Processing of file register's block numbers

(a) When switching from the scan execution type/low-speed 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/low-speed execution type program, and passes them to the interrupt/fixed scan execution type program.

(b) When switching from the interrupt/fixed scan execution type program to the scan execution type/low-speed execution type program

The CPU module restores the saved block numbers of the file register.



Execution program	Scan/low-speed execution type program	Switching →	Interrupt/fixed scan execution type program	Restored →	Scan/low-speed execution type program
Block No. of file register	Block 1	Passed Saved	[RSET K0] Block 1→0		Block 1 Restored
Save area	Block 0	Block 1	Block 1	Block 1	Block 1

Figure 9.69 Saving and restoring file register's block numbers

9.7 File Register (R) Note9.9

(1) Definition

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.

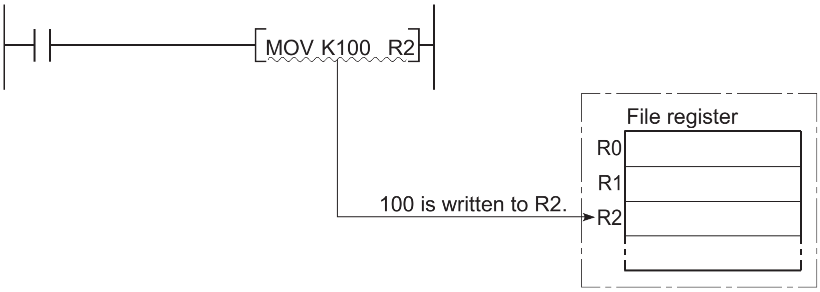


Figure 9.70 Writing to the file register

(2) Bit structure od 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.

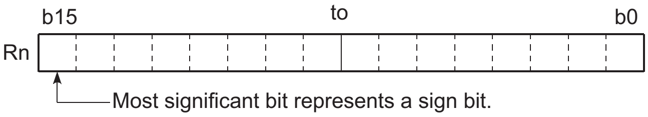


Figure 9.71 Bit structure of the file register

(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.

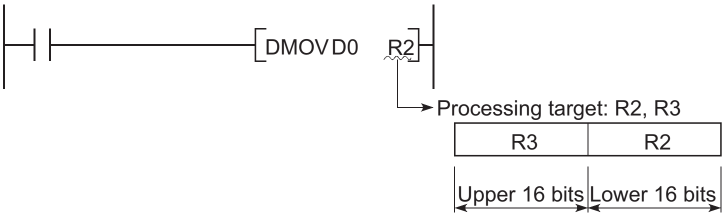


Figure 9.72 Data transfer with a 32-bit instruction and storage location

Data of -2147483648 to 2147483647 or 0H to FFFFFFFFH 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.)

Note9.9 Basic

The Q00JCPU does not have the file register.

(3) Clearing the file register

The data in the file register is held even if the CPU module is powered off or is reset.
(File register data cannot be cleared by latch clear.)

To clear file register data, perform data clear by sequence program or GX Developer.

(a) When clearing by the sequence program

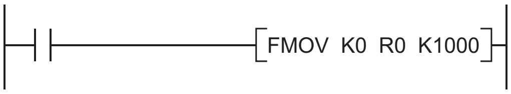


Figure 9.73 Example of clearing the file register R0 to R999

(b) When clearing by GX Developer

Select [Online] → [Clear PLC memory] in GX Developer and clear the data by selecting "Clear all file registers".

9.7.1 File register data storage location

The file register data are stored in the following memories.

- Basic model QCPU:
  Standard RAM
- High Performance model QCPU, Process CPU, and Redundant CPU:
  Standard RAM, SRAM card, or Flash card

9.7.2 File register size

(1) When using the standard RAM

The table below shows the maximum points of file register data that can be stored in the standard RAM.
Note that, however, if the standard RAM is used for an application other than file registers, available points are decreased. (Section 5.1.1,Section 5.2.1)

Table9.10 File register size

CPU module		Points
Basic model QCPU	Q00CPU, Q01CPU	64K
High Performance model QCPU [*1]	Q02CPU	32K
	Q02HCPU, Q06HCPU	64K
	Q12HCPU, Q25HCPU	128K
Process CPU	Q02PHCPU, Q06PHCPU	64K
	Q12PHCPU, Q25PHCPU	128K
Redundant CPU	Q12PRHCPU, Q25PRHCPU	128K

*1: Since the standard RAM size varies by CPU module version, the number of available points is different. (Appendix 2.2)

(2) When using an SRAM card

Up to 1018K points can be stored in one file.
Since one block consists of 32K words, up to 32 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) When using a Flash card

Up to 1018K points can be stored in one file.
Since one block consists of 32K words, up to 32 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 Section 5.2.1

9.7.3 Differences in available accesses by storage memory

Accesses available for the file register vary for each memory.

Table9.11 Differences in accesses available for the file register

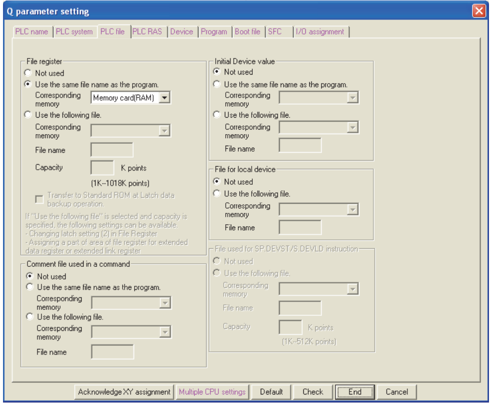
Access		Standard RAM	SRAM card	Flash card
Program writing		○	○	×
Program reading		○	○	○
Writing device memory to programmable controller		○	○	×
Reading device memory from programmable controller		○	○	○
Data modification	Online test operation from GX Developer	○	○	×
	Writing to programmable controller by GX Developer	○	○	×
	Writing to programmable controller by GX Developer (flash ROM)	×	×	○
	Batch write by a serial communication module	○	○	×
	Device data writing from GOT1000 series	○	○	×
	Random write command from GOT1000 series	○	○	×

9.7.4 Registration procedure for the file register Note9.10

To use a file register, register the file of the file register to the CPU module in the following steps.

Start

Setting a file register file



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment

File register
○ Not used
● Use the same file name as the program.
Corresponding memory: Memory card(RAM)
○ Use the following file.
Corresponding memory:
File name:
Capacity: K points (1K--1018K points)
□ Transfer to Standard ROM at Latch data backup operation.
If "Use the following file" is selected and capacity is specified, the following settings can be available.
- Changing latch setting (2) in File Register
- Assigning a part of area of file register for extended data register or extended link register

Comment file used in a command
● Not used
○ Use the same file name as the program.
Corresponding memory:
○ Use the following file.
Corresponding memory:
File name:

Initial Device value
● Not used
○ Use the same file name as the program.
Corresponding memory:
○ Use the following file.
Corresponding memory:
File name:

File for local device
● Not used
○ Use the following file.
Corresponding memory:
File name:

File used for SP.DEVST/S.DEVLD instruction
○ Not used
○ Use the following file.
Corresponding memory:
File name:
Capacity: K points (1K--512K points)

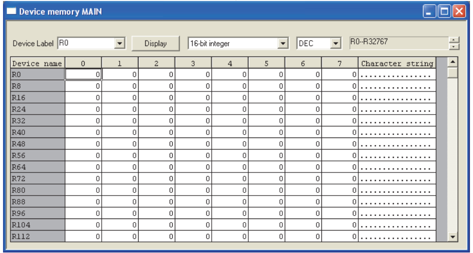
Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

······ PLC file tab of the PLC parameter dialog box

When "Use the following file." is selected

When "Not used" or "Use the same file names as the program." is selected

File register setting



Device memory MAIN

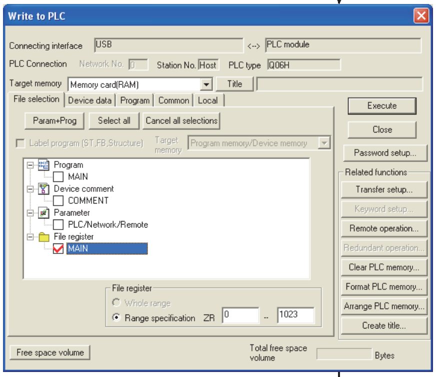
Device Label: R0 | Display | 16-bit integer | DEC | R0--R32767

Device name	0	1	2	3	4	5	6	7	Character string
R0	0	0	0	0	0	0	0	0	....
R8	0	0	0	0	0	0	0	0	....
R16	0	0	0	0	0	0	0	0	....
R24	0	0	0	0	0	0	0	0	....
R32	0	0	0	0	0	0	0	0	....
R40	0	0	0	0	0	0	0	0	....
R48	0	0	0	0	0	0	0	0	....
R56	0	0	0	0	0	0	0	0	....
R64	0	0	0	0	0	0	0	0	....
R72	0	0	0	0	0	0	0	0	....
R80	0	0	0	0	0	0	0	0	....
R88	0	0	0	0	0	0	0	0	....
R96	0	0	0	0	0	0	0	0	....
R104	0	0	0	0	0	0	0	0	....
R112	0	0	0	0	0	0	0	0	....

······ Device memory screen

Writing a file register file

When the file is stored in the standard RAM or SRAM card:
Write to PLC screen



Write to PLC

Connecting interface: USB <--> PLC module
PLC Connection: Network No. 0 | Station No. Host | PLC type Q06H
Target memory: Memory card(RAM) | Title

File selection | Device data | Program | Common | Local

Param+Prog | Select all | Cancel all selections

□ Label program (ST,FB,Structure) | Target memory: Program memory/Device memory

- Program
  - □ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - □ PLC/Network/Remote
- File register
  - ☑ MAIN

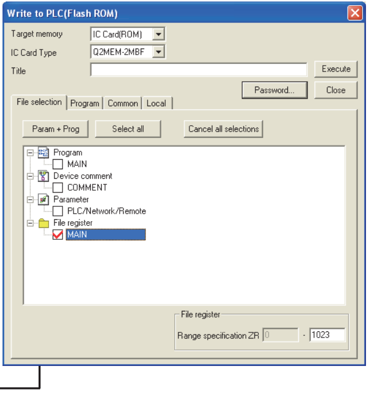
File register
○ Whole range
● Range specification ZR 0 -- 1023

Execute | Close | Password setup...

Related functions: Transfer setup... | Keyword setup... | Remote operation... | Redundant operation... | Clear PLC memory... | Format PLC memory... | Arrange PLC memory... | Create title...

Free space volume | Total free space volume: Bytes

When the file is stored in the Flash card:
Write to PLC (Flash ROM) screen



Write to PLC(Flash ROM)

Target memory: IC Card(ROM)
IC Card Type: Q2MEM-2MBF
Title:

Execute | Password... | Close

File selection | Program | Common | Local

Param + Prog | Select all | Cancel all selections

- Program
  - □ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - □ PLC/Network/Remote
- File register
  - ☑ MAIN

File register
Range specification ZR 0 - 1023

Write the file register file to the CPU module

Write parameters to the CPU module

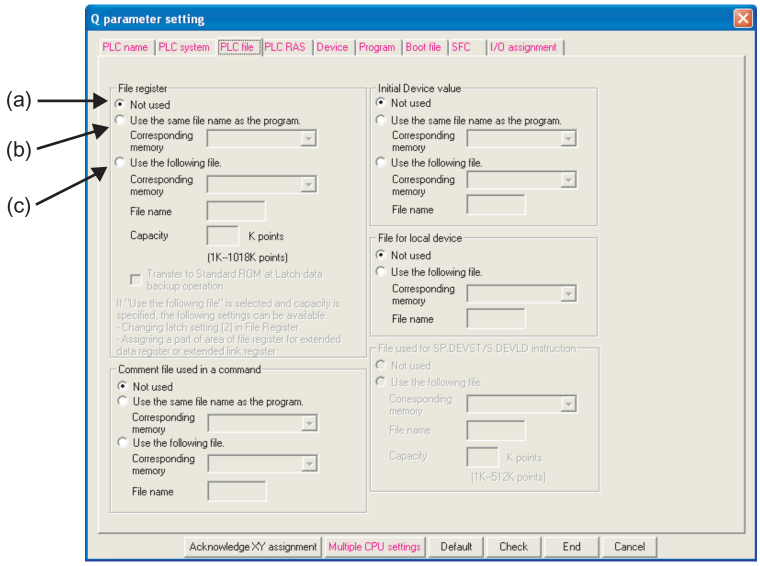
Figure 9.74 Registration procedure for the file register

Note9.10 **Basic**

For the Basic model QCPU, this procedure is not required because the file register is automatically registered to the standard RAM.

(1) Setting the file register

In the PLC file tab of the PLC parameter dialog box, specify the standard RAM or a memory card to use the file register in the sequence program.



File register
● Not used
○ Use the same file name as the program.
Corresponding memory
○ Use the following file.
Corresponding memory
File name
Capacity K points
(1K--1018K points)
□ Transfer to Standard ROM at Latch data backup operation.
If "Use the following file" is selected and capacity is specified, the following settings can be available.
- Changing latch setting (2) in File Register
- Assigning a part of area of file register for extended data register or extended link register

Comment file used in a command
● Not used
○ Use the same file name as the program.
Corresponding memory
○ Use the following file.
Corresponding memory
File name

Initial Device value
● Not used
○ Use the same file name as the program.
Corresponding memory
○ Use the following file.
Corresponding memory
File name

File for local device
● Not used
○ Use the following file.
Corresponding memory
File name

File used for SP.DEVST/S.DEVLD instruction
○ Not used
○ Use the following file.
Corresponding memory
File name
Capacity K points
(1K--512K points)

Figure 9.75 File register setting

(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.

1) 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.

Example 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.

- During execution of program A: Accessing file register A
- During execution of program B: Accessing file register B
- During execution of program C: Accessing file register C

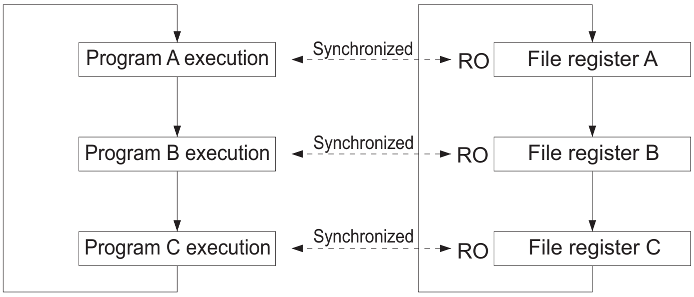


Figure 9.76 When the program is changed

2) Point setting for file registers

Select [Online] → [Write to PLC] in GX Developer and set the number of file register points.

Point

- Only one file register can be created in the standard RAM.
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)

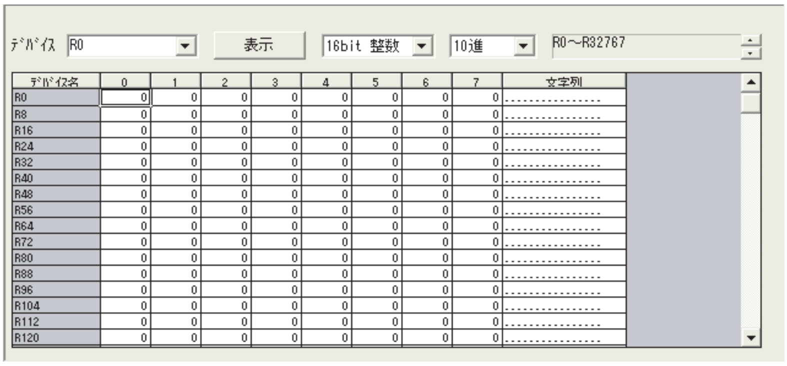
(c) Use the following file.

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).

(2) File register setting

In a new device memory screen, set data for the specified file register.



ﾃﾞﾊﾞｲｽ R0 ▼ 表示 16bit 整数 ▼ 10進 ▼ R0～R32767

ﾃﾞﾊﾞｲｽ名	0	1	2	3	4	5	6	7	文字列
R0	0	0	0	0	0	0	0	0	....
R8	0	0	0	0	0	0	0	0	....
R16	0	0	0	0	0	0	0	0	....
R24	0	0	0	0	0	0	0	0	....
R32	0	0	0	0	0	0	0	0	....
R40	0	0	0	0	0	0	0	0	....
R48	0	0	0	0	0	0	0	0	....
R56	0	0	0	0	0	0	0	0	....
R64	0	0	0	0	0	0	0	0	....
R72	0	0	0	0	0	0	0	0	....
R80	0	0	0	0	0	0	0	0	....
R88	0	0	0	0	0	0	0	0	....
R96	0	0	0	0	0	0	0	0	....
R104	0	0	0	0	0	0	0	0	....
R112	0	0	0	0	0	0	0	0	....
R120	0	0	0	0	0	0	0	0	....

Figure 9.77 Device memory screen

(a) Devices

Setting Rn (R0 in the case shown above) and clicking the Display button will display the file register list.

(b) Data setting

Enter data that are set for the file register.
This step is not needed when you specify only the capacity of file register.

(3) Registering the file register file to the CPU module

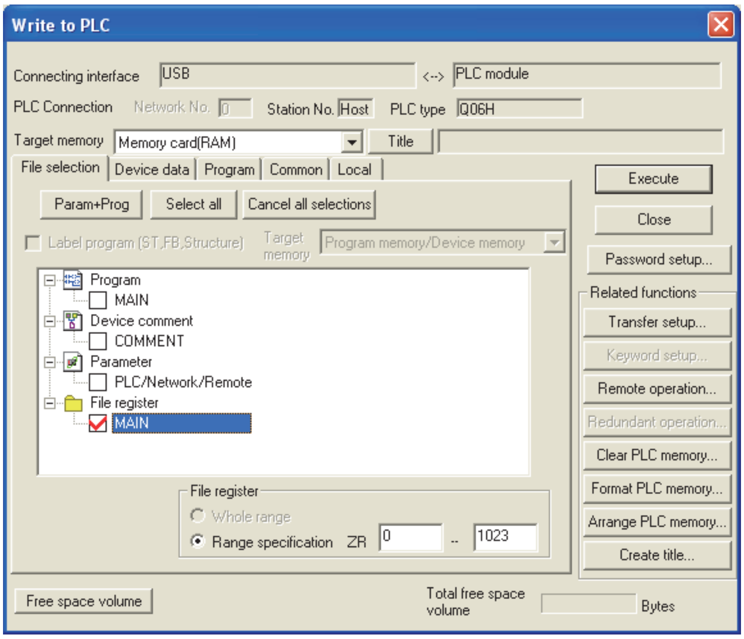
When either of the following is selected in the PLC file tab of the PLC parameter dialog box, the file register must be set (a device memory must be create) before the file register file is registered to the CPU module.

- Not used. (When specifying a file register used in the sequence program) (☞ (1)(a) in this section)
- Use the same name with the program. (☞ (1)(b) in this section)

A file register file can be registered to the CPU module using GX Developer.

(a) When stored in the standard RAM or SRAM card

Select [Online] → [Write to PLC] from the menu of GX Developer and register a file register file on the Write to PLC screen.



Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q06H

Target memory: Memory card(RAM)

- Program
  - □ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - □ PLC/Network/Remote
- File register
  - ☑ MAIN

File register: ○ Whole range ● Range specification ZR 0 -- 1023

Total free space volume: Bytes

Figure 9.78 Write to PLC screen

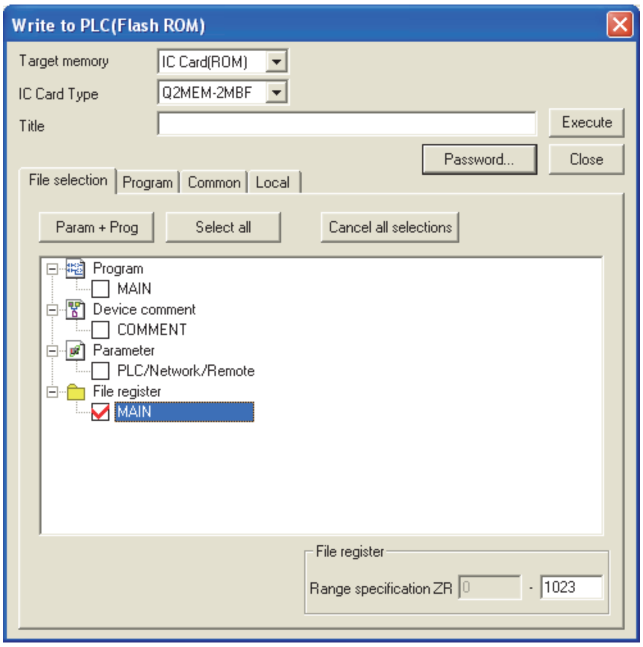
- Select "Standard RAM" or "Memory card (RAM)" in the "Target memory".
- When the "Target memory" is selected, the name of a file register file is displayed.
- Select the file register file.
- When the Execute button is clicked, the file register file is registered to the specified memory in the CPU module for the specified number of points.

Point

The file register size can be set in increments of one point. Note that the file always uses 256 points. Even if the range is not specified from ZR0, the created file will have an assignment from ZR0 to the last number.
For example, when the write range of a file register file is specified to be ZR1000 to ZR1791, the created file will have an assignment from ZR0 to ZR1791. In this case, the data in ZR0 to ZR999 will be undefined. Specify the file register range from ZR0. The file register size is checked in increments of 1K points. Specify the file register size in increments of 1K points starting from R0.

(b) When stored in the Flash card

Select [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] from the menu of GX Developer and register a file register file on the Write to PLC (Flash ROM) screen.



Write to PLC(Flash ROM)

Target memory: IC Card(ROM)

IC Card Type: Q2MEM-2MBF

Title:

File selection

- Program
  - □ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - □ PLC/Network/Remote
- File register
  - ☑ MAIN

File register

Range specification ZR 0 - 1023

Figure 9.79 Write to PLC (Flash ROM) screen

- Select "IC card (ROM)" in the "Target memory".
- When the "Target memory" is selected, the name of a file register file is displayed.
- Select a memory card in the "IC card type".
- Select the file register file.
- When the Execute button is clicked, the file register file is registered to the specified memory in the CPU module for the specified number of points.

9.7.5 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.

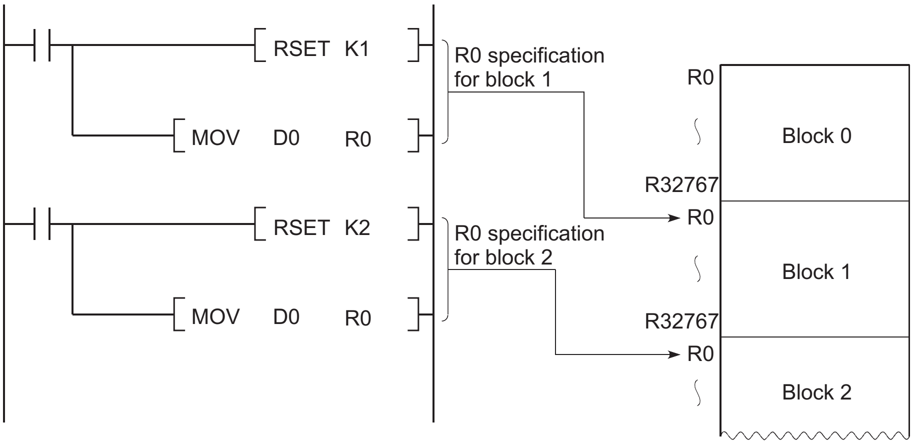


Figure 9.80 Block switching method

(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".

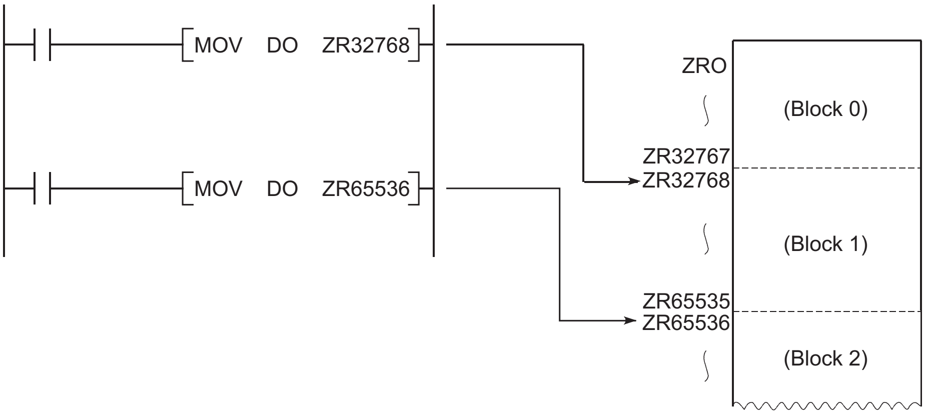


Figure 9.81 Serial number access method

Point

The block numbers and ZR device points that can be specified vary depending on the following.

- Storage location of the file register (Section 9.7.1)
- File register size (Section 9.7.2)

9.7.6 Precautions for using the file register

(1) When using the Basic model QCPU

Even if data are written to or read from the file register area whose number is equal to 64K points or more, no error will occur.

Note that, however, unreliable data may be stored in this kind of reading from the file register.

Point

File register files (MAIN and QDR) cannot be deleted.
However, the contents of the file register can be deleted. (☞ Section 9.7(3))

(2) When using the High Performance model QCPU, Process CPU, or Redundant CPU

(a) No registration or use of an invalid file register number

1) When the file of the file register has not been registered

No error will occur even if data are written to or read from the file register.
However, reading data from the file register will result in the following:

- For the standard RAM, unreliable data are stored.
- For a memory card, "0_H" is stored.

2) When writing to or reading from the file register exceeding the registered size (points)

No error will occur even if data are written to or read from the file register.
However, reading data from the file register will result in the following:

- For the standard RAM, unreliable data are stored.
- For a memory card, "0_H" is stored.

(b) 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.

1) 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).

2) 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), confirm that space of 1K points or more can be ensured after the switching.

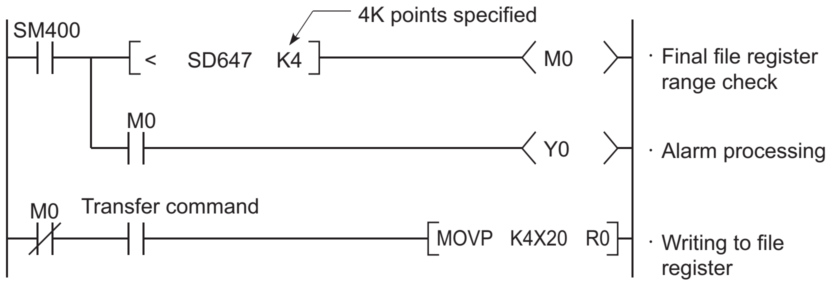
> (File register size) > [32K points × (Switching block No) + 1K points]

3) 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.

