MELSEC Q series

Programmable Controller

User's Manual
(Function Explanation,
Program Fundamentals)

QCPU

01 07 2008
SH(NA)-080484ENG
Version M

MITSUBISHI ELECTRIC INDUSTRIAL AUTOMATION

QCPU

MITSUBISHI

User's Manual

(Function Explanation,
Program Fundamentals)

Q00JCPU
Q00CPU
Q01CPU
Q02CPU
Q02HCPU
Q06HCPU
Q12HCPU
Q25HCPU
Q02PHCPU
Q06PHCPU
Q12PHCPU
Q25PHCPU
Q12PRHCPU
Q25PRHCPU
Q02UCPU
Q03UD(E)CPU
Q04UD(E)HCPU
Q06UD(E)HCPU
Q13UD(E)HCPU
Q26UD(E)HCPU

Mitsubishi
Programmable Controller

MELSEC-Q

●SAFETY PRECAUTIONS●

(Always read these instructions before using this equipment.)

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

In this manual, the safety instructions are ranked as "DANGER" and "CAUTION".

DANGER	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 medium or slight personal injury or physical damage.

Note that the ⚠CAUTION level may lead to a serious consequence according to the circumstances.
Always follow the instructions of both levels because they are important to personal safety.

Please save this manual to make it accessible when required and always forward it to the end user.

[Design Precautions]

⚠DANGER

● Install a safety circuit external to the PLC that keeps the entire system safe even when there are problems with the external power supply or the PLC module. Otherwise, trouble could result from erroneous output or erroneous operation.

(1) Outside the PLC, construct mechanical damage preventing interlock circuits such as emergency stop, protective circuits, positioning upper and lower limits switches and interlocking forward/ reverse operations.

(2) When the PLC detects the following problems,
it will stop calculation and turn off all output in the case of (a).
In the case of (b), it will hold or turn off all output according to the parameter setting.
Note that the AnS/A series module will turn off the output in either of cases (a) and (b).

	Q series module	AnS/A series module
(a) The power supply module has over currentprotection equipment and over voltage protection equipment.	Output OFF	Output OFF
(b) The CPU module self-diagnosis functions, such as the watchdog timer error, detect problems.	Hold or turn off all output according to the parameter setting.	Output OFF

In addition, all output will be turned on when there are problems that the PLC CPU cannot detect, such as in the I/O controller. Build a fail safe circuit exterior to the PLC that will make sure the equipment operates safely at such times.
Refer to "LOADING AND INSTALLATION" in QCPU User's Manual (Hardware Design, Maintenance and Inspection) for example fail safe circuits.

(3) Output could be left on or off when there is trouble in the outputs module relay or transistor. So build an external monitoring circuit that will monitor any single outputs that could cause serious trouble.

[Design Precautions]

⚠DANGER

- When overcurrent which exceeds the rating or caused by short-circuited load flows in the output module for a long time, it may cause smoke or fire. To prevent this, configure an external safety circuit, such as fuse.

- Build a circuit that turns on the external power supply when the PLC main module power is turned on.
  If the external power supply is turned on first, it could result in erroneous output or erroneous operation.

- When there are communication problems with the data link, refer to the corresponding data link manual for the operating status of each station.
  Not doing so could result in erroneous output or erroneous operation.

- When connecting a peripheral device to the CPU module or connecting a personal computer or the like to the intelligent function module / special function module to exercise control (data change) on the running PLC, configure up an interlock circuit in the sequence program to ensure that the whole system will always operate safely.
  Also before exercising other control (program change, operating status change (status control)) on the running PLC, read the manual carefully and fully confirm safety.
  Especially for the above control on the remote PLC from an external device, an immediate action may not be taken for PLC trouble due to a data communication fault.
  In addition to configuring up the interlock circuit in the sequence program, corrective and other actions to be taken as a system for the occurrence of a data communication fault should be predetermined between the external device and PLC CPU.

⚠CAUTION

- Do not bunch the control wires or communication cables with the main circuit or power wires, or install them close to each other.
  They should be installed 100 mm (3.94 inch) or more from each other.
  Not doing so could result in noise that would cause erroneous operation.

- When controlling items like lamp load, heater or solenoid valve using an output module, large current (approximately ten times greater than that present in normal circumstances) may flow when the output is turned OFF to ON.
  Take measures such as replacing the module with one having sufficient rated current.

[Installation Precautions]

⚠CAUTION

- Use the PLC in an environment that meets the general specifications contained in QCPU User's Manual (Hardware Design, Maintenance and Inspection).
  Using this PLC in an environment outside the range of the general specifications could result in electric shock, fire, erroneous operation, and damage to or deterioration of the product.

- While pressing the installation lever located at the bottom of module, insert the module fixing tab into the fixing hole in the base unit until it stops. Then, securely mount the module with the fixing hole as a supporting point.
  Incorrect loading of the module can cause a malfunction, failure or drop.
  When using the PLC in the environment of much vibration, tighten the module with a screw.
  Tighten the screw in the specified torque range.
  Undertightening can cause a drop, short circuit or malfunction.
  Overtightening can cause a drop, short circuit or malfunction due to damage to the screw or module.

- When installing extension cables, be sure that the base unit and the extension module connectors are installed correctly.
  After installation, check them for looseness.
  Poor connections could cause an input or output failure.

- Securely load the memory card into the memory card loading connector.
  After installation, check for lifting.
  Poor connections could cause an operation fault.

- Completely turn off the externally supplied power used in the system before mounting or removing the module. Not doing so could result in damage to the product.Note that the module can be changed online (while power is on) in the system that uses the CPU module compatible with online module change or on the MELSECNET/H remote I/O station.
  Note that there are restrictions on the modules that can be changed online(while power is on), and each module has its predetermined changing procedure.
  For details, refer to QCPU User's Manual (Hardware Design, Maintenance and Inspection) and the online module change section in the manual of the module compatible with online module change.

- Do not directly touch the module's conductive parts or electronic components.
  Touching the conductive parts could cause an operation failure or give damage to the module.

[Wiring Precautions]

⚠DANGER

- ● Completely turn off the externally supplied power used in the system when installing or placing wiring.
  Not completely turning off all power could result in electric shock or damage to the product.

- ● When turning on the power supply or operating the module after installation or wiring work, be sure that the module's terminal covers are correctly attached.
  Not attaching the terminal cover could result in electric shock.

⚠DANGER

● Be sure to ground the FG terminals and LG terminals to the protective ground conductor.
Not doing so could result in electric shock or erroneous operation.

● Use applicable solderless terminals and tighten them with the specified torque.If any solderless spade terminal is used, it may be disconnected when the terminal screw comes loose, resulting in failure.

● When wiring in the PLC, be sure that it is done correctly by checking the product's rated voltage and the terminal layout.
Connecting a power supply that is different from the rating or incorrectly wiring the product could result in fire or damage.

● External connections shall be crimped or pressure welded with the specified tools, or correctly soldered.
Imperfect connections could result in short circuit, fires, or erroneous operation.

● Tighten the terminal screws with the specified torque.
If the terminal screws are loose, it could result in short circuits, fire, or erroneous operation.
Tightening the terminal screws too far may cause damages to the screws and/or the module, resulting in fallout, short circuits, or malfunction.

● Be sure there are no foreign substances such as sawdust or wiring debris inside the module.
Such debris could cause fires, damage, or erroneous operation.

● The module has an ingress prevention label on its top to prevent foreign matter, such as wire offcuts, from entering the module during wiring.
Do not peel this label during wiring.
Before starting system operation, be sure to peel this label because of heat dissipation.

● Before starting system operation, be sure to peel this label because of heat dissipation. Install our PLC in a control panel for use. Wire the main power supply to the power supply module installed in a control panel through a distribution terminal block. Furthermore, the wiring and replacement of a power supply module have to be performed by a maintenance worker who acquainted with shock protection. (QCPU User's Manual (Hardware Design, Maintenance and Inspection))

[Startup and Maintenance Precautions]

◇! DANGER

- Do not touch the terminals while power is on.
  Doing so could cause shock or erroneous operation.

- Correctly connect the battery.
  Also, do not charge, disassemble, heat, place in fire, short circuit, or solder the battery.
  Mishandling of battery can cause overheating or cracks which could result in injury and fires.

- Switch off all phases of the externally supplied power used in the system when cleaning the module or retightening the terminal or module mounting screws.
  Not doing so could result in electric shock.
  Undertightening of terminal screws can cause a short circuit or malfunction.
  Overtightening of screws can cause damages to the screws and/or the module, resulting in fallout, short circuits, or malfunction.

[Startup and Maintenance Precautions]

⚠CAUTION

- The online operations conducted for the CPU module being operated, connecting the peripheral device (especially, when changing data or operation status), shall be conducted after the manual has been carefully read and a sufficient check of safety has been conducted.
  Operation mistakes could cause damage or problems with of the module.

- Do not disassemble or modify the modules.
  Doing so could cause trouble, erroneous operation, injury, or fire.

- Use any radio communication device such as a cellular phone or a PHS phone more than 25cm (9.85 inch) away in all directions of the PLC.
  Not doing so can cause a malfunction.

- Completely turn off the externally supplied power used in the system before mounting or removing the module. Not doing so could result in damage to the product.
  Note that the module can be changed online (while power is on) in the system that uses the CPU module compatible with online module change or on the MELSECNET/H remote I/O station.
  Note that there are restrictions on the modules that can be changed online (while power is on) , and each module has its predetermined changing procedure.
  For details, refer to QCPU User's Manual (Hardware Design, Maintenance and Inspection) and the online module change section in the manual of the module compatible with online module change.

- Do not mount/remove the module to/from the base unit or terminal block more than 50 times (IEC 61131-2 compliant), after the first use of the product.
  Failure to do so may cause module malfunctions.

- Do not drop or give an impact to the battery mounted to the module.
  Doing so may damage the battery, causing the battery fluid to leak inside the battery.
  If the battery is dropped or given an impact, dispose of it without using.

- Before touching the module, always touch grounded metal, etc. to discharge static electricity from human body, etc.
  Not doing so can cause the module to fail or malfunction.

[Disposal Precautions]

⚠CAUTION

● When disposing of this product, treat it as industrial waste.

[Transportation Precautions]

⚠CAUTION

● When transporting lithium batteries, make sure to treat them based on the transport regulations. (Refer to Appendix 8 for details of the controlled models.)

REVISIONS

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

Print Date	Manual Number	Revision
Jun., 2004	SH(NA)-080484ENG-A	First edition
Dec., 2004	SH(NA)-080484ENG-B	Partial correction Section 3.1.3, 4.7.2, 6.7, 6.11.1, 6.12.2, 6.14, 6.17, 8.3, 9.13.1, 10.1.1, 10.1.2, 11.1.2, 11.1.3, 11.2.2, 11.2.3, 11.2.4, Appendix 1, Appendix 2, Appendix 3.1
May, 2005	SH(NA)-080484ENG-C	Addition Appendix 3, Appendix 4.3, Appendix 4.4 Change Change Appendix 3 to 5 to Appendix 4 to 6. Partial correction Section 1.2, 1.4, Chapter 2, Section 5.2.1, 5.2.4, 5.4.3, 5.4.4, 6.5, 6.12.2, 6.14, 6.17, 6.17.1, 6.20, 6.21.2, Chapter 8, Section 8.1.2, 9.2.10, 9.3.2, 9.3.3, 9.10.1, 9.13.1 10.1.2, 10.1.3, Appendix 1, Appendix 2
Aug., 2005	SH(NA)-080484ENG-D	Addition Section 6.27 Partial correction GENERIC TERMS AND ABBREVIATIONS, Chapter 2, Section 3.9.3, 5.1.1, 5.1.4, 9.1, Appendix 1, Appendix 2
Oct.,2005	SH(NA)-080484ENG-E	Addition Section 6.12.3 Partial correction Section 6.15.2, 8.1.2, 10.1.3, Appendix 4
Mar.,2006	SH(NA)-080484ENG-F	Model addition QA65B,QA68B Partial correction GENERIC TERMS AND ABBREVIATIONS, Section 1.1.2, 4.3, 4.7.2, 5.2.7, 5.4.3, 5.4.4, 6.17, 6. 21.2, 6.22.2, 6.22.4, 8.2, 10.1.2, Appendix 2, Appendix 4.3, Index
Nov., 2006	SH(NA)-080484ENG-G	Partial correction SAFETY PRECAUTION, Section 1.3, 3.3, 4.4, 4.7.2, 5.4.3, 6.2, 6.4, 6.6.4, 6.11.3, 6.21.2, 8.3, 9.2.1, 9.3.1, 9.11.3, 9.12.3, Appendix 4.3

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

Print Date	Manual Number	Revision
Apr., 2007	SH(NA)-080484ENG-H	Universal model QCPU model addition, Revision involving High Performance model QCPU and Redundant CPU serial No.09012 Model addition Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q65WRB Addition Section 1.1.5, 5.2.3, 5.2.11, 6.25.1, 6.25.2, 6.29, 6.30, 9.6, 9.6.1, 9.6.2, Appendix 5 Change Section 5.2.3 → 5.2.4, 5.2.4 → 5.2.5, 5.2.5 → 5.2.6, 5.2.6 → 5.2.7, 5.2.7 → 5.2.8, 5.2.8 → 5.2.9, 5.2.9 → 5.2.10, 6.27 → 6.28, Appendix 5 → Appendix 6, Appendix 6 → Appendix 7 Partial correction ABOUT MANUALS, GENERIC TERMS AND ABBREVIATIONS, Chapter 1, Section 1.1.2, 1.1.4, 1.2, 1.3, 1.4, Chapter 2, Section 3.1.1, 3.1.2, 3.1.3, 3.3, 3.3.3, 3.3.4, 3.3.5, 3.3.6, 3.4.1, 3.4.2, 3.4.4, 3.5, 3.7, 3.8, 3.9.1, 3.9.2, 3.9.4, 4.1, 4.2, 4.3, 4.4, 4.6.1, 4.6.2, 5.2.1, 5.2.2, 5.2.4, 5.2.5, 5.2.6, 5.2.7, 5.2.8, 5.2.9, 5.4.1, 5.4.2, 5.4.3, 5.4.4, 6.1, 6.3, 6.4, 6.5, 6.6, 6.6.1, 6.7, 6.10, 6.11, 6.11.1, 6.11.2, 6.11.3, 6.12, 6.12.1, 6.12.2, 6.12.3, 6.13.1, 6.13.3, 6.14, 6.15, 6.15.2, 6.17, 6.17.1, 6.18.2, 6.19, 6.19.1, 6.19.2, 6.20, 6.21, 6.21.1, 6.21.2, 6.22, 6.21.1, 6.21.2, 6.21.3, 6.21.4, 6.24, 6.25, 6.26, 6.28, Chapter 7, Section 7.1, 7.1.2, 7.1.4, 7.2, Chapter 8, Section 8.1.2, 8.2, 8.3, 9.1, 9.2, 9.2.1, 9.2.4, 9.2.5, 9.2.7, 9.2.8, 9.2.9, 9.2.10, 9.2.11, 9.2.13, 9.2.14, 9.3.1, 9.3.2, 9.3.3, 9.4, 9.5.1, 9.6.3, 9.6.4, 9.7, 9.10.1, 9.11.1, 9.11.2, 9.11.3, 9.12.2, 9.12.3, 9.13.1, 10.1.1, 10.1.2, 10.1.3, 10.1.4, 11.2, 11.2.1, 11.2.2, 11.2.3, 11.2.4, 11.2.5, Appendix 1, Appendix 2, Appendix 3, Appendix 4.2, Appendix 4.4, Appendix 6
May, 2007	SH(NA)-080484ENG-I	Partial correction 6.12.3, 9.2.7, 9.2.13, Appendix 5
Jan., 2008	SH(NA)-080484ENG-J	Revision and overall reexamination for compatibility with Redundant CPU serial No.09102 Model addition QA6ADP
Mar., 2008	SH(NA)-080484ENG-K	Universal model QCPU model addition, Revision involving Universal model QCPU serial No.10012 Model addition Q13UDHCPU, Q26UDHCPU Partial correction GENERIC TERMS AND ABBREVIATIONS, Chapter 1, Section 1.1.5, Chapter 2, Section 4.1, 4.2, 4.3, 5.2.1, 5.2.3, 5.4.3, 5.4.4, 6.12.3, 6.14, 7.1.1, 8.1.2, 10.1.2, 10.1.3, 10.1.4, 10.2, 11.2.1, 11.2.4, Appendix 1, Appendix 2, Appendix 3, Appendix 4.1, Appendix 4.4, Appendix 4.5, Appendix 5.1, Appendix 5.2, Appendix 5.3, Appendix 5.4.1, Appendix 5.4.3, Appendix 5.4.4, Appendix 5.4.5, Appendix 5.5.1, Appendix 5.5.2 Addition Section 9.13.1, 9.13.2

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

Print Date	Manual Number	Revision
May, 2008	SH(NA)-080484ENG-L	Process CPU and Universal model QCPU model addition, Revision involving Process CPU, Redundant CPU and Universal model QCPU serial No.10042 Model addition Q02PHCPU, Q06PHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU Addition Section2.1, 2.2, 2.3, 2.4, 2.5, 6.11.4, Chapter 7, Appendix 6, Appendix 7, Section10.8 Change Chapter 7 → Chapter 8, Chapter 8 → Chapter 9, Chapter 9 → Chapter 10, Chapter 10 → Chapter 11, Chapter 11 → Chapter 12, Appendix 6 → Appendix 8, Appendix 7 → Appendix 9, Section 9.8 to 9.13 → Section 10.9 to 10.14 Partial correction A term "MELSECNET/G" has been revised to "CC-Link IE controller network" through this manual. GENERIC TERMS AND ABBREVIATIONS, Chapter 1, Section 1.3, 1.4, Chapter 2, Section 3.1.3, 3.2, 3.3.1, 3.3.2, 3.3.3, 3.3.4, 3.3.5, 3.3.6, 3.4.1, 3.4.4, 3.5, 3.8, 4.1, 4.2, 4.3, 5.1.5, 5.2.1, 5.2.3, 5.2.7, 5.4.3, 5.4.4, 6.1, 6.2, 6.3, 6.5, 6.6.1, 6.6.3, 6.5, 6.6.3, 6.11, 6.11.1, 6.11.3, 6.11.4, 6.12.3, 6.13.3, 6.14, 6.15, 6.15.1, 6.17, 6.19.1, 6.19.2, 6.20, 6.22.3, 6.22.4, 6.26, 6.29, 6.30, Chapter7, Chapter 9, Section 9.1.1, 9.1.2, 9.3, 9.4, 10.1, 10.2, 10.2.1, 10.2.5, 10.2.7, 10.2.8, 10.2.13, 10.2.14, 10.4, 10.7.2, 10.7.4, 10.10, 10.13.2, Chapter 11, Section 11.1.2, 11.1.3, 11.1.4, 11.2, 12.2.1, Appendix 1, Appendix 2, Appendix 3, Appendix 4, Appendix 4.1, Appendix 4.3, Appendix 4.4, Appendix 4.5, Appendix 5.1, Appendix 5.2, Appendix 5.4, Appendix 5.5, Appendix 6, Appendix 7
Jul., 2008	SH(NA)-080484ENG-M	Partial correction Section 10.8

Japanese Manual Version SH-080473-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

Thank you for choosing the Mitsubishi MELSEC-Q Series of General Purpose Programmable Controllers.
Before using the equipment, please read this manual carefully to develop full familiarity with the functions and performance of the Q series PLC you have purchased, so as to ensure correct use.

CONTENTS

CHAPTER11 CPU MODULE PROCESSING TIME 11 - 1 to 11 - 32

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

CONTENTS

CHAPTER11 MAINTENANCE AND INSPECTION

CHAPTER12 TROUBLESHOOTING

APPENDICES

INDEX

(Related manual).................QCPU User's Manual (Multiple CPU System)

CONTENTS

APPENDICES

INDEX

(Related manual)................QnPRHCPU User's Manual (Redundant System)

CONTENTS

CHAPTER7 PROGRAMMING CAUTIONS

CHAPTER8 TROUBLESHOOTING

CHAPTER9 PROCESSING TIME FOR REDUNDANT SYSTEMS

APPENDICES

INDEX

ABOUT MANUALS

The following manuals are also related to this product.
In necessary, order them by quoting the details in the tables below.

Related Manuals

(1) Common to CPU modules

The following table indicates the related manuals common to the Basic model QCPU, High Performance model QCPU, Process CPU and Redundant CPU.

Manual Name	Manual Number (Model Code)
QCPU User's Manual (Hardware Design, Maintenance and Inspection) This manual provides the specifications of the CPU modules, power supply modules, base units, extension cables, memory cards and others. (Sold separately)	SH-080483ENG (13JR73)
QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions) This manual describes how to use the sequence instructions and application instructions. (Sold separately)	SH-080039 (13JF58)
QCPU (Q Mode)/QnACPU Programming Manual (SFC) This manual explains the system configuration, performance specifications, functions, programming, debugging, error codes and others of MELSAP3. (Sold separately)	SH-080041 (13JF60)
QCPU (Q Mode) Programming Manual (MELSAP-L) This manual describes the programming methods, specifications functions, and so on that are necessary to create the MELSAP-L type SFC program. (Sold separately)	SH-080076 (13JF61)
QCPU (Q Mode) Programming Manual (Structured Text) This manual describes the structured text language programming methods. (Sold separately)	SH-080366E (13JF68)

(2) Basic model QCPU

The following table indicates the related manuals of the Basic model QCPU other than the manuals indicated in "(1) Common to CPU modules".

Manual Name	Manual Number (Model Code)
QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions) This manual describes the dedicated instructions used to exercise PID control. (Sold separately)	SH-080040 (13JF59)
QCPU User's Manual (Multiple CPU System) This manual explains the multiple CPU system overview, system configuration, I/O numbers, communication between CPU modules, and communication with the I/O modules or intelligent function modules. (Sold separately)	SH-080485ENG (13JR75)
Q Corresponding MELSEC Communication Protocol Reference Manual This manual explains the communication methods and control procedures through the MC protocol for the external devices to read and write data from/to the CPU module using the serial communication module/Ethernet module. (Sold separately)	SH-080008 (13JF89)

(3) High Performance model QCPU

The following table indicates the related manuals of the High Performance model QCPU other than the manuals indicated in "(1) Common to CPU modules".

Manual Name	Manual Number (Model Code)
QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions) This manual describes the dedicated instructions used to exercise PID control. (Sold separately)	SH-080040 (13JF59)
QCPU User's Manual (Multiple CPU System) This manual explains the multiple CPU system overview, system configuration, I/O numbers, communication between CPU modules, and communication with the I/O modules or intelligent function modules. (Sold separately)	SH-080485ENG (13JR75)

(4) Process CPU

The following table indicates the related manuals of the Process CPU other than the manuals indicated in "(1) Common to CPU modules".

Manual Name	Manual Number (Model Code)
QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions) This manual describes the programming procedures, device names, and other items necessary to implement PID control using process control instructions. (Sold separately)	SH-080316E (13JF67)
QCPU User's Manual (Multiple CPU System) This manual explains the multiple CPU system overview, system configuration, I/O numbers, communication between CPU modules, and communication with the I/O modules or intelligent function modules. (Sold separately)	SH-080485ENG (13JR75)

(5) Redundant CPU

The following table indicates the related manuals of the Redundant CPU other than the manuals indicated in "(1) Common to CPU modules".

Manual Name	Manual Number (Model Code)
QnPRHCPU User's Manual (Redundant System) This manual explains the redundant system configuration, functions, communication with external devices, and troubleshooting for redundant system construction using the Redundant CPU. (Sold separately)	SH-080486ENG (13JR76)
QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions) This manual describes the dedicated instructions used to exercise PID control. (Sold separately)	SH-080040 (13JF59)
QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions) This manual describes the programming procedures, device names, and other items necessary to implement PID control using process control instructions. (Sold separately)	SH-080316E (13JF67)

(6) Universal model QCPU

The following table indicates the related manuals of the Universal model QCPU other than the manuals indicated in "(1) Common to CPU modules".

Manual Name	Manual Number (Model Code)
QCPU User’s Manual (Multiple CPU System) This manual explains the multiple CPU system overview, system configuration, I/O numbers, communication between CPU modules, and communication with the I/O modules or intelligent function modules. (Sold separately)	SH-080485ENG (13JR75)
QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions) This manual describes the dedicated instructions used to exercise PID control. (Sold separately)	SH-080040 (13JF59)
Q Corresponding MELSEC Communication Protocol Reference Manual This manual explains the communication methods and control procedures through the MC protocol for the external devices to read and write data from/to the CPU module using the serial communication module/ Ethernet module. (Sold separately)	SH-080008 (13JF89)

HOW TO SEE THIS MANUAL IS ORGANIZED

CPU modules requiring precautions

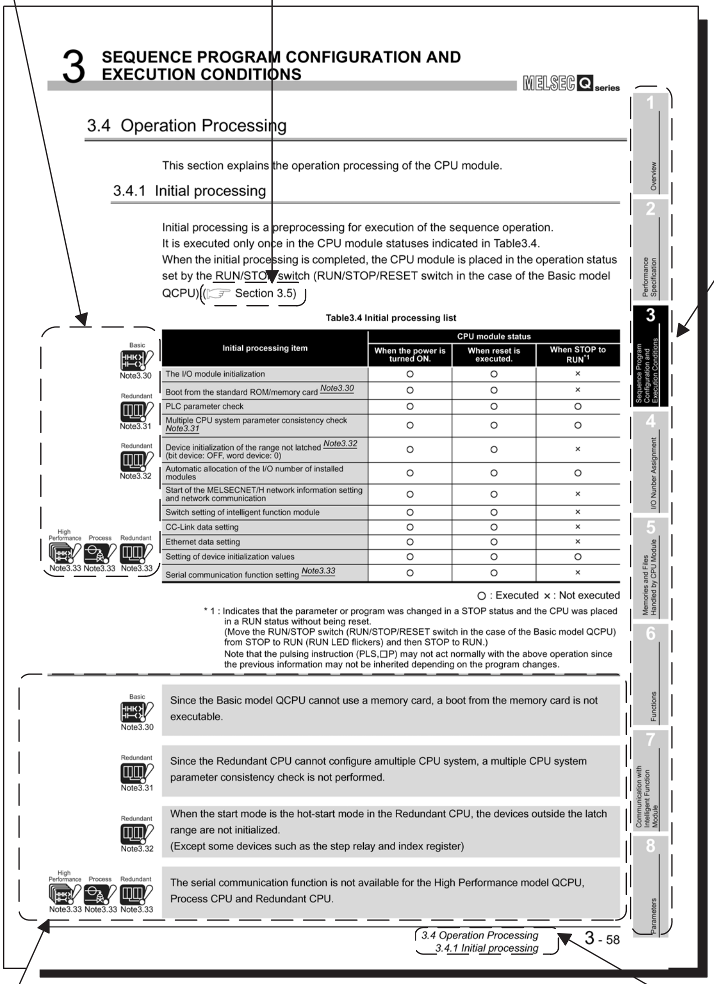
The CPU modules requiring precautions are shown as icons.
"Note ●.▲" under the icon corresponds to "Note ●.▲" in the sentences and at the page bottom. However, "Note ●.▲" is not described in the section title.

Reference destination

A reference destination or reference manual is marked ☞ .

Chapter heading

The index on the right side of the page shows the chapter of the open page at a glance.



MELSEC Q series
1 Overview
2 Performance Specification
3 Sequence Program Configuration and Execution Conditions
4 I/O Nunber Assignment
5 Memories and Files Handled by CPU Module
6 Functions
7 Communication with Intelligent Function Module
8 Parameters
Basic Note3.30
Redundant Note3.31
Redundant Note3.32
High Performance Process Redundant
Note3.33 Note3.33 Note3.33

3 SEQUENCE PROGRAM CONFIGURATION AND EXECUTION CONDITIONS

3.4 Operation Processing

This section explains the operation processing of the CPU module.

3.4.1 Initial processing

Initial processing is a preprocessing for execution of the sequence operation.
It is executed only once in the CPU module statuses indicated in Table3.4.
When the initial processing is completed, the CPU module is placed in the operation status set by the RUN/STOP switch (RUN/STOP/RESET switch in the case of the Basic model QCPU)(☞ Section 3.5)

Table3.4 Initial processing list

Initial processing item	CPU module status		
	When the power is turned ON.	When reset is executed.	When STOP to RUN[*1]
The I/O module initialization	○	○	×
Boot from the standard ROM/memory card Note3.30	○	○	×
PLC parameter check	○	○	○
Multiple CPU system parameter consistency check Note3.31	○	○	○
Device initialization of the range not latched Note3.32 (bit device: OFF, word device: 0)	○	○	×
Automatic allocation of the I/O number of installed modules	○	○	○
Start of the MELSECNET/H network information setting and network communication	○	○	×
Switch setting of intelligent function module	○	○	×
CC-Link data setting	○	○	×
Ethernet data setting	○	○	×
Setting of device initialization values	○	○	○
Serial communication function setting Note3.33	○	○	×

○ : Executed × : Not executed

* 1 : Indicates that the parameter or program was changed in a STOP status and the CPU was placed in a RUN status without being reset.
(Move the RUN/STOP switch (RUN/STOP/RESET switch in the case of the Basic model QCPU) from STOP to RUN (RUN LED flickers) and then STOP to RUN.)
Note that the pulsing instruction (PLS,□P) may not act normally with the above operation since the previous information may not be inherited depending on the program changes.

Since the Basic model QCPU cannot use a memory card, a boot from the memory card is not executable.

Since the Redundant CPU cannot configure amultiple CPU system, a multiple CPU system parameter consistency check is not performed.

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

The serial communication function is not available for the High Performance model QCPU, Process CPU and Redundant CPU.

3.4 Operation Processing
3.4.1 Initial processing

3 - 58

Precautions

Precautions corresponding to the icons are provided.

Section title

The section of the open page is shown at a glance.

Icon					Description
Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU	
Basic	High Performance	Process	Redundant	Universal	The ! marked icon indicates the CPU module does not support a part of the described functions.
Basic	High Performance	Process	Redundant	Universal	The × marked icon indicates the CPU module does not support all of the described functions.

In addition, this manual provides the following explanations.

☒POINT

Explains the matters to be especially noted, the functions and others related to the description.

Remark

Provides the reference destination related to the description on that page and the convenient information.

HOW TO USE THIS MANUAL

This manual is prepared for users to understand memory map, functions, programs, and devices of the CPU modules required for the operation of the Q series programmable controller.

This manual consists of several CHAPTERs as shown below.

1) CHAPTER 1 Describes the overview of the CPU modules.

2) CHAPTER 2 to 5 Describe the performance specifications, executable programs, I/O numbers, and memory of the CPU modules.

3) CHAPTER 6 Describes the functions of the CPU modules.

4) CHAPTER 7 Describes the functions using built-in Ethernet ports of the Built-in Ethernet port QCPU.

5) CHAPTER 8 Describes communication with intelligent function modules.

6) CHAPTER 9 and 10 Describe the parameters and devices of the CPU modules.

7) CHAPTER 11 Describes the processing time in the CPU modules.

8) CHAPTER 12 Describes the procedure for writing parameter settings and programs from GX Developer to the CPU modules.

Remark

This manual does not explain the functions of power supply modules, base units, extension cables, memory cards and batteries of CPU module.
For details, refer to the following manual.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

For the multiple CPU system, refer to the following manual.
☞ QCPU User's Manual (Multiple CPU System)

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

GENERIC TERMS AND ABBREVIATIONS

Unless otherwise specified, this manual uses the following generic terms and abbreviations to explain the Q series CPU modules.

Generic Term/Abbreviation	Description
Basic model QCPU	Generic term for Q00JCPU, Q00CPU, and Q01CPU
High Performance model QCPU	Generic term for Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
Process CPU	Generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU, and Q25PHCPU
Redundant CPU	Generic term for Q12PRHCPU, and Q25PRHCPU
Universal model QCPU	Generic term for Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, and Q26UDEHCPU
Built-in Ethernet port QCPU	Generic term for Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, and Q26UDEHCPU
QnHCPU	Generic term for Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
Qn(H)CPU	Generic term for Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU
QnPHCPU	Generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU and Q25PHCPU
QnPRHCPU	Generic term for Q12PRHCPU and Q25PRHCPU
QnUCPU	Generic term for Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, and Q26UDEHCPU
QnU(D)(H)CPU	Generic term for Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, and Q26UDHCPU
QnUDE(H)CPU	Generic term for Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, and Q26UDEHCPU
CPU module	Generic term for Basic model QCPU, High performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU
Q series	Abbreviation for Mitsubishi MELSEC-Q Series Programmable Controller
AnS series	Another name for compact types of Mitsubishi MELSEC-A Series Programmable Controller
A series	Another name for large types of Mitsubishi MELSEC-A Series Programmable Controller
GX Developer	Product name for Q series compatible SW□D5C-GPPW-E type GPP function software package □ indicates the version. For the GX Developer versions applicable for each CPU module, refer to "System configuration" in the QCPU User's Manual (Hardware Design, Maintenance and Inspection).
PX Developer	Product name for SW□D5C-FBDQ process control FBD software package. □ indicates the version. For the PX Developer versions applicable for each CPU module, refer to "System configuration" in the QCPU User's Manual (Hardware Design, Maintenance and Inspection).
Q3□B	Generic term for Q33B, Q35B, Q38B, and Q312B main base units on which CPU module (except Q00JCPU), Q series power supply module, I/O module and intelligent function module can be mounted
Q3□SB	Generic term for Q32SB, Q33SB, and Q35SB slim type main base units on which Basic model QCPU (except Q00JCPU) High Performance model QCPU, slim type power supply module, I/O module and intelligent function module can be mounted

Generic Term/Abbreviation	Description
Q3□RB	Another name for Q38RB main base unit for redundant power supply system, that CPU modules (except the Q00JCPU), redundant power supply modules, Q series I/O modules, and intelligent function modules can be mounted
Q3□DB	Generic term for Q38DB and Q312DB type Multiple CPU high speed main base unit on which CPU module (except Q00JCPU), Q series power supply module, Q series I/O module and intelligent function module can be mounted
Q5□B	Generic term for Q52B and Q55B extension base unit on which the Q Series I/O and intelligent function module can be mounted
Q6□B	Generic term for Q63B, Q65B, Q68B, and Q612B extension base unit on which Q Series power supply module, I/O module, intelligent function module can be mounted
Q6□RB	Another name for Q68RB extension base unit for redundant power supply system, that redundant power supply modules, Q series I/O modules, and intelligent function modules can be mounted
Q6□WRB	Another name for Q65WRB extension base unit for redundant system, that redundant power supply modules, Q series I/O modules, and intelligent function modules can be mounted
QA1S6□B	Generic term for QA1S65B and QA1S68B extension base units on which with AnS Series power supply module, I/O module and special function module can be mounted
QA6□B	Generic term for QA65B and QA68B extension base units on which the A series power supply module, I/O modules and special function modules can be mounted
A5□B	Generic term for A52B, A55B, and A58B extension base units on which A series I/O module and special function module can be mounted without power supply
A6□B	Generic term for A62B, A65B, and A68B extension base units on which A series I/O module and special function module can be mounted
QA6ADP	Abbreviation for QA6ADP QA conversion adapter module
QA6ADP+A5□B/A6□B	Abbreviation for A large type extension base unit on which QA6ADP is mounted
Main base unit	Generic term for Q3□B, Q3□SB, Q3□RB, and Q3□DB
Extension base unit	Generic term for Q5□B, Q6□B, Q6□RB, Q6□WRB, QA1S6□B, QA6□B, and QA6ADP+A5□B/A6□B
Slim type main base unit	Another name for Q3□SB
Redundant power main base unit	Another name for Q3□RB
Redundant power extension base unit	Another name for Q6□RB
Redundant type extension base unit	Another name for Q6□WRB
Multiple CPU high speed main base unit	Another name for Q3□DB
Base unit	Generic term for main base unit, extension base unit, slim type main base unit, redundant power main base unit, redundant power extension base unit, redundant type extension base unit, and multiple CPU high speed main base unit
Redundant base unit	Abbreviation for redundant power main base unit, redundant power extension base unit, and redundant type extension base unit
Extension cable	Generic term for QC05B, QC06B, QC12B, QC30B, QC50B, QC100B extension cables
Tracking cable	Generic term for QC10TR and QC30TR tracking cable for Redundant CPU
Q series power supply module	Generic term for Q61P-A1, Q61P-A2, Q61P, Q62P, Q63P, Q64P, and Q64PN power supply modules
Slim type power supply module	Abbreviation for Q61SP slim type power supply module
Redundant power supply module	Generic term for Q63RP and Q64RP redundant power supply module
AnS series power supply module	Generic term for A1S61PN, A1S62PN, and A1S63P power supply modules

Generic Term/Abbreviation	Description
A series power supply module	Generic term for A61P, A61PN, A62P, A63P, A61PEU, A62PEU power supply modules
Power supply module	Generic term for Q series power supply modules, AnS series power supply modules, A series power supply modules, slim type power supply module and redundant power supply module
Battery	Generic term for Q6BAT, Q7BAT, and Q8BAT CPU module batteries, Q2MEM-BAT SRAM card battery and Q3MEM-BAT SRAM card battery
SRAM card	Generic term for Q2MEM-1MBS, Q2MEM-2MBS, Q3MEM-4MBS, and Q3MEM-8MBS type SRAM card
Flash card	Generic term for Q2MEM-2MBF and Q2MEM-4MBF type Flash card
ATA card	Generic term for Q2MEM-8MBA, Q2MEM-16MBA, and Q2MEM-32MBA type ATA card
Memory card	Generic term for SRAM card, Flash card and ATA card
GOT	Generic term for Mitsubishi Graphic Operation Terminal GOT-A*** series, GOT-F*** series and GOT1000 series.
PC CPU module	MELSEC-Q series compatible PC CPU module of Contec makes
Control system	Abbreviation for Redundant CPU specified as control system
Standby system	Abbreviation for Redundant CPU specified as standby system
System A	Abbreviation for Redundant CPU specified as System A
System B	Abbreviation for Redundant CPU specified as System B

CHAPTER1 OVERVIEW

This manual describes the programs, I/O number assignment method, functions, devices, and other specifications of the Q series CPU modules (☞ (1) below).

Refer to the following manual, for the other specifications such as the power supply modules, base units, extension cables, memory cards, and batteries.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(1) CPU modules corresponding to the description of this manual

The CPU modules described in this manual are as shown in Table1.1.

Table1.1 List of CPU modules corresponding to the description of this manual

CPU module	Model name
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
Universal model QCPU	Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU

(2) List of Q Series CPU Module manuals

The Q series CPU module manuals are as shown below.
For details such as manual numbers, refer to "About Manuals" in this manual.

(a) Basic model QCPU

Table1.2 List of user's manuals of basic model QCPU

	Maintenance and Inspection	Program Fundamentals	Multi CPU System	Redundant System
Purpose	**QCPU User's Manual (Hardware Design, Maintenance and inspection)**	**QCPU User's Manual (Function Explanation, Program Fundamentals)**	**QCPU User's Manual (Multiple CPU System)**	**QnPRHCPU User's Manual (Redundant System)**
Confirmation of part names and specifications of the CPU module	Details	Outline		
Confirmation of connection methods for the power supply module, base unit and I/O module	Details			
Construction of the single CPU system (confirmation of start-up procedure and I/O number assignment)	Details			
Construction of the multiple CPU system (confirmation of start-up procedure and I/O number assignment)			Details	
Confirmation of the sequence program configuration and memory		Details		
Confirmation of the functions, parameters, and devices of the CPU module		Details		
Confirmation of the troubleshooting and error codes	Details			

Table1.3 List of programming manuals of basic model QCPU

	Common Instructions	PID Control Instructions	Process Control Instruction	SFC	MELSAP-L	Structured Text
Purpose	**QCPU (Q mode)/ QnACPU Programming Manual (Common Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (PID Control Instruction)**	**QnPHCPU/ QnPRHCPU Programming Manual (Process Control Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (SFC)**	**QCPU (Q mode) Programming Manual (MELSAP-L)**	**QCPU (Q mode) Programming Manual (Structured Text)**
Confirmation of usage of sequence instructions, basic instructions, application instructions, etc.	Details					
Confirmation of dedicated instructions for PID control		Outline				
Confirmation of MELSAP3's system configuration, performance specifications, functions, programming, debugging, and error codes				Details		
Confirmation of the programming method, specifications, functions, etc. required for SFC programming of the MELSAP-L type					Details	
Confirmation of the programming method of the structured text language						Details

(b) High Performance Model QCPU

Table1.4 List of user's manuals of high performance model QCPU

	Maintenance and Inspection	Program Fundamentals	Multi CPU System	Redundant System
Purpose	**QCPU User's Manual (Hardware Design, Maintenance and inspection)**	**QCPU User's Manual (Function Explanation, Program Fundamentals)**	**QCPU User's Manual (Multiple CPU System)**	**QnPRHCPU User's Manual (Redundant System)**
Confirmation of part names and specifications of the CPU module	Details	Outline		
Confirmation of connection methods for power supply module, base unit and I/O module	Details			
Construction of the single CPU system (confirmation of start-up procedure and I/O number assignment)	Details			
Construction of the multiple CPU system (confirmation of start-up procedure and I/O number assignment)			Details	
Confirmation of the sequence program configuration and memory		Details		
Confirmation of the functions, parameters, and devices of CPU module		Details		
Confirmation of the troubleshooting and error codes	Details			

Table1.5 List of programming manuals of high performance model QCPU

	Common Instructions	PID Control Instructions	Process Control Instruction	SFC	MELSAP-L	Structured Text
Purpose	**QCPU (Q mode)/ QnACPU Programming Manual (Common Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (PID Control Instruction)**	**QnPHCPU/ QnPRHCPU Programming Manual (Process Control Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (SFC)**	**QCPU (Q mode) Programming Manual (MELSAP-L)**	**QCPU (Q mode) Programming Manual (Structured Text)**
Confirmation of usage of sequence instructions, basic instructions, application instructions, etc.	Details					
Confirmation of dedicated instructions for PID control		Details				
Confirmation of MELSAP3's system configuration, performance specifications, functions, programming, debugging, and error codes				Details		
Confirmation of the programming method, specifications, functions, etc. required for SFC programming of the MELSAP-L type					Details	
Confirmation of the programming method of the structured text language						Details

(c) Process CPU

Table1.6 List of user's manuals of process CPU

	Maintenance and Inspection	Program Fundamentals	Multi CPU System	Redundant System
Purpose	**QCPU User's Manual (Hardware Design, Maintenance and inspection)**	**QCPU User's Manual (Function Explanation, Program Fundamentals)**	**QCPU User's Manual (Multiple CPU System)**	**QnPRHCPU User's Manual (Redundant System)**
Confirmation of part names and specifications of the CPU module	Details	Outline		
Confirmation of connection methods for power supply module, base unit and I/O module	Details			
Construction of the single CPU system (confirmation of start-up procedure and I/O number assignment)	Details			
Construction of the multiple CPU system (confirmation of start-up procedure and I/O number assignment)			Details	
Confirmation of sequence program configuration and memory		Details		
Confirmation of the functions, parameters, and devices of the CPU module		Details		
Confirmation of the troubleshooting and error codes	Details			

Table1.7 List of programming manuals of process CPU

	Common Instructions	PID Control Instructions	Process Control Instruction	SFC	MELSAP-L	Structured Text
Purpose	**QCPU (Q mode)/ QnACPU Programming Manual (Common Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (PID Control Instruction)**	**QnPHCPU/ QnPRHCPU Programming Manual (Process Control Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (SFC)**	**QCPU (Q mode) Programming Manual (MELSAP-L)**	**QCPU (Q mode) Programming Manual (Structured Text)**
Confirmation of usage of sequence instructions, basic instructions, application instructions, etc.	Details					
Confirmation of dedicated instructions for process control			Details			
Confirmation of MELSAP3's system configuration, performance specifications, functions, programming, debugging and error codes				Details		
Confirmation of the programming method, specifications, functions etc. required for SFC programming of the MELSAP-L type					Details	
Confirmation of the programming method of the structured text language						Details

(d) Redundant CPU

Table1.8 List of user's manual of redundant CPU

	Maintenance and Inspection	Program Fundamentals	Multi CPU System	Redundant System
Purpose	**QCPU User's Manual (Hardware Design, Maintenance and inspection)**	**QCPU User's Manual (Function Explanation, Program Fundamentals)**	**QCPU User's Manual (Multiple CPU System)**	**QnPRHCPU User's Manual (Redundant System)**
Confirmation of part names and specifications of the CPU module	Details	Outline		
Confirmation of connection methods for power supply module, base unit and I/O module	Details			
Construction of redundant system (confirmation of start-up procedure and I/O number assignment)				Details
Confirmation of the configuration and memory of sequence programs		Details		
Confirmation of the functions, parameters, devices, etc. of the CPU module		Details		
Confirmation of the troubleshooting				Details
Confirmation of the error codes	Details			

Table1.9 List of programming manuals of redundant CPU

	Common Instructions	PID Control Instructions	Process Control Instruction	SFC	MELSAP-L	Structured Text
Purpose	**QCPU (Q mode)/ QnACPU Programming Manual (Common Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (PID Control Instruction)**	**QnPHCPU/ QnPRHCPU Programming Manual (Process Control Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (SFC)**	**QCPU (Q mode) Programming Manual (MELSAP-L)**	**QCPU (Q mode) Programming Manual (Structured Text Edition)**
Confirmation of usage of sequence instructions, basic instructions, application instructions, etc.	Details					
Confirmation of dedicated instructions for PID control		Details				
Confirmation of dedicated instructions for process control			Details			
Confirmation of MELSAP3's system configuration, performance specifications, functions, programming, debugging and error codes				Details		
Confirmation of the programming method, specifications, functions, etc. required for SFC programming of the MELSAP-L type					Details	
Confirmation of the programming method of the structured text language						Details

(e) Universal model QCPU

Table1.10 List of user's manual of Universal model QCPU

	Maintenance and Inspection	Program Fundamentals	Multi CPU System	Redundant System
Purpose	**QCPU User's Manual (Hardware Design, Maintenance and inspection)**	**QCPU User's Manual (Function Explanation, Program Fundamentals)**	**QCPU User's Manual (Multiple CPU System)**	**QnPRHCPU User's Manual (Redundant System)**
Confirmation of part names and specifications of the CPU module	Details	Outline		
Confirmation of connection methods for power supply module, base unit and I/O module	Details			
Construction of the single CPU system (confirmation of start-up procedure and I/O number assignment)	Details			
Construction of the multiple CPU system (confirmation of start-up procedure and I/O number assignment)			Details	
Confirmation of the sequence program configuration and memory		Details		
Confirmation of the functions, parameters, and devices of CPU module		Details		
Confirmation of the troubleshooting and error codes	Details			

Table1.11 List of programming manuals of Universal model QCPU

	Common Instructions	PID Control Instructions	Process Control Instruction	SFC	MELSAP-L	Structured Text
Purpose	**QCPU (Q mode)/ QnACPU Programming Manual (Common Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (PID Control Instruction)**	**QnPHCPU/ QnPRHCPU Programming Manual (Process Control Instruction)**	**QCPU (Q mode)/ QnACPU Programming Manual (SFC)**	**QCPU (Q mode) Programming Manual (MELSAP-L)**	**QCPU (Q mode) Programming Manual (Structured Text)**
Confirmation of usage of sequence instructions, basic instructions, application instructions, etc.	Details					
Confirmation of dedicated instructions for PID control		Details				
Confirmation of MELSAP3's system configuration, performance specifications, functions, programming, debugging, and error codes				Details		
Confirmation of the programming method, specifications, functions, etc. required for SFC programming of the MELSAP-L type					Details	
Confirmation of the programming method of the structured text language						Details

1.1 Features

This section explains the features of the CPU modules.

1.1.1 Features of Basic model QCPU

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

(1) Cost performance optimum for small-scaled system

The Basic model QCPU is a module targeted for a small-scaled system and optimum for controlling a simple, compact system.

The Basic model QCPU realizes the cost performance optimum for a small-scaled system.

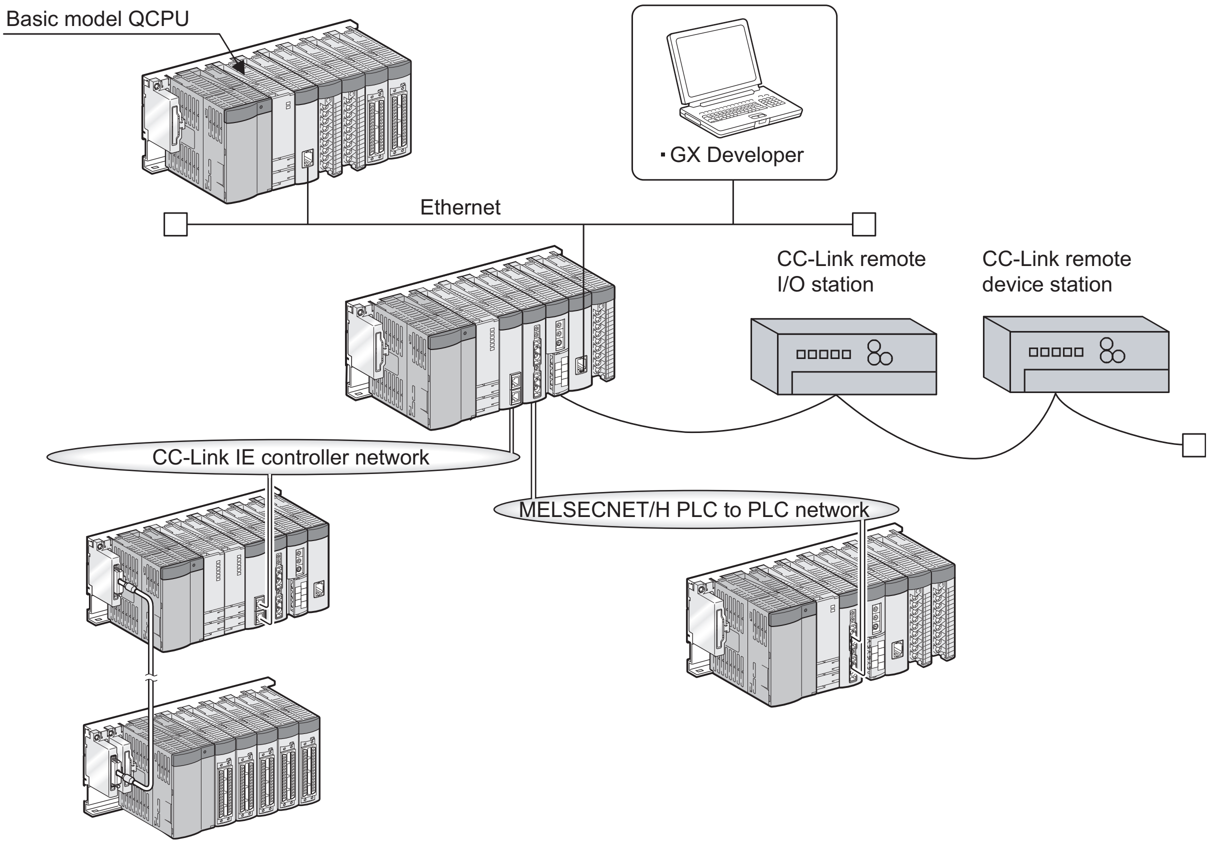


Figure 1.1 System example using the Basic model QCPU

(2) Communication with personal computer and/or display device using serial communication function (☞ Section 6.24)

The Q00CPU or Q01CPU can communicate with a personal computer and/or display device via the RS-232 interface in the MELSEC communication protocol (hereinafter the MC protocol).

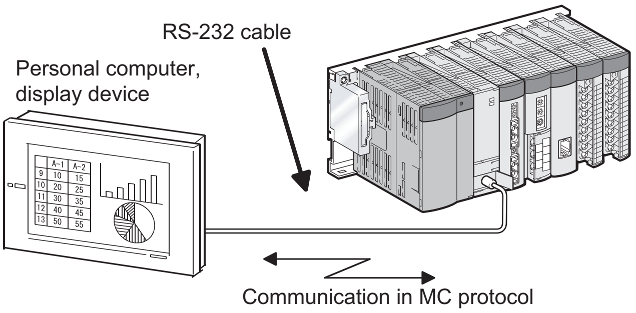

RS-232 cable
Personal computer, display device
Communication in MC protocol

Figure 1.2 Communication with personal computer and/or display device

☒**POINT**

New functions of a CPU module may be added depending on the upgrade of the serial number of the CPU module or the function version of GX Developer.
For the list of functions have been added with the upgrade of the Basic model QCPU, refer to Appendix 4.1.

1.1.2 Features of 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 is a module targeted for small-scaled to large-scaled systems and capable of high-speed massive data processing.

The High Performance model QCPU realizes construction of the optimum and high-performance system.

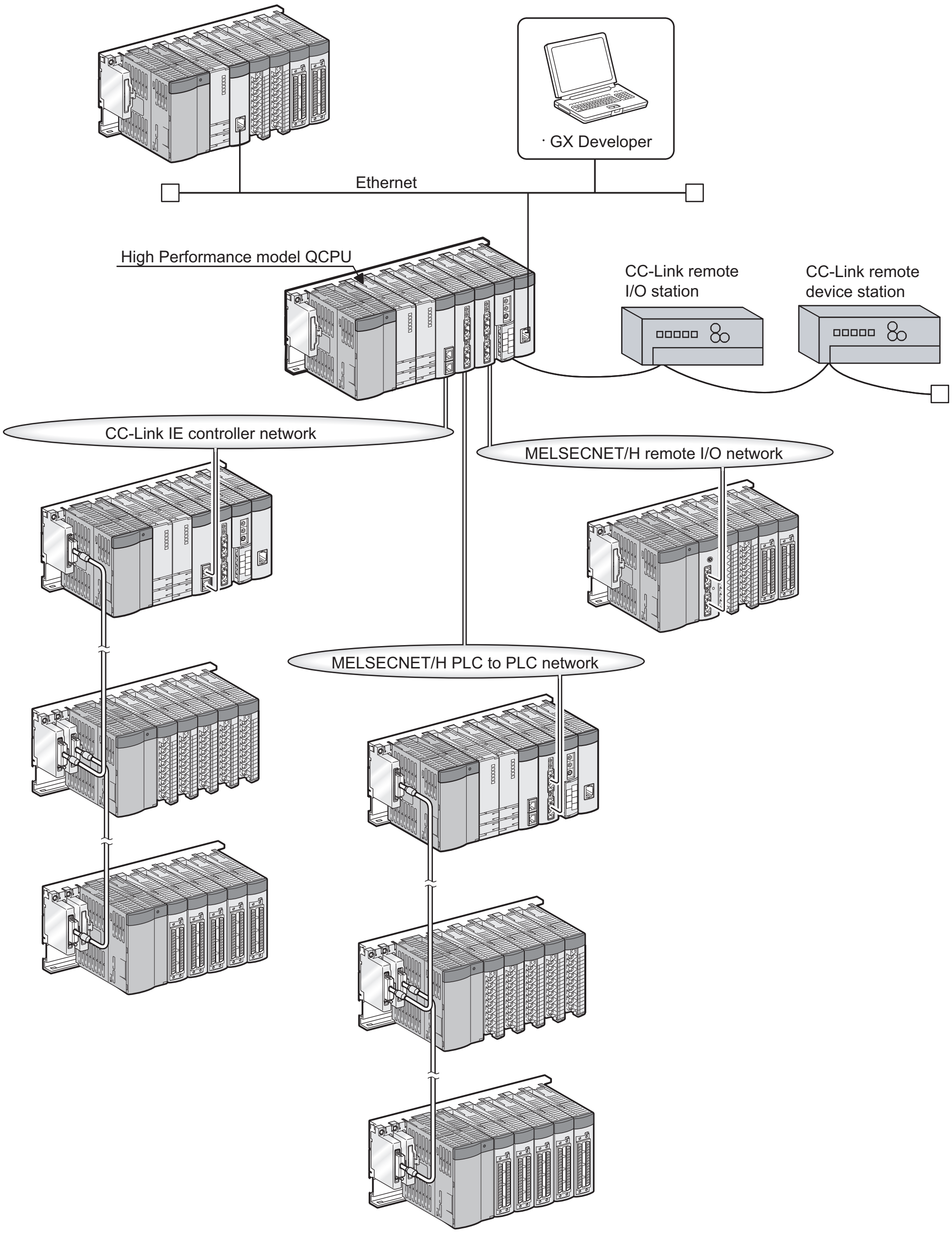

· GX Developer
Ethernet
High Performance model QCPU
CC-Link remote
I/O station
CC-Link remote
device station
CC-Link IE controller network
MELSECNET/H remote I/O network
MELSECNET/H PLC to PLC network

Figure 1.3 System example using the High Performance model QCPU

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

The AnS/A series compatible extension base units (QA1S6□B, QA6□B and QA6ADP+A5□B/A6□B) allow the High Performance model QCPU to use the AnS/A series I/O modules and special function modules.

☒POINT

New functions of a CPU module may be added depending on the upgrade of the serial number of the CPU module or the function version of GX Developer.
For the list of functions have been added with the upgrade of the High Performance model QCPU, refer to Appendix 4.2.

1.1.3 Features of Process CPU

The features specific to the Process CPU are described below.

(1) Additional 52 process control functions

Fifty-two instructions supporting high-leveled process controls have been added to the CPU based on the High Performance model QCPU.

Refer to the following manual for details of the added instructions.

☞ QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions)

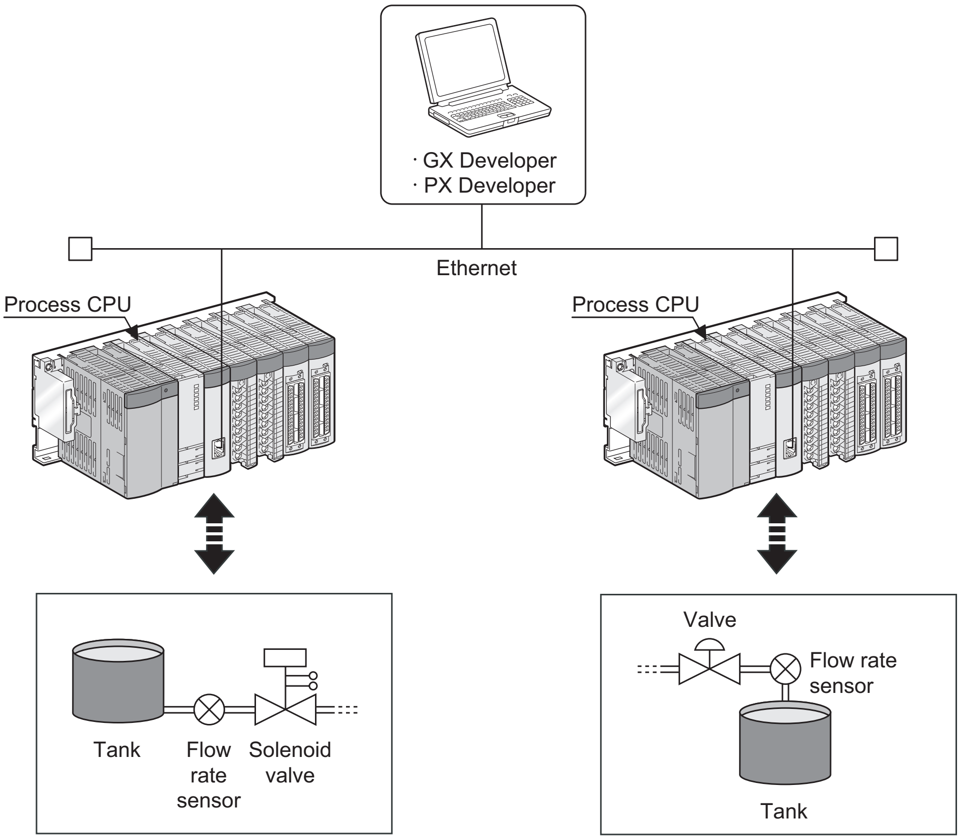


Figure 1.4 Operations of Process CPU

(2) Two-degree-of-freedom PID control system

The Two-degree-of-freedom PID control system has been adopted to enable optimum responses to both set value changes and disturbance.

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

The auto tuning function enables automatic control parameter tuning, decreases the tuning time and labor of operators and control engineers, and eliminates differences between individuals in tuning results.

Refer to the following manual for details of the auto tuning function.

☞ QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions)

(4) Module can be replaced online (Online module change)

When a module becomes faulty, it can be replaced without the system being stopped. Modules available for this are the Q series I/O modules, and the A/D converter modules, D/A converter modules, temperature input modules, temperature control modules and pulse input modules of function version C.

Refer to the following manuals for details of the online module change.

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

☞ Manuals of the modules compatible with the online module change

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

A MELSECNET/H multiplexed remote I/O system can be configured,mounting a remote master station in the MELSECNET/H network system.

(6) Supporting software package dedicated to process control

Using the process control software package (PX Developer), PID control programs can be created easily with function blocks.

The combination of the Process CPU and the process control software package (PX Developer) offers an excellent engineering environment.

☒POINT

1. When using the Q12PHCPU and Q25PHCPU, use GX Developer Version 7.10L or later.
   When using the Q02PHCPU and Q06PHCPU, use GX Developer Version 8.68W or later.
2. Use PX Developer together with GX Developer Version 7.20W or later.
   Refer to the PX Developer manual for details.
   When using the Q12PHCPU and Q25PHCPU, use PX Developer Version 1.00A or later.
   When using the Q02PHCPU and Q06PHCPU, use PX Developer Version 1.18U or later.
3. New functions of a CPU module may be added depending on the upgrade of the serial number of the CPU module or the function version of GX Developer.
   For the list of functions have been added with the upgrade of the Process CPU, refer to Appendix 4.3.

1.1.4 Features of Redundant CPU

The features specific to the Redundant CPU are described below.

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

(a) Redundant system using Redundant CPU

With the Redundant CPU, a redundant system as a whole including base units, power supply modules, and CPU modules (Redundant CPUs) can be configured. The redundant system realizes high reliability since the standby system takes over the control even if a failure occurs in the control system.

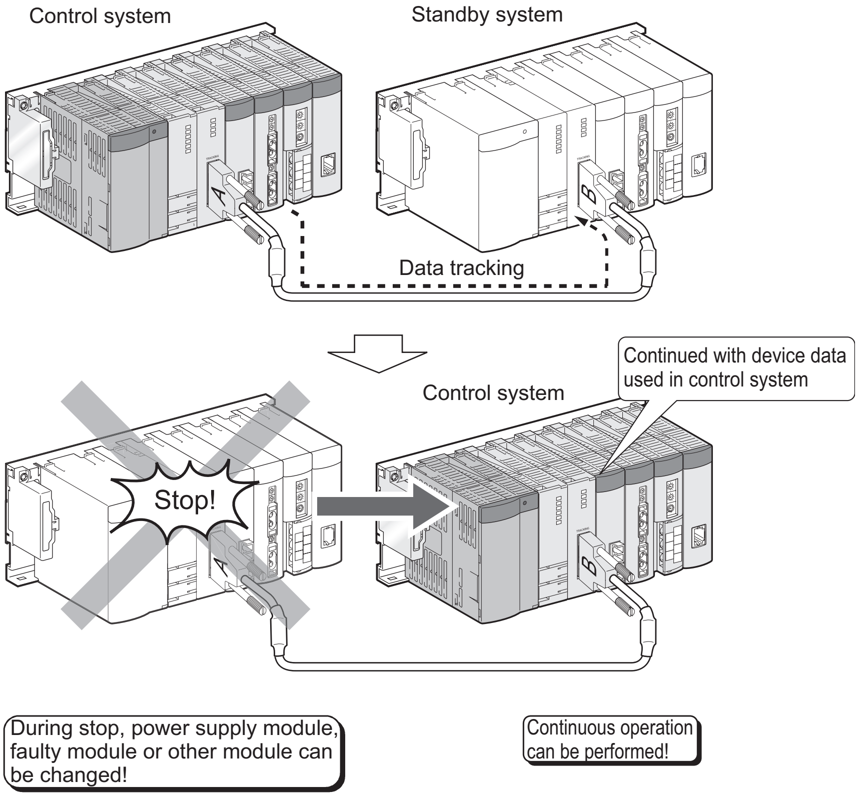

Control system
Standby system
A
B
Data tracking
Continued with device data used in control system
Control system
Stop!
A
B
During stop, power supply module, faulty module or other module can be changed!
Continuous operation can be performed!

Figure 1.5 Operation of 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) together with the remote I/O station, redundant power supply on the remote I/O station side can be realized.
This enables the change of the power supply module without stopping the system even when the power supply module on the remote I/O station side fails.

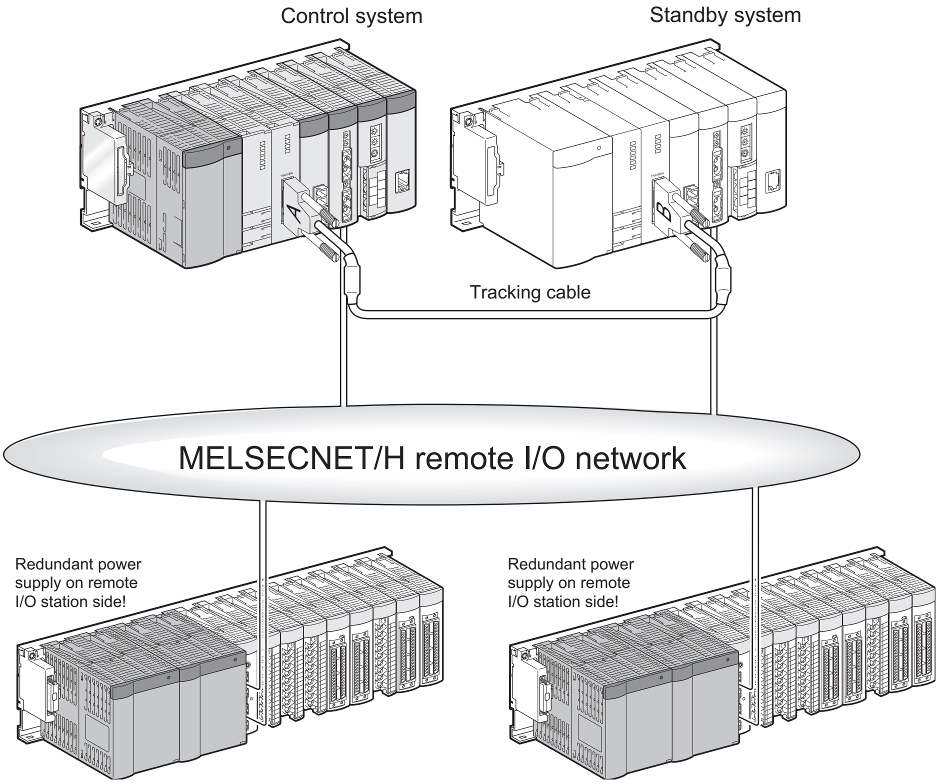

Control system
Standby system
A
B
Tracking cable
MELSECNET/H remote I/O network
Redundant power supply on remote I/O station side!
Redundant power supply on remote I/O station side!

Figure 1.6 Redundant power supply system

☒**POINT**

1. When using the Redundant CPU, use GX Developer Version 8.18U or later.
2. Use PX Developer in combination with GX Developer Version.
   When using the Redundant CPU, use PX Developer Version 1.06G or later.
   Refer to the PX Developer manual for details of PX Developer.
3. New functions of a CPU module may be added depending on the upgrade of the serial number of the CPU module or the function version of GX Developer.
   For the list of functions have been added with the upgrade of the Redundant CPU, refer to Appendix 4.4.

Remark

Refer to the following manual for detailed features, functions and others of the Redundant CPU.

☞ QnPRHCPU User's Manual (Redundant System)

1.1.5 Features of Universal model QCPU

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

(1) Realization of higher speed processing

The Universal model QCPU speeds up the basic instruction processing time, floating point arithmetic processing time, and access processing time to file registers faster than other CPUs.

Standard device registers (Z)[*1] also realize higher speed processing between the register operations (the transfer instruction).

[* 1] An index register used between the register operations is called a standard device register. (☞ Section 10.6.2)

(2) Large-capacity file register

File register whose capacity is 640k points maximum (4086k points maximum when a memory card is used)[*2] can be set inside the CPU module.

[* 2] For the Q26UDHCPU and Q26UDEHCPU only.

(3) Use of double-precision floating-point operation instructions

The double-precision floating-point operation instructions (64-bit instructions) are newly added to the existing single-precision floating-point operation instructions. (☞ Section 3.9.4)

This enables more accurate analog control and positioning control.

(4) Using file register area as data register and link register

The file register area can be used as a pseudo extended area of data register (D) and link register (W). (☞ Section 10.8)

Programming, in addition to internal user devices, using extended data register (D) or extended link register (W) whose capacity is 640k points maximum (4086k points maximum when a memory card is used)[*4] is available.

[* 3] Extended areas of data register (D) and link register (W) are called extended data register (D) and extended link register (W), respectively.

[* 4] For the Q26UDHCPU and Q26UDEHCPU only.

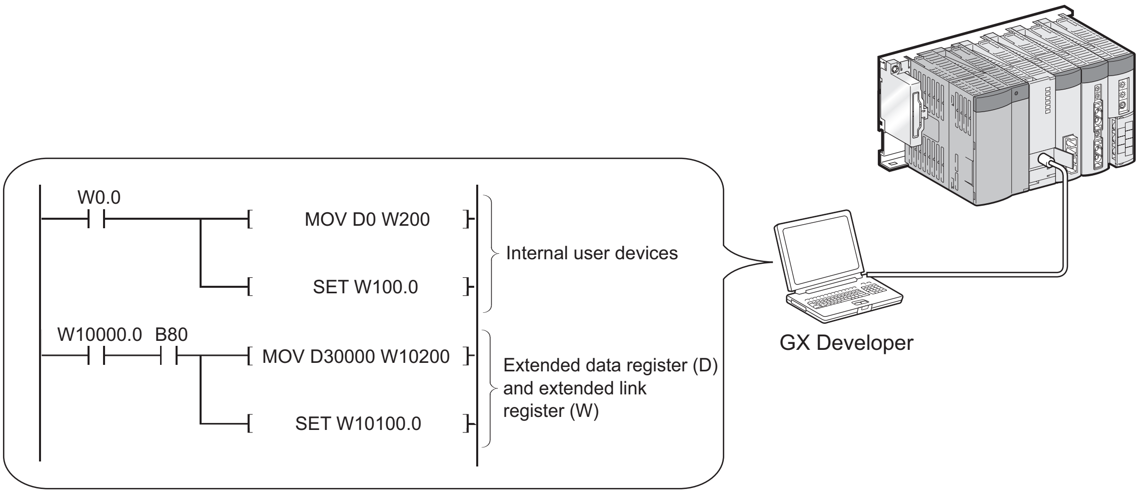


Figure 1.7 Extended data register (D) and extended link register (W)

(5) Indexing in the range of 32-bit

Indexing for entire area of the file register is possible by expanding the indexing range to 32 bits.

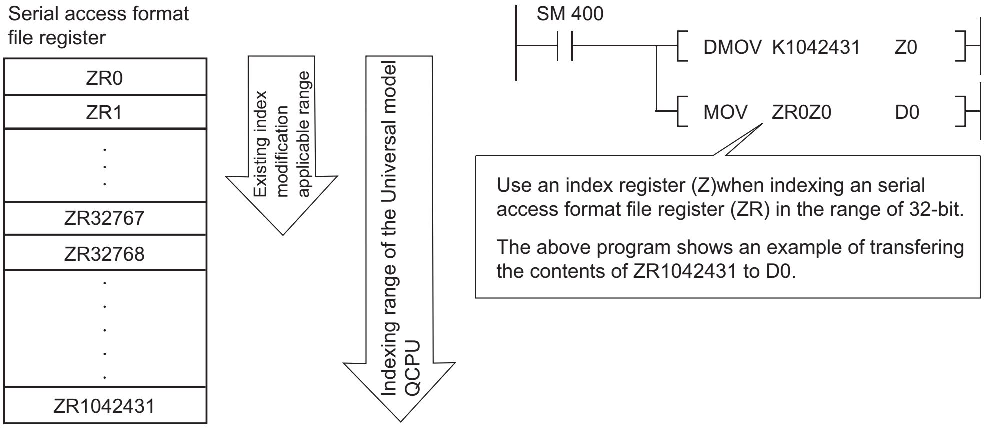

Serial access format file register
ZR0
ZR1
ZR32767
ZR32768
ZR1042431
Existing index modification applicable range
Indexing range of the Universal model QCPU
SM 400
DMOV K1042431 Z0
MOV ZR0Z0 D0
Use an index register (Z)when indexing an serial access format file register (ZR) in the range of 32-bit.
The above program shows an example of transfering the contents of ZR1042431 to D0.

Figure 1.8 32-bit indexing

(6) Communication using built-in Ethernet ports of CPU module

The Built-in Ethernet port QCPU can communicate with MELSOFT devices and/or GOT using the built-in Ethernet ports of the CPU module.

Features of the functions using the built-in Ethernet ports of the CPU module are described below.

(a) GX Developer/GOT connection (☞ Section 7.2)

- The Find CPU function makes it possible to find the Built-in Ethernet port QCPU connected to the same hub as GX Developer and displays a list.
- MELSOFT connection allows access to CPU modules on the network, such as internal LAN, via routers.

(b) GX Developer direct connection (simple connection) (☞ Section 7.3)

- When connecting the Built-in Ethernet port QCPU with GX Developer, the direct connection (simple connection), which connects them using only one Ethernet cable (not using a hub) is available. The direct connection enables communication with only specifying connection target. IP address setting is not required.

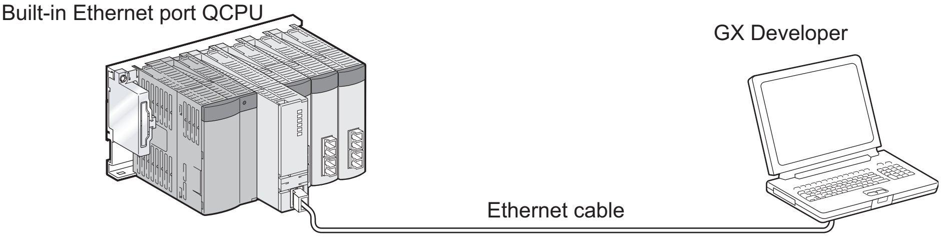


Figure 1.9 Direct connection (simple connection)

(c) MC protocol communication (☞ Section 7.4)

- External devices such as personal computers and display devices read/write device data from/to the Built-in Ethernet port QCPU. This enables external devices to monitor the operation of the CPU module, analyze data, and manage production.

(d) Time setting function (SNTP client) (☞ Section 7.5)

- The time setting function enables automatic time setting for the Built-in Ethernet port QCPU, resulting in reducing the maintenance cost for time setting.
- By synchronizing the Built-in Ethernet port QCPU connected via Ethernet, error occurrence order for each process can be checked and troubleshooting becomes easy.
- Since time setting can be performed automatically at programmable controller CPU power ON, the CPU module can start operation with accurate time data.

(e) File transfer function (FTP) (☞ Section 7.6)

- An external device with a FTP client function can read/write files from/to the Built-in Ethernet port QCPU in units of files, so that data such as in the program file and/or file register can be transferred easily.

(f) Remote password (☞ Section 7.7)

- The Built-in Ethernet port QCPU can prevent unauthorized access from outside of the system and enhance the security by setting the remote password.

☒POINT

New functions of a CPU module may be added depending on the upgrade of the serial number of the CPU module or the function version of GX Developer.
For the list of functions have been added with the upgrade of the Universal model QCPU, refer to Appendix 4.5.

1.2 Program Storage and Operation

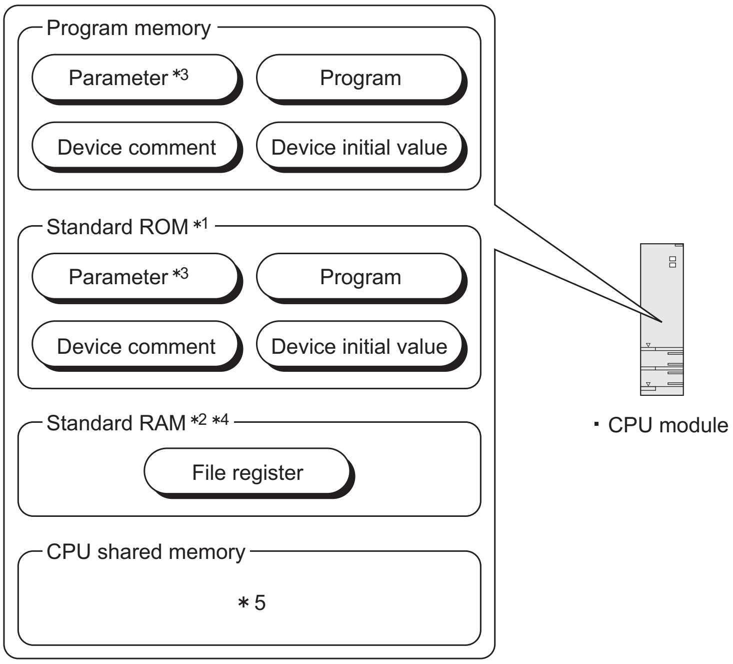
(1) Program storage

Basic
Note1.1

(a) Storage of program created by GX Developer

The program created by GX Developer can be stored into the program memory, standard ROM or memory card Note1.1 of the CPU module.

1) Basic model QCPU



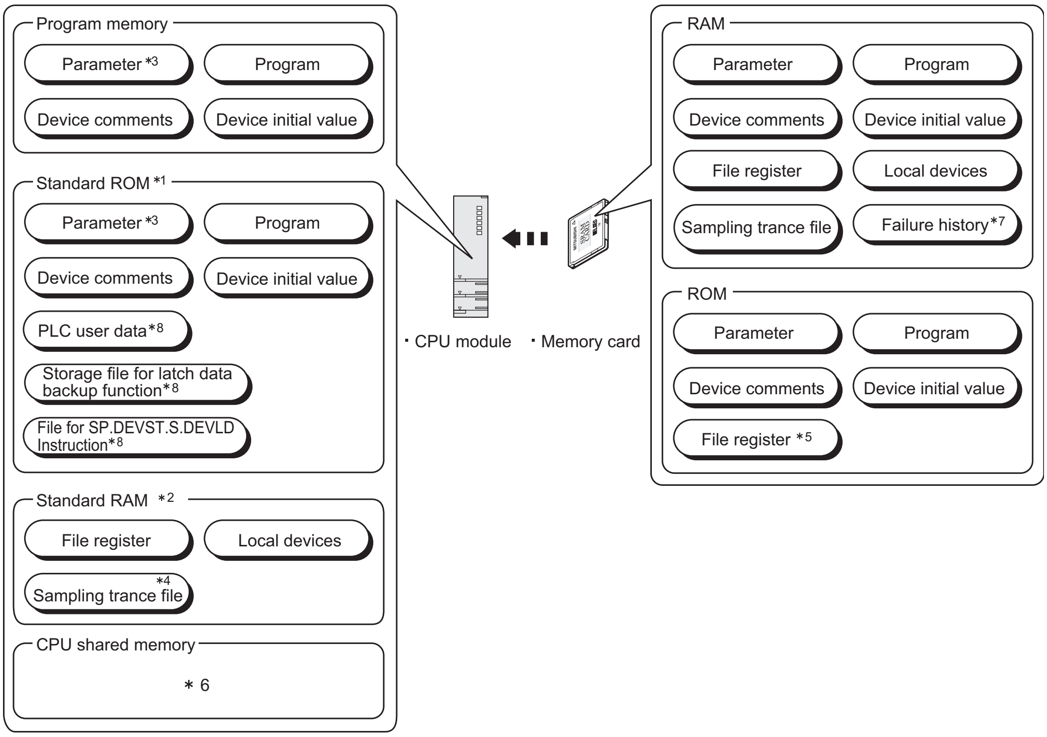
* 1 : The standard ROM is used to ROM the program memory.
* 2 : The standard RAM is a memory used for the file registers.
* 3 : The intelligent function module parameters set by GX Configurator are included.
* 4 : The Q00JCPU does not have the standard RAM.
 The file registers are unavailable.
* 5 : Refer to the following manual for the CPU shared memory.
 ☞ QCPU User's Manual (Multiple CPU System)

Figure 1.10 Memory configuration and storage destinations of Basic model QCPU

Basic
Note1.1

The Basic model QCPU does not support the use of a memory card.

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



* 1 : The standard ROM is used in the following cases.
•For High Performance model QCPU, Process CPU, and Redundant CPU
Writing a program memory to ROM.
•For Universal model QCPU
Storing device comment or user data.

* 2 : The standard RAM is a memory provided for using the file registers, local devices and sampling trace file without installing a memory card.
The standard RAM is used to speed up access to the file registers.

* 3 : The intelligent function module parameters set by GX Configurator are included.

* 4 : When the sampling trace file is stored in the standard RAM, check the version of the CPU module and GX Developer. (Appendix 4)

* 5 : Only available for the Flash card.
Only read is enabled in PLC program.

* 6 : Refer to the following manual for the CPU shared memory.
QCPU User's Manual (Multiple CPU System)

* 7 : The Universal model QCPU cannot store an error history in the memory card.

* 8 : Can be stored only when the Universal model QCPU is used.

Figure 1.11 Memory configuration and storage destinations of High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

(b) Program execution

The CPU module operates the program stored in the program memory.

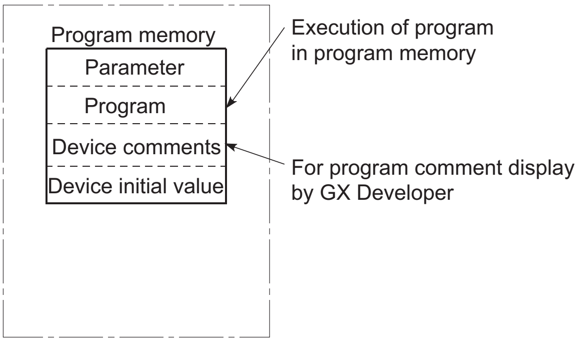

Program memory
Parameter
Program
Device comments
Device initial value
Execution of program in program memory
For program comment display by GX Developer

Figure 1.12 Execution of stored program

Basic Note1.2

(c) Execution of program stored in standard ROM/memory card Note1.2

Programs and data can also be stored into the standard ROM/memory card.

The programs stored in the standard ROM/memory card can be booted (read) to the program memory and executed when the PLC is powered ON or the CPU module is reset.

By storing programs and data into the standard ROM/memory card, they can be saved without battery backup.

1) Basic model QCPU

To execute boot from the standard ROM to the program memory, it is necessary to make boot file setting in the PLC Parameter dialog box of GX Developer.(☞ Section 5.1.5)

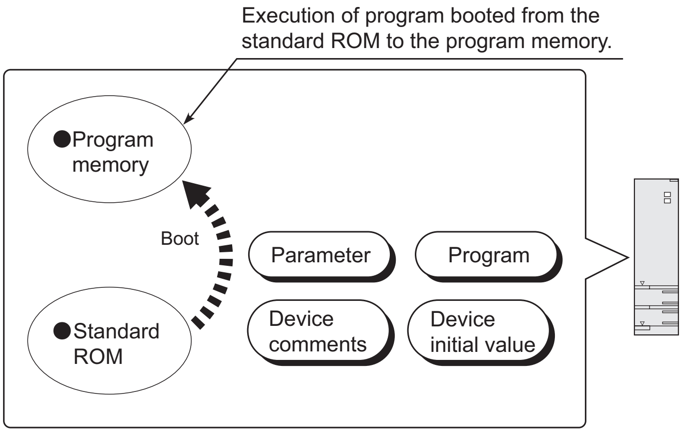


Figure 1.13 Boot run of Basic model QCPU

Basic Note1.2

Since the Basic model QCPU cannot use a memory card, it cannot store programs in the memory card.

Universal UD Note1.3

Universal UD Note1.4

2) **High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU**

To boot Note1.3 the program data to the program memory, make setting on the Boot file setting tab of PLC parameter in GX Developer and set the parameter-valid drive with DIP switches.Note1.4 (☞ Section 5.2.9)

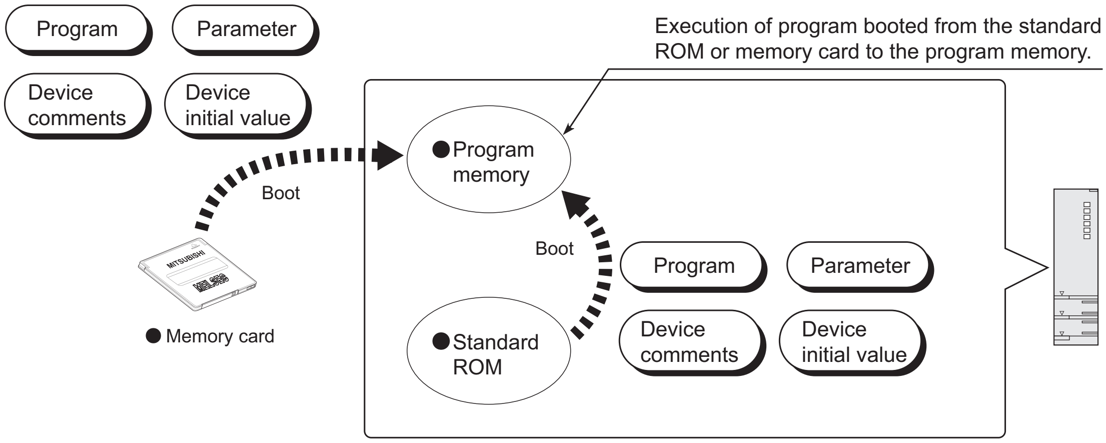


Figure 1.14 Boot operation of High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

Universal UD Note1.3

The Universal model QCPU cannot boot data from the standard ROM to the program memory. (☞ Section 5.2.3)

Universal UD Note1.4

The Universal model QCPU cannot make the parameter-valid drive settings with the DIP switch setting. (☞ Section 5.2.11)

(2) Structured programs

CPU module programs can be structured.
A program can be created according to processes and functions by structuring it.
As program structuring, structuring in the same program (☞ (2)(a) in this section) and file-divided structuring (☞ (2)(b) in this section) are available.

(a) Structuring in the same program

Structuring in the same program is achieved by creating subroutine programs (☞ Section 3.1.2) or interrupt programs (☞ Section 3.1.3).

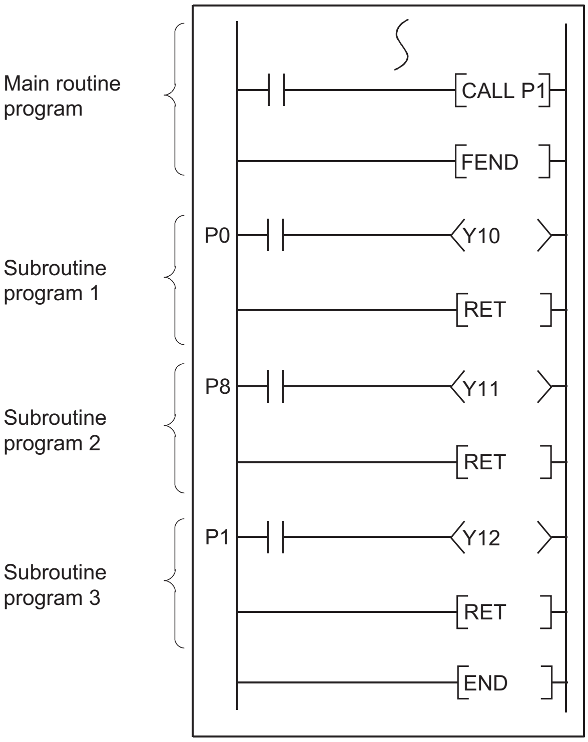


Figure 1.15 Example of structuring in the same program

Basic Note1.5

(b) File-divided structuring Note1.5

Programs are stored in file format.*1
Changing the file name allows the CPU module to store multiple programs.

* 1 : The program storage destination changes depending on the CPU module.
(☞ CHAPTER 5)

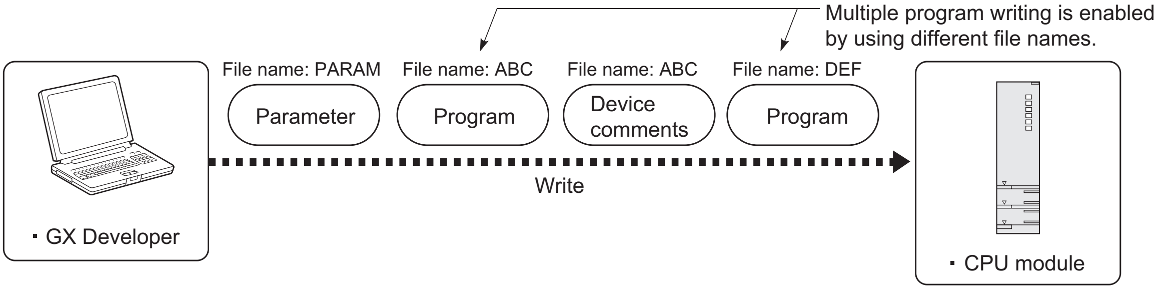
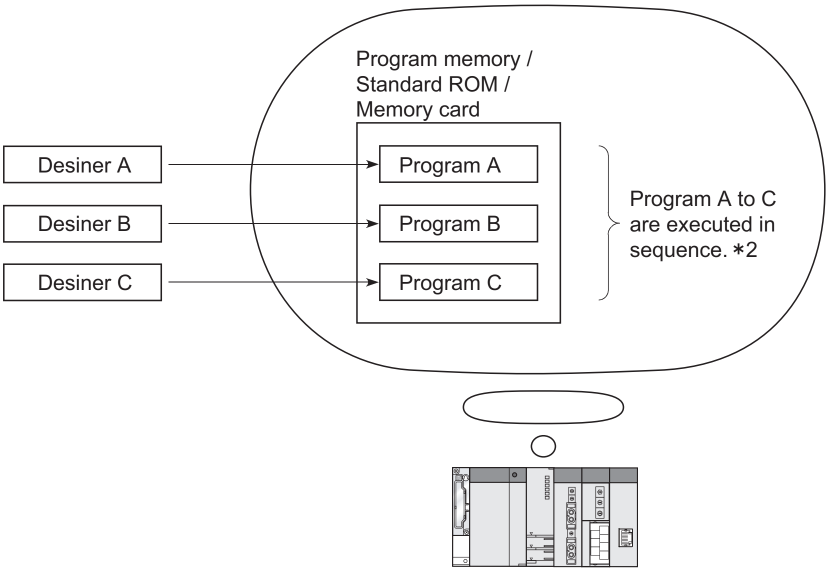


Figure 1.16 File-divided structuring

Hence, a program can be created separately by multiple designers, or can be divided and managed/maintained individually according to the processes and functions.
If specifications are changed, it is necessary to correct/debug only the corresponding program.

1) When program is created separately by multiple designers


Program memory / Standard ROM / Memory card
Desiner A
Desiner B
Desiner C
Program A
Program B
Program C
Program A to C are executed in sequence. *2

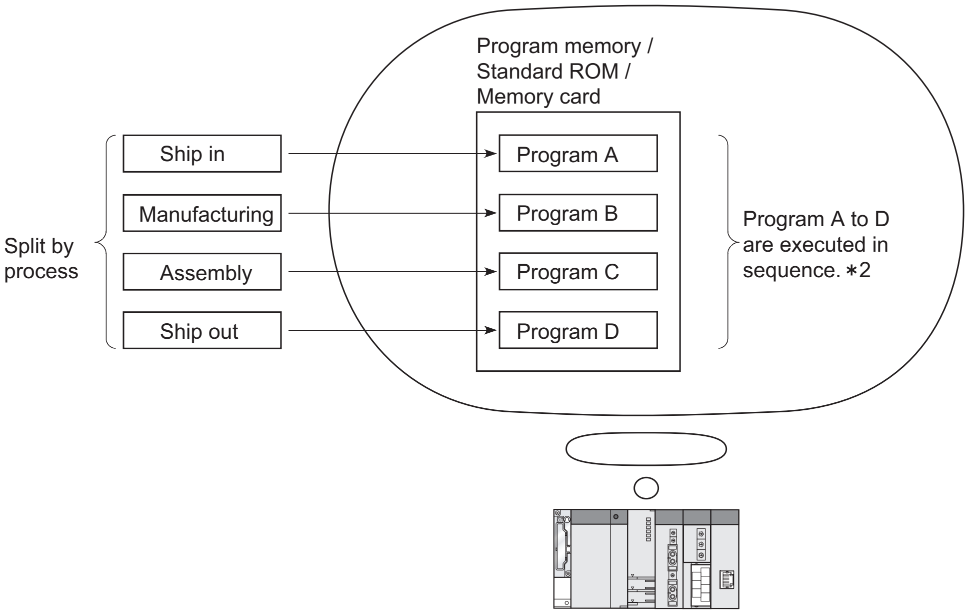
* 2 : The execution order and execution conditions of the program can be set by program setting (☞ Section 3.3.6(1)).

Figure 1.17 Separate creation of program (by designers)

Basic Note1.5

Since the Basic model QCPU cannot store multiple programs by changing the file name, it cannot execute file-divided structuring.

2) When program is divided according to processes*1

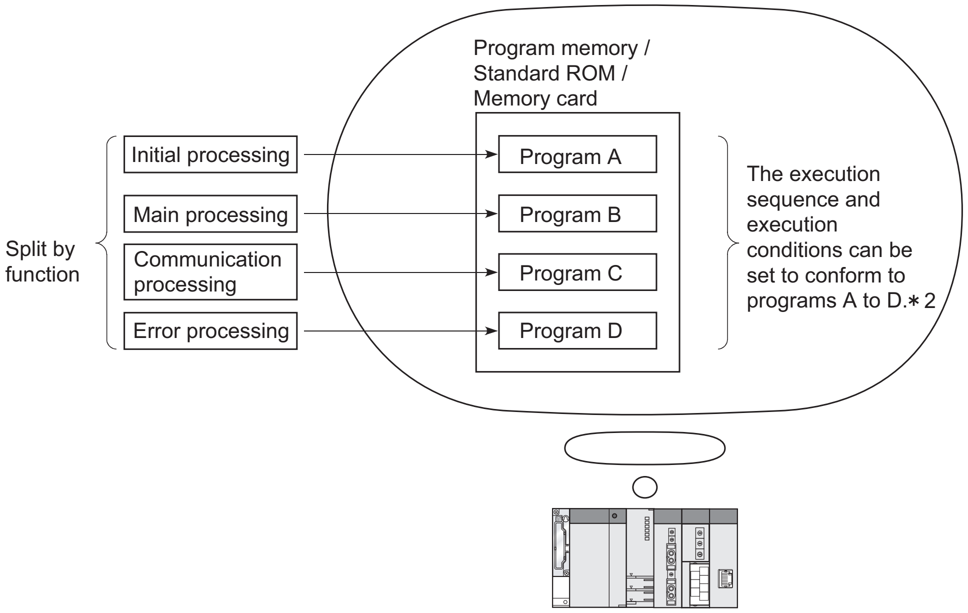


* 1 : The processings performed according to the processes can further be managed separately according to the functions.

* 2 : The execution order and execution conditions of the program can be set by program setting (☞ Section 3.3.6(1)).

Figure 1.18 Separate creation of program (by processes)

3) When program is divided according to functions



* 2 : The execution order and execution conditions of the program can be set by program setting (☞ Section 3.3.6(1)).

Figure 1.19 Separate creation of program (by functions)

1.3 Devices and Instructions Convenient for Programming

The CPU module has devices and instructions convenient for program creation.
The main devices and instructions are outlined below.

(1) Flexible device designation

The Q series CPU modules allow devices to be specified flexibly.

(a) Word device bits are handled as contacts/coils

By specifying the bit of the word device, each bit of the word device can be handled as a contact/coil.

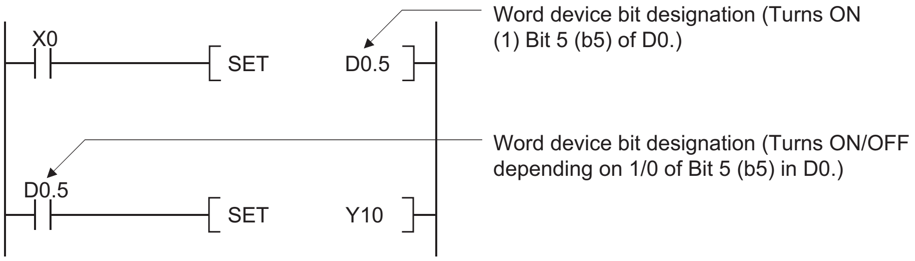

X0
SET
D0.5
Word device bit designation (Turns ON (1) Bit 5 (b5) of D0.)
Word device bit designation (Turns ON/OFF depending on 1/0 of Bit 5 (b5) in D0.)
D0.5
SET
Y10

Figure 1.20 Designation of word device bit

(b) Easy direct processing in 1-point units

Using the direct access input (DX□) and direct access output (DY□), direct processing can be easily performed in 1-point units in a program. (☞ Section 3.8.2)

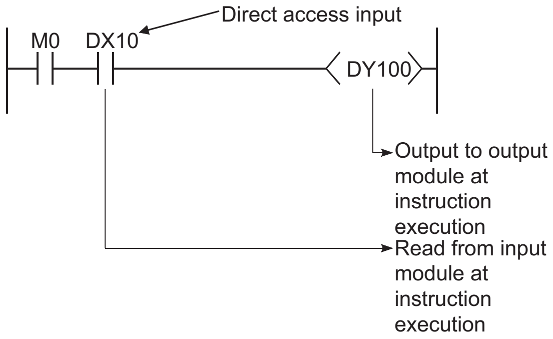

Direct access input
M0
DX10
DY100
Output to output module at instruction execution
Read from input module at instruction execution

Figure 1.21 Direct processing in 1-point units

(c) Input need not be pulsed by use of differential contact

An input need not be pulsed by use of a differential contact(┤↑├/┤↓├).

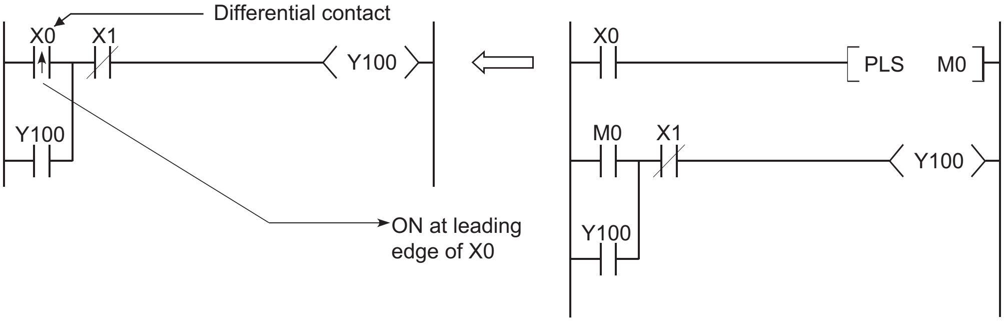


Figure 1.22 Use of differential contact

(d) Direct access to intelligent function module buffer memory

The intelligent function module buffer memory can be handled as devices for programming. (☞ Section 10.5)

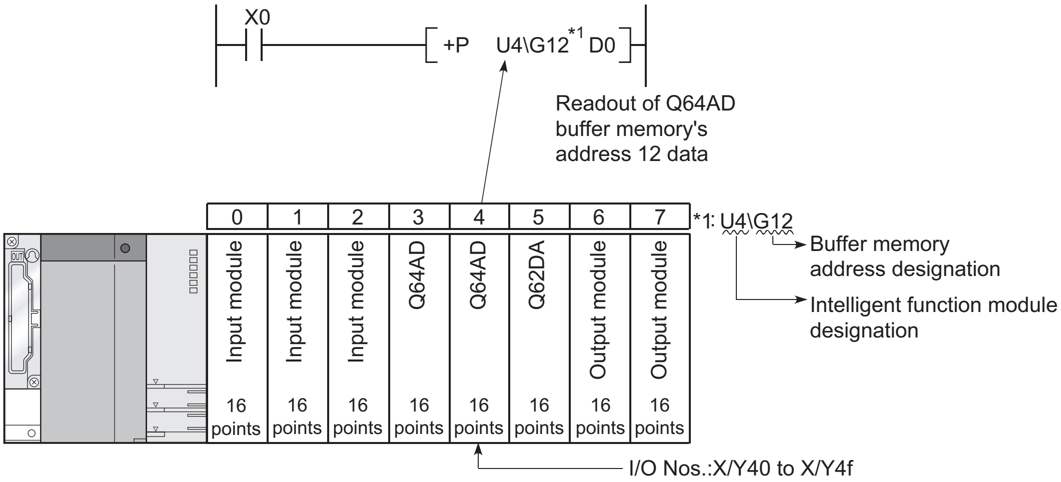


Figure 1.23 Direct access to intelligent function module buffer memory

(e) Direct access to link devices

Access can be made to the link device (LX, LY, LB, LW, SB, or SW) of the CC-Link IE controller network module and the MELSECNET/H network module without the refresh setting.(☞ Section 10.4)

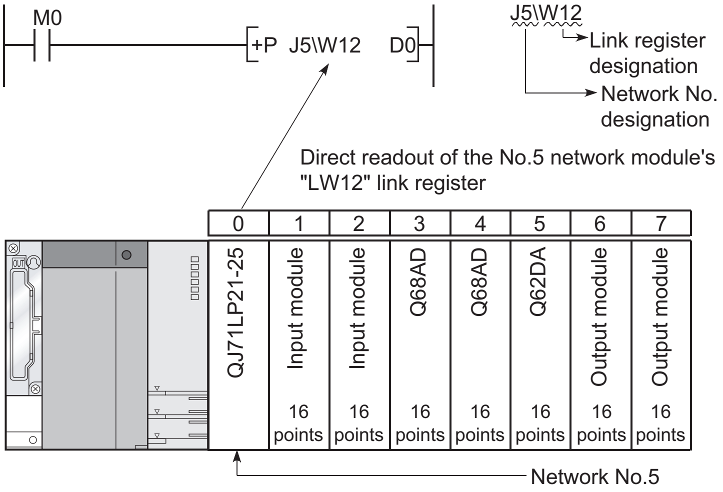


Figure 1.24 Direct access to link devices

(2) Structural description of programs

Using the index registers and edge relays, programs including pulse processing can be structured easily. (☞ Section 10.2.6)

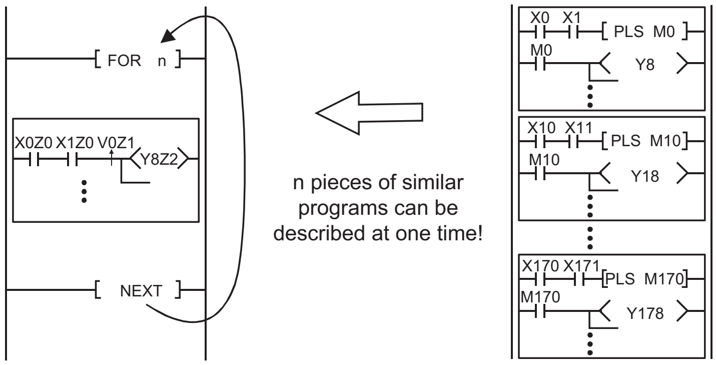


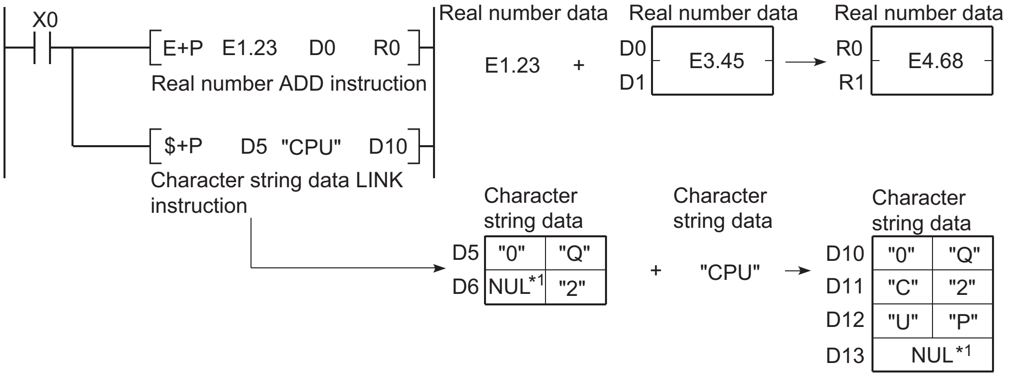
Figure 1.25 Structuring of programs including pulse processing

(3) Ease of data processing

(a) Real numbers and character string constants used unchanged

Basic
Note1.6

The real numbers (floating-point data) and character string constants can be used unchanged in programming.[Note1.6]



* 1 : NUL indicates "00$_H$ (end of character string)".

Figure 1.26 Use of real numbers and character string constants

Basic
Note1.6

When using real numbers (floating-point data) on the Basic model QCPU, check the version of the CPU module. (☞ Appendix 4.1 (2))
On the Basic model QCPU, character strings are available for only the $MOV, STR, DSTR, VAL, DVAL, ESTR and EVAL instructions.

(b) High-speed processing of massive data

The data processing instructions, such as the table processing instruction, have been reinforced to enable high-speed processing of massive data.

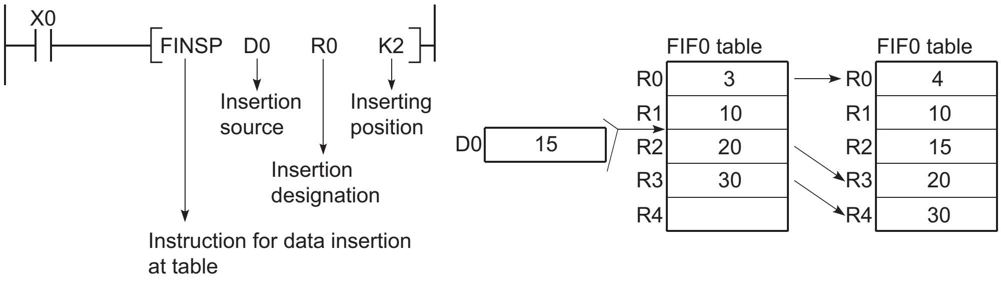


Figure 1.27 Data processing by table processing instruction

(4) Flexible management of subroutine program

(a) Subroutine program sharing

A subroutine program can be shared to reduce the number of program steps. Programs can also be created and managed easily.

Basic
Note1.7

A subroutine program can be created and called in the same program. Using a common pointer, however, a subroutine program in the other program can be also called. Note1.7

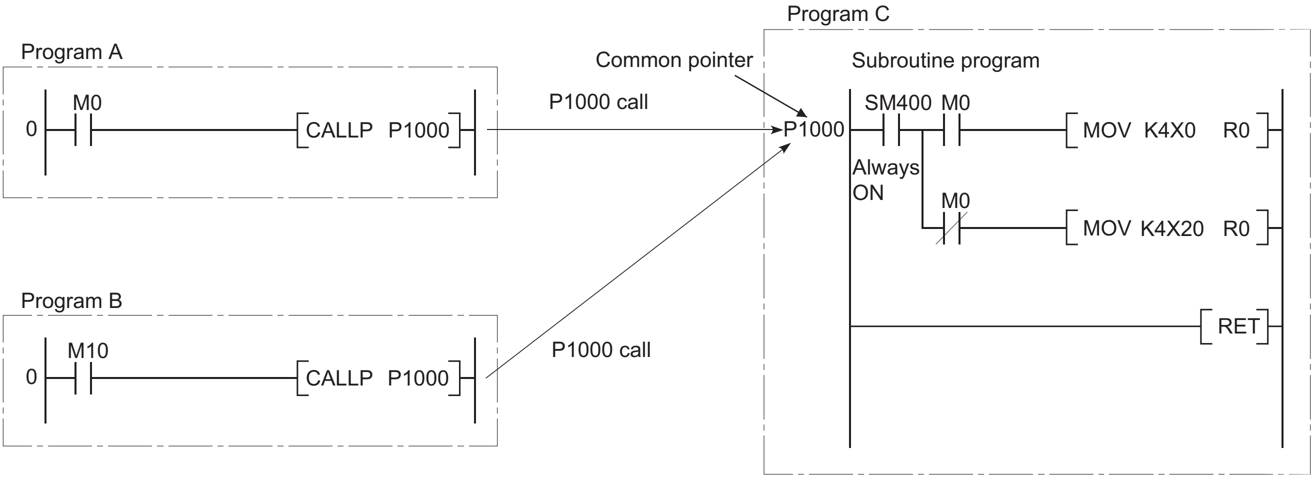


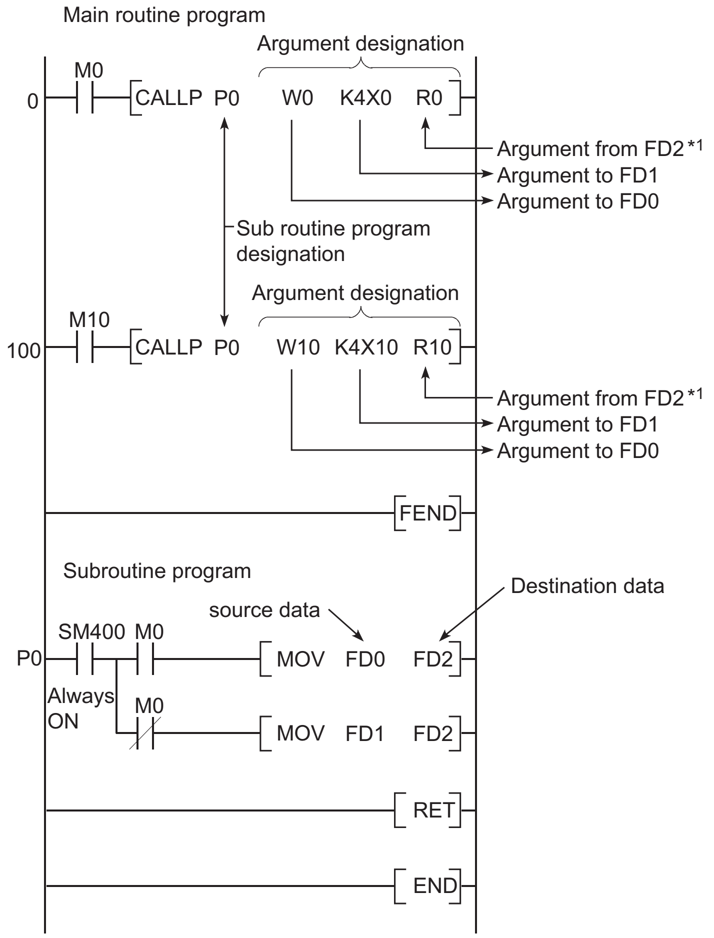
Figure 1.28 Subroutine program sharing

Basic
Note1.7

Since the Basic model QCPU cannot execute multiple programs, it cannot use subroutine programs in the other programs.

(b) Subroutine call instructions with arguments

A subroutine program called several time can be created easily by the subroutine call instructions with arguments.



* 1 : Refer to Section 10.3.1 for the input/output condition of the argument.

Figure 1.29 Subroutine call instructions with arguments

1.4 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.

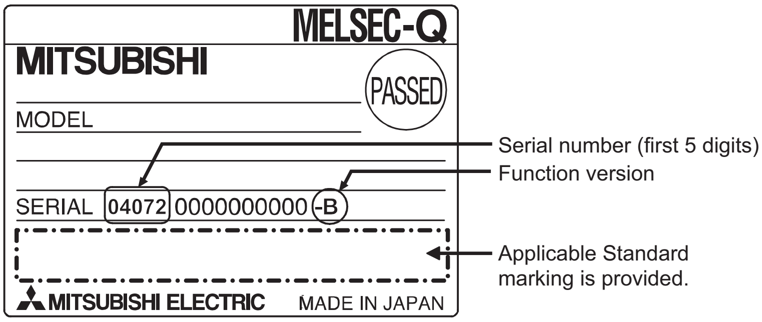

MELSEC-Q
MITSUBISHI
PASSED
MODEL
SERIAL 04072 0000000000 -B
MITSUBISHI ELECTRIC
MADE IN JAPAN
Serial number (first 5 digits)
Function version
Applicable Standard marking is provided.

Figure 1.30 Rating plate

(2) Checking on the front of the module

The serial number written on the rating plate is displayed on the front (at the bottom) of the module.This display is not supported on the following CPU modules.

- Modules manufactured in mid-September, 2007 or earlier
- Redundant CPUs manufactured in March 2008 or earlier
- Q00JCPU

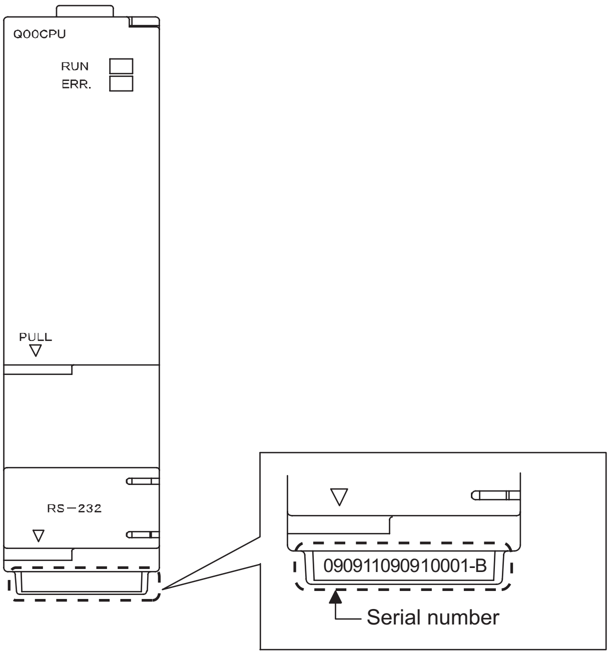


Figure 1.31 Display on the front of the module

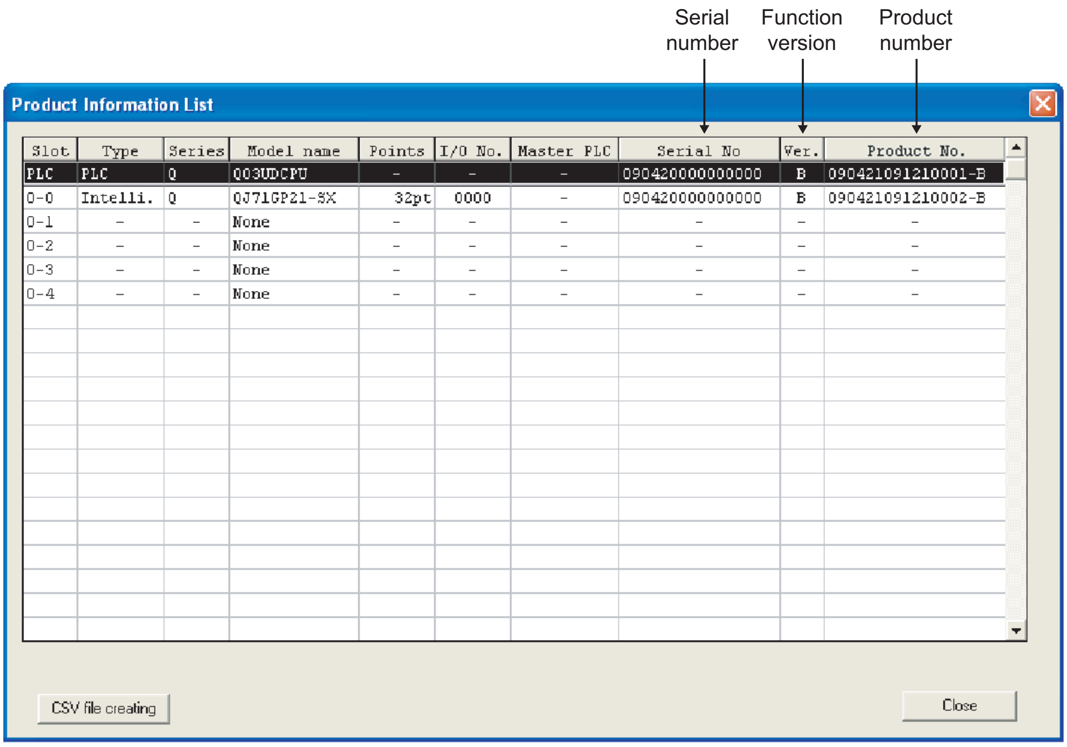
Remark

Serial number of Redundant CPU is displayed on the front of the module from March 2008 production.
Products manufactured during switching period may not have the serial number on the front of the module.

(3) Checking on the System monitor screen (Product information list screen)

To display the screen for checking the serial number and function version, select [Diagnostics] → [System monitor] and click the Product Inf. List button in GX Developer.

On the 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.	Product No.
PLC	PLC	Q	Q03UDCPU	-	-	-	090420000000000	B	090421091210001-B
0-0	Intelli.	Q	QJ71GP21-SX	32pt	0000	-	090420000000000	B	090421091210002-B
0-1	-	-	None	-	-	-	-	-	-
0-2	-	-	None	-	-	-	-	-	-
0-3	-	-	None	-	-	-	-	-	-
0-4	-	-	None	-	-	-	-	-	-

Figure 1.32 System monitor (Product information list)

[Serial No., Ver., and Product No.]

- 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.
- The serial number (product number) printed on the rating plate of the module is displayed in the "Product No." column.[*1]
  Note that "-" is displayed for a modules that does not support the product number display.

[*1] The product number is displayed only for the Universal model QCPU.

☒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 is upgraded when a new function is added.

CHAPTER2 PERFORMANCE SPECIFICATION

The table below shows the performance specifications of the CPU module.

2.1 Basic model QCPU

Table2.1 Performance Specifications

Item		Basic model QCPU		
		Q00JCPU	Q00CPU	Q01CPU
Control method		Sequence program control method		
I/O control mode		Refresh mode (Direct access I/O is available by specifying direct access I/O (DX□, DY□).)		
Program language	Sequence control language	Relay symbol language, logic symbolic language, MELSAP3 (SFC), MELSAP-L, function block, and structured text (ST)		
	Process control language	----		
Processing speed (sequence instruction)	LD X0	200ns	160ns	100ns
	MOV D0 D1	700ns	560ns	350ns
Processing speed (redundant function)	Tracking execution time (extended scan time)	----		
Constant scan (Function for keeping regular scan time)		1 to 2000ms (Setting available in1ms unit.) (Setting by parameters)		
Program capacity *1, *2		8k steps (32k bytes)		14k steps (56k bytes)
Memory capacity*1	Program memory (drive 0)	58k bytes	94k bytes	
	Memory card (RAM) (drive 1)	----		
	Memory card (ROM) (drive 2)	----		
	Standard RAM (drive 3)	0	128k bytes *3	
	Standard ROM (drive 4)	58k bytes	94k bytes	
	CPU shared memory*3, *4	----	1k bytes	

*1 : The unit of the file size stored in the memory area varies depending on the CPU module. For the details, refer to the manual below.
☞ Section 5.4.4

*2 : The maximum number of executable sequence steps is as shown below.(Program capacity) - (File header size (default: 34 steps))
For the details, refer to the manual below.
☞ Refer to CHAPTER 5 for details of the program capacity and file.

*3 : The capacity has been increased due to the function upgrade of the CPU module. (☞ Appendix 4.1)

*4 : The CPU shared memory is not latched.
(☞ QCPU User's Manual (Multiple CPU System)
The CPU shared memory is cleared by turning on the PLC or resetting the CPU module.

Table2.1 Performance Specifications

Item			Basic model QCPU		
			Q00JCPU	Q00CPU	Q01CPU
Max. number of files stored	Program memory		6 *5		
	Memory card (RAM)		----		
	Memory card (ROM)	Flash card	----		
		ATA card	----		
	Standard RAM		----	1	
	Standard ROM		6 *5		
No. of times of writing data into the standard ROM			Max. 100000 times		
No. of I/O device points (No. of points usable on program)			2048 points (X/Y0 to 7FF)		
No. of I/O points (No. of points accessible to the actual I/O module)			256 points (X/Y0 to FF)	1024 points (X/Y0 to 3FF)	
No. of device points	Internal relay [M]		8192 points by default (M0-8191) (changeable)		
	Latch relay [L]		2048 points by default (L0 to 2047) (changeable)		
	Link relay [B]		2048 points by default (B0 to 7FF) (changeable)		
	Timer [T]*6		512 points by default (T0 to 511) (changeable) (Sharing of low- and high-speed timers)		
			The low- and high-speed timers are specified by the instructions. The measurement unit of the low- and high-speed timers is set up by parameters. (Low-speed timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)		
	Retentive timer [ST]*6		0 point by default (sharing of the low- and high-speed retentive timers) (changeable) The low- and high-speed retentive timers are specified by the instructions. The measurement unit of the low- and high-speed retentive timers is set up by parameters. (Low-speed retentive timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed retentive timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)		
	Counter [C]*6		• Normal counter: 512 points by default (C0 to 511) (changeable) • Interrupt counter: 128 points max. (0 point by default, setting by parameters)		
	Data register [D]*6		11136 points by default (D0 to 11135) (changeable)		
	Link register [W]*6		2048 points by default (W0 to 7FF) (changeable)		
	Annunciator [F]*6		1024 points by default (F0 to 1023) (changeable)		
	Edge relay [V]*6		1024 points by default (V0 to 1023) (changeable)		
	File register	[R]	----	32768 points (R0 to 32767)/block (The number of device points is fixed)	
		[ZR]	----	65536 points (ZR0 to 65535) (The number of device points is fixed)	
	Link special relay [SB]		1024 points (SB0 to 3FF) (The number of device points is fixed)		
	Link special register [SW]		1024 points (SW0 to 3FF) (The number of device points is fixed)		

*5 : Each of parameter, intelligent function module parameter, sequence program, SFC program, device comment, and device initial value files can be stored.

*6 : The number of points can be changed within the setting range. (☞ Section 10.2)

Table2.1 Performance Specifications

Item			Basic model QCPU		
			Q00JCPU	Q00CPU	Q01CPU
No. of device points	Step relay [S] [*7]		2048 points (S0 to 127/block) (The number of device points is fixed)		
	Index register [Z]		10 points (Z0 to 9) (The number of device points is fixed)		
	Pointer [P]		300 points (P0 to 299) (The number of device points is fixed)		
	Interrupt pointer [I]		128 points (I0 to 127) The cyclic interval of system interrupt pointers I28 to 31 can be set up by parameters. (2 to 1000ms, 1ms unit) Default values I28: 100ms, I29: 40ms, I30: 20ms, I31: 10ms		
	Special relay [SM]		1024 points (SM0 to 1023) (The number of device points is fixed)		
	Special register [SD]		1024 points (SD0 to 1023) (The number of device points is fixed)		
	Function input [FX]		16 points (FX0 to F) (The number of device points is fixed)		
	Function output [FY]		16 points (FY0 to F) (The number of device points is fixed)		
	Function register [FD]		5 points (FD0 to 4) (The number of device points is fixed)		
No. of device tracking words			----		
Link direct device			Device for accessing the link device directly. Exclusively used for CC-Link IE controller network and MELSECNET/H. Specified form: J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□		
Intelligent function module device			Device for accessing the buffer memory of the intelligent function module directly. Specified form: U□□\G□□		
Latch range			L0 to 2047 (default) (Latch range can be set up for B, F, V, T, ST, C, D, and W.) (Setting by parameters.)		
RUN/PAUSE contact			One contact can be set up in X0 to 7FF for each of RUN and PAUSE. (Setting by parameters.)		
Timer function			Year, month, date, hour, minute, second and day-of-week (leap year automatically identified) Accuracy: -3.2 to +5.27s(TYP.+1.98s)/d at 0℃ Accuracy: -2.57 to +5.27s(TYP.+2.22s)/d at 25℃ Accuracy: -11.68 to +3.65s(TYP.-2.64s)/d at 55℃		
Allowable instantaneous power failure period			20ms or less (100VAC or more)	Varies depending on the power supply module.	
5VDC internal current consumption			0.26A [*8]	0.25A	0.27A
External dimensions		H	98mm (3.86inch)	98mm (3.86inch)	
		W	245mm(9.65inch) [*9]	27.4mm (1.08inch)	
		D	98mm (3.86inch)	89.3mm (3.52inch)	
Weight			0.66kg [*9]	0.13kg	

[*7] : The step relay is the device for the SFC function.

[*8] : The value includes those of the CPU module and base unit.

[*9] : The value includes those of the CPU module, base unit, and power supply module.

Remark

Refer to the following manual for the general specifications.

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

2.2 High performance model QCPU

Table2.2 Performance Specifications

Item		High performance model QCPU				
		Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU
Control method		Sequence program control method				
I/O control mode		Refresh mode (Direct access I/O is available by specifying direct access I/O (DX□, DY□).)				
Program language	Sequence control language	Relay symbol language, logic symbolic language, MELSAP3 (SFC), MELSAP-L, function block, and structured text (ST)				
	Process control language	----				
Processing speed (sequence instruction)	LD X0	79ns	34ns			
	MOV D0 D1	237ns	102ns			
Processing speed (redundant function)	Tracking execution time (extended scan time)	----				
Constant scan (Function for keeping regular scan time)		0.5 to 2000ms (Setting available in 0.5ms unit.) (Setting by parameters.)				
Program capacity *1, *2		28k steps (112k bytes)		60k steps (240k bytes)	124k steps (496 bytes)	252k steps (1008 bytes)
Memory capacity*1	Program memory (drive 0)	112k bytes		240k bytes	496k bytes	1008k bytes
	Memory card (RAM) (drive 1)	Capacity of the installed memory card (2M bytes max.)				
	Memory card (ROM) (drive 2)	Capacity of the installed memory card (Flash card: 4M bytes max., ATA card: 32M bytes max.)				
	Standard RAM (drive 3)	64k bytes	128k bytes *3		256k bytes *3	
	Standard ROM (drive 4)	112k bytes		240k bytes	496k bytes	1008k bytes
	CPU shared memory *3 , *4	8k bytes				

*1 : The unit of the file size stored in the memory area varies depending on the CPU module. For the details, refer to the manual below.
(☞ Section 5.4.4)

*2 : The maximum number of executable sequence steps is as shown below.(Program capacity) - (File header size (default: 34 steps))For the details, refer to the manual below.
☞ Refer to CHAPTER 5 for details of the program capacity and file.

*3 : The capacity has been increased due to the function upgrade of the CPU module. (☞ Appendix 4)

*4 : The CPU shared memory is not latched. (☞ QCPU User's Manual (Multiple CPU System)
The CPU shared memory is cleared by turning on the PLC or resetting the CPU module.

Table2.2 Performance Specifications

Item			High performance model QCPU				
			Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU
Max. number of files stored	Program memory		28		60	124	252 *6
	Memory card (RAM)		287 (When the Q2MEM-2MBS is used)				
	Memory card (ROM)	Flash card	288				
		ATA card	512				
	Standard RAM		3 *5				
	Standard ROM		28		60	124	252
No. of times of writing data into the standard ROM			Max. 100000 times				
No. of I/O device points (No. of points usable on program)			8192 points (X/Y0 to 1FFF)				
No. of I/O points (No. of points accessible to the actual I/O module)			4096 points (X/Y0 to FFF)				
No. of device points	Internal relay [M]		8192 points by default (M0-8191) (changeable)				
	Latch relay [L]		8192 points by default (L0 to 8191) (changeable)				
	Link relay [B]		8192 points by default (B0 to 1FFF) (changeable)				
	Timer [T]		2048 points by default (T0 to 2047) (sharing of low- and high-speed timers) (changeable)				
			The low- and high-speed timers are specified by the instructions. The measurement unit of the low- and high-speed timers is set up by parameters. (Low-speed timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)				
	Retentive timer [ST]		0 point by default (sharing of the low- and high-speed retentive timers) (changeable) The low- and high-speed retentive timers are specified by the instructions. The measurement unit of the low- and high-speed retentive timers is set up by parameters. (Low-speed retentive timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed retentive timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)				
	Counter [C]		• Normal counter, 1024 points by default (C0 to 1023) (changeable) • Interrupt counter: 256 points max. (0 point by default, setting by parameters)				
	Data register [D]		12288 points by default (D0 to 12287) (changeable)				
	Link register [W]		8192 points by default (W0 to 1FFF) (changeable)				
	Annunciator [F]		2048 points by default (F0 to 2047) (changeable)				
	Edge relay [V]		2048 points by default (V0 to 2047) (changeable)				

*5 : Extended by the upgraded functions of CPU module (☞ Appendix 4.2)

*6 : The CPU module can execute up to 124 programs. The CPU module cannot execute 125 or more programs.

*7 : The number of points can be changed within the setting range. (☞ Section 10.2)

Table2.2 Performance Specifications

Item				High performance model QCPU				
				Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU
No. of device points	File register*9	[R]	Standard RAM	32768 points (R0 to 32767)	32768 points(R0 to 32767) Up to 65536 points can be used by block switching in units of 32768 points		32768 points(R0 to 32767) Up to 131072 points can be used by block switching in units of 32768 points	
			SRAM card (1M bytes)	Up to 517120 points can be used by block switching in units of 32768 points (R0 to 32767).				
			SRAM card (2M bytes)	Up to 1041408 points can be used by block switching in units of 32768 points (R0 to 32767).				
			Flash card (2M bytes)	Up to 1041408 points can be used by block switching in units of 32768 points (R0 to 32767).				
			Flash card (4M bytes)	Up to 1042432 points can be used by block switching in units of 32768 points (R0 to 32767).				
		[ZR]	Standard RAM	32768 points (R0 to 32767)	65536 point (ZR0 to 65535) Block switching not required.		131072 points (ZR0 to 131071) Block switching not required.	
			SRAM card (1M bytes)	517120 points (ZR0 to 517119), Block switching not required.				
			SRAM card (2M bytes)	1041408 points (ZR0 to 1041407),Block switching not required.				
			Flash card (2M bytes)	1041408 points (ZR0 to 1041407),Block switching not required.				
			Flash card (4M bytes)	1042432 points (ZR0 to 1042431),Block switching not required.				
	Link special relay [SB]			2048 points (SB0 to 7FF) (The number of device points is fixed)				
	Link special register [SW]			2048 points (SW0 to 7FF) (The number of device points is fixed)				
	Step relay [S] *8			8192 points (S0 to 8191) (The number of device points is fixed)				
	Index register [Z]			16 points (Z0 to 15) (The number of device points is fixed)				
	Pointer [P]			4096 points (P0 to 4095) (The number of device points is fixed), The use ranges of the local pointers and common pointers can be set up by parameters.				
	Interrupt pointer [I]			256 points (I0 to 255) (The number of device points is fixed) The constant cyclic interval of system interrupt pointers I28 to 31 can be set up by parameters. (0.5 to 1000ms, 0.5ms unit) Default values I28: 100ms, I29: 40ms, I30: 20ms, I31: 10ms				

*8 : Read only when Flash card is used. ATA card is unusable.

*9 : The step relay is the device for the SFC function.

Table2.2 Performance Specifications

Item		High performance model QCPU				
		Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU
No. of device points	Special relay [SM]	2048 points (SM0 to 2047) (The number of device points is fixed)				
	Special register [SD]	2048 points (SM0 to 2047) (The number of device points is fixed)				
	Function input [FX]	16 points (FX0 to F) (The number of device points is fixed)				
	Function output [FY]	16 points (FY0 to F) (The number of device points is fixed)				
	Function register [FD]	5 points (FD0 to 4) (The number of device points is fixed)				
No. of device tracking words		----				
Link direct device		Device for accessing the link device directly Dedicated to CC-Link IE controller network and MELSECNET/H. Specified form: J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□				
Intelligent function module device		Device for accessing the buffer memory of the intelligent function module directly. Specified form: U□□\G□□				
Latch range		L0 to 8191 (default) (Latch range can be set up for B, F, V, T, ST, C, D, and W.) (Setting by parameters)				
RUN/PAUSE contact		One contact can be set up in X0 to 1FFF for each of RUN and PAUSE. (Setting by parameters)				
Timer function		Year, month, date, hour, minute, second, and day-of-week (leap year automatically identified) Accuracy: -3.18 to +5.25s(TYP.+2.12s)/d at 0℃ Accuracy: -3.93 to +5.25s(TYP.+1.90s)/d at 25℃ Accuracy: -14.69 to +3.53s(TYP.-3.67s)/d at 55℃				
Allowable instantaneous power failure period		Varies depending on the power supply module.				
5VDC internal current consumption		0.60A	0.64A			
External dimensions	H	98mm (3.86inch)				
	W	27.4mm (1.08inch)				
	D	89.3mm (3.52inch)				
Weight		0.20kg				

Remark

Refer to the following manual for the general specifications.

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

2.3 Process CPU

Table2.3 Performance Specifications

Item		Process CPU			
		Q02PHCPU	Q06PHCPU	Q12PHCPU	Q25PHCPU
Control method		Sequence program control method (Direct access I/O is available by specifying direct access I/O (DX□, DY□))			
I/O control mode		Refresh mode			
Program language	Sequence control language	Relay symbol language, logic symbolic language, MELSAP3 (SFC), MELSAP-L, function block and structured text (ST)			
	Process control language	FBD for process control (Programming by PX Developer)			
Processing speed (sequence instruction)	LD X0	34ns			
	MOV D0 D1	102ns			
Processing speed (redundant function)	Tracking execution time (extended scan time)	----			
Constant scan (Function for keeping regular scan time)		0.5 to 2000ms (Setting available in 0.5ms unit.) (Setting by parameters)			
Program capacity [*1, *2]		28k steps (112k bytes)	60k steps (240k bytes)	124k steps (496k bytes)	252k steps (1008k bytes)
Memory capacity[*1]	Program memory (drive 0)	112k bytes	240k bytes	496k bytes	1008k bytes
	Memory card (RAM) (drive 1)	Capacity of the installed memory card (2M bytes max.)			
	Memory card (ROM) (drive 2)	Capacity of the installed memory card (Flash card: 4M bytes max., ATA card: 32M bytes max.)			
	Standard RAM (drive 3)	128k bytes		256k bytes	
	Standard ROM (drive 4)	112k bytes	240k bytes	496k bytes	1008k bytes
	CPU shared memory [*3]	8k bytes			

*1 : The unit of the file size stored in the memory area varies depending on the CPU module. For the details, refer to the manual below.
(☞ Section 5.4.4)

*2 : The maximum number of executable sequence steps is as shown below.(Program capacity) - (File header size (default: 34 steps))
For the details, refer to the manual below.
(☞ Refer to CHAPTER 5 for details of the program capacity and file.)

*3 : The CPU shared memory is not latched. (☞ QCPU User's Manual (Multiple CPU System)
The CPU shared memory is cleared by turning on the PLC or resetting the CPU module.

Table2.3 Performance Specifications

Item			Process CPU			
			Q02PHCPU	Q06PHCPU	Q12PHCPU	Q25PHCPU
Max. number of files stored	Program memory		28	60	124	252 *5
	Memory card (RAM)		287(When the Q2MEM-2MBS is used)			
	Memory card (ROM)	Flash card	288			
		ATA card	512			
	Standard RAM		3 *4			
	Standard ROM		28	60	124	252
No. of times of writing data into the standard ROM			Max. 100000 times			
No. of I/O device points (No. of points usable on program)			8192 points (X/Y0 to 1FFF)			
No. of I/O points (No. of points accessible to the actual I/O module)			4096 points (X/Y0 to FFF)			
No. of device points	Internal relay [M]*6		8192 points by default (M0 to 8191) (changeable)			
	Latch relay [L]*6		8192 points by default (L0 to 8191) (changeable)			
	Link relay [B]*6		8192 points by default (B0 to 1FFF) (changeable)			
	Timer [T]*6		2048 points by default (T0 to 2047) (sharing of low- and high-speed timers) (changeable) The low- and high-speed timers are specified by the instructions. The measurement unit of the low- and high-speed timers is set up by parameters. (Low-speed timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)			
	Retentive timer [ST]*6		0 point by default (sharing of low- and high-speed retentive timers) (changeable) The low- and high-speed retentive timers are specified by the instructions. The measurement unit of the low- and high-speed retentive timers is set up by parameters. (Low-speed retentive timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed retentive timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)			
	Counter [C]*6		• Normal counter, 1024 points by default (C0 to 1023) (changeable) • Interrupt counter: 256 points max. (0 point by default, setting by parameters)			
	Data register [D]*6		12288 points by default (D0 to 12287) (changeable)			
	Link register [W]*6		8192 points by default (W0 to 1FFF) (changeable)			
	Annunciator [F]*6		2048 points by default (F0 to 2047) (changeable)			
	Edge relay [V]*6		2048 points by default (V0 to 2047) (changeable)			

*4 : Extended by the upgraded functions of CPU module (☞ Appendix 4.3)

*5 : The CPU module can execute up to 124 programs. The CPU module cannot execute 125 or more programs.

*6 : The number of points can be changed within the setting range. (☞ Section 10.2)

Table2.3 Performance Specifications

Item				Process CPU			
				Q02PHCPU	Q06PHCPU	Q12PHCPU	Q25PHCPU
No. of device points	File register*7	[R]	Standard RAM	Up to 65536 points can be used by block switching in units of 32768 points (R0 to 32767).		Up to 131072 points can be used by block switching in units of 32768 points (R0 to 32767).	
			SRAM card (1M bytes)	Up to 517120 points can be used by block switching in units of 32768 points (R0 to 32767).			
			SRAM card (2M bytes)	Up to 1041408 points can be used by block switching in units of 32768 points (R0 to 32767).			
			Flash card (2M bytes)	Up to 1041408 points can be used by block switching in units of 32768 points (R0 to 32767).			
			Flash card (4M bytes)	Up to 1042432 points can be used by block switching in units of 32768 points (R0 to 32767).			
		[ZR]	Standard RAM	65536 points (ZR0 to 131071), block switching not required.		131072 points (ZR0 to 131071), block switching not required.	
			SRAM card (1M bytes)	517120 points (ZR0 to 517119), block switching not required.			
			SRAM card (2M bytes)	1041408 points (ZR0 to 1041407), block switching not required.			
			Flash card (2M bytes)	1041408 points (ZR0 to 1041407), block switching not required.			
			Flash card (4M bytes)	1042432 points (ZR0 to 1042431), block switching not required.			
	Link special relay [SB]			2048 points (SB0 to 7FF) (The number of device points is fixed)			
	Link special register [SW]			2048 points (SW0 to 7FF) (The number of device points is fixed)			
	Step relay [S] *8			8192 points (S0 to 8191) (The number of device points is fixed)			
	Index register [Z]			16 points (Z0 to 15) (The number of device points is fixed)			
	Pointer [P]			4096 points (P0 to 4095) (The number of device points is fixed), The use ranges of the local pointers and common pointers can be set up by parameters.			
	Interrupt pointer [I]			256 points (I0 to 255) (The number of device points is fixed), The constant cyclic interval of system interrupt pointers I28 to 31 can be set up by parameters. (0.5 to 1000ms, 0.5ms unit) Default values I28: 100ms, I29: 40ms, I30: 20ms, I31: 10ms			

*7 : Read only when Flash card is used. ATA card is unusable.

*8 : The step relay is the device for the SFC function.

Table2.3 Performance Specifications

Item		Process CPU			
		Q02PHCPU	Q06PHCPU	Q12PHCPU	Q25PHCPU
No. of device points	Special relay [SM]	2048 points (SM0 to 2047) (The number of device points is fixed)			
	Special register [SD]	2048 points (SD0 to 2047) (The number of device points is fixed)			
	Function input [FX]	16 points (FX0 to F) (The number of device points is fixed)			
	Function output [FY]	16 points (FY0 to F) (The number of device points is fixed)			
	Function register [FD]	5 points (FD0 to 4) (The number of device points is fixed)			
No. of device tracking words		----			
Link direct device		Device for accessing the link device directly. Exclusively used for CC-Link IE controller network and MELSECNET/H. Specified form: J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□			
Intelligent function module device		Device for accessing the buffer memory of the intelligent function module directly. Specified form: U□□\G□□			
Latch range		L0 to 8191(default) (Latch range can be set up for B, F, V, T, ST, C, D, and W.) (Setting by parameters)			
RUN/PAUSE contact		One contact can be set up in X0 to 1FFF for each of RUN and PAUSE. (Setting by parameters)			
Timer function		Year, month, date, hour, minute, second, and day-of-week (leap year automatically identified) Accuracy: -3.18 to +5.25s(TYP.+2.12s)/d at 0℃ Accuracy: -3.93 to +5.25s(TYP.+1.90s)/d at 25℃ Accuracy: -14.69 to +3.53s(TYP.-3.67s)/d at 55℃			
Allowable instantaneous power failure period		Varies depending on the power supply module.			
5VDC internal current consumption		0.64A			
External dimensions	H	98mm (3.86inch)			
	W	27.4mm (1.08inch)			
	D	89.3mm (3.52inch)			
Weight		0.20kg			

Remark

Refer to the following manual for the general specifications.

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

2.4 Redundant CPU

Table2.4 Performance Specifications

Item		Redundant CPU	
		Q12PRHCPU	Q25PRHCPU
Control method		Sequence program control method (Direct access I/O is available by specifying direct access I/O (DX□, DY□))	
I/O control mode		Refresh mode (Programming by PX Developer)	
Program language	Sequence control language	Relay symbol language, logic symbolic language, MELSAP3 (SFC), MELSAP-L, function block and structured text (ST)	
	Process control language	FBD for process control	
Processing speed (sequence instruction)	LD X0	34ns	
	MOV D0 D1	102ns	
Processing speed (redundant function)	Tracking execution time (extended scan time)	Device memory 48k words: 10ms, Device memory 100k words: 15ms (☞ QnPRHCPU User's Manual (Redundant System))	
Constant scan (Function for keeping regular scan time)		0.5 to 2000ms (Setting available in 0.5ms unit.) (Setting by parameters)	
Program capacity [*1, *2]		124k steps (496 bytes)	252k steps (1008 bytes)
Memory capacity[*1]	Program memory (drive 0)	496k bytes	1008k bytes
	Memory card (RAM) (drive 1)	Capacity of the installed memory card (2M bytes max.)	
	Memory card (ROM) (drive 2)	Capacity of the installed memory card (Flash card: 4M bytes max., ATA card: 32M bytes max.)	
	Standard RAM (drive 3)	Capacity of the installed memory card (2M bytes max.)	
	Standard ROM (drive 4)	496k bytes	1008k bytes
	CPU shared memory	----	

*1 : The unit of the file size stored in the memory area varies depending on the CPU module. For the details, refer to the manual below.
☞ Section 5.4.4

*2 : The maximum number of executable sequence steps is as shown below.(Program capacity) - (File header size (default: 34 steps))
For the details, refer to the manual below.
☞ Refer to CHAPTER 5 for details of the program capacity and file.

Table2.4 Performance Specifications

Item			Redundant CPU	
			Q12PRHCPU	Q25PRHCPU
Max. number of files stored	Program memory		124	252 *4
	Memory card (RAM)		287 (When the Q2MEM-2MBS is used)	
	Memory card (ROM)	Flash card	288	
		ATA card	512	
	Standard RAM		3 *3	
	Standard ROM		124	252
No. of times of writing data into the standard ROM			Max. 100000 times	
No. of I/O device points (No. of points usable on program)			8192 points (X/Y0 to 1FFF)	
No. of I/O points (No. of points accessible to the actual I/O module)			4096 points (X/Y0 to FFF)	
No. of device points	Internal relay [M]*5		8192 points by default (M0 to 8191) (changeable)	
	Latch relay [L]*5		8192 points by default (L0 to 8191) (changeable)	
	Link relay [B]*5		8192 points by default (B0 to 1FFF) (changeable)	
	Timer [T]*5		2048 points by default (T0 to 2047) (sharing of low- and high-speed timers) (changeable) The low- and high-speed timers are specified by the instructions. The measurement unit of the low- and high-speed timers is set up by parameters. (Low-speed timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)	
	Retentive timer [ST]*5		0 point by default (sharing of low- and high-speed retentive timers) (changeable) The low- and high-speed retentive timers are specified by the instructions. The measurement unit of the low- and high-speed retentive timers is set up by parameters. (Low-speed retentive timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed retentive timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)	
	Counter [C]*5		• Normal counter, 1024 points by default (C0 to 1023) (changeable) • Interrupt counter: 256 points max. (0 point by default, setting by parameters)	
	Data register [D]*5		12288 points by default (D0 to 12287) (changeable)	
	Link register [W]*5		8192 points by default (W0 to 1FFF) (changeable)	
	Annunciator [F]*5		2048 points by default (F0 to 2047) (changeable)	
	Edge relay [V]*5		2048 points by default (V0 to 2047) (changeable)	

*3 : Extended by the upgraded functions of CPU module (☞ Appendix 4.4)

*4 : The CPU module can execute up to 124 programs. The CPU module cannot execute 125 or more programs.

*5 : The number of points can be changed within the setting range. (☞ Section 10.2)

Table2.4 Performance Specifications

Item				Redundant CPU	
				Q12PRHCPU	Q25PRHCPU
No. of device points	File register*6	[R]	Standard RAM	Up to 131072 points can be used by block switching in units of 32768 points (R0 to 32767).	
			SRAM card (1M bytes)	Up to 517120 points can be used by block switching in units of 32768 points (R0 to 32767).	
			SRAM card (2M bytes)	Up to 1041408 points can be used by block switching in units of 32768 points (R0 to 32767).	
			Flash card (2M bytes)	Up to 1041408 points can be used by block switching in units of 32768 points (R0 to 32767).	
			Flash card (4M bytes)	Up to 1042432 points can be used by block switching in units of 32768 points (R0 to 32767).	
		[ZR]	Standard RAM	131072 points (ZR0 to 131071), block switching not required.	
			SRAM card (1M bytes)	517120 points (ZR0 to 517119), block switching not required.	
			SRAM card (2M bytes)	1041408 points (ZR0 to 1041407), block switching not required.	
			Flash card (2M bytes)	1041408 points (ZR0 to 1041407), block switching not required.	
			Flash card (4M bytes)	1042432 points (ZR0 to 1042431), block switching not required.	
	Link special relay [SB]			2048 points (SB0 to 7FF) (The number of device points is fixed)	
	Link special register [SW]			2048 points (SW0 to 7FF) (The number of device points is fixed)	
	Step relay [S] *7			8192 points (S0 to 8191) (The number of device points is fixed)	
	Index register [Z]			16 points (Z0 to 15) (The number of device points is fixed)	
	Pointer [P]			4096 points (P0 to 4095) (The number of device points is fixed), The use ranges of the local pointers and common pointers can be set up by parameters.	
	Interrupt pointer [I]			256 points (I0 to 255) (The number of device points is fixed), The constant cyclic interval of system interrupt pointers I28 to 31 can be set up by parameters. (0.5 to 1000ms, 0.5ms unit) Default values I28: 100ms, I29: 40ms, I30: 20ms, I31: 10ms	

*6 : Read only when Flash card is used. ATA card is unusable.

*7 : The step relay is the device for the SFC function.

Table2.4 Performance Specifications

Item			Redundant CPU	
			Q12PRHCPU	Q25PRHCPU
No. of device points	Special relay [SM]		2048 points (SM0 to 2047) (The number of device points is fixed)	
	Special register [SD]		2048 points (SD0 to 2047) (The number of device points is fixed)	
	Function input [FX]		16 points (FX0 to F) (The number of device points is fixed)	
	Function output [FY]		16 points (FY0 to F) (The number of device points is fixed)	
	Function register [FD]		5 points (FD0 to 4) (The number of device points is fixed)	
No. of device tracking words			Max. 100k words (☞ QnPRHCPU User's Manual (Redundant System))	
Link direct device			Device for accessing the link device directly. Exclusively used for CC-Link IE controller network and MELSECNET/H. Specified form: J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□	
Intelligent function module device			Device for accessing the buffer memory of the intelligent function module directly. Specified form: U□□\G□□	
Latch range			L0 to 8191 (default) (Latch range can be set up for B, F, V, T, ST, C, D, and W.) (Setting by parameters)	
RUN/PAUSE contact			One contact can be set up in X0 to 1FFF for each of RUN and PAUSE. (Setting by parameters)	
Timer function			Year, month, date, hour, minute, second, and day-of-week (leap year automatically identified) Accuracy: -3.2 to +5.27s (TYP.+2.07s)/d at 0℃ Accuracy: -2.77 to +5.27s (TYP.+2.22s)/d at 25℃ Accuracy: -12.14 to +3.65s (TYP.-2.89s)/d at 55℃	
Allowable instantaneous power failure period			Varies depending on the power supply module.	
5VDC internal current consumption			0.89A	
External dimensions		H	98mm (3.86inch)	
		W	55.2mm (2.17inch)	
		D	89.3mm (3.52inch)	
Weight			0.30kg	

Remark

Refer to the following manual for the general specifications.

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

2.5 Universal model QCPU

Table2.5 Performance Specifications

Item			Universal model QCPU					
			Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	Q13UDHCPU	Q26UDHCPU
				Q03UDECPU	Q04UDEHCPU	Q06UDEHCPU	Q13UDEHCPU	Q26UDHECPU
Control method			Sequence program control method					
I/O control mode			Refresh mode (Direct access I/O is available by specifying direct access I/O (DX□, DY□))					
Program language	Sequence control language		Relay symbol language, logic symbolic language, MELSAP3 (SFC), MELSAP-L, function block and structured text (ST)					
	Process control language		----					
Processing speed (sequence instruction)	LD X0		40ns	20ns	9.5ns			
	MOV D0 D1		80ns	40ns	19ns			
Processing speed (redundant function)	Tracking execution time (extended scan time)		----					
Constant scan (Function for keeping regular scan time)			0.5 to 2000ms(Setting available in 0.5ms unit.) (Setting by parameters)					
Program capacity *1, *2			20k steps (80k bytes)	30k steps (120k bytes)	40k steps (160k bytes)	60k steps (240k bytes)	130k steps (520k bytes)	260k steps (1040k bytes)
Memory capacity*1	Program memory (drive 0)		80k bytes	120k bytes	120k bytes	240k bytes	520k bytes	1040k bytes
	Memory card (RAM) (drive 1)		Capacity of the installed memory card (8M bytes max.)					
	Memory card (ROM) (drive 2)		Capacity of the installed memory card (Flash card: 4M bytes max., ATA card: 32M bytes max.)					
	Standard RAM (drive 3)		128k bytes	192k bytes	256k bytes	768k bytes	1024k bytes	1280k bytes
	Standard ROM (drive 4)		512k bytes	1024k bytes			2048k bytes	4096k bytes
	CPU shared memory *3	QCPU standard memory	8k bytes					
		Multiple CPU high speed transmission area	----	32k bytes				

*1 : The unit of the file size stored in the memory area varies depending on the CPU module. For the details, refer to the manual below.
☞ Section 5.4.4

*2 : The maximum number of executable sequence steps is as shown below.(Program capacity) - (File header size (default: 34 steps))
For the details, refer to the manual below.
☞ Refer to CHAPTER 5 for details of the program capacity and file.

*3 : The capacity has been increased due to the function upgrade of the CPU module. (☞ Appendix 4)

Table2.5 Performance Specifications

Item			Universal model QCPU					
			Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	Q13UDHCPU	Q26UDHCPU
				Q03UDECPU	Q04UDEHCPU	Q06UDEHCPU	Q13UDEHCPU	Q26UDHECPU
Max. number of files stored	Program memory		64	124			252 *4	
	Memory card (RAM)		319 (When the Q3MEM-8MBS is used) (One file register, one local device and sampling trace file only)					
	Memory card (ROM)	Flash card	288					
		ATA card	511					
	Standard RAM		3 (One file for each file register, local device, and sampling trace)					
	Standard ROM		128	256				
No. of times of writing data into the program memory			Max. 100000 times*5					
No. of times of writing data into the standard ROM			Max. 100000 times*6					
No. of I/O device points (No. of points usable on program)			8192 points (X/Y0 to 1FFF)					
No. of I/O points (No. of points accessible to the actual I/O module)			2048 points (X/Y0 to 7FF)	4096 points (X/Y0 to FFF)				
No. of device points	Internal relay [M]*7		8192 points by default (M0-8191) (changeable)					
	Latch relay [L]*7		8192 points by default (L0-8191) (changeable)					
	Link relay [B]*7		8192 points by default (B0 to 1FFF) (changeable)					
	Timer [T]*7		2048 points by default (T0 to 2047) (changeable) (Sharing of low- and high-speed timers) The low- and high-speed timers are specified by the instructions. The measurement unit of the low- and high-speed timers is set up by parameters. (Low-speed timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)					
	Retentive timer [ST]*7		0 point by default (sharing of the low- and high-speed retentive timers) (changeable) The low- and high-speed retentive timers are specified by the instructions. The measurement unit of the low- and high-speed retentive timers is set up by parameters. (Low-speed retentive timer: 1 to 1000ms, 1ms unit, 100ms by default) (High-speed retentive timer: 0.1 to 100ms, 0.1ms unit, 10ms by default)					
	Counter [C]*7		• Normal counter, 1024 points by default (C0 to 1023) (changeable)					
	Data register [D]*7		12288 points by default (D0 to 12287) (changeable)					
	Extended data register [D]		0 points by default (changeable)					
	Link register [W]*7		8192 points by default (W0 to 1FFF) (changeable)					
	Extended link register [W]		0 points by default (changeable)					
	Annunciator [F]*7		2048 points by default (F0 to 2047) (changeable)					
	Edge relay [V]*7		2048 points by default (V0 to 2047) (changeable)					
	Link special relay [SB]*7		2048 points by default (SB0 to 7FF) (changeable)					
	Link special register [SW]*7		2048 points by default (SW0 to 7FF) (changeable)					

*4 : The CPU module can execute up to 124 programs. The CPU module cannot execute 125 or more programs.

*5 : The one write operation may not be counted as one writing to a program memory.
The number of writing to the program memory can be checked by the special register (SD682 and SD683).

*6 : The one write operation may not be counted as one writing to the standard ROM.
The number of writing to the standard ROM can be checked by the special register (SD687 and SD688).

*7 : The number of points can be changed within the setting range. (☞ Section 10.2)

Table2.5 Performance Specifications

Item				Universal model QCPU					
				Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	Q13UDHCPU	Q26UDHCPU
					Q03UDECPU	Q04UDEHCPU	Q06UDEHCPU	Q13UDEHCPU	Q26UDHECPU
No. of device points	File register*8	[R]	Standard RAM	32768 points (R0 to 32767) Up to 65536 points can be used by block switching in units of 32768 points	32768 points (R0 to 32767) Up to 98304 points can be used by block switching in units of 32768 points	32768 points (R0 to 32767) Up to 131072 points can be used by block switching in units of 32768 points	32768 points (R0 to 32767) Up to 393216 points can be used by block switching in units of 32768 points	32768 points (R0 to 32767) Up to 524288 points can be used by block switching in units of 32768 points	32768 points (R0 to 32767) Up to 655360 points can be used by block switching in units of 32768 points
			SRAM card (1M bytes)	32768 points (R0 to 32767) Up to 517120 points can be used by block switching in units of 32768 points					
			SRAM card (2M bytes)	32768 points (R0 to 32767) Up to 1041408 points can be used by block switching in units of 32768 points					
			SRAM card (4M bytes)	32768 points (R0 to 32767) Up to 2087936 points can be used by block switching in units of 32768 points					
			SRAM card (8M bytes)	32768 points (R0 to 32767) Up to 4184064 points can be used by block switching in units of 32768 points					
			Flash card (2M bytes)	32768 points (R0 to 32767) Up to 1041408 points can be used by block switching in units of 32768 points					
			Flash card (4M bytes)	32768 points (R0 to 32767) Up to 2087936 points can be used by block switching in units of 32768 points					
		[ZR]	Standard RAM	65536 points (ZR0 to 65535) Block switching not required.	98304 points (ZR0 to 98303) Block switching not required.	131072 points (ZR0 to 131071) Block switching not required.	393216 points (ZR0 to 393215) Block switching not required.	524288 points (ZR0 to 524287) Block switching not required.	655360 points (ZR0 to 655359) Block switching not required.
			SRAM card (1M bytes)	517120 points (ZR0 to 517119), Block switching not required.					
			SRAM card (2M bytes)	1041408 points (ZR0 to 1041407), Block switching not required.					
			SRAM card (4M bytes)	2087936 points (ZR0 to 2087935), Block switching not required.					
			SRAM card (8M bytes)	4184064 points (ZR0 to 4182015), Block switching not required.					
			Flash card (2M bytes)	1041408 points (ZR0 to 1041407), Block switching not required.					
			Flash card (4M bytes)	2087936 points (ZR0 to 2087935), Block switching not required.					
	Step relay [S] *9			8192 points (S0 to 8191) (The number of device points is fixed)*10					
	Index register / Standard devise register [Z]			max. 20 points (Z0 to 19) (The number of device points is fixed)					
	Index register [Z] (32-bit modification specification of ZR device)			max. 10 points (Z0 to 18) (Index register (Z) is used in double words.)					
	Pointer [P]			4096 points (P0 to 4095), The use ranges of the local pointers and common pointers can be set up by parameters.					

*8 : Read only when Flash card is used. ATA card is unusable.

*9 : Step relay is a device for SFC functions.

*10 : In the Universal model QCPU whose serial number (first five digits) is "10042" or later, 0 points can be set.

Table2.5 Performance Specifications

Item			Universal model QCPU					
			Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	Q13UDHCPU	Q26UDHCPU
				Q03UDECPU	Q04UDEHCPU	Q06UDEHCPU	Q13UDEHCPU	Q26UDHECPU
No. of device points	Interrupt pointer [I]		256 points (I0 to 255) The constant cyclic interval of system interrupt pointers I28 to 31 can be set up by parameters. (0.5 to 1000ms, 0.5ms unit) Default values I28: 100ms, I29: 40ms, I30: 20ms, I31: 10ms					
	Special relay [SM]		2048 points (SM0 to 2047) (The number of device points is fixed)					
	Special register [SD]		2048 points (SM0 to 2047) (The number of device points is fixed)					
	Function input [FX]		16 points (FX0 to F) (The number of device points is fixed)					
	Function output [FY]		16 points (FY0 to F) (The number of device points is fixed)					
	Function register [FD]		5 points (FD0 to 4) (The number of device points is fixed)					
No. of device tracking words			----					
Link direct device			Device for accessing the link device directly. Dedicated to CC-Link IE controller network and MELSECNET/H. Specified form: J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□					
Intelligent function module device			Device for accessing the buffer memory of the intelligent function module directly. Specified form: U□□\G□□					
Specifications of built-in Ethernet ports of CPU module[11]	Data transfer speed		----	100/10Mbps				
	Communication mode		----	Full-duplex/Half-duplex				
	Transmission mode		----	Base band				
	Maximum distance between hub and node		----	100m (328.08feet)				
	Maximum number of nodes/ connections	10BASE -T	----	Cascade connection: Four stages at maximum				
		100BASE -TX	----	Cascade connection: Two stages at maximum				
	Number of connections[12]		----	16 for MELSOFT connection and MC protocol, 1 for FTP				
Latch range			L0 to 8191 (8192 points by default) (Latch range can be set up for B, F, V, T, ST, C, D, and W.) (Setting by parameters)					
RUN/PAUSE contact			One contact can be set up in X0 to 1FFF for each of RUN and PAUSE. (Setting by parameters)					
Timer function			Year, month, date, hour, minute, second, and day-of-week (leap year automatically identified) Accuracy: -2.96 to +3.74s (TYP.+1.42s)/d at 0℃ Accuracy: -3.18 to +3.74s (TYP.+1.50s)/d at 25℃ Accuracy: -13.20 to +2.12s (TYP.-3.54s)/d at 55℃					
Allowable instantaneous power failure period			Varies depending on the power supply module.					
5VDC internal current consumption			0.23A	0.33A[13]	0.39A[14]			
External dimensions		H	98mm (3.86inch)					
		W	27.4mm (1.08inch)					
		D	89.3mm (3.52inch)[15]					
Weight			0.20kg[15]					

*11 : The Built-in Ethernet port QCPU is targeted.

*12 : The number indicates the total number of TCP/IP and UDP/IP protocols.

*13 : The current consumption in the Q03UDECPU is 0.46A.

*14 : The current consumption in the Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, and Q26UDEHCPU is 0.49A.

*15 : The values for the Built-in Ethernet port QCPU are as follows.
External dimensions(D): 115mm (4.53inch)
Weight: 0.22kg

Remark

For the general specifications, refer to the following manual.

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

CHAPTER3 SEQUENCE PROGRAM CONFIGURATION AND EXECUTION CONDITIONS

There are three types of programs that can be executed in the CPU module: sequence program, SFC program, and ST program.

This manual does not explain the SFC program and ST program.

Refer to the following manuals for the SFC program and ST program.

☞ QCPU (Q Mode)/QnACPU Programming Manual (SFC)

☞ QCPU (Q Mode) Programming Manual (Structured Text)

(1) Program execution order of Basic model QCPU

The Basic model QCPU executes a program in the following order (Figure 3.1).

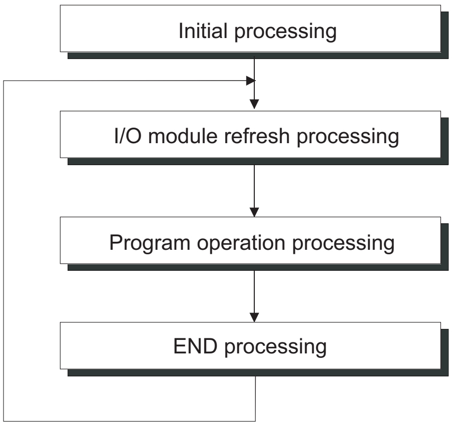


Figure 3.1 Program execution order of Basic model QCPU

(2) Program execution order of High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU

The High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU executes a program in the following order (Figure 3.2).

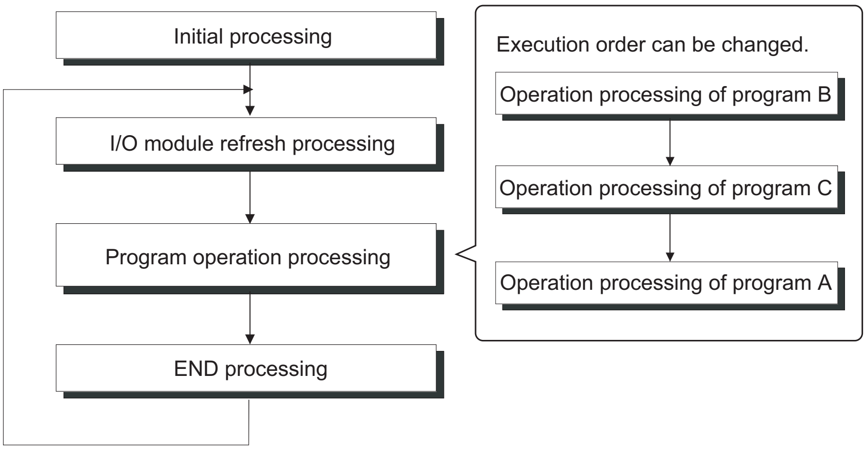
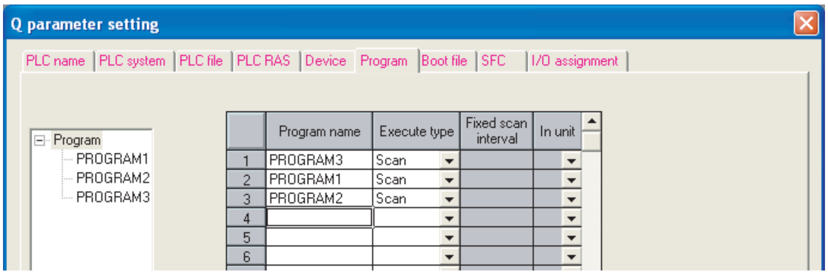

Initial processing
I/O module refresh processing
Program operation processing
END processing
Execution order can be changed.
Operation processing of program B
Operation processing of program C
Operation processing of program A

Figure 3.2 Program execution order of High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU

☒POINT

The High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU can store multiple programs.

Set the program execution order on the Program tab of PLC parameter in GX Developer. (☞ Section 3.3.6)



	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Scan		
2	PROGRAM1	Scan		
3	PROGRAM2	Scan		
4				
5				
6				

3.1 Sequence Program

Sequence program is a program created using sequence instructions, basic instructions, and application instructions.

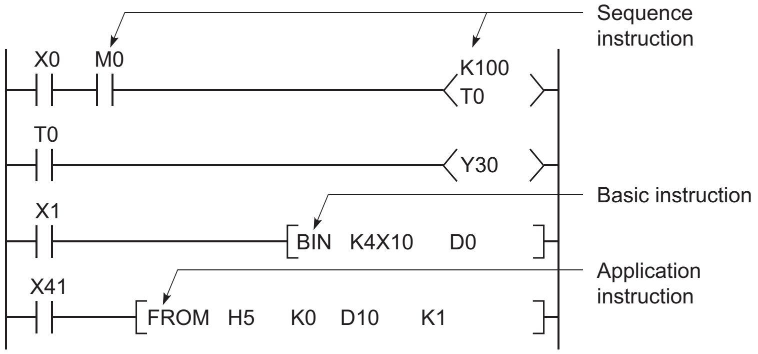


Figure 3.3 Sequence program

Remark

Refer to the following manual for the sequence instructions, basic instructions and application instructions.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

(1) Sequence program description method

There are two different methods for describing sequence programs: ladder mode and list mode.

(a) Ladder mode

The ladder mode is a mode based on the concept of a sequence circuit of relay control.

In the ladder mode, programming close to the image of sequence figure is possible.

Programming is performed in units of ladder blocks.

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

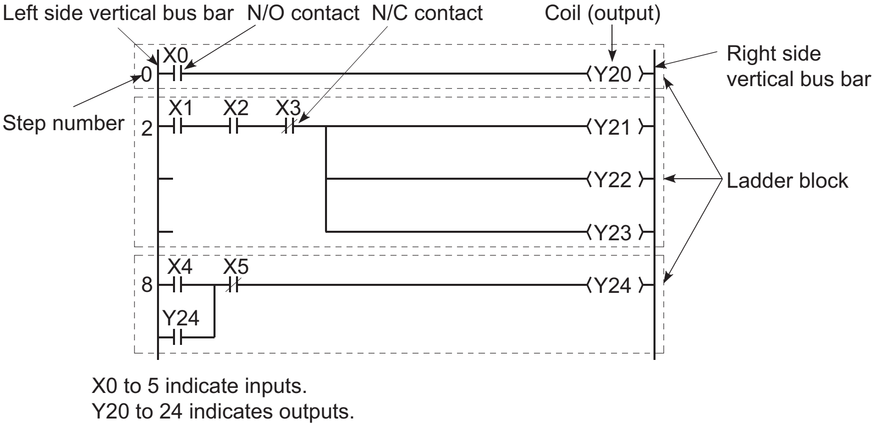


Figure 3.4 Ladder mode

(b) List mode

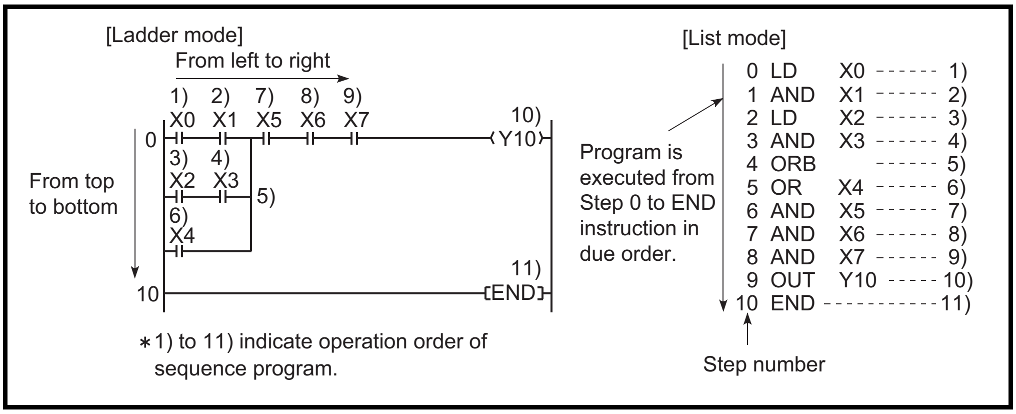
When programming in the list mode, the symbols, such as contacts and coils, used in the ladder mode are replaced with dedicated instructions.

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

(2) Sequence program operation

Program operation is executed sequentially from Step 0 to the END/FEND instruction. In the ladder mode, operation is performed from the left side vertical bus bar to the right end for each ladder block, and from the top rung to the bottom.



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

∗1) to 11) indicate operation order of sequence program.

Figure 3.5 Comparison between ladder and list modes

(3) Sequence program classification

Sequence programs are classified into the following three types.

- Sequence programs are classified into the following three types.
- Main routine program ☞ Section 3.1.1
- Subroutine program ☞ Section 3.1.2
- Interrupt program ☞ Section 3.1.3

Basic Note3.1

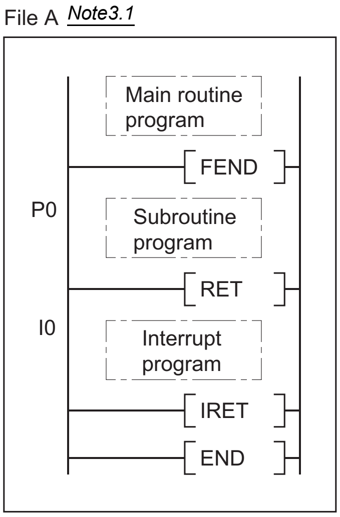
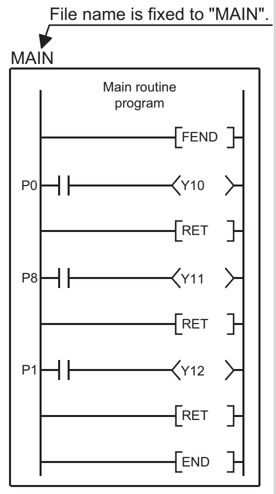


Figure 3.6 Sequence program classification

Basic Note3.1

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



3.1.1 Main routine programs

(1) Definition of main routine program

A main routine program is a program from Step 0 to the END/FEND instruction.

(2) Execution operation of main routine program

When the main routine program is executed, it operates as described below.

(a) When only one program is executed

The main routine program is executed from Step 0 to the END/FEND instruction, where END processing is performed.

After the END processing, the program restarts operation from Step 0.

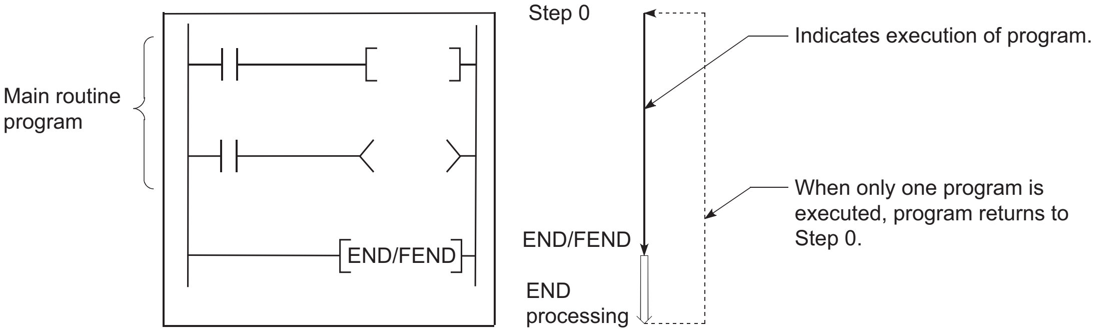

Main routine program
END/FEND
Step 0
Indicates execution of program.
When only one program is executed, program returns to Step 0.
END/FEND
END processing

Figure 3.7 Main routine program

(b) When multiple programs are executed

The main routine program operation after execution of the END/FEND instruction varies depending on the preset execution conditions.

Basic
Note3.2

Redundant Universal
Note3.3 Note3.3

(3) Execution types for main routine programs Note3.2

When multiple programs are to be executed, the following five different execution types can be set to main routine programs depending on the application.

• Initial execution type program	☞ Section 3.3.1
• Scan execution type program	☞ Section 3.3.2
• Low speed execution type program	☞ Section 3.3.3 Note3.3
• Stand-by type program	☞ Section 3.3.4
• Fixed scan execution type program	☞ Section 3.3.5

☒POINT

When no execution type is set for execution of only one program(☞ Section 3.3.6) , the main routine program operates as a scan execution type program.

Remark

Refer to the following manual for details of the END/FEND instruction.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic
Note3.2

The Basic model QCPU cannot execute multiple programs.
Therefore, it is not necessary to set the program execution type.

Redundant Universal
Note3.3 Note3.3

The low speed execution type program is not available for the Redundant CPU or Universal model QCPU.

3.1.2 Subroutine programs

(1) Definition of subroutine program

A subroutine program is a program section from a pointer (P□) to the REET instruction.

The subroutine program is executed only when it is called by a subroutine program call instruction (e.g. CALL(P), FCALL(P)) from the main routine program.

(2) Subroutine program applications

Using a subroutine program as described below reduces the number of program steps.

- Changing a section, which is executed several times during one scan, to a subroutine program reduces the number of steps in the whole program.
- Changing a section, which is executed only when a certain condition is satisfied, to a subroutine program reduces the number of steps in a normally executed program.

(3) Subroutine program management

Subroutine programs are created after a main routine program (after the FEND instruction).

Subroutine programs can also be managed as a single program.

(a) Creating subroutine programs after main routine program

1) Location of creating subroutine programs

Basic Note3.4

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

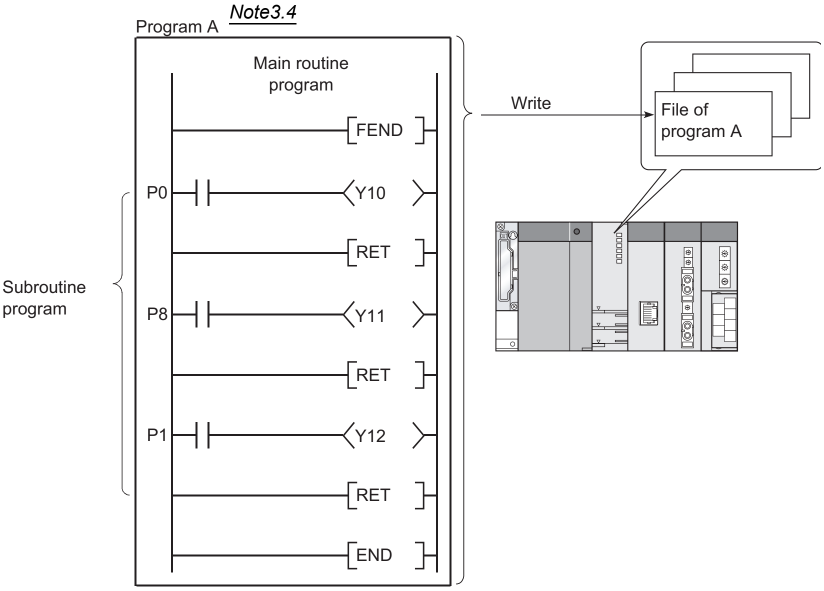
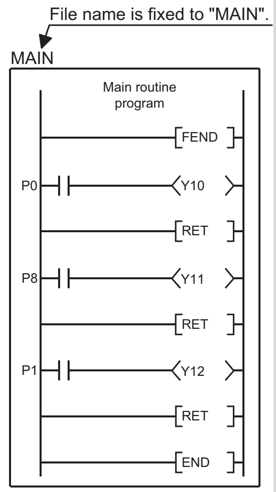


Figure 3.8 Subroutine programs

Basic Note3.4

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



2) **Restrictions on creation order**
When creating multiple subroutine programs, it is not necessary to set the pointer numbers in ascending order.

3) **Available pointers**

Basic
Note3.5

Local pointers and common pointers are available for subroutine programs.Note3.5
Note that when a local pointer is used, the subroutine program cannot be called from another program. (☞ Section 10.10)

Remark

Refer to Section 10.9 for the nesting of subroutine programs.

Basic
Note3.6

(b) **Managing subroutine program as another program** Note3.6
Subroutine programs can be managed as one program (Stand-by type program). (☞ Section 3.3.4)

Basic
Note3.5

Since the Basic model QCPU cannot execute multiple programs, it has no distinction between the local and common pointers.

Basic
Note3.6

Since the Basic model QCPU cannot execute multiple programs, subroutine programs cannot be managed as another program.

3.1.3 Interrupt programs

(1) Definition of interrupt program

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

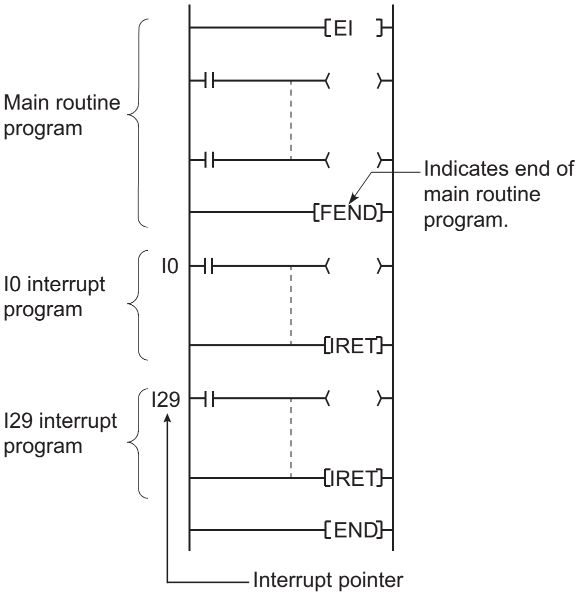


Figure 3.9 Interrupt programs

The interrupt factor varies depending on the interrupt pointer (I□) number. (☞ Section 10.11)

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

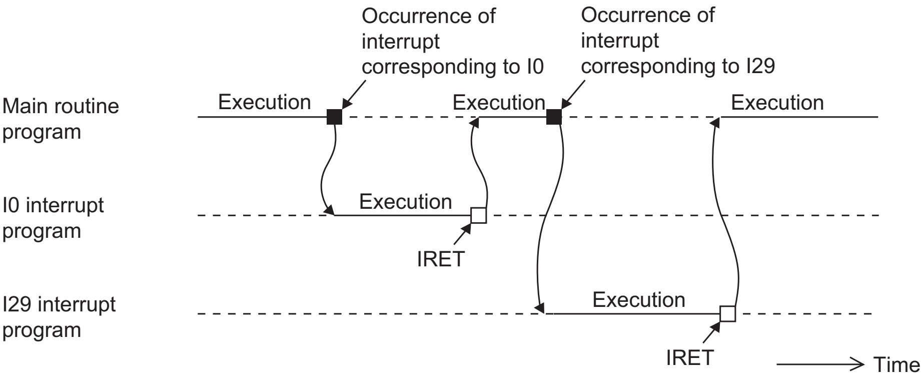


Figure 3.10 Interrupt program execution timing

Basic Note3.7 Process Note3.7 Redundant Note3.7

Universal UD Note3.7

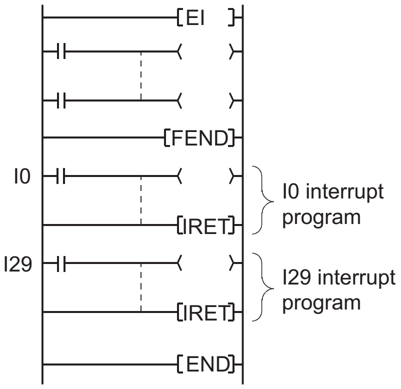
High Performance Note3.8

☒POINT

1. A pointer dedicated to the high-speed interrupt function (I49) is available as an interrupt pointer. Note3.7,Note3.8

   When using I49, do not execute the following:
   - Other interrupt pointers (Interrupt pointers other than I49)
   - Interrupt program
   - Fixed scan execution type program

   If any of the above is executed, the interrupt program of I49 cannot be executed in the preset interrupt cycles.
2. Only one interrupt program can be created with one interrupt pointer number.



Remark

Refer to Section 10.11 for details of the interrupt factors and interrupt pointers.

Basic Note3.7 Process Note3.7 Redundant Note3.7

Universal UD Note3.7

The pointer dedicated to the high-speed interrupt function (I49) is not available for the Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU.

High Performance Note3.8

In the case of the High Performance model QCPU, the pointer dedicated to the high-speed interrupt function (I49) is available for the QnHCPU only.
It is not available for the other CPU modules.

(2) IInterrupt program management

Interrupt programs are created after the main routine program (after the FEND instruction).

The interrupt programs can also be managed as a single program.

(a) Creating interrupt programs after main routine program

1) Location of creating interrupt programs

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

Basic
Note3.9

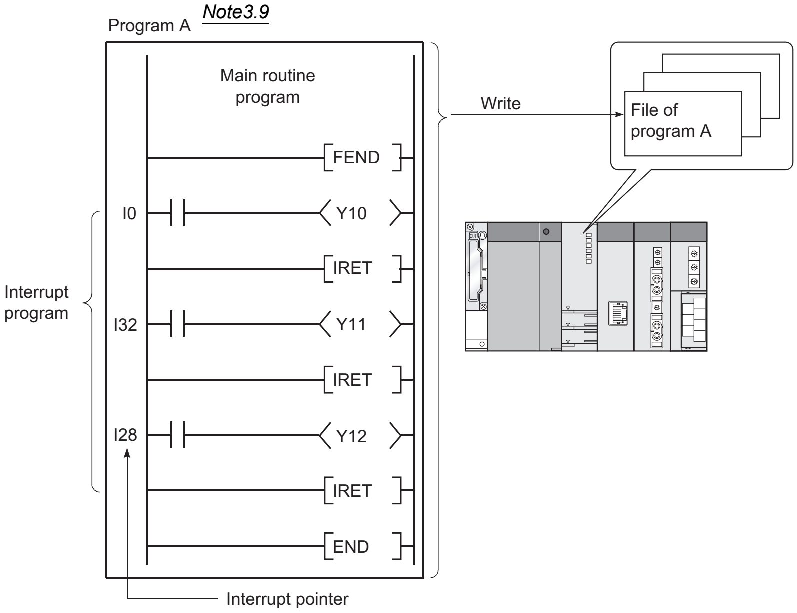


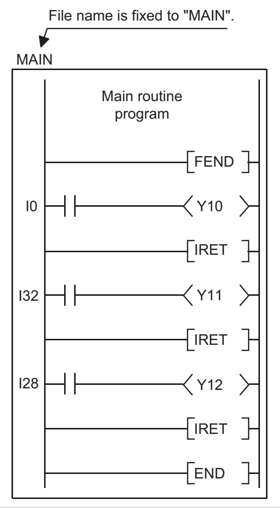
Figure 3.11 Interrupt programs

2) Restrictions on creation order

When creating multiple interrupt programs, it is not necessary to set the interrupt pointer numbers in ascending order.

Basic
Note3.9

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



Basic Note3.10

(b) **Managing interrupt programs as another program** Note3.10

Interrupt programs can be managed as one program (stand-by type program). (☞ Section 3.3.4)

(3) Before executing interrupt programs

Before executing interrupt programs, execute the following instructions to enable the interrupts.

(a) **Basic model QCPU**

Execute the EI instruction to enable the interrupts.

(b) **High Performance model QCPU, Process CPU or Redundant CPU**

When executing interrupt programs of interrupt pointers I32 to 47, execute the IMASK and EI instructions to enable the interrupts.

The interrupt programs of interrupt pointers I0 to 31 or I48 to 255 can be executed after enabling the interrupts with the EI instruction.

(c) **Universal model QCPU**

The interrupt programs of interrupt pointers I0 to 15, I28 to 31, I45, I50 to 255 can be executed after enabling the interrupts with the EI instruction.

Remark

Refer to the following manual for details of the IMASK and EI instructions.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Note3.10

Since the Basic model QCPU cannot execute multiple programs, interrupt programs cannot be managed as another program.

(4) When interrupt factor occurs

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

(a) When interrupt factor occurs before interrupts are enabled

The CPU module stores the interrupt factor that has occurred.
As soon as interrupts are enabled, the interrupt program corresponding to the stored interrupt factor is executed.

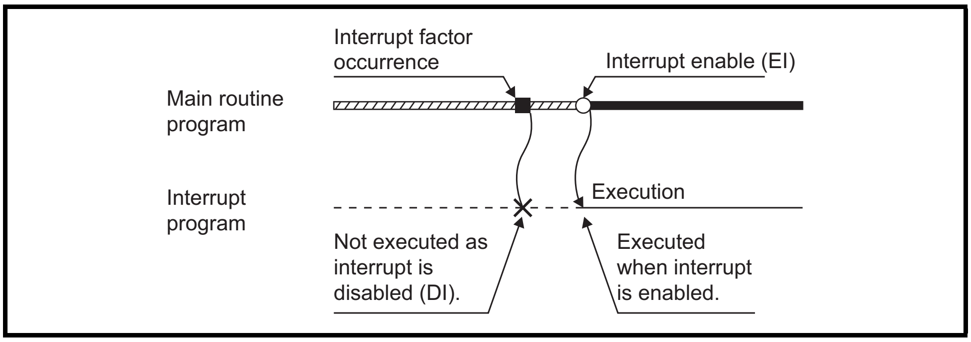

Interrupt factor occurrence
Interrupt enable (EI)
Main routine program
Interrupt program
Execution
Not executed as interrupt is disabled (DI).
Executed when interrupt is enabled.

Figure 3.12 When interrupt factor occurs before interrupts are enabled

When the same interrupt factor occurs several times before interrupts are enabled, the operation is performed as described below.

1) **Basic model QCPU**
   Interrupt factors of I0 to I15, I28 to I31 and I50 to I127 are stored only once.

2) **High Performance model QCPU, Process CPU or Redundant CPU**
   Interrupt factors of I0 to 27, I28 to 31, I50 to 255, and in the fixed scan execution type program are stored only once.
   For those of I32 to 41 and I49, if factors occurred in the interrupt-disabled status, they are discarded.
   Also, all factors occurred are discarded if they are masked by the IMASK instruction.

3) **Universal model QCPU**
   Interrupt factors of I0 to 15, I28 to 31, I45, I50 to 255, and in the fixed scan execution type program are stored only once.
   Note, however, that all factors occurred are discarded if they are masked by the IMASK instruction.

(b) When interrupt factor occurs in STOP/PAUSE status

When an interrupt factor occurs in the STOP/PAUSE status, the CPU module execute the interrupt program corresponding to the interrupt factor as soon as an interrupts are enabled after the CPU module status changes to RUN.

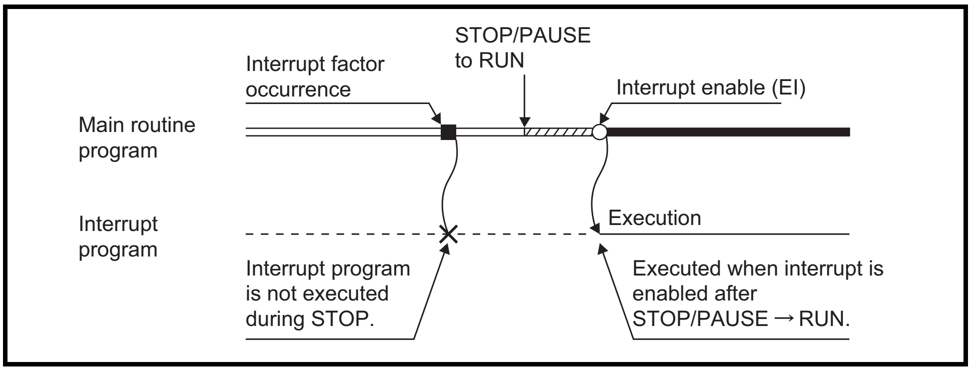

STOP/PAUSE to RUN
Interrupt factor occurrence
Interrupt enable (EI)
Main routine program
Interrupt program
Execution
Interrupt program is not executed during STOP.
Executed when interrupt is enabled after STOP/PAUSE → RUN.

Figure 3.13 When interrupt factor occurs in STOP/PAUSE status

(c) When multiple interrupt factors occur simultaneously in interrupt enable status

The interrupt programs are executed in the order of descending preferences of their interrupt pointers (I□) .(☞ Section 10.11)

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

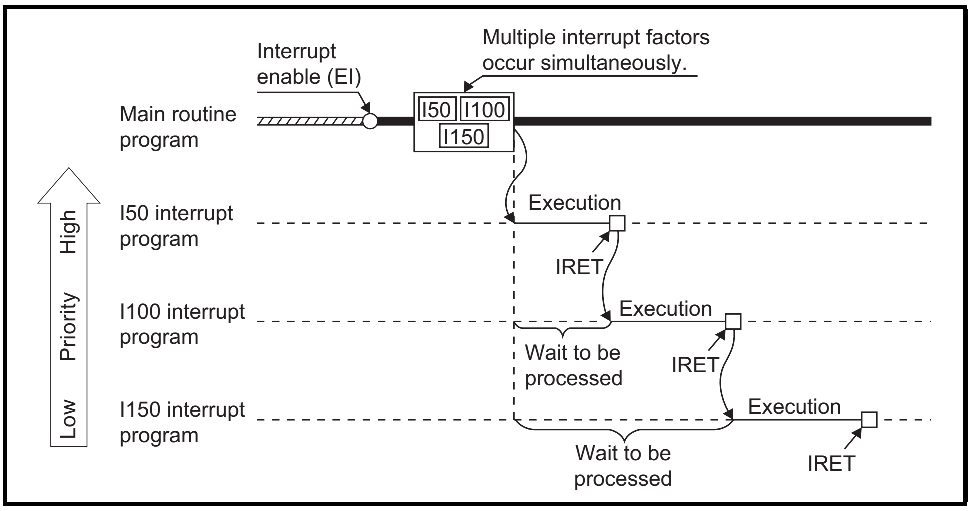


Figure 3.14 When multiple interrupt factors occur simultaneously in interrupt enable status

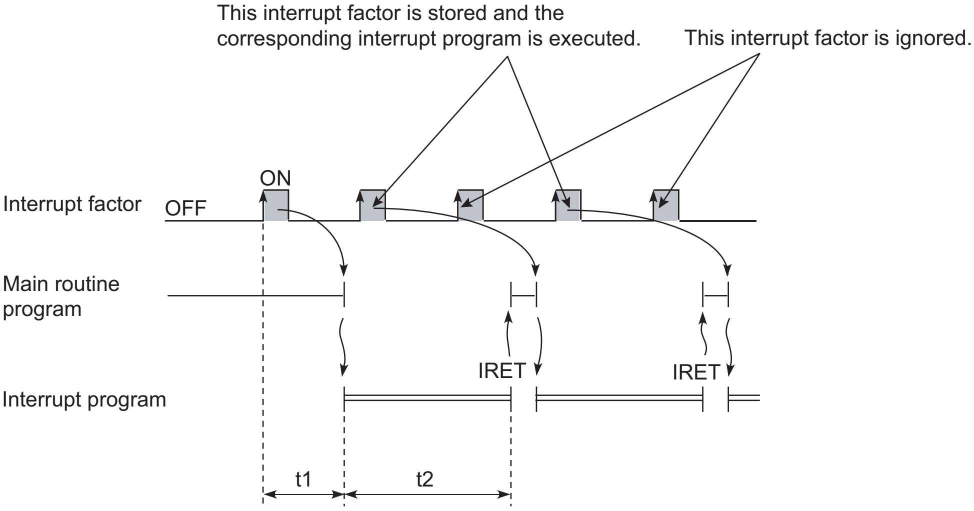
(d) When the same interrupt factor as that of the interrupt program in execution occurs

The operation will be as described below when the same interrupt factor as that of the interrupt program currently being executed occurs before completion of the program processing.

1) Basic model QCPU

Interrupt factors of I0 to I15, I28 to I31, and I50 to I127 are stored only once and the interrupt program for each stored interrupt factor is executed after completion of the current program execution.

Even if the same interrupt factor occurs several times, the first one only is stored and those occurred after that are ignored.


This interrupt factor is stored and the corresponding interrupt program is executed.
This interrupt factor is 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 3.15 When the same interrupt factor as that of the interrupt program in execution occurs

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

- Interrupt factors of I0 to 27 and I50 to 255 are stored only once, and then the interrupt program of each stored interrupt factor is executed after completion of the current interrupt program execution.
  For those of I28 to 31 and in the fixed scan execution type program, all interrupt factors occurred are stored, and then all the interrupt programs corresponding to those factors are executed after completion of the current interrupt program execution.
- Interrupt factors of I32 to I41 and I49 are discarded.

3) **Universal model QCPU**

Interrupt factors of I0 to 15, I45, and I50 to 255 are stored only once, and then the interrupt program of each stored interrupt factor is executed after completion of the current interrupt program execution.
For those of I28 to 31 and in the fixed scan execution type program, all interrupt factors occurred are stored, and then all the interrupt programs corresponding to those factors are executed after completion of the current interrupt program execution.

(e) **When instruction is being executed**

The interrupt program may be executed during execution of the main routine program instruction with the instruction execution being suspended.
When 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) **Transfer device data to another device**

Do not specify the device, which is to be written into the interrupt program, directly in the main routine program, but use it in another device by transferring it with the transfer instruction.

2) **Disable interrupt with DI instruction**
Disable the instructions that may cause inconvenience for the main routine program with the DI instruction.
During access to the device of the corresponding argument of the instruction, however, the interrupt program will not be inserted and therefore data inconsistency will not occur in argument units. Note3.11

Basic
Note3.11

(f) **When interrupt occurs during link refresh**
When an interrupt occurs during link refresh, link refresh is suspended and an interrupt program is executed.
Even if the cyclic data block has been assured for each station in the CC-Link IE controller network or the MELSECNET/H network system, it is not available when the device set as a refresh target is used in the interrupt program.
In the interrupt program, do not use the device set as the link refresh target.

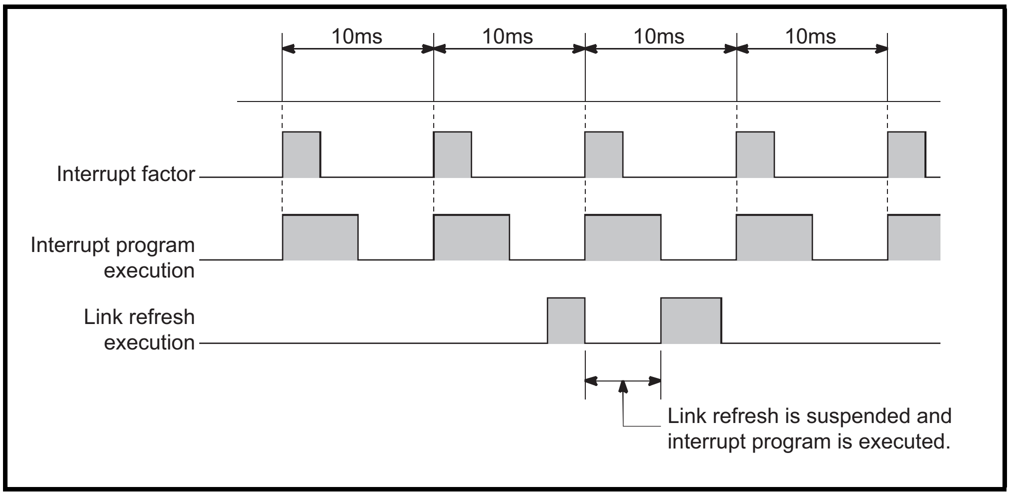

10ms
10ms
10ms
10ms
Interrupt factor
Interrupt program execution
Link refresh execution
Link refresh is suspended and interrupt program is executed.

Figure 3.16 When interrupt occurs during link refresh

Basic
Note3.11

When using the Basic model QCPU, data inconsistency may be occurred in argument unit.

Remark

Refer to the following manual for the block guarantee of cyclic data per station.

☞ CC-Link IE Controller Network Reference Manual

☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

(g) Interrupt during END processing

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

(5) Save and restoration of index register data and file register No.

Basic Redundant Universal

Note3.12 Note3.12 Note3.12

When an interrupt program is executed by default of the CPU module, the index register data and file register block No. are saved and restored at the time of switching between the scan execution type program/low speed execution type program Note3.12 and the interrupt program.(☞ Section 10.6.4)

(6) High speed execution setting and overhead time of interrupt program

When the "High speed execution" item for interrupt programs is checked on the PLC system tab of PLC parameter in GX Developer, the index registers are neither saved nor restored at the time of switching from the main routine program to the interrupt program.

This reduces the overhead time of the interrupt program.(☞ CHAPTER 11)

Basic Redundant Universal

Note3.12 Note3.12 Note3.12

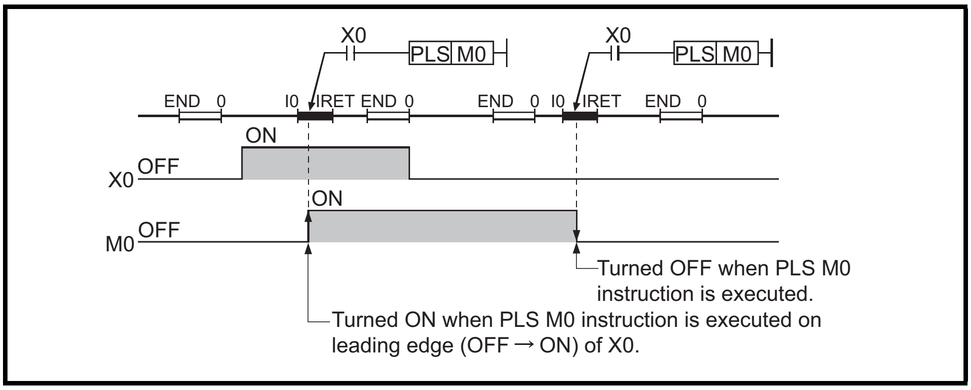
The Basic model QCPU, Redundant CPU and Universal model QCPU cannot be use low speed execution type program.

(7) Restrictions on programming

Restrictions on interrupt programming will be explained.

(a) Device turned ON/OFF by instruction such as PLS

When using an instruction such as PLS, by which an execution condition turns ON from OFF in the next step and it turns the operation device ON, the device remains ON until the same instruction is executed.
When using an instruction such as PLF, by which an execution condition turns OFF from ON in the next step and it turns the operation device ON, the device remains ON until the same instruction is executed.



Turned OFF when PLS M0 instruction is executed.

Turned ON when PLS M0 instruction is executed on leading edge (OFF → ON) of X0.

Figure 3.17 Device turned ON by PLS in interrupt program

(b) EI/DI instruction

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/DI instruction during interrupt program execution.

(c) Timers (T) and counters (C)

Timers and counters are not available for interrupt programs.
The timers (T) and counters (C) update the current values and turn ON/OFF the contacts when the OUT T□ and OUT C□ instructions are executed, respectively.
Using a timer (T) and/or a counter (C) in an interrupt program update the current value only during execution of the interrupt program.
Do not use them since measurement cannot be made correctly.

(d) Instructions not available for interrupt programs

For instructions not available in interrupt programs, refer to each instruction of programming manual.

(e) When interrupt/fixed scan execution type program is executed for execution time measurement, etc.

Basic
Note3.13

When an interrupt/fixed scan execution type program Note3.13 is executed to measure the scan time or execution time using special registers, the time for the program is added to the measured time.

The values stored into the special registers and the monitor values (measured times) of GX Developer shown below increase when the program is executed.

1) Basic model QCPU

- Special registers

SD520, SD521	Current scan time
SD524, SD525	Minimum scan time
SD526, SD527	Maximum scan time
SD540, SD541	END processing time
SD542, SD543	Constant scan wait time
SD548, SD549	Scan execution type program execution time

- Monitor values of GX Developer
  Scan time measurement
  Constant scan

Basic
Note3.13

The fixed scan execution type programs are not available for the Basic model QCPU.

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

- Special registers

  SD520, SD521 : Current scan time
  SD522, SD523 : Initial scan time
  SD524, SD525 : Minimum scan time
  SD526, SD527 : Maximum scan time
  SD528, SD529 : Current scan time for low speed execution type program[Note3.14]
  SD532, SD533 : Minimum scan time for low speed execution type program[Note3.14]
  SD534, SD535 : Maximum scan time for low speed execution type program[Note3.14]
  SD540, SD541 : END processing time
  SD542, SD543 : Constant scan wait time
  SD544, SD545 : Cumulative execution time for low speed execution type programs[Note3.14]
  SD546, SD547 : Execution time for low speed execution type programs
  SD548, SD549 : Scan execution type program execution time
  SD551, SD552 : Service interval time[Note3.15]

- Monitor values of GX Developer

  Execution time measurement
  Scan time measurement
  Constant scan

Redundant Universal UD
Note3.14

Universal UD
Note3.15

Redundant Universal UD
Note3.14

Since the Redundant CPU or Universal model QCPU cannot use low speed execution type programs, no special register for the low speed execution type programs is provided.

Universal UD
Note3.15

Since the Universal model QCPU cannot use service interval time read, no special register for service interval time read is provided.

3.2 Settings for Execution of Only One Sequence Program

A sequence program performs operation from Step 0 to the END/FEND instruction.
It performs an END processing when the END/FEND instruction is executed.
After the END processing, operation restarts from Step 0.
As described above, the sequence program repeats the operation from Step 0 to the END/FEND instruction.

☒**POINT**

This section explains the case where only one sequence program is created.
For creation of multiple sequence programs, the execution type can be specified for each program, e.g. a program started only once at startup or a program executed at fixed intervals.(☞ Section 3.3)

(1) Scan time

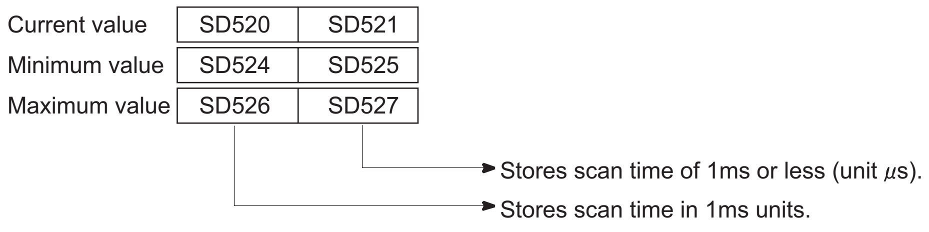
Scan time is a period from the time when the CPU module starts the sequence program operation from Step 0 until it executes Step 0 of the same sequence program again.
The scan time consists of the sequence program execution time and the END processing time.
When either of the following programs is executed, the execution time of that program is added to the scan time.

- Interrupt program
- Fixed scan execution type program<u>*Note3.16*</u>

Basic Note3.16

(a) Scan time storage location

The CPU module measures the current value and minimum and maximum values of the scan time and stores them into the special registers (SD520, SD521, SD524 to 527).
The scan time can be checked by monitoring SD520, SD521 and SD524 to 527.



Current value	SD520	SD521
Minimum value	SD524	SD525
Maximum value	SD526	SD527

Stores scan time of 1ms or less (unit μs).
Stores scan time in 1ms units.

Figure 3.18 Scan time storage location

When SD520 is 3 and SD521 is 400, the scan time is 3.4ms.

(b) Accuracy and measurement of scan time

The accuracy of each scan time stored into the special registers is ±0.1ms.
Even if the watchdog timer reset instruction (WDT) is executed in the sequence program, the measurement of each scan time is continued.

Basic Note3.16

The fixed scan execution type programs are not available for the Basic model QCPU.

(c) Scan time watch

The CPU module has scan time watch timers (watchdog timers).(☞ (2) in this section)

(2) WDT (Watchdog timer)

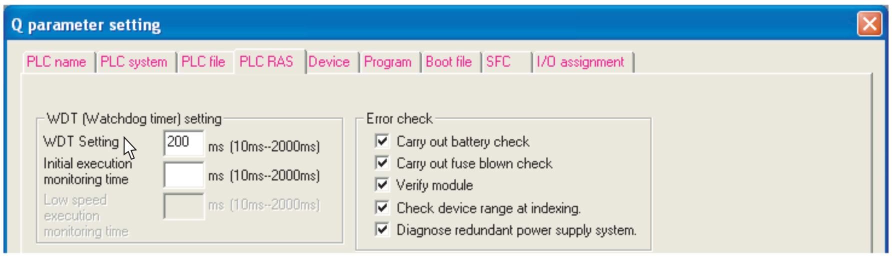
The watchdog timer (hereafter abbreviated to the WDT) watches the scan time.
The default value is 200ms.

(a) WDT error

A WDT error is 10ms.
When the WDT (t) is set to 10ms, a "WDT ERROR" occurs within a scan time range of 10ms<t<20ms.

(b) WDT Setting

The WDT setting can be changed within a range of 10ms to 2000ms in the PLC RAS of the PLC parameter dialog box. (Setting unit: 10ms)



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.

Figure 3.19 PLC RAS (WDT Setting)

Remark

The execution time of the program being executed can be checked in the program list monitor of GX Developer.(☞ Section 6.13.1)

(3) Function that repeats program at fixed intervals

The constant scan function(☞ Section 6.2)allows a program to be executed repeatedly at fixed intervals.
When the constant scan is set, a program is executed at intervals of the preset constant scan time.

3.3 Settings for Creation and Execution of Multiple Sequence Programs

Basic
Note3.17

When multiple sequence programs are created, the execution type can be specified for each program, e.g. a program started only once at startup or a program executed at fixed intervals.

(1) Applications for multiple sequence programs creation

A program can be divided into multiple programs on the basis of each control unit and stored into the CPU module. (They can also be stored as a single program.)
This enables programming to be shared by each designer for each processing unit.

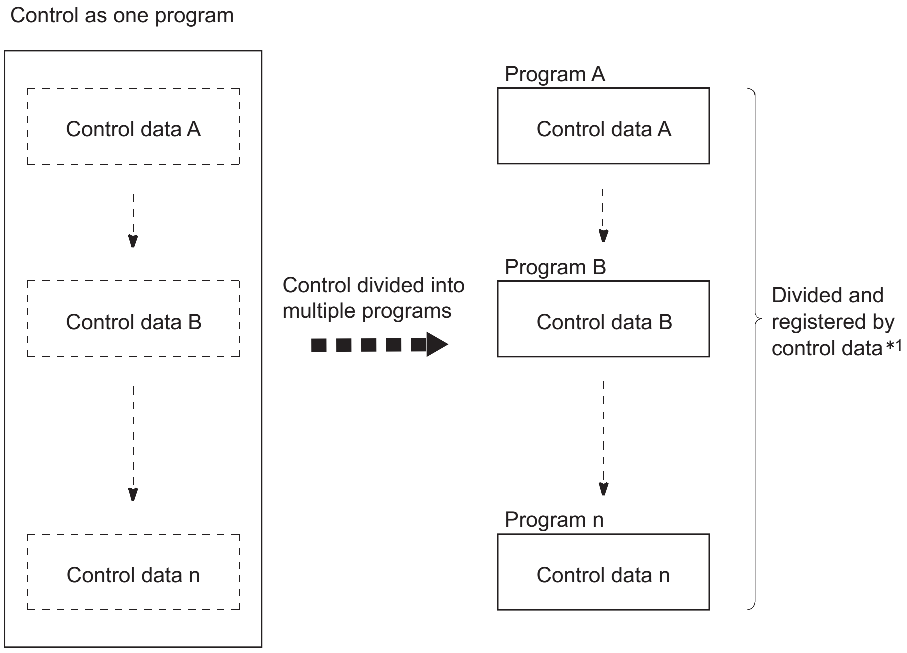


Figure 3.20 Control divided into multiple programs

* 1 : Refer to the following sections for the program storage location.

•Basic model QCPU : ☞ Section 5.1.1
•High Performance model QCPU, Process CPU, Redundant CPU, or Universal model QCPU : ☞ Section 5.2.1

(2) Settings required for execution of multiple programs

When multiple programs are to be executed by the CPU module, the file names (program names) and execution conditions of the programs must be preset.
Make the program settings in the PLC parameter dialog box of GX Developer.
(☞ Section 3.3.6)

Basic
Note3.17

Since the Basic model QCPU cannot execute multiple programs, the settings for creation and execution of multiple sequence programs are not available.

(3) Program storage location

The programs executed by the CPU module can be stored into the following memories.

- Program memory
- Standard ROM
- Memory card

(4) Available execution types

The following program execution types can be set on the CPU module.

- Initial execution type program ☞ Section 3.3.1
- Scan execution type program ☞ Section 3.3.2
- Low speed execution type program ☞ Section 3.3.3 Note3.18
- Stand-by type program ☞ Section 3.3.4
- Fixed scan execution type program ☞ Section 3.3.5

Redundant Universal UD
Note3.18 Note3.18

Redundant Universal UD
Note3.18 Note3.18

The low speed execution type programs are not available for the Redundant CPU or Universal model QCPU.

1 Overview
2 Performance Specification
3 Sequence Program Configuration and Execution Conditions
4 I/O Nunber Assignment
5 Memories and Files Handled by CPU Module
6 Functions
7 Communication using built-in Ethernet ports of CPU module
8 Communication with Intelligent Function Module

Basic Note3.19

3.3.1 Initial execution type program

(1) Definition of initial execution type program

An initial execution type program is executed only once when the PLC is powered ON or the STOP status is changed to the RUN status.

The initial execution type program can be used for a program that need not be executed from the next scan or later once it is executed, e.g. initial processing for the intelligent function module.

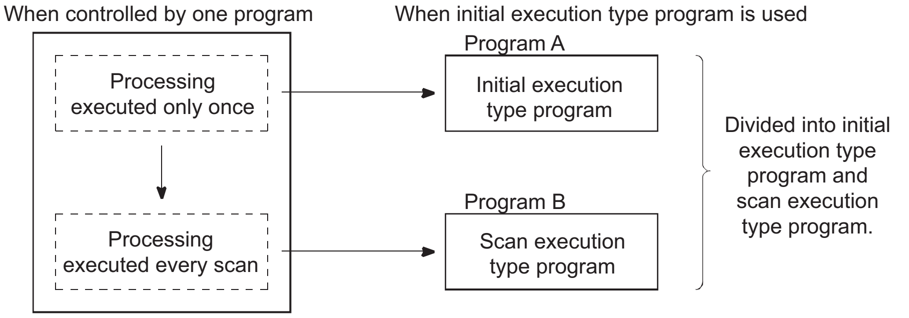

When controlled by one program
Processing executed only once
Processing executed every scan
When initial execution type program is used
Program A
Initial execution type program
Program B
Scan execution type program
Divided into initial execution type program and scan execution type program.

Figure 3.21 When processing to be executed only once is separated as initial execution type program

Basic Note3.19

The initial execution type programs are not available for the Basic model QCPU.

(2) Initial execution type program processing

(a) Execution sequence

When the execution of all the initial execution type program is completed, an END processing is performed and a scan execution type program is executed at the next scan.

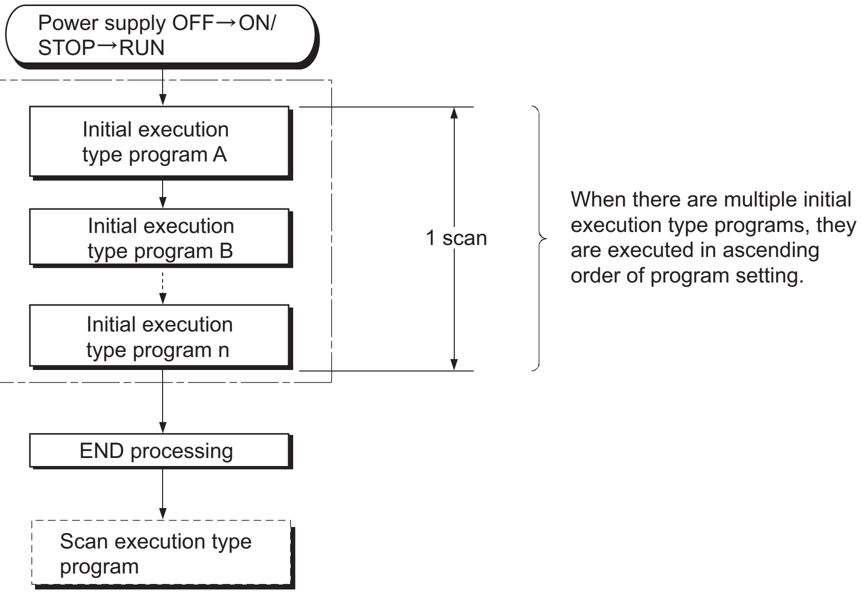

Power supply OFF→ON/ STOP→RUN
Initial execution type program A
Initial execution type program B
Initial execution type program n
1 scan
When there are multiple initial execution type programs, they are executed in ascending order of program setting.
END processing
Scan execution type program

Figure 3.22 Execution sequence of initial execution type program

(b) Initial scan time

The initial scan time is the execution time of an initial execution type program. When multiple initial execution type programs are executed, the initial scan time is the time taken until the execution of all initial execution type programs is completed.

1) Initial scan time storage location

The CPU module measures the initial scan time and stores it into the special registers (SD522, SD523).

The initial scan time can be checked by monitoring SD522 and SD523.

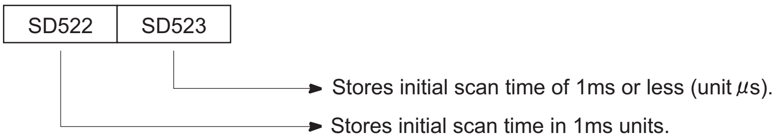

SD522
SD523
Stores initial scan time of 1ms or less (unit μs).
Stores initial scan time in 1ms units.

Figure 3.23 Initial scan time storage location

Example: When SD522 is 3 and SD523 is 400, the scan time is 3.4ms.

2) Accuracy and measurement of initial scan time

The accuracy of the initial scan time stored into the special registers is ± 0.1ms.

The measurement of the initial scan time is continued even if the watchdog timer reset instruction (WDT) is executed in a sequence program.

3) **When interrupt program/fixed scan execution type program is executed**
When an interrupt program/fixed scan execution type program is executed before completion of the initial execution type program execution, the interrupt program/fixed scan execution type program execution time is added to the initial execution type program execution time.

(c) **Initial execution monitoring time**
The initial execution monitoring time is a timer for monitoring the initial scan time.
The setting range is 10 to 2000ms (setting unit: 10ms).
The initial execution monitoring time is not set by default. (There is no default value.) Set the time on the PLC RAS setting tab of PLC parameter. (☞ (4)(b) in this section)

1) **Exceeding the initial execution monitoring time**
A "WDT ERROR" occurs when the initial scan time exceeds the preset initial execution monitoring time.
The CPU module stops the operation.

☒POINT

Redundant Universal
Note3.20 Note3.20

1. When an initial execution type program and low speed execution type program<u>Note3.20</u> are to be executed, the low speed execution type program(☞ Section 3.3.3)is executed after completion of the initial execution type program.
As the initial execution monitoring time, set a value greater than the total execution times of the initial execution type program and low speed execution type program.
2. When the initial execution monitoring time is set, the error of the measurement value is 10ms.
When the initial execution monitoring time (t) is set to 10ms, a "WDT ERROR" occurs within a scan time range of 10ms<t<20ms.

(3) Precautions for initial execution type program creation

In initial execution type programs, instructions that require several scans until execution completion (instructions where completion devices exist) cannot be used.
Example: SEND, RECV and similar instructions

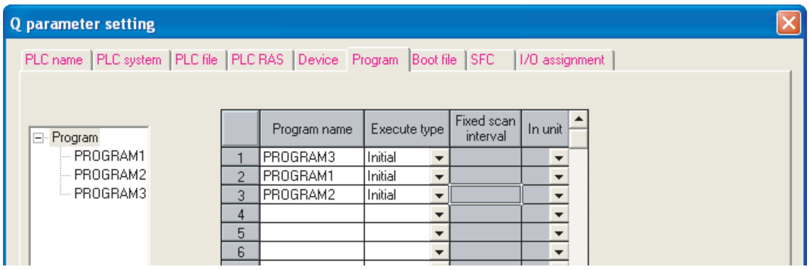
Redundant Universal
Note3.20 Note3.20

Since the Redundant CPU and Universal model QCPU cannot use low speed execution type programs, it is not necessary to take into account the low speed execution type program execution time when setting the initial execution monitoring time.

(4) Settings for initial execution type program execution

(a) Program

Set the execution type to "Initial" in the program of the PLC parameter dialog box. When using multiple initial execution type programs, register them in the order of execution.



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment

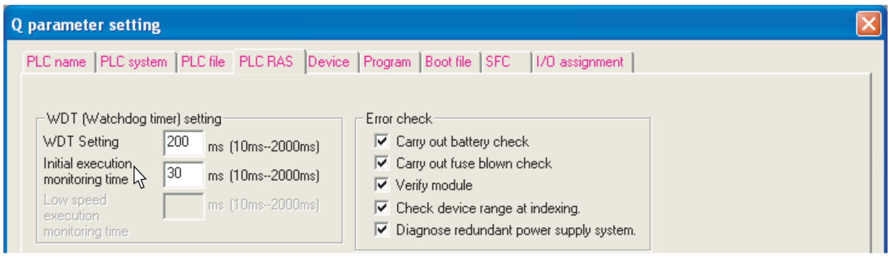
- Program
  - PROGRAM1
  - PROGRAM2
  - PROGRAM3

	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Initial		
2	PROGRAM1	Initial		
3	PROGRAM2	Initial		
4				
5				
6				

Figure 3.24 Program

(b) Initial execution monitoring time

When watching the initial execution type program execution time, set the initial execution monitoring time within a range of 10ms to 2000ms in the PLC RAS of the PLC parameter dialog box. (Setting unit: 10ms)



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment

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 3.25 PLC RAS (Initial execution monitoring time)

Basic
Note3.21

3.3.2 Scan execution type program

(1) Definition of scan execution type program

An scan execution type program is executed once for each scan, starting at the next scan after execution of the initial execution type program.

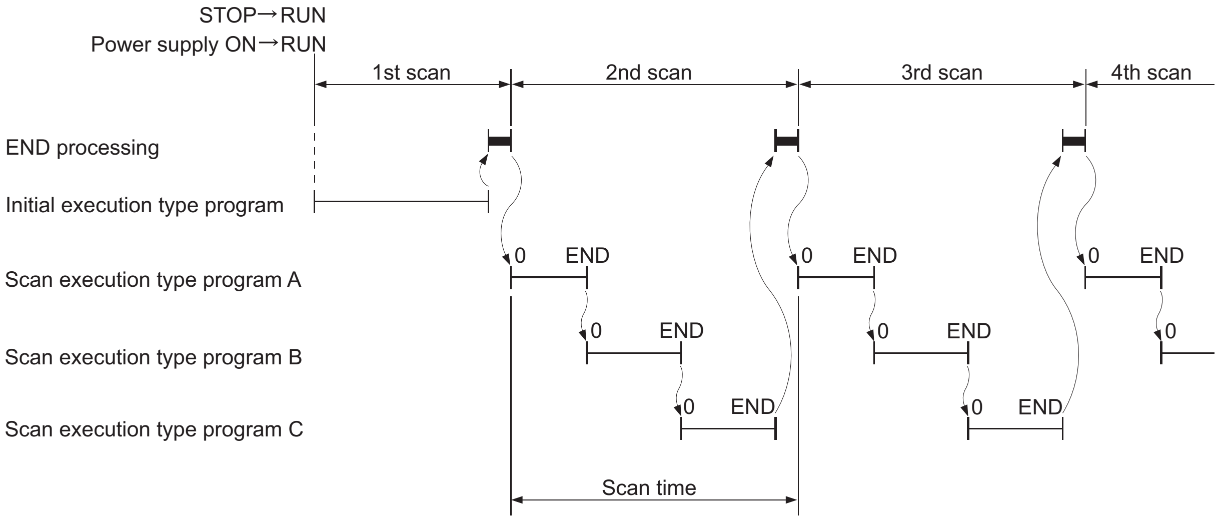

STOP→RUN
Power supply ON→RUN
1st scan
2nd scan
3rd scan
4th scan
END processing
Initial execution type program
Scan execution type program A
Scan execution type program B
Scan execution type program C
0
END
Scan time

Figure 3.26 Execution order of scan execution type programs

(2) Scan execution type program processing

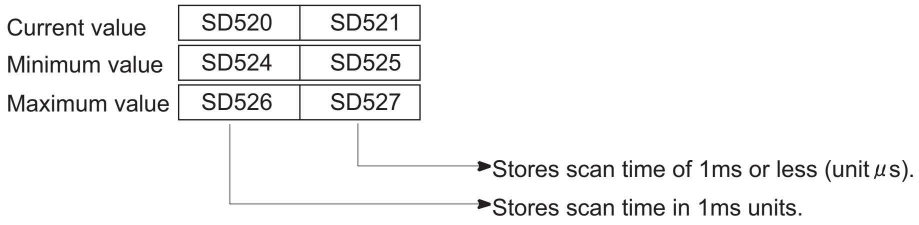
(a) Scan time

Scan time is the sum of the scan execution type program execution time and END processing time.

1) **Scan time storage places**

The CPU module measures the current value and minimum and maximum values of the scan time and stores them into the special registers (SD520, SD521, SD524 to 527).

The scan time can be checked by monitoring SD520, SD521 and SD524 to 527.



Current value	SD520	SD521
Minimum value	SD524	SD525
Maximum value	SD526	SD527

Stores scan time of 1ms or less (unit μs).

Stores scan time in 1ms units.

Figure 3.27 Scan time storage location

Example : When SD520 is 3 and SD521 is 400, the scan time is 3.4ms.

Basic
Note3.21

The Basic model QCPU cannot use multiple scan execution type programs.
Refer to Section 3.2 for details of program execution in the Basic model QCPU.

2) Accuracy and measurement of scan time

The accuracy of each scan time stored into the special registers is ± 0.1ms. Even if the watchdog timer reset instruction (WDT) is executed in a sequence program, the measurement of each scan time is continued.

3) Execution of multiple scan execution type programs

When multiple scan execution type programs are executed, the scan execution type program execution time is the time taken until the execution of all scan execution type programs is completed.

When an interrupt program/fixed scan execution type program is executed, the interrupt program/fixed scan execution type program execution time is added to the scan time.

(b) END processing

When all scan execution type programs are executed, an END processing is performed and the first scan execution type program is executed again.

By placing the COM instruction at the end of each scan execution type program, the END processing (network refresh) can be performed for each program.

(c) WDT (Watchdog timer)

The watchdog timer (hereafter abbreviated to the WDT) watches the scan time. The default value is 200ms.

1) WDT error

A WDT error is 10ms.

When the WDT (t) is set to 10ms, a "WDT ERROR" occurs within a scan time range of 10ms<t<20ms.

2) WDT setting

The WDT setting can be changed in the PLC RAS setting of the PLC parameter dialog box.

(☞ (3)(b) in this section)

Remark

The execution time of the program being executed can be checked in the program list monitor of GX Developer.(☞ Section 6.13.1)

(d) Scan execution type program can be repeated at fixed intervals

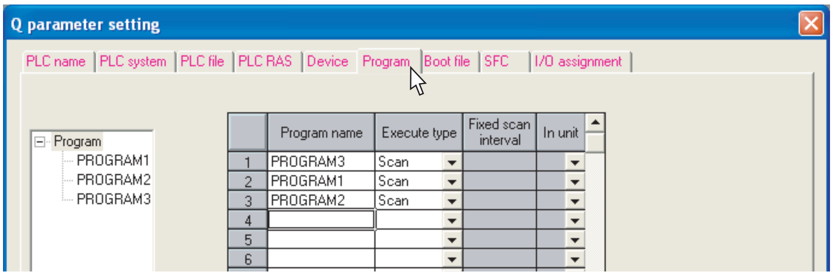
When the constant scan function (☞ Section 6.2) is used, a scan execution type program can be executed repeatedly at fixed intervals.

When the constant scan is set, a scan execution type program is executed at intervals of the preset constant scan time.

(3) Settings for execution of scan execution type programs

(a) Program

Set the execution type to "Scan" in the program of the PLC parameter dialog box. When using multiple scan execution type programs, register them in the order of execution.



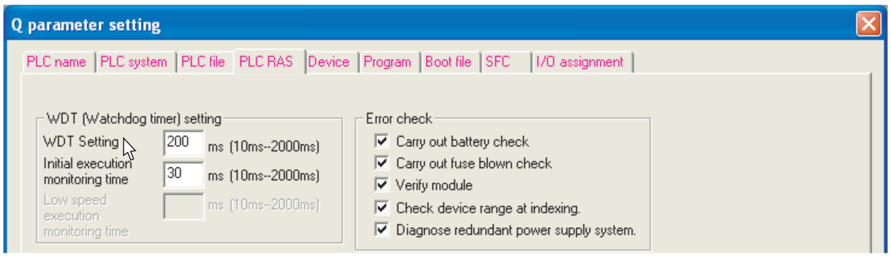
- Program
  - PROGRAM1
  - PROGRAM2
  - PROGRAM3

	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Scan		
2	PROGRAM1	Scan		
3	PROGRAM2	Scan		
4				
5				
6				

Figure 3.28 Program

(b) WDT Setting

When changing the WDT from the default value, set the WDT within a range of 10ms to 2000ms in the WDT setting of the PLC parameter dialog box. (Setting unit: 10ms)



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 3.29 PLC RAS (WDT Setting)

Basic Note3.22 Redundant Note3.22 Universal Note3.22

3.3.3 Low speed execution type program

(1) Definition of low speed execution type program

A low speed execution type program is executed only during the excess time of constant scan or the preset low speed program execution time.

The low speed execution type program can be used for the program that need not be executed every scan (e.g. output to a printer).

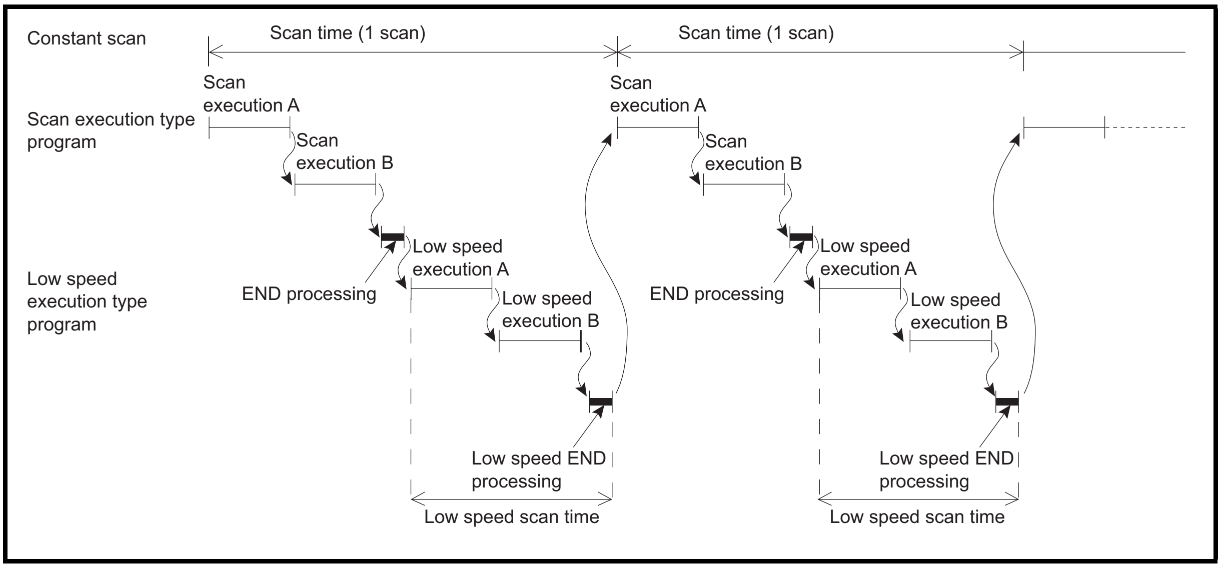


Figure 3.30 Low speed execution type program execution (When executed during excess constant scan time)

(2) Low speed execution type program processing

(a) Execution operation

The low speed execution type program execution varies depending on the following settings.

Use a desired setting as necessary.

1) **When giving priority to the control accuracy at fixed scan time**
Set the constant scan.

2) **When securing the execution time for the low speed execution type program**
Set the low speed program execution time.

Make the above settings in the PLC RAS setting of the PLC parameter dialog box.(☞ (4) in this section)

☒POINT

When executing a low speed execution type program, set either the constant scan or the low speed program execution time.

Basic Note3.22 Redundant Note3.22 Universal Note3.22

The low speed execution type program is not available for the Basic model QCPU , Redundant CPU and Universal model QCPU.

(b) **When there is excess time after completion of all low speed execution type program execution within one scan**

The processing performed after completion of low speed execution type program operation varies depending on the ON/OFF status of the special relay SM330 and low speed execution type program execution condition.

1) **Asynchronized tracking mode (SM330 = OFF)**

The low speed execution type program operation is continued within excess time.

2) **Synchronized tracking mode (SM330 = ON)**

If there is excess time, operation is started at the next scan without the low speed execution type program operation being continued.

Table3.1 Operation mode and setting

Low speed execution type program operation mode	Set status of SM330	Low speed execution type program execution condition	
		When constant scan is set	When low speed program execution time is set
Asynchronized tracking mode	OFF	Low speed execution type programs are re-executed*1	Low speed execution type programs are re-executed*2
Synchronized tracking mode	ON	Constant scan waiting time occurs*3	Scan execution type program operation starts*4

* 1 : Low speed execution type programs are executed repeatedly during the excess time of constant scan.
Hence, the low speed execution type program execution time differs between scans.
(☞ Figure 3.31)

* 2 : Low speed execution type programs are executed repeatedly during the preset low speed execution type program execution time.
Hence, the scan time differs between scans.(☞ Figure 3.33)

* 3 : The waiting time is the excess time after completion of low speed END processing.
When the preset constant scan time is reached, the scan execution type program is executed.
Hence, the scan time is invariable at each scan.(☞ Figure 3.32)

* 4 : The excess time after completion of low speed END processing is ignored, and the scan execution type program operation is started.
Hence, the scan time differs between scans.(☞ Figure 3.34)

1 Overview
2 Performance Specification
3 Sequence Program Configuration and Execution Conditions
4 I/O Nunber Assignment
5 Memories and Files Handled by CPU Module
6 Functions
7 Communication using built-in Ethernet ports of CPU module
8 Communication with Intelligent Function Module

When constant scan is set

The following timing charts show the operations performed when low speed execution type programs are executed under the conditions given below.

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

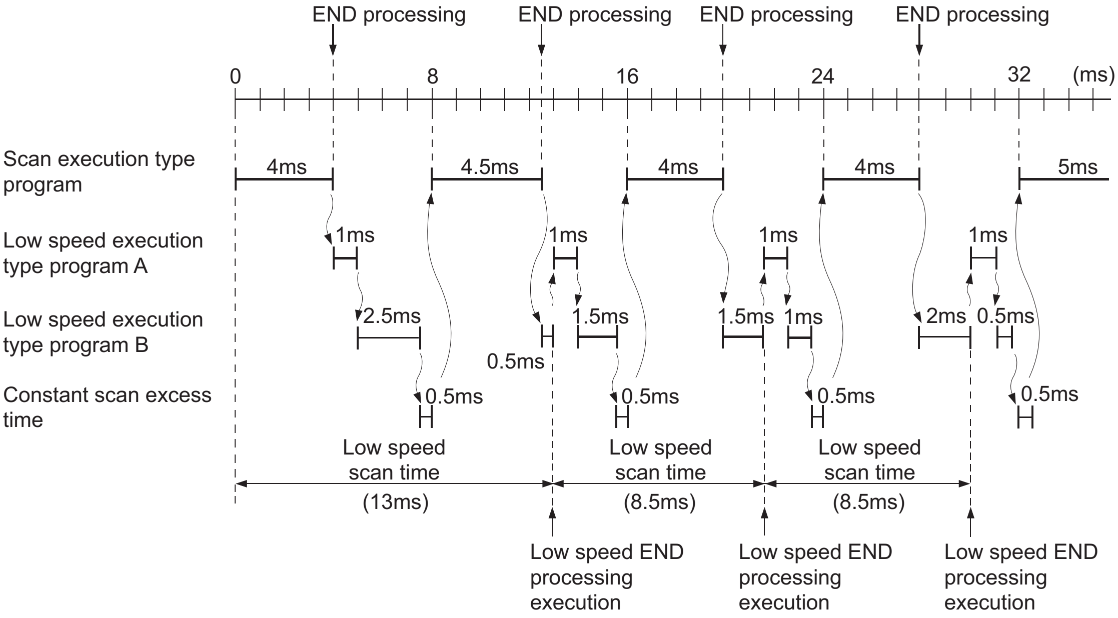


Figure 3.31 Asynchronized tracking mode (SM330: OFF)

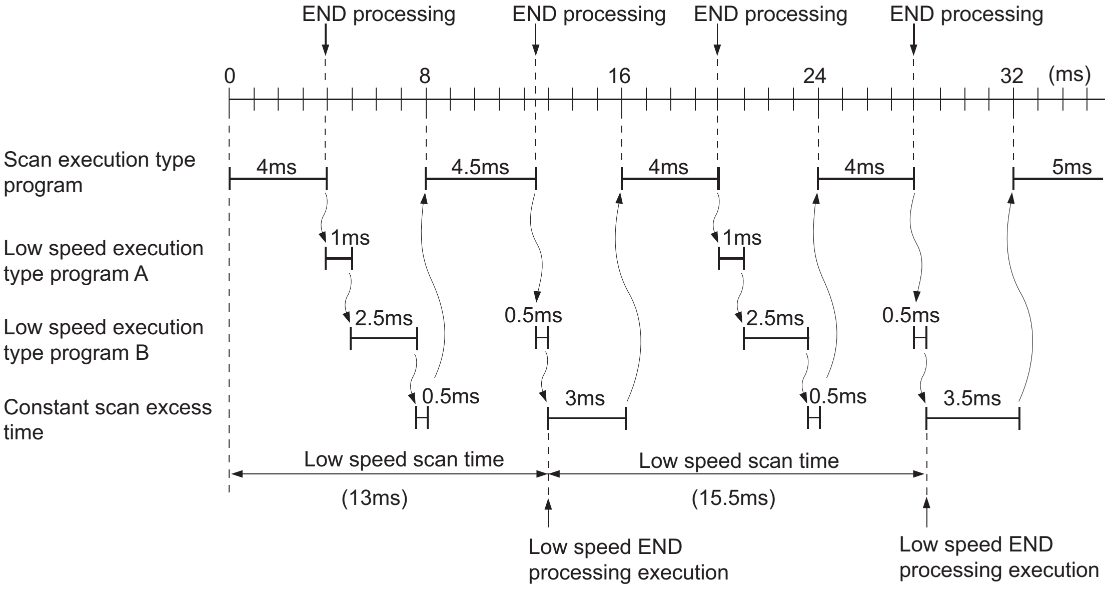

END processing
END processing
END processing
END processing
0
8
16
24
32
(ms)
Scan execution type program
Low speed execution type program A
Low speed execution type program B
Constant scan excess time
4ms
4.5ms
4ms
4ms
5ms
1ms
1ms
2.5ms
0.5ms
2.5ms
0.5ms
0.5ms
3ms
0.5ms
3.5ms
Low speed scan time
(13ms)
Low speed scan time
(15.5ms)
Low speed END processing execution
Low speed END processing execution

Figure 3.32 Synchronized tracking mode (SM330: ON)

When low speed program execution time is set

The following timing charts show the operations performed when low speed execution type programs are executed under the conditions given below.

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

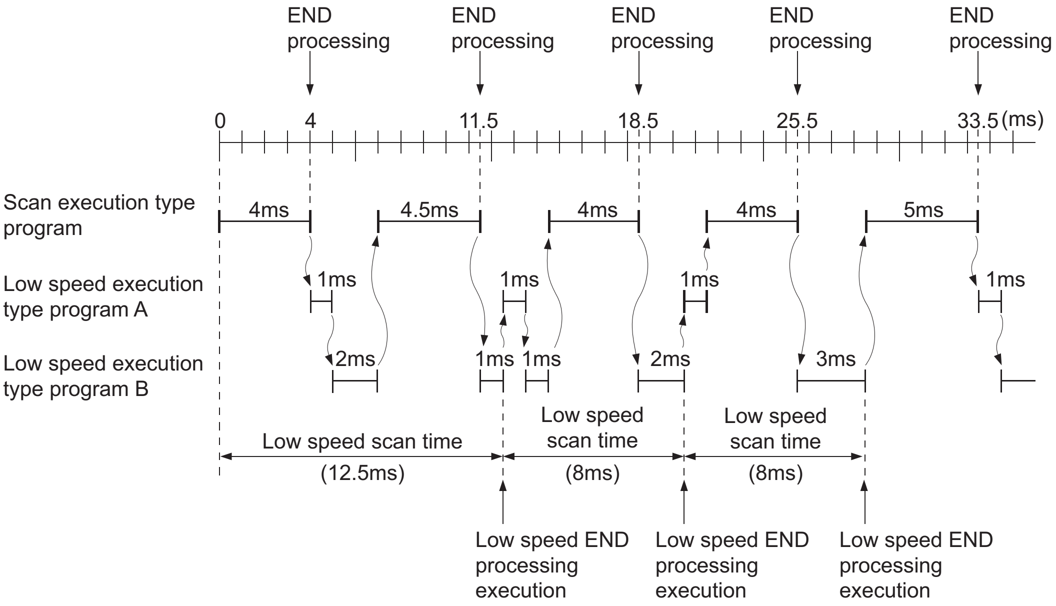


Figure 3.33 Asynchronized tracking mode (SM330: OFF)

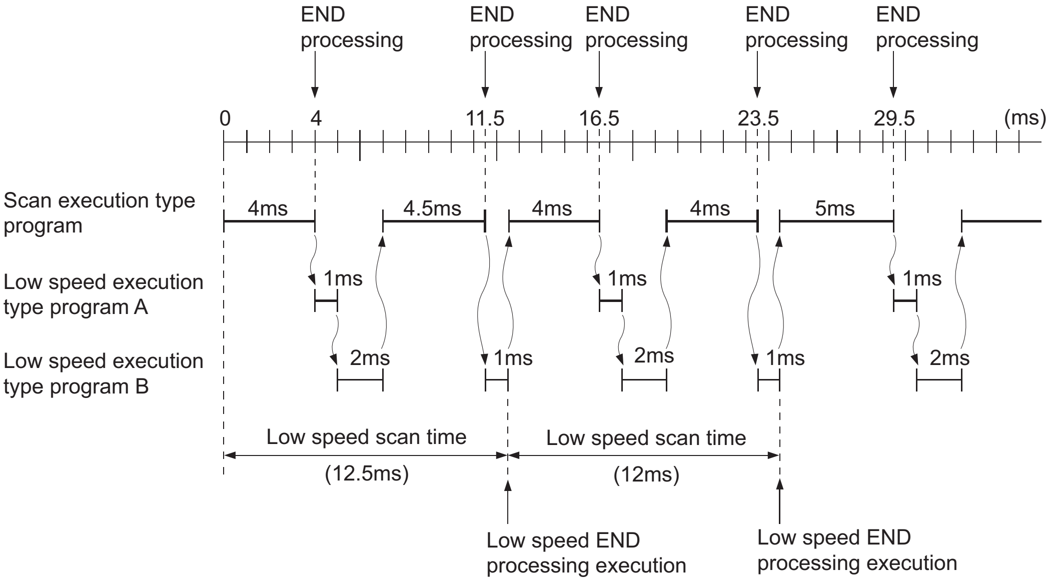


Figure 3.34 Synchronized tracking mode (SM330: ON)

(c) **When low speed execution type programs could not be processed within the excess time of constant scan or the low speed execution program execution time**

Program execution is suspended once and the remaining programs are executed at the next scan.

(d) **Low speed END processing**

When the low speed execution type programs are all executed, low speed END processing is performed.

Low speed END processing performs the following:

- Setting of special relays/special registers for low speed execution type programs[*1]
- Online change hereafter online change refers to the operation of writing a data in RUN status of low speed execution type programs
- Measurement of low speed scan time
- Reset of watchdog timers for low speed execution type programs

When low speed END processing is finished, the first low speed execution type program is executed again.

* 1 : The special relays/special registers for low speed execution type programs are given below.
SM330,SM404,SM405,SM510
SD430,SD510,SD528 to 535,SD544 to 547

☒POINT

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

Remark

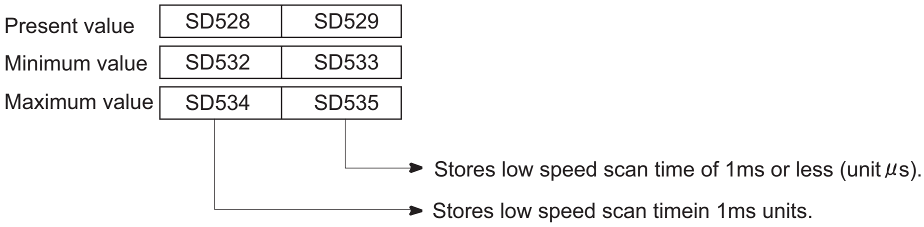
Refer to Figure 3.30 for differences between low speed END processing and END processing.

(e) **Low speed scan time**

Low speed scan time is the sum of the execution time of all low speed execution type programs and the low speed END processing time.
Refer to Figure 3.30 for differences between the low speed scan time and the scan time.

1) **Low speed scan time storage location**

The CPU module measures the low speed scan time and stores it into the special registers (SD528 to 535).
The low speed scan time can be checked by monitoring SD528 to 535.



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

Stores low speed scan time of 1ms or less (unit μs).
Stores low speed scan timein 1ms units.

Figure 3.35 Low speed scan time storage location

Example: When SD528 is 50 and SD529 is 400, the low speed scan time is 50.4ms.

Remark

The low speed scan time and program processing time can be checked on the screen displayed by selecting [Online] → [Monitor] → [Program monitor list] in GX Developer.(☞ Section 6.13.3)

2) **Accuracy and measurement of low speed scan time**

The accuracy of each scan time stored into the special register is ± 0.1ms.
Even if the watchdog timer reset instruction (WDT) is executed in a sequence program, the measurement of each scan time is continued.

3) **When interrupt program/fixed scan execution type program is executed**

The execution time of the interrupt program/fixed scan execution type program is added to the low speed scan time.

(f) **Low speed execution watch time**

The low speed execution watch time is measured by the timer that watches the low speed scan time.
The setting range is 10 to 2000ms (setting unit: 10ms).
No default time is preset to the low speed execution watch time (There is no default value.).

1) **When low speed execution watch time is exceeded**

A "PRG. TIME OVER (error code: 5010)" occurs when the low speed scan time exceeds the preset low speed execution watch time.

☒POINT

When a low speed execution type program and an initial execution type program are to be executed, the low speed execution type program is executed after completion of the initial execution type program(☞ Section 3.3.1).
In that case, set a value greater than the sum of the execution times of the initial execution type program and low speed execution type program as the initial execution monitoring time.

(3) Precautions for low speed execution type program creation

(a) Low speed program execution time setting
Setting the low speed program execution time increases the scan time since the set length of the time is secured.
Set the low speed program execution time or increase the WDT setting to make the scan time shorter than the WDT.

(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 during the scan where the initial execution type program has been executed.
To prevent low speed execution type programs from being executed after the initial execution type program execution, provide interlocks with the special relays (SM402, SM403).

(d) Setting range
Set only either the constant scan time or the low speed program execution time.
When both the constant scan time and the low speed program execution time are set, "PRG. TIME OVER (error code: 5010)" occurs under the condition of (excess time of constant scan) < (low speed program execution time).

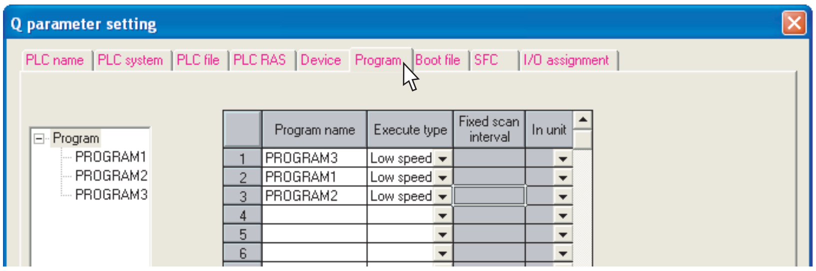
(e) Index register at program switching
Refer to Section 10.6.3 for the index register processing performed when a scan execution type program is switched to a low speed execution type program.
Refer to Section 10.6.4 for the index register processing performed when an interrupt program/fixed scan execution type program is executed during execution of a low speed execution type program.

(4) Settings for low speed execution type program execution

(a) Program

Set the execution type to "Low speed" in the program of the PLC parameter dialog box.

When using multiple low speed execution type programs, register them in the order of execution.



Program
- PROGRAM1
- PROGRAM2
- PROGRAM3

	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Low speed		
2	PROGRAM1	Low speed		
3	PROGRAM2	Low speed		
4				
5				
6				

Figure 3.36 Program

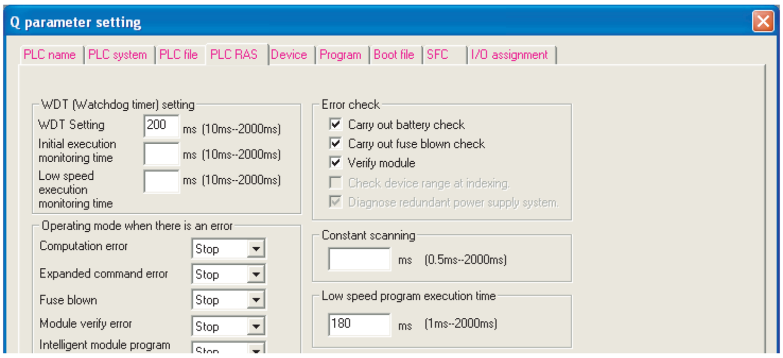
(b) Setting of constant scanning, low speed program execution time and low speed execution monitoring time

Set the time for constant scan and low speed program execution on the PLC RAS setting tab of PLC parameter.

The constant scanning setting range is 0.5ms to 2000ms.
(Setting unit: 0.5ms)

The low speed program execution time setting range is 1ms to 2000ms.
(Setting unit: 1ms)

When watching the low speed execution type program execution time, set a low speed execution monitoring time within a range of 10ms to 2000ms.
(Setting unit: 10ms)



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 3.37 PLC RAS (Constant scanning, low speed program execution time, low speed execution monitoring time)

☒POINT

When executing low speed execution type programs, be sure to set either the constant scanning or low speed program execution time.

Basic
Note3.23

3.3.4 Stand-by type program

(1) Definition of stand-by type program

A stand-by type program is executed only when its execution is requested.
It can also be changed to another execution type by a sequence program instruction.

(2) Applications of stand-by type program

The stand-by type program is used in the following applications.

(a) Library of programs

This application is used to set a subroutine program and an interrupt program as a stand-by type program and manage it separately from the main routine program.
Multiple subroutine programs and interrupt programs can also be created in a single stand-by type program.

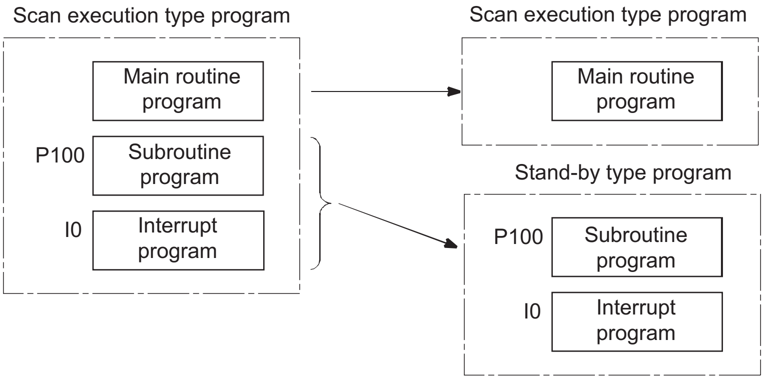


Figure 3.38 Library of programs using stand-by type program

(b) Program setup change

This is applicable when only a necessary program is to be executed among pre-created programs corresponding to all systems.
For example, the program preset as a stand-by (wait) in the PLC parameter dialog box can be changed to a scan execution type program and executed by a sequence program.

Basic
Note3.23

The stand-by type program is not available for the Basic model QCPU.

(3) Stand-by type program execution method

A stand-by type program can be executed 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.
  (☞ (3)(a) in this section)
- Change a stand-by type program to any other execution type using an instruction
  (☞ (3)(b) in this section)

(a) When placing subroutine or interrupt programs in a single stand-by type program

When creating subroutine or interrupt programs in a single stand-by type program, start it from Step 0.

The FEND instruction used for creation of subroutine or interrupt programs is not required after the main routine program.

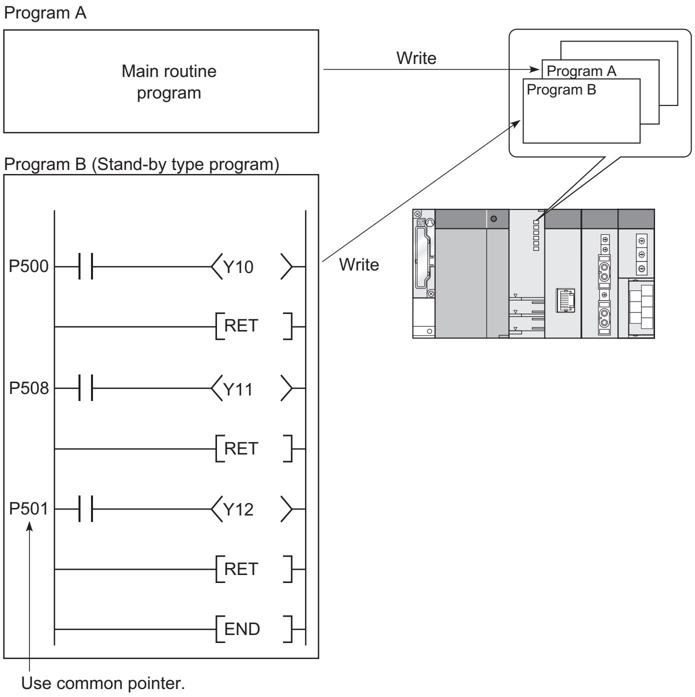


Figure 3.39 When placing subroutine programs in stand-by type program

1) Operation of subroutine and interrupt programs in stand-by type program

When the execution of the stand-by type program is finished, a program in the stand-by type program is called and its execution is resumed.

Figure 3.40 shows the operation performed when the subroutine and interrupt programs in the stand-by type program is executed.

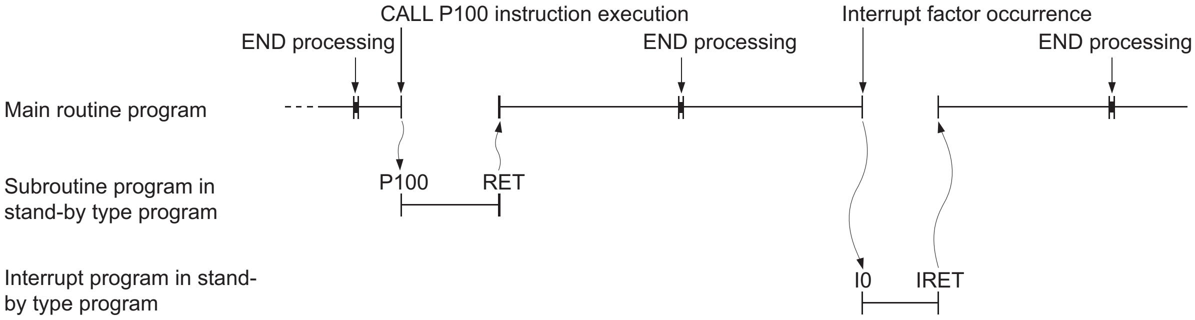

CALL P100 instruction execution
Interrupt factor occurrence
END processing
END processing
END processing
Main routine program
Subroutine program in stand-by type program
P100
RET
Interrupt program in stand-by type program
I0
IRET

Figure 3.40 Operation of subroutine and interrupt programs in stand-by type program

☒POINT

1. Refer to the following section for restrictions on subroutine or interrupt program creation.
   - Subroutine program : ☞ Section 3.1.2
   - Interrupt program : ☞ Section 3.1.3
2. Use common pointers.(☞ Section 10.10.2)
   When local pointers are used, the subroutine programs in the stand-by type program cannot be executed from another program.

Redundant Note3.24 Universal UD Note3.24

(b) When changing execution type by instruction

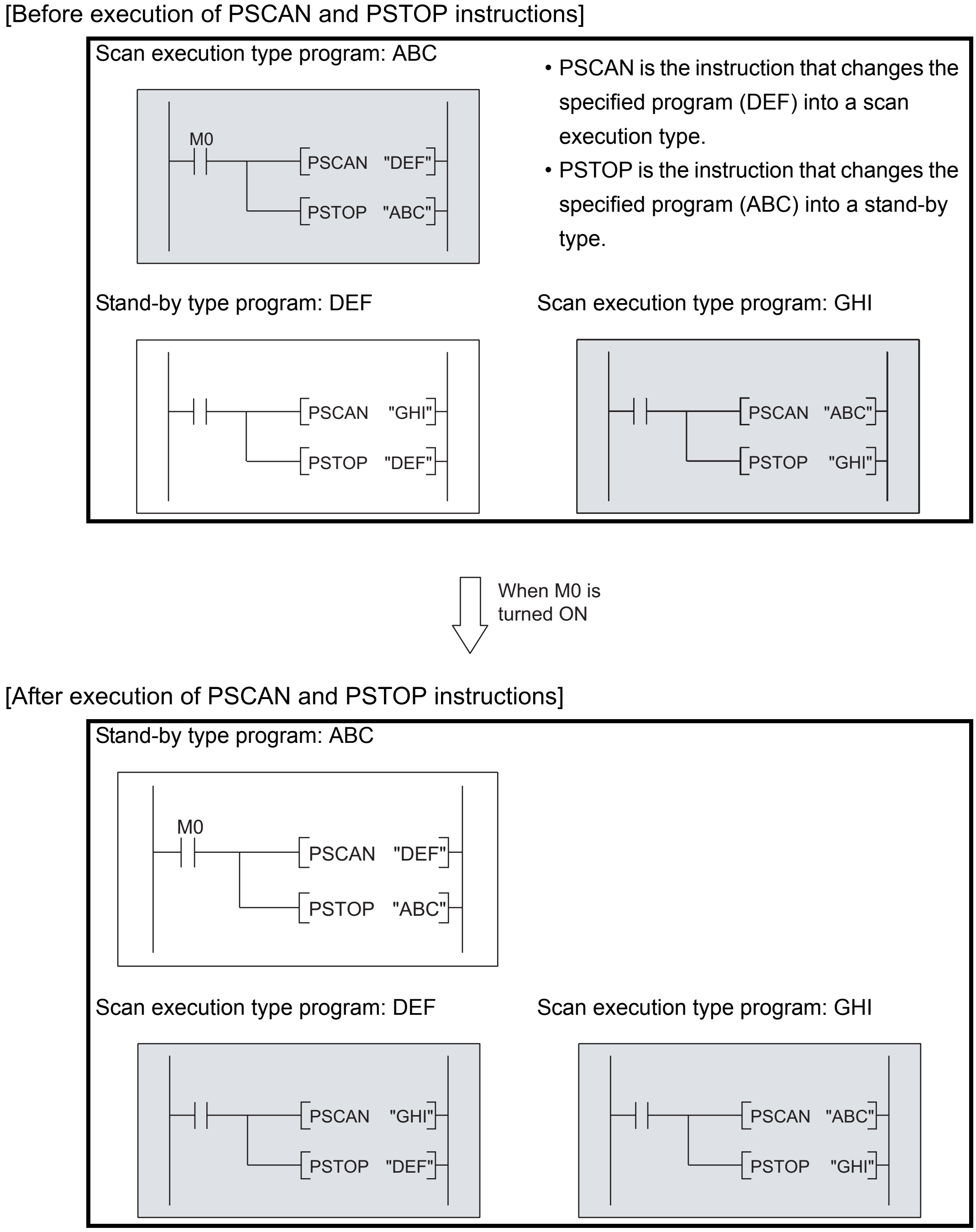
Use the PSCAN, PLOW Note3.24, PSTOP and/or POFF instruction to change the execution type.

Redundant Note3.24 Universal UD Note3.24

Since the Redundant CPU and Universal model QCPU cannot use low speed execution type programs, it cannot change the execution type using the PLOW instruction.

1) **Example of changing the execution type in scan execution type program**

- Set programs "ABC" and "GHI" as scan execution type programs.
  Set program "DEF" as a stand-by type program.
- When the condition is satisfied (the internal relay (M0) in Figure 3.41 turns ON), "DEF" is changed into a scan execution type program and "ABC" into a stand-by type program.

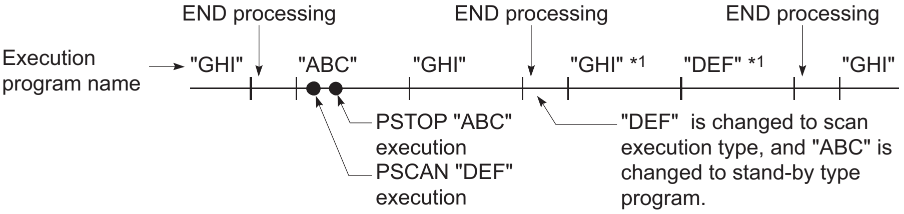

[Before execution of PSCAN and PSTOP instructions]
Scan execution type program: ABC
M0
PSCAN "DEF"
PSTOP "ABC"
Stand-by type program: DEF
PSCAN "GHI"
PSTOP "DEF"
Scan execution type program: GHI
PSCAN "ABC"
PSTOP "GHI"
When M0 is turned ON
[After execution of PSCAN and PSTOP instructions]
Stand-by type program: ABC
M0
PSCAN "DEF"
PSTOP "ABC"
Scan execution type program: DEF
PSCAN "GHI"
PSTOP "DEF"
Scan execution type program: GHI
PSCAN "ABC"
PSTOP "GHI"

- PSCAN is the instruction that changes the specified program (DEF) into a scan execution type.
- PSTOP is the instruction that changes the specified program (ABC) into a stand-by type.

Figure 3.41 Example of changing the execution type in scan execution type program

2) **Timing for execution type change**

The program execution type is changed in the END processing.
Hence, it is not changed midway through program execution.
When different types are set to the same program in the same scan, it is changed to the execution type executed by the latest instruction.



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

Figure 3.42 Timing for execution type change

(4) Precautions for stand-by type program creation

(a) Unavailable devices

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

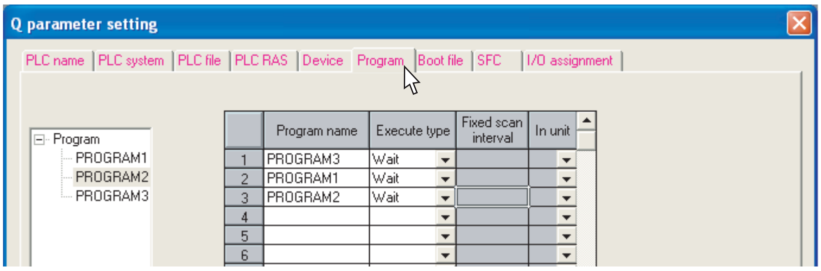
(b) Execution of subroutine program using local device

Refer to Section 10.14.1 for execution of the subroutine program using a local device.

(5) Settings for stand-by type program execution

(a) Program

When setting the created program as a stand-by type program in advance, set the execution type to "Wait" in the program setting of the PLC parameter dialog box.



	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Wait		
2	PROGRAM1	Wait		
3	PROGRAM2	Wait		
4				
5				
6				

Figure 3.43 Program

3.3.5 Fixed scan execution type program

Basic Note3.25

(1) Definition of fixed scan execution type program

This program is executed at the specified time intervals.
It can be executed at fixed cycle intervals for each file without description of interrupt pointers and IRET instructions.

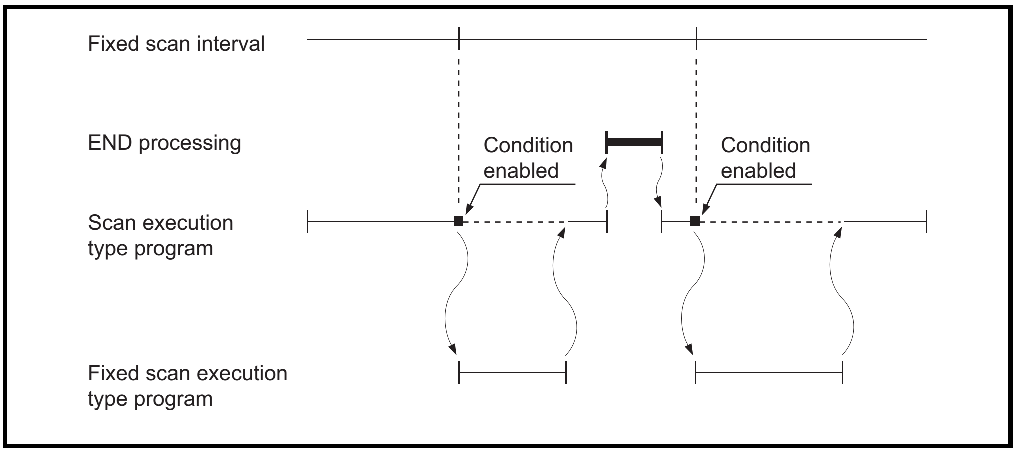


Figure 3.44 Execution of fixed scan execution type program

☒POINT

When executing the fixed scan execution type program, put the initial execution type program/scan execution type program into the interrupt allowable status by the EI instruction.

(2) Fixed scan execution type program processing

Fixed scan execution type program processing is explained below.

(a) When there are multiple fixed scan execution type programs

Fixed scan execution type programs are executed at the specified time intervals. When multiple fixed scan execution type programs reach the specified time at the same time, they are executed in ascending order of the numbers set in the program setting of the PLC parameter dialog box. (☞ (4) in this section)

(b) When there are fixed scan execution type program and interrupt program

When the fixed scan execution type program and an interrupt program (I28 to 31) reach the specified time at the same time, the interrupt program is executed first with priority.

Basic Note3.25

The fixed scan execution type program is not available for the Basic QCPU.

(c) Execution during network refresh

When the execution condition for the fixed scan execution type program is satisfied during network refresh, network refresh is suspended and the fixed scan execution type program is executed.

Even if the cyclic data block has been assured for each station in the CC-Link IE controller network or the MELSECNET/H network system, it is not available when the 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 the device set as a network refresh target.

Remark

Refer to the following manual for the block guarantee of cyclic data per station.

☞ CC-Link IE Controller Network Reference Manual

☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

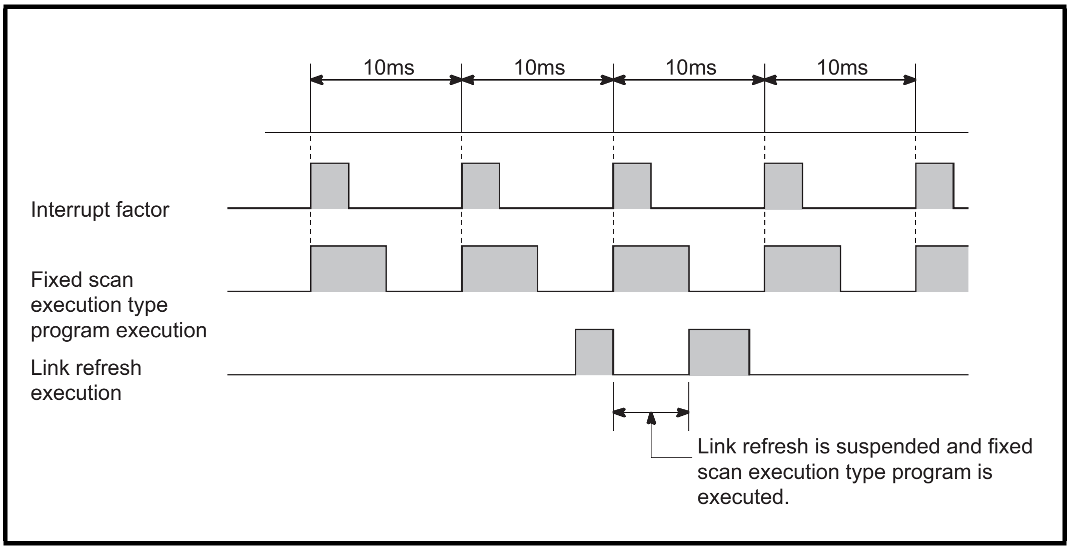

10ms
10ms
10ms
10ms
Interrupt factor
Fixed scan execution type program execution
Link refresh execution
Link refresh is suspended and fixed scan execution type program is executed.

Figure 3.45 Fixed scan execution type program execution during network refresh

(d) Execution during END processing

When the constant scan is set and the execution condition for the fixed scan execution type program is established during the wait time in the END processing, the fixed scan execution type program is executed.

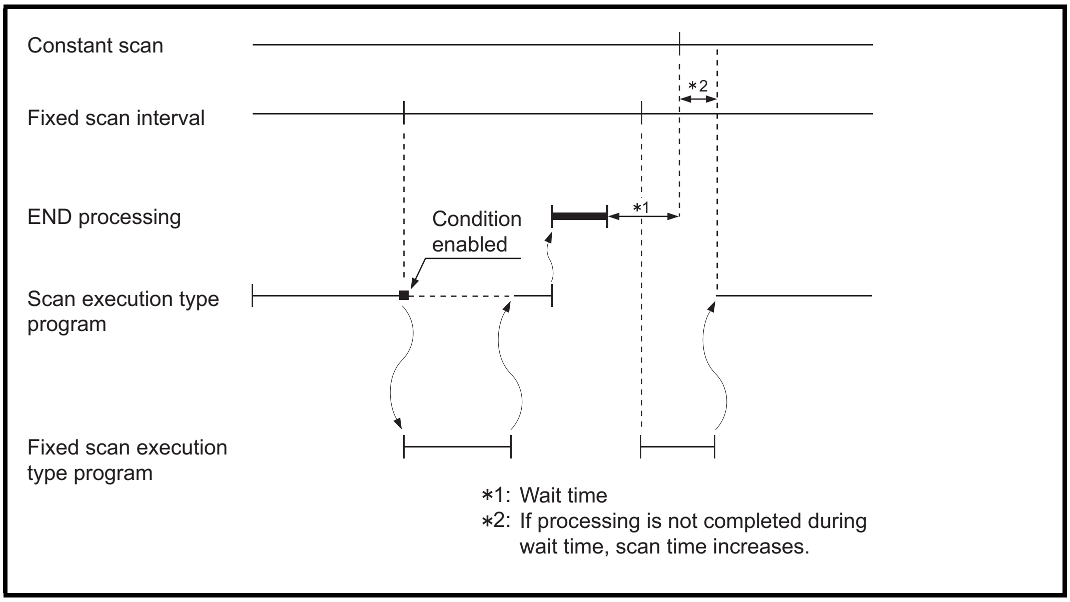

Constant scan
Fixed scan interval
END processing
Scan execution type program
Fixed scan execution type program
Condition enabled
*1
*2
*1: Wait time
*2: If processing is not completed during wait time, scan time increases.

Figure 3.46 Fixed scan execution type program execution during waiting time

Redundant Universal
Note3.26 Note3.26

(e) Index register processing

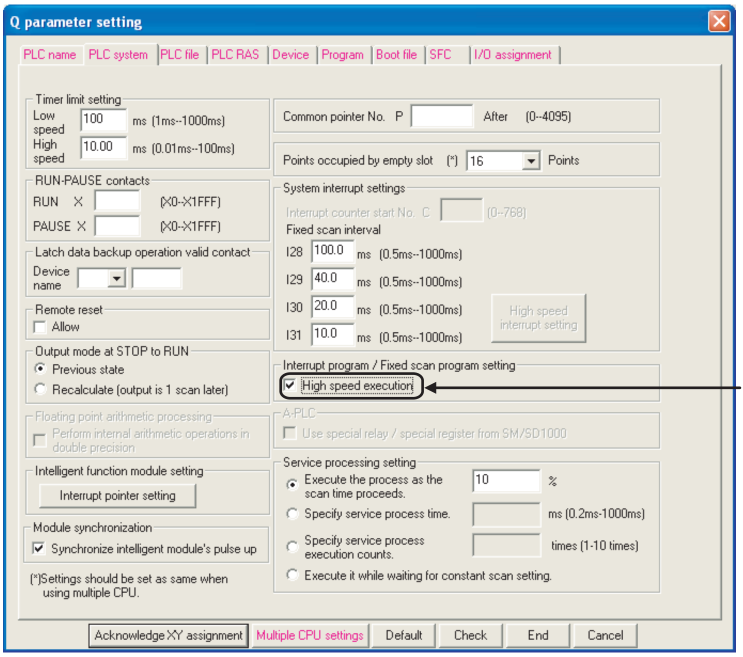
Refer to Section 10.6.4 for the index register processing when the program is switched from the scan execution type program/low speed execution type program Note3.26 to the fixed scan execution type program.

(f) High speed execution setting and overhead time of fixed scan execution type program

The following processing is performed when the fixed scan execution type program is executed.

- Save and restoration of index register data
- Save and restoration of file name of file register being used

The above processing is not executed when "High speed execution" of an interrupt program/fixed scan execution type program is selected in the PLC system of the PLC parameter dialog box.



Q parameter setting

Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.00 ms (0.01ms--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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)
High speed interrupt setting

Interrupt program / Fixed scan program setting
☑ High speed execution

Select.

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

Service processing setting
● Execute the process as the scan time proceeds. 10 %
○ 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.

Figure 3.47 PLC system screen

The overhead time of the fixed scan execution type program can be shortened.(☞ Section 11.1.2)

Redundant Universal
Note3.26 Note3.26

The Redundant CPU and Universal model QCPU do not support low speed execution type programs.

(3) Precautions for programming

(a) Device turned ON/OFF by PLS or similar instruction

When using an instruction such as PLS, by which an execution condition turns ON from OFF in the next step and it turns the operation device ON, the device remains ON until the same instruction is executed.

When using an instruction such as PLF, by which an execution condition turns OFF from ON in the next step and it turns the operation device ON, the device remains ON until the same instruction is executed.

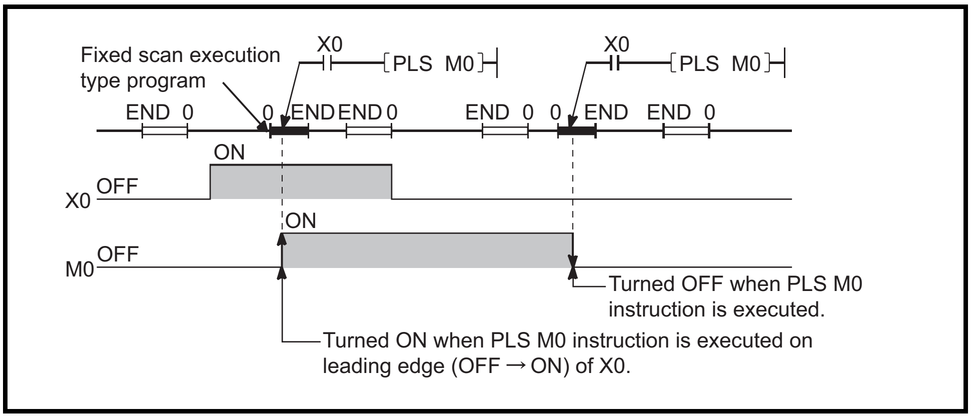

Fixed scan execution type program
X0
PLS M0
X0
PLS M0
END 0
0 END END 0
END 0 0 END END 0
ON
X0 OFF
ON
M0 OFF
Turned OFF when PLS M0 instruction is executed.
Turned ON when PLS M0 instruction is executed on leading edge (OFF → ON) of X0.

Figure 3.48 Device turned ON by PLS instruction in fixed scan execution type program

(b) EI/DI instruction

During execution of a fixed scan execution type program, interrupts are disabled (DI) so that any other interrupt is not enabled.

Do not execute the EI/DI instruction during a fixed scan execution type program execution.

(c) Timer (T) and counter(C)

Timer (T) and counter (C) cannot be used in fixed scan execution type programs.

Timer (T) cannot perform measurement normally when executing fixed scan execution type program multiple times during one scan.

Counter (C) in a fixed scan execution type program cannot perform measurement normally when executing fixed scan execution type program multiple times during one scan or due to the OUT C□ instruction status at execution.

(d) Instructions not available in fixed scan execution type programs

For instruction not available in fixed scan exection type programs, refer to each instruction of programming manual.

(e) **When interrupt/fixed scan execution type program is executed for execution time measurement, etc.**

When an interrupt/fixed scan execution type program is executed to measure scan time or execution time using special registers, the time of the above program is added to the measurement time. The following values stored into the special registers and the monitor values (measurement times) of GX Developer increase when the above program is executed.

- Special registers

SD520, SD521 : Current scan time
SD522, SD523 : Initial scan time
SD524, SD525 : Minimum scan time
SD526, SD527 : Maximum scan time
SD528, SD529 : Current scan time for low speed execution type program[Note3.27]

Redundant Note3.27 Universal UD Note3.27

SD532, SD533 : Minimum scan time for low speed execution type program[Note3.27]
SD534, SD535 : Maximum scan time for low speed execution type program[Note3.27]
SD540, SD541 : END processing time
SD542, SD543 : Constant scan wait time

Universal UD Note3.28

SD544, SD545 : Cumulative execution time for low speed execution type program[Note3.27]
SD546, SD547 : Execution time low speed execution type program[Note3.27]
SD548, SD549 : Scan execution type program execution time
SD551, SD552 : Service interval time[Note3.28]

- Monitor values of GX Developer

Execution time measurement
Scan time measurement
Constant scan

Redundant Note3.27 Universal UD Note3.27

Since the Redundant CPU and Universal model QCPU cannot use low speed execution type programs, no special register for low speed execution type programs is provided.

Universal UD Note3.28

Since the Universal model QCPU cannot use module service interval time read, no special register for module service interval time read is provided.

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

Note that the execution interval of a fixed scan execution type program may increase from the set interval depending on the time set for disabling interrupts by the DI instruction (interrupt disabled time).

When the interrupt disabled time by the DI instruction is set to longer, 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* < Interrupt disabled time ••• Expression 1)

* Means the highest common factor of fixed scan execution interval set to several fixed scan execution type programs.

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

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

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 expression 1) is 0.5ms < 5ms.

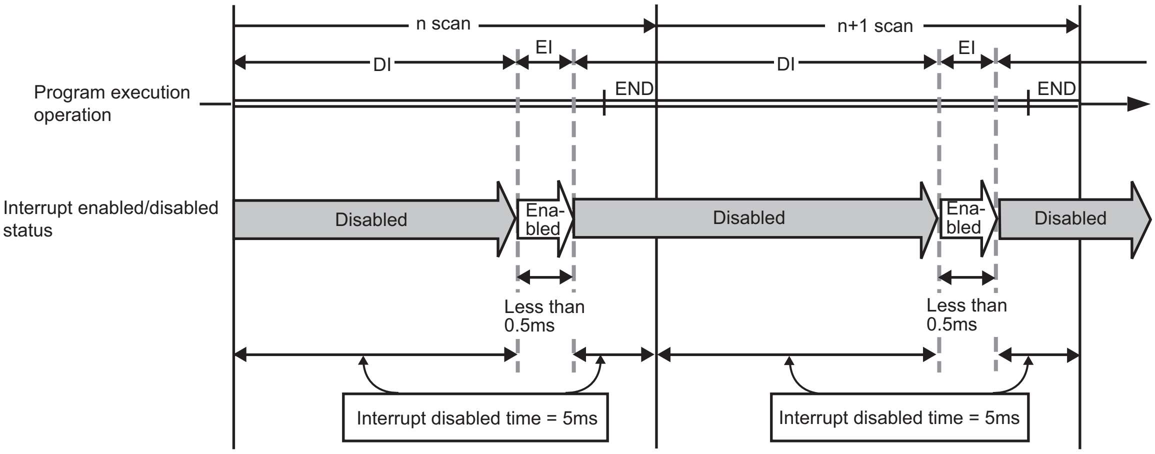

n scan
n+1 scan
DI
EI
DI
EI
END
END
Program execution operation
Interrupt enabled/disabled status
Disabled
Ena-bled
Disabled
Ena-bled
Disabled
Less than 0.5ms
Less than 0.5ms
Interrupt disabled time = 5ms
Interrupt disabled time = 5ms

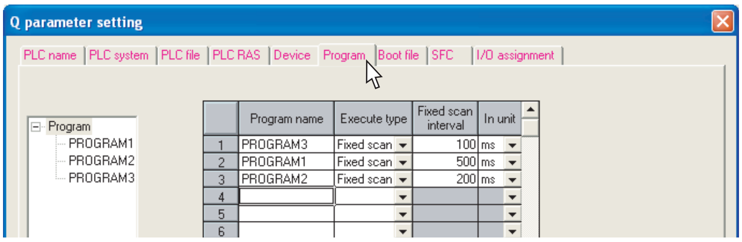
Figure 3.49 Program execution operation and interrupt enabled status

Therefore, 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.

(4) Settings for execution of fixed scan execution type program

(a) Program

Set the execution type to "Fixed scan" in the program setting of the PLC parameter dialog box. When using multiple fixed scan execution type programs, register them in the order of execution.



	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM3	Fixed scan	100	ms
2	PROGRAM1	Fixed scan	500	ms
3	PROGRAM2	Fixed scan	200	ms
4				
5				
6				

Figure 3.50 Program

After setting the execution type to "Fixed scan", set the fixed scan interval and unit.

They can be set within the following ranges.

- When the unit is ms : 0.5 to 999.5ms(0.5ms units)
- When the unit is s : 1 to 60s (1s units)

Basic

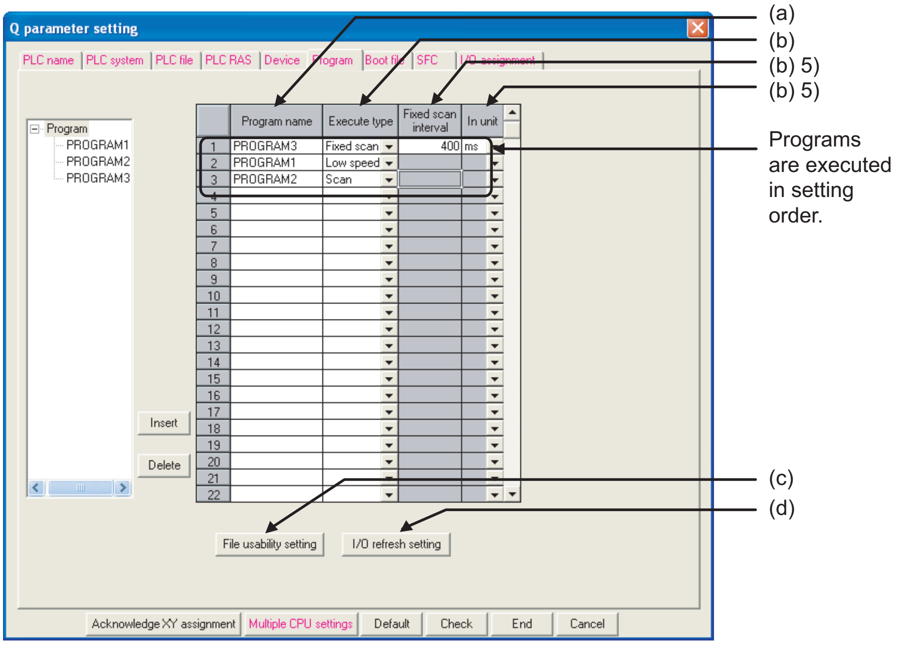
Note3.29

3.3.6 Execution type setting and example of type changing

(1) Execution type setting

Program setting necessary for executing multiple programs is explained in this section. Set the program execution type in the program of the PLC parameter dialog box of GX Developer.

The CPU module executes the programs of set execution types in the setting order.



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

Programs are executed in setting order.

Figure 3.51 Program

(a) Program name

Set the program name (file name) of the program to be executed by the CPU module.

(b) Execution type

Select the execution type of the file set in Program name.

1) Initial execution type (Initial)

This program is executed only once when the PLC is powered ON or is switched from STOP to RUN.(☞ Section 3.3.1)

2) Scan execution type (Scan)

This program is executed once for each scan, starting at the scan after execution of the initial execution type program.(☞ Section 3.3.2)

Redundant Universal UD

Note3.30

3) Low speed execution type (Low speed) Note3.30

This program is executed only when the constant scan or low speed type program execution time is set.(☞ Section 3.3.3)

Basic

Note3.29

Since the Basic model QCPU cannot execute multiple programs, the execution type setting and instruction-triggered changing are not available.

Redundant Universal UD

Note3.30

Since the Redundant CPU and Universal model QCPU cannot use low speed execution type programs, "Low speed" cannot be selected as the execution type.

4) **Stand-by type (Wait)**

This program is executed only when an execution is requested. (☞ Section 3.3.4)

5) **Fixed scan execution type (Fixed scan)**

This program is executed at the time intervals set to "Fixed scan interval" and "Unit".(☞ Section 3.3.5)

- Fixed scan interval

  Set the execution interval of a fixed scan execution type program.

  The setting range of the fixed scan interval changes depending on the set unit.

  •When the unit is ms : 0.5 to 999.5ms (0.5ms units)

  •When the unit is s : 1 to 60s (1s units)

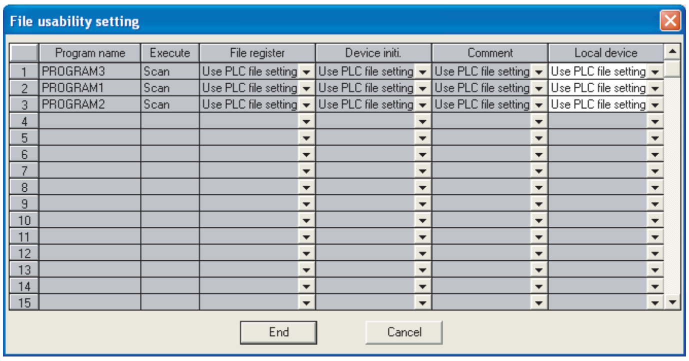
- Unit

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

Universal UD Note3.31

(c) **File usability setting**Note3.31

Make setting to each program whether to use the file specified for the file register, initial device value, device comment, and local device on the PLC file setting tab of PLC parameter.



File usability setting

	Program name	Execute	File register	Device initi.	Comment	Local device
1	PROGRAM3	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
2	PROGRAM1	Scan	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 3.52 File usability setting

The default is set to "Use PLC file setting (Use PLC file setting unchanged)". When "Not used" is selected, the processing of the file usability setting will be as described in Table3.2.

Table3.2 Processing when "Not used" is set in the file usability setting

Setting item	When "Not use" is selected
File registers	File registers cannot be used in the program.
Device initial values	The device initial values are not set when they are the same as those of the program.
Comments	The device comments cannot be used in the program.
Local devices	The local device data are not saved and restored at program switching.

Universal UD Note3.31

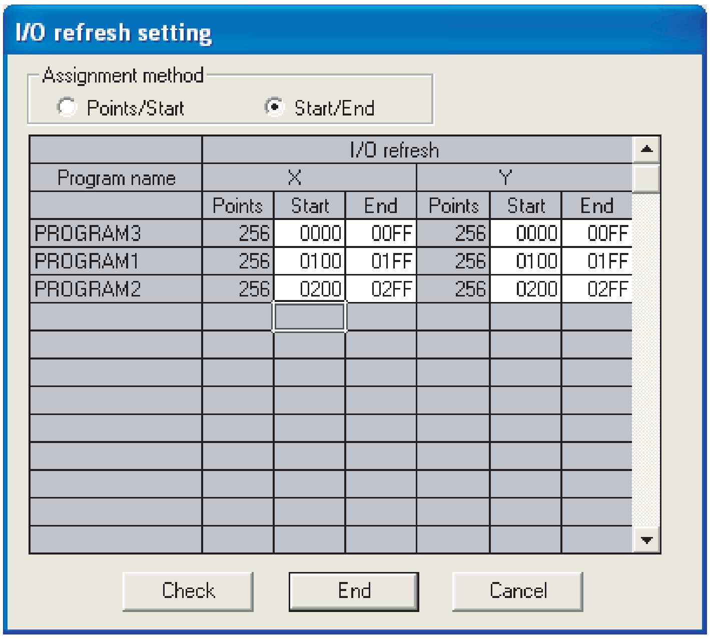
When using a Universal model QCPU, only local device can be set.

When using the file usability setting with a Universal model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.5)

Universal UD Note3.32

(d) I/O refresh setting[Note3.32]

The CPU module updates the I/O of the I/O modules and intelligent function modules by block I/O refresh.(☞ Section 3.8.1)
When I/O refresh setting is performed, the I/O refresh in the specified range can be made for each set program.



I/O refresh setting

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 3.53 I/O refresh setting

1) I/O refresh setting application

The input (X) used before execution of a fixed scan execution type program can be imported, and the output (Y) turned ON/OFF in the fixed scan execution type program can be provided to the outside.

POINT

The scan time of the program being executed (except the fixed scan execution type program) can be checked in the program list monitor.
(☞ Section 6.13.1)

Universal UD Note3.32

I/O refresh setting is not available for the Universal model QCPU.

(2) Each program flow of CPU module

Figure 3.54 shows each program flow in the case where the PLC is powered ON or the CPU module is switched from STOP to RUN.

Redundant Note3.33 Universal Note3.33

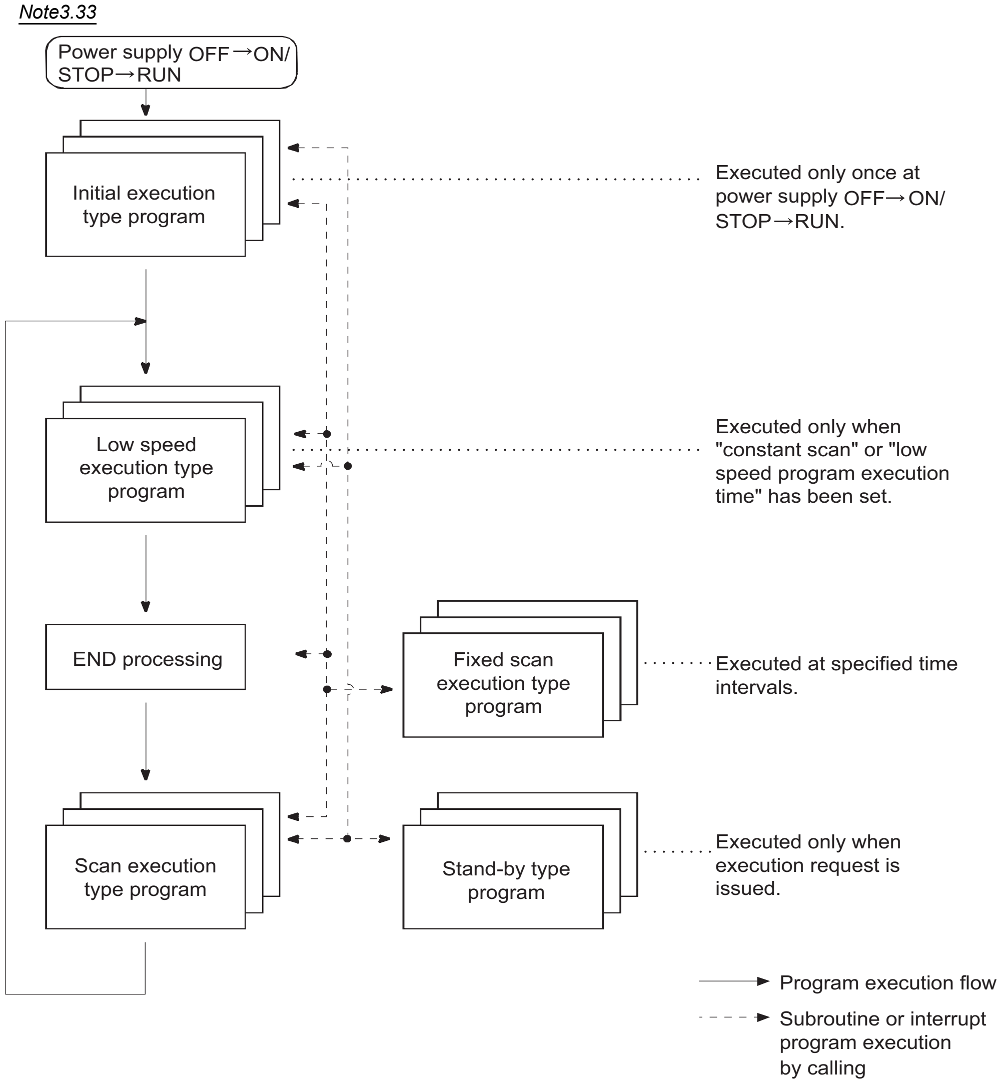

Note3.33
Power supply OFF→ON/ STOP→RUN
Initial execution type program
Executed only once at power supply OFF→ON/ STOP→RUN.
Low speed execution type program
Executed only when "constant scan" or "low speed program execution time" has been set.
END processing
Fixed scan execution type program
Executed at specified time intervals.
Scan execution type program
Stand-by type program
Executed only when execution request is issued.
Program execution flow
Subroutine or interrupt program execution by calling

Figure 3.54 Each program flow

☒POINT

Redundant Note3.33 Universal Note3.33

Use the initial execution type program, low speed execution type program Note3.33, stand-by type program and fixed scan execution type program as necessary.

Redundant Note3.33 Universal Note3.33

The low speed execution type program is not available for the Redundant CPU and Universal model QCPU.

(3) Example of instruction-triggered execution type changing

(a) Execution type changing instruction

Using this instruction can change the execution type even during sequence program execution.

Redundant Universal
Note3.34 Note3.34

Use any of the PSCAN, PLOW<u>Note3.34</u>,PSTOP and/or POFF instructions to change the execution type.

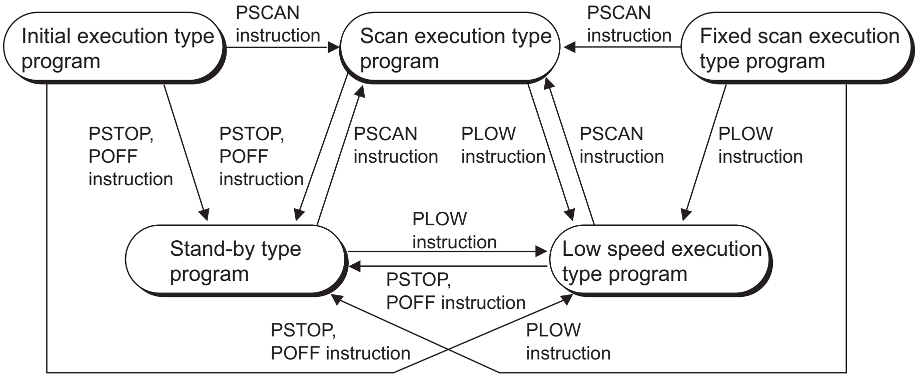


Figure 3.55 Instruction-triggered execution type changing pattern

Redundant Universal
Note3.34 Note3.34

Table3.3 Instruction-triggered execution type changing timing

Execution type before change	Execution instructions			
	PSCAN	PSTOP	POFF	PLOW<u>Note3.34</u>
Scan execution type	Remains unchanged from the scan execution type.	Changes to the stand-by type.	Turns OFF the output at the next scan. Changes to the stand-by type at the next or later scan.	Changes to the low speed execution type.
Initial execution type	Changes to the scan execution type.			
Stand-by type		Remains unchanged from the stand-by type.	No operation	
Low speed execution type <u>Note3.34</u>	Suspends the execution of the low speed execution type and changes to the scan execution type at the next scan. (Executed at Step 0)	Suspends the execution of the low speed execution type and changes to the stand-by type at the next or later scan.	Suspends the execution of the low speed execution type and turns OFF the output at the next scan. Changes to the stand-by type at the next or later scan.	Remains unchanged from the low speed execution type.
Fixed scan execution type	Changes to the scan execution type.	Changes to the stand-by type.	Turns OFF the output at the next scan. Changes to the stand-by type at the next or later scan.	Changes to the low speed execution type.

Redundant Universal
Note3.34 Note3.34

Since the Redundant CPU and Universal model QCPU cannot use low speed execution type programs, it cannot change the execution type using the PLOW instruction.

☒POINT

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

(b) **Example of execution type changing**

In a control program, the stand-by type program matching the preset condition is changed to the scan execution type program.

An unused scan execution type program can also be changed to the stand-by type program.

The following example (Figure 3.56) shows the case where the execution types of the stand-by type programs "ABC", "DEF", "GHI", "JKL" are changed in the control program.

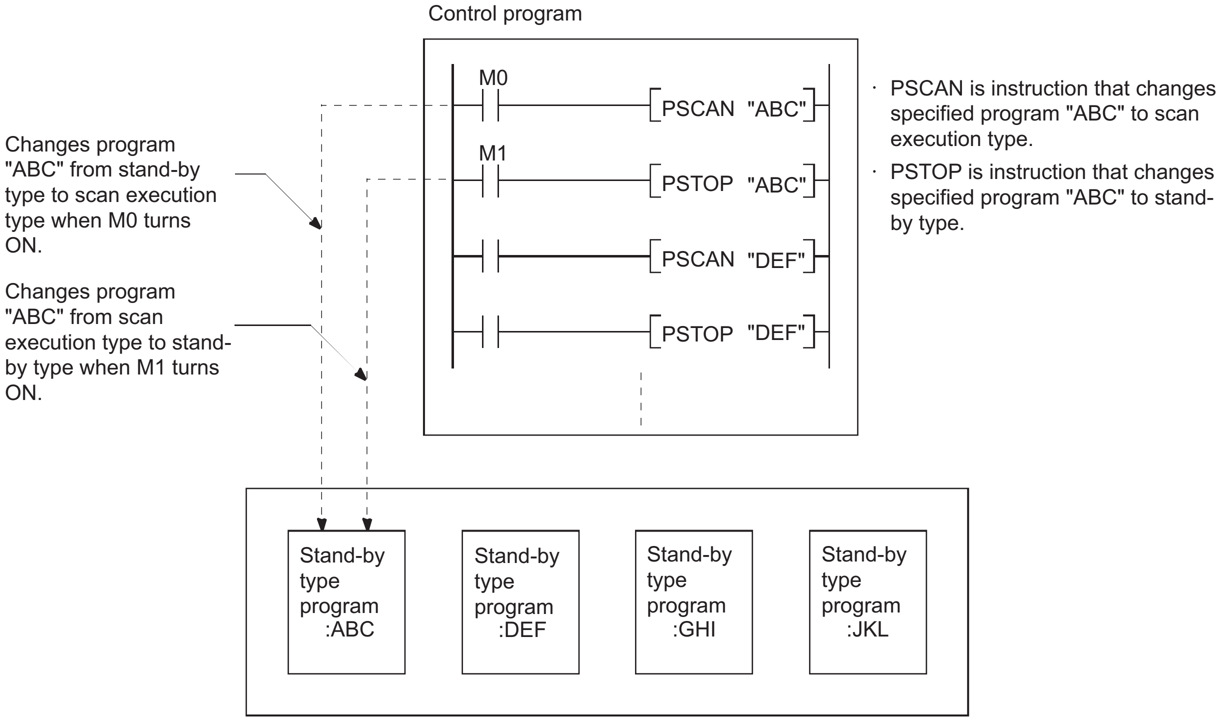

Control program
M0
PSCAN "ABC"
M1
PSTOP "ABC"
PSCAN "DEF"
PSTOP "DEF"
Changes program "ABC" from stand-by type to scan execution type when M0 turns ON.
Changes program "ABC" from scan execution type to stand-by type when M1 turns ON.
· PSCAN is instruction that changes specified program "ABC" to scan execution type.
· PSTOP is instruction that changes specified program "ABC" to stand-by type.
Stand-by type program :ABC
Stand-by type program :DEF
Stand-by type program :GHI
Stand-by type program :JKL

Figure 3.56 Example of instruction-triggered execution type changing

(4) Changing execution type from the program list monitor

The execution type can be changed from the program list monitor of GX Developer. For details, refer to Section 6.13.1 (3).

3.4 Operation Processing

This section explains the operation processing of the CPU module.

3.4.1 Initial processing

Initial processing is a pre-processing to execute sequence operations.

Initial processing is executed once when the actions described in Table3.4 are taken to the CPU module.

When the initial processing is completed, the CPU module is placed in the operation status set by the RUN/STOP switch (RUN/STOP/RESET switch in the case of the Basic model QCPU and Universal model QCPU).(☞ Section 3.5)

Table3.4 Initial processing list

Initial processing item	CPU module status		
	When the power is turned ON	When the reset operation is performed	When the status is switched from STOP to RUN [*1]
The I/O module initialization	○	○	×
Boot from the standard ROM/memory card [*2 *6]	○	○	×
PLC parameter check	○	○	○
Multiple CPU system parameter consistency check[*3]	○	○	○
Device initialization of the range not latched[*4] (bit device: OFF, word device: 0)	○	○	×
Automatic I/O number assignment of mounted modules	○	○	○
CC-Link IE controller network information setting	○	○	×
MELSECNET/H network information setting	○	○	×
Switch setting of intelligent function module	○	○	×
CC-Link information setting	○	○	×
Ethernet information setting	○	○	×
Setting of device initialization values	○	○	○
Serial communication function setting [*5]	○	○	×

○ : Executed × : Not executed

* 1: Indicates that the parameter or program was changed in a STOP status and the CPU was placed in a RUN status without being reset.
(Move the RUN/STOP switch (RUN/STOP/RESET switch in the case of the Basic model QCPU and Universal model QCPU) from STOP to RUN (RUN LED flickers) and then STOP to RUN.)
Note that the pulsing instruction (PLS,□P) may not act normally with the above operation since the previous information may not be inherited depending on the program changes.

* 2: Boot from a memory card is inexecutable since the memory card cannot be used in the Basic model QCPU.

* 3: Multiple CPU system parameter consistency check is not performed for the Redundant CPU, since it cannot configure the multiple CPU system.

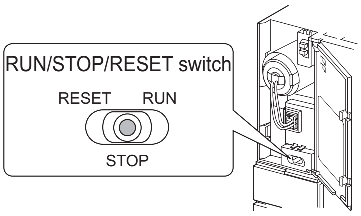
* 4: When the start mode is the hot-start mode in the Redundant CPU, devices outside the latch range are not initialized. (Except some devices such as a step relay and an index register.)

* 5: The serial communication function is not available for the High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU.

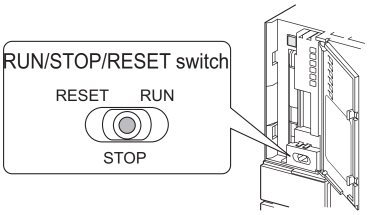
* 6: Boot from a standard ROM is inexecutable for the Universal model QCPU. (☞ Section 5.2.3)

☒POINT

1. The STOP, RUN and RESET switches of the CPU module are provided differently depending on the CPU module.
   - Switch of Basic model QCPU and Universal model QCPU

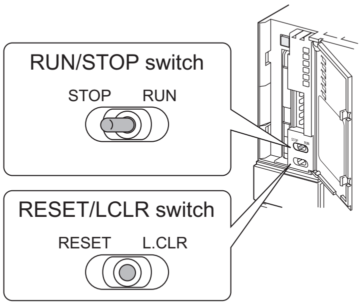


Basic model QCPU



Universal model QCPU

   - Switches of High Performance model QCPU, Process CPU and Redundant CPU



2. When a parameter or program has been changed in the STOP status, reset the CPU with the RESET/L.CLR switch (RUN/STOP/RESET switch in the case of the Basic model QCPU and Universal model QCPU).

3.4.2 I/O refresh (I/O module refresh processing)

In the I/O refresh, ON/OFF data are input from the input module or intelligent function module to the CPU module, and are output from the CPU module to the output module or intelligent function module.

(1) I/O refresh timing

The I/O refresh is executed before the sequence program operation starts.
When the constant scan is set, the I/O refresh is performed after the constant scan waiting time has elapsed.
(The I/O refresh is executed at each constant scan cycle.)

3.4.3 Automatic refresh of the intelligent function module

When the automatic refresh setting of the intelligent function module has been made, access is made to the set data of the intelligent function module.
Refer to the manual of the intelligent function modules, for details on the automatic refresh setting of intelligent function modules.

3.4.4 END processing

End processing is a post-processing to end the whole sequence program operation processing in one scan and to return the operation to step 0.

(1) Operations of END processing

The END processing includes the following.

Table3.5 END processing list

END processing item	CPU modules performing END processing					Reference
	Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU	
CC-Link IE controller network refresh processing	○	○	○	○	○	----
MELSECNET/H or CC-Link refresh processing	○	○	○	○	○	----
Automatic refresh processing of intelligent function module	○	○	○	○	○	Section 8.1.1
Self-diagnostic processing	○	○	○	○	○	Section 6.17
Communication with external device such as GX Developer	○	○	○	○	○	----
Processing of instructions dedicated to intelligent function modules	○	○	○	○	○	----
Collection of device data by sampling trace function (when trace point is set to every scan (after END instruction execution)	×	○	○	○	○	Section 6.14
Automatic refresh processing of CPU shared memory	○	○	○	×	○	QCPU User's Manual (Multiple CPU System)
Watchdog timer reset processing	○	○	○	○	○	Section 6.16
Setting values to special relays/ special registers (when setting is made at END processing)	○	○	○	○	○	Appendix 1 Appendix 2

○ : Executed × : Not executed

☒POINT

1. When the constant scan function(☞ Section 6.2)is set, END processing time result is stored until when END processing is completed or the next scan starts.
2. When executing the low speed execution type program, the low speed END processing starts after the all low speed execution type programs are completed.Note3.35(☞ Section 3.3.3)

Basic Redundant Universal
Note3.35 Note3.35 Note3.35

Basic Redundant Universal
Note3.35 Note3.35 Note3.35

Since the Basic model QCPU, Redundant CPU and Universal model QCPU cannot use low speed execution type programs, the low speed END processing is not performed.

3.5 RUN, STOP, PAUSE Operation Processing

CPU module has three types of operation status; RUN, STOP and PAUSE status.
CPU module operation processing is explained below:

(1) RUN Status Operation Processing

RUN status is a status where the sequence program operations are repeatedly performed from step 0 to the END (FEND) instruction, and back to step 0.

(a) Output status when changing into RUN status

The processing time between the status change from STOP to RUN and the sequence program operation start varies depending on the system configuration and parameter settings. (Normally 1 to 3 seconds) Note that the processing time may increase depending on the conditions.(☞ Section 6.4)

(b) Processing time before operation start

The processing time taken from switching STOP to RUN until the operation start of the sequence program varies with the system configuration and parameter settings. (Normally 1 to 3 seconds)
Note that it may increase depending on the conditions.

(2) STOP Status Operation Processing

STOP status is a status where the sequence program operations are stopped by the RUN/STOP switch (the RUN/STOP/RESET switch in the case of the Basic model QCPU and Universal model QCPU) or the remote STOP function (☞ Section 6.6.1)
The CPU module is also placed in the STOP status when a stop error occurs.

(a) Output status when changing into STOP status

When changing into the STOP status, the CPU module saves the output (Y) status and turns all output points OFF.
The device memory of other than the output (Y) is retained.

(3) PAUSE Status Operation Processing

PAUSE status is a status where the sequence program operations are stopped by the remote PAUSE function (☞ Section 6.6.2) while retaining the output and device memory status after executing the program for one scan.

(4) CPU module operation processing by switch operation

Table3.6 Operation processing by switch operation

RUN/STOP status	CPU module operation processing			
	Sequence program operation processing	External output	Device memory	
			M,L,S,T,C,D	Y
RUN→STOP	Executes up to the END instruction and stops.	Saves the output (Y) status immediately before switching to the STOP status, and turns all points OFF.	Saves the device memory status immediately before switching to the STOP status.	Saves the output (Y) status immediately before switching to the STOP status, and turns all points OFF.
STOP→RUN	Starts at step 0.	Determined by the "STOP→RUN-time output mode" in the PLC parameter dialog box.	Saves the device memory status immediately before switching to the STOP status. However, when the initial device values have been set, the values are used. Clears the local devices*1.	Determined by the "STOP→RUN-time output mode" in the PLC parameter dialog box. (☞ Section 6.4)

* 1: Since the Basic model QCPU cannot execute multiple programs, it cannot use subroutine programs in the other programs.

☒POINT

The CPU module performs the following in any of RUN, STOP, and PAUSE status:

- I/O module refresh processing
- Network module refresh processing
- Automatic refresh processing of intelligent function module
- Self-diagnostic processing
- Communication processing with external device such as GX Developer
- Processing of instructions dedicated to intelligent function modules (completion processing only)
- Operation processing of Multiple CPU high speed transmission function (Universal model QCPU only)

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

- I/O monitor or test operation using GX Developer
- Read/write from/to the external device using the MC protocol
- Communication with other stations using CC-Link IE controller network and 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 momentary power failure occurs for a period shorter than the permitted power failure time

The output is maintained when the momentary power failure occurs, and error history are logged. Then the system interrupts the operation processing.
(The timer clock continues.)

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

When a resume start is specified for the SFC program, the system is saved.

(b) When recovered from momentary power failure

When a momentary power failure ends, the operation processing is resumed.

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

Even if the operation is interrupted due to momentary power failure, the watchdog timer (WDT) measurement continues. For example, if the GX Developer PLC parameter mode WDT setting is set at 200 ms, when a momentary power failure of 15 ms occurs at scan time 190 ms, the watchdog timer error is set.

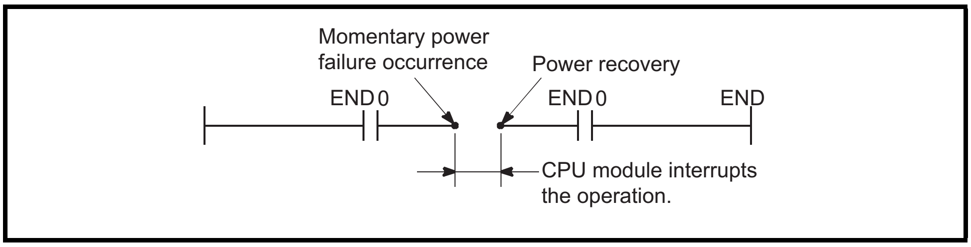

Momentary power failure occurrence
Power recovery
END 0
END 0
END
CPU module interrupts the operation.

Figure 3.57 Operation during momentary power failure

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

CPU module starts initially.

The same operation processing as that after the following operation occurs.

- Power ON
- Resetting using RESET/L. CLR switch (RUN/STOP/RESET switch in the case of the Basic model QCPU and Universal model QCPU)
- Remote setting using GX Developer

☒POINT

1. In the redundant power supply system, operation is not suspended if a momentary power failure occurs in either of the power supply modules, while it is suspended if a momentary power failure occurs under the condition where the power was supplied to only one of them.
2. The information of a momentary power failure for the redundant power supply system is stored in SM1782 to 1783 and SD1782 to 1783.
   The information of a momentary power failure for not redundant power supply system is stored in SM53 and SD53. (☞Appendix 1, Appendix 2)

3.7 Data Clear Processing

This section explains how to clear the CPU module data and the settings about the latch data clearing.

(1) Clearing data in the CPU module

Data in the CPU module can be cleared when the reset operation with the RESET/ L.CLR switch (the RUN/STOP/RESET switch in the case of the Basic model QCPU and Universal model QCPU) is performed or when the power is switched from OFF to ON.

However, the data in (a) below cannot be cleared by the above operations.

(a) Data that cannot be cleared by reset operation

- Program memory data
- Standard ROM data
- Memory card data[Note3.36]
- Latch-specified device data(☞ (2) in this section)
- File register data

Basic Note3.36

(b) How to clear data that cannot be cleared by reset operation

The data indicated in (a) above can be cleared by the following methods.

1) Program memory data

Clear the data in either of the following methods.

- Check the box and set "Clear program memory" on the Boot file setting tab of PLC parameter.[Note3.37]
- Clear the program memory on the screen displayed by selecting [Online] → [Delete PLC data] in GX Developer.

Basic Note3.37

2) Standard ROM data

Automatically cleared when data are written to the standard ROM.

3) Memory card data[Note3.36]

Basic Note3.36

Clear the data in the memory card on the screen displayed by selecting [Online] → [Delete PLC data] in GX Developer.

4) Latch-specified device data

Refer to (2) in this section.

5) File register data

Reset the data using the RST instruction or transfer K0 using the MOV/FMOV instruction.

Or, execute "Clear device's whole memory" from the screen displayed by selecting [Online] → [Clear PLC memory] in GX Developer.

Basic Note3.36

The memory card is not available for the Basic model QCPU.

Basic Note3.37

The Basic model QCPU cannot set "Clear program memory" on the Boot file setting tab of PLC parameter.

(2) Device latch specification

Make the latch specification (latch range setting) for each device in the device setting of the PLC parameter dialog box of GX Developer.(☞ Section 6.3(5))

(a) Latch range setting

The latch range can be set by the following 2 methods using GX Developer.

Basic Universal UD Note3.38 Note3.38

1) **Latch clear operation enable (Latch (1) first/last)**
   Set the latch range that can be cleared by the latch clear operation performed by the RESET/L.CLR switch[Note3.38] or the remote latch clear.

Basic Universal UD Note3.38 Note3.38

2) **Latch clear operation disable (Latch (2) first/last)**
   Set the latch range that cannot be cleared by the latch clear operation performed by the RESET/L.CLR switch[Note3.38] or the remote latch clear.

Basic Universal UD Note3.38 Note3.38

(b) Clearing device data set to latch clear operation enable

- Latch clear operation by the RESET/L.CLR switch[Note3.38]
- Clear operation by the remote latch clear(☞ Section 6.6.4)from GX Developer

(c) Clearing device data set to latch clear operation disable

The device data set to latch clear operation disable can be cleared only by the instruction or the clear operation using GX Developer.

1) **Clearing by instruction**
   Reset the device data using the RST instruction or transfer K0 using the MOV/FMOV instruction.
   ☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

2) **Clearing by GX Developer**
   Choose [Online] → [Clear PLC memory] and execute device memory all clear (including latch).

☒POINT

1. When the start mode of the Redundant CPU is the hot-start mode, the data outside the latch range are also held.
   (Except some devices such as the step relay and index register.).
2. For the Universal model QCPU, the scan time is prolonged when the device is latched. When latching a device, consider the prolonging scan time.
   (☞ Section 10.1.2 (11))

Remark

Refer to the following manual for the operation of GX Developer.
☞ GX Developer Operating Manual

Basic Universal UD Note3.38 Note3.38

The Basic model QCPU and Universal model QCPU cannot perform latch clear by switch operation.

3.8 I/O Processing and Response Lag

The CPU module performs I/O processing in a refresh mode.
Using direct access I/O in a sequence program, however, allows the CPU module to perform direct mode I/O processing corresponding to each instruction.
This section explains these I/O processing modes and response lags of the CPU module.

(a) Refresh mode(☞ Section 3.8.1)
The refresh mode batch-accesses I/O modules before start of sequence program operation.

(b) Direct access mode(☞ Section 3.8.2)
The direct mode accesses an I/O module when each instruction is executed in a sequence program.
To access an I/O module in the direct mode, use the direct access input or direct access output in a sequence program.

(1) Differences between refresh mode and direct access mode

Since the direct access mode makes direct access to an I/O module at execution of an instruction, it imports input data faster than the refresh mode.
However, its instruction processing time is longer than that of the refresh mode.
The direct access mode is available for the I/O used with the I/O modules and intelligent function modules mounted on the main base unit and extension base units.
The differences between the refresh mode and the direct access mode are shown in Table3.7.

Table3.7 List of differences between refresh mode and direct access mode

Item	Refresh mode	Direct access mode
I/O module mounted on base unit	Available	Available
I/O of intelligent function module mounted on base unit		
I/O of I/O link mounted on extension base unit[*1]		
I/O used in CC-Link IE controller network, MELSECNET/H network system or CC-Link system	Available	N/A

[* 1] The Basic model QCPU cannot use the I/O of the I/O link mounted on extension base units.

3.8.1 Refresh mode

(1) Definition of refresh mode

The refresh mode batch-accesses I/O modules before start of sequence program operation.

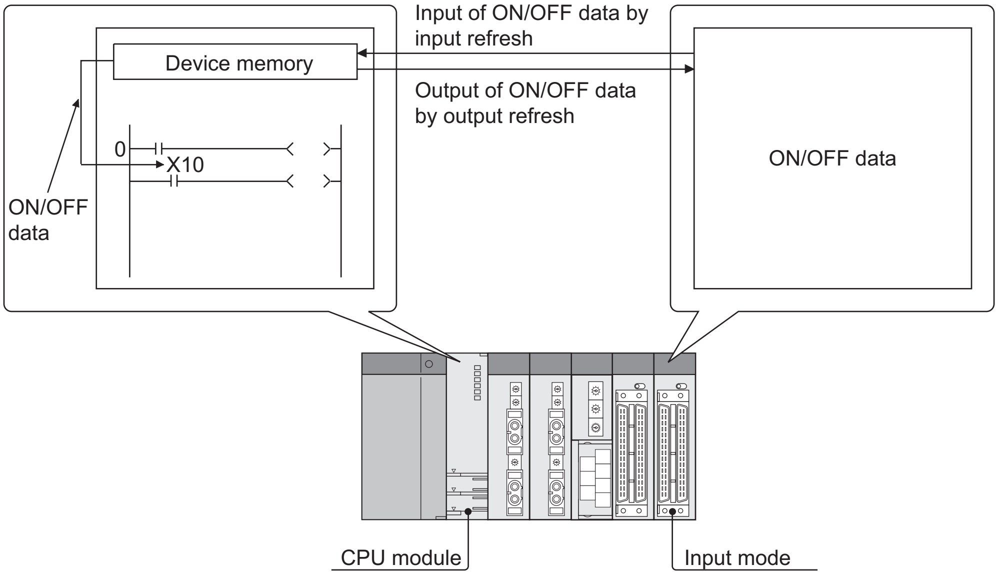

Device memory
Input of ON/OFF data by input refresh
Output of ON/OFF data by output refresh
0
X10
ON/OFF data
ON/OFF data
CPU module
Input mode

Figure 3.58 Refresh mode

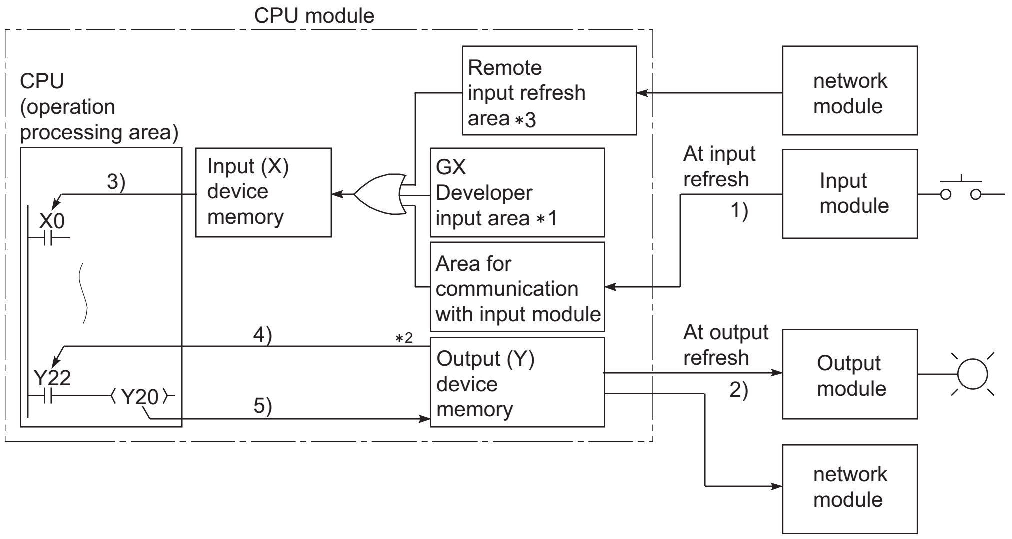
(2) Input

The ON/OFF information of the input module is batch-imported to the area for communication with the input module inside the CPU module before start of sequence program operation.

When a sequence program is executed, the ON/OFF data of the input (X) device memory are used for the operation.

(3) Output

The operation result in an output (Y) sequence program is output to the output (Y) device memory in the CPU module every time the operation is performed, and the ON/OFF data of the output (Y) device memory are batch-output to the output module before start of sequence program operation.



- Input refresh:

  Before start of the sequence program operation, input data are batch-read from the input module (1) and ORed with the GX Developer input area or remote input refresh area data, and the results are stored into the input (X) device memory.

- Output refresh:

  Data in the output (Y) device memory is output in a batch 2) to the output module before sequence program operation starts.

- **When an input contact instruction has been executed:**

  Input information is read 3) from the input (X) device memory, and a sequence program is executed.

- When an output contact instruction has been executed:

  Output information is read 4) from the output (Y) device memory, and a sequence program is executed.

- When an output OUT instruction has been executed:

  The sequence program operation result 5) is stored in the output (Y) device memory.

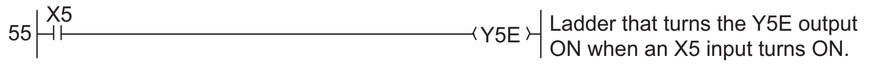
Figure 3.59 I/O Information Flow in Refresh Mode

* 1 : The input area for GX Developer can be turned ON/OFF by:
  - Test operation using GX Developer
  - Write from network module
* 2 : The output (Y) device memory can be turned ON/OFF by:
  - Test operation using GX Developer
  - Link refresh of CC-Link IE controller network,
  - Link refresh of MELSECNET/H network system
  - Write from external device using MC protocol
  - Automatic refresh of CC-Link
* 3 : The remote input refresh area indicates the area used when automatic refresh setting is made to the input (X) with CC-Link IE controller network, MELSECNET/H and CC-Link.
  Automatic refresh of the remote input refresh area is performed during END processing.

(4) Response lag

An output change lags a maximum of two scans behind an input module change depending on the ON timing of an external contact.

Ladder examples


55
X5
Y5E
Ladder that turns the Y5E output ON when an X5 input turns ON.

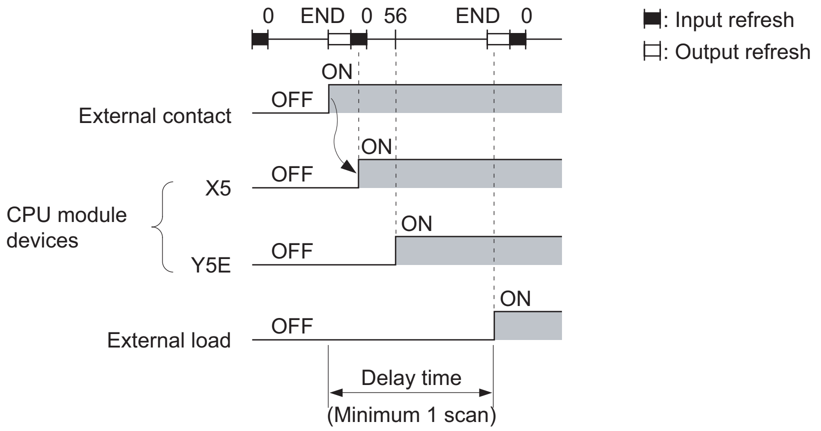


Figure 3.60 When Y5E turns ON fastest

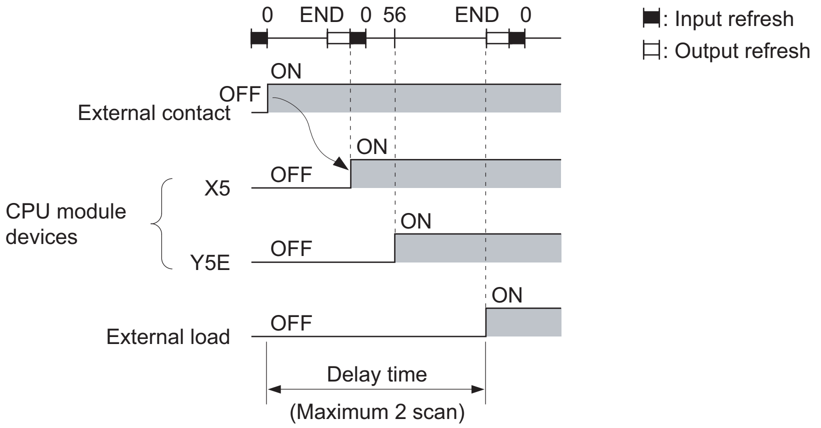


Figure 3.61 When Y5E turns ON slowest

3.8.2 Direct mode

(1) Definition of direct mode

The direct mode accesses an I/O module when each instruction is executed in a sequence program.

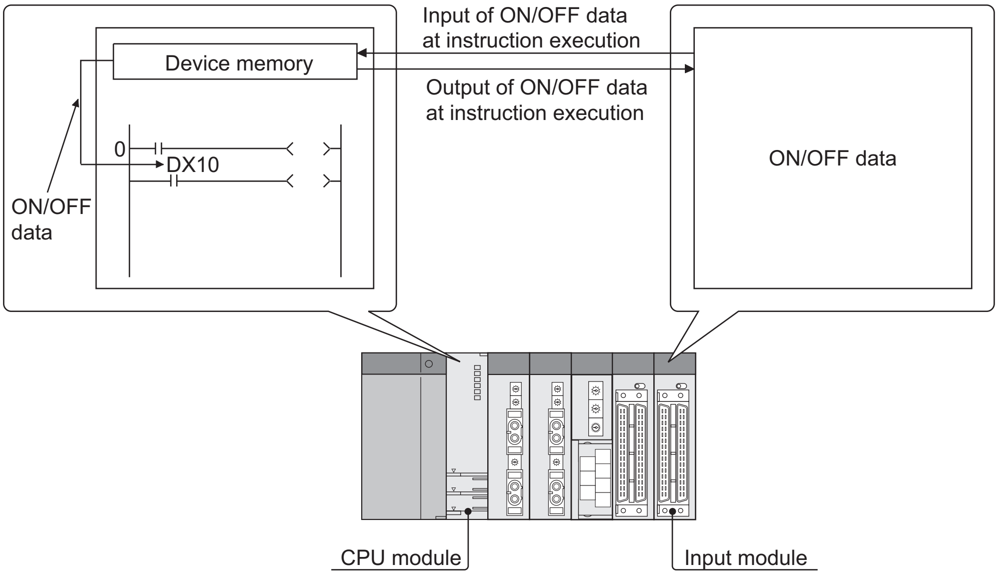
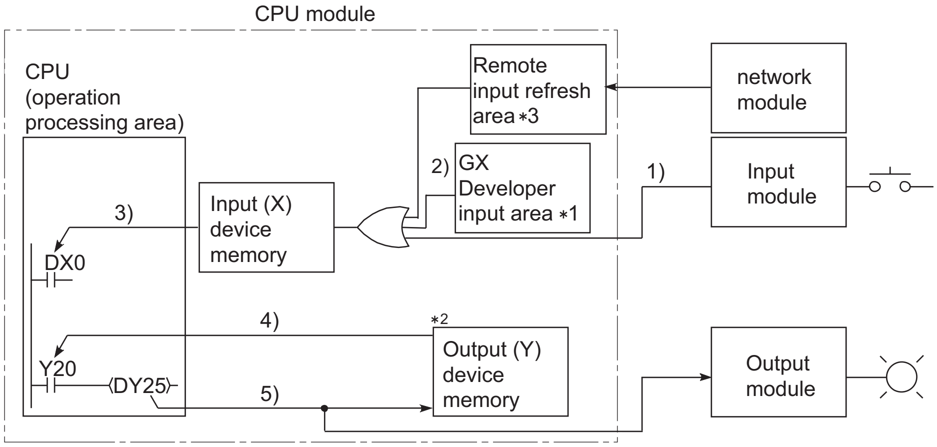


Figure 3.62 Direct mode

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



- When an input contact instruction has been executed:
  The input module's input information 1) is ORed with GX Developer input area's input information 2) or remote input refresh area data, and the result is stored into the input (X) device memory. Then this data is then used as input information 3) at sequence program execution.
- When an output contact instruction has been executed:
  Output information 4) is read from the output (Y) device memory, and a sequence program is executed.
- When an output OUT instruction has been executed:
  The sequence program's operation result 5) is output to the output module, and is stored in the output (Y) device memory.

Figure 3.63 I/O Information Flow in Direct Mode

* 1 : The input area for GX Developer can be turned ON/OFF by:
- Test operation using GX Developer
- Write from network module

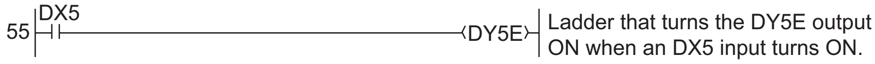
* 2 : The area for the device memory output (Y) can be turned ON/OFF by:
- Test operation using GX Developer
- Link refresh of CC-Link IE controller network
- Link refresh of MELSECNET/H network system
- Write from external device using MC protocol
- Automatic refresh of CC-Link

* 3 : The remote input refresh area indicates the area used when automatic refresh setting is made to the input (X) with CC-Link IE controller network, MELSECNET/H and CC-Link.
Automatic refresh of the remote input refresh area is performed during END processing.

(2) Response lag

An output change lags a maximum of one scan behind an input module change depending on the ON timing of an external contact.

Ladder examples



Ladder that turns the DY5E output ON when an DX5 input turns ON.

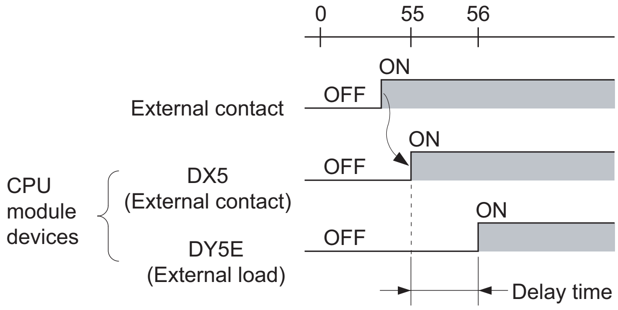


Figure 3.64 When DY5E turns ON fastest

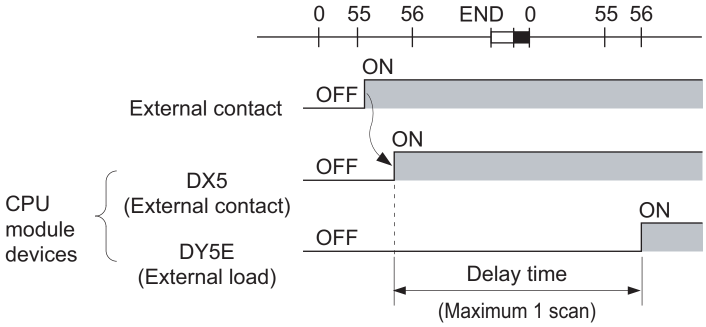


Figure 3.65 When DY5E turns ON slowest

3.9 Numeric Values which can be Used in Sequence Programs

Numeric and alphabetic data are expressed by "0" (OFF) and "1" (ON) numerals in the CPU module.
This expression form is called "binary code" (BIN).
The hexadecimal (HEX) expression form in which BIN data are expressed in 4-bit units, and the BCD (binary coded decimal) expression form are applicable to the CPU module.
Real numbers may also be used.(☞ Section 3.9.4)
Table3.8 shows the numeric expressions of BIN, HEX, BCD and DEC (decimal).

Table3.8 BIN, HEX, BCD, and Decimal Numeric Expressions

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) Numeric value input from outside to CPU module

When setting a numeric value from an external digital switch or similar device to the CPU module, BCD (binary coded decimal) can be used as the same setting in DEC (decimal) by the method given in (b).

(a) Numeric values handled in CPU module

The CPU module performs operation in BIN (binary).
If the value set in BCD is used as-is, the CPU module recognizes the set value as a BIN and performs operation.
Hence, operation is performed using the value different from the set value.
(☞ (b) below)

(b) How to enter numeric value without taking into account BIN notation

Use the BIN instruction to convert the data set in BCD into BIN used in the CPU module.
Using the BIN instruction allows users to set numeric value data from the outside without taking into account BIN notation.

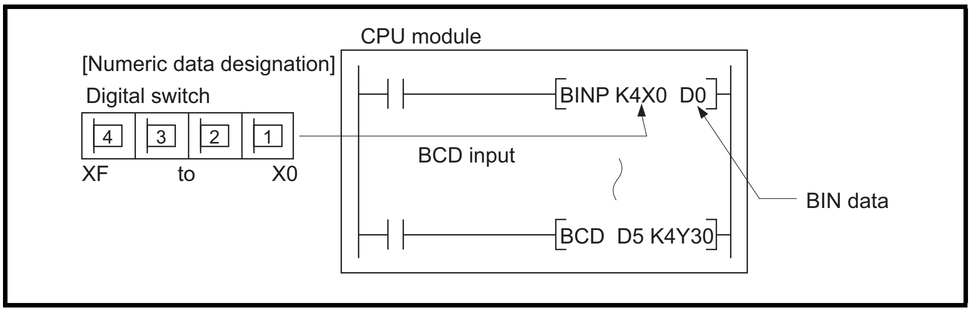


Figure 3.66 Import of data from digital switch to CPU module

Remark

Refer to the following manual for details of the BIN instruction.
☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

(2) Numeric value output from CPU module to outside

A digital display or similar device is available to externally display the numeric value operated by the CPU module.

(a) How to output numeric value

The CPU module performs operation in BIN.
If binary values used in the CPU module are output as they are to a digital display, they will not displayed correctly.
Therefore, the BCD instruction is used to convert the data operated in BIN into BCD used by the external display or similar device.
Using the BCD instruction allows the same display as in DEC (decimal) to be provided on the external display or similar device.

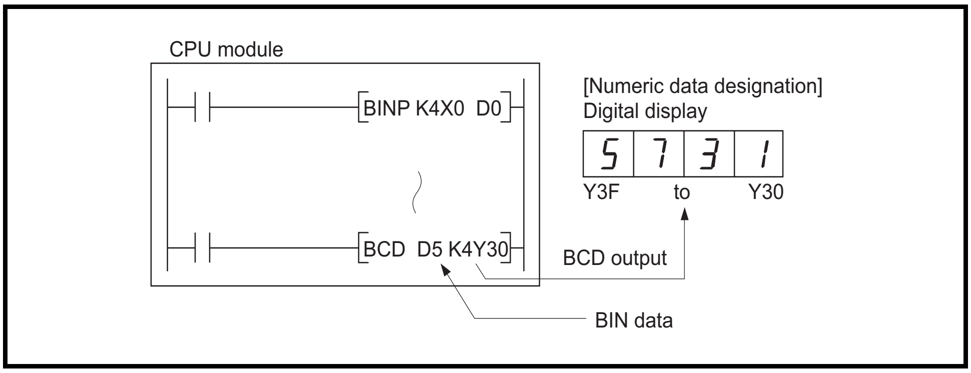


Figure 3.67 Display of CPU module operation data by digital display

Remark

Refer to the following manual for details of the BCD instruction.
☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

3.9.1 BIN (Binary Code)

(1) Binary code

Binary date is represented by 0 (OFF) and 1 (ON).

Decimal notation uses the numerals 0 through 9. When counting beyond 9, a 1 is placed in the 10s column and a 0 is placed in the 1s column to make the number 10.

In binary notation, the numerals 0 and 1 are used. A carry occurs after 1 and the number becomes 10 (decimal 2).

Table3.9 shows the numerical notation by BIN and DEC.

Table3.9 Comparison between Binary and Decimal Notations

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) Binary numeric expression

(a) Bit configuration in BIN notation used in CPU module

Each CPU module register (data registers, link registers, etc.) consists of 16 bits.

(b) Numeric data available for CPU module

Each CPU module register can store numeric values of -32768 to 32767.

The following shows the numeric notation for CPU module registers.

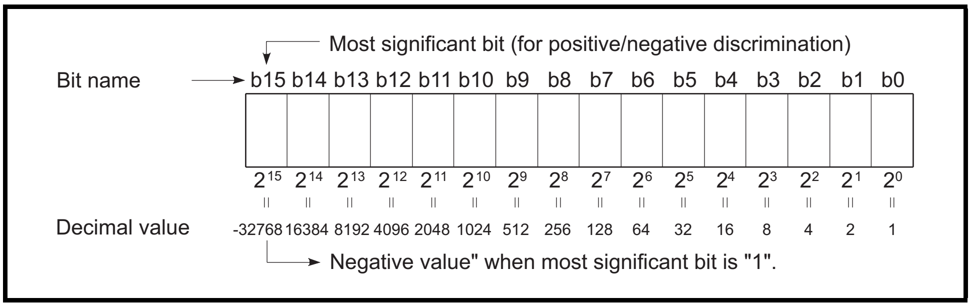


Figure 3.68 Numeric Expressions for CPU module Registers

☒POINT

To each bit of each register, a 2^n value is assigned.
Note that an unsigned (no sign) binary number (0 to 65535) cannot be used since the most significant bit is used for discrimination of sign (positive or negative).
1) When most significant bit is "0"...Positive
2) When most significant bit is "1"...Negative

3.9.2 HEX (Hexadecimal)

(1) Hexadecimal notation

In hexadecimal notation, 4 binary bits are expressed in 1 digit.

If 4 binary bits are used in binary notation, 16 different values from 0 to 15 can be represented.

Since hexadecimal notation represents 0 to 15 in 1 digit, letters AH to FH are used to represent the numbers 10 to 15.

Then, a carry occurs after FH.

Table3.10 shows the numeric expressions of BIN, HEX and DEC (decimal).

Table3.10 Comparison of BIN, HEX, and DEC Numeric Expressions

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) Hexadecimal numeric expression

CPU module registers (data registers, link registers, etc.) consist of 16 bits.

For 16 bits, 0 to FFFFH can be specified in hexadecimal.

3.9.3 BCD (Binary Coded Decimal)

(1) BCD notation

BCD (binary coded decimal) is a numbering system in which one digit of DEC (decimal) is expressed in BIN (binary).

Though it uses 4-bit representation like hexadecimal notation, it dose not use letters AH to FH.

Table3.11 shows the numeric expressions of BIN, BCD and DEC.

Table3.11 Comparison of BIN, BCD, and DEC Numeric Expressions

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) BCD numeric expression

CPU module registers (data registers, link registers, etc.) consist of 16 bits. Therefore, the numeric values expressed in BCD notation can be stored in each register within 0 to 9999 range.

3.9.4 Real numbers (Floating-point data)

Real number data includes the single-precision floating-point data and the double-precision floating-point data.
The Universal model QCPU can use the double-precision floating-point data.

(1) Single-precision floating-point data

(a) Internal expression of real number data

The CPU module internal expression of received real number data is explained below.
Real number data is expressed as shown below, using 2 word devices.

[Sign] 1.[Mantissa] × $2^{\text{(exponent part)}}$

The bit configuration of internal expression for the real number data is as follows:

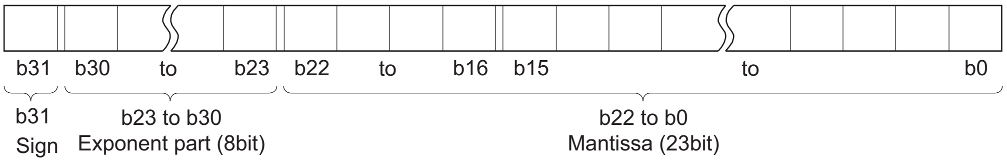


Figure 3.69 Bit configuration of real number data

1) **Sign**
The sign is expressed at b31 as follows.
0: Positive
1: Negative

2) **Exponent part**
The “n” of 2n is expressed by b23 to b30, depending on the BIN value in b23 to b30.

b30 to b23	FF_H	FE_H	FD_H			81_H	80_H	$7F_H$	$7E_H$			02_H	01_H	00_H
n	Not used	127	126			2	1	0	-1			-125	-126	Not used

Figure 3.70 Relation between value stored in exponent part and exponent

3) **Mantissa:**
For a binary number “1.XXXXXX.....”, the “XXXXXX.....” portion is expressed by 23 bits (b0 to b22).

(b) Calculation example

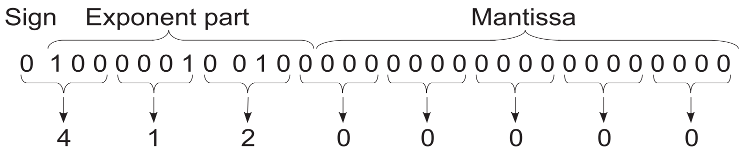
Calculation examples are shown below (the nnnnn "X" indicates an X-system data expression).

1) Storing "10"

$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.01000.....\times 2^3)_2$

SignPositive to 0
Exponent part3 to 82_H to $(1000\ 0010)_2$
Mantissa$(010\ 00000\ 00000\ 00000\ 00000)_2$

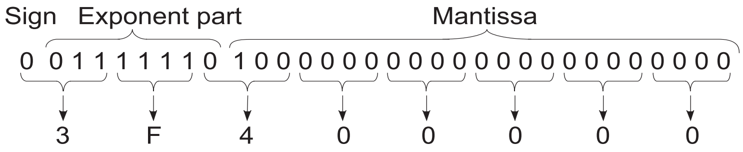
Therefore, the data expression will be 41200000_H, as shown below.



2) Storing "0.75"

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

SignPositive to 0
Exponent part-1 to $7E_H$ to $(0111\ 1110)_2$
Mantissa$(100\ 00000\ 00000\ 00000\ 00000)_2$
Therefore, the data expression will be $3F400000_H$, as shown below.



☒POINT

In binary notation, the portion of the value following the decimal point is calculated as follows:

Example) $(0.1101)_2$

0.	1	1	0	1
	↑	↑	↑	↑
	This bit expresses2^{-1}	This bit expresses2^{-2}	This bit expresses2^{-3}	This bit expresses2^{-4}

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

Basic Process Redundant
Note3.39 Note3.39 Note3.39

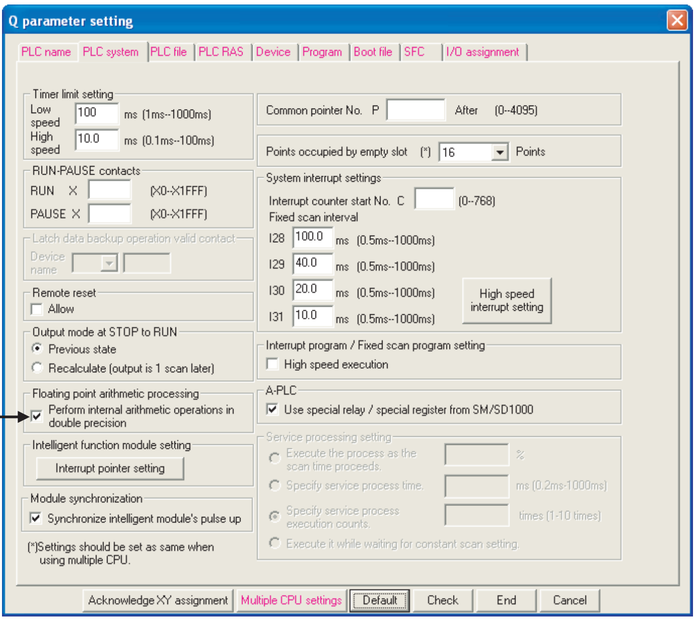
Universal
Note3.39

(c) Method and precautions for internal operation processing with double precisionNote3.39

The High Performance model QCPU permits selection of "Perform internal operation processing with double precision" or "Do not perform internal operation processing with double precision".

1) Settings for internal operation processing with double precision

Make settings in the floating-point operation processing in the PLC system of the PLC parameter dialog box (Default: Perform internal operation processing with double precision).



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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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.

To turn off "Perform internal arithmetic operations in double precision", uncheck the check box.

Figure 3.71 PLC system screen

2) Operation result

The operation result is single precision regardless of the floating-point operation processing setting.

In the internal operation processing with double precision, internal operation only is performed with double precision (64 bits).

Basic Process Redundant
Note3.39 Note3.39 Note3.39

Universal
Note3.39

The internal operation processing with double precision is not available for the Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU.
It is available only for the High Performance model QCPU.

3) **Applications and precautions**
The following explains the applications and precautions for the cases where internal operation processing is performed with double precision and not performed.

- When internal operation processing is performed with double precision
This option is used when accuracy is required to ensure compatibility with the conventional models.
When using an instruction such as SIN or COS, with which many real number operations are performed internally, accuracy is increased by performing internal operation processing with double precision.
- When internal operation processing is not performed with double precision
This option is used when real number operation speed must be increased. However, since internal operation is performed with single precision (32 bits), real number operation speed increases but accuracy may decrease.

☒POINT

1. The monitor function for GX Developer permits monitoring the real number data of the CPU module.However, if an attempt is made to monitor the data that cannot be represented as a real number, e.g. "FFFFH", "_____" is displayed.
2. For a "0" value, "0" will be indicated at all the b0 to b31 bits.

Basic Note3.40 High Performance Note3.40 Process Note3.40

Redundant Note3.40

(2) Double-precision floating-point dataNote3.40

(a) Internal expression of real number data

The following shows the internal expression of real number data dealt with the Universal model QCPU.

Using four word deices, the real number data is expressed as follows:

[Sign] 1.[Mantissa] × $2^{\text{(exponent part)}}$

The bit configuration of internal expression for the real number data is as follows:

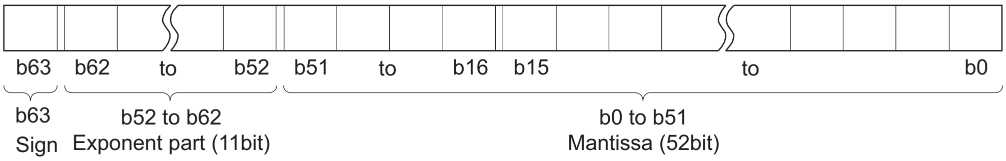


Figure 3.72 Bit configuration of real number data

1) Sign

The b63 shows a sign.

0: Positive

1: Negative

2) Exponent part

The “n” of 2n is expressed by b52 to b62, depending on the BIN value in b52 to b62.

b52 to b62	7FF$_H$	7FE$_H$	7FD$_H$			400$_H$	3FF$_H$	3FE$_H$	3FD$_H$	3FC$_H$			02$_H$	01$_H$	00$_H$
n	Not used	1023	1022			2	1	0	-1	-2			-1021	-1022	Not used

Figure 3.73 Relation between value stored in exponent part and exponent

3) Mantissa:

For a binary number “1.XXXXXX.....”, the “XXXXXX.....” portion is expressed by 52 bits (b0 to b51).Note39

Basic Note3.40 High Performance Note3.40 Process Note3.40

Redundant Note3.40

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU cannot use the double-precision floating-point data.

(b) Calculation example

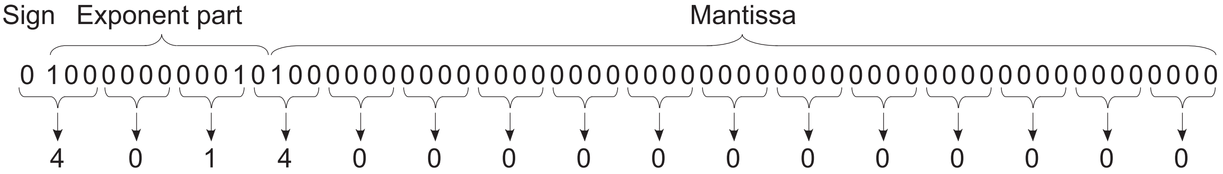
Calculation examples are shown below (the nnnnn "X" indicates an X-system data expression).

1) Storing "10"

$$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.01000.....\times 2^3)_2$$

SignPositive to 0
Exponent part3 to 401H to $(100\ 0000\ 0001)_2$
Mantissa$(0100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000)_2$

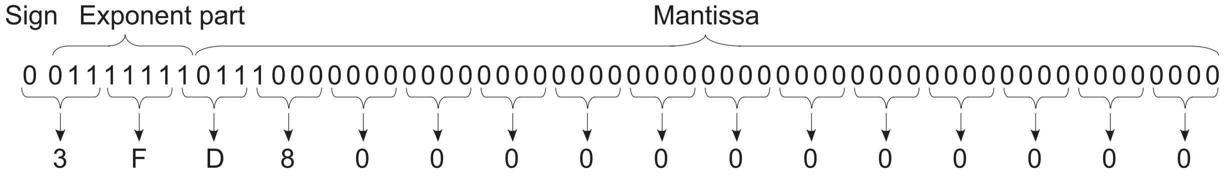
Therefore, the data expression will be 4014000000000000H, as shown below.



2) Storing "0.75"

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

SignPositive to 0
Exponent part--1 to 3FDH to $(011\ 1111\ 1101)_2$
Mantissa$(1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000)_2$
Therefore, the data expression will be 3FD8000000000000H, as shown below.



☒POINT

In binary notation, the portion of the value following the decimal point is calculated as follows:

Example) $(0.1101)_2$

0.	1	1	0	1
	↑	↑	↑	↑
	This bit expresses2^{-1}	This bit expresses2^{-2}	This bit expresses2^{-3}	This bit expresses2^{-4}

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

Basic
Note3.41

3.10 Character String Data

(1) Character String Data Note3.41

The CPU module uses ASCII code data.

(2) ASCII code character strings

ASCII code character strings are shown in Table3.12.

"00H" (NUL code) in Table3.12 is used at the end of a character string.

Table3.12 ASCII code character strings

b8									0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	b7								0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
		b6							0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
			b5						0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
									Column															
b8	b7	b6	b5	b4	b3	b2	b1	Low	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
				0	0	0	0	0	NUL		(SP)	0	@	P	`	p								
				0	0	0	1	1			!	1	A	Q	a	q								
				0	0	1	0	2			"	2	B	R	b	r								
				0	0	1	1	3			#	3	C	S	c	s								
				0	1	0	0	4			$	4	D	T	d	t								
				0	1	0	1	5			%	5	E	U	e	u								
				0	1	1	0	6			&	6	F	V	f	v								
				0	1	1	1	7			'	7	G	W	g	w								
				1	0	0	0	8			(	8	H	X	h	x								
				1	0	0	1	9			)	9	I	Y	i	y								
				1	0	1	0	A			*	:	J	Z	j	z								
				1	0	1	1	B			+	;	K	[	k	{								
				1	1	0	0	C			(Comma) ,	<	L	\	l	|								
				1	1	0	1	D			(Minus) -	=	M	]	m	}								
				1	1	1	0	E			(Period) z	>	N	^	n	—								
				1	1	1	1	F			/	?	O	(Under line) _	o									

Basic
Note3.41

For the Basic model QCPU, character strings can be used only in the $MOV, STR, DSTR, VAL, DVAL, ESTR and EVAL instructions.

CHAPTER4 I/O NUMBER ASSIGNMENT

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

4.1 Relationship between Number of Slots and Main Base Unit

(1) Available main base units

Table4.1 indicates the main base units and/or slim type main base units available for each CPU module.

Table4.1 CPU modules and available main base units

CPU module	Available main base units
Q00JCPU	Not required*1
Q00CPU, Q01CPU	Q3□B, Q3□SB, Q3□RB, Q3□DB*2
Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	Q3□B, Q3□RB, Q3□DB*2
Q12PRHCPU, Q25PRHCPU	
Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	Q3□B, Q3□SB, Q3□RB, Q3□DB

* 1 : The Q00JCPU is a CPU module integrated with a power supply module and a main base unit. It requires no power supply module and main base unit.

* 2 : The high speed access function between Multiple CPUs cannot be used.

(2) Number of available slots

The number of available slots (modules) includes empty slots.
(If Slot 2 is set to "Empty, 0 point" as shown in Figure 4.1, it occupies one slot.)
The number of available slots (modules) varies depending on each main base unit.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection).
For the concepts of base unit assignment and I/O number assignment, refer to Section 4.4 and Section 4.6.

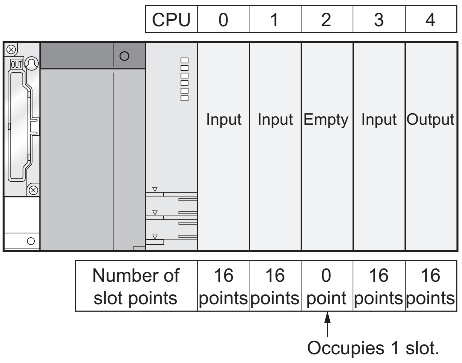


Figure 4.1 Number of modules including empty slot

4.2 Relationship between No. of Extension Stages and No. of Slots

CPU modules can be used in the following system configuration.

- Configuration of one main base unit and extension base units
- Configuration of one slim type main base unit

(1) Number of stages and number of slots

Table4.2 indicates the number of base units (stages) and the number of slots available for each CPU module.

Table4.2 Number of extension stages (extension base units) and numbers of slots

CPU module	Extension base unit (Number of extension stages)	Slim type main base unit[*1]	Number of available slots[*2] (Number of available modules)
Q00JCPU	○ (2 stages)	×	16 slots (modules)
Q00CPU, Q01CPU	○ (4 stages)	○	24 slots (modules)
Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	○ (7 stages)	○	64 slots (modules)
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU[*3]	○ (7 stages)	×	64 slots (modules)
Q12PRHCPU, Q25PRHCPU[*3]	○ (7 stages)	×	63 slots (modules)
Q02UCPU	○ (4 stages)	○	36 slots (modules)
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	○ (7 stages)	○	64 slots (modules)

○ : Available, × : Not Available

[*1] : Extension base units cannot be connected to the slim type main base unit.

[*2] : The number of available slots (modules) includes empty slots.
(If Slot 2 is set to "Empty, 0 point" as shown in Figure 4.2, it occupies one slot.)

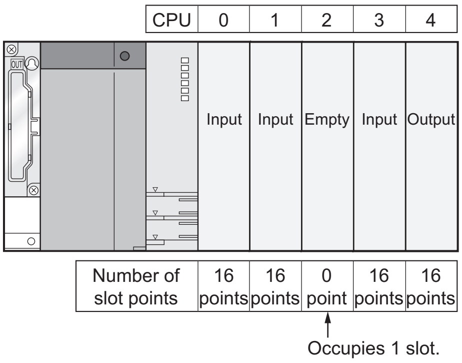


Figure 4.2 Number of modules including empty slot

[*3] : The Redundant CPU whose serial number (first five digits) is "09012" or later is targeted.
When using the Redundant CPU whose serial number (first five digits) is "09011" or earlier, the extension base unit cannot be connected to the main base unit.(☞ Appendix 4.4)
Therefore, the number of available slots will be 11.

(2) Precautions for the number of mounted modules

Mount modules within the range of the allowed number of slots.

Even if the total number of slots for a main base unit and extension base unit is greater than the number of available slots (for example, six 12-slot base units are used), no error will occur when modules are mounted in slots whose number is within a valid range.

An error ("SP. UNIT LAY ERR.") will occur if any module is mounted after the slot exceeding the number of available slots.

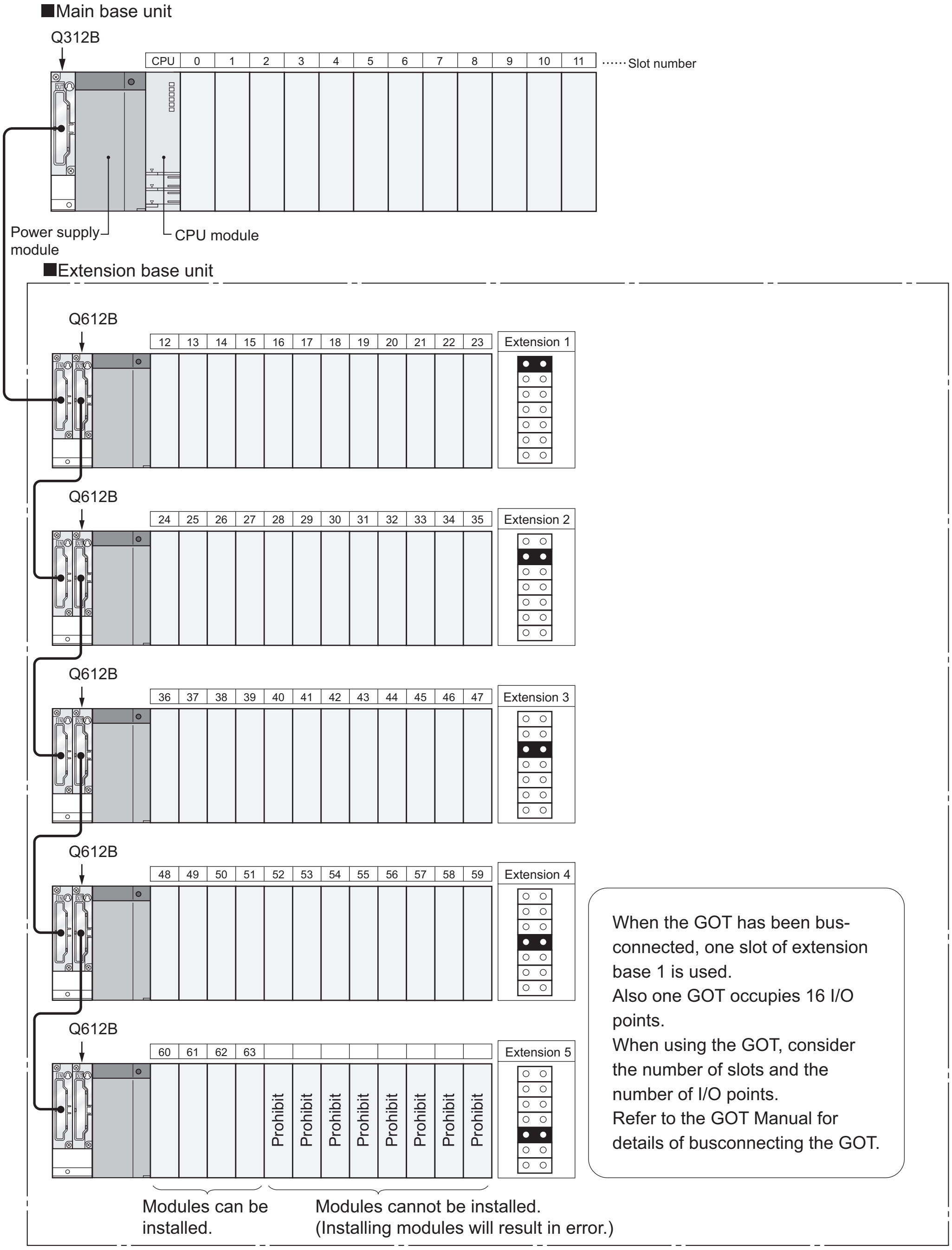


Figure 4.3 Configuration example for High Performance model QCPU

4.3 Installing Extension Base Units and Setting the Number of Stages

The extension base units shown in Table4.3 are available.

Table4.3 Available extension base units

CPU module	Available extension base unit
Q00JCPU	Q5□B,Q6□B,Q6□RB
Q00CPU, Q01CPU	
Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	Q5□B,Q6□B,Q6□RB, QA1S6□B, QA65B, QA6ADP+A5□B/A6□B
Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	Q5□B,Q6□B,Q6□RB
Q12PRHCPU, Q25PRHCPU	Q6□WRB (fixed to first extension stage), Q6□RB (fixed to second extension stage or later stages)
Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	Q5□B,Q6□B,Q6□RB

☒**POINT**

Extension base units cannot be connected to the slim type main base unit.

(1) Extension stage number setting and setting order

When using extension base units for extension, set the extension stage numbers with the stage number setting connectors on the extension base units.

Set the extension stage numbers in order of connection, starting from the extension base unit connected to the main base unit.

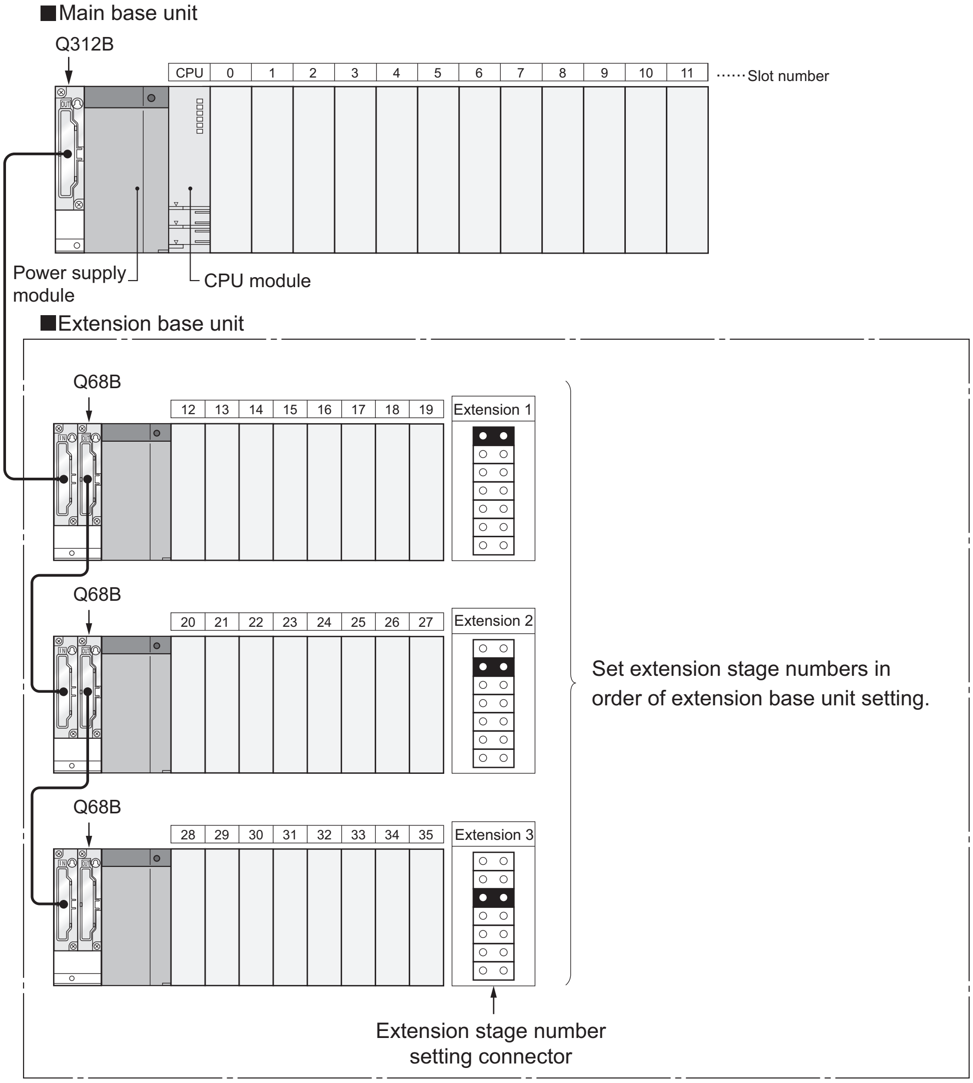


Figure 4.4 Extension stage number setting and setting order

(2) Precautions for extension stage number setting

(a) Extension stage number setting order

Set the extension stage numbers consecutively.

If any extension stage number is skipped in the auto mode(☞ Section 4.4(1))of base unit assignment, 0 slot is set to the skipped stage and the number of empty slots does not increase.

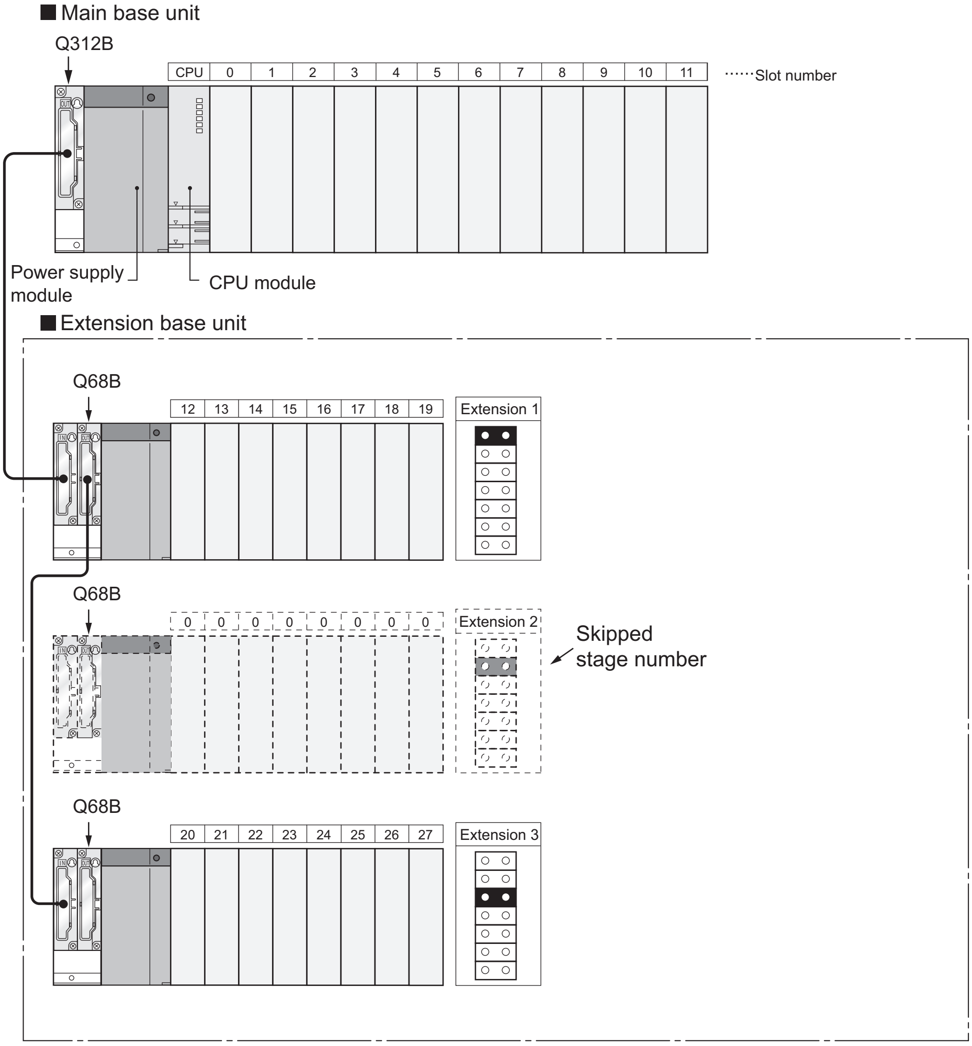


Figure 4.5 Number of slots when extension stage number is skipped

(b) When the same extension stage number is set

The same extension stage number cannot be set to multiple extension base units.

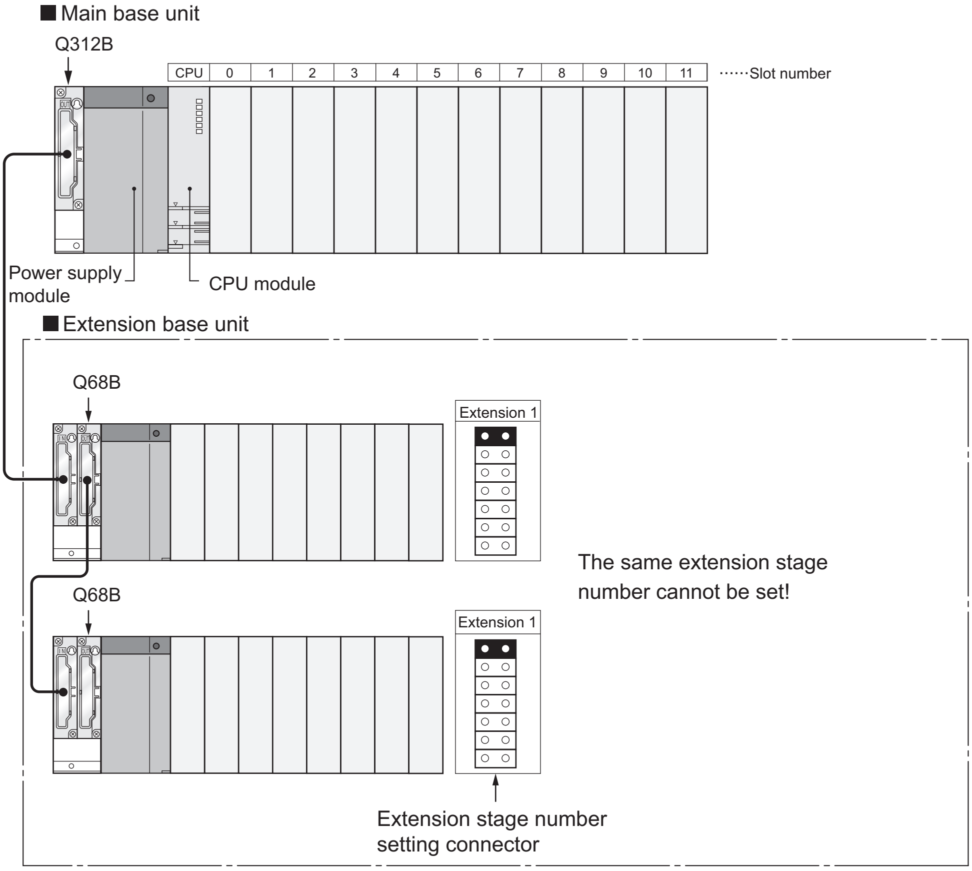


Figure 4.6 When the same extension stage number is set

(c) When connector pins are inserted into two or more positions or no connector pin is inserted

Extension base units cannot be used with connector pins inserted in two or more positions.

Also, they cannot be used without connector pins being inserted.

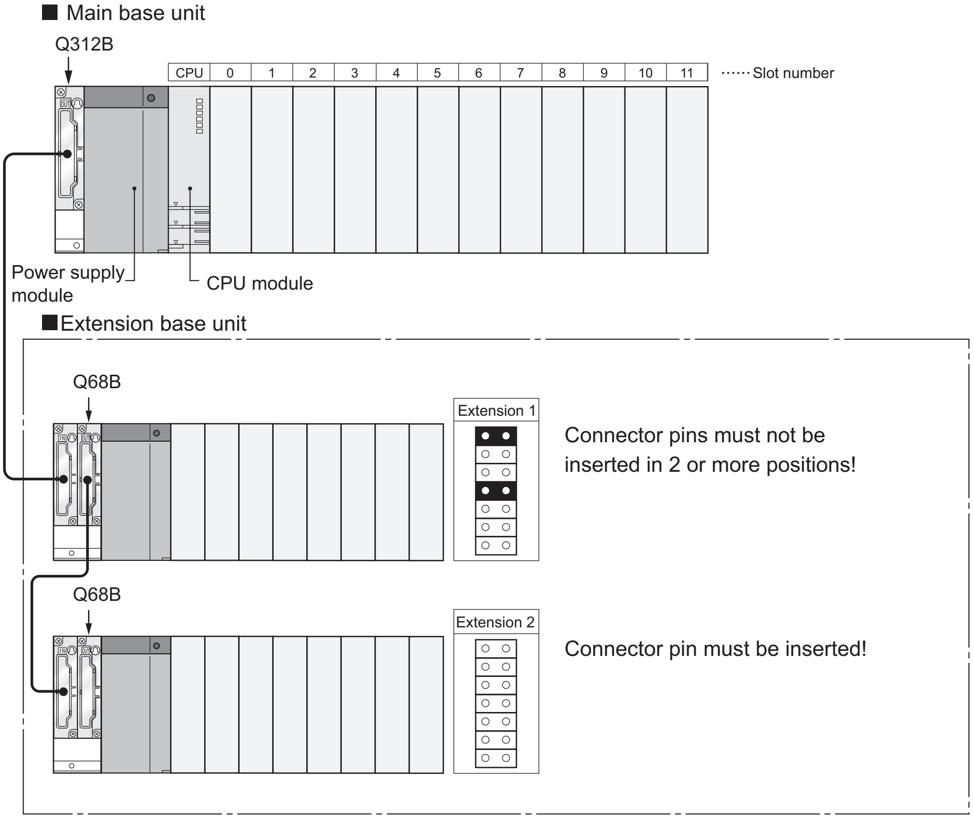
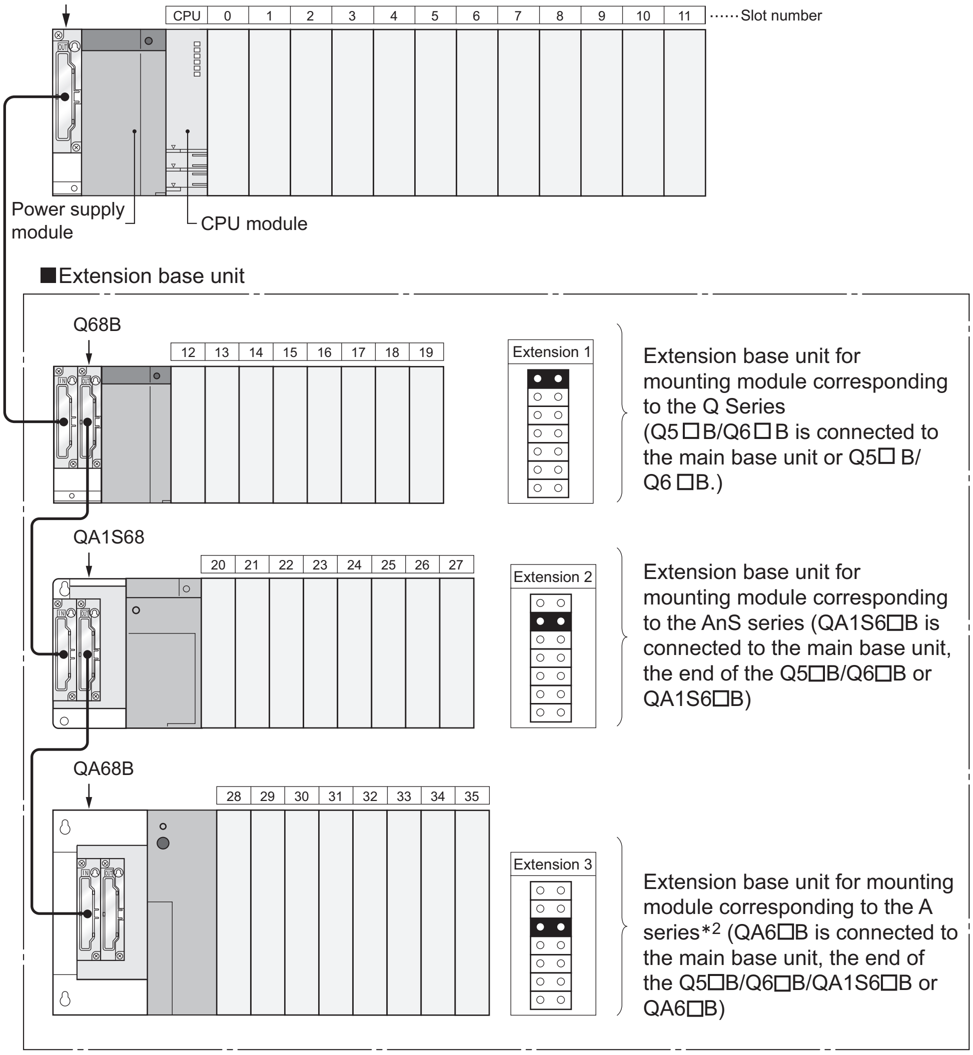


Figure 4.7 When connector pins are inserted into two or more positions or no connector pin is inserted

Basic Note4.1 Process Note4.1 Redundant Note4.1 Universal Note4.1

(d) Extension position of using AnS/A series compatible extension base unit (QA1S6□B, QA6□B, QA6ADP+A5□B/A6□B)Note4.1

When using the Q5□B, Q6□B, QA1S6□B, QA6□B, and QA6ADP+A5□B/A6□B [*1] are used together, mount the Q5□B/Q6□B, QA1S6□B, QA6□B, and QA6ADP+A5□B/A6□B in order from the nearest position of the main base unit.