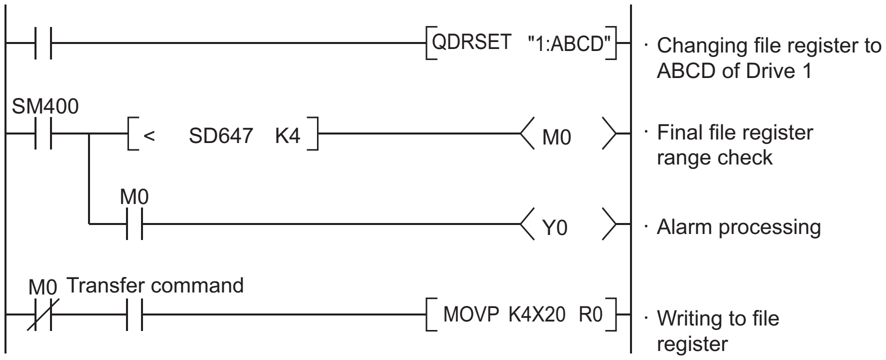
[Program example 1]

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.


QDRSET "1:ABCD"
· Changing file register to ABCD of Drive 1
SM400
< SD647 K4
M0
· Final file register range check
M0
Y0
· Alarm processing
M0
Transfer command
MOVP K4X20 R0
· Writing to file register

[Program example 3]

When a block is switched to another:

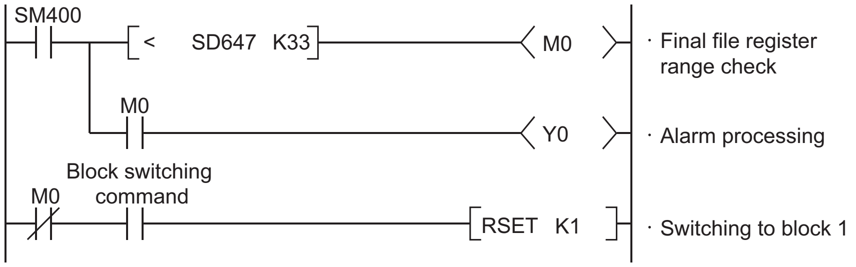


Figure 9.82 Program examples of file register checking

(c) When using the SRAM card which stores file register for 1018K points or more

For the High Performance model QCPU whose serial number (first five digits) is "16021" or later, Process CPU, and Redundant CPU, the file register up to the area of 2M bytes can be used. Even if the file register for 1018K points or more is specified using parameter or instruction, a stop error will not occur.

(3) Deleting a file register file

To delete an unnecessary file register file, select [Online] → [Delete PLC data] in GX Developer.

(4) Differences in the file register processing time among CPU module versions Note9.11

When the serial number (first five digits) of the High Performance model QCPU is "02092" or later, if the file register is specified by the serial number access method (ZR□) using an access instruction to the standard RAM, more processing time will be required per instruction than that of the High Performance model QCPU whose serial number (first five digits) is "02091" or earlier.

- Q02CPU: 1.1 μs (average)
- QnHCPU: 0.65 μs (average)

Table9.12 below shows processing times when the MOV instruction is used.

Table9.12 Differences in processing time between file register specification methods

Instruction	Q12HCPU		Q02CPU	
	"02092" or later	"02091" or earlier	"02092" or later	"02091" or earlier
MOV K0 R0	0.11	0.11	0.26	0.26
MOV K0 ZR0	3.55	2.88	7.71	6.64

(Unit: μs)

Note9.11 Basic Process Redundant

For the Basic model QCPU, Process CPU, and Redundant CPU, there are no differences in the file register processing time among CPU module versions.

9.8 Nesting (N)

(1) Definition

Nesting (N) is a device used in the master control instructions (MC and MCR instructions) to program operation conditions in a nesting structure.

(2) 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.

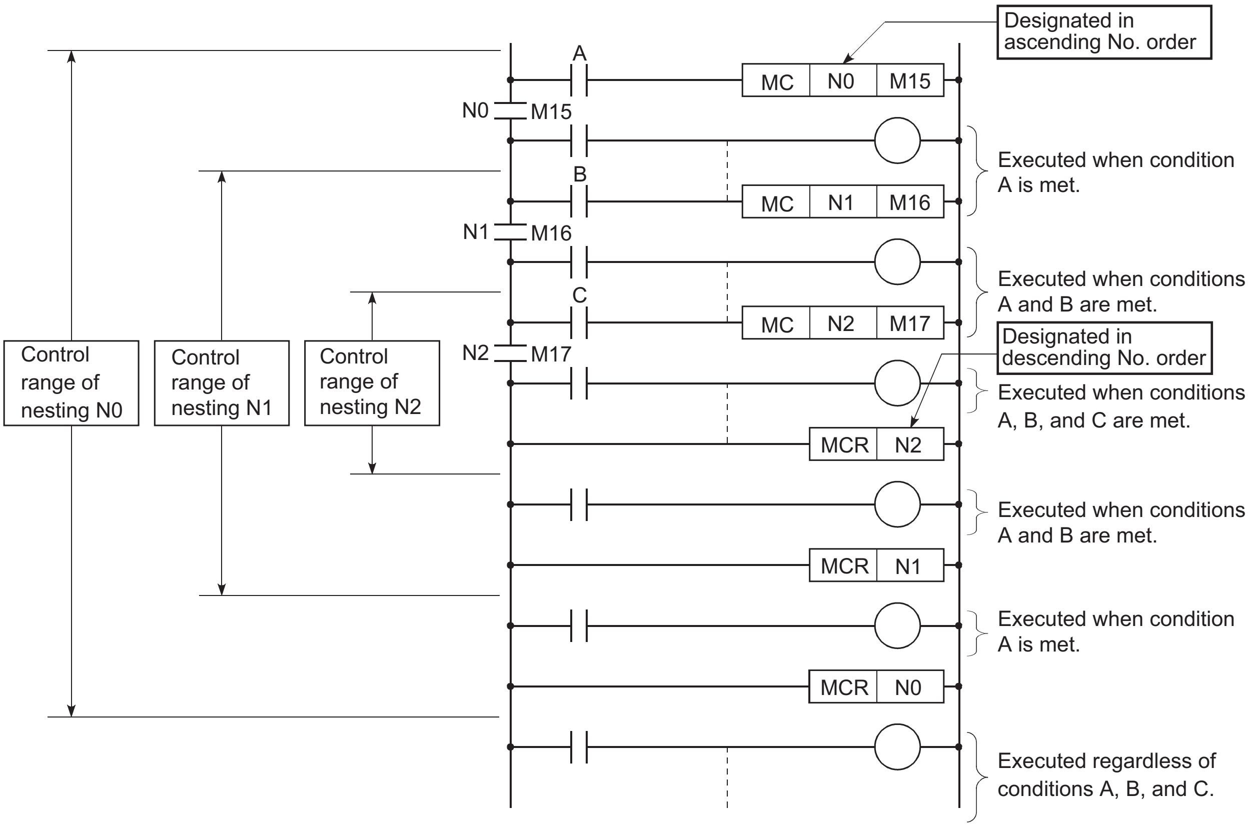

Designated in ascending No. order
A
MC N0 M15
N0 M15
Executed when condition A is met.
B
MC N1 M16
N1 M16
Executed when conditions A and B are met.
C
MC N2 M17
Designated in descending No. order
N2 M17
Control range of nesting N0
Control range of nesting N1
Control range of nesting N2
Executed when conditions A, B, and C are met.
MCR N2
Executed when conditions A and B are met.
MCR N1
Executed when condition A is met.
MCR N0
Executed regardless of conditions A, B, and C.

Figure 9.83 Programming example using the nesting

Remark

For use of the nesting, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

9.9 Pointer (P)

(1) Definition

The pointer (P) is a device used in jump instructions (CJ, SCJ, or JMP) or subroutine call instructions (such as CALL).

(2) 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)

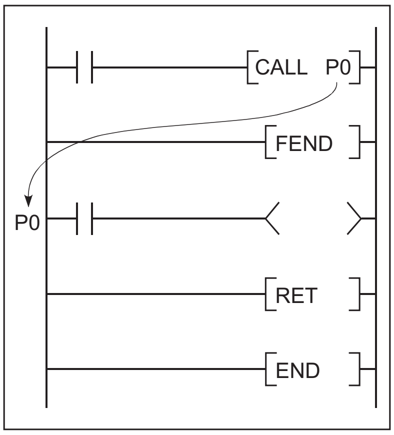


Figure 9.84 Program using a pointer

(3) Pointer types

(a) Basic model QCPU

Since the Basic model QCPU cannot execute multiple programs, it has no distinction between the local and common pointers.

(b) High Performance model QCPU, Process CPU, and Redundant CPU

There are the following two different pointer types.

- Local pointer (☞ Section 9.9.1) :
  The pointer used independently in each program
- Common pointer (☞ Section 9.9.2) :
  The pointer that can be called in all running programs by the subroutine call instruction.

(4) Number of available pointer points

- Basic model QCPU: 300 points
- High Performance model QCPU, Process CPU, and Redundant CPU: 4096 points

Remark

For the jump instructions and subroutine call instructions, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

9.9.1 Local pointer Note9.12

(1) Definition

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.

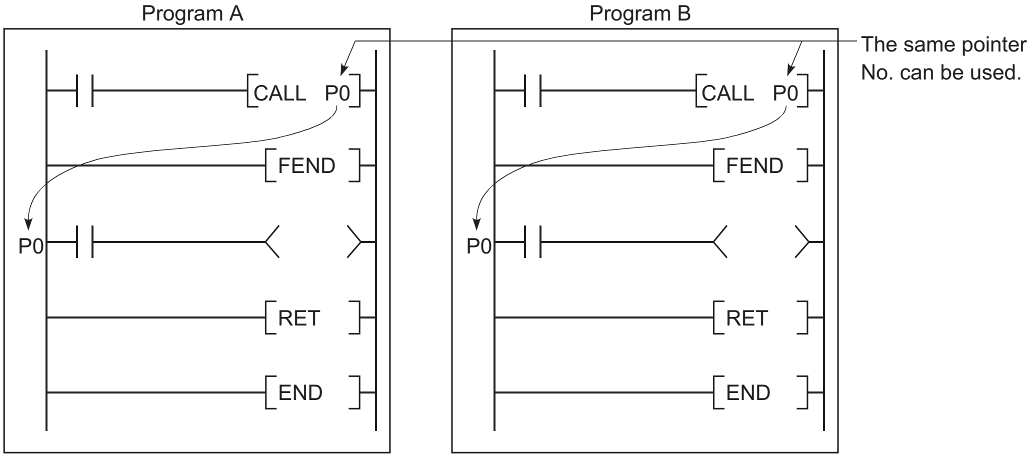


Figure 9.85 Using the same pointer in respective programs (local pointer)

(2) Number of local pointer points

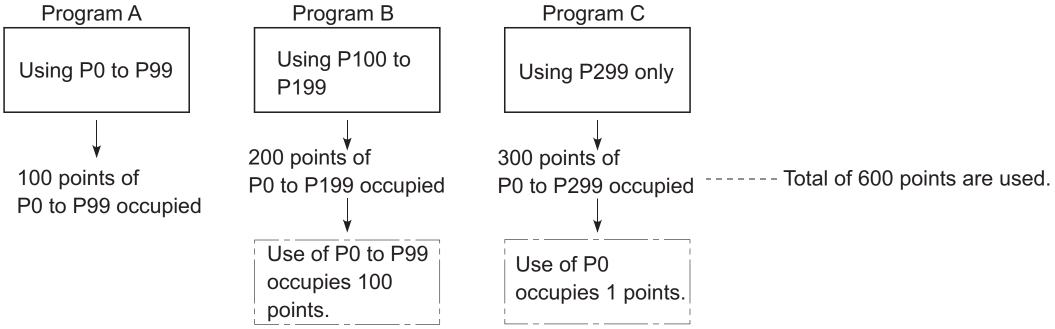
The local pointer can be divided for use of all the programs stored in the program memory.

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.

Example The total is 600 points when the pointer is used as shown below.



Program A
Using P0 to P99
100 points of P0 to P99 occupied
Program B
Using P100 to P199
200 points of P0 to P199 occupied
Use of P0 to P99 occupies 100 points.
Program C
Using P299 only
300 points of P0 to P299 occupied
Use of P0 occupies 1 points.
Total of 600 points are used.

Figure 9.86 Concept of the local pointer points

Note9.12 Basic

The Basic model QCPU is irrelevant to the difference between local pointers and common pointers.

(3) Precautions for using the local pointer

(a) Program where the local pointer is described

A jump from another program is not allowed.
jump instructions and sub-routine CALL instructions.
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 4096 points, a "Pointer configuration error" (error code: 4020) occurs.

9.9.2 Common pointer Note9.13

(1) Definition

The common pointer is used to call subroutine programs from all programs that are being executed.

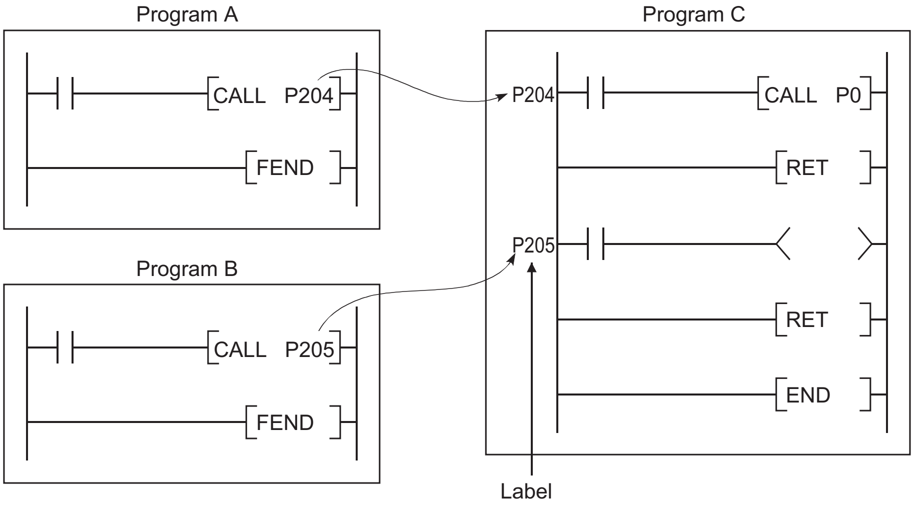


Figure 9.87 Calling pointers in another program (common pointer)

Note9.13 Basic

The Basic model QCPU has no distinction between the local and common pointers.

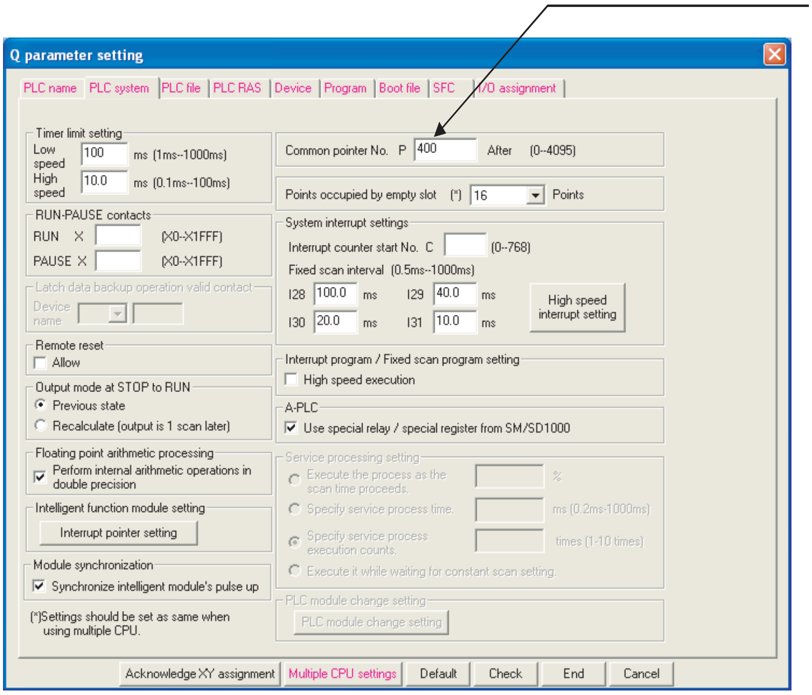
(2) Common pointer range

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.

Set the start number of the common pointer.



Q parameter setting

Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0-X1FFF)
PAUSE X (X0-X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
□ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting
Interrupt pointer setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P 400 After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms
High speed interrupt setting

Interrupt program / Fixed scan program setting
□ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting
PLC module change setting

Figure 9.88 Dialog box for setting the common pointer

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.

(3) Precautions

1) The same pointer number cannot be used as a label.
Doing so will result in a "Pointer configuration error" (error code: 4021).

2) 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.

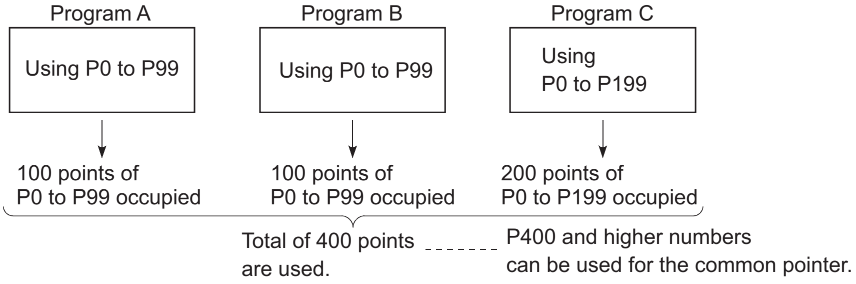

Program A
Using P0 to P99
Program B
Using P0 to P99
Program C
Using P0 to P199
100 points of P0 to P99 occupied
100 points of P0 to P99 occupied
200 points of P0 to P199 occupied
Total of 400 points are used.
P400 and higher numbers can be used for the common pointer.

Figure 9.89 Concept of the common pointer range

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.

9.10 Interrupt Pointer (I)

(1) Definition

The interrupt pointer (I) is used as a label at the start of an interrupt program, and can be used in any programs.

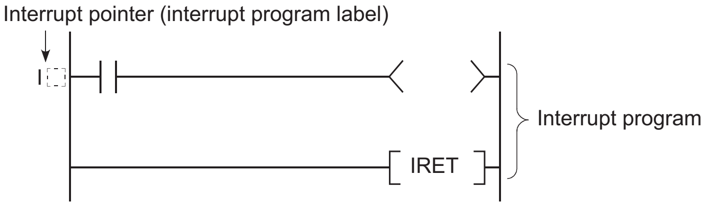


Figure 9.90 Interrupt pointer

(2) Number of available points

- Basic model QCPU: 128 points (I0 to I127)
- High Performance model QCPU, Process CPU, and Redundant CPU: 256 points (I0 to I255)

(3) Interrupt factors

Interrupt factors are listed in Table9.13.

Table9.13 Classification of interrupt factors

Interrupt factor	Interrupt pointer No.	Description	CPU module type			
			Basic model QCPU	High Performance model QCPU	Process CPU	Redun-dant CPU
Interrupt by an interrupt module [*1]	I0 to 15	Interrupt input from an interrupt module	○	○	○	○
Interrupt by a sequence-started module	I16 to I27	Interrupt from an AnS/A series special function module[*6] that is capable of starting an interrupt in the CPU module.	×	○	×	×
Interrupt by the internal timer	I28 to 31	Fixed scan interrupt to execute synchronized control with the operation cycle of a motion controller	○	○	○	○
	I49		×	△[*2]	×	×
Interrupt by an error [*3]	I32 to I39	Interrupt caused by an error which enables continuation of sequence program operation.	×	○	○	○
	I40, I41		×	×	×	○
Intelligent function module interrupt	I50 to 255[*4]	Interrupt from an intelligent function module [*5]	○	○	○	○

*1: For available interrupt modules, refer to the following.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

*2: For the Q02CPU, interrupt pointer I49 (the pointer dedicated to the high speed interrupt function) cannot be used.

*3: An interrupt is executed only when the operation mode after the error can be set to "Continue" in the PLC RAS tab of the PLC parameter dialog box.

*4: For the Basic model QCPU, the interrupt pointer numbers are I50 to I127.

*5: 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.

*6: This module can be an intelligent communication module.
For details, refer to the manual for each module.

*7: When using an interrupt module together with the High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.2)

Point

To use the intelligent function module interrupt (☞ Section 6.23), the intelligent function module setting (interrupt pointer setting) is required in the "PLC system" tab of the PLC parameter dialog box. (☞ Section 8.1.1(2), Section 8.1.2(2))

9.10.1 List of interrupt pointer numbers and interrupt factors

The list of interrupt pointer numbers and interrupt factors are shown below.

(1) Basic model QCPU

Table9.14 List of interrupt pointer numbers and interrupt factors (Basic model QCPU)

I No.	Interrupt factor		Priority
I0	Interrupt by interrupt module (QI60)	1st point	5
I1		2nd point	6
I2		3rd point	7
I3		4th point	8
I4		5th point	9
I5		6th point	10
I6		7th point	11
I7		8th point	12
I8		9th point	13
I9		10th point	14
I10		11th point	15
I11		12th point	16
I12		13th point	17
I13		14th point	18
I14		15th point	19
I15		16th point	20
I16 to I27	N/A	—	—
I28	Interrupt by internal timer [*1]	100ms	4
I29		40ms	3
I30		20ms	2
I31		10ms	1
I32 to I49	N/A	—	—
I50 to I127 [*2 *3]	Intelligent function module interrupt/Interrupt by interrupt module (QI60)	Specify the intelligent function module or interrupt module (QI60) with a parameter.	21 to 98

*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 2ms to 1000ms in increments of 1ms.

*2: 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 Section 6.23.)

*3: I50 has the highest priority (priority 21), and I127 has the lowest priority (priority 98).

(2) High Performance model QCPU

(a) When a Q series interrupt module is mounted

Table9.15 List of interrupt pointer numbers and interrupt factors (High Performance model QCPU)

I No.	Interrupt factor		Priority	I No.	Interrupt factor		Priority
I0	Interrupt by interrupt module (QI60)	1st point	2 to 223	I32 *5	Interrupt by an error *5 *6	All of stop errors	1
I1		2nd point		I33		SINGLE PS. DOWN *3 *4	
I2		3rd point		I34		UNIT VERIFY ERR., FUSE BREAK OFF, SP. UNIT ERROR, MULTI CPU ERROR	
I3		4th point					
I4		5th point					
I5		6th point		I35		OPERATION ERROR, SFCP OPE. ERROR, SFCP EXE. ERROR, EX.POWER OFF	
I6		7th point					
I7		8th point					
I8		9th point					
I9		10th point		I36		ICM. OPE ERROR, FILE OPE. ERROR	
I10		11th point					
I11		12th point		I37		Empty	
I12		13th point		I38		PRG. TIME OVER	
I13		14th point		I39		CHK instruction execution, annunciator detection	
I14		15th point					
I15		16th point		I40 to 48	—	Empty	—
I16	Interrupt by a sequence-started module *1	1st module	224				
I17		2nd module	225	I49	Interrupt by internal timer	0.2ms to 1ms *7	*10
I18		3rd module	226				
I19		4th module	227	I50 to 255	Intelligent function module interrupt *8 *9	Specify the intelligent function module or interrupt module (QI60) with a parameter.	2 to 223
I20		5th module	228				
I21		6th module	229				
I22		7th module	230				
I23		8th module	231				
I24		9th module	232				
I25		10th module	233				
I26		11th module	234				
I27		12th module	235				
I28	Interrupt by internal timer *2	100ms	239				
I29		40ms	238				
I30		20ms	237				
I31		10ms	236				

*1: Among the sequence-started modules on the base unit(s), the module closest to the High Performance model QCPU is assigned to the 1st module and the others are assigned in ascending order.

*2: 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.

*3: This is applicable to the module whose serial number (first five digits) is "07032" or later.

*4: In the case of a multiple CPU system, this is applicable only to CPU No.1 when serial numbers (first five digits) of all CPU modules are "07032" or later.

*5: After processing for "I32 (all of stop errors)" is completed, the High Performance model QCPU stops.

*6: Execution of I32 to I48 will be disabled (DI) if the system is powered on or when the High Performance model QCPU is reset.
To use any of I32 to I48, enable the interrupt with the IMASK instruction.
For the IMASK instruction, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

*7: Set an interval value of the internal timer in the High speed interrupt setting dialog box, which can be opened from the System interrupt settings area in the PLC system tab of the PLC parameter dialog box.
Set a value within the range of 0.2ms to 1.0ms in increments of 0.1ms.

*8: 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 Section 6.23.)

*9: I50 has the highest priority (priority 2), and I255 has the lowest priority (priority 223).

*10: When I49 is set in the PLC parameter dialog box, do not execute other interrupt programs (I0 to I48 and I50 to I255) and fixed scan execution type program.
Once a program such as the fixed scan execution type program is executed, the interrupt program of I49 cannot be executed at the specified interrupt intervals.

(b) When a A series interrupt module is mounted

Table9.16 List of interrupt pointer numbers and interrupt factors (High Performance model QCPU)

I No.	Interrupt factor		Priority	I No.	Interrupt factor		Priority
I0	Interrupt by interrupt module (A1SI61)	1st point	220	I32 *5	Interrupt by an error *5 *6	All of stop errors	1
I1		2nd point	221	I33		SINGLE PS. DOWN *3 *4	
I2		3rd point	222	I34		UNIT VERIFY ERR., FUSE BREAK OFF, SP. UNIT ERROR, MULTI CPU ERROR	
I3		4th point	223				
I4		5th point	224				
I5		6th point	225	I35		OPERATION ERROR, SFCP OPE. ERROR, SFCP EXE. ERROR, EX.POWER OFF	
I6		7th point	226				
I7		8th point	227				
I8		9th point	228				
I9		10th point	229	I36		ICM. OPE ERROR, FILE OPE. ERROR	
I10		11th point	230				
I11		12th point	231	I37		Empty	
I12		13th point	232	I38		PRG. TIME OVER	
I13		14th point	233	I39		CHK instruction execution, annunciator detection	
I14		15th point	234				
I15		16th point	235	I40 to 48	—	Empty	—
I16	Interrupt by a sequence-started module *1	1st module	208				
I17		2nd module	209	I49	Interrupt by internal timer	0.2ms to 1ms *7	*10
I18		3rd module	210				
I19		4th module	211	I50 to 255	Intelligent function module interrupt *8 *9	Specify the intelligent function module or interrupt module (QI60) with a parameter.	2 to 207
I20		5th module	212				
I21		6th module	213				
I22		7th module	214				
I23		8th module	215				
I24		9th module	216				
I25		10th module	217				
I26		11th module	218				
I27		12th module	219				
I28	Interrupt by internal timer *2	100ms	239				
I29		40ms	238				
I30		20ms	237				
I31		10ms	236				

*1: Among the sequence-started modules on the base unit(s), the module closest to the High Performance model QCPU is assigned to the 1st module and the others are assigned in ascending order.

*2: 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.

*3: This is applicable to the module whose serial number (first five digits) is "07032" or later.

*4: In the case of a multiple CPU system, this is applicable only to CPU No.1 when serial numbers (first five digits) of all CPU modules are "07032" or later.

*5: After processing for "I32 (all of stop errors)" is completed, the High Performance model QCPU stops.

*6: Execution of I32 to I48 will be disabled (DI) if the system is powered on or when the High Performance model QCPU is reset.
To use any of I32 to I48, enable the interrupt with the IMASK instruction.
For the IMASK instruction, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

*7: Set an interval value of the internal timer in the High speed interrupt setting dialog box, which can be opened from the System interrupt settings area in the PLC system tab of the PLC parameter dialog box.
Set a value within the range of 0.2ms to 1.0ms in increments of 0.1ms.

*8: 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 Section 6.23.)

*9: I50 has the highest priority (priority 2), and I255 has the lowest priority (priority 207).

*10: When I49 is set in the PLC parameter dialog box, do not execute other interrupt programs (I0 to I48 and I50 to I255) and fixed scan execution type program.
Once a program such as the fixed scan execution type program is executed, the interrupt program of I49 cannot be executed at the specified interrupt intervals.

(3) Process CPU

Table9.17 List of interrupt pointer numbers and interrupt factors (Process CPU)

I No.	Interrupt factor		Priority	I No.	Interrupt factor		Priority
I0	Interrupt by interrupt module (QI60)	1st point	208	I32 [*4]	Interrupt by an error [*4 *5]	All of stop errors	1
I1		2nd point	209	I33		SINGLE PS. DOWN [*2 *3]	
I2		3rd point	210	I34		UNIT VERIFY ERR., FUSE BREAK OFF, SP. UNIT ERROR, MULTI CPU ERROR	
I3		4th point	211				
I4		5th point	212				
I5		6th point	213	I35		OPERATION ERROR, SFCP OPE. ERROR, SFCP EXE. ERROR, EX.POWER OFF	
I6		7th point	214				
I7		8th point	215				
I8		9th point	216				
I9		10th point	217	I36		ICM. OPE ERROR, FILE OPE. ERROR	
I10		11th point	218				
I11		12th point	219	I37		Empty	
I12		13th point	220	I38		PRG. TIME OVER	
I13		14th point	221	I39		CHK instruction execution, annunciator detection	
I14		15th point	222				
I15		16th point	223	I40 to 49	—	Empty	—
I16 to I27	—	Empty	—	I50 to 255	Intelligent function module interrupt [*6 *7]	Specify the intelligent function module or interrupt module (QI60) with a parameter.	2 to 207
I28	Interrupt by internal timer [*1]	100ms	227				
I29		40ms	226				
I30		20ms	225				
I31		10ms	224				

*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 applicable to the module whose serial number (first five digits) is "07032" or later.

*3: In the case of a multiple CPU system, this is applicable only to CPU No.1 when serial numbers (first five digits) of all CPU modules are "07032" or later.

*4: After processing for "I32 (all of stop errors)" is completed, the Process CPU stops.

*5: Execution of I32 to I48 will be disabled (DI) if the system is powered on or when the Process CPU is reset.
To use any of I32 to I48, enable the interrupt with the IMASK instruction.
For the IMASK instruction, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

*6: 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 Section 6.23.)

*7: I50 has the highest priority (priority 2), and I255 has the lowest priority (priority 207).

(4) Redundant CPU

Table9.18 List of interrupt pointer numbers and interrupt factors (Redundant CPU)

I No.	Interrupt factor		Priority	I No.	Interrupt factor		Priority
I0	Interrupt by interrupt module (QI60)	1st point	208	I32 *2	Interrupt by an error *2 *3	All of stop errors	1
I1		2nd point	209	I33		SINGLE PS. DOWN	
I2		3rd point	210	I34		UNIT VERIFY ERR., FUSE BREAK OFF, SP. UNIT ERROR	
I3		4th point	211				
I4		5th point	212				
I5		6th point	213	I35		OPERATION ERROR, SFCP OPE. ERROR, SFCP EXE. ERROR, EX.POWER OFF	
I6		7th point	214				
I7		8th point	215				
I8		9th point	216				
I9		10th point	217	I36		ICM. OPE ERROR, FILE OPE. ERROR	
I10		11th point	218				
I11		12th point	219	I37		Empty	
I12		13th point	220	I38		PRG. TIME OVER	
I13		14th point	221	I39		CHK instruction execution, annunciator detection	
I14		15th point	222				
I15		16th point	223	I40		CAN'T SWITCH	
I16 to I27	–	Empty	–	I41		STANDBY	
				I42 to 49	–	Empty	–
I28	Interrupt by internal timer *1	100ms	227	I50 to 255	Intelligent function module interrupt *4 *5	Specify the intelligent function module or interrupt module (QI60) with a parameter.	2 to 207
I29		40ms	226				
I30		20ms	225				
I31		10ms	224				

*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: After processing for "I32 (all of stop errors)" is completed, the Redundant CPU stops.

*3: Execution of I32 to I48 will be disabled (DI) if the system is powered on or when the Redundant CPU is reset.
To use any of I32 to I48, enable the interrupt with the IMASK instruction.
For the IMASK instruction, refer to the following.
☞ MELSEC-Q/L Programming Manual (Common Instruction)

*4: 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 Section 6.23.)

*5: I50 has the highest priority (priority 2), and I255 has the lowest priority (priority 207).

9.11 Other Devices

9.11.1 SFC block device (BL)

The SFC block is used to check that the specified block in the SFC program is activated.

☞ MELSEC-Q/L/QnA Programming Manual (SFC)

9.11.2 SFC transition device (TR) Note9.14

This device is used for checking whether or not the transition condition for the specified SFC program block is set to forced transition.

☞ MELSEC-Q/L/QnA Programming Manual (SFC)

9.11.3 Network No. specification device (J)

(1) Definition

The network No. specification device is used to specify the network number in the link dedicated instructions.

(2) Specification method

In the link dedicated instruction, this device is specified as shown in Figure 9.91.

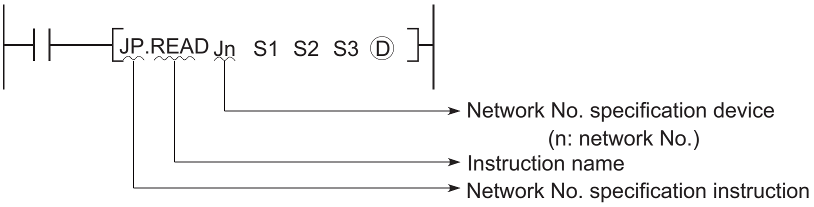

JP.READ Jn S1 S2 S3 Ⓓ
Network No. specification device
(n: network No.)
Instruction name
Network No. specification instruction

Figure 9.91 How to use the network No. specification device

Remark

For details of the link dedicated instructions, refer to the following.

☞ Manual for each network module

Note9.14 Basic

For the Basic model QCPU, the SFC transition device (TR) cannot be used.

9.11.4 I/O No. specification device (U)

(1) Definition

The I/O No. specification device is used to specify I/O numbers in the intelligent function module dedicated instructions.

(2) Specification method

In the intelligent function module dedicated instruction, this device is specified as shown in Figure 9.92.

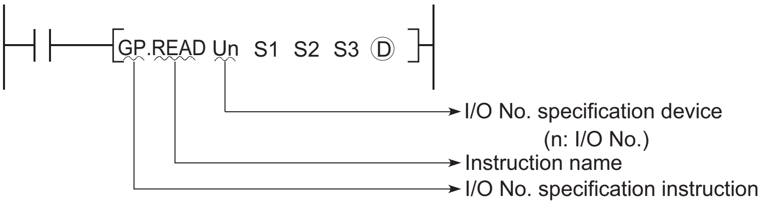

GP.READ Un S1 S2 S3 D
I/O No. specification device
(n: I/O No.)
Instruction name
I/O No. specification instruction

Figure 9.92 How to use the I/O No. specification device

Remark

For details of the intelligent function module dedicated instructions, refer to the following.

☞ Manual for the intelligent function module used

9.11.5 Macro instruction argument device (VD)

(1) Definition

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.

(2) Specification method

Among the devices used in the ladders registered as macros in GX Developer, 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.

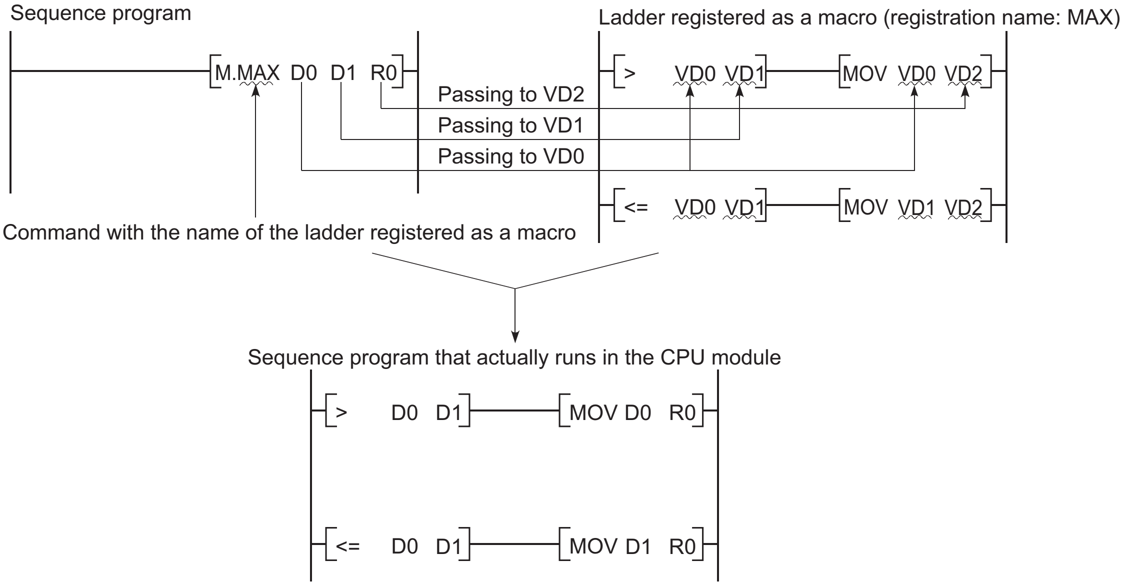

Sequence program
Ladder registered as a macro (registration name: MAX)
M.MAX D0 D1 R0
Passing to VD2
Passing to VD1
Passing to VD0
> VD0 VD1
MOV VD0 VD2
<= VD0 VD1
MOV VD1 VD2
Command with the name of the ladder registered as a macro
Sequence program that actually runs in the CPU module
> D0 D1
MOV D0 R0
<= D0 D1
MOV D1 R0

Figure 9.93 Macro instruction argument device specification

Point

- With the macro instruction argument device, VD0 to VD9 can be used in one macro registration ladder.
- The GX Developer read mode provides an option to view a program in macro instruction format.
  To change the display, select [View] → [Display macro instruction format].

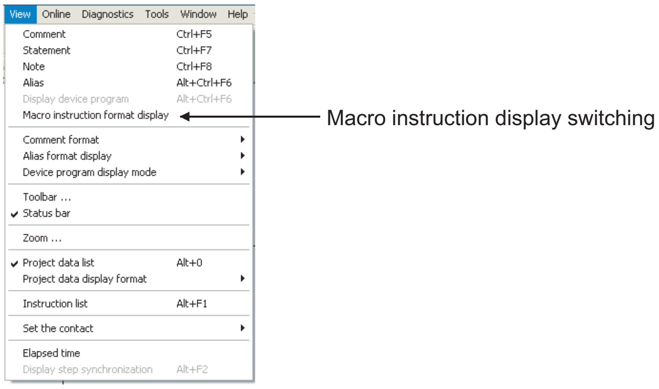

View Online Diagnostics Tools Window Help
Comment Ctrl+F5
Statement Ctrl+F7
Note Ctrl+F8
Alias Alt+Ctrl+F6
Display device program Alt+Ctrl+F6
Macro instruction format display
Comment format
Alias format display
Device program display mode
Toolbar ...
Status bar
Zoom ...
Project data list Alt+0
Project data display format
Instruction list Alt+F1
Set the contact
Elapsed time
Display step synchronization Alt+F2
Macro instruction display switching

Figure 9.94 Setting for macro instruction display

9.12 Constants

9.12.1 Decimal constant (K)

(1) Definition

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). (☞ Section 2.4.1)

(2) 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.

9.12.2 Hexadecimal constant (H)

(1) Definition

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). (☞ Section 2.4.2)

(2) 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)

9.12.3Real number (E) Note9.15

(1) Definition

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). (☞ Section 2.4.4)

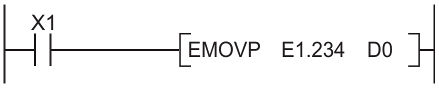


Figure 9.95 Real number specification

(2) Specification range

(a) Real number setting range

$-2^{128} <$ Device $\leqq -2^{-126}, 0, 2^{-126} \leqq$ Device $< 2^{128}$

(b) When an overflow or underflow has occurred

Table9.19 shows the operation of the CPU module when an overflow or underflow has occurred during arithmetic operation.

Table9.19 When an overflow or underflow has occurred

CPU module	Overflow	Underflow
Basic model QCPU	"OPERATION ERROR" (error code: 4100)	"OPERATION ERROR" (error code: 4100)
High Performance model QCPU		
Process CPU		Turned to 0 without any error*1
Redundant CPU		

*1: For the Process CPU, if the serial number (first five digits) is "07031" or earlier, "OPERATION ERROR" (error code: 4140) occurs.

(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. (For the High Performance model QCPU, an error occurs only when the internal operation is set to single precision.)

*1: The special values are -0, normalized numbers, nonnumeric characters, ±, and ∞.

(3) 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.

  Example 10.2345 can be specified as E10.2345.

- Exponential expression ••• A numeric value is specified by (Value) × 10^n.

  Example 234 is specified as E1.234 + 3.*1

*1: + 3 represents 10^3 in E1.234 + 3.

Note9.15 Basic

When using the real number operation function for the Basic model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 2.1)

9.12.4 Character string (" ") Note9.16

(1) Definition

The character string is a device used to specify a character string in sequence program.
Characters enclosed in quotation marks (example: "ABCD1234") are specified.

(2) Available characters

The shift JIS code can be used for character strings.
The CPU module distinguishes between upper and lower case characters.

(3) Number of specified characters

A string from the specified character to the NUL code (00H) is one unit.
Note that, however, up to 32 characters can be specified for an instruction using a character string, such as $MOV.

Note9.16 **Basic**

The Basic model QCPU can use character strings for only the $MOV, STR, DSTR, VAL, DVAL, ESTR and EVAL instructions.

9.13 Convenient Usage of Devices Note9.17

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.

9.13.1Global 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.

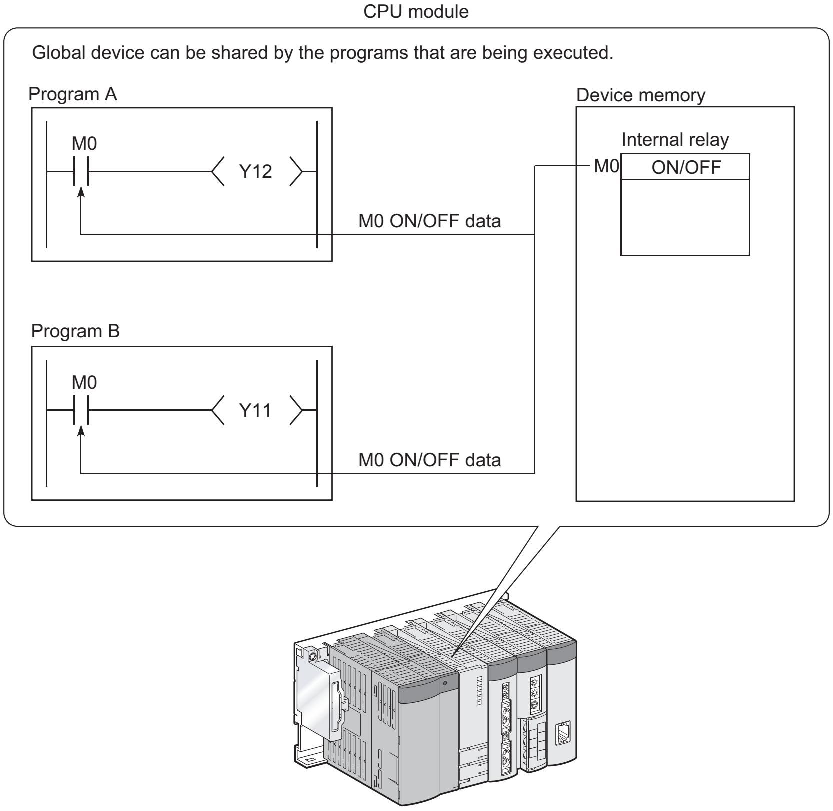


Figure 9.96 Using a global device

Note9.17 Basic

The Basic model QCPU is not support the function that makes each program independent by local device designation.

Point

- All of the devices that have not been set as local devices (☞ Section 9.13.2) 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 (☞ Section 9.13.2) must be specified in advance.

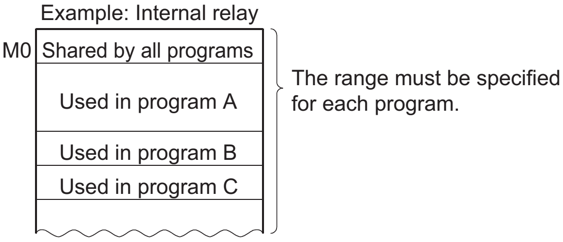


Figure 9.97 Range specification for a device

9.13.2 Local device

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.

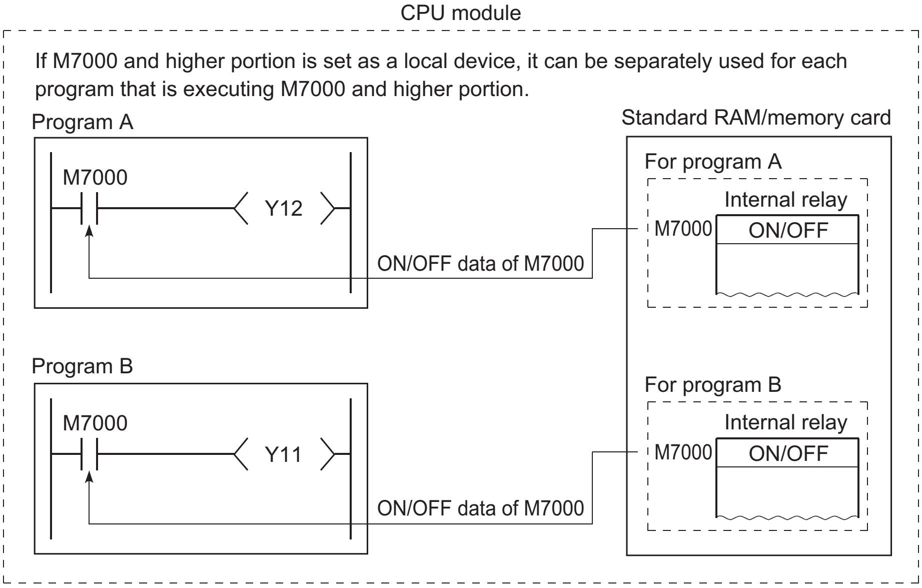


Figure 9.98 Using local devices

(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)

(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.

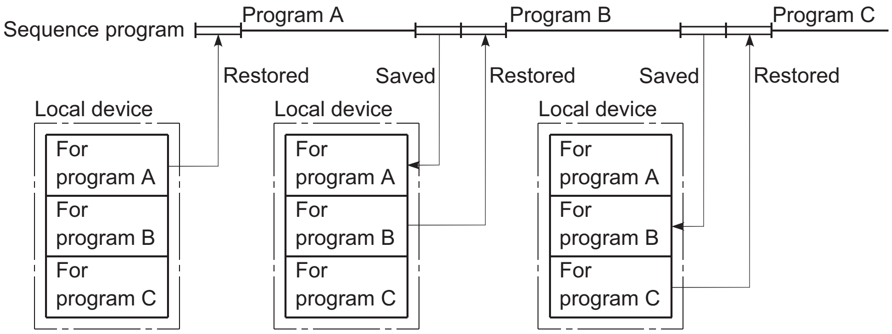


Figure 9.99 Saving and restoring local device files

Remark

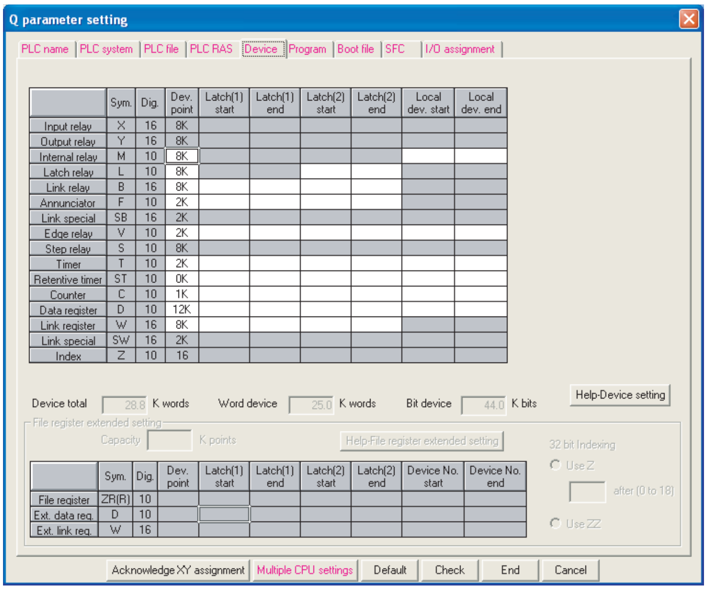
1. There are some instructions for which a local device cannot be specified.
   For details, refer to the pages describing devices available for each instruction in the following manual.
   MELSEC-Q/L Programming Manual (Common Instruction)

2. For the concept of the number of words used for the local devices, refer to Section 9.2.

(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.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Local dev. start	Local dev. end
Input relay	X	16	8K						
Output relay	Y	16	8K						
Internal relay	M	10	8K						
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	12K						
Link register	W	16	8K						
Link special	SW	16	2K						
Index	Z	10	16						

Device total 28.8 K words Word device 25.0 K words Bit device 44.0 K bits

File register extended setting

Capacity K points

	Sym.	Dig.	Dev. point	Latch(1) start	Latch(1) end	Latch(2) start	Latch(2) end	Device No. start	Device No. end
File register	ZR(R)	10							
Ext. data reg.	D	10							
Ext. link reg.	W	16							

32 bit Indexing

○ Use Z

after (0 to 18)

○ Use ZZ

Figure 9.100 Device

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.

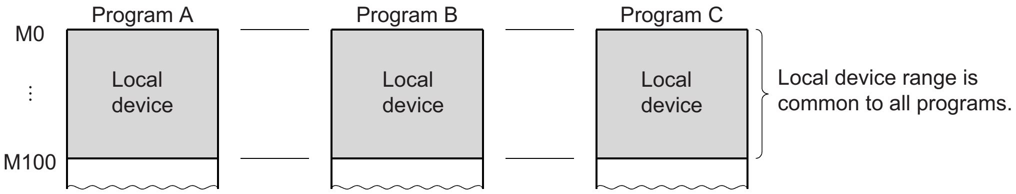
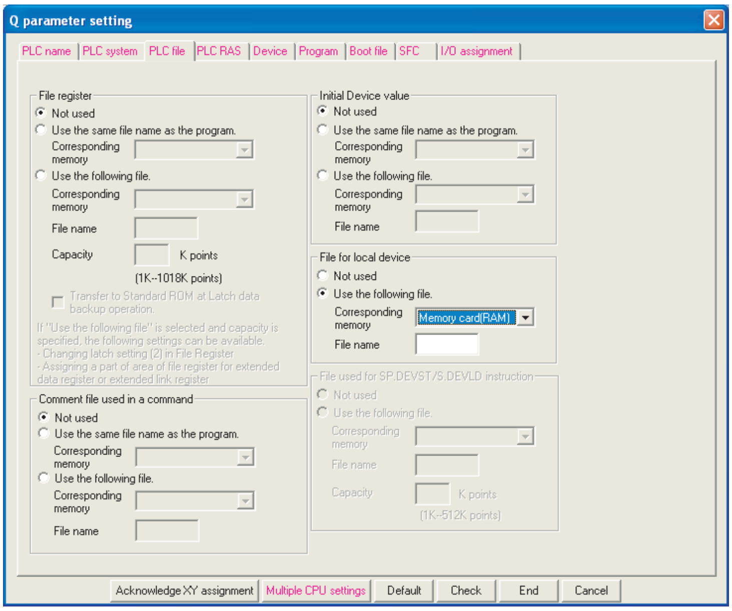


Figure 9.101 Local device range

(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.



Q parameter setting

File register
- ● Not used
- ○ Use the same file name as the program. Corresponding memory
- ○ Use the following file. Corresponding memory
- File name
- Capacity K points (1K--1018K points)
- □ Transfer to Standard ROM at Latch data backup operation.