Extension base unit for mounting module corresponding to the Q Series (Q5□B/Q6□B is connected to the main base unit or Q5□ B/Q6 □B.)

Extension base unit for mounting module corresponding to the AnS series (QA1S6□B is connected to the main base unit, the end of the Q5□B/Q6□B or QA1S6□B)

Extension base unit for mounting module corresponding to the A series*2 (QA6□B is connected to the main base unit, the end of the Q5□B/Q6□B/QA1S6□B or QA6□B)

Figure 4.8 Layout for use of AnS/A series compatible extension base units

* 1 : When using the QA1S6□B as the extension base unit, the QA6ADP+A5□B/A6□B cannot be connected.

* 2 : When using the QA6ADP+A5□B/A6□B, connect it after connecting the QA6□B.

Basic Note4.1 Process Note4.1 Redundant Note4.1 Universal Note4.1

The Basic model QCPU, Process CPU, Redundant CPU, and Universal model QCPU cannot use the AnS/A series compatible extension base unit (QA1S6□B, QA6□B and QA6ADP+A5□B/A6□B).

4.4 Base Unit Assignment (Base Mode)

Basic Process Redundant
Note4.2 Note4.2 Note4.2

The CPU module has the Auto mode and Detail mode, in which the number of modules mounted on the main base unit, slim type main base unit[Note4.2],and/or extension base unit can be assigned.
Set the base mode in the I/O assignment setting of the PLC parameter dialog box.
(☞ (3) in this section)

(1) Auto mode

In Auto mode, the slot numbers are assigned to the main and extension base units according to the number of slots than can be occupied.
The I/O numbers are assigned according to the modules which can be mounted to the current base unit.

Because 8-slot setting is fixed to the AnS/A series main base unit or extension base unit, a 3-slot or 5-slot base unit occupies 8 slots.
Since the Q series CPU module occupies only the mountable slots of the base unit, only 3 slots are occupied when a 3-slot base unit is used.

Basic Process Redundant
Note4.2 Note4.2 Note4.2

For the Q00JCPU, Process CPU and Redundant CPU, slim type main base units are not available.

(a) For 3-slot base unit: 3 slots are occupied

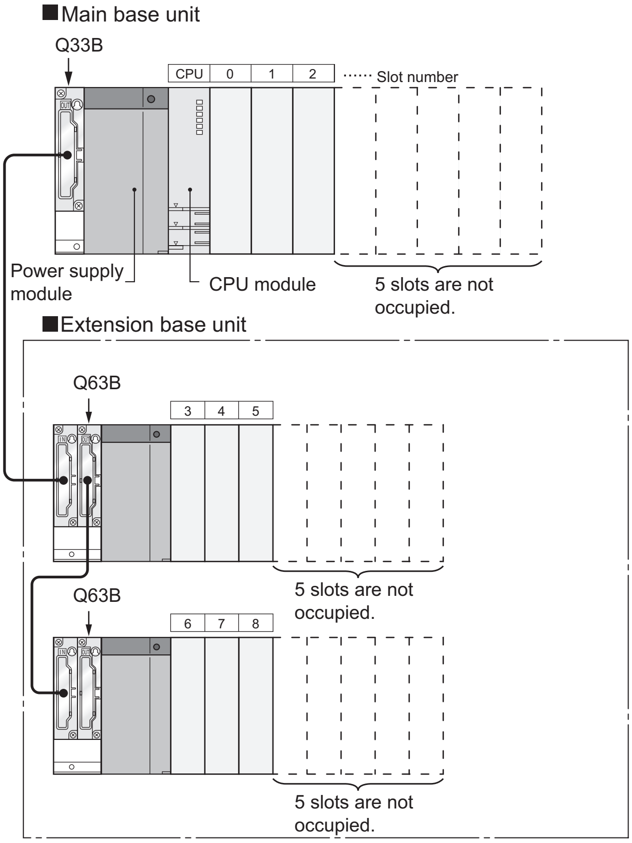


Figure 4.9 For 3-slot base units

(b) For 5-slot base unit/Q00JCPU: 5 slots are occupied

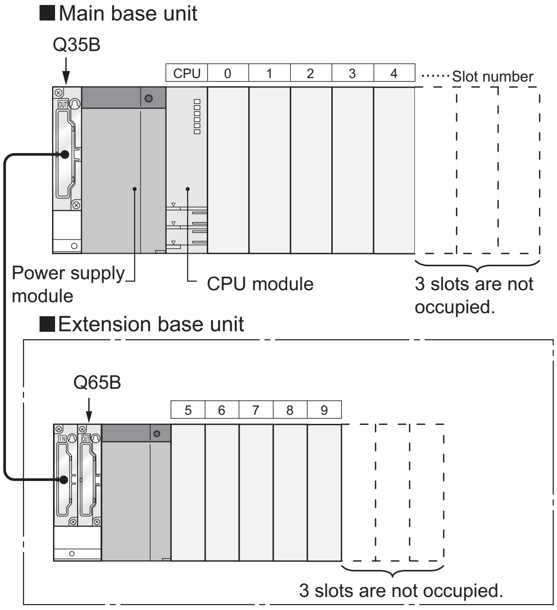


Figure 4.10 For 5-slot base units

(c) For 8-slot base unit: 8 slots are occupied

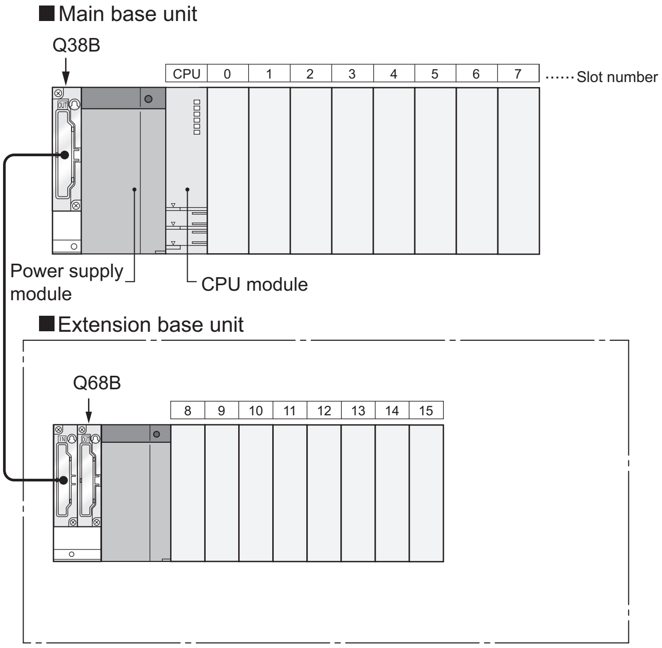


Figure 4.11 For 8-slot base units

(d) For 12-slot base unit: 12 slots are occupied

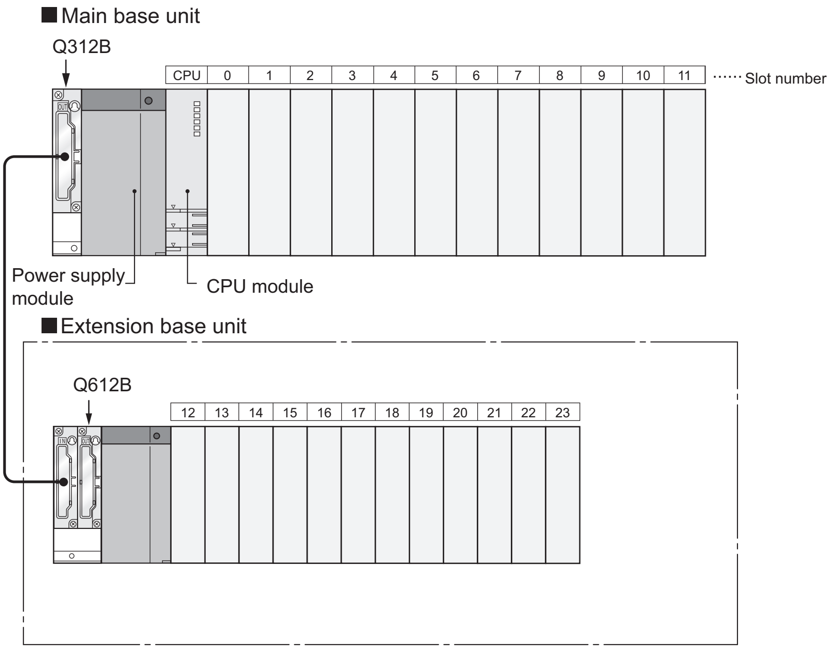


Figure 4.12 For 12-slot base units

(2) Detail mode

In Detail mode, set the number of mountable modules to each base unit in the I/O assignment setting of the PLC parameter dialog box.

(a) Applications

This mode can be used when matching the number of mountable modules with the number of modules (fixed to 8 slots) occupied by the AnS/A series base unit, for example.
This mode is also used to make the slots after an empty one unrecognized since an empty slot occupies 1 slot even if it is set to 0 points in the I/O assignment.

(b) Cautions on setting the number of slots

The number of slots can be set regardless of the number of the module being used.

☒POINT

Set the number of slots for all used base units.
If the number of slot is not set for all the base units, I/O assignment may not work correctly.

The followings result if the preset number of slots differs from that of the installed base units.

1) When the preset number of slots is greater than the number of actually used slots

The preset number of slots is occupied.
The slots of numbers after the used ones are assigned as empty slots.
For example, when 8 slots are designated for a 5-slot base unit, 3 slots will be empty slots.

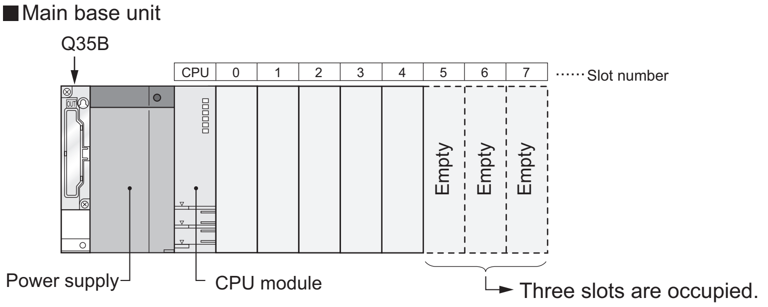


Figure 4.13 When the preset number of slots is greater than the number of actually used slots

The number of points for the empty slots is the one designated at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box. (Default value is 16 points.)

2) When the preset number of slots is less than the number of actually used slots

The slots other than those designated are disabled.
For example, when 8 slots are designated for a 12-slot base unit, the 4 slots on the right of the base unit are disabled.
(If a module is mounted to the prohibited slot, an error [SP. UNIT LAY ERR.] occurs.)

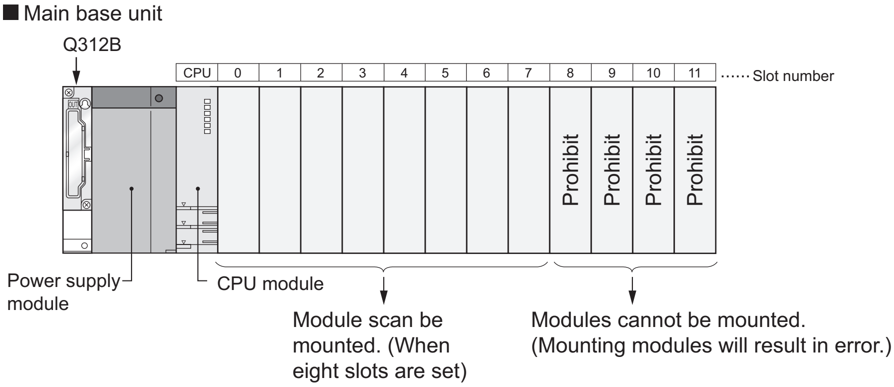
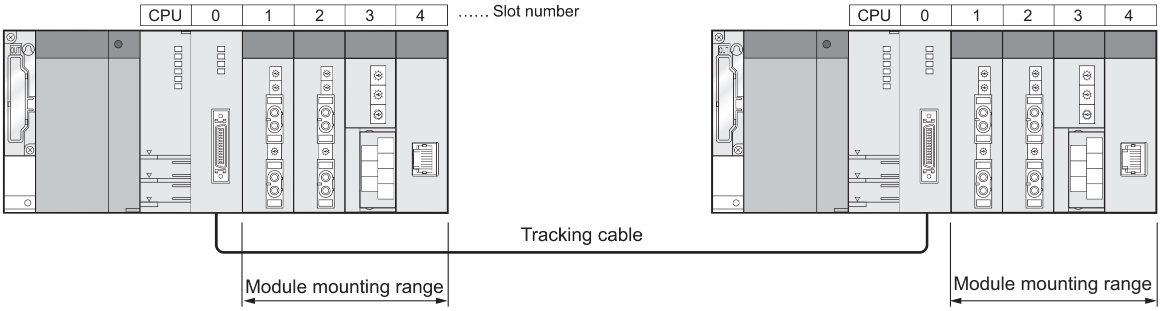

■ Main base unit
Q312B
CPU
0
1
2
3
4
5
6
7
8
9
10
11
…… Slot number
Prohibit
Prohibit
Prohibit
Prohibit
Power supply module
CPU module
Module scan be mounted. (When eight slots are set)
Modules cannot be mounted. (Mounting modules will result in error.)

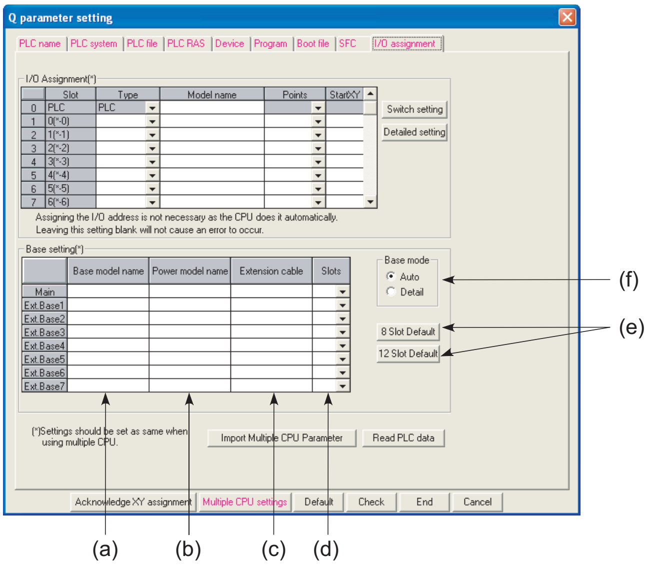
Figure 4.14 When the preset number of slots is less than the number of actually used slots

☒POINT

One Redundant CPU occupies 2 slots.
Hence, the number of mountable modules is (the preset number of slots - 1).
For example, when a 5-slot main base unit is used, the number of mountable slots is 4.



(3) Setting screen and setting items for Base mode of GX Developer



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 4.15 I/O assignment

(a) Base model name

Set the mounted base unit model name within 16 characters.
CPU module does not use the designated model name. (It is used as a user's memo or for parameter printing)

(b) Power model name

Set the mounted power supply module model name within 16 characters.
CPU module does not use the designated model name. (It is used as a user's memo or for parameter printing)

(c) Extension cable

Set the used extension cable model name within 16 characters.
CPU module does not use the designated model name. (It is used as a user's memo or for parameter printing)

(d) Slots (Used with CPU module)

Select the number of points for the slot of the base unit being used from the followings:

- 2(2slots)
- 3(3slots)
- 5(5slots)
- 8(8slots)
- 10(10slots)
- 12(12slots)

(e) 8slot Default/12slot Default (Used with CPU module)

Select either option to designate the number of slots for all base units to the same number.

(f) Auto/Detail

Select whether base unit assignment will be made in Auto mode or Detail mode.

4.5 Definition of I/O Number

I/O numbers indicate the addresses used in a sequence program to input or output ON/OFF data between the CPU module and other modules.

(1) Input and output of ON/OFF data

Input (X) is used to input ON/OFF data to the CPU module, and output (Y) is used to output ON/OFF data from the CPU module.

(2) I/O number expressions

I/O numbers are expressed as hexadecimal.
When using 16-point I/O modules, I/O numbers are consecutively assigned to the slots having□□0 to □□F, 16 points, as shown in Figure 4.16.

The module mounted in the base unit assigns the following:

- For the input module, "X" is assigned at the beginning of the I/O number.
- For the output module, "Y" is assigned at the beginning of the I/O number.

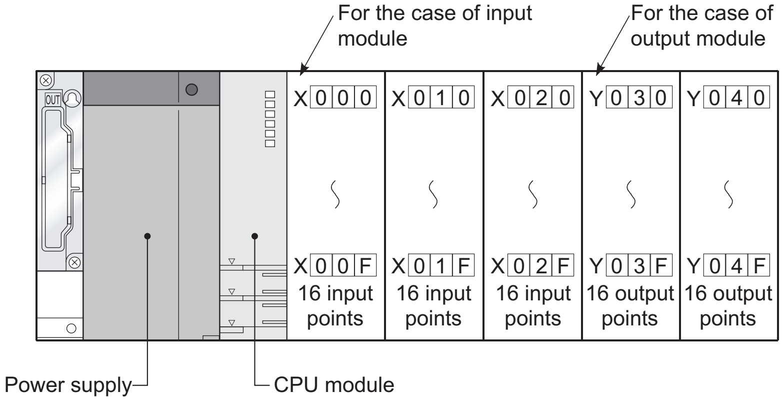
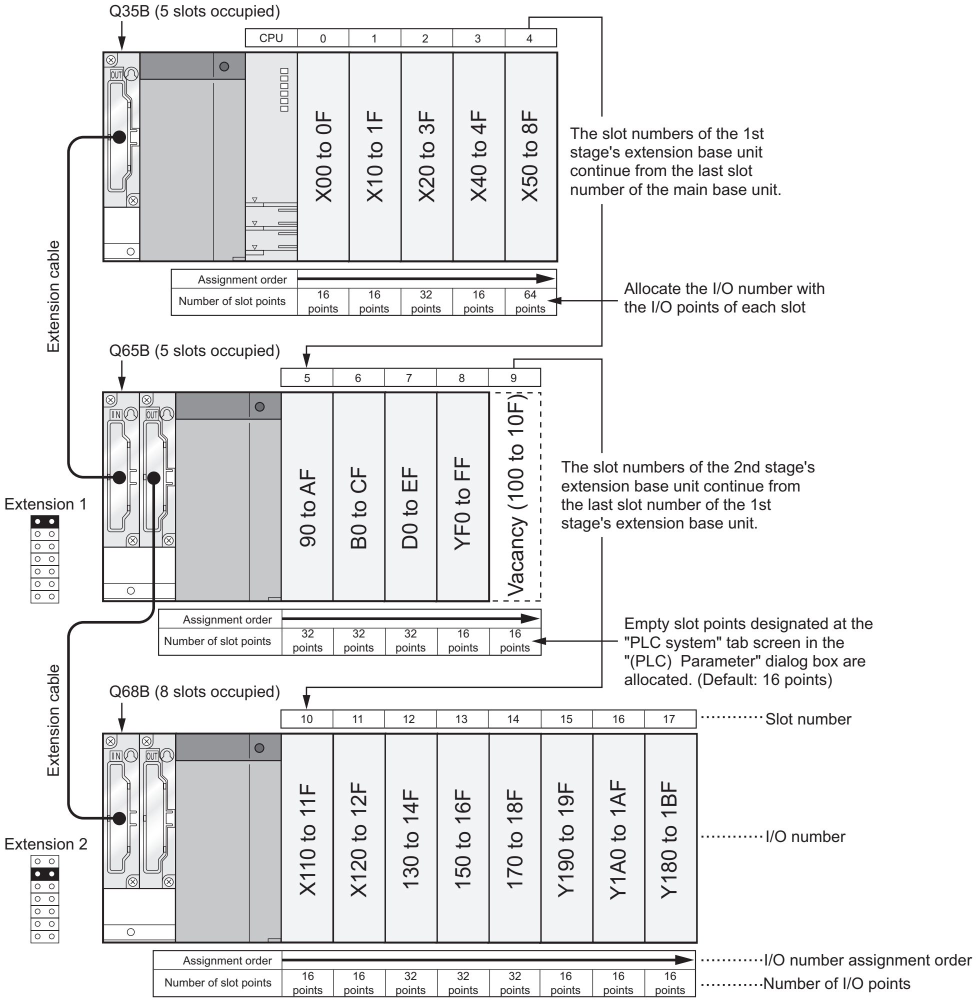


Figure 4.16 Input and output numbers

4.6 Concept of I/O Number Assignment

4.6.1 I/O numbers of base unit

The CPU module assigns I/O numbers at power-on or reset.
Figure 4.17 shows the example of the I/O number assignment when the base unit is set in Auto mode without I/O assignment.



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

Figure 4.17 I/O number assignment example

The above example assumes that the I/O points of the intelligent function module are 32 points.
The number of I/O points changes depending on the intelligent function module.
Assign the I/O numbers after confirming the number of I/O points in the manual of the used intelligent function module.

☒POINT

The CPU module can control even if I/O assignment has not been made by GX Developer.

To assign I/O numbers, follow the items below:

(1) Number of slots of base units

The number of slots of base units is set in Base mode.(☞ Section 4.4)

(a) In Auto mode

The number of slots is determined as the available number of modules mounted to each base unit.

5 slots are assigned to a 5-slot base unit, and 12 slots are assigned to a 12-slot base unit.

(b) In Detail mode

The number of slots set in the I/O assignment setting of the PLC parameter dialog box is used.

(2) Order of I/O number assignment

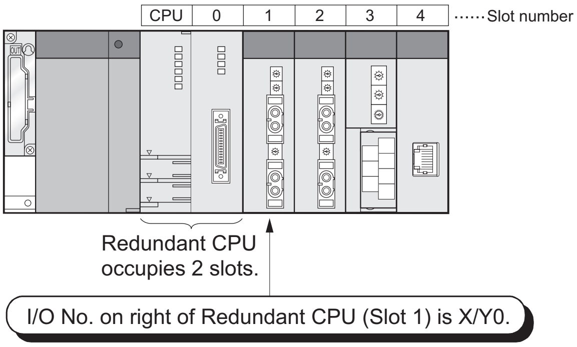
Basic Note4.3 Process Note4.3 Redundant Note4.3

The I/O numbers are assigned to the modules from left to right consecutively, starting from 0H assigned to the module on the right of the CPU module in the main base unit.Note4.3

☒POINT

One Redundant CPU occupies 2 slots.
Hence, the I/O number of Slot 1 (on the right of the Redundant CPU) is X/Y0.

■ Main base unit (5 slots)


CPU
0
1
2
3
4
Slot number
Redundant CPU occupies 2 slots.
I/O No. on right of Redundant CPU (Slot 1) is X/Y0.

Basic Note4.3 Process Note4.3 Redundant Note4.3

For the Q00JCPU, Process CPU and Redundant CPU, slim type main base units are not available.

(3) Order of I/O number assignment for extension base units

The I/O numbers for extension base units continue from the last number of the I/O number of the main base unit.

The I/O numbers are assigned to the extension base units from left (I/O 0) to right consecutively as shown in Figure 4.18, in the order in which the setting connectors of the extension base unit are set.

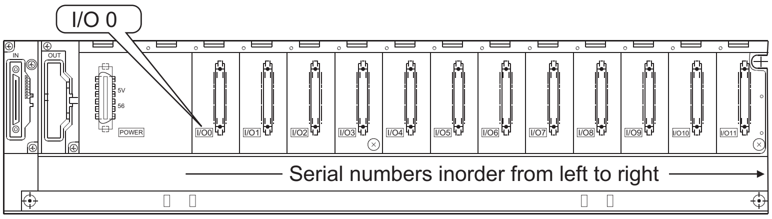


Figure 4.18 I/O number assignment order

(4) I/O numbers of each slot

Each slot of base units occupies the points of I/O numbers of the mounted I/O modules or intelligent function modules .

When 32-point input module is mounted on the right of CPU module, X0 to X1F are assigned as I/O numbers.

(5) I/O numbers of empty slots

If the base unit has empty slots mounted with no I/O modules or no intelligent function modules (special function modules) are mounted, the points designated at the "I/O assigment" tab screen in the "(PLC) Parameter" dialog box are assigned to the empty slots. (Default value is 16 points.)

☒POINT

When the assignment of base units is conducted in Auto mode, the number of empty extension stages is not assured even if the extension stage is skipped at the stage number setting connector of the base unit.(☞ Section 4.3(2)(a))

To reserve empty extension stages for future expansion, use the PLC parameter to set the base unit.

4.6.2 I/O numbers of remote station

Basic
Note4.4
High Performance
Note4.5

It is possible to allocate CPU module device input (X) and output (Y) to remote station I/O modules and intelligent function modules and control the modules in the MELSECNET/H remote network[Note4.4], [Note4.5], the CC-Link and other remote I/O systems.
Input (X) and output (Y) can be used for the refresh target (CPU module side device) of the MELSECNET/H link I/O (LX/LY).

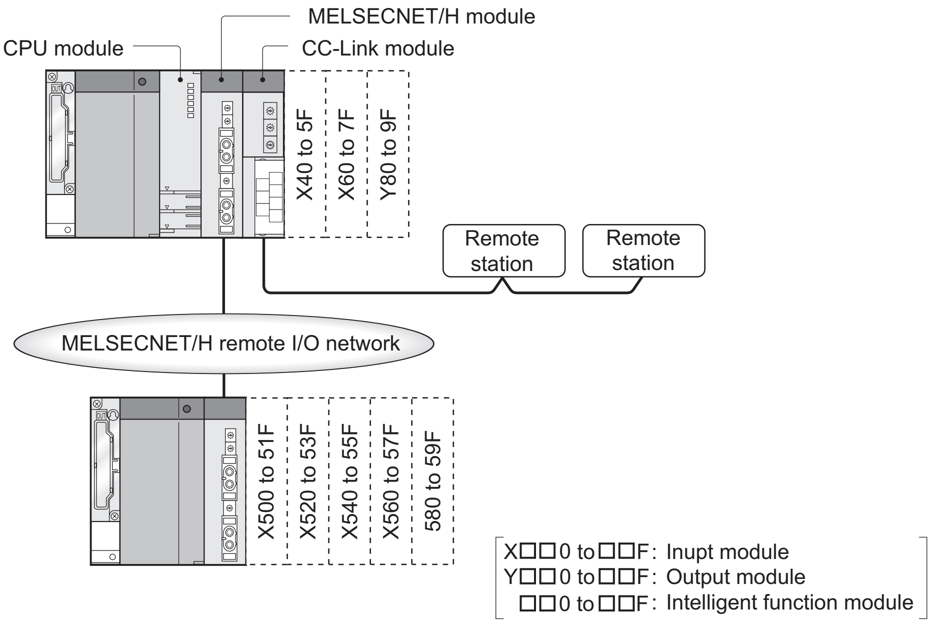


Figure 4.19 Remote station I/O numbers

(1) CPU module I/O numbers available at remote stations

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

Example: When X/Y0 to X/Y3FF (1024 points) are used by the CPU module side I/O modules and intelligent function modules, X/Y400 and later can be used at the remote stations.

Basic
Note4.4

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

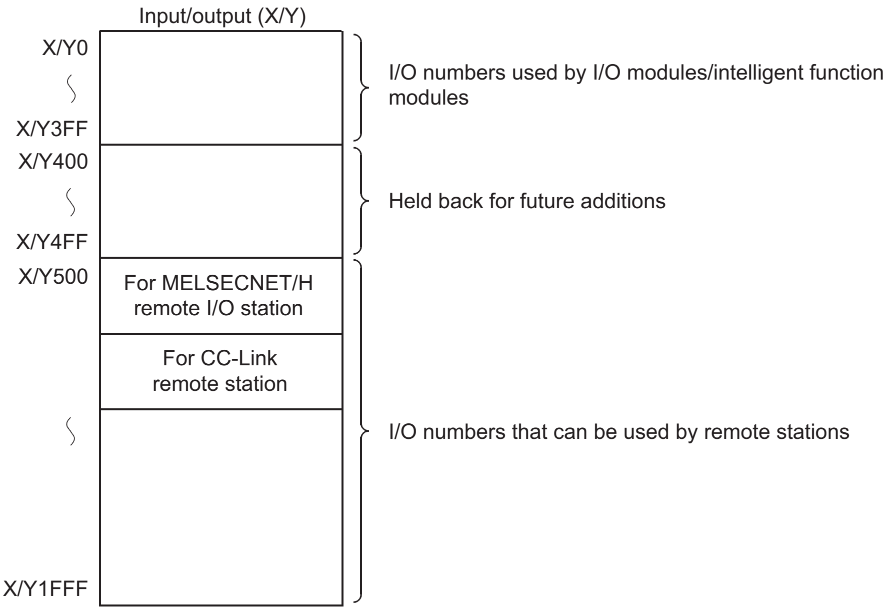
High Performance
Note4.5

When using the MELSECNET/H remote I/O network with the High Performance model QCPU, check the versions of the CPU module and GX Developer.(☞ Appendix 4.2)

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

(a) Setting in consideration of future extension

When using the input (X) and output (Y) of the CPU module for the I/O numbers of remote stations, set them in consideration of extension of I/O modules and/or intelligent function modules on the CPU module side.(☞ Figure 4.20)


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

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

Figure 4.20 Remote station I/O number assignment

(b) When MELSECNET/H and CC-Link are used

Do not assign the same I/O numbers to the refresh target (CPU module side device) of MELSECNET/H and to the CC-Link remote I/O system.

(c) Number of CPU module I/O device points

The number of I/O device points changes depending on the CPU module.
Refer to Chapter 2 for the number of CPU module I/O device points.

☒POINT

1. When network parameter setting has not been made in the CC-Link system, the following points are assigned to the CC-Link master/local modules of lower numbers.
   - Basic model QCPU
     X/Y400 to 7FF (1024 points)
   - High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU
     X/Y1000 to 17FF (2048 points)
2. There is restriction on the order of allocating I/O numbers for MELSECNET/H remote I/O networks, CC-Link or other networks.
3. Space can be made between the I/O area for the MELSECNET/H remote I/O station and the I/O area for the CC-Link remote I/O station.

4.7 I/O Assignment Using GX Developer

This section describes the I/O assignment using GX Developer.

4.7.1 Purpose of I/O assignment using GX Developer

The I/O assignment is set using GX Developer in the following cases.

(1) Reserving points for the change in the future

The number of points can be reserved in advance so that the I/O numbers do not need to be modified when the current module is changed to the one with the different number of occupied I/O points in the future.
For example, a 32-point input module can be assigned for the slot where a 16-point input module is actually mounted.

(2) Preventing I/O numbers from changing when converting modules

You can avoid the change in the I/O numbers when an I/O module other than 16-point module or intelligent function module (special function module) is removed due to a malfunction.

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

When the designed program's I/O numbers are different from the actual system I/O numbers, each module's I/O number of base units can be set to program-I/O number.

(4) Setting the input response time (I/O response time)

The input response times of input modules, I/O composite modules and interrupt modules can be changed according to the system.(☞ Section 6.7)

(5) Error-time output mode setting

The error-time output mode can be set to output modules, I/O composite modules, intelligent function modules and interrupt modules.(☞ Section 6.8)

(6) Setting the switch of intelligent function modules and interrupt modules

The switches of an intelligent function modules and interrupt modules can be set.(☞ Section 6.10)

(7) Setting outputs during CPU module error

The output status (hold/clear) of the output module and/or intelligent function module when the CPU module stops operation due to a stop error can be set.(☞ Section 6.8)

(8) Setting CPU module operation during a hardware error of intelligent function modules

The operation status (continue/stop) of the CPU module when a hardware fault occurs in the intelligent function module can be set.(☞ Section 6.9)

☒POINT

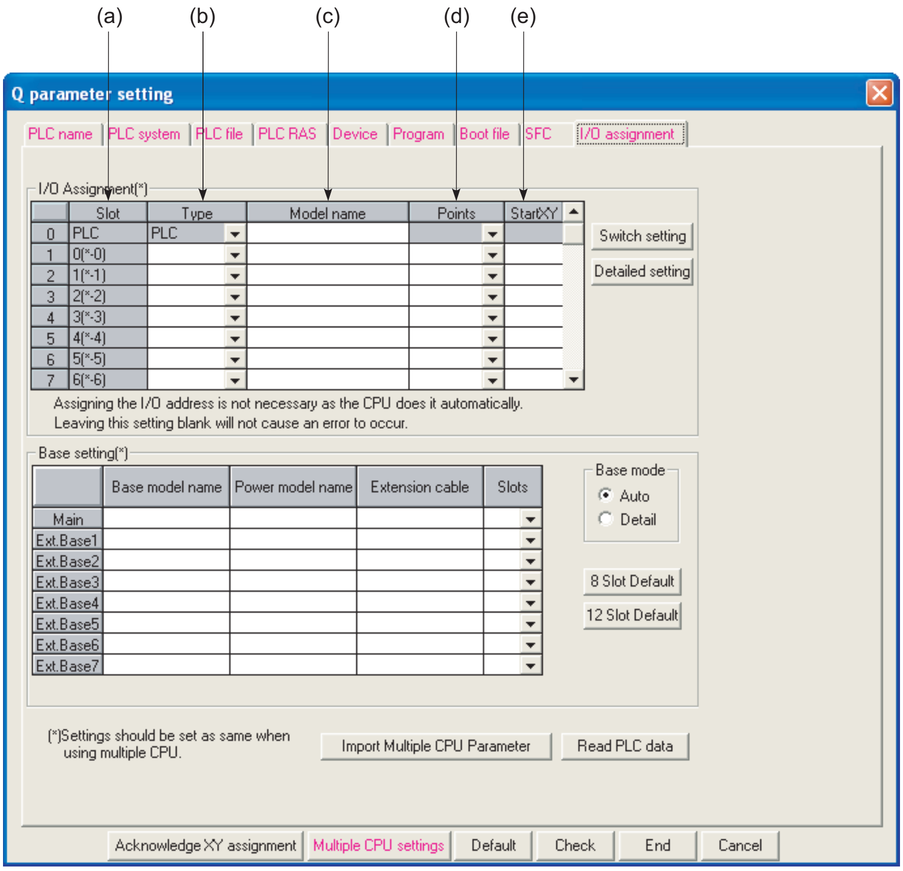
1. The I/O assignment setting becomes valid when the PLC is powered OFF and then ON or the CPU module is reset.
2. The I/O assignment setting is necessary for changing the response time of the input modules and the switch of intelligent function modules.
3. If any of the I/O modules other than the 16-point modules fails without I/O assignment setting being made using GX Developer, the I/O numbers after that module may change, leading to a malfunction. Therefore, it is recommended to make I/O assignment setting using GX Developer.

4.7.2 Concept of I/O assignment using GX Developer

In I/O assignment, the "Type (module type)", "Points (I/O points)" and "Start XY" (starting I/O number) can be set for each slot of the base units.
For example, to change the number of occupied I/O points of the designated slot, only the number of occupied I/O points can be designated.
The items other than designated are set to the status where the base unit is installed.

(1) I/O assignment

The I/O assignment is conducted at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box.


(a)
(b)
(c)
(d)
(e)
Q parameter setting
PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment
Switch setting
Detailed setting
8 Slot Default
12 Slot Default
Import Multiple CPU Parameter
Read PLC data
Acknowledge XY assignment
Multiple CPU settings
Default
Check
End
Cancel

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 4.21 I/O assignment

(a) Slot

The slot No. and what number of the slot on what number of the stage are displayed for each slot.
When the base unit is set in Auto mode, "*" indicates the base unit stage number and the slot number counted from Slot 0 of the main base unit is displayed.

Basic Process Redundant
Note4.6 Note4.6 Note4.6
Universal
UD
Note4.6

(b) Type (For CPU module)

Select the type of module being mounted from the followings:

- Empty (Empty slot)
- Input (Input module)
- Hi Input (Q Series high speed module)*1
- Output (Output module)
- I/O Mix (I/O mixed module)
- Intelligent (Intelligent function module or AnS/A series compatible special function module)Note4.6
- Interrupt (Q Series interruption module)*2

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

* 1 : "Hi input" can be set using GX Developer Version 5 (products after SW5D5C-GPPW-E)
* 2 : "Interrupt" can be set using GX Developer Version 6 (products after SW6D5C-GPPW-E)

(c) Model name

Set the mounted module model name within 16 characters.
The specified model name is not used for the CPU module. (It is used as a user's memo.)

(d) Points (Used with CPU module)

To change the number of occupied I/O points of each slot, select it from the followings:

• 0(0 point)	• 16(16 points)	• 32(32 points)
• 48(48 points)	• 64(64 points)	• 128(128 points)
• 256(256 points)	• 512(512 points)*1	• 1024(1024 points)*1

If the number of occupied I/O points is not designated for a slot, the one of the actually mounted module is used.

* 1 : Not available for the Q00JCPU.
 (Because the I/O points of the Q00JCPU are 256.)

(e) Start XY (Used with CPU module)

When the I/O number of each slot is changed, you should designate the head I/O number according to the change.
If Start XY is not designated for a slot, the I/O number continuing from the last number of the currently designated slot is assigned.

Basic Process Redundant
Note4.6 Note4.6 Note4.6
Universal
UD
Note4.6

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the AnS/A series corresponding modules.

(2) Precautions for I/O assignment

(a) Slot status after I/O assignment

When I/O assignment setting has been made to a slot, that setting has precedence over the mounted module.

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

The I/O points for actually mounted modules are decreased.
For example, when a slot, to which a 32-point input module is installed, is set to a 16-point input module in I/O assignment setting, the second half 16 points of the 32-point input module becomes invalid.

2) When the preset number of points is less than the number of mounted intelligent function module points

An error (SP.UNIT LAY ERR.) occurs.

3) When the preset number of points is greater than the number of I/O points of modules actually mounted

The points in excess of the mounted module points are not used by the I/O module.

4) Mounted module and I/O assigned module type

The mounted module type and the set type in the I/O assignment setting must be the same.
If not, normal operation will not be performed.
For the intelligent function module, make sure that the numbers of I/O points are the same.
Table4.4 describes the operations performed when the mounted module type differs from the one in the I/O assignment setting.

Table4.4 List of operations performed when mounted module differs from I/O assignment

Mounted module	I/O assignment setting	Result
Input module Output module Input / Output conposite module	Intelli. / Interrupt	Error(SP.UNIT.LAY.ERR.)
Intelligent function module	Input / Hi. input Output / I/O mix	Error(SP.UNIT.LAY.ERR.)
Empty slot	Input / Hi. input Output / I/O mix Intelli. / Interrupt	Empty slot
All modules	Empty	Empty slot
Other combinations		Error does not occur but normal operation is not performed.

5) **Last I/O number**
In I/O assignment, set the last I/O number not to exceed the maximum value(☞CHAPTER 2)of the I/O points.
An error ("SP. UNIT LAY ERR.") will occur if the last I/O number exceeds the maximum value of the I/O points. (System monitor of GX Developer shows "***" as an I/O address.)

☒POINT

1. When the Q5□B/Q6□B is used together with the QA1S6□B, QA6□B and QA6ADP+A5□B/A6□B[*1] as the extension base unit, pay attention to the following points.
   - Mount the Q5□B/Q6□B, QA1S6□B, QA6□B, and QA6ADP+A5□B/A6□B in order from the nearest position of the main base unit.
   - Assign the I/O number of a module mounted on the base unit, putting each series in block and in order of "Q series → A series" or "A series → Q series".
     An error (“SP.UNIT LAY ERR.”) will occur when the I/O number of the module is not assigned in the above order.
2. The number of slots of the QA6ADP+A5□B/A6□B is always displayed in 8 slots, regardless of the number of slots of the extension base unit to be used in combination with the QA6ADP.

[*1] : When using the QA1S6□B as the extension base unit, the QA6ADP+A5□B/A6□B cannot be connected.

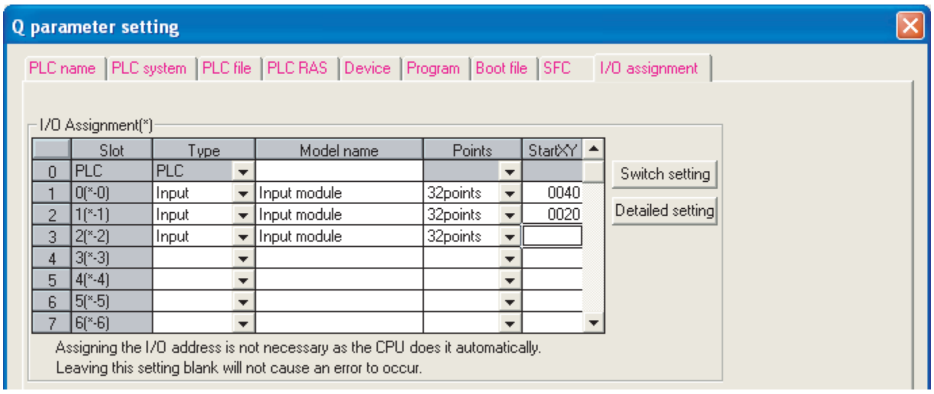
(b) Precautions for automatic start XY assignment by CPU module

When the start XY is not yet entered, the CPU module automatically assigns it. In the case of 1) or 2) below, therefore, the start XY setting of each slot may overlap the one assigned by the CPU module.

1) Settings of I/O numbers were exchanged in the start XY

2) There are slots with start XY setting and those without start XY setting (automatically assigned slot)

The following example Figure 4.22 shows overlapping start XY.



	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)	Input	Input 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 4.22 I/O assignment with overlapping start XY

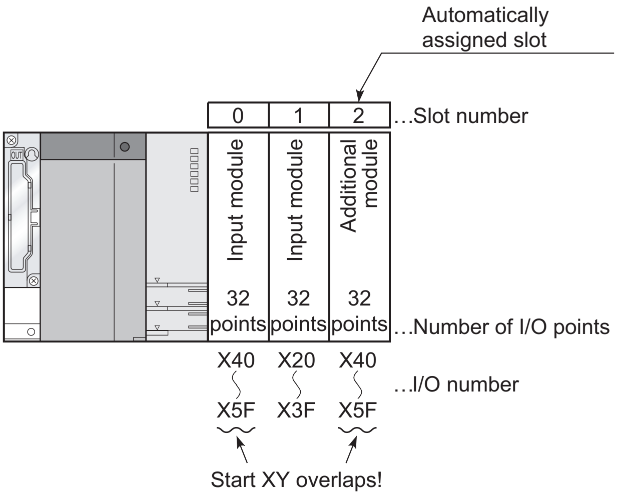


Figure 4.23 Start XY set by above I/O assignment

Be extremely careful not to overlap the start XY of each slot.
Overlapping start XY will result in an error ("SP. UNIT LAY ERR.").

4.8 Examples of I/O Number Assignment

The following example shows I/O number assignment made when I/O assignment setting is performed using GX Developer.

(1) When changing the number of points of an empty slot from 16 to 32 points:

Reserve 32 points so that the I/O numbers of Slot No. 4 and later do not change when a 32-point input module is mounted in the currently empty slot position (Slot No. 3) in the future. (The empty Slot No. 12 has 16 points, which do not change.)

(a) System configuration and I/O number assignment before the I/O assignment with GX Developer

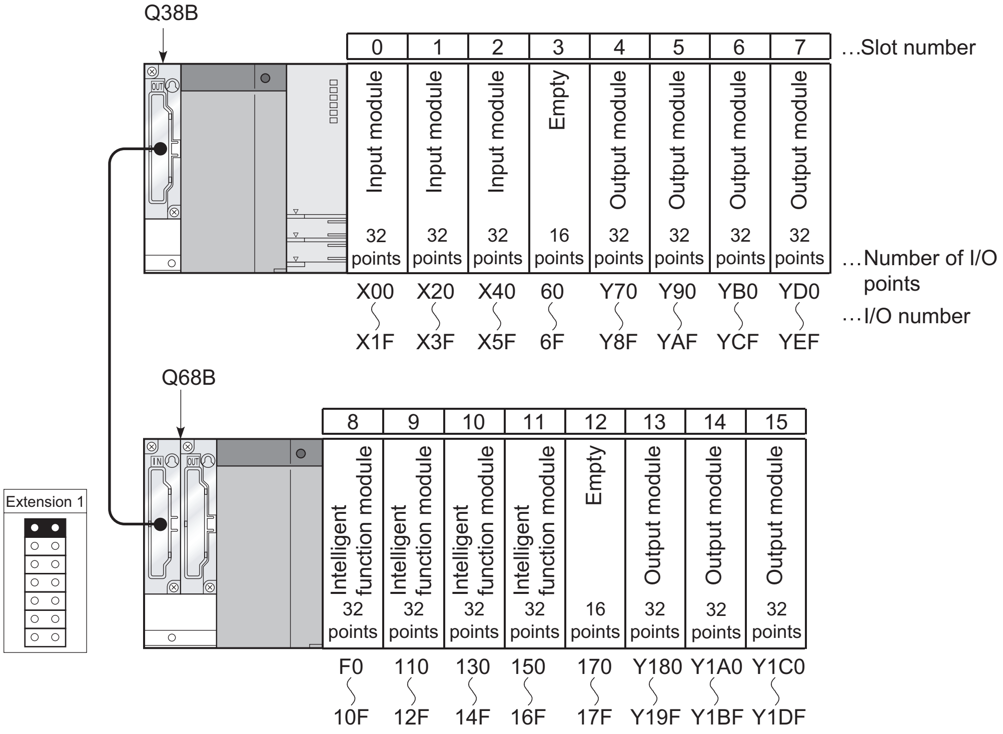
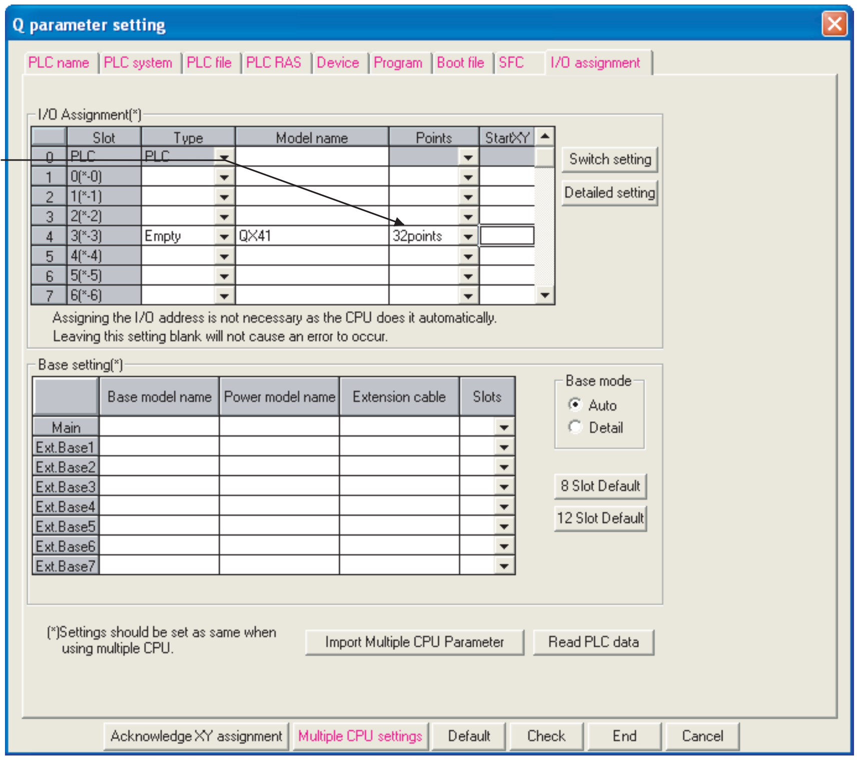

Q38B
0 1 2 3 4 5 6 7 …Slot number
Input module
Input module
Input module
Empty
Output module
Output module
Output module
Output module
32 points 32 points 32 points 16 points 32 points 32 points 32 points 32 points …Number of I/O points
X00 X20 X40 60 Y70 Y90 YB0 YD0 …I/O number
X1F X3F X5F 6F Y8F YAF YCF YEF
Q68B
Extension 1
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
32 points 32 points 32 points 32 points 16 points 32 points 32 points 32 points
F0 110 130 150 170 Y180 Y1A0 Y1C0
10F 12F 14F 16F 17F Y19F Y1BF Y1DF

Figure 4.24 I/O number assignment (Before changing points for empty slot)

(b) I/O assignment with GX Developer

Designate slot No. 3 to "32 points" at the "I/O assignment" tab screen of GX Developer.



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

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)				
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.

Switch setting
Detailed setting

Base setting(*)

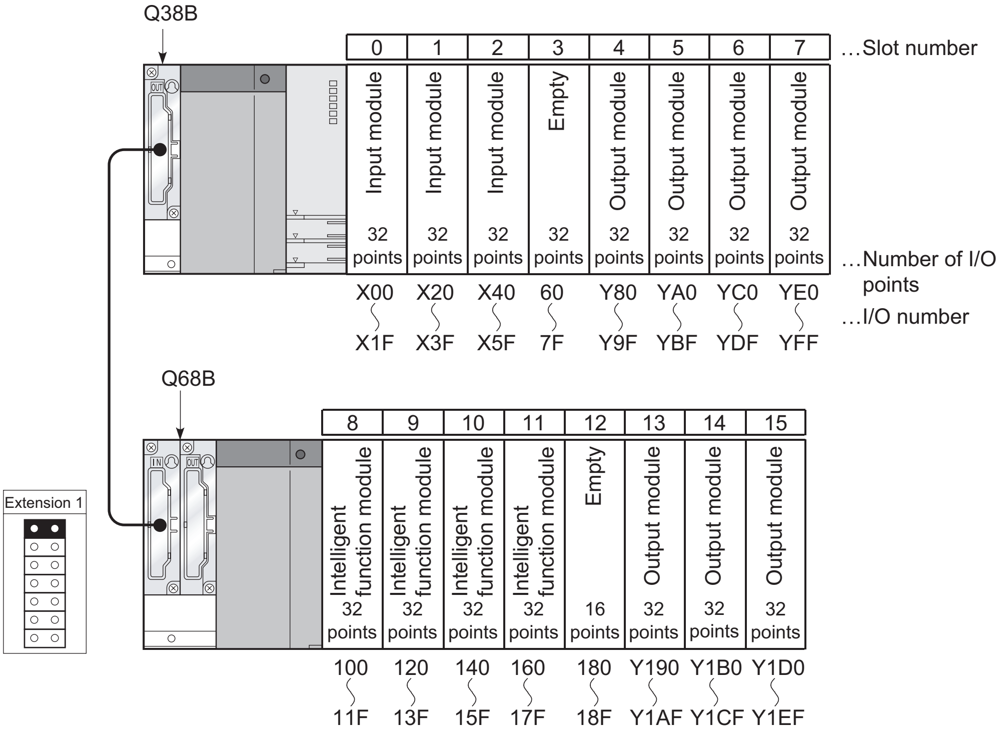
	Base model name	Power model name	Extension cable	Slots
Main				
Ext.Base1				
Ext.Base2				
Ext.Base3				
Ext.Base4				
Ext.Base5				
Ext.Base6				
Ext.Base7				

Base mode
● Auto
○ Detail

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

Figure 4.25 I/O assignment (When changing points of empty Slot 3)

(c) I/O number assignment after the I/O assignment with 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	32 points	32 points	32 points	32 points	32 points
I/O number	X00 ∫ X1F	X20 ∫ X3F	X40 ∫ X5F	60 ∫ 7F	Y80 ∫ Y9F	YA0 ∫ YBF	YC0 ∫ YDF	YE0 ∫ YFF

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	100 ∫ 11F	120 ∫ 13F	140 ∫ 15F	160 ∫ 17F	180 ∫ 18F	Y190 ∫ Y1AF	Y1B0 ∫ Y1CF	Y1D0 ∫ Y1EF

Figure 4.26 I/O numbers after I/O assignment (After changing points of empty slot)

(2) Changing the I/O number of slots

Change the I/O number of an empty slot (slot No. 3) to X200 through 21F so that the I/O numbers of slot No. 4 and later slots do not change when a 32-point input module is mounted to the empty slot (slot No. 3).

(a) System configuration and I/O number assignment before the I/O assignment with GX Developer

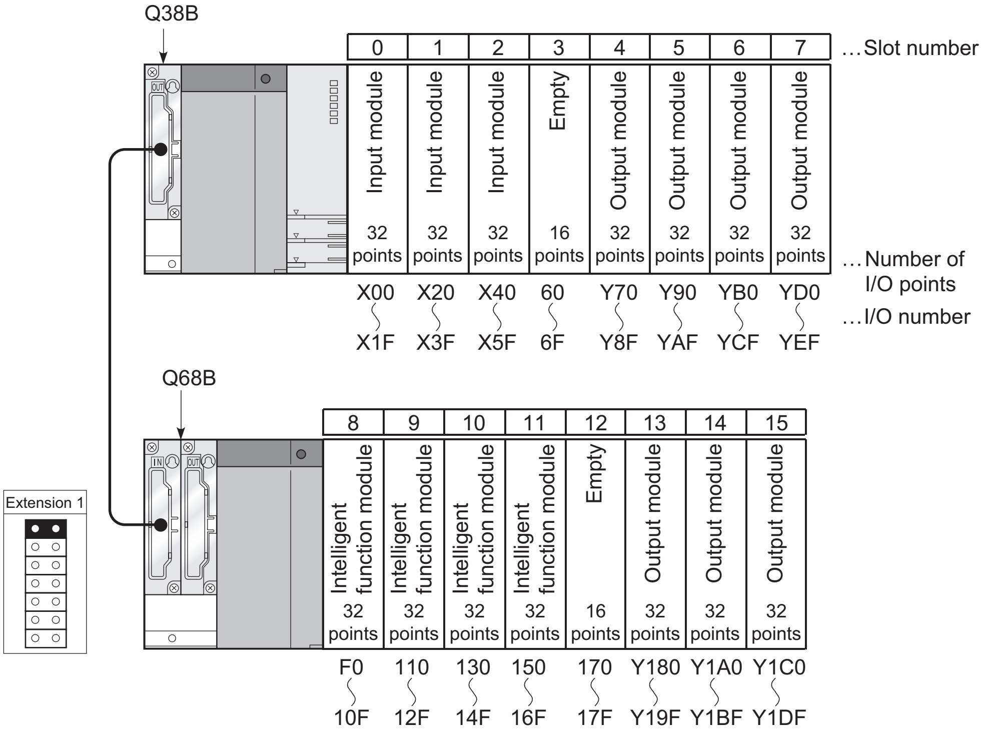
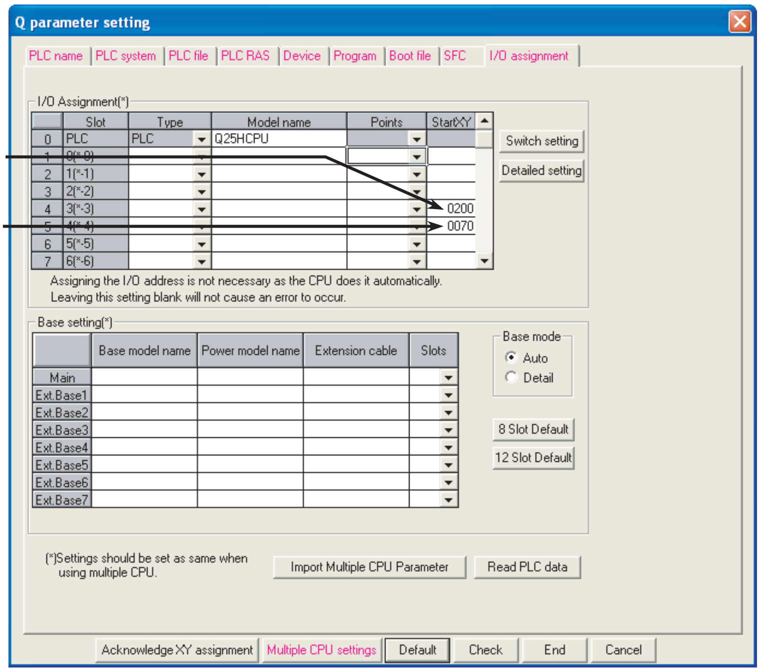


Figure 4.27 I/O number assignment (Before changing slot I/O numbers)

(b) I/O assignment with GX Developer

Designate the head I/O number of slot No. 3 to "200" and that of slot No. 4 to "70" at the "I/O assignment" tab screen of GX Developer.



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	Q25HCPU		
1	0(*-0)				
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				0200
5	4(*-4)				0070
6	5(*-5)				
7	6(*-6)				

Switch setting | Detailed setting

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.

Import Multiple CPU Parameter | Read PLC data

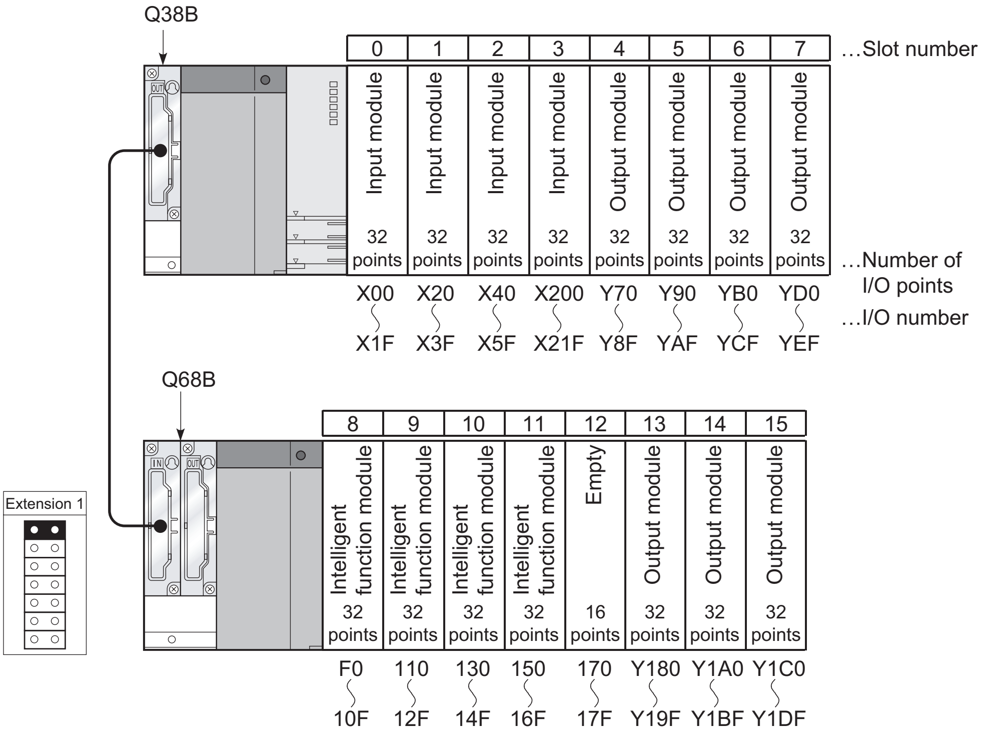
Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

"200" is designated as the head I/O number.

"70" is designated as the head I/O number. (When the head I/O number is not designated, the I/O number following the 3rd slot will be assigned.)

Figure 4.28 I/O assignment (When changing I/O numbers of Slot 3)

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



Slot number	0	1	2	3	4	5	6	7
	Input module	Input module	Input module	Input module	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 ∫ X1F	X20 ∫ X3F	X40 ∫ X5F	X200 ∫ X21F	Y70 ∫ Y8F	Y90 ∫ YAF	YB0 ∫ YCF	YD0 ∫ 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 ∫ 10F	110 ∫ 12F	130 ∫ 14F	150 ∫ 16F	170 ∫ 17F	Y180 ∫ Y19F	Y1A0 ∫ Y1BF	Y1C0 ∫ Y1DF

Figure 4.29 I/O number assignment (After changing slot I/O numbers)

4.9 Checking the I/O Numbers

System monitor of GX Developer allows the check of the mounted modules of CPU module and their I/O numbers.(☞ Section 6.20)

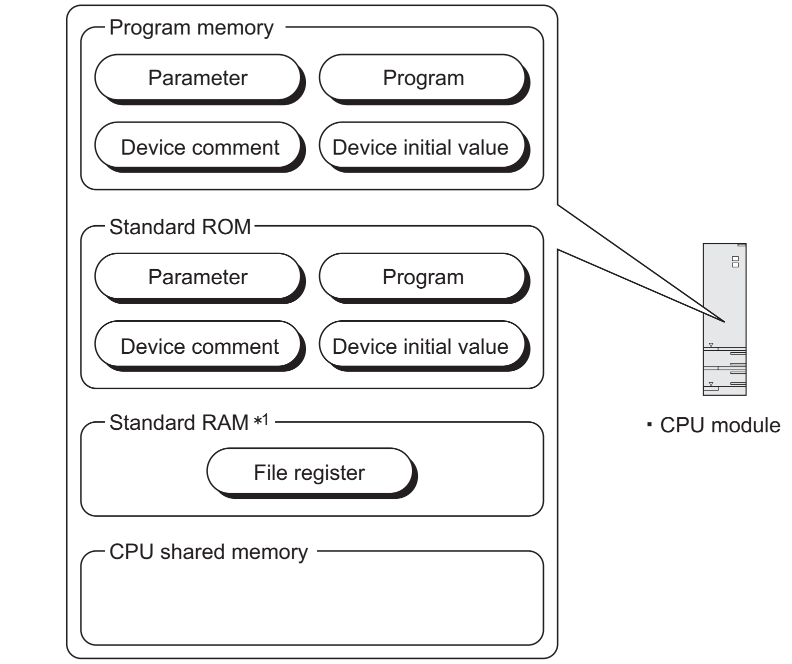
CHAPTER5 MEMORIES AND FILES USED IN CPU MODULE

5.1 Basic Model QCPU

5.1.1 Memory configuration and storable data

This section explains the memories used in the Basic model QCPU and the data can be stored in those memories.

(1) Memory configuration



* 1 : The Q00JCPU does not have the standard RAM.
File registers are unavailable.

Figure 5.1 Data handled by Basic model QCPU

(a) Program memory (☞ Section 5.1.2)
A memory for storing programs required for operations in the Basic model QCPU.

(b) Standard ROM (☞ Section 5.1.3)
A memory for performing the boot operation in the Basic model QCPU.

(c) Standard RAM (☞ Section 5.1.4)
A memory for file registers.

(d) CPU shared memory (☞ QCPU User's Manual (Multiple CPU System))
A memory for writing/reading data between CPU modules in a multiple CPU system.

(2) Data that can be stored into memories

Table5.1 indicates the data that can be stored into the program memory, standard RAM and standard ROM and the corresponding drive Nos.

Table5.1 Storable data and storage locations

	CPU module built-in memories			File name and extension
	Program memory	Standard RAM*5	Standard ROM	
Drive No.	0	3	4	
Parameter	◎	×	○	PARAM.QPA
Intelligent function module parameter	○	×	○	IPARAM.QPA
Sequence program	◎*4	×	○*1	MAIN.QPG
SFC program	◎*4	×	○*1	MAIN-SFC.QPG
File register	×	○*3	×	MAIN.QDR
Device comment	○*2	×	○*2	MAIN.QCD
Device initial value	○	×	○	MAIN.QDI
User setting system area*6	○	×	×	--

◎ : Necessary data, ○ : Storable data, × : Unstorable data

* 1 : To execute the program stored in the standard ROM, boot file setting to the program memory is required in the PLC parameter dialog box.

* 2 : Data can be written only from GX Developer.
Device comments cannot be used in sequence program instructions.

* 3 : Only one file of file register can be stored into the standard RAM.
Refer to CHAPTER 2 for the number of storable file register points.

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

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

* 6 : Set the area used by the system. (☞ Section 5.1.2(3)(b))

(3) Memory capacities and formatting necessities

Table5.2 indicates the memory capacity and formatting necessity of each memory.

Table5.2 Memory capacity and formatting necessity

	Q00JCPU	Q00CPU	Q01CPU	Formatting
Program memory	58k byte	94k byte	94k byte	Necessary*1
Standard ROM	58k byte	94k byte	94k byte	Unnecessary
Standard RAM	None	128k byte		Necessary*1

* 1 : Before use, be sure to format the memory using GX Developer.

* 2 : Make sure to format the standard RAM by GX Developer before using Q00CPU or Q01CPU of the function version A
Q00CPU and Q01CPU of the function version B don't need to be formatted.
(Formatting the standard RAM using GX Developer to Q00CPU or Q01CPU of the function B, an error code "4150H" will be displayed on the GX Developer

☒POINT

When data are written to each memory, the unit of storage capacity changes depending on the target CPU module and the memory area. (☞ Section 5.4.4)

5.1.2 Program memory

(1) Definition of program memory

The program memory is a memory for storing programs required for operations in the Basic model QCPU.

The programs stored in the standard ROM are booted (read) to the program memory to perform operations.

(2) Storable data

The program memory can store parameters, intelligent function module parameters, programs, device comments, device initial values and user setting system area data. (☞ Section 5.1.1(2))

☒POINT

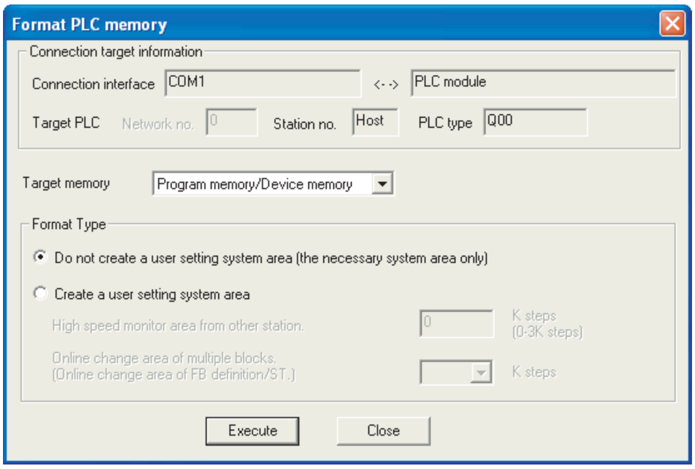
If the total volume of the data to be stored into the program memory exceeds its capacity, examine reducing the user setting system area data.

(3) Before using the program memory

Before using the program memory, be sure to format it by GX Developer.

(a) Formatting

To format the program memory, choose [Online] → [Format PLC memory] on GX Developer, and select "Program memory/Device memory" as the 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.) K steps

Figure 5.2 Program memory formatting

(b) **Create a user setting system area**

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

1) **Do not create a user setting system area**

The program memory is formatted without the user setting system area being created.

2) **Create a user setting system area**

The user setting system area is created during formatting.
There are the following user setting system areas (Table5.3).

Table5.3 User setting system area type

System area type	Description
High speed monitor area from other station	Setting this area speeds up monitoring from GX Developer connected to a module such as a serial communication module. When monitor is to be executed simultaneously from multiple locations, this area is used to register the monitor data from GX Developers connected to modules such as serial communication modules.
Online change area of multiple blocks (Online change area of FB definition/ST)	Setting this area enables multiple blocks of data to be changed online. Refer to the following manual for the number of blocks to which online change can be made in this area setting. ☞ GX Developer Operating Manual

☒POINT

When the user setting system area is created, the available area decreases by the number of created area steps.
The memory capacity can be checked from the Read from PLC screen of GX Developer. (☞ (3)(c)in this section)

(c) Checking the memory capacity after formatting

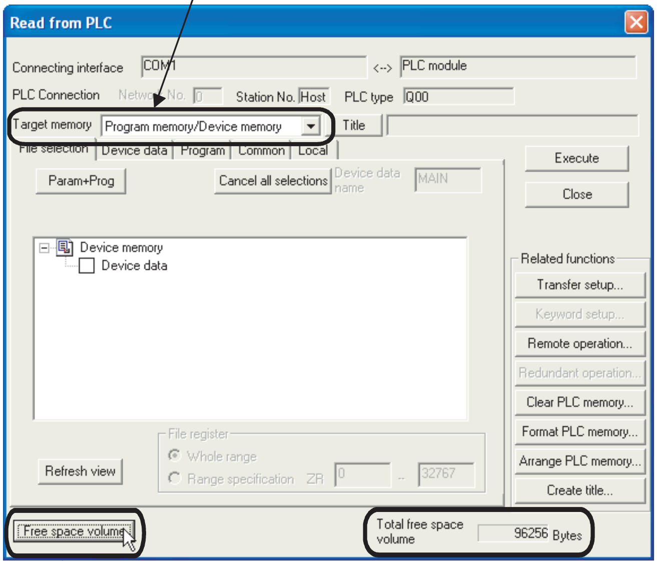
To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

1) Select "Program memory/Device memory" as the target memory on the Read from PLC screen.

2) Click the Free space volume button.

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

1) Select 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 Title

Device data name MAIN

Device memory
□ Device data

File register
● Whole range
○ Range specification ZR 0 .. 32767

Total free space volume 96256 Bytes

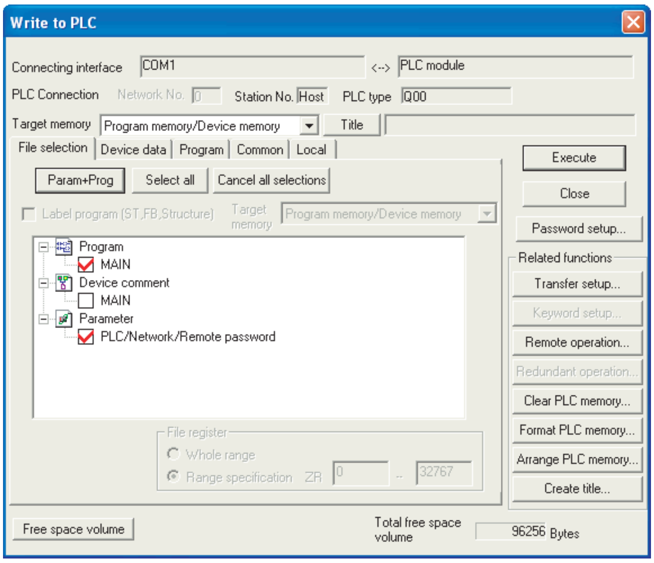
2) Click Free space volume button. 3) Memory capacity is displayed.

Figure 5.3 Memory capacity checking procedure

(4) Write to program memory

To write data to the program memory, choose [Online] → [Write to PLC] on GX Developer.

Select "Program memory/Device memory" as the target memory on the Write to PLC screen and write data to the PLC.



Connecting interface: COM1 <--> PLC module

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

Target memory: Program memory/Device memory Title

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

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

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

Total free space volume: 96256 Bytes

Figure 5.4 Write to PLC screen

☒POINT

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

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

5.1.3 Standard ROM

(1) Definition of standard ROM

The standard ROM is a memory for performing the boot operation in the Basic model QCPU.

The programs stored in the standard ROM are booted (read) to the program memory (☞ Section 5.1.2) to perform operations.

(2) Storable data

The standard ROM can store parameters, intelligent function module parameters, programs, device comments and device initial value data.
(☞ Section 5.1.1(2))

(3) Checking the memory capacity

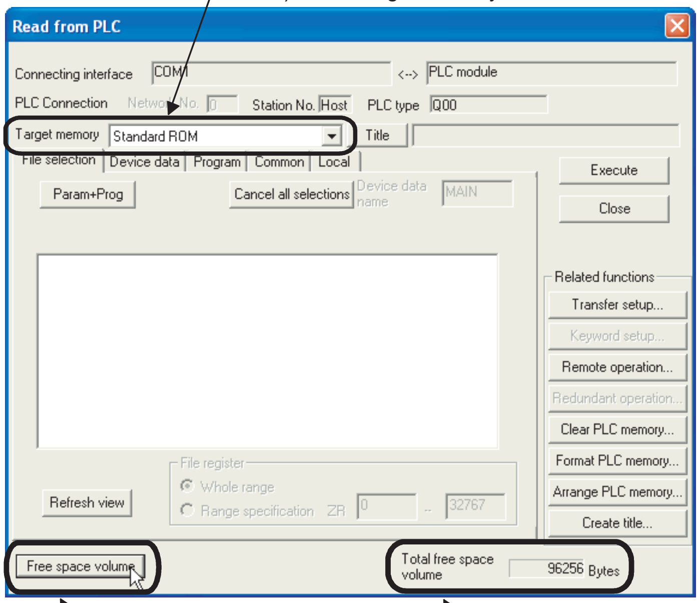
To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

1) Select "Standard ROM" as the target memory on the Read from PLC screen.

2) Click the Free space volume button.

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

1) Select 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 Title

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

Total free space volume: 96256 Bytes

2) Click Free space volume button.   3) Memory capacity is displayed.

Figure 5.5 Memory capacity checking procedure

(4) Write to standard ROM

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

☒POINT

The file size has the minimum unit. (☞ Section 5.4.4)
The occupied memory capacity may be greater than the actual file size.

(5) How to use the program stored in the standard ROM

Since operation cannot be executed by the program stored in the standard ROM, use that program by booting (reading) it to the program memory. (☞ Section 5.1.5)

5.1.4 Standard RAM

(1) Definition of standard RAM

The standard RAM is a memory for file registers. (Q00CPU and Q01CPU only)
File registers in the standard RAM allow fast access like data registers.

(2) Storable data

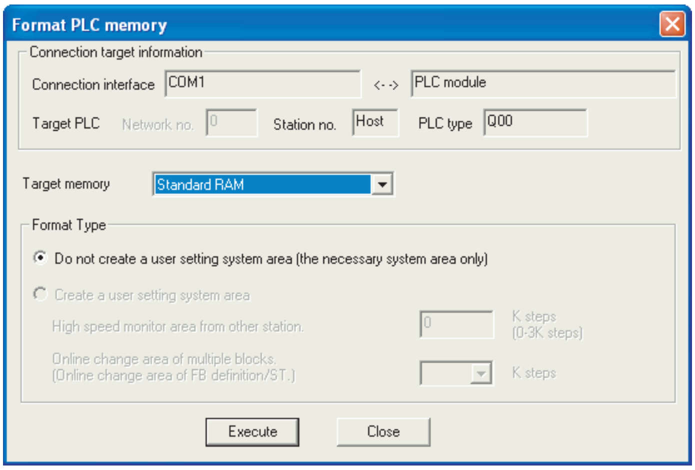
The standard RAM can store one file of file register.
(☞ Section 5.1.1(2))

(3) Before using the standard RAM

- Make sure to format the standard RAM by GX Developer before using Q00CPU or Q01CPU of the function version A
- Q00CPU and Q01CPU of the function version B do not need to be formatted. (Formatting the standard RAM using GX Developer to Q00CPU or Q01CPU of the function B, an error code "4150H" will be displayed on the GX Developer.)

(a) Formatting

To format the standard RAM, choose [Online] → [Format PLC memory] on GX Developer, and select "Standard RAM" as the 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: 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 Standard RAM formatting

(b) Checking the memory capacity after formatting

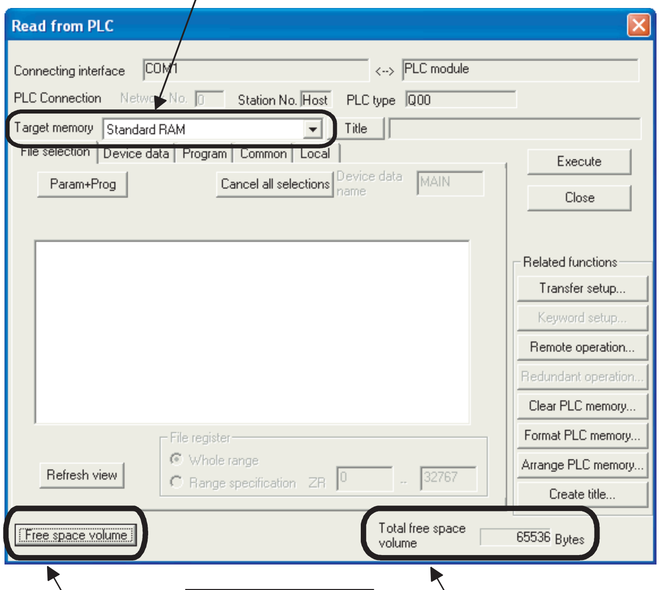
To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

1) Select "Standard RAM" as the target memory on the Read from PLC screen.

2) Click the Free space volume button.

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

1) Select target memory.



Read from PLC

Connecting interface COM1 <--> PLC module

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

Target memory Standard RAM

Title

Device data name MAIN

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

Total free space volume 65536 Bytes

2) Click Free space volume button.

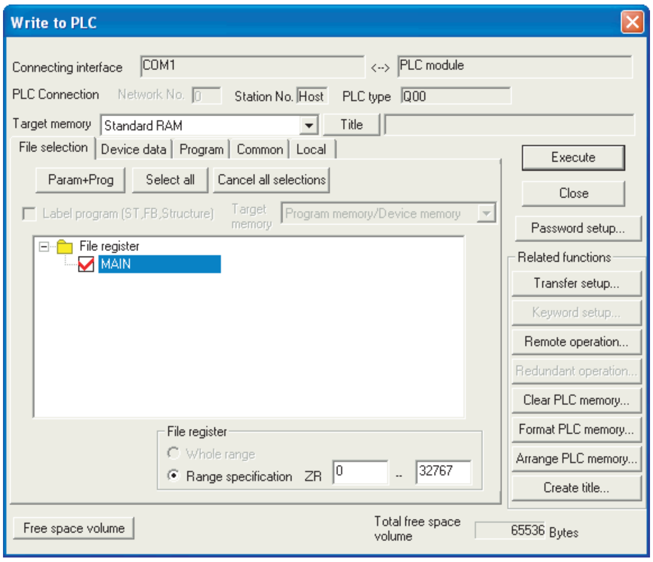
3) Memory capacity is displayed.

Figure 5.7 Memory capacity checking procedure

(4) Write to standard RAM

To write data to the standard RAM, choose [Online] → [Write to PLC] on GX Developer.

Select "Standard RAM" as the target memory on the Write to PLC screen and write data to the PLC.



Write to PLC

Connecting interface: COM1 <--> PLC module

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

Target memory: Standard RAM

Title

File selection | Device data | Program | Common | Local

☐ 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 the minimum unit. (☞ Section 5.4.4)

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

5.1.5 Standard ROM program execution (boot run) and writing

(1) Standard ROM program execution (boot run)

(a) Standard ROM program execution

The Basic model QCPU performs operation of the program stored in the program memory.
It does not operate the program stored in the standard ROM.

The program stored in the standard ROM is booted (read) to the program memory to perform operation.

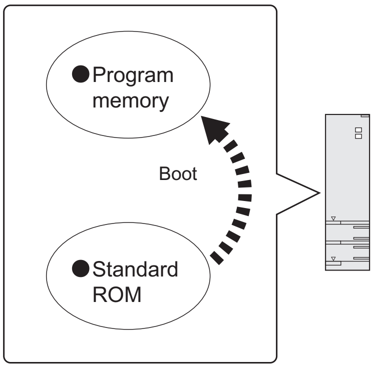


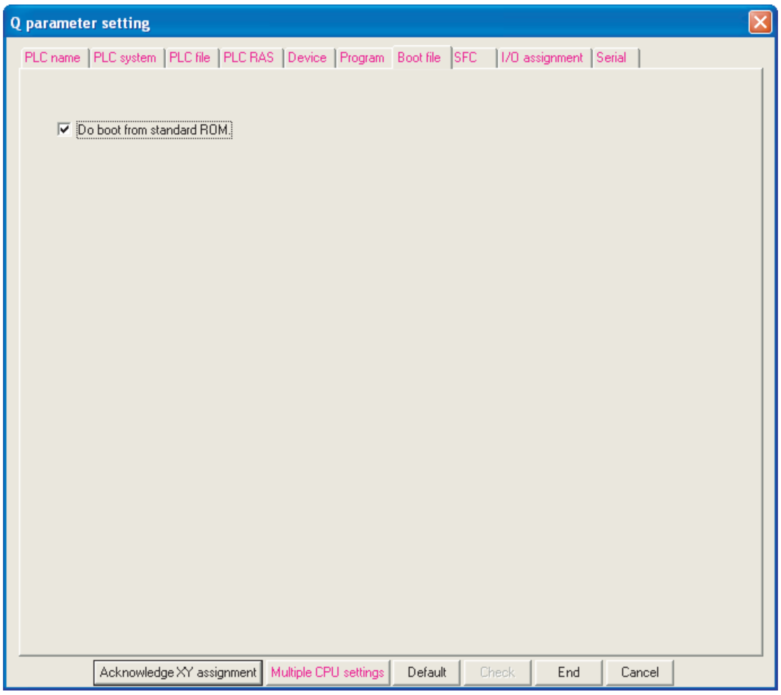
Figure 5.9 Boot run

(b) Procedure for boot run

The following provides the procedure for boot run.

1) **Program creation by GX Developer**
Create a program for executing boot run.

2) **Boot file by GX Developer**
Set "Do boot from Standard ROM" in the Boot file of the PLC parameter dialog box.



☑ Do boot from standard ROM.

Figure 5.10 Boot file

3) **Write to standard ROM by GX Developer**

- Choose [Online] → [Write to PLC] on GX Developer and write the files to the program memory.
- Choose [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM...] on GX Developer, and write to the standard ROM the files written to the program memory. (☞ (2) in this section)

4) **Program execution**

Resetting with the RUN/STOP/RESET switch of the Basic model QCPU boots the program from the standard ROM.

5) **Check for normal boot completion**

Whether the boot is normally completed or not can be checked by the special relay (SM660) status.

Refer to Appendix 1 for the special relay.

(c) **Operation to stop boot run**

Perform the following operation using GX Developer to stop boot run and execute operation by the parameter program written to the program memory.

1) Format the program memory.

2) Choose [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM...].
(Parameters and sequence program data in the standard ROM are cleared.)

3) Write parameters and sequence program data to the program memory.

(d) **Precautions for standard ROM program execution**

1) **Files stored into standard ROM**

Before executing boot run, store the following files into the standard ROM.

- Parameter *1
- Intelligent function module parameter
- Program *2
- Device comment
- Device initial value

* 1 : Must be stored into the standard ROM.
* 2 : Any of the sequence program, ST program and SFC program is necessary.

2) **Online change during boot run**

If online change is made to a program in the program memory during boot run from the standard ROM, the change is not updated on the program in the boot source standard ROM.

Hence, write the program to the standard ROM (☞ (2) in this section) when the CPU module is put in a STOP status.

3) **When program memory contents change at power OFF → ON or reset**

If the sequence program written to the program memory changes when the PLC is powered ON or reset, boot run may be in execution.

Refer to "(1)(c)Operation to stop boot run" in this section, and stop the boot run.

(2) Write to standard ROM

The program memory files are written to the standard ROM by batch-copying them to the standard ROM.

(a) Before write

Check the following points before writing the files to the standard ROM.

1) **Saving the standard ROM files**

When files are written to the standard ROM, all files previously stored in the standard ROM are automatically deleted.
Before writing files to the standard ROM, choose [Online] → [Read from PLC] on GX Developer and save the stored files using GX Developer in advance.

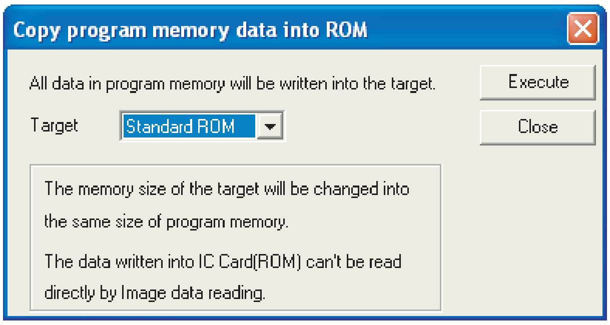
2) **Preparation of files to be written**

Since all files stored in the standard ROM are automatically deleted when files are to be written to the standard ROM, prepare all files to be stored in advance.

(b) Write procedure

The procedure to write files to the standard ROM will be explained.

1) Choose [Online] → [Write to PLC (Flash ROM)] → [Copy program memory data into ROM] on GX Developer.
2) The Write the program memory to ROM screen appears.



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

3) Select the write destination and copy the program memory files to the standard ROM.

(3) Additions/changes to standard ROM files

Since all files stored in the standard ROM are automatically deleted when files are to be written to the standard ROM, additions/changes to the stored files cannot be made directly.

Observe the following steps.

1) Choose [Online] → [Read from PLC] on GX Developer and read all files from the standard ROM.
2) Make necessary additions/changes to the read files.
3) Write the modified files to the program memory.
4) Choose [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM...], and copy these files to the program memory.

(4) Precautions

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

When files are written to the standard ROM, the communication is checked for 180 seconds if the communication time check period ("Check at communication time") is set to 180 seconds or less in GX Developer.

If an error occurs, extend the set time on the Transfer setup screen in GX Developer.

(b) CC-Link Writing files from GX Developer on another station via CC-Link

Since long processing time is required for writing files to the standard ROM, set the CPU monitoring time setting (SW0A) of CC-Link to 180 seconds or more.

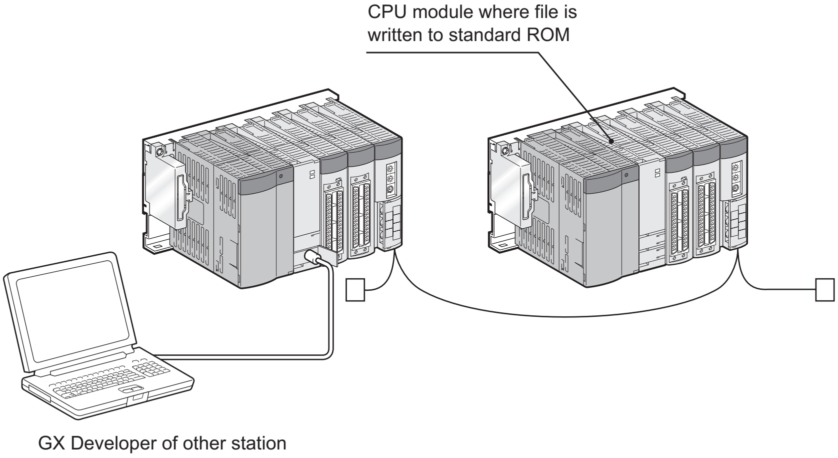


Figure 5.12 Write from GX Developer of other station (via CC-Link)

Remark

Refer to the following manual for details on the communication time check period ("Check at communication time") setting and the CPU monitoring time ("Watchdog time") setting.

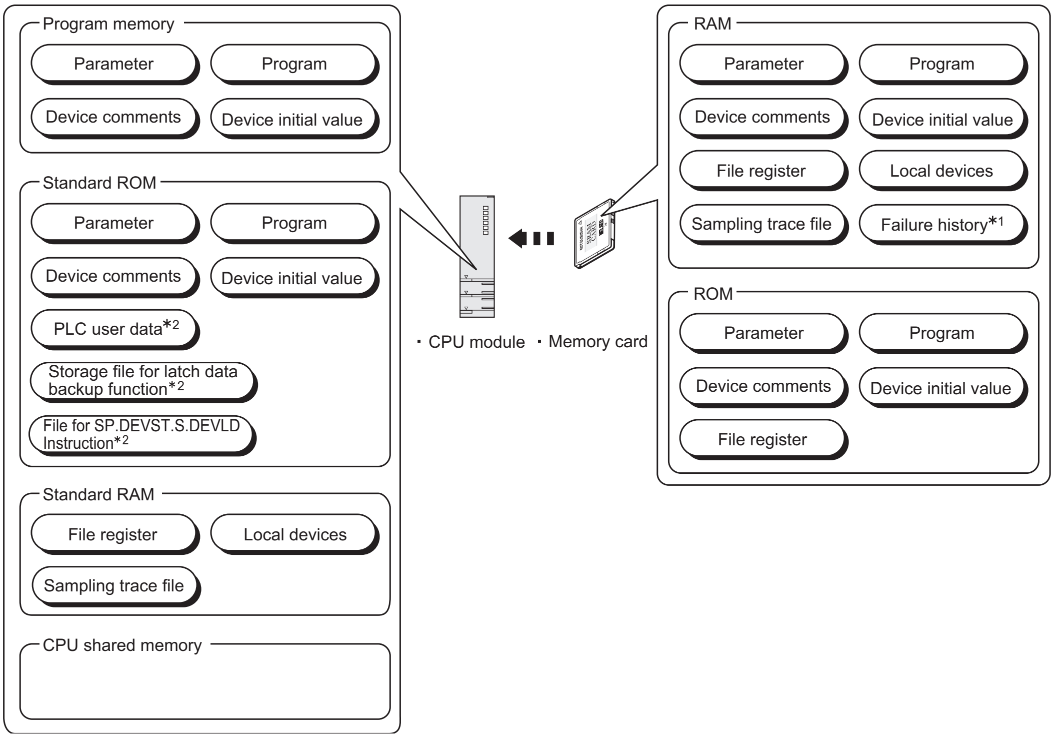
☞ GX Developer Version 8 Operating Manual

5.2 High Performance Model QCPU, Process CPU, Redundant CPU, Universal Model QCPU

5.2.1 Memory configuration and storable data

This section explains the memories used in the High Performance model QCPU, Process CPU, Redundant CPU and Universal model QCPU, and the data can be stored in those memories.

(1) Memory configuration



* 1 : The Universal model QCPU cannot store the error history in the memory card.
* 2 : Can be stored only when the Universal model QCPU is used.

Figure 5.13 Memory configuration of the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU

(a) **Program memory (☞ Section 5.2.2)**
A memory for storing programs required for operations in the High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU.

Universal UD Note5.1

The programs stored in the standard ROM Note5.1 and memory card are booted (read) to the program memory to perform operations. (☞ Section 5.2.9)

(b) **Standard ROM (☞ Section 5.2.4)**
A memory for storing data such as parameters and programs.

Universal UD Note5.1

This memory can be specified as a storage drive of the boot source programs in the High Performance model QCPU, Process CPU, and Redundant CPU. Note5.1

(c) **Standard RAM (☞ Section 5.2.5)**
A memory for using file registers, local devices, and sampling trace files without installing a memory card.

(d) **Memory card (☞ Section 5.2.6)**
A memory for the expansion use of the built-in memory in the High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU. SRAM card, Flash card, and ATA card are available.

(e) **CPU shared memory (☞ QCPU User's Manual (Multiple CPU System))**
A memory for writing/reading data between CPU modules in a multiple CPU system.

Universal UD Note5.1

The Universal model QCPU cannot boot programs from the standard ROM to the program memory. (☞ Section 5.2.3)

(2) Data that can be stored into memories

Table5.4 indicates the data that can be stored into the program memory, standard RAM, standard ROM and memory cards and the corresponding drive Nos.

Table5.4 Storable data and storage locations

	CPU module built-in memories			Memory card (RAM)	Memory cards (ROM)		File name and extension	Remarks
	Program memory	Standar d RAM	Standar d ROM	SRAM card	Flash card	ATA card		
Drive No.*9	0	3	4	1	2			
Parameter	○	×	○	○	○	○	PARAM.QPA	1 data/drive
Intelligent function module parameter*12	○	×	○	○	○	○	IPARAM.QPA	1 data/drive
Program	◎	×	○*1	○*1	○*1	○*1	***.QPG	--
Device comment	○*2	×	○*3	○*3	○*3	○*3	***.QCD	--
Device initial value	○	×	○	○	○	○	***.QDI	--
Device data	×	×	△*11	×	×	×	***.QST	--
File register	×	○*5*6	×	○	○*4	×	***.QDR	--
Local device	×	○*5	×	○	×	×	***.QDL	1 data/CPU module
Sampling trace file	×	○*5*10	×	○	×	×	***.QTD	--
Failure history data*13	×	×	×	○	×	×	***.QFD	--
PLC user data	×	×	△*11	×	×	○*6	***.***	--
User setting system area*7	○	×	×	×	×	×	--	--

◎:Necessary data, ○:Storable data, △:Restricted, ×:Unstorable data

* 1 : To execute the program stored in the standard ROM or memory card, boot setting to the program memory is required in the PLC parameter dialog box. Note that the Universal model QCPU cannot boot data from the standard ROM to the program memory.(☞ Section 5.2.3)

* 2 : Data can be written by operating from the GX Developer.
Device comments cannot be used in an instruction of a sequence program.

* 3 : The read from a sequence program requires several scans.

* 4 : A sequence program allows the read only.
No data can be written through access from a sequence program.

* 5 : One file register file, one local device file and one sampling trace file can be stored into the standard RAM.

* 6 : Refer to CHAPTER 2 for the number of storable file register points.

* 7 : Data can be written or read with the following instructions:
- SP.FREAD (allows the batch read from a specified file on a memory card)
- SP.FWRITE (allows the batch write to a specified file on a memory card)

* 8 : Set the area used by the system. (☞ Section 5.2.2(3)(b))

* 9 : Drive No. is used to specify the target memory to be written/read to/from the external device using a sequence program or MC protocol.
Since the target memory name is specified on GX Developer, it is not necessary to take into account the drive No.

* 10 : When the sampling trace file is stored in the standard RAM, check the version of the CPU module and GX Developer. (☞ Appendix 4)

* 11 : Only the Universal models QCPU can be stored in the standard RAM.

* 12 : Store intelligent function module parameters in the same drive with parameters.
Intelligent function module parameters do not become valid when stored to the different drive.

* 13 : The error history data cannot be stored into a memory card for the Universal model QCPU.
(☞ Section 6.18.2)

(3) Memory capacities and formatting necessities

Table5.5 shows the memory capacity and formatting necessity of each memory.

Table5.5 Memory capacity and formatting necessity

		Q02CPU*2	Q02HCPU*2	Q06HCPU*2	Q12HCPU*2	Q25HCPU*2	Formatting
Program memory		112kbytes (28ksteps)	112kbytes (28ksteps)	240kbytes (60ksteps)	496kbytes (124ksteps)	1008kbytes (252ksteps)	*1
Standard ROM		112kbytes	112kbytes	240kbytes	496kbytes	1008kbytes	Unnecessary
Standard RAM		64k bytes	128k bytes		256k bytes		*1
Memory card	SRAM card	Q2MEM-1MBS : 1M bytes Q2MEM-2MBS : 2M bytes					Necessary (Executed by 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 (Executed by GX Developer)

		Q02PHCPU	Q06PHCPU	Q12PHCPU	Q25PHCPU	Q12PRHCPU	Q25PRHCPU	Formatting
Program memory		112kbytes (28ksteps)	240kbytes (60ksteps)	496kbytes (124ksteps)	1008kbytes (252ksteps)	496kbytes (124ksteps)	1008kbytes (252ksteps)	*1
Standard ROM		112kbytes	240kbytes	496kbytes	1008kbytes	496kbytes	1008kbytes	Unnecessary
Standard RAM		256k bytes						*1
Memory card	SRAM card	Q2MEM-1MBS : 1M bytes Q2MEM-2MBS : 2M bytes						Necessary (Executed by 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 (Executed by GX Developer)

		Q02UCPU	Q03UDCPU	Q04UDHCPU	Q06UDHCPU	Q13UDHCPU	Q26UDHCPU	Formatting
			Q03UDECPU	Q04UDEHCPU	Q06UDEHCPU	Q13UDEHCPU	Q26UDEHCPU	
Program memory		80kbytes (20ksteps)	120kbytes (30ksteps)	160kbytes (40ksteps)	240kbytes (60ksteps)	520kbytes (130ksteps)	1040kbytes (260ksteps)	*1
Standard ROM		512kbytes	1024kbytes	1024kbytes	1024kbytes	2048kbytes	2048kbytes	Unnecessary
Standard RAM		128k bytes	192k bytes	256k bytes	768k bytes	1024k bytes	1280k bytes	*1
Memory card	SRAM card	Q2MEM-1MBS : 1M bytes Q2MEM-2MBS : 2M bytes Q3MEM-4MBS : 4M bytes Q3MEM-8MBS : 8M bytes						Necessary (Executed by 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 (Executed by GX Developer)

* 1 : When the memory contents become indefinite in an initial status or due to a dead battery, the memory is automatically formatted when the PLC is powered off and on, or reset. Before use, however, be sure to format the memory using GX Developer.

* 2 : The standard RAM capacity of the High Performance model QCPU varies depending on function version upgrades. (☞ Appendix 4.2)

☒POINT

1. When data are written to each memory, the unit of storage capacity changes depending on the target CPU module and memory area. (☞ Section 5.4.4)
2. For memory capacity calculation, 1 step is equal to 4 bytes.

5.2.2 Program memory

(1) Definition of program memory

The program memory is a memory for storing programs required for operations in the High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU.

The Universal model QCPU transfers the program from the program memory to the program cache memory to perform operations. (☞ Section 5.2.3)

Universal UD Note5.2

The program stored in the standard ROM[Note5.2] or memory card is booted (read) to the program memory to perform operation. (☞ Section 5.2.9)

(2) Storable data

The program memory can store parameters, intelligent function module parameters, programs, device comments, device initial values and user setting system area data. (☞ Section 5.2.1(2))

☒POINT

The capacity of the program memory is the same as the program capacity of each module (☞ Section 5.2.1(3)).

If the total volume of the data to be stored into the program memory exceeds its capacity, examine the following.

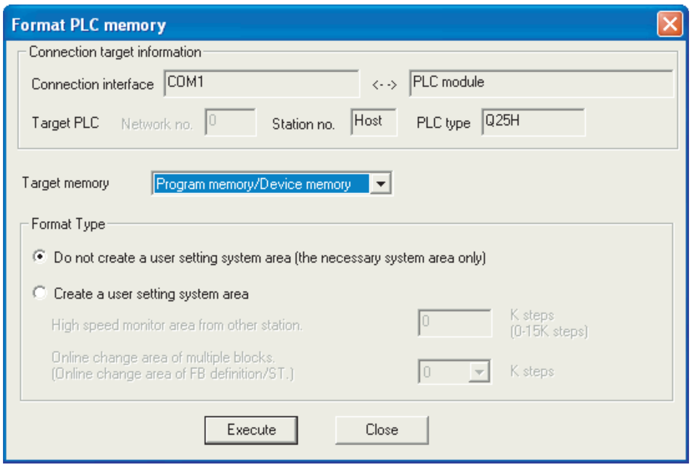
1. Reduce the user setting system area.
2. Transfer the data other than the programs to the standard ROM or a memory card.

(3) Before using the program memory

Before using the program memory, be sure to format it by GX Developer.

(a) Formatting

To format the program memory, choose [Online] → [Format PLC memory] on GX Developer, and select "Program memory/Device memory" as the target memory.



Format PLC memory

Connection target information

Connection interface: COM1 <--> PLC module

Target PLC: Network no. 0 Station no. Host PLC type Q25H

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 Program memory formatting

Universal UD Note5.2

The Universal model QCPU cannot boot programs from the standard ROM to the program memory. (☞ Section 5.2.3)

(b) Create a user setting system area

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

1) Do not create a user setting system area

The program memory is formatted without the user setting system area being created.

2) Create a user setting system area

The user setting system area is created during formatting.
There are the following user setting system areas (Table5.6).

Table5.6 User setting system area type

System area type	Description
High speed monitor area from other station	Setting this area speeds up monitoring from GX Developer connected to a module such as a serial communication module. When using RS-232 and USB concurrently on GX Developer, this area is used to register the monitor data from GX Developer connected to a module such as a serial communication module.
Online change area of multiple blocks (Online change area of FB definition/ST)	• High Performance model QCPU, Process CPU or Redundant CPU Setting this area enables multiple blocks of data to be changed online. • Universal model QCPU Multiple blocks of data can be changed online without setting this area. (Setting of this area is not required.) Refer to the following manual for the number of blocks to which online change can be made in this area setting. ☞ GX Developer Operating Manual

☒POINT

When the user setting system area is created, the available area decreases by the number of created area steps.
The memory capacity can be checked from the Read from PLC screen of GX Developer. (☞ (3)(c)in this section)

(c) **Checking the memory capacity after formatting**

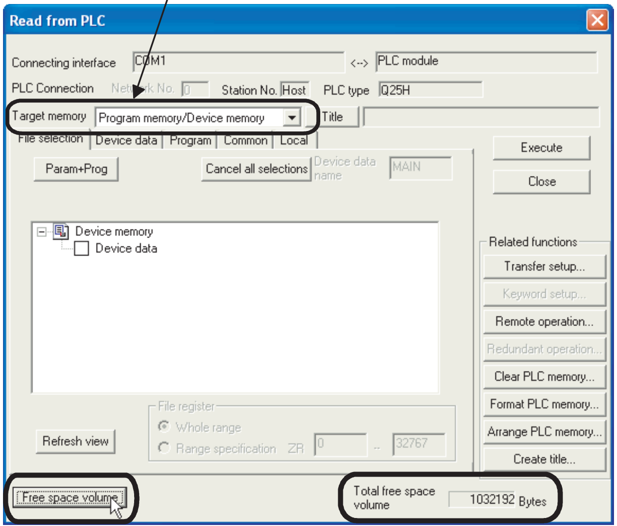
To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

1) Select "Program memory/Device memory" as the target memory on the Read from PLC screen.

2) Click the Free space volume button.

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

1) Select target memory.



2) Click Free space volume button.

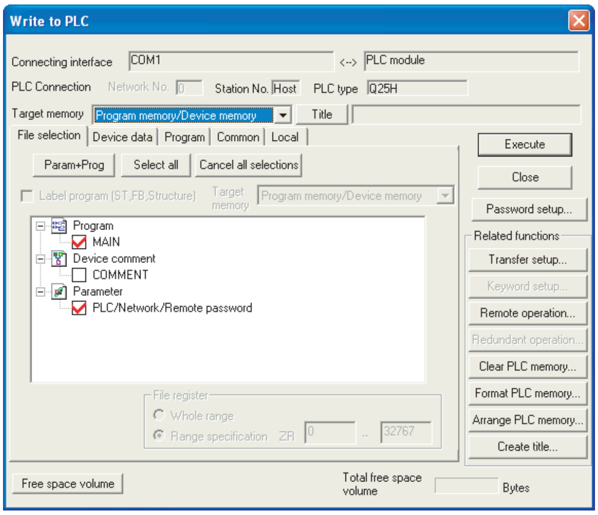
3) Memory capacity is displayed.

Figure 5.15 Memory capacity checking procedure

(4) Write to program memory

To write data to the program memory, choose [Online] → [Write to PLC] on GX Developer.

Select "Program memory/Device memory" as the target memory on the Write to PLC screen and write data to the PLC.



Write to PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q25H

Target memory: Program memory/Device memory

File selection:

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

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

Figure 5.16 Write to PLC screen

☒POINT

The file size has the 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 rises, the difference between the occupied memory capacity and actual file size increases.

High Performance Note5.3 Process Note5.3 Redundant Note5.3

5.2.3 Program cache memory (Universal model QCPU only)

(1) Program cache memory overview

The Universal model QCPU cannot boot programs from the standard ROM to the program memory.

The Universal model QCPU transfers programs stored in the program memory to the program cache memory to perform operations. (☞ (3) in this section)

The following shows the transfer timing from the program memory to the program cache memory.

- During initial processing when the programmable controller is powered ON
- During initial processing when the reset operation of the CPU module is performed

Figure 5.17 shows the flow of program operations.

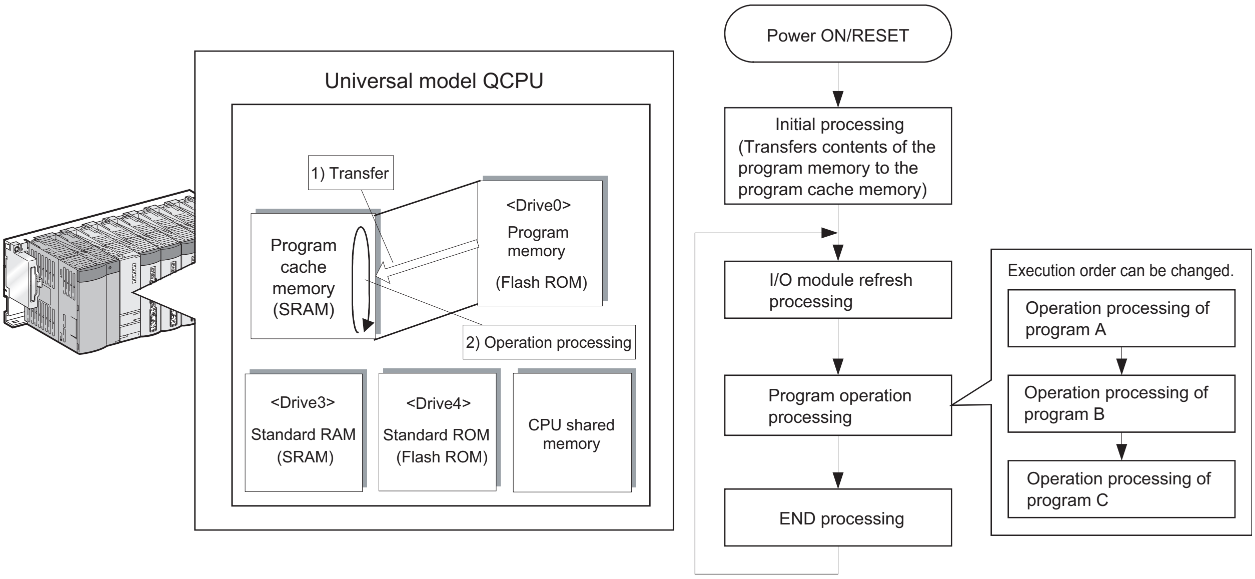

Universal model QCPU
1) Transfer
Program cache memory (SRAM)
<Drive0> Program memory (Flash ROM)
2) Operation processing
<Drive3> Standard RAM (SRAM)
<Drive4> Standard ROM (Flash ROM)
CPU shared memory
Power ON/RESET
Initial processing (Transfers contents of the program memory to the program cache memory)
I/O module refresh processing
Program operation processing
END processing
Execution order can be changed.
Operation processing of program A
Operation processing of program B
Operation processing of program C

Figure 5.17 Flow of program operations

High Performance Note5.3 Process Note5.3 Redundant Note5.3

Since the High Performance model QCPU, Process CPU and Redundant CPU do not use the program cache memory, it is not necessary to be conscious of this description.

(2) Writing programs

When executing the write to PLC operation from GX Developer, programs and parameters are written into the program cache memory of the CPU module once, and then transferred to the program memory after writing is completed.
Figure 5.18 shows the flow of writing programs

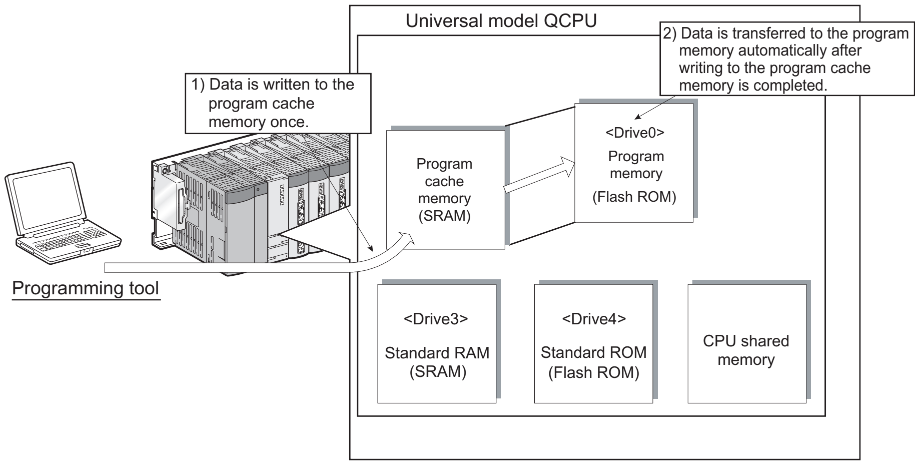

Universal model QCPU
1) Data is written to the program cache memory once.
2) Data is transferred to the program memory automatically after writing to the program cache memory is completed.
Programming tool
Program cache memory (SRAM)
<Drive0>
Program memory
(Flash ROM)
<Drive3>
Standard RAM (SRAM)
<Drive4>
Standard ROM (Flash ROM)
CPU shared memory

Figure 5.18 Flow of writing programs

(3) Transferring data to a program memory by GX Developer

The batch transfer of data from a program cache memory to a program memory can also be performed by [Online] → [Program memory batch transfer] of GX Developer. Also, transfer time will be prolonged by the time shown in Table 5.7 if performing batch transfer of data to the program memory when the CPU module in the RUN status.

Table5.7 Transfer time of program batch transfer

CPU model	Transfer time
Q02UCPU	Ts × 320 + 4.8 (s)
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU	Ts × 260 + 4.7 (s)
Q13UDHCPU, Q13UDEHCPU	Ts × 600 + 8.0 (s)
Q26UDHCPU, Q26UDEHCPU	Ts × 1100 + 15.0 (s)

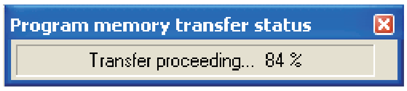
Ts: Scan time (s)

(4) Checking the transfer status of data to a program memory

Transfer status to the program memory can be checked either on the Program memory transfer status screen of GX Developer or in the special relay/special register.

(a) Checking on the Program memory transfer status screen

Figure 5.19 shows the Program memory transfer status screen of GX Developer.


Program memory transfer status

Transfer proceeding... 84 %

Figure 5.19 Program memory transfer status screen

(b) Checking method in the special relay/special register

Transfer status to the program memory can be checked in SM681 or SD681.

Universal UD Note5.4

(5) Checking the transfer status of data to a program memory Note5.4

Transfer status from the program cache memory to the program memory can be checked by the special relay (SM165).

Universal UD Note5.4

When checking the transfer status to the program memory with a Universal model QCPU, check the version of the CPU module. (☞ Appendix 4.5)

5.2.4 Standard ROM

Universal UD Note5.5

(1) Definition of standard ROM

The standard ROM is a memory for storing data such as parameters and programs. This memory can be specified as a storage drive of the boot source programs in the High Performance model QCPU, Process CPU, and Redundant CPU.Note5.5

The standard ROM can save parameters and programs without battery backup.

(2) Storable data

The standard ROM can store parameters, intelligent function module parameters, programs, device comments and device initial value data.

(☞ Section 5.2.1(2))

Universal UD Note5.5

The Universal model QCPU cannot boot programs from the standard ROM to the program memory. (☞ Section 5.2.3)

(3) Checking the memory capacity

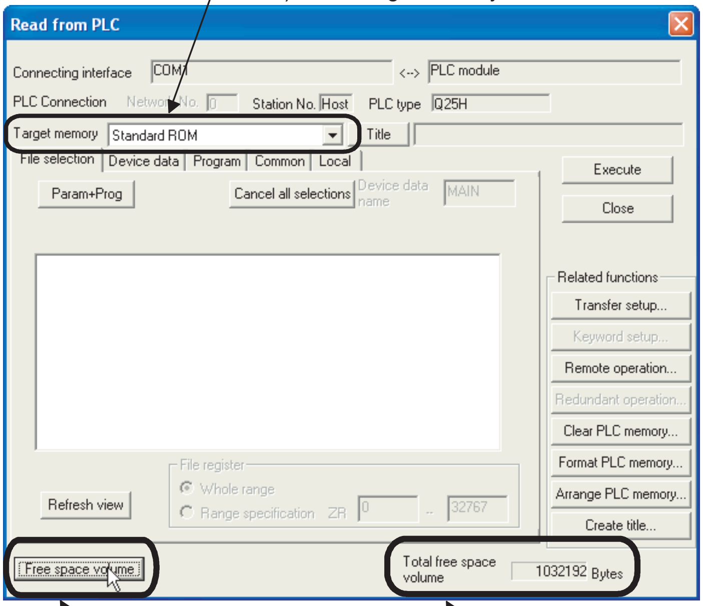
To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

1) Select "Standard ROM" as the target memory on the Read from PLC screen.

2) Click the Free space volume button.

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

1) Select target memory.



Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q25H

Target memory: Standard ROM

Device data name: MAIN

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

Total free space volume: 1032192 Bytes

2) Click Free space volume button.

3) Memory capacity is displayed.

Figure 5.20 Memory capacity checking procedure

(4) Write to standard ROM

The method of writing data to the standard ROM differs depending on the CPU module.

(a) High Performance model QCPU, Process CPU or Redundant CPU

- Choose [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM...] on GX Developer to batch-copy the program memory data to the standard ROM. (☞ Section 5.2.7)
- Choose [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] on GX Developer to write data to the standard ROM. (☞ Section 5.2.7)
- Use the automatic write of all data from memory card to standard ROM. (☞ Section 5.2.8)

(b) Universal model QCPU

- Choose [Online] → [Write to PLC] on GX Developer, and select "Standard ROM" as the target memory to write the program memory data. (☞ Section 5.2.2) Select "Standard ROM" as the target memory to write the program memory data.

☒POINT

The file size has the 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 rises, the difference between the occupied memory capacity and actual file size increases.

Remark

Refer to the sections above for details of the selection, applications, procedures and precautions for the standard ROM wiring methods.

(5) How to use the program stored in the standard ROM

Universal UD Note5.6

The programs stored in the standard ROM are booted (read) to the program memory to perform operations. Note5.6 (☞ Section 5.2.9)

Universal UD Note5.6

The Universal model QCPU cannot boot programs from the standard ROM to the program memory. (☞ Section 5.2.3)

5.2.5 Standard RAM

(1) Definition of standard RAM

The standard RAM is a memory for using file registers, local devices, and sampling trace files without installing a memory card.

The standard RAM used as file registers allows fast access like the data registers.

(2) Storable data

The standard RAM can store one file register file, one local device file and one sampling trace file (a total of three files).

(☞ Section 5.2.1(2))

☒POINT

1. If the volume of the files to be stored into the standard RAM exceeds the standard RAM capacity, examine the following.
   - Store the file into a memory card.
   - Examine to reduce points of the file register, local device and sampling trace.

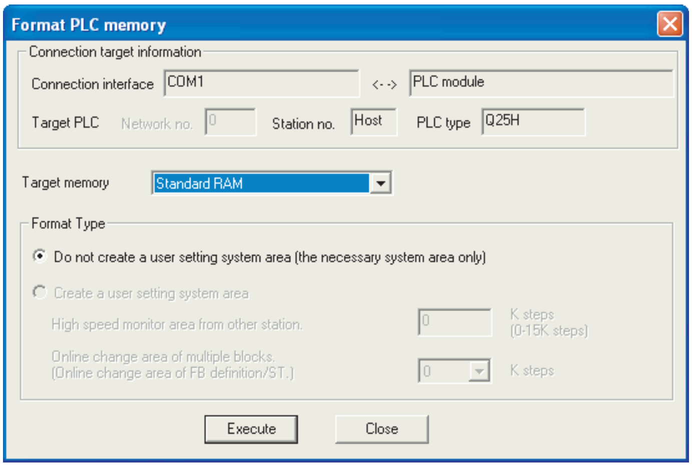
   However, note that in the case of 1. above, access speed will be lower than that of the standard RAM.
2. If the sampling trace file is stored into the standard RAM, check the version of the CPU module and GX Developer. (☞ Appendix 4)

(3) Before using the standard RAM

Before using the standard RAM, be sure to format it by GX Developer.

(a) Formatting

To format the standard RAM, choose [Online] → [Format PLC memory] on GX Developer, and select "Standard RAM" as the target memory.



Format PLC memory

Connection target information

Connection interface COM1 <--> PLC module

Target PLC Network no. 0 Station no. Host PLC type Q25H

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

Execute Close

Figure 5.21 Standard RAM formatting

(b) Checking the memory capacity after formatting

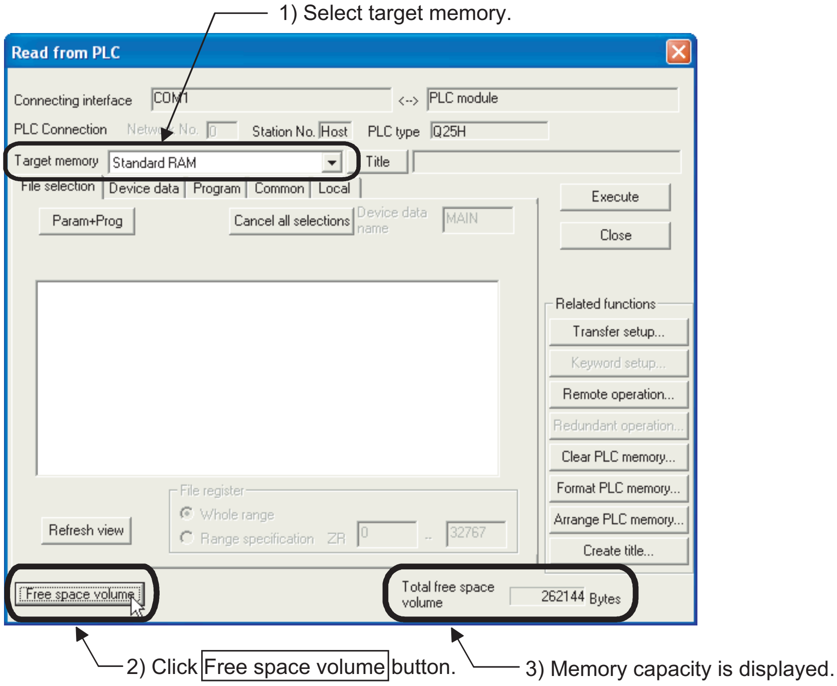
To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

1) Select "Standard RAM" as the target memory on the Read from PLC screen.

2) Click the Free space volume button.

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

1) Select target memory.



Read from PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q25H

Target memory: Standard RAM

Title:

Device data name: MAIN

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

Total free space volume: 262144 Bytes

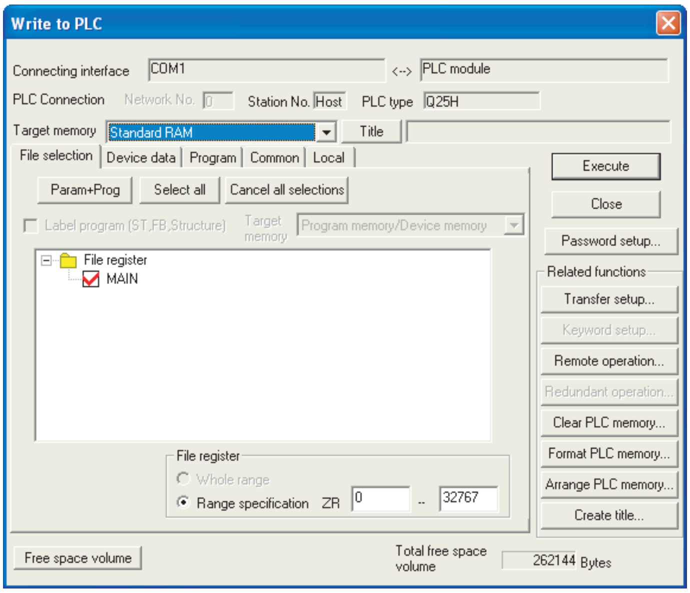
2) Click Free space volume button. 3) Memory capacity is displayed.

Figure 5.22 Memory capacity checking procedure

(4) Write to standard RAM

To write data to the standard RAM, choose [Online] → [Write to PLC] on GX Developer.

Select "Standard RAM" as the target memory on the Write to PLC screen and write data to the PLC.



Write to PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q25H

Target memory: Standard RAM

File register:
- MAIN ☑

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

Total free space volume: 262144 Bytes

Figure 5.23 Write to PLC screen

☒POINT

The file size has the 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 rises, the difference between the occupied memory capacity and actual file size increases.

5.2.6 Memory card

(1) Memory card

A memory card is used to increase memory of the CPU module.
Available memory cards are the SRAM card, Flash card and ATA card.

(a) SRAM card

The file register can be written/read to/from the SRAM card by a sequence program.
The SRAM card is used in the following cases.

- When the number of file register points is greater than the standard RAM capacity
- When using the sampling trace function (☞ Section 6.14)
- When saving 17 or more error history data[Note5.7] (☞ Section 6.18)

Universal UD Note5.7

When using the SRAM card as a file register, data can be written/read to/from a sequence program up to the following points.

- High Performance model QCPU, Process CPU, Redundant CPU......Up to 1017k points
- Universal model QCPU......Up to 4086k points

(b) Flash card

Data can be read by a sequence program only.
Write data by GX Developer and read it by a sequence program.
Use the Flash card when data change will not be made.
A file register can be stored up to the following points.

- High Performance model QCPU, Process CPU, Redundant CPU......Up to 1018k points
- Universal model QCPU.......Up to 2039k points

(c) ATA card

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

Universal UD Note5.7

The Universal model QCPU cannot store the error history in the memory card. (☞ Section 6.18.2)

(2) Storable data

The following data indicated in Table5.8 can be stored into the memory cards.

Table5.8 Data that can be stored into memory cards

Data name	Memory card (RAM)	Memory cards (ROM)	
	SRAM card	Flash card	ATA card
Parameter	○	○	○
Intelligent function module parameter	○	○	○
Program	○	○	○
Device comment	○	○	○
Device initial value	○	○	○
File register	○	○	×
Local device	○	×	×
Sampling trace file	○	×	×
Failure history dat[*1]	○	×	×
PLC user data	×	×	○

○ : Storable data, × : Unstorable data

* 1: The Universal model QCPU cannot store the error history in the memory card. (☞ Section 6.18)

Refer to Section 5.2.1(2) for the list of data that can be stored into each memory.

(3) Before using the SRAM or ATA card

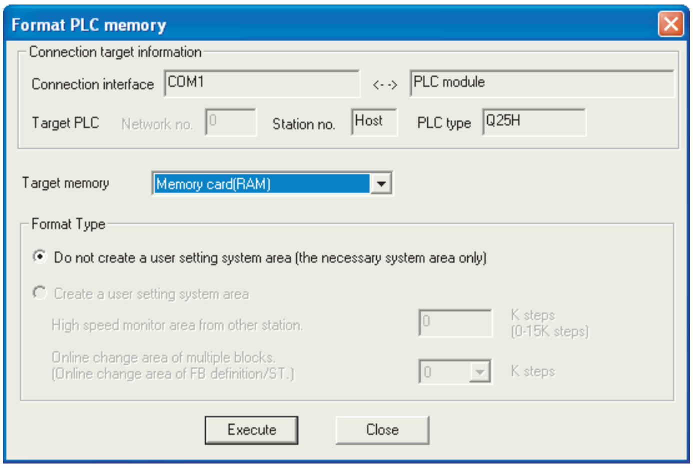
Before using the SRAM or ATA card, be sure to format it by GX Developer.

(a) Formatting

To format the program memory, choose [Online] → [Format PLC memory] on GX Developer.

When formatting the SRAM card, select "Memory card (RAM)" as the target memory.

When formatting the ATA card, select "Memory card (ROM)" as the target memory.



Format PLC memory

Connection target information

Connection interface: COM1 <--> PLC module

Target PLC: Network no. 0, Station no. Host, PLC type Q25H

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 5.24 SRAM or ATA card formatting

☒POINT

1. Use only GX Developer to format the ATA card.

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

2. When the SRAM or ATA card is formatted, the memory card information area is automatically secured. Therefore, the memory capacity is reduced by that amount.

   The memory capacity can be checked from the Read from PLC screen of GX Developer. (☞ (3)(b)in this section)

Remark

The Flash card need not be formatted.

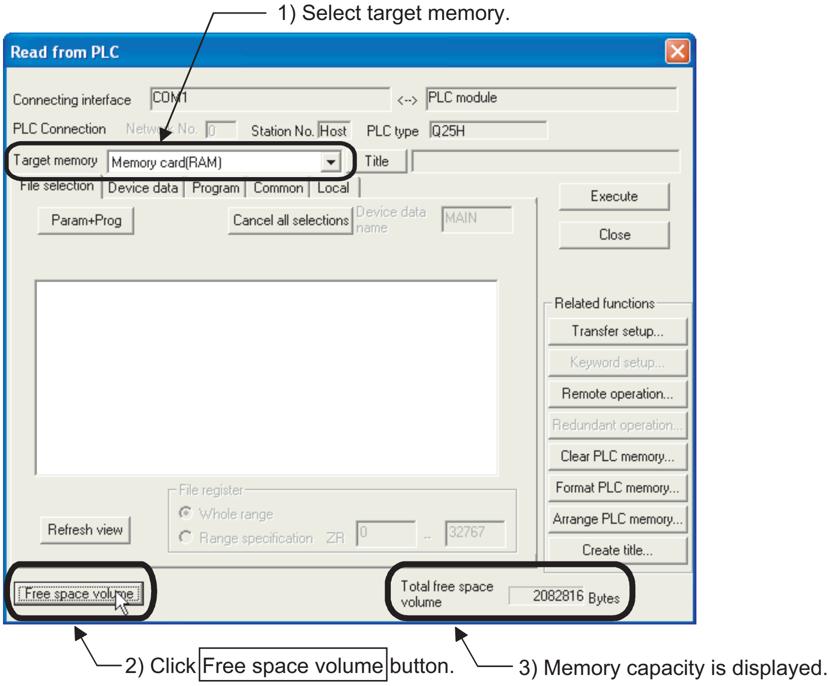
(b) **Checking the memory capacity after formatting**

To check the memory capacity, choose [Online] → [Read from PLC] on GX Developer.

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

2) Click the Free space volume button.

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



Target memory: Memory card(RAM)

Total free space volume: 2082816 Bytes

1) Select target memory.

2) Click Free space volume button.

3) Memory capacity is displayed.

Figure 5.25 Memory capacity checking procedure

(4) Write to memory card

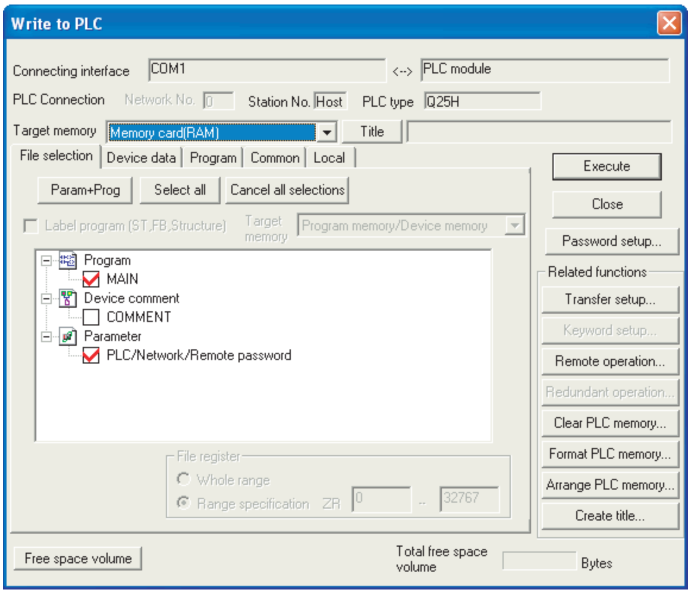
The following explains the pre-write operation and the types of wriring methods.

(a) Write to SRAM or ATA card

To write data to the SRAM or ATA card, choose [Online] → [Write to PLC] on GX Developer.

When writing data to the SRAM card, select "Memory card (RAM)" as the target memory on the Write to PLC screen.

When writing data to the ATA card, select "Memory card (ROM)" as the 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 Q25H

Target memory: Memory card(RAM)

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

Program
☑ MAIN
Device comment
☐ COMMENT
Parameter
☑ PLC/Network/Remote password

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

Total free space volume ___ Bytes

Figure 5.26 Write to PLC screen

(b) Write to Flash card

To write data to the Flash card, there are the following two methods.

- Choose [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM...] on GX Developer to batch-write the program memory data to the Flash card. (☞ Section 5.2.7)
- Choose [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] on GX Developer to write data to the Flash card. (☞ Section 5.2.7)

☒POINT

The file size has the 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 rises, the difference between the occupied memory capacity and actual file size increases.

Remark

Refer to Section 5.2.7 for details of the selection, applications, procedures and precautions for Flash card writing methods.

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

The programs stored in the memory card are booted (read) to the program memory to perform operations. (☞ Section 5.2.9)

5.2.7 Writing data to standard ROM and Flash card using GX Developer

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

Figure 5.27 shows the methods of writing data to the standard ROM and Flash card.

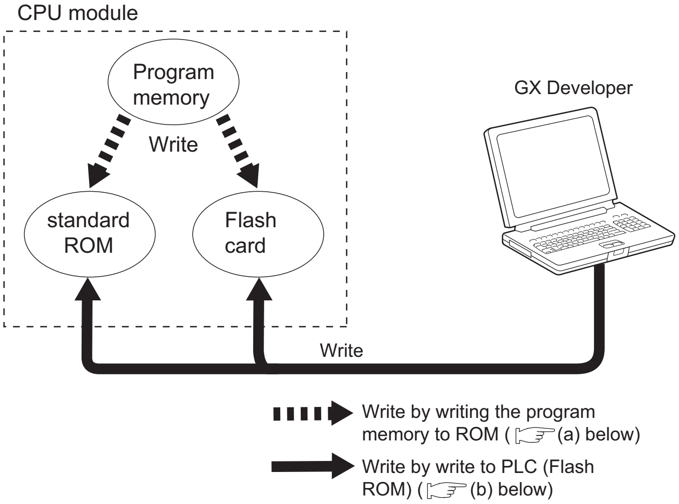

CPU module
Program memory
Write
standard ROM
Flash card
GX Developer
Write
Write by writing the program memory to ROM (☞(a) below)
Write by write to PLC (Flash ROM) (☞(b) below)

Figure 5.27 Methods of writing data to the standard ROM and Flash card

(a) Method using [Write the program memory to ROM...] of GX Developer

The data in the program memory are batch-written to the standard ROM or Flash card.

Universal UD Note5.8

Select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] to execute writing with this method.Note5.8

Writing the program memory to ROM is used in the following cases.

- When performing debugging in the program memory program, copying the debugged data to the standard ROM or Flash card, and executing boot run (☞ Section 5.2.9)
- When saving the program memory data into the standard ROM or Flash card without battery backup.

Universal UD Note5.8

(b) Method using [Write to PLC (Flash ROM)] of GX Developer

The files specified in GX Developer are batch-written to the standard ROM or Flash card.

Select [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM] to execute writing with this method.Note5.8

- When storing the parameters, device initial values and device comments, which volume is too large to be saved in the program memory, to the standard ROM or Flash card.
- When using the file registers stored on the Flash card.

Universal UD Note5.8

The Universal model QCPU cannot write data to the standard ROM by [Write the program memory to ROM...] and [Write to PLC (Flash ROM)].

(2) Write to standard ROM or Flash card

The following explains the pre-write operation and writing methods to the standard ROM or Flash card.

(a) Before writing

Check the following points before writing the files to the standard ROM or Flash card.

1) Saving files in standard ROM or Flash card

When files are to be written to the standard ROM or Flash card, all files stored in the standard ROM or Flash card are automatically deleted.
Before writing files to the standard ROM, choose [Online] → [Read from PLC] on GX Developer and save the stored files using GX Developer in advance.

2) Preparation of files to be written

Since all files stored in the standard ROM or Flash card are automatically deleted when files are to be written to the standard ROM or Flash card, prepare all files to be stored in advance.

3) When executing boot run

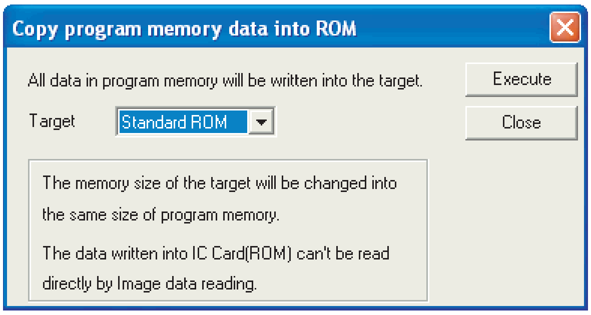
When storing the parameters into the standard ROM or Flash card before boot run, make the boot file setting described in Section 5.2.9.

(b) Write procedure

How to write files to the standard ROM or Flash card is explained here.

1) Using [Copy program memory data into ROM] of GX Developer

- Choose [Online] → [Write to PLC (Flash ROM)] → [Copy program memory data into ROM] on GX Developer.
- The Copy 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: 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 5.28 Copy program memory data into ROM screen

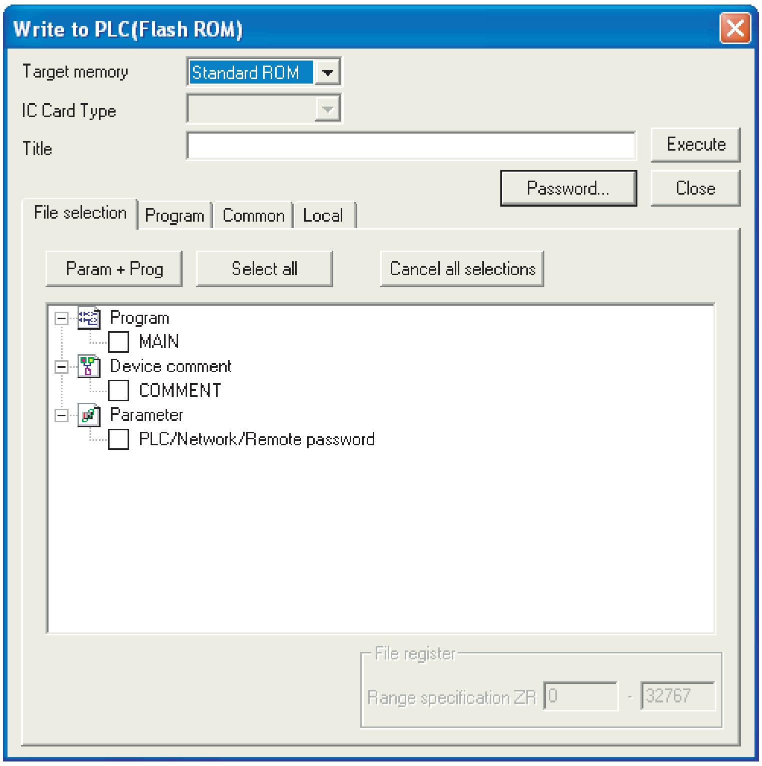
- Select the write destination and copy the program memory files to the standard ROM or Flash card.

☒POINT

1. When files have been written by [Write the program memory to ROM], the memory capacity used by the write destination is the capacity used by the program memory.
   To fully use the memory capacity of the write destination, use [Write to PLC (Flash ROM)] to write the files.
2. When writing the data, which cannot be stored into the program memory (file register), to the Flash card, use [Write to PLC (Flash ROM)] to write the data.

2) Using [Write to PLC (Flash ROM)] of GX Developer

- Choose [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] on GX Developer.
- The Write to PLC (Flash ROM)screen appears.


Write to PLC(Flash ROM)

Target memory: Standard ROM

IC Card Type:

Title:

Figure 5.29 Write to PLC (Flash ROM)screen

- Select the target memory.
- Select the files to be written, and write them to the standard ROM or Flash card.

(3) Additions/changes to standard ROM or Flash card files

Since all files stored in the standard ROM or Flash card are automatically deleted when files are to be written to the standard ROM or Flash card, additions/changes to the stored files cannot be made directly.

Use the following procedures.

(a) When [Write the program memory to ROM...] of GX Developer was used for writing

- Choose [Online] → [Read from PLC] on GX Developer and read all files of the program memory.
- Make necessary additions/changes to the read files.
- Write the modified files to the program memory.
- Choose [Online] → [Write to PLC (Flash ROM)] → [Write the program memory to ROM...], and write the files to the standard ROM or Flash card.

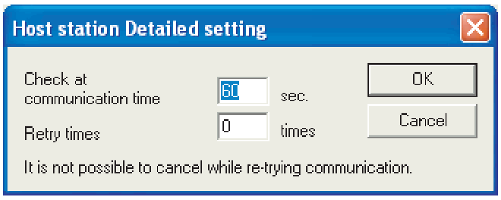
(b) When [Write to PLC (Flash ROM)] of GX Developer was used for writing

- Choose [Online] → [Read from PLC] on GX Developer and read all files of the standard ROM or Flash card.
- Make necessary additions/changes to the read files.
- Choose [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)], and write the modified files to the standard ROM or Flash card.

(4) Precautions

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

Since long processing time is required for writing files to the standard ROM or Flash card, set the GX Developer's check at communication time to 60 seconds or more. If the time for the check at communication time is too 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 5.30 Check at communication time

(b) Writing files from GX Developer on another station via CC-Link

Since long processing time is required for writing files to the standard ROM or Flash card, set the CPU monitoring time setting (SW000A) of CC-Link to 60 seconds or more.
(The default value of 90 seconds can be used.)

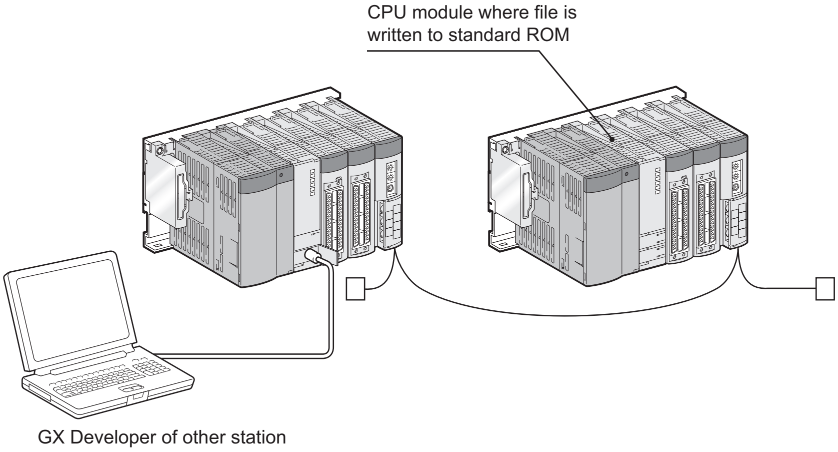

CPU module where file is written to standard ROM
GX Developer of other station

Figure 5.31 Writing files from GX Developer on another station (via CC-Link)

Remark

Refer to the following manual for details on the communication time check period ("Check at communication time") setting and the CPU monitoring time ("Watchdog time") setting.

☞ GX Developer Version 8 Operating Manual

(c) Time required for write to PLC (Flash ROM)

Using the Write to PLC (Flash ROM) writes data over the entire space of the standard ROM or Flash card.

Therefore, even if a program written to the Flash card has a small number of steps, long time will be taken until completion since data are written to the entire space of the Flash card.

(Approx. 14 minutes will be taken when writing to the Q2MEM-4MBF at a communication speed of 115.2kbps using an RS-232 interface.)

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

Long communication time will be also required for the write to PLC (Flash ROM) from the other station.

(d) Online change (enabled for write to PLC (Flash ROM) only)

Write to PLC (Flash ROM) can be executed online.

In either of the following cases, however, execute the write to PLC (Flash ROM) after setting the PLC to the STOP status.

1) The file registers on the Flash card are used in the sequence program.

2) File registers are set to "Not used" in the PLC parameter dialog box but they are used in the sequence program.

If write to PLC (Flash ROM) is performed in either of above 1) or 2), an error may occur, stopping the High Performance model QCPU, Process CPU or Redundant CPU.

(e) Write/read data during write to PLC (Flash ROM)

While writing to PLC (Flash ROM) is in execution, writing/reading data to/from other modules cannot be executed.

Therefore, timeout may occur in the other modules.

(f) When write to PLC (Flash ROM) is performed in STOP status

When write to PLC (Flash ROM) is performed in the STOP status, do not select the RUN status during writing.

The CPU module cannot be turned to RUN during write to PLC (Flash ROM).

Set it to RUN after completion of write to PLC (Flash ROM).

High Performance Note5.9

Universal Note5.10

5.2.8 Automatic all data write from memory card to standard ROM

(1) Definition of automatic all data write from memory card to standard ROM

The automatic all data write from memory card to standard ROM (hereafter abbreviated to the automatic write to standard ROM) is the function that automatically writes the parameters and programs previously written to a memory card to the standard ROM.

As shown in Figure 5.32, the automatic write to standard ROM function boots the parameters, sequence programs, etc. from the memory card to the program memory and writes the booted ones from the program memory to the standard ROM.

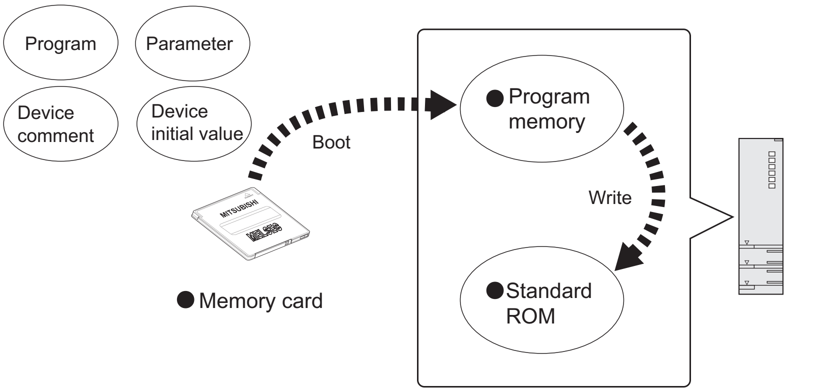


Figure 5.32 Automatic write to standard ROM

(2) Applications of automatic write to standard ROM

Using the automatic write to standard ROM, the parameters and programs written to the memory card in advance can be written to the standard ROM without use of GX Developer (personal computer).

This function is convenient for the following cases.

- When writing the same parameters and programs to multiple CPU modules.
- When creating the same environment at a remote site.

High Performance Note5.9

When using the automatic write to standard ROM on the High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.2)

When the memory card set for the automatic write to standard ROM is installed in the High Performance model QCPU incompatible with the function, it performs boot run from the standard ROM.

Universal Note5.10

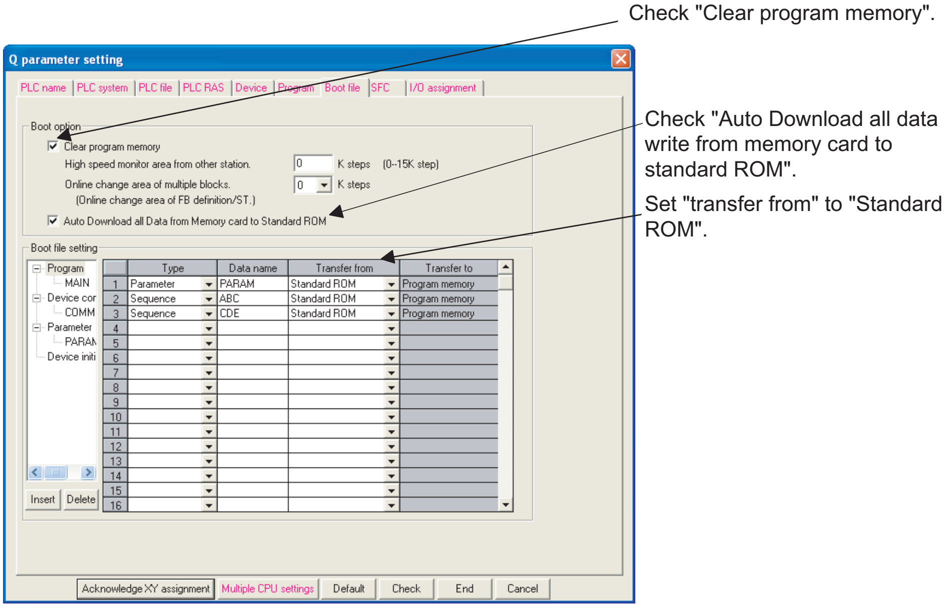
The Universal models QCPU cannot Automatic all data write from memory card to standard ROM.

(3) Execution procedure for automatic write to standard ROM

Perform automatic write to standard ROM in the following procedure.

(a) Operation on GX Developer (Setting of automatic write to standard ROM)

1) In the boot file setting of the PLC parameter dialog box, check "Clear the program memory" and "Auto Download all data from memory card to standard ROM".
In the boot file setting, set the parameters, programs, etc. to be booted.
(Set the "transfer from" to "Standard ROM".)



Check "Clear program memory".

Check "Auto Download all data write from memory card to standard ROM".

Set "transfer from" to "Standard ROM".

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				

Figure 5.33 Setting of Auto Download all data from memory card to standard ROM

2) Store the set parameters and programs to be booted into the memory card.
(☞ Section 5.2.6(4))

(b) Operation on High Performance model QCPU, Process CPU or Redundant CPU (Automatic write to standard ROM)

1) Power OFF the PLC.

2) Install the memory card, which stores the parameters and programs to be booted, into the High Performance model QCPU, Process CPU or Redundant CPU.

3) Set to the DIP switches of the High Performance model QCPU, Process CPU or Redundant CPU to positions indicating a parameter for valid drive.

- When SRAM card is installed••••••• SW2 : ON,SW3 : OFF
- When Flash/ATA card is installed •• SW2 : OFF,SW3 : ON

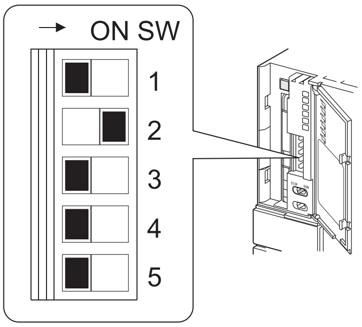


Figure 5.34 DIP switch settings at the time of SRAM card installation

4) Power ON the PLC.
Boot the files specified for the memory card to the program memory.
After completion of boot, write the program memory contents to the standard ROM.

5) Upon completion of the automatic write to standard ROM, the BOOT LED and ERR. LED flicker and the High Performance model QCPU, Process CPU or Redundant CPU is placed in a stop error status.

6) Power OFF the PLC.

7) Remove the memory card, and set the DIP switches for valid drive of the High Performance model QCPU, Process CPU or Redundant CPU to the standard ROM setting.

- Standard ROM•••••SW2 : ON,SW3 : ON

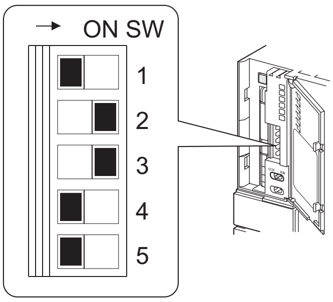


Figure 5.35 When parameter-valid drive DIP switches are set to standard ROM

8) Power ON the PLC.
When the PLC is powered ON, data are booted from the standard ROM to the program memory and actual operation is enabled.

(4) Precautions

The following provides the precautions for the automatic write to standard ROM.

(a) When file of the same file name exists in the program memory

When the program memory has a file whose name is the same as that of the file to be booted from the memory card, that file is overwritten by the file data of the memory card.

When the program memory does not have a file whose name is the same as that of the file to be booted from the memory card, the file is added to the program memory.

If the program memory capacity is exceeded at this time, "FILE ERROR (error code: 2401)" will occur.

(b) Program memory clear at boot

When files are to be booted from the memory card to the program memory, whether boot will be executed after program memory clear or without program memory clear can be selected.

Selecting boot after clearing the program memory clear for the automatic write to standard ROM prevents the program memory capacity from being exceeded at a boot.

(c) Parameter-valid drive DIP switch setting for use of this function

The "Automatic all data write from memory card to standard ROM" setting becomes valid only when the parameter-valid drive DIP switches are set to the memory card.

Hence, when actual operation is started after completion of the automatic write to standard ROM, the check mark in the "Automatic all data write from memory card to standard ROM" check box in the boot file setting need not be removed.

5.2.9 Execution of standard ROM/memory card programs (boot run)

Universal UD Note5.11

This section explains how to operate the programs stored in the standard ROM Note5.11 or memory card.

(1) How to execute the standard ROM/memory card programs

The CPU module operates the programs stored in the program memory.
It does not operate the programs stored in the standard ROM or memory card.
To do this, make setting to boot (read) the programs in the standard ROM or memory card to the program memory at power-on or reset.
(☞ (3), (4) below)

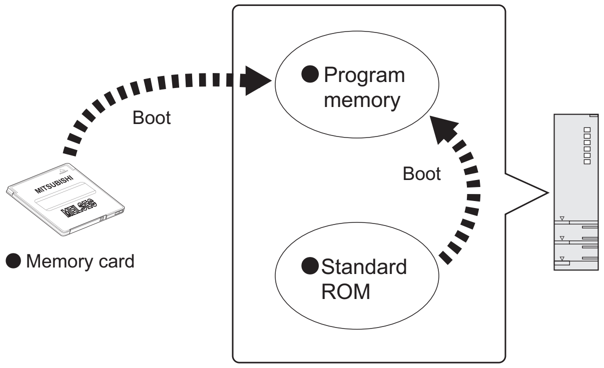


Figure 5.36 Boot run

The programs having the file names specified in the boot file setting are booted from the standard ROM or memory card to the program memory at power-on or reset.

Universal UD Note5.11

In the Universal model QCPU, a program cannot be executed from the standard ROM.

(2) Types of files can be booted and a transfer source and transfer destination

The file can be booted, its transfer source or transfer destination depends on the CPU module.

The following table shows the executable combination for boot run.

(a) High performance model QCPU, Process CPU, and Redundant CPU

Table5.9 Available files for boot run

File name	Transfer source		Transfer destination
	Memory card	Standard ROM	
Parameter	○	○	Program memory
Sequence program	○	○	
Device comment	○	○	
Device initial value	○	○	

○ : Boot available, × : Boot not available

(b) Universal model QCPU

Table5.10 Available files for boot run

File name	Transfer source	Transfer destination	
		Program memory	Standard ROM
Parameter	Memory card	○	○
Sequence program		○	×
Device comment		○	○
Device initial value		○	○
Label program		○	○

○ : Boot available, × : Boot not available

(3) Procedure before boot run

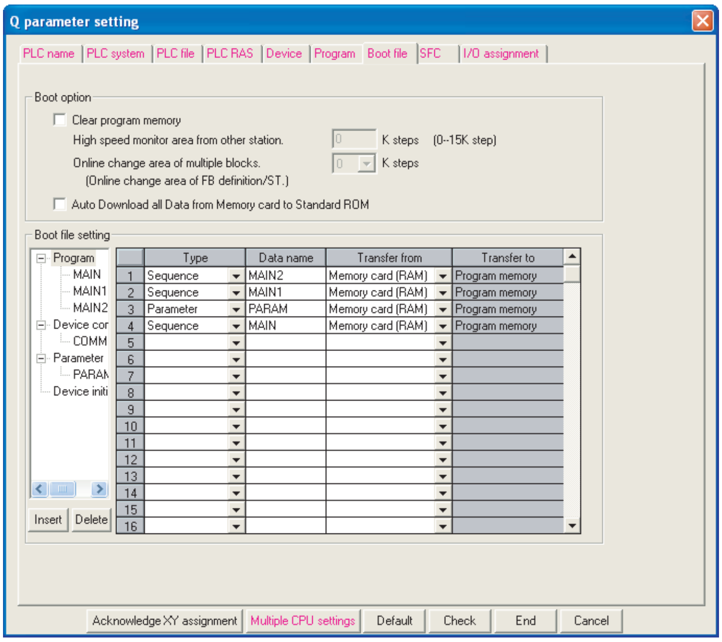
The following describes the procedure before boot run.

(a) Program creation by GX Developer

Create programs for boot run.

(b) Boot file by GX Developer

To execute the programs in the standard ROM or memory card, set the names of files to be booted (read) to the program memory in the boot file 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

- Program
  - MAIN
  - MAIN1
  - MAIN2
- Device cor
  - COMM
- Parameter
  - PARAN
- Device initi

	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 5.37 Boot file

(c) Hardware setting for boot run

Universal UD Note5.12

Set the parameter-valid drive DIP switches to indicate the memory that will store the parameters. Note5.12

When executing boot run, set the parameter storage location to the standard ROM or memory card. (☞ (5)(a)in this section)

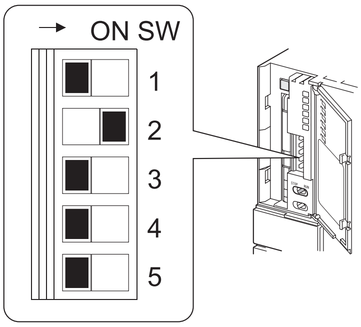


Figure 5.38 When parameter-valid drive DIP switches are set to SRAM card

Universal UD Note5.12

The Universal model QCPU does not require hardware settings for boot run.

(d) **Installation of memory card**

When storing files to a memory card by boot run, install the memory card into the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU.

(e) **Writing parameters and programs by GX Developer**

1) **High Performance model QCPU, Process CPU, Redundant CPU**

Write the parameters to the memory set by the parameter-valid drive DIP switches.

Also, write the files set in the boot file setting in (b) to the transfer source memory.

2) **Universal model QCPU**

Write the files set in the boot file settings and parameters to a memory card.

(f) **Program execution**

Reset the CPU module using the RESET/L.CLR switch (RUN/STOP/RESET switch in the case of the Universal model QCPU).

The BOOT LED turns on upon completion of the boot from the specified memory.

(g) **Checking whether boot is normally completed**

Whether a boot is normally completed or not can be checked by the following.

- The BOOT LED turns on.
- The special relay (SM660) turns ON.
- Choose [Online] → [Verify with PLC] on GX Developer and verify the data written to the transfer source memory and the data in the program memory.

(4) Operation for stopping boot run

To stop boot run and execute operation using the parameters and program files written to the program memory, perform the following operation by GX Developer.

(a) High Performance model QCPU, Process CPU, or Redundant CPU

1) Write the parameters, which are not set as a boot file, to the program memory.
2) Set the parameter-valid drive DIP switches of the CPU module to the "program memory". (SW2: OFF, SW3: OFF)

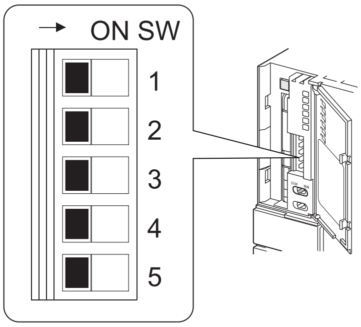


Figure 5.39 When parameter-valid drive DIP switches are set to program memory

3) Power on the PLC again or reset the CPU module.

(b) Universal model QCPU

1) Remove the memory card and write the parameters for which boot file is not set to the program memory.
2) Power on the PLC again or reset the CPU module.

Redundant Universal

Note5.13 Note5.13

(5) Changing the program file during RUN Note5.13

(a) Changing method

During RUN of the High Performance model QCPU, Process CPU or Universal model QCPU, files can be added, changed and deleted from the standard ROM or memory card to the program memory with the following instructions in a sequence program.

- 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)

Refer to the following manual for details of the PLOADP instruction, PUNLOADP instruction and PSWAPP instruction.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

(b) Program setting made when program file is changed

Even if a program file has been changed during RUN of the High Performance model QCPU, Process CPU or Universal model QCPU, the program setting in the PLC parameter dialog box is not changed.

Hence, when the High Performance model QCPU, Process CPU or Universal model QCPU is set to a STOP status, correct the program setting of the PLC parameter dialog box to the current settings made during RUN of the High Performance model QCPU, Process CPU or Universal model QCPU (addition/change/deletion of program name).

Otherwise, an error will occur when the CPU is switched from STOP to RUN status.

Redundant Universal

Note5.13 Note5.13

Redundant CPU and Universal model QCPU cannot change the program file during RUN using the PLOADP instruction, PUNLOADP instruction, and PSWAPP instruction.

(6) Precautions when executing the programs in the standard ROM/memory card

(a) Parameter storage location

1) High Performance model QCPU, Process CPU and Redundant CPU

When executing boot run, store the parameters (PLC parameters) specified in the boot file setting to the standard ROM or memory card.

When the parameters are stored in the program memory, the CPU module ignores the boot file settings of PLC parameter even if the parameter-valid drive of DIP switches are set to the program memory. A boot is not executed when the PLC is powered on or RESET status of the CPU module is cleared.

2) Universal model QCPU

When executing boot run, store the parameters (PLC parameters) specified in the boot file setting to the memory card.

When executing boot run, store the parameters (PLC parameters) specified in the boot file setting to the memory card.

The CPU module ignores the boot file settings of PLC parameter if the parameters are stored in the program memory or standard ROM. A boot is not executed when the PLC is powered on or RESET status of the CPU module is cleared. (☞ Section 5.2.11)

When the following conditions a) and b) are met, the CPU module is not operated by the parameters in the memory card, but operated by the parameters in the program memory.

a) Parameters exist in the program memory.

b) The parameters in the memory card are not set as a boot file.

(b) Online change during boot run

1) Memory card (RAM)

When online change (☞ Section 6.12) is made to the program in the program memory, the change can be updated to the program of the boot source memory card (RAM).

2) Standard ROM/memory card (ROM)

Even if online change has been made to the program in the program memory, the change is not updated to the program in the boot source standard ROM/ memory card (ROM).

Therefore, write the same file as the one written to the program memory to the standard ROM/memory card (ROM) by performing the write to PLC (Flash ROM). (☞ Section 5.2.7)

(c) Maximum number of boot files that can be set

The maximum number of boot files that can be set in the boot file setting of the PLC parameter dialog box should be the same as the number of files that can be stored into the program memory.

However, in the following cases, the number of boot files decreases by 1 for each.

- When a heading is set
- When the PLC parameters (set in the boot file setting) stored in the standard ROM/memory card are booted

(d) Boot run executed when ATA card is used

When boot run is executed in either of the following states, a maximum of 200ms may be taken per 1k step (4k bytes) at a boot.

- When boot is executed from the ATA card
- When boot is executed from the standard ROM with the ATA card installed

(e) When data in the program memory are changed at power OFF → ON or reset

If the data in the program memory are changed when the programmable controller is powered OFF → ON or the reset operation of the CPU module is performed after sequence programs are written to the program memory, it may be considered that the boot operation is still active.

The boot operation is active if the BOOT LED on the front of the High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU is on.

Refer to (4) in this section and stop the boot operation.

(f) Capacity after boot from memory card

The size unit of the file stored in each memory differs between the memory card and the program memory. (☞ Section 5.4.4)

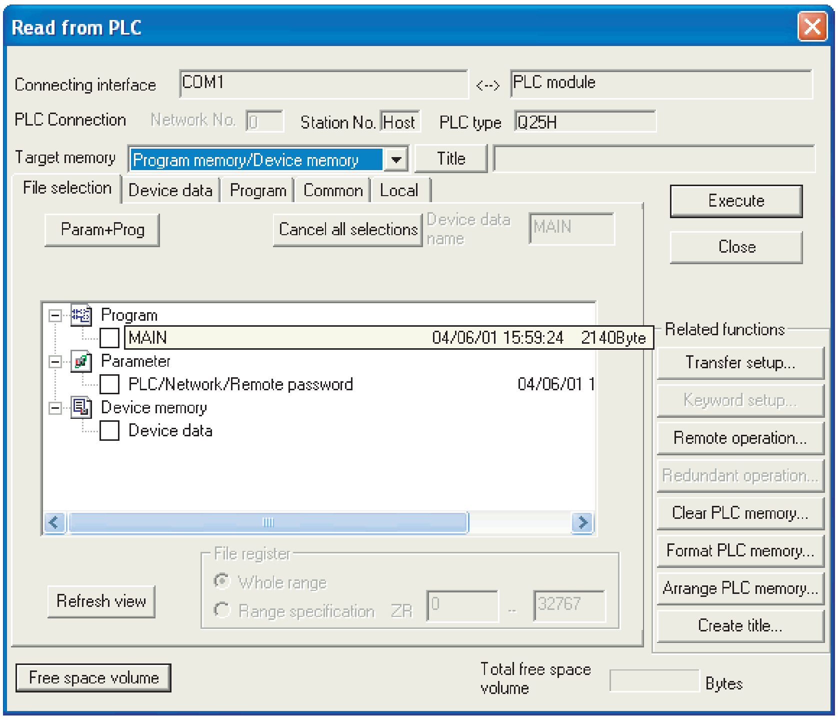
Therefore, note that the file transferred from the memory card to the program memory changes in memory capacity between before and after the transfer.

(g) Program written to memory card

The PLC type (the model of the CPU module) of programs written to the memory card (programs set in the boot file settings) and the model of the CPU module targeted for boot operation must be same.

5.2.10 Details of written files

The file name, file size, written date and time, etc. set when creating a file by GX Developer are added to each file written to the CPU module.
Choose [Online] → [Read from PLC] on GX Developer and monitor the files. The files are displayed as shown below.



Read from PLC

Connecting interface: COM1 <--> PLC module
PLC Connection: Network No. 0 Station No. Host PLC type Q25H
Target memory: Program memory/Device memory

- Program
  - □ MAIN 04/06/01 15:59:24 2140Byte
- Parameter
  - □ PLC/Network/Remote password 04/06/01 1
- Device memory
  - □ Device data

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

Total free space volume Bytes

Figure 5.40 Display of file details

(a) File name

1) File name structure and file specification

Each file has a file name (up to 8 single-byte characters/4 double-byte characters) and extension (3 single-byte characters).
Create file name with capital letters only.
Extensions are automatically added depending on types set at the time of creating a file.

2) Characters unavailable for file names

The following words reserved for MicrosoftO WindowsO cannot be used for file names.

- COM1 to COM9
- LPT1 to LPT9
- AUX
- CON
- PRN
- NUL
- CLOCK$

3) **File name in a sequence program**
How to specify a file name in a sequence program differs depending on the CPU module.

- For High Performance model QCPU, Process CPU, and Redundant CPU
Since the single-byte capital letters are distinguished from the single-byte small letters, specify the file name with capital letters.
* "ABC" and "abc" are distinguished.
- For Universal model QCPU
Since the single-byte small letters and the single-byte capital letters are not distinguished, specify the file name with either letters.
* Both "ABC" and "abc" are treated as "ABC".

Remark

Since the double-byte capital letters are distinguished from the double-byte small letters, specify the file name with the capital letters.
In double-byte letters, "ABC" and "abc" are distinguished.

(b) **Date, time**
The date and time when a file was written to the CPU module by GX Developer. Note that the indicated date and time are those set on GX Developer (personal computer).

(c) **Size**
The file size of when it was written to the CPU module using GX Developer is displayed in units of bytes. (Click the Refresh view button on the screen displayed by selecting [Online] → [Read from PLC] to display the latest data.)
At least 64 bytes (136 bytes for program files) are added to the file size created by users, except for file register files.

5.2.11 Specifying valid parameters (Parameter-valid drive setting)

The parameter-valid drive setting function specifies the drive (memory) in which parameters to be valid are stored.
The method for specifying the parameter-valid drive differs depending on the CPU module.

(1) High Performance model QCPU, Process CPU and Redundant CPU

Specify the drive with the DIP switches of the CPU module. (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(a) Parameter-valid drive specification method

Specify a parameter-valid drive in the setting of the DIP switches (SW2, SW3) on the CPU module.

Table5.11 Drive specification by 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 3 (Standard RAM)

(b) Valid parameter determination timing

The valid parameters are determined at the following timing.

- When the programmable controller is powered ON (at power ON)
- When the reset operation of the CPU module is performed (at reset)

The CPU module validates the parameters in the drive specified by SW2 and SW3 at the timing above, and operates using the valid parameters.

(2) Universal model QCPU

The system automatically determines and specifies the parameter-valid drive. Users cannot specify the parameter-valid drive.

(a) Parameter-valid drive specification method

The valid parameters are decided by the order of priority set to the drives where parameters are stored.

Table5.12 shows the priority order of the drives.

Write the parameters need to be valid to the drive.

Table5.12 Priority order of the drives

Priority		Parameter-valid drive
High	1	Drive 0 (Program memory)
↑	2	Drive 1 (Memory card RAM)
↓	3	Drive 2 (Memory card ROM)
Low	4	Drive 3 (Standard ROM)

The following describes the operation flowchart for the CPU module to specify a parameter storage drive.

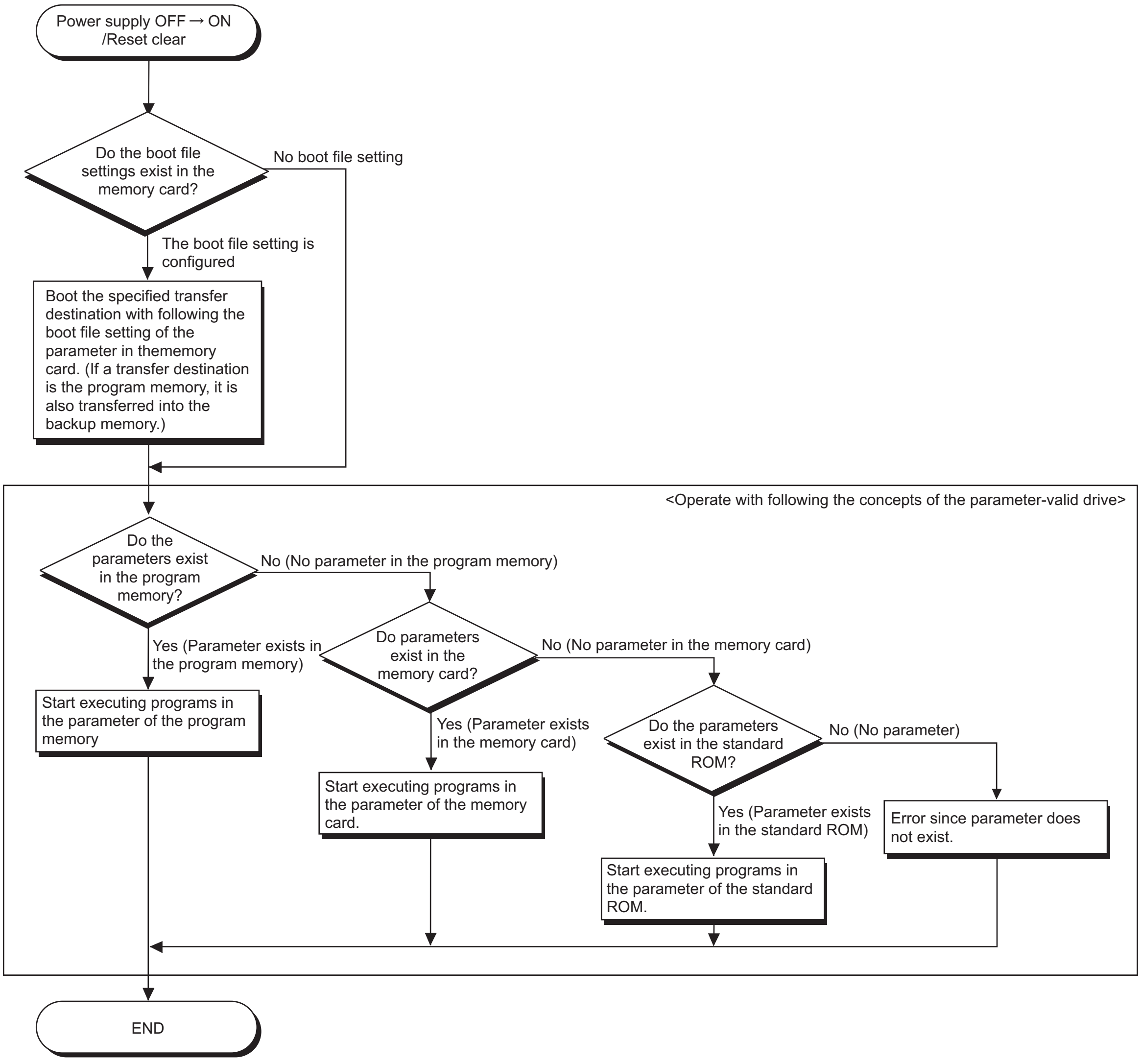

Power supply OFF → ON /Reset clear
Do the boot file settings exist in the memory card?
No boot file setting
The boot file setting is configured
Boot the specified transfer destination with following the boot file setting of the parameter in thememory card. (If a transfer destination is the program memory, it is also transferred into the backup memory.)
<Operate with following the concepts of the parameter-valid drive>
Do the parameters exist in the program memory?
No (No parameter in the program memory)
Yes (Parameter exists in the program memory)
Start executing programs in the parameter of the program memory
Do parameters exist in the memory card?
No (No parameter in the memory card)
Yes (Parameter exists in the memory card)
Start executing programs in the parameter of the memory card.
Do the parameters exist in the standard ROM?
No (No parameter)
Yes (Parameter exists in the standard ROM)
Start executing programs in the parameter of the standard ROM.
Error since parameter does not exist.
END

Figure 5.41 Operation flowchart to specify a parameter storage drive

(b) **Valid parameter determination timing**

The valid parameters are determined at the following timing.

- When the programmable controller is powered ON (at power ON)
- When the reset operation of the CPU module is performed (at reset)

The CPU module automatically searches for the parameters at the timing above, and operates with the parameter setting of the drive where the detected parameters are stored.

When storing the parameters using the Write to PLC operation of GX Developer, the valid parameter determination timing differs depending on the drive where parameters are stored.

1) **When parameters are stored to the drive different from the one where the parameters in operation are stored.**

Parameters are validated according to the order of priority set to the drives at power ON or reset.

2) **When parameters are stored to the drive same as the one where the parameters in operation are stored.**

Only the Device setting in parameters are validated at the time of completion of the Write to PLC operation.

To validate all parameter settings, turn the power OFF and then ON, or perform the reset operation.

Memo

5.3 Program File Structure

A program file consists of a file header, execution program and allocate memory for online change.

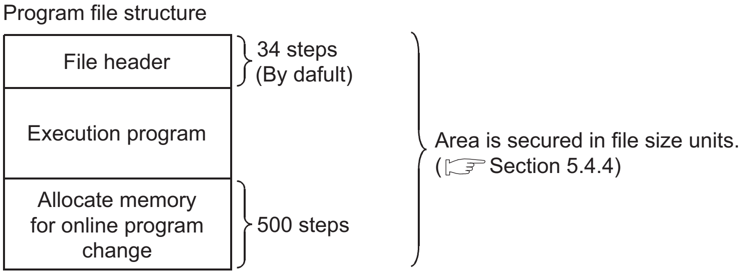


Figure 5.42 Program file structure

(1) Structure details

The capacity of the program stored in the program memory of the CPU module is the total of the above three areas.

(a) File header

This area stores the file name, size, creation date, etc.
The file header size ranges from 25 to 35 steps (100 to 140 bytes) depending on the device setting of the PLC parameter dialog box.
(Default: 34 steps)

(b) Execution program

This area stores the created program.

(c) Allocate memory for online change

This area is used when online change that increases the number of steps is performed by GX Developer.
When such an online change is performed by GX Developer, the number of remaining allocate memory for online change is displayed.

1) Default number of allocate memory for online change

The default setting is 500 steps (2000 bytes).

2) Changing the number of allocate memory for online change

The number of allocate memory for online change can be changed by GX Developer (by choosing [Online] → [Write to PLC] →<Program>, tab).
When the number of steps is insufficient for online change, it can be set again.
(☞ Section 6.12.1)

(2) Display of program capacity by GX Developer

During programming by GX Developer, the program capacity (sum of the file header capacity and the numbers of steps in the created program) is displayed in terms of the number of steps as shown in Figure 5.43.

When a program is created, the capacity of the crea.ted program can be confirmed.

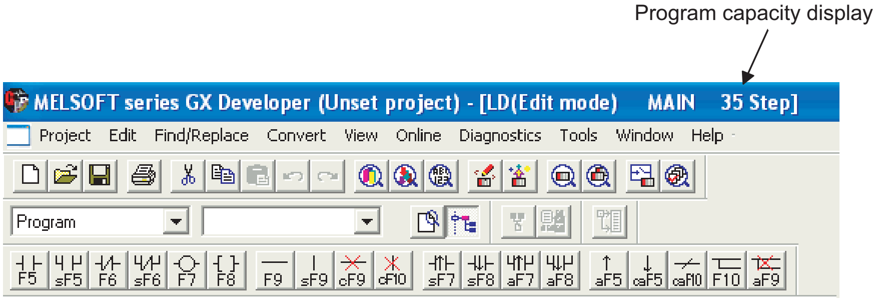
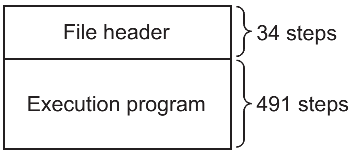

Program capacity display

Figure 5.43 Program capacity display

☒POINT

1. The program capacity displayed during programming by GX Developer is the capacity of the file header and execution program and does not include the capacity of the allocate memory for online change (500 steps).

   (Example) The capacity of the program having the execution program area of 491 steps is displayed on GX Developer as shown below. (The file header is fixed to 34 steps.)



Display on GX Developer:
34 steps + 491 steps = 525 steps.

Figure 5.44 File status on GX Developer

2. Since a file is stored in file size units on the program memory, the program capacity displayed during programming by GX Developer may differ from the capacity of the program file on the CPU module. (☞ Section 5.4.4)

5.4 File Operation by GX Developer and Handling Precautions

5.4.1 File operation

Basic Note5.14

Basic Note5.15 Universal UD Note5.15

The file operations shown in Table5.14 can be performed for the files stored in the program memory, standard ROM or memory card*Note5.14* by the online operation of GX Developer. However, the executable file operations varies depending on the password registration made by GX Developer, the system protect switch*Note5.15* status of the CPU module, and the RUN/STOP status of the CPU module.

Table5.13 List of file operations that can be executed from GX Developer

File operation	Operation details	Operability*1 (Basic model QCPU)			Operability*1 (High Performance model QCPU)				
		A	B	C	A	B	C	D	
Read from PLC	Reads files from the target memory.	○	△	○	○	△	○	○	
Write to PLC	Writes files to the program memory or SRAM card.	△	△	×	△	△	○	×	
	Writes files to the standard ROM.	×	×	×	×	×	×	×	
Verify with PLC	Verifies the target memory and GX Developer files.	△	△	○	△	△	○	○	
Write the program memory to ROM	Batch-writes the 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 the specified files to the Flash card by GX Developer.	×	×	×	△	△	○	×	
	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 the memory.	○	○	×	○	○	×	×	
Arrange PLC memory	Rearranges the discontinuous files on the memory.	○	○	×	○	○	×	×	
Online change in ladder mode	Writes the contents changed in the ladder mode to the program memory.	△	△	○	△	△	○	×	

Basic Note5.14

The Basic model QCPU does not support the use of a memory card.

Basic Note5.15 Universal UD Note5.15

The Basic model QCPU and Universal model QCPU do not support the system protect switch.

	Operability*1 (Process CPU)				Operability*1 (Redundant CPU)				Operability*1 (Universal model QCPU)		
	A	B	C	D	A	B	C	D	A	B	C
	○	△	○	○	○	△	○	○	○	△	○
	△	△	○	×	△	△	○	×	△	△	○
	×	×	×	×	×	×	×	×	△	△	○
	△	△	○	○	△	△	○	○	△	△	○
	○	○	○	×	○	○	○	×	○	○	○
	○	○	○	×	○	○	○	×	×	×	×
	△	△	○	×	△	△	○	×	△	△	○
	△	△	○	×	△	△	○	×	×	×	×
	△	△	×	×	△	△	×	×	△	△	×
	○	○	×	×	○	○	×	×	○	○	×
	○	○	×	×	○	○	×	×	○	○	×
	△	△	○	×	△	△	○	×	△	△	○

○ : Can be executed, △ : Can be executed on password match, × : Cannot be executed

* 1 : The following table indicates the meanings of the symbols in the operability field.

Symbol	Description
A	When a write inhibit password is set to the file
B	When a read/write inhibit password is set to the file
C	When the CPU module is in a RUN status
D	When the system protect switch of the CPU module is ON

5.4.2 Precautions for handling files

(1) Power OFF (including reset) during file operation

When the PLC is powered OFF or the CPU module is reset during any file operation, the files of the corresponding memory will be as described in Table5.14.

Table5.14 File status at power OFF during file operation

CPU module	Memory status
Basic model QCPU	The files of the corresponding memory may be corrupted.
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	The files of the corresponding memory are not corrupted. (When the memory card is used, it is assumed that the PLC is powered ON without removal of the memory card from the CPU module in power OFF status.)

☒POINT

When the PLC is powered OFF during operation that transfers file(s), 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) Simultaneous write from multiple GX Developers to the same file

The CPU module does not allow access from other GX Developers to the file being written.

Also, it does not allow data to be written from other GX Developers to the file being accessed.

When writing data from multiple GX Developers to the same file, start the processing of next GX Developer after the processing of one GX Developer is completed.

(3) Simultaneous access from multiple GX Developers to different files

The CPU module allows simultaneous access from other GX Developers to a maximum of 10 different files of the same CPU module.

5.4.3 Memory capacities of files

The sizes of the files used by the CPU module change depending on their types.
This section indicates the memory capacities of the files for each CPU module.
When files are written to the memory area, the units of the storage capacities change depending on the target CPU module and memory area. (☞ Section 5.4.4)

(1) When Basic model QCPU is used

When using the program memory, standard RAM or standard ROM, calculate the rough size of each file according to Table5.15.

Table5.15 Memory capacity calculation for files (Basic model QCPU)

<table>
<tr><th>Function</th><th>Rough file capacity (unit: byte)</th></tr>
<tr><td>Drive heading</td><td>64</td></tr>
<tr><td>Parameter</td><td>Default: 522 (increases depending on the parameter setting)
Reference
Boot setting → 94
CC-Link IE controller network setting made → Max. 7214 increased
MELSECNET/H setting made → Max. 6180 increased
Ethernet setting made → Max. 922 increased
CC-Link setting made → Max. value of the table below increased
(The values of the table below indicate the 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>module 1</td><td>550byte</td><td>572byte</td><td>624byte</td></tr>
<tr><td>module 2 to module 4</td><td>536byte</td><td>558byte</td><td>610byte</td></tr>
<tr><td>module 5</td><td>550byte</td><td>566byte</td><td>618byte</td></tr>
<tr><td>module 6 to module 8</td><td>536byte</td><td>558byte</td><td>610byte</td></tr>
</table>
Remote password setting made → 64 + 20 + (number of target modules × 10), max. 164 increased</td></tr>
<tr><td>Sequence program</td><td>136[*1]+ (4 × ((number of steps) + (number of allocate memory for online change)))</td></tr>
<tr><td>Device comment</td><td>74 + (sum of comment data sizes of devices)
• Comment data size of one device = 10 + 10250 × a + 40 × b
• a : Quotient of ((device points)/256)
• b : Remainder of ((device points)/256)</td></tr>
<tr><td>File register</td><td>2 × (number of file register points)</td></tr>
<tr><td>Device initial value</td><td>66 + 44 × n + 2 × (total number of device points set for the device initial value)
• n : Number of set device initial values</td></tr>
<tr><td>Intelligent parameter</td><td>68 + (24 × number of set modules) + parameter sizes of utilities</td></tr>
<tr><td>User setting area</td><td>Value set at formatting (0 to 3k)</td></tr>
<tr><td>Multi-block online program change</td><td>Value set at formatting (0/1.25k/2.5k)</td></tr>
</table>

* 1 : 136 is the default value. (It can be increased or decreased by parameter setting.)

(2) When High Performance model QCPU, Process CPU, Redundant CPU is used

When using the program memory, standard RAM, standard ROM or memory card, calculate the rough size of each file according to Table5.16.

Table5.16 Memory capacity calculation for files (High Performance model QCPU, Process CPU, Redundant CPU)

<table>
<tr><th>Function</th><th>Rough file capacity (unit: byte)</th></tr>
<tr><td>Drive heading</td><td>64</td></tr>
<tr><td>Parameter</td><td>Default (increases depending on the parameter setting)
<table>
<tr><th>CPU module</th><th>File capacity</th></tr>
<tr><td>High Performance model QCPU</td><td>564</td></tr>
<tr><td>Process CPU</td><td>558</td></tr>
<tr><td>Redundant CPU</td><td>764</td></tr>
</table>
Reference
Boot setting → 70 + (18 × (number of files))
CC-Link IE controller network setting made → Max. 7214/module increased
MELSECNET/H setting made → Max. 6180/module increased
Ethernet setting made → Max. 922/module increased
CC-Link setting made → Max. value of the table below increased
(The values of the table below indicate the 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>module 1</td><td>550byte</td><td>572byte</td><td>624byte</td></tr>
<tr><td>module 2 to module 4</td><td>536byte</td><td>558byte</td><td>610byte</td></tr>
<tr><td>module 5</td><td>550byte</td><td>566byte</td><td>618byte</td></tr>
<tr><td>module 6 to module 8</td><td>536byte</td><td>558byte</td><td>610byte</td></tr>
</table>
Remote password setting made → 64 + 20 + (number of target modules × 10),
max. 164 increased</td></tr>
<tr><td>Sequence program</td><td>136*1+ ((4 × (number of steps) + (number of allocate memory for online change)))</td></tr>
<tr><td>Device comment</td><td>74 + (sum of comment data sizes of devices)
• Comment data size of one device = 10 + 10250 × a + 40 × b
• a : Quotient of ((device points)/256)
• b : Remainder of ((device points)/256)</td></tr>
<tr><td>Device initial value</td><td>66 + 44 × n + 2 × (total number of device points set for the device initial value)
• n : Number of set device initial values</td></tr>
<tr><td>User setting area</td><td>Value set at formatting (0 to 15k)</td></tr>
<tr><td>Multi-block online program change</td><td>Value set at formatting (0/2k/4k)</td></tr>
<tr><td>File register</td><td>2 × (number of file register points)</td></tr>
</table>

(Continued on next page)

Table5.16 Memory capacity calculation for files (High Performance model QCPU, Process CPU, Redundant CPU) (Continued)

Function	Rough file capacity (unit: byte)
Sampling trace file	362 + (number of word device points + number of bit device points) × 12 + (N1 +N2 + N3 + number of word device points × 2 + (number of bit device points/16) × 2) × number of traces (total number of traces) *2 • The following values are added to the N1 to N3 corresponding to the items set in trace Additional information of trace condition settings screen. (☞ Section 6.14 (4)(b)) N1 : Add "4" at setting time N2 : Add "10" at setting step No. N3 : Add "8" at setting program name
Failure history data	72 + 54 × (number of failures stored)
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 the default value. (It can be increased or decreased by parameter setting.)
* 2 : Round up "number of bit device points/16" to the right of the decimal point.

Remark

Refer to Section 5.4.4 for the memory capacity calculation example.

(3) When Universal model QCPU is used

When using the program memory, standard RAM, standard ROM or memory card, calculate the rough size of each file according to Table 5.18
.