If "Use the following file" is selected and capacity is specified, the following settings can be available.
- Changing latch setting (2) in File Register
- Assigning a part of area of file register for extended data register or extended link register

Comment file used in a command
- ● Not used
- ○ Use the same file name as the program. Corresponding memory
- ○ Use the following file. Corresponding memory
- File name

Initial Device value
- ● Not used
- ○ Use the same file name as the program. Corresponding memory
- ○ Use the following file. Corresponding memory
- File name

File for local device
- ○ Not used
- ● Use the following file.
- Corresponding memory: Memory card(RAM)
- File name

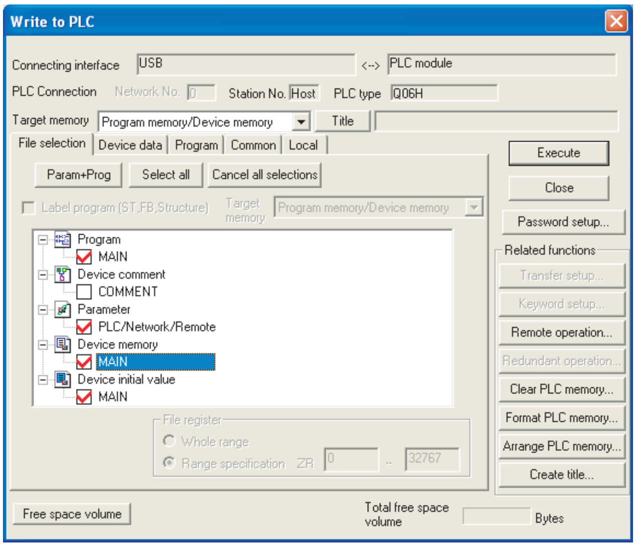
File used for SP.DEVST/S.DEVLD instruction
- ○ Not used
- ○ Use the following file. Corresponding memory
- File name
- Capacity K points (1K--512K points)

Figure 9.102 PLC file

(c) Writing the setting data

Write the data set in (a) and (b) to the CPU module.

Select [Online] → [Write to PLC] in GX Developer.



Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q06H

Target memory: Program memory/Device memory Title

□ Label program (ST,FB,Structure) Target memory: Program memory/Device memory

- Program
  - ☑ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - ☑ PLC/Network/Remote
- Device memory
  - ☑ MAIN
- Device initial value
  - ☑ MAIN

File register
- ○ Whole range
- ● Range specification ZR 0 -- 32767

Total free space volume Bytes

Figure 9.103 Device memory writing

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.

  1) Click the Trace result PLC read button on the Sampling trace dialog box to read the trace result into the personal computer. (☞ Section 6.14(5)(e))

  2) Click the Trace result button to display the trace result.

  3) Click the Create CSV file button to store the trace results in CSV format.

- All of the devices that have not been set as local devices are global devices.

(4) Setting of whether to use a local device (for each program)

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.

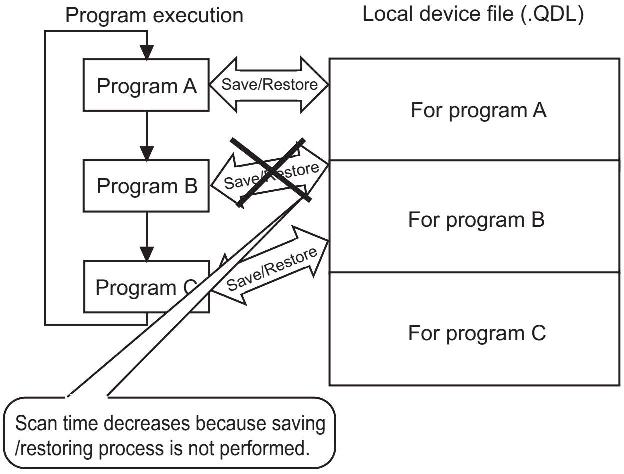
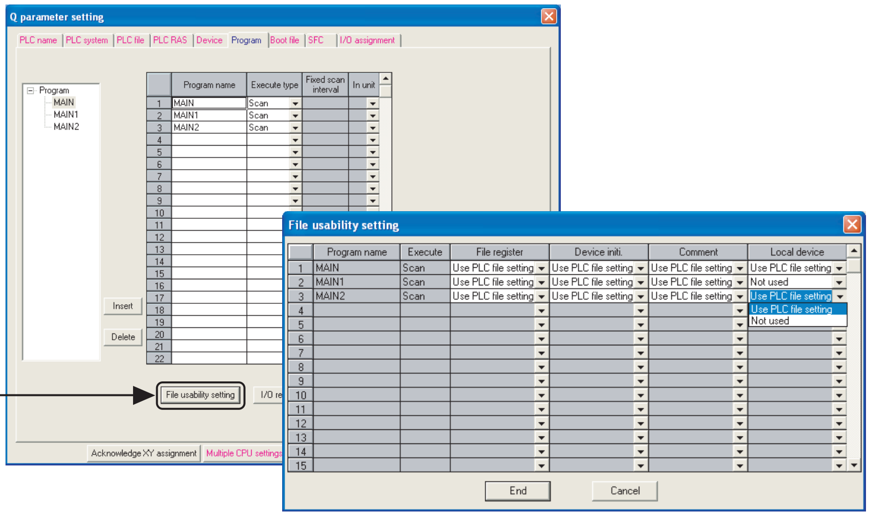


Figure 9.104 Save area configuration of a local device file

(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.



Q parameter setting

	Program name	Execute type	Fixed scan interval	In unit
1	MAIN	Scan		
2	MAIN1	Scan		
3	MAIN2	Scan		

File usability setting

	Program name	Execute	File register	Device initi.	Comment	Local device
1	MAIN	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
2	MAIN1	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Not used
3	MAIN2	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting

Click the File usability setting button.

Figure 9.105 File usability setting dialog box

(b) Precautions

1) 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.

2) Conditions for creating a local device file

Creation of a local device file depends on the PLC parameter settings.

Table9.20 shows the conditions for creating a local device file.

Table9.20 Conditions for creating a local device file

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	×	-

○ : Creates a file, × : Not create a file

*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 executing a subroutine program, you can utilize the local device corresponding to the file where the subroutine program is stored.

Use of the relevant local device is set by ON/OFF of SM776.

Table9.21 Local device switching by ON/OFF of the special relay (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

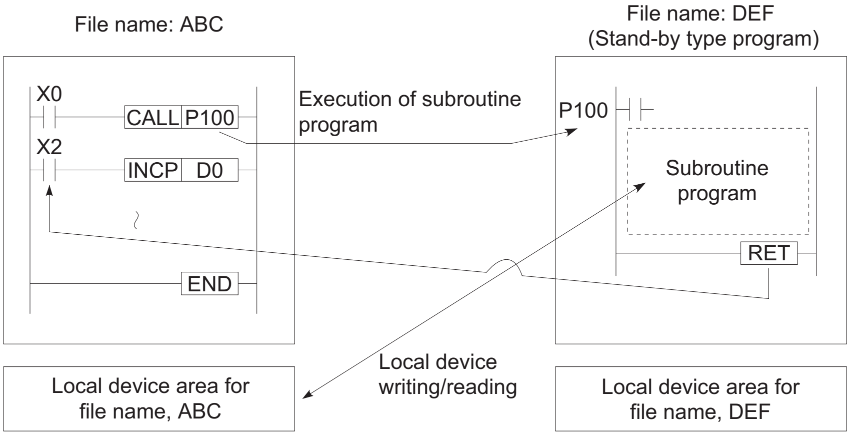

File name: ABC
File name: DEF
(Stand-by type program)
X0
CALL P100
X2
INCP D0
END
Execution of subroutine program
P100
Subroutine program
RET
Local device writing/reading
Local device area for file name, ABC
Local device area for file name, DEF

Figure 9.106 When SM776 is off

(b) When SM776 is on

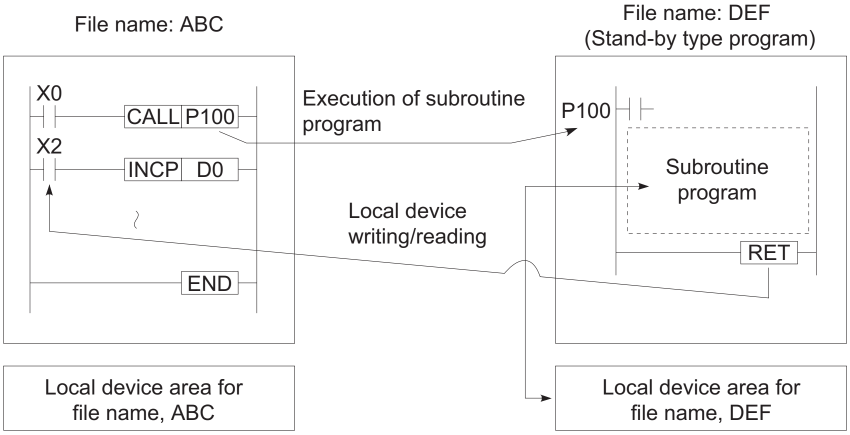

File name: ABC
File name: DEF
(Stand-by type program)
X0
CALL P100
X2
INCP D0
END
Execution of subroutine program
P100
Subroutine program
RET
Local device writing/reading
Local device area for file name, ABC
Local device area for file name, DEF

Figure 9.107 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) When executing an interrupt/fixed scan execution type program

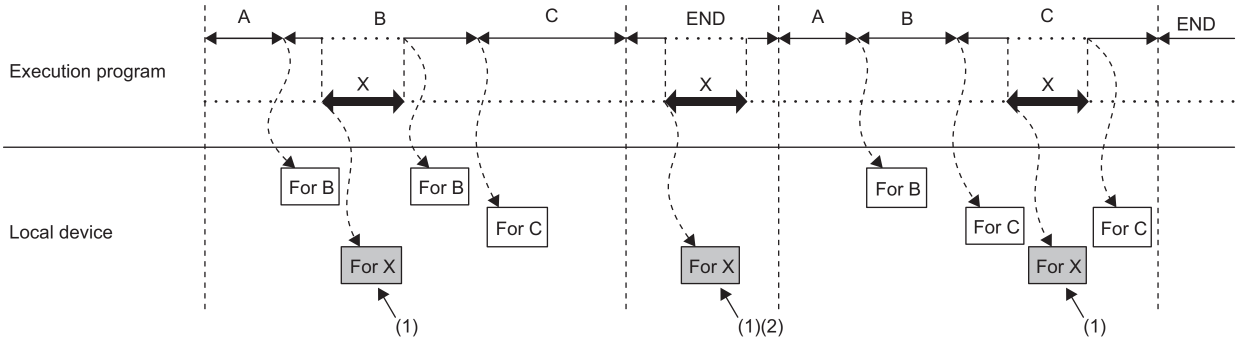
Set SM777(Enable/disable local device in interrupt program) to ON when a local device is used on an interrupt/ fixed scan execution type program. If SM777 set to OFF, SM777 does not operate normally.[*1]

[*1] For the index register that is set as the local device, the local device of the program executed before execution the interrupt/fixed scan execution type program is used regardless of the SM777 setting

Example Operation when SM777 is on with the following setting

Table9.22 Execution type of each program and local device enabled/disabled

Program name	Execution type	Local device enabled/disabled
A	Scan	Not used
B	Scan	Used
C	Scan	Used
X	Fixed	Used



(1) Use the local device of program X.

(2) When the interrupt/fixed scan execution type program is executed during the end processing, the end processing time is increased because the local device data of program C are saved and read out before execution of the end processing.

Figure 9.108 Operation when SM777 is on

(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.
- When the local device monitor is executed, the monitor switches to the applicable local device.
  Consequently, if SM777 is off, when an interrupt occurs immediately after switching, and a local device is accessed, the local device being monitored by the local device monitor is used. (The local device for the program being run prior to the interrupt (program immediately before END) is not accessed.)

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 from GX Developer.

CHAPTER10 CPU MODULE PROCESSING TIME

This chapter describes the CPU module processing time.

10.1 Scan Time

This section describes the scan time structures and CPU module processing time.

10.1.1 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) Scan time structure of the Basic model QCPU

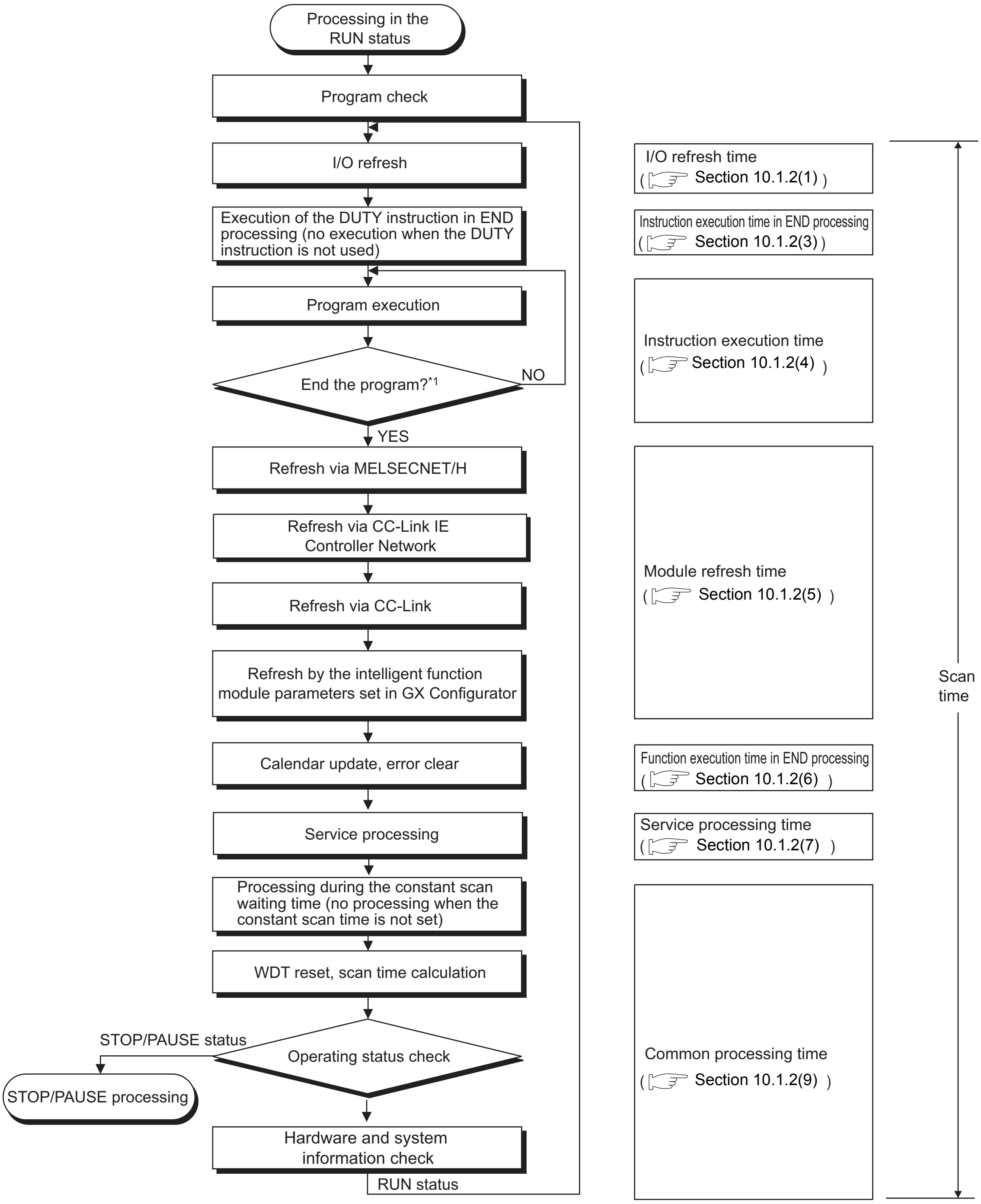

Processing in the RUN status
Program check
I/O refresh
Execution of the DUTY instruction in END processing (no execution when the DUTY instruction is not used)
Program execution
End the program?*1
NO
YES
Refresh via MELSECNET/H
Refresh via CC-Link IE Controller Network
Refresh via CC-Link
Refresh by the intelligent function module parameters set in GX Configurator
Calendar update, error clear
Service processing
Processing during the constant scan waiting time (no processing when the constant scan time is not set)
WDT reset, scan time calculation
STOP/PAUSE status
Operating status check
STOP/PAUSE processing
Hardware and system information check
RUN status
I/O refresh time (☞ Section 10.1.2(1))
Instruction execution time in END processing (☞ Section 10.1.2(3))
Instruction execution time (☞ Section 10.1.2(4))
Module refresh time (☞ Section 10.1.2(5))
Function execution time in END processing (☞ Section 10.1.2(6))
Service processing time (☞ Section 10.1.2(7))
Common processing time (☞ Section 10.1.2(9))
Scan time

Figure 10.1 Scan time structure of the Basic model QCPU

*1: End of a program indicates the timing when the END, GOEND, FEND, or STOP instruction is executed.

(2) Scan time structure of the High Performance model QCPU and Process CPU

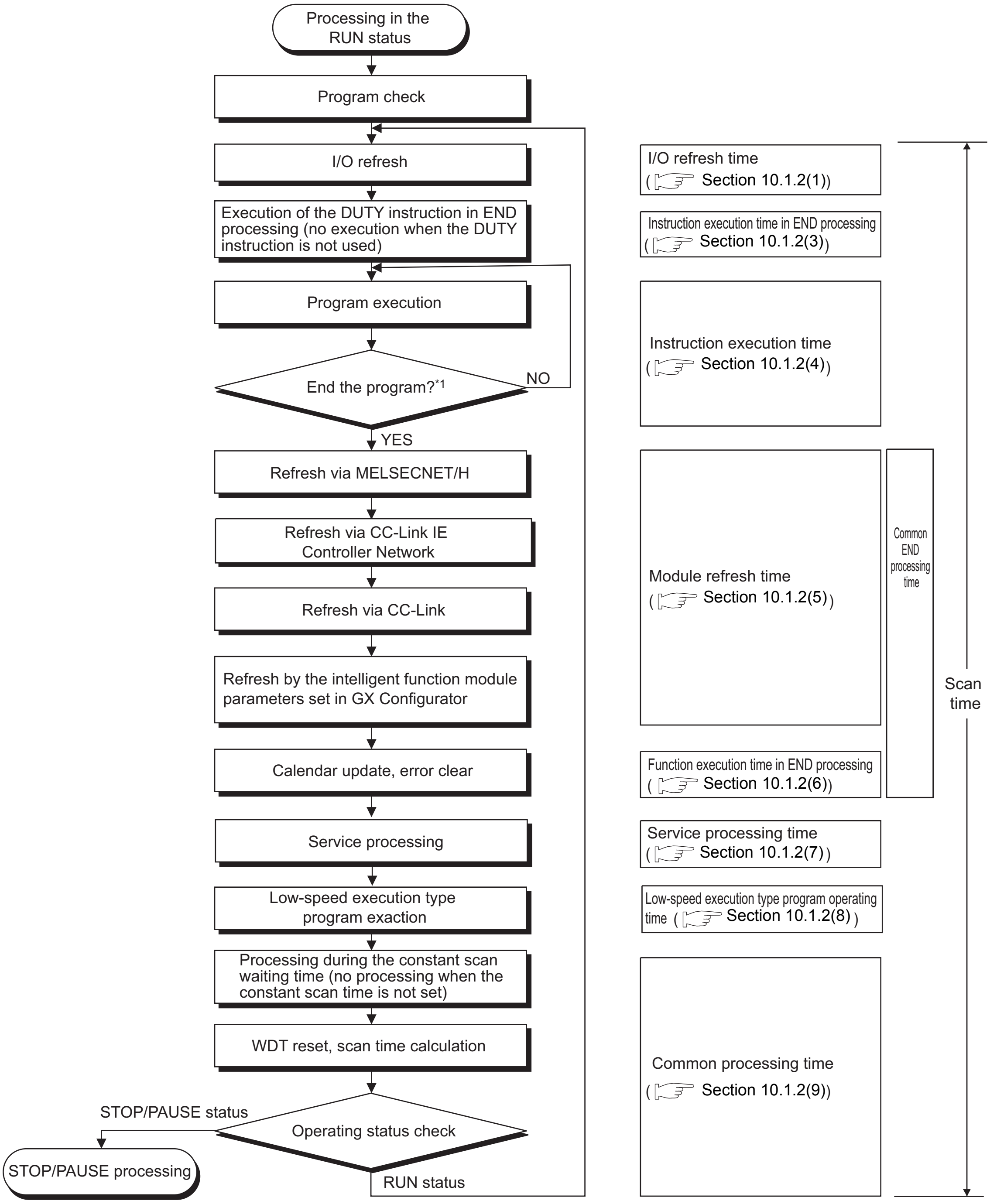

Processing in the RUN status
Program check
I/O refresh
Execution of the DUTY instruction in END processing (no execution when the DUTY instruction is not used)
Program execution
End the program?*1
NO
YES
Refresh via MELSECNET/H
Refresh via CC-Link IE Controller Network
Refresh via CC-Link
Refresh by the intelligent function module parameters set in GX Configurator
Calendar update, error clear
Service processing
Low-speed execution type program exaction
Processing during the constant scan waiting time (no processing when the constant scan time is not set)
WDT reset, scan time calculation
Operating status check
STOP/PAUSE status
STOP/PAUSE processing
RUN status
I/O refresh time (☞ Section 10.1.2(1))
Instruction execution time in END processing (☞ Section 10.1.2(3))
Instruction execution time (☞ Section 10.1.2(4))
Module refresh time (☞ Section 10.1.2(5))
Function execution time in END processing (☞ Section 10.1.2(6))
Common END processing time
Service processing time (☞ Section 10.1.2(7))
Low-speed execution type program operating time (☞ Section 10.1.2(8))
Common processing time (☞ Section 10.1.2(9))
Scan time

Figure 10.2 Scan time structure of the High Performance model QCPU and Process CPU

*1: End of a program indicates the timing when the END, GOEND, FEND, or STOP instruction is executed.

(3) Scan time structure of the Redundant CPU

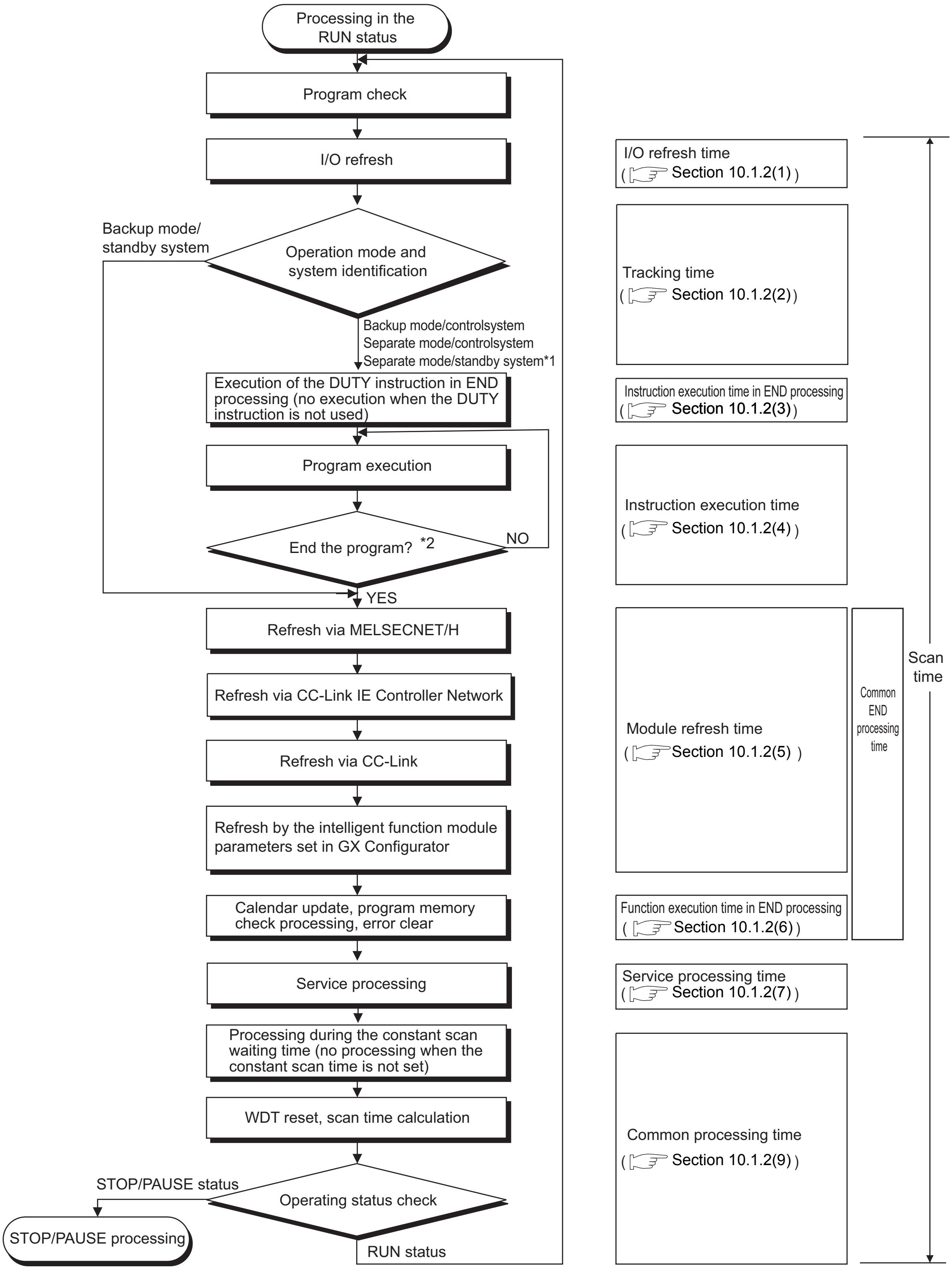

Processing in the RUN status
Program check
I/O refresh
Backup mode/ standby system
Operation mode and system identification
Backup mode/controlsystem
Separate mode/controlsystem
Separate mode/standby system*1
Execution of the DUTY instruction in END processing (no execution when the DUTY instruction is not used)
Program execution
End the program? *2
NO
YES
Refresh via MELSECNET/H
Refresh via CC-Link IE Controller Network
Refresh via CC-Link
Refresh by the intelligent function module parameters set in GX Configurator
Calendar update, program memory check processing, error clear
Service processing
Processing during the constant scan waiting time (no processing when the constant scan time is not set)
WDT reset, scan time calculation
STOP/PAUSE status
Operating status check
STOP/PAUSE processing
RUN status
I/O refresh time (Section 10.1.2(1))
Tracking time (Section 10.1.2(2))
Instruction execution time in END processing (Section 10.1.2(3))
Instruction execution time (Section 10.1.2(4))
Module refresh time (Section 10.1.2(5))
Function execution time in END processing (Section 10.1.2(6))
Service processing time (Section 10.1.2(7))
Common processing time (Section 10.1.2(9))
Common END processing time
Scan time

Figure 10.3 Scan time structure of the Redundant CPU

*1: When the CPU modules are switched from backup mode to separate mode, the CPU module in the standby system (the RUN LED turns on) cannot execute any program.

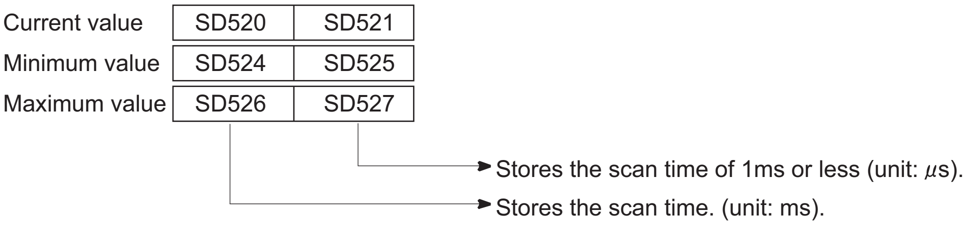
*2: End of a program indicates the timing when the END, GOEND, FEND, or STOP instruction is executed.

(4) 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.



Current value	SD520	SD521
Minimum value	SD524	SD525
Maximum value	SD526	SD527

Stores the scan time of 1ms or less (unit: μs).
Stores the scan time. (unit: ms).

Figure 10.4 Scan time storage location

Example If the stored values in SD520 and SD521 are 3 and 400 respectively, the scan time is 3.4ms.

10.1.2 Time required for each processing included in scan time

This section describes how to calculate the processing and execution time described in Section 10.1.1.

(1) I/O refresh time

The I/O refresh time is 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.

(I/O refresh time) = (number of input points/16) × N1 + (number of output points/16) × N2

For N1 and N2, refer to Table10.1.

Table10.1 I/O refresh time

CPU module	Q3□B, Q3□SB, Q3□RB, Q3□DB		Q5□B, Q6□B, Q6□RB		Q6□WRB		QA1S5□B, QA1S6□B		QA6□B, QA6ADP + A5□B, QA6ADP + A6□B	
	N1	N2	N1	N2	N1	N2	N1	N2	N1	N2
Q00JCPU	2.05μs	1.25μs	2.95μs	2.20μs	-	-	-	-	-	-
Q00CPU	2.00μs	1.20μs	2.75μs	2.05μs						
Q01CPU	1.95μs	1.15μs	2.70μs	2.00μs						
Q02CPU	2.2μs	1.3μs	2.9μs	2.1μs	-	-	4.9μs	3.9μs	5.6μs	4.8μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	1.7μs	1.3μs	2.4μs	2.1μs	-	-	4.3μs	3.9μs	5.0μs	4.8μs
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU							-	-	-	-
Q12PRHCPU, Q25PRHCPU					2.4μs	2.1μs				

Remark

The number of available base units differs depending on the CPU module.

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

(2) Tracking time

This is the processing time required when the tracking function is used in a redundant system.
For tracking time, refer to the following.

☞ QnPRHCPU User's Manual (Redundant System)

(3) Instruction execution time in END processing

This is the processing time of the DUTY instruction in END processing.
The user timing clock (SM420 to 424 and SM430 to SM434) specified with the DUTY instruction is turned on/off during the END processing. Note10.1

Table10.2 Instruction execution time in END processing

CPU module	Processing time in END processing	
	When set to 1	When set to 5
Q00JCPU	0.15ms	0.21ms
Q00CPU	0.14ms	0.19ms
Q01CPU	0.12ms	0.16ms
Q02CPU	0.02ms	0.02ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.01ms	0.01ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(4) Instruction execution time

The instruction execution time is the time required for all instructions used in the program to be executed.

Remark

For the processing time required for each instruction, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

Note10.1 Basic

The Basic model QCPU does not support the use of SM430 to SM434.

(a) Overhead time at execution of interrupt and fixed scan execution type programs

When calculating instruction execution time, add the overhead time given in the following table to the instruction execution time, which is described in (4).

Two kinds of overhead time (pre-start and program-end) need to be added to interrupt programs.

Table10.3 Pre-start overhead time for interrupt programs

CPU module	Fixed scan interrupt (I28 to I31)		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
Q00JCPU	175μs	150μs	350μs	325μs
Q00CPU	145μs	125μs	285μs	265μs
Q01CPU	135μs	120μs	270μs	255μs
Q02CPU	190μs	85μs	205μs	100μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	85μs	40μs	90μs	45μs
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	85μs	40μs	90μs	45μs
Q12PRHCPU, Q25PRHCPU (when no extension base unit is connected)	85μs	40μs	90μs	45μs
Q12PRHCPU, Q25PRHCPU (when extension base unit is connected)	85μs	40μs	1090μs[*2]	1045μs[*2]

*1: Indicates the value when the QI60 is mounted on the slot 0 of the main base unit.

*2: When any extension base unit is connected, the Q160 cannot be used. The values in the above list indicate overhead time for interrupt processing from an intelligent function module.

Table10.4 Program-end overhead time for interrupt programs

CPU module	Without high-speed start	With high-speed start
Q00JCPU	175μs	150μs
Q00CPU	145μs	125μs
Q01CPU	135μs	120μs
Q02CPU	180μs	75μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	80μs	35μs
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

Table10.5 Overhead time for fixed scan execution type programs

CPU module	Without high-speed start	With high-speed start
Q02CPU	380μs	230μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	165μs	100μs
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

1) Overhead time when local devices in the interrupt program are enabled

When SM777 (Enable/disable local device in interrupt program) is turned on, the time given in Table10.6 and Table10.7 will be added to the overhead time given in Table10.3 and Table10.4.

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

Table10.6 When a local device file in the standard RAM is used

CPU module	Additional time to the pre-start overhead time for interrupt programs (Table10.3)	Additional time to the program-end overhead time for interrupt programs (Table10.4)
Q02CPU	$(0.35 \times n + 0.05) \times 10^3 \mu s$	$(0.35 \times n + 0.05) \times 10^3 \mu s$
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$(0.15 \times n + 0.03) \times 10^3 \mu s$	$(0.15 \times n + 0.03) \times 10^3 \mu s$
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

Table10.7 When a local device file in a SRAM card is used

CPU module	Additional time to the pre-start overhead time for interrupt programs (Table10.3)	Additional time to the program-end overhead time for interrupt programs (Table10.4)
Q02CPU	$(1.15 \times n + 0.30) \times 10^3 \mu s$	$(1.15 \times n + 0.30) \times 10^3 \mu s$
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$(0.85 \times n + 0.15) \times 10^3 \mu s$	$(0.85 \times n + 0.15) \times 10^3 \mu s$
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(5) Module refresh time

Module refresh time is the total time required for the CPU module to refresh data with CC-Link IE Controller Network, MELSECNET/H, and CC-Link modules.

(a) Refresh via CC-Link IE Controller Network

This is the time required for refreshing data between link devices in a CC-Link IE Controller Network module and devices in the CPU module.

(b) Refresh via MELSECNET/H

This is the time required for refreshing data between link devices in a MELSECNET/H network module and devices in the CPU module.

(c) Auto refresh via CC-Link

This is the time required for refreshing data between a CC-Link system master/local module and CPU module.

Remark

For each refresh time, refer to the following.

☞ Manual for each network module

(d) 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.

Use intelligent function module utility package (GX Configurator) for auto refresh settings.

■ Calculation method

Use the following expression to calculate the auto refresh time with an intelligent function module.

(Refresh time) = KN1 + KN2 × (number of refresh points)

For KN1 and KN2, use the values given in Table10.8 and Table10.9.

Table10.8 When an intelligent function module is mounted on the main base unit

CPU module	KN1	KN2
Q00JCPU	115μs	55μs
Q00CPU	91μs	46μs
Q01CPU	85μs	41μs
Q02CPU	53μs	13μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	27μs	6μs
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

Table10.9 When an intelligent function module is mounted on the extension base unit

CPU module	KN1	KN2
Q00JCPU	120μs	56μs
Q00CPU	92μs	48μs
Q01CPU	86μs	43μs
Q02CPU	61μs	15μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	29μs	8μs
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

Example When the number of auto refresh points for the analog-digital converter module (Q64AD) is 4 points (when the module is mounted on the Q01CPU main base unit)

0.249(ms) = 0.085 + 0.041 × 4

(6) Function execution time in END processing

This is the time required for updating calender, clearing error in END processing, or checking the memory.

(a) Calendar update processing time

When the clock data set request (SM210 changes from off to on) or the clock data read request (SM213 turns on) is issued, the processing time for changing or reading the clock data is required in END processing.

Table10.10 Calendar update processing time

CPU module	Processing time in END processing	
	When the clock data set request is issued	When the clock data read request is issued
Q00JCPU	1.25ms	0.04ms
Q00CPU	0.99ms	0.03ms
Q01CPU	0.98ms	0.02ms
Q02CPU	0.26ms	0.01ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.11ms	0.005ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(b) Error clear processing time

Upon the rising edge (changes from off to on) of SM50 (Error reset), the processing time for clearing the continuation error stored in SD50 is required.

Table10.11 Error clear processing time

CPU module	Processing time in END processing	
	When the error is cleared (the one detected by the annunciator)	When the error is cleared
Q00JCPU	2.1ms	2.0ms
Q00CPU	1.75ms	1.7ms
Q01CPU	1.45ms	1.35ms
Q02CPU	1.15ms	0.41ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.84ms	0.21ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(c) Memory check processing time

When the "Check Program memory" parameter is selected in the PLC RAS(2) tab of the PLC parameter dialog box, the following processing time is required.

[Processing time required]

$$= \frac{\text{Size to be checked (steps)}^{*1}\text{ per scan}}{1024} \times 3.5\text{ ms}$$

*1: Indicates the number of steps set in the PLC RAS(2) tab of the PLC parameter dialog box. (☞ Section 8.1.2(5))

(7) Service processing time

Service processing is the communication processing with GX Developer and external devices.

(a) Basic model QCPU

When monitoring device data by GX Developer, the processing time shown in Table10.12 is required.

Table10.12 Processing time to monitor device data

Function	GX Developer is connected to an RS-232 interface of the host station CPU module			GX Developer is connected to an RS-232 interface of another station CPU module[*4]		
	Q00JCPU	Q00CPU	Q01CPU	Q00JCPU	Q00CPU	Q01CPU
Read a program from the programmable controller[*1]	1.6ms	1.3ms	1.2ms	2.3ms	1.9ms	1.8ms
Device monitor[*2]	1.2ms	1.0ms	0.9ms	2.4ms	2.0ms	1.9ms
Online change[*3]	1.0ms	1.0ms	1.0ms	1.9ms	1.6ms	1.5ms

[*1] Time required for reading an 8K-step program from the program memory.

[*2] Time required when 32 points is set in registration monitor.

[*3] Time required when a 100-step ladder is added.

[*4] Time required when accessed via the MELSECNET/H, Ethernet, CC-Link, or serial communication module.

For the time required for communicating with the serial communication module and Ethernet module, refer to the following.

MELSEC Communication Protocol Reference Manual

(b) High Performance model QCPU, Process CPU, and Redundant CPU

When monitoring device data, reading programs, and setting monitor conditions in GX Developer, the processing time shown in Table10.13 or Table10.14 is required.

Table10.13 Processing time to monitor device data and read programs

CPU module	Processing time
Q02CPU	0.017ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.011ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	
Q12PRHCPU, Q25PRHCPU	

Table10.14 Processing time to set monitor conditions

CPU module	Processing time	
	Specified step match	Specified device match
Q02CPU	0.05ms	0.01ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.03ms	0.01ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(8) Low speed program operation time

The low speed program operation time is the sum of processing times of the instructions used in the low speed execution type program to be executed by the CPU module.

Remark

For the processing time required for each instruction, refer to the following.

☞ MELSEC-Q/L Programming Manual (Common Instruction)

(9) Common processing time

The CPU module performs common processing by the system. The common processing time shown in Table10.15 is required.

Table10.15 Common processing time

CPU module	Processing time
Q00JCPU	0.66ms
Q00CPU	0.60ms
Q01CPU	0.52ms
Q02CPU	0.40ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.16ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	
Q12PRHCPU, Q25PRHCPU	0.5ms

10.1.3 Factors that increase the scan time

When executing any of the functions or operations described in this section, add the given processing time to the time value calculated in Section 10.1.2.

(1) Sampling trace

When the sampling trace function (☞ Section 6.14) is executed, the processing time shown in Table10.16 is required.

Table10.16 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	Memory card
Q02CPU	0.16ms	0.24ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.06ms	0.12ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU,		
Q12PRHCPU, Q25PRHCPU		

(2) Use of local devices

When local devices are used, the processing time shown in Table10.17 is required.
"n" in the table indicates the number of programs to be executed.

Table10.17 Processing time (when local devices are used)

CPU module	Processing time	
	Standard RAM	Memory card
Q02CPU	$3.52 \times n \times 10^{3} \mu s$[*1]	$10.73 \times n \times 10^{3} \mu s$[*1]
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$1.54 \times n \times 10^{3} \mu s$[*1]	$8.16 \times n \times 10^{3} \mu s$[*1]
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU,		
Q12PRHCPU, Q25PRHCPU		

*1: Indicates the processing time when 10K points of local devices are used.

(a) When local devices in a subroutine program are enabled

When SM776 (Enable/disable local device at CALL) is turned on, the processing time shown in Table10.18 or Table10.19 is required for each subroutine call.

"n" in the table indicates the number of local device points (unit: K words).

Table10.18 Processing time (when a local device file in the standard RAM is used)

CPU module	Processing time when a subroutine program in the same file is called	Processing time when a subroutine program in a different file is called
Q02CPU	$(0.35 \times n + 0.05) \times 10^3 \mu s$	$(0.70 \times n + 0.10) \times 10^3 \mu s$
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$(0.15 \times n + 0.03) \times 10^3 \mu s$	$(0.30 \times n + 0.05) \times 10^3 \mu s$
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

Table10.19 Processing time (when a local device file in a SRAM card is used)

CPU module	Processing time when a subroutine program in the same file is called	Processing time when a subroutine program in a different file is called
Q02CPU	$(1.15 \times n + 0.30) \times 10^3 \mu s$	$(2.30 \times n + 0.60) \times 10^3 \mu s$
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$(0.85 \times n + 0.15) \times 10^3 \mu s$	$(1.65 \times n + 0.30) \times 10^3 \mu s$
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(3) Execution of multiple programs

When multiple programs are executed, the processing time shown in Table10.20 is required for each program.

Table10.20 Processing time for each program (when multiple programs are executed)

CPU module	Processing time
Q02CPU	0.08 × n ms[*1]
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.03 × n ms[*1]
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	
Q12PRHCPU, Q25PRHCPU	

*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 shown in Table10.21 is required only for one scan where a memory card is removed or inserted.

Table10.21 Processing time (when a memory card is removed or inserted)

CPU module	Processing time	
	When a memory card is inserted	When a memory card is removed
Q02CPU	0.16ms	0.10ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.08ms	0.04ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

(5) 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 shown in Table10.22 is required.

When "Use the following file." is selected, the scan time will not be increased.

Table10.22 Processing time (when the file register is used)

CPU module	Processing time	
	Standard RAM	Memory card
Q02CPU	1.03ms	1.14 × n ms[*1]
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.41ms	0.50 × n ms[*1]
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

*1: "n" indicates the number of program files.

(6) 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 shown in Table10.23 is required.

Table10.23 Processing time (online change (ladder mode))

CPU module	Processing time	
	The reserved area for online change is not changed.	The reserved area for online change is re-set.
Q00JCPU	Up to 2.1ms	Up to 30ms
Q00CPU	Up to 1.7ms	Up to 26ms
Q01CPU	Up to 1.7ms	Up to 36ms
Q02CPU	Up to 4ms (Up to 10ms[*1])	Up to 30ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	Up to 2ms (Up to 4ms[*1])	Up to 90ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

*1: A value when a local pointer is used in the program targeted for online change

(b) Online change (files)

When a file is written to the running CPU module, the processing time shown in Table10.24 is required. When an ATA card is used, 1.25 seconds is required for each 30K steps.

Table10.24 Processing time (online change (files))

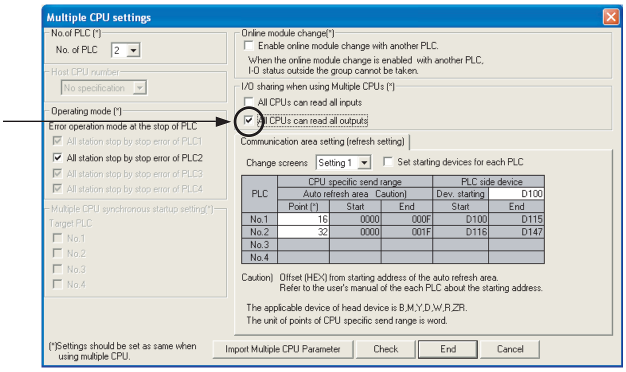
CPU module	Free area is available in the program memory.	Free area is available in a memory card (except an ATA card).
Q02CPU	Up to 90ms (Up to 200ms[*2])	Up to 150ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	Up to 350ms (Up to 650ms[*2])	Up to 650ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU		
Q12PRHCPU, Q25PRHCPU		

*2: A value for the SFC program

(7) 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.



Multiple CPU settings

No.of PLC (*)
No. of PLC: 2

Host CPU number: No specification

Operating mode (*)
Error operation mode at the stop of PLC
☑ All station stop by stop error of PLC1
☑ All station stop by stop error of PLC2
☑ All station stop by stop error of PLC3
☑ All station stop by stop error of PLC4

Multiple CPU synchronous startup setting(*)
Target PLC
□ No.1
□ No.2
□ No.3
□ No.4

Online module change(*)
□ Enable online module change with another PLC.
When the online module change is enabled with another PLC, I-O status outside the group cannot be taken.

I/O sharing when using Multiple CPUs (*)
□ All CPUs can read all inputs
☑ All CPUs can read all outputs

Communication area setting (refresh setting)

Change screens: Setting 1 □ Set starting devices for each PLC

PLC	CPU specific send range			PLC side device	
	Auto refresh area Caution)			Dev. starting	D100
	Point (*)	Start	End	Start	End
No.1	16	0000	000F	D100	D115
No.2	32	0000	001F	D116	D147
No.3					
No.4					

Caution) Offset (HEX) from starting address of the auto refresh area.
Refer to the user's manual of the each PLC about the starting address.

The applicable device of head device is B,M,Y,D,W,R,ZR.
The unit of points of CPU specific send range is word.

(*)Settings should be set as same when using multiple CPU.

Figure 10.5 Multiple CPU settings screen

(8) Functions that increase the scan time only when the Basic model QCPU is used

When any of the following functions is executed, the scan time increases by the time required for its processing.

- System monitor
- Battery check
- Fuse blown check
- Module verification
- General data processing

When 12 intelligent function modules are mounted on one extension base unit and system monitor operation is performed, the processing time in the following table is required.

Table10.25 Processing time for system monitor (when 12 intelligent function modules are mounted)

CPU module	Processing time
Q00JCPU	0.036ms
Q00CPU	0.015ms
Q01CPU	0.011ms

(9) Memory check

When the memory check function is executed in the CPU module, the scan time increases by the time required for its processing. For the memory check processing time, refer to Section 6.27(4).

(10)Scan time measurement

When the scan time is measured by GX Developer, the processing time shown in Table10.26 is required. (☞ Section 6.13.3)

Table10.26 Processing time (when the scan time is measured)

CPU module	Processing time
Q02CPU	120.0 + 16.5 × number of branch instructions μs[*1]
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	52.0 + 7.0 × number of branch instructions μs[*1]
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	
Q12PRHCPU, Q25PRHCPU	

[*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

10.1.4 Factors that shorten the scan time

Scan time can be shortened by changing parameter settings as described in this section.

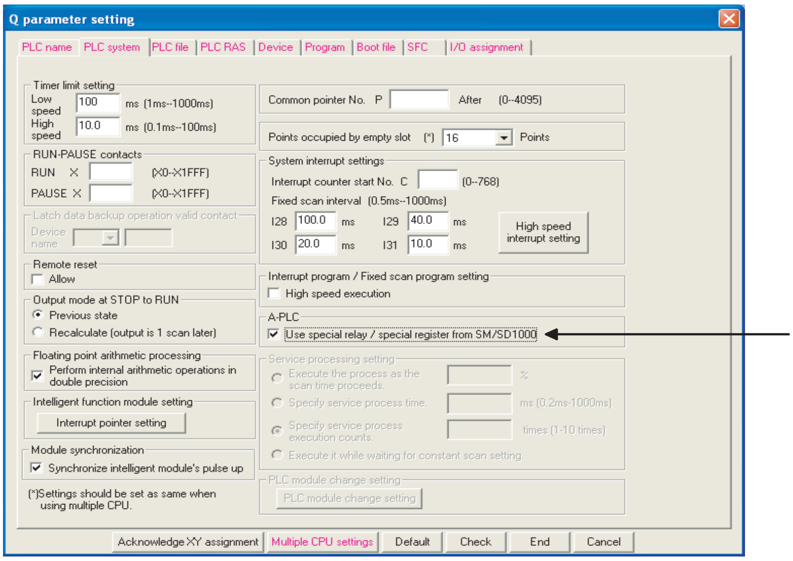
(1) A series CPU compatibility setting ("A-PLC") (☞ Section 8.1.2(2))

Note10.2

Scan time can be shortened by the time shown in Table10.27. To shorten the scan time, deselect the "Use special relay / special register from SM/SD1000" item in the PLC system tab of the PLC parameter dialog box.

Table10.27 Processing time that can be shortened (with "A-PLC" parameter setting)

CPU module	Processing time
Q02CPU	0.07ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.03ms
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment

Timer limit setting
Low speed: 100 ms (1ms--1000ms)
High speed: 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
□ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting
Interrupt pointer setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms
High speed interrupt setting

Interrupt program / Fixed scan program setting
□ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting
PLC module change setting

Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

Deselect this item.

Figure 10.6 A series CPU compatibility setting

Point

When deselect the "Use special relay / special register from SM/SD1000" item, the special relay and special register must be replaced as described below.

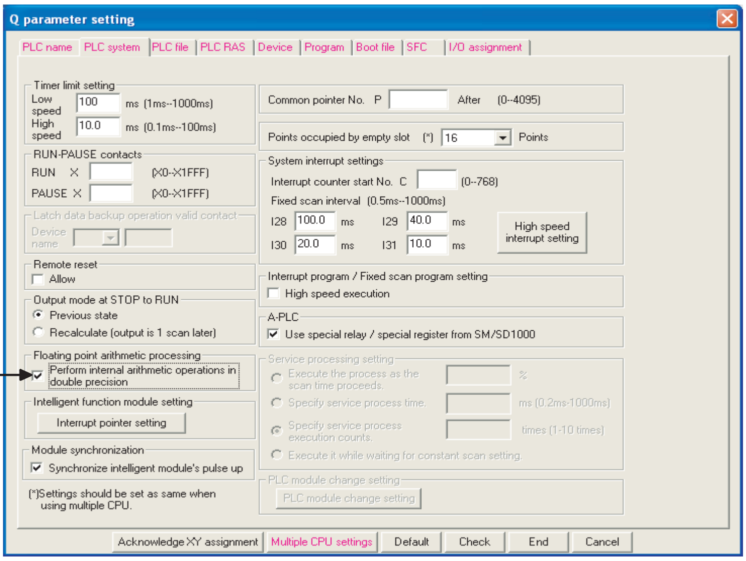
- A series-compatible special relay (SM1000 to SM1299)
  → Q series dedicated special relay (SM0 to SM999)
- A series-compatible special register (SD1000 to SD1299)
  → Q series dedicated special register (SD0 to SD999)

Note10.2 Basic Process Redundant

The A series CPU compatibility setting is not available for the Basic model QCPU, Process CPU, and Redundant CPU.

(2) Floating-point operation processing ("Floating point arithmetic processing") (☞ Section 8.1.2(2)) Note10.3

The time required for processing instructions that uses floating-point data can be shortened. To shorten the processing time, deselect the "Perform internal arithmetic operations in double precision" item in the PLC system tab of the PLC parameter dialog box.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.0 ms (0.1ms--100ms)

RUN-PAUSE contacts
RUN X (X0--X1FFF)
PAUSE X (X0--X1FFF)

Latch data backup operation valid contact
Device name

Remote reset
☐ Allow

Output mode at STOP to RUN
● Previous state
○ Recalculate (output is 1 scan later)

Floating point arithmetic processing
☑ Perform internal arithmetic operations in double precision

Intelligent function module setting

Module synchronization
☑ Synchronize intelligent module's pulse up

(*)Settings should be set as same when using multiple CPU.

Common pointer No. P After (0--4095)

Points occupied by empty slot (*) 16 Points

System interrupt settings
Interrupt counter start No. C (0--768)
Fixed scan interval (0.5ms--1000ms)
I28 100.0 ms I29 40.0 ms
I30 20.0 ms I31 10.0 ms

Interrupt program / Fixed scan program setting
☐ High speed execution

A-PLC
☑ Use special relay / special register from SM/SD1000

Service processing setting
○ Execute the process as the scan time proceeds. %
○ Specify service process time. ms (0.2ms-1000ms)
● Specify service process execution counts. times (1-10 times)
○ Execute it while waiting for constant scan setting.

PLC module change setting

Deselect this item.

Figure 10.7 Floating-point operation processing

Remark

For the operation processing time of the instruction that uses floating point data, refer to the following.

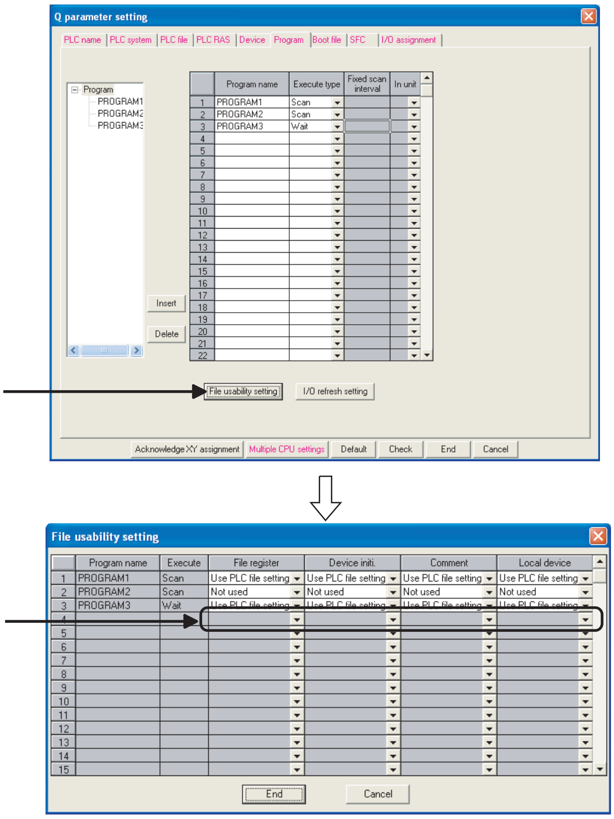
☞ QCPU Programming Manual (Common Instructions)

Note10.3 **Basic** **Process** **Redundant**

The floating-point operation setting is not available for the Basic model QCPU, Process CPU, and Redundant CPU.