Table5.17 Memory capacity calculation for files (Universal model QCPU)

<table>
<tr><th>Function</th><th>Rough file capacity (unit: byte)</th></tr>
<tr><td>Drive heading</td><td>72</td></tr>
<tr><td>Parameter</td><td>Default: 464 (increases depending on the parameter setting)*1
Reference
Boot setting → 84 + (18 × (number of files))*2
CC-Link IE controller network setting made → 72+(Total of parameter sizes per module)+ (Routing setting size) + (Size setting of transfer between data links)
• Parameter size per module:
Up to 10368 (1826 + 16 × (Number of refresh transfer points) for setting LB/LW(1) only)
• Routing setting size: 6 + 8 × (Number of routing settings)
• Size setting of transfer between data links:
6+12 × (Number of transfersettings)+86 × (Number of modules)
MELSECNET/H setting made → Max. 6180/module increased
Ethernet setting made → Max. 922/module increased
CC-Link setting made → Max. value of the table below increased
(The values of the table below indicate the 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>module 1</td><td>550byte</td><td>572byte</td><td>624byte</td></tr>
<tr><td>module 2 to module 4</td><td>536byte</td><td>558byte</td><td>610byte</td></tr>
<tr><td>module 5</td><td>550byte</td><td>566byte</td><td>618byte</td></tr>
<tr><td>module 6 to module 8</td><td>536byte</td><td>558byte</td><td>610byte</td></tr>
</table>
Remote password setting made → 92 + (number of target modules × 10), max. 172 increased</td></tr>
<tr><td>Sequence program</td><td>148*3+ ((4 × (number of steps) + (number of allocate memory for online change)))</td></tr>
<tr><td>Device comment</td><td>74 + 8 + (sum of comment data sizes of devices)
• Comment data size of one device = 10 + 10250 × a + 40 × b
• a : Quotient of ((device points)/256)
• b : Remainder of ((device points)/256)</td></tr>
<tr><td>Device initial value</td><td>66 + 44 × n + 2 × (total number of device points set for the device initial value) + 8
• n : Number of set device initial values</td></tr>
<tr><td>User setting area</td><td>Value set at formatting (0 to 15k)</td></tr>
<tr><td>File register</td><td>2 × (number of file register points)</td></tr>
</table>

(Continued on next page)

Table5.16 Memory capacity calculation for files (Universal model QCPU) (Continued)

Function	Rough file capacity (unit: byte)
Sampling trace file	362 + (number of word device points + number of bit device points) × 12 + (N1 +N2 + N3 + number of word device points × 2 + (number of bit device points/16) × 2) × number of traces (total number of traces) *4 • The following values are added to the N1 to N3 corresponding to the items set in trace Additional information of trace condition settings screen. (☞ Section 6.14 (4)(b)) N1 : Add "4" at setting time N2 : Add "10" at setting step No. N3 : Add "8" at setting program name
File for device data backup	Value set at formatting (2 to 1024k)
Local device	70 + 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 using local device)*5 • 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 : The total number of bytes becomes multiple of four, which is adjusted by the system.
* 2 : The number of bytes becomes multiple of four, which is adjusted by the system.
* 3 : 148 is the default value. (It can be increased or decreased by parameter setting.)
* 4 : Round up "number of bit device points/16" to the right of the decimal point.
* 5 : For the Universal model QCPU whose first 5 digits of the serial number are "10011" or earlier, it will be the number of executing programs.

Remark

Refer to Section 5.4.4 for the memory capacity calculation example.

5.4.4 File size units

(1) What is file size unit?

When files are written to the memory area, the unit of storage capacity changes depending on the target CPU module and memory area.
This unit is called a file size unit.

(a) File size units classified by memory areas

The following table indicates the file size units classified by the target CPU modules and memory areas.

Table5.18 File size units of CPU modules (classified by memory areas)

Module name	Memory area	
	File size unit of program memory, standard ROM, Flash card[*1]	Standard RAM
Q00JCPU	1 step/4 bytes	----
Q00CPU, Q01CPU		4 byte
Q02CPU, Q02HCPU, Q06HCPU	128 steps/512 bytes[*2]	512 byte
Q12HCPU	256 steps/1024 bytes[*2]	1024 byte[*4]
Q25HCPU	512 steps/2048 bytes[*2]	
Q02PHCPU, Q06PHCPU	128 steps/512 bytes	512 byte
Q12PHCPU	256 steps/1024 bytes[*3]	1024 byte
Q25PHCPU	512 steps/2048 bytes[*3]	
Q12PRHCPU	256 steps/1024 bytes	
Q25PRHCPU	512 steps/2048 bytes	
Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU	program memory : 1 steps/4 bytes standard ROM, Flash card : 128 steps/512 bytes	512 byte
Q13UDHCPU, Q13UDEHCPU	program memory : 1 steps/4 bytes standard ROM : 256 steps/1024 bytes Flash card : 128 steps/512 bytes	
Q26UDHCPU, Q26UDEHCPU	program memory : 1 steps/4 bytes standard ROM : 512 steps/2048 bytes Flash card : 128 steps/512 bytes	

Basic
Note5.16

* 1 : The file size unit of the Flash card applies to the case where files are written to the Flash card by selecting [Online] → [Write to PLC (Flash ROM)] → [[Write the program memory to ROM] in GX Developer. (☞ Section 5.2.7(1)(a)) Note5.16

* 2 : 1024 steps/4096 bytes for the High Performance model QCPU whose first 5 digits of serial No. are "04121" or earlier.

* 3 : 1024 steps/4096 bytes for the Process CPU whose first 5 digits of serial No. are "07031" or earlier.

* 4 : 512 bytes for the Q12HCPU and Q25HCPU whose first 5 digits of serial No. are "02091" or earlier.

Basic
Note5.16

The Basic model QCPU does not support the use of a Flash card.

Basic
Note5.17

(b) File size units classified by memory cards Note5.17

Table5.19 File size units (classified by memory cards)

Type	Memory card model name	File size unit (cluster size)
SRAM card	Q2MEM-1MBS	512 bytes
	Q2MEM-2MBS	1024 bytes
	Q3MEM-4MBS*2	1024 bytes
	Q3MEM-8MBS*2	4096 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 applies to the following cases.
•When files are written to the Flash card by selecting [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] in GX Developer, (☞ Section 5.2.7(1)(b))
•When files are written to the Flash card directly using GX Developer without the CPU module being relayed.
* 2: The Basic model QCPU, High Performance model QCPU, Process CPU, and Universal model QCPU do not support the use of the Q3MEM-4MBS and Q3MEM-8MBS.

(2) Memory capacity calculation example

The following indicates an example of calculating the memory capacity when the parameter and sequence program files are written to the program memory.

(a) Conditions

1) Write target CPU module: Q25HCPU

2) Written files

Table5.20 File capacities

File name	File capacity*1
PARAM.QPA (Parameter file)	564 bytes
MAIN.QPG (Sequence program)	525 steps/2100 bytes*2

* 1 : Refer to Section 5.4.3 for the file capacity.
* 2 : Indicates the program capacity (file header + execution program) displayed by GX Developer. (☞ Section 5.3)

3) Allocate memory for online change: 500 steps/2000 byets

Basic
Note5.17

The Basic model QCPU cannot use the memory card.

(b) Memory capacity calculation

The memory capacity is calculated on the basis of the file size unit of the write target CPU module. (☞ (1) in this section)

The file size unit of the Q25HCPU in this example is 512 steps/2048 bytes according to (1) in this section.

1) Calculation of parameter file capacity

Although the parameter file capacity is 564bytes, the file occupies a capacity of 512 steps/2048 bytes since it is stored on the program memory in file size unit.

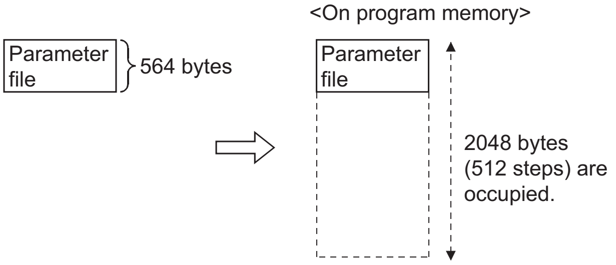


Figure 5.45 Capacity occupied by file stored in file size unit (Parameter file)

2) Calculation of program capacity

The program capacity is the sequence program capacity + allocate memory for online change

In this example, it is 525 steps + 500 steps = 1025 steps. However, the file occupies a capacity of 1536 steps/6144 bytes since it is stored on the program memory in file size unit (512-step unit for the Q25HCPU in this example).

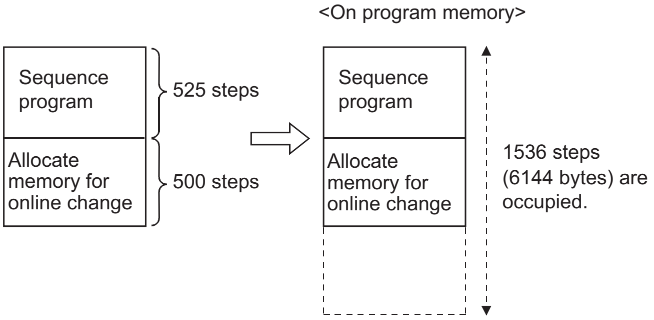

<On program memory>
Sequence program
525 steps
Allocate memory for online change
500 steps
Sequence program
Allocate memory for online change
1536 steps (6144 bytes) are occupied.

Figure 5.46 Capacity occupied by file stored in file size unit (Program file)

3) **Calculation result**

Table5.21 Memory capacity calculation result

File name	File capacity		Memory capacity
PARAM.QPA	564 bytes		512 steps (2048 bytes)
MAIN.QPG	Sequence program capacity	525 steps	1536 steps (6144 bytes)
	Allocate memory 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 whose first 5 digits of serial No. is "04122" or later.
- Process CPU whose first 5 digits of serial No. is "07032" or later.

Hence, note the following points.

1. Depending on the file capacity, the file executed by the CPU module of above serial No. or later may not be stored into the CPU module of above serial No. or before.
2. Combination of PLC write and GX Developer
   The following table indicates the combination of High Performance model QCPU/Process CPU and GX Developer versions when a file is read from the High Performance model QCPU/Process CPU to GX Developer and then written to another High Performance model QCPU/Process CPU.

(High Performance model QCPU)

Legends ◎ : Write to PLC enabled ○ : Restriction on write to PLC

Write destination	Write source			
	GX Developer Version8		GX Developer Version7	
	File executed by CPU of serial No. "04122" or later	File executed by CPU of serial No. "04121" or earlier	File executed by CPU of serial No. "04122" or later	File executed by CPU of serial No. "04121" or earlier
Serial No. "04122" or later	◎	◎	◎*2	○*2
Serial No. "04121" or earlier	○*1	◎	○*1 *2	○*2

* 1 : Because of different file size units, the file may not be stored into the High Performance model QCPU depending on the file capacity.

* 2 : Unless the number of allocate memory for online change is decreased, the file may not be stored into the High Performance model QCPU depending on the file capacity.

(Process CPU)

Legends ◎ : Write to PLC enabled ○ : Restriction on write to PLC

Write destination	Write source			
	GX Developer Version8		GX Developer Version7	
	File executed by CPU of serial No. "07032" or later	File executed by CPU of serial No. "07031" or earlier	File executed by CPU of serial No. "07032" or later	File executed by CPU of serial No. "07031" or earlier
Serial No. "07032" or later	◎	◎	◎*2	○*2
Serial No. "07031" or earlier	○*1	◎	○*1 *2	○*2

* 1 : Because of different file size units, the file may not be stored into the High Performance model QCPU depending on the file capacity.

* 2 : Unless the number of allocate memory for online change is decreased, the file may not be stored into the High Performance model QCPU depending on the file capacity.

CHAPTER6 FUNCTIONS

This chapter explains the functions of the CPU module.

6.1 Function List

Table6.2 lists the functions of the CPU module.

The numbers in the "CPU module" column correspond to the CPU modules as indicated in Table6.1.

Table6.1 Number in the "CPU module" column and corresponding CPU module

No.	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU
5)	Universal model QCPU

Table6.2 CPU module function list

Item	Description	CPU module					Reference
		1)	2)	3)	4)	5)	
Constant scan	Executes the program in a set time interval regardless of the program scan time.	○	○	○	○	○	Section 6.2
Latch function	Retains 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	Stops or executes operations in the CPU module.	○	○	○	○	○	Section 6.6.1
Remote PAUSE	Stops operations in the CPU module while retaining the status of output (Y).	○	○	○	○	○	Section 6.6.2
Remote RESET	Resets the CPU module when the CPU module is in a STOP status.	○	○	○	○	○	Section 6.6.3
Remote latch clear	Clears the latch data of the CPU module when the CPU module is in a STOP status.	○	○	○	○	○	Section 6.6.4
Input response time selection	Selects the input response time for the Q series compatible input modules, I/O combined modules, high-speed input modules, and interrupt modules.	○	○	○	○	○	Section 6.7

○ : Supported, × : Not supported

(Continued on next page)

Table6.2 CPU module function list (Continued)

Item	Description	CPU module					Reference
		1)	2)	3)	4)	5)	
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
Hardware error time CPU 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
Switch setting of intelligent function module	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 using GX Developer.	○	○	○	○	○	Section 6.11
Monitor condition setting	Specifies the monitoring timing of the CPU module with device condition or step number.	×	○	○	○	○[*1]	Section 6.11.1
Local device monitor/test	Monitors and/or tests the local devices of the specified program using GX Developer.	×	○	○	○	○	Section 6.11.2
Enforced ON/OFF of external I/O	Forcibly turns ON/OFF the external I/O of the CPU module using GX Developer.	×	○	○	○	○[*1]	Section 6.11.3
Executional conditioned device test	Changes a device value within the specified step of a sequence program.	×	×	×	×	○	Section 6.11.4
Online program change	Writes programs when the CPU module is in the RUN status.	○	○	○	○	○	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 execution count of interrupt programs.	○	○	○	○	○	Section 6.13.2
Scan time measurement	Measures the execution time of the area specified by the steps in a program.	×	○	○	○	○[*1]	Section 6.13.3
Sampling trace function	Samples the data of the specified device in the CPU module at a set interval.	×	○	○	○	○	Section 6.14
Multiple-user debugging function	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 into the memory as error history data.	○	○	○	○	○	Section 6.18
System protect	Prevents the programs from being modified from GX Developer, serial communication module Ethernet module or like.	△[*2]	○	○	○	△[*2]	Section 6.19

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

(Continued on next page)

Table6.2 CPU module function list (Continued)

Item	Description	CPU module					Reference
		1)	2)	3)	4)	5)	
Password registration	Prohibits writing/reading data to/from each file in the CPU module using GX Developer.	○	○	○	○	○	Section 6.19.1
Remote password	Prevents illegal access from external sources with serial communication modules and Ethernet modules.	○	○	○	○	○	Section 6.19.2
System display	Monitors the system configuration using GX Developer.	○	○	○	○	○	Section 6.20
LED display	Displays the CPU module operation status using LEDs located on the front of the module.	○	○	○	○	○	Section 6.21
Priority setting	Sets the priority for error messages stored in SD220 to 227 (LED display data) at the time of error occurrence. LED can be set to non-display.	○	○	○	○	×	Section 6.21.2
High speed interrupt function	Executes an interrupt program by fixed scan interrupt at 0.2ms to 1.0ms intervals using the interrupt pointer I49.	×	○	×	×	×	Section 6.22
Interrupt from intelligent function module	Executes an interrupt program at the time of interrupt request from the intelligent function module.	○	○	○	○	○	Section 6.23
Serial communication function	Connects the RS-232 interface of the Q00CPU or Q01CPU and the personal computer/display device with the RS-232 cable, and performs data communication using the MC protocol.	○	×	×	×	×	Section 6.24
Module service interval time read	Monitors the service interval time (time from service acceptance to next service acceptance) of intelligent function modules, network modules or GX Developer.	×	○	○	○	×	Section 6.25
Service processing setting	Specifies the service processing count or time to be executed in END processing.	×	×	×	×	○	Section 6.25
Device initial value	Registers data required for operations in a program to devices or buffer memory of intelligent function modules/special function modules (without using programs).	○	○	○	○	○	Section 6.26
Battery life-prolonging function	Prolongs a battery life by holding only clock data in the battery.	×	×	×	×	○	Section 6.27
Memory check function	Checks whether the data in memories of the CPU module are not modified due to excessive electric noise.	×	×	×	×	○	Section 6.28
Latch data backup function (to standard ROM)	Backs up latch data such as device data and error history without using a battery.	×	×	×	×	○	Section 6.29
Write/read device data to/from standard ROM	Writes/reads device data to/from standard ROM.	×	×	×	×	○	Section 6.30

○ : Supported, × : Not supported

(Continued on next page)

Table6.2 CPU module function list (Continued)

Item	Description	CPU module					Reference
		1)	2)	3)	4)	5)	
Online module change	Replaces the Q series I/O module, the intelligent function module of function version C, or the I/O module or intelligent function module mounted on a MELSECNET/H remote I/O station or an extension base unit online. In a redundant power supply system, the power supply module can also be changed online.	×	×	○	○	×	*3
Auto tuning function	Makes the initial setting of the PID constants. Auto tuning can be used for the process with relatively slow response such as temperature adjustment in the loop that uses the S.PID or S.2PID instruction.	×	×	○	○	×	*4
Redundant function	Configures a redundant system including two sets of CPU modules, power supply modules, network modules or main base units.	×	×	×	○	×	*5
System switching (between control system and standby system) function	Control system and standby system (switches the control system to the standby system or the standby system to the control system). There are two different methods: system switching and user switching.	×	×	×	○	×	
Operation mode change	Changes between the separate mode and backup mode.	×	×	×	○	×	
Tracking function	Used to share data between the control system and standby system. (Transfers the control system data to the standby system.) If system switching occurs at the time of a control system failure or error, control can be continued with the same data.	×	×	×	○	×	
Redundant online change function	Transfers the data, which was written to the control system by write to PLC or online change, to the standby system.	×	×	×	○	×	
Memory copy from control system to standby system	Copies the program memory contents of the control system to the standby system. (When the standby system CPU module is replaced, the program memory contents of the control system and standby system can be matched.) There are two different methods: by GX Developer and by special relay and special register.	×	×	×	○	×	

○ : Supported, × : Not supported

* 1 : When using these functions in the Universal model QCPU, check the versions of CPU module and GX Developer. (Appendix 4.5)

* 2 : The Basic model QCPU and Universal model QCPU do not support the system protect using DIP switches.

* 3 : Refer to the following manual

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

* 4 : Refer to the following manual.

QnPHCPU/QnPRHCPU Programming Manual (Process Control Instructions)

* 5 : Refer to the following manual.

QnPRHCPU User's Manual (Redundant System)

6.2 Constant scan

(1) Definition of Constant Scan

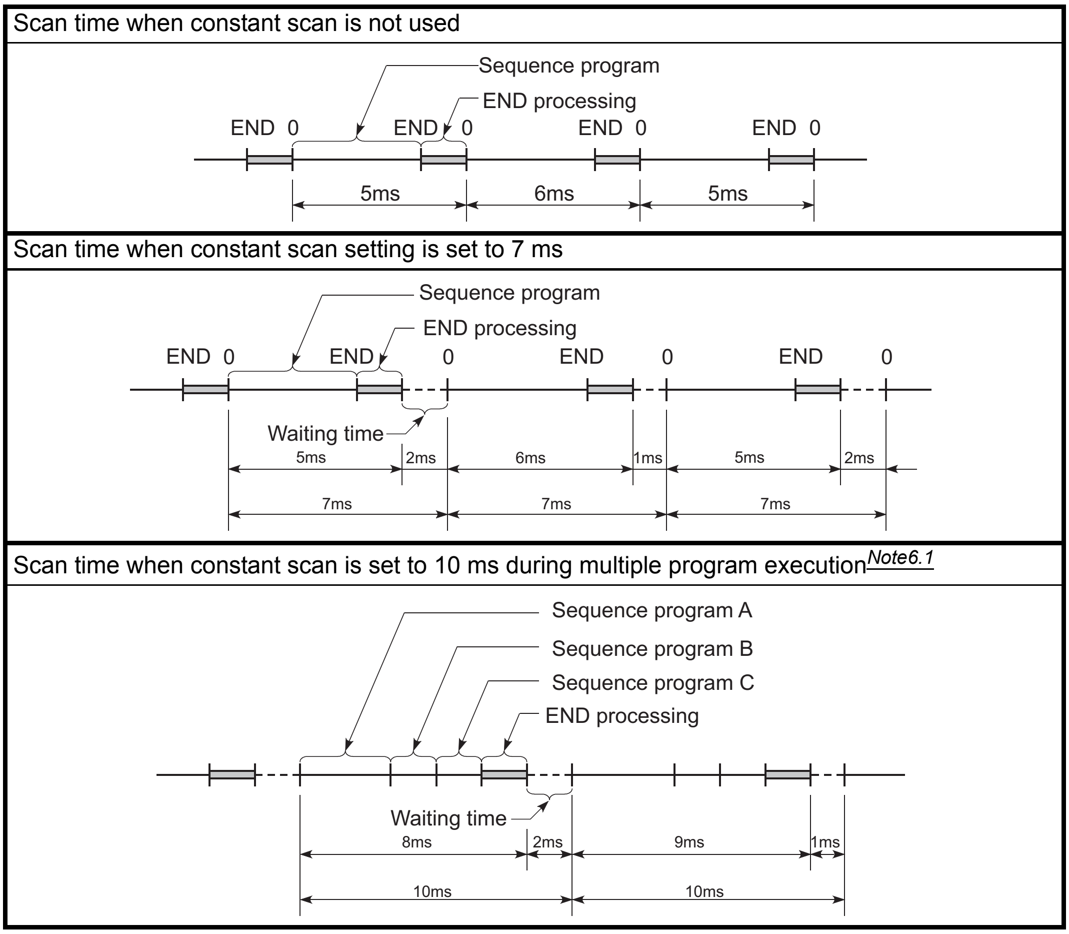
The scan time differs because the processing time differs depending on whether the instruction, which is used in the sequence program, is executed or not.

Constant scan is a function to execute the sequence program repeatedly while maintaining the scan time at a constant time.

(2) Applications of constant scan

I/O refresh is performed before sequence program execution.

Using the constant scan function, the I/O refresh intervals can be made constant if the sequence program execution time varies.


Scan time when constant scan is not used

Scan time when constant scan setting is set to 7 ms

Basic
Note6.1

Scan time when constant scan is set to 10 ms during multiple program execution*Note6.1*

Figure 6.1 Constant scan operation

Basic
Note6.1

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) Setting the constant scanning time

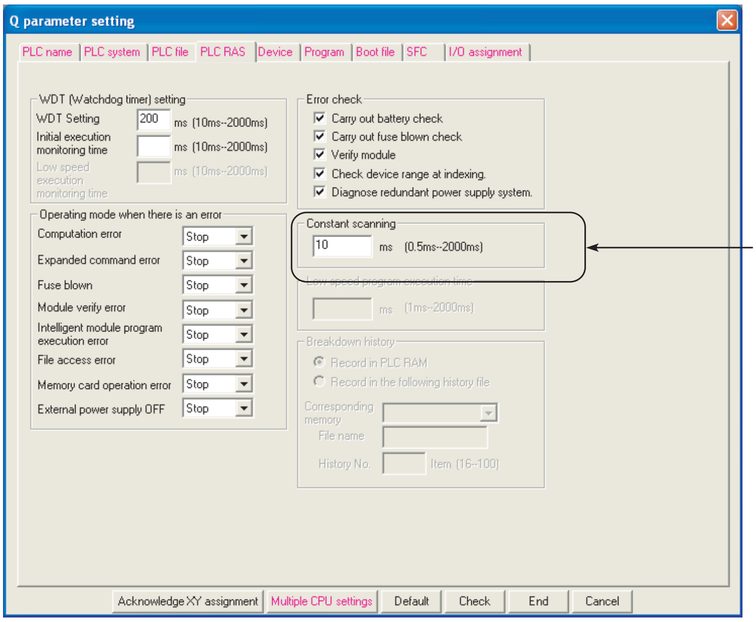
The constant scanning time is set at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box.

The constant scanning time can be set within the following range.

- For Basic model QCPU
  1 to 2000ms (set in 1ms unit)
- For High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU

When executing constant scanning, set the constant scanning time.

When not executing a constant scanning, leave the constant scanning time 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)

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 execution error Stop
File access error Stop
Memory card operation error Stop
External power supply OFF Stop

Constant scanning
10 ms (0.5ms--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)

Set the constant scanning time

Figure 6.2 When constant scanning is set to 10ms

(a) Setting time condition

As the constant scan time, set a value that satisfies the following relational expression.

(WDT Set Time) > (Constant Scan Set 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 is ignored and the sequence program is executed based on its scan time.

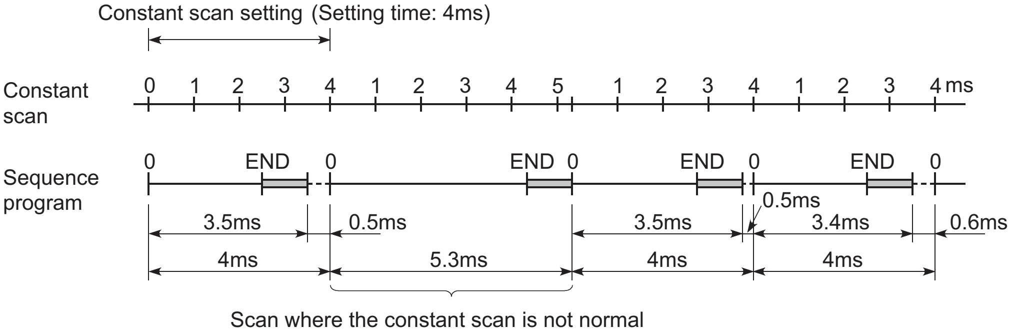

Constant scan setting (Setting time: 4ms)
Constant scan
0 1 2 3 4 1 2 3 4 5 1 2 3 4 1 2 3 4 ms
Sequence program
0 END 0 END 0 END 0 END 0
3.5ms
0.5ms
3.5ms
0.5ms
3.4ms
0.6ms
4ms
5.3ms
4ms
4ms
Scan where the constant scan is not normal

Figure 6.3 Operation when the Scan Time is longer than the Constant Scan

If the sequence program scan time is longer than the WDT setting time, the CPU module detects a WDT error.
In this case, the program execution is 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 the END processing of a sequence program is executed until the next scan starts.

Basic Redundant Universal
Note6.2 Note6.2 Note6.2

(a) When low speed execution type program is executed Note6.2

When a low speed execution type program is being executed, the execution of the low speed execution type program is stopped for the following period.

(Constant scan) - 0.5ms

(b) When interrupt factor occurs during waiting time until next scan starts

When an interrupt factor occurs during the waiting time until the next scan starts, either of the following programs is executed.

- Interrupt program
- Fixed scan execution type programNote6.2

Basic
Note6.2

Basic High Performance Process
Note6.3 Note6.3 Note6.3

Redundant
Note6.3

(c) When service processing setting is specified Note6.3

The communication service processing with peripherals (such as GX Developer) and intelligent function modules is enabled during the waiting time until next scan starts by specifying the service processing setting in GX Developer. (☞ Section 6.25.2)

Basic Redundant Universal
Note6.2 Note6.2 Note6.2

Since the Basic model QCPU cannot use low speed execution type programs and fixed scan execution type programs, it is not necessary to be conscious of this description.
Since the Redundant CPU, Universal model QCPU cannot use low speed execution type programs, it is not necessary to be conscious of this description.

Basic High Performance Process
Note6.3 Note6.3 Note6.3

Redundant
Note6.3

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not support the service processing setting.

(5) Constant scan accuracy

Refer to Section 11.2 for the constant scan accuracy.
However, when the program indicated in (a) or (b) below is being executed, the constant scan may be delayed.

Basic Redundant Universal
Note6.4 Note6.4 Note6.4

(a) When low speed execution type program is executedNote6.4

There is a waiting time of 0.5ms when a low speed execution type program is being executed. (☞Figure 3.31,Figure 3.32)

- When maximum processing time of one instruction is within 0.5ms
  The constant scan error is the same as the constant scan accuracy shown in Section 11.2.
- When maximum processing time of one instruction exceeds 0.5ms
  The constant scan time may increase by an excess over 0.5ms.
  Also, PRG. TIME OVER (error code: 5010) is detected.

Refer to the following manual for the maximum processing time of one instruction.

☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic
Note6.4

(b) When interrupt program/fixed scan execution type program is executedNote6.4

Interrupt is enabled while an interrupt program/fixed scan execution type program is executed. If constant scan time runs out when an interrupt program/fixed scan execution type program is executed, constant scan cannot be finished. When an interrupt program/fixed scan execution type program is used, constant scan time could be shifted by the execution time of an interrupt program/fixed scan execution type program.

Basic Redundant Universal
Note6.4 Note6.4 Note6.4

Since the Basic model QCPU cannot use low speed execution type programs and fixed scan execution type programs, it is not necessary to be conscious of this description.
Since the Redundant CPU, Universal model QCPU cannot use low speed execution type programs, it is not necessary to be conscious of this description.

6.3 Latch Function

(1) Definition of Latch Functions

The values of each High Performance model QCPU device are set back to the default (bit device: OFF and word device: 0) when;

- The PLC is powered OFF and then ON
- The reset operation is performed.
- Power failure longer than the permissible momentary power failure time occurs

Latch function maintains the device information when the above conditions occur.

(2) Operation performed when latch is used

Program operation is the same, independently of whether latch is used or not.

(3) Applications of latch function

Latch can be used to hold the data, which is being managed by continuous control, to continue control if the PLC is powered OFF, reset operation is performed, or a power failure longer than the permissible momentary power failure time occurs.

(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)

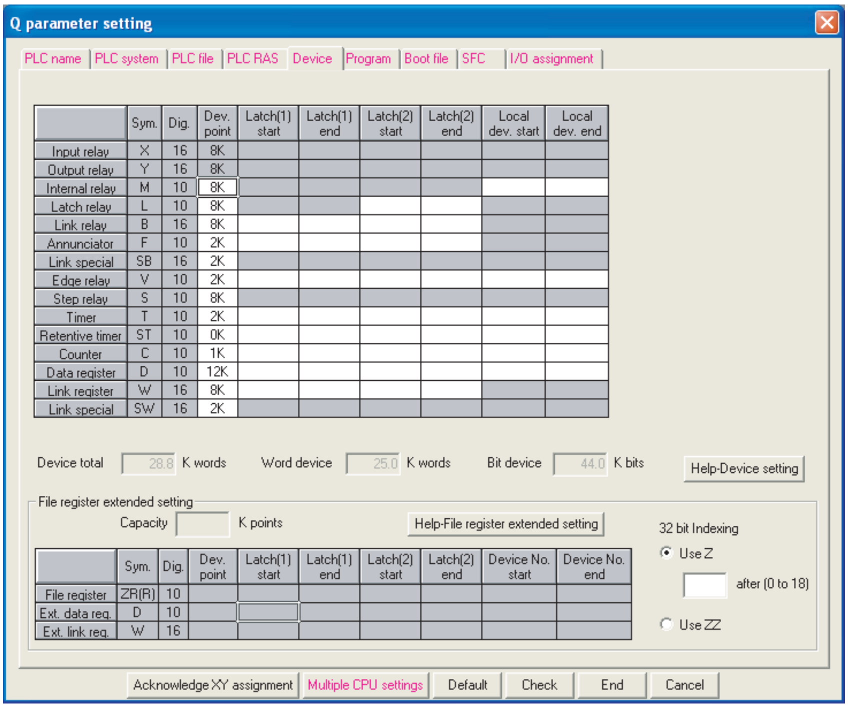
☒**POINT**

When the battery life-prolonging function (Universal model QCPU only) is set, the latch relay cannot be latched.

(5) Latch range setting

The latch range is set in the Device tab of the PLC parameter dialog box on GX Developer.

The latch range can be set in two types: the latch clear operation enable range and the latch clear operation disable range.



	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						

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 6.4 Latch range setting

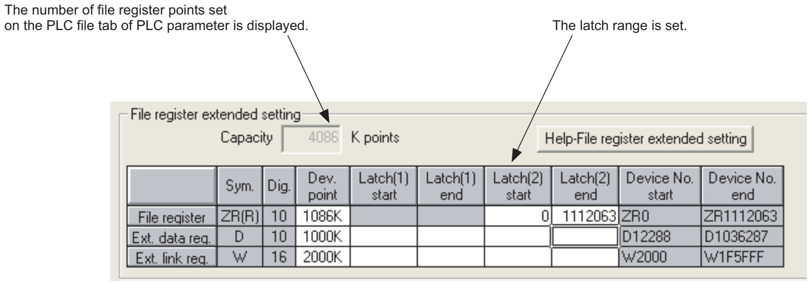
☒POINT

The latch range of the file register (ZR), extended data register (D), and extended link register (W) can be set in the Universal model QCPU.

The data outside the latch range is cleared by powering the programmable controller OFF → ON or performing the reset operation of the CPU module.

After selecting "Use the following file." and setting the capacity of the file register on the PLC file tab of PLC parameter, set the latch range of the file register (ZR), extended data register (D), and extended link register (W). (For the extended data register (D) and extended link register (W), assign a part of the file register area.)

If "Use the same file name as the program." is selected on the PLC file tab of PLC parameter, the latch range of the file register (ZR), extended data register (D), and extended link register (W) cannot be set. (All data in the file register is latched.)



The number of file register points set on the PLC file tab of PLC parameter is displayed.

The latch range is set.

File register extended setting

Capacity 4086 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	1086K			0	1112063	ZR0	ZR1112063
Ext. data reg.	D	10	1000K					D12288	D1036287
Ext. link reg.	W	16	2000K					W2000	W1F5FFF

Figure 6.5 Latch range setting of file register (ZR), extended data register (D), and extended link register (W)

When switching the file register file to be used with the QDRSET instruction, the latch range setting of the file register will be invalid. After switching, regardless of latch range setting, whole range of file register can be held without power supply.

(6) How to hold latch range device data and influence on scan time

(a) Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

Data is latched at the same time as the data is written to the devices in the latch range.

There is no influence on scan time due to latch since processing for latch is not performed.

(b) Universal model QCPU

Perform processing for latch at the time of END processing, which prolongs the scan time. When latching a device, consider the prolonging scan time.
(☞ Section 10.1.2(11))

☒POINT

For the Universal model QCPU, reduce the number of latch points (latch (1) setting, latch (2) setting, and latch relay (L)) as much as possible to minimize the prolonging scan time due to latch.

The following alternative methods can also reduce the number of latch points.

- Move data to be latched to a file register.
- Store the device data that is updated not so often to the standard ROM using the SP.DEVST instruction. (The device data stored in the standard ROM can be read with the S(P).DEVLD instruction.)

  ☞ QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)

(7) Clearing the Latch Range Device Data

The status of devices to which "latch clear" is made is shown in Table6.3.

Table6.3 Status when lath clear is performed

Latch setting	Clear/retention after "latch clear"
Devices not specified to be within latch range	Clear
Latch (1) setting (Devices with "latch clear" option)	Clear
Latch (2) setting (Devices without "latch clear" option)	Retain*

* : Refer to Section 3.7 for the clearing method.

☒**POINT**

File registers (R, ZR) cannot be cleared with latch clear.
Clear the file registers (R, ZR) by performing data clear operation using a sequence program or GX Developer. (☞Section 10.7(3))

(8) Precautions

Basic
Note6.5

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

Even if the device has been latch-specified, it will not be latched the when the local device[Note6.5] or the device initialization is specified.

(b) Use of battery

The device details of the latch range are maintained with the battery attached to the CPU module.

- If boot run is executed by the CPU module, the battery is required for latch.
- Take care that, if the battery connector is disconnected from the connector of the CPU module when PLC is turned off, the latch range device memory is not retained but becomes undefined.

(c) When start mode of Redundant CPU is hot-start mode

When the start mode is the hot-start mode, the data not set to be within the latch range are also held. (Except some devices such as the index register and step relay)

To clear these data, perform latch clear operation (☞Section 3.7(2)(b)).

Basic
Note6.5

The Basic model QCPU does not support local device designation.

6.4 Setting Output (Y) Status when Changing between STOP and RUN

(1) Definition of Setting Output (Y) Status when Changing between STOP and RUN

When changed from the RUN or other status to the STOP status, the CPU module stores the output (Y) in the RUN status into the PLC and turns all outputs (Y) OFF. Status when changing from STOP to RUN can be selected from the following two options with parameters in GX Developer.

- The output (Y) status prior to STOP is output.
- The output (Y) is cleared.

(2) Setting applications

Using a holding circuit or similar, it is possible to select whether the output is resumed from the previous status or not when the STOP status is changed to the RUN status.

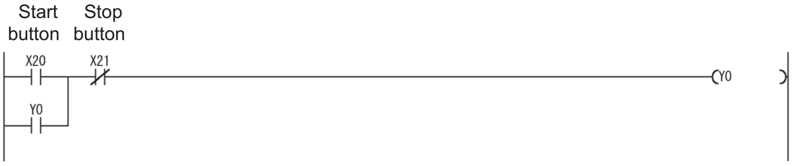


Figure 6.6 Holding circuit

- When the output (Y) status prior to STOP is set to output

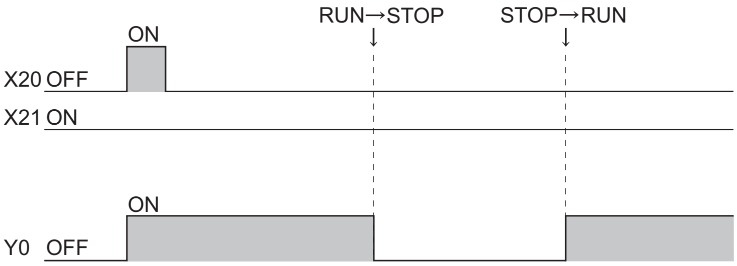


Figure 6.7 Timing chart when output (Y) status prior to STOP is set to output

- When output (Y) is set to clear

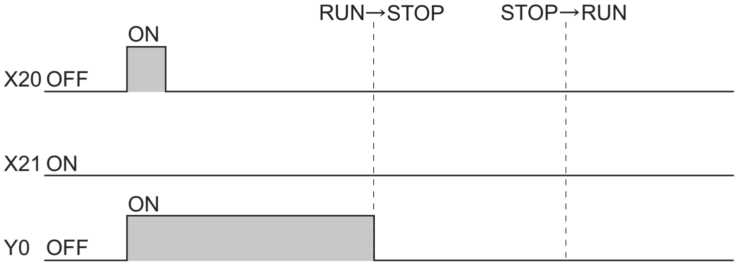


Figure 6.8 Timing chart when output (Y) is set to clear

(3) Operation when changing from STOP status to RUN status

(a) Output (Y) status prior to STOP is output (Default)

After the output (Y) status before the STOP status is output, the sequence program calculations are performed.

(b) Output (Y) is cleared

The output (Y) is OFF status.

The output (Y) will be output after the sequence program operation is executed.

Refer to (5) for the operation when forcibly turning ON the output (Y) in STOP status.

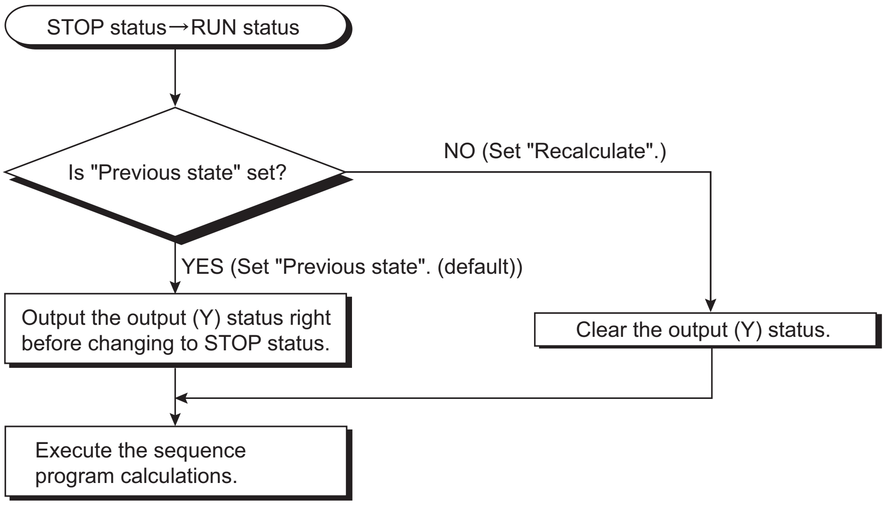
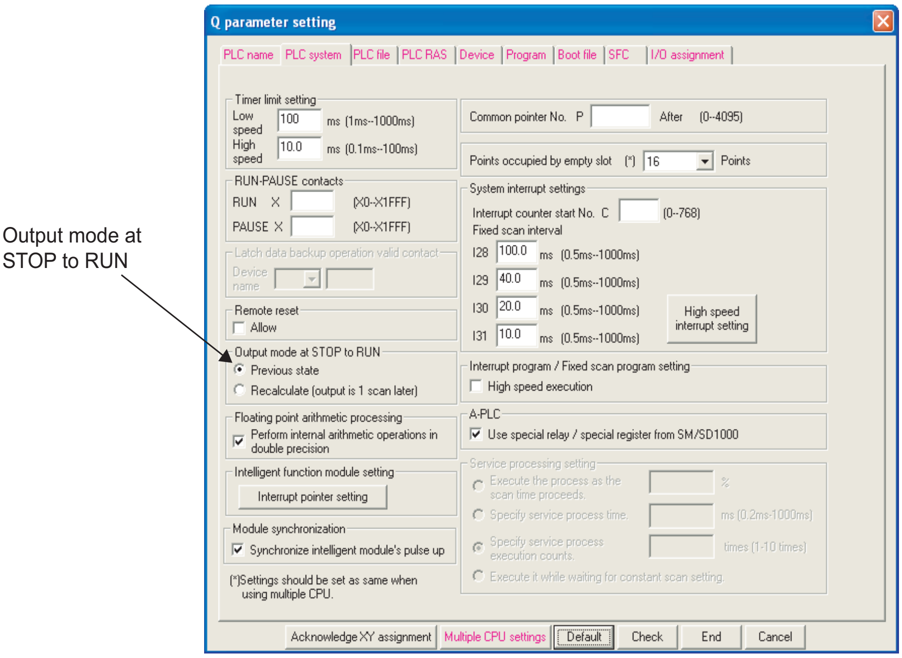

STOP status→RUN status
Is "Previous state" set?
NO (Set "Recalculate".)
YES (Set "Previous state". (default))
Output the output (Y) status right before changing to STOP status.
Clear the output (Y) status.
Execute the sequence program calculations.

Figure 6.9 Operation when Change from STOP Status to RUN Status

(4) Setting the Output (Y) Status when Changing from STOP Status to RUN Status

Set the output (Y) status when changing from the STOP status to the RUN status in the PLC system of the PLC parameter dialog box.



Output mode at STOP to RUN

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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)
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.

Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

Figure 6.10 PLC system screen

(5) Precaution

If an output (Y) is forcibly turned ON when the CPU is in STOP status, the output (Y) when changing from STOP status to RUN status is as shown in Table6.4.

Table6.4 Output when changing from STOP status to RUN status after forcibly turning ON output (Y) in STOP status

Output mode when changing from STOP to RUN	Output when changing from STOP status to RUN status
Output (Y) status prior to STOP is output	The status prior to STOP is output. If output (Y) is OFF prior to STOP, ON status is not maintained.
Output (Y) is cleared	ON status is maintained.

6.5 Clock Function

(1) Definition of Clock Function

The clock function reads the internal clock data of the CPU module to use it for time management.

The clock data is used by the CPU module system to perform time management, e.g. storage of date into the error history.

Remark

The Built-in Ethernet port QCPU can set the time in the module automatically using the time setting function (SNTP client). (☞ Section 7.5)

(2) Clock operation at power OFF and momentary power failure

Clock operation is continued by the internal battery of the CPU module during power OFF of the PLC or when a power failure longer than the permissible momentary power failure time occurs.

(3) Clock Data

Clock data is used in the CPU module and includes the data indicated in Table6.5.

Table6.5 Clock data details

Data Name	Contents	
Year	Four digits in AD (Countable from 1980 to 2079)	
Month	1 to 12	
Day	1 to 31(Automatic leap year calculation)	
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

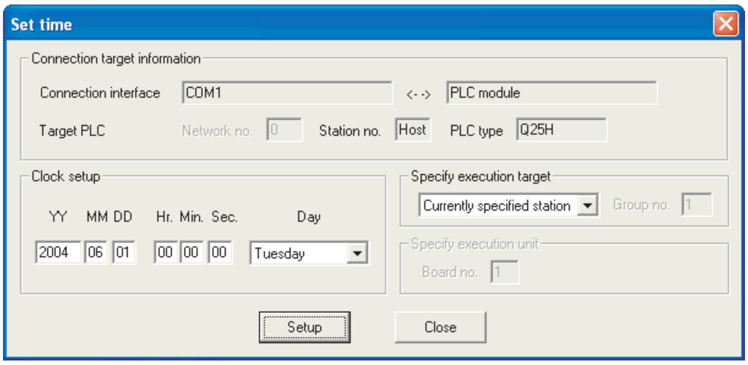
(4) Changing and reading the clock data

(a) Changing the clock data

The clock data can be changed either by GX Developer or program.

1) Method to write from GX Developer

When using GX Developer, choose [Online] → [Set time] to display the clock setting window, and change the clock data.



Set time

Connection target information

Connection interface: COM1 <--> PLC module

Target PLC: Network no. 0, Station no. Host, PLC type Q25H

Clock setup

YY	MM	DD	Hr.	Min.	Sec.	Day
2004	06	01	00	00	00	Tuesday

Specify execution target: Currently specified station, Group no. 1

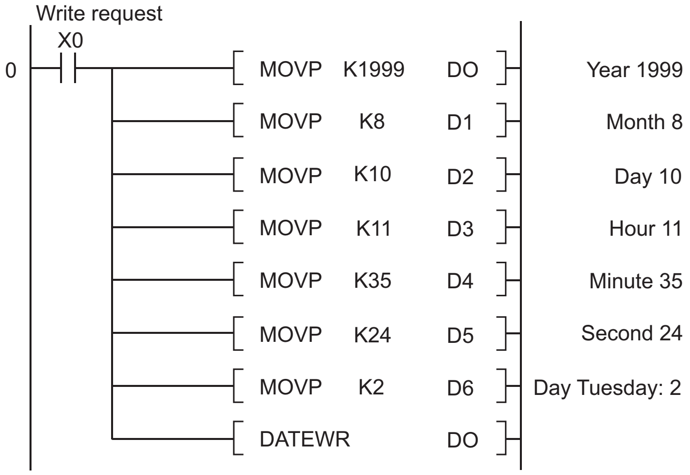
Specify execution unit: Board no. 1

Figure 6.11 Set time screen

2) Method to Write from the Program

When using a program, change the clock data with the clock data write instruction (DATEWR).

Figure 6.12 shows a program for writing the set clock data to D0 to D6 with the clock data write instruction (DATEWR).



```
   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 Tuesday: 2
       └─[ DATEWR         DO ]
```

Figure 6.12 Program example of writing clock data

Refer to the following manual for details of the DATEWR instruction.

☞ Q QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

POINT

1. When the clock data is changed with GX Developer or program, the clock of 1/1000 sec. is reset to 0.
2. Year data can be set up to 2037 in the clock data on GX Developer.

(b) Reading Time Data

When the time data is read to the data register, use either of the following instructions in the program.

- Time data read instruction (DATERD)
- Expanded time data read instruction (S(P).DATERD)

The figure below shows an example of a program used to read the clock data with the DATERD instruction and then store it in D10 to D16.

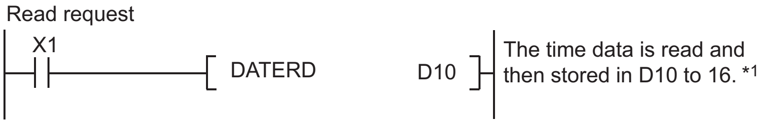


Figure 6.13 Program that stores clock data

* 1 : Figure 6.14 shows the clock data stored in D10 to 16.

D10	2004	4 digits in AD
D11	4	Month
D12	1	Date
D13	11	Hour
D14	35	Minute
D15	24	Second
D16	2	Day of the week

(☞ Section (3)(b))

Figure 6.14 Clock data stored

Refer to the following manual for details of the DATERD instruction and S(P).DATERD instruction.

☞ Q QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

☒POINT

Time Data can be written to and read from by special relays (SM210 to SM213) and special registers (SD210 to SD213).

Refer to Appendix 1 for details on special relay. Refer to Appendix 2 for details on special registers.

(5) Precautions

(a) Initial clock data setting

The clock data is not factory-set.

The clock data is used by the CPU module system and intelligent function modules for error history, etc.

When using the CPU module for the first time, be sure to set the precise time.

(b) Clock data correction

If part of the clock data is corrected, all data must be written to the CPU module again.

(c) Range of new clock data

Write new clock data within the range given in (3) of this section.

If impossible time data is written to the CPU module, normal clock operation cannot be performed.

However, if the data is impossible as time, an error will not occur when it is within the range given in (3) of this section.

Table6.6 Example of data written to clock data

	Write to CPU module	CPU module operation status
February 30	Executed	Error is not detected
32 of month 13	Not executed	When DATEWR instruction is executed: OPERATION ERROR (Error code 4100) When SM210 is on: SM211 is on

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

1) Function that clock data of 1/1000 sec. can be use

The clock data can be used with only the following instructions.

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

The clock data cannot be used with instructions other than above.

(Read using SM/SD, occurrence time of error stored in error history, read using GX Developer and read using dedicated instruction of another module, etc.)

2) When clock data is changed

When clock data is changed using GX Developer or instructions (including dedicated instruction of another module), the clock of 1/1000 sec. is reset to 0.

(6) Accuracy of Clock Data

The accuracy of the clock function differs with the ambient temperature, as shown below:

Table6.7 Accuracy of Basic model QCPU

Ambient Temperature (℃)	Accuracy (Day difference, S)
0	- 3.2 to + 5.27(TYP.+ 1.98)
+ 25	- 2.57 to + 5.27(TYP.+ 2.22)
+ 55	- 11.68 to + 3.65(TYP.- 2.64)

Table6.8 Accuracy of High Performance model QCPU and Process CPU

Ambient Temperature (℃)	Accuracy (Day difference, S)
0	- 3.18 to + 5.25(TYP.+ 2.12)
+ 25	- 3.93 to + 5.25(TYP.+ 1.9)
+ 55	- 14.69 to + 3.53(TYP.- 3.67)

Table6.9 Accuracy of Redundant CPU

Ambient Temperature (℃)	Accuracy (Day difference, S)
0	- 3.2 to + 5.27(TYP.+ 2.07)
+ 25	2.77 to + 5.27(TYP.+ 2.22)
+ 55	- 12.14 to + 3.65(TYP.- 2.89)

Table6.10 Accuracy of Universal model QCPU

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

(7) Comparison of Clock Data

To compare High Performance model QCPU's clock data with a sequence program, use the DATERD instruction to read the clock data.

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

6.6 Remote Operation

Remote operation is an operation to change operating status of the CPU module from the outside (GX Developer, external device using MC protocol, link dedicated instruction of the CC-Link IE controller network module or MELSECNET/H network module, and remote contacts)

The following four options are available for 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 of Remote RUN/STOP

The remote RUN/STOP performs RUN/STOP of the CPU module externally with the CPU module RUN/STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU) at RUN.

(2) Applications of remote RUN/STOP

Using remote RUN/STOP for the following remote operations are useful:

- When the CPU module is at a position out of reach
- When performing RUN/STOP of the control board CPU module externally

(3) Operations during remote RUN/STOP

Programs operate as follows with the remote RUN/STOP:

(a) Remote STOP

Executes the program to the END instruction and enters the STOP status.

(b) Remote RUN

When remote RUN is performed while in the STOP status using remote STOP, the status changes to RUN and executes the program from step 0.

(4) Methods of performing remote RUN/STOP

There are three methods for performing remote RUN/STOP.

- Method with RUN contact
- Method with GX Developer or external devices using MC protocol
- Method with link dedicated instructions of the CC-Link IE controller network module or MELSECNET/H network module

(a) Method with RUN contact

The RUN contact is set at the PLC system tab screen in the (PLC) Parameter dialog box of GX Developer.

The device ranges that can be set are shown in Table6.11.

Table6.11 Device ranges that can be set to RUN contact

CPU module	Available device range
Basic model QCPU	Input X0 to 7FF
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	Input X0 to 1FFF

By turning the set RUN contact ON/OFF, the remote RUN/STOP can be performed.

- When the RUN contact is OFF, the CPU module enters the RUN status.
- When the RUN contact is ON, the CPU module enters the STOP status.

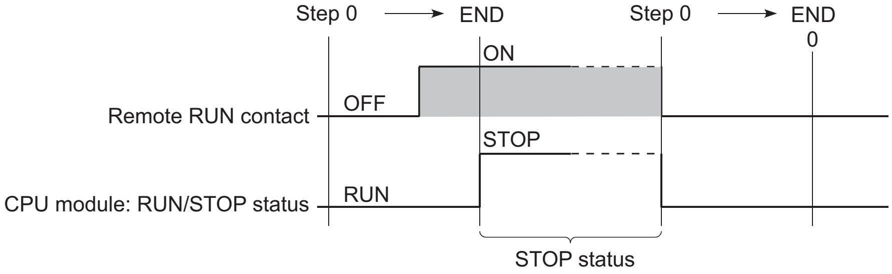


Figure 6.15 Time Chart for RUN/STOP with RUN Contact

(b) **Method by GX Developer or external device using MC protocol**

RUN/STOP of the CPU module can be executed by performing remote RUN/STOP operation with GX Developer or external device using MC protocol.

Operate GX Developer by choosing [Online] → [Remote operation].
Execute remote RUN/STOP from an external device using the MC protocol command.

☞ Q Corresponding MELSEC Communication Protocol Reference Manual

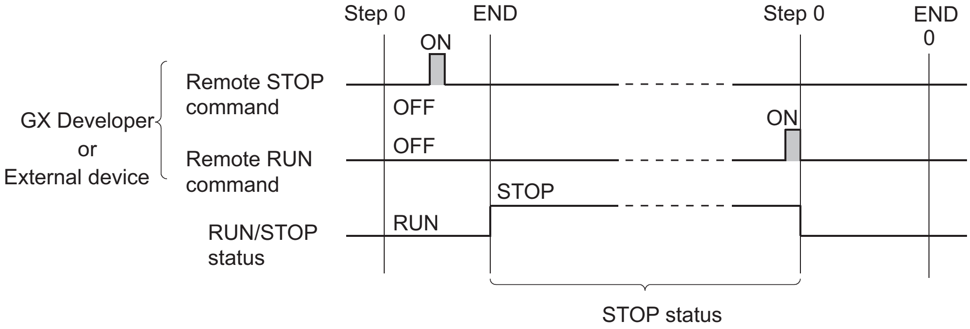


Figure 6.16 Remote RUN/STOP by GX Developer or external device

(c) **Method with link dedicated instructions of the CC-Link IE controller network module or MELSECNET/H network module**

The remote RUN/STOP operation by the link dedicated instructions of the CC-Link IE controller network module or MELSECNET/H network module can change the RUN/STOP status of the CPU module. Refer to the following manuals for details.

☞ CC-Link IE Controller Network Reference Manual

☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

(5) Precautions

Take note of the following, because STOP has priority in CPU module:

(a) Timing of changing to STOP status

The CPU module is put in the STOP status when remote STOP is executed from any of the followings: RUN contact, GX Developer and external device using MC protocol.

(b) To put CPU module in RUN status again after remote STOP

When placing the CPU module in the RUN status again after putting it in the STOP status by remote STOP, perform remote RUN in the order that remote STOP was executed first.

(c) When redundant system is configured using Redundant CPU

1) When both systems are not specified on remote operation screen

Remote RUN/STOP is performed for only the system specified in the connection target settings.

2) When both systems are specified on remote operation screen

Remote RUN/STOP is performed for both the control system and standby system.

However, remote operation for both systems is enabled only when the Redundant CPU is in the backup mode.

☒POINT

1 . The RUN/STOP status is described below:
- RUN Status....................Status in which the calculations are repeatedly executed from step 0 to the END/FEND instruction in the sequence program.
- STOP Status..................Status in which the sequence program calculations are stopped and the output (Y) is all OFF.

2 . After being reset, the CPU module is put to RUN/STOP status according to the RUN/STOP switch setting.

6.6.2 Remote PAUSE

(1) Definition of Remote PAUSE

The remote PAUSE performs PAUSE of the CPU module externally with the CPU module RUN/STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU) at RUN position.

In the PAUSE status, the CPU module operation is stopped with the ON/OFF states of all outputs (Y) held.

(2) Applications of remote PAUSE

This can be used to maintain the output (Y) on even if the CPU module is changed to STOP status, in such areas as process control.

(3) Remote PAUSE method

Remote PAUSE operation can be performed either by the PAUSE contact or by GX Developer or external device using MC protocol.

(a) Method with PAUSE Contact

The PUASE contact is set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box of GX Developer.

The device ranges that can be set are shown in Table6.12.

Table6.12 Device ranges that can be set to PAUSE contact

CPU module	Available device range
Basic model QCPU	Input X0 to 7FF
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	Input X0 to 1FFF

- The PAUSE status contact (SM204) is turned on when the END processing is executed for the scan with both PAUSE contact and PAUSE permission flag (SM206) on.
- When the PAUSE contact is off or SM206 is turned off, the PAUSE status is canceled, and the sequence program calculation is performed again from step 0.

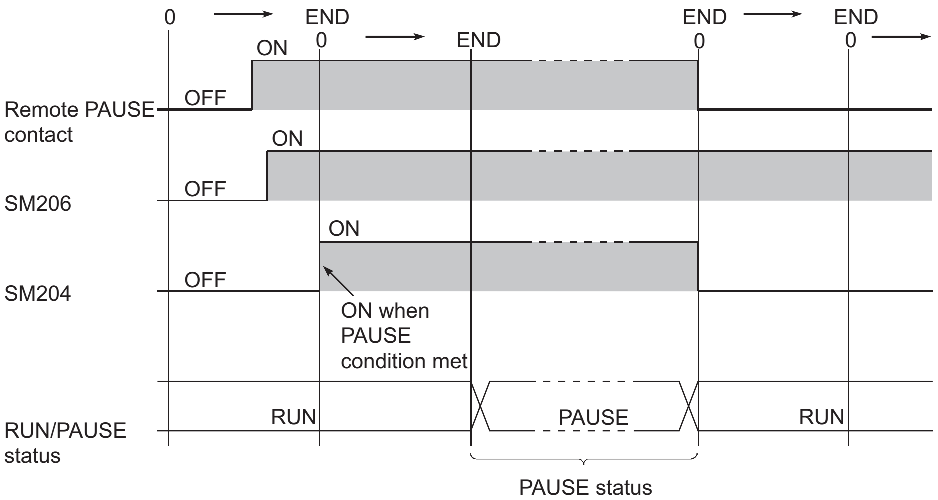


Figure 6.17 PAUSE Time Chart with PAUSE Contact

☒POINT

Setting of only the PAUSE contact is not allowed.
When setting the PAUSE contact, also set the RUN contact.

(b) **Method with GX Developer, Serial Communication Module etc.**

The remote PAUSE operation can be performed from the GX Developer or by using serial communication module.
Operate GX Developer by choosing [Online] → [Remote operation].

When executing remote PAUSE from an external device using MC protocol, use the MC protocol command.

☞Q Corresponding MELSEC Communication Protocol Reference Manual

- When the END processing is performed for the scan where the remote PAUSE command was accepted, the PAUSE status contact (SM204) is turned on.
  When the scan after the PAUSE status contact is turned on is executed to the END process, it enters the PAUSE status and stops the calculations.
- When the remote RUN command is received, the sequence program calculations are performed again from step 0.

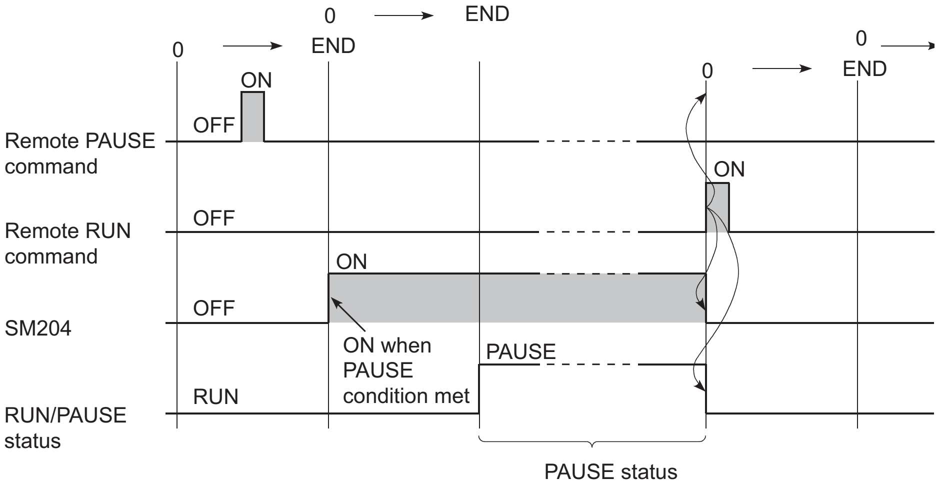


Figure 6.18 PAUSE Time Chart with GX Developer

(4) Precaution

(a) When forcibly keeping output ON or OFF

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

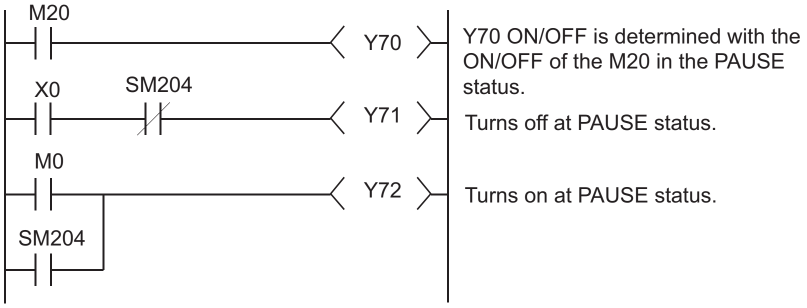

M20
Y70
Y70 ON/OFF is determined with the ON/OFF of the M20 in the PAUSE status.
X0
SM204
Y71
Turns off at PAUSE status.
M0
Y72
Turns on at PAUSE status.
SM204

Figure 6.19 Program example for forcibly keeping output (Y) ON or OFF in PAUSE status

(b) When redundant system is configured using Redundant CPU

1) **When both systems are not specified on remote operation screen**
Remote PAUSE is executed for only the system specified in the target connection settings.

2) **When both systems are specified on remote operation screen**
Remote PAUSE is performed for both the control system and standby system. However, remote operation for both systems is enabled only when the Redundant CPU is in the backup mode.

6.6.3 Remote RESET

(1) Definition of Remote RESET

The remote RESET resets the CPU module externally when the CPU module is in STOP status.

Even if RUN/STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU) is in RUN, the reset can be performed when the CPU module is stopped and an error that can be detected by the self-diagnosis function occurs.

(2) Applications of remote RESET

Remote RESET allows the faulty CPU module in a remote location to be reset by remote operation.

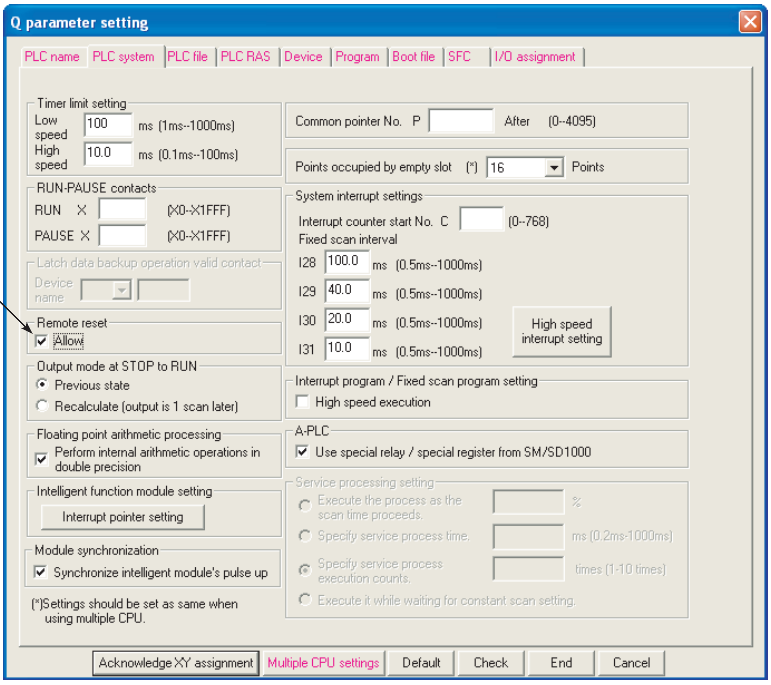
(3) Remote RESET method

Remote RESET operation can be performed by only GX Developer or external device using MC protocol.

To perform the remote RESET, follow the following steps:

- At the "PLC system" tab screen in the "(PLC) Parameter" dialog box, check the "Allow" check box at the "Remote reset" section, and then write parameters to the CPU module.

Allow the "Remote reset"



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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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.

Figure 6.20 Setting necessary for remote RESET

- When the CPU module is in RUN status, use remote STOP to arrange the STOP status.
- Reset CPU module by the remote RESET operation.
  For the GX Developer, this is performed by [Online] →[Remote operation.]
  The serial communication module and Ethernet interface module are controlled by commands complying with the MC protocol.
  ☞Q Corresponding MELSEC Communication Protocol Reference Manual

(4) Precautions

(a) Setting for remote RESET

To perform the remote RESET, check the "Allow" check box of the "Remote reset" section at the "PLC system" tab screen in the "(PLC) Parameter" dialog box, and then write parameters into CPU module.

If the "Allow" check box is not checked, a remote RESET operation is not performed.

(b) Remote RESET in RUN status

Remote RESET cannot be performed when the CPU module is in RUN status. Perform remote RESET after placing the CPU module in the STOP status by performing remote STOP or similar operation.

(c) Status after reset processing completion

After the reset processing is complete, the CPU module will enter operation status set by the RUN/STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU).

- With the RUN/ STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU) in the STOP position, the CPU module enters into the STOP status.
- With the RUN/ STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU) in the RUN position, the CPU module enters into the RUN status.

☒POINT

1. If remote RESET is performed with the CPU module stopping due to an error, note that the CPU module is placed in the operation status set by the RUN/ STOP switch (RUN/STOP/RESET switch for the Basic model QCPU and Universal model QCPU) upon completion of the reset processing.
2. Even if "Remote reset" is set as "Allow" at the "PLC system" tab screen in the "(PLC) Parameter" dialog box, the remote process of the GX Developer is completed.
   However, the CPU module is not reset since the reset process is not performed in it.
   If the status of the CPU module does not change though a reset process is performed from GX Developer, check if the "Remote reset" is set as "Allow" at the "PLC System" tab screen in the "(PLC) Parameter" dialog box.

(d) When error occurs due to noise

Take care that Remote RESET does not reset CPU module if an error occurs in the CPU module due to noise.

If Remote RESET does not rest, use the RESET/L.CLR switch (RUN/STOP/ RESET switch for the Basic model QCPU and Universal model QCPU) to reset or turn the PLC off.

(e) **When redundant system is configured using Redundant CPU**

In the backup mode, perform remote RESET for the control system, and then perform remote reset to reset both systems.

In the separate mode or debug mode, perform remote RESET for only the system specified the connection target settings.

1) **Remote RESET target in backup mode**

In the backup mode, perform remote RESET for the control system. (When remote RESET is performed for the control system, both systems are reset.)

If remote RESET is executed for the standby system, an error (error code: 4240_H) will occur.

2) **When RUN/STOP status differs between control system and standby system in backup mode**

When executing remote RESET with the control system in the STOP status, also place the standby system in the STOP status.

System switching will occur if remote RESET is performed with the control system in the STOP status and the standby system in the RUN status.

If remote RESET is performed with the control system in the RUN status and the standby system in the STOP status, only the standby system is reset.

3) WDT error has occurred in standby system in backup mode

If a WDT error has occurred in the standby system, the standby system is not reset when remote RESET is executed for the control system.

In this case, perform remote RESET in the following path (communication path where the tracking cable is not relayed).

- Connect the personal computer (GX Developer) directly to the Redundant CPU of the standby system.
- Execute remote RESET via the standby system module (e.g. remote RESET using MC protocol).

4) When remote operation is performed from the other path in backup mode

As described in Section 6.6.5 (3), remote operation cannot be performed from other GX Developer for the CPU module that is executing remote operation.

When remote operation is performed from other path for the control system and standby system as shown in Figure 6.21, the standby system may not be reset if remote RESET is executed for the control system.

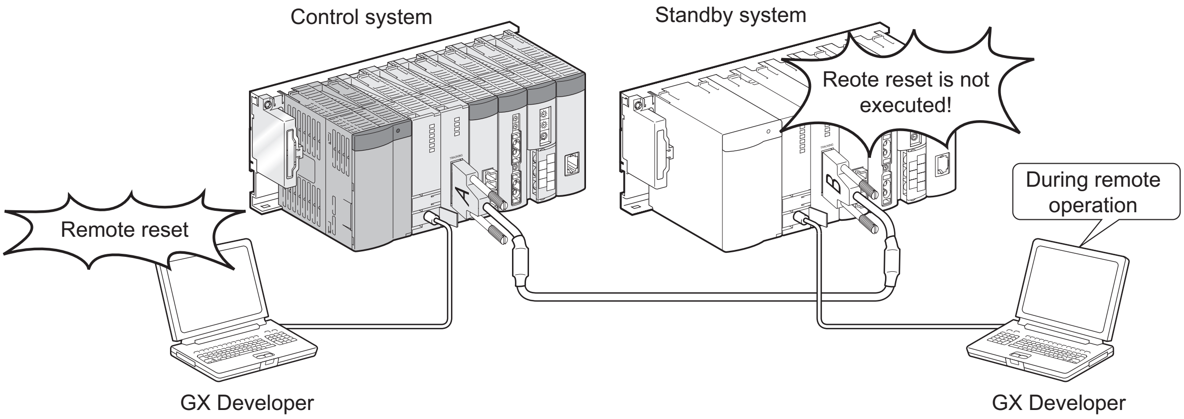


Figure 6.21 When remote operation is performed from other path

When remote operation is performed for the standby system, cancel the remote operation for the standby system and then execute remote RESET for the control system.

6.6.4 Remote latch clear

(1) Definition of Remote Latch Clear

Remote latch clear resets the latched device data from GX Developer or similar device when the CPU module is in the STOP status.

(2) Applications of remote latch clear

Remote latch clear is useful when the CPU module is in the following areas. In these cases, the operations are performed in combination with the remote RUN/STOP.

- When the CPU module is at a position out of reach
- When externally performing latch clear of the CPU module inside a control panel.

(3) Remote Latch Clear Method

Remote latch clear can be executed by performing operation from GX Developer, external device using MC protocol or similar device.
To perform the remote latch clear, follow the following steps:

- Use the remote STOP to bring the CPU module to STOP status.
- Use the Latch Clear to bring the CPU module to the Latch Clear status.
  Operate GX Developer by choosing [Online] → [Remote operation].
  Use the MC protocol command to execute control from an external device using the MC protocol.
  ☞Q Corresponding MELSEC Communication Protocol Reference Manual
- To return the High Performance model QCPU to RUN status after the remote latch clear, perform a remote RUN operation.

(4) Precautions

(a) Latch clear in RUN status

Latch clear cannot be performed when the CPU module is in the RUN status.

(b) Range where latch clear operation is enabled/disabled

The latch clear operation can be set to be enabled/disabled for the device latch range in the device setting of the PLC parameter dialog box.
Remote latch clear operation resets only the device range set as the "latch (1)" range.
Refer to Section 3.7 (2)(c) for how to reset the device set as latch clear disabled.

(c) Devices that are reset at execution of remote latch clear

Devices that are not latched are cleared when the remote latch clear is performed.

(d) When redundant system is configured using Redundant CPU

1) **When both systems are not specified on remote operation screen**
Remote latch clear is performed for only the system specified in the connection target settings.

2) **When both systems are specified on remote operation screen**
Remote latch clear is performed for both the control system and standby system.
However, remote operation is enabled for both systems (specified) only when the Redundant CPU is in the backup mode.

6.6.5 Relationship of remote operation and CPU's RUN/STOP status

(1) Relationship of the Remote Operation and CPU module Switch

The CPU module operation status is as shown in Table6.13 with the combination of remote operations to RUN/STOP switch.

Table6.13 Relation between RUN/STOP status and remote operation

RUN/STOP status	Remote operation				
	RUN [*1]	STOP	PAUSE [*2]	RESET [*3]	Latch clear
RUN	RUN	STOP	PAUSE	Cannot operate [*4]	Cannot operate [*4]
STOP	STOP	STOP	STOP	RESET [*5]	Latch clear

[*1] : When performing the operation with RUN contact, "RUN-PAUSE contact" must be set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

[*2] : When performing the operation with PAUSE contact, "RUN-PAUSE contact" must be set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box. In addition, the remote PAUSE enable coil (SM206) must be set ON.

[*3] : "Remote reset enable" must be set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

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

[*5] : This includes a situation where the CPU module is stopped due to error.

(2) Remote Operations from the Same GX Developers

When remote operations are performed from the same GX Developer, the status of the remote operation that is executed last will be effective.

(3) Remote Operations from Multiple GX Developers

Remote operations from multiple GX Developers via other stations cannot be performed to the CPU module where remote operations from single GX Developer are being performed.

(a) Remote RUN executed by GX Developer that is performing remote operation

When performing remote operation from another GX Developer, execute remote RUN using the GX Developer that is performing remote operation and cancel the remote operation.

For example, a remote PAUSE operation is being performed by one GX Developer, the PAUSE status will remain active even if a remote STOP/ remote RUN operation is attempted by another GX Developer.
When a remote RUN operation is performed by the GX Developer that is performing a remote PAUSE operation, and then that remote operation is cancelled, a new remote operation can be performed by another GX Developer.

6.7 Q Series Compatible Module Input Response Time Selection (I/O Response Time)

(1) Input response time selection

This function changes the input response times of each Q series corresponding module.

Table6.14 indicates the modules of which input response time can be changed and the applicable setting times.

Table6.14 Modules of which input response time can be changed

Module of which input response time can be changed	Type	Applicable setting times
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 above Q series corresponding modules import external input in the set input response time.

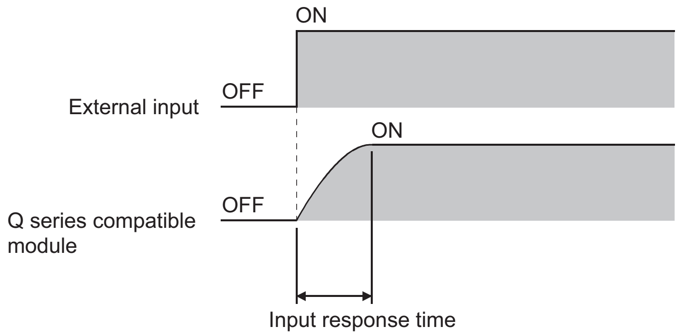


Figure 6.22 Input response time

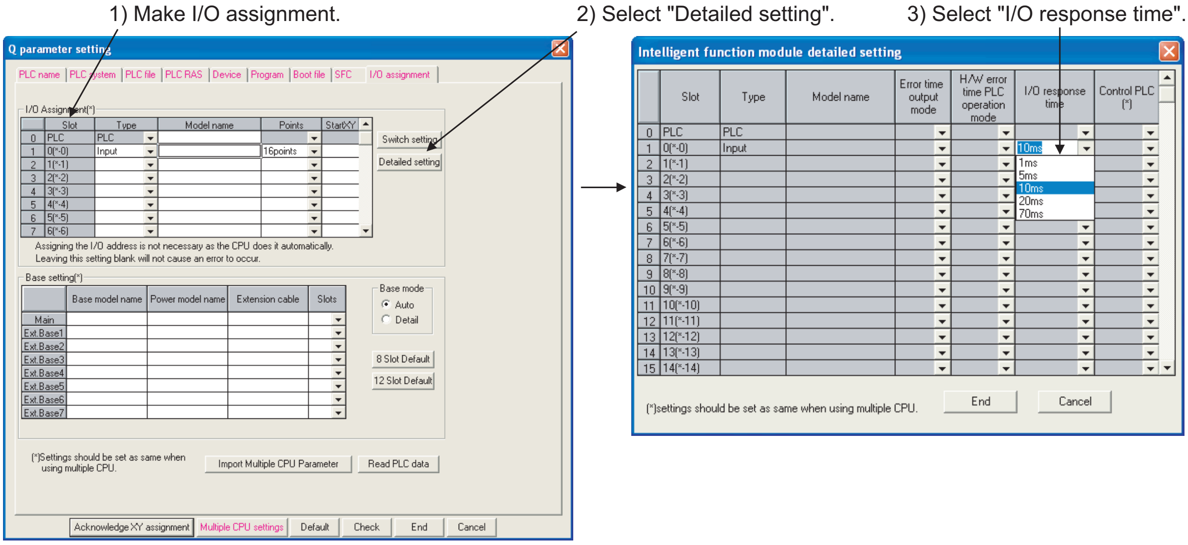
(2) Input response time setting

Set the input response time in the I/O assignment of the PLC parameter dialog box.

1) Make I/O assignment.

2) Select the Detailed setting button.

3) Set the input response time on the I/O module, intelligent function module detail setting screen.



I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)	Input		16points	
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.

	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	
2	1(*-1)					1ms	
3	2(*-2)					5ms	
4	3(*-3)					10ms	
5	4(*-4)					20ms	
6	5(*-5)					70ms	
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 Input response time setting

(3) Precautions

(a) Restrictions on GX Developer version and the relevant modules

When changing the input response time of the high-speed input module or interrupt module, use GX Developer version indicated in Table6.15.

If GX Developer version earlier than the version indicated in Table6.15 is used, the module will operate with the default value of the input response time.

Table6.15 Applicable GX Developer version

Module	Applicable GX Developer version
High-speed input module	GX Developer Version 5 (SW5D5C-GPPW-E) or later
Interrupt module	GX Developer Version 6 (SW6D5C-GPPW-E) or later

(b) When input response time is shortened

If the input response time is set shorter, an input module becomes susceptible to noise.

Set the desired input response time by taking into consideration the operating environment of an input module in use.

Basic Note6.6 Process Note6.6 Redundant Note6.6

Universal UD Note6.6

(c) When AnS/A series corresponding input module is used Note6.6

No change can be made to the input response time of an AnS/A Series corresponding input module.

When the input response speed is set to the slot of the AnS/A series compatible input module or interrupt module, the settings will be ignored.

(d) Timing when settings are ensured

The input response speed setting is valid in the following cases.

- When the PLC is powered OFF and then ON
- When the CPU module's RESET switch is set to neutral position

Basic Note6.6 Process Note6.6 Redundant Note6.6

Universal UD Note6.6

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the AnS/A series corresponding modules.

6.8 Error Time Output Mode Setting

(1) Error time Output Mode Setting

The error time output mode setting is to set whether the output to the Q series corresponding output module, I/O combined module, intelligent function module or interrupt module will be cleared or held when the CPU module results in a stop error.

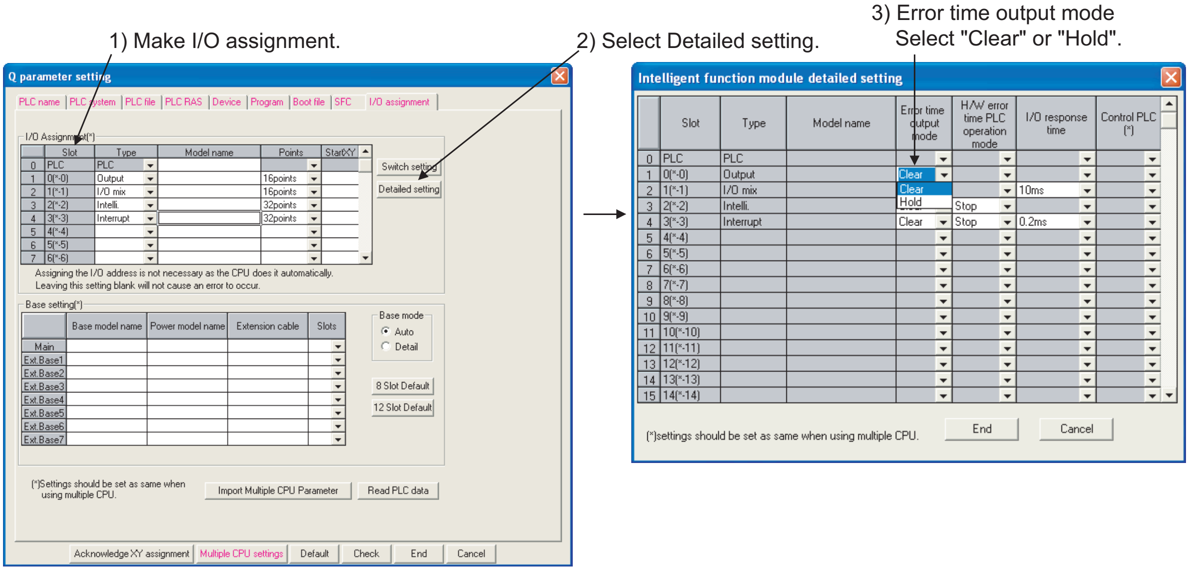
(2) Error time Output Mode Setting

Make the error time output mode setting in the I/O assignment setting of the PLC parameter dialog box.

1) Make I/O assignment.

2) Select the Detailed setting button.

3) Choose "Clear" or "Hold" for the slot to which the error time output mode is set. (The default is "Clear".)



1) Make I/O assignment.

2) Select Detailed setting.

3) Error time output mode Select "Clear" or "Hold".

Q parameter setting

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 Clear Hold			
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 6.24 Error time output mode setting sequence

(3) Precautions

The error time output mode setting is made valid when:

- When the PLC is powered OFF and then ON
- When the CPU module's RESET switch is set to neutral position

Failure to perform either operation after changing the error time output mode setting will result in "PARAMETER ERROR (error code: 3000)".

6.9 Hardware Error Time PLC Operation Settings

(1) Hardware Error time PLC Operation Mode Setting

The hardware error time PLC operation mode setting is to set whether the operation of the CPU module will be stopped or continued when a hardware error occurs in the intelligent function module or interrupt module.

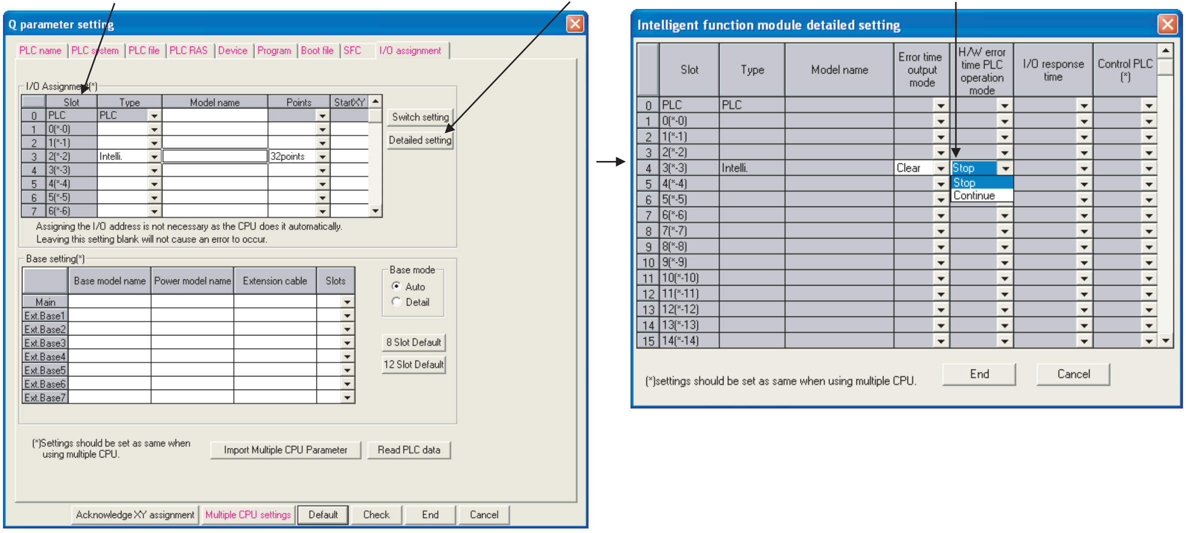
(2) Hardware Error time PLC Operation Mode Setting

Make the hardware error time PLC operation mode setting in the I/O assignment setting of the PLC parameter dialog box.

1) Make I/O assignment setting.

2) Select the Detailed setting button.

3) Choose the hardware error time PLC operation mode of the slot to which the hardware error time PLC operation mode is set.
(The default is "Stop".)



1) Make I/O assignment.

2) Select Detailed setting.

3) Hardware error time PLC operation mode Select "Stop" or "Continue".

I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)				
2	1(*-1)				
3	2(*-2)	Intelli.		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.

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)						
4	3(*-3)	Intelli.		Clear	Stop		
5	4(*-4)				Stop		
6	5(*-5)				Continue		
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 6.25 Hardware error time PLC operation setting sequence

(3) Precautions

The hardware error time PLC operation mode setting is made valid when:

- When the PLC is powered OFF and then ON
- When the CPU module's RESET switch is set to neutral position

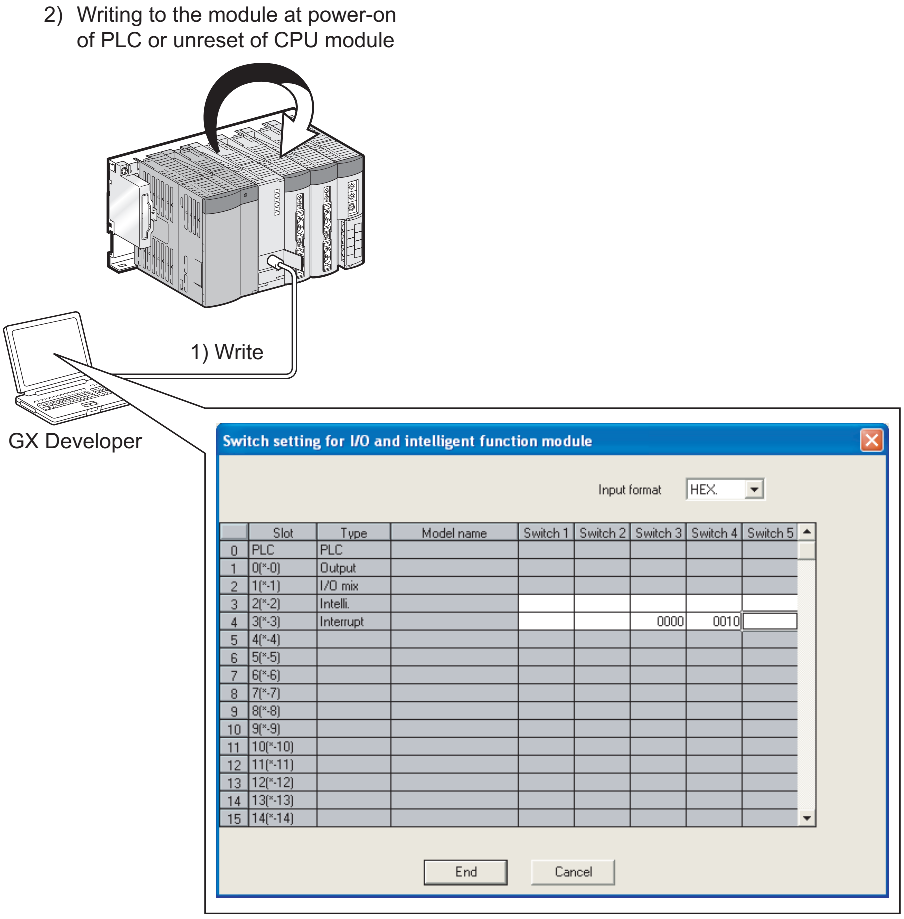
6.10 Intelligent Function Module Switch Setting

(1) Definition

The intelligent function module/interrupt module switch setting is to set the switches of the Q series compatible intelligent function modules using GX Developer.

(2) Timing when switch setting is written

The specified switch setting is written from the CPU module to the corresponding intelligent function modules and interrupt modules when the PLC is powered on or the CPU module is unreset, i.e., CPU module's RESET switch is set to the neutral position.


2) Writing to the module at power-on of PLC or unreset of CPU module
1) Write
GX Developer
Switch setting for I/O and intelligent function module
End
Cancel

Input format HEX.

	Slot	Type	Model name	Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
0	PLC	PLC						
1	0(*-0)	Output						
2	1(*-1)	I/O mix						
3	2(*-2)	Intelli.						
4	3(*-3)	Interrupt				0000	0010	
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)							

Intelligent function module switch settings

Figure 6.26 Sequence until switch setting is written to modules

(3) Setting the Switches of the Intelligent Function Modules and Interrupt Modules

At the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box, specify the desired switch setting. Select "Intelli." in the "Type" column of a slot for which to set the switches of the intelligent function modules and interrupt modules.

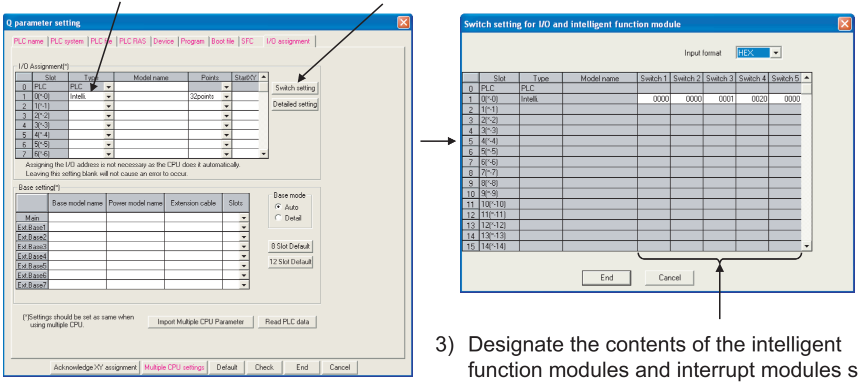
1) Make I/O assignment.

2) Select the switch setting button.

3) Set the module switches.

1) Make I/O assignment.

2) Select "Switch Setting".



I/O Assignment(*)

	Slot	Type	Model name	Points	StartXY
0	PLC	PLC			
1	0(*-0)	Intelli.		32points	
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.

Input format: HEX.

	Slot	Type	Model name	Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
0	PLC	PLC						
1	0(*-0)	Intelli.		0000	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)							

3) Designate the contents of the intelligent function modules and interrupt modules switch.

Figure 6.27 Switch setting sequence

(4) Precautions

Basic Process Redundant
Note6.7 Note6.7 Note6.7
Universal
UD
Note6.7

(a) **When AnS/A series corresponding module is used**Note6.7

In switch setting, do not set the AnS/A series corresponding module.
An error ("SP. PARA. ERROR") will occur if switch setting is made to any AnS/A series corresponding module.

(b) **Switch setting of each module**

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

(c) **Setting differences between GX Developer versions**

The switch settings for interruption modules with the GX Developer Version 6 (SW6D5C-GPPW-E) or later are made by setting the type to "Interruption".
When using GX Developer version 5 (SW5D5C-GPPW-E) or earlier to set the interrupt module switches, set the type to "Intelli.".
Refer to the following manual for further details on the interruption module's switch settings.

☞ Building Block I/O Module User's Manual

(d) **Timing when settings are ensured**

The switch setting of the intelligent function modules and interrupt modules become valid in the following cases.

- At power-on of the CPU module
- At reset cancel

Basic Process Redundant
Note6.7 Note6.7 Note6.7
Universal
UD
Note6.7

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the AnS/A series corresponding modules.

6.11 Monitor Function

(1) Definition of Monitoring Function

This is a function to read the program, device and intellignet function module status of the CPU module by using GX Developer.

Table6.16 Monitor function list and applicable CPU

Monitor function	Applicability for CPU modules					Reference
	Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU	
Ladder monitor	○	○	○	○	○	GX Developer Operating Manual
Device/buffer memory batch monitor	○	○	○	○	○	
Device registration monitor	○	○	○	○	○	
Device test	○	○	○	○	○	
Ladder registration monitor	○	○	○	○	○	
Monitor condition setting	×	○	○	○	○*3	Section 6.11.1
Local device monitor/test	×*1	○	○	○	○	Section 6.11.2
Forced ON/OFF for external I/O	△*2	○	○	○	○*4	Section 6.11.3
Executional conditioned device test	×	×	×	×	○*5	Section 6.11.4

○ : Available △ : Partly applicable × : N/A

* 1 : The Basic model QCPU is irrelevant to the difference between global devices and local devices (☞ Section 10.14.1).
Therefore, it is not necessary to make the monitor setting of the local devices on GX Developer.
* 2 : The Basic model QCPU can only execute forced ON/OFF by device test.
* 3 : When using the Monitor condition setting in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)
* 4 : When using the Forced ON/OFF of external I/O in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)
* 5 : When using the Executional conditioned device test in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

(2) Monitor request processing timing and displayed data

The CPU module performs the END processing to handle monitor requests from GX Developer.

The results of CPU module END processing are displayed on the GX Developer side.

Basic Universal
Note6.8 Note6.8

(3) Monitor with monitor condition setting[Note6.8]

By setting the monitor condition on GX Developer for debugging, the CPU module operation status can be monitored under the specified condition.
It is also possible to maintain the monitoring status under the specified conditions by setting the monitoring stop conditions.

Basic
Note6.9

(4) Local device monitor[Note6.9]

The use of local devices for execution of multiple programs makes it possible to monitor local device data.

Basic Universal
Note6.8 Note6.8

The Basic model QCPU cannot execute monitor with monitor conditions setting.
When setting the monitor condition in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

Basic
Note6.9

The Basic model QCPU is irrelevant to the difference between global devices and local devices (☞ Section 10.14.1).Therefore, it is not necessary to make the monitor setting of the local devices on GX Developer.

Basic Note6.10

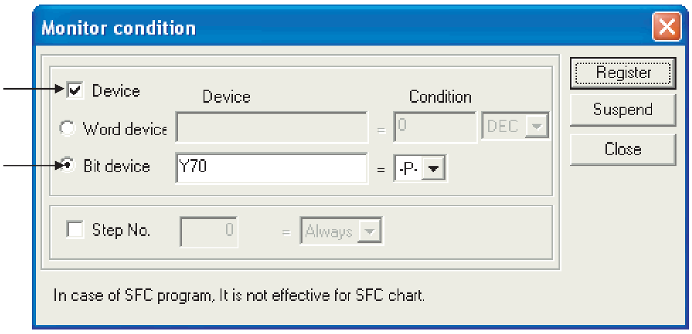
Universal UD Note6.10

6.11.1 Monitor condition setting

Set the monitor condition when executing monitor under the specified condition.

(1) Monitor execution condition setting for ladder monitor

Choose [Online] → [Monitor] → [Monitor condition setup] to open the Monitor Condition dialog box. The following shows an example in which to start a monitoring operation at the leading edge of Y70.



Monitor condition

☑ Device

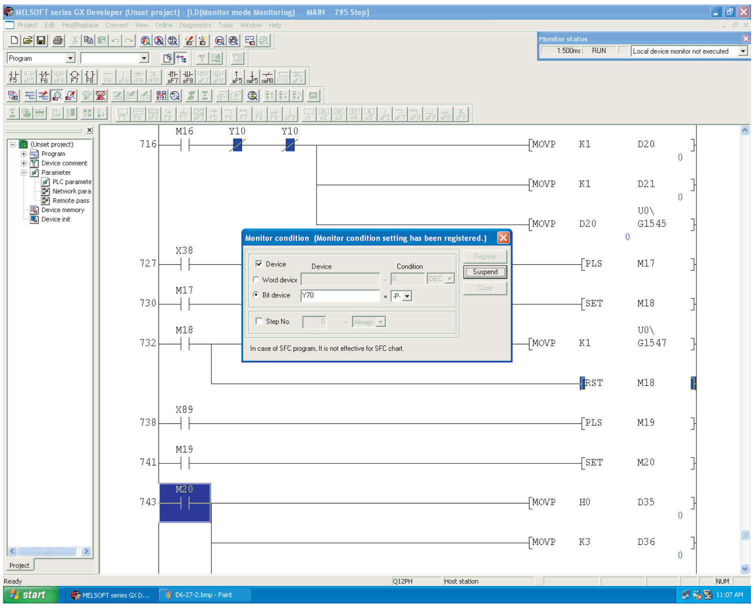
	Device		Condition
○ Word device		=	0 DEC
● Bit device	Y70	=	-P-

□ Step No. 0 = Always

In case of SFC program, It is not effective for SFC chart.

Select when monitoring by setting the contents of the device

Select when monitoring by setting Step No.



MELSOFT series GX Developer (Unset project) - [LD(Monitor mode Monitoring) MAIN 795 Step]
Monitor status
Monitor condition (Monitor condition setting has been registered.)
716
727
730
732
738
741
743
MOVP
PLS
SET
RST

When condition is enabled, monitor is executed (operation status when condition is enabled is displayed).

Figure 6.28 Monitor condition screen

Basic Note6.10 Universal UD Note6.10

The Basic model QCPU cannot execute monitor with monitor condition setting.
When setting the monitor condition in the Universal model QCPU, check the versions of CPU module and GX Developer.(☞ Appendix 4.5)

(a) When only "Step No." is specified:

The monitor data sampled when the status previous to execution of the specified step becomes "the specified".

The specification method for the execution status is indicated below:

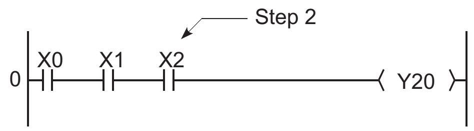
- When changing from non-execution status to executing status : < -P->
- When changing from executing status to non-execution status : < -F->
- Always when executing only : <ON>
- Always when not executing only : <OFF>
- Always regardless of status : <Always>

☒POINT

1 . If a step between the AND/OR blocks is specified as a monitor condition, monitor data is sampled 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 6.29, the monitor execution condition is established when both "X0" and "X1" are ON.

Ladder mode



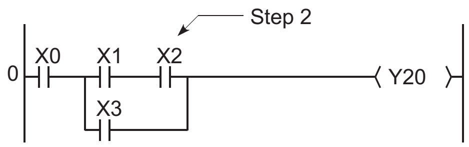
List mode

```
0  LD   X0
1  AND  X1
2  AND  X2
3  OUT  Y20
```

Figure 6.29 When the Step2 is connected by the AND instruction

- When the step 2 is connected in the middle of the AND/OR block
  The monitor execution condition is established when "X1" turns ON in Figure 6.30. (The ON/OFF status of "X0" does not affect the establishment of the monitor execution condition.)

Ladder mode



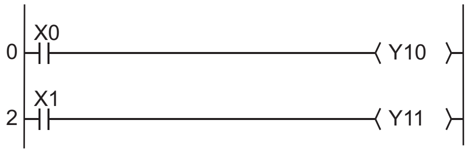
List mode

```
0  LD   X0
1  LD   X1
2  AND  X2
3  OR   X3
4  ANB
5  OUT  Y20
```

Figure 6.30 When the step 2 is connected in the middle of the AND/OR block

- If the beginning of a ladder block not at Step 0 is specified in Step No. 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.

Ladder mode



List mode

```
0  LD   X0
1  OUT  Y10
2  LD   X1
3  OUT  Y11
```

Figure 6.31 When the beginning of a ladder block not at Step 0 is specified in Step No.

2 . Be sure to set the condition of the step set as step No.0 to "Always".

(b) **When only Device is specified:**
Word Device or Bit Device can be specified.

1) **When Word Device is selected:**
The monitor data is sampled is when the current value of the specified word device becomes the specified value.
Type a current value in decimal digits or hexadecimal digits.

2) **When Bit Device is specified:**
The monitor data is sampled is when the execution status of the specified bit device becomes the specified value.
Either the leading edge or the fall can be specified for execution condition.

(c) **When Step No. and Device is selected:**
The monitor data is sampled when the status previous to execution of the specified status or the status (current value) of the specified bit device (word device) is specified.

☒POINT

When "Step No.[100]=<-P->, Word Device [D1]=[K5]" is specified as the detailed condition in the following circuit, a monitor execution condition is established at the leading edge of the Step 100 where D1=5.

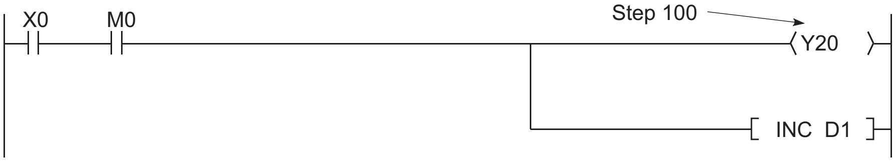


Figure 6.32 When the leading edge of Step 100 and [D1=5] are the execution conditions

Besides, 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 conditions are met at the monitor timing of GX Developer.

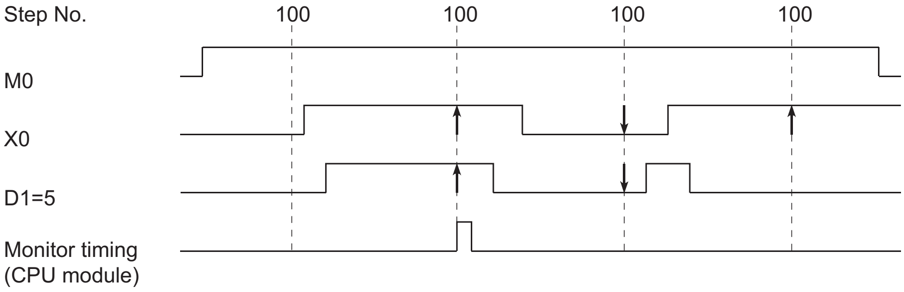
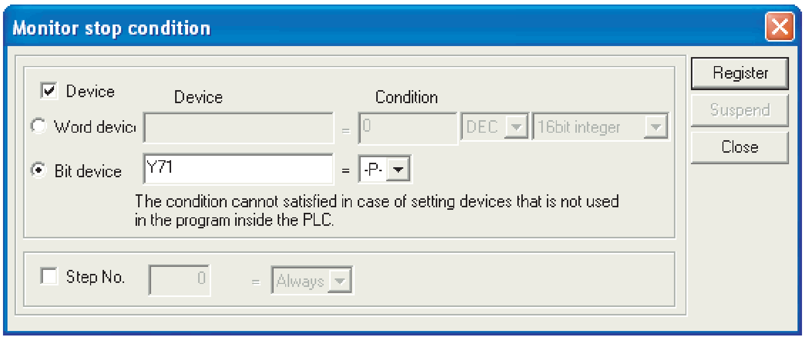


Figure 6.33 Monitor timing of GX Developer

(2) Monitor Stop Condition Set Up

Choose [Online] → [Monitor] → [Monitor stop condition setup] to open the [Monitor Stop Condition] dialog box. Figure 6.34 shows an example of stopping a monitoring operation at the leading edge of Y71.



☑ Device

○ Word devic: = 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 6.34 Monitor stop condition screen

(a) When "Step No." is specified:

Monitoring is stopped when the status at execution of the specified step becomes as specified.

The specification method for the execution status is shown below:

- 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>

When "Step No." is not specified, the monitoring operation is stopped after the CPU module END processing.

(b) When Device is specified:

Word Device or Bit Device can be specified.

1) When Word Device is selected:

The monitoring operation is stopped when the current value of the specified word device becomes the specified value.

A current value can be expressed in decimal digits, hexadecimal digits, 16-bit integral numbers, 32-bit integral numbers, or real numbers.

2) When Bit Device is specified:

The monitoring operation is stopped when the execution status of the specified bit device becomes the specified value.

Either the leading edge or the fall can be specified for execution condition.

(3) Precautions

(a) Monitored CPU module file

When monitoring after setting the monitor condition, the file displayed on GX Developer is monitored.

Choose [Online] → [Read from PLC] on GX Developer, and match the file name of the CPU module to be monitored with the file name on GX Developer.

(b) When there is no file register setting

When monitoring the file register which is not specified, 0 is displayed. (For the Universal model QCPU, FFFF_H is displayed.)

(c) Device assignment of CPU module and GX Developer for monitor

When executing monitor, match the device assignment of the CPU module with that of GX Developer.

(d) When monitoring the intelligent function module buffer memory

When monitoring the buffer memory of the intelligent function module, the scan time takes longer, as well as when executing the FROM/TO instruction.

(e) When multiple users are performing simultaneously

Multiple users can perform monitoring at the same time.

When multiple users are performing monitoring at the same time, take note of the following:

- High speed monitoring can be performed by increasing 1k step in the system area for other station's monitor file when formatting of program memory.
  Up to 15 stations can be set as the station monitor file, but the program space will be reduced.
- If the monitor condition and/or monitor stop condition are set, only one user can perform monitoring.

(f) Status where monitor stop condition can be set

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

(g) When the same device is specified for conditions

When the same device is specified as the monitor condition and monitor stop condition, specify "ON" or "OFF".

(h) When step No. is specified for monitor condition

The monitoring conditions will not be established unless the following specified steps commands are executed when "Step No." has been specified for the monitoring conditions.

- When skipping steps specified with the CJ instruction, the SCJ instruction and the JMP instruction.
- When the specified step is the END instruction, the FEND instruction exists while the program is running, and the END instruction is not executed.

(i) During monitor condition registration

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

(j) When executing monitor with monitor condition setting

When executing monitor with monitor condition setting, other applications on the same personal computer cannot execute online functions using the same route used for monitor. The following applications must be noted.

- GX Developer
- Application using MX Component
- MX Sheet

If online functions of other applications are executed using the same route for monitor execution, the following situations may occur.

- No response is returned for online functions from the CPU module. (An online communication function time-out occurs.)
- The CPU module detects an error (error code: 4109_H) for online functions. (Universal model QCPU only)
- Results of monitor with monitor condition setting are not updated even though the monitor condition is satisfied in the CPU module. (Universal model QCPU only)

Basic

Note6.11

6.11.2 Local device monitor/test

Local devices specified at the Device tab screen in the (PLC) Parameter dialog box can be monitored or tested by operating from GX Developer. This function is useful when debugging a program and monitoring local devices in a program monitored by GX Developer. (☞Section 10.14.1)

(1) Monitoring the Local Devices

Table6.17 shows the status of three programs "A", "B", and "C" being executed on the CPU module, with local devices D0 to 99 specified. It assumes that these three programs are executed in the order of A → B → C → (END processing) → A → B ....

Table6.17 Data displayed when three programs are executed

Setting	Monitor Device	
	D0 (Local device)	D100 (Global device)
When local devices are specified	D0 in Program "C" is monitored.	D100 is monitored after Program "C" is executed.
When local devices are not specified	D0 in the displayed program is monitored.	D100 is monitored after Program "C" is executed.

Basic

Note6.11

The Basic model QCPU is irrelevant to the difference between global devices and local devices (☞Section 10.14.1). Therefore, it is not necessary to make the monitor setting of the local devices on GX Developer.

If the local device monitor setting is made and Program "B" is displayed, for example, this makes it possible to monitor the local devices in Program "B".

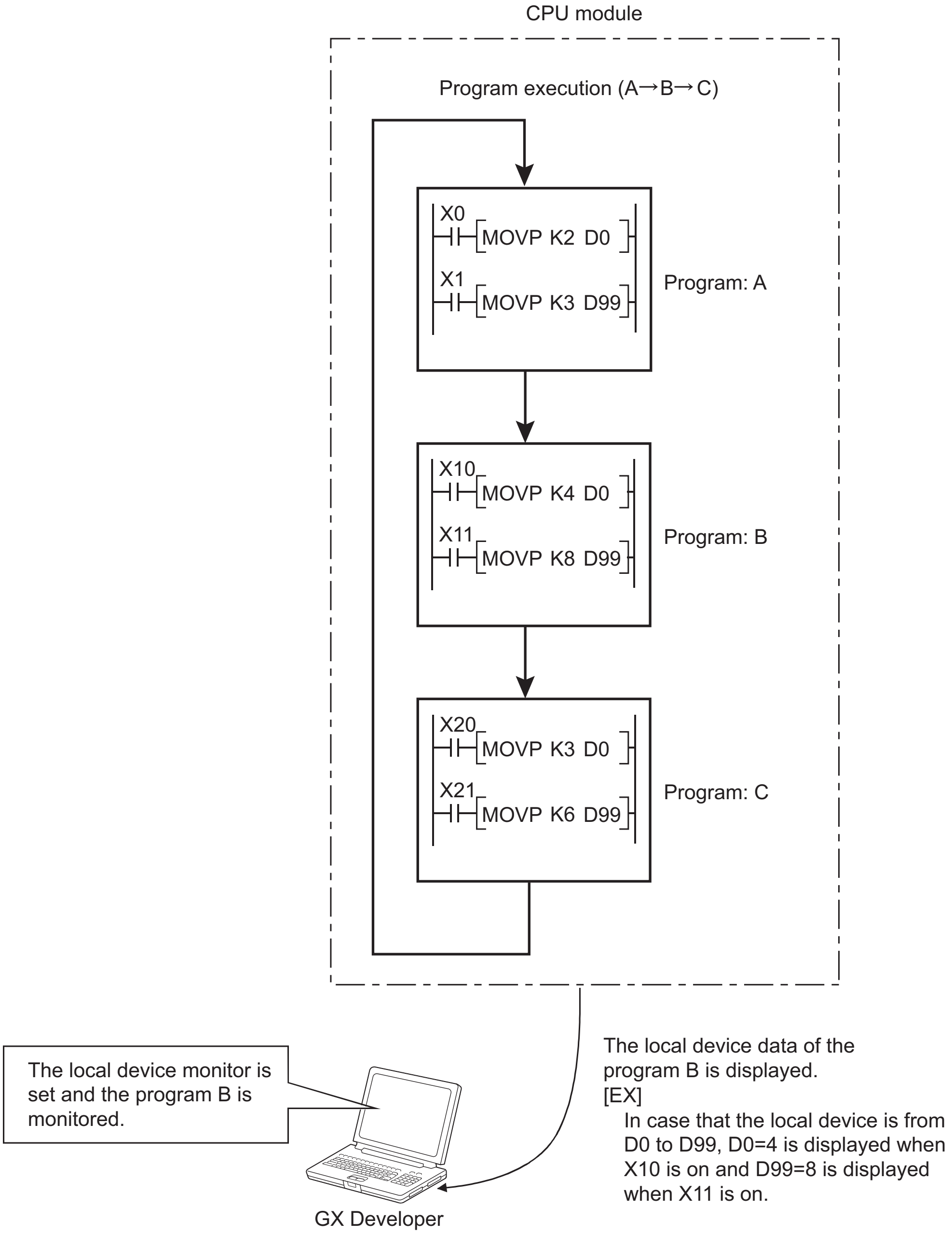

CPU module
Program execution (A→B→C)
X0
MOVP K2 D0
X1
MOVP K3 D99
Program: A
X10
MOVP K4 D0
X11
MOVP K8 D99
Program: B
X20
MOVP K3 D0
X21
MOVP K6 D99
Program: C
The local device monitor is set and the program B is monitored.
The local device data of the program B is displayed.
[EX]
In case that the local device is from D0 to D99, D0=4 is displayed when X10 is on and D99=8 is displayed when X11 is on.
GX Developer

Figure 6.35 Local device monitor example

(2) Monitoring the Local Devices

Monitor local devices in the following steps:

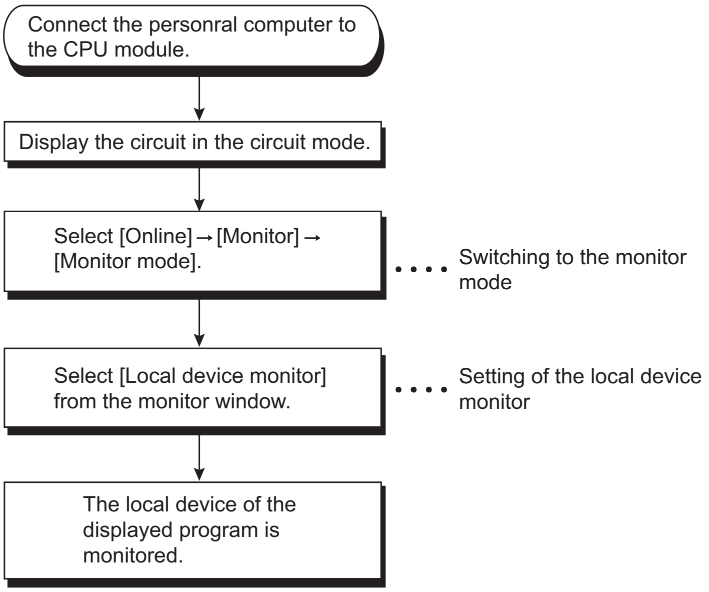

Connect the personral computer to the CPU module.
Display the circuit in the circuit mode.
Select [Online]→[Monitor]→[Monitor mode].
Switching to the monitor mode
Select [Local device monitor] from the monitor window.
Setting of the local device monitor
The local device of the displayed program is monitored.

Figure 6.36 Local device monitor procedure flowchart

(3) Precautions

(a) Local devices that can be monitored/tested by one GX Developer

It is only a single program that local devices can be monitored or tested by operating from a single GX Developer. Local devices in multiple programs cannot be monitored or tested by operating from a single GX Developer.

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

It is a maximum of 16 programs that local devices can be monitored or tested by operating from multiple GX Developers connected to a RS-232 serial communication module of the CPU module.

(c) Monitor of local devices in stand-by type program

When the local devices in the stand-by type program are monitored, the scan time increases since the local device data are saved and restored.
(☞ Section 10.14.1)

(d) Local device monitor of fixed scan execution type program

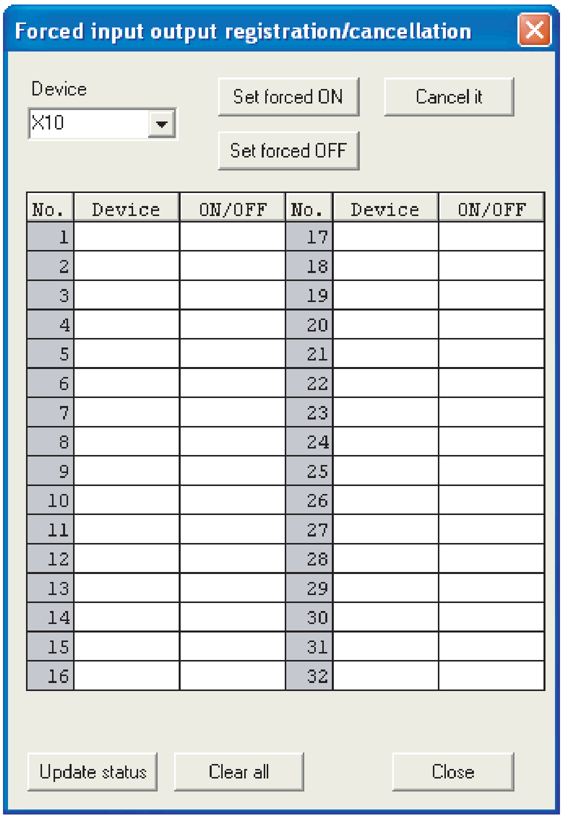
When the local devices in the fixed scan execution type program are monitored, device data is not acquired and "0" is displayed.

6.11.3 Forced ON/OFF for external I/O

High Performance Note6.12

Universal UD Note6.12

The external I/O can be forcibly turned ON/OFF on the screen displayed by selecting [Online] → [Debug] → [Forced input output registration/cancellation] in GX Developer. The information registered for ON/OFF will be cancelled with GX Developer operations.



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 6.37 Forced input output registration/cancellation screen

High Performance Universal UD Note6.12 Note6.12

When using the Forced ON/OFF for external I/O function in the High Performance model QCPU and Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.2 and 4.5)

(1) Operation when forced ON/OFF is registered

Basic

Note6.13

This function can perform three operations: forced ON ("Set forced ON"), forced OFF ("Set forced OFF"), and forced ON/OFF cancellation ("Cancel it").Note6.13

Table6.18 shows the I/O operation when operations described above are performed.

Table6.18 I/O operation when Forced ON/OFF function is executed

Operation	Input (X) operation	Output (Y) operation
Forced ON/OFF cancellation	Performs sequence program operations with external input.	Outputs the results of sequence program operations externally.*1
Forced ON	Performs sequence program operations in the forced ON status.	Outputs "ON" externally regardless of the results of sequence program operations.*1
Forced OFF	Performs sequence program operations in the forced OFF status.	Outputs "OFF" externally regardless of the results of sequence program operations.*1

* 1: In the backup mode, the enforced ON/OFF cannot be executed for the output (Y) of the standby system.

Basic

Note6.13

The Basic model QCPU does not support the forced ON/OFF registration operation from the Forced input output registration/cancellation screen (☞(3) in this section).
The Basic model QCPU can forcibly turns ON/OFF external I/O only by executing a device test in GX Developer.

Figure 6.38 shows the I/O operation when the Forced ON/OFF function is executed.

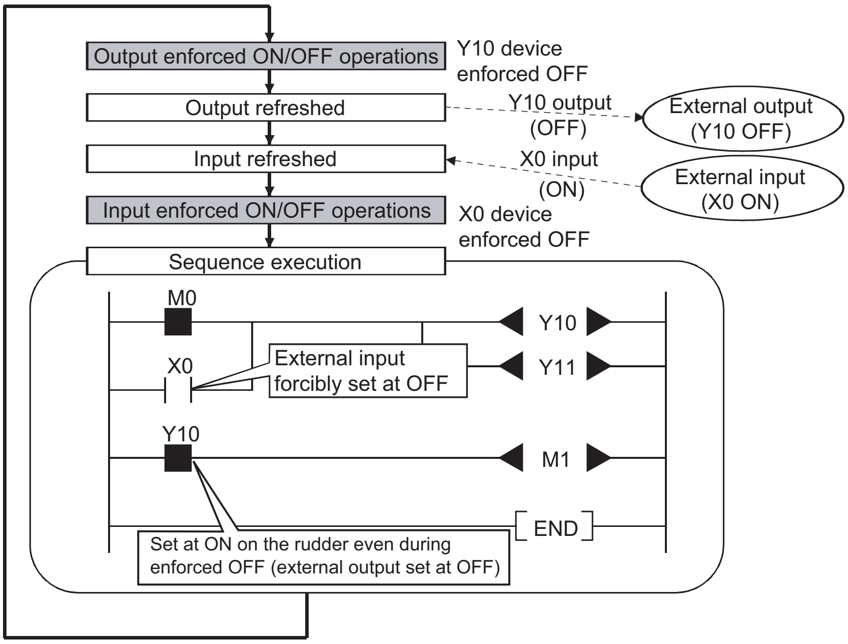

Output enforced ON/OFF operations
Y10 device enforced OFF
Output refreshed
Y10 output (OFF)
External output (Y10 OFF)
Input refreshed
X0 input (ON)
External input (X0 ON)
Input enforced ON/OFF operations
X0 device enforced OFF
Sequence execution
M0
Y10
X0
External input forcibly set at OFF
Y11
Y10
M1
END
Set at ON on the rudder even during enforced OFF (external output set at OFF)

Figure 6.38 I/O operation when the Forced ON/OFF function is executed

☒POINT

When system is switched in the redundant system, the new control system CPU module continues forced ON/OFF using the data registered in the old control system CPU module.

(2) Explanation of specifications

(a) **Status of the CPU module for which forced ON/OFF is available**

Forced ON/OFF can be registered regardless of the status (RUN/STOP) of the CPU module.

Note, however, that only input can be forcibly turned ON/OFF during a stop error. Output is performed only to device Y.

(b) **Registerable devices**

The number of devices as many as CPU module's I/O device points can be registered.

(☞ CHAPTER 2)Note6.14

Basic
Note6.14

(c) **Target I/O for forced ON/OFF**

Inputs and outputs targeted for forced ON/OFF are described below.

- I/O (X/Y) of modules on main base unit.
- I/O (X/Y) of CPU module to which LX/LY of CC-Link IE controller network module or MELSECNET/H module is refreshed
- I/O (X/Y) of CPU module to which RX/RY of CC-Link to be refreshed

When forcibly turning ON/OFF the devices outside the above refresh ranges (ex: empty slots), only I/O in the CPU module device memory are turned ON/OFF, and the results are not output externally.

☒POINT

In a multiple CPU system, inputs and outputs of controlled modules are targeted for forced ON/OFF.

Even when inputs and outputs of non-controlled modules are registered for forced ON/OFF, I/O devices in other CPU modules and I/O of modules controlled by other CPU modules are not forcibly turned ON/OFF.

(I/O devices in host CPU module turn ON/OFF.)

For details on the multiple CPU system, refer to the following manual.

☞ QCPU User's Manual (Multiple CPU System)

Basic
Note6.14

The Basic model QCPU does not support the forced ON/OFF registration operation from the Forced input output registration/cancellation screen (☞ (3) in this section).

The Basic model QCPU can forcibly turns ON/OFF external I/O only by executing a device test in GX Developer.

Basic

Note6.15

(d) Canceling ON/OFF registration data Note6.15

The registered forced ON/OFF data can be canceled using GX Developer.
Once the registered data is canceled, the status of the forced ON/OFF registered devices will be as follows.

Table6.19 Status of devices after forced ON/OFF registration data is canceled

Forced ON/OFF registered device		ON/OFF performed with sequence programs	ON/OFF not performed with sequence programs
Input	Input from modules mounted on the base unit	Assumes the ON/OFF status received from the module.	
	Input of CPU module to be refreshed from LX of CC-Link IE controller network or MELSECNET/H	Assumes the refreshed ON/OFF status from CC-Link IE controller network or MELSECNET/H.	
	Input of CPU module to be refreshed from RX of CC-Link	Assumes the refreshed ON/OFF status from CC-Link.	
	Input other than above (outside of the refresh range)	Operation result in the sequence program is reflected.	Maintains the enforced ON/OFF status.
Output	Output from modules mounted on the base unit	Outputs the results of the sequence program operations.	Retains the registered ON/OFF status.
	Output of CPU module to be refreshed from LX of CC-Link IE controller network or MELSECNET/H	Outputs the results of the sequence program operations.	Retains the registered ON/OFF status.
	Output of CPU module to be refreshed from RX of CC-Link	Outputs the results of the sequence program operations.	Retains the registered ON/OFF status.
	Output other than above (outside of the refresh range)	Assumes the result of the sequence program operations.	Maintains the enforced ON/OFF status.

Forced ON/OFF setting can be cleared by any of the following operations.

- Power supply OFF → ON
- Reset operation using the RESET/L.CLR switch or RUN/STOP/RESET switch of the CPU module
- Reset with remote reset operations

Basic

Note6.15

The Basic model QCPU does not support the forced ON/OFF registration operation from the Forced input output registration/cancellation screen (☞ (3) in this section).
The Basic model QCPU can forcibly turns ON/OFF external I/O only by executing a device test in GX Developer.

(e) Forced ON/OFF timing of external I/O

Table6.20 shows the forced ON/OFF timing of external I/O.

Table6.20 Forced ON/OFF timing

Refresh area	Input	Output
I/O modules on the base unit (X, Y)	• During END processing (input refresh) • During execution of the COM instruction (input refresh) • During execution of instructions using direct access inputs (DX) (LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF) • During execution of the RFS or MTR instruction • During execution of instructions used for a system interrupt • (UDCNT1, UDCNT2, SPD)	• During END processing (output refresh) • During execution of the COM instruction (output refresh) • During execution of instructions using direct access outputs (DY) (OUT, SET, DELTA, RST, PLS, PLF, FF, LDF, MC) • During execution of the RFS or MTR instruction • During execution of instructions used for a system interrupt • (PLSY, PWM)
I/O of CPU module to be refreshed from LX, LY of CC-Link IE controller network or MELSECNET/H	• During END processing (CC-Link IE controller network or MELSECNET/H refresh) • During execution of the COM instruction • During execution of the ZCOM instruction	
I/O of CPU module to be refreshed from RX, RY of CC-Link	• During END processing (auto refresh) • During execution of the COM instruction (auto refresh) • During execution of the ZCOM instruction (auto refresh)	

Basic
Note6.16

(f) Number of registerable devices Note6.16

The number of devices can be registered for forced ON/OFF is 32 in total.

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

Sequence program operations take precedence when used with an output Y contact.

Basic
Note6.16

The Basic model QCPU does not support the forced ON/OFF registration operation from the Forced input output registration/cancellation screen (☞ (3) in this section).
The Basic model QCPU can forcibly turns ON/OFF external I/O only by executing a device test in GX Developer.

Basic
Note6.16

Redundant
Note6.17

(h) **Checking forced ON/OFF registration and/or cancellation status**Note6.16

The forced ON/OFF registration and/or cancellation (including no-setting) status can be checked using GX Developer.

The MODE LED can be used for checking the status if at lease one device is registered. (The MODE LED flashes.)Note6.17

The Universal model QCPU can also use the special register (SD840) for checking the status.

☒POINT

1. The MODE LED also flashes in green when the Executional conditioned device test function is used. To check the registration status using the MODE LED, check the status of the Executional conditioned device test function as well. (☞ Section 6.11.4)
2. When using SD840 to check the registration and cancel status of forced ON/OFF, remind that SD840 is used to check the status for the Executional conditioned device test function as well.

Basic
Note6.16

(i) **Forced ON/OFF registration from multiple GX Developer**Note6.16

Forced ON/OFF of external I/O can be registered to the same CPU module from multiple GX Developer connected via network.

When forced ON/OFF registration is performed to the same CPU module from multiple GX Developer, note that the latest forced ON/OFF status will be registered in the CPU module.

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 executed the Forced ON/OFF function first.

When the Forced ON/OFF function is executed from multiple GX Developer, click the Update status button to update the registered data and then execute the function.

Basic
Note6.16

The Basic model QCPU does not support the forced ON/OFF registration operation from the Forced input output registration/cancellation screen (☞ (3) in this section).
The Basic model QCPU can forcibly turns ON/OFF external I/O only by executing a device test in GX Developer.

Redundant
Note6.17

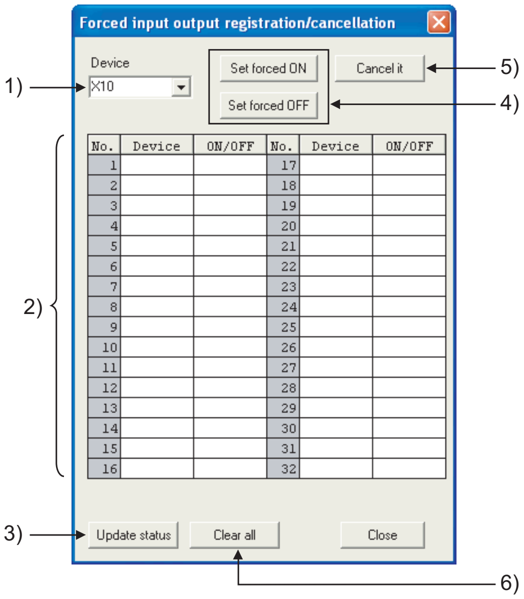
In the redundant system, the MODE LED of Redundant CPUs in both system flashes.

Basic
Note6.18

(3) Operation procedure[Note6.18]

The operation procedure is explained below.

- Register enforced ON/OFF for the specified device.
  [Online] → [Debug] → [Forced input output registration/cancellation]
- It is possible to perform enforced ON or enforced OFF for a specified device by selecting [Set forced ON] or [Set forced OFF] after the device has been specified on the [Registration forced ON/OFF] setup screen.



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 6.39 Forced input output registration/cancellation screen

Table6.21 Enforced I/O registration/cancel screen items

No.	Name of setup field	Function description
1)	Device	Enter the I/O number for which enforced ON/OFF is to be set, or for which enforced ON/OFF is to be cancelled.
2)	Registration status displayed area	Displays the registration status of registered enforced input and output.
3)	Update	Displays the registration status loaded from CPU module.
4)	Set forced ON/OFF	Performed enforced ON/OFF registration for specified devices.
5)	Cancel it	Cancels the enforced ON/OFF for registered devices.
6)	Clear all	Cancels all registered enforced I/O registrations.

Basic
Note6.18

The enforced ON registration and enforced OFF registration cannot be executed for Basic model QCPU by selecting [Online] → [Debug] → [Forced input output registration/cancellation]. This operation can be executed for Basic model QCPU by conducting device test with GX Developer.

(4) Precautions for using the Redundant CPU

(a) Systems to be registered/canceled

When the Redundant CPU is used, register/cancel the enforced I/O for the control system. (Enforced I/O cannot be registered/canceled for the control system and standby system individually.)

After system switching, register/cancel the enforced I/O for the new control system (system that was changed from the standby system to the control system by system switching).

An error (error code: 4240H) will occur if the enforced I/O is registered/canceled for the standby system.

(b) Power OFF to ON or reset in separate mode

When the control system is powered OFF or reset in the separate mode, the registered enforced I/O is canceled.

The operation performed after cancel is the same as the operation indicated in "During canceling (no operations)" in Table6.18.

However, the output of the module on the MELSECNET/H remote I/O station is processed as described below.

1) **During power OFF or reset**

The output status at power OFF or reset of the control system is held.

2) **When power is switched OFF and then ON or Reset switch is set to neutral position**

The operation result of the sequence program is output.
(As in enforced I/O cancellation)

Universal
Note6.19
Basic
Note6.20
High Performance
Note6.20
Process
Note6.20
Redundant
Note6.20

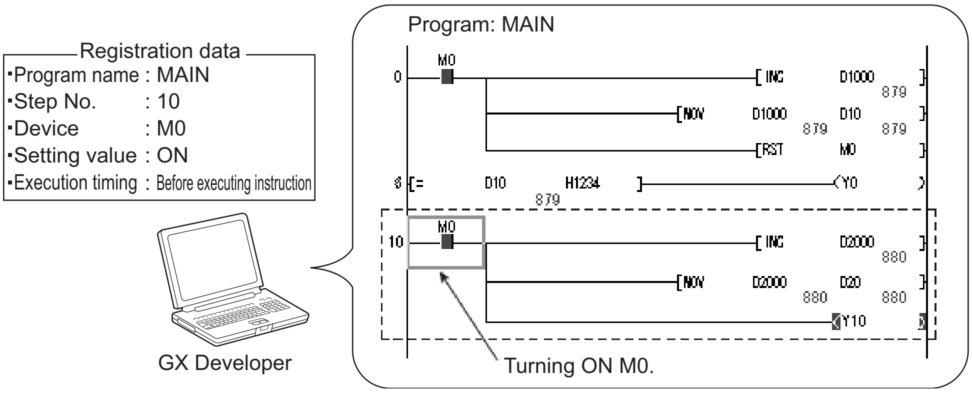
6.11.4 Executional conditioned device test

This function changes a device value within the specified step of a sequence program. This enables debugging of the specified ladder block without modifying the sequence program.*1

* 1 : The Executional conditioned device test is targeted for sequence programs only. (SFC programs are not supported.)

(1) Operation of Executional conditioned device test

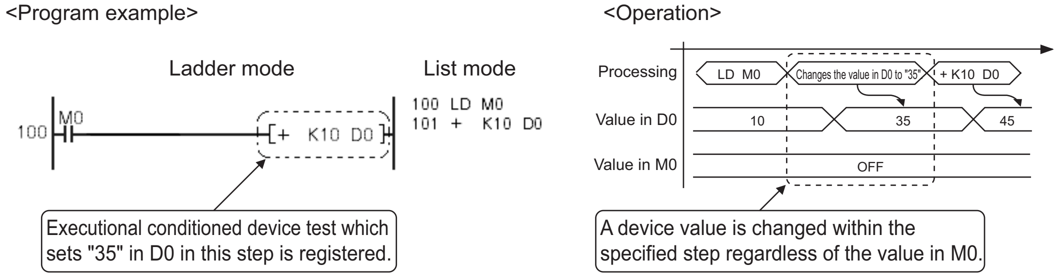
A device value will be changed based on the registration data once after the executional conditioned device test setting is registered.
The changed device value becomes effective in the step after the specified step number and later.



Registration data
- Program name : MAIN
- Step No. : 10
- Device : M0
- Setting value : ON
- Execution timing : Before executing instruction

Figure 6.40 Operation of Executional conditioned device test

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



```
100 LD M0
101 +  K10 D0
```

Executional conditioned device test which sets "35" in D0 in this step is registered.

A device value is changed within the specified step regardless of the value in M0.

Figure 6.41 Operation example of Executional conditioned device test

Universal
Note6.19

When using the Executional conditioned device test in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

Basic High Performance Process
Note6.20 Note6.20 Note6.20
Redundant
Note6.20

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not support the Executional conditioned device test.

(2) Specifiable devices and number of settable devices

Table6.22 Specifiable devices and number of settable devices

Type	Specifiable device	Number of settable devices
Bit device	X, Y, M, L, B, F, SB, V, SM, T (contact), ST (contact), C (contact), J□\ X, J□\ Y, J□\ B, J□\ SB, FX, FY, DX, and DY	32 in total (maximum)
Word device	T (current value), ST (current value), C (current value), D[*1], SD, W[*2], SW, R, ZR, Z, U□\ G□, U3E□\ G□, J□\ W□ , J□\ SW□ , and FD	
	Digit-specified bit device: X, Y, M, L, F, SM, V, B, SB, J□\ X, J□\ Y, J□\ B, and J□\ SB	
	Indirect specification (@D0): D, SD, W, SW, R, and ZR (devices specified with @)	

[*1] The extended data register (D) is also included.
[*2] The extended link register (W) is also included.

(3) Checking execution status of Executional conditioned device test

The execution status of registered Executional conditioned device test can be checked in three different ways:

- By the flash of the MODE LED in green
- By the ON status of the first bit of SD840 (Debug function usage) (☞ Appendix 2)
- By the display on the screen for checking the registration status in GX Developer (☞ (4)(d) in this section)

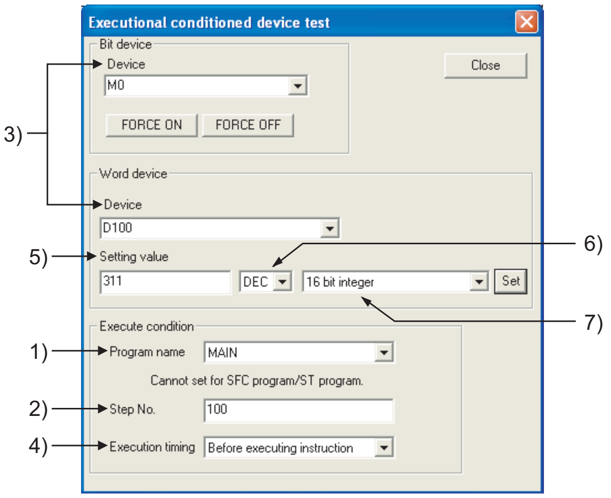
☒POINT

1. The MODE LED also flashes in green when the Forced ON/OFF function for external I/O is used. To check the execution status using the MODE LED, check the status of the Forced ON/OFF function as well. (☞ Section 6.11.3)
2. When using SD840 to check the execution status, remind that SD840 is used to check the status for the Forced ON/OFF function for external I/O as well.

(4) Operation method

(a) Registering executional conditioned device test settings

Select the registration target step number on the program editing screen in GX Developer. Then, select [Online] → [Debug] → [Executional conditioned device test] → [Register executional conditioned device test].


Executional conditioned device test

Bit device
Device: M0
FORCE ON
FORCE OFF
Close

Word device
Device: D100
Setting value: 311 DEC 16 bit integer Set

Execute condition
Program name: MAIN
Cannot set for SFC program/ST program.
Step No.: 100
Execution timing: Before executing instruction

Figure 6.42 Screen for registering executional conditioned device test settings

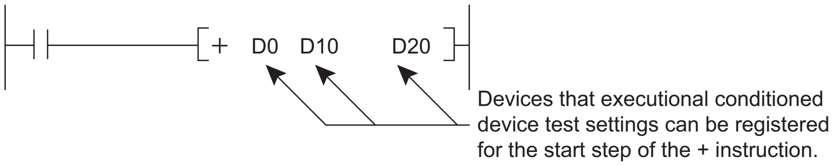
Table6.23 Items on the screen for registering executional conditioned device test settings

No.	Item	Setting range	
		For bit device	For word device
1)	Program name	Name of a program registered in the CPU module	
2)	Step No.	0 to (Step No. for the END instruction)	
3)	Device	Device listed at (2) in this section	
4)	Execution timing	Before executing instruction / After executing instruction	
5)	Setting value	- (Not specified)	Setting range corresponding to each value format
6)	DEC / HEX	- (Not specified)	DEC (decimal) / HEX (hexadecimal)
7)	Value format[*1]	- (Not specified)	16 bit integer / 32 bit integer / Real number (single precision) / Real number (Double precision)

[*1] : When a word device which is index-modified or indirectly-specified is set, only "16 bit integer" can be specified.

1) Multiple executional conditioned device test registrations for the same step number

Multiple executional conditioned device test settings can be registered for one step number.



Devices that executional conditioned device test settings can be registered for the start step of the + instruction.

Note, however, that multiple executional conditioned device test settings with same device name cannot be registered for the same step number. If this occurs, the registration data will be overwritten. (Even though the same device is specified, if the execution timing differs, two settings can be registered for one step; one for "Before executing instruction" and the other for "After executing instruction".)

☒POINT

1. When setting a word device with a different value format, a device is regarded as the same device.
   [Example]
   When a word device is set in the following order, "D100 (16 bit integer)", and then "D100 (Real number (single precision))", "D100 (Real number (single precision))" is registered.
2. When setting a device with a different modification method (such as a bit-specified word device, digit-specified bit device or index-modified device), a device is regarded as a different device.
   [Example]
   When a word device is set in the following order, "D100.F", and then "D100Z0 (Real number (single precision)), both devices are registered.

2) Step specification for executional conditioned device test registration

Any step number (0 to step number for the END instruction) in a sequence program can be specified.

☒POINT

Be sure to specify the start step of each instruction.

3) Execution timing specification

Timing of changing a device value can be specified. A device value can be changed either before or after an instruction of the specified step is executed. Figure 6.43 shows the module operation based on the execution timing.

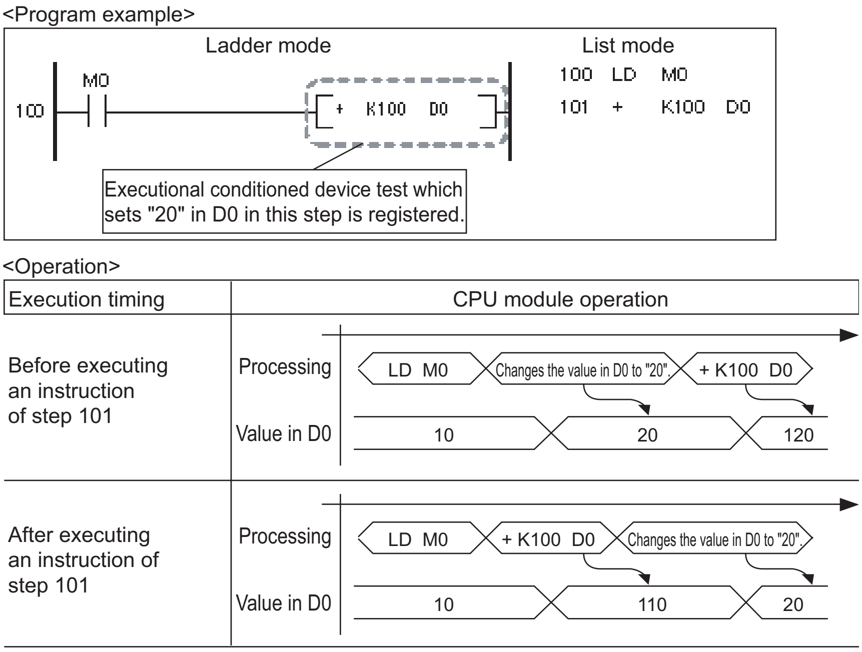

<Program example>
Ladder mode
List mode
M0
100
+ K100 D0
100 LD M0
101 + K100 D0
Executional conditioned device test which sets "20" in D0 in this step is registered.
<Operation>
Execution timing
CPU module operation
Before executing an instruction of step 101
Processing
LD M0
Changes the value in D0 to "20".
+ K100 D0
Value in D0
10
20
120
After executing an instruction of step 101
Processing
LD M0
+ K100 D0
Changes the value in D0 to "20".
Value in D0
10
110
20

Figure 6.43 CPU module operation based on execution timing

Note that there may be a case where a device value will not be changed depending on the execution timing even though the specified step is executed.
The following instructions should be noted when registering executional conditioned device test setting.

- Instructions that do not change device values[*1]
  A device value will not be changed by executing the Executional conditioned device test when the execution timing has been set to "After executing instruction", specifying the step for instructions that do not execute the next step, such as branch instructions.

  [*1] If the execution condition of an instruction is not satisfied, a device value will be changed based on the registration data.

Table6.24 lists the instructions that do not change device values.

Table6.24 Instructions that do not change device values

No.	Classification	Instruction	Operation
1	Stop	STOP	<When the execution condition for an instruction is satisfied> A device value will not be changed even when the specified step is executed. <When execution condition for an instruction is not satisfied> A device value will be changed after the specified step is executed.
2	Jump	CJ	
3		SCJ	
4		GOEND	
5	Repeat (Loop)	BREAK(P)	
6	Subroutine program call	CALL(P)	
7		FCALL(P)	
8		ECALL(P)	
9		EFCALL(P)	
10		XCALL	
11	End	FEND	A device value will not be changed even when the specified step is executed.
12	Jump	JMP	
13	Return from subroutine program	RET	
14	Return from interrupt program	IRET	

- FOR/NEXT instructions
  When the executional conditioned device test setting is registered specifying the step for the FOR or NEXT instruction, timing of device value change is different from the timing when steps for other instructions are specified. Table6.25 shows the device value change timing based on the execution timing of the executional conditioned device test setting.

Table6.25 Device value change timing when step for FOR or NEXT instruction is specified

Instruction of the specified step	Execution timing setting	
	Before executing instruction	After executing instruction
FOR	Executed once before the start of loop processing.	Executed once after the start of loop processing. (Executed before the operation of the program between the FOR and NEXT instructions.)
NEXT	Executed every loop processing. (Executed after the operation of the program between the FOR and NEXT instructions.)	Executed once after the start of loop processing.

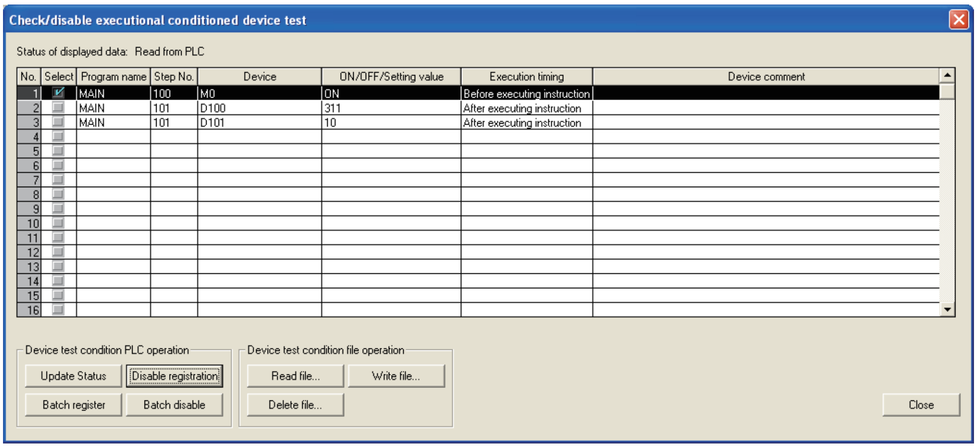
- END instruction
  When the executional conditioned device test setting is registered specifying the step for the END instruction, the execution timing is restricted to "Before executing instruction" only. If "After executing instruction" is set, the CPU module returns a registration error to the programming tool.

4) **Number of settings that can be registered simultaneously in one scan**

Eight executional conditioned device test settings can be registered into the CPU module simultaneously in one scan. When nine or more executional conditioned device test settings are to be registered simultaneously using a programming tool, they will be registered over multiple scans.

(b) Disabling executional conditioned device test settings

Select the disabling target step number on the program editing screen in GX Developer. Then, select [Online] → [Debug] → [Executional conditioned device test] → [Check/disable executional conditioned device test].



Check/disable executional conditioned device test

Status of displayed data: Read from PLC

No.	Select	Program name	Step No.	Device	ON/OFF/Setting value	Execution timing	Device comment
1	☑	MAIN	100	M0	ON	Before executing instruction	
2	□	MAIN	101	D100	311	After executing instruction	
3	□	MAIN	101	D101	10	After executing instruction	
4	□						
5	□						
6	□						
7	□						
8	□						
9	□						
10	□						
11	□						
12	□						
13	□						
14	□						
15	□						
16	□						

Figure 6.44 Screen for checking/disabling executional conditioned device test settings

Table6.26 Items on the screen for checking/disabling executional conditioned device test settings

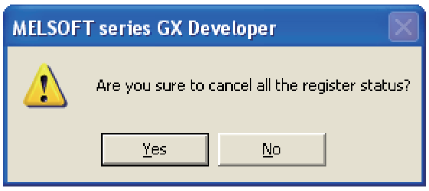
Item	Setting range	
	For bit device	For word device
Program name	Name of a program registered in the CPU module	
Step No.	0 to (Step No. for the END instruction)	
Device	Device listed at (2) in this section	
Execution timing	Before executing instruction/After executing instruction	

1) **Number of settings that can be disabled simultaneously in one scan**

Eight executional conditioned device test settings can be disabled simultaneously in one scan. When nine or more executional conditioned device test settings are to be disabled simultaneously using a programming tool, they will be disabled over multiple scans.

(c) Batch-disabling executional conditioned device test settings

Select [Online] → [Debug] → [Executional conditioned device test] → [Batch disable executional conditioned device test] in GX Developer.



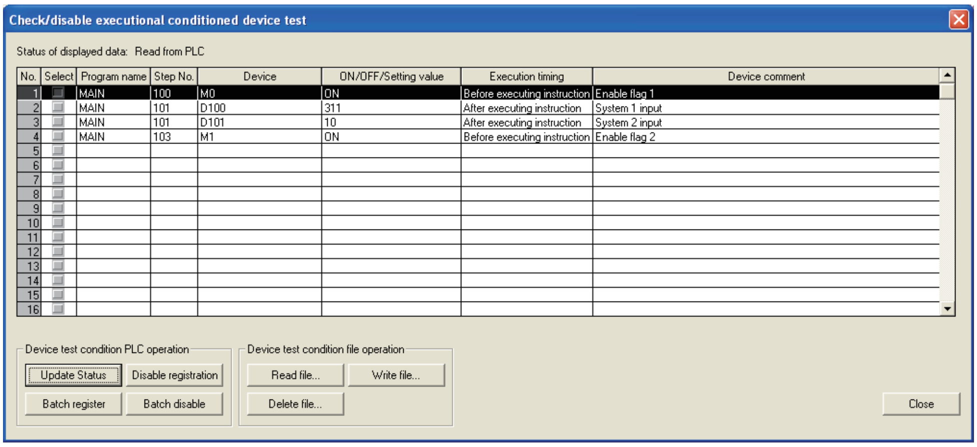
MELSOFT series GX Developer

Are you sure to cancel all the register status?

Figure 6.45 Screen for batch-disabling executional conditioned device test settings

(d) Checking executional conditioned device test settings

Select [Online] → [Debug] → [Executional conditioned device test] → [Check/ disable executional conditioned device test] in GX Developer.



Check/disable executional conditioned device test

Status of displayed data: Read from PLC

No.	Select	Program name	Step No.	Device	ON/OFF/Setting value	Execution timing	Device comment
1	□	MAIN	100	M0	ON	Before executing instruction	Enable flag 1
2	□	MAIN	101	D100	311	After executing instruction	System 1 input
3	□	MAIN	101	D101	10	After executing instruction	System 2 input
4	□	MAIN	103	M1	ON	Before executing instruction	Enable flag 2
5	□						
6	□						
7	□						
8	□						
9	□						
10	□						
11	□						
12	□						
13	□						
14	□						
15	□						
16	□						

Figure 6.46 Screen for checking executional conditioned device test settings

Remark

1. For the operation method of checking or disabling executional conditioned device test settings, refer to the following manual.
   ☞ GX Developer Operating Manual
2. Usage of the Executional conditioned device test can also be checked in the special register (SD840). (☞ Appendix 2)

(e) When executional conditioned device test settings cannot be registered or disabled

In the following cases, executional conditioned device test setting cannot be registered or disabled.

When multiple settings are to be registered, all settings cannot be registered if any of the settings applies to the cases below.

1) When executional conditioned device test settings cannot be registered

- Specified program does not exist.
- Specified step does not exist.
- Specified device does not exist.
- The number of registered executional conditioned device test settings exceeds 32.

2) When executional conditioned device test settings cannot be disabled

- Specified program does not exist.
- Specified step does not exist.
- Specified device does not exist.
- No executional conditioned device test setting has been registered.

(5) Precautions

(a) Operating function from multiple GX Developer

Executional conditioned device test setting can be registered in the same CPU module from multiple GX Developer connected via network.

Note, however, that multiple executional conditioned device test settings are registered with the same device name in the same step, the registration data will be overwritten.

When registering executional conditioned device data setting from multiple GX Developer, update the data first by clicking the Read file button, and then register the setting.

(b) Priority

If any of the following functions is set in the same step number that is specified by the executional conditioned device test setting, the Executional conditioned device test is given the priority to other functions.

- Monitor condition setting
- Sampling trace function (trace point)
- Sampling trace function (trigger point)
- Scan time measurement (start step)
- Scan time measurement (end step)

(c) Disabling executional conditioned device test settings

Executional conditioned device test setting can be disabled by any of the following operations, other than by the operation from GX Developer.

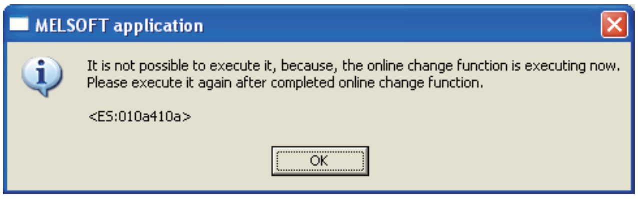
- Powering OFF → ON the programmable controller
- Performing the reset operation of the CPU module
- Writing program files to the program memory of the CPU module in STOP state
- Clearing the program memory of the CPU module in STOP state
- Formatting the program memory of the CPU module in STOP state

(d) Executing Online change (Write during RUN)

The module operation will be as follows if the Executional conditioned device test and Online change (Write during RUN) are executed simultaneously.

1) When the Executional conditioned device test is executed during execution of the Online change (Write during RUN)

The Online change completes normally. However, the Executional conditioned device test cannot be executed. The following message box appears. Execute the function again after the Online change has completed.



MELSOFT application

It is not possible to execute it, because, the online change function is executing now.
Please execute it again after completed online change function.

<ES:010a410a>

Figure 6.47 Message box when Executional conditioned device test is executed during execution of Online change

2) **When the Online change (Write during RUN) is executed during execution of the Executional conditioned device test**
The Online change completes normally. If executional conditioned device test setting has been registered in the program to be changed online, the corresponding setting will be disabled. (☞ (5)(e) in this section)

(e) **Online change (Write during RUN) of the CPU module with Executional conditioned device test registration**

1) **Online change in ladder mode**
If executional conditioned device test setting has been registered in the ladder block to be changed online, the CPU module disables the corresponding setting.
[Example 1]
Step numbers of registrations 1 to 3 are specified in the executional conditioned device test settings. When the ladder block including the registration 2 is changed and the Online change (Write during RUN) is executed, the registration 2 is disabled during execution of the Online change. Since the registrations 1 and 3 are not included in the change target program, they are not disabled.

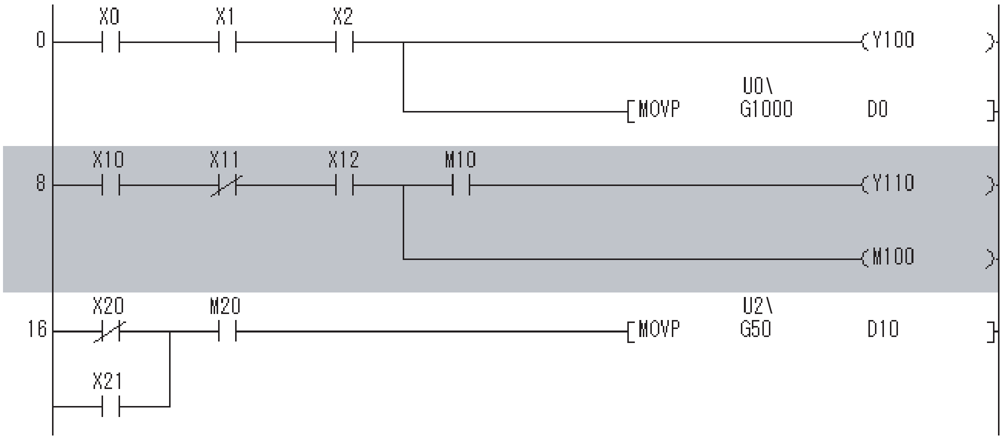
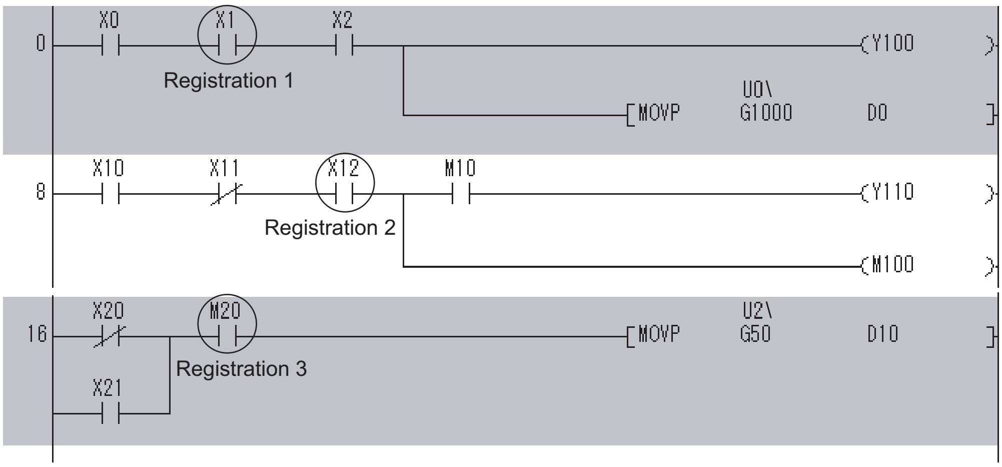


Figure 6.48 Executing Online change for a ladder block

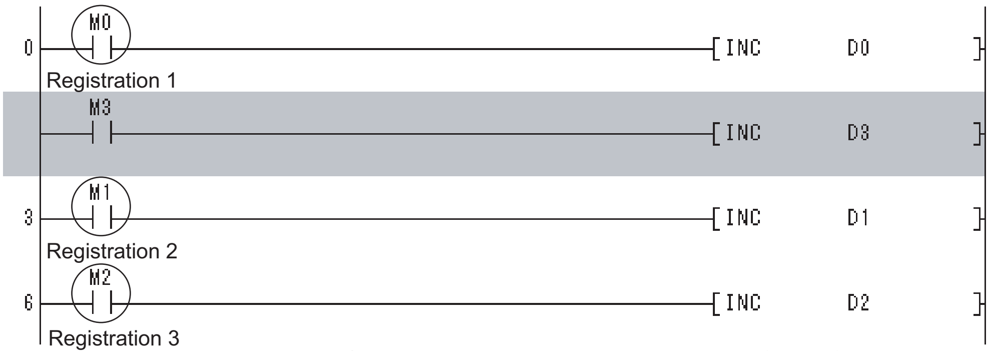
[Example 2]
When multiple ladder blocks are targeted for the Online change, ladder blocks between the change target ladder blocks will be included in the change target. If the Online change shown in Figure 6.49 is executed, the registrations 1 to 3 are all disabled.



*The shaded area is the target ladder block for the Online change.

Figure 6.49 Executing Online change for multiple ladder blocks

[Example 3]
When a ladder block is added by the Online change, the executional conditioned device test setting included in the ladder block followed after the added ladder block. If the Online change shown in Figure 6.50 is executed, the registration 2 is disabled.



*The shaded area is the target ladder block for the Online change.

Figure 6.50 Adding ladder block by Online change

2) **Online change in file units**
All executional conditioned device test settings registered to the program of the Online change target file are disabled.

(f) Device specification by index modification

If index-modified device name is specified to register the executional conditioned device test setting, the module does not check whether the specified device is within the setting range.

If the index-modified device is out of the device range or on the boundary of devices, a device value will not be changed within the specified step.

(g) Device specification by indirect specification

If indirectly-specified device name is specified to register the executional conditioned device test setting, the module does not check whether the specified device is within the setting range.

If the indirectly-specified device is out of the device range or on the boundary of devices, a device value will not be changed within the specified step.

(h) File register specification

If file register is specified to register the executional conditioned device test setting, the module does not check the file register file assignment and the file register number range.

If the file register file is not assigned or the specified file register number is out of the file register range, a file register value will not be changed within the specified step.

6.12 Writing in Program during CPU Module RUN

When the High Performance model QCPU is in the RUN status, you can write programs or files in any of the steps shown in Table6.27.

Table6.27 Online change types

CPU module	Online change type
Basic model QCPU	• Writing data in the circuit mode during RUN (Section 6.12.1) • Writing a batch of files during RUN (Section 6.15.2)
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	• Writing data in the circuit mode during RUN (Section 6.12.1) • File-write online change (Section 6.12.2) • Writing a batch of files during RUN (Section 6.15.2)

POINT

Refer to the following manual for online change (order of write to control system and standby system, whether tracking can be executed at execution of online change, etc.) when the Redundant CPU is used.

☞ QnPRHCPU User's Manual (Redundant System)

6.12.1 Online change in ladder mode

(1) Writing data in the circuit mode during RUN Status

Writing data in the circuit mode during RUN is a function to write a program during the CPU module RUN status.

The program can be changed without stopping the process in CPU module program by performing writing data in the circuit mode during RUN status.

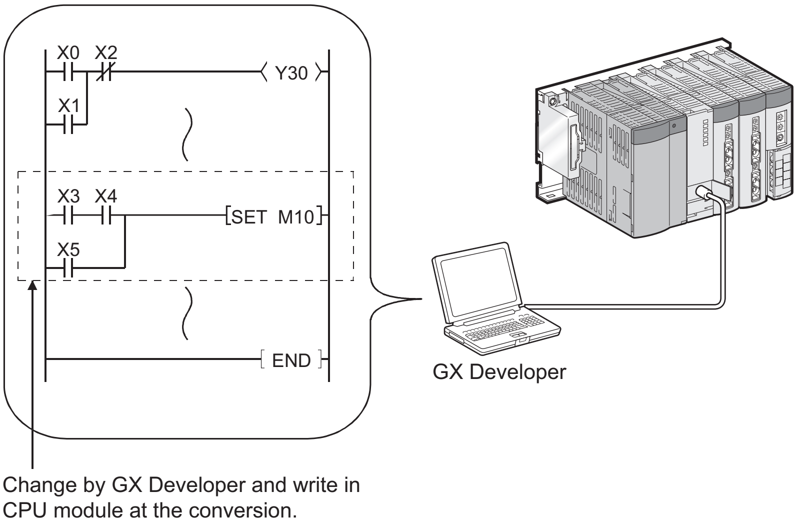

X0 X2
Y30
X1
X3 X4
SET M10
X5
END
GX Developer
Change by GX Developer and write in CPU module at the conversion.

Figure 6.51 Outline of online change in ladder mode

Also, online change is enabled from GX Developer connected to another station on the network.

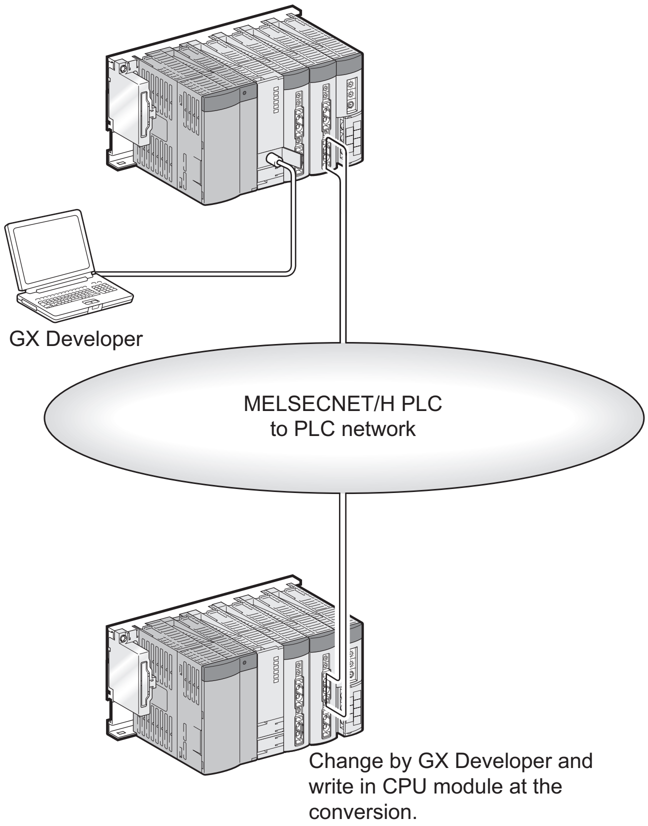

GX Developer
MELSECNET/H PLC to PLC network
Change by GX Developer and write in CPU module at the conversion.

Figure 6.52 Outline of online change via network

(2) Memory enabled for online change

(a) Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU

The memory enabled for online change is the program memory.

(b) Universal model QCPU

The memory enabled for online change is the program cache memory (program memory).

(3) Number of steps enabled for online change at once

A maximum of 512 steps can be written at once during RUN.

Basic Redundant Universal
Note6.21 Note6.21 Note6.21

(4) The execution timing of low speed execution type programNote6.21

In the low speed execution type program, data are written online in the END processing of the next scan after completion of all the program execution.
Note the execution of low speed execution type program is suspended during online change.

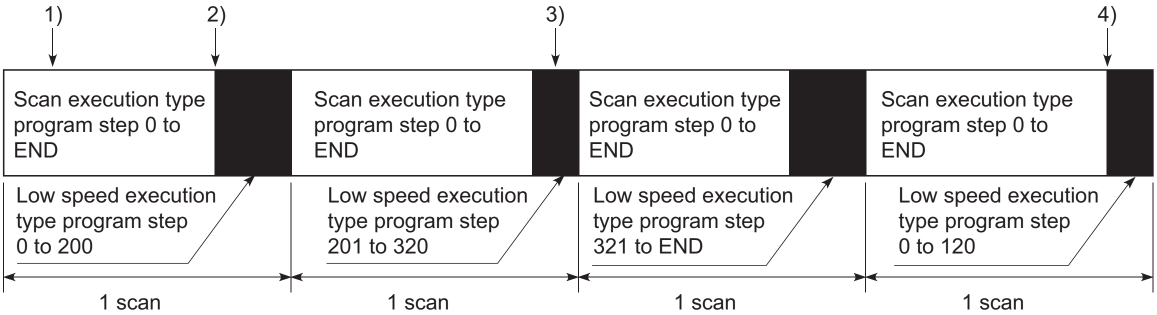

1)
2)
3)
4)
Scan execution type program step 0 to END
Scan execution type program step 0 to END
Scan execution type program step 0 to END
Scan execution type program step 0 to END
Low speed execution type program step 0 to 200
Low speed execution type program step 201 to 320
Low speed execution type program step 321 to END
Low speed execution type program step 0 to 120
1 scan
1 scan
1 scan
1 scan

Figure 6.53 Online change processing at execution of low speed execution type program

1) : RUN write command of scan execution type program
2) : RUN write execution of scan execution type program
3) : RUN write command of the low speed execution type program
4) : RUN write execution of the low speed execution type program

Basic Redundant Universal
Note6.22 Note6.22 Note6.22

(5) Online change during execution of PLOADP, PUNLOADP or PSWAPP instructionNote6.22

If the online change is executed while the PLOADP, PUNLOADP or PSWAPP instruction is executed, the processing will enter into a stand-by status for the online change. If the online change is executed while the PLOADP, PUNLOAD or PSWAPP instruction is executed, the execution of the instruction is delayed until the online change is executed.

Basic Redundant Universal
Note6.21 Note6.21 Note6.21

Since the Basic model QCPU, Redundant CPU and Universal model QCPU cannot use low speed execution type programs, it is not necessary to be conscious of online change of low speed execution type programs.

Basic Redundant Universal
Note6.22 Note6.22 Note6.22

The Basic model QCPU, Redundant CPU and Universal model QCPU cannot change the program files during RUN by the PLOADP, PUNLOADP or PSWAPP instruction.

(6) Changing the "allocate memory for online change" for online change

The following explains the precautions for changing the "allocate memory for online change" for online change.

(a) The allocate memory for online change

A program file has steps secured for online change to support online change that changes the program file capacity.
The program file capacity is the sum of the created program capacity and "allocate memory for online change".

(b) When program file capacity increases from the secured capacity

If the capacity secured for the program file capacity (capacity including the allocate memory for online change) is exceeded at the time of online change, the allocate memory for online change can be re-set for online change.
Hence, online change can be executed when the user memory area has a free area.

(c) Scan time increased when allocate memory for online change are set again

The scan time increases, when the online change reserve step is re-set in online change.
For increased scan time, refer to Section 11.1.3.

Basic High Performance Process
Note6.23 Note6.23 Note6.23
Redundant
Note6.23

(7) For the prohibited operations[Note6.23] when writing data during RUN, when changing T/C setting value, or at the time of transferring from a program cache memory to a program memory

For the prohibited operations when writing data during RUN, when changing T/C setting value, or at the time of transferring from a program cache memory to a program memory, refer to Section 6.12.3 (2).

(8) Instruction which does not operate normally when writing data during RUN

For instructions do not operate normally at write during RUN,
refer to Section 6.12.3 (3) in this section.

Basic High Performance Process
Note6.23 Note6.23 Note6.23
Redundant
Note6.23

Since a program cache memory is not used for the Basic model QCPU, High Performance model QCPU, Process CPU, or Redundant CPU, data transfer from the program cache memory to the program memory can be ignored.

Basic
Note6.24

6.12.2 File-write in RUN

(1) File-Write in RUN function

The file-write in RUN function is used to write a batch of files to the CPU module as shown in Table6.28.

Table6.28 Files that can be written during RUN by the CPU module

Memory Name	CPU module built-in memories			Memory Card (RAM)	Memory Card (ROM)	
	Program Memory	Standard RAM	Standard ROM	SRAM Card	Flash Card	ATA Card
Parameter	×	×	×	×	×	×
Intelligent function module parameter	×	×	×	×	×	×
Program	○	×	△*2	○	×	○
Device comment	○	×	△*1*2	△*1	×	△*1
Device initial value	×	×	×	×	×	×
File register	×	△*1	×	△*1	×	×
Local device	×	×	×	×	×	×
Sampling trace file	×	○	×	○	×	×
Failure history data	×	×	×	×	×	×
PLC user data	×	×	△*2	×	×	○

○ : Writable data, × : Unwritable data, △ : Restricted

* 1 : Writable only when not being accessed by a sequence program.
* 2 : Writable for the Universal model QCPU.

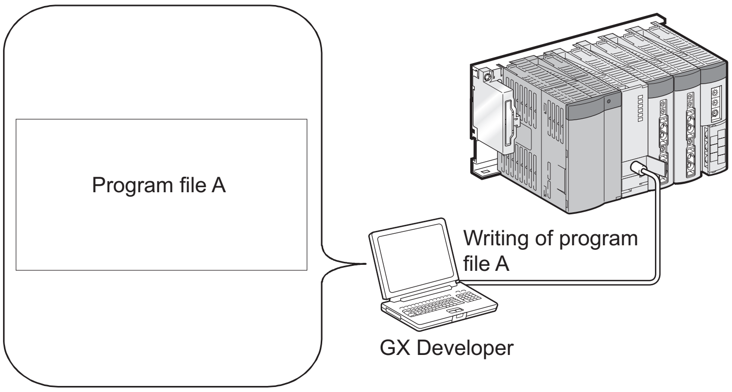


Figure 6.54 Outline of file-write in RUN

Basic
Note6.24

The Basic model QCPU cannot execute file-write in RUN.

(2) Availability of writing a file during RUN

(a) For the High Performance model QCPU, Process CPU, and Redundant CPU

Table 6.24 shows the availability of writing a file during RUN.

Table6.29 Availability of writing a file during RUN

Free area greater than the program file to be written		riting a file during RUN
Program memory	Memory card	
Yes	No/Yes	Executable
No	Yes	Executable*1
No	No	Not executable

* 1: A file cannot be written during RUN if the trailing edge instruction is included in the write program file when [Trailing edge instructions are not executed] is set in the option menu “instruction operating setting at writing during RUN” of GX Developer.
However, if a write program is the SFC program, the file can be written during RUN even if the trailing edge instruction is included in the write program file.
(For the operation of setting [Trailing edge instructions are not executed] in the “instruction operating setting at writing during RUN” of GX Developer, refer to Section 6.12.3.)

* 2: A program file can be written to the program memory during RUN in the Universal model QCPU.

(b) For the Universal model QCPU

A file can be written during RUN regardless of whether empty area exists enough to write a program file.

(3) Increase in scan time

The scan time increases when the file is written during RUN.
For increased scan time, refer to Section 11.1.3.

(4) When file is accessed by sequence program instruction

No access can be made from an instruction in a program to the file during online change.
The instruction to make access to a file during online change will not be executed.

(5) Disabled file-write in RUN from multiple locations

Do not execute file write in RUN simultaneously from multiple locations.
Doing so may erase program file contents.

High Performance Note6.25 | Process Note6.25

Universal Note6.26

(6) Writing a SFC program file in RUN Note6.25, Note6.26

After writing SFC program file in RUN, the program starts from the beginning.

Basic Note6.27 | Redundant Note6.27 | Process Note6.27

(7) For the prohibited operations Note6.27 when writing data during RUN, when changing T/C setting value, or at the time of transferring from a program cache memory to a program memory

For the prohibited operations when writing data during RUN, when changing T/C setting value, or at the time of transferring from a program cache memory to a program memory, refer to Section 6.12.3 (2).

Redundant Note6.27

(8) Instruction which does not operate normally when writing data during RUN

For instructions do not operate normally at write during RUN,
refer to Section 6.12.3 (3) in this section.

High Performance Note6.25 | Process Note6.25

When writing a SFC program file online in High Performance model QCPU or Process CPU, confirm the versions of the CPU module and the GX Developer.
(☞ Appendix 4.2, Appendix 4.3)

Universal Note6.26

The SFC program file cannot be written during RUN in the Universal model QCPU.

Basic Note6.27 | Redundant Note6.27 | Process Note6.27

Redundant Note6.27

Since a program cache memory is not used for the Basic model QCPU, High Performance model QCPU, Process CPU, or Redundant CPU, data transfer from the program cache memory to the program memory can be ignored.

6.12.3 Precautions for online change

The following shows precautions at the time of writing data during RUN.

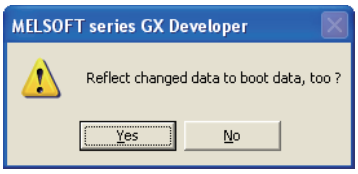
(1) Online change performed during boot run

Operation is performed as described in Table6.30 when online change is performed during boot run.

Table6.30 Status of boot source program when writing data during RUN

Boot source		Status of boot source program			
		Basic model QCPU	▪ High Performance model QCPU ▪ Process CPU	Redundant CPU	Universal model QCPU
Memory card	SRAM card ATA card	----	Changed[*1]	Changed*[1]	Not changed[*2]
	Flash card	----	Not changed[*2]	Changed[*1]	Not changed[*2]
Standard ROM		Not changed[*2]	Not changed[*2]	Changed[*1]	----

* 1: The boot source program can be changed by clicking "Yes"on the following message box. When this operation is executed simultaneously with online change,the online change completion takes time.



MELSOFT series GX Developer

Reflect changed data to boot data, too ?

If “No” is selected, boot source program will not be changed.
Turn OFF power supply of the CPU module after writing data during RUN. Or, write data of a program memory to a memory card Note6.28 or the standard ROM Note6.29 before resetting the CPU module.

Basic
Note6.28

Universal
Note6.29

* 2: Turn OFF power supply of the CPU module after writing data during RUN. Or, write data of a program memory to a memory card Note6.28 or the standard ROM Note6.29 before resetting the CPU module.

* 3: When boot source memory is Flash card or standard ROM, files of boot source are deleted and replaced by files on program memory. Set files in the boot source memory to be transferred to the program memory by the boot file setting of the parameter.

Basic
Note6.28

The Basic model QCPU cannot use a memory card.

Universal
Note6.29

The Universal model QCPU cannot boot data from the standard ROM to program memory.
(☞ Section 5.2.3)

Basic Redundant Process
Note6.30 Note6.30 Note6.30
Redundant
Note6.30

(2) Operations prohibited during online change, TC setting value change or data transfer from the program cache memory to the program memoryNote6.30

Do not execute the following operations during online change, TC setting value change or data transfer from the program cache memory to the program memory.

(a) Power OFF or reset operationNote31

If power OFF or reset operation is performed during online change, TC setting value change or data transfer from the program cache memory to the program memory, the operation does not complete normally.
Execute Write to PLC again if power OFF or reset operation is performed.

(b) Operations from GX Developer

The following operations cannot be executed during online change, TC setting value change or data transfer from the program cache memory to the program memory. Execute them after the executing operation is completed.
GX Developer displays an error when executed.

- Online change (Ladder mode, writing files during RUN, and function block)
- TC setting value change
- Data transfer to the program memory

Basic Redundant Process
Note6.30 Note6.30 Note6.30
Redundant
Note6.30

Since the Basic model QCPU, High Performance model QCPU, Process CPU and Redundant CPU do not use the program cache memory, it is not necessary to be conscious of data transfer from the program cache memory to the program memory.

(3) Instructions do not operate normally at online change

When online change is performed, the following instructions do not operate normally.

- Trailing edge instruction
- Leading edge instruction
- SCJ instruction

(a) Trailing edge instruction

The trailing edge instruction is executed when the instruction is in a writing range even the execution condition(ON → OFF) is not established at the completion of online change.

For preventing the trailing edge instruction execution, refer to POINT in this section.

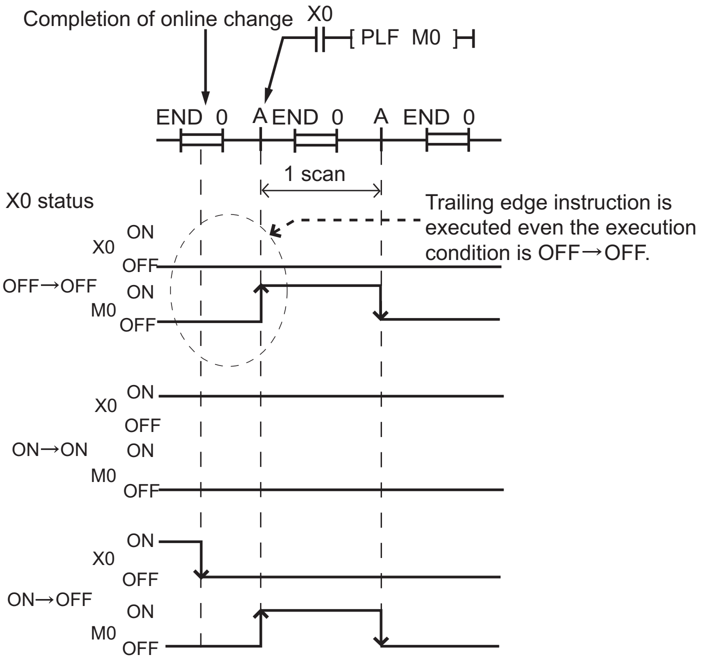

Completion of online change
X0
PLF M0
END 0
A
END 0
A
END 0
1 scan
X0 status
Trailing edge instruction is executed even the execution condition is OFF→OFF.
X0
ON
OFF
OFF→OFF
M0
ON
OFF
X0
ON
OFF
ON→ON
M0
ON
OFF
X0
ON
OFF
ON→OFF
M0
ON
OFF

Figure 6.55 Trailing edge instruction operation

The corresponding instructions are LDF,ANDF,ORF,MEF,PLF,FCALLP,EFCALLP.

(b) Leading edge instruction

The leading edge instruction is not executed when the instruction is in a writing range even the execution condition(OFF → ON) is established at the completion of online.

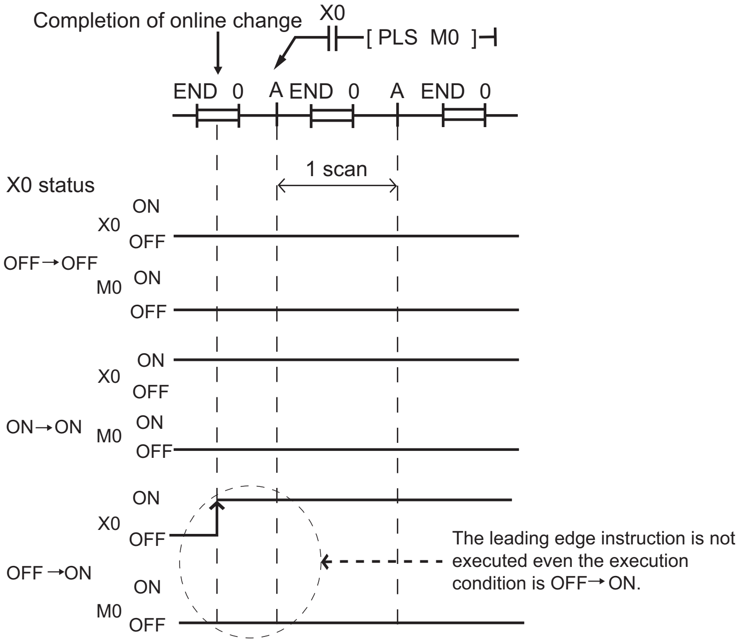

Completion of online change
X0
[PLS M0]
END 0
A END 0
A END 0
1 scan
X0 status
X0
ON
OFF
OFF→OFF
M0
ON
OFF
X0
ON
OFF
ON→ON
M0
ON
OFF
X0
ON
OFF
OFF→ON
M0
ON
OFF
The leading edge instruction is not executed even the execution condition is OFF→ON.

Figure 6.56 Leading edge instruction operation

The corresponding instructions are PLS,□P.

(c) SCJ instruction

When the SCJ instruction is in a writing range and the execution condition is ON at the completion of the writing, a jump to the designated pointer occurs even in a scan cycle.

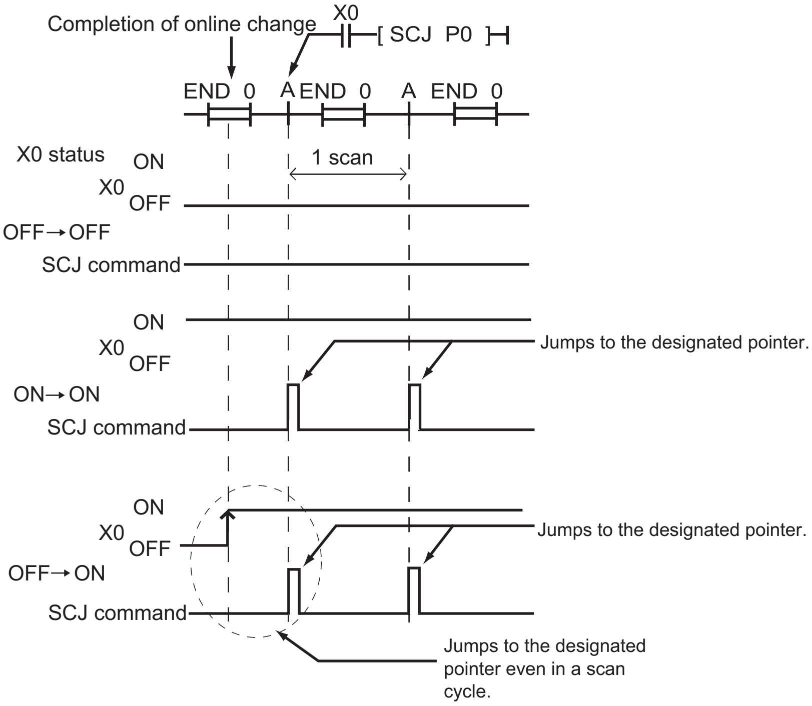

Completion of online change
X0
[SCJ P0]
END 0
A END 0
A END 0
1 scan
X0 status
X0
ON
OFF
OFF→OFF
SCJ command
X0
ON
OFF
ON→ON
SCJ command
Jumps to the designated pointer.
X0
ON
OFF
OFF→ON
SCJ command
Jumps to the designated pointer.
Jumps to the designated pointer even in a scan cycle.

Figure 6.57 SCJ instruction operation

☒POINT

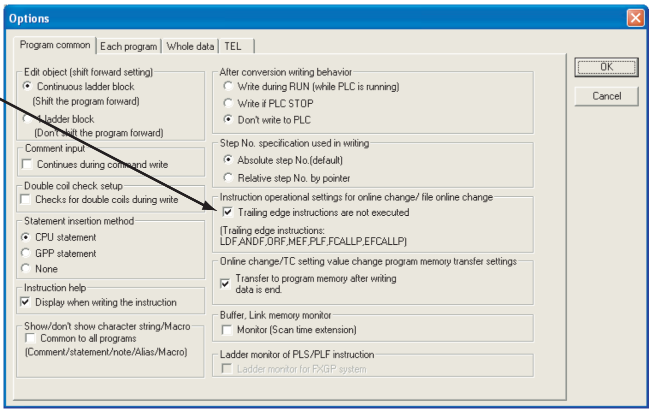
Basic High Performance Process
Note6.31 Note6.31 Note6.31

Redundant
Note6.31

The event that the trailing edge instruction is executed even when the execution condition (ON → OFF) is not established after the completion of online change can be avoided by setting the "Instruction operational settings for online change/ file online change" on the Options screen in GX Developer. Note6.31

The default value depends on the CPU module.

- High Performance model QCPU, Process CPU, and Redundant CPU
  The "Trailing edge instructions are not executed" checkbox is not checked.
- Universal model QCPU
  The "Trailing edge instructions are not executed" checkbox is checked.



The trailing edge instructions whose execution condition set to OFF are not executed by checking the check box.

Options

Program common | Each program | Whole data | TEL

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,ANDF,ORF,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

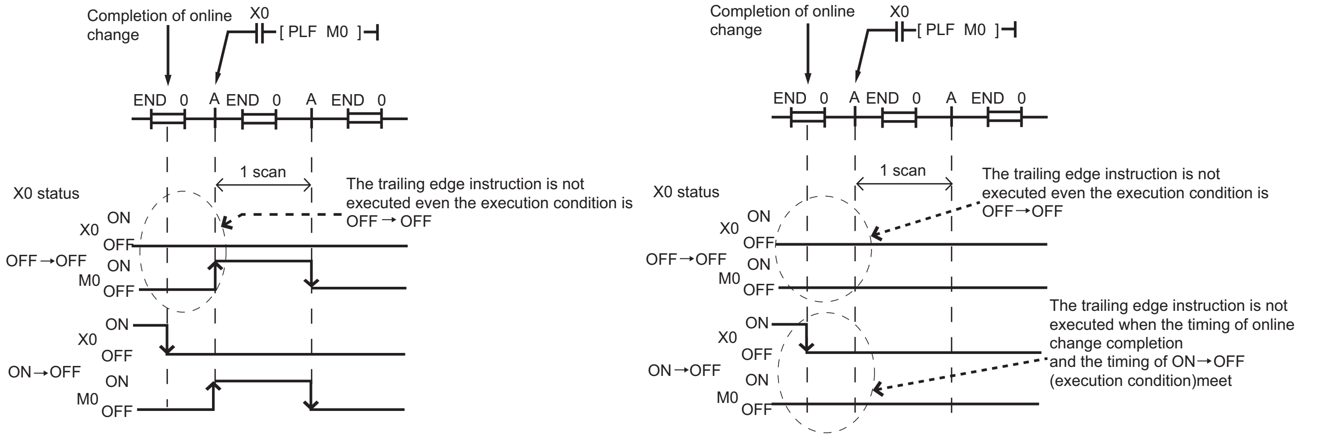
Figure 6.58 Option menu screen in GX Developer

Basic High Performance Process
Note6.31 Note6.31 Note6.31

Redundant
Note6.31

The Basic model QCPU does not support the selection of trailing edge instruction execution/non-execution at online change.
When selecting the trailing edge instruction execution/non-execution at online change in the High Performance model QCPU, Process CPU, and Redundant CPU, check the versions of CPU module and GX Developer. (☞ Appendix 4)

Figure 6.59 shows the trailing edge instruction operation based on the "Instruction operational settings for online change" setting in GX Developer.