(3) File usability setting (☞ Section 8.1.2(7)) Note10.4

Overhead time of a program can be shortened if the program uses no file register file, initial device value file, or device comment file. To shorten the overhead time, select "Not use" in the File usability setting dialog box.



Click this button.

Select "Not used" for the file of the file register, device initial value, or device comment that is not used in the program.

Q parameter setting

	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM1	Scan		
2	PROGRAM2	Scan		
3	PROGRAM3	Wait		

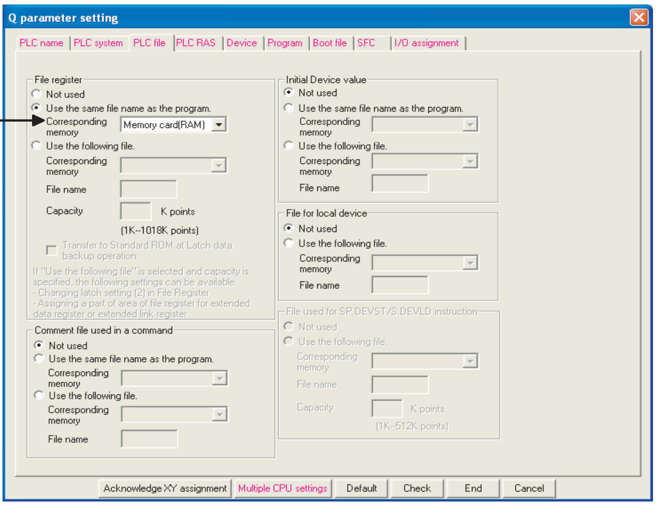
File usability setting

	Program name	Execute	File register	Device initi.	Comment	Local device
1	PROGRAM1	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
2	PROGRAM2	Scan	Not used	Not used	Not used	Not used
3	PROGRAM3	Wait	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting

Figure 10.8 File usability setting

Point

File usability setting is enabled only when the "Use the same file name as the program" item is selected in the PLC file tab of the PLC parameter dialog box.



File usability setting is enabled only when this item is selected.

Q parameter setting — PLC file

File register
- ○ Not used
- ● Use the same file name as the program.
  Corresponding memory: Memory card(RAM)
- ○ Use the following file.
  Corresponding memory
  File name
  Capacity K points (1K--1018K points)
- □ Transfer to Standard ROM at Latch data backup operation.

If "Use the following file" is selected and capacity is specified, the following settings can be available.
- Changing latch setting (2) in File Register
- Assigning a part of area of file register for extended data register or extended link register

Comment file used in a command
- ● Not used
- ○ Use the same file name as the program.
  Corresponding memory
- ○ Use the following file.
  Corresponding memory
  File name

Initial Device value
- ● Not used
- ○ Use the same file name as the program.
  Corresponding memory
- ○ Use the following file.
  Corresponding memory
  File name

File for local device
- ● Not used
- ○ Use the following file.
  Corresponding memory
  File name

File used for SP.DEVST/S.DEVLD instruction
- ○ Not used
- ○ Use the following file.
  Corresponding memory
  File name
  Capacity K points (1K--512K points)

Figure 10.9 PLC file tab

Note10.4 **Basic**

The file usability setting is not available for the Basic model QCPU.

CHAPTER11 PROCEDURES FOR WRITING PROGRAM TO CPU MODULE

This chapter describes procedures for writing a program created by GX Developer to the CPU module.

Remark

For procedures for starting the CPU module, refer to the following.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

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 wrong model names. If the modules are connected with wrong model names, data may not be written or read properly.

11.1 Basic Model QCPU

11.1.1 Items to be Considered for Creating Programs

To create a program, the number of device points, size, and file name of the program must be predetermined.

(1) Program size

Check whether the total size of programs and parameters are within the program size executable in the CPU module used. (☞ Section 5.4.3)
Table11.1 provides the program size executable in each CPU module.

Table11.1 Program size of the Basic model QCPU

CPU module	Program size
Q00JCPU	8k steps (32k bytes)
Q00CPU	8k steps (32k bytes)
Q01CPU	14k steps (56k bytes)

(2) Setting the applications of devices and the number of device points

When creating multiple programs, determine a unit (process/function) for structuring the programs. (☞ CHAPTER 9)

(3) Setting the initial device value

Set data necessary as initial values to the device memory of the Basic model QCPU and the buffer memory of the intelligent function module. (☞ Section 6.27)

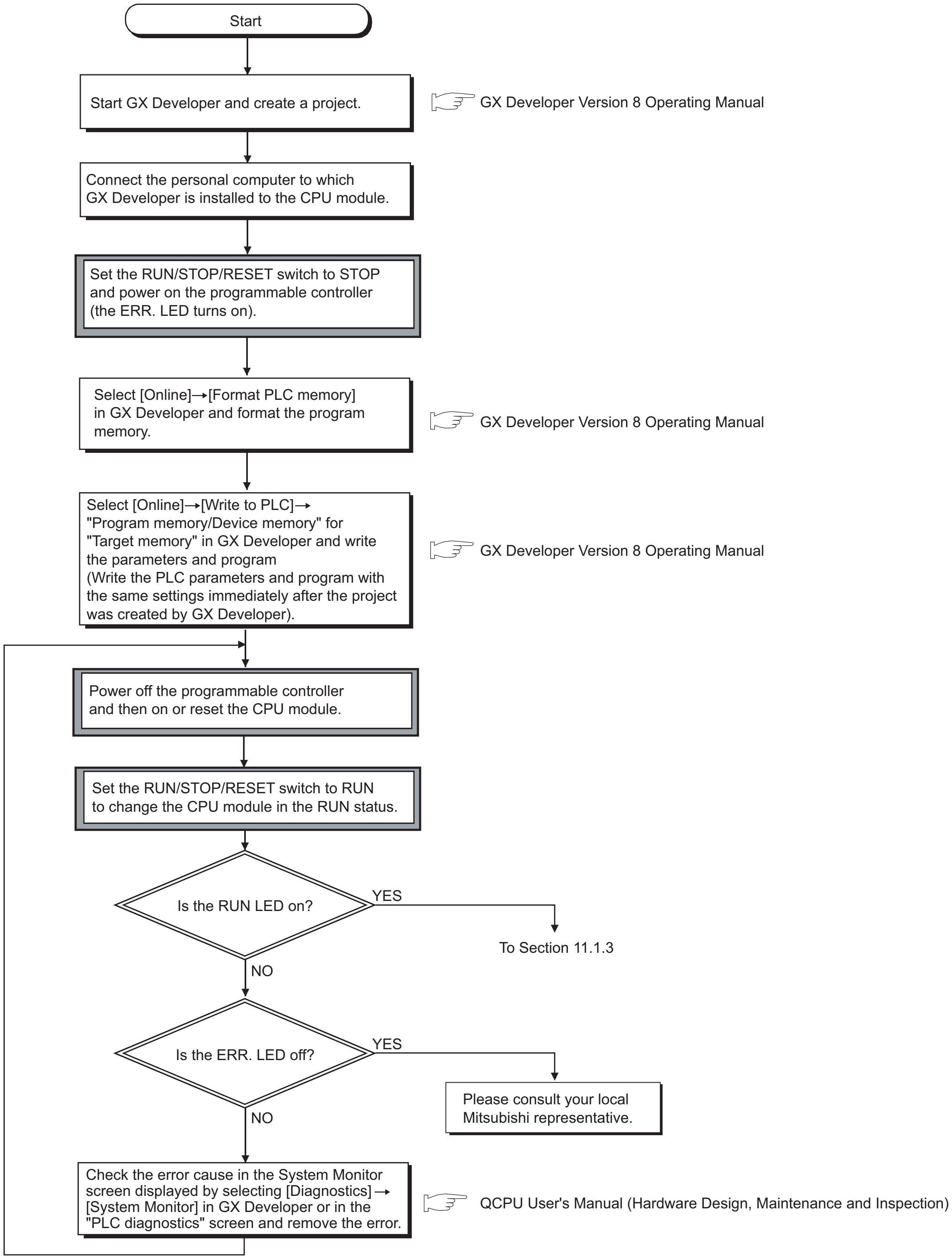
(4) Setting boot operation

When storing a program to the standard ROM, perform boot operation before executing the program.
Before boot operation, configure the settings in the Boot file tab of the PLC parameter dialog box.
(☞ Section 5.1.5, Section 11.1.4)

11.1.2 Hardware Check

This section describes a procedure for checking hardware before writing a created program.

In the following procedure, □ indicates an operation on the CPU module side.



Start
Start GX Developer and create a project.
GX Developer Version 8 Operating Manual
Connect the personal computer to which GX Developer is installed to the CPU module.
Set the RUN/STOP/RESET switch to STOP and power on the programmable controller (the ERR. LED turns on).
Select [Online]→[Format PLC memory] in GX Developer and format the program memory.
GX Developer Version 8 Operating Manual
Select [Online]→[Write to PLC]→ "Program memory/Device memory" for "Target memory" in GX Developer and write the parameters and program (Write the PLC parameters and program with the same settings immediately after the project was created by GX Developer).
GX Developer Version 8 Operating Manual
Power off the programmable controller and then on or reset the CPU module.
Set the RUN/STOP/RESET switch to RUN to change the CPU module in the RUN status.
Is the RUN LED on?
YES
To Section 11.1.3
NO
Is the ERR. LED off?
YES
Please consult your local Mitsubishi representative.
NO
Check the error cause in the System Monitor screen displayed by selecting [Diagnostics]→ [System Monitor] in GX Developer or in the "PLC diagnostics" screen and remove the error.
QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Figure 11.1 Hardware check flowchart

Remark

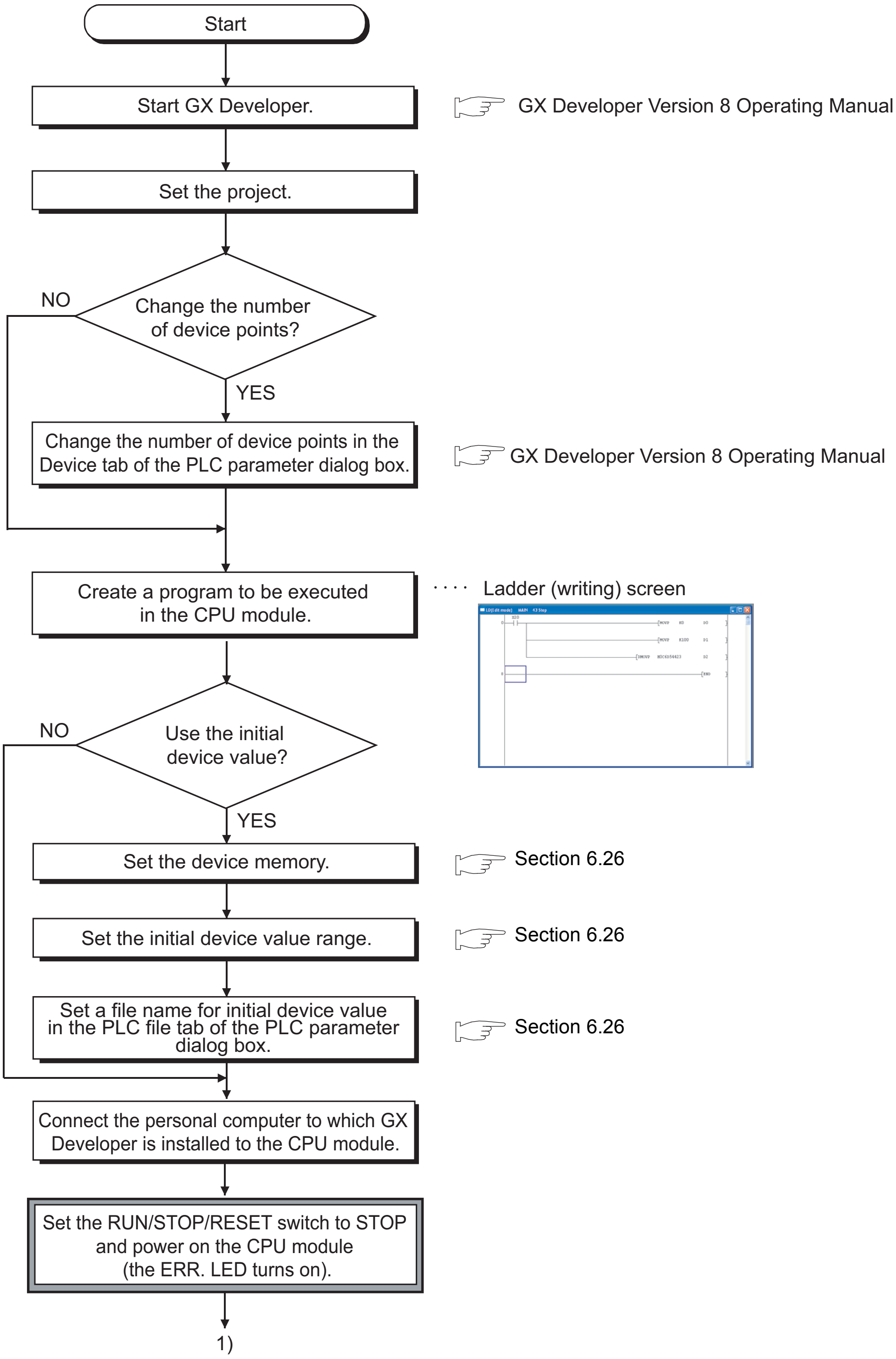
For installation and mounting procedures of the CPU module, refer to the following.

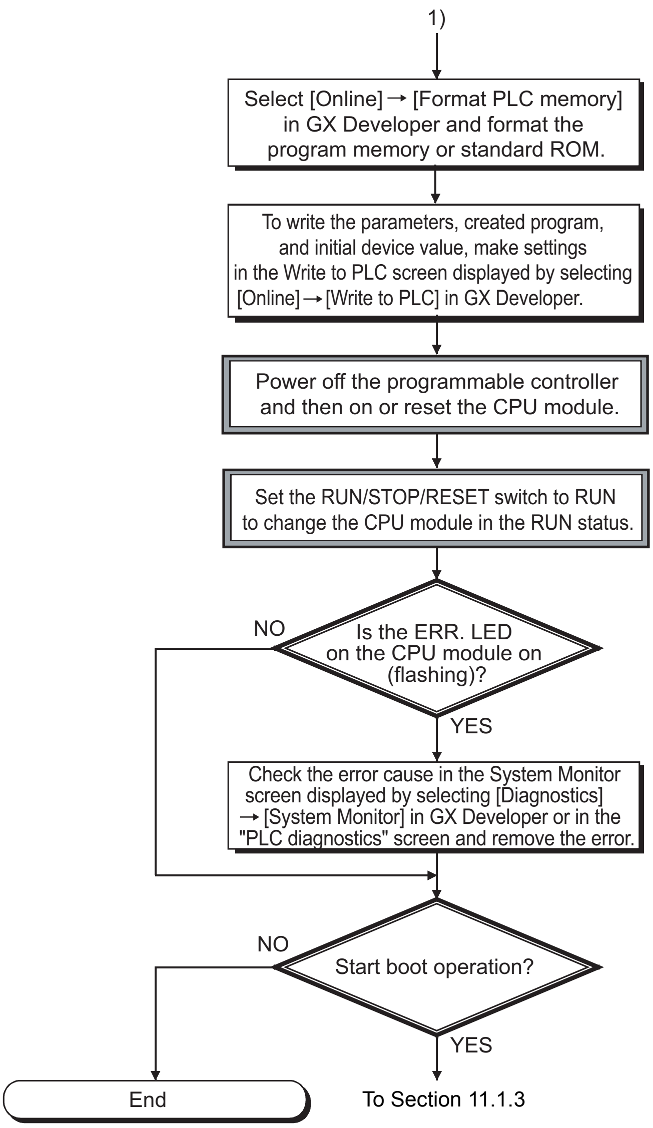
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

11.1.3 Procedure for writing a program

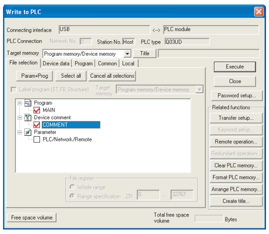
This section describes a procedure for writing a program to the program memory. (☞ Section 5.1.2)
Follow the procedure below and then the procedure provided in Section 11.1.4 before storing the program in the standard ROM for boot operation.

In the following procedure, ▭ indicates an operation on the CPU module side.


Start
Start GX Developer.
☞ GX Developer Version 8 Operating Manual
Set the project.
Change the number of device points?
NO
YES
Change the number of device points in the Device tab of the PLC parameter dialog box.
☞ GX Developer Version 8 Operating Manual
Create a program to be executed in the CPU module.
···· Ladder (writing) screen
Use the initial device value?
NO
YES
Set the device memory.
☞ Section 6.26
Set the initial device value range.
☞ Section 6.26
Set a file name for initial device value in the PLC file tab of the PLC parameter dialog box.
☞ Section 6.26
Connect the personal computer to which GX Developer is installed to the CPU module.
Set the RUN/STOP/RESET switch to STOP and power on the CPU module (the ERR. LED turns on).
1)


1)
Select [Online] → [Format PLC memory] in GX Developer and format the program memory or standard ROM.
To write the parameters, created program, and initial device value, make settings in the Write to PLC screen displayed by selecting [Online] → [Write to PLC] in GX Developer.
Power off the programmable controller and then on or reset the CPU module.
Set the RUN/STOP/RESET switch to RUN to change the CPU module in the RUN status.
Is the ERR. LED on the CPU module on (flashing)?
NO
YES
Check the error cause in the System Monitor screen displayed by selecting [Diagnostics] → [System Monitor] in GX Developer or in the "PLC diagnostics" screen and remove the error.
Start boot operation?
NO
YES
End
To Section 11.1.3

Write to PLC screen



Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q03UD

Target memory: Program memory/Device memory Title

□ Label program (ST,FB,Structure) Target memory Program memory/Device memory

- Program
  - ☑ MAIN
- Device comment
  - ☑ COMMENT
- Parameter
  - □ PLC/Network/Remote

File register: ○ Whole range ● Range specification ZR 0 .. 32767

Total free space volume Bytes

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

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

Figure 11.2 Flowchart for writing a program

11.1.4 Procedure for Boot Operation

This section describes a procedure for boot operation.

In the following procedure, ▭ indicates an operation on the CPU module side.

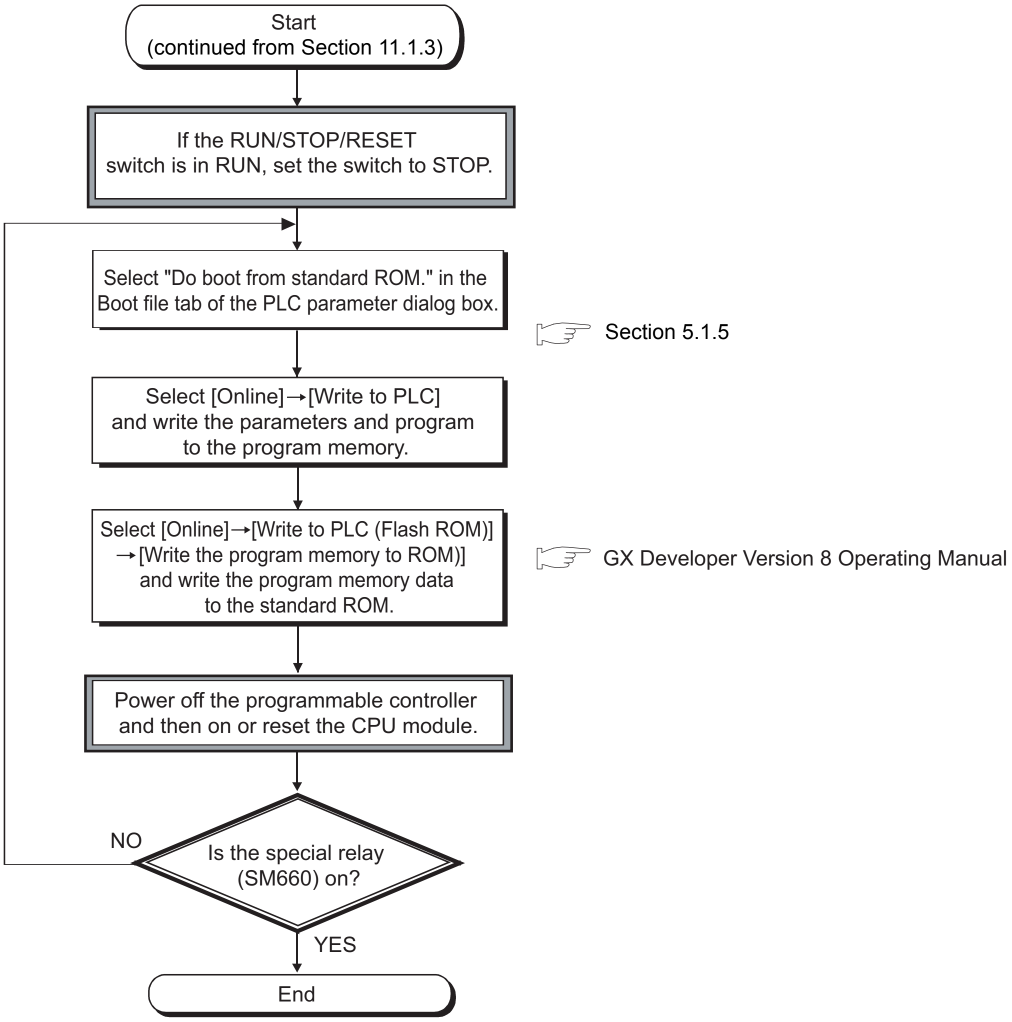

Start
(continued from Section 11.1.3)
If the RUN/STOP/RESET switch is in RUN, set the switch to STOP.
Select "Do boot from standard ROM." in the Boot file tab of the PLC parameter dialog box.
Section 5.1.5
Select [Online]→[Write to PLC] and write the parameters and program to the program memory.
Select [Online]→[Write to PLC (Flash ROM)] →[Write the program memory to ROM)] and write the program memory data to the standard ROM.
GX Developer Version 8 Operating Manual
Power off the programmable controller and then on or reset the CPU module.
Is the special relay (SM660) on?
NO
YES
End

Figure 11.3 Flowchart for boot operation

11.2 High Performance Model QCPU, Process CPU, and Redundant CPU

11.2.1 Items to be Considered for Creating Programs

To create a program, the number of device points, size, and file name of the program must be predetermined.

(1) Program size

Check whether the total size of programs and parameters are within the program size executable in the CPU module used. (☞ Section 5.4.3)
Table11.2 provides the program size executable in each CPU module.

Table11.2 Program size

CPU module	Program size
Q02CPU,Q02HCPU,Q02PHCPU	28k steps (112k bytes)
Q06HCPU,Q06PHCPU	60k steps (240k bytes)
Q12HCPU,Q12PHCPU,Q12PRHCPU	124k steps (496k bytes)
Q25HCPU,Q25PHCPU,Q25PRHCPU	252k steps (1008k bytes)

Whether parameters are stored to the program memory, standard ROM, or memory card can be set.
If the program size above is required for only programs, store the parameters in the standard ROM or memory card.

(2) Determining a unit for structuring the programs

When creating multiple programs, determine a unit (process/function) for structuring the programs

(3) Setting the execution conditions for programs to be created

When executing multiple programs, set their execution conditions to each program. (☞ Section 2.3)
Without the setting, the programs cannot be executed.

(4) Setting the applications of devices and the number of device points

Consider the applications of devices and the number of device points used in the program.
(☞ CHAPTER 9)

(5) Setting the initial device value

Set data necessary as an initial value to the device memory and the buffer memory of the intelligent function module. (☞ Section 6.27)

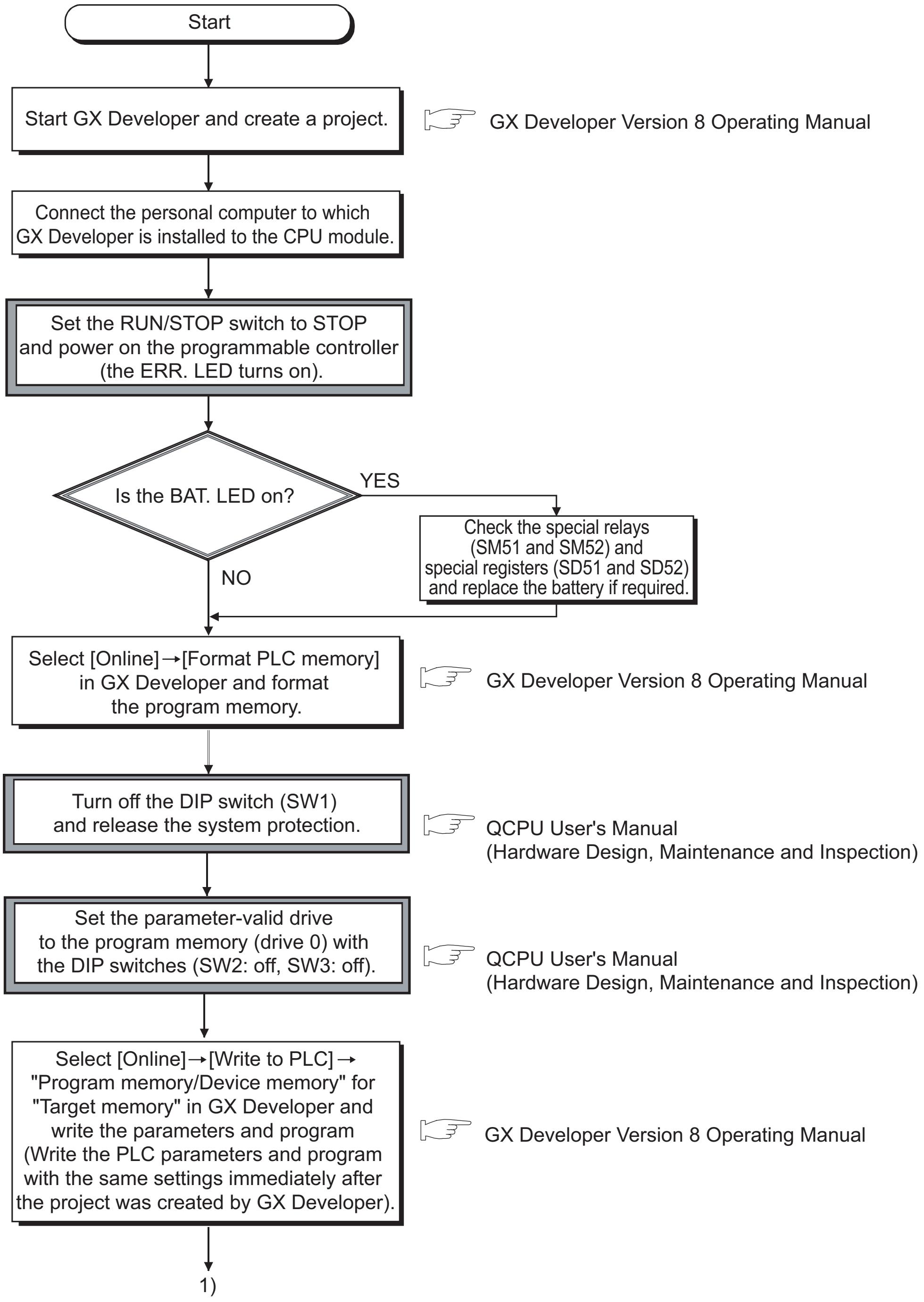
(6) Setting boot operation

When storing a program to the standard ROM or memory card, execute the program after boot operation.
For boot operation, make the setting in the Boot file tab of the PLC parameter dialog box.
(☞ Section 5.2.8, Section 11.2.5)

11.2.2 Hardware Check

This section describes a procedure for checking hardware before writing a created program.

In the following procedure, ▭ indicates an operation on the CPU module side.


Start
Start GX Developer and create a project.
GX Developer Version 8 Operating Manual
Connect the personal computer to which GX Developer is installed to the CPU module.
Set the RUN/STOP switch to STOP and power on the programmable controller (the ERR. LED turns on).
Is the BAT. LED on?
YES
Check the special relays (SM51 and SM52) and special registers (SD51 and SD52) and replace the battery if required.
NO
Select [Online]→[Format PLC memory] in GX Developer and format the program memory.
GX Developer Version 8 Operating Manual
Turn off the DIP switch (SW1) and release the system protection.
QCPU User's Manual (Hardware Design, Maintenance and Inspection)
Set the parameter-valid drive to the program memory (drive 0) with the DIP switches (SW2: off, SW3: off).
QCPU User's Manual (Hardware Design, Maintenance and Inspection)
Select [Online]→[Write to PLC]→ "Program memory/Device memory" for "Target memory" in GX Developer and write the parameters and program (Write the PLC parameters and program with the same settings immediately after the project was created by GX Developer).
GX Developer Version 8 Operating Manual
1)

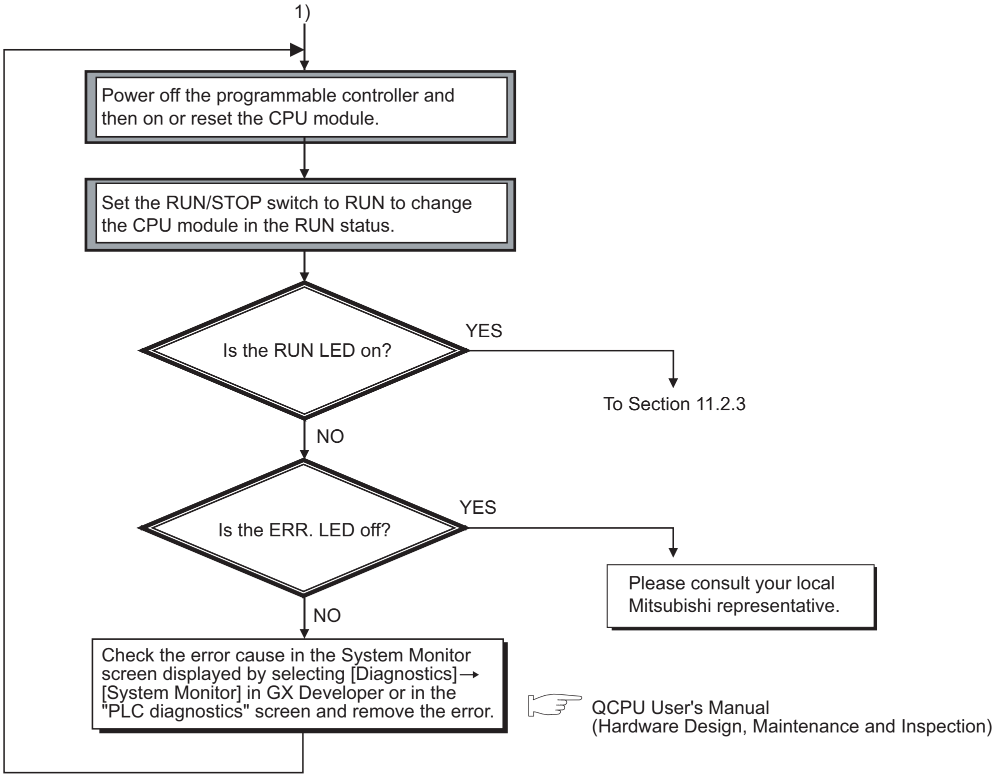

1)
Power off the programmable controller and then on or reset the CPU module.
Set the RUN/STOP switch to RUN to change the CPU module in the RUN status.
Is the RUN LED on?
YES
To Section 11.2.3
NO
Is the ERR. LED off?
YES
Please consult your local Mitsubishi representative.
NO
Check the error cause in the System Monitor screen displayed by selecting [Diagnostics]→[System Monitor] in GX Developer or in the "PLC diagnostics" screen and remove the error.
QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Figure 11.4 Hardware check flowchart

Remark

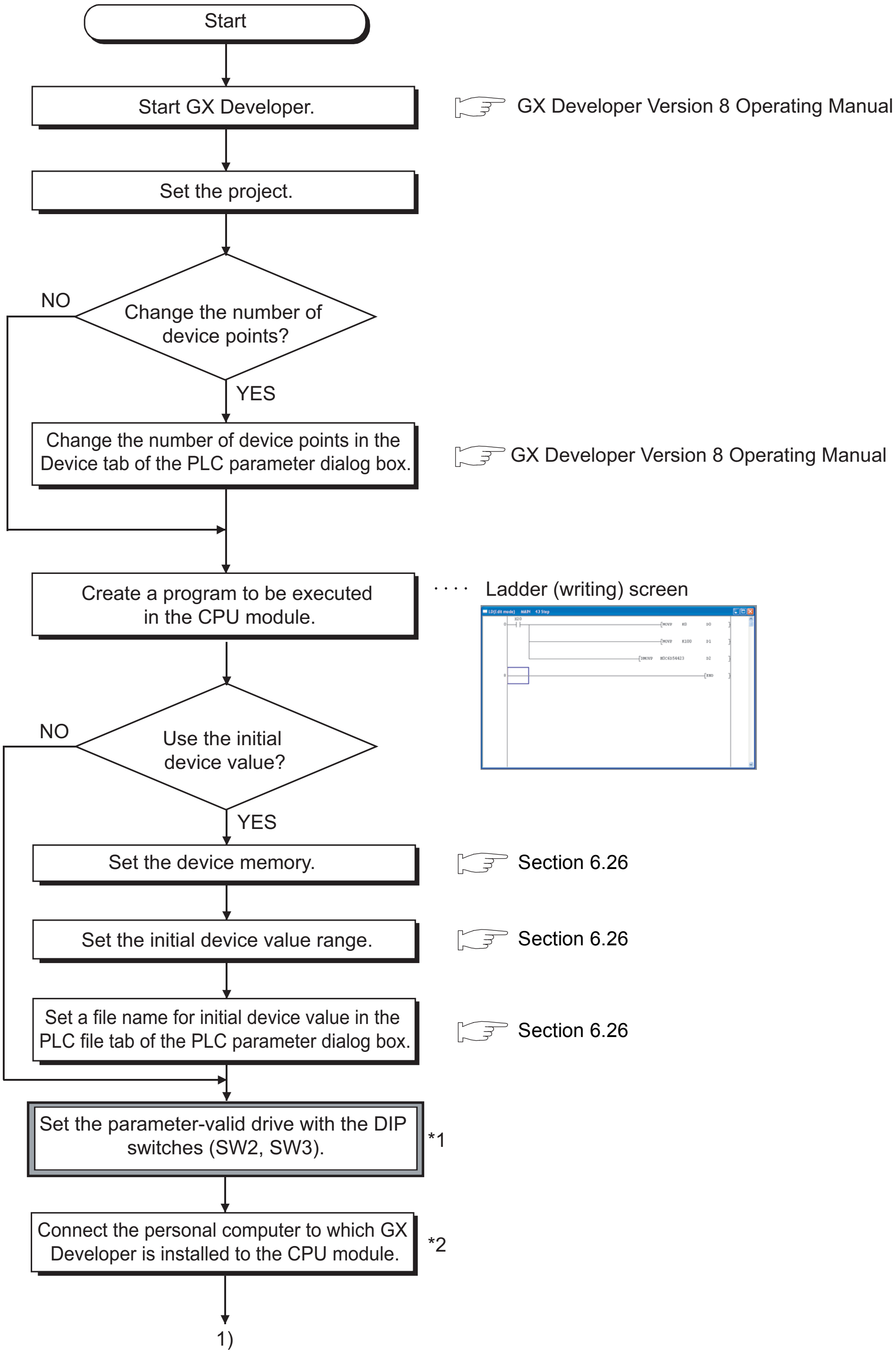
For installation and mounting procedures of the CPU module, refer to the following.

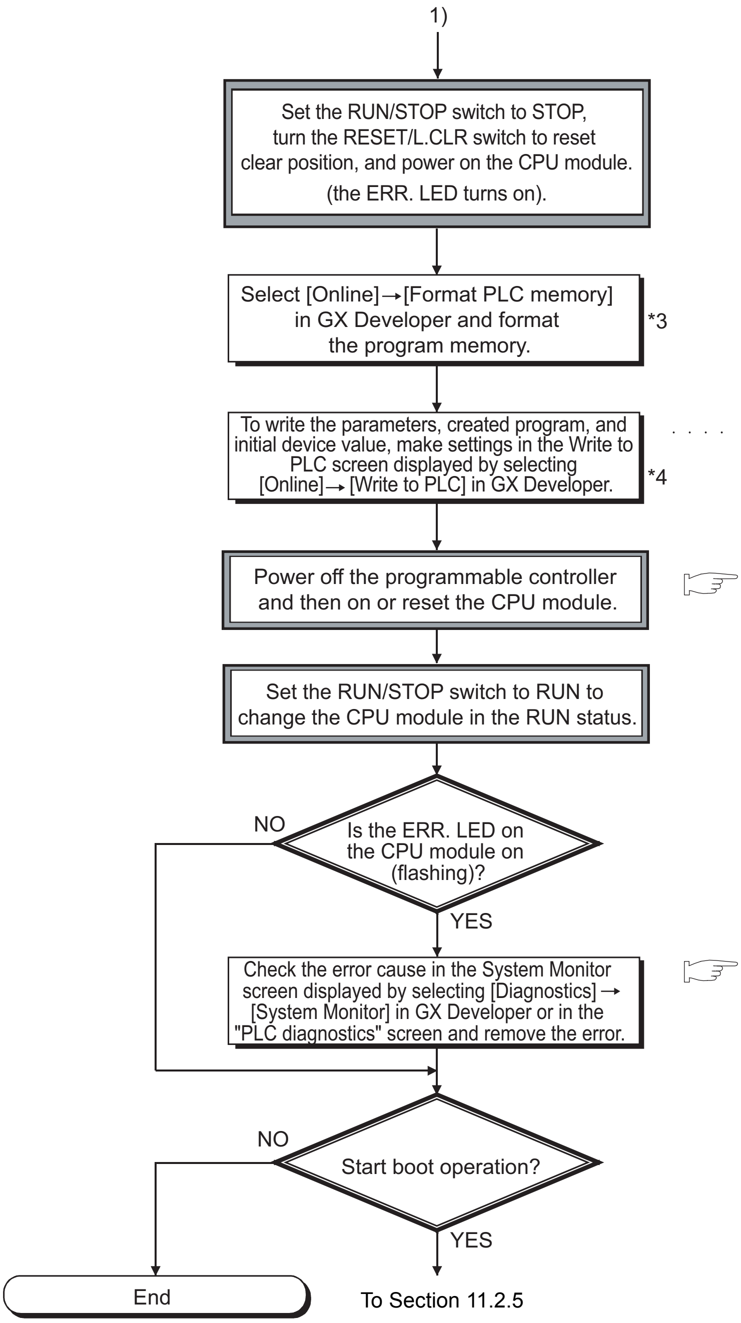
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

11.2.3 Procedure for Writing One Program

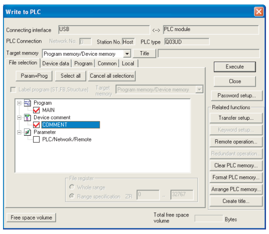
This section describes a procedure for writing a program to the program memory. (☞ Section 5.2.2)
Follow the procedure below and then the procedure provided in Section 11.2.5 before storing the program in the standard ROM or memory card for boot operation.

In the following procedure, ▭ indicates an operation on the CPU module side.


Start
Start GX Developer.
☞ GX Developer Version 8 Operating Manual
Set the project.
Change the number of device points?
NO
YES
Change the number of device points in the Device tab of the PLC parameter dialog box.
☞ GX Developer Version 8 Operating Manual
Create a program to be executed in the CPU module.
···· Ladder (writing) screen
Use the initial device value?
NO
YES
Set the device memory.
☞ Section 6.26
Set the initial device value range.
☞ Section 6.26
Set a file name for initial device value in the PLC file tab of the PLC parameter dialog box.
☞ Section 6.26
Set the parameter-valid drive with the DIP switches (SW2, SW3).
*1
Connect the personal computer to which GX Developer is installed to the CPU module.
*2
1)


1)
Set the RUN/STOP switch to STOP, turn the RESET/L.CLR switch to reset clear position, and power on the CPU module. (the ERR. LED turns on).
Select [Online]→[Format PLC memory] in GX Developer and format the program memory.
*3
To write the parameters, created program, and initial device value, make settings in the Write to PLC screen displayed by selecting [Online]→ [Write to PLC] in GX Developer.
*4
Power off the programmable controller and then on or reset the CPU module.
Set the RUN/STOP switch to RUN to change the CPU module in the RUN status.
Is the ERR. LED on the CPU module on (flashing)?
NO
YES
Check the error cause in the System Monitor screen displayed by selecting [Diagnostics]→ [System Monitor] in GX Developer or in the "PLC diagnostics" screen and remove the error.
Start boot operation?
NO
YES
End
To Section 11.2.5

Write to PLC screen


Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type: Q03UD

Target memory: Program memory/Device memory

□ Label program (ST,FB,Structure) Target memory: Program memory/Device memory

- Program
  - ☑ MAIN
- Device comment
  - ☑ COMMENT
- Parameter
  - □ PLC/Network/Remote

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume: Bytes

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

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

Figure 11.5 Flowchart for writing one program

*1: The parameter-valid drive is set to the program memory (drive 0) with the DIP switches by default (SW2: Off, SW3: On). To store parameters in the standard ROM or memory card, change the settings of the DIP switches.

*2: For procedures when the Redundant CPU is used, refer to the following.

QnPRHCPU User's Manual (Redundant System)

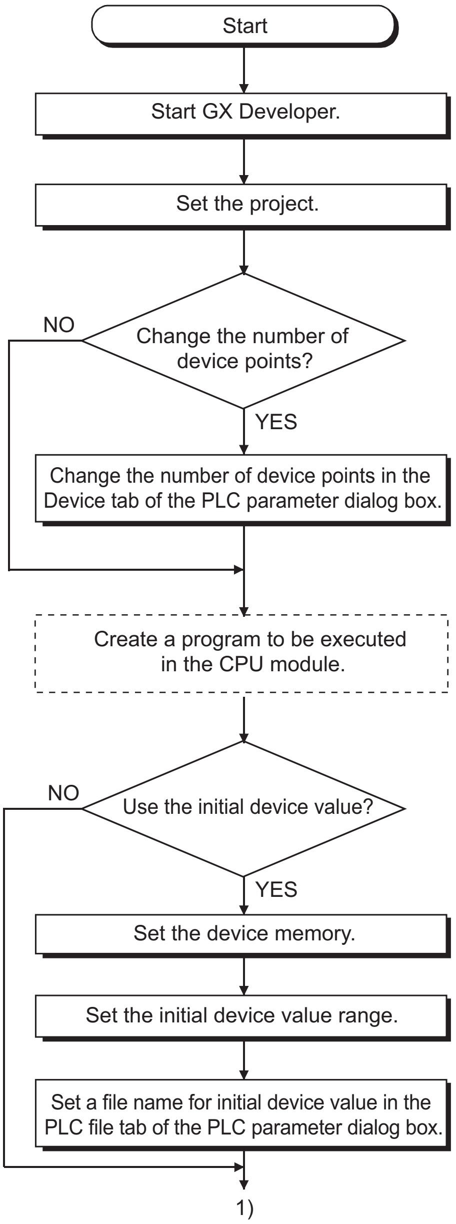
*3: When storing the file register and initial device value to the standard RAM or memory card (except the Flash card), format the memory used.

*4: Write each data in the memory as shown below:
- Program: Program memory
- Parameter: Memory set in the parameter-valid drive
- Initial device value: Memory set in the PLC file tab of the PLC parameter dialog box

11.2.4 Procedure for Writing Multiple Programs

This section describes a procedure for writing multiple programs to the program memory. (☞ Section 5.2.2) Follow the procedure below and then the procedure provided in Section 11.2.5 before storing the programs in the memory card for boot operation.

In the following procedure, □ indicates an operation on the CPU module side.

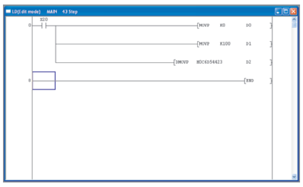


Start
Start GX Developer.
Set the project.
Change the number of device points?
NO
YES
Change the number of device points in the Device tab of the PLC parameter dialog box.
Create a program to be executed in the CPU module.
Use the initial device value?
NO
YES
Set the device memory.
Set the initial device value range.
Set a file name for initial device value in the PLC file tab of the PLC parameter dialog box.
1)

☞ GX Developer Version 8 Operating Manual

☞ GX Developer Version 8 Operating Manual

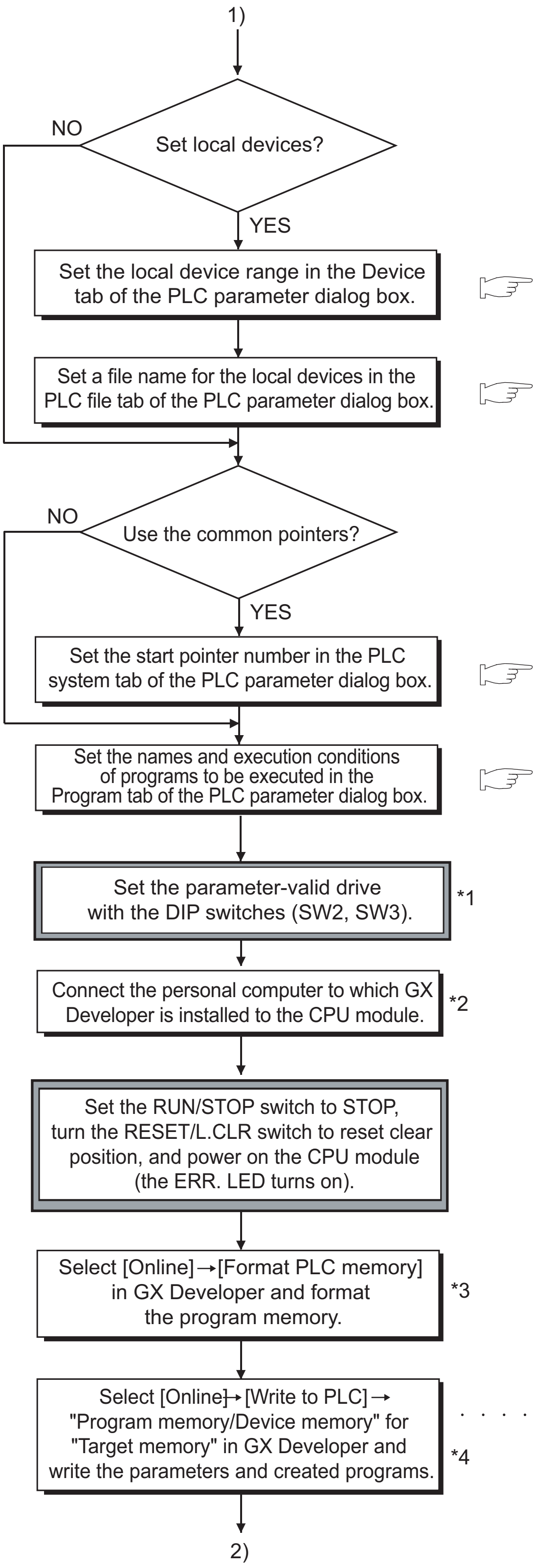
Ladder (writing) screen



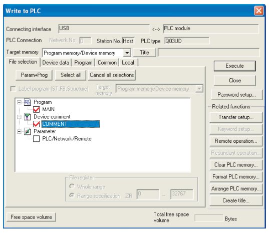
☞ Section 6.25

☞ Section 6.25

☞ Section 6.25


1)
Set local devices?
NO
YES
Set the local device range in the Device tab of the PLC parameter dialog box.
Section 9.13.2
Set a file name for the local devices in the PLC file tab of the PLC parameter dialog box.
Section 9.13.2
Use the common pointers?
NO
YES
Set the start pointer number in the PLC system tab of the PLC parameter dialog box.
Section 9.9
Set the names and execution conditions of programs to be executed in the Program tab of the PLC parameter dialog box.
Section 2.3
Set the parameter-valid drive with the DIP switches (SW2, SW3).
*1
Connect the personal computer to which GX Developer is installed to the CPU module.
*2
Set the RUN/STOP switch to STOP, turn the RESET/L.CLR switch to reset clear position, and power on the CPU module (the ERR. LED turns on).
Select [Online]→[Format PLC memory] in GX Developer and format the program memory.
*3
Select [Online]→[Write to PLC]→"Program memory/Device memory" for "Target memory" in GX Developer and write the parameters and created programs.
*4
2)

Write to PLC screen


Write to PLC

Connecting interface: USB <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q03UD

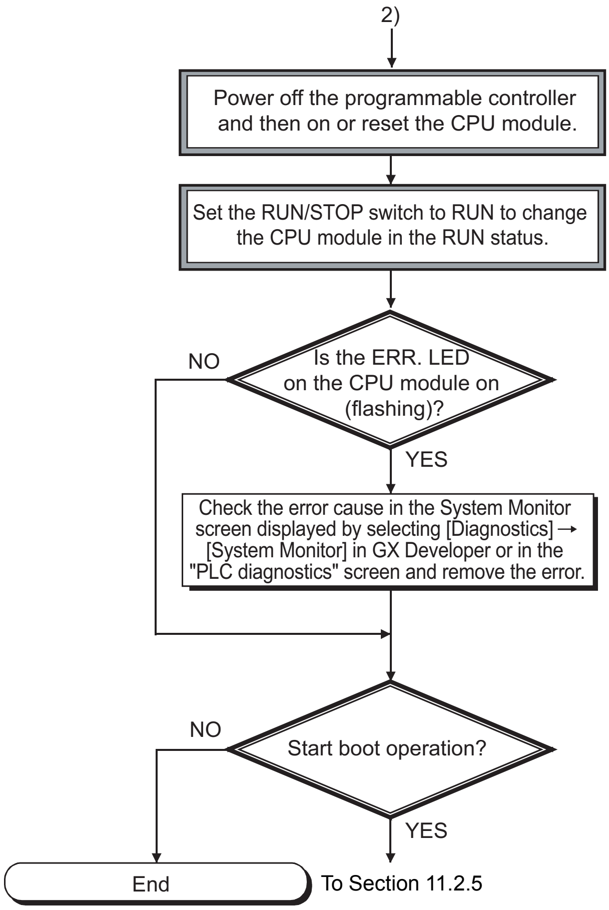
Target memory: Program memory/Device memory

☐ Label program (ST,FB,Structure) Target memory: Program memory/Device memory

- Program
  - ☑ MAIN
- Device comment
  - ☑ COMMENT
- Parameter
  - ☐ PLC/Network/Remote

File register: ○ Whole range ● Range specification ZR 0 -- 32767

Total free space volume Bytes


2)
Power off the programmable controller and then on or reset the CPU module.
Set the RUN/STOP switch to RUN to change the CPU module in the RUN status.
Is the ERR. LED on the CPU module on (flashing)?
NO
YES
Check the error cause in the System Monitor screen displayed by selecting [Diagnostics] → [System Monitor] in GX Developer or in the "PLC diagnostics" screen and remove the error.
Start boot operation?
NO
YES
End
To Section 11.2.5

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

Figure 11.6 Flowchart for writing multiple programs

*1: The parameter-valid drive is set to the program memory (drive 0) with the DIP switches by default (SW2: Off, SW3: On). To store parameters in the standard ROM or memory card, change the settings of the DIP switches.