Completion of online change
X0
[PLF M0]
END 0 A END 0 A END 0
1 scan
X0 status
The trailing edge instruction is not executed even the execution condition is OFF → OFF
ON
X0
OFF
OFF→OFF
ON
M0
OFF
ON
X0
OFF
ON→OFF
ON
M0
OFF
Completion of online change
X0
[PLF M0]
END 0 A END 0 A END 0
1 scan
X0 status
The trailing edge instruction is not executed even the execution condition is OFF→OFF
ON
X0
OFF
OFF→OFF
ON
M0
OFF
The trailing edge instruction is not executed when the timing of online change completion and the timing of ON→OFF (execution condition)meet
ON
X0
OFF
ON→OFF
ON
M0
OFF

(a) Operation when "Trailing edge instructions are not executed" is unchecked.

(b) Operation when "Trailing edge instructions are not executed" is checked.

Figure 6.59 Operation comparison of trailing edge instruction

(4) Precautions for using Universal model QCPU

(a) Write to program memory during online change or TC setting value change

Contents changed during online change or TC setting value change are transferred to the program memory automatically at the same time as data is written to the program cache memory.

The time required for writing data during RUN and changing T/C setting value is prolonged due to automatic transfer of data to a program memory by the time shown in Table 6.26.

Table6.31 Prolonged time due to automatic transfer of data to a program memory

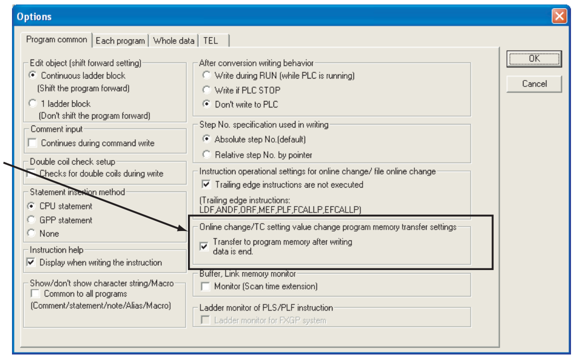
CPU model	Transfer time
Q02UCPU	Ts × 320 + 4.8 (s)
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU	Ts × 260 + 4.7 (s)
Q13UDHCPU, Q13UDEHCPU	Ts × 600 + 8.0 (s)
Q26UDHCPU, Q26UDEHCPU	Ts × 1100 + 15.0 (s)

Ts: Scan time (s)

In addition, since the number of write to the program memory (Flash ROM) is limited (up to hundred thousand times), set the automatic transfer of data to the program memory to be invalid if writing data during RUN or TC setting value change is executed frequently.

☒POINT

1. The automatic transfer to the program memory can be set to invalid in the Options dialog box of GX Developer.



Data is not transferred to the program memory automatically by unchecking.

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,ANDF,ORF,MEF,PLF,FCALLP,EFCALLP)

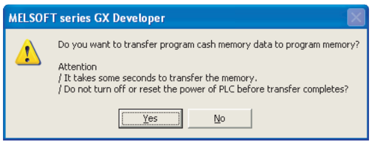
Online change/TC setting value change program memory transfer settings
☑ Transfer to program memory after writing data is end.

Buffer, Link memory monitor
□ Monitor (Scan time extension)

Ladder monitor of PLS/PLF instruction
□ Ladder monitor for FXGP system

Figure 6.60 Online change/TC setting value change program memory transfer settings screen

2. When the automatic transfer is set to invalid, the following message box appears after online change is completed.



MELSOFT series GX Developer

Do you want to transfer program cash memory data to program memory?

Attention
/ It takes some seconds to transfer the memory.
/ Do not turn off or reset the power of PLC before transfer completes?

When "Yes" is selected, data is transferred to the program memory.
When "No" is selected, data is not transferred to the program memory.

3. When "No" is selected in the message box, be sure to execute Program memory batch transfer by GX Developer.
(☞ GX Developer Operating Manual)

Universal UD Note6.32

Program transfer status can be checked by the special relay (SM165).
Note6.32

When the SM165 is ON, the program memory batch transfer has not completed.
When the SM165 is OFF, the program memory batch transfer has completed.

Universal UD Note6.32

When checking the transfer status to the program memory with a Universal model QCPU, check the version of the CPU module. (☞ Appendix 4.5)

6.13 Execution Time Measurement

(1) Definition

This is a function to display the processing time of the program being executed.

(2) Applications and types of execution time measurement

This is used to find out the effect of each program's processing time on the total scan time.

There are three functions to the execution time measurement.

- Program monitor list : (☞ Section 6.13.1)
- Interrupt program monitor list : (☞ Section 6.13.2)
- Scan time measurement : (☞ Section 6.13.3)

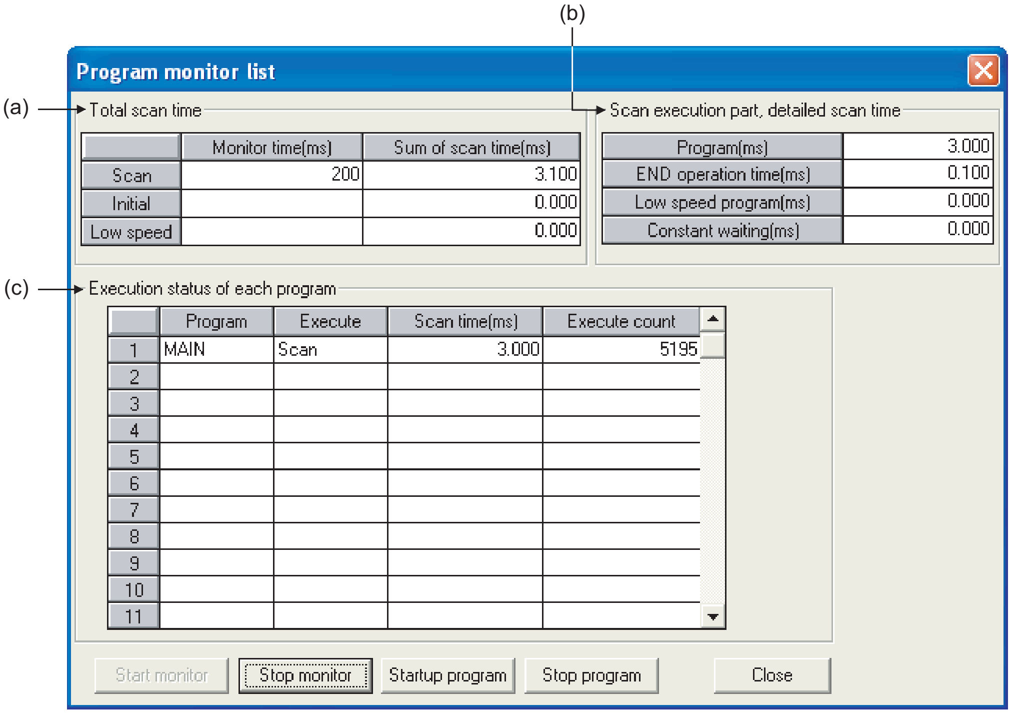
6.13.1 Program list monitor

(1) Definition of Program Monitor List

This is a function to display the processing time of the program being executed.
The scan time, number of times executed, and processing time by item can be displayed for each program.

(2) Using the Program Monitor List

Choose [Online] → [Monitor] → [Program monitor list]. "Program Monitor List" dialog box appears on screen.
Figure 6.61 shows an example of executing program monitor list.



Total scan time

	Monitor time(ms)	Sum of scan time(ms)
Scan	200	3.100
Initial		0.000
Low speed		0.000

Scan execution part, detailed scan time

Program(ms)	3.000
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	3.000	5195
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				

Figure 6.61 Program list monitor

Basic Redundant Universal
UD
Note6.33 Note6.33 Note6.33

(a) **Total Scan Time**[Note6.33]

The monitor time set in WDT(the watchdog timer) of "PLC RAS" tab screen in the (PLC) "Parameter" dialog box and total scan time for each program type are displayed.

1) **Monitor Time**

Indicates the monitor time of each program.
If the scan time exceeds this time, the CPU module displays the watchdog timer error.

2) **Sum of Scan Time**

The total time in each item stated in "Scan Time Details for Scan Execution" are displayed.
"Constant" indicates the constant scan waiting time when the setting is made for constant scan.

(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 operation time is displayed.

Basic Redundant Universal
UD
Note6.34 Note6.34 Note6.34

3) **Low speed program**[Note6.34]

This indicates the total execution time of a low speed execution type program when making the setting for the execution time of a low speed execution type program or constant scan time.

4) **Constant waiting**

The constant scan waiting time is displayed when setting the constant scan time. However, when the low speed execution type program execution time is set as well, this value is 0.000 ms.

Basic Redundant Universal
UD
Note6.33 Note6.33 Note6.33

In the case of the Basic model QCPU, only the monitor time of the scan execution type program can be displayed. In the case of the Redundant CPU and Universal model QCPU does, the monitor time of the low speed execution type program cannot be displayed.

Basic Redundant Universal
UD
Note6.34 Note6.34 Note6.34

In the case of the Basic model QCPU, Redundant CPU and Universal model QCPU, the item for low speed program cannot be displayed.

Basic
Note6.35

(c) Execution status of each program

The execution status of program specified at the Program tab screen in the (PLC) Parameter dialog box is displayed.Note6.35

1) Program

The program name is displayed in the order set in the parameter.

2) Execute

The program type set in the parameter is displayed.

3) Scan Time

The actual scan time (current value) is displayed. At the program stop (wait) status, the scan time is displayed as 0.000 ms.

4) Execute count

The number of times the program was executed is displayed, setting the starting point of when the measurement is started as "0". The number of execution times is displayed up to 65535 times and returns to 0 when the 65536th time measurement is made. The ex times remains 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 executions displayed differs depending on the CPU module.

- High Performance model QCPU, Process CPU, Redundant CPU
  The number of non-execution processing executions is not added.
- Universal model QCPU
  The number of non-execution processing executions is added.

Refer to the following manual for details of the POFF instruction.

☞ QCPU(Q Mode)/ QnACPU Programming Manual (Common Instructions)

Basic
Note6.35

Since the Basic model QCPU cannot execute multiple programs, the execution status of each program is displayed as shown below.

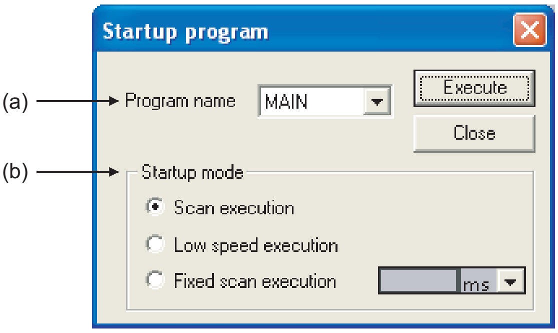
Program	: MAIN or MAIN-SFC
Execute	: Scan

Basic Universal
Note6.36 Note6.36

(3) Startup programNote6.36

Program can be started on the program list monitor screen.

Clicking the Startup program button displays the following dialog box.



Startup program

Program name: MAIN ▾

Execute

Close

Startup mode

● Scan execution

○ Low speed execution

○ Fixed scan execution ms ▾

Figure 6.62 Startup program screen

(a) Program name

Select the program set in the program setting of the PLC parameter dialog box. A program name cannot be entered as desired.

(b) Startup mode

Redundant Universal
Note6.37 Note6.37

The stand-by type program can be changed to "Scan execution", "Low speed execution"Note6.37 or "Fixed scan execution".
The value set in the program setting of the PLC parameter dialog box is displayed as the default value of fixed scan execution. ms or scan be selected as the unit.

Basic Universal
Note6.36 Note6.36

In the case of Basic model QCPU, a program cannot be started from the program list monitor screen.

Redundant Universal
Note6.37 Note6.37

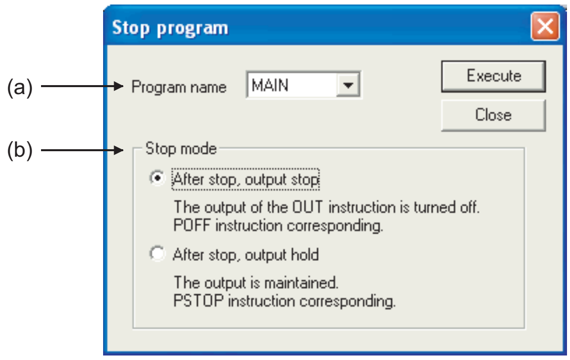
Since the Redundant CPU and Universal model QCPU cannot use low speed execution type programs, "Low speed execution" cannot be selected in the start mode.

Basic Universal
Note6.38 Note6.38

(4) Stop programNote6.38

Program can be stopped on the program list monitor screen.

Clicking the Stop program button on the program list monitor screen (☞ (2) in this section) displays the following dialog box.



Stop program

Program name: MAIN ▾

Execute

Close

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 6.63 Stop program screen

(a) Program name

Select the program set in the program setting of the PLC parameter dialog box.
A program name cannot be entered as desired.

(b) Stop mode

- Executing "After stop, output stop" for the scan execution type turns off the output (non-execution processing) at the next scan. The program is put in the standby status at and after the next scan. (This operation is the same as performed when the POFF instruction is executed.)

Redundant Universal
Note6.39 Note6.39

- Executing "After stop, output stop" for the low speed execution type suspends the execution of the low speed execution type and turns off the output at the next scan. The program is put in the standby status at and after the next scan. Note6.39
  (This operation is the same as performed when the POFF instruction is executed.)
- Executing "After stop, output stop" for the standby program stops the program after one-scan OFF is executed as scan execution.
  For this reason, "Execute count" is also increased by 1.
- "Execute count" is also increased by 1 if an error occurs in the RET/IRET instruction during execution of one-scan OFF in the standby program.
  At this time, the execution type is "scan execution".

Basic Universal
Note6.38 Note6.38

In the case of Basic model QCPU, a program cannot be started from the program list monitor screen.

Redundant Universal
Note6.39 Note6.39

The Redundant CPU and Universal model QCPU does not support low speed execution type programs.

☒POINT

Depending on the instruction, the output may not turn OFF if "After stop, output stop" is executed.
For details, refer to the section of the POFF instruction in the following manual.
☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

(5) **Precaution**
The scan time of a constant scan execution type program being executed is not displayed on screen, but a dash (-) is displayed in the Scan Time column.

6.13.2 Interrupt program monitor list

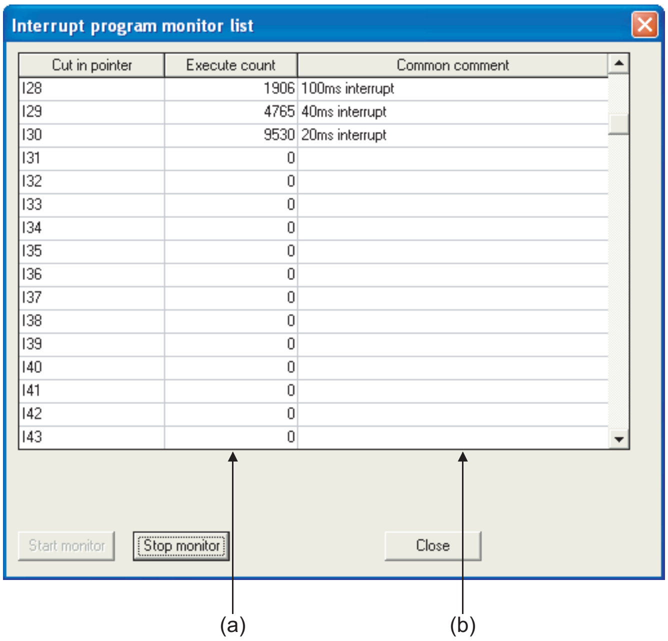
(1) Definition of Interrupt Program Monitor List

This function displays execution count of the interrupt program.
This is used to confirm the execution status of the interrupt program.

(2) Using the Interrupt Program Monitor List

Choose [Online] → [Monitor] → [Interrupt program monitor list]. The "Interrupt Program Monitor List" dialog box appears on screen.

Figure 6.64 shows an execution example of the interrupt program monitor list:



Interrupt program monitor list

Cut in pointer	Execute count	Common comment
I28	1906	100ms interrupt
I29	4765	40ms interrupt
I30	9530	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 6.64 Interrupt program monitor list

(a) Execute count

The number of times the interrupt program was executed is displayed. This function starts counting the number when CPU module is in RUN status (When the number reaches 65536 times, it is reset to 0).

(b) Common Comment

This indicates device comments created on interrupt points.

Basic
Note6.40

Universal
UD
Note6.41

6.13.3 Scan time measurement

(1) Definition of Scan Time Measurement

This function displays the set program interval processing time.

The time for the subroutines and interrupt program can be measured as well.

(2) Scan time measurement range designation

To specify a scan time measurement range, follow either of the following two steps:

- Make the setting on the "Ladder monitor" window.
- Make the setting on the Scan time measurement screen.

(3) When subroutine program call instruction exists

The time includes the time required for processing subroutines, when the subroutine CALL instruction is within the range of scan time measurement.

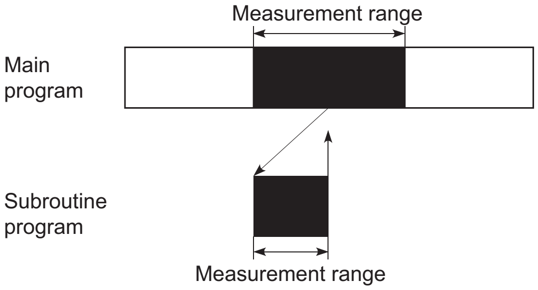


Figure 6.65 When subroutine program is included in measurement range

(4) When interrupt programs/fixed scan execution type programs are executed within the scan time measurement range

If interrupts for executing interrupt programs and/or fixed scan execution type programs occur within the scan time measurement range, the execution time for those programs will be added.

Basic
Note6.40

The Basic model QCPU does not support the scan time measurement.

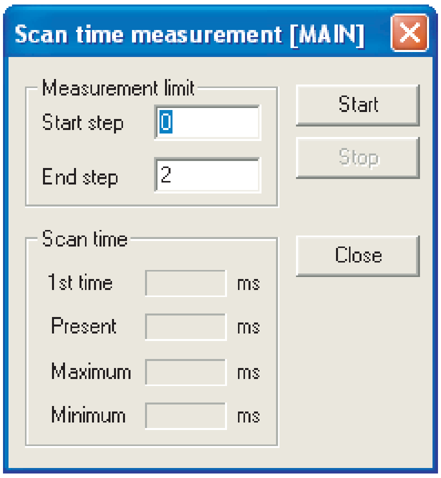
Universal
UD
Note6.41

When using the Scan time measurement in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

(5) Measuring Scan Time

To measure scan time, follow the following steps.

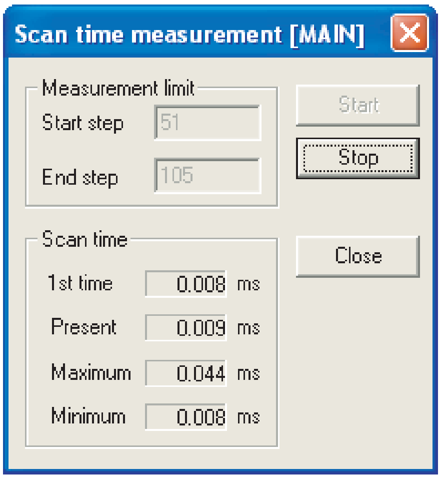
- Display the beginning of the ladder of which scan time will be measured, on GX Developer, select the monitor mode.
- Choose [Online] → [Monitor] → [Scan time measurement] to open "Scan time Measurement screen" dialog box.


Scan time measurement [MAIN]

Measurement limit
Start step: 0
End step: 2

Scan time
1st time: ms
Present: ms
Maximum: ms
Minimum: ms

- Specify the start step and end step, click the Start button.
  [Example] When "52" for the start step and "105" for the end step are specified


Scan time measurement [MAIN]

Measurement limit
Start step: 51
End step: 105

Scan time
1st time: 0.008 ms
Present: 0.009 ms
Maximum: 0.044 ms
Minimum: 0.008 ms

Figure 6.66 Scan time measurement screen

Remark

When the Scan time measurement screen is displayed after specifying the target rage in monitor mode, the start step and end step of the specified range are set in the Start step and End step fields.
To specify the range in monitor mode, use the "Shift" key and click of a mouse.

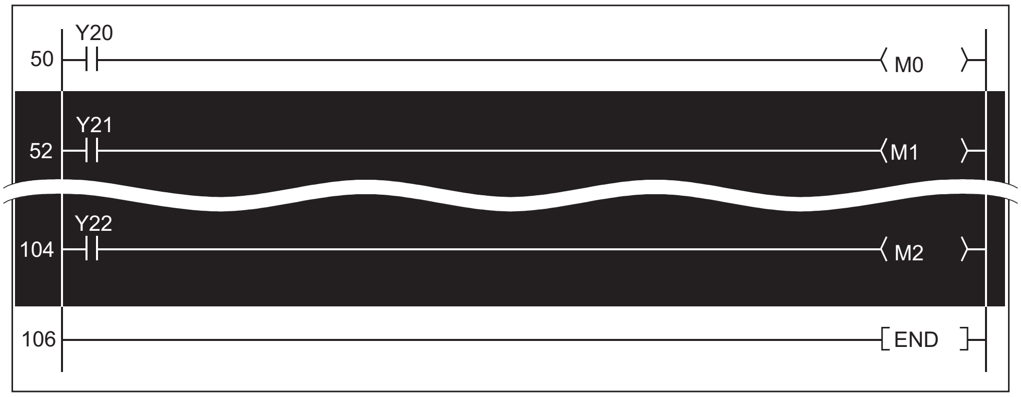


Figure 6.67 Measurement range designation

(6) Precautions

(a) Measurement range setting

Set the measurement range ("Measurement limit") so that the following condition is satisfied.(Start step) < (End step)

(b) Scan time measurement across program files

Scan time cannot be measured across program files.

(c) Measurement time minimum unit

1) **High Performance model QCPU, Process CPU, and Redundant CPU**
The measurement time minimum unit is 0.100ms.
If the measurement time is less than 0.100ms, 0.000ms is displayed.

2) **Universal model QCPU**
The measurement time minimum unit is 0.01ms.
If the measurement time is less than 0.01ms, 0.000ms is displayed.

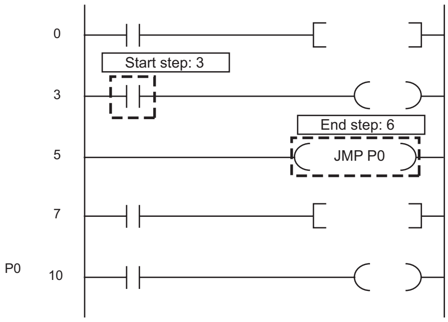
(d) Step specification between the FOR and NEXT instructions

If the steps between the FOR and NEXT instructions are specified, measurement time indicates the execution time of one scan between the specified steps.

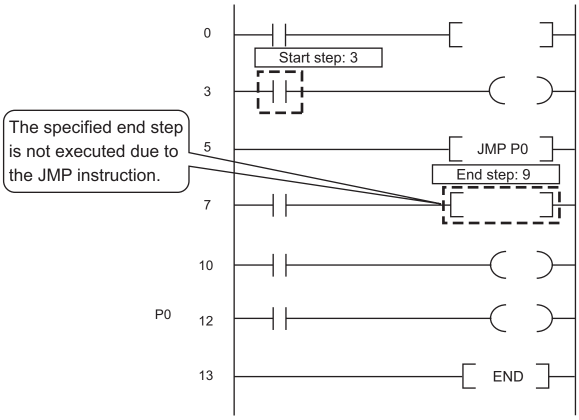
(e) **When scan time cannot be measured**

Scan time cannot be updated on the Scan time measurement screen in the following cases.

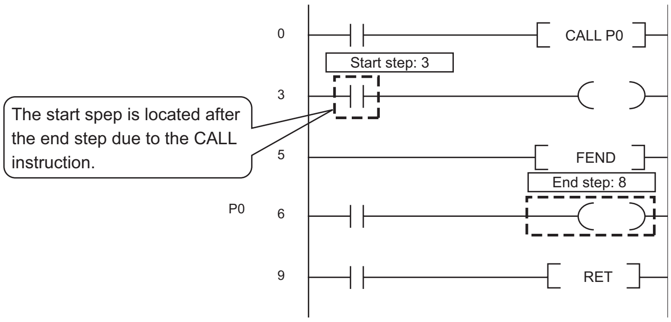
- When branch instruction is specified for end step
  [Example]
  The JMP instruction is specified for the end step.



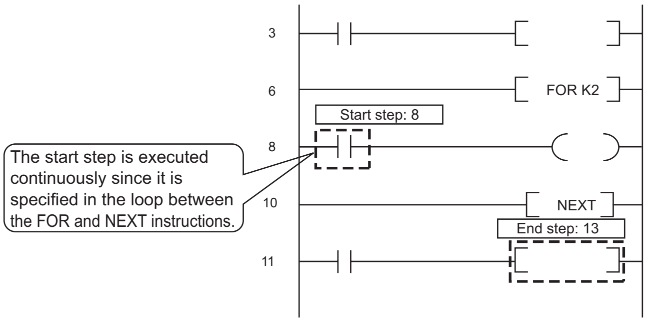
- When only start step is executed
  [Example]
  The specified end step is not executed due to the JMP instruction.


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 end step is executed before start step
  [Example]
  The start step is specified as next step of the CALL instruction, and the end step is specified in a subroutine program called by the CALL instruction.


0
CALL P0
Start step: 3
3
The start spep is located after the end step due to the CALL instruction.
5
FEND
End step: 8
P0
6
9
RET

- When start step is executed continuously
  [Example]
  Only the start step is specified in the loop between the FOR and NEXT instructions.


3
6
FOR K2
Start step: 8
8
The start step is executed continuously since it is specified in the loop between the FOR and NEXT instructions.
10
NEXT
End step: 13
11

6.14 Sampling Trace Function

Basic Note6.42

(1) Definition of Sampling Trace Function?

This function samples the device continuously on the High Performance model QCPU at specified timings.

The sampling trace samples the contents of the specified device at a set interval (sampling cycle), and stores the trace results at the sampling trace file.

(2) Applications of sampling trace

The changed contents of the device that program uses during debugging can be checked at the specified timing.

The sampling trace function reads device contents if trigger conditions are satisfied.

(3) Sampling trace operation

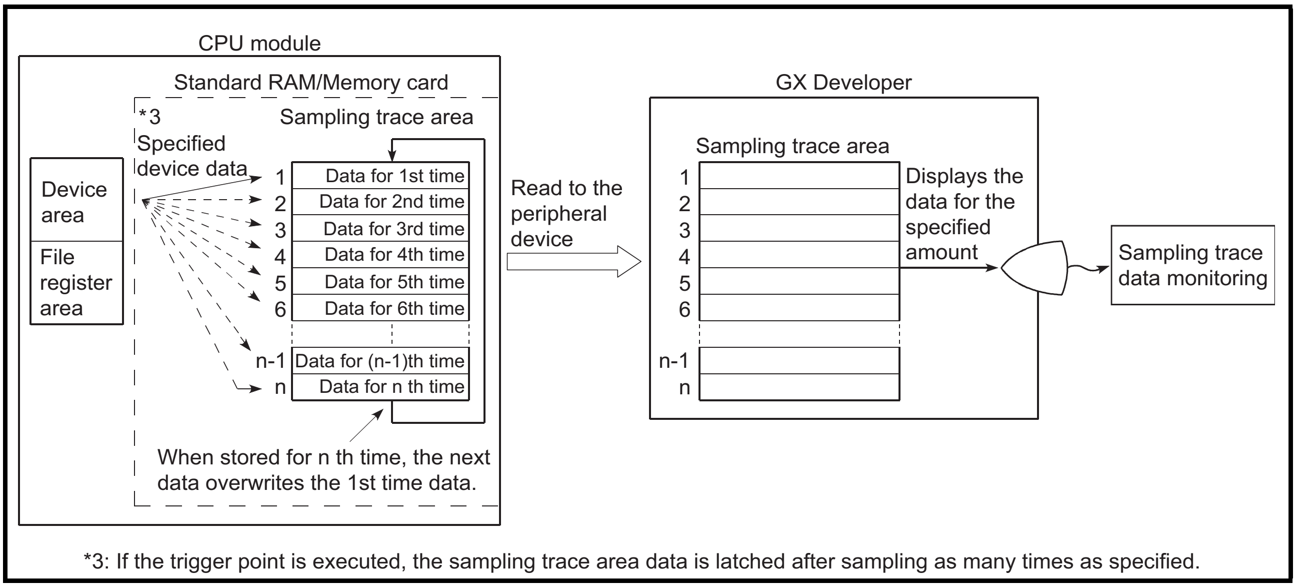
The sampling trace file stores the trace setting and trace result necessary to perform the sampling trace. When trace is started by using GX Developer, the trace is performed as many times as specified.

The area for sampling trace is up to 60k bytes. (For the Universal model QCPU, there is no restriction on capacity.)

The number of traces is a value divided by the number of bytes of the device where the number of bytes of sampling trace area is set (N1+N2+N3 number of word device points × 2+(number of bit device points/16 × 2) [*1] [*2]

[*1] : Round up result of "number of bit device points/16" in the expression to the right of the decimal point.

[*2] : The following values are added to the N1 to N3 corresponding to the items set in trace additional information of trace condition settings screen.
N1 : Add "4" at setting time
N2 : Add "10" at setting step No.
N3 : Add "8" at setting program name



*3: If the trigger point is executed, the sampling trace area data is latched after sampling as many times as specified.

Figure 6.68 Sampling trace operation

Basic Note6.42

The Basic model QCPU does not support the sampling trace function.

☒**POINT**

1. The sampling trace file can be stored into only the standard RAM or SRAM card. (☞ Section 5.2.1(2))
2. When the sampling trace file is stored into the standard RAM, check the version of CPU module and GX Developer.(☞ Appendix 4)

(a) When sampling trace is executed

The execution status of the sampling trace function is stored in the special relay (SM800, SM802, SM804 and SM805).

If an error occurs while the sampling trace function is used, SM826 turns on. By using special relays in a sequence program, the execution status of the sampling trace function can be checked.

1) Preparations for sampling trace

After writing [Trace data conditions] and [Trace condition settings] that are set using GX Developer to the CPU module is completed, SM800 (sampling trace ready) is turned ON. SM800 indicates whether the sampling trace can be executed or not.

2) Start of sampling trace

When a sampling trace start request is accepted, the sampling trace starts and SM802 (sampling trace executing) turns on. SM802 indicates whether the sampling trace is executed or not.

- A trace from GX Developer starts.
- SM801 is turned on.
- Sampling trace auto start (☞ (4)(e) in this section)

3) Checking the trigger conditions

When a next trigger condition is satisfied, SM804 (after sampling trace trigger) turns on. SM804 indicates whether the trigger conditions are satisfied or not.

- A trigger from GX Developer executed.
- The TRACE instruction is executed.
- SM803 is turned on.
- Detailed setting. (Device No. and step No.)

(b) When sampling trace is completed

After the sampling trace is completed, SM805 (sampling trace complete) is turned on.

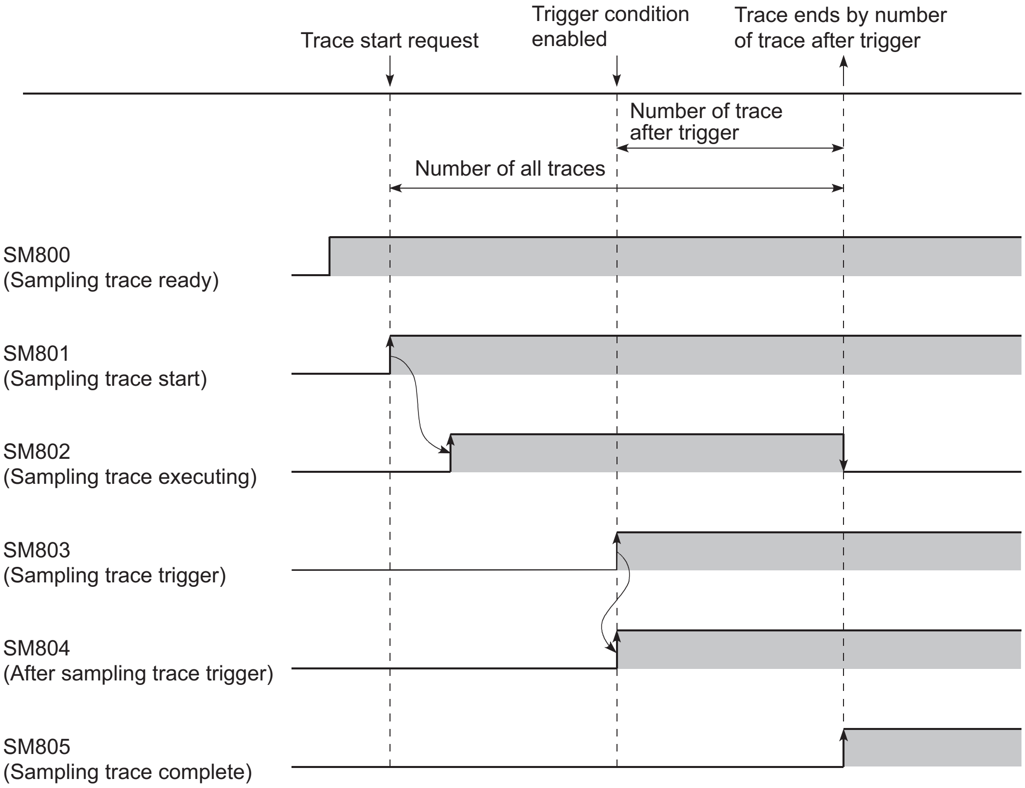

Trace start request
Trigger condition enabled
Trace ends by number of trace after trigger
Number of trace after trigger
Number of all traces
SM800 (Sampling trace ready)
SM801 (Sampling trace start)
SM802 (Sampling trace executing)
SM803 (Sampling trace trigger)
SM804 (After sampling trace trigger)
SM805 (Sampling trace complete)

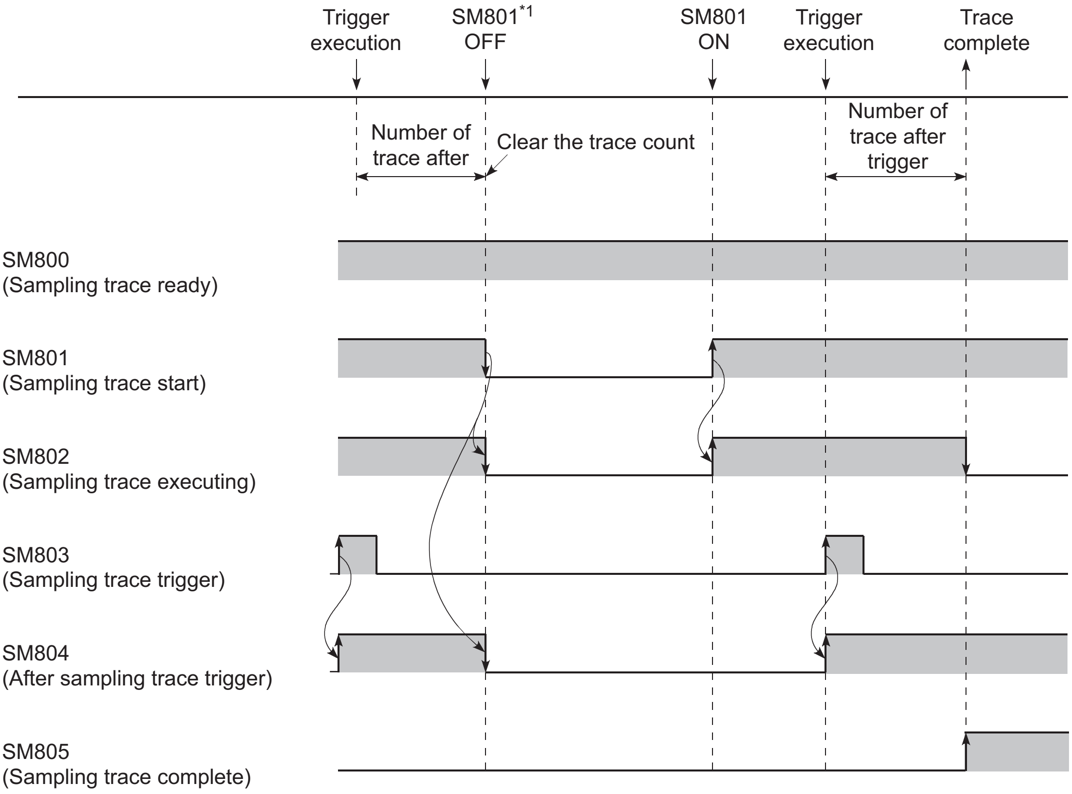
Figure 6.69 Sampling trace timing chart

(c) Sampling trace interrupt

When SM801 (sampling trace start) is turned off during sampling trace, the sampling trace is interrupted.

When sampling trace is interrupted, the number of traces is cleared.

When turning on SM801 again, trace is restarted.


Trigger execution
SM801*1 OFF
SM801 ON
Trigger execution
Trace complete
Number of trace after
Clear the trace count
Number of trace after trigger
SM800 (Sampling trace ready)
SM801 (Sampling trace start)
SM802 (Sampling trace executing)
SM803 (Sampling trace trigger)
SM804 (After sampling trace trigger)
SM805 (Sampling trace complete)

*1: When trace is interrupted from GX Developer, the SM800 is also turned off.

Figure 6.70 Sampling trace timing chart (When sampling trace is interrupted)

(4) Operation procedure

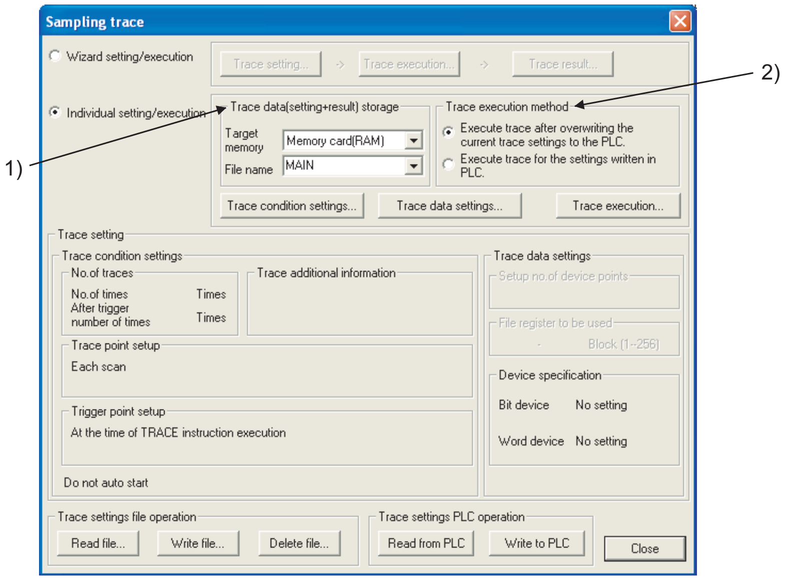
The following methods are available as the sampling trace operation procedure.

- Use the wizard (☞ GX Developer Operating Manual)
- Make detailed settings individually. (☞ as follows)

Perform each operation from the [Online] → [Trace] → [Sampling trace...] from the menu bar.

(a) Trace data (setting+result) storage, Trace execution method

Set the trace data storage destination 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

Do not auto start

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 6.71 Sampling trace screen

1) **Trace data (setting+result) storage**

Set the target memory to store the trace data and the file name to write the trace conditions.

Select either of "Standard RAM" or "Memory card (RAM)" as target memory.

The trace result is stored into the target memory set in this setting with specified file name.

2) **Trace execution method**

Select the trace execution method from below.

- Execute trace after overwriting the current trace setting to the CPU module.

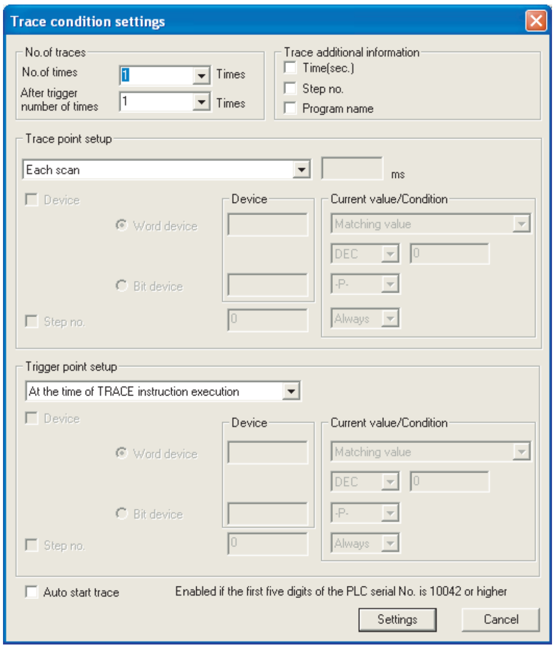
  The trace setting is overwritten to the existing sampling trace file.
- Execute trace specified in [Trace setting] written to the CPU module.

  The trace is executed with the sampling trace file setting specified in [Trace data (setting + result) storage].

(b) Setting the Trace Condition

Click the Trace condition setting button on the Sampling trace screen and set the trace conditions.

On the Trace condition setting screen, number of traces, trace point, trigger point, trace additional information, and auto start setting can be set.



Trace condition settings

No.of traces
No.of times: 1 Times
After trigger number of times: 1 Times

Trace additional information
□ Time(sec.)
□ Step no.
□ Program name

Trace point setup
Each scan ▾ ms
□ Device
● Word device
○ Bit device
□ Step no. 0
Device
Current value/Condition: Matching value ▾, DEC ▾ 0, -P- ▾, Always ▾

Trigger point setup
At the time of TRACE instruction execution ▾
□ Device
● Word device
○ Bit device
□ Step no. 0
Device
Current value/Condition: Matching value ▾, DEC ▾ 0, -P- ▾, Always ▾

□ Auto start trace
Enabled if the first five digits of the PLC serial No. is 10042 or higher

Settings Cancel

Figure 6.72 Trace condition setting

1) No. of traces

The "No. of times" sets the number of times to execute the sampling trace from trace execution to trace complete.

The "After trigger number of times" sets the number of times to executes the sampling trace from trigger execution to trace complete.

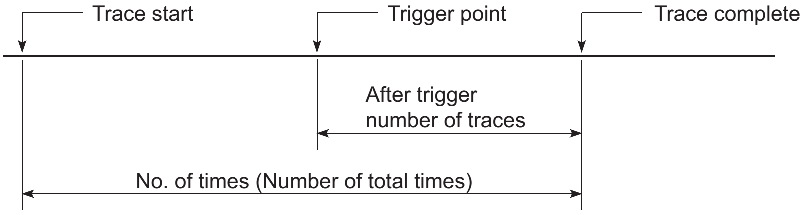

Trace start
Trigger point
Trace complete
After trigger number of traces
No. of times (Number of total times)

Figure 6.73 Relation between total number of times and the number of times after trigger

The setting range for each number of times is shown below:

(After trigger number of times) ≦ (No. of times) ≦ (8192)

2) **Trace Point Setup**

This sets the timing to collect trace data. Select one from the following:

- Each Scan
  Collects trace data for every scan (END processing).
- Interval
  Collects trace data at specified times.
- Per multiple CPU high speed transmission cycle (Universal model QCPU only)
  Data is collected in a cycle of 0.88ms × specified interval of time.
  When the trace point setting is set to [per multiple CPU high speed transmission cycle], the trace data can be collected only when the following conditions are all met, since the trace data is collected at the timing of executing the interrupt program of I45.
  1.Multiple CPU high speed main base unit (Q3□DB) is used.
  2.The multiple CPU system where two or more CPU modules are used and multiple CPU high speed transmission function is set.
  3.Interrupt pointer of I45 exists in a program.
  4.EI status with the interrupt mask of I45 is cancelled.
- Detailed
  Sets the device and step no. The setting method and trace data sampling timing is the same as mentioned in Section 6.11.1, (when setting the monitor condition setting.)
  Also, other than above timings, data can also be collected at the time of changing word devices in the Universal model QCPU.
  The device settable in the detailed setting is as follows:

  •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[*1], SD, W[*2], SW, R, Z, ZR, FD, U□\G, J□\W, J□\SW,
  U3E □\G (Universal model QCPU only)

[* 1] The extended data register (D) is also included.
[* 2] The extended link register (W) is also included.

The following attributes can be set for the above devices:

•Bit device number of digits specification
•Word device bit number specification
•Word device indirect specification
•index modification

3) **Trace additional information**

The information to be added for every trace is set. Multiple items can be selected from the following (of none of the items have to be selected):

- Time
  Stores the time when the trace was executed.
- Step No.
  Stores the step number when the trace was executed.
- Program Name
  Stores the program name that executed the trace.

4) **Trigger Point setup**

This sets the point to execute the trigger. Select one from the following:

- At the time of TRACE order:
  The time of execution of TRACE instruction is set as the trigger.
- At the time of trigger operation:
  The trigger operation from GX Developer device is set as the trigger.
- Detailed setting
  The device and step number is set. The setting method and trigger execution timing is the same as mentioned in Section 6.11.1, (the monitor condition setting.)
  Also, other than above timings, data can also be collected at the time of writing word devices in the Universal model QCPU.
  The device settable in the detailed setting is as follows:

•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[*1], SD, W[*2], SW, R, ZR,

[*1] The extended data register (D) is also included.
[*2] The extended link register (W) is also included.

The following attributes can be set for the above devices:

•Word device bit number specification

Indirect specified devices cannot be set.

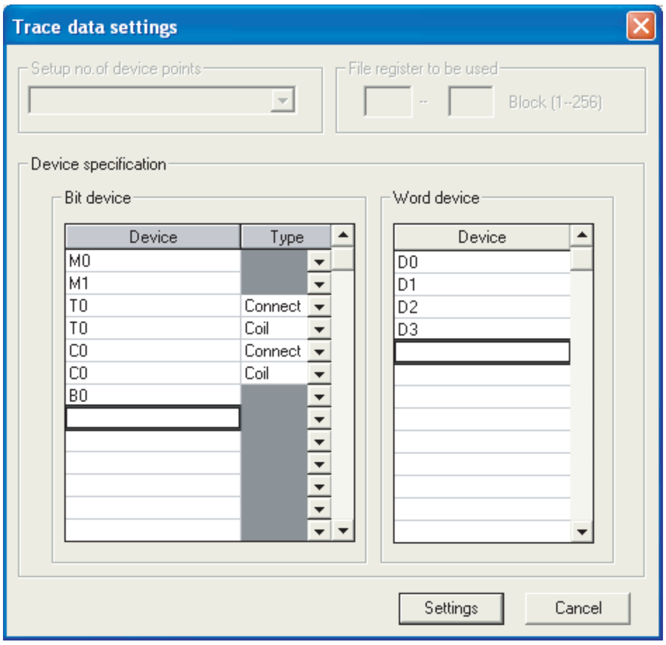
5) **Sampling trace auto start setting ("Auto start trace")**

Sets whether to start the sampling trace automatically. (☞ (4)(e) in this section)

(c) Trace data Setting

Select "Individual setting/execution" on the Sampling trace screen.

Click the Trace data setting button and set the devices to which sampling trace will be executed.



Device	Type
M0	
M1	
T0	Connect
T0	Coil
C0	Connect
C0	Coil
B0	

Device
D0
D1
D2
D3

Figure 6.74 Trace data setting screen

1) Bit Device

Maximum of 50 bit devices can be set as follows.

- 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

2) Word Device

Maximum of 50 word devices can be set as follows.

- T(current value), ST(current value), C(current value), D*1, SD, W*2, SW, R, Z, ZR, U□\G, J□\W, J□\SW, U3E □\G (Universal model QCPU only)

* 1: The extended data register (D) is also included.

* 2: The extended link register (W) is also included.

The following attributes can be set for the above devices:

- Bit device number of digits specification
- Word device bit number specification
- Index modification

Indirect specified devices cannot be set.

(d) Writing trace data settings and trace condition settings

The created trace data settings and trace condition settings are written to the target memory set in [Trace data (setting + result) storage] as sampling trace file.

Use the [Write to PLC] button on the Sampling trace screen to write the sampling trace file to the memory card (SRAM card).

☒POINT

When the sampling trace file is stored into the memory card (SRAM card), multiple 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).

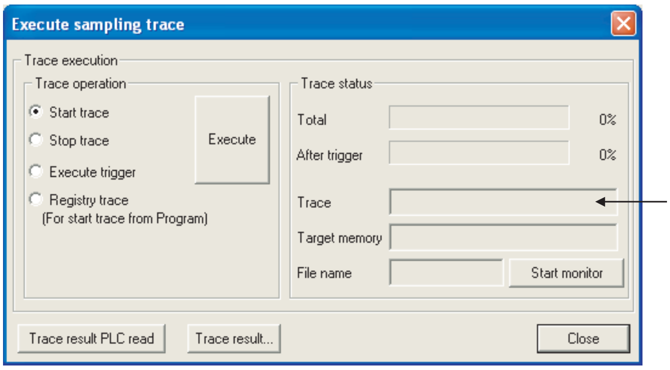
1 Overview
2 Performance Specification
3 Sequence Program Configuration and Execution Conditions
4 I/O Nunber Assignment
5 Memories and Files Handled by CPU Module
6 Functions
7 Communication using built-in Ethernet ports of CPU module
8 Communication with Intelligent Function Module

(e) Sampling trace execution

Sampling trace can be executed in two ways: using GX Developer and using the auto start function.

1) Executing sampling trace using GX Developer

Click the Execute button on the Sampling trace screen to display the execute sampling trace screen.
Sampling trace will be executed.



Displayed only when status display is selected

Figure 6.75 Execution and status display screen

Trace operation can be set on the Execution and status display screen.

- 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.
  (When restarting 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 set number.
- Registry trace (For start trace from Program)
  Registers trace data when a program is executed.

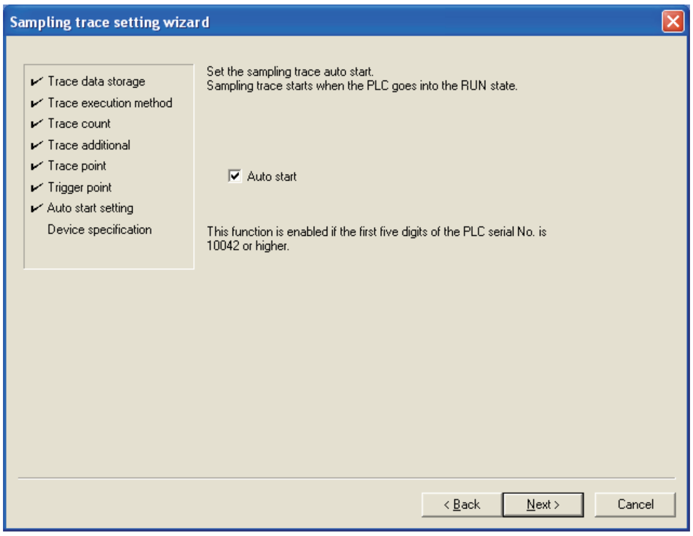
Universal UD Note6.43

Basic Note6.44 High Performance Note6.44 Process Note6.44

Redundant Note6.44

2) Executing sampling trace using the auto start function[Note6.43,Note6.44]

Check the box for "Auto start" on the Sampling trace setting wizard screen.



Sampling trace setting wizard

- ✓ Trace data storage
- ✓ Trace execution method
- ✓ Trace count
- ✓ Trace additional
- ✓ Trace point
- ✓ Trigger point
- ✓ Auto start setting
- Device specification

Set the sampling trace auto start.
Sampling trace starts when the PLC goes into the RUN state.

☑ Auto start

This function is enabled if the first five digits of the PLC serial No. is 10042 or higher.

Figure 6.76 Screen for auto start setting

When the check box for "Auto start" is checked, sampling trace will be started automatically at the following timing.

- When the programmable controller is powered OFF → ON after sampling trace setting is completed.
- When the CPU module is switched from STOP to RUN.

POINT

When re-registration of sampling trace setting is attempted by powering OFF → ON the programmable controller, resetting the CPU module or switching the CPU module from STOP to RUN, the setting is not re-registered regardless of the status of SM829 (Forced registration specification of trace setting) if a trigger condition has already been satisfied. In this case, the module is in the status where the sampling trace setting registration has been canceled, turning SM800 (Trace preparation) OFF.

Universal UD Note6.43

When using the sampling trace auto start function in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

Basic Note6.44 High Performance Note6.44 Process Note6.44

Redundant Note6.44

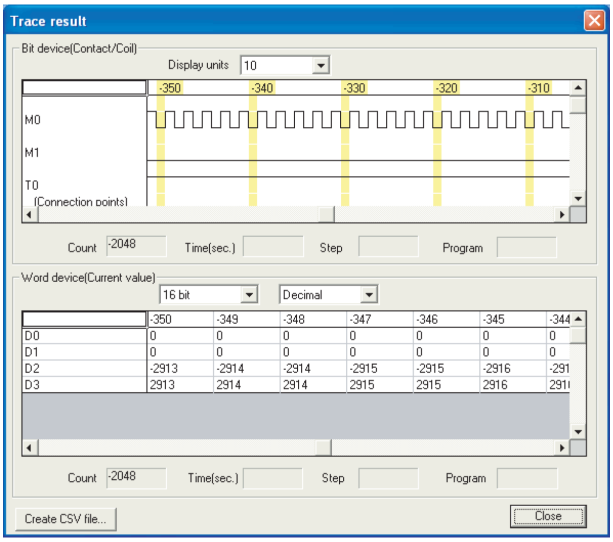
The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not support the sampling trace auto start function.

(f) Trace result display

Read the trace results form the CPU module and display the data.

- Click the Trace result PLC read button on the Sampling trace execution screen to read the trace result from the CPU module.
- The trace result read with the Trace result button will be displayed.

The trace result displays the ON/OFF status of the bit device for the sampling cycle, and the current value of the word device.



Trace result

Bit device(Contact/Coil)

Display units 10

Count -2048 Time(sec.) Step Program

Word device(Current value)

16 bit Decimal

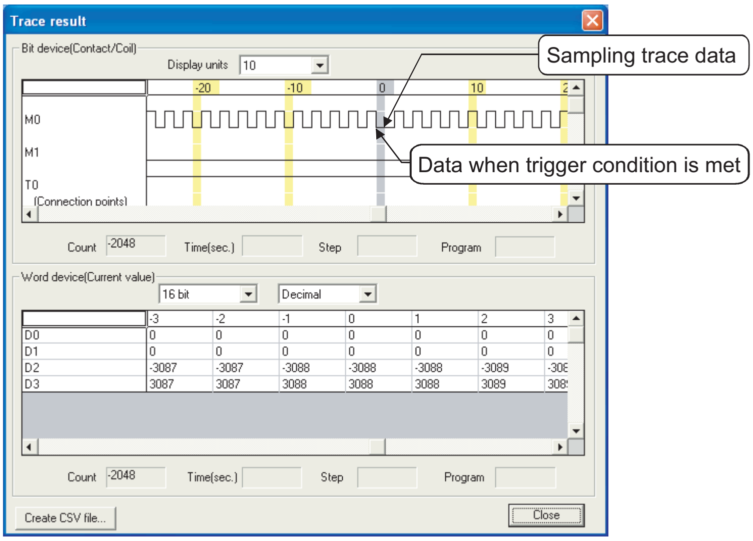
	-350	-349	-348	-347	-346	-345	-344
D0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0
D2	-2913	-2914	-2914	-2915	-2915	-2916	-291
D3	2913	2914	2914	2915	2915	2916	291

Count -2048 Time(sec.) Step Program

Figure 6.77 Trace result display

☒POINT

Device details are read under trigger conditions specified in the trigger point setting. Sampling is performed for each scan. Before the sampling is finished by a trigger operation of a peripheral device, data is sampled twice because the sampling timing is the same as that of trigger conditions.



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 6.78 Trace result display

Universal UD Note6.45

(5) How to clear [Trace execution] status

The [Trace execution] status can be cleared by latch clear using the RESET/L.CLR switch[Note6.45] or the remote latch clear operation. When performing the sampling trace after latch clear, select [Start trace] or [Registry trace] before executing the sampling trace again.

(6) Precautions

(a) Areas where sampling trace can be executed

The sampling trace can be executed from other station on the network or serial communication module. However, the trace cannot be executed from multiple areas at once. The trace can only be executed from one area with CPU module.

(b) How to hold and clear [Trace setting]

The trace setting (sampling trace file) registered in the CPU module is latched. Even if turning power supply ON (OFF→ON)/resetting the CPU module, the sampling trace can be executed again in the trace setting at the time of registration. However, the previous trace result cannot be read.
Also, when the trigger condition of the sampling trace is established, the latched [Trace setting] will be cleared since it is not recognized as the trigger condition. (SM800 (trace ready) is turned OFF.)
Register [Trace setting] again by GX Developer.

1) When "standard RAM" is selected in target memory

- The power supply was turned on (off to on), CPU module was reset, or CPU module status was changed from STOP to RUN after setting to change the size of local device in the standard RAM [*1] and writing PLC parameter to CPU module.
- On the status of sampling trace file corruption, the power supply was turned on (off to on), or CPU module was reset.
  *1 : Creating a new local device is included.

☒POINT

To keep the trace result in your personal computer, perform the following operations.

- Click the Trace result PLC read button on the sampling trace execution screen to read the trace result. (☞ Sampling trace execution in this section)
- Click the Trace result button to display the trace result.
- Click the Create CSV file button to store the trace result in CSV format.

Universal UD Note6.45

Latch clear cannot be executed by switch in the Universal model QCPU.

2) When "Memory card (RAM)" is selected in target memory

- When SRAM card registered the sampling trace file has not been mounted, the power supply is turned on (off to on), or CPU module was reset.
- On the status sampling trace file corruption, the power supply was turned on (off to on), or CPU module is reset.

(c) Connection with CPU module when sampling trace is executed

Execute sampling trace after connecting GX Developer to the CPU module.

(d) Reading trace result at 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.

(e) Trigger condition at sampling trace registration (Other than Universal model QCPU)

When registering the sampling trace, do not establish the trigger condition set by the trigger point.
If established, the setting of the sampling trace cannot be registered.

(f) Registration of the sampling trace when the trigger condition is established(Universal model QCPU only)

When registering the sampling trace in the Universal model QCPU, the setting of the sampling trace can be registered by the following procedures even if the trigger condition is established.

1) Turn ON SM829 (Forced registration specification of trace setting).

2) Select [Individual setting/execution] in Sampling trace screen.

3) Select [Registry trace (For start trace from Program)] from [Trace execution...] and click the Execute **button.**

☒POINT

1. When the sampling trace is registered by the procedures (f) 1) to 3) mentioned above, start the sampling trace in a status where the trigger condition is not established.
   If the trigger condition has been established when the sampling trace is started, trigger may not be executed.
2. When selecting [Wizard setting/execution] in Sampling trace screen mentioned in the above (f) 2), and starting the sampling trace, the trigger may not be executed normally, although the setting of the sampling trace can be registered.

(g) When file register is designated in designation device with detail setting of trace conditions

When file register is designated in designation device with detail setting of trace point and trigger point setting, do not change block No. of file register file and file register after registry trace.
Trace data may not be normally sampled.

(h) Trace point setting Note6.46

High Performance Process Redundant
Note6.46 Note6.46 Note6.46

When setting the trace point setting as [per time] or [per multiple CPU high speed transmission cycle], 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 occur.

(i) Executing sampling trace during execution of sampling trace

The first sampling trace continues normally. However, the second sampling trace cannot be executed.

(j) Executing online change (Write during RUN)

Operation will be as follows when sampling trace and online change are executed simultaneously.

1) **Executing sampling trace during execution of online change**
   When step number is used for specifying the trace point and trigger point, online change can be executed (completes normally) and sampling trace cannot be executed.
   When other than step number is used for specifying the trace point and trigger point, both online change and sampling trace can be executed.
2) **Executing online change during execution of sampling trace**
   Only when step number is used for specifying the trace point and trigger point, sampling trace is stopped and online change can be executed (completes normally).
   When other than step number is used for specifying the trace point and trigger point, both online change and sampling trace can be executed.

High Performance Process Redundant
Note6.46 Note6.46 Note6.46

The High Performance model QCPU, Process CPU, and Redundant CPU cannot set [per multiple CPU high speed transmission cycle].

6.15 Multiple-user debugging function

(1) Debug Execution by Multiple Users

This function debugs from multiple GX Developers connected to such as CPU module or serial communication module.

When a file has been divided according to the procedure or the function, the divided files can be debugged from multiple GX Developers.Note6.47

Basic
Note6.47

(2) Function Description

The debug function combination for multiple users are as follows:

Table6.32 Combined use of Debug function and other functions

Functions being executed	Functions executed later			
	Monitor	Online change	Execution time measurement	Sampling trace
Monitor	○*1	×*2	○	○
Online change	×*2	×*4	×	×
Scan measurement	○	○	×*3	○
Sampling trace*5	○	×	○	×*3

○ : Can be executed simultaneously.

× : Can only be performed from one GX Developer.

Basic Universal
Note6.48 Note6.48

* 1 : Since monitor condition setting (☞ Section 6.11.1)is only available from a single GX Developer, the setting can not be made from other GX Developers.Note6.48

* 2 : Monitoring with monitor conditions can not be executed simultaneously with online change.

* 3 : The items of × cannot be executed simultaneously by multiple users. For executing from the other GX Developers, stop or interrupt the execution time measurement and sampling trace being executed.

* 4: For method of writing to a single file in RUN by multiple users, refer to Section 6.15.2.

* 5: The Basic model QCPU does not support the sampling trace function.:

Basic
Note6.47

The Basic model QCPU does not support the Multiple-user debugging function since files cannot be divided based on procedures or functions.

Basic Universal
Note6.48 Note6.48

The Basic model QCPU does not support the monitor with monitor conditions.
When setting the monitor conditions in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

6.15.1 Simultaneous monitoring execution by multiple users

(1) Simultaneous monitoring execution by multiple users

The CPU module, serial communication module or similar module can be simultaneously monitored by multiple GX Developer, when they are connected.

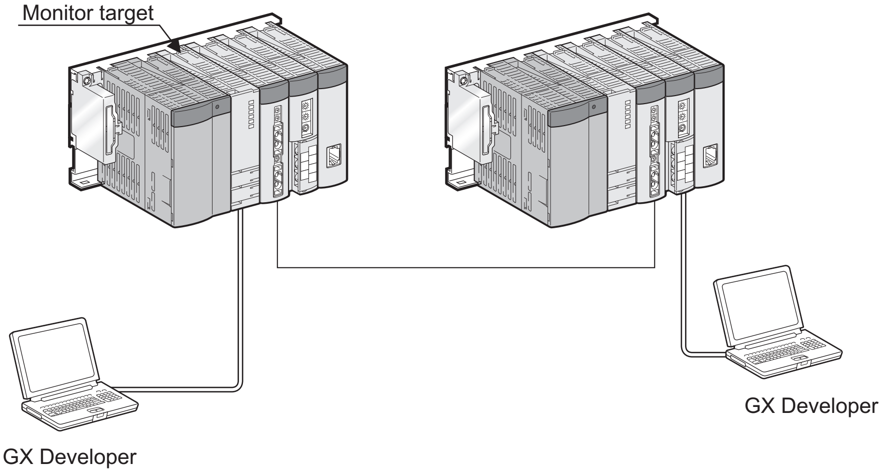


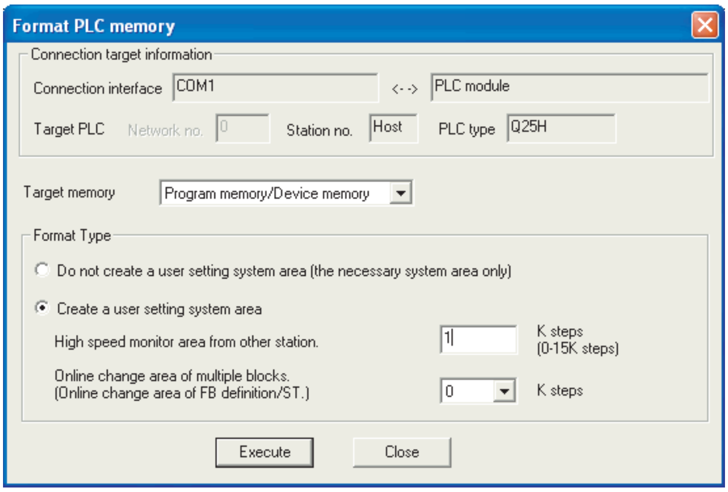
Figure 6.79 Simultaneous monitor

Multiple users can monitor at the same time. By setting a station monitor file, high speed monitoring can be performed. (It is not necessary to set host station monitor file.)

(2) Operation Procedure

For multi-user monitoring operation, create a user-defined system file in the following steps.

- Choose [Online] → [Format PLC memory] on GX Developer to display the PLC memory format window.
- Select "Program memory/device memory" from the "Target Memory list box".
- At the "Format Type" section, select "Create a user setting system area..." so that its radio button is checked.
- Specify the desired k steps in the "System Area" text box.



Format PLC memory

Connection target information

Connection interface: COM1 <--> PLC module

Target PLC: Network no. 0 Station no. Host PLC type Q25H

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 6.80 System area setting (when number of steps is set to 1k steps)

The maximum number of steps that can be set as system areas is as indicated in Table6.33.

Table6.33 Maximum number of steps that can be set as system areas

CPU module	Maximum number of steps that can be set	System area for monitoring from other station
Basic model QCPU	Max. 3k steps	Max. 3 areas
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	Max. 15k steps	Max. 15 areas

Only 1k steps are applicable to one monitor file in other station.

(3) Precautions

Basic Universal
Note6.49 Note6.49

(a) **Monitor condition setting**Note6.49

Monitor condition setting can be made by only one GX Developer.

(b) **Necessity of system area setting**

If the user setting system area is not created, simultaneous monitor from other stations can be executed, but the monitor speed decreases.

The system area is in the same area as the program memory, so the area of the stored program decreases when the system area is set.

(c) **The number of nodes that can be set to monitor at higher speed**

When multiple nodes are monitoring one CPU module simultaneously, a certain number (number of user setting system areas (k steps) + 1) of them can be set to monitor at higher speed.

For example, when 15k steps of user setting system areas are created, Up to 16 nodes can be set to monitor one CPU module at higher speed.

Basic Universal
Note6.49 Note6.49

The Basic model QCPU does not support the monitor with monitor conditions.
When setting the monitor conditions in the Universal model QCPU, check the versions of CPU module and GX Developer. (Appendix 4.5)

6.15.2 Simultaneous online change by multiple users

Basic
Note6.50

(1) Simultaneous online change by multiple users

This function enables multiple users to write to one or more files in RUN. When online change is performed to a single file by multiple users, select "Relative step No. by pointer". When online change is performed to other files by multiple users, checking "Relative step No. by pointer" is unnecessary.

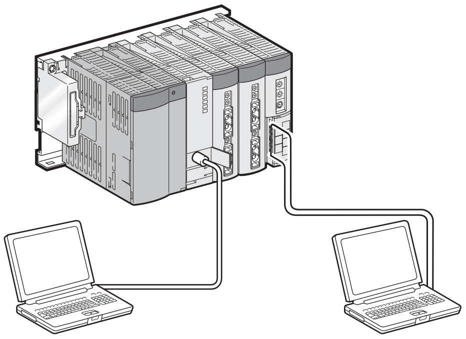


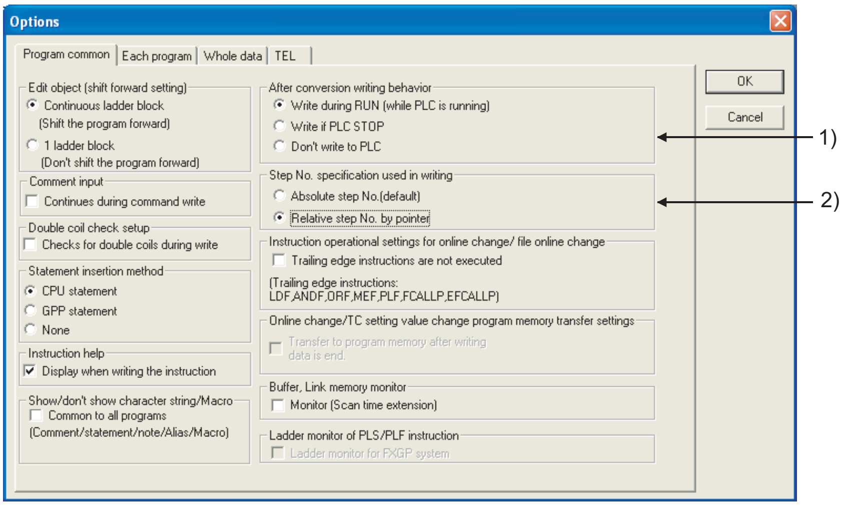
Figure 6.81 Simultaneous online change by multiple users

(2) Operating procedure for writing to a single file in RUN.

When multiple users write to one file simultaneously during Run, it is necessary to set the RUN write pointer and select "Relative step No. by pointer" in advance.

(a) Displaying<<Program common>>tab

- Choose [Tools] → [Options] on GX Developer and select the <Program common> tab.



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,ANDF,ORF,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

Figure 6.82 Options

Basic
Note6.50

When online change is executed by multiple users simultaneously for the Basic model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.1)

(b) **Set Online change and set the Online change method.**

1) Set Online change (while PLC is runnning) in "After conversion writing behavior".

2) Select Absolute step No. (default) or "Relative step No. by pointer" in "Step No. specification used in writing".

(c) **Online change execution**

- The specified circuit of the pointer is displayed to write the circuit after conversion during RUN.

(3) Precautions

Precautions for writing data during RUN are the same as those of writing data during RUN in Section 6.12.3.

6.16 Watchdog Timer (WDT)

(1) Definition of Watchdog Timer (WDT)

The watchdog timer is an internal sequence timer to detect CPU module hardware and sequence program error.

(2) Watchdog Timer Setting and Reset

(a) Watchdog timer setting

The watchdog timer setting can be changed at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box.
The default value of the watchdog timer is 200 ms.
The setting range is 10 to 2000 ms (in 10ms units).

(b) Watchdog timer resetting

CPU module resets the watchdog timer during the END processing.

- When the END/FEND instruction is executed within the set value of the watchdog timer in the sequence program and the CPU module is operating correctly, the watchdog timer does not time out.
- When the scan time of a sequence program is extended due to the CPU module hardware error or execution of interrupt program/fixed scan execution type program[Note6.51], and END/FEND instruction cannot be executed within the set watchdog timer value, the watchdog timer times out.

Basic Note6.51

(3) When watchdog timer expires

When the watchdog timer expires, a watchdog timer error occurs.
The CPU module responds to the watchdog timer error as follows:

1) The CPU module turns off all outputs.
2) The front-mounted RUN LED turned off, and the ERR. LED starts flicking.
3) SM1 turns ON and the error code 5000, 5001 ("WDT ERROR") is stored into SD0.

Basic Note6.51

Since the Basic model QCPU does not support fixed scan execution type programs, scan time is not increased due to fixed scan execution time programs.

(4) Precautions

(a) Watchdog timer error

An error of 0 to 10 ms occurs in the measurement time of the watchdog timer. Set the watchdog timer for a desired value by taking such an error into account.

(b) Watchdog timer reset when program is executed repeatedly by FOR and NEXT instructions

The watchdog timer can also be reset by executing the WDT instruction in a sequence program.

If the watchdog timer expires while the FOR and NEXT instructions are repetitiously executed, reset the watchdog time with the WDT instruction.

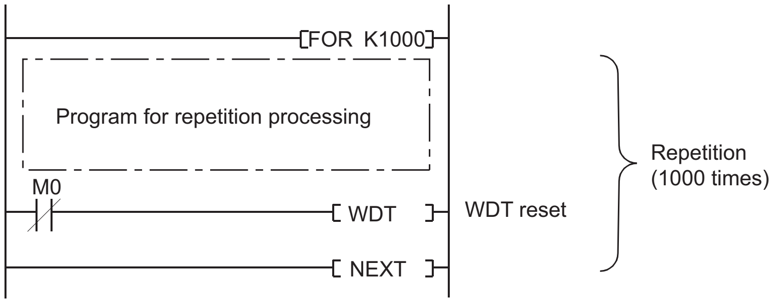


Figure 6.83 Watchdog timer reset when FOR and NEXT instructions are executed

(c) Scan time when WDT instruction is used

The scan time value is not reset even if the watchdog timer is reset in the sequence program.

The scan time value is measured to the END instruction.

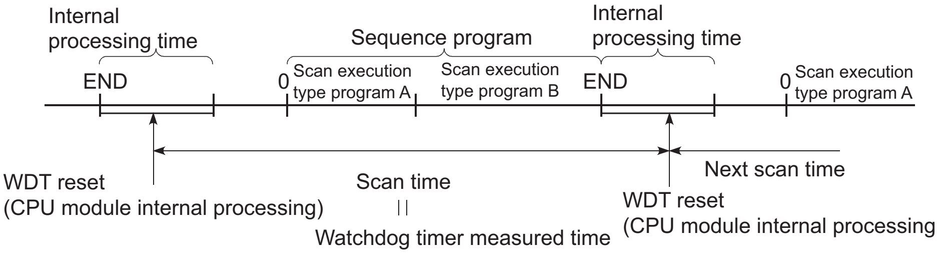

Internal processing time
END
Sequence program
0
Scan execution type program A
Scan execution type program B
Internal processing time
END
0
Scan execution type program A
WDT reset (CPU module internal processing)
Scan time
||
Watchdog timer measured time
Next scan time
WDT reset (CPU module internal processing

Figure 6.84 Watchdog Timer Reset

☒POINT

1. Scan time is the time from when the CPU module starts processing a sequence program at Step 0 until it restarts processing another sequence program with the same filename at Step 0.
   The scan time is not consistent for each and changes depending on the followings.
   - Whether the commands used are executed or not executed.
   - Whether to execute or not an interrupt program and a fixed scan execution type program.Note6.52
2. To execute at the same scan time at every scan, use the constant scan function. (☞ Section 6.2)

Basic
Note6.52

Basic
Note6.52

The Basic model cannot execute fixed scan execution type programs.

6.17 Self-diagnostics Function

(1) Definition of Self-Diagnosis Function

The self-diagnosis is a function performed by the High Performance model QCPU itself to diagnose whether there is an error in the CPU module.
The self-diagnosis function is used to prevent the CPU module erroneous operation as well as preventive maintenance.

(2) Self-diagnostics timing

When an error occurs at power-on of the CPU module or during RUN/STOP of the CPU module, an error is detected and displayed, and the operation of the CPU module will be stopped by the self-diagnostic function.
Note that errors may not be detected by the self-diagnostic function depending on the error occurrence status and the instruction to be executed. Even if the operation will not be stopped by the self-diagnostics, establish a safety circuit outside the PLC so that the entire system operates safely.

(3) Error checking

(a) LED ON

When the CPU module detects an error, it turns on ERR. LEDs.

(b) Error information storage destination and checking

When the CPU module detects an error, it turns ON the special relay (SM0, SM1) and stores the information (error code) into the special register (SD0).
When multiple errors are detected, error codes of the latest errors are stored in the special register SD0. For error detection, use special relays and special registers in programs so that these devices can interlock with sequencers and mechanical systems.

☒POINT

1. In the case of the Redundant CPU, the information of the error that occurred in the other system is stored into the special relay (SM1610 to 1626) and special register (SD1610 to 1636).
   In any of the following cases, the other system error information is not stored.
   - The other system is powered OFF/reset or has developed a hardware failure.
   - A watchdog timer error (error code: 5000, 5001) has occurred.
   - Tracking cable fault (not installed, disconnected, failed, etc.)
2. The information of the errors that indicate the following CPU module statuses are not stored into the special relays (SM0, SM1) or special registers (SD0 to 26). The ERR. LED is not turned on, either.
   The error information is stored into the error history (☞ Section 6.18).
   - System switching (from standby system to control system) switching has occurred (Error message: CONTROL EXE. Error code: 6200)
   - System switching (from control system to standby system) has occurred. (Error message: STANDBY. Error code: 6210)

(4) Error history checking

The CPU module stores 16 latest error codes. (☞ Section 6.18)
The failure history can be checked in the GX Developer function PLC diagnostics mode.
The failure history can be stored even when the power is shut off using the battery backup.

(5) CPU module operation at the time of error detection

(a) Mode at error detection

When an error is detected from the self-diagnosis, there are two types of modes that the CPU module operation can change to.

1) CPU module calculation stop mode

At detection of an error, the CPU module stops operation and turns OFF all external outputs of the modules set to "Clear" (default) in the "error time output mode" in the I/O assignment setting of the PLC parameter dialog box. (The output (Y) on the device memory is held.)
Note that the external outputs of the modules set to "Hold" in the "error time output mode" are held. (The output (Y) of the device memory is held.)

2) CPU module calculation continue mode

At detection of an error, the CPU module executes programs other than the one that has developed the error.

(b) Errors that can be set to continue/stop the operation

The following errors can be set to "continue/stop" the operation when they have occurred.

1) Errors that can be set to continue/stop the operation in PLC RAS setting of PLC parameter dialog box

- Computation error
- Extended instruction error (Setting for future extension)
- Fuse blown
- Module comparison error
- Intelligent module program execution error
- File access error[Note6.53]
- Memory card operation error[Note6.53]
- External power supply OFF (Setting for future extension) [Note6.53]

Basic Note6.53

For example, when the module comparison error is set to "continues", the calculations are continued in the I/O address before the error occurred.
For details of errors, refer to Self-diagnostics list in Section 6.17 (7).

2) Error can be set to continue/stop the operation in detail setting of I/O assignment setting of PLC parameter dialog box

- Intelligent function module error

Basic Note6.53

The operation mode of "File access error", "Memory card operation error", and "External power supply OFF" cannot be set for the Basic model QCPU.

(6) **Selecting error check items**

Whether to execute an error check or not can be set for the following items on the PLC RAS setting tab of PLC parameter in GX Developer.
(The default of all items is set to "Execute (with a checkmark in the checkbox)".)

(a) **Battery check ("Carry out battery check")**

(b) **Fuse blown check ("Carry out fuse blown check")**

Basic Note6.54 High Performance Note6.54 Process Note6.54

Redundant Note6.54

(c) **Module verification ("Verify module")**

(d) **Device range check at index modification ("Check device range at indexing")**Note6.54

Universal Note6.55

(e) **Redundant power supply system diagnostics ("Diagnose redundant power supply system")**Note6.54, Note6.55

Basic Note6.54 High Performance Note6.54 Process Note6.54

Redundant Note6.54

Error check items, "Check device range at indexing" and "Diagnose redundant power supply system", cannot be set for the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU.

Universal Note6.55

When setting the error check item "Diagnose redundant power supply system" in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

(7) Self-diagnostics list

Table6.35 lists the self-diagnostics performed in the CPU module.
The error messages in the "Error message" column can be checked on the screen displayed by selecting [Diagnostics] → [PLC diagnostics] in GX Developer.
The numbers in the "CPU module" column correspond to the CPU modules as indicated in Table6.34.

Table6.34 Number in the "CPU module" column and corresponding CPU module

No.	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU
5)	Universal model QCPU

Table6.35 Self-diagnostics list

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Hardware failure	CPU error	MAIN CPU DOWN	• Always	Stop	OFF	Flash	○	○	○	○	○
	END instruction not executed	END NOT EXECUTE	• When the END instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	SFC program execution error	SFCP. END ERROR	• When SFC program is executed	Stop	OFF	Flash	○	×	○	×	○
	RAM check	RAM ERROR	• When the CPU module is powered on/reset	Stop	OFF	Flash	○	○	○	○	○
	Tracking memory or tracking hardware fault	TRK. CIR. ERROR	• When the CPU module is powered on/reset • When operation is being performed	Stop	OFF	Flash	×	×	×	○	×
	Calculation circuit check	OPE.CIRCUIT ERR.	• When the CPU module is powered on/reset • When END processing is executed	Stop	OFF	Flash	○	○	○	○	○
	Fuse blown *1*2	FUSE BREAK OFF	• Always	Stop/ Continue	OFF/ON	Flash/ON	○	○	○	○	○
	I/O interrupt error	I/O INT. ERROR	• When an interrupt occurs	Stop	OFF	Flash	○	○	○	○	○
	LAN controller error	LAN CTRL. DOWN	• When the CPU module is powered on/reset	Stop	OFF	Flash	×	×	×	×	*8 ○

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Hardware failure	Intelligent function module error *1	SP.UNIT DOWN	• When the CPU module is powered on/reset • When FROM/TO instruction is executed • When intelligent function module dedicated instruction is executed • When END instruction is executed	Stop/ Continue	OFF/ON	Flash/ON	○	○	○	○	○
	Control bus error	CONTROL-BUS ERR.	• When the CPU module is powered on • When END instruction is executed • When FROM/TO instruction is executed • When intelligent function module dedicated instruction is executed • Always	Stop	OFF	Flash	○	○	○	○	○
	Momentary stop occurred	AC/DC DOWN	• Always	Continue	ON	OFF	○	○	○	○	○
	Multiple CPU high speed bus dedicated instruction error	MULTI-C.BUS ERR.	• When the CPU module is powered on/reset	Stop	OFF	Flash	×	×	×	×	○
	Redundant base unit power supply voltage drop	SINGLE PS. DOWN	• Always	Continue	ON	ON	×	*4 ○	*4 ○	○	*9 ○
	Redundant power supply module fault	SINGLE PS. ERROR	• Always	Continue	ON	ON	×	*4 ○	*4 ○	○	*9 ○
	Battery low*3	BATTERY ERROR	• Always	Continue	ON	BAT.ALM LED ON	○	○	○	○	○
	ROM write count error	FLASH ROM ERROR	• At ROM write	Continue	ON	ON	×	×	×	×	○

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Handling error	Module comparison *1 *2	UNIT VERIFY ERR.	• When the END instruction is executed	Stop/ Continue	OFF/ON	Flash/ON	○	○	○	○	○
	Base assignment error	BASE LAY ERROR	• When the CPU module is powered on/reset	Stop	OFF	Flash	○	×	×	○	○
	Intelligent function module allocation error	SP.UNIT LAY ERR.	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	Intelligent program execution error *1	SP.UNIT ERROR	• When the FROM/TO instruction is executed	Stop/ Continue	OFF/ON	Flash/ON	○	*5 ○	○	○	○
	Intelligent function module version error	SP.UNIT VER.ERR	• When the CPU module is powered on/reset	Stop	OFF	Flash	*5 ×	×	○	○	○
	No parameter	MISSING PARA.	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	Boot error	BOOT ERROR	• When the CPU module is powered on/reset	Stop	OFF	Flash	○	○	○	○	○
	Backup error	ESTORE ERROR	• When the CPU module is powered on/reset	Stop	OFF	Flash	×	×	×	×	○
	Memory card operation error *1	ICM.OPE. ERROR	• When the memory card is installed/ removed	Stop/ Continue	OFF/ON	Flash/ON	×	○	○	○	○
	File setting error	FILE SET ERROR	• When the CPU module is powered on/reset • When writing data to the programmable controller	Stop	OFF	Flash	○	○	○	○	○
	File access error *1	FILE OPE. ERROR	• When an instruction is executed	Stop/ Continue	OFF/ON	Flash/ON	×	○	○	○	○
	Instruction execution not possible	CAN'T EXE.PRG.	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Parameter error	Parameter setting check	PARAMETER ERROR	• When the CPU module is powered on/reset • When switched from STOP to RUN • When writing data to the programmable controller	Stop	OFF	Flash	○	○	○	○	○
	Link parameter error	LINK PARA.ERROR	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	SFC parameter error	SFC PARA.ERROR	• When switched from STOP to RUN • When writing data to the programmable controller	Stop	OFF	Flash	○	○	○	○	○
	Intelligent function module parameter error	SP.PARA. ERROR	• When the CPU module is powered on/reset	Stop	OFF	Flash	○	○	○	○	○
Password error		REMOTE PASS.ERR	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
Program error	Instruction code check	INSTRUCT. CODE ERR	• When the CPU module is powered on/reset • When switched from STOP to RUN • When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	No END instruction	MISSING END INS.	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	Pointer setting error	CAN'T SET(P)	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	Pointer setting error	CAN'T SET(I)	• When the CPU module is powered on/reset • When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Program error	Operation error[*1 *6]	OPERATION ERROR	• When an instruction is executed	Stop/ Continue	OFF/ON	Flash/ON	○	○	○	○	○
	FOR to NEXT instruction structure error	FOR NEXT ERROR	• When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	CALL to RET instruction structure error	CAN'T EXECUTE(P)	• When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	Interrupt program error	CAN'T EXECUTE(I)	• When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	Instruction execution not possible	INST.FORMAT ERR.	• When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	Multiple CPU dedicated instruction	MULTI COM. ERROR	• When the instruction is executed	Stop	OFF	Flash	×	×	×	×	○
	SFC program structure error	SFCP.CODE ERROR	• When switched from STOP to RUN	Stop	OFF	Flash	×	○	○	○	×
	SFC block structure error	CAN'T SET(BL)	• When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	SFC step structure error	CAN'T SET(S)	• When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	SFC execution error	SFC.EXE. ERROR	• When switched from STOP to RUN	Continue	ON	ON	○	×	×	×	○
	SFC syntax error	SFCP. FORMAT ERR.	• When switched from STOP to RUN	Stop	OFF	Flash	○	○	○	○	○
	SFC operation check error[*1]	SFCP.OPE. ERROR	• When an instruction is executed	Stop/ Continue	OFF/ON	Flash/ON	×	○	○	○	×
	SFC program execution error	SFCP.EXE. ERROR	• When switched from STOP to RUN	Continue	ON	ON	○	○	○	○	○
	SFC block execution error	BLOCK EXE.ERROR	• When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
	SFC step execution error	STEP EXE.ERROR	• When an instruction is executed	Stop	OFF	Flash	○	○	○	○	○
PLC error	Watchdog error supervision	WDT ERROR	• Always	Stop	OFF	Flash	○	○	○	○	○
	Program time exceeded	PRG.TIME OVER	• Always	Continue	ON	ON	○	○	○	○	○

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Redundant system fault	Program, parameter, DIP switch mismatch	FILE DIFF.	• Always • When the CPU module is powered on/ reset • When tracking cable is connected • When switched to the backup mode • When writing data during RUN • When the system is switched • When both systems are switched to RUN	Stop	OFF	Flash	×	×	×	○	×
	Operation status, key switch mismatch	OPE.MODE DIFF.	• When the CPU module is powered on/ reset • Always	Stop/ Continue	OFF/ON	Flash/ON	×	×	×	○	×
	Module mounting configuration mismatch	UNIT LAY. DIFF.	• Always • When the CPU module is powered on/ reset • When tracking cable is connected • When operation mode is changed	Stop	OFF	Flash	×	×	×	○	×
	Memory card installation status mismatch	CARD TYPE DIFF.	• When the CPU module is powered on/ reset	Stop	OFF	Flash	×	×	×	○	×
	Current mode-time function execution disable	CAN'T EXE. MODE	• Always	Continue	ON	ON	×	×	×	○	×
	Both systems file consistency error Parameter-valid drive settings consistency error	CPU MODE DIFF.	• When the CPU module is powered on/ reset • When END instruction is executed • When tracking cable is connected	Stop	OFF	Flash	×	×	×	○	×
	Tracking data transmission error	TRK.TRANS. ERR.	• Always	Continue	ON	ON	×	×	×	○	×
	Tracking capacity excess error	TRK.SIZE ERROR	• When END instruction is executed	Continue	ON	ON	×	×	×	○	×

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Redundant system fault	Tracking cable fault, tracking hardware failure	TRK.CABLE ERR.	• When the CPU module is powered on/ reset	Stop	OFF	Flash	×	×	×	○	×
	Tracking cable not connected, failure, tracking hardware failure	TRK. DISCONNECT	• Always	Continue	ON	ON	×	×	×	○	×
	Tracking initial error	TRK.INIT. ERROR	• When the CPU module is powered on/ reset	Stop	OFF	Flash	×	×	×	○	×
	System switching (form standby system to control system) occurrence[*7]	CONTROL EXE.	• Always	Continue	ON	OFF	×	×	×	○	×
	System switching (from control system to standby system) occurrence[*7]	STANDBY	• Always	Continue	ON	OFF	×	×	×	○	×
	System switching error	CAN'T SWITCH	• When system switching is executed	Continue	ON	ON	×	×	×	○	×
	Standby system not started/stop error	STANDBY SYS.DOWN	• Always	Continue	ON	ON	×	×	×	○	×
	Control system not started/stop error	CONTROL SYS.DOWN	• Always	Stop	OFF	Flash	×	×	×	○	×
	Program memory cleared	PRG.MEM. CLEAR	• When program memory copy function is executed	Stop	OFF	Flash	×	×	×	○	×
	Memory copy function executed	MEM.COPY EXE.	• When memory copy function is executed	Continue	ON	ON	×	×	×	○	×
	Tracking setting parameter error	TRK.PARA. ERROR	• When the CPU module is powered on/ reset	Stop	OFF	Flash	×	×	×	○	×
	When the CPU module is powered on/reset	CPU LAY ERROR	• When the CPU module is powered on/ reset	Stop	OFF	Flash	×	×	×	○	×
Multiple CPU error	Other CPU major error	MULTI CPU DOWN	• Always • When the CPU module is powered on/reset	Stop	OFF	Flash	○	○	○	×	○
	Multiple CPU system not configurable	MULTI EXE.ERROR	• When the CPU module is powered on/ reset	Stop	OFF	Flash	○	○	○	×	○

○ : Self-diagnostics is performed, × : Self-diagnostics is not performed

(Continued on next page)

Table6.35 Self-diagnostics list (Continued)

Diagnostics		Error message	Diagnostics timing	CPU module status	LED status		CPU module				
					RUN	ERR.	1)	2)	3)	4)	5)
Multiple CPU error	Multiple CPU consistency error	CPU LAY.ERROR	• When the CPU module is powered on/reset	Stop	OFF	Flash	○	○	○	×	○
	Other CPU minor error	MULTI CPU ERROR	• Always	Continue	ON	ON	○	○	○	×	○
File diagnostics check		INCORRECT FILE	• When the CPU module is powered on/reset • When switched from STOP to RUN • At PLC writing	Stop	OFF	OFF	×	×	×	×	○
Annunciator check		F****	• When an instruction is executed	Continue	ON	USER LED ON	○	○	○	○	○
CHK Instruction check		<CHK>ERR***-***	• When an instruction is executed	Continue	ON	USER LED ON	×	○	○	○	×
BOOT OK		BOOT OK	• When the CPU module is powered on/reset	Stop	OFF	Flash	×	○	○	○	×

○ : 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 : The check item can be set to "Not execute (without a checkmark in the checkbox)" with the parameter setting in GX Developer. (The default is set to "Execute (with a checkmark in the checkbox)".) Besides, the check is not executed when SM251 is ON.
* 3 : The check can be set to "Not execute (without a checkmark in the checkbox)" with the parameter setting in GX Developer. (The default is set to "Execute (with a checkmark in the checkbox)".)
* 4 : In a multiple CPU system configuration, it is detected only by CPU module of CPU No.1 when serial No. (first five digits) of all CPU modules is "07032" or later.
* 5 : The CPU module whose function version is B or later is required.
* 6 : Operation errors which occur when device range is checked for index-modified devices are included.
* 7 : Since this diagnostics indicates the 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.
* 8 : Only the Built-in Ethernet port QCPU is targeted.
* 9 : CPU modules whose serial number (first five digits) is "10042" or later are targeted.

☒POINT

In the case of the Redundant CPU, the information of the error that occurred in the other system is stored into the special relay (SM1610 to 1626) and special register (SD1610 to 1636).

In any of the following cases, the other system error information is not stored.

- The other system is powered OFF/reset or has developed a hardware failure.
- A watchdog timer error (error code: 5000, 5001) has occurred.
- Tracking cable fault (not installed, disconnected, failed, etc.)

Basic Note6.56

Universal Note6.56

6.17.1 Interrupt due to error occurrence

The CPU module can execute the interrupt program of the interrupt pointer that is set as the interrupt object when an error occurs.

(1) Interrupt caused by the error that can be set to continue/stop in PLC RAS setting

Only when the error set to "continue" at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box occurs, the High Performance model QCPU executes the interrupt program corresponding to the error. When the error set to "stop" there occurs, the interrupt program (I32) for "Stop all errors" is executed.

High Performance Note6.57 Process Note6.57

(2) Interrupt pointers and corresponding errorsNote6.57

Figure 6.85 indicates the interrupt pointers and corresponding errors.

Interrupt pointer	Corresponding error message
I32	Stop all errors*1
I33	SINGLE PS.DOWN *2
I34	UNIT VERIFY ERR. FUSE BREAK OFF SP.UNIT ERROR MULTI CPU ERROR
I35	OPERATION ERROR SFCP OPE.ERROR SFCP EXE.ERROR EX.POWER OFF
I36	ICM.OPE.ERROR FILE OPE.ERROR
I37	
I38	PRG.TIME OVER
I39	CHK instruction Annunciator detect
I40	CAN'T SWITCH
I41	STANDBY

Errors that occur when the system can continue the drive mode, where or "continue" is selected from "continues/stops".

* 1 : The I32 interrupt program is not executed upon the following serious errors.
  - •MAIN CPU DOWN
  - •END NOT EXECUTE
  - •RAM ERROR
  - •OPE CIRCUIT ERR.

* 2 : Applicable for the CPU module No.1 only when the multiple CPU system is configured by the High Performance model QCPU or the Process CPU.

Figure 6.85 List of interrupt pointers corresponding to interrupts due to error occurrence

Basic Note6.56 Universal Note6.56

Since the Basic model QCPU and Universal model QCPU does support interrupt pointers for error occurrence, it cannot perform interrupts caused by the error occurrence of the CPU module itself.

High Performance Note6.57 Process Note6.57

The High Performance model QCPU and Process CPU do not support the interrupt pointers I33, I40 and I41.
In addition, I33 applies to the CPU of which the serial No. of first 5 digits is "07032" or later.

☒POINT

The interrupt pointers I32 to 41 is at an execution disable mode when the power is started or CPU module is reset. When using I32 to 41, use the IMASK instruction and EI instruction to enable execution.

Refer to the following manual for the IMASK instruction and EI instruction.

☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

(3) Precautions when the Redundant CPU is used

(a) Precautions for using the interrupt program of interrupt pointer I41

I41 is an interrupt pointer used when system switching from control system to standby system occurs.

Note the following points for the I41 interrupt program since it is executed by the new standby system (system that was changed from the control system to the standby system by system switching) after system switching from control system to standby system.

1) Special relays, "Control/Stanby Status flag" (SM1515, SM1516)

Special relays (SM1515: OFF, SM1516: ON) indicate the operation of the standby system.

2) When changing the tracking target device

When programming has been performed to change the tracking target deice in the I41 interrupt program, the device of the new standby system is overwritten by the tracking of the new control system (system that was changed from the standby system to the control system by system switching).

Carry out the following operations when changing the tracking device in the I41 interrupt program.

- In the tracking setting (☞Section 9.2(2)) of the redundant parameter dialog box, remove the device to be changed from the tracking target.
- Performing programming so that the contents of the tracking target device will be transferred to another device and the changed.

3) Outputs to modules and network modules mounted on the main base unit

Since the I41 interrupt program is executed by the new standby system, the following outputs are not executed.

- Output (Y) to the modules mounted on the main base unit
- Output (Y) to the MELSECNET/H remote I/O network
- Transmission of the link relay (B) and link register (W) data to the other stations of MELSECNET/H remote I/O network and MELSECNET/H PLC to PLC network

6.17.2 LED display at the time of error occurrence

At the time of error occurrence, the LED located on the front of the CPU module turns on or flashes. (☞Section 6.21)

6.17.3 Error Clear

CPU module error clear operation can be performed only for error that can continue the CPU module operation.

(1) Procedures for error clear

The error clear is performed as follows:

- Resolve the cause of error.
- Store the error code of the error to be canceled in the special register SD50.
- Switch special relay SM50 from OFF to ON.
- The error is canceled.

(a) Error Clear procedure when multiple errors have occurred

When the latest error (error stored in the special register SD0) is cleared, the error information of the special relays/registers (SM0, SM1, SM5, SM16, SD0 to 26) is cleared, and therefore the information of the errors not cleared cannot be obtained from the special relays/registers.

For the errors that have not been cleared, obtain the past errors from the error history (☞Section 6.18).

(2) Status after error clear

When the CPU module is recovered by clearing the error, the special relay, special register, and LED affected by the error are set to the status before the error occurred.

When the same error occurs after clearing the error, it is logged again in the error history.

(3) Clear of annunciator

For the clear of the annunciator detected multiple times, only the first detected "F" is cleared. (☞Section 10.2.5)

☒POINT

1. When error cancellation is performed by storing the code of the error to cancel is stored in SD50, the lower 2 digits of the code number is ignored.
   [Example]
   When the error codes 2100 and 2101 occur, canceling the error code 2100 will also cancel the error code 2101.
   When 2100 and 2111 occur in the error code and error code 2100 is canceled, error code 2111 is canceled as well.

2. If the cause of the error is not in the CPU module, the error cause cannot be resolved if error cancellation is performed using the special relay (SM50) and special register (SD50).
   [Example]
   Since "SP. UNIT DOWN" is an error that occurred on the Q bus, the error cause cannot be resolved if error cancellation is performed using the special relay (SM50) and special register (SD50).
   Refer to the error code list in the QCPU User's Manual (Hardware Design, Maintenance and Inspection), and resolve the error cause.

6.18 Error History

The CPU module can store the error history (results detected from the self-diagnosis function and the time) in the memory.

The error history can be checked by choosing [Diagnostics] → [PLC diagnostics] on GX Developer.

☒**POINT**

The detection time uses the CPU module internal clock, so make sure to set the correct time when using the CPU module for the first time.

(☞ Section 6.5)

6.18.1 Basic model QCPU

(1) Storage area

The 16 latest errors are stored into the latched error history storage memory of the Basic model QCPU.

(2) Stored data

If the same error occurs two or more times while the PLC power is ON, the data is stored only once into the error history storage memory.

(3) Error history clearing method

To clear the error history storage memory, choose [Diagnostics] → [PLC diagnostics] on GX Developer and click the Clear log button.

Clearing the error history clears all data in the error history storage memory of the Basic model QCPU.

6.18.2 High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

The storage area depends on the CPU module.

(1) Storage Area

(a) High Performance model QCPU, Process CPU, Redundant CPU

The latest 16 errors are stored in the latched CPU module error history storage memory.
When storing more than 17, the history can be stored in the memory card file using PLC RAS setting in the "(PLC) Parameter" box.

(b) Universal model QCPU

All storable data are saved into the error history storage memory of the CPU module.

Table6.36 Error history file storage area

Item	High Performance model QCPU, Process CPU, and Redundant CPU	Universal model QCPU
Storage area	Built-in memory and the file in the set memory card	Built-in memory
Amount that can be stored	Max. 100 (can be changed) *1	Max. 100 *1

* 1 : When the number of storable files is exceeded, the oldest data is overwritten by the latest one.

(2) Storage data

When the number of histories set in the parameter and that stored in the memory card are different after either of the following operations is performed, the history file in the memory card is cleared, then the 16 points data in the CPU module error history storage memory is transferred to the history file in the High Performance model QCPU, Process CPU, or Redundant CPU.

- When the history count in the parameter history file is changed in the middle of operation.
- When a memory card, which has a different history count from that set in the parameter, is mounted.

The storage area in the error history file is as follows:

☒POINT

Even if the failure history file set in the parameter does not exist in the memory card, the error will not occur in the CPU module.
In this case, the CPU module stores the latest error into only the error history storage memory of the CPU module.

(3) Error history clearing method

To clear the error history storage memory/error history file, choose [Diagnostics] → [PLC diagnostics] on GX Developer and click the Clear log button.
Data stored in both the CPU module error history storage memory and error history file within the memory card can be cleared with a error history clear.

6.19 System Protect

The CPU module has several protection functions (system protect) to prevent the programs being modified illegally by a third party other than the designer using GX Developer, serial communication module, and like.

Table6.37 System protect types

Item to protect	Protect valid file	Protection description	Method	Valid Timing	Remarks
CPU (as a whole) *1 *2	All files	Prohibits all write/control instructions to the CPU module	Set the CPU module system setting switch SW1 on.	Always	Valid for devices too
Memory card *1	All files	Performs drive protect for the memory card, and write protect.	Set write-protect switch on the memory card on.	Always	--
File	Program, device comment, and device initial value files	Changes the attribute for each file as follows: • Read/Write display prohibit • Write prohibit	Change the attribute for the file in the Password Registration.	Always	--

* 1: The Basic model QCPU does not support the system protect for the CPU (as a whole) and memory cards.

* 2: The Universal model QCPU does not support the system protect for the CPU (as a whole).

The control instruction, read/write display, and write mentioned above are as follows:

Item	Description
Control instruction	CPU module operation instruction by remote operation. (Remote RUN, remote STOP, etc.)
Read/Write display	Program read/write operations.
Write	Operation related with write processing such as program writes the program and tests.

☒POINT

The following functions set the "(PLC) Parameter" and CPU module dip switches are performed even when the CPU module system's SW1 setup switch is set to ON and the system protect function is activated.Note6.58

- Booting from the standard ROM and the memory card
- Automatic write to standard ROM

Basic Universal
Note6.58 Note6.58

Basic Universal
Note6.58 Note6.58

Since the Basic model QCPU and Universal model QCPU does not include the system protect function by DIP switch setting, it is not necessary to be conscious of the restrictions described in POINT.

6.19.1 Password registration

Password is used to prohibit reading and writing data of the program and comments in CPU module from GX Developer.

(1) Valid password range

The read- and rewrite-prohibited range is set for the specified memory (program memory/standard memory/memory card) program file, device comment file<u>Note6.59</u>, and device initial file.

Basic Note6.59

(2) Operation that can be disabled by password

Either of the following two descriptions is to be registered.

- The file name is not displayed and write/read cannot be performed as well.
- Write cannot be performed to the file. (Read only)

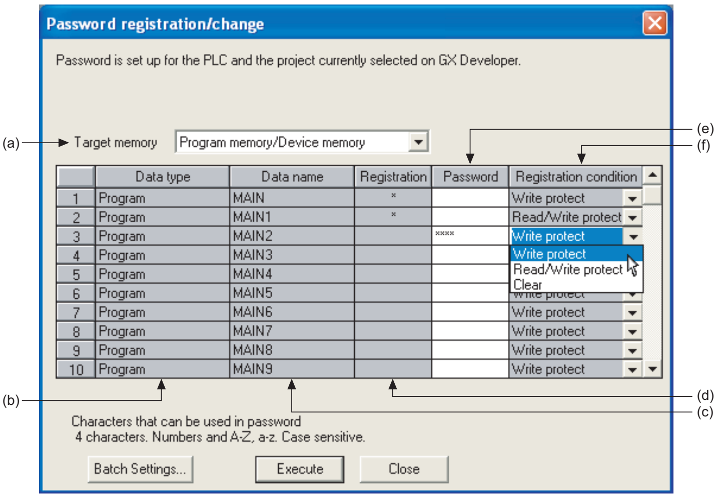
If the password is registered, file operations from GX Developer cannot be performed unless the same password is input.

Basic Note6.59

The valid password range of the Basic model QCPU can set for the program file of the program memory, the device comment file, and the device initial value file.
It cannot be set for the standard ROM and memory card.

(3) Password registration

Register the password on the Password registration/change screen of GX Developer. To display the Password registration/change screen, choose [Online] → [Password setup] on GX Developer and click the Password setup button on the Write to PLC screen.



Password registration/change

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 6.86 Password registration/change screen

Each item is described below:

(a) Target memory

Specifies the memory storing the file whose password is to be registered or changed.

(b) Data type

Displays the type of a file stored in the target memory.

(c) Data name

Displays a filename of a file stored in the target memory.

(d) Registration

Displays an asterisks "*" that indicates a password-protected file.

(e) Password

Defines or changes a password.
The registration condition can be set by specifying the password.

(f) Registration Condition

1) Write Protect

Write operation is restricted by a password.
(Reading is not allowed.)

2) Read/Write protect

Write/read operation is restricted by a password.

3) Clear

Password is cleared.
(Sets password currently registered in "Password".)

(4) Precautions

The password registered to a file can not read out from the file.
If the password can not be remembered, file operation other than following can not be performed.
Take notes of the password registered and keep it on hand.

(a) For Basic model QCPU

• Program memory	: Format PLC memory
• Standard ROM	: Write the program memory to ROM

(b) For High Performance model QCPU, Process CPU, Redundant CPU

• Program memory/Memory card	: Format PLC memory
• Standard ROM	: Write to PLC (Flash ROM)

Basic Note6.60
High Performance Note6.60

6.19.2 Remote password

(1) Definition

The remote password prevents illegal access to the CPU module by users in remote locations.

After remote passwords have been set, a remote password check is performed when the CPU module is accessed from users in remote locations.

(2) Flow from remote password setting to reflection

Set the remote password using GX Developer and write it to the CPU module. (☞ (6) in this section)

When the PLC is powered OFF and then ON or the CPU module's reset switch is set to the neutral position, the remote password is transferred to the modules that accept remote password setting (☞ (3) in this section).

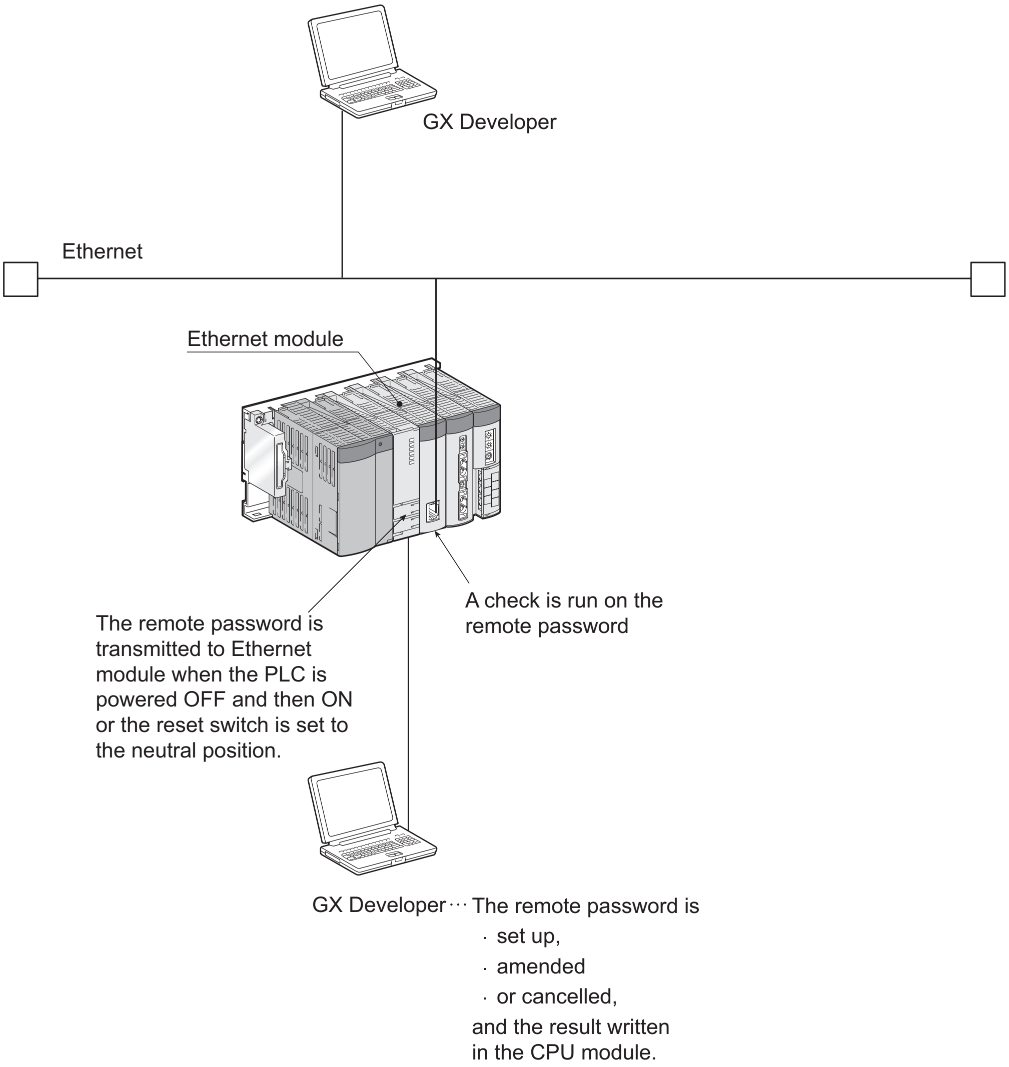

GX Developer
Ethernet
Ethernet module
The remote password is transmitted to Ethernet module when the PLC is powered OFF and then ON or the reset switch is set to the neutral position.
A check is run on the remote password
GX Developer··· The remote password is
· set up,
· amended
· or cancelled,
and the result written in the CPU module.

Figure 6.87 Outline of remote password transfer

Basic Note6.60 High Performance Note6.60

When using the remote password for the Basic model QCPU or High Performance model QCPU, check the versions of the CPU module and GX Developer.
(☞ Appendix 4.1, Appendix 4.2)

(3) Modules that support remote password setting

The following modules support remote password setting.

- Serial communication module
- Ethernet module
- Modem interface module
- Built-in Ethernet port QCPU

☒POINT

For details on the remote password of Built-in Ethernet port QCPU, refer to Section 7.7.

(4) Remote password lock/unlock processing

Unlock the remote password of the serial communication module via a modem or that of the Ethernet module via Ethernet.
When the remote password is matched, this enables the access to the CPU module.

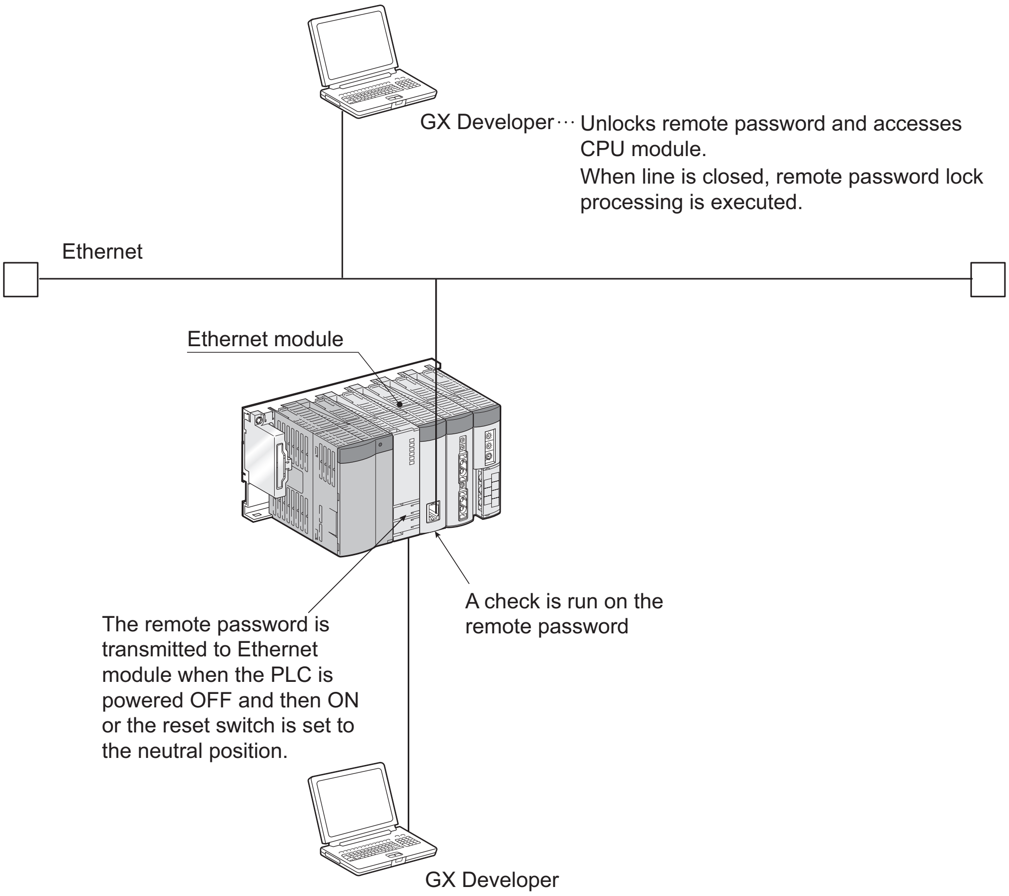

GX Developer··· Unlocks remote password and accesses CPU module.
When line is closed, remote password lock processing is executed.
Ethernet
Ethernet module
The remote password is transmitted to Ethernet module when the PLC is powered OFF and then ON or the reset switch is set to the neutral position.
A check is run on the remote password
GX Developer

Figure 6.88 Outline of remote password lock/unlock processing performed for Ethernet module

(5) Number of remote password-set modules

The number of remote password-set modules changes depending on the version of used GX Developer.

Table6.38 indicates the number of remote password-set modules depending on the GX Developer version.

Table6.38 Number of set modules according to GX Developer version

GX Developer Version	Module Name	Maximum Number of Set Modules
Version 6 Version 7	Ethernet module	4 modules
	Serial communication module	4 modules
	Modem interface module	
Version8 or later	Ethernet module	4 modules
	Serial communication module	8 modules
	Modem interface module	
	Built-in Ethernet port QCPU[*1]	1 modules

[*1] When using the Built-in Ethernet port QCPU, use the GX Developer version 8.68W or later.

☒POINT

The maximum number of set modules in the above table indicates the number of modules for which remote passwords can be set by GX Developer.

It is not the maximum number of modules mounted in the system that uses the CPU module.

Refer to the following manual for the maximum number of modules mounted in the system.

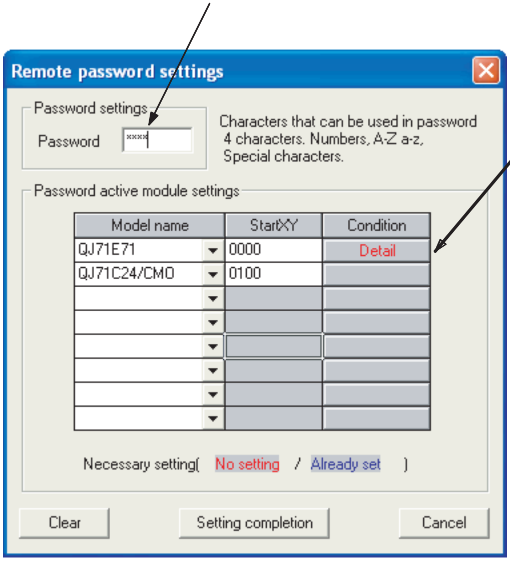
☞QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(6) Remote password setting, changing and canceling procedures

(a) Remote password setting

- In the project data list of GX Developer, choose [Parameter] → [Remote pass] to display the Remote password setting screen.
  Set the remote password.

Remote password setup



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	0100	

Necessary setting(No setting / Already set)

Detail is required with the QnUDE(H)CPU, QJ71E71

Figure 6.89 Remote password setting screen

Table6.39 Setting items on Remote password setting screen

Field			Description	Setup range/Selection range
Password settings			Remote password entry	Within 4 characters (alphanumeric character, special symbol)
Password active module settings	Model name		Model selection	QnUDE(H)CPU/QJ71E71/QJ71C24/QJ71CM0
	Start XY		Module's head address setup	• Basic model QCPU: 0000_H to $03E0_H$ • High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU : 0000_H to $0FE0_H$
Detail			------	------
	User connection No.		User's connection No. setup	Connection No. 1 to Connection No. 16
	System connection.	Auto open UDP port	Adds a check to the valid remote password port	------
		FTP transmission port (TCP/IP)		
		GX Developer transmission port (TCP/IP)		
		GX Developer transmission port (UDP/IP)		
		HTTP port		

Redundant
Note6.61

- Connect GX Developer to the CPU module.
  Write the set remote password to the CPU module.
  When a multiple CPU system is configured, write the remote password to the control CPU of the module to which the remote password is set.[Note6.61]
- When the PLC is powered OFF and then ON or the CPU module is reset, the remote password is made valid for the module.

☒POINT

Basic
Note6.62

Universal
Note6.63

1. After setting the remote password, store the parameters to the program memory (Drive 0).[Note6.62] [Note6.63]
   If the parameters are stored into other than the program memory (Drive 0), the remote password function will not be performed normally.
2. When executing boot run, store the parameter file into the standard ROM or memory card and make setting to transfer the parameter file to the program memory in the boot file setting of GX Developer.[Note6.62] [Note6.63]
   At this time, set the parameter-valid drive settings, i.e., DIP switches to the standard ROM or memory card that stores the parameter file. (If the DIP switches are set to the program memory, boot run will not be executed normally.)

(b) Remote password change

- In the project data list of GX Developer, choose [Parameter] → [Remote pass] to display the Remote password setting screen.
- Change the password and write the new password to the CPU module.

(c) Remote password deletion

- In the project data list of GX Developer, choose [Parameter] → [Remote pass] to display the Remote password setting screen.
- Click the Clear button to delete the set password.
- Write the remote password by GX Developer.

Redundant
Note6.61

The Redundant CPU is inapplicable to a multiple CPU system.

Basic
Note6.62

Since the Basic model QCPU always stores parameters to the program memory, it is not necessary to be conscious of the description in POINT.

Universal
Note6.63

Since the remote password is valid with the Universal model QCPU as long as parameters are stored in the valid drive, it is not necessary to be conscious of the description in POINT.

Remark

Refer to the following manuals for further details on the remote password function.

- When using Built-in Ethernet port QCPU
  ☞ Section 7.7
- When using Serial Communication Modules
  ☞ Q Corresponding Serial Communication Module Users' Manual (Application)
- When using Ethernet Modules
  ☞ Q Corresponding Ethernet Interface Module Users' Manual (Basic)
- When using Modem Interface Module
  ☞ QJ71CMO Modem Interface Module User's Manual (Details)

6.20 CPU Module System Display by GX Developer

After GX Developer is connected to the CPU module, the following items can be checked in the system monitor (refer to Figure 6.90 and Figure 6.91).

- Installed status
- Parameter status
- Module' s detailed information
- Product information

This section uses the system monitor screen of GX Developer Version 8.24A or later for explanation.

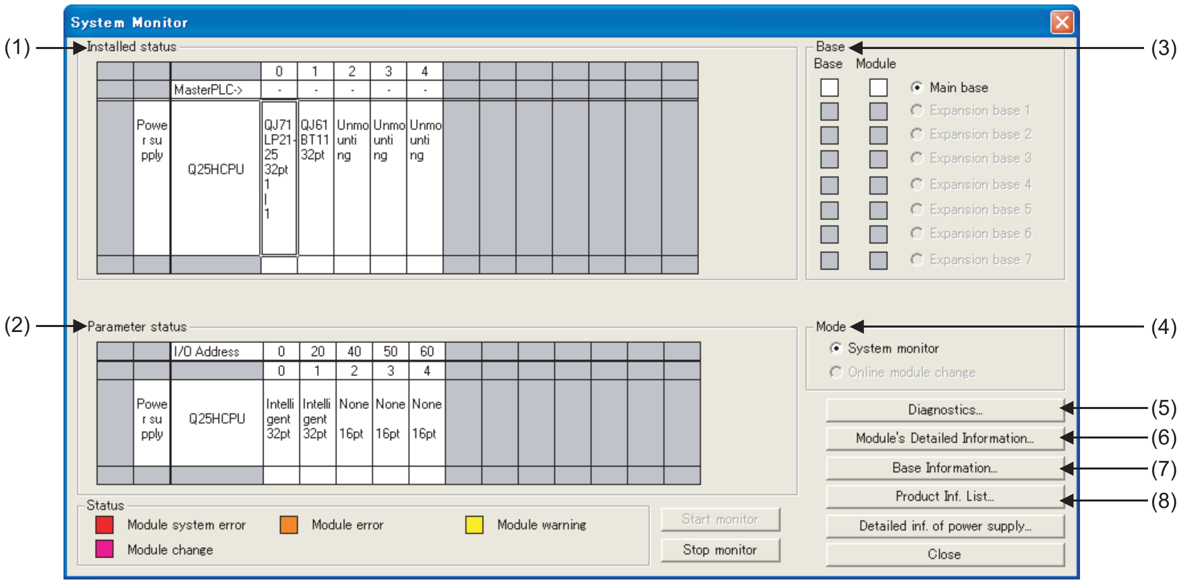


Figure 6.90 System monitor screen (except for Redundant CPU)

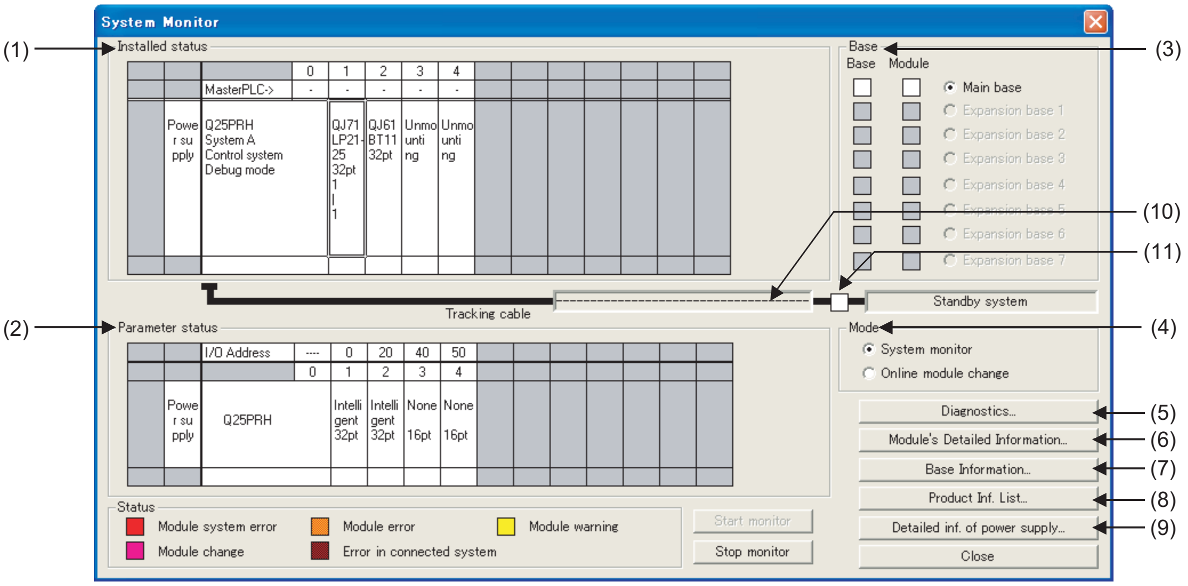


Figure 6.91 System monitor screen (for Redundant CPU)

(1) Installed status

Basic Note6.64

Enables the controlling CPU, the model an07 d the number of modules mounted onto the selected base unit to be confirmed.Note6.64
"Not installed" will be displayed for slots in which modules have not been mounted.
When slots have been set as "Empty" at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box, the module's model will not be displayed when if a module has been mounted.

Basic Universal Note6.65 Note6.66

When the redundant base module is used, the power supply module mounting status is also displayed.Note6.65, Note6.66
When the redundant CPU is used, redundant CPU status is also displayed.

(2) Parameter status

Enables the I/O number, the module type and the number of modules mounted for each of the slots on the selected base unit to be confirmed.
If the operation status shows 0 empty points and an allocation error is displayed, it means that the PLC parameter's I/O allocation and the actual status are different.
Match the I/O assignment of the PLC parameter dialog box to the actual mounting status.

Basic Universal Note6.65 Note6.66

When the Redundant base unit is used, the power supply module mounting status is also displayed.Note6.65, Note6.66

(3) Base

Enables the status of the modules mounted onto the base unit in use to be confirmed.
When there is at least one error module, the Module field turns to the status color of that error module.

(4) ModeNote6.67

Basic High Performance Universal Note6.67 Note6.67 Note6.67

Used when online module change is executed.
Refer to the following manuals for online module change.

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

☞ Manual of the module compatible with online module change

Basic Note6.64

When the Basic model QCPU is used, the control CPU of the module is not displayed.

Basic Note6.65

When the Basic model QCPU is used, the power supply module mounting status.

Universal Note6.66

To monitor the Universal model QCPU mounting status, check the versions of CPU module and GX Developer.

Basic High Performance Universal Note6.67 Note6.67 Note6.67

Since online module change cannot be executed for the Basic model QCPU, High Performance model QCPU and Universal model CPU mode selection is not available.

(5) Diagnostics

This function is used to confirm the status of the CPU module and errors.

(6) Module' s detailed information

This function is used to confirm the detailed information for selected modules.
Refer to the instruction manual for details on the relevant intelligent function module and intelligent function modules.

(7) Base information

Enables the "Overall Information" and "Base Information" to be confirmed.

(a) Overall information

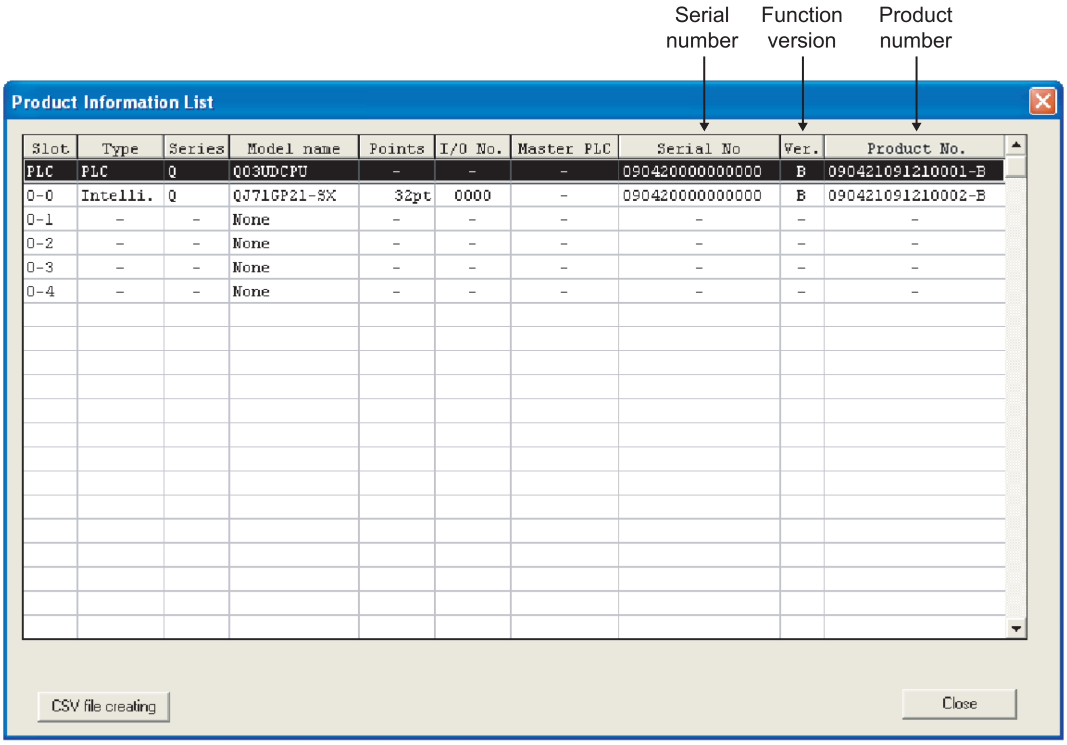
Enables the number of base units in use and the number of modules mounted on the base units to be confirmed.

(b) Base information

Enables the base name, the number of slots, the base type and the number of modules mounted onto the base for the selected base unit to be confirmed.

(8) Product Information List

The individual information (type, series, model, number of points, head I/O, control CPU, serial No., function version, and product number[*1]) of the mounted CPU module, I/O module and intelligent function can be confirmed.



Slot	Type	Series	Model name	Points	I/O No.	Master PLC	Serial No	Ver.	Product No.
PLC	PLC	Q	Q03UDCPU	-	-	-	090420000000000	B	090421091210001-B
0-0	Intelli.	Q	QJ71GP21-SX	32pt	0000	-	090420000000000	B	090421091210002-B
0-1	-	-	None	-	-	-	-	-	-
0-2	-	-	None	-	-	-	-	-	-
0-3	-	-	None	-	-	-	-	-	-
0-4	-	-	None	-	-	-	-	-	-

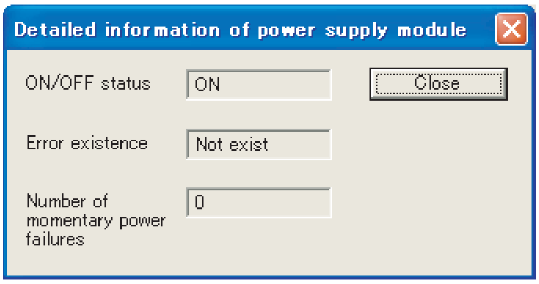
Figure 6.92 Product information list

* 1: The product number is displayed only when the Universal model QCPU is used.
Since the Basic model QCPU, High performance model QCPU, Process CPU, and Redundant CPU do not support the product number display, "-" is displayed.

(9) Detailed information of power supply module

Indicates the ON/OFF status, error occurrence and number of momentary power failures of the power supply module.

The detailed information of power supply module is available when redundant base unit or the power supply module that supports the detailed information of the power supply module is used.



Detailed information of power supply module

ON/OFF status: ON

Error existence: Not exist

Number of momentary power failures: 0

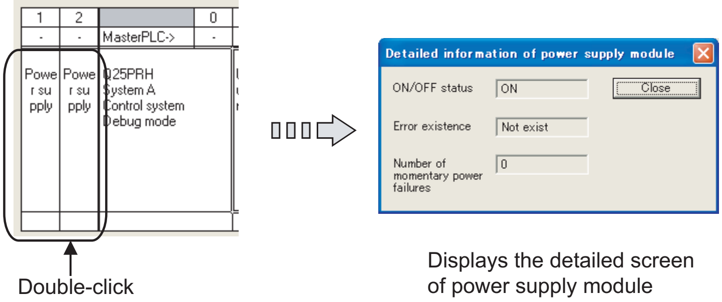
Figure 6.93 Detailed screen of power supply module

Table6.40 Description for detailed screen of power supply module

Item	Description
ON/OFF status	Displays the input power supply status to the redundant power supply module.
Error occurrence	Displays the failure (error) occurrence in the redundant power supply module.
Number of momentary power failures	• When the redundant power supply module of redundant main base unit is selected The number of momentary power failures of the redundant power supply module mounted to the redundant main base unit is displayed.(Display range: 0 to 65535) • When the redundant power supply module of redundant extension base unit is selected The number of momentary power failures is displayed as "-" and not counted.

☒POINT

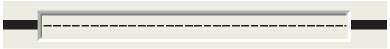
1. In a multiple CPU system configuration, the detailed information of power supply module can be displayed only when GX Developer is connected to the CPU No. 1.
2. In a multiple CPU system configuration, the detailed information of power supply module can be displayed only when all CPU module serial No. of first 5 digits is "07032" or later.
3. Double-clicking the power supply module area of mounting status can also display the detailed screen of power supply module.



(10)Memory copy status

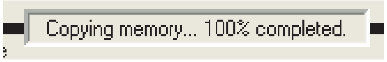
Indicates the execution progress of the memory copy from control system to standby system.

- During normal operation



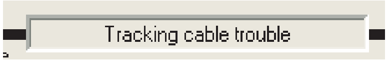
```
--------------------------------
```

- When memory copy is executed from control system to standby system



Copying memory... 100% completed.

- When tracking cable is faulty



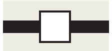
Tracking cable trouble

Figure 6.94 Memory copy status

(11)Other system status

Indicates the status of the other system.

- During normal operation



- When an error has occurred

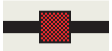


Figure 6.95 Other system status

When the Redundant CPU is in the debug mode, however, it indicates a normal operation status even if an error has occurred in the other system. (Error occurrence is not displayed.)

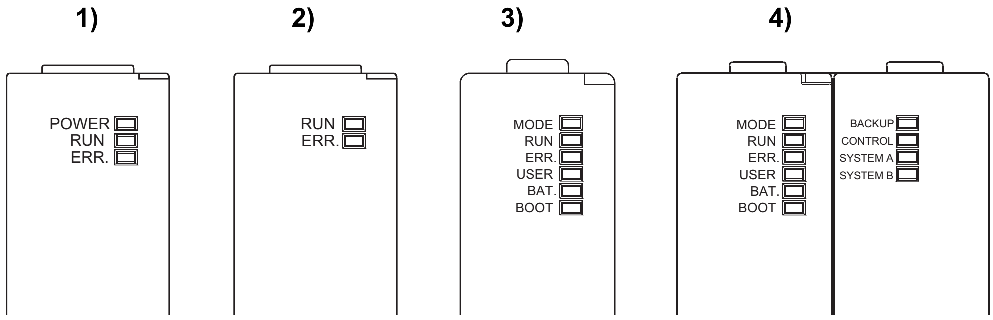
Remark

Refer to the following manual for details of the system monitor of GX Developer.

☞ GX Developer Operating Manual

6.21 LED Display

The CPU module has an LED to indicate the CPU module operation status on the front of the CPU module.



1) : Basic model QCPU (Q00JCPU)
2) : Basic model QCPU (Q00CPU, Q01CPU)
3) : High Performance model QCPU, Process CPU, Universal model QCPU
4) : Redundant CPU

Figure 6.96 LED on CPU module front

Remark

Refer to the following manual for details of the LED indications.

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

6.21.1 Method to turn off the LED

(1) Method to turn off the LED

(a) For Basic model QCPU

To turn off the ERR. LED that is on, remove the cause of the error and then operate the special relay SM50 and special register SD50 to cancel the error. (This does not apply to reset operation.)

(b) For High Performance model QCPU, Process CPU, Redundant CPU, Universall model QCPU

The LED that is on can be turned off by the following operation. (Except for the reset operation.)

Table6.41 LED turning off method

Method to Turn LED Off	Applicable LED			
	ERR.	USER	BAT.	BOOT
Executing the LEDR instruction after resolving the cause of error.	○	○	○	×
After the cause of error is resolved, cancel the error by operating the special relay SM50 and special register SD50. (Only for the operation continue errors.) *1	○	○	○	×
Turn off the LED by operating the special relay SM202 and special register SD202. *1	×	○	×	○

○ : Valid, × : Invalid

* 1 :Special relay and special register contents

SM50•••When switch from OFF to ON, the error is canceled for the error code stored in the SD50.

SD50•••The error code for the error to be canceled is stored.
Refer to the following manual for the error codes.
QCPU User's Manual (Hardware Design/Maintenance and Inspection)

SM202•••When turned from OFF to ON, the LED corresponding to each bit in the SD202 is turned off.

SD202•••This specifies the LED to turn off.

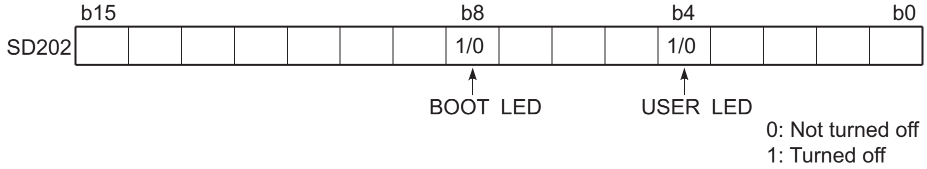


Figure 6.97 Bit configuration of special register SD202

The setting to turn off each LED is as follows. (All in hexadecimal.)

- When turning off both Boot LED and USER LED : SD202=110H
- When turning off only the BOOT LED : SD202=100H
- When turning off only the USER LED : SD202=10H

(2) Method to not turn on the ERR. LED, USER LED, and BAT. LED

It is possible to set the error message display priority for each of The ERR. LED, USER LED and BAT. LED. (Section 6.21.2)

When an error No. for an LED is deleted from this priority, the LED will not turn on even if an error with that error No. occurs.

Universal UD Note6.68

6.21.2 Priority setting

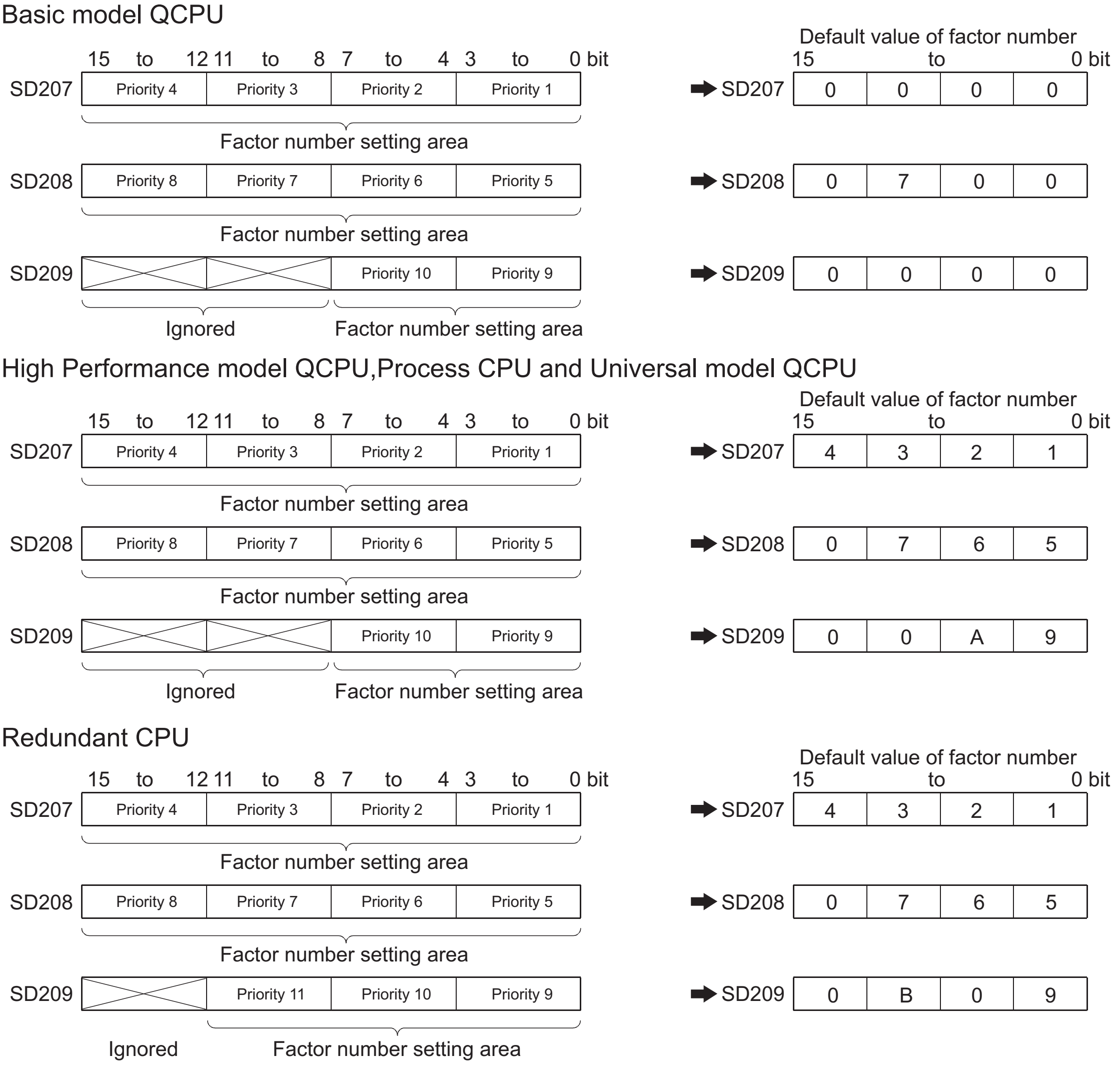
This section explains how to set the priority for error messages stored in SD220 to 227 (LED display data) at the time of error occurrence.

(1) Displayed error messages and their priorities

When multiple factors that can be displayed occur, the display is performed with the following conditions:

- A stop error is unconditionally set to the display device data (SD220 to 227).
- An operation continuation error is displayed in accordance with the priority factor number indicated in this section.
  The priority can be changed. (Set with special registers SD207 to 209)
- When errors with the same priority level occur, the error detected first is displayed.

The priority is set with the special registers SD207 to 209 in the following manner:



Basic model QCPU

	15 to 12	11 to 8	7 to 4	3 to 0 bit		Default value of factor number 15 to 0 bit			
SD207	Priority 4	Priority 3	Priority 2	Priority 1	➡SD207	0	0	0	0
SD208	Priority 8	Priority 7	Priority 6	Priority 5	➡SD208	0	7	0	0
SD209	☒	☒	Priority 10	Priority 9	➡SD209	0	0	0	0

SD207: Factor number setting area
SD208: Factor number setting area
SD209: Ignored / Factor number setting area

High Performance model QCPU,Process CPU and Universal model QCPU

	15 to 12	11 to 8	7 to 4	3 to 0 bit		Default value of factor number 15 to 0 bit			
SD207	Priority 4	Priority 3	Priority 2	Priority 1	➡SD207	4	3	2	1
SD208	Priority 8	Priority 7	Priority 6	Priority 5	➡SD208	0	7	6	5
SD209	☒	☒	Priority 10	Priority 9	➡SD209	0	0	A	9

SD207: Factor number setting area
SD208: Factor number setting area
SD209: Ignored / Factor number setting area

Redundant CPU

	15 to 12	11 to 8	7 to 4	3 to 0 bit		Default value of factor number 15 to 0 bit			
SD207	Priority 4	Priority 3	Priority 2	Priority 1	➡SD207	4	3	2	1
SD208	Priority 8	Priority 7	Priority 6	Priority 5	➡SD208	0	7	6	5
SD209	☒	Priority 11	Priority 10	Priority 9	➡SD209	0	B	0	9

SD207: Factor number setting area
SD208: Factor number setting area
SD209: Ignored / Factor number setting area

Figure 6.98 Special registers and bit configurations related to priority setting

Universal UD Note6.68

The Universal model QCPU does not support the LED display priority setting. Only the LED display status can be set.

(2) Priorities and factor Nos.

The description and default priority for the factor Nos. to be set in the special registers SD207 to 209 are as follows:

Table6.42 List of default factor Nos. and priorities

Priority	Factor number*1 (Hexadecimal)	Displayed error message	Remarks
1	1	AC/DC DOWN SINGLE PS.DOWN*2 SINGLE PS.ERROR*2	• Power shutoff • 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	• Module verification error • Fuse shutoff • Intelligent function module verify error
3	3	OPERATION ERROR SFCP OPE.ERROR SFCP EXE.ERROR	• Calculation error • SFC instruction calculation 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 FLASH ROM ERROR*4	• Memory card operation error • File access error • Operation status, switch mismatch • Current mode-time function execution disabled • Tracking data transmission error • Tracking capacity excess error • Tracking cable not connected, failure • Flash ROM access count over error
5	5	PRG.TIME OVER MULTI CPU ERROR	• Constant scan setting timeout error • Low-speed execution monitoring timeout error*5 • Other CPU error in the multiple CPU system configuration
6	6	CHK instruction*6	--
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/stop error • Memory copy function executed

* 1: The Basic model QCPU can set the priority only for the factor number 7 (annunciator).
* 2: Confirm the version of CPU module for actions to be taken in the High Performance model QCPU and Process CPU. (Appendix 4) The error message is not displayed in the Universal model QCPU.
* 3: The error message is not displayed in the Basic model QCPU, High Performance model QCPU, Process CPU, and Universal model QCPU.
* 4: The error message is not displayed in the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU.
* 5: Since the Redundant CPU and Universal model QCPU do not support low-speed execution type programs, a low-speed execution monitoring timeout error does not occur.
* 6: The Basic model QCPU and Universal model QCPU do not support the CHK instruction.

☒POINT

1. When leaving the LED turned off at the error described above, set the factor No. setting area (each 4 bits), which stores the factor No. corresponding to SD207 to 209 to "0".
   [Example]
   To leave the ERR. LED off when a fuse shutoff error is detected, set the factor No. setting area to "0" where the error No. is "2".

SD209				SD208				SD207			
0	0	A	9	8	7	6	5	4	3	0	1

Figure 6.99 Factor No. stored in SD 207 to 209.

   Because the factor No. "2" is not set, the ERR. LED remains off even if the fuse shutoff is detected. In this case, even if another error with the factor No. "2" (I/O module verify error or intelligent function module verify error) is detected, the ERR. LED remains off.

2. If 0 is set to the factor No. setting area (setting that does not turn on the LED), SM0 (diagnostics error flag) is turned ON, SM1 (self diagnostics error flag) is turned ON, and the error code is stored into SD0 (diagnostics error register).

Basic Note6.69 Process Note6.69 Redundant Note6.69 Universal Note6.69 High Performance Note6.70

6.22 High Speed Interrupt Function

When an interrupt program is created using the interrupt pointer I49, the QnHCPU can run a program by making high speed, fixed-cycle interrupts at intervals of 0.2ms to 1.0ms. And, the QnHCPU improves the I/O response by refreshing the I/O signals and intelligent function module buffer memories in the parameter-set ranges before and after the execution of the high speed interrupt program.
This enables accurate control such as precision position detection.

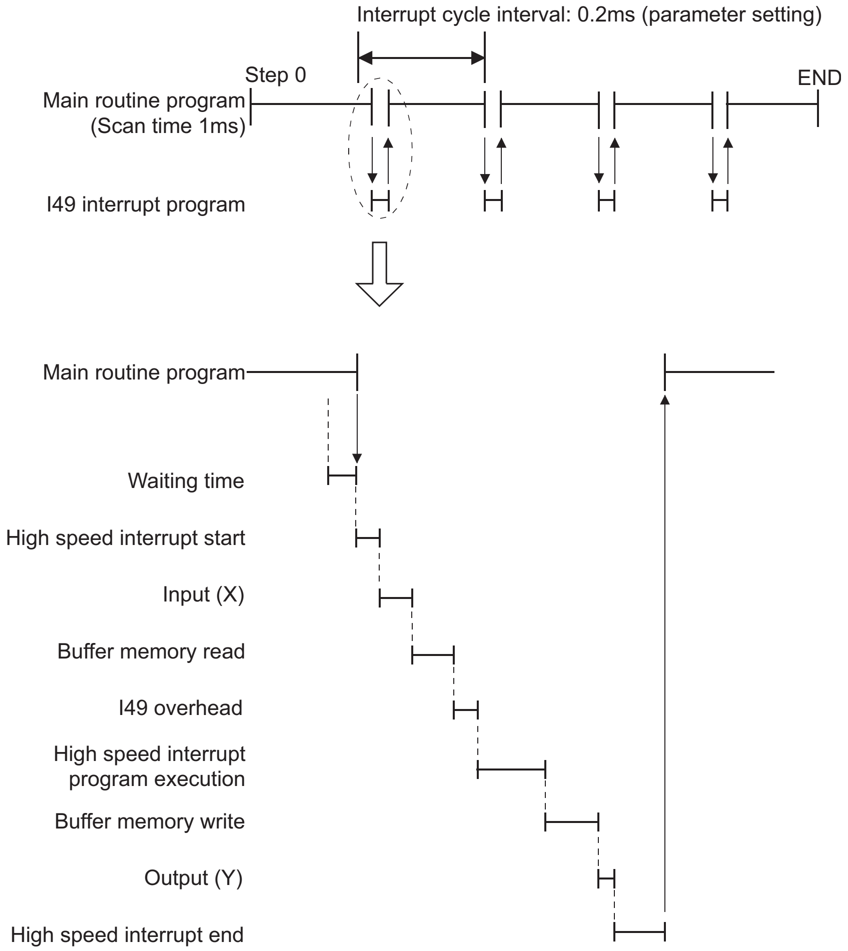


Figure 6.100 High speed interrupt timing chart

Basic Note6.69 Process Note6.69 Redundant Note6.69 Universal Note6.69

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the high-speed interrupt function.

High Performance Note6.70

The Q02CPU does not support the high speed interrupt function.
When using the high speed interrupt function with the QnHCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.2)

(1) Compatible CPUs

Table6.43 CPU modules compatible with high speed interrupt function

Compatible CPU modules	Remarks
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	There are restrictions on CPU module versions (Appendix 4.2(2))

(2) Specifications of high speed interrupt function

Table6.44 Specifications of high speed interrupt function

Item	Description	Remarks
Interrupt cycle interval	0.2 to 1.0ms	0.1ms units
Number of interrupt programs	1	Interrupt pointer I49

To set the high speed interrupt function, choose "PLC system" - "System interrupt settings" - "High speed interrupt setting" on the PLC parameter screen.

(3) Detailed items of high speed interrupt function

Table6.45 Detailed items of high speed interrupt function

Item	Description	Remarks
Interrupt program execution	Runs the interrupt program created with I49.	Section 6.22.1
High speed I/O refresh	Updates I/O signals between the I/O, intelligent function modules and CPU module at interrupt cycle intervals.	Section 6.22.2
High speed buffer transfer	Updates data between the intelligent function module buffer memories and CPU module devices at interrupt cycle intervals.	

☒POINT

Since the high speed interrupt function need to pick up interrupts at very short intervals of 0.2ms to 1.0ms with the interrupt pointer I49, please do not run the interrupt programs, which use the other interrupt pointers and fixed scan execution type programs.
If any of the interrupt and fixed scan execution type programs is executed, interrupt cannot be made at the cycle intervals set for the high speed interrupt function.
Refer to Section 6.22.4 for other restrictions.

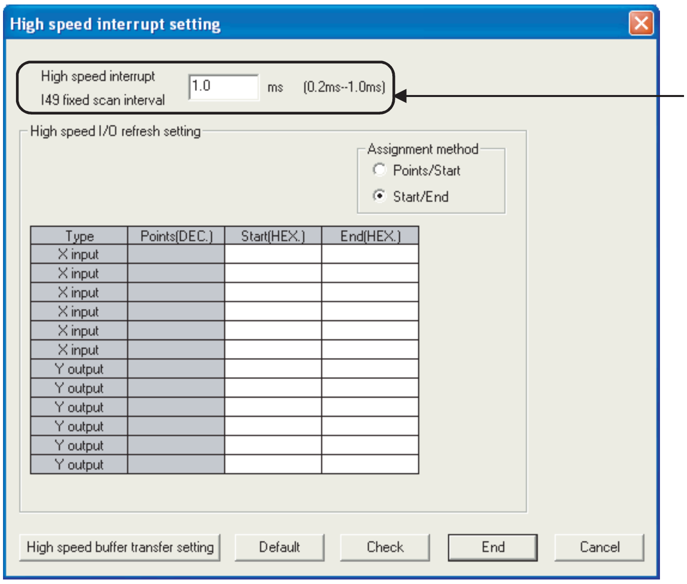
Basic
Note6.71
Process
Note6.71
Redundant
Note6.71
Universal
Note6.71
High Performance
Note6.72

6.22.1 High speed interrupt program execution

The high speed interrupt program execution function is designed to run an interrupt program according to the setting of high speed interrupt pointer I49.

(1) Setting method

Set the high speed interrupt pointer I49 at "High speed interrupt I49 fixed scan interval" after choosing "PLC system" - "System interrupt settings" - "High speed interrupt setting" on the PLC parameter screen.



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 within the range 0.2 to 1.0ms.

Figure 6.101 High speed interrupt setting

(2) Precautions

(a) High speed interrupt while interrupt is disabled

The high speed interrupt program is only executed as interrupt enable.
Refer to Section 6.22.4 (3) for the items that delay high speed interrupt's start from interrupt disable.

(b) When high speed interrupt is ignored

If the interrupt disable period continues longer than the set interrupt cycle interval, high speed interrupt may be ignored.
High speed interrupt is ignored once when it occurs twice during interrupt disable.

Basic Process Redundant
Note6.71 Note6.71 Note6.71
Universal
Note6.71

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the high speed interrupt program function.

High Performance
Note6.72

The Q02CPU does not support the high speed interrupt program execution.
When using the high speed interrupt function with the QnHCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.2)

(c) **High speed interrupt program execution**

This function is executed when the following conditions are all satisfied.

- When the EI instruction is being executed
- When the CPU module is in the RUN status
- When I49 is not masked by the IMASK instruction

By default, I49 is not masked by the IMASK instruction.

Remark

Refer to the following manual for the IMASK instruction and EI instruction.

☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Note6.73 Process Note6.73 Redundant Note6.73 Universal Note6.73 High Performance Note6.74

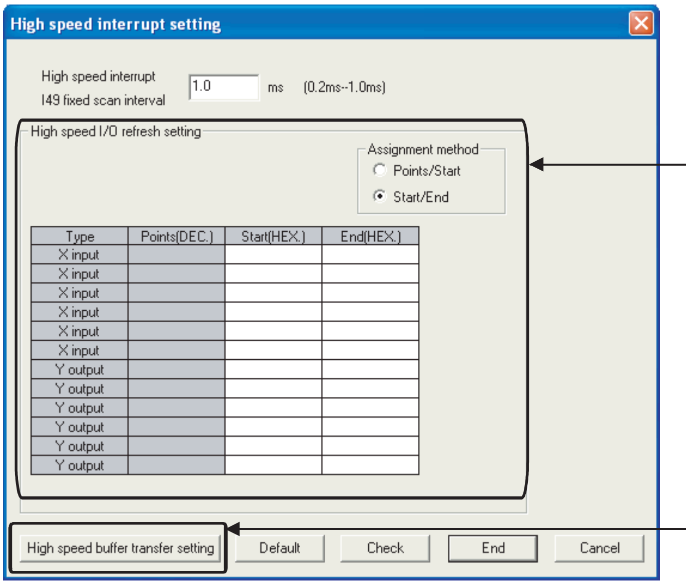
6.22.2 High speed I/O refresh, high speed buffer transfer

High speed I/O refresh is a function that updates I/O signals between the I/O and intelligent function modules and CPU module at interrupt cycle intervals.
High speed buffer transfer is a function that updates data between the intelligent function module buffer memories and CPU module devices at interrupt cycle intervals.

(1) Setting method

To execute this function, set "High speed interrupt I49 fixed scan interval" shown in Section 6.22.1, "High speed I/O refresh setting" and "High speed buffer transfer setting".

(a) High speed I/O refresh 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 the X/Y refresh ranges.

Choose here when setting the high speed buffer transfer ranges.

Figure 6.102 High speed interrupt setting screen

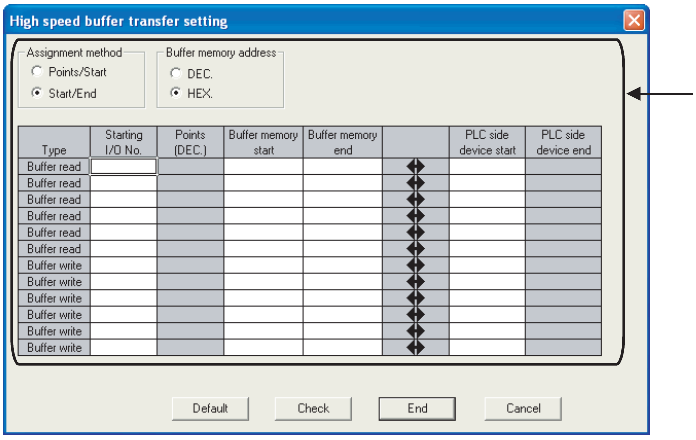
Basic Note6.73 Process Note6.73 Redundant Note6.73 Universal Note6.73

The Basic model QCPU, Process CPU, Redundant CPU, and Universal model QCPU do not support the high speed I/O refresh and high speed buffer transfer function.

High Performance Note6.74

The Q02CPU does not support the high speed I/O refresh and high speed buffer transfer function. Check the versions of CPU module and GX Developer when using the high speed interrupt function in the QnHCPU. (☞ Appendix 4.2)

(b) High speed buffer transfer setting



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					↔		

Set the buffer memory transfer ranges.

Figure 6.103 High speed buffer transfer setting screen

Table6.46 High speed I/O refresh setting and high speed buffer transfer setting

Item	Sub Item	Contents	Restrictions	Number of Settings
High speed I/O refresh setting	Points [DEC.]	Number of transferred bits (16 to 4096)	I/O and intelligent function modules Specify multiples of 16 only *1	Up to 6 points for X input and Y output, respectively.
	Start [HEX.]	Start device number (X0 to 0FF0/Y0 to 0FF0)		
	End [HEX.]	Start device number (X0 to 00FF/ Y0 to 0FFF)		
High speed buffer transfer setting	Starting I/O No.	[Start I/O number] divided by [10H (0 to FFH)]	Intelligent function modules only *2	Up to 6 points for read and write, respectively.
	Points [DEC.]	Number of transferred bits (1, 2 to FFFEH)	Intelligent function modules only Specify even addresses and even words only *3	
	Buffer memory start	Start address (0 to FFFFH)		
	Buffer memory end	Start address (FFFD to FFFFH)		
	PLC side device start	Start device number	D, W, R, ZR only	
	PLC side device start	Start device number		

* 1 : Only multiples of 16 can be set for both the head device No. and the number of transferred bits.

* 2 : Since the AnS/A series-compatible extension base unit (QA1S6□B, QA6□B) cannot be connected, the AnS/A series-compatible special function modules are not included. (A "PARAMETER ERROR" (error code: 3006) occurs when the AnS/A series-compatible extension base unit (QA1S6□B, QA6□B) is connected.)
A "PARAMETER ERROR" (error code: 3006) also occurs when an error is detected in the intelligent function module mounting status check or buffer memory capacity check.

* 3 : Only when the specified number of transferred words is 1, an odd address can be specified.

☒POINT

It is recommended to mount the target modules of this function on the main base unit.
(Access time to modules mounting on the main base unit is shorter than that on the extension base unit.)

(2) Execution of this function

This function is executed when the following conditions are all satisfied.

- When the EI instruction is being executed
- When the CPU module is in the RUN status
- When I49 is not masked by the IMASK instruction

By default, I49 is not masked by the IMASK instruction.

Remark

Refer to the following manual for the IMASK instruction and EI instruction.

☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Note6.75 Process Note6.75 Redundant Note6.75 Universal Note6.75 High Performance Note6.76

6.22.3 Processing times

The following chart shows the processing times of the high speed interrupt function from a start to an end.

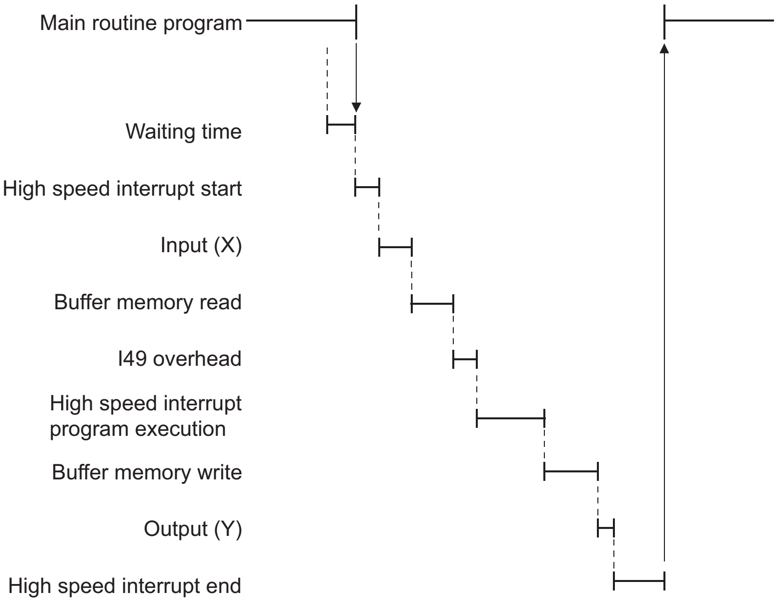


Figure 6.104 Processing from start to end of high-speed interrupt function

Basic Note6.75 Process Note6.75 Redundant Note6.75 Universal Note6.75

The Basic model QCPU, Process CPU, Redundant CPU, and Universal model QCPU do not support the high speed interrupt function.

High Performance Note6.76

The Q02CPU does not support the high speed interrupt function.
Check the versions of CPU module and GX Developer when using the high speed interrupt function in the QnHCPU. (☞ Appendix 4.2)

Table6.47 Processing times related to high speed I/O refresh and high speed buffer transfer

Processing Item	Processing Time
Waiting time	• Max. 37.5μs or more than 37.5μs instruction processing time • Max. 40μs when CC-Link IE controller network, MELSECNET/H, CC-Link or intelligent function modules are mounted on extension base unit.
High speed interrupt start high speed interrupt end	22 μs
X input	<Main base unit> Time = 0.14 × (total number of X points)+0.65 × (number of settings)+0.85 <Extension base unit> Time = 0.21 × (total number of X points)+0.65 × (number of settings)+0.85 (Calculation example) 3.74μs when the module is mounted on the main base unit, the number of settings is 1, and the number of X points is 16
Buffer memory read	<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) 4.74μs when the module is mounted on the main base unit, thenumber of settings is 1, and the number of words is 2
I49 overhead	41μs
High speed interrupt program execution	Depends on the user-created interrupt program.
Buffer memory write	<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) 4.54μs when the module is mounted on the main base unit, the number of settings is 1, and the number of words is 2
Y output	<Main base unit> Time = 0.13 × (total number of Y points) +1.55 <Extension base unit> Time = 0.2 × (total number of Y points)+1.55 (Calculation example) 3.63μs when the module is mounted on the main base unit, the number of settings is 1, and the number of Y points is 16

Basic Note6.77 Process Note6.77 Redundant Note6.77 Universal Note6.77 High Performance Note6.78

6.22.4 Restrictions

This section provides the cautions when executing the high speed interrupt function and the restrictions.
Depending on the cautions, an WDT error may occur, or high speed interrupt may not be executed at preset cycle intervals.
The items are classified into 4 types depending on the restriction level.

(1) Items that are completely disabled.
(2) Items that are restricted on the high speed interrupt.
(3) Items that have priority over the high speed interrupt when interrupt is disabled.
(4) Items other than (1) to (3)

The time taken to run an interrupt program once should not exceed the preset time of the interrupt cycle interval. (Make sure the time required for an interrupt program once is within the preset time of the interrupt cycle interval. Otherwise, high speed interrupt operation cannot be guaranteed.)

(1) Items that are completely disabled

Table6.48 Items that are completely disabled

No.	Item	Restriction	When Used
1	CPU module	Only the CPU modules listed in Section 6.22 (1) support the high speed interrupt function.	Parameter error is detected.
2	Base unit	QA1S6□B or QA6□B base unit is not available.	Parameter error is detected.
3	Multiple CPU system	Multiple CPU system is not available.	Checked at parameter setting of GX Developer.
4	Instruction	PR/PRC, UDCNT1/2, PLSY, PWM, SPD, PLOADP/PUNLOADP/PSWAPP instructions are not available.	Any of the instructions indicated on the left is not available and error is detected.
5	Instruction	Instruction that will take longer processing time than high speed interrupt cycle is not available.	Since interrupt is disabled during instruction execution, high speed interrupt is not available at preset cycle.
6	Programming unit	Programming unit is not available.	Response to instruction search will be slow, or communication error may occur on programming unit side.
7	SFC	The following two SFC functions are not available. • SFC transition monitor check function using SM90 to SM99, SD90 to SD99 • Fixed-time executed block execution function	Functions given on the left are not available and are ignored.
8	Sampling trace	Time-based sampling trace is not available.(Sampling trace can be used every scan or when detailed conditions are executed.)	Sampling trace is not executed (it is ignored). (When reading the trace data, the trace results may not be displayed.)

Basic Note6.77 Process Note6.77 Redundant Note6.77 Universal Note6.77

The Basic model QCPU, Process CPU, Redundant CPU, and Universal model QCPU do not support the high speed interrupt function.

High Performance Note6.78

The Q02CPU does not support the high speed interrupt function.
Check the versions of CPU module and GX Developer when using the high speed interrupt function in the QnHCPU. (☞ Appendix 4.2)

Table6.48 Items that are completely disabled (Continued)

No.	Item	Restriction	When Used
9	Interrupt program (I0 to I48, I50 to I255), fixed scan execution type program	Interrupt program (I0 to I48, I50 to I255) and fixed scan execution type program is not available.	Since multiple interrupts are disabled, high speed interrupt is not available at preset cycle while any interrupt or fixed scan execution type program is run.
10	Online change (Write during RUN) *1	Online change (Write during RUN) *1 is not executable.	Interrupts are disabled during execution of online change. During online change execution, the start of high-speed interrupts is delayed, resulting in not executing high-speed interrupts at preset cycle. (The start will be delayed for the following time. • Maximum 102µs for online change in ladder mode • Maximum 300ms for online change in file units)
11	File registers having the same file name as the program name	File registers having the same file name as the program name is not available.	Since interrupt is disabled when file registers having the same file name as the program name are changed, high speed interrupt is not available at preset cycle.(The following time is taken. • Max. 410µs for standard RAM • 400µs + 100µs × number of program files for SRAM card)
12	Local devices	Local devices are not available.	Since interrupt is disabled when local devices are changed, high speed interrupt is not available at preset cycle. (The following time is taken. • 390µs+170µs × n for standard RAM • 390µs+950µs × n for SRAM card n: number of program files)
13	Command of intelligent function module that accesses CPU module	Issue of CPU access command from intelligent function module that accesses CPU module, e.g. QJ71C24 or QJ71E71, is not available.	Since interrupt is disabled at the time of CPU access command issue, a high speed interrupt start is delayed during that period and high speed interrupt is not available at preset cycle. N points read/write: (0.07 × N+34)µs N points random read/write: (0.07 × N+101)µs
14	Monitor via other station	Monitor via CC-Link IE controller network, MELSECNET/H, QJ71C24 or other intelligent function module is not available during host monitoring.	If host monitor request and intelligent function module-relayed monitor request overlap, interrupt disable processing time increases. Therefore, a high speed interrupt start is delayed (102µs) during that period and high speed interrupt is not available at preset cycle.
15	Interrupt counter	Interrupt counter corresponding to interrupt pointer I49 is not available.	If there is interrupt counter setting, setting for I49 is ignored and high speed interrupt I49 is executed properly. (For other interrupt pointers, interrupt programs are not available and interrupt counters are executed.)

* 1: There are two types of Online change (Write during RUN): online change in ladder mode and online change in file units.

(2) Items that are restriced on the high speed interrupt

Table6.49 Items that are disabled only within high speed interrupt

No.	Item	Restriction	When Used
1	Device comment	In high speed interrupt program, device comment that is the same as program name is not saved/restored.	Device comment in high speed interrupt program is overwritten.
2	Index register	In high speed interrupt sequence, index register is not saved/restored.	Index register in high speed interrupt program is overwritten.
3	Bus access flag SM390	In high speed interrupt program, bus access flag SM390 is not saved/restored.	SM390 value in high speed interrupt program is overwritten.
4	Forced ON/OFF	High speed X/Y refresh area cannot be turned ON/OFF forcibly.	In high speed interrupt, forced ON/OFF is not executed and is ignored. (Does not result in time-out error.)
5	Monitor condition setting	Cannot be specified within high speed interrupt program.	Is not executed properly. (Does not result in time-out error.)
6	Execution time measurement	Cannot be specified within high speed interrupt program.	Is not executed and is ignored. (Does not result in time-out error.)

(3) Items that have priorities over the high speed interrupt when interrupt is disabled

Table6.50 Items that hold high speed interrupt by interrupt disable

No.	Item	Precaution
1	Instruction	Interrupt is disabled during instruction execution.
2	Link refresh	Interrupt is disabled during refresh (bus access). For CC-Link IE controller network, MELSECNET/H, CC-Link or intelligent function module refresh, waiting time is max. 37.5μs when modules are mounted on main base or max. 40μs when they are mounted on extension base.
3	Multiple program execution	When multiple programs are run, interrupt is disabled during program switching. Waiting time is 30μs. When high speed interrupt function has been set, the recommended number of programs is 1.
4	Monitor	For ladder monitor, device batch-monitor or device entry monitor, waiting time is as follows. (0.096 × number of device points+20)μs
5	AC DOWN	Waiting time is max. 20ms for high speed interrupt start.

(4) Items other than (1) to (3)

(a) Interrupt program setting/fixed scan program setting in PLC parameter dialog box

If "High speed execution" is selected at the "Interrupt program/fixed scan execution program setting" section on the "PLC system" tab screen in the "(PLC) parameter" dialog box of GX Developer, this setting is invalid for the high speed interrupt function.

(b) High speed buffer transfer

Using the file registers outside the setting range (range in excess of the maximum number of points) for high speed buffer transfer does not result in an error and does not transfer data to outside the range. (The contents of the other devices are not corrupted.)

(c) Precautions for program creation

As in the other interrupt programs, there are the following items as precautions for program creation. (☞ Section 3.1.3(7))

Basic Note6.79

6.23 Interrupt from the Intelligent Function Module

CPU module executes an interrupt program (I□) by the interrupt request from the intelligent function module.
For example, the serial communication module processes the data reception by an interrupt program when the following data communication functions are executed.

- Data reception during the communication with no handshaking protocol
- Data reception during the communication with bi-directional protocol

Processing data reception with an interrupt program improves the data reception speed of CPU module.

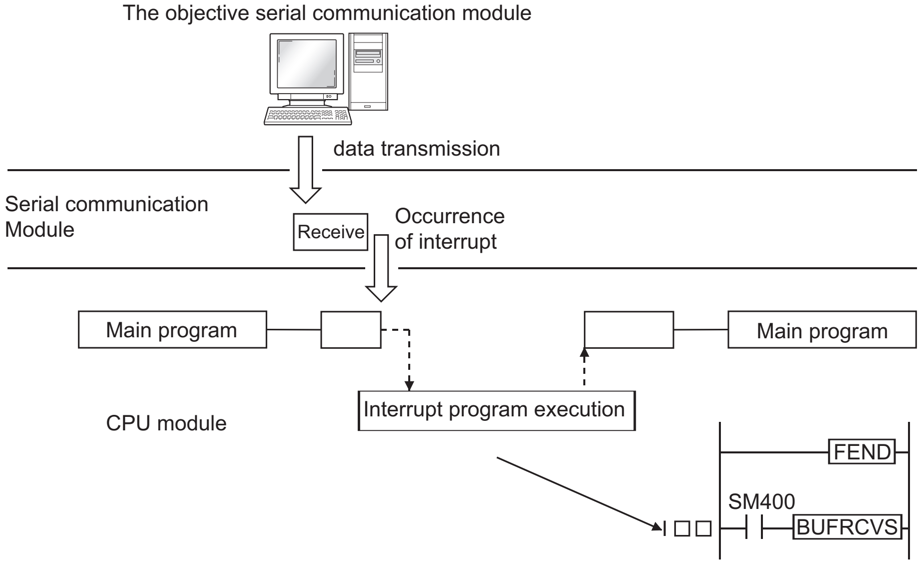


Figure 6.105 Interrupt from serial communication module

(1) Setting an interrupt from the intelligent function module

To execute an interrupt program by the interrupt of the intelligent function module, it is necessary to designate the "Intelligent function module setting (Interrupt pointer setting)" at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.
You should also designate "System setting" at the intelligent function module.
Refer to the following manual when executing an interrupt program by the interrupt of the intelligent function module.

☞ Manual of the intelligent function module

Remark

Refer to Section 10.11 for the available interrupt pointer Nos. for an interrupt from the intelligent function module.

Basic Note6.79

When using an interrupt from the intelligent function module on the Basic model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.1)

Basic
Note6.80

6.24 Serial Communication Function

High Performance
Note6.81
Process
Note6.81
Redundant
Note6.81
Universal
Note6.81

(1) Definition

The serial communication function is designed to make communication in the MC protocol [*1] by connecting the RS-232 interface of the CPU module and personal computer, display device or the like by an RS-232 cable.
The serial communication can be executed by the Q00CPU or Q01CPU only.
(The other CPU modules do not support the serial communication function.)
The following explains the specifications, functions and various settings needed to make communication with a personal computer, display device or the like using the serial communication function.

[*1] : The MC protocol is the abbreviation of the MELSEC communication protocol.
The MELSEC communication protocol is a name of the communication method to make access from the mating equipment to the CPU module in accordance with the communication procedure of the Q series PLC (such as serial communication modules, Ethernet modules)
Refer to the following manual for the MC protocol.
Q Corresponding MELSEC Communication Protocol Reference Manual

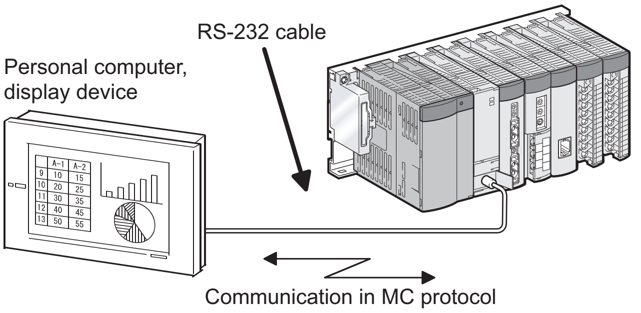


Figure 6.106 Communication with personal computer or display

☒POINT

1. The CPU that can make communication with a personal computer, display device or the like using the serial communication function is only the CPU module that is connected with the personal computer, display device or the like.
Communication cannot be made with the other station of MELSECNET/H, Ethernet or CC-Link via the CPU module that is connected with the personal computer, display device or the like.
2. The serial communication function is not used for connection of GX Developer or GX Configurator and CPU module.

Basic
Note6.80

The Q00JCPU does not support the serial communication function.

High Performance Process Redundant
Note6.81 Note6.81 Note6.81
Universal
Note6.81

The High Performance model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the serial communication function.

(2) Specifications

(a) Transmission specifications

Table6.51 indicates the transmission specifications of RS-232 used for the serial communication function of the CPU module.

Use the serial communication function after making sure that the specifications of the personal computer, Display device or the like match those of Table6.51.

Table6.51 Transmission specifications of serial communication function

Item	Default	Setting Range
Communication system	Full-duplex communication	------
Synchronization system	Asynchronous system	------
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 judgment)	Format 4 (ASCII) Format 5 (binary)	------
Frame*2	QnA-compatible 3C frame QnA-compatible 4C frame	------
Transmission control	DTR/DSR control	------
Sumcheck*1	Yes	Yes, No
Transmission wait time*1	No wait	No wait, 10ms to 150ms (10ms increments)
Online change setting*1	Not enabled	Enabled, Not enabled
Extension distance	15m	------

* 1 : Can be set in the PLC parameter setting of GX Developer.
* 2 : The relationships between the MC protocol formats and frames are indicated in Table6.52.

Table6.52 Relation between MC protocol formats and frames

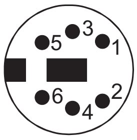
Function		Format 4	Format 5
Communication in ASCII code	QnA-compatible 3C frame	○	×
	QnA-compatible 4C frame	○	×
Communication in binary code	QnA-compatible 4C frame	×	○

○ : Usable, × : Unusable

(b) RS-232 connector specifications

Table6.53 indicates the applications of the RS-232 connector of the CPU module.

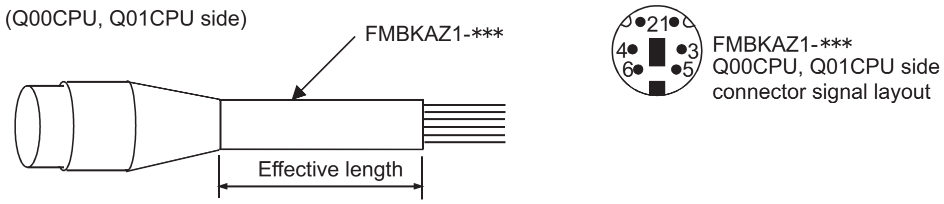
Table6.53 RS-232 connector specifications

Appearance	Pin No.	Signal Symbol	Signal Name
 Mini-Din 6 pins (female)	1	RD (RXD)	Receive data
	2	SD (TXD)	Send data
	3	SG	Signal ground
	4	------	------
	5	DSR (DR)	Data set ready
	6	DTR (ER)	Data terminal ready

(c) RS-232 cable

The following RS-232 cable can be used for connection of the CPU module with the personal computer, display device or the like.

- QC30R2 (cable length: 3m)
- FMBKAZ1-***(KURAMO ELECTRIC) manufactured
  Cable with a mini-DIN connector on one side and without connector on the other side
  ***indicates the cable length, which can be specified up to 15ms
  in 0.1m increments



Pin No.	1	2	3	4	5	6	Metal shell
Signal name	RD	SD	SG	--	DR	ER	
Wire core	Red	Black	Green White	--	Yellow	Brown	Shield

Figure 6.107 RS-232 cable effective length and signal layout

(3) Functions

The serial communication function allows the MC protocol commands in Table6.54 to be executed.

Refer to the following manual for details of the MC protocol.

☞Q-Compatible MELSEC Communication Protocol Reference Manual

Table6.54 List of MC protocol commands supported by serial communication function

Function			Command	Processing	Processing Points
Device memory	Batch read	in bits	0401(00□1)	Reads bit devices by 1 point.	ASCII: 3584 points BIN: 7168 points
		in words	0401(00□0)	Reads bit devices by 16 points.	480 words (7680 points)
				Reads word devices by 1 point.	480 points
	Batch write*1	in bits	1401(00□1)	Writes to bit devices by 1 point.	ASCII: 3584 points BIN: 7168 points
		in words	1401(00□0)	Writes to bit devices by 16 points	4480 words (7680 points)
				Writes to word devices by 1 point.	480 words
	Random read	in words	0403(00□0)	Reads bit devices by 16 points or 32 points by designating the devices at random.	96 points
				Reads word devices by 1 point or 2 points by designating the devices at random.	
	Test*1 (Random write)	in bits	1402(00□1)	Sets/resets bit devices by 1 point by designating the devices at random.	94 points
		in words	1402(00□0)	Sets/resets bit devices by 16 points or 32 points by designating the units at random.	*2
				Writes to word devices by 1 point or 2 points by designating the devices at random.	
	Monitor registration	in words	0801(00□0)	Registers the bit devices to be monitored by 16 points or 32 points.	96 points
				Registers the word devices to be monitored by 1 point or 2 points.	96 points
	Monitor	in words	0802(00□0)	Monitors the devices registered for monitoring.	Number of points registered for monitor

* 1 : When performing online change of the CPU module, set online change setting to "Enable".

* 2 : Set the number of processing points within the range of the following expression.

(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

Table6.55 Devices that can be accessed by serial communication function

Class	Device		Device Code	Device Number Range [1] (Default Value)		Write	Read
Internal system device	Function input		FX[2]	000000 to 00000F	Hexadecimal	×	×
	Function output		FY[2]	000000 to 00000F	Hexadecimal		
	Function register		FD	000000 to 000004	Decimal		
	Special relay		SM	000000 to 001023	Decimal	○	○
	Special register		SD	000000 to 001023	Decimal		
Internal user device	Input		X	000000 to 0007FF	Hexadecimal		
	Output		Y	000000 to 0007FF	Hexadecimal		
	Internal relay		M	000000 to 008191	Decimal		
	Latch relay		L	000000 to 002047	Decimal		
	Annunciator		F	000000 to 001023	Decimal		
	Edge relay		V	000000 to 001023	Decimal		
	Link relay		B	000000 to 0007FF	Hexadecimal		
	Data register		D	000000 to 011135	Decimal		
	Link register		W	000000 to 0007FF	Hexadecimal		
	Timer	Contact	TS	000000 to 000511	Decimal		
		Coil	TC				
		Current value	TN				
	Retentive timer	Contact	SS	------	Decimal		
		Coil	SC				
		Current value	SN				
	Counter	Contact	CS	000000 to 000511	Decimal		
		Coil	CC				
		Current value	CN				
	Link special relay		SB	000000 to 0003FF	Hexadecimal		
	Link special register		SW	000000 to 0003FF	Hexadecimal		
	Step relay		S	000000 to 002047	Decimal	×	
	Direct input		DX	000000 to 0007FF	Hexadecimal	○	
	Direct output		DY	000000 to 0007FF	Hexadecimal		
	Index register		Z	000000 to 000009	Decimal		
	File register		R	000000 to 032767	Decimal		
			ZR	000000 to 00FFFF	Hexadecimal		

○ : Read/write enabled, × : Write disabled

[1] : After changing the number of CPU module device points on GX Developer, use the devices in the new device No. range.
Decimal and hexadecimal indicate that the device specified for the MC protocol command is decimal or hexadecimal.

[2] : Hexadecimal when specified for the command of the MC protocol.

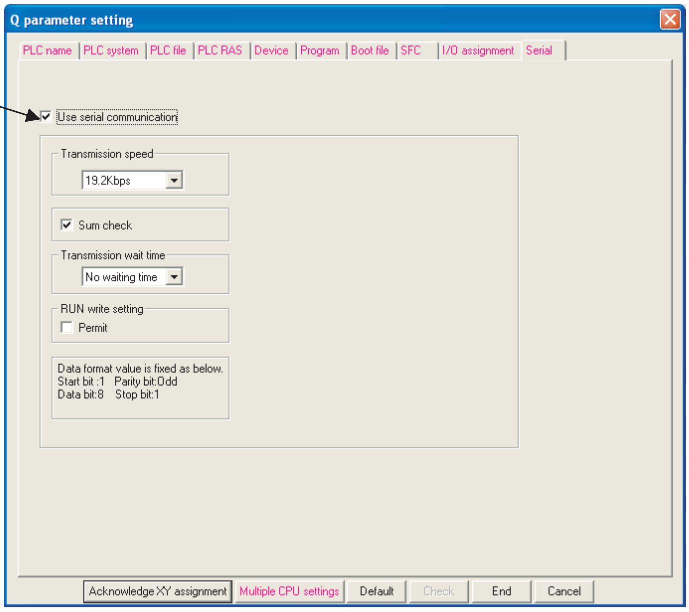
(5) Setting of transmission specifications

Use the serial communication setting PLC parameters to set the transmission speed, sum check, transmission wait time and online change setting of the serial communication function.

- When using the serial communication function to make communication with the personal computer, Display device or the like, specify "Use serial communication".
- Set the transmission speed, sum check, transmission wait time and RUN write setting.

Click here to use the serial communication function.

Selecting "Use serial communication" allows you to change the 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 6.108 Serial communication setting screen

(6) Instructions

(a) When switching the connection target to GX Developer during communication with display device or like

Connection can be switched to GX Developer during communication with the personal computer, Display device or the like using the serial communication function.
However, the personal computer, Display device or the like that was making communication using the serial communication function results in a communication error.
Refer to the manual of the used device for the way to start the personal computer, Display device or the like when the CPU module is reconnected with the personal computer, Display device or the like.

(b) Transmission speed set on Connection setup screen

When "Use serial communication" is selected, the transmission speed set on the Connection setup screen of GX Developer is ignored.

POINT

The data set in serial communication setting is made valid when:

- The PLC is powered on; or
- The CPU module is reset.

(7) Error codes for communication made using serial communication function

Table6.56 indicates the error codes, error definitions and corrective actions that are sent from the CPU module to the external device when errors occur during communication made using the serial communication function.

Table6.56 List of error codes sent from CPU module to external device

Error Code (Hexadecimal)	Error Item	Error Definition	Corrective Action
4000H to 4FFFH	----	• CPU detected error (Error that occurred in other than the serial communication function)	• Refer to the Appendices of the QCPU User's Manual (Hardware Design, Maintenance and Inspection), and take corrective action.
7153H	Frame length error	• The length of the received message is outside the permissible range.	• Reconsider the sent message. • The number of access points of the message should be within the permissible range.
7155H	Unregistered monitor error	• A monitor request was given before monitor registration was made.	• Give a monitor request after registering the device to be monitored.
7164H	Requested data error	• The requested data or device specifying method is in error.	• Check and correct the sent message/requested data of the device on the other end, and restart communication.
7167H	Disabled during RUN	• A write command was specified for the setting of online change disable.	• Change the setting to online change enable and restart communication.
7168H		• The command specified cannot be executed during RUN.	• Set the CPU module to STOP and restart communication.
716DH	Monitor registration error	• The QnA-compatible 3C/4C frame was not used for monitor registration.	• Perform monitor registration again.
7E40H	Command error	• The command or sub-command specified does not exist.	• Check and correct the sent message of the device on the other end and restart communication.
7E41H	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 device on the other end and restart communication.
7E42H	Data count error	• The requested number of points exceeds the range of the command.	• Check and correct the sent message of the device on the other end and restart communication.
7E43H	Device error	• The device specified does not exist. • The device specified cannot be specified for the corresponding command.	• Check and correct the sent message of the device on the other end and restart communication.
7E47H	Continuous request error	• The next request was received before the reply message was returned.	• Do not give continuous requests from the device on the other end. • Match the monitoring time of timer 1 with the time-out period of the device on the other end.
7E4FH	Device point count error	• The number of access points is incorrect.	• Check and correct the sent message of the device on the other end and restart communication.
7E5FH	Request destination module I/O number error	• The request destination module I/O number is in error.	• Correct the module I/O number of the data send destination.
7E64H	Registered point count range error	• The number of registered points (word/bit) is outside the range.	• Correct the set value of the registered points (word/bit).
7F01H	Buffer full error	• The next data was received before completion of received data processing.	• Perform handshake with the device on the other end, for example, to increase the sending intervals.
7F21H	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 device on the other end and restart communication.

(Continued on next page)

Table6.56 List of error codes sent from CPU module to external device (Continued)

Error Code (Hexadecimal)	Error Item	Error Definition	Corrective Action
7F22$_H$	Command error	• The command or device specified does not exist. • The remote password length is in error.	• Check and correct the sent message of the device on the other end and restart communication.
7F23$_H$	MC protocol message error	• The data (e.g. ETX, CR-LF) specified after the character part does not exist or in error.	• Check and correct the sent message of the device on the other end and restart communication.
7F24$_H$	Sum check error	• The calculated sum check does not match the received sum check.	• Reconsider the sum check of the device on the other end.
7F67$_H$	Overrun error	• The next data was received before the CPU module completed receive processing.	• Reduce the communication speed and restart communication. • Check the CPU module for occurrence of an instantaneous power failure. (For the CPU module, use the special register SD53 to check.) When an instantaneous power failure has occurred, remove its cause.
7F69$_H$	Parity error	• The parity bit setting does not match.	• Match the setting of the CPU module with that of the device on the other end.
7F6A$_H$	Buffer full error	• The receive buffer of the OS overflew, resulting in skipped receive data.	• Exercise DTR control to make communication, preventing a buffer full error.
F□□□$_H$	------	• Error detected by the MELSECNET/H network system.	• Check the send message of the device on the other end, make corrections and retry. (The station number may have been specified. Communication with stations on other than MELSECNET/H or Ethernet is not available.)

6.25 Service Processing

Universal
Note6.82

6.25.1 Module service interval time read

The module service interval indicates the time between a transient request such as monitor, test, program write/read.

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 at which access to the CPU occurs from outside.

(1) Reading method

To read the module service interval time, operate the following special relay and special registers shown in Table6.57 and Table6.58.

Table6.57 Special relay

Number	Name	Description
SM551	Module service interval time read	Turning this relay from OFF to ON reads to SD551 and SD552 the module service interval time of the intelligent function module specified in the special register SD550. ON: Read OFF: No processing

Table6.58 Special registers

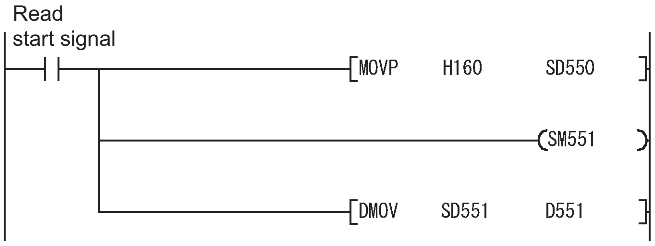
Number	Name	Description
SD550	Module service interval time measured module	Set the I/O number of the module whose module service interval time will be measured. Set the I/O number of the peripheral device connected to the RS-232 or USB interface of the CPU module to FFFFH.
SD551, SD552	Module service interval time	Stores the service interval time from the module specified in SD550 when SM551 is turned on. SD551: 1ms units (range 0 to 65535) SD552: 100μs units (range 0 to 900, stored at intervals of 100μs) (Example) When module service interval time is 123.4ms SD551 = 123, SD552 = 400

Universal
Note6.82

The Universal model QCPU cannot read module interval time.

(2) **(Program example)**

The following program example reads the module service interval of the intelligent function module at X/Y160 (Figure 6.109).



Sets I/O number 160 (hexadecimal) to SD550.

Starts module service interval time read.

Stores module service interval time into D551, D552.

Figure 6.109 Module service interval reading program example

☒**POINT**

1. The access interval in cyclic communication from the network module is not stored.
2. To read the service interval time when access is made from GX Developer of the other station on the network, set the I/O number of the network module.

Basic
Note6.83
High Performance
Note6.83
Process
Note6.83
Redundant
Note6.83

6.25.2 Service processing

(1) Service processing setting overview

The service processing setting specifies the number or time of service processing executed in END processing as desired.

The service processing setting function can reduce increased scan time brought by the improvement of communication response with the peripheral devices and service processing. This enables to configure the service processing environment optimum for the system.

☒**POINT**

Service processing designates the communication service processing with peripheral devices (such as GX Developer) and intelligent function modules. However, link refresh processing such as MELSECNET/H network module or CC-Link network module is not included.

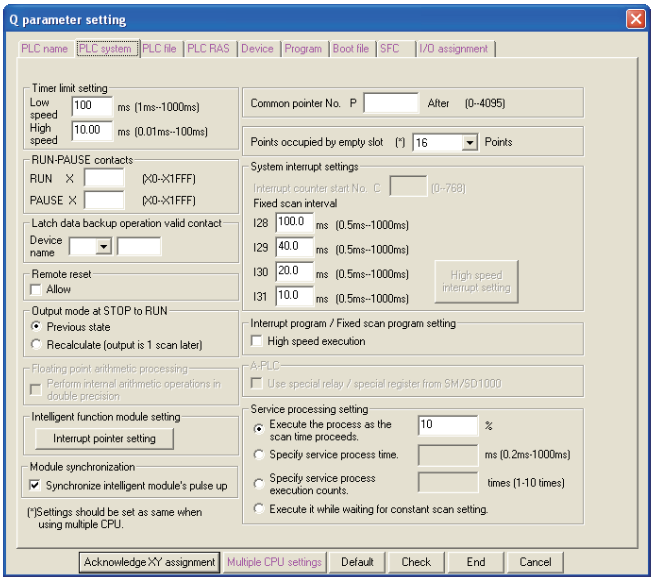
Using the COM instruction enables to perform service processing even during a midway through program execution same as the one performed at END processing. Therefore, the high-speed service processing response can be executed even if the scan time is long.

Basic High Performance Process
Note6.83 Note6.83 Note6.83
Redundant
Note6.83

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU cannot use service processing settings.

(2) Parameter setting

The service processing can be set in the PLC system tab of the PLC parameter dialog box.


Q parameter setting
PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment
Interrupt pointer setting
High speed interrupt setting
Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.00 ms (0.01ms--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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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. 10 %
○ 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.

Figure 6.110 Parameter setting screen

The service processing can be executed by selecting one of the parameter items in Table6.59. Setting values for non-selected parameter items cannot be entered. (Default: Execute the process as the scan time proceeds. =10%)

Table6.59 Parameter item list

Item name	Description	Setting range	Remarks
Execute the process as the scan time proceeds.	Specifies the percentage of service processing for one scan.	• Range: 1 to 99% • Unit: 1%	Default at selection =10%
Specify service process time.	Specifies the time of service processing for one scan.	• Range: 0.2ms to 1000ms • Unit: 0.1ms	Default at selection =0.2ms
Specify service process execution counts.	Specifies the count of service processing for one scan.	• Range: 1 to 10 times • Unit: 1 time	Default at selection =1 time
Execute it while waiting for constant scan setting.	Specifies whether to execute the service processing while waiting for constant scan setting.	------	Even when the waiting time is 0.2ms or less, the service processing time (0.2ms) will be added to the scan time at service processing execution.

(3) Operations for each service processing setting

Operation details for each service processing setting is described below.

(a) Operation details when "Execute the process as the scan time proceeds." is selected

1) Operation when "10%" is set

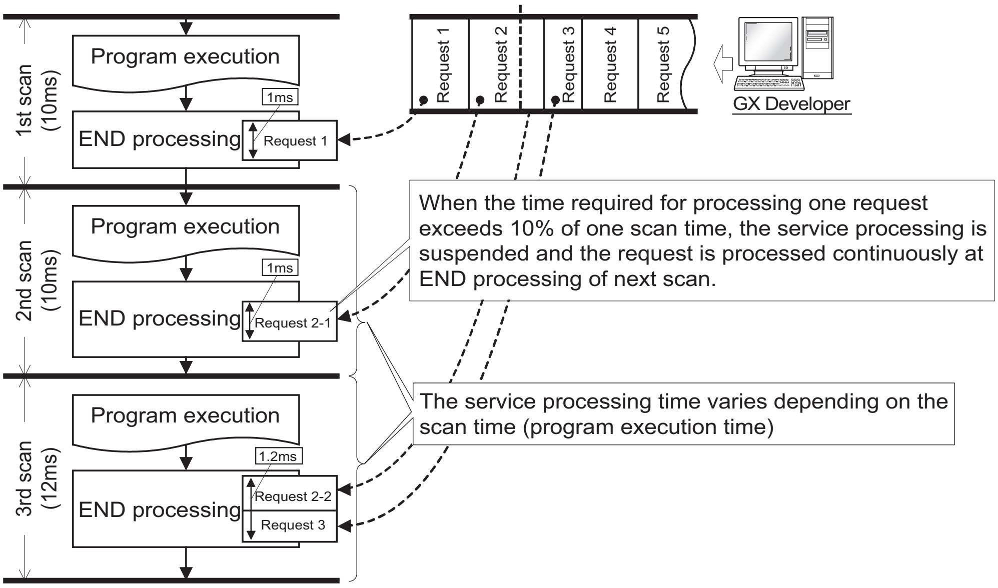

1st scan (10ms)
2nd scan (10ms)
3rd scan (12ms)
Program execution
END processing
1ms
1.2ms
Request 1
Request 2-1
Request 2-2
Request 3
Request 1
Request 2
Request 3
Request 4
Request 5
GX Developer
When the time required for processing one request exceeds 10% of one scan time, the service processing is suspended and the request is processed continuously at END processing of next scan.
The service processing time varies depending on the scan time (program execution time)

Figure 6.111 Parameter setting screen

☒POINT

If no request data for service processing exists, END processing speeds up by the request processing time. (CPU module does not wait for requests.)

2) Operation at the time of setting constant scan

The calculation of the service processing time is a calculation of the percentage toward the time excluding the wait time of the constant scan from the scan time, not a calculation of the percentage toward the scan time.

[Example]

When setting Percentage of scan time = “50%”

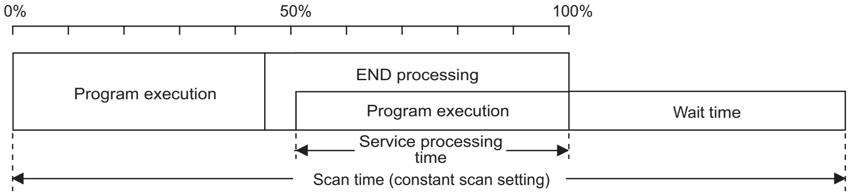


Figure 6.112 Operation at the time of setting constant scan

☒POINT

When setting the constant scan, select [Execute it while waiting for constant scan setting.] to execute the service processing efficiently. (☞ Section 6.25.2(3)(d))

(b) Operation details when "Specify service process execution counts." is selected

1) Operation when "1 time" is set

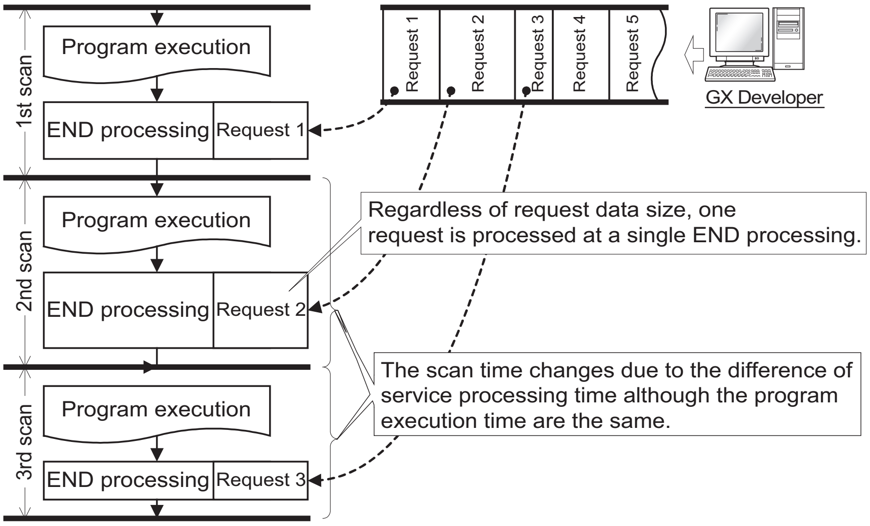

1st scan
2nd scan
3rd scan
Program execution
END processing
Request 1
Request 2
Request 3
Request 4
Request 5
GX Developer
Regardless of request data size, one request is processed at a single END processing.
The scan time changes due to the difference of service processing time although the program execution time are the same.

Figure 6.113 Operation when "1 time" is set

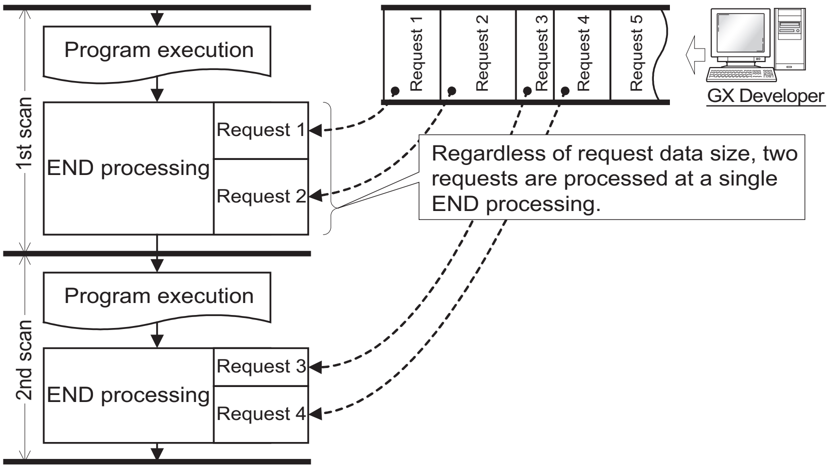


Figure 6.114 Operation when "2 times" is set

☒POINT

1. When several devices are connected to one CPU module, each device requests service processing.
   When requests from several devices are received together, a single END processing can accepts several requests simultaneously if the service processing count is set to the number of connected devices. This improves response performance. (Note, however, that the scan time increases by the service processing time.)

2. If no request data exists when setting service processing count, END processing speeds up by the request processing time. (CPU module does not wait for requests.)

(c) Operation details when "Specify service process time." is selected

1) Operation when "0.5ms" is set

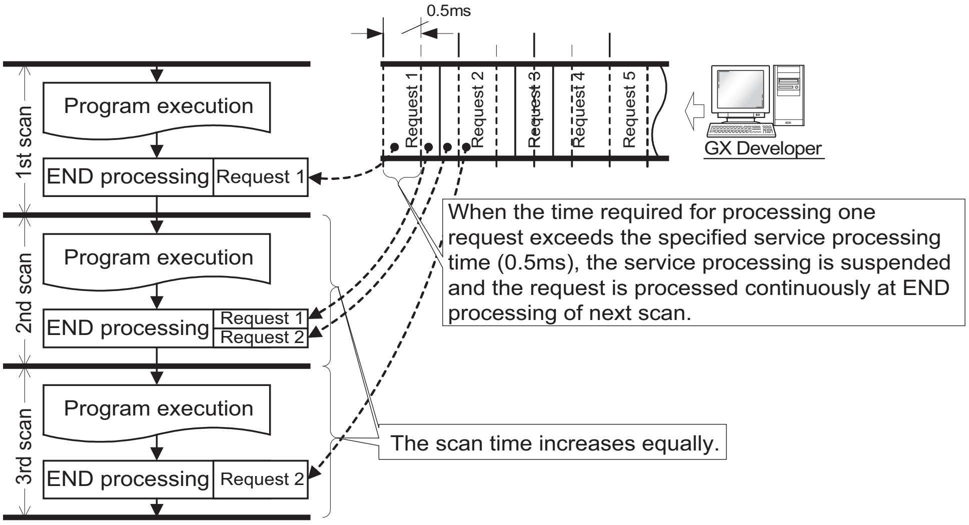

0.5ms
1st scan
2nd scan
3rd scan
Program execution
END processing
Request 1
Request 2
Request 3
Request 4
Request 5
GX Developer
When the time required for processing one request exceeds the specified service processing time (0.5ms), the service processing is suspended and the request is processed continuously at END processing of next scan.
The scan time increases equally.

Figure 6.115 Operation when "0.5ms" is set

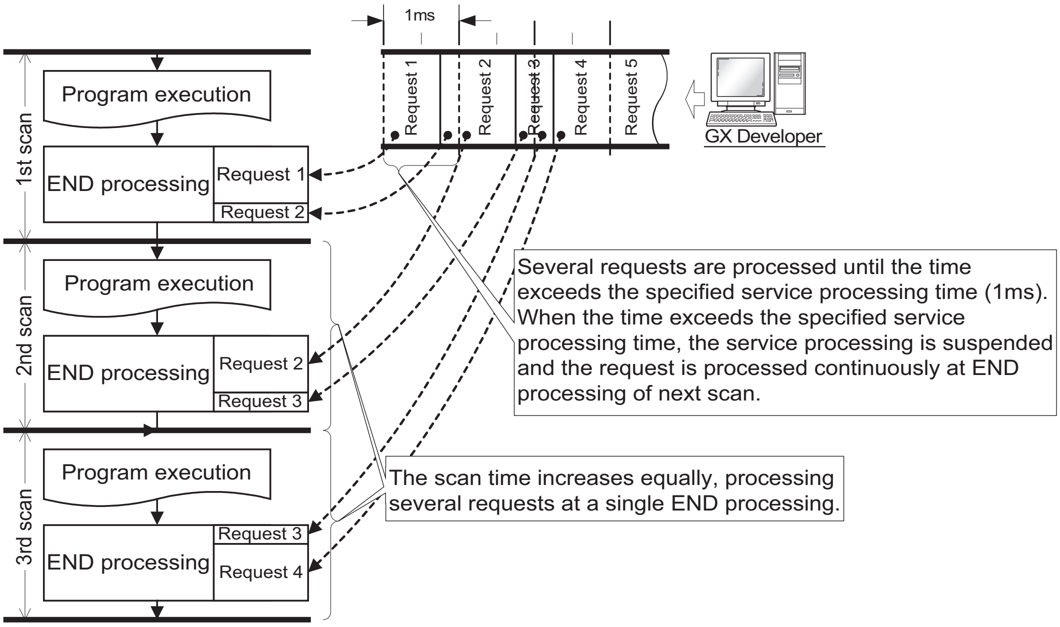

1ms
1st scan
2nd scan
3rd scan
Program execution
END processing
Request 1
Request 2
Request 3
Request 4
Request 5
GX Developer
Several requests are processed until the time exceeds the specified service processing time (1ms). When the time exceeds the specified service processing time, the service processing is suspended and the request is processed continuously at END processing of next scan.
The scan time increases equally, processing several requests at a single END processing.

Figure 6.116 Operation when "1ms" is set

☒POINT

If no request data exists when setting service processing time, END processing speeds up by the request processing time. (CPU module does not wait for requests.)

(d) Operation details when "Execute it while waiting for constant scan setting." is selected

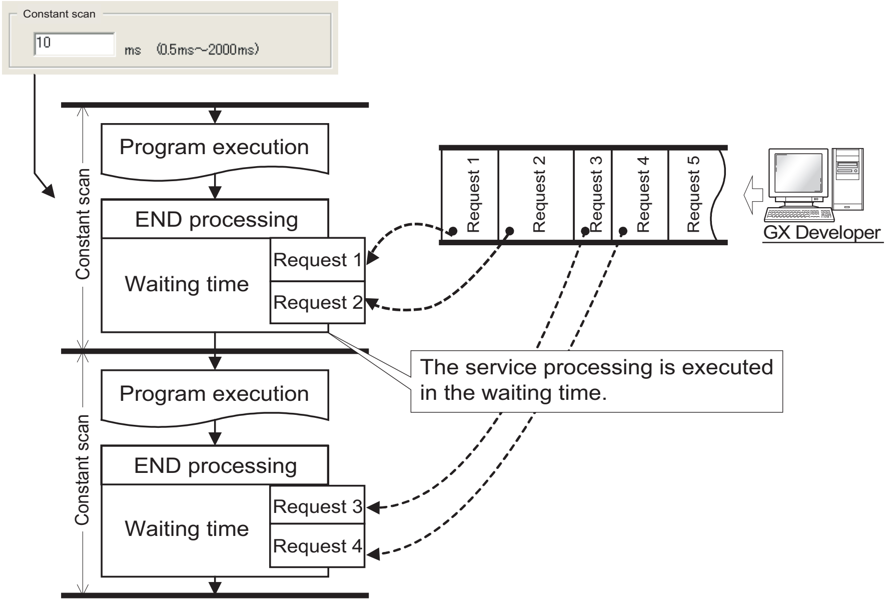
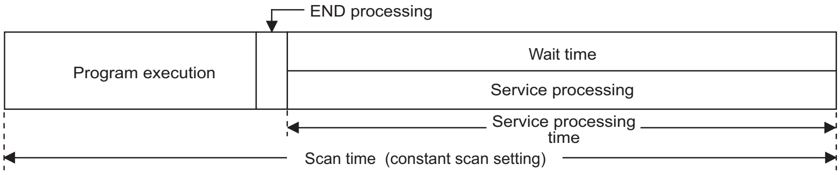


Figure 6.117 Operation when "Execute it while waiting for constant scan setting." is selected

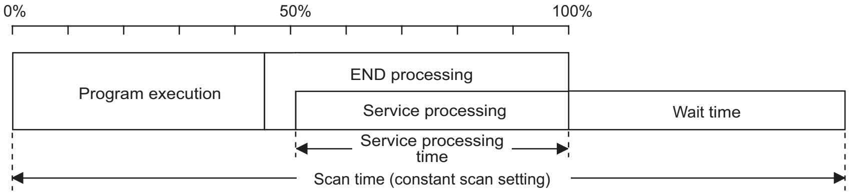
☒POINT

1. When setting the constant scan, selecting [Execute it while waiting for constant scan setting.] enables to reduce the scan time compared with performing the scan in accordance with the scan time percentage.

   - When performing the service processing at wait time of constant scan setting



   - When performing the scan in accordance with the scan time percentage



2. Even when there is no wait time, the service processing (0.2ms) is performed. Therefore, when the wait time is less than 0.2ms, the constant scan time may be exceeded.

(4) Precautions

The following describes the precautions when the service processing settings are configured.

1) On the following functions, scan time will increase longer than the specified time during service processing even if the service processing time specification is set.
   - Online change
   - Change TC setting
   - Local device monitor
   - Program memory backup
   - Writing/reading to/from a file register (The scan time will be prolonged when the write/read size is large.)
   - Buffer memory write/read of the intelligent function module (The scan time increases when the size of write/read data is large.)
   - Access to network module
     a) Diagnostics function
     (Network diagnostics, Ethernet diagnostics, CC-Link diagnostics)
     b) Monitor function (Module access device, Link direct device)

2) Take note that the scan time will increase much longer if multiple requests are received simultaneously when the service processing count specification is set many.

3) When setting the service processing time much shorter than the scan time, the response quality of service processing is extremely reduced. Set the service processing time with considering the time out time of peripheral devices.

4) An error of -20μs to +30μs occurs between the actual processing time and the set service processing time.

6.26 Device Initial Value

Basic Note6.84
Basic Note6.85
Process Note6.85
Redundant Note6.85
Universal Note6.85

(1) Definition

The device initial value is a function that registers the data used in a program to the device or intelligent function module/special function module Note6.85 buffer memory without any program.

(2) Application of device initial value

The use of device initial values provides a shortcut to specify device data in a program without using a device data setting program (initial program).

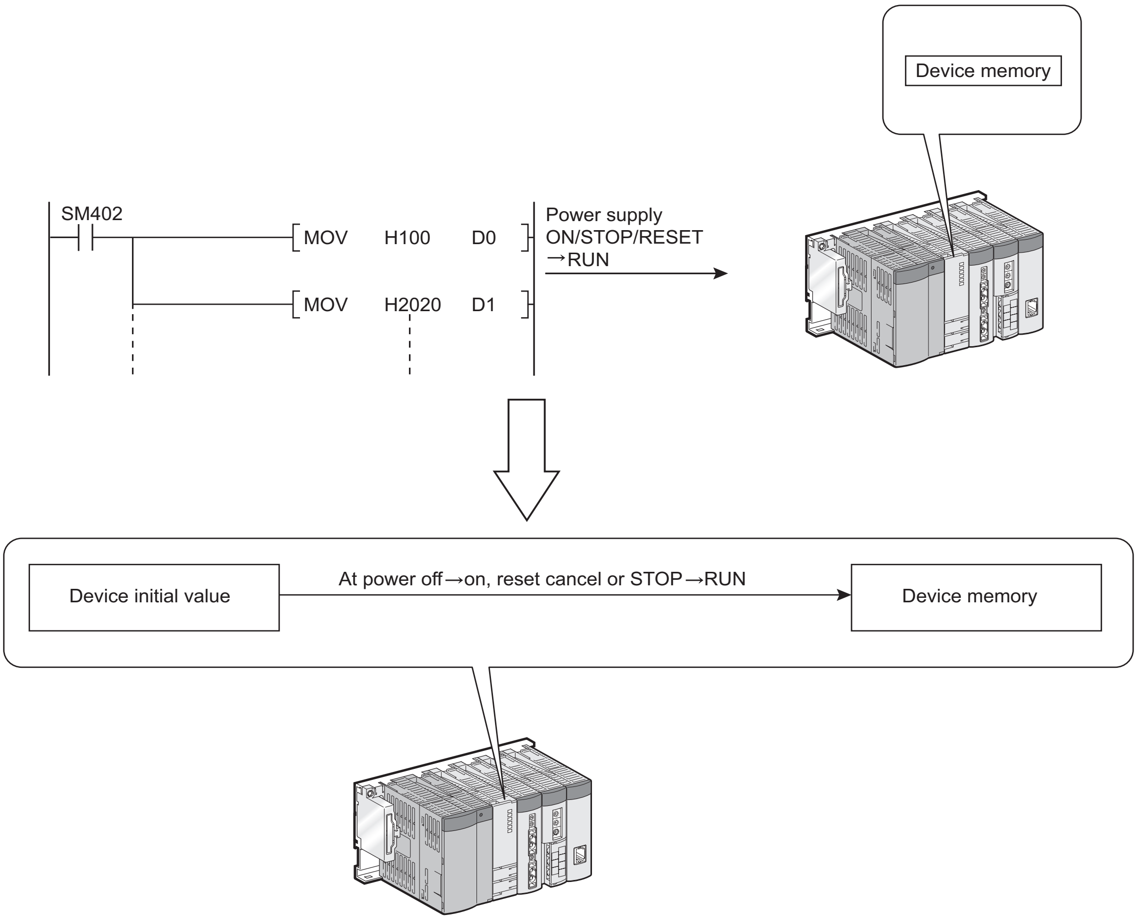


Figure 6.118 Data setting by initial program

Basic Note6.84

When using the device initial value on the Basic model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.1)

Basic Note6.85 Process Note6.85 Redundant Note6.85 Universal Note6.85

The Basic model QCPU, Process CPU, Redundant CPU, and Universal model QCPU do not support the AnS/A series corresponding modules.

(3) Timing when device initial values are written to specified devices

The CPU module writes the data of the specified device initial value file to the specified device or intelligent function module buffer memory when the PLC is powered on or the CPU module is switched from the STOP status to the RUN status.

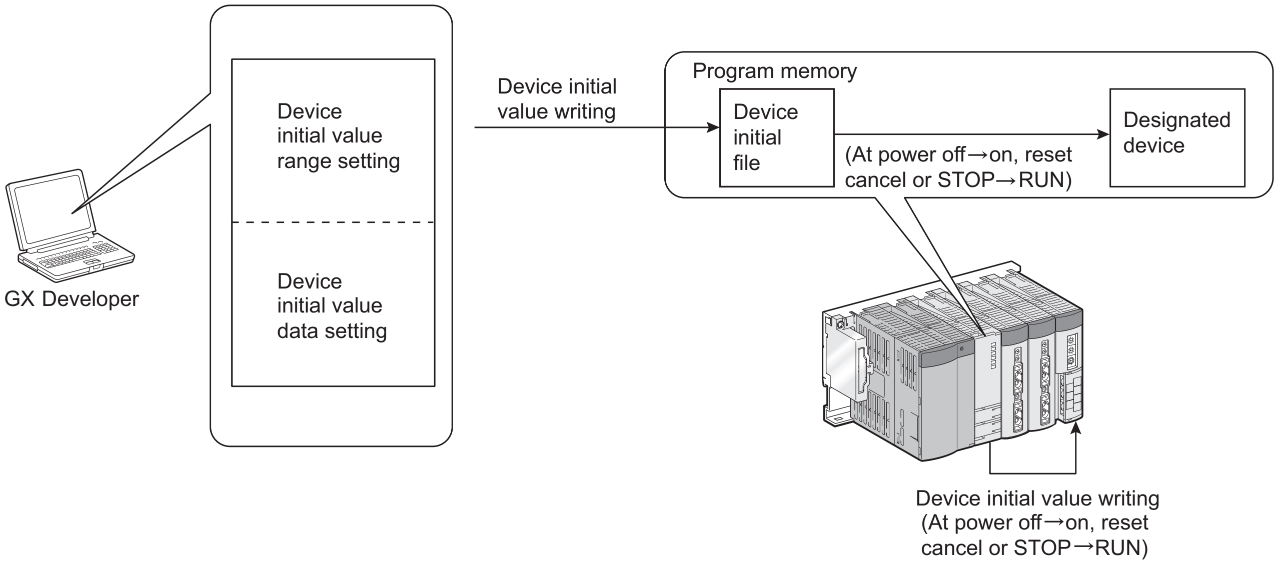


Figure 6.119 Sequence up to device initial value write

(4) Available devices[*1]

Device initial values can be used by the following devices:

- Timer present value (T)
- Retentive timer present value (ST)
- Counter present value (C)
- Data register (D)[*2]
- Special register (SD)
- Link register (W)[*3]
- Link special register (SW)
- File register (R)
- File register (ZR)
- Intelligent function module device (U□\G□)
- Link direct device (J□\W□,J□\SW□)

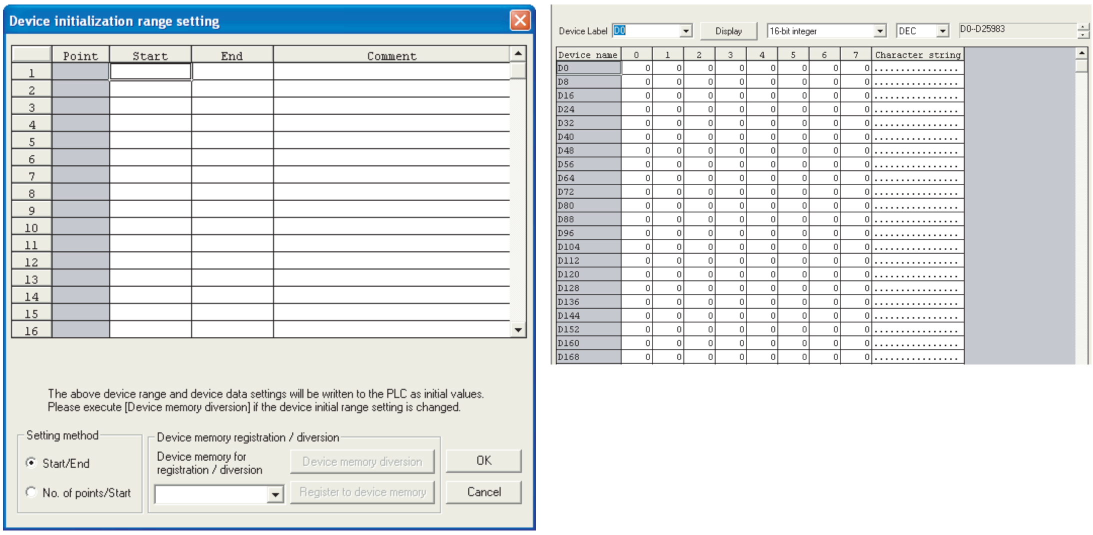
* 1 : Refer to Section 10.1 for the available ranges.
* 2 : The extended data register (D) is also included.
* 3 : The extended link register (W) is also included.

(5) Procedure and setting for use of device initial values

Basic
Note6.86

In order to use the device initial values, the device initial data must be created with GX Developer in advance, and this data must be stored as a device initial value file in the CPU module program memory, standard RAM or memory card.Note6.86

- Add the device initial value data in the project data list of GX Developer.
  As the device initial value range setting screen appears, set the device initial value range.
  The number of points that can be set is up to 8000 points per range setting.
- Add the device memory data on the project data list of GX Developer.
  As the device memory screen appears, set the device initial value data in the device initial 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 | Display | 16-bit integer | DEC | D0--D25983

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	....
D168	0	0	0	0	0	0	0	0	....

Figure 6.120 Device initialization range setting screen, device memory screen

☒POINT

When the setting on the Device initialization range setting screen is modified, be sure to execute the "Device memory registration/diversion" operation. Refer to the following manual for details of the operation.

☞ GX Developer Version 8 Operating Manual

Basic
Note6.86

The Basic model QCPU does not support the use of a memory card.

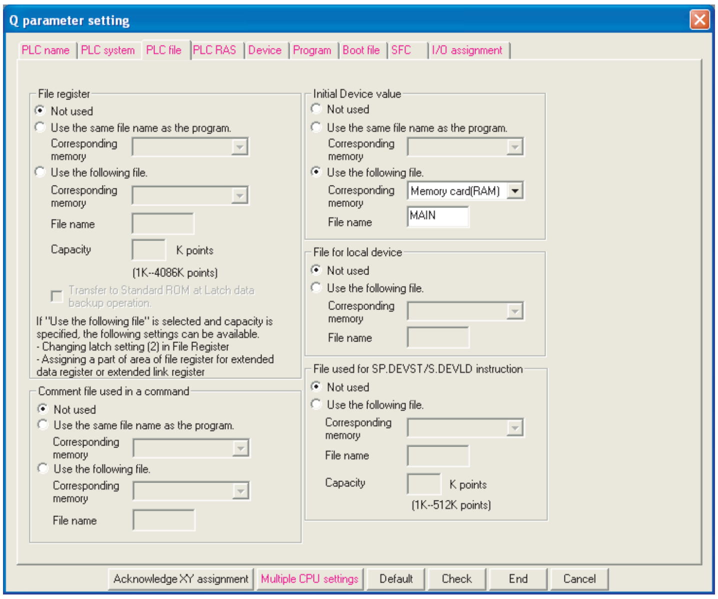
- At the "PLC file" tab screen in the "(PLC) Parameter" dialog box, designate the name of the file where the device initial value data is to be stored.

1) For Basic model QCPU

Set Device initial values to "Use" in the PLC file setting of the PLC parameter dialog box.

2) For High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

Set the file name that stores the used device initial value data in the PLC file setting of the PLC parameter dialog box.



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--4086K 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: Memory card(RAM)
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 6.121 PLC file screen

- Write the device initial value data and parameter settings to the CPU module.

(6) Precautions for the use of device initial values

(a) When device initial value data and latch range data are overlapped

In cases where both device initial value data and latch range data are overlapped, the device initial value data takes priority.

Hence, the latch range data are also changed to the device initial value data when the power supply is switched OFF and then ON.

(b) Disabling the device initial value from being set in a device area at the time of STOP → RUN

The device initial value data are reflected at the time of STOP → RUN by the device initial value function.

To disable the device initial value data from being reflected at the time of STOP → RUN in a certain device area (data that are set at power supply OFF → ON and are changed by a program), use the MOV instruction or similar one in the main routine program so that the initial values will be set to the specified devices only. Use the TO instruction to write data to the buffer memory of the intelligent function module.

(c) Devices that require module synchronization setting

When specifying the following devices in the device initial value range setting, make "Module synchronization setting" in the PLC system setting of the PLC parameter dialog box.

If the module synchronization setting is not made, the device initial 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 device initial value range setting, the device initial value data setting operation, and the operation for writing the device initial values to the CPU module, refer to the following manual.

☞ GX Developer Version 8 Operating Manual

6.27 Battery life-prolonging function

Basic Note6.87
High Performance Note6.87
Process Note6.87
Redundant Note6.87

(1) Battery life-prolonging function

The battery life-prolonging function prolongs the battery life by holding only clock data 3in the battery of the CPU module.
When the battery life-prolonging function is used, data other than clock data is all initialized when turning OFF power supply or canceling reset.

Table6.60 Parameter item list

Data held in the battery			Initializing contents
Error history			The number of error histories is initialized to zero.
Latch device (L)			Cleared to zero.
Device in the latch range			Cleared to zero.
Standard RAM			Formatted. (Cleared to zero.)
	File register assigned to the standard RAM	Set to "Use the same file name as that of the program"	A file is deleted.
		Set to "Use the following file"	A file is deleted. (Re-create a file at power ON or reset clear. (Data is cleared to zero.))

Basic Note6.87 High Performance Note6.87 Process Note6.87
Redundant Note6.87

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not support the battery life-prolonging function.

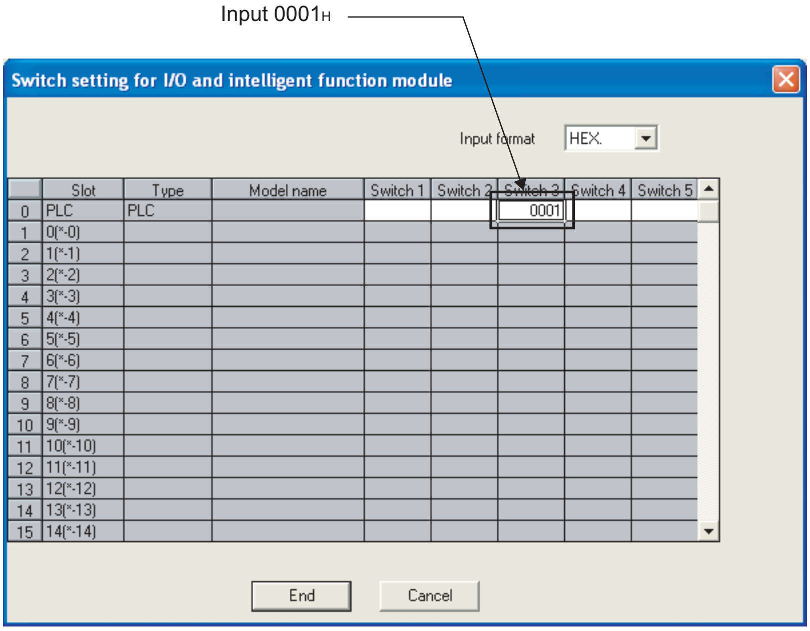
(2) Battery life-prolonging function setting

The battery life-prolonging function is set by the I/O assignment setting of the PLC parameter.

1) Perform I/O assignment setting.

2) Select the [Switch setting] button.

3) Input 0001$_H$ to the switch 3 of the CPU slot. (Even when inputting to other CPU slots, they are ignored.)



Input format: HEX.

	Slot	Type	Model name	Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
0	PLC	PLC				0001		
1	0(*-0)							
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 6.122 Switch setting screen

(3) Battery life

For the battery life of CPU module when the module uses the battery life-prolonging function, refer to the following manual.

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

6.28 Memory Check Function

Basic Note6.88
High Performance Note6.88

The memory check function checks whether the memory contents are not modified due to excessive electric noise.

Memories targeted for memory check depend on each CPU module. Table6.59 shows the target memories.

Table6.61 Parameter item list

CPU model name	Target memory	
	Program memory	Device memory
Process CPU	○	×
Redundant CPU	○	×
Universal model QCPU	○	○

○:Checked, ×:Not checked

(1) Memory check function overview for each CPU module

(a) Process CPU and Redundant CPU

The data in the executing program memory is checked whether to match the based data such as user programs and parameters at the time they was written into the program memory of the CPU module both at STOP to RUN switching and at END processing. (☞ (3) in this section.)

If the data in the executing program memory does not match the based data, the CPU determines that the data in the program memory has been modified and a stop error, RAM ERROR (error code:1106), occurs.

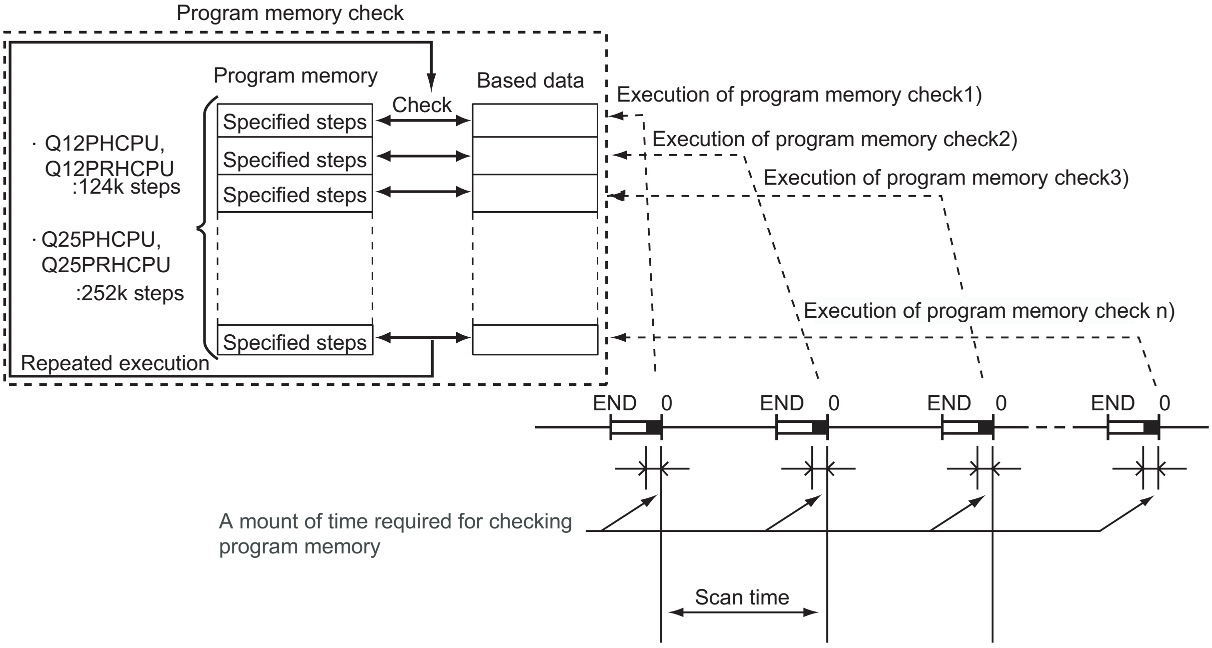

Program memory check
Program memory
Check
Based data
· Q12PHCPU, Q12PRHCPU :124k steps
· Q25PHCPU, Q25PRHCPU :252k steps
Specified steps
Specified steps
Specified steps
Specified steps
Repeated execution
Execution of program memory check1)
Execution of program memory check2)
Execution of program memory check3)
Execution of program memory check n)
END 0
END 0
END 0
END 0
A mount of time required for checking program memory
Scan time

Figure 6.123 Program memory check function overview for Process CPU and Redundant CPU

Basic Note6.88 High Performance Note6.88

The Basic model QCPU and High Performance model QCPU do not support the memory check function.

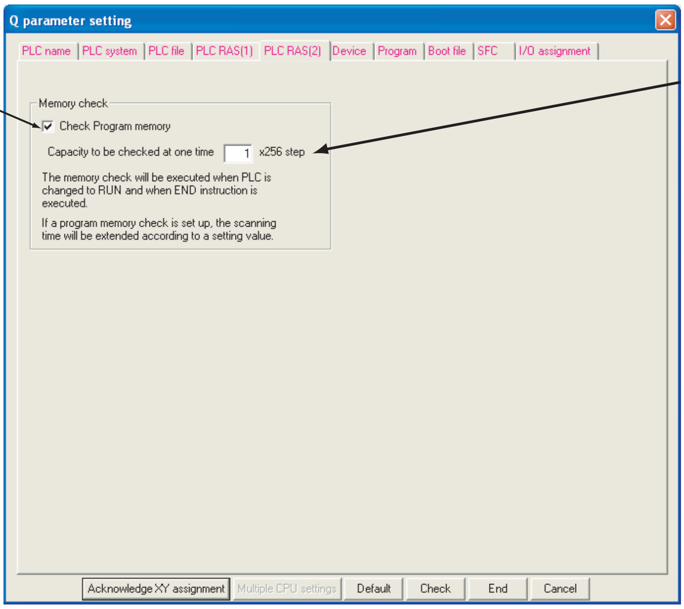
(b) Universal model QCPU

The data in the executing program memory is checked automatically whether to match the based data (user program data) at the time they were written into the program memory of the CPU module.
If the data in the executing program memory does not match the based data, a stop error, RAM ERROR (error code:1160), occurs.
As for the device memory, if the data cannot be read as devices, a stop error, RAM ERROR (error code:1161), occurs.

(2) Setting for executing memory check

(a) Process CPU and Redundant CPU

When executing memory check, check the check box for "Check Program memory" and set the "Capacity to be checked at one time" in the PLC RAS(2) tab of the PLC parameter dialog box.



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 when checking a program memory

Setting range
·Q12PHCPU,Q12PRHCPU:1～496 (256 steps to 124k steps)
·Q25PHCPU,Q25PRHCPU:1～1008 (256 steps to 252k steps)

Figure 6.124 PLC RAS(2) settings screen

Q12PRHCPU and Q25PRHCPU can check program memory by setting "Check Program memory" on the both control system CPU module and standby system CPU module.
However, on standby system CPU module, which is the back up mode for the other one, the program memory must be checked by the fixed "1(256 steps)" regardless of setting for a capacity to be checked at one time.

(b) Universal model QCPU

No setting is required neither for program memory check nor device memory check.

(3) Program memory check execution

(a) Process CPU and Redundant CPU

Memory check is executed both at "STOP to RUN switching" and "END processing" during RUN.

- At STOP to RUN switching:Checks all area of the program memory.
- At END processing :Checks by the capacity set at "Capacity to be checked at one time × 256 step" in the PLC RAS(2) tab of the PLC parameter dialog box.

(b) Universal model QCPU

The program memory, device memory, and parameters are checked automatically at the timing below.

1) Program memory

a) Programs: At program execution

b) Parameters:

- At power OFF to ON
- At reset clear
- At STOP to RUN after Write to PLC

(4) Device memory: At device readProcessing time of memory check

(a) Process CPU and Redundant CPU

1) At STOP to RUN switching

When memory check is executed, STOP to RUN switching delays by the processing time of memory check shown below.

- Q12PHCPU, Q12PRHCPU:434 ms
- Q25PHCPU, Q25PRHCPU:882 ms

2) At END processing during RUN

If a program memory is checked, the scan time takes longer.
In "PLC RAS(2)" tab in Qn(H) Parameter the dialog box, set the capacity to be checked at one time in consideration of the increase in scan time.

Calculate increases in scan time by the following expression.

$$\text{(Increases in scan time)} = 3.5 \ \text{X} \ \frac{\text{(Capacities to be checked at one time)}^{*1} \ \text{X} \ 256}{1024} \quad \text{(ms)}$$

For example, when setting a check capacity 4 in "PLC RAS(2)" tab in Qn(H) Parameter the dialog box at one time, the processing time will be 3.5 × 4 × 256 ÷ 1024=3.5(ms).

* 1 : Indicates "Capacity to be checked at one time" set in "PLC RAS(2)" tab in the dialog box Qn(H) Parameter. (☞ (2) in this section)

(b) Universal model QCPU

It takes no processing time for memory check.

(5) Precautions

Note the following items when executing memory check on the Process CPU and Redundant CPU.

(a) Maximum delay time for detecting errors

From the start of rewriting the program memory data to detecting the rewritten data, maximum delay time will occur as the following expression.

The sequence program continues to execute before the program memory check detects errors.

There is a possibility that the program memory check detects other errors before detecting RAM ERROR or causes an unexpected operation due to erroneous program memories.

$$\text{Maximum delay time for detecting errors} = \frac{\text{(Maximum capacities of program memories)}}{\text{(Capacities to be checked at one time)}^{*3}\ \text{X}\ 256}\ \text{X}\ \text{(Scan time) (ms)}$$

For example, when setting a check capacity 4 in Q12PHCPU at one time, the maximum delay time will be (124 × 1024) ÷ (4 × 256) × 10=1240(ms).

* 2 : The maximum capacity of program memory in each CPU modules is shown below.
Q12PHCPU, Q12PRHCPU: 124k steps (124 × 1024 steps)
Q25PHCPU, Q25PRHCPU: 252k steps (252 × 1024 steps)

If a user setting system area is set at formatting a program memory, the capacity of the program memory will decrease for the set capacities of the user setting system area.

For a user setting system area and confirmation methods for a capacity of program memories, refer to Section 5.2.2.

* 3 : Indicates "Capacity to be checked at one time" set in "PLC RAS(2)" tab in Qn(H) Parameter the dialog box. (☞(2)in this section)

Redundant
Note6.89

(b) **Instructions not available at an execution of memory check**

If the following instructions are executed with setting memory check to be valid in the PLC parameter dialog box, OPERATION ERROR (error code:4105) occurs.Note6.89

- PLOADP instruction
- PUNLOADP instruction
- PSWAPP instruction

(c) **Valid condition of PLC parameter**

If executing the following operations after writing the PLC parameter in the CPU module, the executed or non-executed program memory check function will become valid.

- Turn the PLC off to on
- Reset the CPU module

(d) **Target memory**

Only a program memory is checked with a program memory check function. Device memory, standard RAM, standard ROM and memory card cannot be checked.

(e) **Program memory check when executing COM instruction**

When executing COM instruction, a program memory is not checked.

Redundant
Note6.89

No PLOADP, PUNLOADP and PSWAPP instruction in a redundant CPU.

6.29 Latch Data Backup Function (to Standard ROM)

Basic
Note6.90

High Performance
Note6.90

Process
Note6.90

Redundant
Note6.90

(1) Definition of latch data backup function (to standard ROM)

Latch data backup function (to standard ROM) is a function to hold (backup) latch data such as device data or error history without using a battery when stopping the system for a long period. This function enables to prolong the life of battery.

Remark

When executing the latch data backup function (to standard ROM), the battery life-prolonging function is enabled regardless of parameter setting of the battery life-prolonging function.
Also, the battery life-prolonging function is switched to be disabled after restoring backup data.
The battery life-prolonging function enabled/disabled status can be checked by SD119 (battery life-prolonging function enabled/disabled).
For the battery life-prolonging function, refer to Section 6.27.

(2) Data to be backuped

Data that needs to be backuped is shown in Table 6.56.

Table6.62 List of data to be backuped

Data to be backuped	Description	Remarks
Device data	• Internal user device (M, L, B, F, V, T, ST, C, D, W) • Index register (Z)/standard device register (Z) • File register (R, ZR) • Extended data register (D)*1 • Extended link register (W)*1	The data of the file register, extended data register (D), and extended link register (W) are backed up only when the file register in the standard RAM is used.
Error history	Error history information immediately before latch data backup to the standard ROM	-----
SFC program continuation start information	Information to start the SFC program continuously	-----
Trace setting (Sampling trace file)	Trace condition settings and Trace setting created by sampling trace function	Backup cannot be made in the following cases. • Trace setting is set in a memory card. • Trace setting is not registered in the CPU module.

* 1: The data of the extended data register (D) and extended link register (W) are backed up only when the Universal model QCPU whose serial number (first five digits) is "10042" or later is used.

Basic Note6.90 High Performance Note6.90 Process Note6.90

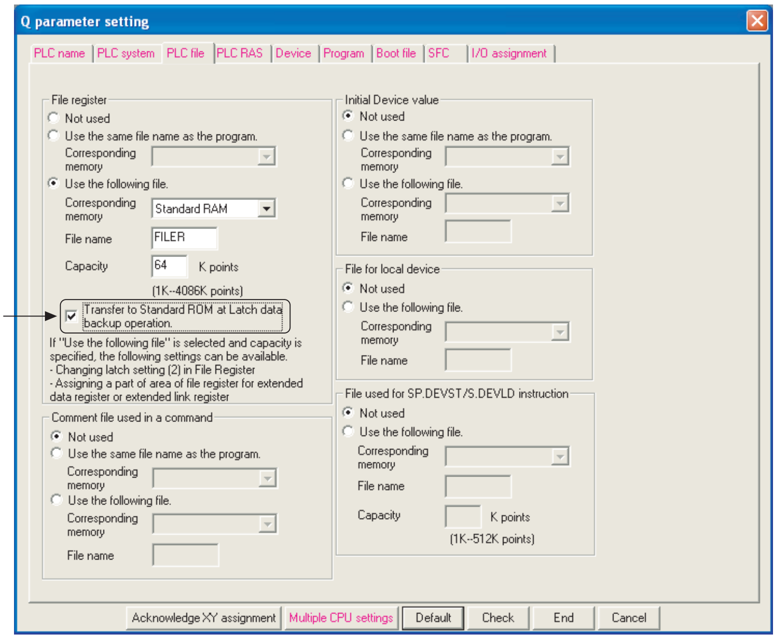
Redundant Note6.90

The Basic model QCPU, High performance model QCPU, Process CPU, and Redundant CPU do not support the latch data backup function to the standard ROM.

☒POINT

When backing up the data of the file register, extended data register (D), and extended link register (W), pay attention to the following points.

- The data is backed up only when the file register in the standard RAM is set to be used.
- Check the "Transfer to Standard ROM at Latch data backup operation." checkbox on the PLC file tab of PLC parameter.



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
● Use the following file.
Corresponding memory: Standard RAM
File name: FILER
Capacity: 64 K points
(1K--4086K 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

If checked, backup is mode

Figure 6.125 PLC file setting screen

(3) File capacity for storing data to be backuped

The system automatically creates files for storing the data that is backuped at the time of executing latch data backup to the standard ROM.

Table 6.57 shows the size of storage file used for the latch data backup function (to standard ROM).

Table6.63 List of data to be backuped

Data to be backuped		File size
Device data	• Internal user device (M, L, B, F, V, T, ST, C, D, W) • Index register (Z)/standard device register (Z)	110818 *1 (At default device assignment setting)*2
Error history		
SFC program continuation start information		
• File register (R, ZR)*3 • Extended data register (D)*3 • Extended link register (W)*3		64+2 × Number of file register points
Trace setting (Sampling trace file)*4		16 + Sampling trace file size

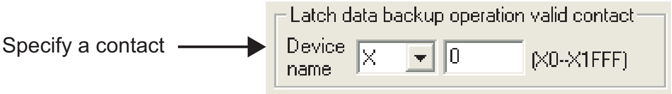
* 1 : The file size of the Q02UCPU is 87138 bytes.
* 2 : Increased or decreased depending on the parameter setting.
* 3 : Only when the data of the file register, extended data register (D), and extended link register (W) are backed up, storage files are created.
* 4 : Only when the sampling trace setting has been registered, storage files are created.

(4) Execution method

(a) Execution by contact

1) Setting method

Set the start contact of the latch data backup to standard ROM by the PLC system setting of PLC parameter. (Device applicable for contact is X, M, or B.)



Specify a contact

Latch data backup operation valid contact

Device name X 0 (X0--X1FFF)

Figure 6.126 Setting screen of start contact of latch data backup to standard ROM

2) Execution method

Backup is started by starting up contacts (OFF to ON).
When backup is completed, BAT.LED of the CPU module flashes (green), which is status where power supply is ready to be turned OFF.

3) Precautions

a) Since data to be backuped is data at the time when contact is ON (at END processing), turn ON power supply again or cancel the reset after execution of this operation.

b) The status of start contact of latch data backup to the standard ROM is checked at the END instruction execution, backup is not made even if a contact is switched ON → OFF → ON or OFF → ON → OFF in one scan processing.

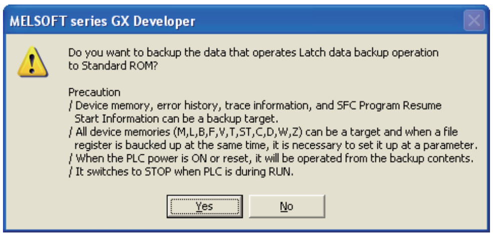
c) In the following cases, backup is not started unless start contact of the latch data backup to standard ROM is turned OFF and turned ON again.

- Start contact of the latch data backup to the standard ROM is set to X, and the power supply is turned ON from OFF without turning OFF contact or cancel the reset after executing backup by turning ON from OFF the contact.
- Start contact of the latch data backup to the standard ROM is set to M or B, and backup is performed by turning ON from OFF the contact.

(b) Execution by remote operation

1) Execution method

Backup of latch data to the standard ROM can be operated by [Online] → [Latch data backup operation] of GX Developer.
When backup is completed, BAT.LED of the CPU module flashes (green), which is status where power supply is ready to be turned OFF.



MELSOFT series GX Developer

Do you want to backup the data that operates Latch data backup operation to Standard ROM?

Precaution
/ Device memory, error history, trace information, and SFC Program Resume Start Information can be a backup target.
/ All device memories (M,L,B,F,V,T,ST,C,D,W,Z) can be a target and when a file register is baucked up at the same time, it is necessary to set it up at a parameter.
/ When the PLC power is ON or reset, it will be operated from the backup contents.
/ It switches to STOP when PLC is during RUN.

Figure 6.127 Remote operation execution screen

Data to be backuped is data at the time of remote operation.

(5) Restoring backup data

The data backuped is automatically stored by the following operations.

- At power-on (OFF to ON)
- At reset cancel

Whether to restore data after executing backup only once or to restore data per the above-mentioned operations is set by SM676 (restore repetitive execution specification) at the time of backup operation.

Table6.64 Status of SM676 and restore operation

Status of SM676 at backup operation	Restore operation
Backup operation is performed when SM676 is OFF	Data is restored only once when the power supply is turned ON from OFF or the reset is cancelled after backup operation execution.
Backup operation is performed when SM676 is ON	Data is restored per turning ON from OFF power supply or canceling the reset. Data is restored repeatedly until the backup data is deleted or the next latch data backup is performed.

When restoring backup data is completed, BAT.LED of the CPU module flashes (green) for five seconds.

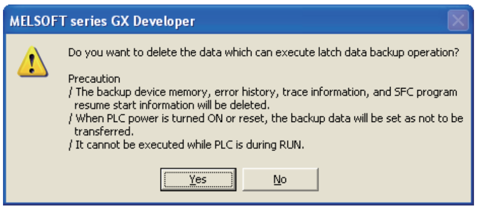
POINT

If the number of device points set by the parameter setting and the device points at the time of backup are different, RESTORE ERROR (error code: 2220) occurs when restoring the backup data, and restoring data is not completed normally. (Data will be restored when turning ON from OFF the power supply or canceling the reset again in the next time.)
Execute any one of the following operations to complete restoring data normally.

- Return data at the time of parameter is backuped.
- Delete the backup data.
- Backup data again.

(6) Deleting backup data

The backup data is deleted by [Online]→[Latch data backup operation] of GX Developer. (Set the CPU to the STOP status, since data cannot be deleted in the RUN status.)
Also, information of special registers (SD671 to SD675) can be initialized (cleared to 0) by deleting the backup data.


MELSOFT series GX Developer

Do you want to delete the data which can execute latch data backup operation?

Precaution
/ The backup device memory, error history, trace information, and SFC program resume start information will be deleted.
/ When PLC power is turned ON or reset, the backup data will be set as not to be transferred.
/ It cannot be executed while PLC is during RUN.

Figure 6.128 Screen at deleting backup data

(7) Checking with special relay and special register

The status of execution of latch data backup to the standard ROM or restore operation can be checked by SM671, SM676, SD671 to SD679.

(8) Precautions

The following shows precautions for backup of latch data.

1) Do not turn OFF power supply of the CPU module or perform reset operation during backup of latch data. If executed, RESTORE ERROR (error code: 2221) occurs, and backup data is not restored. (The backup data is deleted.)
2) Even if the backup data exists, the device initial value has a priority over the backup data when the device initial value is set.
   Therefore, the device where the device initial value setting is made is overwritten by the device data of the device initial value after reflecting the backup data.
3) Even if the latch device or latch range setting is used, backup data has a priority over it.
   Therefore, even when data of latch device and latch range setting is changed, it is overwritten by data backuped at the time of turning ON from OFF power supply and canceling the reset.
4) Devices where local device range setting is made are not backuped. They are initialized (cleared to 0) when turning power supply ON from OFF or canceling the reset.
5) When the number of writing to the standard ROM exceeds 100,000 times (FLASH ROM ERROR (error code: 1610)), backup may not be made normally.
6) Backup data cannot be deleted unless the backup data by remote operation is deleted or storage destination memory (Standard ROM) of backup data is formatted.
7) Perform the following operations after the latch data backup operation has been completed, since they cannot be performed during latch data backup.
   If executed, an error is displayed on GX Developer.
   - PLC memory format (Standard ROM only)
   - Latch data backup by remote operation
   - Write during RUN (Ladder mode, write files during RUN, and function block)
   - Write a program memory to ROM
   - Write to PLC (Flash ROM)

6.30 Write/Read Device Data to/from Standard ROM

Basic
Note6.91
High Performance
Note6.91
Process
Note6.91
Redundant
Note6.91

(1) Definition of write/read device data to/from standard ROM

Write/read the device data to/from the standard ROM is to write any device data to the standard ROM. Writing the fixed value used for operation and operation result to the standard ROM can prevent data disappearance at the time of low battery. Also, the data written to the standard ROM can be read by any timing using the instruction.

(2) Execution method

Device data is written to the standard ROM by the SP.DEVST instruction. Also, the device data written to the standard ROM is read to the specified device by the S.DEVLD instruction.

(3) Device writable to standard ROM

Table6.65 shows the devices writable to the standard ROM.

Table6.65 Device writable to standard ROM and storage area

Device name	Storage area
X, Y, M, L, B, F, V, T, ST, C, D[*1], W[*2], SM, SD, SB, SW	Standard ROM (Device data storage file)
R, ZR	

[*1] The extended data register (D) is also included.

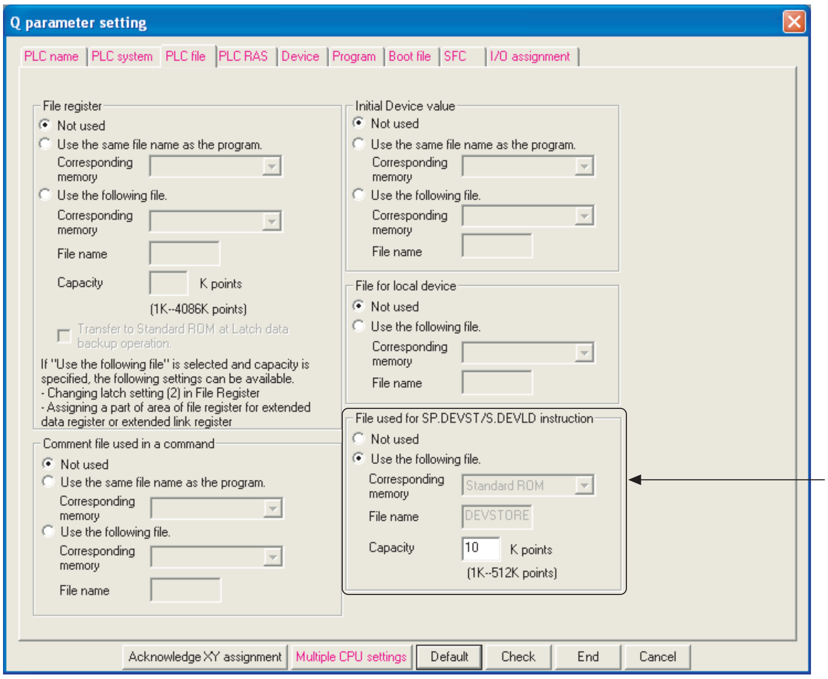
[*2] The extended link register (W) is also included.

Basic High Performance Process
Note6.91 Note6.91 Note6.91
Redundant
Note6.91

The Basic model QCPU, High Performance model QCPU, Process CPU, or Redundant CPU cannot write/read the device data to/from the standard ROM.

(4) Setting method

Set the area to store the device data to the standard ROM in the PLC file setting of the PLC parameter.



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
- ○ Use the following file.
  - Corresponding memory
  - File name
  - Capacity K points
  - (1K--4086K 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: Standard ROM
  - File name: DEVSTORE
  - Capacity: 10 K points
  - (1K--512K points)

Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

Set the file size
(file name is fixed to
DEVSTORE)

Figure 6.129 Setting screen of file for storing device data

Remark

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

☞ QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)

CHAPTER7 COMMUNICATION USING BUILT-IN ETHERNET PORTS OF CPU MODULE

Basic Note7.1
High Performance Note7.1
Process Note7.1
Redundant Note7.1
Universal UD Note7.1

The Built-in Ethernet port QCPU can communicate data by connecting built-in Ethernet ports of the CPU module with personal computers and/or display devices using an Ethernet cable (100BASE-TX, 10BASE-T).

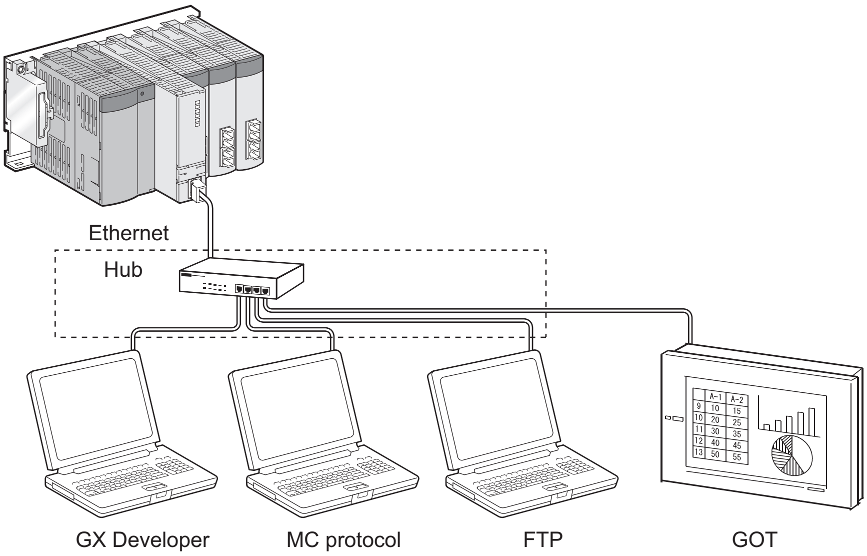


Figure 7.1 Connection example using built-in Ethernet ports of CPU module

For the comparison with the conventional MELSEC-Q series Ethernet module (QJ71E71-100), refer to Appendix 6.
The following table lists the functions using built-in Ethernet ports of the CPU module.

Table7.1 List of functions

Function	Reference
GX Developer/GOT connection	Section 7.2
GX Developer direct connection (simple connection)	Section 7.3
MC protocol communication	Section 7.4
Time setting function (SNTP client)	Section 7.5
File transfer function (FTP server)	Section 7.6
Remote password	Section 7.7

Basic Note7.1
High Performance Note7.1
Process Note7.1
Redundant Note7.1
Universal UD Note7.1

Only the Built-in Ethernet port QCPU has the built-in Ethernet ports.

7.1 Communication specifications

The following table shows the communication specifications for built-in Ethernet ports of the CPU module.

Table7.2 Ethernet communication specifications

Item			Built-in Ethernet port QCPU				
			Q03UDECPU	Q04UDEHCPU	Q06UDEHCPU	Q13UDEHCPU	Q26UDEHCPU
Transmission specifications	Data transfer speed		100/10Mbps				
	Communication mode		Full-duplex/Half-duplex				
	Transmission method		Base band				
	Maximum distance between hub and node		100m (328.08feet)				
	Maximum number of nodes/ connections	10BASE-T	Cascade connection: Four stages at maximum				
		100BASE-TX	Cascade connection: Two stages at maximum				
Number of connections	TCP/IP		16 for MELSOFT connections and MC protocols, 1 for FTP				
	UDP/IP						
Cable used[1]	10BASE-T		Ethernet-compliant cable Category 3 or more (STP/UTP cable)[2]				
	100BASE-TX		Ethernet-compliant cable Category 5 or more (STP cable)				

* 1 : Straight cables can be used.
Crossing cables can also be used for connecting the Built-in Ethernet port QCPU with GOT directly using an Ethernet cable.

* 2 : STP cables are recommended for the use under noisy environment.

Hubs with 10BASE-T or 100BASE-TX ports[3] can be used.
The maximum number of devices that can access to one CPU module simultaneously is 16.

* 3 : The ports must comply with the IEEE802.3 10BASE-T or IEEE802.3 100BASE-TX standards.

☒POINT

1. When connecting with a hub, the Built-in Ethernet port QCPU determines the cable used (10BASE-T or 100BASE-TX) and the communication mode (full-duplex or half-duplex). Set the communication mode to the half-duplex mode on the hub side when the hub that does not have the auto-negotiation function is used.
2. The module operation is not guaranteed if any of the following connection is used. Check the module operation on the user side.
   - Connections using the Internet (general public line) (Connections using Internet connection service provided by Internet service providers and telecommunications carriers)
   - Connections using devices in which a firewall is installed
   - Connections using broadband routers
   - Connections using wireless LAN

7.2 GX Developer/GOT Connection

The following describes the setting method for connecting the Built-in Ethernet port QCPU with GX Developer and/or GOT.

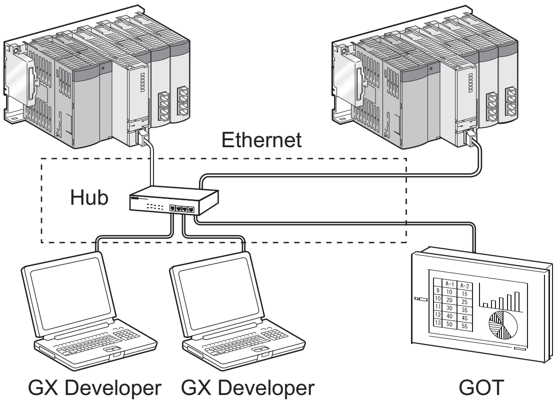


Figure 7.2 Connection example using hub

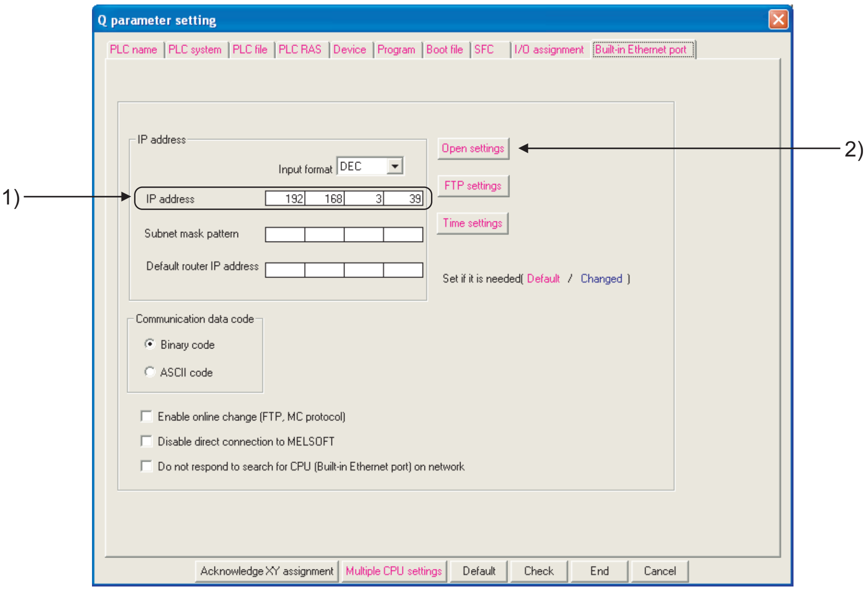
POINT

1. The Built-in Ethernet port QCPU can be connected directly with GX Developer using one Ethernet cable. (Direct connection (simple connection))
   The direct connection (simple connection) enables the QnUDE(H)CPU and GX Developer to communicate without setting IP addresses. For details, refer to Section 7.3.
2. For setting on the GOT side, refer to following manual.
   ☞ GOT1000 Series Connection Manual

(1) Setting on the Built-in Ethernet port QCPU side

Setting on the Built-in Ethernet port QCPU side is described below.

(a) Set the items on the Built-in Ethernet port tab of PLC parameter in GX Developer as shown in Figure 7.3.



IP address
Input format DEC
IP address 192 168 3 39
Subnet mask pattern
Default router IP address

Set if it is needed(Default / Changed)

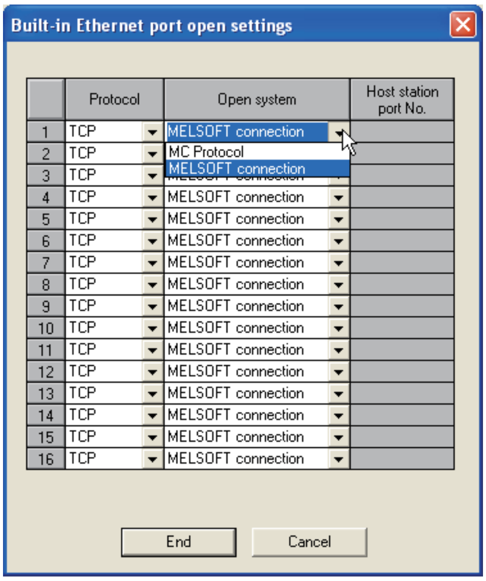
Communication data code
● Binary code
○ ASCII code

□ Enable online change (FTP, MC protocol)
□ Disable direct connection to MELSOFT
□ Do not respond to search for CPU (Built-in Ethernet port) on network

Figure 7.3 Built-in Ethernet port tab

1) **Set the CPU module IP address.**

2) **Set the connection for GX Developer (MELSOFT connection). (Refer to Figure 7.4.)**



Built-in Ethernet port open settings

	Protocol	Open system	Host station port No.
1	TCP	MELSOFT connection	
2	TCP		
3	TCP	MELSOFT connection	
4	TCP	MELSOFT connection	
5	TCP	MELSOFT connection	
6	TCP	MELSOFT connection	
7	TCP	MELSOFT connection	
8	TCP	MELSOFT connection	
9	TCP	MELSOFT connection	
10	TCP	MELSOFT connection	
11	TCP	MELSOFT connection	
12	TCP	MELSOFT connection	
13	TCP	MELSOFT connection	
14	TCP	MELSOFT connection	
15	TCP	MELSOFT connection	
16	TCP	MELSOFT connection	

Figure 7.4 Built-in Ethernet port open settings screen

- Protocol :
  Select the protocol to be used, "TCP" or "UDP" in accordance with the external device.
  Select "TCP" to emphasize communication reliability.
- Open system :
  Select "MELSOFT connection".
  (For the MELSOFT connection, refer to Section 7.4.)
- Host station port No. :
  Set the number when "MC Protocol" is selected. (☞ Section 7.4.)

(b) Using Ethernet direct connection or USB connection, write the settings in PLC parameter to the CPU module by selecting [Online] → [Write to PLC] in GX Developer. After writing the parameter settings, power the programmable controller OFF → ON or perform the reset operation of the CPU module using the RUN/STOP/RESET switch so that the parameters become valid.
For the Ethernet direct connection, refer to the following section.
(☞ Section 7.3)

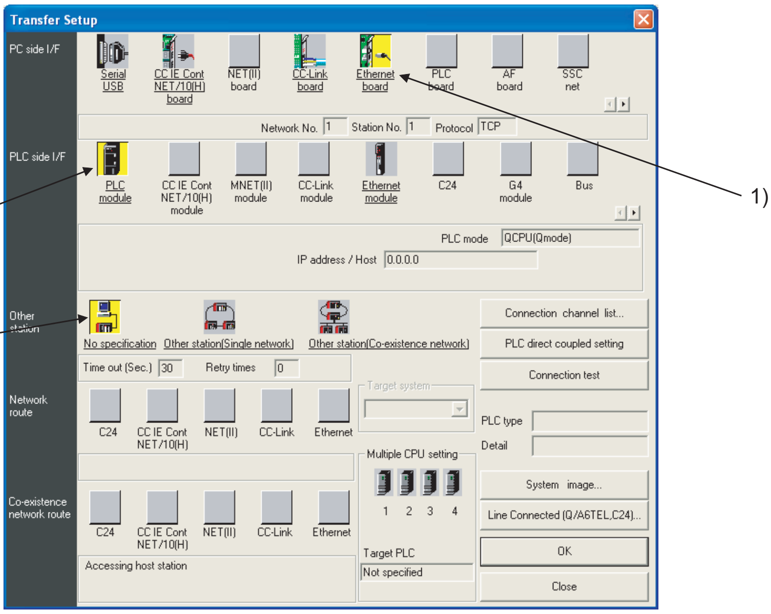
☒POINT

For the reset operation using the RUN/STOP/RESET switch, refer to the following manual.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

(2) Setting on GX Developer side

Setting on GX Developer side is described below.

(a) Set the items on the Transfer Setup screen in GX Developer as shown in Figure 7.5.



Network No. 1 Station No. 1 Protocol TCP

PLC mode QCPU(Qmode)

IP address / Host 0.0.0.0

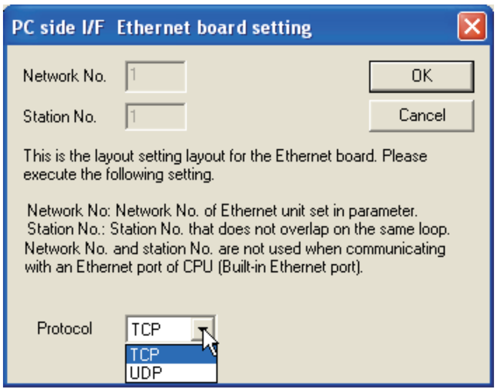
Time out (Sec.) 30 Retry times 0

Target PLC Not specified

Figure 7.5 Transfer Setup screen

1) Select "Ethernet board" for "PC side I/F".

Select the protocol to be used on the detailed setting screen as shown in Figure 7.6.



Network No. 1

Station No. 1

This is the layout setting layout for the Ethernet board. Please execute the following setting.

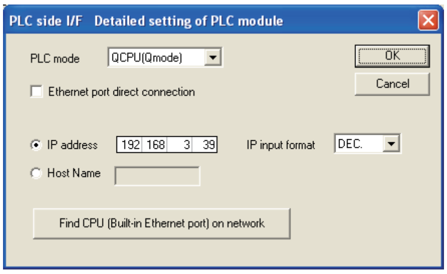
Network No: Network No. of Ethernet unit set in parameter.
Station No.: Station No. that does not overlap on the same loop.
Network No. and station No. are not used when communicating with an Ethernet port of CPU (Built-in Ethernet port).

Protocol TCP

Figure 7.6 PC side I/F Ethernet board setting screen

- Network No., Station No.: Not used.
- Protocol: Select the protocol to be used, "TCP" or "UDP". (Select the same protocol which has been set under the Open settings at (1) (a) 2).)

2) **Select "PLC module" for "PLC side I/F".**
Enter the IP address or the host name of the CPU module on the detailed setting screen as shown in Figure 7.7. (For the host name, enter the name set in Microsoft® or Windows® hosts files.)



PLC side I/F   Detailed setting of PLC module

PLC mode: QCPU(Qmode)

□ Ethernet port direct connection

● IP address: 192.168.3.39   IP input format: DEC.

○ Host Name:

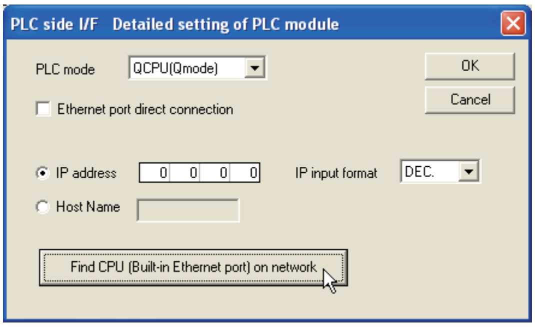
Figure 7.7 PLC side I/F Detailed setting of PLC module screen

3) **Make the setting for "Other station".**

☒POINT

The Find CPU function can be used for specifying the CPU IP address in the connection using a hub.

This function can be activated on the PLC side I/F Detailed setting of PLC module screen, finds the CPU modules connected to the same hub as GX Developer, and displays a list.



PLC side I/F Detailed setting of PLC module

PLC mode: QCPU(Qmode)

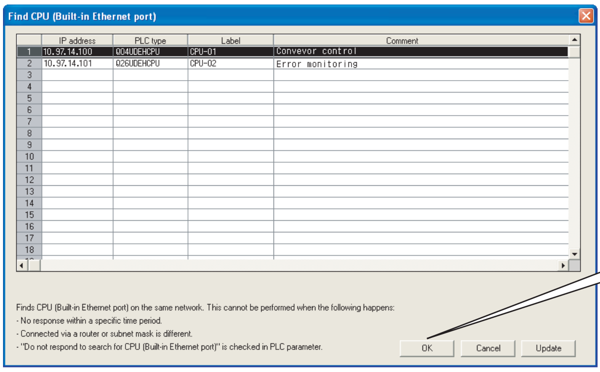
□ Ethernet port direct connection

● IP address: 0 . 0 . 0 . 0 IP input format: DEC.

○ Host Name:

Figure 7.8 PLC side I/F Detailed setting of PLC module screen

A list of the CPU modules found is displayed.



Find CPU (Built-in Ethernet port)

	IP address	PLC type	Label	Comment
1	10.97.14.100	Q04UDEHCPU	CPU-01	Convevor control
2	10.97.14.101	Q26UDEHCPU	CPU-02	Error monitoring
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				

Finds CPU (Built-in Ethernet port) on the same network. This cannot be performed when the following happens:
- No response within a specific time period.
- Connected via a router or subnet mask is different.
- "Do not respond to search for CPU (Built-in Ethernet port)" is checked in PLC parameter.

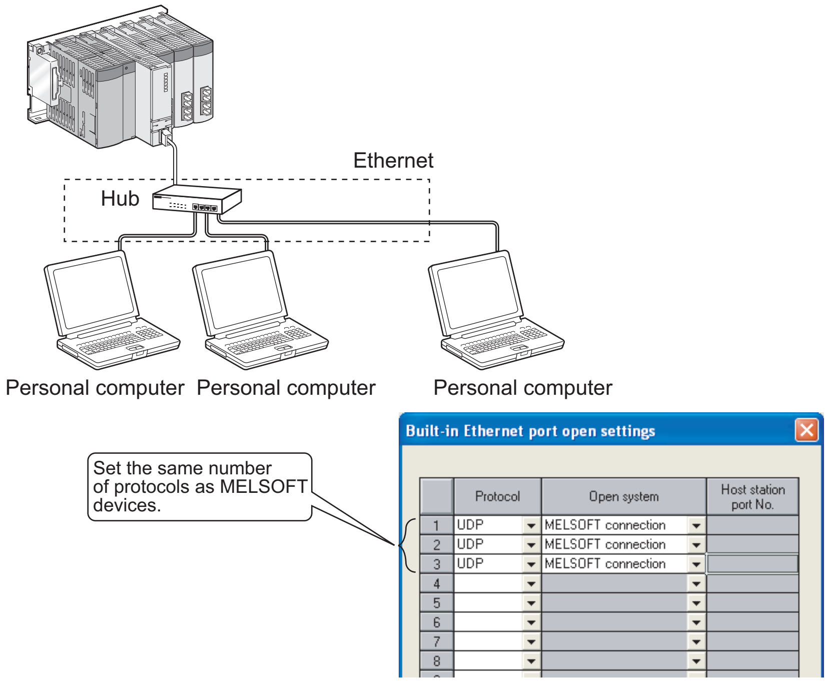
The IP address of the corresponding CPU is input automatically by clicking the OK button.

Figure 7.9 Find CPU (Built-in Ethernet port) screen

- CPU modules connected to a cascade connected hub are also found and displayed in the list.
- CPU modules connected via a router cannot be found.
- CPU modules connected via wireless LAN may not be found since the Ethernet communication does not stabilize due to lost packets.
- If CPU modules with same IP address are found in the list, re-examine the parameter settings of the CPU IP address. If communication is performed with the situation left as is, a communication error occurs.
- CPU modules may not be found if service processing load on CPU modules is heavy.
  Increase the service processing time at "Service processing setting" on the PLC system tab of PLC parameter.

(3) Precautions

- When the protocol is set to "TCP", existence check using the KeepAlive function is performed.
  The CPU module sends an existence check message five seconds after the last message from the external device is received and waits to see whether a response from the external device is received. If there is no response, the CPU module keeps sending an existence check message in five-second intervals.
  When a response has not been confirmed within 45 seconds, the connection is disconnected, considering that no external device exists.
  The connection may be disconnected if the external device does not support the TCP KeepAlive function (response to ACK messages for KeepAlive).
- Establish connections within the number of protocols set for open settings of PLC parameter.
  If connections are established exceeding the setting range from a personal computer, the status of some applications may change into the wait status, making the application inoperable.
- Establish connections within the number of protocols set for open settings of PLC parameter.
  If connections are established exceeding the setting range from a personal computer, the time period causing a time-out error may increase.
- In TCP connection, if no ACK response message for TCP protocol device is received from the external device, resend processing is performed. Resend processing is performed six times, 0.3, 0.6, 1.2, 2.4, 4.8, and 9.6 seconds after the first message. If there is no ACK response within a period of 19.2 seconds after the last message is resent, the connection is disconnected, considering that the external device is not operating normally. (The connection is disconnected within the total period of 38.1 seconds.)
- To communicate with multiple MELSOFT devices using the UDP protocol, set the same number of protocols as MELSOFT devices in PLC parameter.


Ethernet
Hub
Personal computer
Personal computer
Personal computer
Set the same number of protocols as MELSOFT devices.
Built-in Ethernet port open settings

Built-in Ethernet port open settings

	Protocol	Open system	Host station port No.
1	UDP	MELSOFT connection	
2	UDP	MELSOFT connection	
3	UDP	MELSOFT connection	
4			
5			
6			
7			
8			

Figure 7.10 Setting for MELSOFT connection using UDP protocol

- When the sampling trace function is executed using GX Developer which is connected with built-in Ethernet ports of the CPU module, stop the function before powering OFF the programmable controller or resetting the CPU module.
- To execute the remote STOP or remote PAUSE to a CPU module on another station using GX Developer which is connected with built-in Ethernet ports of the CPU module, execute the remote STOP or remote PAUSE before powering OFF the programmable controller or resetting the CPU module.

(4) Communication via routers

MELSOFT connection allows access to CPU modules on the network, such as internal LAN, via routers.

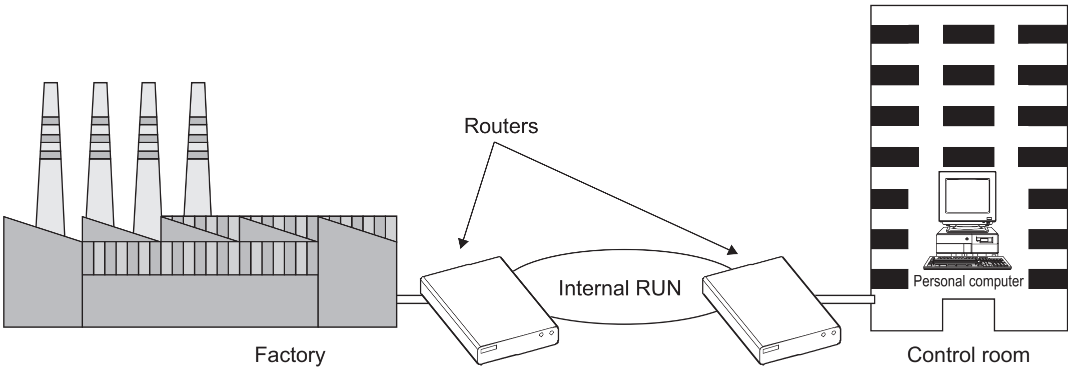
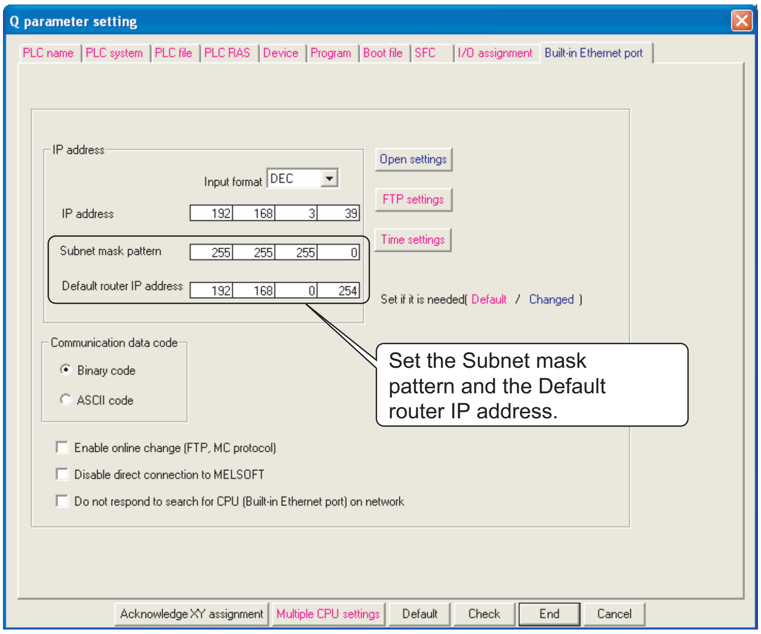


Figure 7.11 Example of router use

When accessing a CPU module via routers, set the items in parameter as shown in Figure 7.12. For other settings, refer to (1) in this section.


Q parameter setting
PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment | Built-in Ethernet port
Open settings
FTP settings
Time settings
Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

IP address

Input format DEC

IP address 192 168 3 39

Subnet mask pattern 255 255 255 0

Default router IP address 192 168 0 254

Set if it is needed(Default / Changed)

Communication data code

● Binary code

○ ASCII code

□ Enable online change (FTP, MC protocol)

□ Disable direct connection to MELSOFT

□ Do not respond to search for CPU (Built-in Ethernet port) on network

Set the Subnet mask pattern and the Default router IP address.

Figure 7.12 Built-in Ethernet port tab

7.3 GX Developer Direct Connection (Simple Connection)

When connecting the Built-in Ethernet port QCPU with GX Developer, the direct connection (simple connection), which connects them using only one Ethernet cable (not using a hub) is available.
The direct connection enables communication with only specifying connection target (Broadcasting). IP address setting is not required.

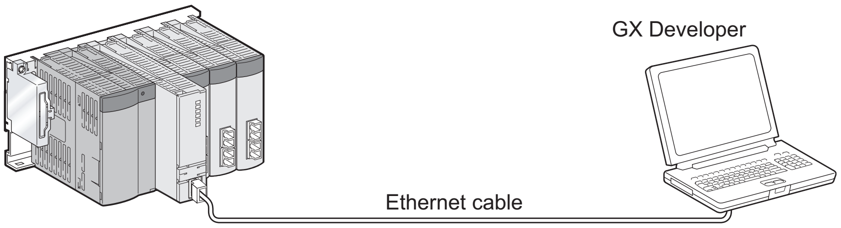
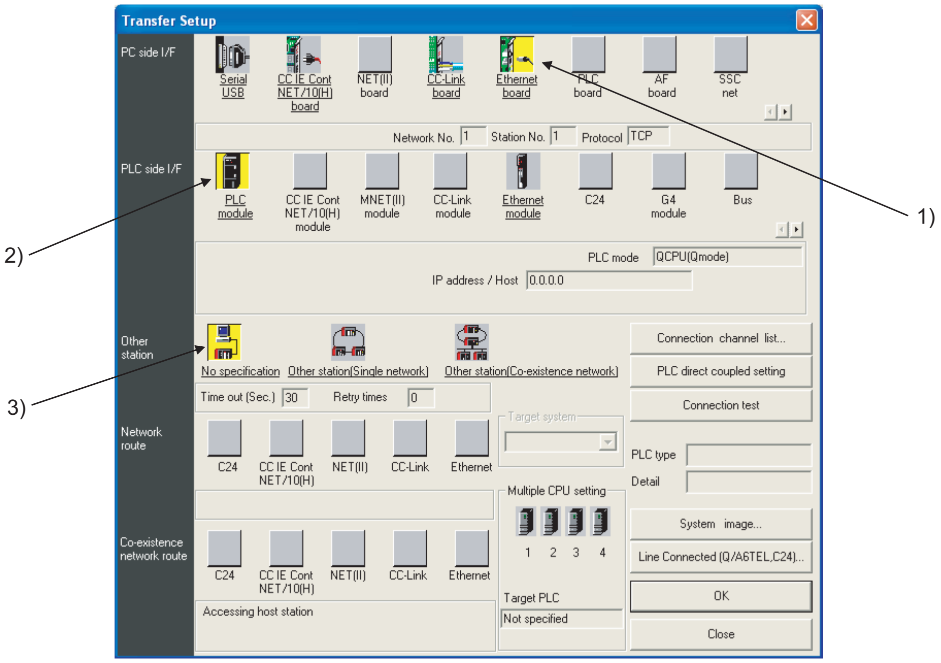


Figure 7.13 Direct connection (simple connection)

(a) Communication setting in GX Developer

Set the items on the Transfer Setup screen in GX Developer as shown in Figure 7.14.



Network No. 1 Station No. 1 Protocol TCP

PLC mode QCPU(Qmode)

IP address / Host 0.0.0.0

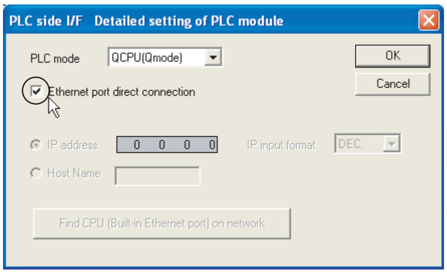
Time out (Sec.) 30 Retry times 0

Target PLC Not specified

Figure 7.14 Transfer Setup screen

1) **Select "Ethernet board" for "PC side I/F".**

2) **Select "PLC module" for "PLC side I/F".**
Check the "Ethernet port direct connection" checkbox on the detailed setting screen as shown in Figure 7.15.


PLC side I/F Detailed setting of PLC module

PLC mode: QCPU(Qmode)

☑ Ethernet port direct connection

○ IP address: 0 0 0 0   IP input format: DEC.

○ Host Name:

Figure 7.15 PLC side I/F Detailed setting of PLC module screen

3) **Make the setting for "Other station".**

(b) Precautions

1) When the CPU module is connected to LAN line, do not perform communication using direct connection. If performed, the communication may put a load to LAN line and adversely affect communications of other devices.

2) If the "Disable direct connection to MELSOFT" checkbox is checked on the Built-in Ethernet port tab of PLC parameter in GX Developer, the direct connection is not available.

3) The system configuration that connects a Built-in Ethernet port QCPU with an external device using a hub as shown in Figure 7.16 is not regarded as direct connection since the hub is used. Do not check the "Ethernet port direct connection" checkbox on the detailed setting screen.

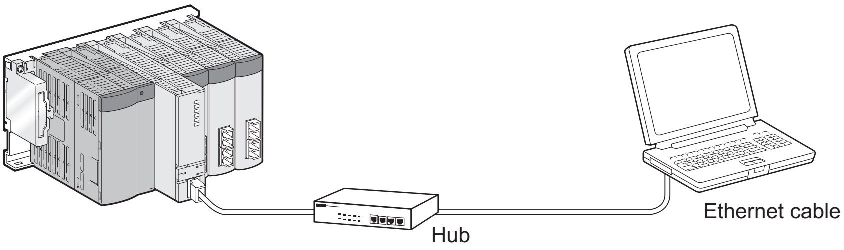


Figure 7.16 Non-direct connection

4) Under the following conditions, direct connection communication may not be available. In that case, check the setting of the CPU module and/or personal computer.
    - In the CPU module IP address, bits corresponding to "0" in the personal computer subnet mask are all ON or all OFF.
    [Example] CPU IP address: 64.64.255.255
    Personal computer IP address: 64.64.1.1
    Personal computer subnet mask: 255.255.000.000
    - In the CPU module IP address, bits corresponding to the host address for each class in the personal computer IP address are all ON or all OFF.
    [Example] CPU module IP address: 64.64.255.255
    Personal computer IP address: 192.168.0.1
    Personal computer subnet mask: 255.255.000.000

7.4 MC Protocol Communication

Built-in Ethernet ports of the Built-in Ethernet port QCPU enable communication using the MC protocol[*1].
External devices such as personal computers and display devices read/write device data from/to the CPU module using the MC protocol.
External devices monitor the operation of the CPU module, analyze data, and manage production by reading/writing device data.
Besides, the remote password function can prevent unauthorized access to the CPU module from outside of the system. (☞ Section 7.7)

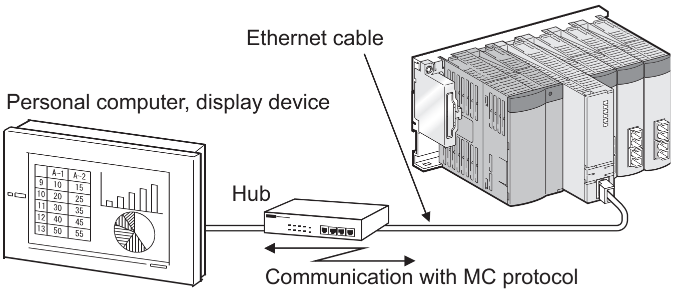

Ethernet cable
Personal computer, display device
Hub
Communication with MC protocol

Figure 7.17 MC protocol communication

☒POINT

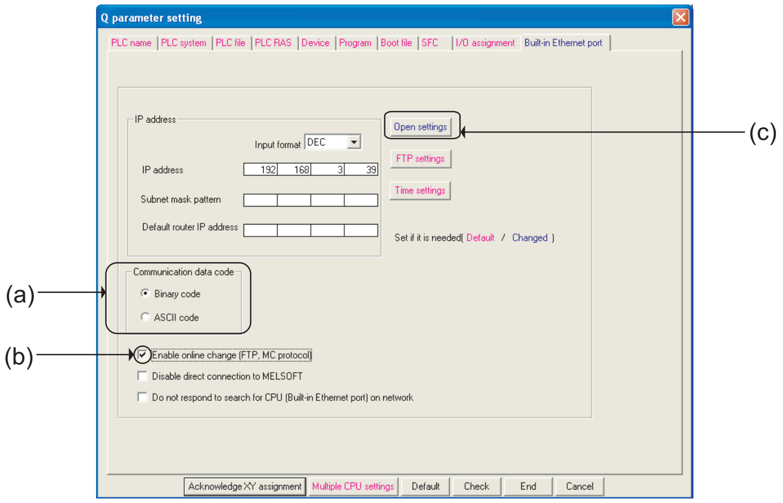
External devices such as personal computers and display devices can communicate with only the connected CPU module using the MC protocol.
An access to any of the following CPU modules is not available.

- CPU modules on another station via network
- Other CPU modules in a multiple CPU system

[*1] : The MC protocol is an abbreviation for the MELSEC communication protocol.
The MELSEC communication protocol is a name of the communication method used to access CPU modules from external devices in accordance with the communication procedure of Q series programmable controllers (such as serial communication modules, Ethernet modules).
For details on the MC protocol, refer to the following manual.
☞ Q Corresponding MELSEC Communication Protocol Reference Manual

(1) Setting for MC protocol communication

Setting for communication using the MC protocol is described below.
Set the items on the Built-in Ethernet port tab of PLC parameter in GX Developer as shown below.



IP address

Input format DEC

IP address 192 168 3 39

Subnet mask pattern

Default router IP address

Set if it is needed(Default / Changed)

Communication data code

● Binary code

○ ASCII code

☑ Enable online change (FTP, MC protocol)

□ Disable direct connection to MELSOFT

□ Do not respond to search for CPU (Built-in Ethernet port) on network

Figure 7.18 Built-in Ethernet port tab

(a) Communication data code

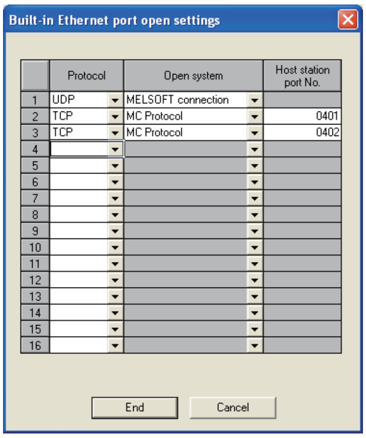
Select a communication data code used for the MC protocol, "Binary code" or "ASCII code".

(b) Enable online change (FTP, MC protocol)

Check the checkbox to enable online change when writing data to the CPU module from the external device that communicates using the MC protocol.
For details on the available functions with this setting, refer to (2) in this section.

(c) Open settings

Set the following items.



	Protocol	Open system	Host station port No.
1	UDP	MELSOFT connection	
2	TCP	MC Protocol	0401
3	TCP	MC Protocol	0402
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			

Figure 7.19 Open settings screen

- Protocol :
  Select a connection used as MC protocol. (Up to 16 CPU modules can be connected.)
- Open system : Select "MC protocol".
- Host station port No. : Set the host station port number (in hexadecimal). (Required) (Setting range : 0401H to 1387H, 1392H to FFFEH)

☒**POINT**

When the "Enable online change (FTP, MC protocol)" box is unchecked, if a data write request is sent from an external device to the CPU module which is in the RUN status, data will not be written to the CPU module and the module returns the NAK message.

(2) Command list

When the Built-in Ethernet port QCPU communicates using the MC protocol, commands listed in Table7.3 can be executed.

For details on the MC protocol, refer to the following manual.

☞ Q Corresponding MELSEC Communication Protocol Reference Manual

Table7.3 List of MC protocol commands supported in MC protocol communication function of Built-in Ethernet port QCPU

Function			Command (Subcommand) *1	Description	Number of processed points	Status of CPU module		
						STOP	RUN	
							Write enabled	Write disabled
Device memory	Batch read	In units of bits	0401 (0001)	Reads bit devices in units of one point.	ASCII: 3584 points BIN: 7168 points	○	○	○
		In units of words	0401 (0000)	Reads bit devices in units of 16 points.	960 words (15360 points)			
				Reads word devices in units of one point.	960 points			
	Batch write	In units of bits	1401 (0001)	Writes bit devices in units of one point.	ASCII: 3584 points BIN: 7168 points	○	○	×
		In units of words	1401 (0000)	Writes bit devices in units of 16 points.	960 words (15360 points)			
				Writes word devices in units of one point.	960 points			
	Random read *2	In units of words	0403 (0000)	Reads bit devices in units of 16 or 32 points by randomly specifying a device or device number.	192 points	○	○	○
				Reads word devices in units of one or two points by randomly specifying a device or device number.				
	Test (Random write)	In units of bits	1402 (0001)	Sets/resets bit devices in units of one point by randomly specifying a device or device number.	188 points	○	○	×
		In units of words *2	1402 (0000)	Sets/resets bit devices in units of 16 or 32 points by randomly specifying a device or device number.	*5			
				Writes word devices in units of one or two points by randomly specifying a device or device number.				
	Monitor registration *2*3*4	In units of words	0801 (0000)	Registers bit devices to be monitored in units of 16 or 32 points.	192 points	○	○	○
				Registers word devices to be monitored in units of one or two points.				
	Monitor	In units of words	0802 (0000)	Monitors devices registered.	Number of registered points	○	○	○
Remote password	Unlock		1630 (0000)	Specifies the remote password and changes the status from locked to unlocked.	---	○	○	○
	Lock		1631 (0000)	Specifies the remote password and changes the status from unlocked to locked.	---	○	○	○

○ : Available × : Not available

* 1 : Subcommand is for the QnA-compatible 3E frame.

* 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 : During monitor registration, monitor condition cannot be set.

* 4 : Do not execute monitor registration from multiple external devices. If executed, the last monitor registration becomes valid.

* 5 : Set the number of processed points so that the following condition is satisfied.

(Number of word access points) × 12 + (Number of double-word access points) × 14 ≦ 1920

- Bit devices are regarded as 16 bits during word access and 32 bits during double-word access.
- Word devices are regarded as one word during word access and two words during double-word access.

(3) Available devices

Table7.4 lists the devices available in commands used in the MC protocol communication function.

Table7.4 List of available devices in Built-in Ethernet port QCPU

Classification	Device		Device code	Device number range (Default)		Remarks
Internal system device	Special relay		SM	000000 to 002047	Decimal	---
	Special register		SD	000000 to 002047	Decimal	
internal user device	Input		X	000000 to 001FFF	Hexadecimal	
	Output		Y	000000 to 001FFF	Hexadecimal	
	Internal relay		M	000000 to 008191	Decimal	• When the device number range is changed, access is possible up to the largest device number after the change. • Local devices cannot be accessed.
	Latch relay		L	000000 to 008191	Decimal	
	Annunciator		F	000000 to 002047	Decimal	
	Edge relay		V	000000 to 002047	Decimal	
	Link relay		B	000000 to 001FFF	Hexadecimal	
	Data register		D	000000 to 012287	Decimal	
	Link register		W	000000 to 001FFF	Hexadecimal	
	Timer	Contact	TS	000000 to 002047	Decimal	
		Coil	TC			
		Current value	TN			
	Retentive timer	Contact	SS	000000 to 002047	Decimal	
		Coil	SC			
		Current value	SN			
	Counter	Contact	CS	000000 to 001023	Decimal	
		Coil	CC			
		Current value	CN			
	Link special relay		SB	000000 to 007FFF	Hexadecimal	
	Link special register		SW	000000 to 007FFF	Hexadecimal	
	Step relay		S	000000 to 008191	Decimal	
	Direct input		DX	000000 to 000FFF	Hexadecimal	*1
	Direct output		DY	000000 to 000FFF	Hexadecimal	
Index register	Index register		Z	000000 to 000019	Decimal	---
File register	File register		R	000000 to 032767	Decimal	
			ZR	000000 to 3FD7FF	Hexadecimal	
Extended data register	Extended data register		D	• Binary: 000000 to 4184063 (4086k points maximum) • ASCII: 000000 to 999999 (976.6k points maximum)	Decimal	If the number of points is set on the PLC file tab of PLC parameter, access is possible up to the largest device number after the setting. However, in the ASCII code communication, the number of points described on the left is the access limit.
Extended link register	Extended link register		W	000000 to 3FD7FF (4086k points maximum)	Hexadecimal	If the number of points is set on the PLC file tab of PLC parameter, access is possible up to the largest device number after the setting.

* 1 : Devices of DX/DY1000 or later are not available. Use X/Y devices to access devices of X/Y1000 or later.

(4) Precautions

(a) Number of connected modules

In the connection with external devices using the MC protocol, the number of CPU modules set as "MELSOFT connection" in the open settings on Built-in Ethernet port tab of PLC parameter can be connected simultaneously.

(b) Data communication frame

Table7.5 shows the frames available in the Built-in Ethernet port QCPU.

Table7.5 Available frames in Built-in Ethernet port QCPU

Communication frame	Built-in Ethernet port QCPU
4E frame	×
QnA-compatible 3E frame	○
A-compatible 1E frame	×

(c) Access range

- Only connected CPU module can be accessed. Accessing a CPU module not connected results in an error.
- In a multiple CPU system, other CPU modules not connected to Ethernet cannot be accessed.
- Accessing a CPU module on another station in CC-Link IE controller network, MELSECNET/H, Ethernet or CC-Link via a connected CPU module is not possible.

(d) Precautions when UDP protocol is selected

- If a new request message is sent to the same UDP port while the port waits for a response message, the new request message is discarded.
- Setting same host station port number to multiple UDP ports is regarded as one setting. When communicating with multiple external devices using the same host station port number, select TCP protocol.

(e) **Response message receive processing**

Figure 7.20 shows an example of the response message receive processing on the external device side.

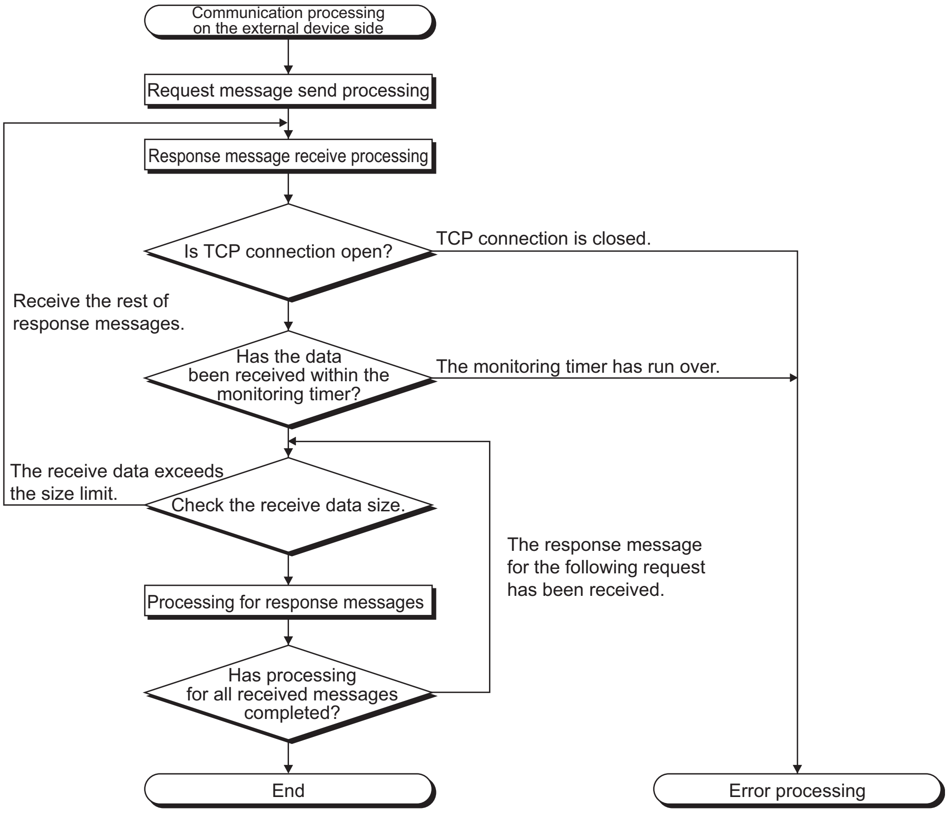

Communication processing on the external device side
Request message send processing
Response message receive processing
Is TCP connection open?
TCP connection is closed.
Receive the rest of response messages.
Has the data been received within the monitoring timer?
The monitoring timer has run over.
The receive data exceeds the size limit.
Check the receive data size.
The response message for the following request has been received.
Processing for response messages
Has processing for all received messages completed?
End
Error processing

Figure 7.20 Example of the response messages receive processing on the external device side

Remark

Personal computers use the TCP socket functions internally for Ethernet communication.
These functions do not have boundary concept. Therefore, when data is sent by executing the "send" function once, the "recv" function needs to be executed once, twice or more to receive the same data. (One execution of the "send" function does not correspond to one execution of the "recv" function.)
For this reason, receive processing described above is required on the external device side.
If the "recv" function is used in blocking mode, data may be read by executing the function once.

(5) Error codes for communication using MC protocol

Table7.6 shows the error codes, error descriptions, and corrective actions that will be sent from the CPU module to an external device when an error occurs during communication using the MC protocol.

Table7.6 List of available devices in QnUDE(H)CPU

No.	Error code (Hexadecimal)	Description	Corrective action
1	4000_H to $4FFF_H$	CPU detected error (Error that occurred in other than communication using the MC protocol)	Refer to the QCPU User's Manual (Hardware Design, Maintenance and Inspection) and take corrective action.
2	0055_H	When the setting for online change is disabled on the Built-in Ethernet port tab of PLC parameter in GX Developer, an external device requested online change to the CPU module.	• Enable the online change setting on the Built-in Ethernet port tab of PLC parameter in GX Developer and write data to the CPU module. • Change the status of the CPU module to STOP and write data.
3	$C050_H$	When the communication data code setting is set to ASCII code in the QnUDE(H)CPU, ASCII code data that cannot be converted to binary code was received.	• Set the communication data code to binary code and restart the QnUDE(H)CPU for communication. • Correct the send data on the external device side and resend the data.
4	$C051_H$ to $C054_H$	The number of device points for reading/writing is outside the allowable range.	Correct the number of device points for reading/writing and resend the data to the QnUDE(H)CPU.
5	$C056_H$	The read/write request data exceeds the allowable address range.	Correct the start address or the number of device points for reading/writing, and resend the data to the QnUDE(H)CPU. (Do not exceed the allowable address range.)
6	$C058_H$	The request data length after the ASCII to binary conversion does not match the data size of the character area (a part of text data).	Check and correct the text data or the request data length of the header data, and resend the data to the QnUDE(H)CPU.
7	$C059_H$	• The command and/or subcommand are specified incorrectly. • The command and/or subcommand not supported in the QnUDE(H)CPU are specified.	• Check the request data. • Use commands and/or subcommands supported in the QnUDE(H)CPU.
8	$C05B_H$	The QnUDE(H)CPU cannot read/write data from/to the specified device.	Check the device for reading/writing data.
9	$C05C_H$	The request data is incorrect. (ex. specifying data in units of bits for reading/writing of word devices)	Correct the request data (such as subcommand correction) and resend the data to the QnUDE(H)CPU.
10	$C05D_H$	Monitor registration is not performed.	Perform the monitor registration before monitor operation.
11	$C05F_H$	The external device sent a request that cannot be executed in the target CPU module.	• Correct the network number, PC number, request destination module I/O number, and request destination module station number. • Correct the read/write request data.
12	$C060_H$	The request data is incorrect. (ex. incorrect specification of data for bit devices)	Correct the request data and resend the data to the QnUDE(H)CPU.
13	$C061_H$	The request data length does not match the data size of the character area (a part of text data)	Check and correct the text data or the request data length of the header data, and resend the data to the QnUDE(H)CPU.
14	$C06F_H$	The CPU module received a request message in ASCII when the communication data code setting is set to binary code, or received a request message in binary when the data code setting is set to ASCII code. (As for this error code, only error history is registered. The error response message is not returned.)	• Send request messages corresponding to the communication data code setting. • Set the communication data code corresponding to the request message.
15	$C070_H$	The device memory extension cannot be specified for the target station.	Read/Write data to the device memory without specifying the extension.
16	$C0B5_H$	Data that cannot communicate in the CPU module or Ethernet module is specified.	• Check the request data. • Stop the current request.
17	$C200_H$	The remote password is incorrect.	Check the remote password and perform unlock/lock processing again.
18	$C201_H$	The communication target port is in the remote password locked status. Or, when the communication data code setting is set to ASCII code, any data of subcommands and later cannot be converted to binary code since the remote password is locked.	Perform remote password unlock processing before communication.
19	$C204_H$	The external device is different from the one that quested remote password unlock processing.	Have the same external device which has requested unlock processing request remote password lock processing.

7.5 Time Setting Function (SNTP Client)

The Built-in Ethernet port QCPU collects time information from a time information server connected to LAN, making it possible to set the CPU time automatically.
The Built-in Ethernet port QCPU time setting function queries a time information server to get the time at the specified timing and sets the time sent from the time information server as clock data for the CPU module.

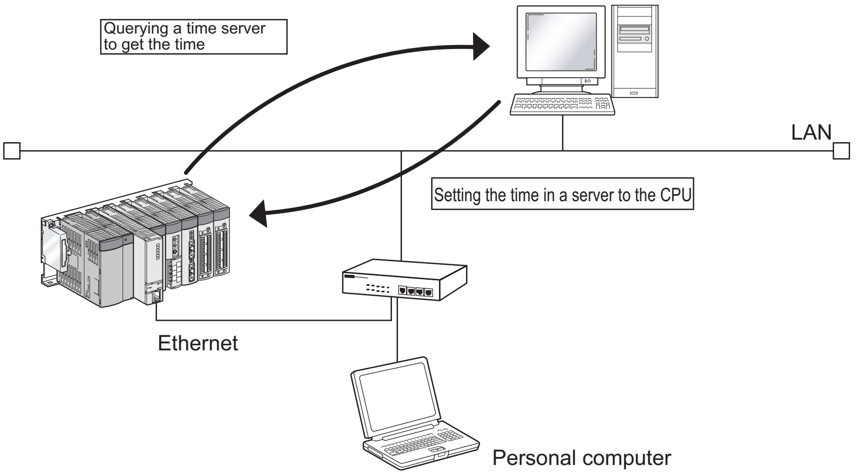

Querying a time server to get the time
LAN
Setting the time in a server to the CPU
Ethernet
Personal computer

Figure 7.21 Image of time setting function

(1) Function details

The time setting operation is executed based on the following timing.

- At programmable controller power ON or CPU module reset.
- At a specified time interval or a specified time.
- At special relay ON.

For details on the specified time setting, refer to (2) in this section.
A communication time-out occurs after 20 seconds. When a communication time-out occurs, the value in SD1270 changes to $0FFFF_H$.

(a) Execution at programmable controller power ON or CPU module reset

Time is set when the programmable controller is powered ON or the CPU module is reset.

(b) Execution at a specified time interval (Execution interval)

Time is set at a specified time interval.
The settable execution interval range is 1 to 1440 minutes.

(c) Execution at a specified time

Time is set at a specified time.
The settable time range is 00:00 to 23:30.

(d) Execution with a special relay

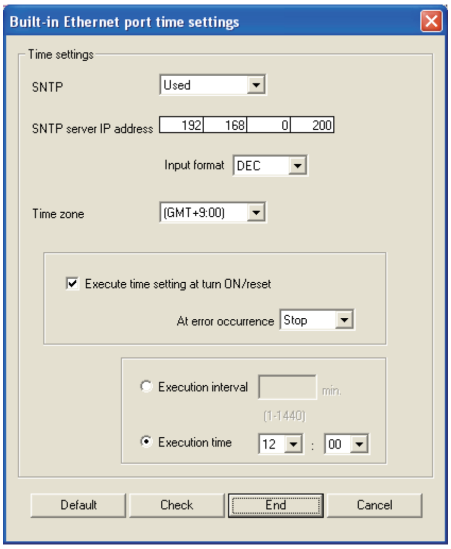
Time is set when the special relay (SM1270) is turned ON for one scan.

☒POINT

1. Check the connection of the hub or the external device first when executing a time setting operation at programmable controller power ON or CPU module reset.
2. The time setting result details can be checked with the special registers (SD1270 to SD1275).
3. Other time setting operations are ignored during execution of the time setting function.

(2) Setting for time setting function

To use the time setting function, set the time on the screen activated from the Built-in Ethernet port tab of PLC parameter in GX Developer.



Built-in Ethernet port time settings

Time settings

SNTP: Used

SNTP server IP address: 192 168 0 200

Input format: DEC

Time zone: (GMT+9:00)

☑ Execute time setting at turn ON/reset

At error occurrence: Stop

○ Execution interval min. (1-1440)

● Execution time 12 : 00

Figure 7.22 Time settings screen

- SNTP: Select whether to use the time setting function. (Required)
- SNTP server IP address: Specify the SNTP server IP address. (Required)
- Time zone:
  Specify the time zone in which the time is to be synchronized. (Required)
  Default is set to Japan Standard Time "GMT + 9:00".
- Execute time setting at turn ON/reset:
  Select whether to execute the time setting function when the programmable controller is powered ON or when the CPU module is reset.
- At error occurrence:
  Select whether to stop or continue when a time setting error occurs when the programmable controller is powered ON or when the CPU module is reset.
- Execution interval:
  Select to execute the time setting function at a specified time interval.
- Execution time:
  Select to execute the time setting function at a specified time.
  (Either of the two options must be selected.)

(3) Precautions

- To use the time setting function, an SNTP server personal computer (time information server) is required on LAN.
- A delay occurs with respect to the time set in the CPU module as a result of the time required for communication with the server. Specify the closest SNTP server possible on the network.
- In a multiple CPU system, enable the time setting function of only the Built-in Ethernet port QCPU No.1.
  When a CPU module other than the Built-in Ethernet port QCPU No. 1 is enabled, the clock data of the Built-in Ethernet port QCPU No. 1 is automatically set.
- This function is available within the year range 1980 to 2079.

7.6 File Transfer Function (FTP)

The Built-in Ethernet port QCPU supports the FTP (File Transfer Protocol) server function. FTP is a protocol for transferring files between CPU modules and external devices.
An external device with a FTP client function can simply access to files within the CPU module directly by using this FTP server function.

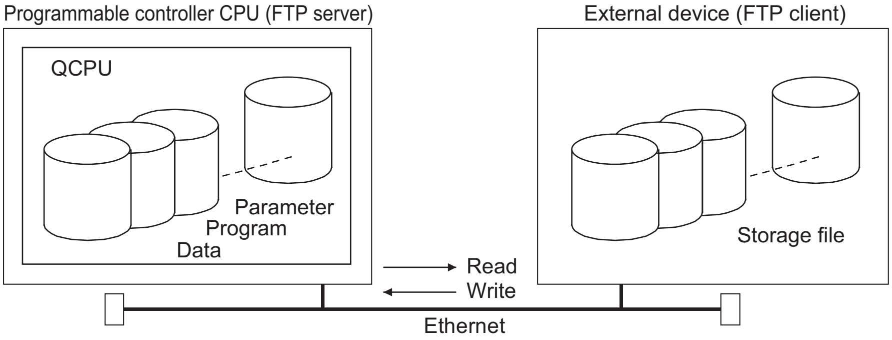

Programmable controller CPU (FTP server)
External device (FTP client)
QCPU
Parameter
Program
Data
Storage file
Read
Write
Ethernet

Figure 7.23 File transfer function (FTP)

The following accesses can be performed to files in the Built-in Ethernet port QCPU from an external device with a FTP client function.

(a) Reading (downloading) files from Built-in Ethernet port QCPU
A function for storing CPU module files in an external device.

(b) Writing (uploading) files to Built-in Ethernet port QCPU
A function for registering files stored in an external device to the CPU module.

(c) Browsing Built-in Ethernet port QCPU file names
A function for checking files registered in the CPU module on the external device side.

☒**POINT**

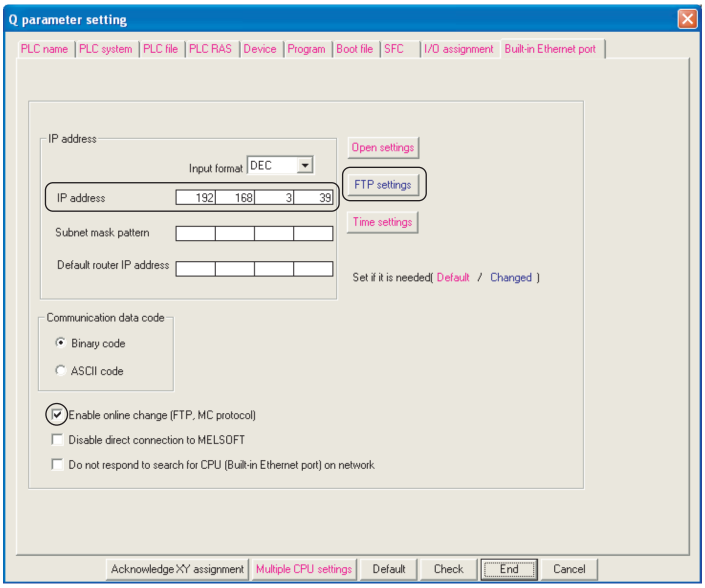
In a multiple CPU system, only the CPU module connected with an Ethernet cable can transfer files.

(1) Setting for FTP communication

Setting for communication using the FTP is described below.

(a) Operation on Built-in Ethernet port QCPU side

Set the items on the Built-in Ethernet port tab of PLC parameter in GX Developer as shown below.



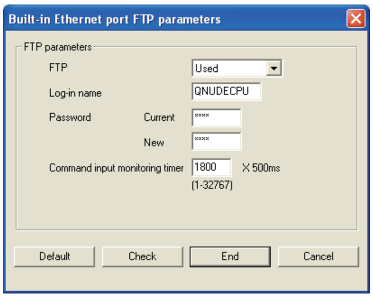
IP address
Input format: DEC
IP address: 192 . 168 . 3 . 39
Subnet mask pattern:
Default router IP address:

Communication data code
● Binary code
○ ASCII code

☑ Enable online change (FTP, MC protocol)
□ Disable direct connection to MELSOFT
□ Do not respond to search for CPU (Built-in Ethernet port) on network

Figure 7.24 Built-in Ethernet port tab

- IP address: Set the CPU IP address
- Enable online change (FTP, MC protocol):
  Check this checkbox to enable online change.
- FTP setting: Set the FTP parameters as shown below.



FTP parameters
FTP: Used
Log-in name: QNUDECPU
Password Current: ****
New: ****
Command input monitoring timer: 1800 × 500ms (1-32767)

Figure 7.25 FTP settings screen

- FTP: Set to "Used".
- Log-in name:
  Set the login name for an external device to request file transfer (log in).
- Password:
  Set the FTP password for an external device to request file transfer (log in).
  To change the password, enter both the old and the new passwords for confirmation.
- Command input monitoring timer:
  Set the monitoring time for command input performed by the CPU module.
  When no command is input within the set period of time, the FTP connection is disconnected. (Setting range : 1 to 32767 (× 500ms)) Set the value so that the time will be larger than the time required for file transfer.

Remark

To access the CPU module via a router, set the "Subnet mask pattern" and the "Default router IP address" settings as well. (☞ Section 7.2)

(b) Operation on external device (FTP client) side

The procedure and required processing on the external device side when using the FTP server function of Built-in Ethernet port QCPU are described below. FTP operation commands used in each operation and their input methods are also described. (<ret> indicates an input of the CR, Enter or Return key.)

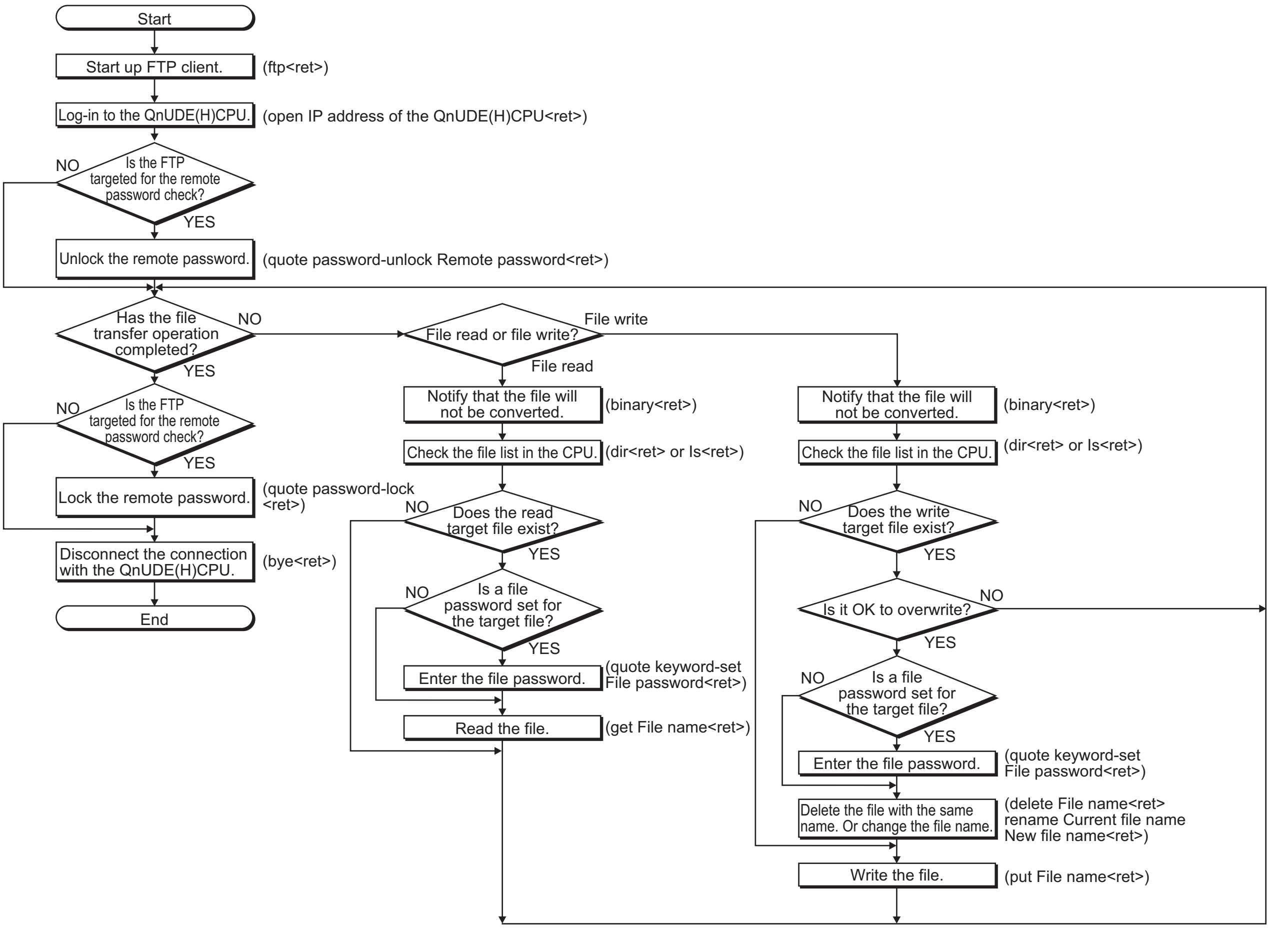
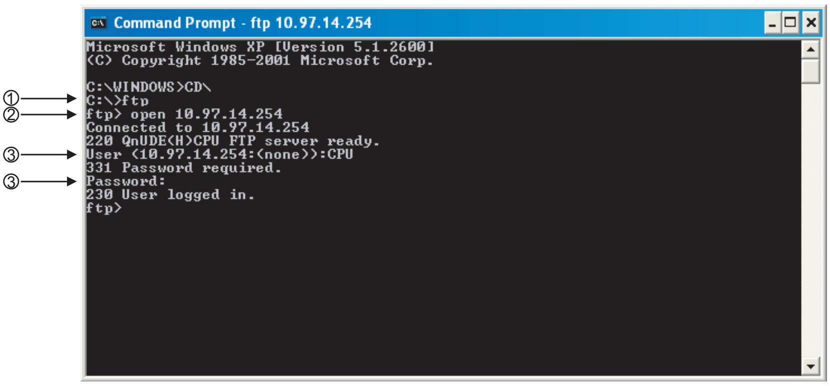

Start
Start up FTP client.
(ftp<ret>)
Log-in to the QnUDE(H)CPU.
(open IP address of the QnUDE(H)CPU<ret>)
Is the FTP targeted for the remote password check?
NO
YES
Unlock the remote password.
(quote password-unlock Remote password<ret>)
Has the file transfer operation completed?
NO
YES
Is the FTP targeted for the remote password check?
NO
YES
Lock the remote password.
(quote password-lock <ret>)
Disconnect the connection with the QnUDE(H)CPU.
(bye<ret>)
End
File read or file write?
File write
File read
Notify that the file will not be converted.
(binary<ret>)
Check the file list in the CPU.
(dir<ret> or ls<ret>)
Does the read target file exist?
NO
YES
Is a file password set for the target file?
NO
YES
Enter the file password.
(quote keyword-set File password<ret>)
Read the file.
(get File name<ret>)
Notify that the file will not be converted.
(binary<ret>)
Check the file list in the CPU.
(dir<ret> or ls<ret>)
Does the write target file exist?
NO
YES
Is it OK to overwrite?
NO
YES
Is a file password set for the target file?
NO
YES
Enter the file password.
(quote keyword-set File password<ret>)
Delete the file with the same name. Or change the file name.
(delete File name<ret> rename Current file name New file name<ret>)
Write the file.
(put File name<ret>)

Figure 7.26 FTP client side operation procedure

1) **Logging in to Built-in Ethernet port QCPU**

Operations from starting the FTP to logging in to the Built-in Ethernet port QCPU are described below.

Example) Start up the FTP from the MS-DOS prompt of a Microsoft® Windows® XP Operating System.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS>CD\
C:\>ftp
ftp> open 10.97.14.254
Connected to 10.97.14.254
220 QnUDE(H)CPU FTP server ready.
User (10.97.14.254:(none)):CPU
331 Password required.
Password:
230 User logged in.
ftp>
```

Figure 7.27 Example of screen for logging in to Built-in Ethernet port QCPU

① Start FTP (FTP<ret>)

② Connect to FTP server (open Built-in Ethernet port QCPU IP address<ret>)

③ Specify login name (Login name (Default: QNUDECPU)<ret>)
Specify password (Password (Default: QNUDECPU)<ret>)

2) **Unlocking/Locking remote password**

When the FTP communication port is specified for a remote password check target in remote password settings, the remote password needs to be unlocked using the following command.
(quote password-unlock Remote password<ret>)

At the end of the operation, change the remote password to the locked status using the following command.
(quote password-lock<ret>)

POINT

When the FTP communication port is specified for a remote password check target in remote password settings, other commands cannot be used until the remote password is unlocked.

3) **Entering file password**

When a file password is set for the target file, the file password needs to be entered using the following command before accessing the file.
(quote keyword-set File password<ret>)

(2) List of transferable files

Table7.7 lists the transferable files using the file transfer function.

Table7.7 List of transferable files

Target memory	Built-in memory			Memory card (RAM)	Memory card (ROM)		File name or extension
File storage drive number	0	3	4	1	2	2	
File type	Program memory *2, *3	Standard RAM	Standard ROM *4	SRAM card *4	Flash card	ATA card *4	
Parameter	○	×	×	×	×	×	PARAM.QPA
Intelligent function module parameter	○	×	×	×	×	×	IPARAM.QPA
Program	○	×	×	×	×	×	.QPG
Device comment	○	×	×	×	×	×	.QCD
Device initial value	○	×	×	×	×	×	.QDI
Device data	×	×	×	×	×	×	.QST
File register	×	○	×	○	○	×	.QDR
Local device	×	×	×	×	×	×	.QDL
Sampling trace file	×	×	×	×	×	×	.QTD
Error history data	×	×	×	×	×	×	.QFD
Programmable controller user data*1	×	×	○	×	×	○	Arbitrary file name
User setting area	×	×	×	×	×	×	---

○ : Transferable × : Not transferable

* 1 : A file for reading/writing data by a sequence program using 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.)

For details on the SP.FREAD/SP.FWRITE instructions, refer to the following manual.

☞ QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)

* 2 : Writing files to the program memory is enabled only when the CPU module is in the STOP status.

* 3 : Write destination is the program cache memory. The user should back up files using the "pm-write" command as required. (☞ (5) in this section)

* 4 : The number of files written using the File transfer function (FTP) is up to the number described below.
[The maximum number of files can be stored in the corresponding drive -1]

(3) List of FTP client side user interface commands

Table7.8 lists the FTP client commands can be used in the Built-in Ethernet port QCPU.

Table7.8 List of FTP client side user interface commands

Command	Function	CPU module status			Remote password		Remarks
		STOP	RUN		Unlocked *2	Locked *2	
			Write enabled *1	Write disabled *1			
binary	Notifies that a file is to be transferred to the FTP server without conversion.	○	○	○	○	×	---
bye	Disconnects the FTP server connection and ends the FTP function.	○	○	○	○	○	
close	Disconnects the FTP server connection.	○	○	○	○	○	
delete	Deletes a file stored in the QnUDE(H)CPU.	○	○	×	○	×	
dir	Displays file information stored in the QnUDE(H)CPU.	○	○	○	○	×	
get	Reads a file from the QnUDE(H)CPU.	○	○	○	○	×	
ls	Displays the file names of files stored in the QnUDE(H)CPU.	○	○	○	○	×	
mdelete	Deletes a file stored in the QnUDE(H)CPU.	○	○	×	○	×	
mdir	Saves information of files stored in the QnUDE(H)CPU.	○	○	○	○	×	
mget	Reads a file from the QnUDE(H)CPU.	○	○	○	○	×	
mls	Saves names of files stored in the QnUDE(H)CPU.	○	○	○	○	×	
mput	Writes a file to the QnUDE(H)CPU.	○	○	×	○	×	
open	Connects to the FTP server.	○	○	○	○	○	
put	Writes a file to the QnUDE(H)CPU.	○	○	×	○	×	
pwd	Displays the current directory name of the QnUDE(H)CPU.	○	○	○	○	×	
quit	Disconnects the FTP server connection and ends the FTP function.	○	○	○	○	○	
quote	Sends an FTP server subcommand.	○	○	○	○	○	*3
rename	Renames a QnUDE(H)CPU file.	○	○	×	○	×	---
user	Inputs the user name and password of QnUDE(H)CPU.	○	○	○	○	○	

○ : Performablel × :Not performable

* 1 : These indicate the online change enable/disable setting set on the Built-in Ethernet port tab of PLC parameter in GX Developer.

* 2 : These indicate the remote password execution status when FTP communication port specified for remote password check target on the Remote password detail settings screen in GX Developer. For details on the remote password, refer to Section 7.7.

* 3 : Table7.9 lists the subcommands that can be used with the "quote" command.

Table7.9 Subcommands that can be used with "quote" command

Command	Function	CPU module status			Remote password		Remarks
		STOP	RUN		Unlocked	Locked	
			Write enabled	Write disabled			
change	Displays/changes the QnUDE(H)CPU file attribute.	○	○	×	○	×	---
keyword-set	Sets/displays/clears the QnUDE(H)CPU file access password.	○	○	○	○	×	
password-lock	Specifies the remote password and changes the status from unlocked to locked.	○	○	○	○	× *4	
password-unlock	Specifies the remote password and changes the status from locked to unlocked.	○	○	○	○	○	
status	Displays the operation information of the QnUDE(H)CPU.	○	○	○	○	×	
run	Changes the QnUDE(H)CPU status to RUN.	○	○	○	○	×	
stop	Changes the QnUDE(H)CPU status to STOP.	○	○	○	○	×	
pm-write	Writes files to the program memory.	○	×	×	○	×	

○ : PerformableI × :Not performable

* 4 : Even if the subcommand is executed, the remote password status remains locked and an error does not occur.

(4) FTP command descriptions and file specification methods

(a) FTP command descriptions

The FTP operation commands (on the FTP client (external device) side) which can be used in the Built-in Ethernet port QCPU are described below.

How to read descriptions

Sections in brackets at [Specification format] can be omitted.

(b) File specification methods

File specification method which is specified during execution of an FTP operation command (on the FTP client side) is described below.

- For the Built-in Ethernet port QCPU, files are specified using drive names and file names.
- When specifying a file in the Built-in Ethernet port QCPU using the FTP function, specify the target file in the following order.

[Specification format] Drive name : \ File name . extension
[Specification example] 3:\MAINSEQ1.QDR
[Specification details]

1) Drive name (Drive No.)
   Specify the drive name of the file transfer target memory.
   For the correspondence between Built-in Ethernet port QCPU target memory and drive name, refer to Table7.7.

2) File name, extension
   - Specify the file transfer target file.
   - Specify the file name according to the rules for naming files using GX Developer.
     Characters that can be used are one-byte alphabets (uppercase), numbers, and symbols. Lowercase alphabets cannot be used.
     File name: 8 characters maximum (one-byte)
     Extension: 3 characters maximum (one-byte)
     Alphabets are not case-sensitive. Lowercase alphabets are treated as uppercase alphabets.
   - For the extension, the name fixed by the Built-in Ethernet port QCPU must be used.
     Specify the extension for the target file in accordance with Table7.7.
   - For details on the file names and extensions, refer to the GX developer operating manual.
   - When using FTP operation commands which can target multiple files, specify the file names and extensions within the file path name using a wild card (* and/or ?).
     * : Used for specifying all files having arbitrary characters (including no characters) after the location where "*" is used.
     ? : Used for specifying all files having arbitrary characters (including no characters) at the location where the "?" is used. (Multiple "?" can be used.)
   - Some FTP clients have restrictions on characters that can be used for file names other than those described above.

(5) FTP operation commands

Functions and usages of FTP operation commands (on the FTP client side) which can be used in the Built-in Ethernet port QCPU are described below.

☒POINT

Note that some FTP operation commands (on the FTP client side) which can be used in the Built-in Ethernet port QCPU may not operate as described in this manual, depending on the FTP application used on the FTP client side.
Check the functions and operation methods, referring to the manuals on the FTP client side.

(a) FTP server support commands

● binary

[Function]

Notifies that a file is to be transferred to the FTP server without conversion.
Neither return codes nor kanji codes are converted.
This command is set automatically in the Built-in Ethernet port QCPU.

[Specification format] binary (abbreviated format bin)

● bye

[Function]

Disconnects the FTP server connection and ends the FTP function.

[Specification format] bye

[Identical function] quit

● close

[Function] Disconnects the FTP server connection.

[Specification format] close

● delete

[Function] Deletes a file stored in the Built-in Ethernet port QCPU.

[Specification format] delete File path name

[Specification example] When deleting a file stored in the ATA card

delete2:\MAINSEQ1.USR

[Similar function] mdelete

● dir

[Function]

Displays the names, created dates, and sizes of files stored in the Built-in Ethernet port QCPU.

[Specification format] dir [Drive name:\]

[Specification example]

When displaying the detailed information of files stored in the RAM area of the memory card

dir1:\

[Similar function] ls

● get

[Function] Reads a file from the Built-in Ethernet port QCPU.

[Specification format]

get Transfer source file path name [Transfer destination file path name]

[Specification example1]

When reading a file stored in the SRAM card and saving that file under the same file name

get1:\MAINSEQ1.QDR

[Specification example2]

When reading a file stored in the SRAM card and saving that file under a different file name

get1:\SEQ1BAK.QDR\SEQ10LD.QDR

[Note]

When the transfer destination file path name (FTP client side) is not specified, the file is saved on the FTP client side using the same file name as the transfer source file name (Built-in Ethernet port QCPU side).
The transfer destination is on the current directory of which when FTP is started up and connected to the server.

● ls

[Function] Displays the names of files stored in the Built-in Ethernet port QCPU.

[Specification format] ls [Drive name:\]

[Specification example]

When displaying the names of files stored in the SRAM card

ls1:\

[Similar function] dir

● mdelete

[Function]

Deletes a file stored in the Built-in Ethernet port QCPU. To delete multiple files, specify the file names and extensions within the file path name using a wild card (* and/or ?).

[Specification format]

mdelete File path name (abbreviated format: mdel)

[Specification example]

When deleting all files whose extensions are "QPG" among the files stored in the ATA card

mdelete2:*.QPG

[Similar function] delete

- mdir

[Function]

Saves detailed information (file names, created dates, and sizes) of files stored in the Built-in Ethernet port QCPU as log data in a file on the FTP client side.

[Specification format]

```
mdir Transfer source drive name:\ Transfer destination file path name
```

[Specification example]

When saving detailed information of files stored in the SRAM card to the S990901.LOG file

```
mdir1:\S990901.LOG
```

[Note]

- Specify "\" immediately after the transfer source drive name.
- Specify the transfer source drive name when specifying the transfer destination file path name (FTP client side).
- When the transfer destination file path name is not specified, the file is saved using a file name determined by the FTP application on the FTP client side.
- The transfer destination is on the current directory of which when FTP is started up and connected to the server.

[Similar function] mls

- mget

[Function]

Reads a file from the Built-in Ethernet port QCPU.

To read multiple files, specify the file names and extensions within the file path name using a wild card (* and/or ?).

When reading multiple files, receive status is checked for each file transfer.

[Specification format] mget File path name

[Specification example]

When reading all files whose extensions are "USR" among the files stored in the ATA card

```
mget2:\*.USR
```

[Note]

The read file is saved on the FTP client side under the same file name.

The save destination is on the current directory of which when FTP is started up and connected to the server.

● mls

[Function]

Saves the names of files stored in the Built-in Ethernet port QCPU as log data in a file on the FTP client side.

[Specification format]

mls Transfer source drive name:\ Transfer destination path name

[Specification example]

When saving the names of files stored in the SRAM card to the S990901F.LOG file

mls1:\S990901F.LOG

[Note]

- Specify "\" immediately after the transfer source drive name.
- Specify the transfer source drive name when specifying the transfer destination file path name (FTP client side).
- When the transfer destination file path name is not specified, the file is saved using a file name determined by the FTP application on the FTP client side.
- The transfer destination is on the current directory of which when FTP is started up and connected to the server.

[Similar function] mdir

● mput

[Function]

Writes a file to the Built-in Ethernet port QCPU.

To write multiple files, specify the file names and extensions within the file path name using a wild card (* and/or ?).

When writing multiple files, send status is checked for each file transfer.

[Specification format] mput Transfer source file path name

[Specification example] When writing all files whose extensions are "USR"

mput*.USR

[Note]

The save destination file name is the same as that on the FTP client side.

The transfer destination is the memory in which the parameter files used during the current operation are stored.

● open

[Function]

Connects to the FTP server based on a specified host name or IP address and port number on the FTP server side.

[Specification format]

open Host name [Port number]
open IP address [Port number]

Host name: Host name set in Microsoft® Windows® hosts file
IP address: Built-in Ethernet port QCPU IP address
Port number: Port number to be used

[Specification example1]

When connecting to the FTP server by specifying a host name

open HOST

[Specification example2]

When connecting to the FTP server by specifying an IP address

open 192.0.1.254

[Note]

Connection is also possible by specifying an IP address when FTP is started up.

● put

[Function] Writes a file to the Built-in Ethernet port QCPU.

[Specification format]

put Transfer source file path name [Transfer destination file path name]

[Specification example1]

When writing the MAINSEQ1.QDR file to the SRAM card with the same file name

put MAINSEQ1.QDR1:\MAINQDR1.QDR

[Specification example2]

When writing the MAINSEQ.QDR file to the SRAM card with a different file name

put MAINSEQ.QDR1:\MAINQDR1.QDR

[Note]

- When the transfer destination file path name (FTP client side) is not specified, the files on the current directory of which when FTP is started up and connected to the server are written.
- When the transfer destination file path name (FTP server side) is not specified, the files are saved in the memory in which the parameter files used during the current operation are stored.

● pwd

[Function] Displays the current directory name of the Built-in Ethernet port QCPU.

[Specification format] pwd

[Note] "\" is displayed as the execution result of the "pwd" command.

● quit

[Function] Disconnects the FTP server connection and ends the FTP function.

[Specification format] quit

[Identical function] bye

- quote

[Function]

Sends an FTP server subcommand (Built-in Ethernet port QCPU dedicated subcommand).

[Specification format] quote Built-in Ethernet port QCPU dedicated subcommand

[Specification example] quote password-lock

[Note]

Only Built-in Ethernet port QCPU dedicated subcommands can be specified. Refer to (b).

- rename

[Function] Renames a Built-in Ethernet port QCPU file.

[Specification format]

rename Old file path name New file path name (abbreviated format: ren)

[Specification example] When renaming a file stored in the SRAM card

rename 1:\MAINSEQ1.QDR1:\SEQ1OLD.QDR

[Note] The following response codes are displayed upon normal completion.

350 Need more info.
250 Rename successful.

- user

[Function] Inputs the user name and password of the connected FTP server.

[Specification format]

user User name [FTP password]
User name: Login name set in Built-in Ethernet port QCPU parameter
FTP password: FTP password set in Built-in Ethernet port QCPU parameter

[Specification example1] When specifying a user name

user CPU

[Specification example2] When specifying a user name and password

user CPU CPU

(b) **Built-in Ethernet port QCPU dedicated subcommands**

The Built-in Ethernet port QCPU dedicated subcommands added to and sent with the FTP operation "quote" command are described below.

● change

[Function] Displays/changes Built-in Ethernet port QCPU file attribute.

[Specification format1] When displaying file attribute

quote change File path name

One of the following is displayed as the execution result upon normal completion.

- When the specified file is a read-only file : -----R
- When the specified file is a read/write-enabled file : -----W

[Specification format2] When changing file attribute

quote change File path name Attribute

Use one of the following for specifying the attribute.

- To change the file to read-only file : r
- To change the file to a read/write-enabled disk file : w

[Specification example1]

When displaying file attribute stored in the SRAM card

quote change 1:\MAINSEQ1.QDR

[Specification example2]

When changing file attribute stored in the SRAM card

quote change 1:\MAINSEQ1.QDR r

● keyword-set

[Function]

Sets a file password registered in the file transfer target file into the Built-in Ethernet port QCPU.*1

Or, displays/clears the password set for FTP settings in parameter.

* 1 : This command is used only when a file password is registered in the file transfer target file. When accessing the specified file, the Built-in Ethernet port QCPU checks the file password.

[Specification format] quote keyword-set [File password]

File password:

Specify the file password registered in the Built-in Ethernet port QCPU file.

To clear the set file password, specify "****".

One of the following is displayed as the execution result upon normal completion.

- When setting a file password: 200 Command successful
- When displaying a file password: 200 Keyword is "File password"
- When clearing a file password: 200 Command successful

[Specification example1] When setting the password (1234)

quote keyword-set 1234

[Specification example2]

When displaying the password currently set for FTP settings in parameter

quote keyword-set

[Specification example3]

When clearing the password currently set for FTP settings in parameter

quote keyword-set ****

[Note]

- One file password can be set for the FTP settings in Built-in Ethernet port QCPU parameter.
  When the file targeted for file transfer changes, re-set the file password of the target file when a file password is also registered for the change target file.
- When logging in to the Built-in Ethernet port QCPU, the file password is initialized (cleared) to "****".

● password-unlock

[Function]

Specifies the remote password set in the Built-in Ethernet port QCPU and performs unlock processing.[*2]

[Specification format] quote password-unlock [Remote password]

Remote password:

Specify the remote password set in Built-in Ethernet port QCPU parameter.

- The following is displayed as the execution result upon normal completion.
  200 Command 0key
- The following is displayed when the entered remote password and set remote password do not match.
  556 Password Error
- The following is displayed when another command is requested before unlock processing of the remote password is performed.
  555 Password Locked

[Specification example] When specifying the remote password (1234)

```
quote password-unlock 1234
```

[Note]

- The remote password is locked when logged in to the Built-in Ethernet port QCPU if the FTP communication port is specified for a remote password check target.
- Performing unlock processing by executing this command before the various FTP operations enables file operations in the Built-in Ethernet port QCPU.
- Processing completes normally if the remote password unlock processing is performed when the FTP communication port is not specified for a remote password check target.

● password-lock

[Function]

Performs lock processing of the remote password set in the Built-in Ethernet port QCPU.[*3]

[Specification format] quote password-lock

The following is displayed as the execution result upon normal completion.

200 Command 0key

[Specification example] When locking the remote password

```
quote password-lock
```

[*2] This command is used only when FTP communication port is specified for a remote password check target.

[*3] This command is used only when FTP communication port is specified for a remote password check target.

● run

[Function]

Changes the Built-in Ethernet port QCPU status to RUN. (Executes remote RUN.) When changing the QnUDE(H)CPU status to RUN, device memory clear can be specified.

[Specification format] quote run [Mode [Clear mode]]

Mode: Specify whether to execute remote RUN forcibly.

0 : Normal RUN (default)
1 : Forced RUN

Clear mode:

Specify the clear (initialization) processing of the Built-in Ethernet port QCPU device memory when an operation starts by executing remote RUN.

0 : Do not clear device (default)
1 : Clear all devices excluding latch range
2 : Clear all devices including latch range

The following message is displayed as the execution result upon normal completion.

Displayed message: 200 Command successful

[Specification example1]

When executing remote RUN, specifying "Normal RUN" and "Do not clear device"

```
quote run
```

[Specification example2]

When executing remote RUN, specifying "Forced RUN" and "Do not clear device"

```
quote run 1
```

[Specification example3]

When executing remote RUN, specifying "Forced RUN" and "Clear all devices excluding latch range"