*2: For procedures when the Redundant CPU is used, refer to the following.
QnPRHCPU User's Manual (Redundant System)

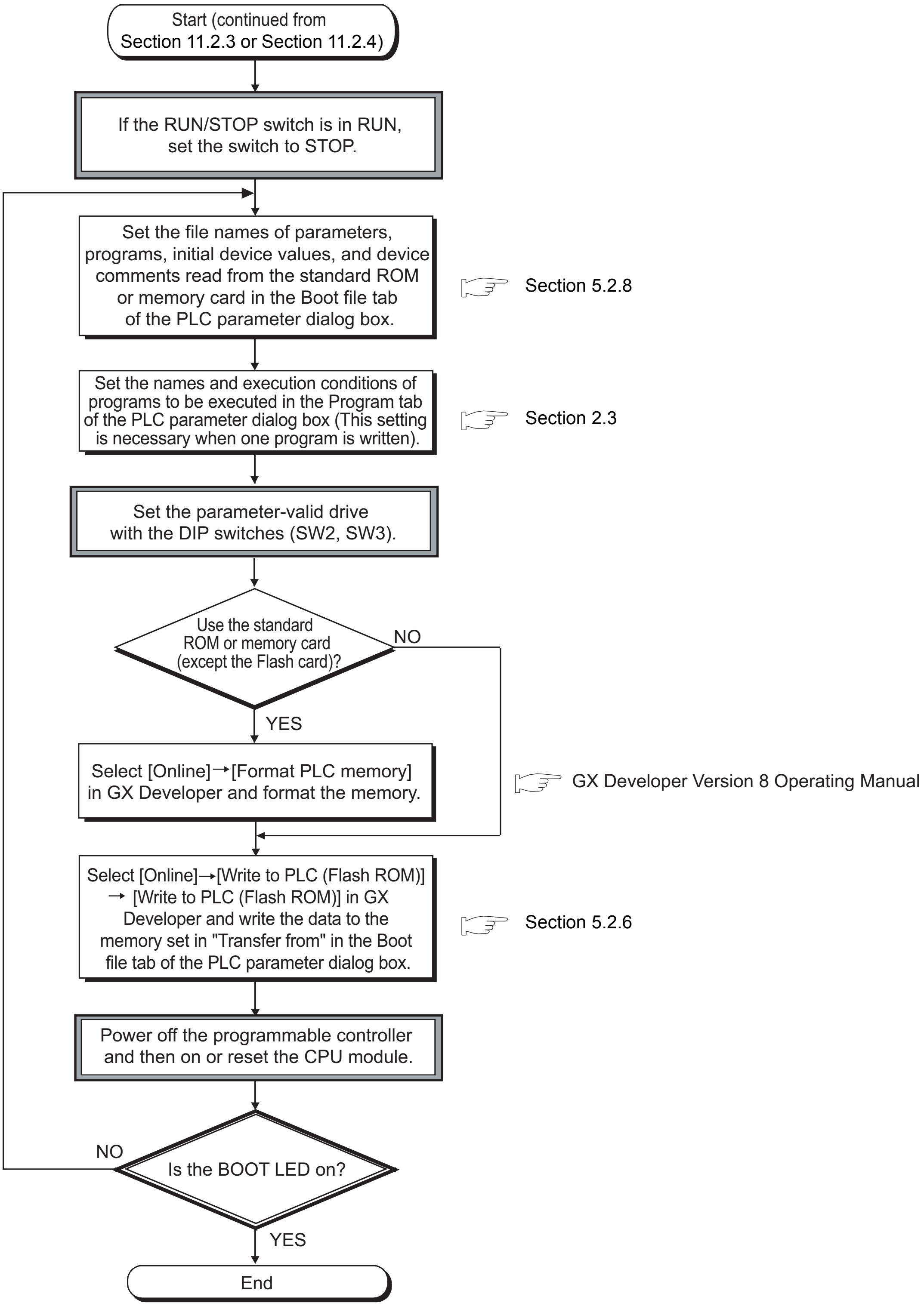
*3: When storing the file register and initial device value to the standard RAM or memory card (except the Flash card), format the memory used.

*4: Write each data in the memory as shown below:
- Program: Program memory
- Parameter: Memory set in the parameter-valid drive
- Initial device value: Memory set in the PLC file tab of the PLC parameter dialog box

11.2.5 Procedure for Boot Operation

This section describes a procedure for boot operation.

In the following procedure, ▭ indicates an operation on the CPU module side.



Start (continued from Section 11.2.3 or Section 11.2.4)
If the RUN/STOP switch is in RUN, set the switch to STOP.
Set the file names of parameters, programs, initial device values, and device comments read from the standard ROM or memory card in the Boot file tab of the PLC parameter dialog box.
☞ Section 5.2.8
Set the names and execution conditions of programs to be executed in the Program tab of the PLC parameter dialog box (This setting is necessary when one program is written).
☞ Section 2.3
Set the parameter-valid drive with the DIP switches (SW2, SW3).
Use the standard ROM or memory card (except the Flash card)?
NO
YES
Select [Online]→[Format PLC memory] in GX Developer and format the memory.
☞ GX Developer Version 8 Operating Manual
Select [Online]→[Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] in GX Developer and write the data to the memory set in "Transfer from" in the Boot file tab of the PLC parameter dialog box.
☞ Section 5.2.6
Power off the programmable controller and then on or reset the CPU module.
Is the BOOT LED on?
NO
YES
End

Figure 11.7 Flowchart for boot operation

APPENDICES

Appendix 1 List of Parameter Numbers

Each parameter number will be stored in the special register (SD16 to SD26) when an error occurs in the parameter settings.

TableAPPX.1 lists the parameter items and corresponding parameter numbers.

For explanation of mn, **, M, and N shown in the "Parameter No." column, refer to Section 8.3.

TableAPPX.1 List of parameter numbers

Item		Parameter No.	Reference
Label		0000_H	Section 8.1
Comment		0001_H	
I/O assignment	Type	0400_H	Section 4.2.2, Section 8.1
	Model name		
	Points		
	Start XY (Start I/O number)		
Basic setting	Base model name	0401_H	Section 4.1.2, Section 8.1
	Power model name		
	Extension cable		
	Slots		
Detailed setting	Error time output mode	0403_H	Section 6.8, Section 8.1
	I/O response time	0405_H	Section 6.7, Section 8.1
	Control PLC	0406_H	Section 8.1, QCPU User's Manual (Multiple CPU System)
Switch setting		0407_H	Section 6.10, Section 8.1,
Group No.		$05mn_H$	Section 8.3
		$0Amn_H$	Section 8.3
Redundant parameters		$0D00_H$	Section 8.1, QCPU User's Manual (Multiple CPU System)
No. of PLC		$0E00_H$	Section 8.1, QCPU User's Manual (Multiple CPU System)
Operating mode		$0E01_H$	
I/O sharing when using Multiple CPUs	All CPUs can read all inputs	$0E04_H$	
	All CPUs can read all outputs		
Timer limit setting	Low speed	1000_H	Section 8.1
	High speed		
RUN-PAUSE contacts	RUN	1001_H	Section 6.6.1, Section 8.1
	PAUSE		Section 6.6.2, Section 8.1
Remote reset		1002_H	Section 6.6.3, Section 8.1
Output mode at STOP to RUN		1003_H	Section 6.4, Section 8.1
Floating point arithmetic processing		1004_H	Section 2.4.4, Section 8.1.2
Common pointer No.		1005_H	Section 9.9.2, Section 8.1
Points occupied by empty slot		1007_H	Section 4.1.1, Section 8.1
Interrupt program/Fixed scan program setting		1008_H	Section 2.2.3, Section 2.3.5, Section 8.1
System interrupt settings	Interrupt counter start No.		Section 8.1
	Fixed scan interval (n: 28 to 31)		
Intelligent function module setting (Interrupt pointer setting)		$100A_H$	Section 9.10, Section 8.1
Module synchronization		$100C_H$	Section 8.1
A-PLC		$100D_H$	Section 8.1, Section 10.1.4

(To the next page)

TableAPPX.1 List of parameter numbers (continued)

Item		Parameter No.	Reference
Use serial communication		100E$_H$	Section 6.24, Section 8.1
Transmission speed			
Sum check			
Transmission wait time			
RUN write setting			
System interrupt settings (High speed interrupt setting)	X input	100F$_H$	Section 6.22, Section 8.1
	Y output	1010$_H$	
	Buffer read	1011$_H$	
	Buffer write	1012$_H$	
File register		1100$_H$	Section 9.7, Section 8.1
Comment file used in a command		1101$_H$	Section 8.1
Initial Device value		1102$_H$	Section 6.26, Section 8.1
File for local device		1103$_H$	Section 9.13.2, Section 8.1
Device points		2000$_H$	Section 9.1, Section 8.1
Latch (1) start/end		2001$_H$	Section 3.7, Section 6.3, Section 8.1
Latch (2) start/end		2002$_H$	
Local device start/end		2003$_H$	Section 9.13.2, Section 8.1
WDT (watchdog timer) setting	WDT setting	3000$_H$	Section 6.16, Section 8.1
	Initial execution monitoring time		Section 2.3.1, Section 8.1
Error check	Carry out battery check	3001$_H$	Section 6.17, Section 8.1
	Carry out fuse blown check		
	Verify module		
Operating mode when there is an error	Computation error	3002$_H$	
	Expanded command error		
	Fuse blown		
	Module verify error		
	Intelligent module program execution error		
	File access error		
	Memory card operation error		
	External power supply OFF		
Constant scanning		3003$_H$	Section 6.2, Section 8.1
Error history		3005$_H$	Section 6.18, Section 8.1
Low-speed program execution time		3006$_H$	Section 2.3.3, Section 8.1
Memory check	Check Program memory	3008$_H$	Section 6.27, Section 8.1
Detailed setting	H/W error time PLC operation mode	4004$_H$	Section 6.9, Section 8.1
Number of modules on MELSECNET/H		5000$_H$	Section 8.3
Valid module during other station access		5001$_H$	
Interlink transmission parameters		5002$_H$	
Routing parameters		5003$_H$	
Starting I/O No.		5NM0$_H$	
Network No.			
Total stations			
Mode		5NM0$_H$	

(To the next page)

TableAPPX.1 List of parameter numbers (continued)

Item		Parameter No.	Reference
Refresh parameters		5NM1$_{H}$	Section 8.3
Common parameters		5NM2$_{H}$	
Station inherent parameters		5NM3$_{H}$	
Sub-master parameters		5NM5$_{H}$	
Common parameters 2		5NMA$_{H}$	
Station inherent parameters 2		5NMB$_{H}$	
Interrupt settings			
Program		7000$_{H}$	Section 2.3, Section 8.1
Boot option	Clear program memory	7000$_{H}$	Section 5.2.8, Section 8.1
	Auto Download all Data from Memory card to Standard ROM		
Boot file setting			
SFC program start mode		8002$_{H}$	Section 8.1
Start conditions		8003$_{H}$	
Output mode when the block is stopped		8006$_{H}$	
Number of modules on Ethernet		9000$_{H}$	Section 8.3
Starting I/O No.		9N00$_{H}$	
Network No.			
Group No.			
Station No.			
Operational settings			
Initial settings		9N01$_{H}$	
Open settings		9N02$_{H}$	
Router relay parameter		9N03$_{H}$	
Station No.<->IP information		9N05$_{H}$	
FTP Parameters		9N06$_{H}$	
E-mail settings		9N07$_{H}$	
	News setting	9N08$_{H}$	
Interrupt settings		9N09$_{H}$	
Routing parameters		9N04$_{H}$	
Number of modules on CC-Link IE Controller Network		A000$_{H}$	Section 8.3
Interlink transmission parameters		A002$_{H}$	
Starting I/O No.		ANM0$_{H}$	
Network No.			
Total stations			
Station No.			
Mode		ANM0$_{H}$	
Refresh parameters		ANM1$_{H}$	
Common parameters		ANM2$_{H}$	
Station inherent parameters		ANM3$_{H}$	

(To the next page)

TableAPPX.1 List of parameter numbers (continued)

Item	Parameter No.	Reference
Number of Modules	C000$_H$	Section 8.3
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)		
Special Register (SW)		
Starting I/O No.	CNM2$_H$	
Operational setting		
All connect count		
Retry count	CNM2$_H$	Section 8.3
Automatic reconnection station count		
Standby master station No.		
PLC down select		
Scan mode setting		
Delay information setting		
Station information setting		
Remote device station initial setting		
Initial settings		
Interrupt setting		
Start mode setting	D001$_H$	Section 8.2, QnPRHCPU User's Manual (Redundant System)
Standby system watch setting		
Debug mode setting		
Backup mode setting		
Tracking characteristics setting	D002$_H$	
Tracking device settings	D003$_H$	
Signal flow memory tracking setting		
Device detail settings		
Tracking block No.		
Do auto forward Tracking block No.1 (Auto ON SM1520)		
Device range settings		
File register file settings		
Group settings	D004$_H$	Section 8.3
Redundant settings	D5**$_H$	
	D9**$_H$	
	DA**$_H$	
Communication area setting (refresh setting)	E002$_H$ E003$_H$	Section 8.1, QCPU User's Manual (Multiple CPU System)
Online module change	E006$_H$	

Appendix 2 Upgrade by Function Addition

The CPU module is upgraded when any function is added or specifications are changed.
Therefore, the functions and specifications can be used differ depending on the function version and serial number of the CPU module.

Appendix 2.1 Upgrade of the Basic model QCPU

(1) Specifications comparisons

TableAPPX.2 Specifications comparisons

Specifications		First 5 digits of serial No.	
		Function version A	Function version B
		"04121" or earlier	"04122" or later
Standard RAM capacity	Q00JCPU	×	
	Q00CPU	64K bytes	128K bytes
	Q01CPU	64K bytes	128K bytes
CPU shared memory	Q00JCPU	×	
	Q00CPU	×	○
	Q01CPU	×	○

○: Available/supported, ×: Not available/not supported

(2) Availability of new functions depending on the versions of the CPU module and GX Developer

TableAPPX.3 Availability of new functions depending on the versions of the CPU module and GX Developer

Function	Function version	First 5 digits of serial No.	GX Developer
Function block (☞ GX Developer Version 8 Operating Manual (Function Block))	A	"04121" or earlier	Version 8.00A or later
Structured text (ST) language (☞ MELSEC-Q/L Programming Manual (Structured Text))			
MELSAP3 (☞ MELSEC-Q/L/QnA Programming Manual (SFC))	B	"04122" or later	
PID operation*1 (☞ MELSEC-Q/L/QnA Programming Manual (PID Control Instructions))			
Real number operation*1 (☞ Section 9.12.3)			
Interrupt from intelligent function module (☞ Section 6.23)			
Initial device value automatic setting (☞ Section 6.26)			
Remote password setting (☞ Section 6.19.2)			
E-mail parameter (☞ Manual for the module that has the E-mail function)			
Online change with a pointer (☞ Section 6.15.2)			
Extension of file register R*2 (☞ Section 9.7)			-
Use in multiple CPU systems (☞ QCPU User's Manual (Multiple CPU System))*2			Version 8.00A or later
Online change (multiple blocks) (☞ Section 6.15.2)			
Use of CC-Link remote net additional mode (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)		"06112" or later	Version 8.03D or later
Communication with CC-Link IE Controller Network (☞ MELSEC-Q CC-Link IE Controller Network Reference Manual)	-	-	Version 8.68W or later

-: Not related to function version, serial No., or GX Developer

*1: When the CPU module instructions supported by GX Developer Version 8 are read by GX Developer Version 7 or earlier, they are processed as "instruction code error" by the GX Developer.

*2: The Q00JCPU does not support this function.

(3) Differences among the Basic model QCPU models

TableAPPX.4 Differences among the Basic model QCPU models

Item			Q00JCPU	Q00CPU	Q01CPU
CPU module			Integrated type of CPU module, power supply module, and main base unit (5 slots)	CPU module only	
Main base unit/slim type main base unit			Unnecessary	Necessary	
Extension base unit			Connectable (Cannot be connected to a slim type main base unit.)		
Number of extension bases			Up to 2 bases	Up to 4 bases (Cannot be connected to a slim type main base unit.)	
Number of mountable modules			16	24	
Power supply module					
	Main base unit		Unnecessary	Necessary	
	Slim type main base unit		Unnecessary	Necessary	
	Extension base unit	Q52B, Q55B	Unnecessary		
		Q63B, Q65B, Q68B, Q68RB, Q612B	Necessary		
Extension cable			QC05B, QC06B, QC12B, QC30B, QC50B, QC100B		
Memory card interface			None		
External interface	RS-232		Available (transmission speed: 9.6Kbps, 19.2Kbps, 38.4Kbps, 57.6Kbps, 115.2Kbps)		
	USB		None		
Processing speed (sequence instruction)	LD X0		0.20μs	0.16μs	0.10μs
	MOV D0 D1		0.70μs	0.56μs	0.35μs
Program size*1			8K steps (32K bytes)	8K steps (32K bytes)	14K steps (56K bytes)
Memory capacity	Program memory		58K bytes	94K bytes	
	Standard RAM		-	128K bytes*2	
	Standard ROM		58K bytes	94K bytes	
	CPU shared memory*3		None	1K byte (user setting area: 320 words)	
Number of writes to the standard ROM			Up to 100,000 times		
Device memory capacity			The number of device points can be changed within the range of 16.4K words.		
Number of I/O device points (including remote I/O)			2048 points		
Number of I/O points			256 points	1024 points	
File register			None	Available	
Serial communication function			None	Available (RS-232 interface of the CPU module is used.)	

*1: 1 step of the program size is 4 bytes.

*2: The memory capacity of the CPU module of function version A is 64K bytes.

*3: This memory is added to the CPU module of function version B.
Data in the CPU shared memory cannot be latched.
Data in the CPU shared memory will be cleared when the CPU module is powered off and then on or reset.

Appendix 2.2 Upgrade of the High Performance model QCPU

(1) Specifications comparisons

TableAPPX.5 Specifications comparisons

Specifications		First 5 digits of serial No.					
		Function version A		Function version B			
		"02091" or earlier	"02092" or later	"02112" or later	"03051" or later	"04012" or later	"16021" or later
Standard RAM capacity	Q02CPU	64K bytes					
	Q02HCPU	64K bytes				128K bytes	
	Q06HCPU	64K bytes				128K bytes	
	Q12HCPU	64K bytes	256K bytes				
	Q25HCPU	64K bytes	256K bytes				
CPU shared memory		×	×	○	○	○	○
Life extension of SRAM card battery		×	×	×	×	○	○
Support of 2M-byte SRAM card		×	×	×	×	○	○
Support of 4M-byte SRAM card		×	×	×	×	×	○

○: Available/supported, ×: Not available/not supported

(2) Availability of new functions depending on the versions of the CPU module and GX Developer

TableAPPX.6 Availability of new functions depending on the versions of the CPU module and GX Developer

Function	Function version	First 5 digits of serial No.	GX Developer
Automatic write to standard ROM (☞ Section 5.2.7)	A	"02092" or later	Version 6 or later
External input/output forced on/off (☞ Section 6.11.3)			
Remote password setting (☞ Section 6.19.2)			
Communication with MELSECNET/H remote I/O network (☞ Section 4.2.1)			
Use of interrupt modules (☞ Section 9.10)			
Use of programming modules (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection))			-
Use in multiple CPU systems (☞ QCPU User's Manual (Multiple CPU System))	B	"02122" or later	Version 7 or later
Use of the PC CPU module in multiple CPU systems (☞ QCPU User's Manual (Multiple CPU System))		"03051" or later	Version 7.10L or later
High-speed interrupt (☞ Section 6.22)		"04012" or later	Version 8 or later
Index modification with a device specifying an intelligent function module (☞ Manual for the intelligent function module that can use dedicated instructions)			-
Refresh item selection for the COM instruction (☞ MELSEC-Q/L Programming Manual (Common Instruction))			-

(To the next page)

TableAPPX.6 Availability of new functions depending on the versions of the CPU module and GX Developer (continued)

Function	Function version	First 5 digits of serial No.	GX Developer
Online change (files) of SFC program (☞ Section 6.12.2)	B	"04122" or later	Version 8 or later
File size unit (☞ Section 5.4.4)			
Use of CC-Link remote net additional mode (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual))		"05032" or later	Version 8.03D or later
Incomplete differentiation PID operation (☞ MELSEC-Q/L/QnA Programming Manual (PID Control Instructions))			
High-speed processing of floating-point data comparison instructions			-
Reading of the SFC active step comment (☞ MELSEC-Q/L/QnA Programming Manual (SFC))		"07012" or later	-
Detection of errors in the redundant power supply system (☞ Section 6.20)		"07032" or later	Version 8.23Z or later
Clock data in units of 1/1000 seconds (☞ Section 6.5)			-
Storage of sampling trace files in the standard RAM (☞ Section 6.14)			Version 8.23Z or later
Individual settings of refresh devices in multiple CPU systems (☞ QCPU User's Manual (Multiple CPU System))			
Selection of execution of the fall instruction during online change (☞ Section 6.12.3)		"07092" or later	Version 8.27D or later
"Block data assurance per station" setting for CC-Link (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)		"08032" or later	Version 8.32J or later
CC-Link parameter setting of up to 8 modules (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)			
Communication with CC-Link IE Controller Network (☞ MELSEC-Q CC-Link IE Controller Network Reference Manual)		"09012" or later	Version 8.45X or later
Change of ATA card (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection))			
Support of 4M-byte SRAM card (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection))		"16021" or later	-

-: Not related to GX Developer

Appendix 2.3 Upgrade of the Process CPU

(1) Availability of new functions depending on the versions of the CPU module and GX Developer

TableAPPX.7 Availability of new functions depending on the versions of the CPU module and GX Developer

Function	Function version	First 5 digits of serial No.	GX Developer
Index modification with a device specifying an intelligent function module (☞ Manual for the intelligent function module that can use dedicated instructions)	C	"07032" or later	-
Refresh item selection for the COM instruction (☞ MELSEC-Q/L Programming Manual (Common Instruction))			
Online change (files) of SFC program (☞ Section 6.12.2)			Version 8 (Version 8.22Y or earlier)
File size unit (☞ Section 5.4.4)			
Use of CC-Link remote net additional mode (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)			
Memory check function (☞ Section 6.26)			Version 8.23Z or later
Reading of the SFC active step comment (☞ MELSEC-Q/L/QnA Programming Manual (SFC))			-
Detection of errors in the redundant power supply system (☞ Section 6.20)			Version 8.23Z or later
Clock data in units of 1/1000 seconds (☞ Section 6.5)			-
Storage of sampling trace files in standard RAM (☞ Section 6.14)			Version 8.23Z or later
Individual settings of refresh devices in multiple CPU systems (☞ QCPU User's Manual (Multiple CPU System))			
Selection of execution of the fall instruction during online change (☞ Section 6.12.3)		"07092" or later	Version 8.27D or later
"Block data assurance per station" setting for CC-Link (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)		"08032" or later	Version 8.32J or later
CC-Link parameter setting of up to 8 modules (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)			
Communication with CC-Link IE Controller Network (☞ MELSEC-Q CC-Link IE Controller Network Reference Manual)		"10042" or later	Version 8.68W or later
Support of 4M-byte SRAM card (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection))		"16021" or later	-

-: Not related to GX Developer

Appendix 2.4 Upgrade of the Redundant CPU

(1) Availability of new functions depending on the versions of the CPU module and GX Developer

TableAPPX.8 Availability of new functions depending on the versions of the CPU module and GX Developer

Function	Function version	First 5 digits of serial No.	GX Developer
Reading of the SFC active step comment (☞ MELSEC-Q/L/QnA Programming Manual (SFC))	D	"07032" or later	-
Clock data in units of 1/1000 seconds (☞ Section 6.5)			
Storage of sampling trace files in standard RAM (☞ Section 6.14)			Version 8.23Z or later
Selection of execution of the fall instruction during online change (☞ Section 6.12.3)		"07092" or later	Version 8.27D or later
Use of extension base units (☞ QnPRHCPU User's Manual (Redundant System))		"09012" or later	Version 8.45X or later
CC-Link parameter setting of up to 8 modules (☞ MELSEC-Q CC-Link System Master/Local Module User's Manual)		"09102" or later	Version 8.58L or later
Communication with CC-Link IE Controller Network (☞ MELSEC-Q CC-Link IE Controller Network Reference Manual)		"10042" or later	Version 8.68W or later
Support of 4M-byte SRAM card (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection))		"16021" or later	-

-: Not related to GX Developer

Appendix 3 Device Point Assignment Sheet

(1) For the Basic model QCPU

TableAPPX.9 Device point assignment sheet

Device name	Sym-bol	Numeric notation	Number of device points *2		Restriction check	
			Points	Range	Size (words) *3	Points (bits)*2
Input relay*1	X	16	2K (2048)	X0000 to 07FF	/16 128	× 1 2048
Output relay*1	Y	16	2K (2048)	Y0000 to 07FF	/16 128	× 1 2048
Internal relay	M	10	K ( )	M0 to	/16	× 1
Latch relay	L	10	K ( )	L0 to	/16	× 1
Link relay	B	16	K ( )	B0000 to	/16	× 1
Annunciator	F	10	K ( )	F0 to	/16	× 1
Link special relay*1	SB	16	1K (1024)	SB0000 to 03FF	/16 64	× 1 1024
Edge relay	V	10	K ( )	V0 to	/16	× 1
Step relay *1	S	10	2K (2048)	S0 to 2047	/16 128	× 1 2048
Timer	T	10	K ( )	T0 to	$\times \frac{18}{16}$	× 2
Retentive timer	ST	10	K ( )	ST0 to	$\times \frac{18}{16}$	× 2
Counter	C	10	K ( )	C0 to	$\times \frac{18}{16}$	× 2
Data register	D	10	K ( )	D0 to	× 1	-
Link register	W	16	K ( )	W0000 to	× 1	-
Link special register*1	SW	16	1K (1024)	SW0000 to 03FF	× 1 1024	-
Total					(16704 or less)	

*1: The points are fixed for the system. (Cannot be changed)
*2: Up to 32K points can be set for each device.
*3: Enter the values multiplied (or divided) by the number shown in the Size (words) column.

(2) For the High Performance model QCPU, Process CPU, and Redundant CPU

TableAPPX.10 Device point assignment sheet

Device name	Sym-bol	Numeric nota-tion	Number of device points *2		Restriction check	
			Points	Range	Size (words)*3	Points (bits)*2
Input relay*1	X	16	8K (8192)	X0000 to 1FFF	/16 512	× 1 8192
Output relay*1	Y	16	8K (8192)	Y0000 to 1FFF	/16 512	× 1 8192
Internal relay	M	10	K ( )	M0 to	/16	× 1
Latch relay	L	10	K ( )	L0 to	/16	× 1
Link relay	B	16	K ( )	B0000 to	/16	× 1
Annunciator	F	10	K ( )	F0 to	/16	× 1
Link special relay*1	SB	16	2K (2048)	SB0000 to 07FF	/16 128	× 1 2048
Edge relay	V	10	K ( )	V0 to	/16	× 1
Step relay *1	S	10	8K (8192)	S0 to 8191	/16 512	× 1 8192
Timer	T	10	K ( )	T0 to	$\times\frac{18}{16}$	× 2
Retentive timer	ST	10	K ( )	ST0 to	$\times\frac{18}{16}$	× 2
Counter	C	10	K ( )	C0 to	$\times\frac{18}{16}$	× 2
Data register	D	10	K ( )	D0 to	× 1	-
Link register	W	16	K ( )	W0000 to	× 1	-
Link special register*1	SW	16	2K (2048)	SW0000 to 07FF	× 1 2048	-
Total					(29696 or less)	(65536 or less)

*1: The points are fixed for the system. (Cannot be changed)

*2: Up to 32K points can be set for each device.

*3: Enter the values multiplied (or divided) by the number shown in the Size (words) column.

INDEX

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)-080808ENG-M(1904)MEE
MODEL: QNCPU-U-KP-E
MODEL CODE: 13JZ28

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

When exported from Japan, this manual does not require application to the Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.