```
quote run 1 1
```

[Note]

- Forced RUN mode is used to forcibly execute remote RUN to the Built-in Ethernet port QCPU from other devices when a trouble occurs in the device which executed remote STOP to the Built-in Ethernet port QCPU (the device can no longer execute remote RUN to the Built-in Ethernet port QCPU).
  When Normal RUN mode is specified, the status will not change to RUN if the Built-in Ethernet port QCPU is already set in the STOP/PAUSE status by another device.
- Specify the clear mode at the start of an operation based on the system arrangement.
  After the specified clear processing is performed, the Built-in Ethernet port QCPU changes to the RUN status in accordance with the parameter settings ("Initial Device value" setting on the PLC file tab of PLC parameter).

- status

  [Function]

  Displays the operation information of the Built-in Ethernet port QCPU. This command checks the Built-in Ethernet port QCPU operation information when executing file transfer to the Built-in Ethernet port QCPU.

  [Specification format] quote status

  One of the following is displayed as the execution result upon normal completion.

  • When the Built-in Ethernet port QCPU is in the RUN status: "RUN"
  • When the Built-in Ethernet port QCPU is in the STOP status: "STOP"
  • When the Built-in Ethernet port QCPU is in the PAUSE status: "PAUSE"

- stop

  [Function]

  Changes the Built-in Ethernet port QCPU status to STOP. (Executes remote STOP.)

  [Specification format] quote stop

  The following message is displayed as the execution result upon normal completion.

  Displayed message: 200 Command successful

  [Note]

  Write files to the program memory after the Built-in Ethernet port QCPU status is changed to STOP using this command.

- pm-write

  [Function]

  Transfers files in the program cache memory to the program memory.

  [Specification format] quote pm-write

  The following message is displayed as the execution result upon normal completion.

  Displayed message: 200 Command successful

  [Note]

  Write files after the Built-in Ethernet port QCPU status is changed to STOP.

(6) Precautions

Precautions for using the file transfer function are described below.

(a) FTP clients

- FTP command specifications may differ from those described in this manual, depending on the FTP client. In such a case, check the functions and operation methods, referring to the manuals on the FTP client side.
- FTP operations cannot be performed from Microsoft® Internet Explorer. If performed, an error occurs with Internet Explorer.

(b) Processing in Built-in Ethernet port QCPU

- Files only in the drives of the host station Built-in Ethernet port QCPU can be accessed.
- During file access, do not turn OFF the power supply, reset the CPU module or install/remove a memory card. If any of these operations are performed, files may corrupt.
- During file access, do not perform any file operations from peripherals such as GX Developer. When a file operation is performed from another peripheral during FTP function operation, an error may occur on the peripheral side.
- During online change, do not perform any file operations using the FTP function. When a file operation is performed, an error may occur.
- During file access using the FTP function, do not perform any online operations from peripherals such as GX Developer. When an online operation is performed, an error may occur on the peripheral side.

(c) Communication processing

- When a time-out error occurs during file transfer, the TCP connection will be closed (disconnected). To restart the file transfer, log in to the Built-in Ethernet port QCPU once again from an FTP client.
- In a communication using FTP connections, an existence check of external devices is performed. For operation details, refer to Section 7.2 (3).
- The file transfer processing time depends on the following factors.
  1) Load rate of the Ethernet line (line traffic)
  2) Number of connections simultaneously used
     (communication processing of other connections)
  3) System configuration
- The number of external devices (FTP clients) that can be simultaneously logged in to the Built-in Ethernet port QCPU is only one. When another FTP client attempts the connection to the Built-in Ethernet port QCPU which is in the logged-in status, connection cannot be established and an error occurs.
- If the File transfer function using the FTP is executed simultaneously with other communication functions (MELSOFT connection, MC protocol) using the UDP, an error, such as a time-out error, may occur. Execute other communication functions after file transfer is completed or communicate using the TCP.

(d) **Precautions for writing files**

- Operation to overwrite an existing file cannot be performed. Either delete the file using a file delete command (delete, mdelete) or rename the file using a file name change command (rename) before writing the file.
- Files whose attribute is read-only or files which are locked by other devices/ functions cannot be written. If files are written, a write error occurs.
- File transfer (writing files) cannot be executed when a write-protected memory card is used. If files are written, a write error occurs.
- When a file is written, a temporary file (FTP_I***.TMP) is created automatically. The name of this file is changed to the name of written file upon normal completion. However, if the power failure or reset of the CPU module occurs during the file write, this temporary file may remain. If this occurs, delete the temporary file.
- When file registers are used in the standard RAM or SRAM card, change the CPU status to STOP first, and then write /delete files to/in these drives.
- When file registers used in the standard RAM or SRAM card are set as auto refresh devices, do not write/delete files to/in these drives.
- If large-capacity files are written to the ATA card during RUN, a load on write processing increases and a communication error may occur. Change the CPU status to STOP to write large-capacity files.

(e) **Precautions for deleting files**

- The timing for deleting files is to be arranged by the user considering the overall system, including Built-in Ethernet port QCPU and GX Developer.
- Files whose attribute is read-only or files which are locked by other devices/ functions cannot be deleted. If files are deleted, a delete error occurs.
- Deleting files cannot be executed when a write-protected memory card is used. If files are deleted, a delete error occurs.

7.7 Remote Password

This function checks a remote password when the communication using the following connections is requested.

- Communication using GX Developer
- Communication using the MC protocol
- File transfer

☒POINT

The remote password function is a function to prevent unauthorized access (ex. corrupting data and programs) from external devices. However, this function cannot completely prevent unauthorized access.
Users should incorporate safety measures by themselves if security of the programmable controller system needs to be maintained against unauthorized access from external devices. We cannot be held responsibility for any problems caused by unauthorized access.

[Examples of measures against unauthorized access]

- Install a firewall.
- Set up a personal computer as a relay station and control the relay of communication data using an application program.
- Set up an external device that control access authority as a relay station.
- Please consult a network provider or equipment sales company regarding external devices that can control access authority.

(1) Communication method when a remote password is set

The communication procedure with the Built-in Ethernet port QCPU with a preset remote password is described below.

(a) Access enable processing (unlock processing)

An external device such as a personal computer performs remote password unlock processing for the CPU module.

When unlock processing is not performed, an error occurs in the external device since the CPU module does not accept access.

(b) Access processing

Access is enabled after the remote password unlock processing is completed normally.

(c) Access disable processing (lock processing)

Remote password lock processing is performed from the external device to disable subsequent access after the access is completed.

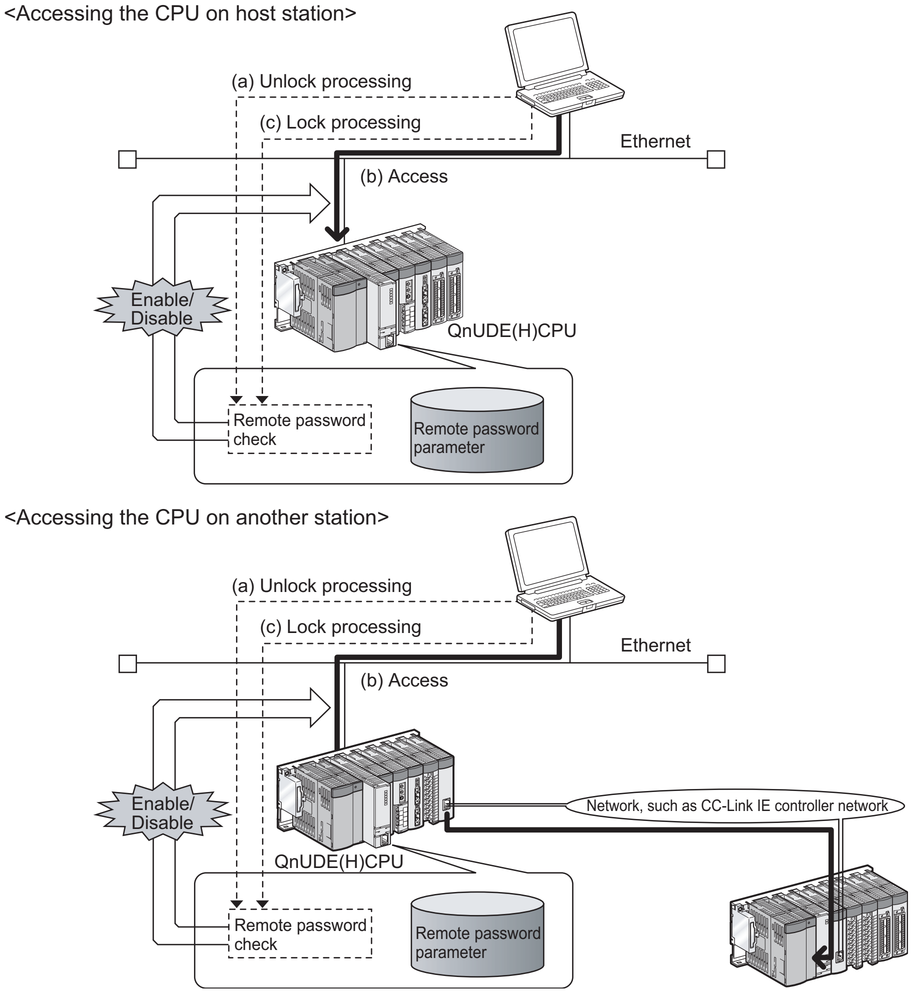


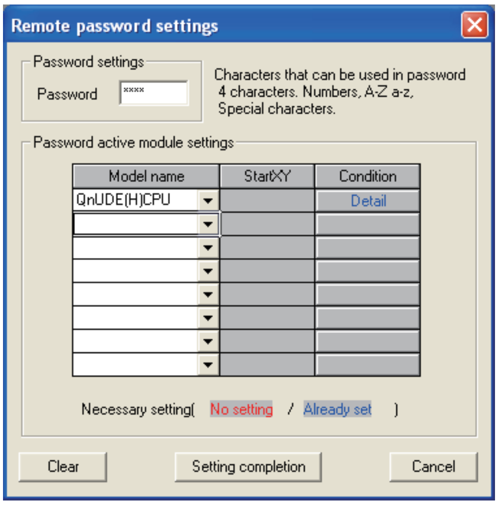
Figure 7.28 Remote password operation overview

(2) Remote password setting

Setting method for remote password is described below.

(a) Setting remote password and writing parameters to CPU module

Set the remote password and the connection target in GX Developer and write the parameter settings to the CPU module.



Remote password settings

Password settings

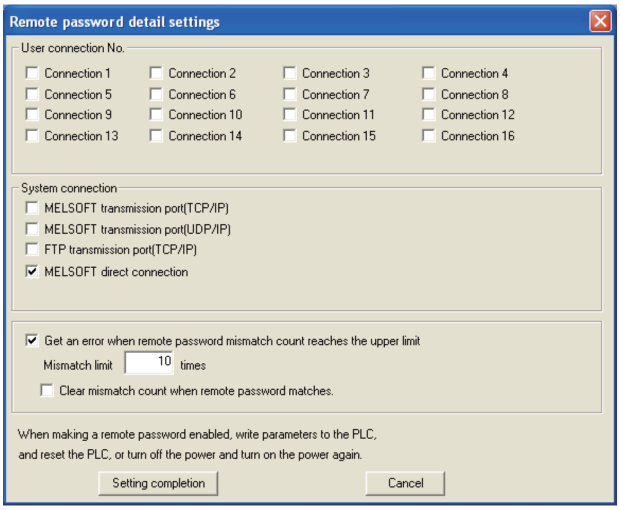
Password ****

Characters that can be used in password 4 characters. Numbers, A-Z a-z, Special characters.

Password active module settings

Model name	StartXY	Condition
QnUDE(H)CPU		Detail

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

System connection

□ MELSOFT transmission port(TCP/IP)
□ MELSOFT transmission port(UDP/IP)
□ FTP transmission port(TCP/IP)
☑ MELSOFT direct connection

☑ Get an error when remote password mismatch count reaches the upper limit

Mismatch limit 10 times

□ Clear mismatch count when remote password matches.

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 7.29 Screens for setting remote password

Table7.10 Setting items on the screens for setting remote password

Item		Description	Setting range
Password settings		Enter the remote password to be set in the QCPU.[*1]	Up to four one-byte characters
Password active module settings	Model name	Select the CPU model to enable the remote password for built-in Ethernet ports of the QCPU module. Only one CPU model is available.	QnUDE(H)CPU
	Condition	Click this to display the Remote password detail settings screen.	Detail
User connection No.	Connection 1 to 16	Select the connection port for which the remote password is to be enabled. (Setting to the unused connection or to the MELSOFT connection is ignored.)	Check the checkbox of target connection.
System connection	MELSOFT transmission port (TCP/IP) [*2]	Select the connection port for which the remote password is to be enabled.	
	MELSOFT transmission port (UDP/IP) [*2*3]		
	FTP transmission port (TCP/IP)		
	MELSOFT direct connection		
Get an error when remote password mismatch count reaches the upper limit		Check the checkbox to perform this operation. (To detect unauthorized access.)	
Mismatch limit		Set the number of mismatch limits.	1 to 65535
Clear mismatch count when remote password matches		Check the checkbox to perform this operation. (To ignore the entry error.)	Check the checkbox of target connection.

* 1 : Characters that can be used for remote passwords are one-byte alphanumeric characters and one-byte special characters. (Case-sensitive)
Refer to the following when setting a remote password.
- Avoid using character strings having simple numbers or letters only.
- Combine numbers, letters, and special characters.
- Avoid using character strings, including user's name or date of birth.

* 2 : To enable the remote password for the port for which the open system is set to "MELSOFT connection" in PLC parameter, check the checkbox as follows.
- When protocol is set to "TCP", check "MELSOFT transmission port (TCP/IP)".
- When protocol is set to "UDP", check "MELSOFT transmission port (UDP/IP)".

* 3 : When connecting GOT with the Built-in Ethernet port QCPU via Ethernet, do not check "MELSOFT transmission port (UDP/IP)".

(b) **Enabling parameter settings**

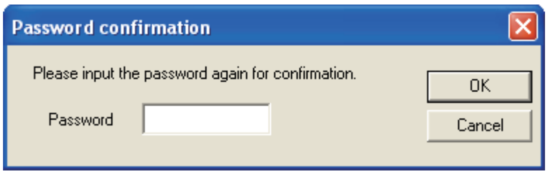
After writing parameter settings to the CPU module, settings are enabled by powering the programmable controller ON or resetting the CPU module.

(c) **Performing remote password unlock/lock processing**

Perform the remote password unlock/lock processing from an external device using respective protocol as follows.

1) **Using MELSOFT connection**

Enter the remote password in the following dialog box that appears during communication.
GX Developer performs unlock processing upon entering the remote password and then access the CPU module.


Password confirmation

Please input the password again for confirmation.

Password

Figure 7.30 Password confirmation screen

2) **Using MC protocol**

Use the dedicated commands for the MC protocol.
For details on the MC protocol dedicated commands, refer to the following manual.
☞ Q Corresponding MELSEC Communication Protocol Reference Manual

3) **Using file transfer (FTP server) function**

Use the "password-lock"/"unlock" commands. (☞ Section 7.6 (5))

(3) Precautions for remote password

Precautions for the remote password check function are described below.

(a) **Setting remote password to UDP connections**

- Determine the communication target external device and do not communicate with unspecified external devices. (The communication target external device needs to be determined since UDP/IP connections can communicate with any other devices, including the device that has performed unlock processing, after the remote password unlocking processing is completed normally.)
- At the end of data communication, always perform the remote password lock processing. (If lock processing is not performed, unlock status is held until a time-out occurs. When there is no communication for 10 minutes, the CPU module automatically performs time-out based lock processing.)

To prevent unauthorized access more strictly using the remote password, it is recommended to set all connection protocols to TCP/IP and disable GX Developer direct connection in parameter.

(b) **Performing close processing before lock processing for TCP/IP connections**

When close processing is performed before lock processing for TCP/IP connections, the CPU module automatically performs lock processing.

(c) Remote password valid range

The remote password is valid only for access from the Built-in Ethernet port QCPU for which the parameter settings were made.

When multiple CPU modules are used in a multiple CPU system, set a remote password for each setting target CPU module respectively.

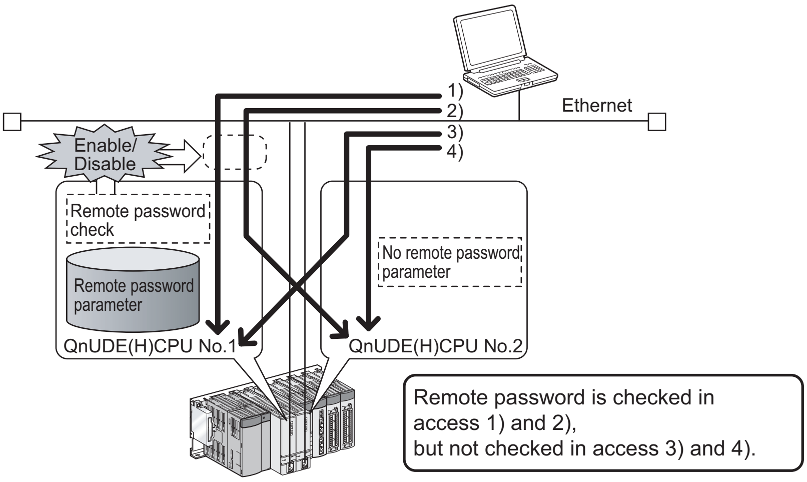

1)
2)
3)
4)
Ethernet
Enable/
Disable
Remote password check
Remote password parameter
No remote password parameter
QnUDE(H)CPU No.1
QnUDE(H)CPU No.2
Remote password is checked in access 1) and 2), but not checked in access 3) and 4).

Figure 7.31 Remote password in multiple CPU system

(d) When remote password mismatch count is set

If the checkbox of "Get an error when remote password mismatch count reaches the upper limit" is checked on the Remote password detail settings screen in GX Developer, an error may occur due to entry errors accumulated in a long period of time by the user.

To prevent such errors, perform the following operations.

- Check the checkbox of "Clear mismatch count when remote password matches" on the Remote password detail settings screen in GX Developer.
- Clear the accumulated count of remote password mismatches using the special relay (SM1273).

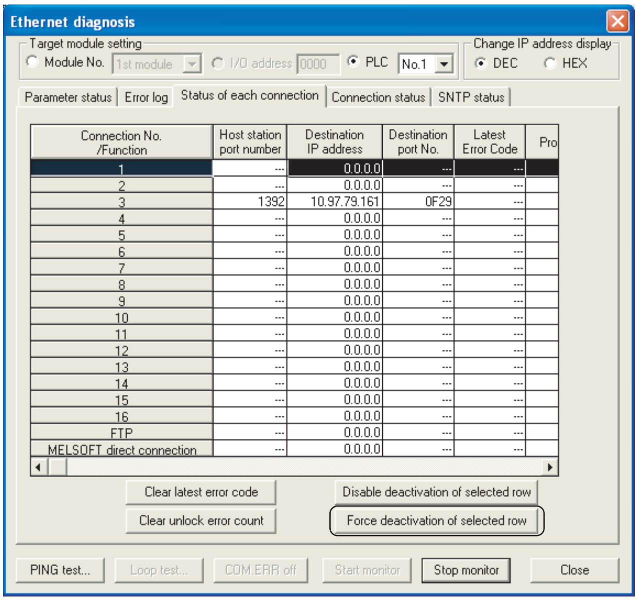
(4) When remote password mismatch count reaches the upper limit during unlock processing

When remote password mismatch count reaches the upper limit during unlock processing, an error (error code: 2700) occurs in the Built-in Ethernet port QCPU. The error cause is considered to be unauthorized access from outside of the system. Perform the following corrective action as needed.

1) Monitor the Remote password count (SD979 to SD999) and check the connection that remote mismatch count reaches the upper limit during unlock processing.

2) Disable the communication by making the corresponding connection invalid. The corresponding connection can be set to invalid by the operation described below.

- By selecting the target connection on the Status of each connection tab, which is displayed by clicking the "Built-in Ethernet port diagnostics" button on the PLC diagnostics screen in GX Developer. (Refer to Figure 7.32.)



Ethernet diagnosis

Target module setting: ○ Module No. 1st module ○ I/O address 0000 ● PLC No.1

Change IP address display: ● DEC ○ HEX

Parameter status | Error log | Status of each connection | Connection status | SNTP status

Connection No. /Function	Host station port number	Destination IP address	Destination port No.	Latest Error Code	Pro
1	---	0.0.0.0	---	---	
2	---	0.0.0.0	---	---	
3	1392	10.97.79.161	0F29	---	
4	---	0.0.0.0	---	---	
5	---	0.0.0.0	---	---	
6	---	0.0.0.0	---	---	
7	---	0.0.0.0	---	---	
8	---	0.0.0.0	---	---	
9	---	0.0.0.0	---	---	
10	---	0.0.0.0	---	---	
11	---	0.0.0.0	---	---	
12	---	0.0.0.0	---	---	
13	---	0.0.0.0	---	---	
14	---	0.0.0.0	---	---	
15	---	0.0.0.0	---	---	
16	---	0.0.0.0	---	---	
FTP	---	0.0.0.0	---	---	
MELSOFT direct connection	---	0.0.0.0	---	---	

Figure 7.32 Ethernet diagnosis screen

- By turning ON the forced invalidation flag (bit) of the corresponding connection using the special registers (SD1276, SD1277).

☒POINT

For details on the Ethernet diagnostics of the Built-in Ethernet port QCPU, refer to the following manual.

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

3) Clear the error (error code: 2700).
The remote password count (SD979 to SD999) is cleared in accordance with clearing of the error.

4) Inform the system manager that the number of error completions of unlock processing is exceeding the mismatch limit and take appropriate corrective actions.

CHAPTER8 COMMUNICATION WITH INTELLIGENT FUNCTION MODULE

(1) Description of intelligent function modules

The intelligent function module is a module that allows the CPU module to process analog values or high speed pulses which cannot be processed with I/O modules.
For example, an analog value is converted into a digital value with the analog/digital conversion module, one of the intelligent function modules, before being used.

(2) Communication with intelligent function modules

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

8.1 Communication Between CPU Module and Intelligent Function Modules

The following methods enable the communication between the CPU module and intelligent function modules:

- Initial setting or automatic refresh setting using GX Configurator [1] (☞ Section 8.1.1)
- Initial setting by device initial value (☞ Section 8.1.2)
- FROM/TO instruction (☞ Section 8.1.3)
- Intelligent function module device (☞ Section 8.1.4)
- Instructions dedicated for intelligent function modules[1] (☞ Section 8.1.5)

[1] : Not supported by the AnS/A series corresponding special function modules.

Table8.1 shows the communication timing for the communication methods with intelligent function modules described above:

Table8.1 Timing of communication with intelligent function modules

Communication method with intelligent function modules		Communication timing				
		Power ON	High Performance model QCPU reset	STOP ➡ RUN [3]	Instruction execution	END processing
GX Configurator [1]	Initial setting	○	○	○	--	--
	Automatic refresh setting	--	--	--	--	○
Device initial value		○	○	○	--	--
FROM/TO instruction [2]		--	--	--	○	--
Intelligent function module device [2]		--	--	--	○	--
Instructions dedicated for intelligent function modules [1][2]		--	--	--	○	--

○ : Executed -- : Not executed

[1] : Not supported by the AnS/A series compatible special function modules.

[2] : Represents the program using the intelligent function module device, the FROM/TO instruction, or the instructions dedicated for intelligent function modules.

[3] : Indicates the case where the RUN/STOP switch (RUN/STOP/RESET switch of the Basic model QCPU, Universal model QCPU) is set from STOP to RUN (RUN LED flickers) to STOP to RUN.

☒POINT

1. The data used for communication with the intelligent function modules (initial setting, etc.) are stored into the CPU module.
   Refer to Section 5.1.1 and Section 5.2.1 for the storage destination in the CPU module.
2. To enable the initial settings made by GX Configurator to be ensured by powering the PLC OFF and then ON or resetting the CPU module, set the RUN/STOP switch (RUN/STOP/RESET switch of the Basic model QCPU, Universal model QCPU) to RUN, and then power the PLC OFF and then ON or reset the CPU module.

8.1.1 Initial setting and auto refresh setting by GX Configurator

The initial setting and auto refresh setting of the intelligent function modules can be performed by adding in intelligent function module-compatible GX Configurator to GX Developer.
Making initial setting and auto refresh setting of the intelligent function modules on GX Configurator enables write/read of data without creation of a program for communication with the intelligent function modules.

(1) Starting GX Configurator

To start GX Configurator, choose [Tools] → [Intelligent function module utility] → [Start] on GX Developer.

(2) Setting using the GX Configurator

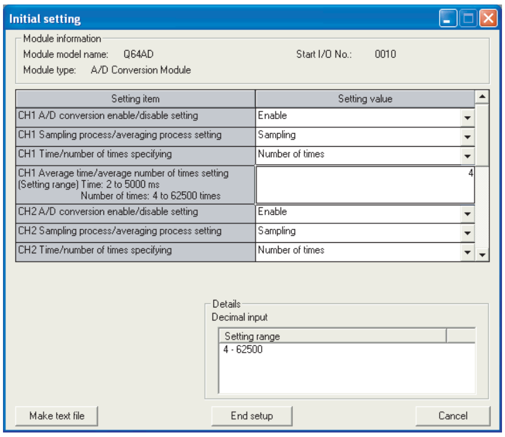
This section describes the example to set the initial setting and automatic refresh setting of A/D conversion module Q64AD.

(a) Initial setting

The initial setting of Q64AD offers the following four settings:

- A/D conversion enable/disable setting
- Sampling process/averaging process setting
- Time/number of times specifying
- Average time/average number of times setting

The initial setting of Q64AD is designated on the following initial setting screen of GX Configurator (Figure 8.1).



Initial setting

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 8.1 Initial setting screen

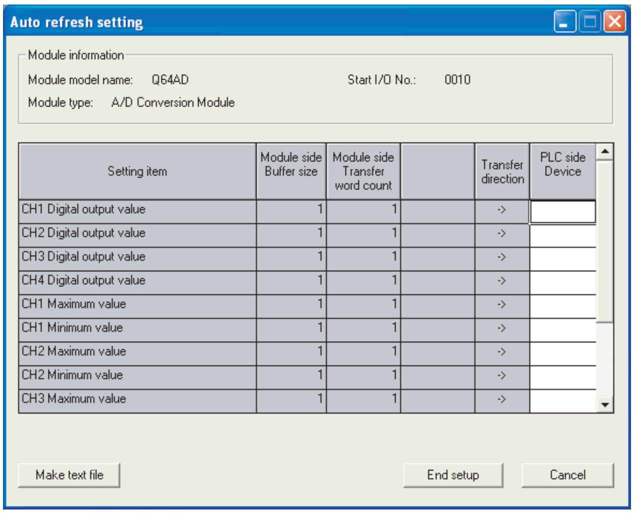
The set initial setting data are stored into the intelligent function module parameters of the CPU module.

(b) Auto refresh setting

For the auto refresh setting, designate the device at the CPU module to store the following data.

- Digital output of Q64AD
- Maximum/minimum values of Q64AD
- Error code

The auto refresh setting of Q64AD is designated on the following auto refresh setting screen of GX Configurator (Figure 8.2).



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 8.2 Auto refresh setting screen

The designated auto refresh setting data is stored in the intelligent function parameters of the CPU module.

(3) Restrictions on the number of parameter settings

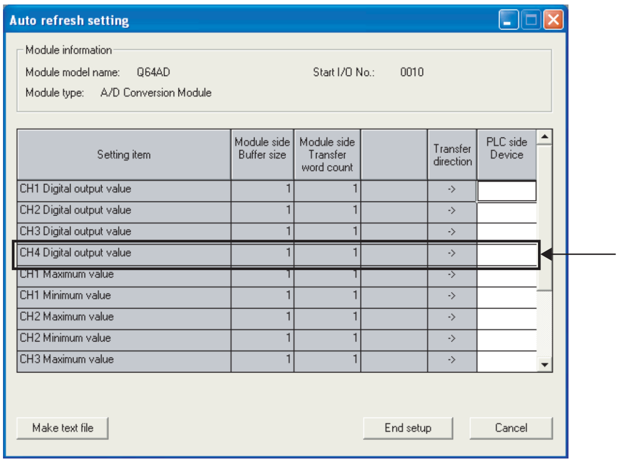
There are restrictions on the number of parameter (initial setting or auto refresh setting) set by GX Configurator.

When mounting the multiple intelligent function modules, make the setting of the GX Configurator so that the total number of parameter setting of intelligent function modules does not exceed the maximum number of parameter settings shown in Table8.2.

Table8.2 Number of parameter settings that can be set by GX Configurator

CPU module	Number of parameter settings	
	Initial setting	Auto refresh setting
Q00JCPU, Q00CPU, Q01CPU, Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU, MELSECNET/H remote I/O station	512	256
Q02UCPU	2048	1024
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	4096	2048

<<Concept of number of parameter settings>>



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 setting. Blanks are not counted. All the setting items in this setting screen are added, which is summed up with the number of other intelligent function modules.

Figure 8.3 Concept of number of parameter settings

(4) Precautions

The AnS/A series corresponding special function module does not support the settings made by GX Configurator.

Remark

For the details of GX Configurator, refer to the manual of the intelligent function module being used.

8.1.2 Initial setting by device initial value

(1) Device initial value

Using the device initial values, the initial setting of the intelligent function module can be made without a program. (☞ Section 6.26)

The set device initial values are written from the CPU module to the intelligent function module when the PLC is powered OFF and then ON or the CPU module is reset or switched from STOP to RUN.

(2) Designation of the device initial value

Make the following setting using GX Developer.

- Set to the device memory the intelligent function module data to be used as the device initial values. (☞ Section 10.5)
- In the device initial value setting, specify the range of the intelligent function module device to be used as the device initial values.

8.1.3 Communication by FROM/TO instruction

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

The TO instruction writes the data of the designated device to the buffer memory of the intelligent function module.

Remark

- For the details of the FROM/TO instruction, refer to the following manuals.

  ☞ QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)
- For the details of the buffer memory of the intelligent function module, refer to the manual of the intelligent function module being used.

8.1.4 Communication by intelligent function module device

(1) Intelligent function module device

The intelligent function module device is the buffer memory of the intelligent function module represented as a device of the CPU module program. (☞ Section 10.5)
The data stored in the intelligent function module buffer memory can be handled by the sequence instruction like the device memory.
For example, programming can be performed as shown below when "100" is written to the buffer memory address 0 of the intelligent function module whose I/O numbers are X/Y20 to X/Y2F.

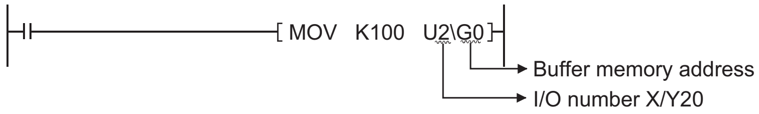


Figure 8.4 How to use the intelligent function module device

(2) Difference from the FROM/TO instruction

The intelligent function module device can be handled as a device of the CPU module, enabling the processing of data read from the intelligent function module with one instruction.
For example, the following programming will be processed as the same as when reading and adding data from the intelligent function module and storing the result into D2.

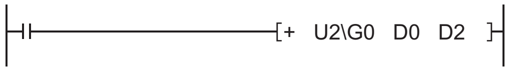


Figure 8.5 Application example of the intelligent function module device

This saves the number of steps in the entire program.
The instruction processing time is the time obtained by adding the time for access to/ from the intelligent function modules to the instruction execution time.

☒POINT

The intelligent function module device accesses the intelligent function module every time the instruction is executed.
When writing/reading buffer memory data using multiple intelligent function module devices in a sequence program, make sure to write/read the data in one position of the program using the FROM/TO instruction.

When writing using multiple intelligent function module devices.

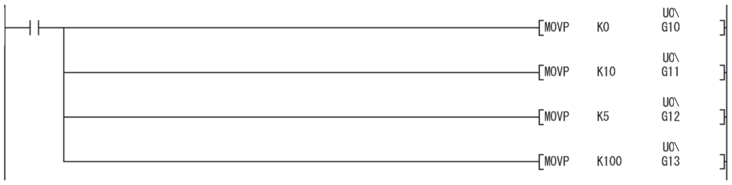


Figure 8.6 When writing using multiple intelligent function module devices

When writing using the TO instruction.

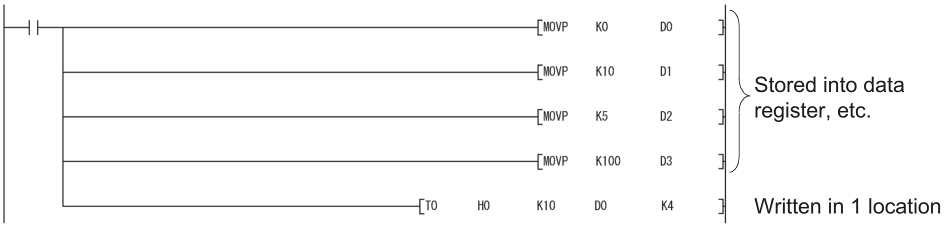


Figure 8.7 When writing using the TO instruction

8.1.5 Communication by instructions for Intelli. function modules

(1) Description of the instructions dedicated for intelligent function modules

The instructions dedicated for intelligent function modules are the instructions that facilitate programming using the functions of the intelligent function modules.

(a) Example of serial communication module dedicated instruction (OUTPUT instruction)

Use of the OUTPUT instruction allows communication with the external device by the no-procedure protocol without being conscious of the buffer memory address of the serial communication module.

[Transmission by OUTPUT instruction]

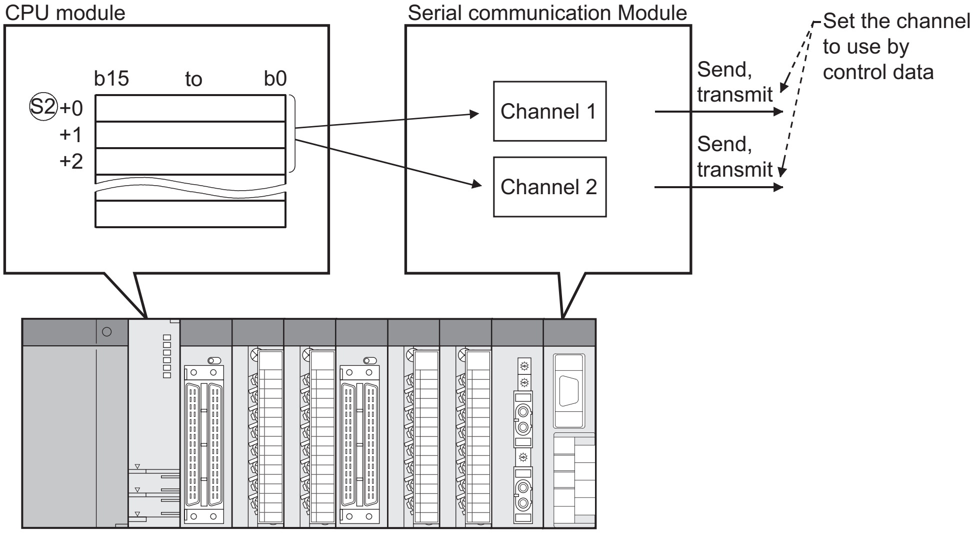


Figure 8.8 Communication with external device using OUTPUT instruction

(2) Processing of intelligent function module dedicated instruction

Some intelligent function module dedicated instructions allow the completion device to be specified.

This completion device turns ON in one scan when the execution of the instruction is completed.

To use two or more instructions dedicated for intelligent function modules to one intelligent function module, be sure to execute the next instruction dedicated for intelligent function modules after the completion device turns ON.

(3) Precautions

(a) When RUN is switched to STOP before completion device turns ON

If the instruction dedicated for intelligent function modules are executed and the CPU module is switched from RUN to STOP before the completion device turns ON, the completion device turns ON one scan later when the CPU module is switched to RUN next time.

(b) Available range

The intelligent function module dedicated instructions cannot be executed for the intelligent function modules mounted on the remote I/O stations of MELSECNET/H.

They can be executed for the intelligent function modules mounted on the main base unit, slim main base unit and extension base units. (Except the AnS/A series corresponding special function modules)

Remark

For the instruction dedicated for intelligent function modules and the completion device, refer to the manual of the intelligent function module being used.

8.2 Access to AnS/A Series Corresponding Special Function Module

Basic Note8.1
Process Note8.1
Redundant Note8.1
Universal Note8.1

(1) Effects of quicker access to the special function module

The Q series CPU module can process at higher speed, taking shorter time to scan. When the FROM/TO instruction is frequently executed for the special function module during a short scan, the processing of the target special function module may not be normally terminated.

In this case, match the FROM/TO instruction execution interval to the processing time/conversion time using the timer, constant scan, etc. of the special function module.

(☞ (2) below)

(2) Countermeasures against the effects of quicker access to the special function module

To use the AnS/A compatible special function module, adjust the execution time with SM415 (2n-ms clock) and SD415 (2n-ms clock setting).

The initial value of SD415 is "30." When SD415 is used for interlock of the FROM/TO instruction, the FROM/TO instruction is executed at every 120 ms.

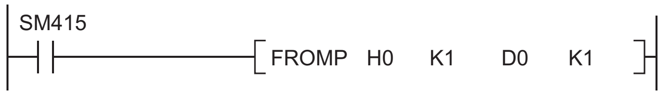


Figure 8.9 Countermeasure program example

POINT

To change the clock value of SM415, store the new value in SD415.
For details of SM415, refer to Appendix 1. For details of SD415, refer to Appendix 2.

Basic Note8.1 Process Note8.1 Redundant Note8.1
Universal Note8.1

Since the Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the AnS/A series correponding special function modules, it is not necessary to be conscious of the description in this section.

CHAPTER9 PARAMETERS

This chapter explains the parameters that are set when a PLC system is configured.

(1) Parameter types

There are the following CPU module parameters.

- PLC parameter (☞ Section 9.1)
  Set when the PLC is used independently.
- Redundant CPU parameter (☞ Section 9.2)
  Set when a redundant system is configured using the Redundant CPU.
- Network parameter (☞ Section 9.3)
  The network parameter is set to use the CC-Link IE controller network module, MELSECNET/H module, Ethernet module, or CC-Link module in combination with the PLC.
- Remote password (☞ Section 9.4)
  Set when the remote password function of the Ethernet, serial communication and modem interface modules is used.

(2) Parameter setting method

Set the PLC parameters, redundant parameters, network parameters and remote password parameters by GX Developer.
Refer to the following manual for the setting operation on GX Developer.

☞ GX Developer Version 8 Operating Manual

☒POINT

In GX Developer, since the functions are not available to the CPU module being used, it is not necessary to set the setting items displayed in gray (cannot be selected) that are not explained in this section.

Remark

- When an error occurs in the parameter setting, the corresponding parameter number indicated in the tables of this chapter is stored in the special register (SD16 to 26).
- Refer to Appendix 2 for details of the special register (SD16 to 26).
- Identify the parameter error location from the parameter number.
  Refer to Appendix 3 for the list of parameter numbers.
- Refer to CHAPTER 12 for the parameter reflection procedure.

9.1 PLC Parameters

This section explains the PLC parameter list and parameter details.

9.1.1 Basic model QCPU

(1) PLC name

Set the label and comment of the used CPU module.

Setting the label and comment in the PLC name does not affect the actual operation.

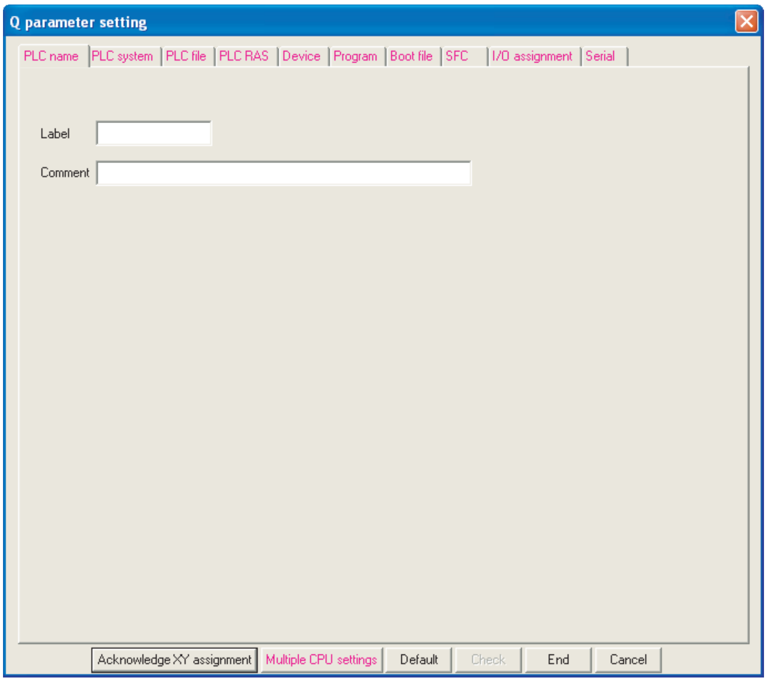


Figure 9.1 PLC name

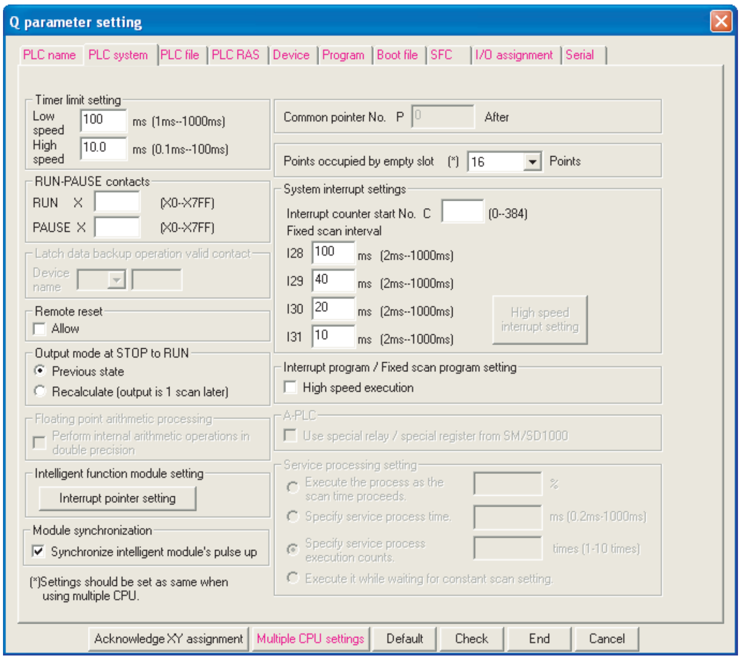
Table9.1 PLC name list

Item	Parameter No.	Description	Setting range	Default value	Reference
Label	0000_H	Set the label (name, application) of the CPU module.	Max. 10 characters	No setting	----
Comment	0001_H	Set the comment of the CPU module label.	Max. 64 characters	No setting	----

(2) PLC system

Make the settings necessary to use the CPU module.

The parameters may be the default values to perform control.



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

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
I28 100 ms (2ms--1000ms)
I29 40 ms (2ms--1000ms)
I30 20 ms (2ms--1000ms)
I31 10 ms (2ms--1000ms)

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.

Figure 9.2 PLC system

Table9.2 PLC system setting list

Item		Parameter No.	Description	Setting range	Default value	Reference
Timer limit setting	Low speed	1000_H	Set the time limit of the low speed timer/high speed timer.	1ms to 1000ms (1ms unit)	100ms	Section 10.2.10
	High speed			0.1ms to 100.0ms (0.1ms unit)	10.0ms	Section 10.2.10
RUN-PAUSE contact	RUN X	1001_H	Set the contact that controls RUN/PAUSE of the CPU module. Setting of only the PAUSE contact is not allowed. (The RUN contact or RUN contact and PAUSE contact can be set.)	X0 to 7FF	No setting	Section 6.6.1
	PAUSE X					Section 6.6.2
Remote reset		1002_H	Set enable/disable of remote reset operation from GX Developer.	Enable/Disable	Disable	Section 6.6.3
Output mode at STOP to Run		1003_H	Set the output (Y) status when the STOP status is switched to the RUN status.	Provide output (Y) status before STOP/Clear output (Y) (output one scan later)	Provide output (Y) status before STOP	Section 6.4
Intelligent function module setting (interrupt pointer setting)		$100A_H$	Set the assignment of the interrupt pointers (I50 to 127) and the start I/O No. and start SI No. of the intelligent function module.	Start I/O No. Start SI No. I50 to 127	No setting	Section 10.11
Points occupied by empty slot		1007_H	Set the number of empty slots on the main base unit/extension base unit.	[Q00JCPU] 0 points/16 points/32 points/64 points/128 points/256 points [Q00CPU, Q01CPU] 0 points/16 points/32 points/64 points/128 points/256 points/512 points/1024 points	16 points	Section 4.6.1(5)

(Continued on next page)

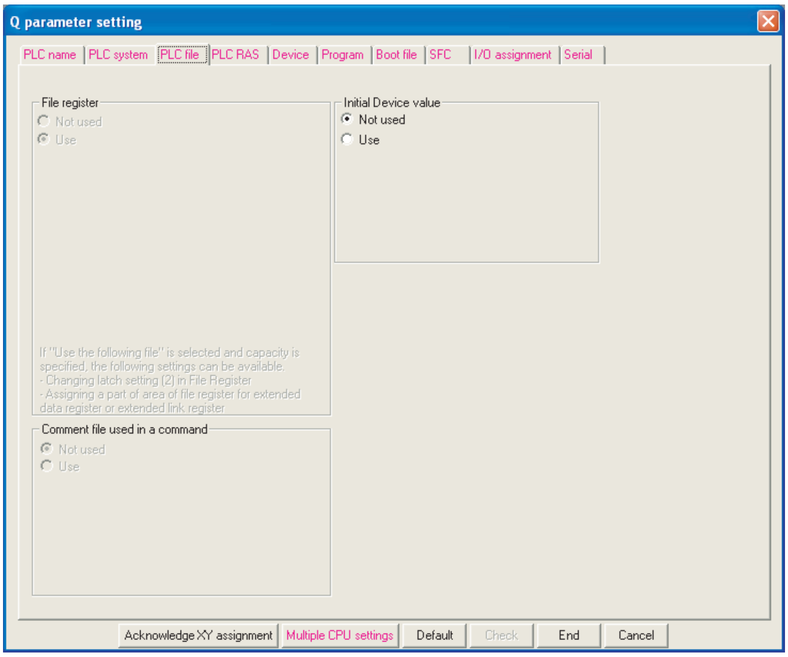
Table9.2 PLC system setting list (Continued)

Item		Parameter No.	Description	Setting range	Default value	Reference
System interrupt setting	Interrupt counter start No.	1008H	Set the start No. of the interrupt counters.	C0 to 13184 (Can be set up to the number of counter setting points - 128)	No setting	Section 10.2.11(4)
	Fixed scan interval (n : 28 to 31)		Set the execution intervals of the interrupt pointers (I28 to 31).	2ms to 1000ms (1ms unit)	I28 : 100ms I29 : 40ms I30 : 20ms I31 : 10ms	Section 10.11
Interrupt program/fixed scan program setting*1		1008H	Set whether an interrupt program is executed at high speed or not.	Execute at high speed/Do not execute at high speed	Do not execute at high speed	Section 3.1.3
Module synchronization		100CH	Set whether a CPU module startup is synchronized with an intelligent function module startup.	Synchronize/Do not synchronize with intelligent function module startup	Synchronize with intelligent function module startup	----

* 1: The Basic model QCPU does not support fixed scan execution type programs.

(3) PLC file

St the file used by the CPU module.



Q parameter setting

File register
○ Not used
● Use

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

Initial Device value
● Not used
○ Use

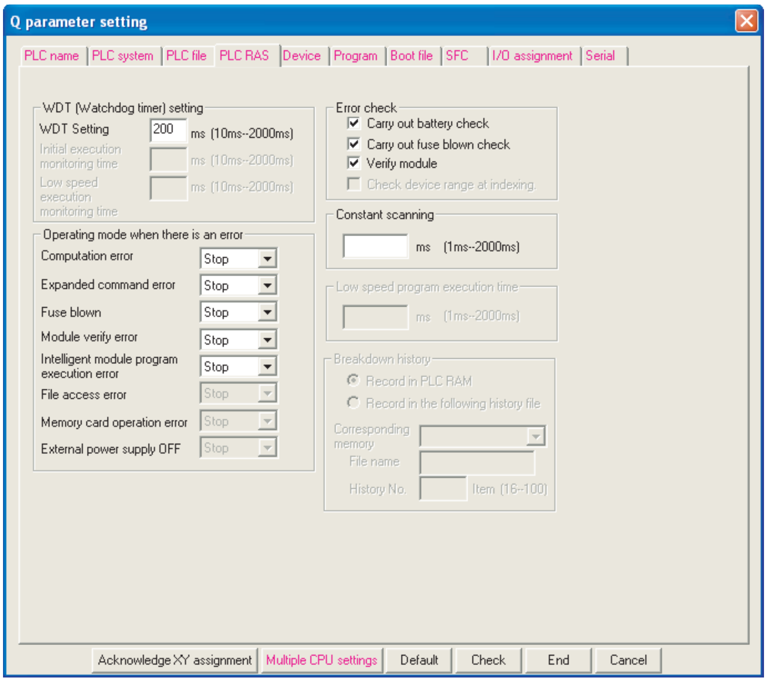
Figure 9.3 PLC file

Table9.3 PLC file list

Item	Parameter No.	Description	Setting range	Default value	Reference
Initial device value	1102H	Set the device initial value file used by the CPU module.	• Not used • Use	Not used	Section 6.26

(4) PLC RAS

Make the various settings for the RAS function.



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.

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 9.4 PLC RAS

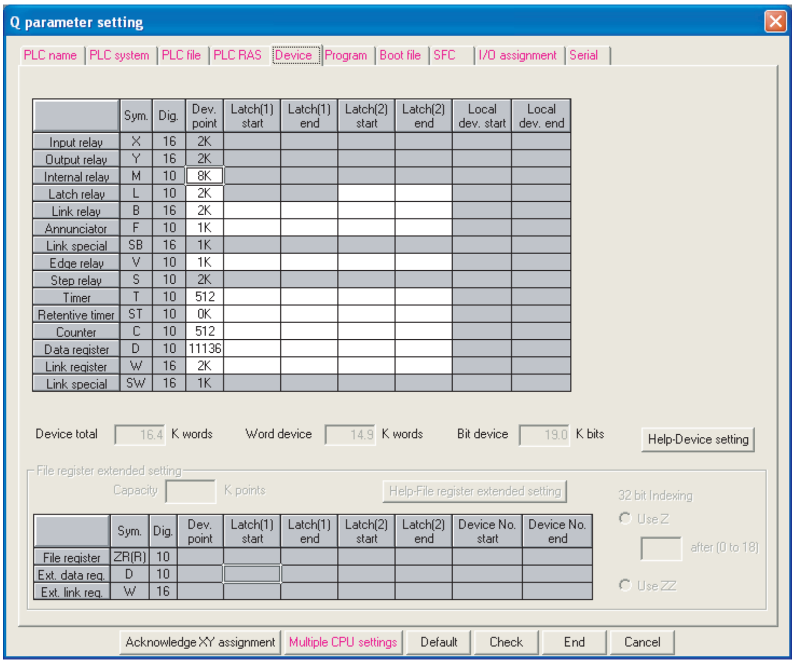
Table9.4 PLC RAS list

Item		Parameter No.	Description	Setting range	Default value	Reference
WDT (watchdog timer) setting	WDT setting	3000H	Set the watchdog timer value of the CPU module.	10ms to 2000ms (10ms unit)	200ms	Section 3.2
Operating mode when there is an error	Computation error	3002H	Set the operation mode of the CPU module at error detection.*2	Stop/Continue	Stop	Section 6.17
	Extended instruction error*1					
	Fuse blown					
	I/O module comparison error					
	Intelligent module program execution error					
Error check	Carry out battery check	3001H	Set whether the specified error will be checked or not.	Check/Do not check	Check	Section 6.17
	Carry out fuse blown check					
	Carry out I/O module comparison					
Constant scanning		3003H	Set the constant scan time.	1ms to 2000ms (1ms unit)	No setting	Section 6.2

* 1: The Extended instruction error is setting for future extension.
* 2: For details of errors, refer to Self-diagnostics list in Section 6.17 (7).

(5) Device

Set the number of used points and latch range 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						

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

32 bit Indexing

○ Use Z

after (0 to 18)

○ Use ZZ

	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							

Figure 9.5 Device

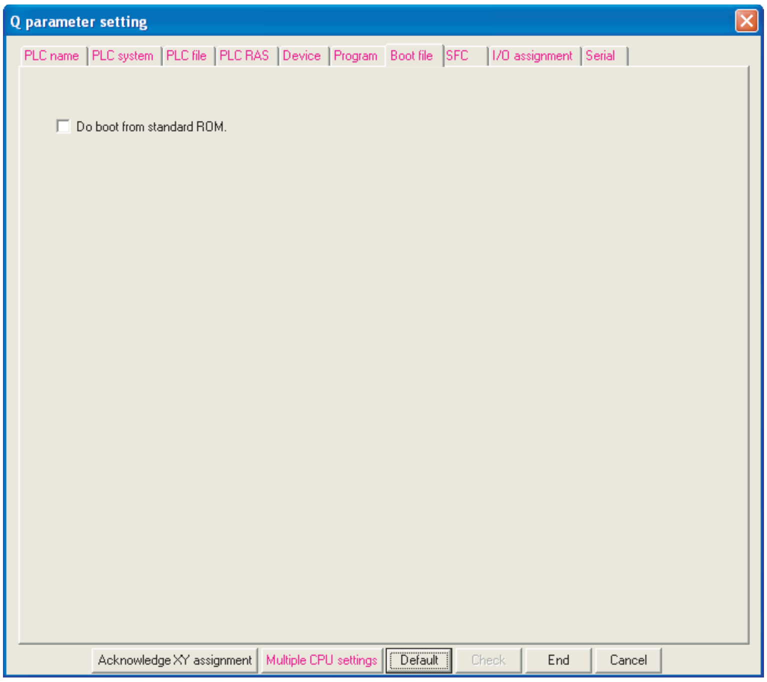
Table9.5 Device list

Item	Parameter No.	Description	Setting range	Default value	Reference
Device points[*1]	2000H	Set the number of used device points according to the system.	X (2k points), Y (2k points), S (2k points), SB (1k points) and SW (1k points) are fixed. Can be set within the range of total 16.4k words, including the above number of points (1.5k words). • 1 device: Max. 32k points (There is no restriction on the total number of bit device points.)	X : 2k points Y : 2k points M : 8k points L : 2k points B : 2k points F : 1k points SB : 1k points V : 1k points S : 2k points T : 512 points ST : 0k points C : 512 points D : 11136points W : 2k points SW : 1k points	Section 10.1 Section 10.2
Latch (1) start/end (Latch clear valid)	2001H	Set the latch range (first device number/last device number) in which clear is enabled by remote latch clear operation.	Set only one range to each of the B, F, V, T, ST, C, D and W devices.	No setting	Section 3.7 Section 6.3
Latch (2) start/end (Latch clear invalid)	2002H	Set the latch range (first device number/last device number) in which clear is disabled by remote latch clear operation.	Set only one range to each of the L, B, F, V, T, ST, C, D and W devices.	No setting	Section 3.7 Section 6.3

* 1 : When changing the device points, the refresh ranges of the network modules and the auto refresh ranges of the intelligent modules must be set so as not to be out of the corresponding device points.
If the device points are set exceeding the corresponding device range, the data may be written into other devices or an error may occur.

(6) Boot file

Set whether a boot from the standard ROM will be executed or not.



□ Do boot from standard ROM.

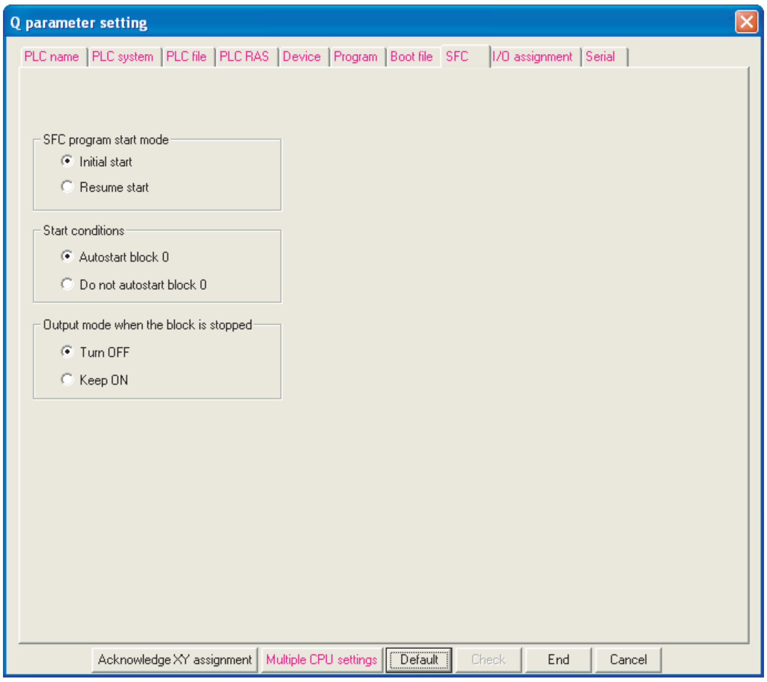
Figure 9.6 Boot file

Table9.6 Boot file list

Item	Parameter No.	Description	Setting range	Default value	Reference
Boot file	----	Set whether a boot from the standard ROM will be executed or not.	Do not execute boot/Execute boot	Do not execute boot	Section 5.1.5

(7) SFC

Set the SFC program start mode, start condition and block stop-time output mode when an SFC program is used.


Q parameter setting
PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment | Serial
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

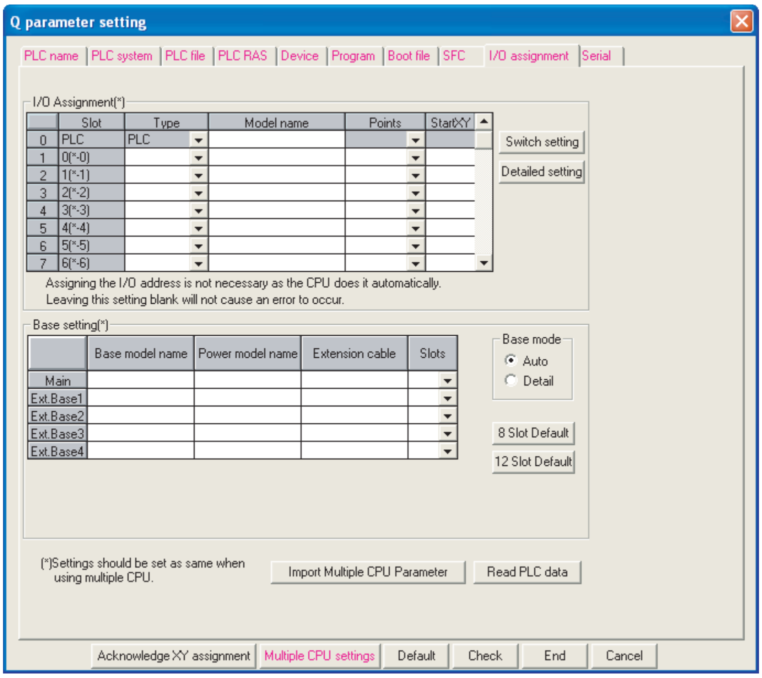
Figure 9.7 SFC

Table9.7 SFC list

Item	Parameter No.	Description	Setting range	Default value	Reference
SFC program start mode	8002H	Set the SFC program start mode, start condition and block stop-time output mode when an SFC program is used.	QCPU (Q Mode)/QnACPU Programming Manual (SFC)	Initial start	----
Start condition	8003H			Autostart Block 0	
Output mode when the block is stopped	8006H			Turn OFF	

(8) I/O assignment

Set the mounting status of each module in the system.



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 9.8 I/O assignment

Table9.8 I/O assignment list

Item		Parameter No.	Description	Setting range	Default value	Reference
I/O assignment	Type	0400H	Set the type of the mounted module.	• CPU No. 2 to No. 3: No. n/ empty (Set "CPU (Empty)" on the slot not mounted with the CPU module.) • Empty, input, high-speed input, output, intelli., hybrid I/ O, interrupt	No setting	Section 4.7
	Model name		Set the model name of the mounted module. (User memo. Not used for the CPU module.)	16 characters	No setting	
	points		Set the number of points of each slot.	[Q00JCPU] 0 points, 16 points, 32 points, 48 points, 64 points, 128 points, 256 points [Q00CPU, Q01CPU] 0 points, 16 points, 32 points, 48 points, 64 points, 128 points, 256 points, 512 points, 1024 points	No setting	
	Start XY (Start I/O No.)		Set the start I/O number of each slot.	[Q00JCPU] 0H to F0H [Q00CPU, Q01CPU] 0H to 3F0H	No setting	

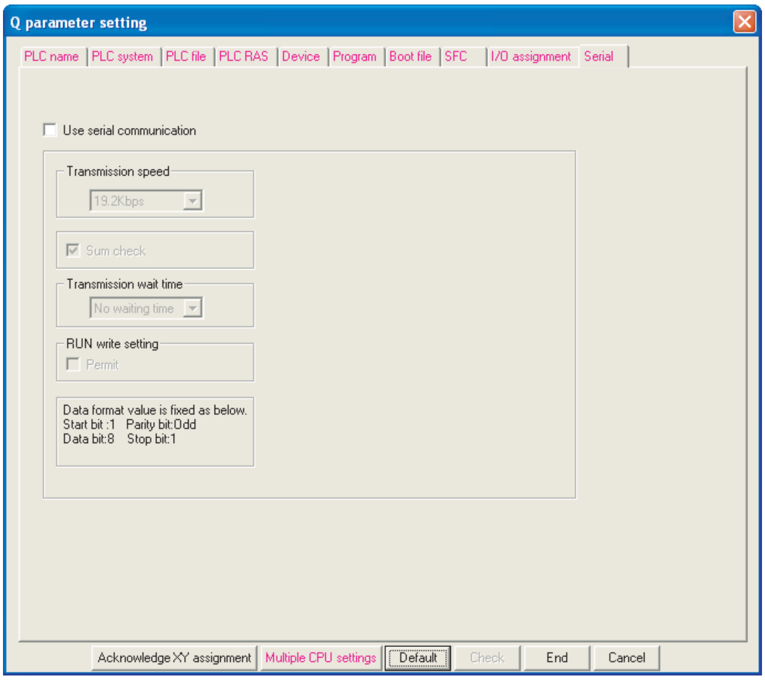
(Continued on next page)

Table9.8 I/O assignment list (Continued)

Item		Parameter No.	Description	Setting range	Default value	Reference
Basic setting	Base model name	0401$_H$	Set the model name of the used main base unit or extension base unit. (User memo. Not used for the CPU module.)	16 characters	No setting	Section 4.4
	Power model name		Set the model name of the power supply module mounted on the main base unit or extension base unit. (User memo. Not used for the CPU module.)	16 characters	No setting	
	Extension cable		Set the extension cable model name. (User memo. Not used for the CPU module.)	16 characters	No setting	
	Slots		Set the number of slots of the main base unit or extension base unit. Set the number of slots for all base units.	2,3,5,8,10,12	No setting	
Switch setting		0407$_H$	Set various switches of the intelligent function module.	Refer to the manual of the used intelligent function module.	No setting	Section 6.10
Detail setting	Error time output mode	0403$_H$	Set whether the output will be cleared or held when the control CPU results in a stop error.	Clear/Hold	Clear	Section 6.8
	Hardware error time PLC operation mode	4004$_H$	Set whether the control CPU will stop or continue operation when the hardware fault of the intelligent function module occurs.	Stop/Continue	Stop	Section 6.9
	I/O response time	0405$_H$	Set the response time of the input module, high-speed input module, hybrid I/O module or interrupt module.	• Input, hybrid I/O : 1ms, 5ms, 10ms, 20ms, 70ms • High-speed input, interrupt : 0.1ms, 0.2ms, 0.4ms, 0.6ms, 1ms	Input, hybrid I/O : 10ms High-speed input, interrupt : 0.2ms	Section 6.7
	Control PLC	0406$_H$	Set the control CPU of the I/O module and intelligent function module.	• No. 1, No. 2, No. 3	No. 1	QCPU User's Manual (Multiple CPU System)

(9) Serial communication

Set the transmission speed, sum check, message waiting time and online change enabled/disabled when the serial communication function of the Q00CPU or Q01CPU is used.



□ 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

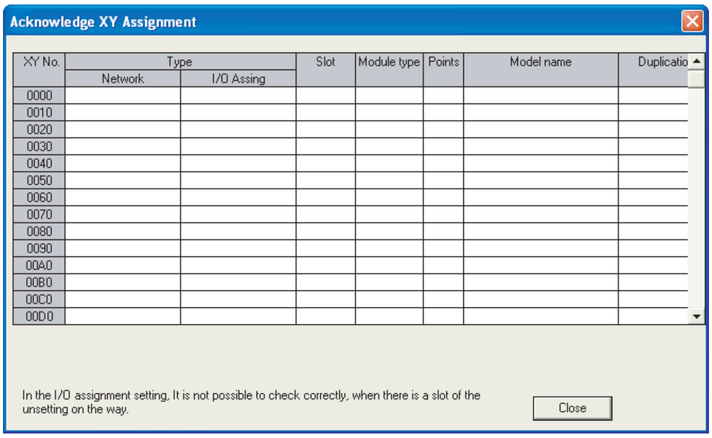
Figure 9.9 Serial communication

Table9.9 Serial communication list

Item	Parameter No.	Description	Setting range	Default value	Reference
Use serial communication	100EH	Check the check box when using the serial communication function.	Do not use/Use serial communication function	Do not use serial communication function	Section 6.24
Transmission speed		Set the 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 or not a sum check code is added to the transmission message and receive message according to the external device specifications when data communication is made by the serial communication function.	No/Yes	Yes	
Transmission wait time		Set the Basic model QCPU side transmission waiting time when the external device cannot receive data immediately after receipt of data.	No waiting/10ms to 150ms (10ms unit)	No waiting time	
RUN write setting		Set whether the external device will write data to the CPU module if the CPU module is during RUN.	Disable/Enable	Disable	

(10) X/Y assignment

Check the data set in the I/O assignment, Ethernet/CC IE/MELSECNET setting and CC-Link setting.



Acknowledge XY Assignment

XY No.	Type		Slot	Module type	Points	Model name	Duplicatio
	Network	I/O Assing					
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.

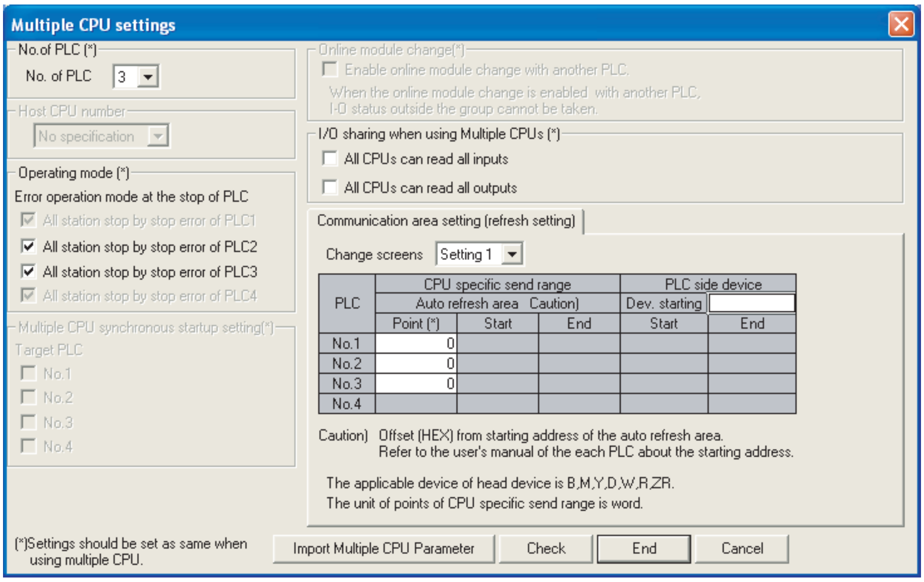
Figure 9.10 X/Y assignment

Table9.10 X/Y assignment list

Item	Parameter No.	Description	Setting range	Default value	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 setting

Make the settings to configure a multiple CPU system.



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 set as same when using multiple CPU.

Figure 9.11 Multiple CPU setting

Table9.11 Multiple CPU setting list

Item		Parameter No.	Description	Setting range	Default value	Reference
Number of PLC		$0E00_H$	Set the number of CPU modules used by the multiple CPU system.	1 to 3 modules	1 module	Refer to QCPU User's Manual (Multiple CPU System).
Operation mode		$0E01_H$	Set the multiple CPU system operation to be performed when the CPU module No. 2 or No. 3 results in a stop error. When the CPU No. 1 results in a stop error, the multiple CPU system stops. (Fixed)	Stop/Do not stop all CPUs at error of CPU No. n	Stop all CPUs at error of CPU No. n	
I/O sharing when using Multiple CPUs	All CPUs can read all inputs	$0E04_H$	Set whether the input status of the input module or intelligent function module controlled by the other CPU will be imported or not.	Do not import non-group input status/Import non-group input status	Do not import non-group input status	
	All CPUs can read all outputs		Set whether the output status of the output module controlled by the other CPU will be imported or not.	Do not import non-group output status/Import non-group output status	Do not import non-group output status	
Refresh setting		$E002_H$ $E003_H$	Set the device and its number of points to/from which data will be written/read by auto refresh between the CPU modules of the multiple CPU system.	[CPU specific send range] No.1: 0 to 320 (2 point units) No.2, 3: 0 to 2048 (2 point units) per module Max. 4416/system [PLC side device] B,M,Y,D,R,ZR The device points set in the transmission range are occupied, starting from the specified device number. • 16 points of B, M or Y are occupied in the transmission range of one point. • 1 point of D, W, R or ZR is occupied in the transmission range of one point.	No setting	

9.1.2 High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

(1) PLC name

Set the label and comment of the used CPU module.

Setting the label and comment in the PLC name does not influence the actual operation.

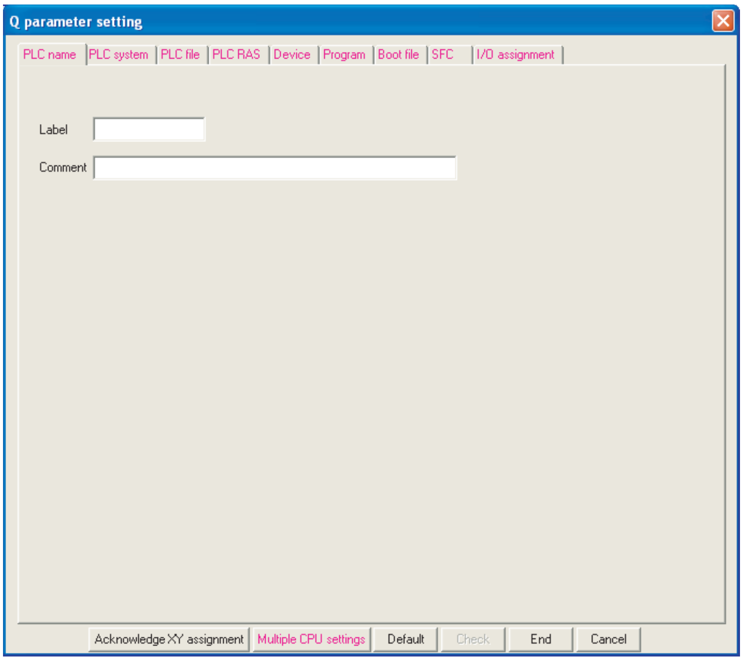


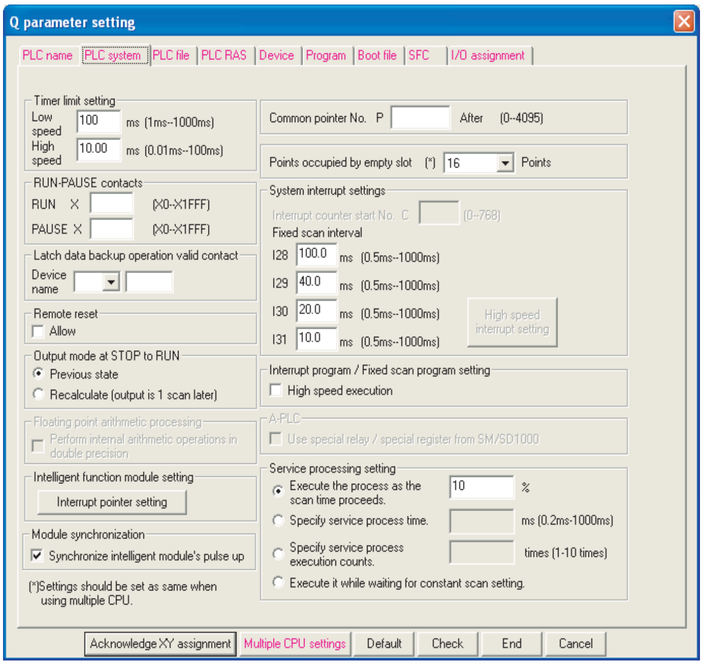
Figure 9.12 PLC name

Table9.12 PLC name list

Item	Parameter No.	Description	Setting range	Default value	Reference
Label	0000H	Set the label (name, application) of the CPU module.	Max. 10 characters	No setting	----
Comment	0001H	Set the comment of the CPU module label.	Max. 64 characters	No setting	----

(2) PLC system

Make setting necessary to use the CPU module.
The parameters may be the default values to perform control.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.00 ms (0.01ms--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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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. 10 %
○ 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.

Figure 9.13 PLC system

Table9.13 PLC system list

Item		Parameter No.	Description	Setting range	Default value	Reference
Timer limit setting	Low speed	1000$_H$	Set the time limit of the low speed timer/high speed timer.	1ms to 1000ms (1ms unit)	100ms	Section 10.2.10
	High speed[2]			0.1ms to 100.0ms (0.1ms unit)	10.0ms	Section 10.2.10
RUN-PAUSE contact	RUN X	1001$_H$	Set the contact that controls RUN/PAUSE of the CPU module. Setting of only the PAUSE contact is not allowed. (The RUN contact or RUN contact and PAUSE contact can be set.)	X0 to 1FFF	No setting	Section 6.6.1
	PAUSE X					Section 6.6.2
Remote reset		1002$_H$	Set enable/disable of remote reset operation from GX Developer.	Enable/disable	Disable	Section 6.6.3
Output mode at STOP to RUN		1003$_H$	Set the output (Y) status when the STOP status is switched to the RUN status.	Provide output (Y) status before STOP/Clear output (Y) (output one scan later)	Provide output (Y) status before STOP	Section 6.4
Floating point arithmetic processing[1]		1004$_H$	Set whether the floating-point operation processing is performed with double precision or not.	Perform/Do not perform floating-point operation processing with double precision	Perform floating-point operation processing with double precision	Section 3.9.4
Intelligent function module setting (interrupt pointer setting)		100A$_H$	Set the assignment of the interrupt pointers (I50 to 255) and the start I/O No. and start SI No. of the intelligent function module.	Start I/O No. Start SI No. I50 to 255	No setting	Section 10.11
Common pointer No.		1005$_H$	Set the start No. of the pointers used as common pointers.	P0 to 4095	No setting	Section 10.10.2

* 1 : Cannot be set when the Process CPU, Redundant CPU or Universal model QCPU is used.
* 2 : The setting range for the Universal model QCPU is 0.01 to 100.0ms (0.01ms units).

(Continued on next page)

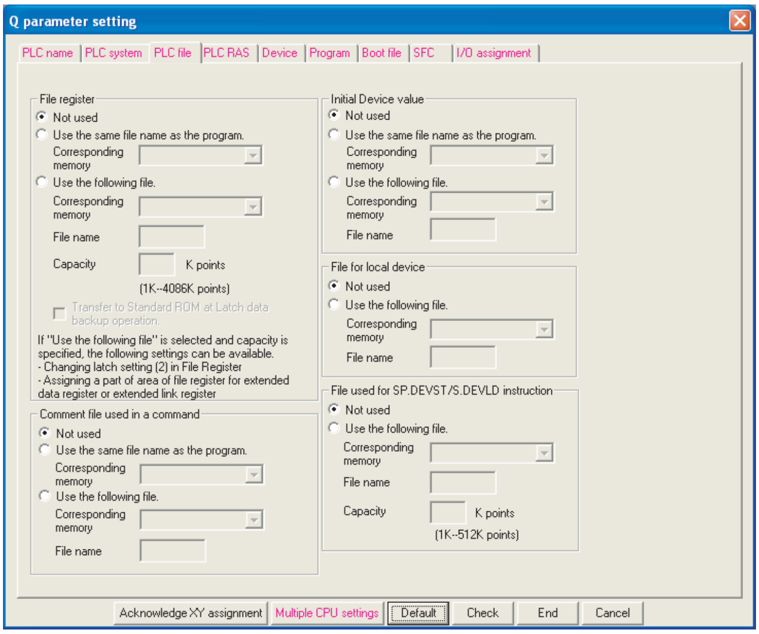
Table9.13 PLC system list (Continued)

Item			Parameter No.	Description	Setting range	Default value	Reference
Points occupied by empty slot *4			1007$_H$	Set the number of empty slots on the main base unit/extension base unit.	0 points/16 points/32 points/64 points/128 points/256 points/512 points/1024 points	16 points	Section 4.6.1(5)
System interrupt setting	Interrupt counter start No.		1008$_H$	Set the start No. of the interrupt counters.*4	(Can be set up to the number of counter setting points - 256)	No setting	Section 10.2.11(4)
	Fixed scan interval (n : 28 to 31)			Set the execution intervals of the interrupt pointers (I28 to 31).	0.5ms to 1000ms (0.5ms unit)	I28:100.0ms I29:40.0ms I30:20.0ms I31:10.0ms	Section 10.11
	High-speed interrupt setting *2	X Input	100F$_H$	St the fixed scan interval, high-speed I/O refresh setting and high-speed buffer transfer setting of the high-speed interrupt pointer I49.	I49 : 0.2ms to 1.0ms (0.1ms unit)	No setting	Section 3.1.3 Section 6.22
		Y Output	1010$_H$				
		Buffer read	1011$_H$				
		Buffer write	1012$_H$				
Service processing setting*3			1013$_H$	Select the service processing execution method from the followings. • Execute the process as the scan time proceeds • Specify service process time • Specify service process execution counts • Execute it while waiting for constant scan setting	• 1 to 99 % (1 percent units) • 1 to 10 times (1 time units) • 0.2ms to 1000ms (0.1ms units) • With/without setting	Execute the process as the scan time proceeds: 10%	Section 6.25.2
Latch data backup operation enabled contact*3			1014$_H$	Sets the enabled contact device No. at the time of backup of latch data to the standard ROM.	----	Not used	Section 6.29
Interrupt program/fixed scan program setting			1008$_H$	Set whether an interrupt program is executed at high speed or not.	Execute at high speed/Do not execute at high speed	Do not execute at high speed	Section 3.1.3
Module synchronization			100C$_H$	Set whether a CPU module startup is synchronized with an intelligent function module startup.	Synchronize/Do not synchronize with intelligent function module startup	Synchronize with intelligent function module startup	----
A-PLC*1			100D$_H$	Set whether the special relays/ special registers for MELSEC-A series (SM1000/SD1000 to SM1299/SD1299) will be used or not.	Use/Do not use the special relays/special registers SM1000/ SD1000 and later	Use the special relays/special registers SM1000/ SD1000 and later	Section 10.3.2 Section 10.3.3

* 1 : Cannot be set when the Redundant CPU and Universal model QCPU is used.
* 2 : High-speed interrupt setting can be made only when the High Performance model QCPU is used.
* 3 : Can be set only when the Universal model QCPU is used.
* 4 : The Universal model QCPU cannot set the head number of the interrupt counter.

(3) PLC file

Set various files used by the CPU module.



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--4086K 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 9.14 PLC file

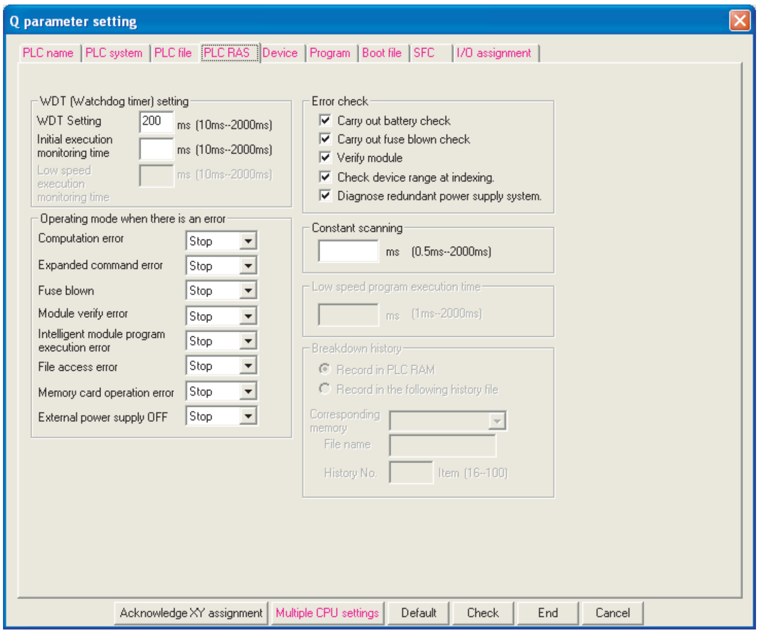
Table9.14 PLC file list

Item	Parameter No.	Description	Setting range	Default value	Reference
File register	1100H	Set the file register file used by the program.	• Not used • Use the same file name as the program • Use the following file	Not used	Section 10.7
Comment file used in a command	1101H	Set the device comment file used by the program.	• Not used • Use the same file name as the program • Use the following file	Not used	----
Initial Device value	1102H	Set the device initial value file used by 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	1103H	Set the local device file used by the program.	• Not used • Use the following file	Not used	Section 10.14.1
Transfer data to the standard ROM at the time of latch data backup operation*1	1104H	Set whether to file register batch transfer or not at the time of latch data backup to the standard ROM.	Whether to perform batch transfer of the file register or not	Not used	Section 6.29
File used for the SP.DEVST/S.DEVLD instruction*1	1105H	Set the file for storing device data used for writing/reading the device data to/from the standard ROM.	• Not used • Use the following file.	Not used	Section 6.30

* 1 : Can be set only when the Universal model QCPU is used.

(4) PLC RAS (PLC RAS (1)[*1])

Set various items for the RAS function.



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)

Breakdown history

● Record in PLC RAM

○ Record in the following history file

Corresponding memory

File name

History No. Item (16--100)

Figure 9.15 PLC RAS

Table9.15 PLC RAS list

Item		Parameter No.	Description	Setting range	Default value	Reference
WDT (watchdog timer) setting	WDT setting	3000H	Set the watchdog timer value of the CPU module.	10ms to 2000ms (10ms unit)	200ms	Section 3.2
	Initial execution monitoring time		Set the watchdog timer value when an initial execution type program is used.	10ms to 2000ms (10ms unit)	No setting	Section 3.3.1
	Low speed execution monitoring time[*2]		Set the watchdog timer value when a low speed execution type program is used.	10ms to 2000ms (10ms unit)	No setting	Section 3.3.3

[*1] : When the PLC parameters of the Process CPU or Redundant CPU are set, "PLC RAS (1)" is displayed on the PLC parameters screen of GX Developer.

[*2] : Cannot be set when the Redundant CPU and Universal model QCPU is used.

(Continued on next page)

Table9.15 PLC RAS list (Continued)

Item		Parameter No.	Description	Setting range	Default value	Reference
Operating mode when there is an error	Computation error	3002_H	Set the operation mode of the CPU module at error detection.*4	Stop/Continue	Stop	Section 6.17
	Extended instruction error*3					
	Fuse blown					
	Module comparison error					
	Intelligent module program execution error					
	File access error					
	Memory card operation error					
	External power supply OFF*3					
Error check	Carry out battery check	3001_H	Set whether the specified error will be checked or not.	Check/Do not check	Check	Section 6.17
	Carry out fuse blown check					
	Carry out module comparison					
	Check device range at indexing *6					
	Diagnose redundant power supply system*7					
Constant scanning		3003_H	Set the constant scan time.	0.5ms to 2000ms (0.5ms unit)	No setting	Section 6.2
Low speed program execution time*2		3006_H	Set the execution time of a low speed program every scan.	1ms to 2000ms	No setting	Section 3.3.3
Breakdown history		3005_H	Set the storage destination of the CPU module failure history.*5	Store into program memory/ Store into the following history file	Store into program memory	Section 6.18

* 2: Cannot be set when the Redundant CPU and Universal model QCPU is used.

* 3: TheExtended instruction error and External power supply OFF is setting for future extension.

* 4: For details of errors, refer to Self-diagnostics list in Section 6.17 (7).

* 5: Cannot be set when the Universal model QCPU is used.

* 6: Can be set only when the Universal model QCPU is used.

* 7: This item cannot be set when the High Performance model QCPU, Process CPU or Redundant CPU is used. When setting this item in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

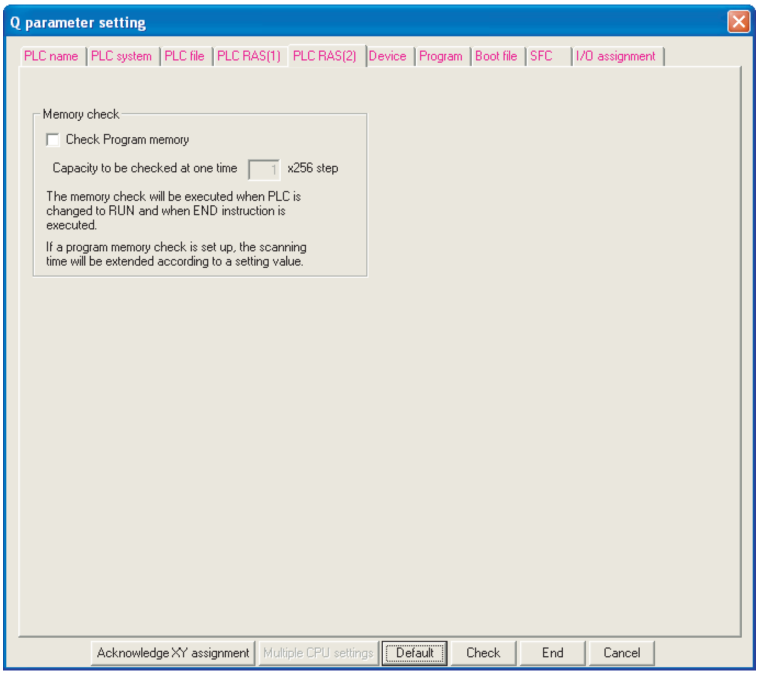
Process

Note9.1

(5) PLC RAS (2)Note9.1

Make the various settings for the RAS function in the Process CPU or Redundant CPU.

his setting can be made only when the Redundant CPU or Process CPU is used.



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 9.16 PLC RAS (2)

Table9.16 PLC RAS (2) list

Item		Parameter No.	Description	Setting range	Default value	Reference
Memory check	Check Program memory	3008H	Set whether the user area of the program memory (except the system area) will be checked for data corruption or not.	Do not make/Make program memory check	Do not make program memory check	Section 6.28
			When "Make program memory check" is selected, set the number of steps to be checked.	Q12PHCPU, Q12PRHCPU: 1 to 496 points (256 to 124k steps) Q25PHCPU, Q25PRHCPU: 1 to 1008 points (256 to 252k steps)	1 × 256 steps	

☒POINT

When checking all areas of the program memory, follow the procedure below to calculate the number of steps for the check target.

- Check free space in the program memory using the PC call-up screen in GX Developer. To do that, first format the program memory so that the memory has no file written in it. (☞ Section 5.1.2(3)(c))
- Use the following expression to obtain the number of steps for the check target.

$$\text{Number of steps for check targets} = \frac{\text{All free space}}{4}$$

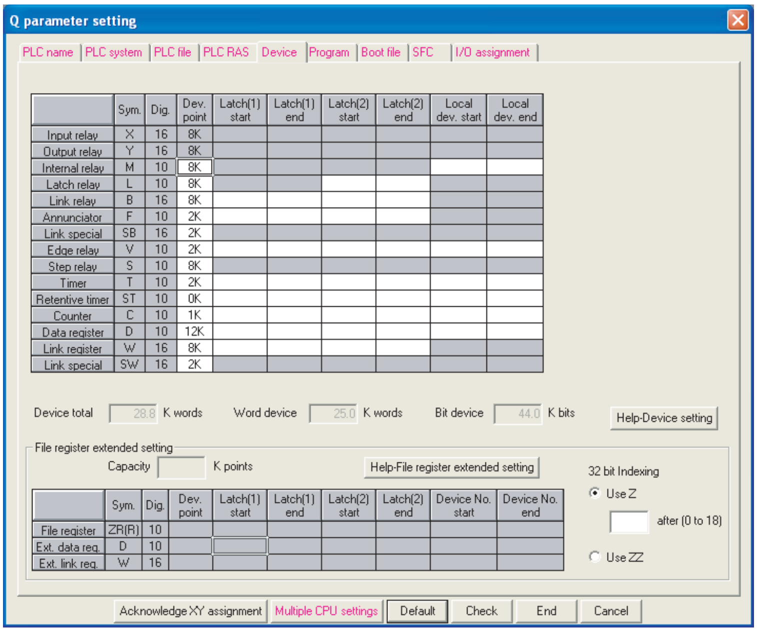
Process

Note9.1

Check the version of the CPU module and GX Developer before making the settings for memory-checking using the Process CPU.(☞ Appendix 4.3)

(6) Device

Set the number of used points, latch range and local device range 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						

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 9.17 Device

Table9.17 Device list

Item	Parameter No.	Description	Setting range	Default value	Reference
Device points*1	2000_H	Set the number of used device points according to the system.	X (8k points), Y (8k points), S (8k points)*5, SB (2k points)*3 and SW (2k points)*3 are fixed. Can be set within the range of total 29k words, including the above number of points (3.7k words). 1 device: Max. 32k points Total of bit devices: 64k points	X:8k points,Y:8k points M:8k points, L:8k points*4 B:8k points, F:2k points SB:2k points, V:2k points S:8k points*5, T:2k points ST:0k points, C:1k points D:12k points, W:8k points SW:2k points	Section 10.1 Section 10.2
Latch (1) start/end*4	2001_H	Set the latch range (first device number/last device number) in which clear is enabled by the RESET/L.CLR switch*2 or remote latch clear operation.	Set only one range to each of the B, F, V, T, ST, C, D and W devices.	No setting	Section 3.7 Section 6.3
Latch (2) start/end*4	2002_H	Set the latch range (first device number/last device number) in which clear is disabled by the RESET/L.CLR switch*2 or remote latch clear operation.	Set only one range to each of the L, B, F, V, T, ST, C, D and W devices.	No setting	Section 3.7 Section 6.3
Local device start/end	2003_H	Set the device range (first device number/last device number) used as a local device.	Set only one range to each of the M, V, T, ST, C and D devices.	No setting	Section 10.14.1

Table9.18 Device list (Continued)

Item		Parameter No.	Description	Setting range	Default value	Reference
File register extended setting[*6]	Dev. point	2000_H	Set the number of file register (ZR), extended data register (D), and extended link register (W) points.	• Number of points set for the file register (ZR) • Assign a part of the points set for the file register to the extended data register (D) or the extended link register.	No setting	Section 10.2 Section 10.8
	Latch (1) start/end (Latch clear enabled)	2004_H	Set the latch range (start device number/end device number) that can be cleared with the remote latch clear operation.	Each latch range of the file register (ZR), extended data register (D), and extended link register (W)	No setting	Section 3.7 Section 6.3
	Latch (2) start/end (Latch clear disabled)	2005_H	Set the latch range (start device number/end device number) that cannot be cleared with the remote latch clear operation.	Each latch range of the file register (ZR), extended data register (D), and extended link register (W)	No setting	Section 3.7 Section 6.3
Indexing setting of ZR device[*7]		2000_H	Set whether to use the Z device or the ZZ device with the 32-bit index modification.	• Z0 to 18 (When using the Z device)	Use Z	QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

* 1: When changing the device points, the refresh ranges of the network modules and the auto refresh ranges of the intelligent function modules must be set so as not to be out of the out of the corresponding device points.
If the device points are set exceeding the corresponding device range, the data may be written into other devices or an error may occur.

* 2: For the Universal model QCPU, the latch clear cannot be made by the switch operation.

* 3: Setting in the range is allowed for the Universal model QCPU.

* 4: For the Universal model QCPU, the scan time is prolonged when the device is latched. When latching a device, consider the prolonging scan time. (☞ Section 10.1.2 (11))

* 5: In the Universal model QCPU whose serial number (first five digits) is "10042" or later, 0 point can be set for step relays (S).

* 6: Setting is allowed only when the Universal model QCPU is used.

* 7: Setting is allowed only when the Universal model QCPU whose serial number (first five digits) is "10042" or later.

(7) Program

When writing multiple programs to the CPU module, set the file names and execution types (execution conditions) of the programs.

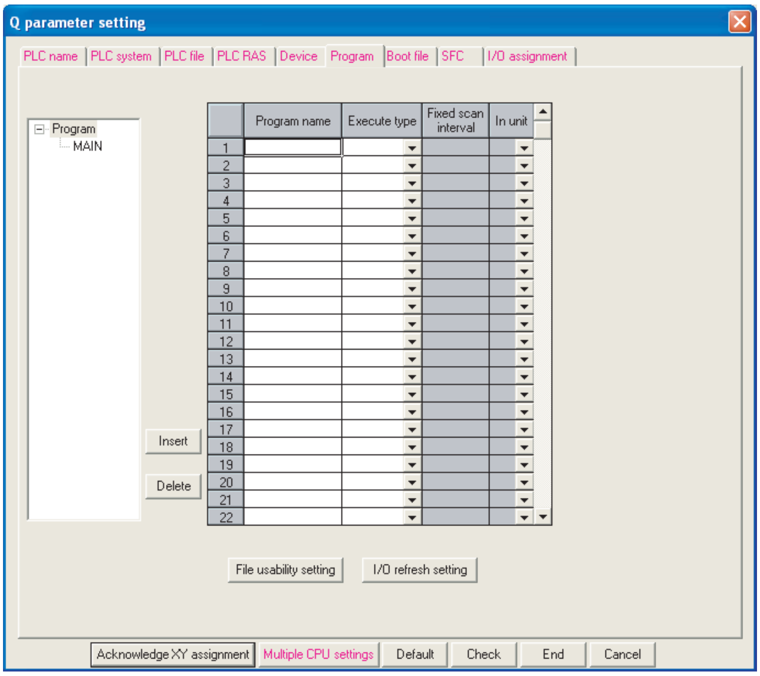


Figure 9.18 Program

Table9.19 Program list

Item	Parameter No.	Description	Setting range	Default value	Reference
Program	7000$_H$	When writing multiple programs to the CPU module, set the file names and execution types (execution conditions) of the programs. Also set the fixed scan interval (execution interval of fixed scan execution type program).	• Program name • Execution type (fixed scan interval when fixed scan execution is selected) • File using method setting *1 • I/O refresh setting *2	No setting	Section 3.3.6

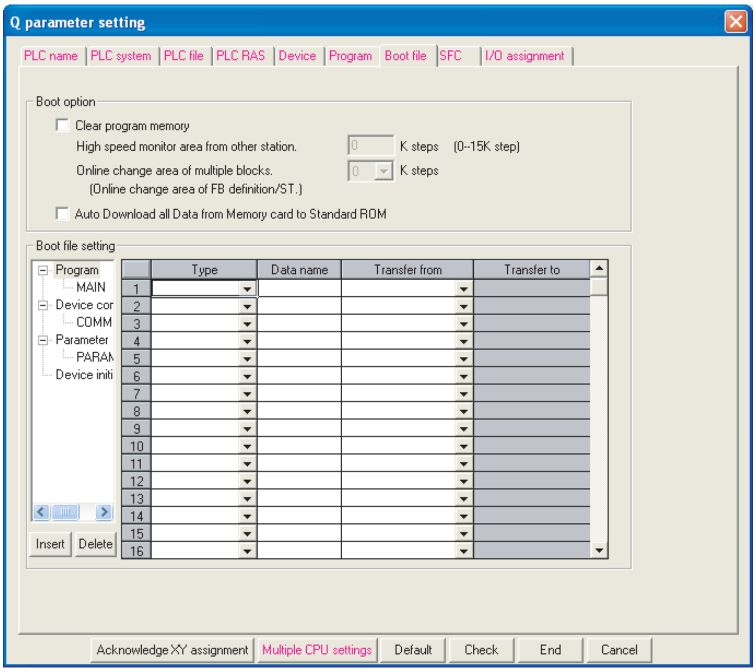
* 1: When using a Universal model QCPU, only local device can be set.
When using the file usability setting with a Universal model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.5)

* 2: Not settable when the Universal model QCPU is used.

(8) Boot file

Universal UD Note9.2

Make the settings to execute boot run and automatic write to standard ROM[Note9.2].



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

Figure 9.19 Boot file

Table9.20 Boot file setting list

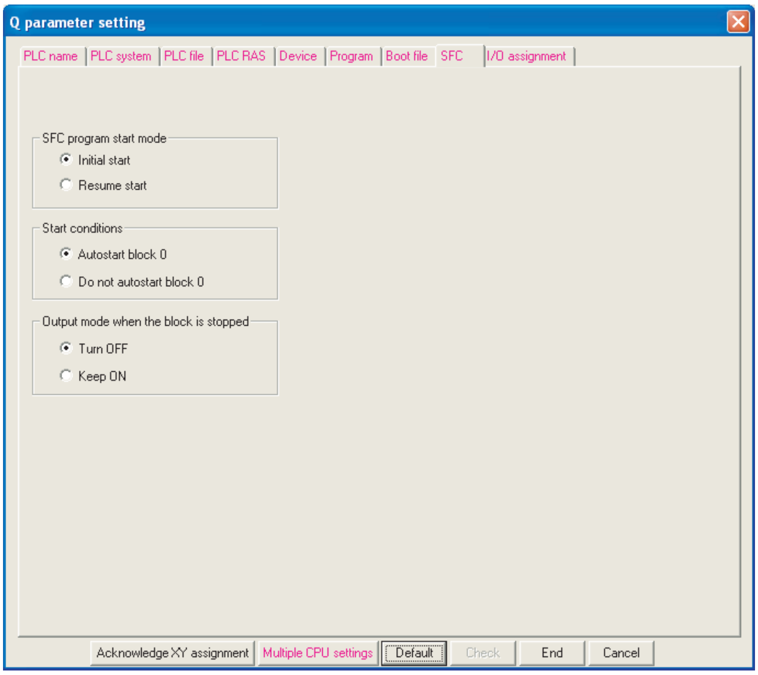
Item		Parameter No.	Description	Setting range	Default value	Reference
Boot option	Clear program memory	7000$_H$	Set whether the program memory will be cleared or not at boot.	Do not clear/Clear program memory at boot	Do not clear program memory at boot	Section 5.2.8 Section 5.2.9
	Auto Download all Data from Memory card to Standard ROM		Set whether the memory card data will be automatically written to the standard ROM or not at a boot.	Do not perform/Perform automatic all data write from memory card to standard ROM	Do not perform automatic all data write from memory card to standard ROM	
Boot file setting			Set the type, data name and transfer source drive of the boot-run file.	Type, data name, transfer source drive (transfer source drive is automatically set by the program memory)	No setting	

Universal UD Note9.2

Automatic write to standard ROM is not available for the Universal model QCPU.

(9) SFC

Set the SFC program start mode, start condition and block stop-time output mode when an SFC program is used.



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

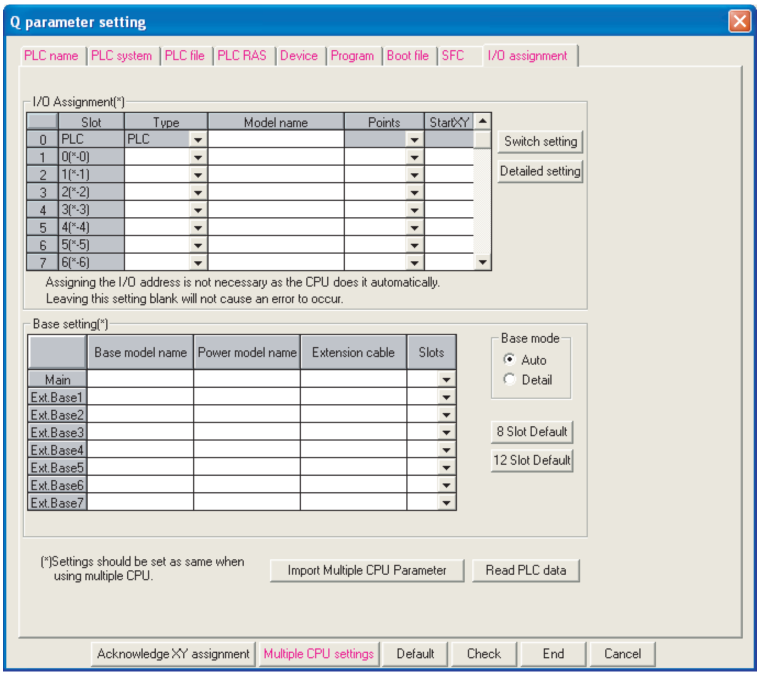
Figure 9.20 SFC

Table9.21 SFC list

Item	Parameter No.	Description	Setting range	Default value	Reference
SFC program start mode	8002H	Set the SFC program start mode, start condition and block stop-time output mode when an SFC program is used.	Refer to the QCPU (Q Mode)/ QnACPU Programming Manual (SFC).	Initial start	----
Start condition	8003H			Automatically start Block 0	
Output mode when the block is stopped	8006H			Turn OFF	

(10)I/O assignment

Set the mounting status of each module in the system.



	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 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 9.21 I/O assignment

Table9.22 I/O assignment list

Item		Parameter No.	Description	Setting range	Default value	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)" on the slot not mounted with the CPU module.) • Empty, input, high-speed input, output, intelli., hybrid I/ O, interrupt	No setting	Section 4.7
	Model name		Set the model name of the mounted module. (User memo. Not used for the CPU module.)	16 characters	No setting	
	points		Set the number of points of each slot.	0 points, 16 points, 32 points, 48 points, 64 points, 128 points, 256 points, 512 points, 1024 points	No setting	
	Start XY (Start I/O No.)		Set the start I/O number of each slot.	0_H to $FF0_H$	No setting	

(Continued on next page)

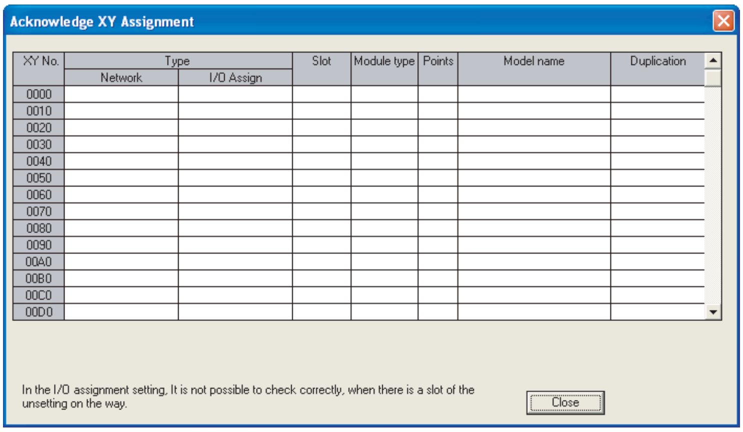
Table9.22 I/O assignment list (Continued)

	Item	Parameter No.	Description	Setting range	Default value	Reference
Basic setting	Base model name	0401H	Set the model name of the used main base unit or extension base unit. (User memo. Not used for the CPU module.)	16 characters	No setting	Section 4.4
	Power model name		Set the model name of the power supply module mounted on the main base unit or extension base unit. (User memo. Not used for the CPU module.)	16 characters	No setting	
	Extension cable		Set the extension cable model name. (User memo. Not used for the CPU module.)	16 characters	No setting	
	Slots		Set the number of slots of the main base unit or extension base unit. Set the number of slots for all base units.	2,3,5,8,10,12	No setting	
Switch setting		0407H	Set various switches of the intelligent function module.	Refer to the manual of the used intelligent function module.	No setting	Section 6.10
Detail setting	Error time output mode	0403H	Set whether the output will be cleared or held when the control CPU results in a stop error.	Clear/Hold	Clear	Section 6.8
	Hardware error time PLC operation mode	4004H	Set whether the control CPU will stop or continue operation when the hardware fault of the intelligent function module occurs.	Stop/Continue	Stop	Section 6.9
	I/O response time	0405H	Set the response time of the input module, high-speed input module, hybrid I/O module or interrupt module.	• Input, hybrid I/O: 5ms, 10ms, 20ms, 70ms • High-speed input, interrupt: 0.1ms, 0.2ms, 0.4s, 0.6ms, 1ms	Input, hybrid I/O: 10ms High-speed input, interrupt: 0.2ms	Section 6.7
	Control PLC*1	0406H	Set the control CPU of the I/O module and intelligent function module.	• No. 1, No. 2, No. 3, No. 4	No. 1	QCPU User's Manual (Multiple CPU System)

* 1 : Cannot be set when the Redundant CPU is used.

(11) X/Y assignment

Check the data set in the I/O assignment, Ethernet/CC IE/MELSECNET setting and CC-Link setting.



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 9.22 X/Y assignment

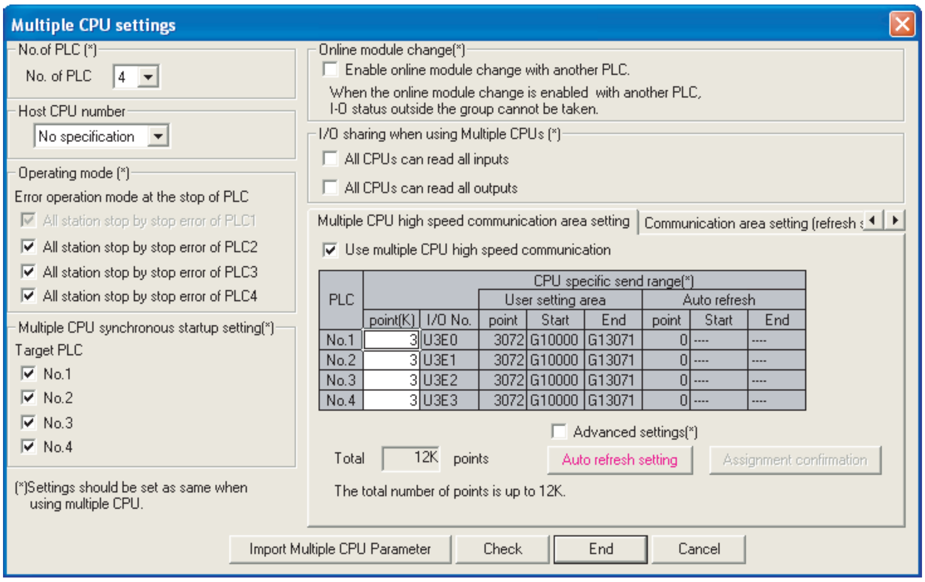
Table9.23 X/Y assignment list

Item	Parameter No.	Description	Setting range	Default value	Reference
X/Y assignment	----	The data set in the I/O assignment, Ethernet/CC IE/ MELSECNET setting and CC-Link setting can be checked.	----	----	----

Redundant
Note9.3

(12) Multiple CPU setting*Note9.3*

Make setting to configure a multiple CPU system.



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

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

Online module change(*)
☐ Enable online module change with another PLC.
When the online module change is enabled with another PLC, I-0 status outside the group cannot be taken.

I/0 sharing when using Multiple CPUs (*)
☐ All CPUs can read all inputs
☐ All CPUs can read all outputs

Multiple CPU high speed communication area setting | Communication area setting (refresh s

☑ Use multiple CPU high speed communication

PLC	CPU specific send range(*)							
			User setting area			Auto refresh		
	point(K)	I/0 No.	point	Start	End	point	Start	End
No.1	3	U3E0	3072	G10000	G13071	0	----	----
No.2	3	U3E1	3072	G10000	G13071	0	----	----
No.3	3	U3E2	3072	G10000	G13071	0	----	----
No.4	3	U3E3	3072	G10000	G13071	0	----	----

☐ Advanced settings(*)

Total 12K points

The total number of points is up to 12K.

Figure 9.23 Multiple CPU setting

Table9.24 Multiple CPU setting list

Item	Parameter No.	Description	Setting range	Default value	Reference
No. of PLC	0E00H	Set the number of CPU modules used by the multiple CPU system.	1 to 4 modules	1 module	QCPU User's Manual (Multiple CPU System)
Host CPU number *1	E00CH	Specify the CPU number for which the multiple CPU setting parameters are set. (Specify the host CPU number.)	PLC No.1 to PLC No.4	No specification	
Operating mode	0E01H	Set the multiple CPU system operation to be performed when any of the CPU modules No. 2 to No. 4 results in a stop error. When the CPU No. 1 results in a stop error, the multiple CPU system stops. (Fixed)	Stop/Do not stop all CPUs at error of CPU No. n	Stop all CPUs at error of CPU No. n	
Multiple CPU synchronous startup setting *1	E00BH	Set whether to synchronize the startup of CPU modules in the multiple CPU system.	PLC No.1 to PLC No.4	All CPU modules are set for synchronous startup.	

* 1: Can be set only when the Universal model QCPU (except the Q02UCPU) is used.

(Continued on next page)

Redundant
Note9.3

The Redundant CPU does not support the multiple CPU setting since a multiple CPU system cannot be configured.

Table9.24 Multiple CPU setting list (Continued)

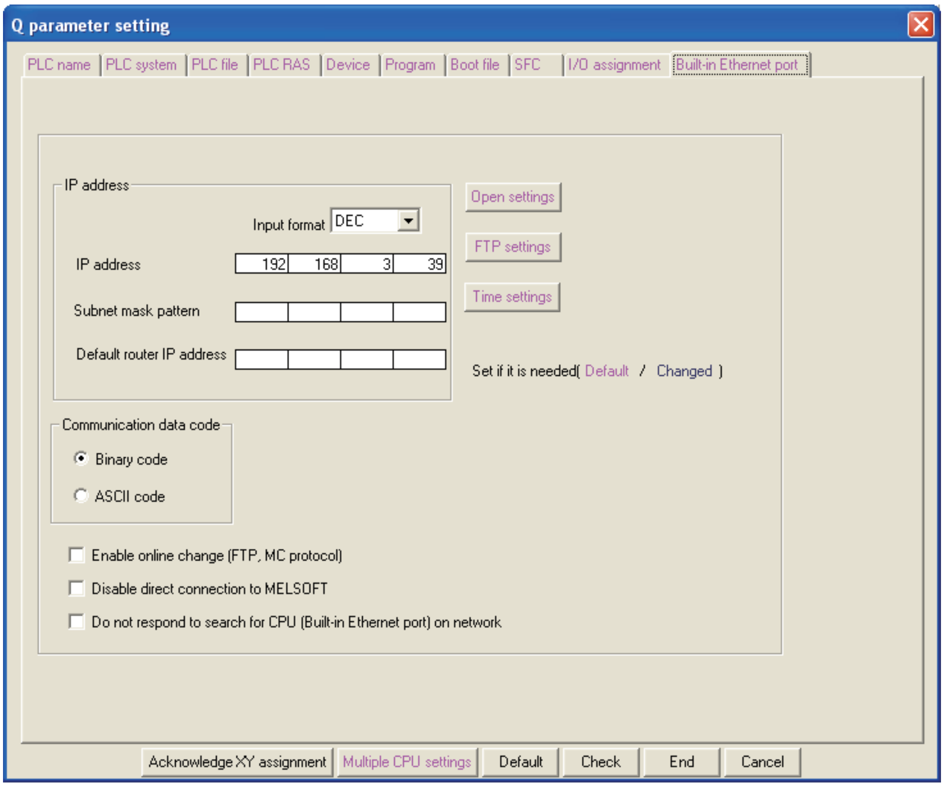
Item			Parameter No.	Description	Setting range	Default value	Reference
Online module change (hot swapping) setting *2			E006H	Set whether online module change (hot swapping) is enabled or disabled in the multiple CPU system. (When enable is selected, the non-group I/O status cannot be imported.	Enable/Disable online module change (hot swapping) of other CPU modules.	• High Performance model QCPU: Disable • Process CPU: Enable • Universal model QCPU: Disable	QCPU User's Manual (Multiple CPU System)
I/O sharing when using Multiple CPUs	All CPUs can read all inputs		0E04H	Set whether the input status of the input module or intelligent function module controlled by the other CPU will be imported or not.	Do not import non-group input status/Import non-group input status	Do not import non-group input status	
	All CPUs can read all outputs			Set whether the output status of the output module controlled by the other CPU will be imported or not.	Do not import non-group output status/Import non-group output status	Do not import non-group output status	
Multiple CPU high speed communication area setting	CPU specific send range *1		E008H	Set the multiple CPU high-speed transmission area size assigned for each CPU module configuring the multiple CPU system.	• For simple setting: 0 to 12k points (1k units) • For high-speed setting: 0 to 16k points (0.5k units)	3k points	
	Auto refresh setting *1	Auto refresh device setting *1	E009H E00AH	Set a range for data communication performed with the auto refresh function in the user area of multiple CPU high-speed transmission area.	Available devices*3: X, Y, M, L, B, D, W, R, ZR, SM, SD, SB, SW	----	
Communication area setting (refresh setting)			E002H E003H	Set the device and its number of points to/from which data will be written/read by auto refresh between the CPU modules of the multiple CPU system.	[Set the head device to each CPU] Do not set the head device to each CPU/set the head device to each CPU	Do not set the head device to each CPU	
					[Transmission range of each CPU module] 0 to 2048 points (2 point units)/module Max. 4k points (4096 points)/ system	No setting	
Refresh parameter detailed device specification *1			E007H	Specify the transfer destination device and the number of transfer points for transferring the data in the multiple CPU shared memory to the device in the CPU module when the start device is set in the CPU module. Specification is available for each CPU module respectively.	Available devices: D, W, R, ZR, B, M, Y	No setting	

* 1 : Settable only when the Universal model QCPU (except Q02UCPU) is used.
* 2 : Not settable in the Q02UCPU.
* 3 : SM, SD, SB and SW are usable only when they are selected as a send device.

Basic Note9.4
High Performance Note9.4
Process Note9.4
Redundant Note9.4
Universal Note9.4

(13) Built-in Ethernet port Note9.4

Make settings required for using built-in Ethernet ports of the Built-in Ethernet port QCPU.



Q parameter setting

PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment | Built-in Ethernet port

IP address

Input format: DEC

IP address: 192 . 168 . 3 . 39

Subnet mask pattern:

Default router IP address:

Communication data code

● Binary code

○ ASCII code

□ Enable online change (FTP, MC protocol)

□ Disable direct connection to MELSOFT

□ Do not respond to search for CPU (Built-in Ethernet port) on network

Set if it is needed(Default / Changed)

Figure 9.24 Built-in Ethernet port

Table9.25 Built-in Ethernet por setting list

Item	Parameter No.	Description	Setting range	Default value	Reference
IP address	1016H	IP address : Specify the QnUDE(H)CPU IP address. Subnet mask pattern: Specify the subnet mask for communication via routers. Default router IP address: Specify the router IP address for communication via routers.	IP address : 0.0.0.1 to 223.255.255.254 (00000001$_H$ to 0DFFFFFFE$_H$) Subnet mask pattern: None/ 192.0.0.0 to 255.255.255.252 (0C0000000$_H$ to 0FFFFFFFC$_H$) Default router IP address: None/ 0.0.0.1 to 223.255.255.254 (00000001$_H$ to 0DFFFFFFE$_H$)	IP address : 192.168.3.39 Subnet mask pattern: None Default router IP address: None	Section 7.2
Communication data code		Select the code, Binary or ASCII, for MC protocol communication.	Binary code/ASCII code	Binary code	Section 7.4

Basic Note9.4 | High Performance Note9.4 | Process Note9.4 | Redundant Note9.4 | Universal Note9.4

Only the Built-in Ethernet port QCPU can use the parameter settings on the Built-in Ethernet port tab.

Table9.25 Built-in Ethernet por setting list (Continued)

Item	Parameter No.	Description	Setting range	Default value	Reference
Open settings	1016H	Set items when the MC protocol communication function is used.	---	No settings	Section 7.2 Section 7.4
FTP settings		Set items when the File transfer function (FTP) is used.	---	No settings	Section 7.6
Time settings		Set items when the Time setting function is used.	---	No settings	Section 7.5
Enable online change (FTP, MC protocol)		Specify whether to enable online change (writing device data or files while the CPU is in the RUN status) in the communication using the MC protocol or the FTP function.	Checked/Unchecked (Enabled/ Disabled)	Unchecked (Disabled)	Section 7.4 Section 7.6
Disable direct connection to MELSOFT		Specify whether to disable MELSOFT direction connection. Disable direct connection to emphasize the security by setting a remote password.	Checked/Unchecked (Disabled/ Enabled)	Unchecked (Enabled)	---
Do not respond to search for CPU (Built-in Ethernet port) on network		Disable CPU modules connected to MELSOFT devices for responding the Find CPU function. Disable the response to emphasize the security.	Checked/Unchecked (Disabled/ Enabled)	Unchecked (Enabled)	---

Basic Note9.5
High Performance Note9.5
Process Note9.5
Universal Note9.5

9.2 Redundant Parameter

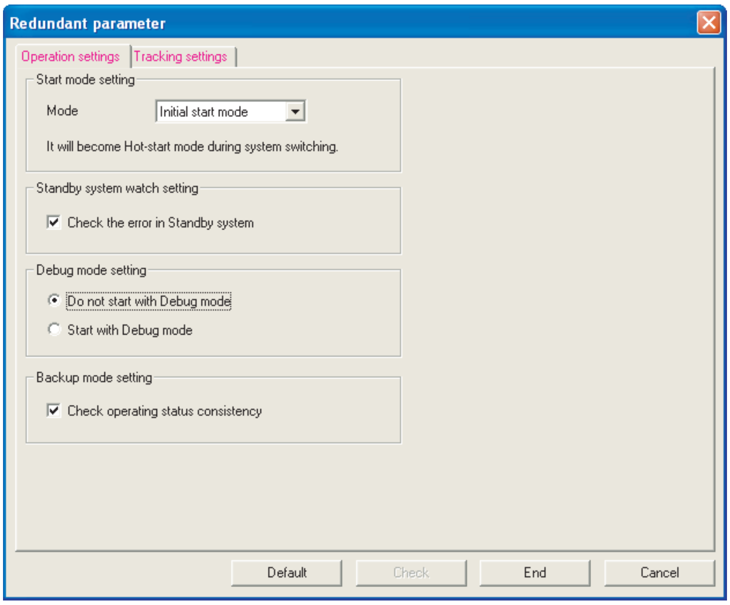
This section explains the redundant parameter list and parameter details.

Table9.26 Redundant parameters

Item	Parameter No.	Description	Setting range	Default value	Reference
Redundant Parameter	0D00H	Set the operation mode and tracking in the Redundant CPU.	----	----	----

(1) Operation mode setting

Set the Redundant CPU operation to be performed when the redundant system is powered ON.



Redundant parameter

Operation settings | Tracking settings

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 9.25 Operation setting

Table9.27 Operation setting list

Item	Parameter No.	Description	Setting range	Default value	Reference
Start mode setting	D001H	Set the start mode of the Redundant CPU when the system 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 a standby system error will be watched or not.	Check for a standby system error/Do not check for a standby system error	Check for a standby system error	
Debug mode setting		Set whether the Redundant CPU will be started in the debug mode or not.	Do not start in debug mode/Start in debug mode	Do not start in debug mode	
Backup mode setting		Set whether an operating status consistency check between system A and B will be performed or not when the Redundant CPU is operating in the backup mode.	Perform the operation status consistency check/Do not perform the operation status consistency check	Perform the operation status consistency check	

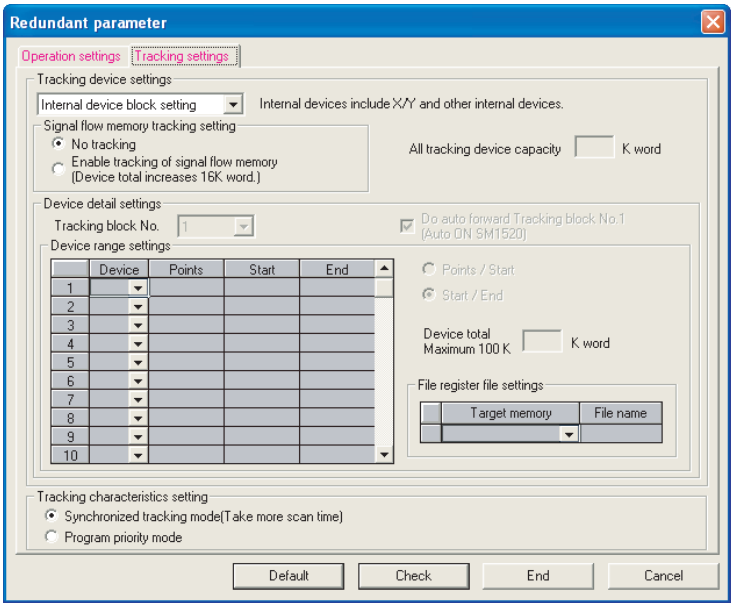
Basic Note9.5
High Performance Note9.5
Process Note9.5
Universal Note9.5

module The Basic model QCPU, High Performance model QCPU, Process CPU and Universal model QCPU cannot use the redundant parameters.

These parameters can be set only when the Redundant CPU is used.

(2) Tracking settings

Make the settings for the tracking function of the Redundant CPU.



Redundant parameter

Operation settings | Tracking settings

Tracking device settings

Internal device block setting — Internal devices include X/Y and other internal devices.

Signal flow memory tracking setting
- ● No tracking
- ○ Enable tracking of signal flow memory (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 9.26 Tracking settings

Table9.28 Tracking settings list

Item		Parameter No.	Description	Setting range	Default value	Reference
Tracking device settings		D003H	Set whether the range of the tracking device data will be set freely or not.	Internal device block setting/ Device detail settings	Internal device block setting	QnPRHCPU User's Manual (Redundant System)
Signal flow memory tracking setting			Set whether signal flow memory tracking will be done or not.	No tracking/Do tracking	No tracking	
Device detail settings			Set the tracking device when the user sets the tracking device freely.	----	----	
	Tracking block No.		Select the block No. of the tracking device. Select the block No. and set the tracking device in the device range settings.	1 to 64	1	
	Do auto forward Tracking block No.1 (Auto ON SM1520)		Set whether the tracking block No. 1 will be automatically tracked or not.	Auto tracking enabled/Auto tracking disabled	Auto tracking enabled	
	Device range settings		Set the tracking device and range.	QnPRHCPU User's Manual (Redundant System)	----	
	File register file settings		When the file register has been set in the device range settings, set the target memory, file name, etc. of the tracking file register.		----	
Tracking characteristics settings		D002H	Set the tracking operation when executing tracking.	Synchronized tracking mode/ Program priority mode	Synchronized tracking mode	

Remark

Refer to the following manual for the precautions for the tracking function and redundant parameter setting.

☞ QnPRHCPU User's Manual (Redundant System)

9.3 Network Parameters

This section explains the network parameter list and parameter details.

□ mn, **, N and M in the Parameter No. field of this section

mn, **, N and M in the Parameter No. field of this section indicate the following.

mn	: Indicates a "start I/O No. ÷ 16" value.
**	: Indicates any value.
N	: Indicates the module number.
M	: Indicates the network type.

Table9.29 Setting for CC-Link IE controller network and MELSECNET/H (☞ (1), (2) in this section)

Basic

Note9.6

M	Network type
1_H	CC IE Control (Control station) Note9.6, MELSECNET/H mode (Control station), MELSECNET/H. Extended mode (Control station), MELSECNET/10 mode (Control station)
2_H	CC IE Control (Normal station), MELSECNET/H mode (Normal station), MELSECNET/H. Extended mode (Normal station), MELSECNET/10 mode (Normal station)
5_H	MELSECNET/H (Remote master)
A_H	MELSECNET/H (Standby station)
B_H	MELSECNET/H mode multiplexed remote I/O network master station
D_H	MELSECNET/H mode multiplexed remote I/O network sub-master station (When there is no parameter setting)
E_H	MELSECNET/H mode multiplexed remote I/O network sub-master station (When there is parameter setting)

Table9.30 Setting for CC-Link (☞ (4) in this section)

M	Network type
0_H	Master station
1_H	Local station
2_H	Standby master station

Basic

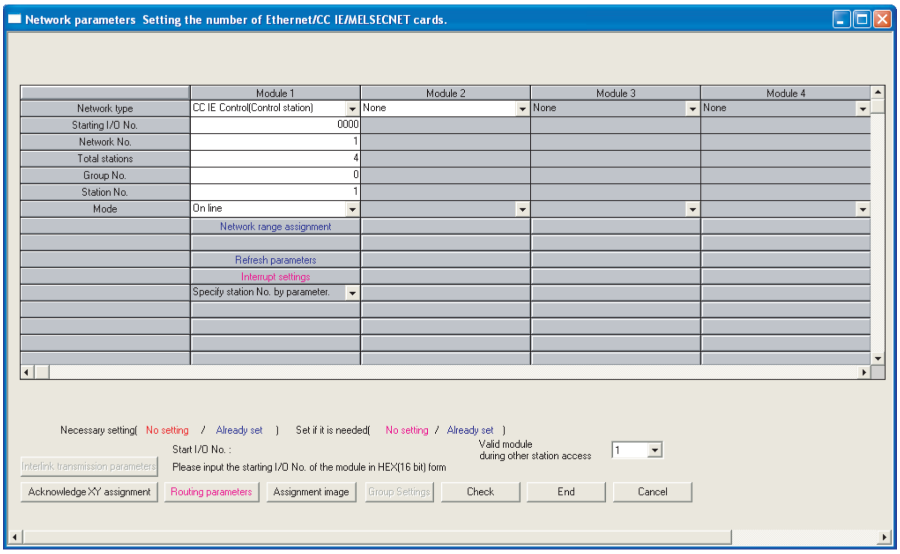
Note9.6

The Basic model QCPU cannot specify "CC IE Control (Control station)".

(1) CC-Link IE controller network setting Note9.7

The network parameter of the CC-Link IE controller network is set.

Basic
Note9.7



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 9.27 Number of modules on Ethernet/CC IE/MELSECNET.
(When CC-Link IE controller network is set)

Table9.31 CC-Link IE controller network setting list

Item	Parameter No.	Description	Setting range	Default value	Reference
Number of modules on CC-Link IE controller network	A000$_H$	Sets the network parameter of the CC-Link IE controller network.	Refer to the manual for the Q corresponding 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$_H$				
Setup Common parameters	ANM2$_H$				
Station inherent parameters	ANM3$_H$				
Group setting*1	D004$_H$				
Redundant settings*1	DA**$_H$				
Interlink transmission parameters	A002$_H$				
Routing parameters	A003$_H$				

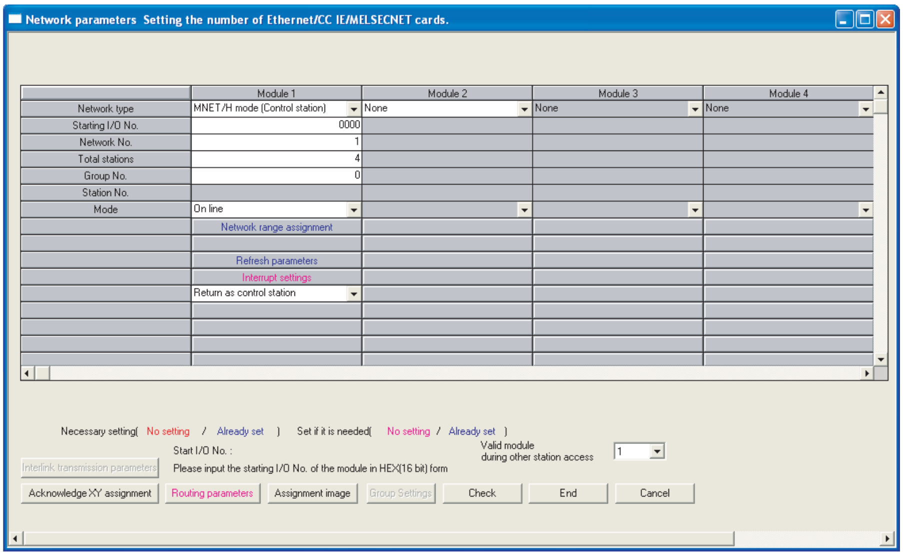
* 1 : Can be set only when the Redundant CPU is used.

Basic
Note9.7

The Basic model QCPU cannot specify "CC IE Control (Control station)".

(2) MELSECNET/H setting

The network parameter of the MELSECNET/H is 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)	None	None	None
Starting I/O No.	0000			
Network No.	1			
Total stations	4			
Group No.	0			
Station No.				
Mode	On line			
	Network range assignment			
	Refresh parameters			
	Interrupt settings			
	Return as control station			

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 9.28 Number of modules on Ethernet/CC IE/MELSECNET. (for MELSECNET/H setting)

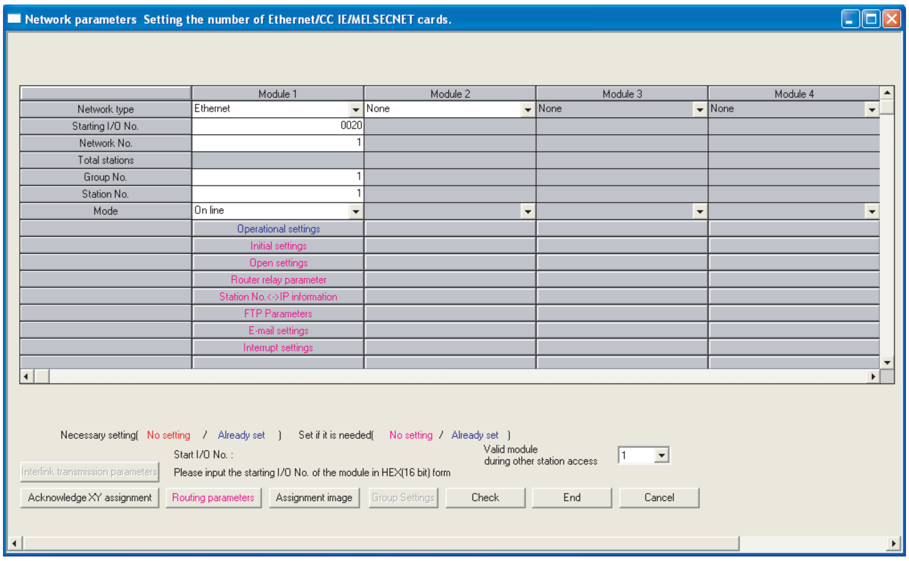
Table9.32 MELSECNET/H setting list

Item	Parameter No.	Description	Setting range	Default value	Reference
Number of modules on MELSECNET/H	5000$_H$	Set the MELSECNET/H network parameters.	Refer to the manual for the Q corresponding MELSECNET/H	----	----
Starting I/O No.	5NM0$_H$				
Network No.					
Total stations					
Group No.	05mn$_H$				
Mode	5NM0$_H$				
Refresh parameters	5NM1$_H$				
Setup Common parameters	5NM2$_H$				
Station inherent parameters	5NM3$_H$				
Sub-master parameters*2	5NM5$_H$				
Setup Common parameters 2	5NMA$_H$				
Station inherent parameters 2	5NMB$_H$				
Interrupt settings					
Group setting*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 : Can be set only when the Redundant CPU is used.
* 2 : Can be set only when the Process CPU or Redundant CPU is used.

(3) Ethernet setting

The network parameter of the Ethernet network is 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 9.29 Number of modules on Ethernet/CC IE/MELSECNET. (for Ethernet setting)

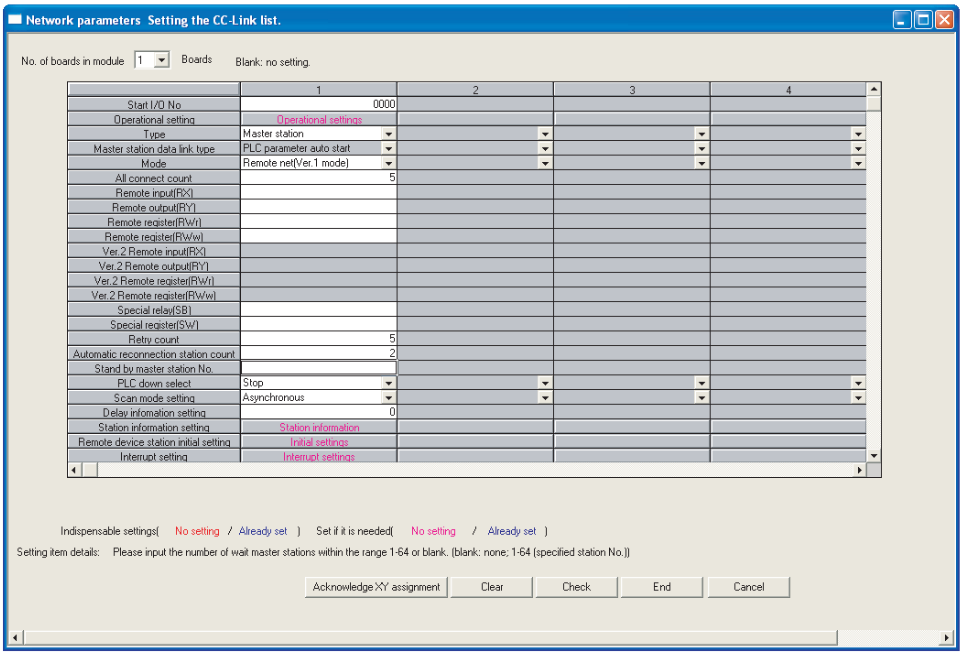
Table9.33 Ethernet setting list

Item	Parameter No.	Description	Setting range	Default value	Reference
Nunber of Ethernet	9000_H	Set the Ethernet network parameters.	Refer to the manual for the Q corresponding Ethernet.	----	----
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	$9N09_H$				
Redundant settings[*1]	$D9**_H$				
Routing parameters	$9N04_H$				
Group setting[*1]	$D004_H$				

* 1 : Can be set only when the Redundant CPU is used.

(4) CC-Link setting

The network parameter of the CC-Link is set.


Network parameters Setting the CC-Link list.
Acknowledge XY assignment
Clear
Check
End
Cancel

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	5			
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	5			
Automatic reconnection station count	2			
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: Please input the number of wait master stations within the range 1-64 or blank. (blank: none; 1-64 (specified station No.))

Figure 9.30 Network parameters Setting the CC-Link list

Table9.34 Network parameters Setting the CC-Link list

Item	Parameter No.	Description	Setting range	Default value	Reference
Number of CC-Link	C000H	Set the CC-Link parameters.	Refer to the CC-Link Manual.	----	----
Starting I/O No.	CNM2H				
Operational settings					
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					

* 1 : Can be set only when the High Performance model QCPU or Process CPU is used.

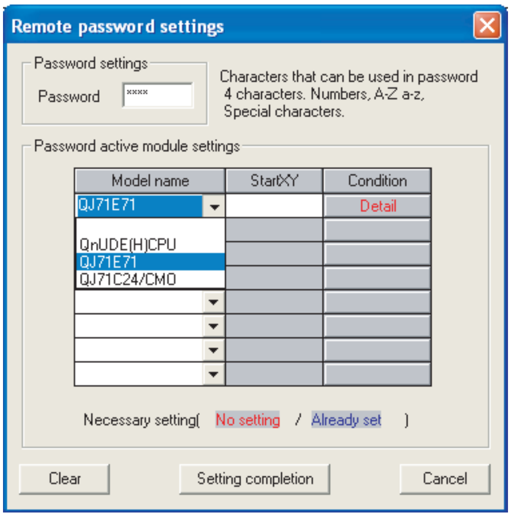
(Continued on next page)

Table9.34 Network parameters Setting the CC-Link list (Continued)

Item	Parameter No.	Description	Setting range	Default value	Reference
Special relay (SB)	CNM1H	Set the CC-Link parameters.	Refer to the CC-Link Manual.	----	----
Special register (SW)					
Retry count	CNM2H	Set the CC-Link parameters.	Refer to the CC-Link Manual.	----	----
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 settings					

9.4 Remote Password

This section explains the remote password-related parameter list and parameter details.



Remote password settings

Password settings

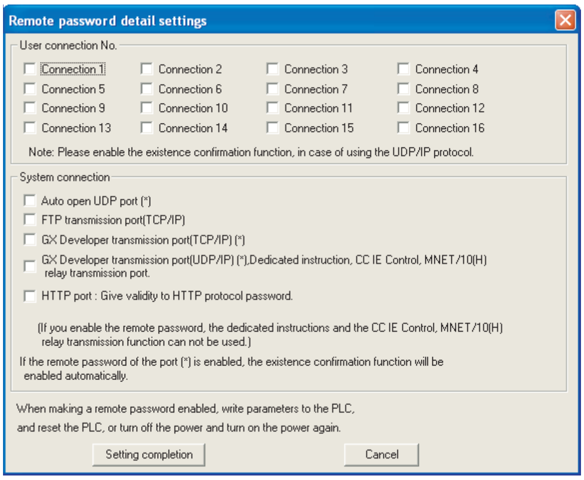
Password: ****

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
QnUDE(H)CPU QJ71E71 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 9.31 Remote password setting screens

Set the remote password of the Ethernet module, serial communication module, modem interface module and Built-in Ethernet port QCPU.

Table9.35 Remote password setting list

Item		Parameter No.	Description	Setting range	Default value	Reference
Password settings		----	Enter the remote password.	Within 4 characters (alphanumeric characters, special symbols).	----	• QnUDE(H)CPU : Section 7.7 • QJ71E71 : Manuals of the Ethernet module • QJ71C24 : Manuals of the serial communication module • QJ71CMO : Manuals of the modem interface module
Password active module settings	Model name	----	Select the module model name that will be checked for the remote password set to the CU module.	QnUDE(H)CPU QJ71E71 QJ71C24/CMO	----	
	Start XY	----	Set the starting address of the module checked for the remote password.	<Basic model QCPU> 0000H to 03E0H <High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU> 0000H to 0FE0H	----	
Detail		----	Set the remote password details in the QJ71E71.	--	----	
	User connection No.	----	Set the user connection No.	Connection 1 to 16	----	
	System connection	----	Specify the remote password-valid port of the system connection.	Specify the remote password-valid port. • 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	----	

CHAPTER10 DEVICE EXPLANATION

This chapter describes all devices that can be used in the CPU module.

10.1 Device List

The names and data ranges of devices which can be used in the CPU module are shown in Table10.1 and Table10.2.

(1) Basic model QCPU

Table10.1 Device List

Class	Type	Device Name	Default Values			Parameter Designated Setting Range	Reference Section
			Number of Points	Range Used			
Internal user devices	Bit devices	Input	2048	X0 to 7FF	Hexadecimal	Changeable within 16.4k words.*2	Section 10.2.1
		Output	2048	Y0 to 7FF	Hexadecimal		Section 10.2.2
		Internal relay	8192	M0 to 8191	Decimal		Section 10.2.3
		Latch relay	2048	L0 to 2047	Decimal		Section 10.2.4
		Annunciator	1024	F0 to 1023	Decimal		Section 10.2.5
		Edge relay	1024	V0 to 1023	Decimal		Section 10.2.6
		Step relay	2048	S0 to 127/block	Decimal		Section 10.2.9
		Link relay	2048	B0 to 7FF	Hexadecimal		Section 10.2.7
		Special link relay	1024	SB0 to 3FF	Hexadecimal		Section 10.2.8
	Word devices	Timer*1	512	T0 to 511	Decimal		Section 10.2.10
		Retentive timer*1	0	(ST0 to 511)	Decimal		
		Counter*1	512	C0 to 511	Decimal		Section 10.2.11
		Data register	11136	D0 to 11135	Decimal		Section 10.2.12
		Link register	2048	W0 to 7FF	Hexadecimal		Section 10.2.13
		Link special register	1024	SW0 to 3FF	Hexadecimal		Section 10.2.14
Internal system devices	Bit devices	Function input	16	FX0 to F	Hexadecimal	Unchangeable	Section 10.3.1
		Function output	16	FY0 to F	Hexadecimal		Section 10.3.1
		Special relay	1024	SM0 to 1023	Decimal		Section 10.3.2
	Word devices	Function register	5	FD0 to 4	Decimal		Section 10.3.1
		Special register	1024	SD0 to 1023	Decimal		Section 10.3.3
Link direct devices	Bit device	Link input	8192	Jn\X0 to 1FFF	Hexadecimal	Unchangeable	Section 10.4
		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	Hexadecimal		
Intelligent function module device	Word device	Intelligent function module device	65536	Un\G0 to 65535*3	Decimal	Unchangeable	Section 10.5
Index register	Word device	Index register	10	Z0 to 9	Decimal	Unchangeable	Section 10.6

(Continued on next page)

Table10.1 Device List (Continued)

Class	Type	Device Name		Default Values			Parameter Designated Setting Range	Reference Section
				Number of Points	Range Used			
File register	Word device	File register	Q00JCPU	0	--	--	Unchangeable	Section 10.7
			Q00CPU, Q01CPU	64k	R0 to 32767 ZR0 to 65535	Decimal		
Nesting	--	Nesting		15	N0 to 14	Decimal	Unchangeable	Section 10.9
Pointers	--	Pointer		300	P0 to 299	Decimal	Unchangeable	Section 10.10
		Interrupt pointer		128	I0 to 127	Decimal		Section 10.11
Other	Bit device	SFC block device		128	BL0 to 127	Decimal	Unchangeable	Section 10.12.1
	--	Network No. designation device		239	J1 to 239	Decimal	Unchangeable	Section 10.12.3
		I/O No. designation device	Q00JCPU	--	U0 to F	Hexadecimal		Section 10.12.4
			Q00CPU, Q01CPU	--	U0 to 3F	Hexadecimal		
	--	Macro instruction argument device		--	VD0 to □	Decimal	Unchangeable	Section 10.12.5
Constants	--	Decimal constants		K-2147483648 to 2147483647				Section 10.13.1
		Hexadecimal constants		H0 to FFFFFFFF				Section 10.13.2
		Real number constant		E ± 1.17550 - 38 to E ± 3.40282 + 38				Section 10.13.3
		Character string constants		"ABC" and "123"[*4]				Section 10.13.4

* 1 : For the timers, retentive timers and counters, their contacts and coils are bit devices and their current values are word devices.

* 2 : 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 10.2)

* 3 : The number of actually usable points changes depending on the intelligent function module. For the number of buffer memory points, refer to the manual of the intelligent function module used.

* 4 : The character string is available for only the $MOV, STR, DSTR, VAL, DVAL, ESTR and EVAL instructions.
It is unavailable for the other instructions.

(2) High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

Table10.2 Device List

Class	Type	Device Name	Default Values			Parameter Designated Setting Range	Reference Section
			Number of Points	Range Used			
Internal user devices	Bit devices	Input	8192	X0 to 1FFF	Hexadecimal	Changeable within 29k words.[*3]	Section 10.2.1
		Output[*3]	8192	Y0 to 1FFF	Hexadecimal		Section 10.2.2
		Internal relay	8192	M0 to 8191	Decimal		Section 10.2.3
		Latch relay	8192	L0 to 8191	Decimal		Section 10.2.4
		Annunciator	2048	F0 to 2047	Decimal		Section 10.2.5
		Edge relay	2048	V0 to 2047	Decimal		Section 10.2.6
		Step relay	8192	S0 to 511 / block	Decimal		Section 10.2.9
		Link special relay	2048	SB0 to 7FF	Hexadecimal		Section 10.2.8
		Link relay	8192	B0 to 1FFF	Hexadecimal		Section 10.2.7
	Word devices	Timer[*1]	2048	T0 to 2047	Decimal		Section 10.2.10
		Retentive timer[*1]	0	(ST0 to 2047)	Decimal		
		Counter[*1]	1024	C0 to 1023	Decimal		Section 10.2.11
		Data register	12288	D0 to 12287	Decimal		Section 10.2.12
		Link register	8192	W0 to 1FFF	Hexadecimal		Section 10.2.13
		Link special register	2048	SW0 to 7FF	Hexadecimal		Section 10.2.14
Internal system devices	Bit devices	Function input	5	FX0 to F	Hexadecimal	Unchangeable	Section 10.3.1
		Function output	5	FY0 to F	Hexadecimal		Section 10.3.1
		Special relay	2048	SM0 to 2047	Decimal		Section 10.3.2
	Word devices	Function register	5	FD0 to 4	Decimal		Section 10.3.1
		Special register	2048	SD0 to 2047	Decimal		Section 10.3.3
Link direct devices	Bit device	Link input	8192	Jn\X0 to 1FFF	Hexadecimal	Unchangeable	Section 10.4
		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	Hexadecimal		
Module access devices	Word device	Intelligent function module device	65536	Un\G0 to 65535[*2]	Decimal	Unchangeable	Section 10.5
		Common device for multiple CPU[*4]	4096 (14336)[*5]	U3En\G0 to 4095 (U3En\G10000 to 24335)[*6]	Decimal	Changeable	

(Continued on next page)

Table10.2 Device List (Continued)

Class	Type	Device Name	Default Values			Parameter Designated Setting Range	Reference Section
			Number of Points	Range Used			
Index register/ Standard device resister	Word device	Index register/ Standard device resister	16 (20)[*5]	Z0 to 15 (Z0 to 19)[*6]	Decimal	Unchangeable	Section 10.6
File register	Word device	File register	0	--	--	0 to 1018 k points (1 k units)[*5*11]	Section 10.7
Extended data register[*12]	Word device	Extended data register	0	--	--		Section 10.8
Extended link register[*12]	Word device	Extended link register	0	--	--		Section 10.8
Nesting	--	Nesting	15	N0 to 14	Decimal	Unchangeable	Section 10.9
Pointers	--	Pointer	4096	P0 to 4095	Decimal	Unchangeable	Section 10.10
		Interrupt pointer	256	I0 to 255	Decimal		Section 10.11
Other	Bit devices	SFC block device	320	BL0 to 319	Decimal	Unchangeable	Section 10.12.1
		SFC transition device[*9]	512	TR0 to 511	Decimal		Section 10.12.2
	--	Network No. specification device	255	J1 to 255	Decimal		Section 10.12.3
		I/O No. specification device	--	U0 to FF	Hexadecimal		Section 10.12.4
		Macro instruction argument device	--	VD0 to □	Decimal		Section 10.12.5
Constants	--	Decimal constants	K-2147483648 to 2147483647				Section 10.13.1
		Hexadecimal constants	H0 to FFFFFFFF				Section 10.13.2
		Real number constants	Single-precision floating-point data: E ± 1.17549435 - 38 to E ± 3.40282347 + 38 Double-precision floating-point data[*7*8]: E ± 2.2250738585072014 - 308 to E ± 1.7976931348623157 + 308				Section 10.13.3
		Character string constants	"ABC" and "123"				Section 10.13.4

Process Redundant Universal
Note10.1 Note10.1 Note10.1

* 1 : For the timer, retentive timer, and counter, bit devices are used for the "contact" and the "coil", and the word device is used for the "present value".

* 2 : The actual number of usable points varies according to the intelligent/special function module *Note10.1*.

For details regarding the buffer memory's "number of points", refer to the Intelligent/Special Function Module Manual.

* 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 10.2)

In the Universal model QCPU whose serial number (first five digits) is "10042" or later, 0 point can be set for step relays.

* 4 : Can be used only when a multiple CPU system is configured.

* 5 : The points in the parentheses are points when using the Universal model QCPU

* 6 : The points in the parentheses are available when the Universal model QCPU is used.

* 7 : The double-precision floating-point data can only be used for the Universal model QCPU.

* 8 : Up to 15 digits can be input by GX Developer.

* 9 : The Universal model QCPU cannot use the SFC transition switch.

* 10: Device points and range for the Q02UCPU are 128 points (BL0 to 127).

* 11: The setting value indicates the total number of points for the file register, extended data register (D), and extended link register (W).

* 12: The extended data register (D) and extended link register (W) can only be used for the Universal model QCPU.

Process Redundant Universal
Note10.1 Note10.1 Note10.1

The Process CPU, Redundant CPU and Universal model QCPU do not support the AnS/A series corresponding modules.

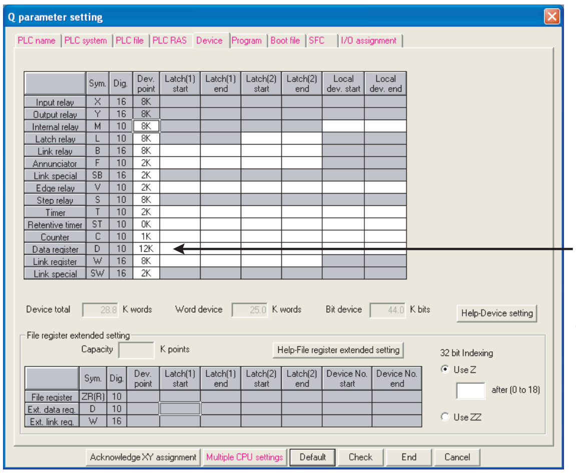
10.2 Internal User Devices

(1) Definition

Internal user devices can be used for various user applications.

The "number of usable points" setting is designated in advance (default value) for internal user devices.

However, this setting can be changed at the "Device" tab screen in the "(PLC) Parameter" dialog box.



	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						

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

Default value
For device whose number of points can be changed, number of used points can be changed.

Figure 10.1 Device in PLC parameter dialog box

(2) Internal user device setting range

Universal UD Note10.2

Universal UD Note10.3

The number of used internal user devices other than the input (X), output (Y), step relay (S)Note10.2, link special relay (SB)Note10.3and link special register (SW)Note10.3of the CPU module can be changed within the following range in the device setting of the PLC parameter dialog box. (Table10.3)

Table10.3 Internal user device setting range

CPU module	Changeable range
Basic model QCPU	16.4k words (Including 1.5k words for the above devices)
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	29k words (Including 3.7k words for the above devices)

The following gives more information.

Universal UD Note10.2

In the Universal model QCPU whose serial number (first five digits) is "10042" or later, 0 points can be set for the step relay. (☞ Section 10.2.9)

Universal UD Note10.3

For the Universal model QCPU, the number of usable points of the link special relay (SB) and link special register (SW) can be changed within the range indicated in Table10.3 in the device setting of the PLC parameter dialog box.

(a) **Setting range**

The number of device points can be specified in units of 16-point.

Up to 32k/60k points can be set for one device. [*1]

One point is calculated as two points for timers, retentive timers, and counters (one for coil, one for contact).

[*1] : In the Universal model QCPU whose serial number (first five digits) is "10042" or later, up to 60k points can be set for internal relays and link relays.

☒POINT

1. When changing the device points, the refresh ranges of the network modules and the auto refresh ranges of the intelligent function modules must be set so as not to be out of the out of the corresponding device points.
   If the device points are set exceeding the corresponding device range, the data may be written into other devices or an error may occur.
2. The total number of points for internal relays, latch relays, annunciators, edge relays, link relays, link special relays, step relays, timers, retentive timers, and counters can be set within the following range.

Table10.4 The total number of points for internal relays, latch relays, annunciators, edge relays, link relays, link special relays, step relays, timers, retentive timers, and counters

CPU module	Total number of points
High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU whose serial number (first five digits) is "10041" or earlier	Up to 64k points
Universal model QCPU whose serial number (first five digits) is "10042" or later	No limit (Up to 29k words for internal user devices)

(3) Memory capacity

Use the following expression to obtain the memory capacity of an internal user device.

(For the Basic model QCPU)

(Bit device capacity) + (Word device capacity) + (Timer, retentive timer and counter capacity) ≦ 16.4k words

(For the High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU)

(Bit devices capacity) + (Word devices capacity) + (Timer, retentive timer and counter capacity) ≦ 29k

(a) For bit devices:

For bit devices, 16 points are calculated as 1 word.

$$(\text{Bit device capacity}) = \frac{(X+Y+M+L+B+F+SB+V+S)}{16}\ (\text{Word})$$

(b) For timer (T) retentive timer (ST), and Counter (C):

For the timer, retentive timer, and counter, 16 points are calculated as 18 words.

$$(\text{Timer, retentive, counter capacity}) = \frac{(T + ST + C)}{16} \times 18\ (\text{words})$$

(c) For word devices:

As for data register (D), link register (W), and link special register (SW), 16 points is calculated as 16 words.

$$(\text{Word device capacity}) = \frac{(D + W + SW)}{16} \times 16\ (\text{words})$$

☒POINT

When an internal user device's "number of usable points" setting is changed, the following files which were created under the previous setting cannot be used as they are.

- The sequence program
- The SFC program
- ST program

When the number of used internal user device points is changed, read the above program from the CPU module to GX Developer, and write it to the CPU module again.

(4) Device point assignment example

A device point assignment example for the High Performance model QCPU is shown in Table10.5.

Table10.5 is based on the device point assignment sheet shown in Appendix 9.

Table10.5 Device point assignment example (for High Performance model QCPU)

Device name	Symbol	Numeric notation	Number of device points[1][2]		Restriction check	
			Number of points	Number	Capacity (Word)[3]	Number of bit points[2]
Input relay	X	16	8k (8192) points	X0000 to 1FFF	÷ 16 512 words	× 1 8192 points
Output relay	Y	16	8k (8192) points	Y0000 to 1FFF	÷ 16 512 words	× 1 8192 points
Internal relay	M	10	16k (16384) points	M0 to 16383	÷ 16 1024 words	× 1 16384 points
Latch relay	L	10	4k (4096) points	L0 to 4095	÷ 16 256 words	× 1 4096 points
Link relay	B	16	4k (4096) points	B0000 to 0FFF	÷ 16 256 words	× 1 4096 points
Annunciator	F	10	1k (1024) points	F0 to 1023	÷ 16 64 words	× 1 1024 points
Link special relay	SB	16	2k (2048) points	SB0000 to 07FF	÷ 16 128 words	× 1 2048 points
Edge relay	V	10	1k (1024) points	V0 to 1023	÷ 16 64 words	× 1 1024 points
Step relay	S	10	8k (8192) points	S0 to 8191	÷ 16 512 words	× 1 8192 points
Timer	T	10	2k (2048) points	T0 to 2047	$\times \frac{18}{16}$ 2304 words	× 2 4096 points
Retentive timer	ST	10	2k (2048) points	ST0 to 2047	$\times \frac{18}{16}$ 2304 words	× 2 4096 points
Counter	C	10	1k (1024) points	C0 to 1023	$\times \frac{18}{16}$ 1152 words	× 2 2048 points
Data register	D	10	14k (14336) points	D0 to 14335	× 1 14336 words	--
Link register	W	16	4k (4096) points	W0000 to 4095	× 1 4096 words	--
Link special register	SW	16	2k (2048) points	SW0000 to 07FF	× 1 2048 words	--
Device total					29568 words (29696 words or less)	63488 points (65536 points or less)

[1] : □ The hatched number of points is fixed. (Unchangeable)

[2] : The maximum number of points of one device is 32k points.

[3] : Enter the value that is obtained by multiplying (or dividing) the number of device points by the numeral indicated in the capacity (Word) field.

10.2.1 Input (X)

(1) Definition

Inputs transmit commands or data to the High Performance model QCPU from an external device such as push-button switches, selector switches, limit switches, digital switches.

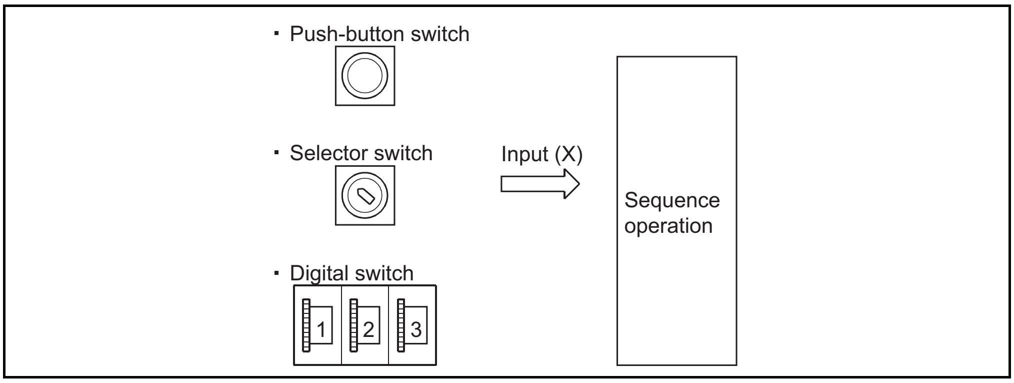


Figure 10.2 Commands from external devices to CPU module

(2) Concept of input (X)

If the input point is the Xn virtual relay inside the High Performance model QCPU, the program uses the Xn's N/O contact or N/C contact.

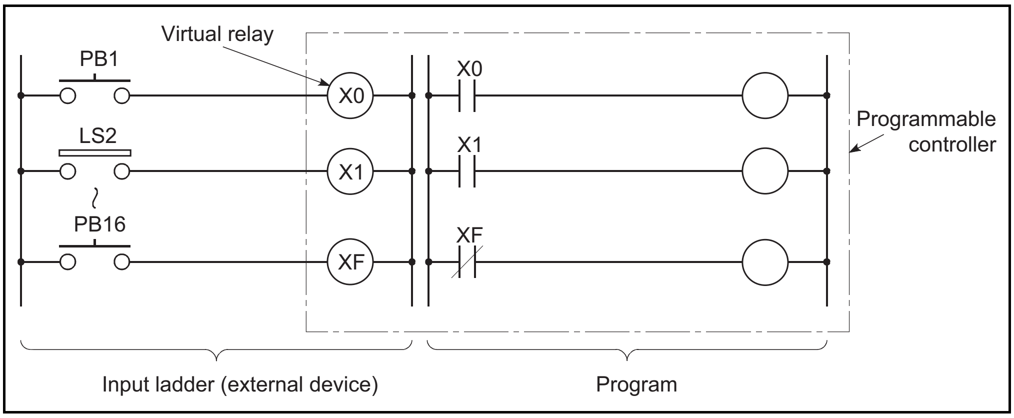


Figure 10.3 Inputs(X)

(3) Number of used N/O and N/C contacts

There are no restrictions on the number of Xn N/O contacts and N/C contacts used in a program, provided the program capacity is not exceeded.

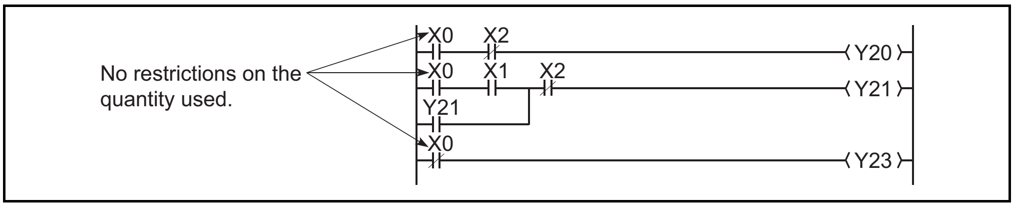


Figure 10.4 Input(X) Used in Program

☒POINT

1. When debugging a program, an input (X) can be set to ON/OFF as described below.
   - GX Developer test operation
   - OUT Xn instruction

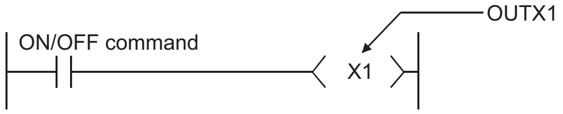


Figure 10.5 Input(X) ON/OFF by the OUT Xn instruction

2. The input (X) can also be used in the following cases.
   - Refresh target (CPU module side) device of RX of CC-Link
   - Refresh target (CPU module side) device of the CC-Link IE controller network or MELSECNET/H

10.2.2 Output (Y)

(1) Definition

Outputs give out the program control results to the external devices such as solenoid, electromagnetic switch, signal lamp and digital display.

Outputs give out the result equivalent to one N/O contact.

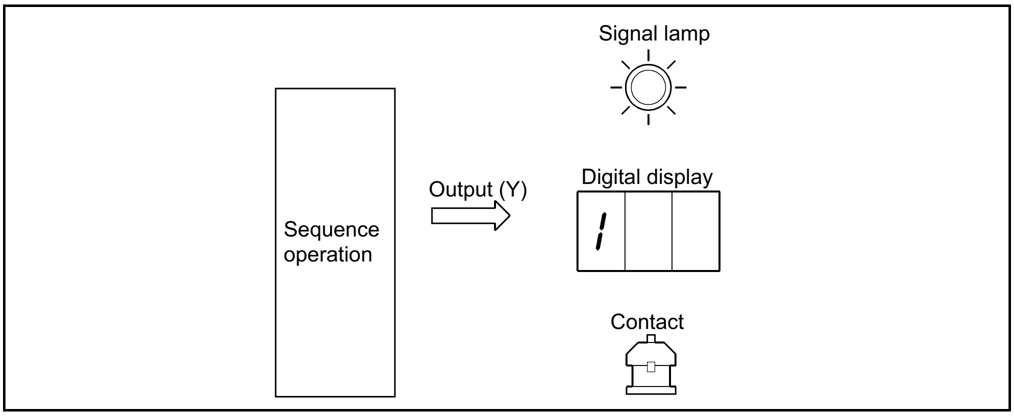


Figure 10.6 Output from CPU module to external devices

(2) Number of used N/O and N/C contacts

There are no restrictions on the number of output Yn N/O contacts and N/C contacts used in a program, provided the program capacity is not exceeded.

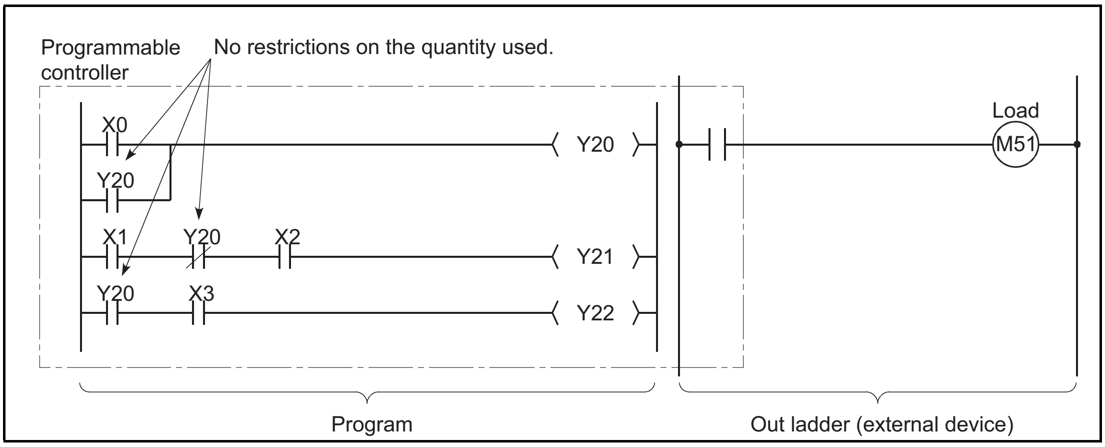

Programmable controller
No restrictions on the quantity used.
X0
Y20
Y20
X1
Y20
X2
Y21
Y20
X3
Y22
Load
M51
Program
Out ladder (external device)

Figure 10.7 Use of output (Y) in program

(3) Using outputs as internal relays (M)

Output (Y) corresponding to the slots installed with input modules and empty slots can serve as internal relays (M).

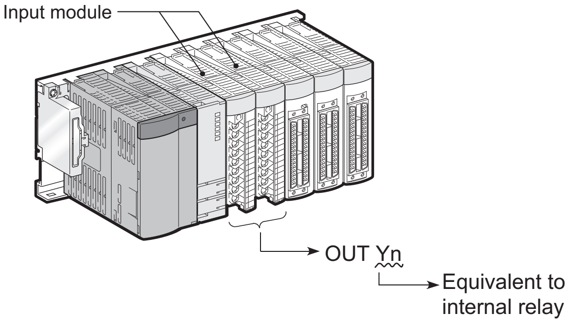


Figure 10.8 Substitute for internal relay

10.2.3 Internal relay (M)

(1) Definition

Internal relays are auxiliary relays used in the CPU module.
All internal relays are switched OFF at the following times:

- When the PLC is powered OFF and then ON.
- When the CPU module is reset.
- When latch clear operation is executed.(☞ Section 6.3)

(2) Latch execution of internal devices (backed up for power failure)

(a) For Basic model QCPU, High Performance model QCPU, Process CPU, Universal model QCPU

The internal relays cannot be latched (backed up for power failure).

(b) For Redundant CPU

When the start mode is set to the hot start mode, the internal relays are latched (backed up for power failure). When the start mode is set to the initial start mode, the internal relays are not latched (backed up for power failure).

(3) Number of used N/O and N/C contacts

There are no restrictions on the number of contacts (N/O contacts, N/C contacts) used in the program, provided the program capacity is not exceded.

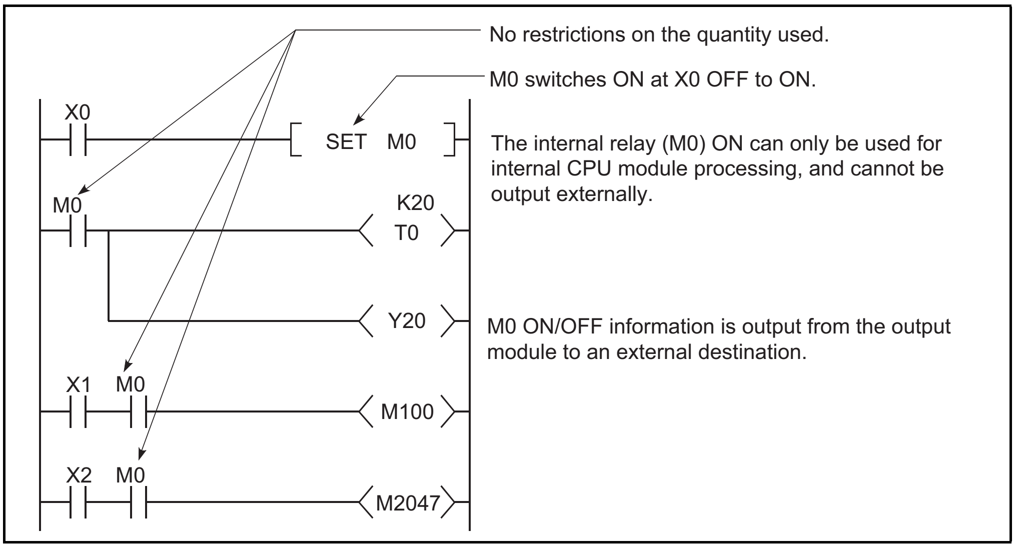

No restrictions on the quantity used.
M0 switches ON at X0 OFF to ON.
X0
SET M0
The internal relay (M0) ON can only be used for internal CPU module processing, and cannot be output externally.
M0
K20
T0
Y20
M0 ON/OFF information is output from the output module to an external destination.
X1
M0
M100
X2
M0
M2047

Figure 10.9 Use of internal relays in program

(4) Procedure for external outputs

Outputs (Y) are used to output sequence program operation results to an external destination.

☒POINT

Latch relays (L) should be used when a latch (memory backup) is required.(☞ Section 10.2.4)

10.2.4 Latch relay (L)

(1) Definition

Latch relays are auxiliary relays which can be latched by the programmable controller's internal latch (memory backup).

Latch relay operation results (ON/OFF information) are saved even in the following cases:

- When the PLC is powered OFF and then ON
- When the CPU module is reset

The latch is backed up by the CPU module battery.

(2) Latch relay clear

Basic Universal
Note10.4 Note10.4

Turn OFF the latch relays by latch clear operation.(☞ Section 3.7)
However, the latch relay set to "Latch (2) first/last" in the device setting of the PLC parameter dialog box cannot be turned OFF if latch clear is performed by the RESET/L.CLR switch Note10.4.

(3) Number of used N/O and N/C contacts

There are no restrictions on the number of contacts (N/O contacts, N/C contacts) used in the program, provided the program capacity is not exceeded.

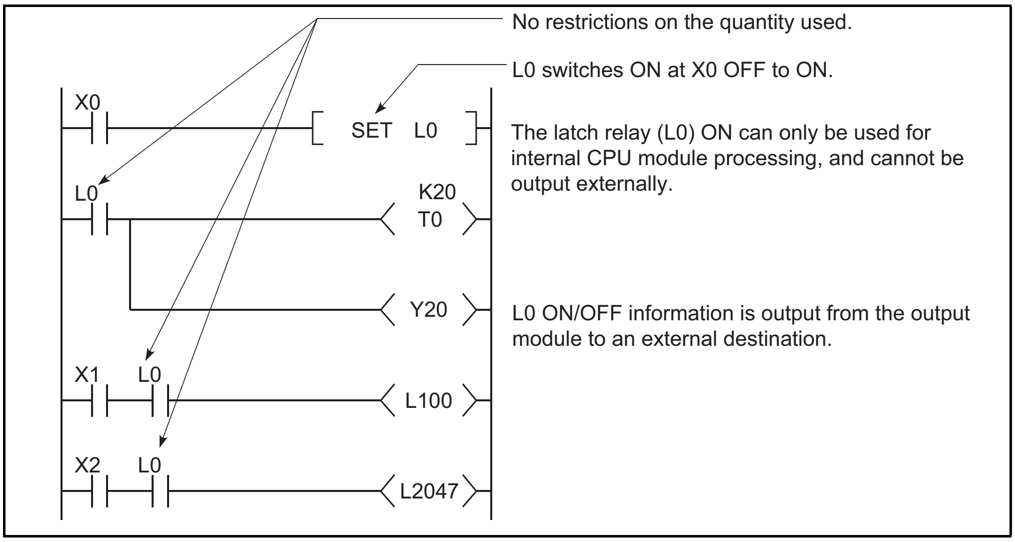

No restrictions on the quantity used.
L0 switches ON at X0 OFF to ON.
X0
SET L0
The latch relay (L0) ON can only be used for internal CPU module processing, and cannot be output externally.
L0
K20
T0
Y20
L0 ON/OFF information is output from the output module to an external destination.
X1 L0
L100
X2 L0
L2047

Figure 10.10 Latch Relay

☒POINT

For the Universal model QCPU, the scan time is prolonged when the latch relay (L) is used. Reducing the number of latch relay (L) points can reduce the prolonging scan time..(☞Section 10.1.2 (11))

Basic Universal
Note10.4 Note10.4

In the case of the Basic model QCPU and Universal model QCPU, latch clear cannot be performedd by switch operation.

(4) Procedure for external outputs

Outputs (Y) are used to output sequence program operation results to an external destination.

☒POINT

1. Internal relays (M) should be used when a latch (memory backup) is not required.(☞ Section 10.2.3)
2. For latch clear, the latch clear disabled range can be set to each device in the device setting of the PLC parameter dialog box.(☞ Section 6.3)

10.2.5 Annunciator (F)

(1) Definition

Annunciators are internal relays used for fault detection programs created by the user.

(2) Special relay and special registers at annunciator ON

When annunciators switch ON, a special relay (SM62) switches ON, and the Nos. and quantity of the annunciators which switched ON are stored at the special registers (SD62 to 79).

• Special relay	:	SM62 • • • • •	Switches ON if even one annunciator switches ON.
• Special register	:	SD62 • • • • •	No. of first annunciator which switched ON is stored here.
		SD63 • • • • •	The number (quantity) of annunciators which are ON is stored here.
		SD64 to 79 • • •	Annunciator Nos. are stored in the order in which they switched ON. (The same annunciator No. is stored at SD62 and SD64.)

The annunciator No. stored at SD62 is also registered in the "error history strage area".

☒POINT

For the Basic model QCPU, only one annunciator No. is stored into the error history strage area while the PLC power is ON.

(3) Applications of annunciators

Using annunciators for a fault detection program, an equipment fault or fault presence/absence (annunciator number) can be checked by monitoring the special register (SD62 to 79) when the special relay (SM62) switches ON.

Example

The program which outputs the No. of the ON annunciator (F5).

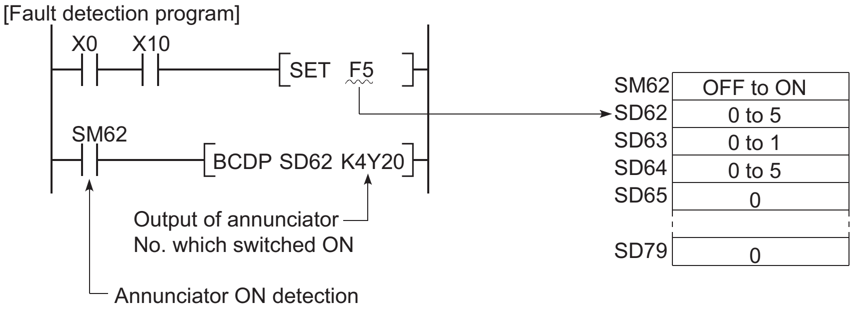


Figure 10.11 Detection and storage of annunciator ON

(4) Number of used N/O and N/C contacts

There are no restrictions on the number of contacts (N/O contacts, NC contacts) used in the program, provided the program capacity is not exceeded.

(5) Annunciator ON procedure

(a) Annunciator ON procedure

The annunciator can be turned ON by either of the following instructions.

1) SET F□ instruction

The SET F□ instruction turns ON the annunciator only on the leading edge (OFF to ON) of the input condition.
If the input condition turns OFF, the annunciator is held ON.
The scan time can be reduced by using many annuciators, compared with the OUT F□ instruction.

2) OUT F□ instruction

The annunciator can be turned ON/OFF by the OUT F□ instruction, but it takes longer time than the SET F instruction since it performs processing every scan.

Basic Note10.5

If the annunciator is turned OFF by the OUT F□ instruction, the RST F□ instruction/LEDR instruction[Note10.5]/BKRST instruction must be executed. For these reasons, use the SET F□ instruction to turn ON the annunciator.

☒POINT

1. If switched ON by any method other than the SET F□ and OUT F□ instructions, the annunciator functions in the same way as the internal relay.
   Does not switch ON at SM62, and annunciator Nos. are not stored at SD62, SD64 to 79.
2. When the annunciator is tracked from the control system to the standby system by the tracking function of the Redundant CPU, the annunciator in the standby system will not turn ON.
   Refer to the following manual for the processing when the annunciator is tracked.
   ☞QnPRHCPU User's Manual (Redundant System)

Basic Note10.5

The Basic model QCPU cannot use the LEDR instruction.

(b) Processing at annunciator ON

1) **Data stored at special registers (SD62 to 79)**

- Nos. of annunciators which switched ON are stored in order at SD64 to 79.
- The annunciator No. which was stored at SD64 is stored at SD62.
- "1" is added to the SD63 value.

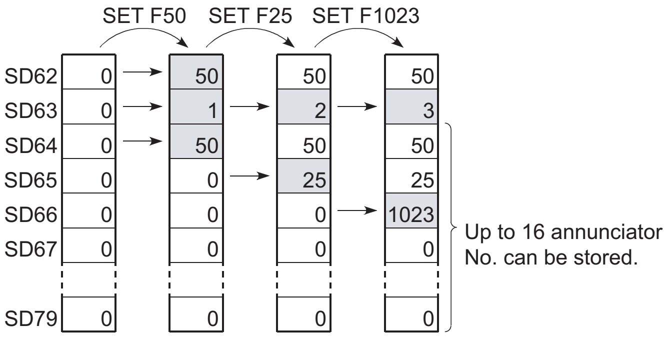


Figure 10.12 Processing at annunciator ON

2) **Processing at CPU**

- When Basic model QCPU is used
  The ERR LED on the module front turns ON.
- When High Performance model QCPU, Process CPU or Redundant CPU, Universal model QCPU is used
  The USER LED (red) on the module front turns ON.

3) **Selection of LED ON/OFF**

Setting LED indication priority to SD207 to SD209 when an error occurs enables to select to turn ON or OFF USER.LED when an annunciator is turned ON. (For the Basic model QCPU, ERR.LED.)

(6) Annunciator OFF procedure and processing content

(a) Annunciator OFF procedure

The annunciator can be turned OFF by any of the following instructions.

1) **RST F□ instruction**

An annunciator No. which has been switched ON by the SET F□ instruction can be switched OFF by the RST F□ instruction.

Basic Note10.6

2) **LEDR instruction**Note10.6

The LEDR instruction is used to switch OFF the annunciator Nos. stored at SD62 and SD64.

3) **BKRST instruction**

The BKRST instruction is used to switch OFF all the annunciator Nos. within a specified range.

4) **OUT F□ instruction**

The OUT F□ instruction can execute ON/OFF of the annunciator No. by the same instruction.

However, if an annunciator is switched OFF by the OUT F□ instruction, the "processing at annunciator OFF" ((6)(b) in this section) is not performed.

Execute the RST F□, LEDR or BKRST instructions after the annunciator has been switched OFF by the OUT F□ instruction.

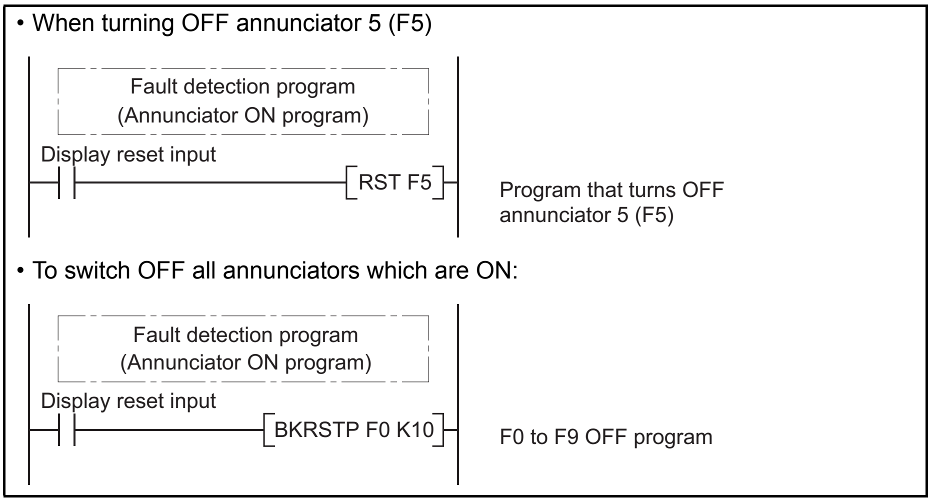


Figure 10.13 Example of turning OFF the annunciators

Remark

Refer to the following manual for details of each instruction.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Note10.6

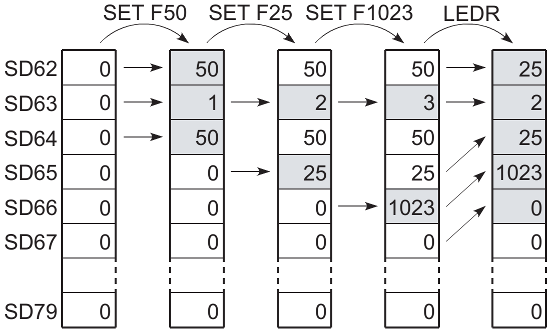
The Basic model QCPU cannot use the LEDR instruction.

(b) Processing at annunciator OFF

1) Special register (SD62 to 79) data operation at execution of LEDR instruction Note10.7

Basic
Note10.7

- The annunciator No. stored in SD64 is deleted, and the annunciator Nos. stored in SD65 and later are shifted up.
- The annunciator No. stored in SD64 is stored into SD62.
- "-1" is subtracted from the SD63 value.
- If the SD63 value is "0", SM62 is switched OFF.



SD62	0	50	50	50	25
SD63	0	1	2	3	2
SD64	0	50	50	50	25
SD65	0	0	25	25	1023
SD66	0	0	0	1023	0
SD67	0	0	0	0	0
SD79	0	0	0	0	0

Figure 10.14 Processing at annunciator OFF (when LEDR instruction is executed)

Basic
Note10.7

The Basic model QCPU cannot use the LEDR instruction.

2) **Special register (SD62 to 79) data operation when annunciator is tunred OFF by executing the RST F⬚ instruction or BKRST instruction**

- The annunciator No. specified by the RST instruction/BKRST instruction is deleted, and the stored annunciator Nos. after the deleted annunciator No. are shifted up.
- If the annunciator No. stored at SD64 was switched OFF, the new annunciator No. which is stored at SD64 is stored at SD62.
- 1 is subtracted from the SD63 value.
- If the SD63 value is "0", SM62 is switched OFF.

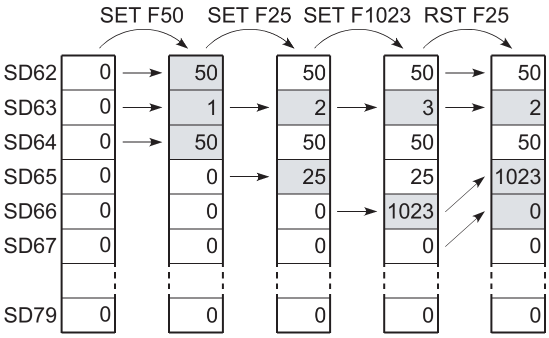


Figure 10.15 Processing at annunciator OFF (when REST F⬚ instruction is executed)

3) **LED indication**

When the annunciator Nos. in SD64 to 79 all turn OFF, the LED (refer to (5)(b) in this section), which was turned ON as the annunciator turned ON, turns OFF.

☒POINT

If the LEDR instruction is executed when annunciator ON and the operation continuation error having higher priority (☞ Section 6.21.2) than the annunciator occur simultaneously, the LEDR instruction clears the error having higher priority (☞ Section 6.21.2) than the annunciator.
Hence, the annunciator cannot be turned OFF if the LEDR instruction is executed.
When using the LEDR instruction to turn OFF the annunciator, execute the LEDR instruction after removing the cause of the error having higher priority than the annunciator. Note10.8

Basic
Note10.8

Basic
Note10.8

The Basic model QCPU cannot use the LEDR instruction.

10.2.6 Edge relay (V)

(1) Definition

An edge relay is a device which stores the operation results (ON/OFF information) from the beginning of the ladder block.

Edge relays can only be used at contacts, and cannot be used as coils.

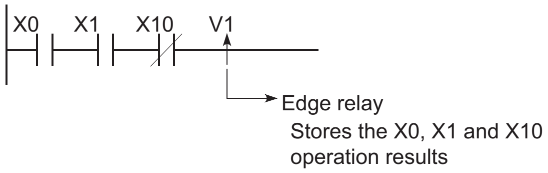
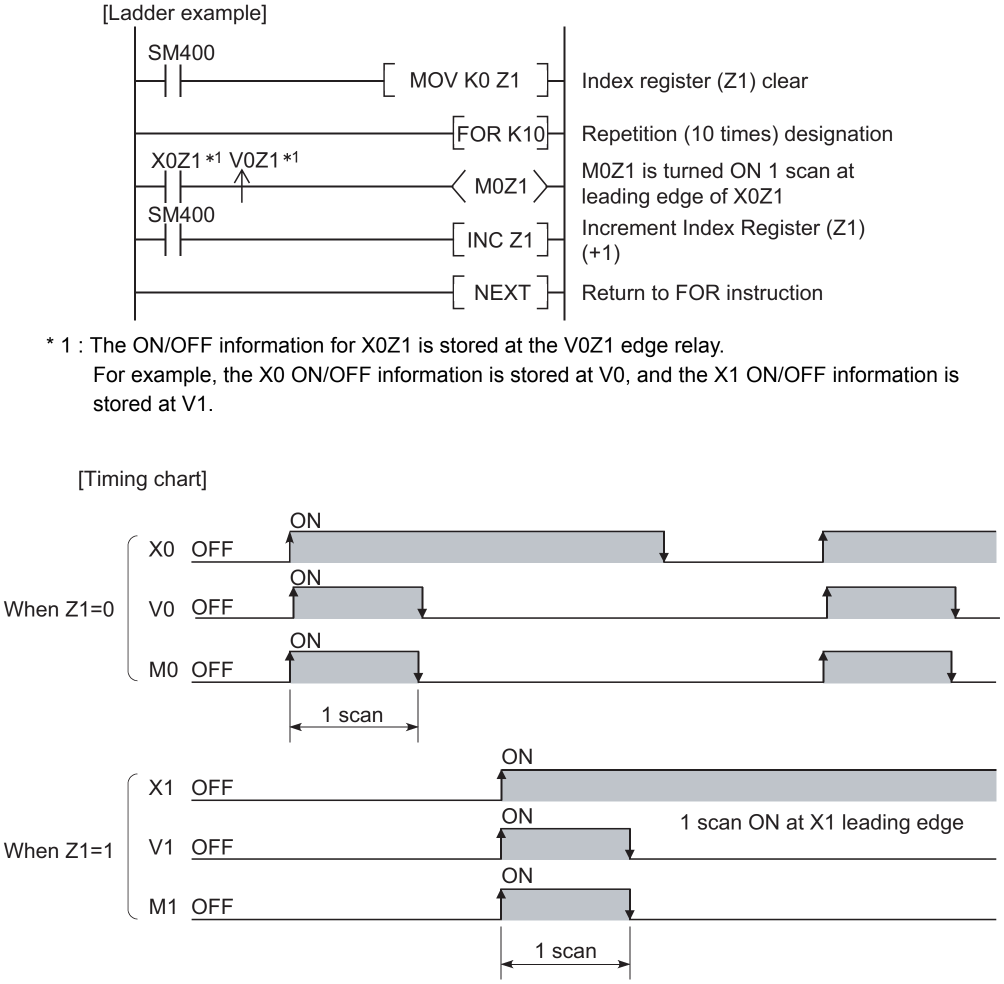


Figure 10.16 Edge relay

(2) Edge relay applications

Edge relays are used for detecting the leading edge (OFF to ON) in programs configured using index modification.


[Ladder example]
SM400
MOV K0 Z1
Index register (Z1) clear
FOR K10
Repetition (10 times) designation
X0Z1 *1
V0Z1 *1
M0Z1
M0Z1 is turned ON 1 scan at leading edge of X0Z1
SM400
INC Z1
Increment Index Register (Z1) (+1)
NEXT
Return to FOR instruction
[Timing chart]
When Z1=0
X0
V0
M0
ON
OFF
1 scan
When Z1=1
X1
V1
M1
ON
OFF
1 scan ON at X1 leading edge
1 scan

* 1 : The ON/OFF information for X0Z1 is stored at the V0Z1 edge relay.
For example, the X0 ON/OFF information is stored at V0, and the X1 ON/OFF information is stored at V1.

Figure 10.17 Application example and timing chart of edge relay

(3) Precautions

The edge relay of the same No. cannot be set in multiple steps of a program.

10.2.7 Link relay (B)

(1) Definition

Link relays are CPU module side relays used when refreshing the link relay (LB) data of the MELECNET/H network module, etc. to the CPU module or when refreshing the CPU module data to the link relays (LB) of the MELECNET/H network module, etc.

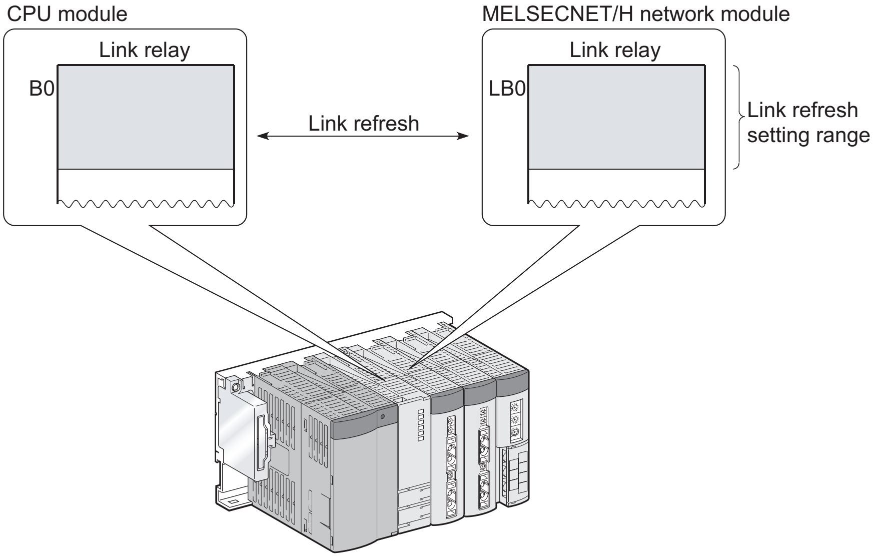


Figure 10.18 Link refresh

(2) Number of used N/O and N/C contacts

There are no restrictions on the number of contacts (N/O contacts, N/C contacts) used in the program.

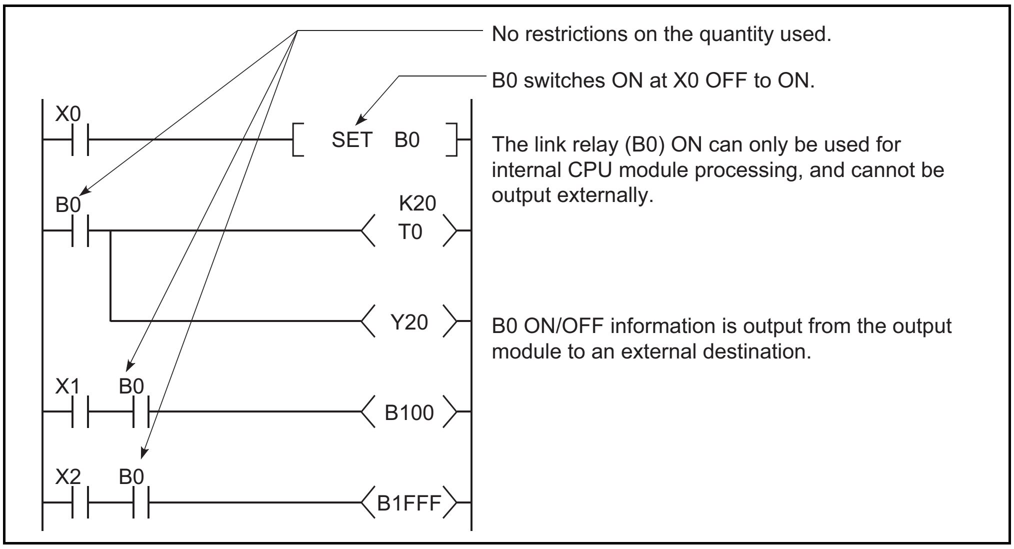

No restrictions on the quantity used.
B0 switches ON at X0 OFF to ON.
X0
SET B0
The link relay (B0) ON can only be used for internal CPU module processing, and cannot be output externally.
B0
K20
T0
Y20
B0 ON/OFF information is output from the output module to an external destination.
X1
B0
B100
X2
B0
B1FFF

Figure 10.19 Link Relay

(3) Using link relays in the network system

In order to use link relays in the network system, a network parameter setting is required.

The link relay range where network parameter setting has not been made (not used by the MELSECNET/H network system, etc.) is available as the internal relays or latch relays.

- Range where no link relay latch is performed••Internal relay
- Range where link relay latch is performed••••••Latch relay

☒POINT

- Although the number of device points for link relays in the CC-Link IE controller network module is 32768 points, the default number of points for link relays in the CPU module is 2048 points (Basic model QCPU) or 8192 points (High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU).
- Although the number of device points for link relays in the MELSECNET/H network module is 16384 points, the default number of points for link relays in the CPU module is 2048 points (Basic model QCPU) or 8192 points (High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU).

To use link relays exceeding the point described above, change the device point setting for link relays on the Device tab of PLC parameter in GX Developer.

Remark

Refer to the following manual for the network parameters.

☞ CC-Link IE Controller Network Reference Manual

☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

10.2.8 Link special relay (SB)

(1) Definition

A link special relay indicates the communication status and error detection of an intelligent function module, such as CC-Link IE controller network module and MELSECNET/H network module.

ON/OFF of the link special relays are controlled by various causes that occur during data link.

By monitoring the link special relays, the communication status, error status and others of data link can be grasped.

(2) Number of link special relay points

The number of link special relay points is as described in Table10.6.

Table10.6 Number of link special relay points of each CPU module

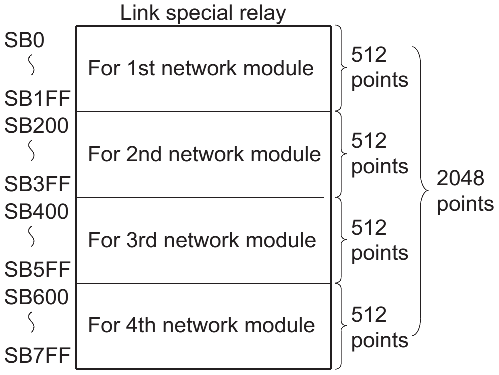
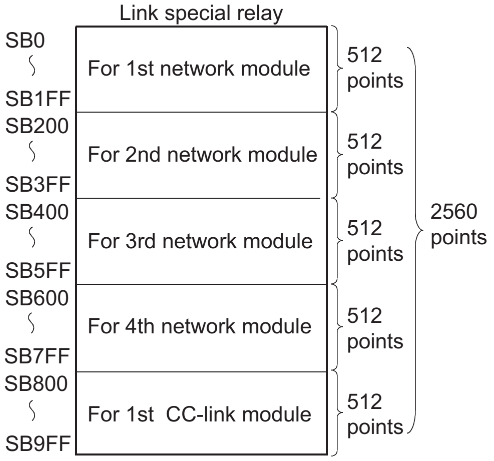
CPU module	Number of link special relay points
Basic model QCPU	1024 points (SB0 to 3FF). 512 points assigned for the intelligent function module having link special relays, such as CC-Link IE controller network module and MELSECNET/H network module.
High Performance model QCPU Process CPU Redundant CPU	2048 points (SB0 to 7FF). 512 points assigned to the intelligent function module having link special relays, such as CC-Link IE controller network module and MELSECNET/H network module. The link special relays can be assigned as shown below.  Link special relay SB0 SB1FF For 1st network module 512 points SB200 SB3FF For 2nd network module 512 points SB400 SB5FF For 3rd network module 512 points SB600 SB7FF For 4th network module 512 points 2048 points

Table10.6 Number of link special relay points of each CPU module (continued)

CPU module	Number of link special relay points
Universal model QCPU	2048 points (SB0 to 7FF) However, this setting can be changed in the device setting of the PLC parameter dialog box. (Section 10.1(2)) For the intelligent function module, such as the CC-Link IE controller network module and MELSECNET/H network module, having link special relays, 512 points are assigned. Assigning the link special relay as shown below enables refreshing of the CC-Link link special relay (SB) for the link special relay (SB) of the CPU module.  Link special relay SB0 ⟆ SB1FF For 1st network module 512 points SB200 ⟆ SB3FF For 2nd network module 512 points SB400 ⟆ SB5FF For 3rd network module 512 points SB600 ⟆ SB7FF For 4th network module 512 points SB800 ⟆ SB9FF For 1st CC-link module 512 points 2560 points

Remark

For details of the link special relays, refer to the manual of the intelligent function module having link special relays.

10.2.9 Step relay (S)

Step relays are devices for SFC programs.

Refer to the following manual for how to use the step relays.

☞QCPU (Q Mode)/QnACPU Programming Manual (SFC)

☒POINT

Because the step relay is a device exclusively for the SFC program, it cannot be used as an internal relay in the sequence program.

If used in this manner a SFC error will occur, and system operation will be stopped (system down).

10.2.10 Timer (T)

(1) Definition

A timer (T) is a device that starts counting when its coil turns ON, and times-out and turns ON its contact when the current value reaches or exceeds the set value.
The timer is of an up-counting type.
The current value matches the set value when a "time-out" occurs.

(2) Timer types

There are two types of timers: a low/high speed that allows the current value to return to "0" when a timer coil switches OFF, and a retentive timer that retains the current value even when a timer coil switches OFF.

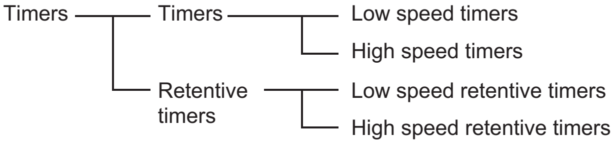


Figure 10.20 Timer types

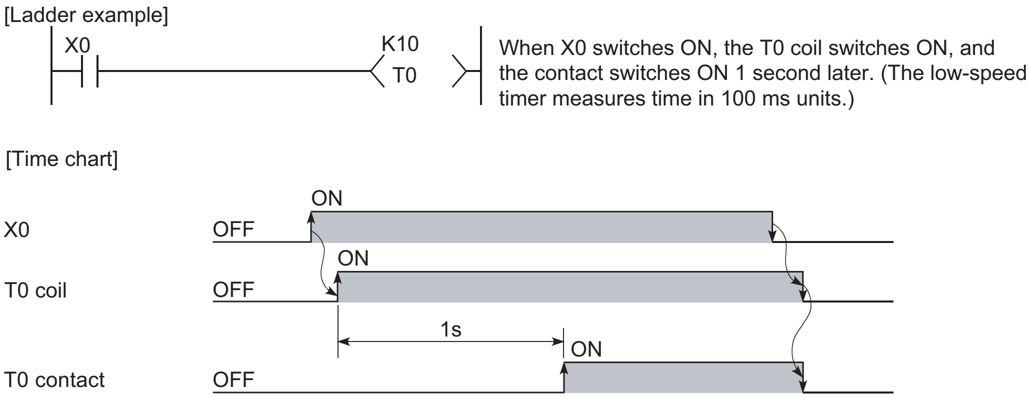
(3) How to use timers

With a timer setting (instruction format), a device is assigned for a low speed timer or high speed timer. The OUT T0 instruction is used to assign a device for a low -speed timer. The OUTH T0 instruction is used to assign a device for a high speed timer.
With a timer setting (instruction format), a device is assigned for a low speed retentive timer or high speed retentive timer. The OUT T0 instruction is used to assign a device for a low speed retentive timer. The OUTH T0 instruction is used to assign a device for a high speed retentive timer.

(4) Low-speed timers

(a) Definition

Low-speed timers perform counting in 1 to 1000ms units.
The timer is valid only while its coil is ON.
The time measurement begins when the timer's coil switches ON, and the contact switches ON when a "time-out" occurs. When the timer's coil switches OFF, the current value becomes "0", and the contact switches OFF.



When X0 switches ON, the T0 coil switches ON, and the contact switches ON 1 second later. (The low-speed timer measures time in 100 ms units.)

Figure 10.21 Ladder example and timing chart of low-speed timer

(b) Measurement units

The default time measurement units setting for low speed timers is 100 ms.

The time measurement units setting can be designated in 1 ms units within a 1 ms to 1000 ms range.

This setting is designated at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

(5) High-speed timers

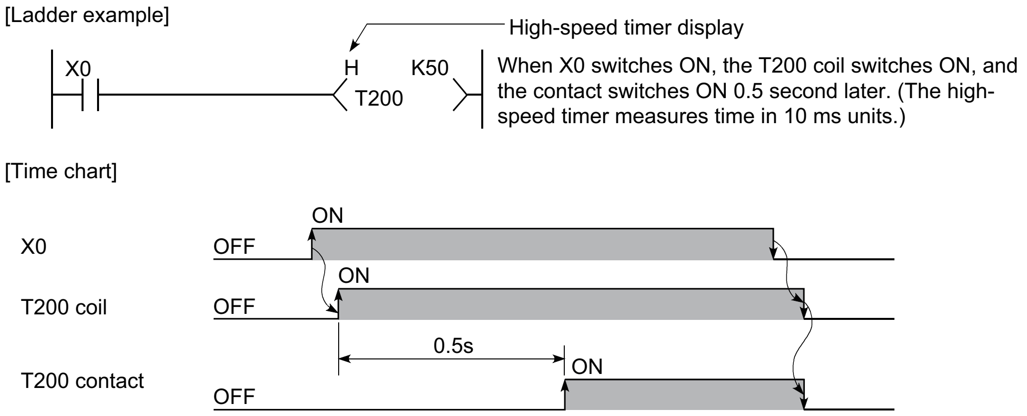
(a) Definition

High-speed timers provide higher measurement accuracy than low-speed timers.

The measurement units for each CPU are shown in Table 9.

The timer is valid only while its coil is ON, and has a symbol "H".

The time measurement begins when the timer's coil switches ON, and the contact switches ON when the time elapses. When the timer's coil switches OFF, the current value becomes "0", and the contact switches OFF.



When X0 switches ON, the T200 coil switches ON, and the contact switches ON 0.5 second later. (The high-speed timer measures time in 10 ms units.)

Figure 10.22 Ladder example and timing chart of high-speed timer

(b) Measurement units

Measurement units are set in the PLC system tab of the PLC parameter dialog box.

Table10.7 Measurement units in each CPU

CPU module	Basic model QCPU, High Performance model QCPU, Process CPU and Redundant CPU	Universal model QCPU
Measurement units	0.1 to 100ms(0.1ms units)	0.01 to 100ms(0.01ms units)

(6) Retentive timers

(a) Definition

Retentive timers measure the "coil ON" time.

The measurement begins when the timer coil switches ON, and the contact switches ON when a time-out (coil OFF) occurs.

Even when the timer coil is OFF, the current value and the contact ON/OFF status are saved. When the coil is switched ON again, the time measurement resumes from the current value which was saved.

(b) Retentive timer types

There are 2 retentive timer types: low speed retentive timer, and high speed retentive timer.

(c) Retentive timer clear

The RST ST□ instruction is used to clear (reset) the current value and switch the contact OFF.

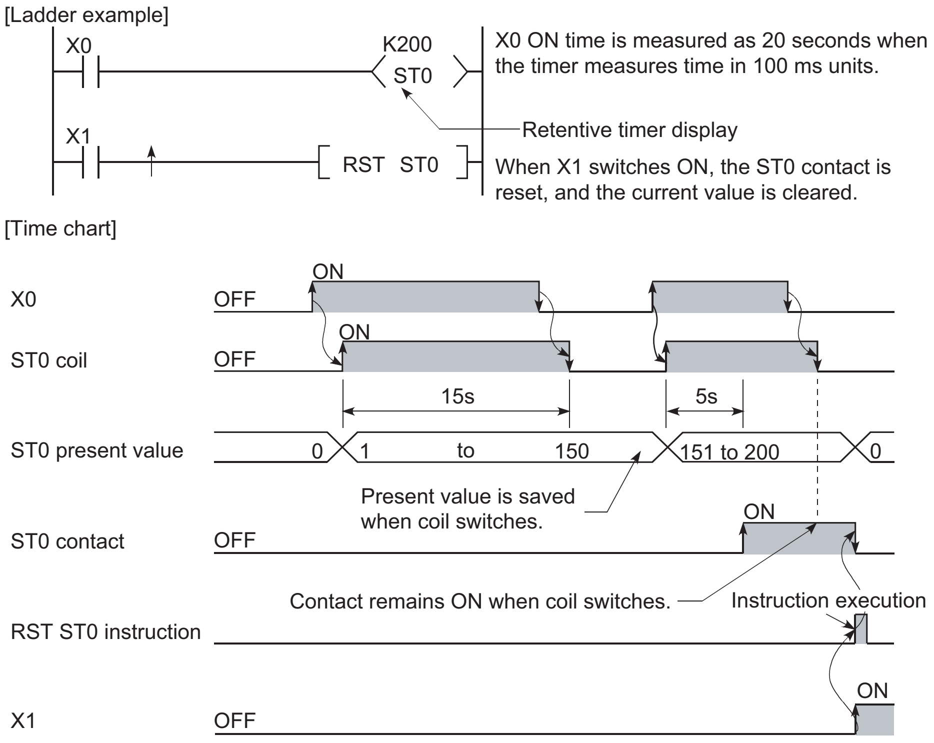

[Ladder example]
X0
K200
ST0
X0 ON time is measured as 20 seconds when the timer measures time in 100 ms units.
Retentive timer display
X1
RST ST0
When X1 switches ON, the ST0 contact is reset, and the current value is cleared.
[Time chart]
X0
ON
OFF
ST0 coil
ON
OFF
15s
5s
ST0 present value
0
1
to
150
151 to 200
0
Present value is saved when coil switches.
ST0 contact
OFF
ON
Contact remains ON when coil switches.
Instruction execution
RST ST0 instruction
X1
OFF
ON

Figure 10.23 Ladder example and timing chart of retentive timer

(d) Measurement units

The measurement units settings for retentive timers are the same as those for low speed timers and high speed timers.

- Low speed retentive timer : Same as low speed timer
- High speed retentive timer : Same as high speed timer

☒POINT

In order to use retentive timers, a retentive timer "number of points used" setting must be designated at the "Device" tab screen in the "(PLC) Parameter" dialog box.

(7) Timer Processing and accuracy

(a) Processing method

When an OUT T□ instruction is executed, the following is processed: timer coil ON/OFF, current value update and contact ON/OFF processing. Timer current value update and contact ON/OFF processing are not performed at END processing.

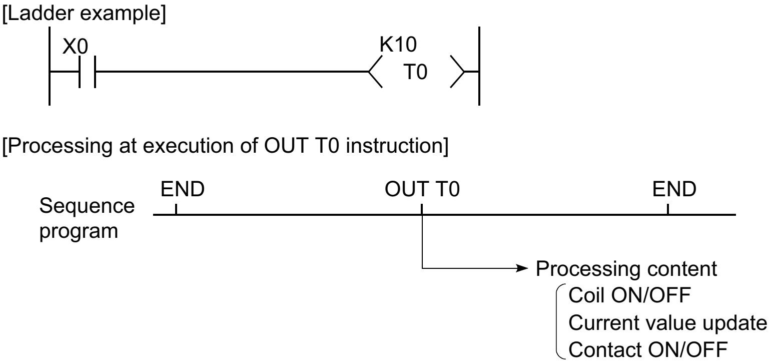
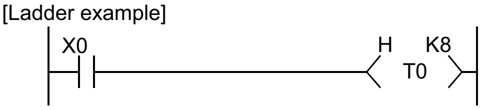


Figure 10.24 Processing at execution of OUT TO instruction

(b) Accuracy

Measured value at END instruction is added to the current value when the OUT T□ instruction is executed..



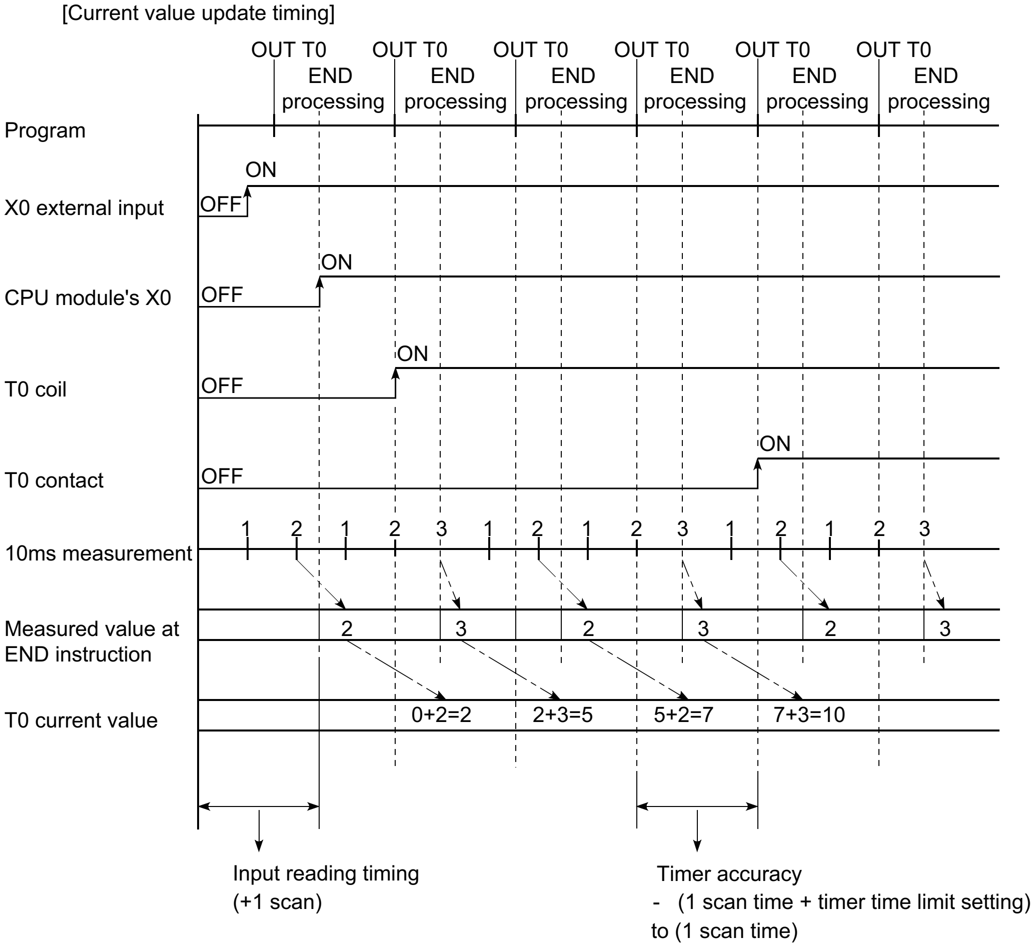


Figure 10.25 Accuracy of timer (For 10ms)

The timer response accuracy from when reading input (X), until when outputing it is + (2-scan time + timer time limit setting).

(8) Precautions for using timers

The following are a few precautions regarding timer use:

(a) Use of the same timer

A given timer cannot be designated (by OUT T□) more than once in a single scan.

This designation results in measurement, since the timer current value is updated at execution of each OUT T□ instruction.

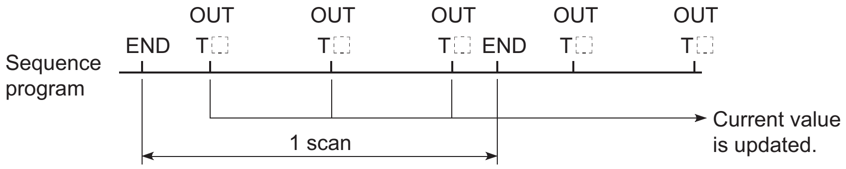


Figure 10.26 When the same timer is used

(b) When the timer is not executed in every scan

When a timer (for example. T1) coil is ON, the OUT T1 instruction cannot be skipped using a CJ instruction, and so forth.

If the OUT T□ instruction is skipped, the timer current value will not be updated. When the timer exists in the subroutine program, the subroutine call, which includes OUT T1 instruction, should be executed only once per every scan while the timer coil (ex: T1) turns on

If the subroutine call is not executed in every scan, the timer current value will not be updated.

Basic Note10.9

(c) Program that cannot use timers

Timers cannot be used in interrupt programs and fixed scan execution programs. Note10.9

(d) When set value is 0

If the timer set value is "0", the contact turnes ON when the OUT T□ instruction is executed.

(e) When set value is changed after time-out

If the set value changes to a value which is higher than the current value following a timer "time-out", the "time-out" status will remain in effect, and timer operation will not be performed.

Basic Note10.1 Redundant Note10.10 Universal Note10.1

(f) When low speed execution type program is used Note10.10

If a timer is used at a low speed execution type program, the current value will be added to the low speed scan time when the OUT T□ instruction is executed.(☞ Section 3.3.3)

Basic Note10.9

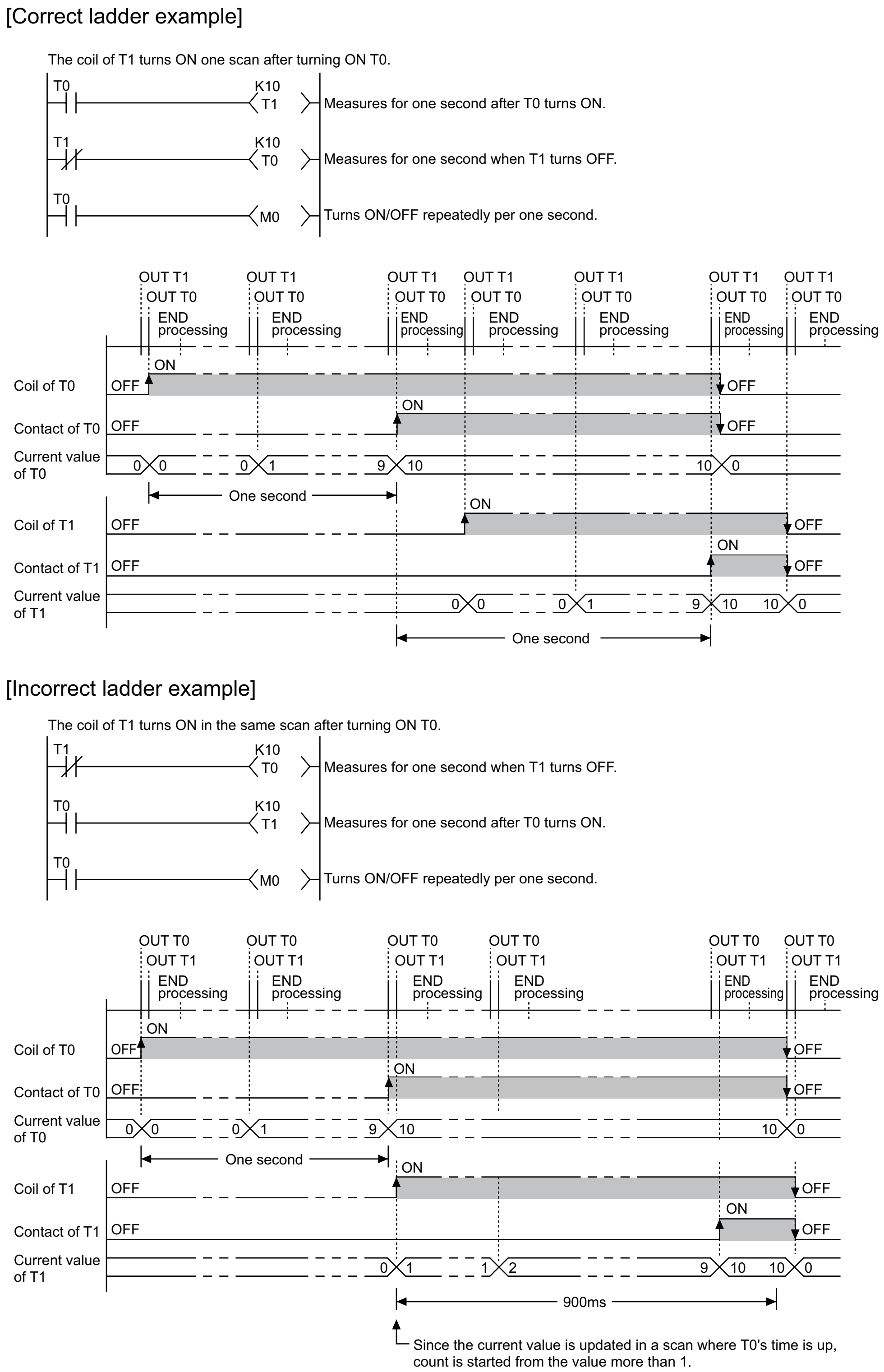
The Basic model QCPU cannot use fixed scan execution type programs.

Basic Note10.1 Redundant Note10.1 Universal Note10.1

The Basic model QCPU, Redundant CPU and Universal model QCPU cannot use low speed execution type programs.

(g) When using the multiple timers

When using the multiple timers to update the current value of a timer at the time of executing each OUT T□ instruction, pay attention to the order of ladders.
For example, when creating ON/OFF ladder using two timers, create a ladder as shown in Figure 10.27.



[Correct ladder example]

[Incorrect ladder example]

The coil of T1 turns ON one scan after turning ON T0.
T0
K10
T1
Measures for one second after T0 turns ON.
T1
K10
T0
Measures for one second when T1 turns OFF.
T0
M0
Turns ON/OFF repeatedly per one second.
OUT T1
OUT T0
END processing
ON
OFF
Coil of T0
Contact of T0
Current value of T0
One second
Coil of T1
Contact of T1
Current value of T1
One second
The coil of T1 turns ON in the same scan after turning ON T0.
T1
K10
T0
Measures for one second when T1 turns OFF.
T0
K10
T1
Measures for one second after T0 turns ON.
T0
M0
Turns ON/OFF repeatedly per one second.
OUT T0
OUT T1
END processing
Coil of T0
Contact of T0
Current value of T0
One second
Coil of T1
Contact of T1
Current value of T1
900ms
Since the current value is updated in a scan where T0's time is up, count is started from the value more than 1.

Figure 10.27 ON/OFF ladder using two timers

10.2.11 Counter (C)

(1) Definition

A counter is a device which counts the number of input condition leading edges in sequence programs.

When the count value matches the set value, the counter counts up and its contact turns ON.

The counter is of an up-counting type.

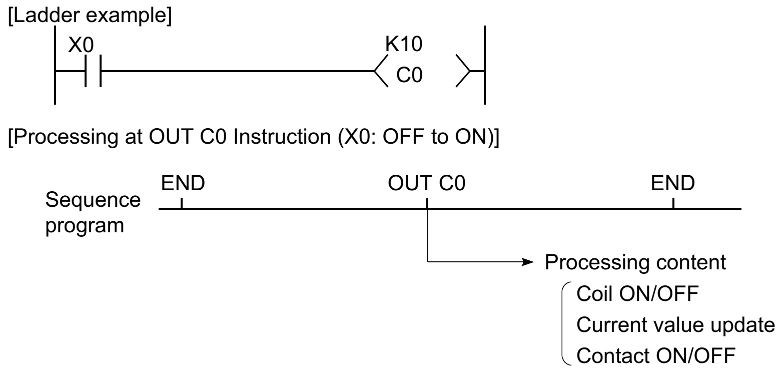
(2) Counter types

There are two counter types: counters which count the number of input condition start-ups (leading edges) in sequence programs, and counters which count the number of interrupt factor occurrences.

(3) Count processing

(a) When OUT C□ instruction is executed

When and OUT C□ instruction is executed, the following counter processing occurs: coil ON/OFF, current value update (count value + 1), and contact ON/OFF. Counter current value update and contact ON/OFF processing are not performed at END processing.


[Ladder example]

[Processing at OUT C0 Instruction (X0: OFF to ON)]

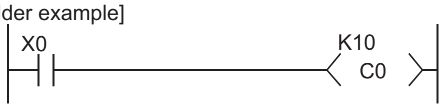
Figure 10.28 Execution and processing of OUT C□ instruction

(b) Current value update (count value + 1)

The current value update (count value + 1) is performed at the leading edge (OFF to ON) of the OUT C□ instruction.

The current value is not updated in the following OUT C□ instruction statuses: OFF, ON to ON, ON to OFF

[Ladder example]



[Current value update timing]

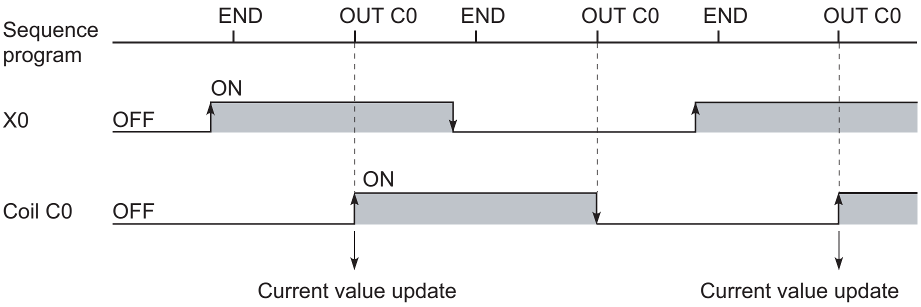


Figure 10.29 Current value update timing

☒POINT

Multiple counters can be used within a single scan to achieve the maximum counting speed.

In such cases, the direct access input (DX□) method should be used for the counter input signals.(☞ Section 3.8.2)

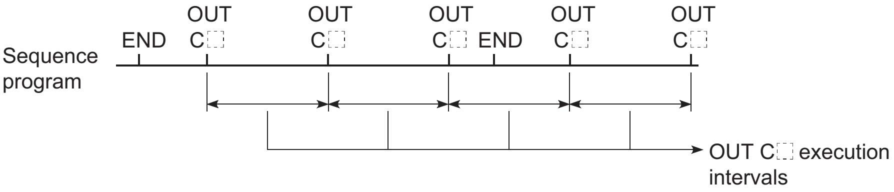
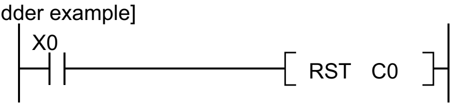


Figure 10.30 Ladder example for increasing the maximum counting speed

(c) Resetting the counter

Counter current values are not cleared even if the OUT C□ instruction switches OFF. Use the RST C□ instruction to clear the counter's current value and switch the contact OFF.
The count value is cleared and the contact is switched OFF at execution of when the RST C□ instruction.

[Ladder example]



[Counter reset timing]

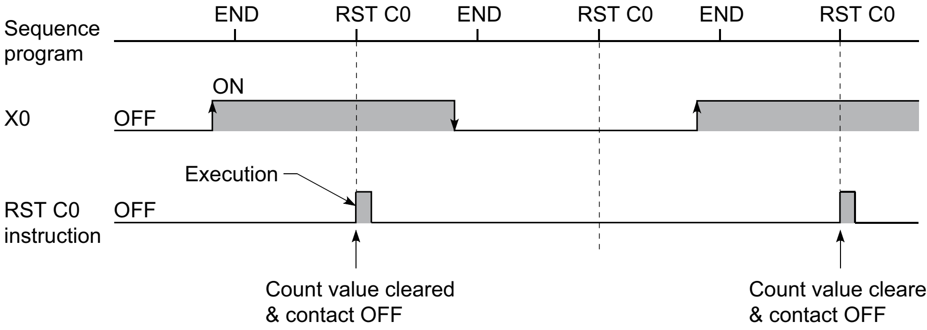


Figure 10.31 Counter resetting

1) Precautions for resetting the counter

When the RST C□ instruction is executed, the coil of C□ also turns OFF.
If the execution condition of the OUT C□ instruction is still ON after execution of the RST C□ instruction, the coil of C□ is turned ON at the execution of the OUT C□ instruction to update the current value (increment the count value by 1).

[Ladder example]

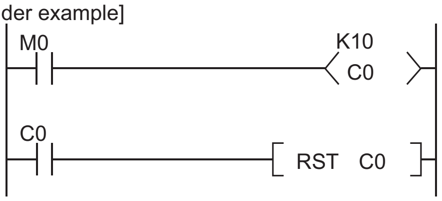


Figure 10.32 Counter resetting ladder example

In the above ladder example, when M0 turns from OFF to ON, the coil of C0 turns ON, updating the current value. When C0 reaches the preset value finally, the contact of C0 turns ON, and the execution of the RST C0 instruction clears the current value of C0. At this time, the coil of C0 also turns OFF. When M0 is still ON in the next scan, the current value is updated since the coil of C0 turns from OFF to ON at the execution of the OUT C0 instruction. (The current value turns to 1.)

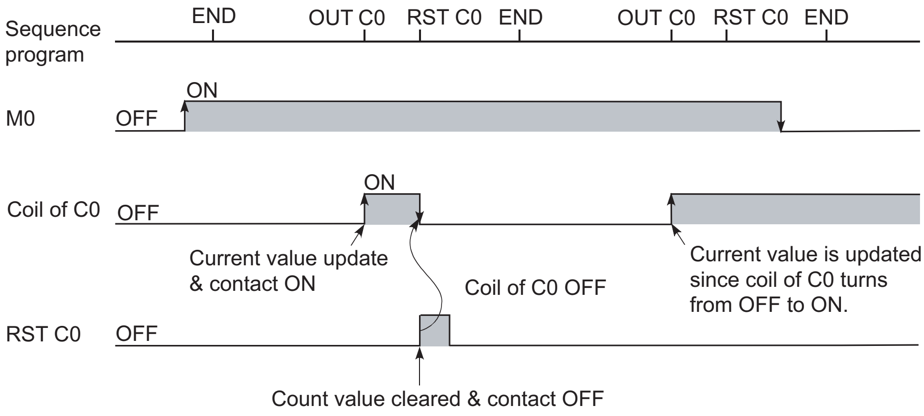

Sequence program
END
OUT C0
RST C0
END
OUT C0
RST C0
END
M0
ON
OFF
Coil of C0
ON
OFF
Current value update & contact ON
Coil of C0 OFF
Current value is updated since coil of C0 turns from OFF to ON.
RST C0
OFF
Count value cleared & contact OFF

Figure 10.33 Current value update timing

To prevent the above, it is recommended to insert the N/C contact of the OUT C0 instruction as the execution condition of the RST C0 instruction so that the coil of C0 does not turn OFF while the execution condition (M0) of the OUT C0 instruction is ON.

[Modified ladder example]

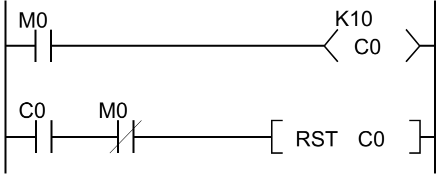


Figure 10.34 Counter resetting ladder example (recommended example)

(d) Maximum counting speed

The counter can count only when the input condition ON/OFF time 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 : The "duty" is the count input signal's ON-OFF time ratio expressed as a percentage value.

- When T1≧T2, $n = \frac{T2}{T1+T2} \times 100\%$
- When T1＜T2, $n = \frac{T1}{T1+T2} \times 100\%$

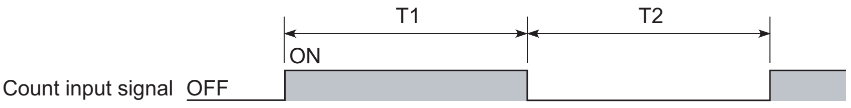


Figure 10.35 Duty ratio

Universal UD Note10.11

(4) Interrupt counters Note10.11

(a) Definition

Interrupt counters are devices which count the number of interrupt factor occurrences.

(b) Count processing

1) When interrupt occurs

The interrupt counter's current value is updated when an interruption occurs. It is not necessary to create a program which includes an interrupt counter function.

2) Counting of interrupt counter

Interrupt counter operation requires more than the simple designation of a set value.

To use the interrupt counter for control purposes, comparison instructions (=, <=, etc.) must also be used to enable comparisons with the set value, with an internal relay (M), etc., being switched ON or OFF according to the comparison result.

Figure 10.36 shows a sample program in which M0 is switched ON after 10 interrupt inputs are performed. (In this example, "C300" is the interrupt counter No. corresponding to I0.)

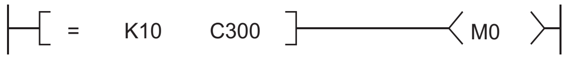


Figure 10.36 Ladder example when interrupt counter is used for control

Universal UD Note10.11

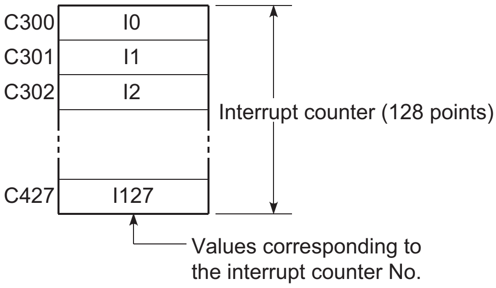
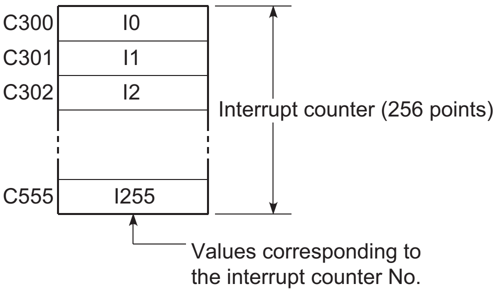
The Universal model QCPU dose not support interrupt counters.

(c) Setting the interrupt counter

In order to use interrupt counters, set Interrupt counter start No. at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

The number of points indicated in Table10.8 starting from the set counter number is used as interrupt counters.

Table10.8 Number of interrupt counter points of 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 interrupt counters. When the first interrupt counter No. is set to C300, C300 to 427 are used as interrupt counters.  C300 C301 C302 C427 I0 I1 I2 I127 Interrupt counter (128 points) Values corresponding to the interrupt counter No.	256 points, starting from the set counter number, are used as interrupt counters. When the first interrupt counter No. is set to C300, C300 to 555 are used as interrupt counters.  C300 C301 C302 C555 I0 I1 I2 I255 Interrupt counter (256 points) Values corresponding to the interrupt counter No.

In order to use an interrupt counter, an "interruption permitted" status must be established by E1 instruction at the main routine program.

(5) Precautions

(a) Execution of interrupt counter and interrupt program

One interrupt pointer is insufficient to execute interrupt counter and interrupt program operation.

Moreover, an interrupt program cannot be executed by an interrupt counter setting designated at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

(b) Processing that delays count processing

If the processing items shown below are in progress when an interruption occurs, the counting operation will be delayed until processing of these items is completed.

The count processing starts after the execution of programs is completed.

Even if the same interruption occurs again while processing of these items is in process, only one interruption will be counted.

- During execution of sequence program instructions
- During interrupt program execution
- During execution of a fixed scan execution type program Note10.12

Basic
Note10.12

(c) Maximum counting speed of interrupt counter

The maximum counting speed of the interrupt timer is determined by the longest processing time of the items shown below.

- Instruction with the longest processing time among the instructions used in the program
- Interrupt program processing time
- The processing time of a fixed scan execution type program Note10.12

Basic
Note10.12

(d) When many interrupt counters are used

The use of too many interrupt counters will increase the sequence program processing time, and may cause a "WDT ERROR".

If this occurs, reduce the number of interrupt counters or the counting speed for the input pulse signal.

(e) Resetting the interrupt counter

The interrupt counter's count value can be reset by using the RST C□ instruction in the sequence program prior to the FEND instruction.

(f) Reading the count value

The interrupt counter's count value can be read out by using the sequence program MOV instruction.

Basic
Note10.12

The Basic model QCPU cannot execute fixed scan execution type programs.

10.2.12 Data register (D)

(1) Definition

Data registers are memory devices which store numeric data (-32768 to 32767, or 0000H to FFFFH).

(2) Bit configuration of data register

(a) Bit configuration and read and write units

Data registers, which consist of 16 bits per point, read and write data in 16-bit units.

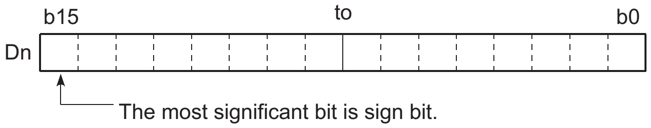


Figure 10.37 Bit configuration of data register

☒POINT

Data register data are handled as signed data.
For HEX (hexadecimal), 0000H to FFFFH can be stored. However, since the most significant bit is a sign bit, the range of a value that can be specified is -32768 to 32767.

(b) When data register is used for 32-bit instruction

If the data registers are used for 32-bit instructions, the data will be stored in registers Dn and Dn + 1. The lower 16 bits of data are stored at the data register No. (Dn) designated in the sequence program, and the higher 16 bits of data are stored in the designated register No. + 1 (Dn + 1). For example, if register D12 is designated in the DMOV instruction, the lower 16 bits are stored in D12, and the upper 16 bits are stored in D13.

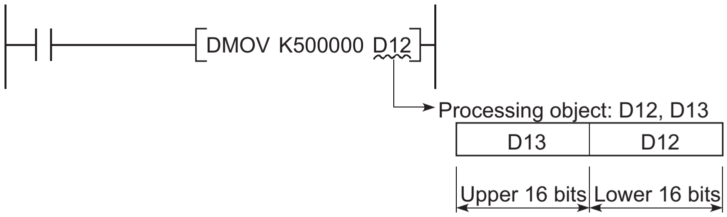


Figure 10.38 Data transfer by 32-bit instruction and storage destination

Two data registers can store a range of numeric data from -2147483648 to 2147483647 or from 0H to FFFFFFFFH. (The most significant bit in a 32-bit configuration is a sign bit.)

(3) Holding of stored data

The data stored in the data register is held until the other data is stored.
The data stored in the data register is initialized when the PLC is powered OFF or the CPU module is reset.

10.2.13 Link register (W)

(1) Definition

A link register is the CPU module memory used to refresh the CPU module with data from the link registers (LW) of intelligent function modules including MELSECNET/H network module.

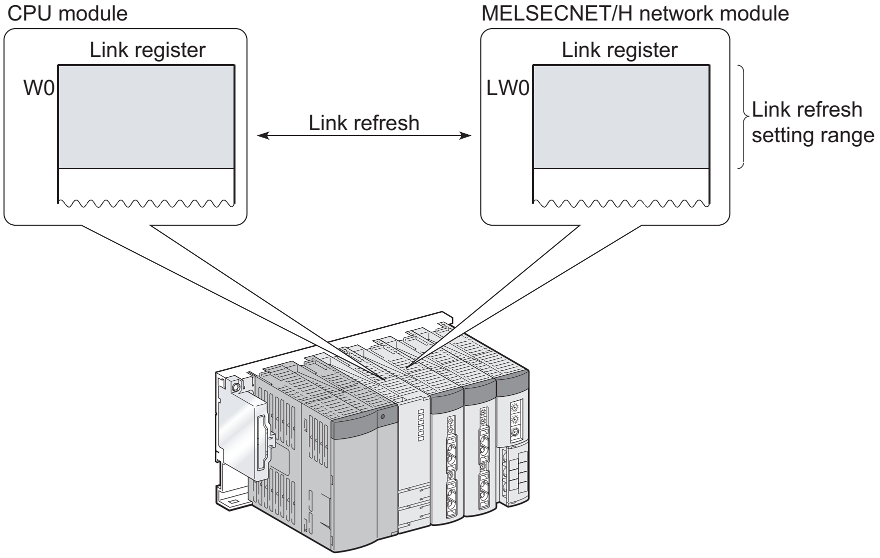


Figure 10.39 Link refresh

Link registers are used to store numeric data (-32768 to 32767, or 0000H to FFFFH).

(2) Bit configuration of link register

(a) Bit configuration and read and write units

Link registers, which consist of 16 bits per point, read and write data in 16 bit units.

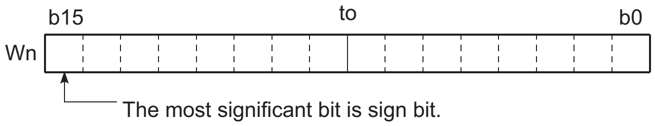


Figure 10.40 Bit configuration of link register

☒POINT

1. Link register data are handled as signed data.
   For HEX (hexadecimal), 0000H to FFFFH can be stored. However, since the most significant bit is a sign bit, the range of a value that can be specified is -32768 to 32767.
2. When used outside the MELSECNET/H network system's range, link registers can serve as data registers.

(b) **When link register is used for 32-bit instruction**

If the link registers are used for 32-bit instructions, the data is stored in registers Wn and Wn + 1. The lower 16 bits of data are stored in the link register No. (Wn) designated in the sequence program, and the higher 16 bits of data are stored in the designated register No. + 1 (Wn + 1).

For example, if link register W12 is designated in the DMOV instruction, the lower 16 bits are stored in W12, and the upper 16 bits are stored in W13.

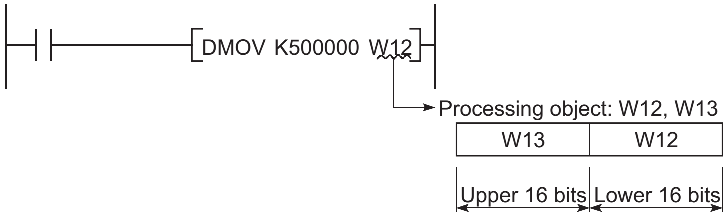


Figure 10.41 Data transfer by 32-bit instruction and storage destination

Two link registers can store a range of numeric data from -2147483648 to 2147483647 or from 0H to FFFFFFFFH. (The most significant bit in a 32-bit configuration is a sign bit.)

(3) Holding of stored data

Data stored by the link register is maintained until another data is save.

The data stored in the link register is initialized when the PLC is powered OFF or the CPU module is reset.

☒POINT

- Although the number of device points for link registers in the CC-Link IE controller network module is 131072 points, the default number of points for link registers in the CPU module is 2048 points (Basic model QCPU) or 8192 points (High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU).
- Although the number of device points for link registers in the MELSECNET/H network module is 16384 points, the default number of points for link registers in the CPU module is 2048 points (Basic model QCPU) or 8192 points (High Performance model QCPU, Process CPU, Redundant CPU, and Universal model QCPU).

To use link registers exceeding the point described above, change the device point setting for link registers on the Device tab of PLC parameter in GX Developer or use file registers.

(4) Using link registers in a network system

In order to use link registers in the network system, network parameter settings must be made.

Link registers not set in the network parameter settings can be used as data registers.

Remark

Refer to the following manual for the network parameters.

☞ CC-Link IE Controller Network Reference Manual

☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

10.2.14 Link special register (SW)

(1) Definition

Link special registers are used to store data on the communication status and errors of an intelligent function module such as CC-Link IE controller network module and MELSECNET/H network module.

Because the data link information is stored as numeric data, the link special registers serve as a tool for identifying the locations and causes of faults.

(2) Number of link special register points

The number of link special register points is as described in Table10.9.

Table10.9 Number of link special register points of each CPU module

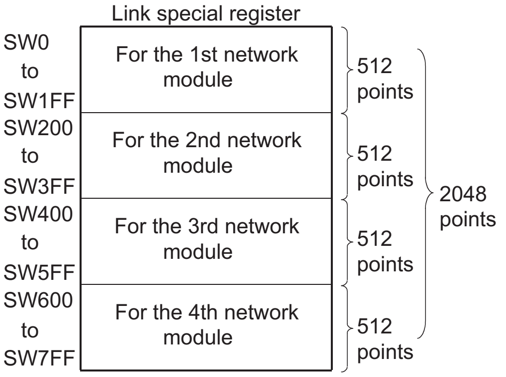
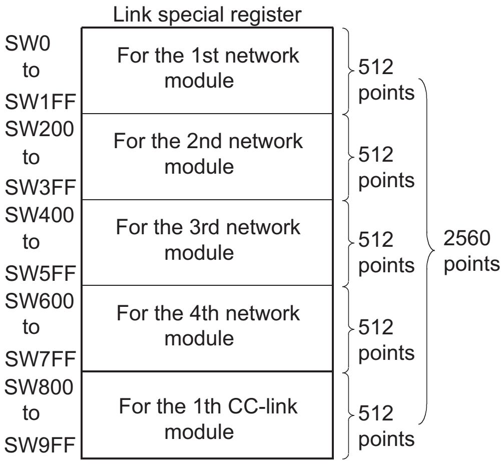
CPU module	Number of link special relay points
Basic model QCPU	1024 points (SW0 to 3FF). 512 points assigned to each intelligent function module such as CC-Link IE controller network module and MELSECNET/H network module.
High Performance model QCPU Process CPU Redundant CPU	2048 points (SW0 to 7FF). 512 points assigned to each intelligent function module such as CC-Link IE controller network module and MELSECNET/H network module. The link special registers can be assigned as shown below.  Link special register SW0 to SW1FF For the 1st network module 512 points SW200 to SW3FF For the 2nd network module 512 points SW400 to SW5FF For the 3rd network module 512 points SW600 to SW7FF For the 4th network module 512 points 2048 points

Table10.9 Number of link special register points of each CPU module (continued)

CPU module	Number of link special relay points
Universal model QCPU	2048 points (SB0 to 7FF) However, this can be changed in the device setting of the PLC parameter dialog box. (☞Section 10.1(2)) For the intelligent function module, such as the CC-Link IE controller network module and MELSECNET/H network module, having link special relays, 512 points are assigned. Assigning the link special registers as shown below enables refreshing of the CC-Link special register (SW) for the link special register (SW) of the CPU module.  Link special register SW0 to SW1FF For the 1st network module 512 points SW200 to SW3FF For the 2nd network module 512 points SW400 to SW5FF For the 3rd network module 512 points SW600 to SW7FF For the 4th network module 512 points SW800 to SW9FF For the 1th CC-link module 512 points 2560 points

Remark

For details of the link special registers, refer to the manual of the intelligent function module having link special registers.

10.3 Internal System Devices

Internal system devices are used for system operations.
The allocations and sizes of internal system devices are fixed, and cannot be changed by the user.

10.3.1 Function devices (FX, FY, FD)

(1) Definition

Function devices are used in subroutine programs with arguments.
The function devices write/read data between a subroutine call source with argument and a subroutine program with argument.

Example

If FX0, FY1 and FD2 are used at the subroutine program, and if X0, M0, and D0 are designated by the subroutine program CALL instruction, the X0 ON/OFF data is transferred to FX0, the FY1 ON/OFF data is transferred to M0, and the D0 data is transferred to FD2.

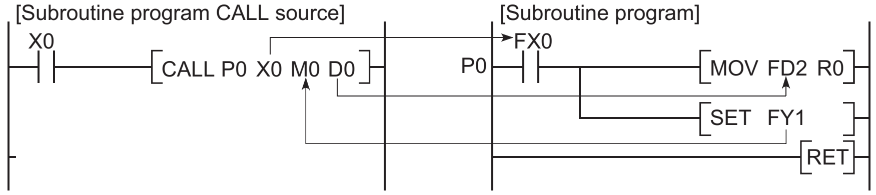


Figure 10.42 Application example of function devices

(2) Applications of function devices

Because the function devices used for each subroutine program CALL source can be set, the same subroutine program can be used without regard to other subroutine CALL sources.

(3) Types of function devices

There are 3 function device types: function input devices (FX), function output devices (FY), and function register devices (FD).

(a) Function input devices (FX)

- These devices are used to designate inputs of ON/OFF data to a subroutine program.
- In the subroutine program, these devices are used for reading and processing bit data designated by subroutine with argument CALL instruction.
- All the CPU module bit data designation devices can be used.

(b) Function output devices (FY)

- These devices are used to designate outputs of subroutine program operation results (ON/OFF data) to the subroutine program CALL source.
- The operation results are stored at the device designated by using subroutine programs with arguments.
- All bit data designation devices except CPU module inputs (X, DX) can be used.

(c) **Function registers (FD)**

- Function registers are used to perform write/read of data between the subroutine call source and the subroutine program.
- The function register I/O condition is automatically determined by the High Performance model QCPU. If the subroutine program data is the source data, the data is designated as subroutine input data.
  If the subroutine program data is the destination data, the data is designated as subroutine output data.
- 1 function register occupies a maximum of 4 words.
  The number of words used depends on an instruction in a subroutine program.

A one-word instruction requires 1 word.

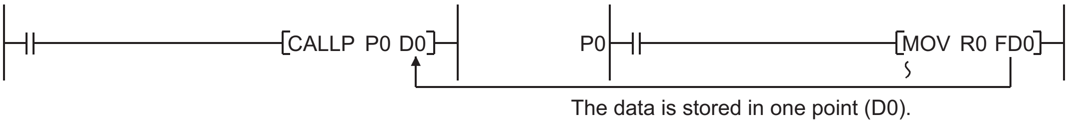


Figure 10.43 When 1 word is occupied by 1 function register

A two-words instruction requires 2 words.

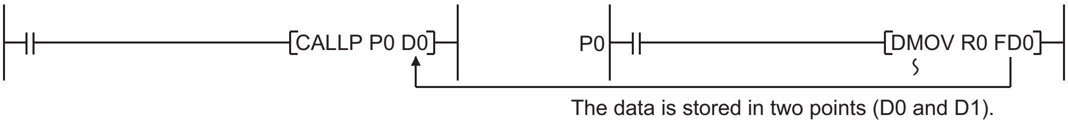


Figure 10.44 When 2 words are occupied by 1 function register

The destination of 32-bit multiplication/division operation requires 4 words.

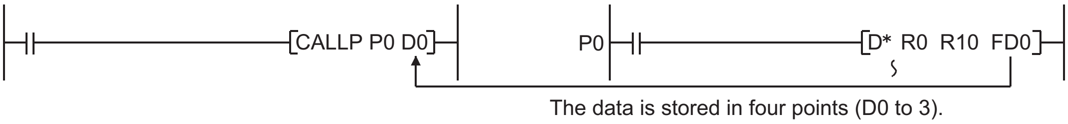


Figure 10.45 When 4 words are occupied by 1 function register

CPU module word data devices can be used.

☒POINT

Valid devices cannot be used in a subroutine program that contains arguments. If devices assigned for function registers are used, values of the function registers will not correctly be returned to a calling program.

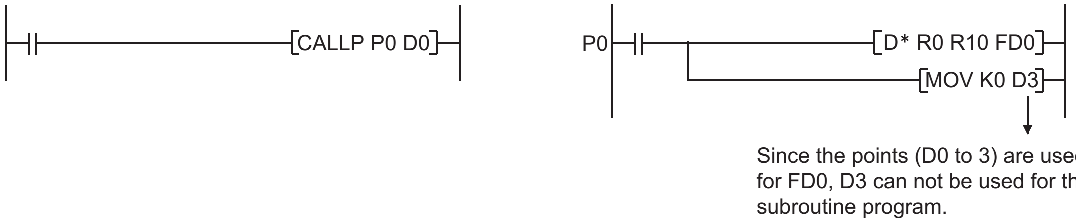

CALLP P0 D0
P0
D* R0 R10 FD0
MOV K0 D3
Since the points (D0 to 3) are used for FD0, D3 can not be used for the subroutine program.

Figure 10.46 Ladder example where the devices are unusable in the sub-routine program that contains arguments.

Remark

Refer to the following manual for how to use the function devices.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

10.3.2 Special relay (SM)

(1) Definition

A special relay is used to store CPU module status data.

(2) Special relay classifications

Special relays are classified according to their applications, as shown in Table10.10.

Table10.10 Special relay classification list

Classification	Special relay	CPU module				
		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU
Diagnostics information	SM0 to 99[*1] SM0 to 199[*2]	○	○	○	○	○
Serial communication function	SM100 to 129[*1]	○	×	×	×	×
System information	SM200 to 399	○	○	○	○	○
System clock/system counter	SM400 to 499	○	○	○	○	○
Scan information	SM500 to 599	○	○	○	○	○
Memory card	SM600 to 699	○	○	○	○	○
Instruction related	SM700 to 799	○	○	○	○	○
debugging	SM800 to 899	×	○	○	○	○
Latch area	SM900 to 999	×	○	○	○	○
A → Q/QnA conversion compatibility	SM1000 to 1299[*3]	×	○	○	×	×
Process control instruction	SM1500 to 1509	×	×	○	○	×
Redundant system (host system CPU information)	SM1510 to 1599	×	×	×	○	×
Redundant system (other system CPU information)	SM1600 to 1699	×	×	×	○	×
Redundant system (tracking information)	SM1700 to 1779	×	×	×	○	×
Redundant power supply module information	SM1780 to 1799	×	○	○	○	○

○ : Usable special relay is included
× : Usable special relay is not included

* 1 : In the Basic model QCPU, SM0 to 99 will be the area for error diagostics and SM100 to 129 will be the area for a serial communication function.
* 2 : In the High Performance model QCPU, Process CPU or Redundant CPU, SM0 to 199 will be the area for error diagnostics.
* 3 : Usable only when "Use the special relays/special registers SM1000/SD1000 and later" is selected in the PLC system setting of the PLC parameter dialog box.

Remark

For details on special relays which can be used by the CPU module, refer to Appendix 1.

10.3.3 Special register (SD)

(1) Definition

A special register is used to store CPU module status data (diagnosis and system information).

(2) Special register classifications

Special registers are classified according to their applications, as shown in Table10.11.

Table10.11 Special register classification list

Classification	Special register	CPU module				
		Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU
Diagnostics information	SD0 to 99[*1]	○	○	○	○	○
	SD0 to 199[*2]					
Serial communication function	SD100 to 129[*1]	○	×	×	×	×
Fuse blown module	SD130 to 149[*1]	○	×	×	×	×
Check of I/O modules	SD150 to 199[*1]	○	×	×	×	×
System information	SD200 to 399	○	○	○	○	○
System clock/system counter	SD400 to 499	○	○	○	○	○
Scan information	SD500 to 599	○	○	○	○	○
Memory card	SD600 to 699	○	○	○	○	○
Instruction related	SD700 to 799	○	○	○	○	○
debugging	SD800 to 899	×	○	×	×	○
Latch area	SD900 to 929	×	○	×	×	○
Redundant function (power failure backup information)	SD952	×	×	×	○	×
A → Q/QnA conversion compatibility	SD1000 to 1299	×	○	○	×	×
Fuse blown module	SD1300 to 1399[*2]	×	○	○	○	○
Check of I/O modules	SD1400 to 1499[*2]	×	○	○	○	○
Process control instruction	SD1500 to 1509	×	×	○	○	×
Redundant system (host system CPU information)	SD1510 to 1599	×	×	×	○	×
Redundant system (other system CPU information)	SD1600 to 1699	×	×	×	○	×
Redundant system (tracking information)	SD1700 to 1779	×	×	×	○	×
Redundant power supply module information	SD1780 to 1799	×	○	○	○	○

○ : Usable special relay is included
× : Usable special relay is not included

* 1 : In the Basic model QCPU, SD0 to 99 will be the area for error diagnostics.
The serial communication function, fuse blown module and check of I/O modules data are stored into SD100 to 199.

* 2 : In the High Performance model QCPU, Process CPU or Redundant CPU, SD0 to 199 will the area for error diagnostics.
The fuse blown module and check of I/O modules data are stored into SD1300 to 1499.

* 3 : Usable only when "Use the special relays/special registers SM1000/SD1000 and later" is selected in the PLC system setting of the PLC parameter dialog box.

Remark

For details on special relays refer to Appendix 2.

10.4 Link direct device

(1) Definition

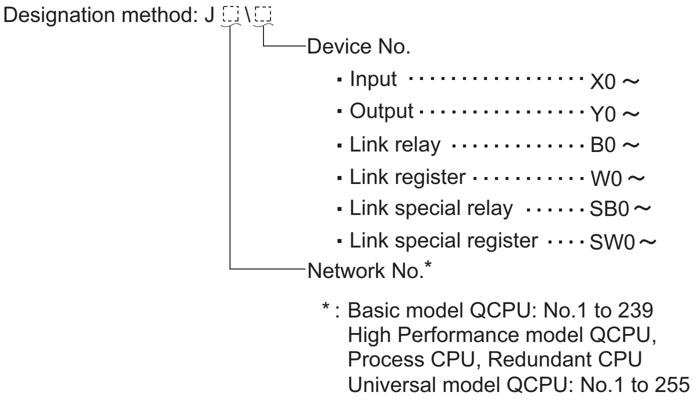
The link direct device is a device to directly access the link device in the CC-Link IE controller network module or MELSECNET/H network module.

The CPU module can directly write/read data to/from the link device in the CC-Link IE controller network module or MELSECNET/H network module using the sequence program regardless of the link refresh.

(2) Designation method and application example of link direct device

(a) Designation method

Link direct devices are designated by network No. and device No.



Designation method: J □ \ □

Device No.

▪ Input	X0 ~
▪ Output	Y0 ~
▪ Link relay	B0 ~
▪ Link register	W0 ~
▪ Link special relay	SB0 ~
▪ Link special register	SW0 ~

Network No.*

* : Basic model QCPU: No.1 to 239
High Performance model QCPU,
Process CPU, Redundant CPU
Universal model QCPU: No.1 to 255

Figure 10.47 Link direct device designation method

(b) Application example

For link register 10 (W10) of network No.2, the designation would be "J2\W10"

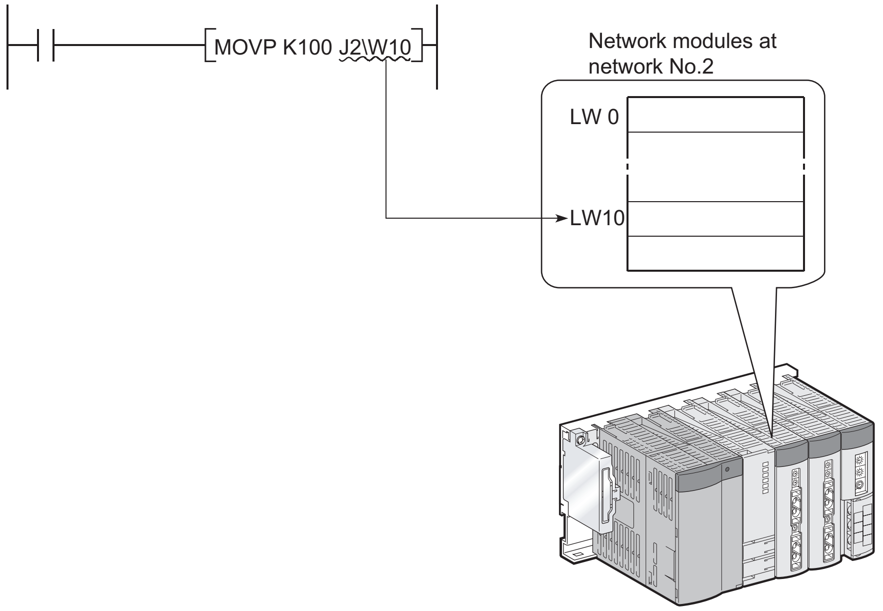


Figure 10.48 Link direct device application example

- For a bit device (X, Y, B, SB), digit designation is necessary.
  Designation example : J1\K1X0, J10\K4B0

(3) Designation range

The link direct device can specify the link device of the network module.
The link device range that is not set by the network refresh parameter can be specified.

(a) Writing

- Writing is executed within that part of the link device range set as the send range in the common parameters of the network parameters that is outside the range specified as the "refresh range" in the network refresh parameters.

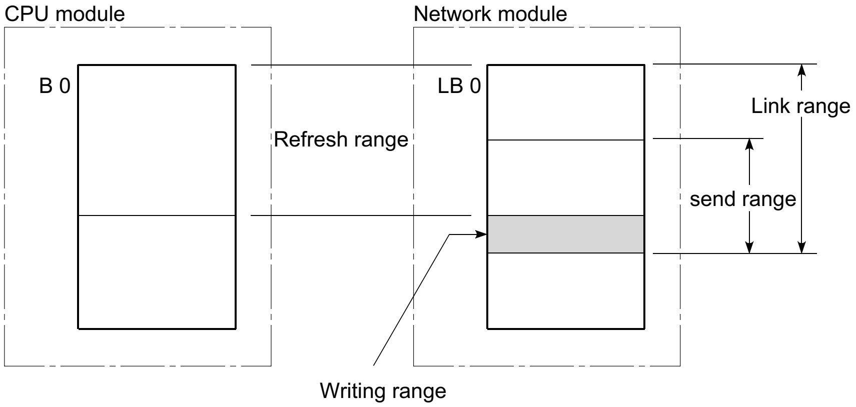


Figure 10.49 Link direct device write range

- Although writing is also allowed in the "refresh range" portion of the link device range (specified by refresh parameters), the link module's link device data will be rewritten when a refresh operation occurs.
  When writing by link direct device, the same data should also be written to the CPU module related devices designated by refresh parameter.

[Refresh parameter settings]

- Network No. : 1
- CPU module (W0 to 3F)↔Network module (LW0 to 3F)

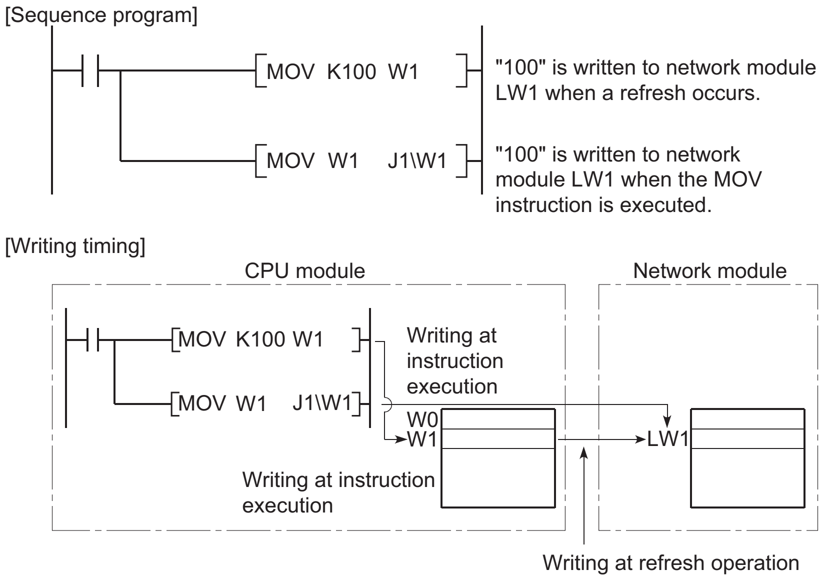

[Sequence program]
MOV K100 W1
"100" is written to network module LW1 when a refresh occurs.
MOV W1 J1\W1
"100" is written to network module LW1 when the MOV instruction is executed.
[Writing timing]
CPU module
Network module
MOV K100 W1
MOV W1 J1\W1
Writing at instruction execution
W0
W1
LW1
Writing at instruction execution
Writing at refresh operation

Figure 10.50 Write to link device set to refresh range

- When data is written to another station's writing range using a link direct device, the data which is received from that station will replace the written data.

(b) Reading

Reading by link direct device is allowed in the entire link device range of network modules.

☒POINT

Basic
Note10.13

The CPU module can write/read data to/from only one network module on the same network using a link direct device.

If two or more network modules are mounted on the same network, the network module whose slot number is younger is targeted for writing/reading data to/from the CPU module using a link direct device.Note10.13

In the case where the station No.1 and No.2 network modules are mounted on the network No.1 as shown in Figure 10.51, the station No.2 will be the target network module.

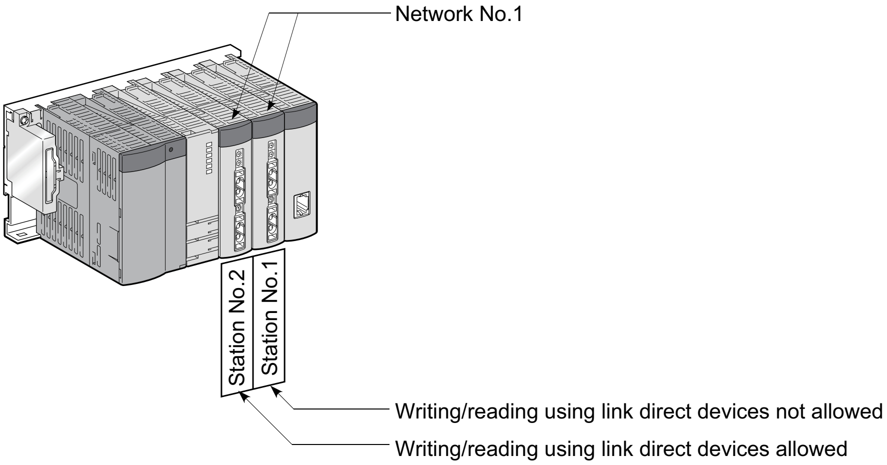

Network No.1
Station No.2
Station No.1
Writing/reading using link direct devices not allowed
Writing/reading using link direct devices allowed

Figure 10.51 When two or more network modules are mounted on the same network

Basic
Note10.13

When the Basic model QCPU is used, mounting two or more network modules on the same network number is not allowed. In this case, it is not necessary to be conscious of the description in the POINT.

(4) Differences between "link direct devices" and "link refresh"

The differences between "link direct devices" and "link refresh" are shown in Table10.12.

Table10.12 Differences Between "Link Direct Devices" and "Link Refresh"

Item		Link Direct Device	Link Refresh
Program notation method	Link relay	J□\K4B0 or later	B0 or later
	Link register	J□\W0 or later	W0 or later
	Link special relay	J□\K4SB0 or later	SB0 or later
	Link special register	J□\SW0 or later	SW0 or later
Number of steps		2 steps	1 step
Network module access range		J□\□0 to 3FFF	Refresh parameter designated range
Access data guarantee range		Word units (16 bits)	

Remark

1. For details on the CC-Link IE controller network system, refer to the following manuals:
   ☞ CC-Link IE Controller Network Reference Manual
2. For details on the MELSECNET/H network system, refer to the following manuals:
   ☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
   ☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)
3. For details on network parameters, common parameters, and network refresh parameters, refer to the following manuals:
   - Detailed information :
     ☞ CC-Link IE Controller Network Reference Manual
     ☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
     ☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)
   - Setting procedures :
     ☞ GX Developer Operating Manual

10.5 Module Access Devices

10.5.1 Intelligent function module device (U□\G□)

(1) Definition

The intelligent function module devices allow the CPU module to directly access the buffer memories of intelligent function modules/special function modules which are mounted on at the main base unit and extension base units.

(2) Designation method and application example of intelligent function module device

(a) Designation method

Intelligent function module devices are designated by the intelligent function module/special function module I/O No., and the buffer memory address.

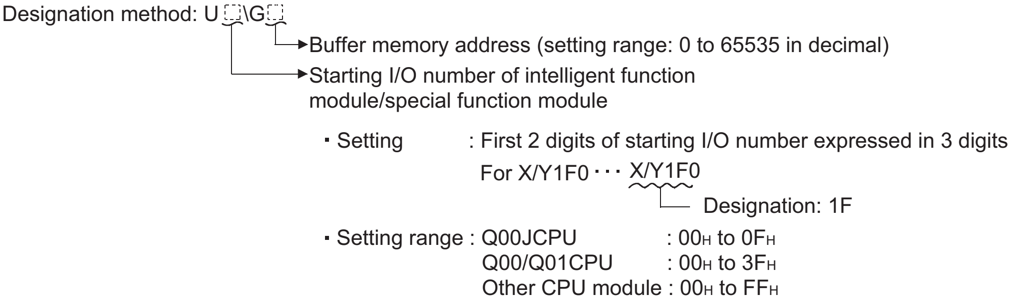

Designation method: U□\G□
Buffer memory address (setting range: 0 to 65535 in decimal)
Starting I/O number of intelligent function module/special function module
・Setting : First 2 digits of starting I/O number expressed in 3 digits
For X/Y1F0 ・・・ X/Y1F0
Designation: 1F
・Setting range : Q00JCPU : 00H to 0FH
Q00/Q01CPU : 00H to 3FH
Other CPU module : 00H to FFH

Figure 10.52 Intelligent function module device designation method

(b) Application example

Specify the intelligent function module device as shown in Figure 10.53 when storing into D0 to 3 the digital output values of CH. 1 to 4 on the Q64AD analog/digital converter module (X/Y020 to 02F) mounted at the I/O No. 020.

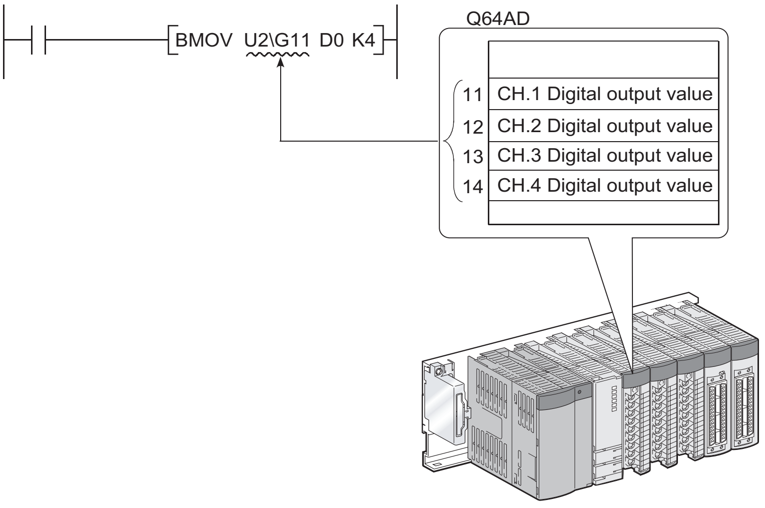

BMOV U2\G11 D0 K4
Q64AD
11 CH.1 Digital output value
12 CH.2 Digital output value
13 CH.3 Digital output value
14 CH.4 Digital output value

Figure 10.53 Intelligent function module device application example

☒POINT

Using the intelligent function module device allows device comments to be attached to the buffer memory.

☞ GX Developer Operating Manual

(3) Processing speed

The processing speed for intelligent function module devices is;

- The processing speed of write/read by the intelligent function module is slightly higher than that of write/read by the FROM/TO instruction.(For example, "MOV U2\G11 D0")

Basic Process Redundant
Note10.14 Note10.14 Note10.14
Universal
Note10.14

- To conduct reading the buffer memory of the intelligent function module/special function module Note10.14 and another process in a single instruction, add the processing speed of FROM/TO instruction and processing speed of instruction to setup the reference value.
  (For example, case of "+ U2\G11 D0 D10")

☒POINT

When using the intelligent function module device in the sequence program twice or more, and writing/reading data of the buffer memory, writing/reading data in one place of the program with the FROM/TO instruction enables to speed up the processing speed.

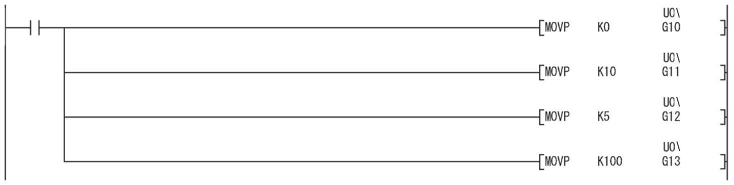


Figure 10.54 When writing to multiple devices in multiple steps

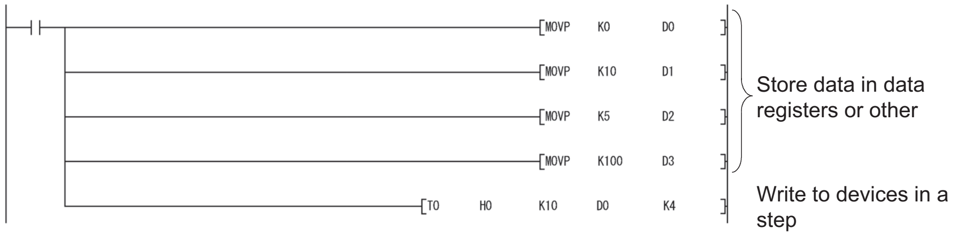

MOVP K0 D0
MOVP K10 D1
MOVP K5 D2
MOVP K100 D3
TO H0 K10 D0 K4
Store data in data registers or other
Write to devices in a step

Figure 10.55 When writing to multiple devices in a step

Remark

Basic Process Redundant
Note10.14 Note10.14 Note10.14
Universal
Note10.14

1) For details on buffer memory addresses and applications, refer to the intelligent function module/special function module manual. Note10.14
2) Refer to the following manual for the FROM/TO instruction.
☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Process Redundant
Note10.14 Note10.14 Note10.14
Universal
Note10.14

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the special function modules.

Redundant

Note10.15

10.5.2 Common device for multiple CPU (U3En\G□)

(1) Definition of common device for multiple CPU

The common device for multiple CPU allows the CPU module to access the CPU shared memory in each CPU module of a multiple CPU system.

The common device for multiple CPU allows the CPU module to access the CPU shared memory in each CPU module of a multiple CPU system.

(2) Features of common device for multiple CPU

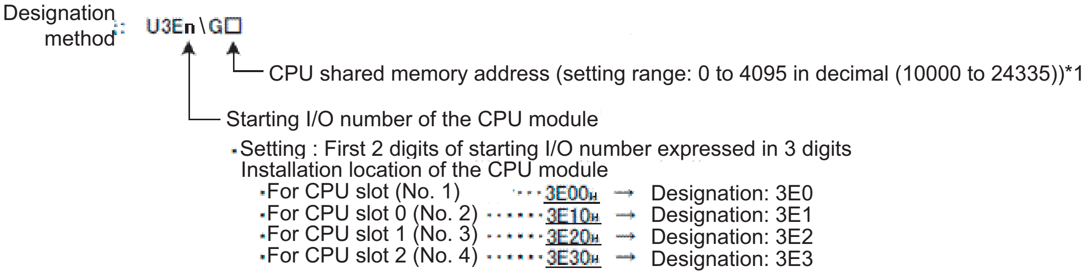
The features of the common device for multiple CPU are described below.

- The device allows transferring at a higher speed than writing (S.TO/TO instruction)/reading (FROM instruction) of the CPU shared memory, reducing steps used in the program.
- Using the common device for multiple CPU, bit operation can be performed.
- Setting device comments for the common device for multiple CPU can improve program readability.
- Since this can directly specify the information on the CPU shared memory as an argument of the instruction, the devices for interlock are not required.

(3) Specifying method and application example of common device for multiple CPU

(a) Specifying method

The common device for multiple CPU is specified using the I/O number of a CPU module and the CPU shared memory address.



Designation method : U3En\G□

CPU shared memory address (setting range: 0 to 4095 in decimal (10000 to 24335))*1

Starting I/O number of the CPU module

- Setting : First 2 digits of starting I/O number expressed in 3 digits

Installation location of the CPU module

- For CPU slot (No. 1) ···3E00H → Designation: 3E0
- For CPU slot 0 (No. 2) ······3E10H → Designation: 3E1
- For CPU slot 1 (No. 3) ······3E20H → Designation: 3E2
- For CPU slot 2 (No. 4) ······3E30H → Designation: 3E3

*1: ( ) is only for the universal model QCPU.

Figure 10.56 Specifying method and application example of common device for multiple CPU

Remark

For details of the common device for multiple CPU, refer to the following manual.

☞ QCPU User's Manual (Multiple CPU System)

Redundant

Note10.15

The common device for multiple CPU is not available for the Redundant CPU.

10.6 Index Register(Z) / Standard Device Resister(Z)

10.6.1 Index Register (Z)

Universal UD Note10.16

(1) Definition

Index registers are used in the sequence program for indirect setting (index qualification) designations.

An index register point is used for index modification.Note10.16

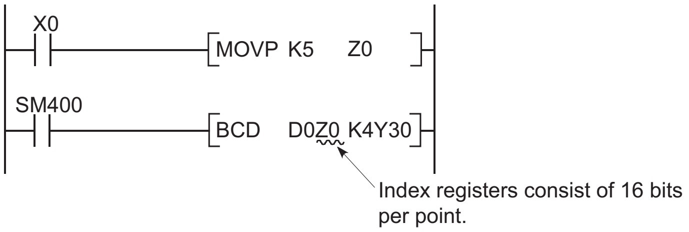


Figure 10.57 Index register

The number of index register points is as described below.
Basic model QCPU: Z0 to 9 (10 points)
High Performance model QCPU,
Process CPU, Redundant CPU: Z0 to 15 (16 points)
Universal model QCPU : Z0 to 19 (20 points)

(2) Bit configuration of index register

(a) Bit configuration and write/read units

The index register, configured in 16 bits per point, writes/reads data in units of 16 bits.

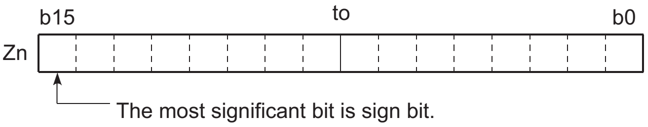


Figure 10.58 Bit configuration of index register

☒POINT

Index register data are handled as signed data.
F or HEX (hexadecimal), 0000H to FFFFH can be stored. However, since the most significant bit is a sign bit, the range of a value that can be specified is -32768 to 32767.

Universal UD Note10.16

For the Universal model QCPU, the serial access format file register (ZR) can perform 32-bit indexing with using 2 points of index register.

(b) When index register is used for 32-bit instruction

If the index registers are used for 32-bit instructions, the data is stored in registers Zn and Zn +1.

The lower 16 bits of data are stored in the index register No. (Zn) designated in the sequence program, and the upper 16 bits of data are stored in the designated index register No. + 1.

For example, if register Z2 is designated in the DMOV instruction, the lower 16 bits are stored in Z2, and the upper 16 bits are stored at Z3.

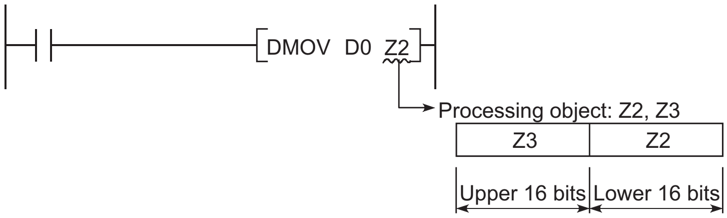


Figure 10.59 Data transfer by 32-bit instruction and storage destination

Remark

Refer to the following manual for indexing using the index register.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

10.6.2 Standard device register (Z)

(1) Definition

The Universal model QCPU can perform operation at the faster speed if the index register is used between the register operations.

The index register used in this case is called General-purpose operation resistor.

(2) Device number

The standard device register is the same device as the index register.

When performing indexing, pay attention not to use the same device number as the standard device register.

☒POINT

The operation speed can be higher only for the Universal model QCPU.

This effect cannot be expected in the Basic model QCPU, High Performance model QCPU, Process CPU or Redundant CPU.

Remark

For the processing time when using the standard device register, refer to the following manual.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Note10.17

Redundant Note10.17

Universal Note10.17

10.6.3 Switching between scan execution and low speed execution types

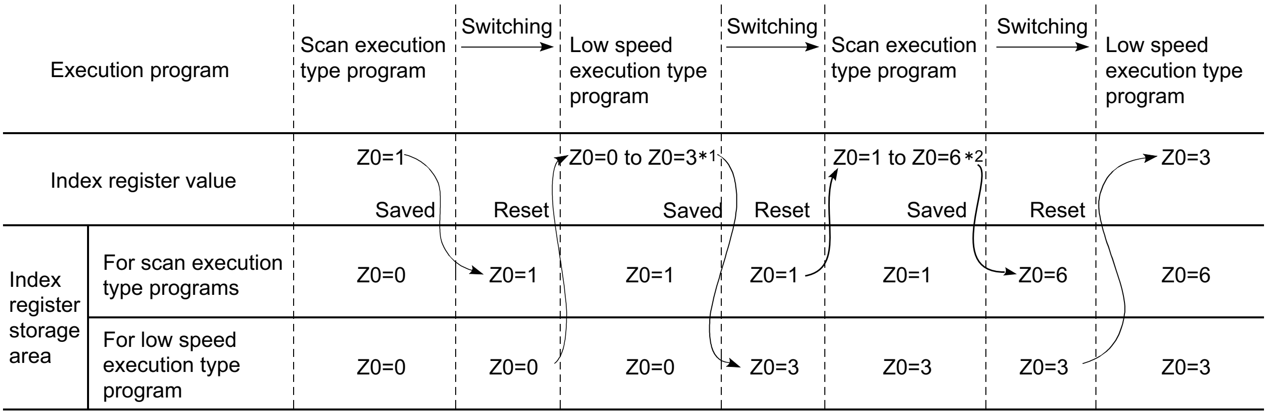
The CPU module saves (protects) and restores the index register (Z0 to 15) contents when switching between a scan execution type program and a low speed execution type program.

(1) When switching from a scan execution type program to a low speed execution type program occurs.

The CPU module saves the index register value of the scan execution type program and restores the index register value of the low speed execution type program.

(2) When switching from a low speed execution program to a scan execution type program occurs.

The CPU module saves the index register value of the low speed execution type program and restores the index register value of 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 Saved	Reset	Z0=0 to Z0=3*1 Saved	Reset	Z0=1 to Z0=6*2 Saved	Reset	Z0=3
Index register storage area	For scan execution type programs	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

*1: For low-speed execution type program, Z0 is changed to 3.

*2: For scan execution type program, Z0 is changed to 6.

Figure 10.60 Saving and restoration of index register value at switching between scan and low speed execution type programs

(3) Exchanges of index register data

Word devices should be used for exchanges of index register data between scan execution type programs and low speed execution type programs.

Basic Redundant Universal

Note10.17 Note10.17 Note10.17

The Basic model QCPU, Redundant CPU and Universal model QCPU cannot use low speed execution type programs. When the Basic model QCPU, Redundant CPU or Universal model QCPU is used, it is not necessary to be conscious of the description in this section.

Basic
Note10.18

Redundant Universal
Note10.19 Note10.19

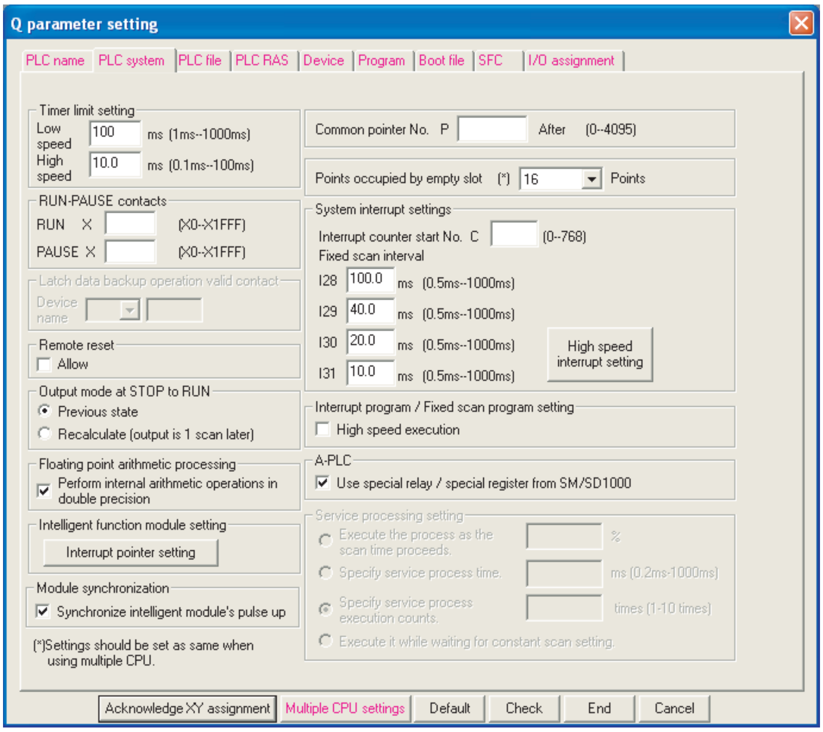
10.6.4 Switching scan/low speed exec. to Interrupt/fixed scan exec.

The CPU module performs the following processing when switching between a scan/ low speed execution type program and an interrupt/fixed scan execution type program.

- Index register value is saved (protected)/restored.
- File register block No. is saved (protected)/restored.

(1) Save (protection)/restoration setting

For the index register, when not writing date to index registers check by using an interrupt program/fixed scan execution type program, the "High speed execution" check box in the "Interrupt program/Fixed scan program setting" section at the "PLC system" tab screen in the "(PLC) Parameter" dialog box. This will switch between programs quickly.



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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)
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.

Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel

Figure 10.61 Interrupt program/fixed scan program setting

Basic
Note10.18

Basic model QCPU does not support the low speed execution type program and fixed scan execution type program.
Replace "scan/low speed execution type program" with "main routine program/sub-routine program" and "interrupt/fixed scan execution type program" with "interrupt execution type program", respectively, throughout this section.

Redundant Universal
Note10.19 Note10.19

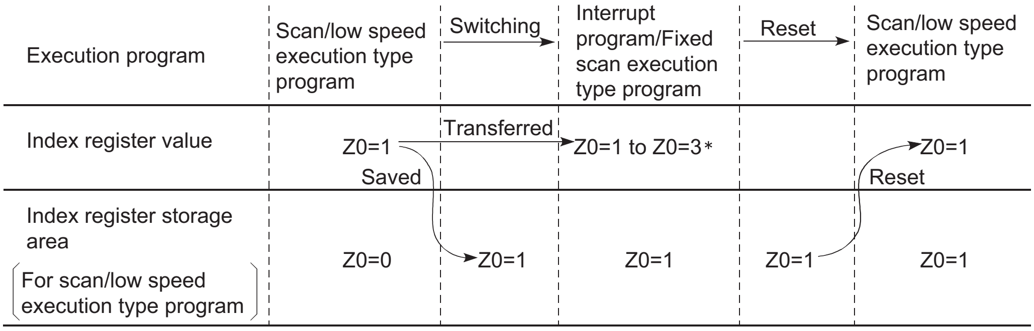
Redundant CPU and Universal model QCPU does not support the low speed execution type program.Replace "scan/low speed execution type program" with "scan execution type program" throughout this section.

(2) Index register processing

(a) When "High-speed execution" is not selected

1) **When switching from scan/low speed execution type program to interrupt/fixed scan execution type program**
   The CPU module saves the index register value of the scan/low speed execution type program and passes it to the interrupt/fixed scan execution type program.

2) **When switching from interrupt/fixed scan execution type program to scan/low speed execution type program**
   The CPU module restores the saved index register value.


Execution program
Scan/low speed execution type program
Switching
Interrupt program/Fixed scan execution type program
Reset
Scan/low speed execution type program
Index register value
Z0=1
Transferred
Z0=1 to Z0=3∗
Z0=1
Saved
Reset
Index register storage area
For scan/low speed execution type program
Z0=0
Z0=1
Z0=1
Z0=1
Z0=1

∗For interrupt program, Z0 is changed to 3.

Figure 10.62 Index register saving (protection)/restoration (When "High-speed execution" is not selected)

☒POINT

Word devices should be used to transfer index register data from an interrupt or fixed scan execution type program to a scan or low speed execution type program.

(b) When "High-speed execution" is selected

1) When switching from scan/low speed execution type program to interrupt/fixed scan execution type program

The CPU module does not save/restore the index register value.

2) When switching from interrupt/fixed scan execution type program to scan/low speed execution type program

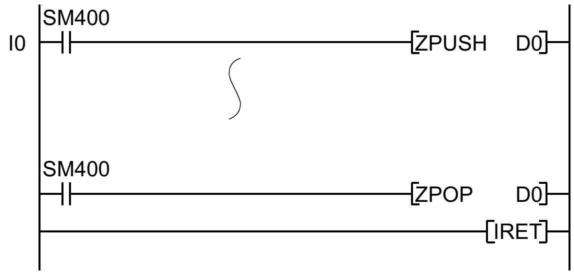
If data is written to index registers by using an interrupt program/fixed scan execution type program, the values of index registers used for an scan/low speed execution type program will be corrupted.

Execution program	Scan/low speed execution type program	Switching →	Interrupt program/Fixed scan execution type program	Reset →	Scan/low speed execution type program
Index register value	Z0=1 →	Transferred →	Z0=1 to Z0=3* →	Transferred →	Z0=3
Index register storage area (For scan/low speed execution type program)	Z0=0	Z0=1	Z0=0	Z0=0	Z0=0

* : For interrupt program, Z0 is changed to 3.

Figure 10.63 Index register saving (protection)/restoration (When "High-speed execution" is selected)

Before writing data to index registers by using an interrupt program/fixed scan execution type program, use the ZPUSH/ZPOP instruction to save/restore the data.



The points after D0 stores the data (Z0 to 15).

The data after D0 is stored in points (Z0 to 15).

Figure 10.64 When data is written to index register by interrupt/fixed scan execution type program

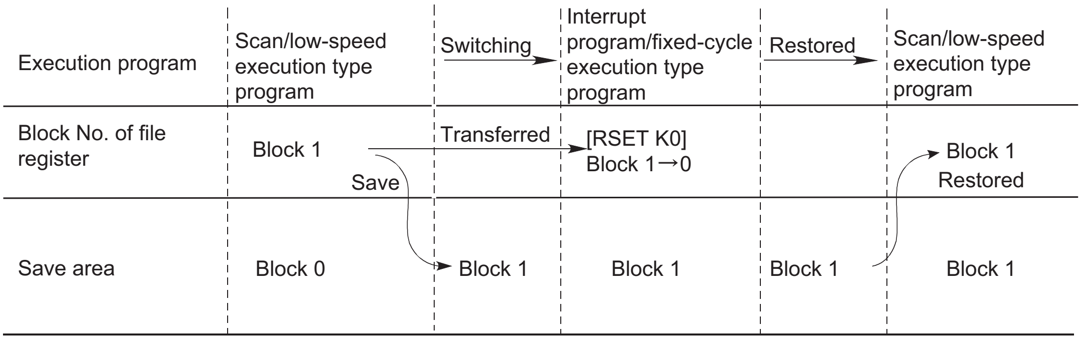
(3) Processing of file register block No.

(a) When switching from scan/low speed execution type program to interrupt/fixed scan execution type program

The CPU module saves the file register block No. of the scan/low speed execution type program and passes it to the interrupt/fixed scan execution type program.

(b) When switching from interrupt/fixed scan execution type program to scan/low speed execution type program

The CPU module restores the saved file register block No.



Execution program	Scan/low-speed execution type program	Switching →	Interrupt program/fixed-cycle execution type program	Restored →	Scan/low-speed execution type program
Block No. of file register	Block 1 Save	Transferred	[RSET K0] Block 1→0		Block 1 Restored
Save area	Block 0	Block 1	Block 1	Block 1	Block 1

Figure 10.65 File register block No. saving (protection)/restoration

10.7 File Register (R)

(1) Definition

File registers are expansion devices for data registers.

The file registers can be used at the same processing speed as the data registers.

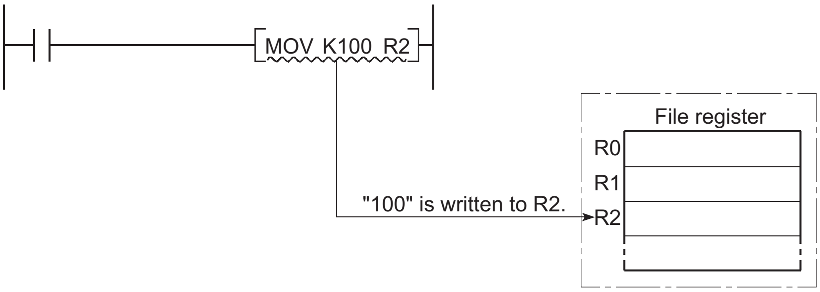


Figure 10.66 Write to file register

(2) Bit configuration of file register

(a) Bit configuration and write/read units

File registers, which consist of 16 bits per point, writes/reads data in 16bit units.

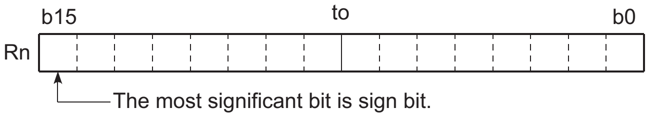


Figure 10.67 Bit configuration of file register

(b) When file register is used for 32-bit instruction

If the file registers are used for 32-bit instructions, the data will be stored in registers Rn and Rn + 1.

The lower 16 bits of data are stored in the file register No. (Rn) designated in the sequence program, and the upper 16 bits of data are stored in the designated file register No.+ 1.

For example, if file register R2 is designated in the DMOV instruction, the lower 16 bits are stored in R2, and the upper 16 bits are stored in R3.

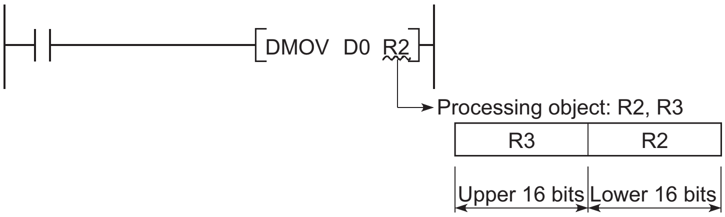


Figure 10.68 Data transfer by 32-bit instruction and storage destination

Two file registers can store a range of numeric data from -2147483648 to 2147483647 or from 0H to FFFFFFFFH. (The most significant bit in a 32-bit configuration is a sign bit.)

(3) Clearing the file register

Universal UD Note10.20

The file register contents are backed up by the battery built in the CPU module. They are backed up if the PLC is powered OFF or the CPU is reset. (It is not initialized if latch clear is conducted.) Note10.20

To initialize the file register contents, perform data clear operation in a sequence program or using GX Developer.

(a) When clearing by sequence program

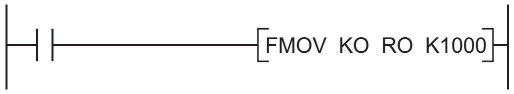


Figure 10.69 Example of clearing file registers R0 to 999

(b) When clearing by GX Developer

Choose [Online] → [Clear PLC memory] on GX Developer, and select File register all clear to clear the data.

10.7.1 File register data storage location

As the memory for storing the file register data, there are three different memories: standard RAM, SRAM card and Flash card.

The file register data storage location changes depending on the CPU module.

The file register data storage location of each CPU module is as described in Table10.13.

Table10.13 File register data storage locations

CPU module		Storage location
Basic model QCPU	Q00JCPU	None (File registers unavailable)
	Q00CPU, Q01CPU	Standard RAM
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU		Standard RAM, memory card (SRAM card, Flash card)

Universal UD Note10.20

The Universal model QCPU can set the latch range of the file register. Set the latch range of file register in the device setting of the PLC parameter. (☞ Section 6.3 (5))

10.7.2 File register capacity

(1) Using the Standard RAM

Table10.14 shows the maximum points of file registers can be stored in the standard RAM.

Note, however, that if the standard RAM stores devices other than file registers, the points can be used for file registers are decreased. (Section 5.1.1, Section 5.2.1)

Table10.14 File register capacity of each CPU module

CPU module		Number of points
Basic model QCPU	Q00JCPU	File registers unavailable
	Q00CPU, Q01CPU	64k points
High Performance model QCPU *1	Q02CPU	32k points
	Q02HCPU, Q06HCPU	64k points
	Q12HCPU, Q25HCPU	128k points
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	128k points
Redundant CPU	Q12PRHCPU, Q25PRHCPU	128k points
Universal model QCPU	Q02UCPU	64k points
	Q03UDCPU, Q03UDECPU	96k points
	Q04UDHCPU, Q04UDEHCPU	128k points
	Q06UDHCPU, Q06UDEHCPU	384k points
	Q13UDHCPU, Q13UDEHCPU	512k points
	Q26UDHCPU, Q26UDEHCPU	640k points

* 1 : The number of available points changes since the standard RAM capacity changes depending on the CPU module version. (Appendix 4)

Basic
Note10.21

(2) Using the SRAM Card Note10.21

The number of points or blocks that can be expanded depends on the CPU module.

- For High Performance model QCPU, Process CPU, and Redundant CPU
  Up to 1017k points can be stored in one file.
  Up to 32 blocks can be stored (32k words/block).
- For Universal model QCPU
  Up to 4086k points can be stored in one file.
  Up to 128 blocks can be stored (32k words/block).

Note that the number of points or blocks that can be expanded depends on the capacity of the program and device comment stored in a memory card.

Basic
Note10.21

The Basic model QCPU does not support the use of a memory card.

Basic
Note10.21

(3) Using the Flash Card Note10.21

The number of points or blocks that can be expanded depends on the CPU module.

- For High Performance model QCPU, Process CPU, and Redundant CPU
  Up to 1018k points can be stored in one file.
  Up to 32 blocks can be stored (32k words/block).
- For Universal model QCPU
  Up to 2039k points can be stored in one file.
  Up to 64 blocks can be stored (32k words/block).

Note that the number of points or blocks that can be expanded depends on the capacity of the memory card to be used and the capacity of the program and device comment stored in a memory card.

Remark

For details regarding the High Performance model QCPU memory cards, refer to Section 5.2.6.

Basic
Note10.21

The Basic model QCPU does not support the memory card.

10.7.3 Differences in access methods by storage destination memory

The file register access method changes depending on the memory.

Basic
Note10.21

Table10.15 File register access method for each memory

How to Access		Standard RAM	SRAM Card *Note10.21*	Flash Card *Note10.21*
Writing a program		○	○	×
Reading a program		○	○	○
PLC write to device memory		○	○	×
PLC read to device memory		○	○	○
How to Modify the Stored Data	Online test operation from GX Developer	○	○	×
	PLC write from GX Developer	○	○	×
	PLC write from GX Developer (Flash ROM)	×	×	○
	Batch write from serial communication module	○	○	×
	Device write from GOT900/1000 Series	○	○	×
	Random write command from GOT 900/1000 Series	○	○	×

Remark

The memory that can store file register data changes depending on the CPU module. (☞Section 10.7.1)

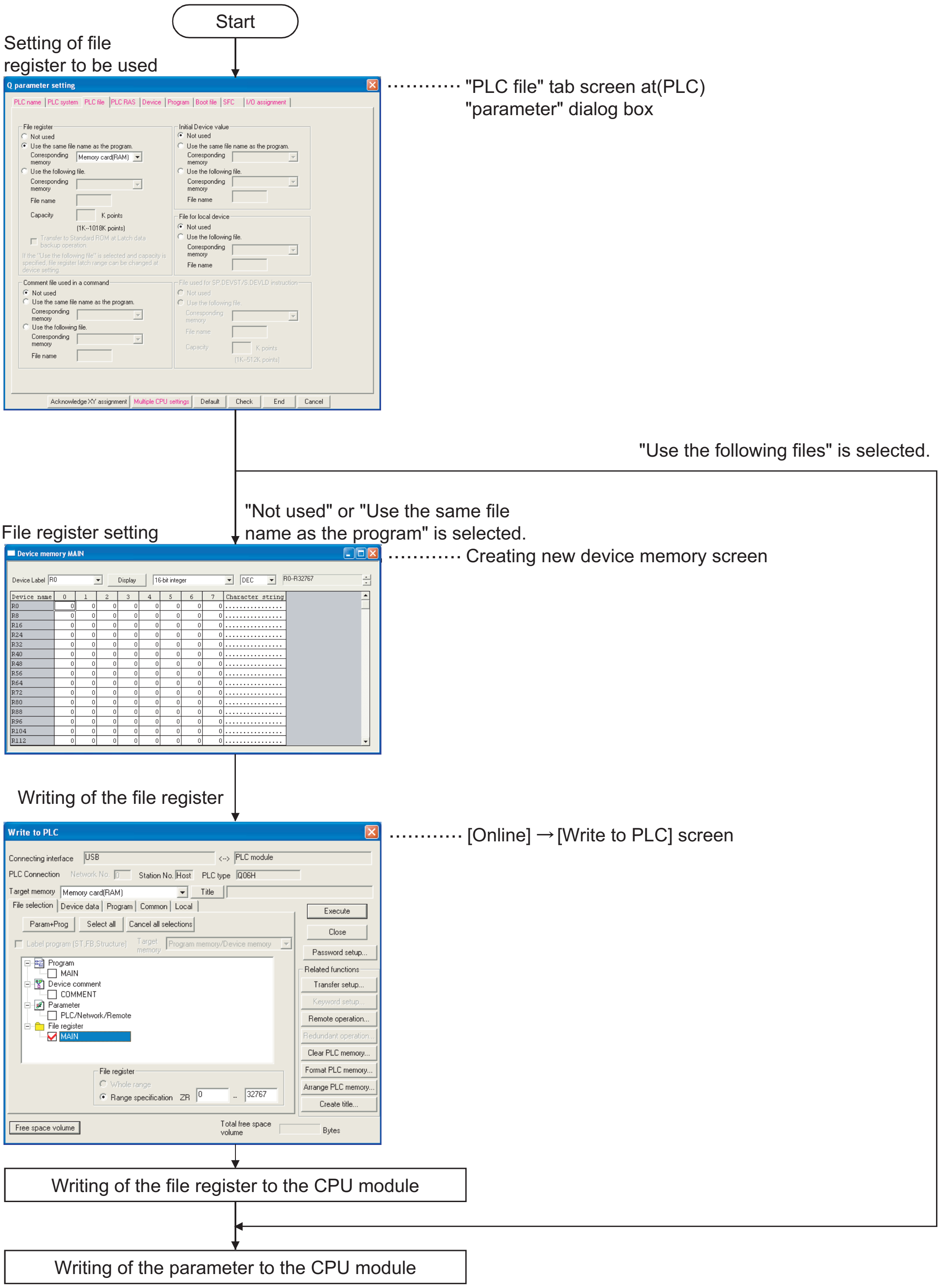
Basic
Note10.21

The Basic model QCPU does not support the memory card.

Basic Note10.22

10.7.4 File register registration procedure

To use file registers, register the file registers with the CPU module in the following steps.


Start
Setting of file register to be used
"PLC file" tab screen at(PLC) "parameter" dialog box
"Use the following files" is selected.
"Not used" or "Use the same file name as the program" is selected.
File register setting
Creating new device memory screen
Writing of the file register
[Online] → [Write to PLC] screen
Writing of the file register to the CPU module
Writing of the parameter to the CPU module

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 the "Use the following file" is selected and capacity is specified, file register latch range can be changed at device setting.

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:

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:

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

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	....

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 -- 32767

Free space volume | Total free space volume: Bytes

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...

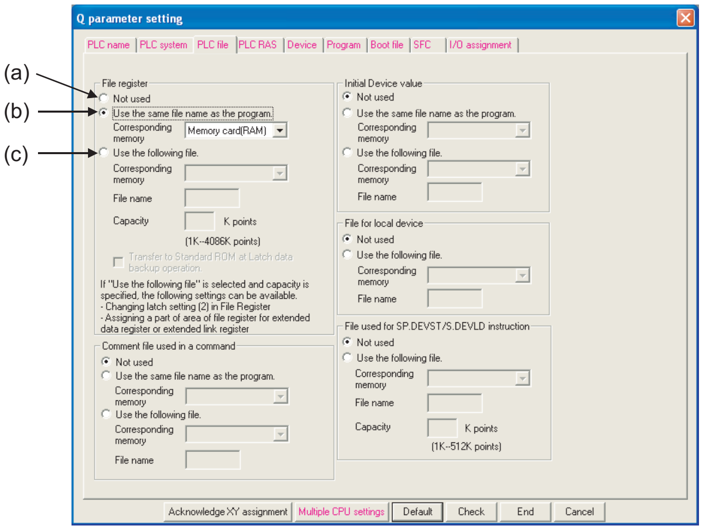
Figure 10.70 File register registration procedure

Basic Note10.22

When using the Basic model QCPU, it is not necessary to register the file registers.
The file registers are automatically registered to the standard RAM.

(1) Designating file registers for use

The standard RAM or the memory card file registers which are to be used in the sequence program are determined at the "PLC file" tab screen in the "(PLC) Parameter" dialog box.



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--4086K 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 10.71 File register setting

(a) Not used

This setting should be selected for the following cases:

- When not using the file registers
- When designating the file registers to be used in the sequence program. (The QDRSET instruction is used to designate which file registers are to be used.)

(b) **Use the same file name as the program**

This setting should be selected when the file registers having the same file name as the sequence program are to be used.

1) **Operation performed when program is changed**

If the program is changed, the file registers are automatically changed to conform to the new program name.

There are also cases where it is convenient to use the file registers as local devices which can only be used with the program currently being executed.

2) **Setting of file register points**

Choose [Online] → [Write to PLC] on GX Developer to set the number of available file register points.

Example

When file registers (A to C) having the same name as the programs (A to C) are to be used, operation is as shown below.

- At program A is execution ········ File register A is accessed.
- At program B is execution ········ File register B is accessed.
- At program C is execution ········ File register C is accessed.

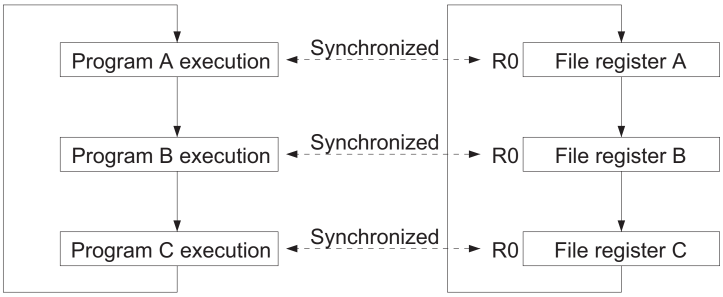


Figure 10.72 File register when program is changed

POINT

File registers dedicated to each program may not be designated with some instructions.

Refer to the allowable device in the programming manual of each instruction for details.

(c) Use the following file

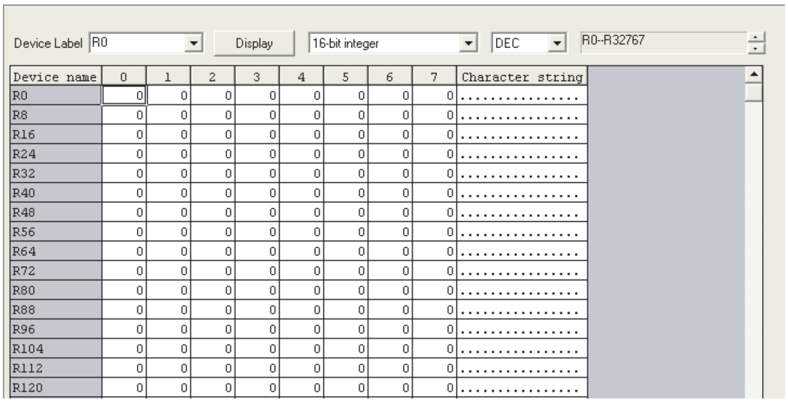
This setting should be selected when a given file register is to be shared by all executed programs.

Specify the desired parameters in the "Corresponding memory", "File name", and "Capacity" text boxes. The High Performance model QCPU creates a file register file with the specified parameters. If a parameter is not specified in the Capacity test box, this may result in the following:

- If a file register file with the specified filename is stored on the specified drive, that file register file is used.
- If a file register file with the specified filename is not found on the specified drive, a "PARAMETER ERROR (3002)" will occur.
- When an ATA card is used, a memory card (ROM) cannot be registered with the targeted memory. (File register data cannot be stored into the ATA card.) If a memory card is registered with the targeted memory, a parameter error (3000) will occur when a file register file is written to the High Performance model QCPU.

(2) File Register Setting

Use the device memory screen to specify a filename of a file register file.



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	....
R120	0	0	0	0	0	0	0	0	....

Figure 10.73 Device memory screen

(a) Device

Type "Rn" in the list box to view a listing of file registers.

Set Rn (R0 in the above case) and click the Display button to display the file register list.

(b) Setting the parameters

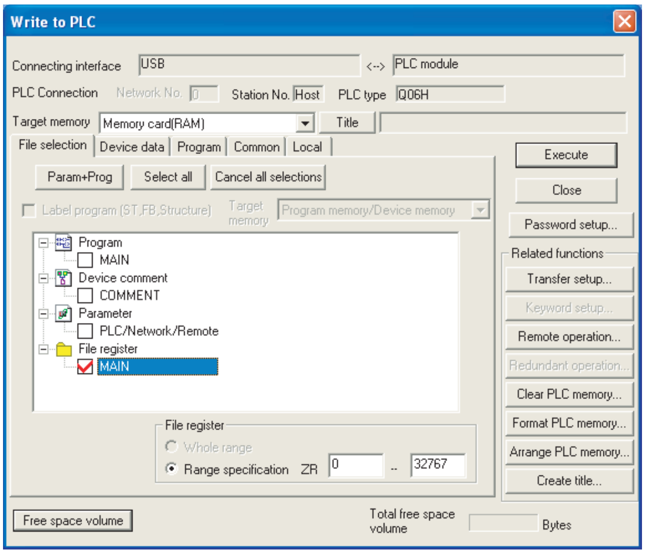
Enter the desired data in columns to specify file registers. This step is not needed when you specify only the capacity of file registers.

(3) Registering the File Register File with the CPU module

If you click on the following check boxes at the "PLC file" tab screen in the "(PLC) Parameter" dialog box, you must register a file register file with the CPU module:

- Not used (☞ (1)(a) in this section)
- Use the same file name as the program (☞ (1)(b) in this section)

Choose [Online] → [Write to PLC] on GX Developer to register the file to the CPU module.



Write to PLC

Connecting interface: USB <--> PLC module

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

Target memory: Memory card(RAM)

File selection:
- Program
  - □ MAIN
- Device comment
  - □ COMMENT
- Parameter
  - □ PLC/Network/Remote
- File register
  - ☑ MAIN

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

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

Total free space volume Bytes

Figure 10.74 File register file registration

(a) Target memory

Choose the standard RAM, memory card (RAM), or memory card (ROM) from this list box to specify a memory to store file registers. If you want to use the same filename as that of a program, store a file register file in the memory specified in the PLC File sheet of the PLC Parameter dialog box.

(b) Selecting a file register file

If a memory for file registers is selected, a filename of a file register file is displayed. Select the desired filename of a file register file.

(c) Specifying the capacity and filename

This section is used to specify the capacity of file registers and a filename of the file register file to be written onto the CPU module (QCPU-side filename).

1) File register capacity of CPU module

The file register capacity can be set in 1 point units.

However, note that the capacity is secured in 256 point units as a file. If file registers cannot be assigned from ZR0, this will result in a file register file that contains points from ZR0 to the last point. For example, if the storing range of file registers are designated from ZR1000 to ZR1791, a file register file will contain points from ZR0 to ZR1791. Specify file registers from ZR0 because undefined data is from ZR0 to ZR999. A check on the capacity of file registers is made in the units of 1k points. The capacity of file registers should be specified from R0 in the units of 1k points.

(d) Storing a file register file in the CPU module memory

This button is used to store a file register file with the specified number of points in the specified CPU module memory.

10.7.5 File register designation method

(1) Block switching format

The block switching format designates the number of file register points in 32k point (R0 to 32767) units.

If multiple blocks are used, switch to the block No. to be used in the RSET instruction for further file register settings.

Settings are designated in the R0 to 32767 range for each block.

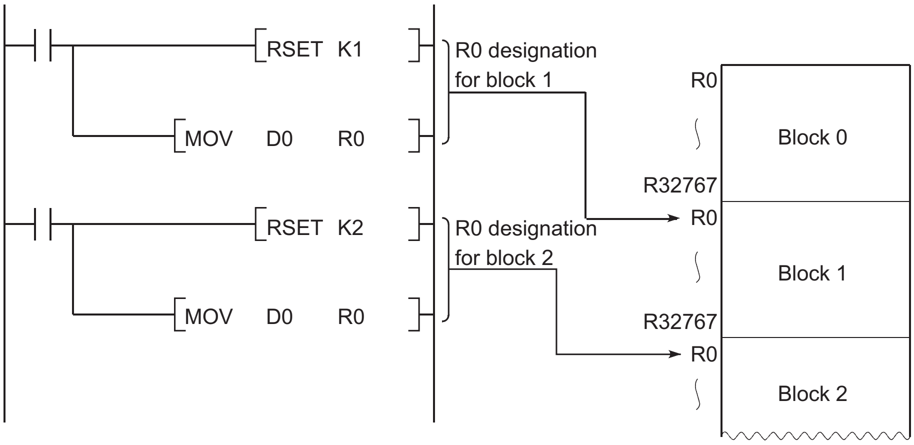


Figure 10.75 Block switching format

(2) Serial number access format

This format is used for designating file register beyond 32k points by device name.

Multiple blocks of file registers can be used as a continuous file register.

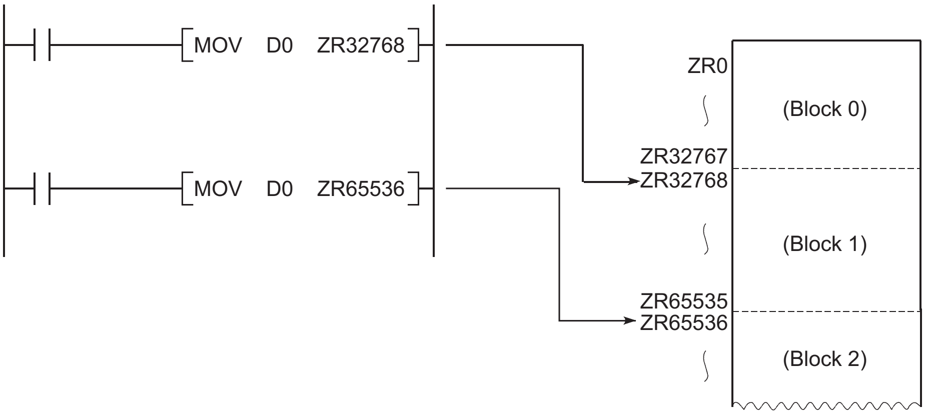


Figure 10.76 Serial No. access format

☒POINT

The block No. and number of block ZR device points that can be specified change depending on the file register storage location (☞Section 10.7.1) or file register capacity (☞Section 10.7.2).

10.7.6 Precautions for using file registers

(1) When Basic model QCPU is used

An error will not occur if data are written/read to/from the file register numbers of 64k points or more.
However, note that undefined data will be stored if data are read from the file registers.

☒POINT

The file register file (MAIN.QDR) of the Basic model QCPU cannot be deleted.
However, the file register contents can be deleted.(☞ Section 10.7(3))

(2) When High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU is used

(a) When an unregistered file register number or a file register outside the registered capacity is used

1) When the file of the file register is not registered

a) High Performance model QCPU, Process CPU, Redundant CPU
Even if data are written/read to/from a file register of which file is not registered for the CPU module, an error will not occur.
However, the following status will be assumed if data are read from the file register.
- For the standard RAM, an undefined data is stored.
- For the memory card, "0H" is stored.

b) Universal model QCPU
If data are written/read to/from a file register of which file is not registered for the CPU module, OPERATION ERROR (error code: 4101) is reported.

2) Writing/reading to/from a file register outside the registered capacity (points)

a) High Performance model QCPU, Process CPU, Redundant CPU
An error will not occur even if data are written/read to/from a file register outside the registered capacity (Points).
However, the following status will be assumed if data are read from the file register.
- For the standard RAM, an undefined data is stored.
- For the memory card, "0H" is stored.

b) Universal model QCPU
If data are written/read to/from a file register outside the registered capacity (points), OPERATION ERROR (error code: 4101) will be reported.

(b) File register capacity check

When writing/reading data to/from the file register, check the file register capacity so that data is written/read within the capacity (points) set to the CPU module.

1) File register capacity checking method

The available file register capacity can be checked in the file register capacity storage register (SD647).*1

The file register capacity is stored in SD647 in 1k point units.

* 1 : If a file register file is switched to another, the file register capacity of the currently selected file register file is stored in SD647.

☒POINT

The "less than 1k points" surplus portion of a file register capacity is not stored. In order to ensure an accurate "range of use" check, be sure to designate the file register setting in 1k point (1024 points) units.

2) Checking timing

- A file register capacity check should be executed at step 0 of programs in which file registers are used.
- After switching to another file register file using the QDRSET instruction, execute a file register capacity check.
- When using the RSET instruction to switch blocks, confirm that the switching destination block has a capacity of 1k points or more before executing the RSET instruction.

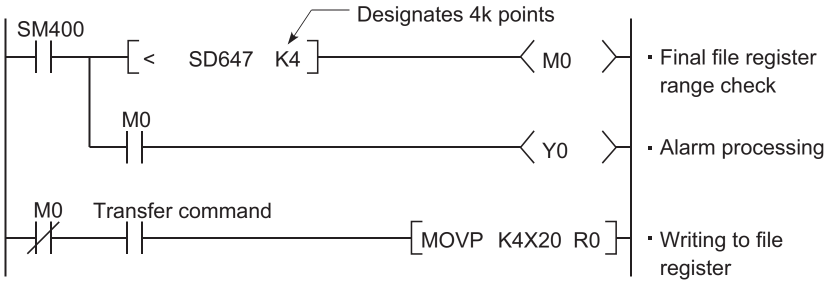
(File register capacity) > [32k points × (switching block No.) + 1k points]

3) **Checking the file register capacity**

- Check The file register capacity used for each sequence program.
- Determine if the file register capacity exceeds the number of points used, on the basis of the total file register capacity set in SD647 in the sequence program.

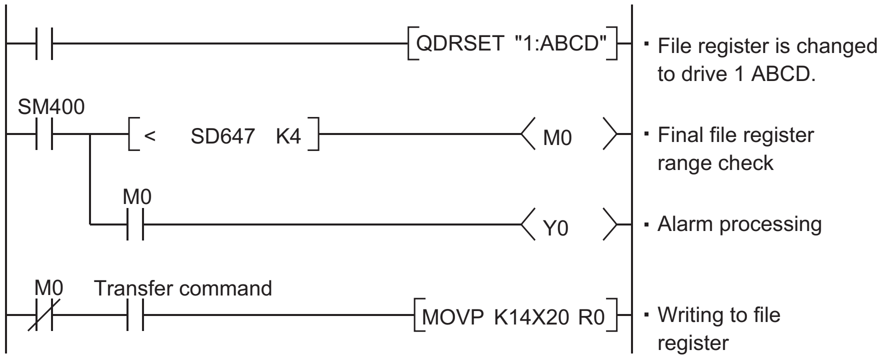
[Program example 1]

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 executing the QDRSET instruction.


QDRSET "1:ABCD"
· File register is changed to drive 1 ABCD.
SM400
< SD647 K4
M0
· Final file register range check
M0
Y0
· Alarm processing
M0
Transfer command
MOVP K14X20 R0
· Writing to file register

[Program example 3]

For block switching.

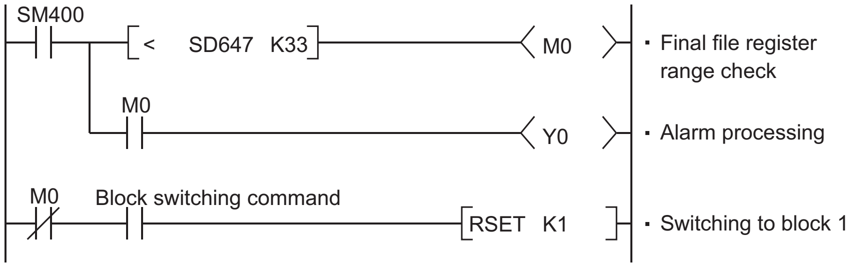


Figure 10.77 File register checking program example

(c) **File register file deletion**

Choose [Online] → [Delete PLC data] on GX Developer to delete the unnecessary file register file.

Basic Process Redundant
Note10.23 Note10.23 Note10.23
Universal
UD
Note10.23

(d) Change of file register processing time depending on CPU module version Note10.23

When the file register is specified in the serial access format (ZR □) for the access instruction to the standard RAM on the High Performance model QCPU of which first 5 digits of serial No. are 02092 or later, the processing time per instruction becomes longer than that of the High Performance model QCPU of 02091 or earlier. (The increased time per instruction is 1.1 μs for the Q02CPU and 0.65 μs for the QnHCPU on average.)
Table10.16 indicates the processing time taken when the MOV instruction is used.

Table10.16 Change of processing time depending on file register specifying method

Unit : μs

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

Basic Process Redundant
Note10.23 Note10.23 Note10.23
Universal
UD
Note10.23

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not change in file register processing time depending on the CPU module version.

10.8 Extended Data Register (D) and Extended Link Register (W)

Basic Note10.24
High Performance Note10.24
Process Note10.24
Redundant Note10.24
Universal UD Note10.25

(1) Definition

The extended data register (D) and extended link register (W) are devices to be used as an extended area of the data register (D) and link register (W) in the large-capacity file register (ZR) area. These devices can be programmed as the data register (D) and link register (W) in the file register (ZR) area.

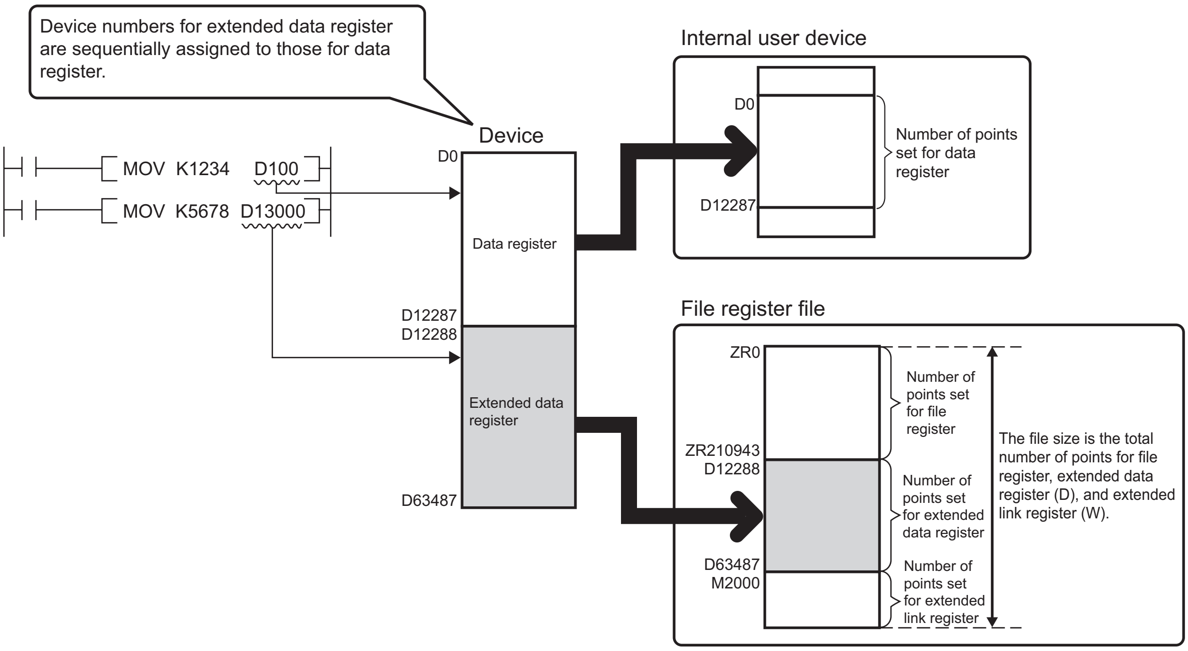

Device numbers for extended data register are sequentially assigned to those for data register.
MOV K1234 D100
MOV K5678 D13000
Device
D0
Data register
D12287
D12288
Extended data register
D63487
Internal user device
D0
D12287
Number of points set for data register
File register file
ZR0
Number of points set for file register
ZR210943
D12288
Number of points set for extended data register
D63487
M2000
Number of points set for extended link register
The file size is the total number of points for file register, extended data register (D), and extended link register (W).

Figure 10.78 Operation of extended data register (D) and extended link register (W)

Basic Note10.24 High Performance Note10.24 Process Note10.24 Redundant Note10.24

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not support the use of the extended data register (D) and extended link register (W).

Universal UD Note10.25

When using the extended data register (D) and extended link register (W) in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

(2) Device number

Device numbers for the extended data register (D) and extended link register (W) can be sequentially assigned to those for the internal user devices, data register (D) and link register (W).

☒POINT

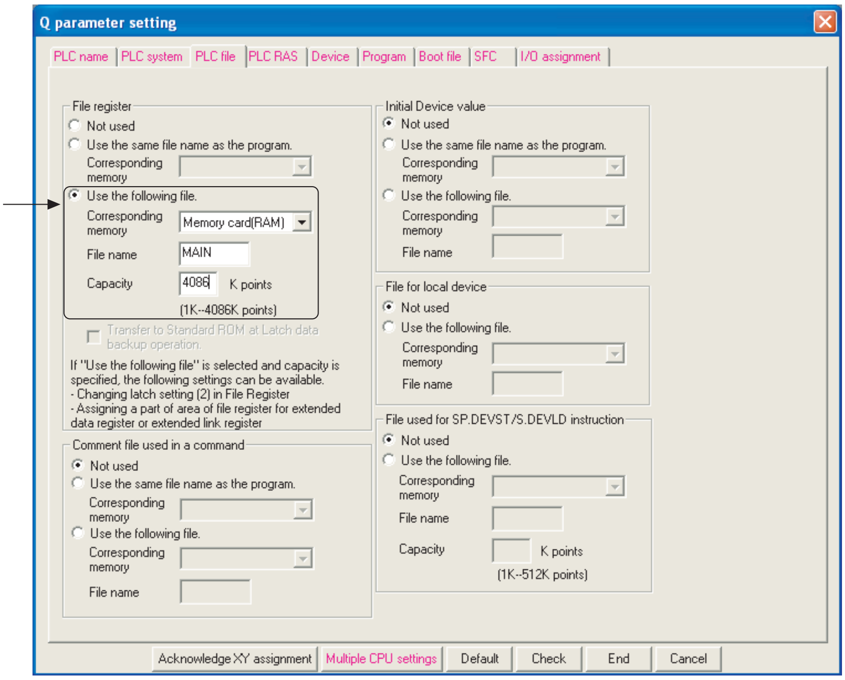
1. Even though device numbers are sequentially assigned, physical areas of the data register (D) and the extended data register (D) inside the CPU module are not connected. The same holds for the link register (W) and the extended link register (W). To use the devices as one continuous area, set the number of points for the data register (D) or link register (W) to "0" on the Device tab of PLC parameter and use only the extended data register (D) or extended link register (W).
2. When using the file register (ZR), extended data register (D) or extended link register (W) in the auto refresh setting of GX Configurator, set the device within the number of points set at File register extended setting in the Device setting of PLC parameter.

(3) Setting method

Since the extended data register (D) and extended link register (W) use the file register area, settings for both file register and device are required.

(a) File register setting

Select the "Use the following file." item at File register on the PLC file tab of PLC parameter in GX Developer and set the items in Table 10.17. Do not select the "Use the same file name as the program." item.



Q parameter setting

File register
- ○ Not used
- ○ Use the same file name as the program.
  - Corresponding memory:
- ● Use the following file.
  - Corresponding memory: Memory card(RAM)
  - File name: MAIN
  - Capacity: 4086 K points (1K--4086K 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)

Set corresponding memory, file name, and capacity.

Figure 10.79 Screen for setting the number of points for the file register

Table10.17 File register setting items

Item	Corresponding memory		File name	Capacity[*1]	Remarks
"Use the following file."	Memory card (RAM)		Any	1 to 4086k points[*2]	--
	Memory card (ROM)		Any	1 to 2039k points[*3]	Read only
	Standard RAM	Q02UCPU	Any	1 to 64k points	--
		Q03UDCPU, Q03UDECPU		1 to 96k points	
		Q04UDHCPU, Q04UDEHCPU		1 to 128k points	
		Q06UDHCPU, Q06UDEHCPU		1 to 384k points	
		Q13UDHCPU, Q13UDEHCPU		1 to 512k points	
		Q26UDHCPU, Q26UDEHCPU		1 to 640k points	

* 1: The capacity is the total number of points for the file register (ZR), extended data register (D), and extended link register (W).
* 2: The capacity indicates the maximum number of points when SRAM card (8M bytes) is used.
* 3: The capacity indicates the maximum number of points when Flash card (4M bytes) is used.

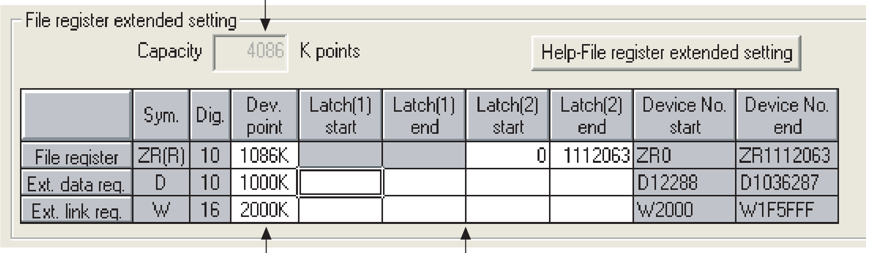
(b) **Device setting**

Set the number of points for each file register (ZR), extended data register (D), and extended link register (W) at File register extended setting on the Device tab of PLC parameter in GX Developer.
Assign a part of points set for the file register (ZR) on the PLC file tab of PLC parameter to the extended data register (D) and extended link register (W).

Set the latch range if data needs to be latched.

- Latch (2) of the file register (ZR)
- Latch (1) and (2) of the extended data register (D)
- Latch (1) and (2) of the extended link register (W)

The number of point set for file register on the PLC file tab is displayed.



File register extended setting

Capacity 4086 K points

Help-File register extended setting

	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	1086K			0	1112063	ZR0	ZR1112063
Ext. data reg.	D	10	1000K					D12288	D1036287
Ext. link reg.	W	16	2000K					W2000	W1F5FFF

Set the number of device points so that the total equals to the point set for file register on the PLC file tab.

Set the latch range if data needs to be latched.

Figure 10.80 File register extended setting

Once the number of device points for the extended data register (D) and extended link register (W) is set, areas for these devices are reserved in the file register file.

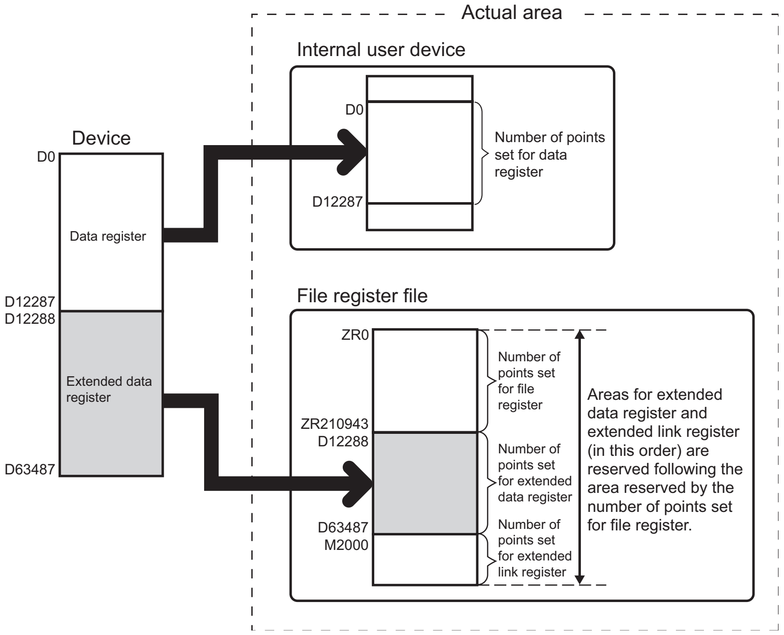

Actual area
Internal user device
D0
D12287
Number of points set for data register
Device
D0
Data register
D12287
D12288
Extended data register
D63487
File register file
ZR0
ZR210943
D12288
D63487
M2000
Number of points set for file register
Number of points set for extended data register
Number of points set for extended link register
Areas for extended data register and extended link register (in this order) are reserved following the area reserved by the number of points set for file register.

Figure 10.81 Areas for extended data register (D) and extended link register (W)

(4) Checking the number of device points in special register

The number of device points assigned for each file register (ZR), extended data register (D), and extended link register (W) can be checked in the following special registers.

- SD306 and 307: File register (ZR)
- SD308 and 309: Extended data register (D)
- SD310 and 311: Extended link register (W)

(5) Precautions

Precautions for using the extended data register (D) and extended link register (W) are described below.

1) When the extended data register (D) and extended link register (W) are specified, the values of the following items will be the same as those for the file register.
    - Number of steps in a program
    - Instruction processing time
    - Auto refresh processing time with network modules
    - Auto refresh processing time with intelligent function modules
    - Auto refresh processing time between CPUs in a multiple CPU system

2) The file register capacity cannot be changed during RUN.

3) The file register cannot be switched using the QDRSET instruction (OPERATION ERROR (error code: 4100).

4) Set the refresh range for the following auto refresh settings so that the range does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).
    - Auto refresh with network modules
    - Auto refresh with intelligent function modules
    - Auto refresh between CPUs in a multiple CPU system

5) Set the following items so that the specification does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).
    - Index modification
    - Indirect specification
    - Specification for the instructions which target data block

* 1: Data block indicates the following data.
- Data used in the instructions, such as FMOV, BMOV, BK+, which multiple words are targeted for operation
- Control data, composed of two or more words, specified in the instructions, such as SP.FWRITE, SP.FREAD
- Data whose data type is 32-bit or more (BIN 32-bit, real number, indirect address of the device)

For details on index modification and indirect specification with the extended data register (D) and extended link register (W), refer to the following manual.

☞QCPU(Q Mode)/QnACPU Programming Manual (Common Instructions)

6) To access the extended data register (D) or extended link register (W) from a module which does not support the use of those devices, device number needs to be specified by that of the file register (ZR).
Calculation formulas to obtain device number of the file register (ZR) to be specified to access the extended data register (D) or extended link register (W) and their calculation examples are described below.

Table10.18 Calculation formulas to obtain device number of file register (ZR) to be specified to access extended data register (D) or extended link register (W)*1

Item	Calculation formula
Device number of the file register (ZR) to access the extended data register (D)	$ED_{ZN} = ZR_C + (ED_N - D_C)$
Device number of the file register (ZR) to access the extended link register (W)	$EW_{ZN} = ZR_C + ED_C + (EW_N - W_C)$

* 1: Variables in the table above indicate the following.
- ZR_C: Number of points set for the file register (ZR)
- ED_{ZN}: Device number of the file register (ZR) to access the extended data register (D)
- ED_N: Access target device number of the extended data register (D)
- D_C: Number of points set for the data register (D)
- ED_C: Number of points set for the extended data register (D)
- EW_{ZN}: Device number of the file register (ZR) to access the extended link register (W)
- EW_N: Access target device number of the extended link register (W) (hexadecimal)
- W_C: Number of points set for the link register (W)

[Calculation example]
- D_C: Number of points set for the data register (D) ••• 12288 points
- W_C: Number of points set for the link register (W) ••• 8192 points
- ZR_C: Number of points set for the file register (ZR) ••• 2048 points
- ED_C: Number of points set for the extended data register (D) ••• 2048 points

① Device number of the device register (ZR) to access D13000
$ED_{ZN} = 2048 + (13000 - 12288)$
$= 2760$

② Device number of the device register (ZR) to access W2100
$EW_{ZN} = 2048 + 2048 + (2100_H - 8192)$
$= 2048 + 2048 + (8448 - 8192)$
$= 4352$

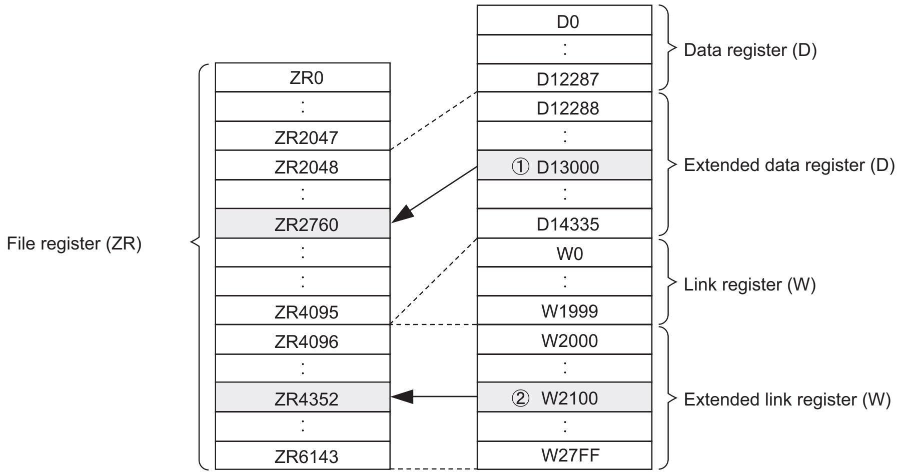


Figure 10.82 Access to extended data register (D) and extended link register (W)

10.9 Nesting (N)

(1) Definition

Nesting is a device used in the master control instruction (MC instruction, MCR instruction) to program operation conditions in a nesting structure.

(2) Specifying method in master control instruction

The master control instruction opens/closes a common ladder bus to create a sequence program of efficient ladder switching.

Specify nesting in ascending order (in order of N0 to N14), starting from the outside of the nesting structure.

Refer to the following manual for how to use nesting.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

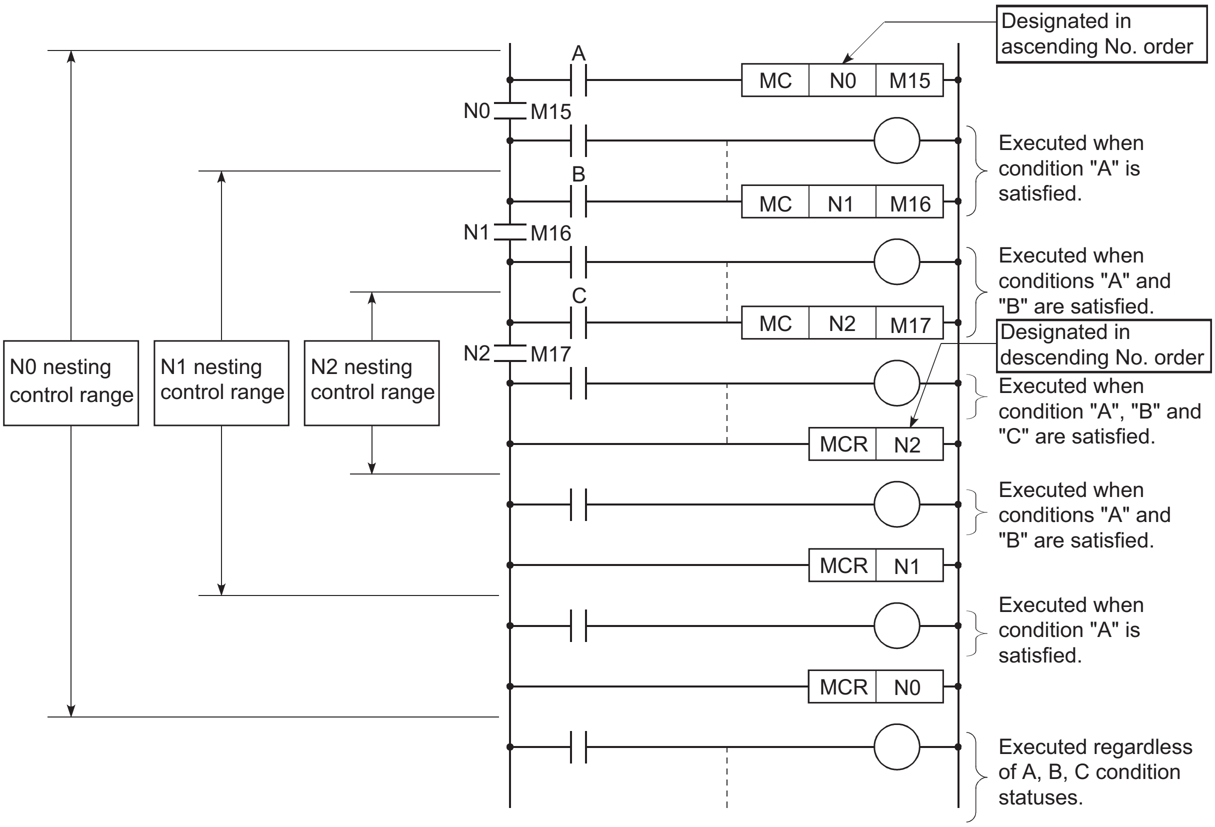

Designated in ascending No. order
A
MC N0 M15
N0 M15
Executed when condition "A" is satisfied.
B
MC N1 M16
N1 M16
Executed when conditions "A" and "B" are satisfied.
C
MC N2 M17
Designated in descending No. order
N2 M17
Executed when condition "A", "B" and "C" are satisfied.
MCR N2
Executed when conditions "A" and "B" are satisfied.
MCR N1
Executed when condition "A" is satisfied.
MCR N0
Executed regardless of A, B, C condition statuses.
N0 nesting control range
N1 nesting control range
N2 nesting control range

Figure 10.83 Program example using nesting

10.10 Pointer (P)

(1) Definition

Pointer devices are used in jump instructions (CJ, SCJ, JUMP) or subroutine call instructions (CALL).

(2) Pointer applications

The pointers can be used in the following applications.

- Pointers are used in jump instructions (CJ, SCJ, JMP) to designate jump destinations and labels (jump destination beginning).
- Pointers are used in subroutine CALL instructions (CALL, CALLP) to designate the CALL destination and label (subroutine beginning).

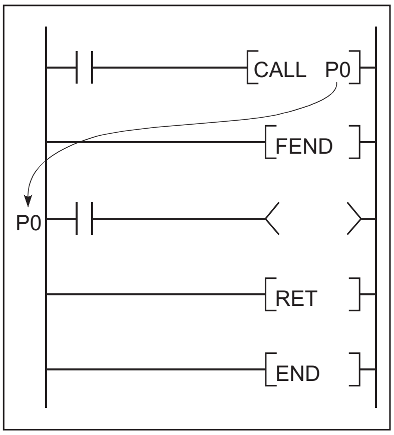


Figure 10.84 Program using pointer

(3) Pointer types

(a) Basic model QCPU

Since the Basic model QCPU cannot execute multiple programs, it is irrelevant to the difference between local pointers and common pointers.

(b) High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

There are the following two different pointer types.

- Local pointer (☞Section 10.10.1)
  Pointer used independently in each program.
- Common pointer (☞Section 10.10.2)
  Pointer that can be called by all executed programs using the subroutine call instruction.

(4) Number of available pointer points

The number of available pointer points changes depending on the CPU module.

Table10.19 Number of available pointer points of each CPU module

CPU module	Number of points
Basic model QCPU	300 points
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	4096 points

Remark

Refer to the following manual for the jump instruction and subroutine call instruction.

☞QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic
Note10.26

10.10.1 Local pointer

(1) Definition

Local pointers are pointers which can be used independently in program jump instructions and subroutine call instructions.

The same pointer No. can be used in each of the programs.

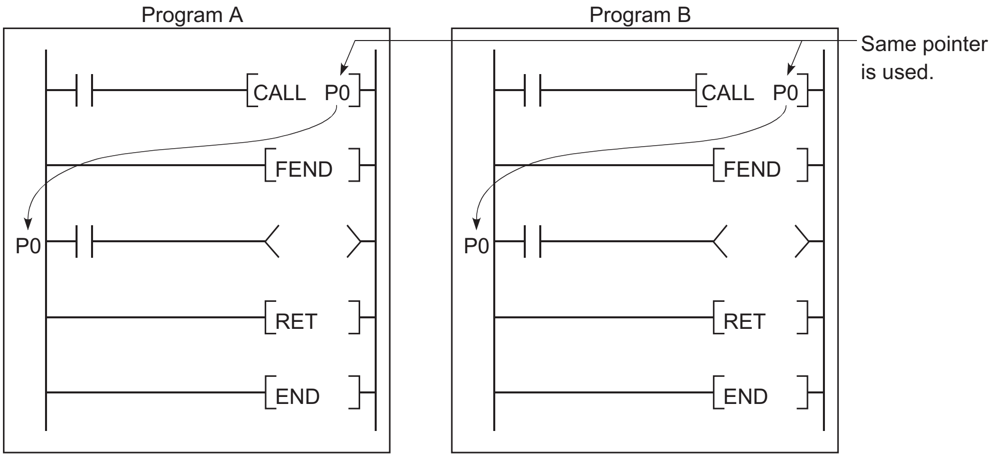
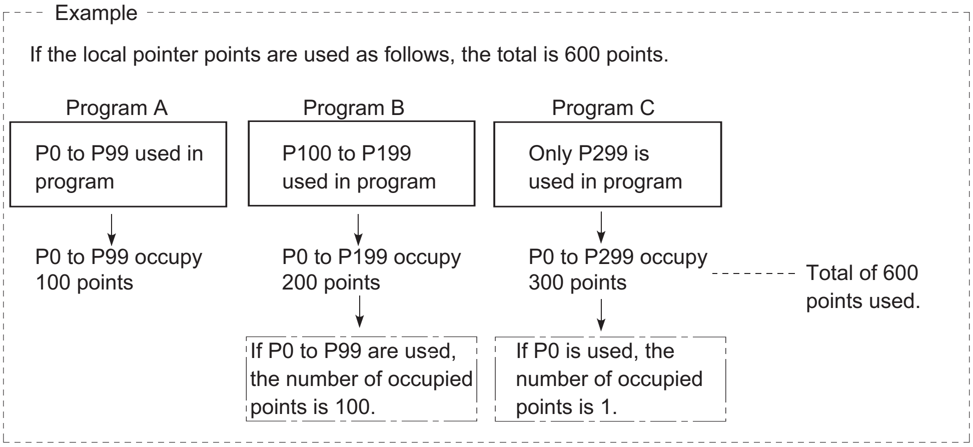


Figure 10.85 Use of the same pointer in each program (Local pointer)

(2) Number of local pointer points

Local pointers can be divided among all the programs stored in the program memory.
The local pointer No. ranges from P0 to the highest No. of the local pointer in use. (The CPU module OS computes the number of points used.)
Even if only P99 is used in a program, for example, the number of points used will be counted as 100 between P0 and P99.
When local pointers are used at several programs, the pointer settings should begin from P0.



Example

If the local pointer points are used as follows, the total is 600 points.

Program A
P0 to P99 used in program
P0 to P99 occupy 100 points
Program B
P100 to P199 used in program
P0 to P199 occupy 200 points
If P0 to P99 are used, the number of occupied points is 100.
Program C
Only P299 is used in program
P0 to P299 occupy 300 points
If P0 is used, the number of occupied points is 1.
Total of 600 points used.

Figure 10.86 Concept of the number of local pointer points

Basic
Note10.26

Since the Basic model QCPU cannot execute multiple programs, it is irrelevant to the difference between local pointers and common pointers.
When the Basic model QCPU is used, it is not necessary to be conscious of the description in this section.

(3) Precautions for using local pointers

(a) Program where local pointers are described

Local pointers cannot be used from other program jump instructions and subroutine CALL instructions.

Use an ECALL instruction to call a subroutine subprogram in a program file that contains local pointers.

(b) Total number of local pointer points

If the total number of pointers (total for all programs) exceeds 4096 points, a pointer configuration error (error code:4020) occurs.

Basic
Note10.27

10.10.2 Common pointer

(1) Definition

Common pointers are used to call subroutine programs from all programs being executed in the High Performance model QCPU.

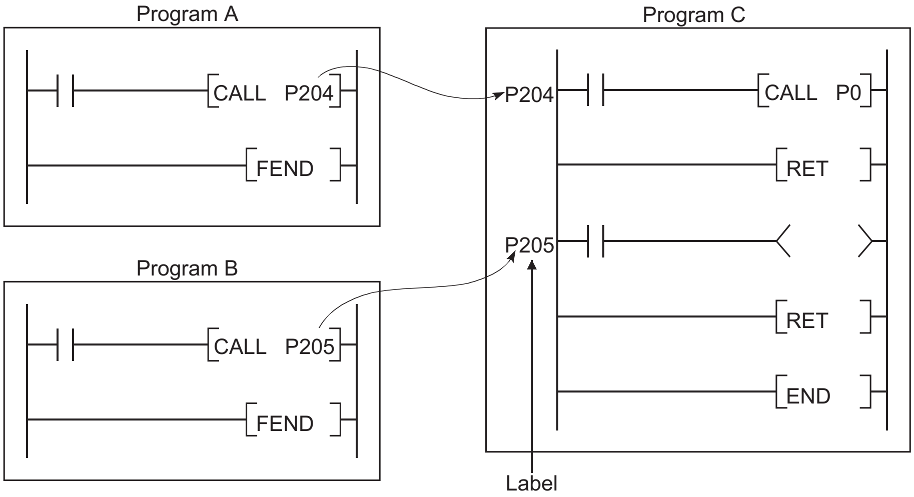


Figure 10.87 Calling the pointers in other programs (Common pointers)

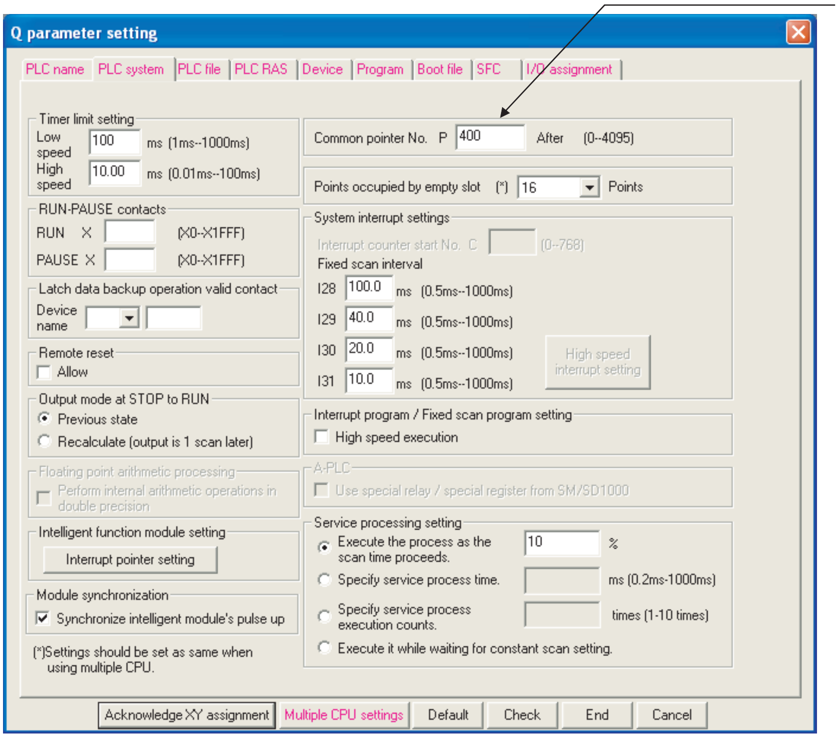
Basic
Note10.27

Since the Basic model QCPU cannot execute multiple programs, it is irrelevant to the difference between local pointers and common pointers.
When the Basic model QCPU is used, it is not necessary to be conscious of the description in this section.

(2) Common pointer range of use

In order to use common pointers, the first common pointer No. must be designated at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

A range of common pointers starts from a specified pointer number to P4095. However, only pointer numbers subsequent to the local pointer range can be designated by parameter setting as common pointers.



Timer limit setting
Low speed 100 ms (1ms--1000ms)
High speed 10.00 ms (0.01ms--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 400 After (0--4095)

Points occupied by empty slot (*) 16 Points

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

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. 10 %
○ 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.

Set the head number of the common pointers here.

Figure 10.88 Common pointer No. setting screen

If a total of 400 points are used in three programs (100 points in Program A, 100 points in Program B, and 200 points in Program C), for example, all local pointers after P400 can be used as common pointers.

9 Parameters
10 Device Explanation
11 CPU Module Processing Time
12 Procedure for Writing Program to CPU Module
Appendices
Index

(3) Precautions for using common pointers

(a) When the same pointer No. is used as label

The same pointer No. cannot be used again as a label.
Such use will result in a pointer configuration error (error code:4021).

(b) When the total number of local pointer points exceeds the first number of common pointer

If the last number of local pointers used in several programs overlaps the first number of common pointers, a "pointer configuration error (Error Code: 4020) will occur.

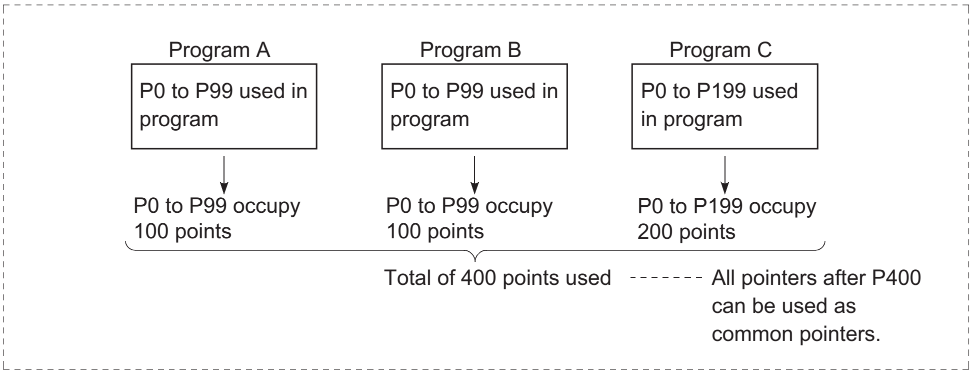

Program A
P0 to P99 used in program
P0 to P99 occupy 100 points
Program B
P0 to P99 used in program
P0 to P99 occupy 100 points
Program C
P0 to P199 used in program
P0 to P199 occupy 200 points
Total of 400 points used
All pointers after P400 can be used as common pointers.

Figure 10.89 Concept of common pointer using range

☒POINT

In the jump instruction, jumping to common pointers in other programs is not allowed.
Common pointers should be used only with subroutine call instructions.

10.11 Interrupt pointer (I)

(1) Definition

Interrupt pointers are used as labels at the beginning of interrupt programs.

The interrupt pointers can be used in all running programs.

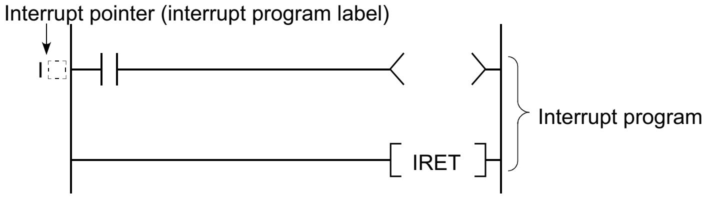


Figure 10.90 Interrupt pointer

(2) Number of available interrupt pointer points

Table10.20 indicates the number of available interrupt pointer points.

Table10.20 Number of available interrupt pointer points of each CPU module

CPU module	Number of points
Basic model QCPU	128 points (I0 to 127)
High Performance model QCPU Process CPU Redundant CPU Universal model QCPU	256 points (I0 to 255)

(3) Interrupt factors

Table10.21 indicates the interrupt factors of the interrupt pointers.

Table10.21 Interrupt factor classification

Interrupt factor	Interrupt pointer No.	Description	Applicable CPU module				
			Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	Universal model QCPU
Interrupt module[*1, *10]	I0 to 15	Interrupt input from the interrupt module[*1]	○	○	○	○	○
Special function module for starting sequence	I16 to 27	Interrupt from the AnS/A series compalible special function module that can make the CPU module start an interrupt.	×	○	×	×	×
Internal timer	I28 to 31, I49	Fixed scan interrupt by the internal timer of the CPU module.	△[*2]	△[*3]	△[*2]	△[*2]	△[*2]
Error[*5]	I32 to 41	Interrupt made by occurrence of error that continues sequence program operation.	×	△[*4]	△[*4]	○	○
Multiple CPU synchronization	I45	The fixed scan interrupt to execute control with synchronizing the operation cycle of the Motion controller with interrupt cycle.	×	×	×	×	○[*6]
Intelligent function module	I50 to 255[*7]	Interrupt from the intelligent function module.[*8]	○	○	○	○	○

○ : Available △ : Some interrupt pointers unavailable × : N/A

* 1 : Refer to the following manual for the available interrupt modules.
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)
* 2 : The interrupt pointer I49 (high-speed interrupt function dedicated pointer) is available for the QnHCPU only.
* 3 : The Q02CPU cannot use the interrupt pointer I49 (high-speed interrupt function dedicated pointer).
* 4 : The interrupt pointers I40, I41 are available for the Redundant CPU only.
* 5 : For the error to which the error-time operation mode can be set in the PLC RAS setting of the PLC parameter dialog box, an interrupt is executed only when the setting is "Continue".
* 6 : Applicable for the Universal model QCPU and Motion controller that are compatible with the multiple CPU high speed main base unit (Q3□ DB) and multiple CPU high speed transmission.
Not applicable for the single CPU system using the Universal model QCPU.
* 7 : The Basic model QCPU is I50 to 127.
* 8 : The target module is serial communication module, MELSECNET/H module, Ethernet module, and high-speed counter module.
For details, refer to the manual for each module.
* 9 : The target module is the intelligent communication module etc.
For details, refer to the manual for each module.
* 10 : When using the interrupt module with the High Performance model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.2)

POINT

To use the intelligent function module interrupt (☞ Section 6.23), the intelligent function module setting (interrupt points setting) is required at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

10.11.1 List of interrupt pointer Nos and interrupt factors

The following table lists the interrupt pointer Nos. and interrupt factors of each CPU module.

(1) Basic model QCPU

Table10.22 Interrupt pointer No. and interrupt factor list (Basic model QCPU)

I No.	Interrupt factors		Priority level
I0	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 27	N/A	--	--
I28	Internal timer [*1]	100ms	4
I29		40ms	3
I30		20ms	2
I31		10ms	1
I32 to 49	N/A	--	--
I50 to 127 [*2*3]	Intelligent function module/ Interrupt module (QI60)	Set which intelligent function module/interrupt module (QI60) is used with parameters.	21 to 98

* 1 : The time-out period of the internal timer is a default value.
It can be changed in 1ms units within the range 2ms to 1000ms in the PLC system setting of the PLC parameter dialog box.

* 2 : To use the intelligent function module interrupt, the intelligent function module setting (interrupt pointer setting) is required in the PLC system setting of the PLC parameter dialog box. (Refer to Section 6.23 for the interrupt from the intelligent function module.)

* 3 : Among I50 to 127, I50 has the highest priority (priority 21) and I127 has the lowest priority (priority 98).

(2) High Performance model QCPU

Table10.23 Interrupt pointer No. and interrupt factor list (High Performance model QCPU)

I No.	Interrupt Factors		Priority Ranking
I0	Interrupt module (QI60/A1SI61)	1st point	220
I1		2nd point	221
I2		3rd point	222
I3		4th point	223
I4		5th point	224
I5		6th point	225
I6		7th point	226
I7		8th point	227
I8		9th point	228
I9		10th point	229
I10		11th point	230
I11		12th point	231
I12		13th point	232
I13		14th point	233
I14		15th point	234
I15		16th point	235
I16	Special function module for starting sequence*1	1st point	208
I17		2nd point	209
I18		3rd point	210
I19		4th point	211
I20		5th point	212
I21		6th point	213
I22		7th point	214
I23		8th point	215
I24		9th point	216
I25		10th point	217
I26		11th point	218
I27		12th point	219
I28	Internal timer*2	100ms	239
I29		40ms	238
I30		20ms	237
I31		10ms	236

I No.	Interrupt Factors		Priority Ranking
I32*5	Error*5*6	Errors that stop operation	1
I33		SINGLE PS DOWN *3 *4	
I34		UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR MULTI CPU ERROR	
I35		OPERATION ERROR SFCP OPE. ERROR SFCP EXE. ERROR EX. POWER OFF	
I36		ICM. OPE ERROR FILE OPE. ERROR	
I37		Empty	
I38		PRG. TIME OVER	
I39		CHK instruction execution Annunciator detection	
I40 to 48	--	Empty	--
I49	Internal timer*2	0.2ms to 1.0ms*7	*10
I50 to 255	Intelligent function module *8*9	Specifies which intelligent function module is used with parameters.	2 to 207

* 1 : 1st to 12th points are allocated in order, beginning from the sequence start generator module installed closest to the High Performance model QCPU.

* 2 : The time-out period of the internal timer is a default value.
It can be changed in 0.5ms units within the range 0.5ms to 1000ms in the PLC system setting of the PLC parameter dialog box.

* 3 : This applies to the CPU module with serial No. of first 5 digits "07032" or later.

* 4 : In a multiple CPU system configuration, it can be used only by CPU No.1 when all CPU modules serial No. of first 5 digits is "07032" or later.

* 5 : When an error interrupt with "I32 (error that stops operation)" occurs, the High Performance model QCPU stops after performing the processing of I32 at error occurrence.

* 6 : I32 to 48 are disabled from execution (DI) when the system is powered on/the High Performance model QCPU is reset.
When using interrupt pointers I32 to 48, execute the IMASK instruction to enable interrupts.
Refer to the following manual for the IMASK instruction.
☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

* 7 : Set the time-out period of the internal timer by choosing "PLC system" - "System interrupt setting" - "High speed interrupt setting" on the PLC parameter screen.
Set it in the setting range 0.2ms to 1.0ms in 0.1ms increments.

* 8 : To use the intelligent function module for interrupt, the intelligent function module setting (interrupt pointer setting) must be made in the PLC system setting of the PLC parameter dialog box. (Refer to Section 6.23 for the interrupt from the intelligent function module.)

* 9 : Among I50 to 255, I50 has the highest priority (priority 2) and I255 has the lowest priority (priority 207).

* 10 : When I49 is set in the PLC parameters, other interrupt programs (I0 to I48, I50 to I255) or fixed scan execution type programs must not be executed. If or fixed scan execution type program or the like is executed, the interrupt program using I49 cannot be executed at the set interrupt cycle intervals.

(3) Process CPU

Table10.24 Interrupt pointer number and interrupt factor list (Process CPU)

I No.	Interrupt Factors		Priority Ranking
I0	Interrupt module (QI60/A1SI61)	1st point	208
I1		2nd point	209
I2		3rd point	210
I3		4th point	211
I4		5th point	212
I5		6th point	213
I6		7th point	214
I7		8th point	215
I8		9th point	216
I9		10th point	217
I10		11th point	218
I11		12th point	219
I12		13th point	220
I13		14th point	221
I14		15th point	222
I15		16th point	223
I16	--	Empty	--
I17			
I18			
I19			
I20			
I21			
I22			
I23			
I24			
I25			
I26			
I27			
I28	Internal timer *1	100ms	227
I29		40ms	226
I30		20ms	225
I31		10ms	224
I32*4	Error *4*5	Errors that stop operation	1
I33		SINGLE PS DOWN *2*3	
I34		UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR MULTI CPU ERROR	
I35		OPERATION ERROR SFCP OPE. ERROR SFCP EXE. ERROR EX.POWER OFF	
I36		ICM. OPE ERROR FILE OPE. ERROR	
I37		Empty	
I38		PRG. TIME OVER	
I39		CHK instruction execution Annunciator detection	
I40 to 49	--	Empty	--
I50 to 255	Intelligent function module *6*7	Specifies which intelligent function module is used with parameters.	2 to 207

* 1 : The time-out period of the internal timer is a default value.
It can be changed in 0.5ms units within the range 0.5ms to 1000ms in the PLC system setting of the PLC parameter dialog box.
* 2 : This applies to the CPU module with serial No. of first 5 digits "07032" or later.
* 3 : In a multiple CPU system configuration, it can be used only by CPU No.1 when all CPU modules serial No. of first 5 digits is "07032" or later.
* 4 : When an error interrupt with "I32 (error that stops operation)" occurs, the Process CPU stops after performing the processing of I32 at error occurrence.
* 5 : I32 to 48 are disabled from execution (DI) when the system is powered on/the Process CPU is reset.
When using interrupt pointers I32 to 48, execute the IMASK instruction to enable interrupts.
Refer to the following manual for the IMASK instruction.
☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)
* 6 : To use the intelligent function module for interrupt, the intelligent function module setting (interrupt pointer setting) must be made in the PLC system setting of the PLC parameter dialog box. (Refer to Section 6.23 for the interrupt from the intelligent function module.)
* 7 : Among I50 to 255, I50 has the highest priority (priority 2) and I255 has the lowest priority (priority 207).

(4) Redundant CPU

Table10.25 Interrupt pointer No. and interrupt factor list (Redundant CPU)

I No.	Interrupt Factors		Priority Ranking	I No.	Interrupt Factors		Priority Ranking
I0	Interrupt module (QI60)	1st point	208	I32*2	Error*2*3	Errors that stop operation	1
I1		2nd point	209	I33		SINGLE PS. DOWN	
I2		3rd point	210	I34		UNIT VERIFY ERR.	
I3		4th point	211			FUSE BREAK OFF	
I4		5th point	212			SP. UNIT ERROR	
I5		6th point	213	I35		OPERATION ERROR	
I6		7th point	214			SFCP OPE. ERROR	
I7		8th point	215			SFCP EXE. ERROR	
I8		9th point	216			EX.POWER OFF	
I9		10th point	217	I36		ICM. OPE ERROR	
I10		11th point	218			FILE OPE. ERROR	
I11		12th point	219	I37		Empty	
I12		13th point	220	I38		PRG. TIME OVER	
I13		14th point	221	I39		CHK instruction execution	
I14		15th point	222			Annunciator detection	
I15		16th point	223	I40		CAN'T SWITCH	
I16	--	Empty	--	I41		STANDBY	
I17				I42 to 49	--	Empty	--
I18							
I19				I50 to 255	Intelligent function module *4*5	Specifies which intelligent function module is used with parameters.	2 to 207
I20							
I21							
I22							
I23							
I24							
I25							
I26							
I27							
I28	Internal timer *1	100ms	227				
I29		40ms	226				
I30		20ms	225				
I31		10ms	224				

* 1 : The time-out period of the internal timer is a default value. It can be changed in 0.5ms units within the range 0.5ms to 1000ms in the PLC system setting of the PLC parameter dialog box.

* 2 : When an error interrupt with "I32 (error that stops operation)" occurs, the Redundant CPU stops after performing the processing of I32 at error occurrence.

* 3 : I32 to 48 are disabled from execution (DI) when the system is powered on/the Redundant CPU is reset.
When using interrupt pointers I32 to 48, execute the IMASK instruction to enable interrupts.
Refer to the following manual for the IMASK instruction.
☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

* 4 : To use the intelligent function module for interrupt, the intelligent function module setting (interrupt pointer setting) must be made in the PLC system setting of the PLC parameter dialog box. (Refer to Section 6.23 for the interrupt from the intelligent function module.)

* 5 : Among I50 to 255, I50 has the highest priority (priority 2) and I255 has the lowest priority (priority 207).

(5) Universal model QCPU

Table10.26 Interrupt pointer No. and interrupt factor list (Universal model QCPU)

I No.	Interrupt Factors		Priority Ranking
I0	Interrupt module (QI60)	1st point	6
I1		2nd point	7
I2		3rd point	8
I3		4th point	9
I4		5th point	10
I5		6th point	11
I6		7th point	12
I7		8th point	13
I8		9th point	14
I9		10th point	15
I10		11th point	16
I11		12th point	17
I12		13th point	18
I13		14th point	19
I14		15th point	20
I15		16th point	21
I16 to I27	--	Empty	--
I28*4	Internal timer *1	100ms	5
I29*4		40ms	4
I30*4		20ms	3
I31*4		10ms	2

I No.	Interrupt Factors		Priority Ranking
I32 to 44	--	Empty	--
I45*2*4	Multiple CPU synchronization	0.88ms	1
I46 to 49	--	Empty	--
I50 to 255	Intelligent function module *3*5/Interrupt module (QI60)	Specifies which intelligent function module is used with parameters.	22 to 227

* 1 : The time-out period of the internal timer is a default value. It can be changed in 0.5ms units within the range 0.5ms to 1000ms in the PLC system setting of the PLC parameter dialog box.

* 2 : Available for the Multiple CPU system using the QnUCPU.

* 3: Setting for the "Intelligent function module setting (Interrupt pointer setting)" on the PLC system setting tab of PLC parameter is required to use interrupts from intelligent function modules. (Refer to Section 6.23 for details on interrupts from intelligent function modules.)

* 4 : Among I50 to 255, I50 has the highest priority (Priority order: 22), and I255 has the lowest priority (Priority order: 227).

* 5 : When an interrupt occurs, CAN'T EXECUTE(I) (error code:4220) does not occur even though no interrupt pointer exists in the program.

10.12 Other Devices

10.12.1 SFC block device (BL)

This device is used for checking if the block designated by the SFC program is valid.
Refer to the following manual for how to use the SFC block device.

☞QCPU (Q Mode)/QnACPU Programming Manual (SFC)

10.12.2 SFC transition device (TR)

Basic Note10.28

Universal Note10.28

This device is used for checking if a forced transition is designated for a specified transition condition in a specified SFC program block.
Refer to the following manual for how to use the SFC transition device.

☞QCPU (Q Mode)/QnACPU Programming Manual (SFC)

10.12.3 Network No. designation device (J)

(1) Definition

The network No. designation device is used to designate the network No. in link instructions.

(2) Designating network No. designation device

The network No. designation device is designated in the link instruction as shown in Figure 10.91.

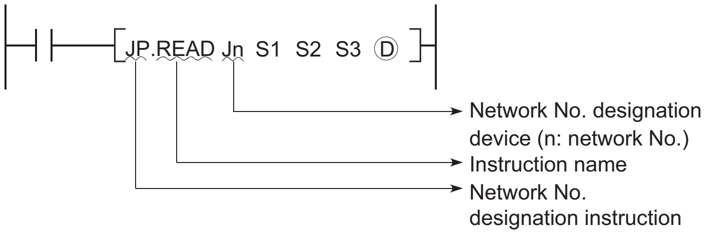

JP.READ Jn S1 S2 S3 Ⓓ
Network No. designation device (n: network No.)
Instruction name
Network No. designation instruction

Figure 10.91 How to specify the network No. designation device

Remark

Refer to the following manual for the link instructions.

☞Q Corresponding MELSECNET/H Network System Reference Manual

Basic Note10.28 Universal Note10.28

The Basic model QCPU and Universal model QCPU cannot use the SFC transition device (TR).

10.12.4 I/O No. designation device (U)

(1) Definition

I/O No. designation devices are used with instructions dedicated to intelligent function module to designate I/O numbers.

(2) Designating the I/O No. designation device

I/O No. designation devices are designated with the intelligent function module instructions as shown in Figure 10.92.

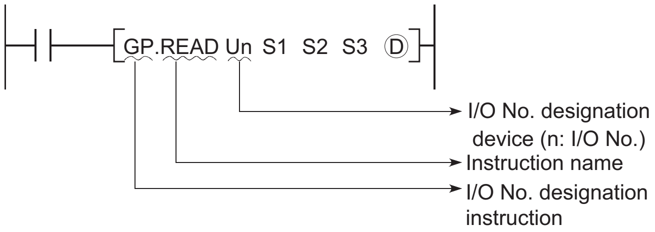

GP.READ Un S1 S2 S3 D
I/O No. designation device (n: I/O No.)
Instruction name
I/O No. designation instruction

Figure 10.92 How to specify the I/O No. designation device

Remark

For details on intelligent function module instructions, refer to the corresponding manual for the intelligent function module to be used.

10.12.5 Macro instruction argument device (VD)

(1) Definition

Macro instruction argument devices are used with ladders registered as macros.

When a VD□ setting is designated for a ladder registered as a macro, conversion to the designated device is performed when the macro instruction is executed.

(2) Designating macro instruction argument devices

Specify the devices transferred from sequence programs to macro registration ladders as macro instruction argument devices among the devices used in the ladders registered as macro with GX Developer.

Designate devices that correspond to the macro argument devices used in the macro registration ladders in ascending order, when using macro instructions in a sequence program.

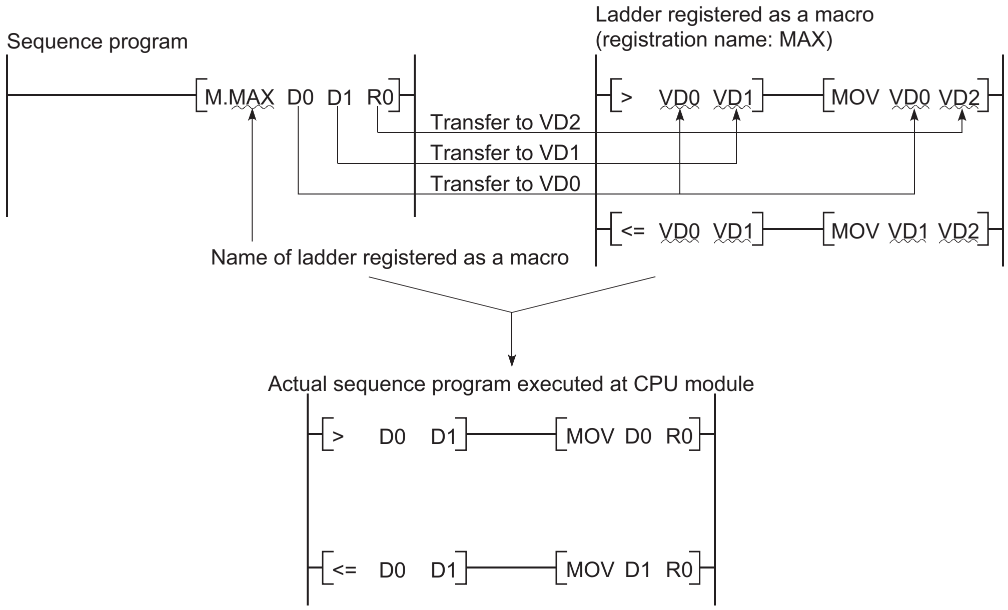
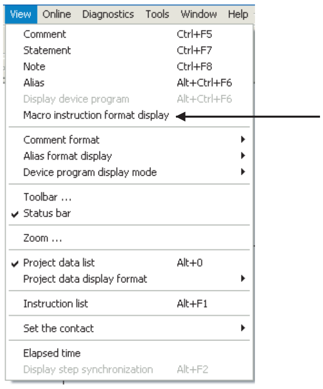


Figure 10.93 Designating the macro instruction argument device

☒POINT

1. With the macro instruction argument device, VD0 to 9 can be used in one ladder registered as a macro instruction.
2. The GX Developer read mode provides an option to view a program in macro instruction format.
   (To change the display, choose [View] → [Display macro instruction format].)



Change of macro instruction display

Figure 10.94 Settings for macro instruction display

10.13 Constants

10.13.1 Decimal constant (K)

(1) Definition

Decimal constants are devices that designate decimal data in sequence programs.

Specify it as K□(example: K1234) in a sequence program.

It is stored in binary (BIN) into the CPU module. (☞Section 3.9.1)

(2) Designation range

The designation ranges for decimal constants are as follows:

- For word data (16 bits) •••••••••••••••••••••••• K-32768 to 32767
- For 2-word data (32 bits)•••••••••••••••••••••• K-2147483648 to 2147483647

☒POINT

The most significant bit is a sign bit.

10.13.2 Hexadecimal constant (H)

(1) Definition

Hexadecimal constants are devices which designate hexadecimal or BCD data in sequence programs.
(For BCD data designations, 0 to 9 digit designations are used.)

Hexadecimal constants are designated as "H□ " settings (e.g. H1234).
(☞Section 3.9.2)

(2) Designation range

The setting ranges for hexadecimal constants are as follows:

- For word data (16 bits) •••••••••••••••••••••••• H0 to FFFF
  (H0 to 9999 for BCD)
- For 2-word data (32 bits)•••••••••••••••••••••• H0 to FFFFFFFF
  (H0 to 99999999 for BCD)

Basic
Note10.29

10.13.3 Real number (E)

(1) Definition

Real numbers are devices which designate real numbers in the sequence program.

Specify it as E□ (example: K1234) in a sequence program.(☞ Section 3.9.4)

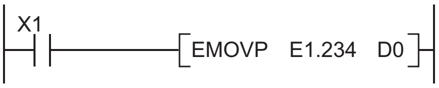


Figure 10.95 Specifying the real number

(2) Designation range

(a) Setting range of real number

Basic High Performance Process
Note10.30 Note10.30 Note10.30
Redundant
Note10.30

The setting range of the real number is as follows:

- For the single-precision floating-point data

  $-2^{128} <$ Device $\leqq -2^{-126}$, 0 , $2^{-126} \leqq$ Device $< 2^{128}$

- For the double-precision floating-point data Note10.30

  $-2^{1024} <$ Device $\leqq -2^{-1022}$, 0 , $2^{-1022} \leqq$ Device $< 2^{1024}$

(b) The operations when an overflow or underflow occurs

The following table shows the operation of the CPU module when an overflow or underflow occurs at the time of arithmetic operation.

Table10.27 Operations when an overflow or underflow occurs

CPU module	Operation at overflow	Operation at underflow
Basic model QCPU	4100 Error occurrence	4100 Error occurrence
High Performance model QCPU	4100 Error occurrence	4100 Error occurrence
Process CPU	4100 Error occurrence	4100 Error occurrence
Redundant CPU	4100 Error occurrence	Becomes 0 without error occurrence
Universal model QCPU	4141 Error occurrence	Becomes 0 without error occurrence

Basic
Note10.29

When using the real number operation function for the Basic model QCPU, check the versions of the CPU module and GX Developer. (☞ Appendix 4.1)

Basic High Performance Process
Note10.30 Note10.30 Note10.30
Redundant
Note10.30

The Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU cannot use the double-precision floating-point data.

(c) The operations when a special value is entered[*1]

The following table shows the operation when an arithmetic operation where the input data is a special value is performed.

Table10.28 Operations when a special value is entered

CPU module	Nonnumeric number input
Basic model QCPU	4100 Error occurrence
High Performance model QCPU	4100 Error occurrence[*2]
Process CPU	4100 Error occurrence
Redundant CPU	4100 Error occurrence
Universal model QCPU	4140 Error occurrence

* 1 : The special values are -0, denormalized number, nonnumeric number, and ±∞.

* 2 : An error occurs only when the internal operation is set to the single precision.

(3) Designation method

Real numbers can be designated in sequence programs by a "normal expression" or an "exponential expression".

- Normal expression ••••••••••••••• The specified value is designated as it is.
  For example, 10.2345 becomes E10.2345.
- Exponential expression ••••••••• The specified value is multiplied by a " × 10^n" exponent.
  For example, 1234 becomes E1.234 + 3. [*1]

* 1 : + 3 in E1.234 + 3 indicates 10^3.

Basic
Note10.31

10.13.4 Character string (" ")

(1) Definition

Character string constants are devices used to designate character strings in sequence programs.

They are designated by quotation marks (e.g. "ABCD1234").

(2) Usable characters

All ASCII code characters can be used in character strings.

The QCPU is sensitive to uppercase and lowercase characters.

(3) Number of designated characters

Character strings extend from the designated character to the NUL code (00H).

You can use up to 32 characters for a character string in an instruction such as $MOV.

Basic
Note10.31

The Basic model QCPU can use character strings for only the $MOV, STR, DSTR, VAL, DVAL, ESTR and EVAL instructions.

10.14 Convenient Usage of Devices

Basic Note10.32

When executing multiple programs in the CPU module, local devices among the internal user devices can be designated to execute each of the programs in an independent manner. Note10.32

Basic Note10.33

CPU module devices are classified into "global devices" shared by all the programs being executed and "local devices" used independently by each of the programs. Note10.33

10.14.1 Global devices

Global devices can be shared by all the programs being executed in the CPU module. Global device data are stored in the CPU module device memory, and can be used by all programs.

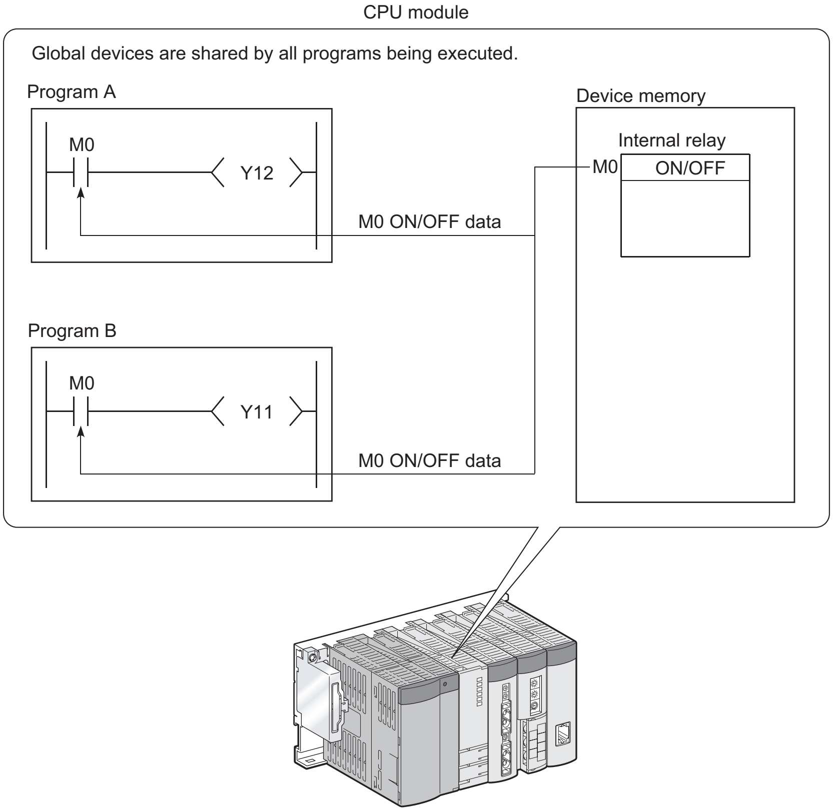


Figure 10.96 Using the global device

Basic Note10.32

Since the Basic model QCPU cannot execute multiple programs, it does not support the function that makes each program independent by local device designation.

Basic Note10.33

Since the Basic model QCPU cannot execute multiple programs, it is irrelevant to the difference between local devices and common devices. When the Basic model QCPU is used, it is not necessary to be conscious of the description in this section.

☒POINT

1. The devices that have not been set as local devices (☞ Section 10.14.2) are all global devices.
2. When executing multiple programs, the "shared range" for all programs, and the "independent range" for each program must be designated in advance. (☞ Section 10.14.2)

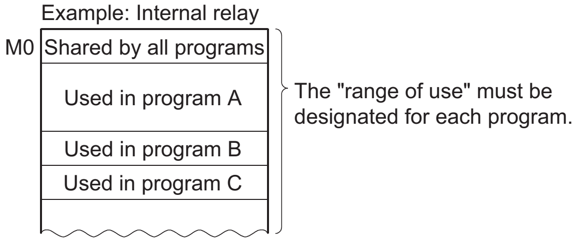


Figure 10.97 Determination of device using range

10.14.2 Local devices

Local devices are used independently by the programs.
The use of local devices permits programming of multiple "independent execution" programs without regard to other programs.
Note that the local device data can be stored to the standard RAM and the memory card (SRAM) only.

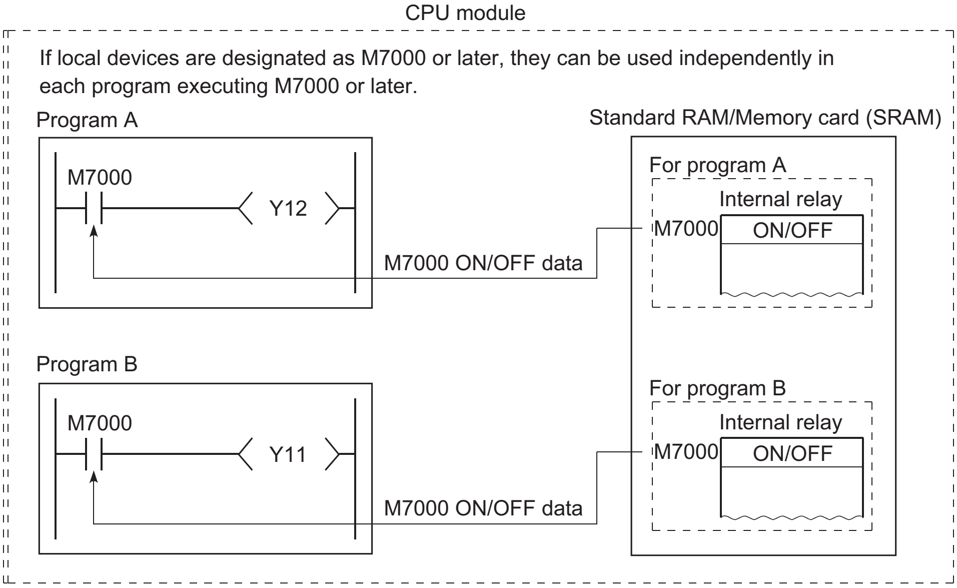

CPU module
If local devices are designated as M7000 or later, they can be used independently in each program executing M7000 or later.
Program A
M7000
Y12
M7000 ON/OFF data
Standard RAM/Memory card (SRAM)
For program A
Internal relay
M7000
ON/OFF
Program B
M7000
Y11
M7000 ON/OFF data
For program B
Internal relay
M7000
ON/OFF

Figure 10.98 Using the local devices

(1) Devices that can be used as local devices

The following devices can be used as local devices.

- internal relays (M)
- edge relays (V)
- timers (T, ST)
- counters (C)
- data registers (D)

(2) Saving and restoration of local device file

Programs which use the local device exchange the local device file data stored in the standard RAM or the memory card (SRAM) with the data in the device memory of the CPU module.
Therefore the scan time is extended by this data exchange time.

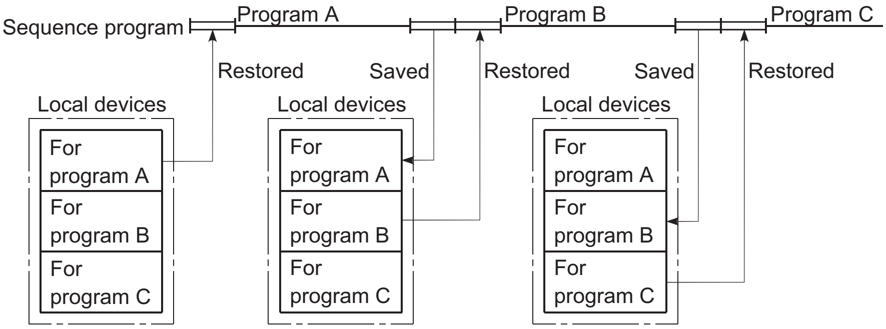


Figure 10.99 Saving and restoration of local device file

☒POINT

The local device may not be designated with some instructions.
Refer to the allowable device in the programming manual of each instruction for details.

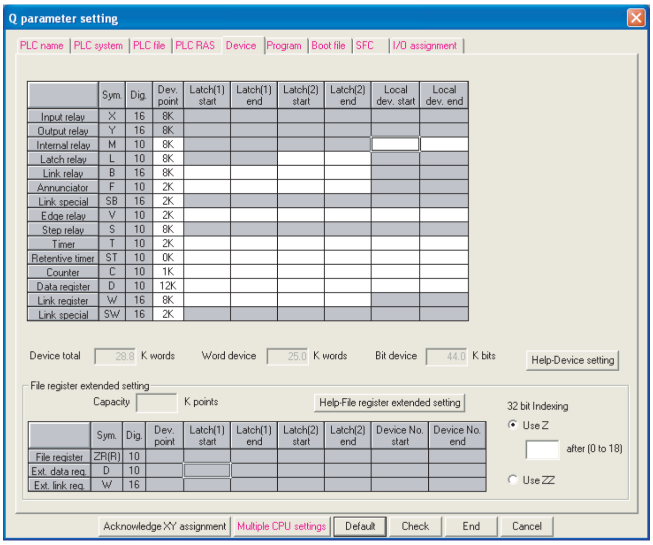
Remark

Refer to Section 10.2 for the concept of the number of words of the devices used as local devices.

(3) Local device designation

(a) Setting the ranges of devices used as local devices

When using as local devices, set the ranges of the devices used as local devices in the device of the PLC parameter dialog box.



	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						

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 10.100 Device

Note that the range designated for local devices applies to all programs, and cannot be changed for individual programs.
For example, if the local device range is designated as M0 to M100, this range will be used for local devices in all programs.

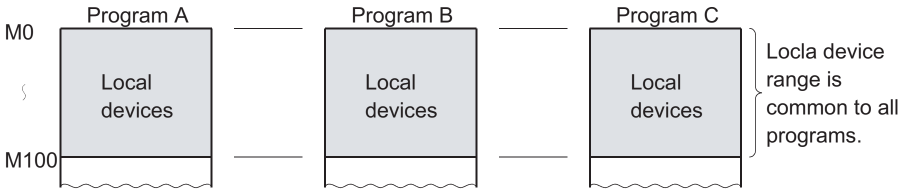
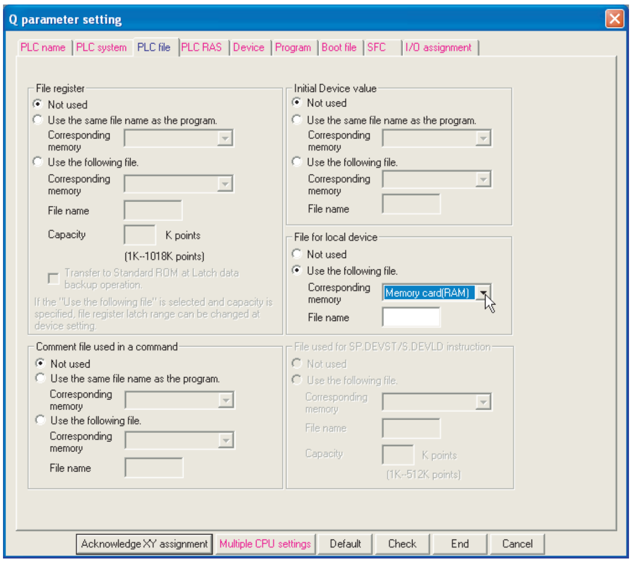


Figure 10.101 Local device ranges

(b) Setting the drive and file name where local devices will be stored

After setting the ranges of the devices used as local devices, set the drive and file name, where the local device file will be stored, in the PLC file of the PLC parameter dialog box.



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 the "Use the following file" is selected and capacity is specified, file register latch range can be changed at device setting.

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

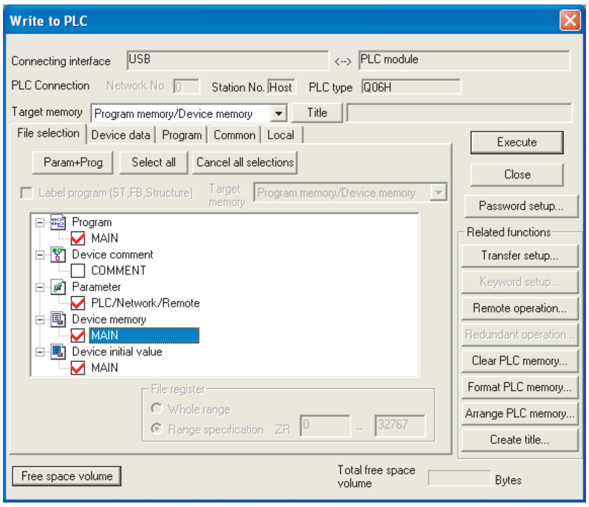
Comment file used in a command
● Not used
○ Use the same file name as the program.
Corresponding memory
○ Use the following file.
Corresponding memory
File name

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 10.102 PLC file

(c) Write of settings

Write the settings made in (a) and (b) to the CPU module.
To write them, execute [Write to PLC] on 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

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
- Device memory
  - ☑ MAIN
- Device initial value
  - ☑ MAIN

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

Free space volume | Total free space volume Bytes

Figure 10.103 Write to device memory

☒POINT

1. Performing the setting to change the size of local device in the standard RAM, with the sampling trace file stored in the standard RAM, clears the sampling trace file.
   To keep the trace result in your personal computer, perform the following operations.
   - Click the [Trace result PLC read] button on the sampling trace execution screen to read the trace result in the personal computer. (☞ Section 6.14 (4))
   - Click the [Trace result] button to display the trace result.
   - Click the [Create CSV file] button to store the trace result in CSV format.
2. Unless designated as "local devices", all devices are global devices.

Universal UD Note10.34

(4) Setting whether to use a local device (use for each program)Note10.34

Setting whether to use a local device for each program can reduce the scan time. For the Universal model QCPU, the size of the local device file can be reduced because the programs not using a local device does not create the area for saving and restoring data.

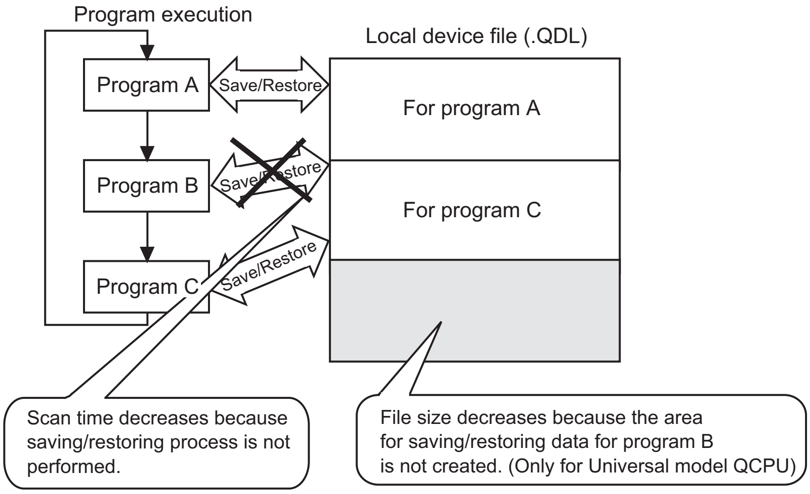
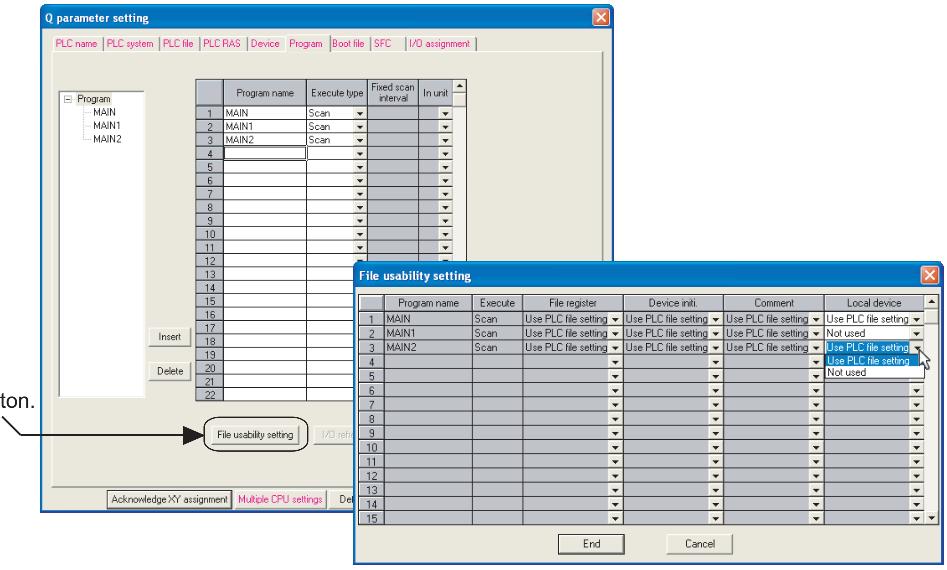

Program execution
Local device file (.QDL)
Program A
Program B
Program C
Save/Restore
Save/Restore
Save/Restore
For program A
For program C
Scan time decreases because saving/restoring process is not performed.
File size decreases because the area for saving/restoring data for program B is not created. (Only for Universal model QCPU)

Figure 10.104 Configuration of local device file restoration area

(a) Setting method

In addition to the setting in (3) in this section, set the following:
Select "File usability setting" from "Program" in "PLC parameter", and set programs that use local devices.

Click the File usability setting button.



	Program name	Execute type	Fixed scan interval	In unit
1	MAIN	Scan		
2	MAIN1	Scan		
3	MAIN2	Scan		
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				

	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 Use PLC file setting Not used
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Figure 10.105 File usability setting screen

Universal UD Note10.34

When setting local devices for each program in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

(b) Precautions

1) **Changing local device**

For programs set to "Not used" for local device, do not change or refer to the local device.

When the local device is changed by a program that does not use local device, the changed value will not be held.

2) **Creation condition of local device file**

Creation of a local device file depends on the PLC parameter setting.

Table 9.26 shows the creation condition of local device file.

Table10.29 Creation condition of 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	×	• Universal model QCPU only: PARAMETER ERROR (Error code: 3000) • Other than Universal model QCPU: FILE SET ERROR (Error code: 2400)
		Not used	×	----
	Not set	Use PLC file setting	×	----
		Not used	×	----

○: Create a file , ×: Do not create a file

* 1 : It indicates the range setting for local devices in the device setting.

(5) Using local devices used by the file where a subroutine program is stored

It is possible to use local devices that are used by the file where a subroutine program is stored when executing a subroutine program.

Whether or not such local devices are used is set by special relay (SM776) ON/OFF setting.

Table10.30 Switching over local devices by setting ON/OFF for a special relay (SM776)

SM776	Operation
OFF	Executes calculation by the local devices that are used by the file where the subroutine program was called.
ON	Executes calculation by the local devices that are used by the file where the subroutine program is stored.

(a) Operation at "SM776 : OFF"

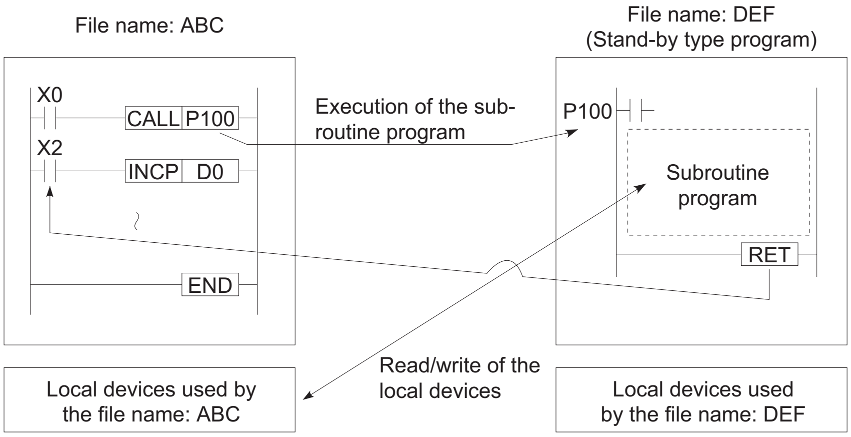

File name: ABC
File name: DEF
(Stand-by type program)
X0
CALL P100
X2
INCP D0
END
Execution of the sub-routine program
P100
Subroutine program
RET
Read/write of the local devices
Local devices used by the file name: ABC
Local devices used by the file name: DEF

Figure 10.106 When SM776 is OFF

(b) Operation at "SM776 : ON"

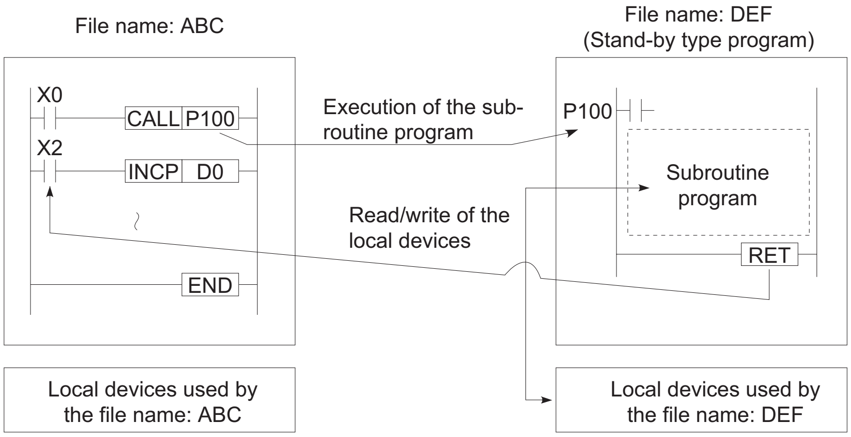

File name: ABC
File name: DEF
(Stand-by type program)
X0
CALL P100
X2
INCP D0
END
Execution of the sub-routine program
Read/write of the local devices
P100
Subroutine program
RET
Local devices used by the file name: ABC
Local devices used by the file name: DEF

Figure 10.107 When SM776 is ON

(c) Precautions

- If SM776 is ON, the local device data is read when the subroutine program is called and the local device data is saved after the execution of the RET instruction. Accordingly, scan time is elongated by the time as when a subroutine program is executed once with the setting of "SM776: ON".
- ON/OFF setting of SM776 is enabled in CPU modules.
  Setting in file units is not enabled.
- If the ON/OFF setting of SM776 is changed while a sequence program is executed, the control is made according to the information after change.

Remark

For details on SM776, refer to Appendix 1.

(6) Using local devices when executing an interrupt/fixed scan execution type program

It is possible to use local devices in the file where an interrupt/fixed scan execution type program is stored when executing an interrupt/fixed scan execution type program.

The local devices can be set available/unavailable by special relay "SM777" ON/OFF setting.

Table10.31 Switching over local devices by setting ON/OFF for a special relay (SM777)

SM777	Operation
OFF	Executes operation with the local devices in the file which was executed before the execution of the interrupt/fixed scan execution type program.
ON	Executes operation with the local devices in the file where the interrupt/fixed scan execution type program is stored.

(a) Operation at "SM777 : OFF"

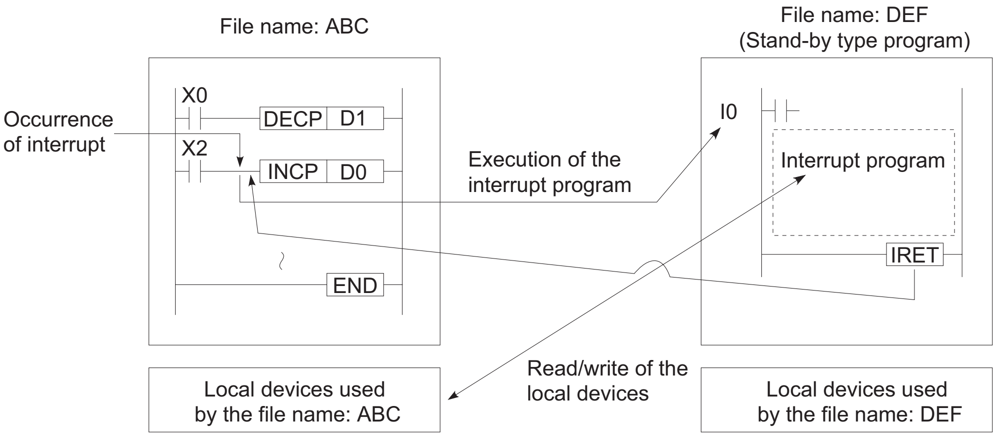


Figure 10.108 When SM777 is OFF

(b) Operation at "SM777 : ON"

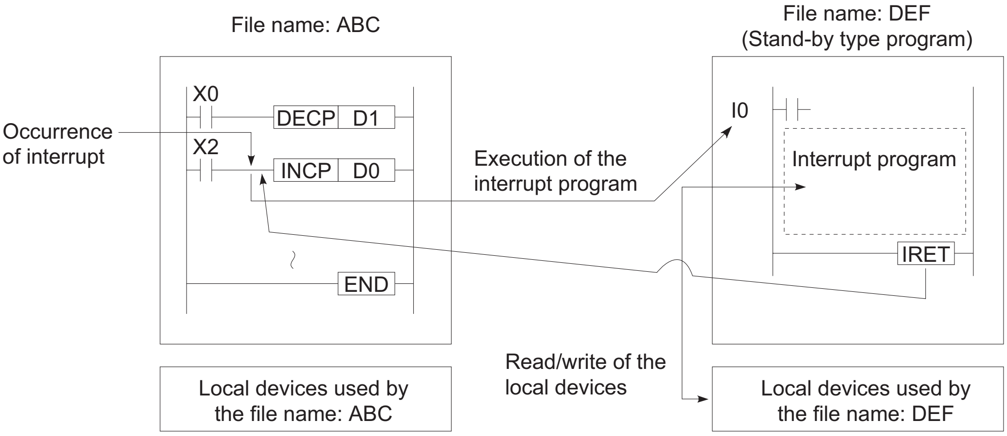

File name: ABC
File name: DEF
(Stand-by type program)
X0
DECP D1
Occurrence of interrupt
X2
INCP D0
Execution of the interrupt program
I0
Interrupt program
IRET
END
Read/write of the local devices
Local devices used by the file name: ABC
Local devices used by the file name: DEF

Figure 10.109 When SM777 is ON

(c) Precautions

- If SM777 is ON, the local device data is read before the interrupt/fixed scan execution type program is executed and the local device data is saved after the execution of the IRET instruction.
  Accordingly, scan time increases when an interrupt/fixed scan execution type program is executed once with the setting of "SM777: ON".
- ON/OFF setting of SM777 is enabled in CPU module units.
  Setting in file unit is not enabled.
- If the ON/OFF setting of SM777 is changed while a sequence program is executed, the control is made according to the information after change.

Remark

For details on SM777, refer to Appendix 1.

(7) Clearing the Local Device Data

Clear the local device data in either of the following methods.

- The PLC is powered on or the CPU module is reset.
- The CPU module enters into the RUN status from the STOP status.

The local device data cannot be cleared by operating from the GX Developer.

CHAPTER11 CPU MODULE PROCESSING TIME

This chapter explains the CPU module processing time.

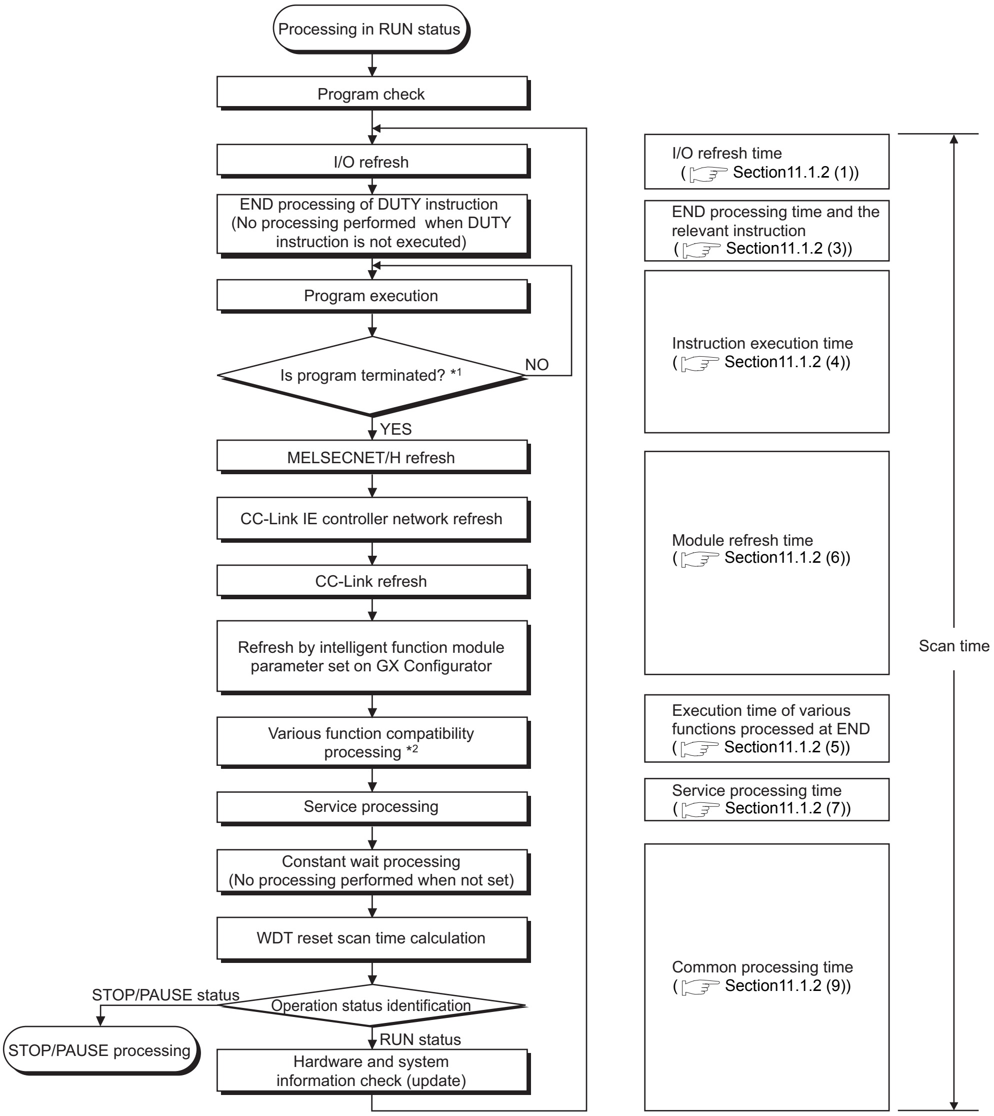
11.1 Scan Time

This section explains the scan time structures and CPU module processing time.

11.1.1 Scan time structure

The CPU module scan time consists of the followings processings.
The CPU module performs the following processings cyclically in the RUN status.

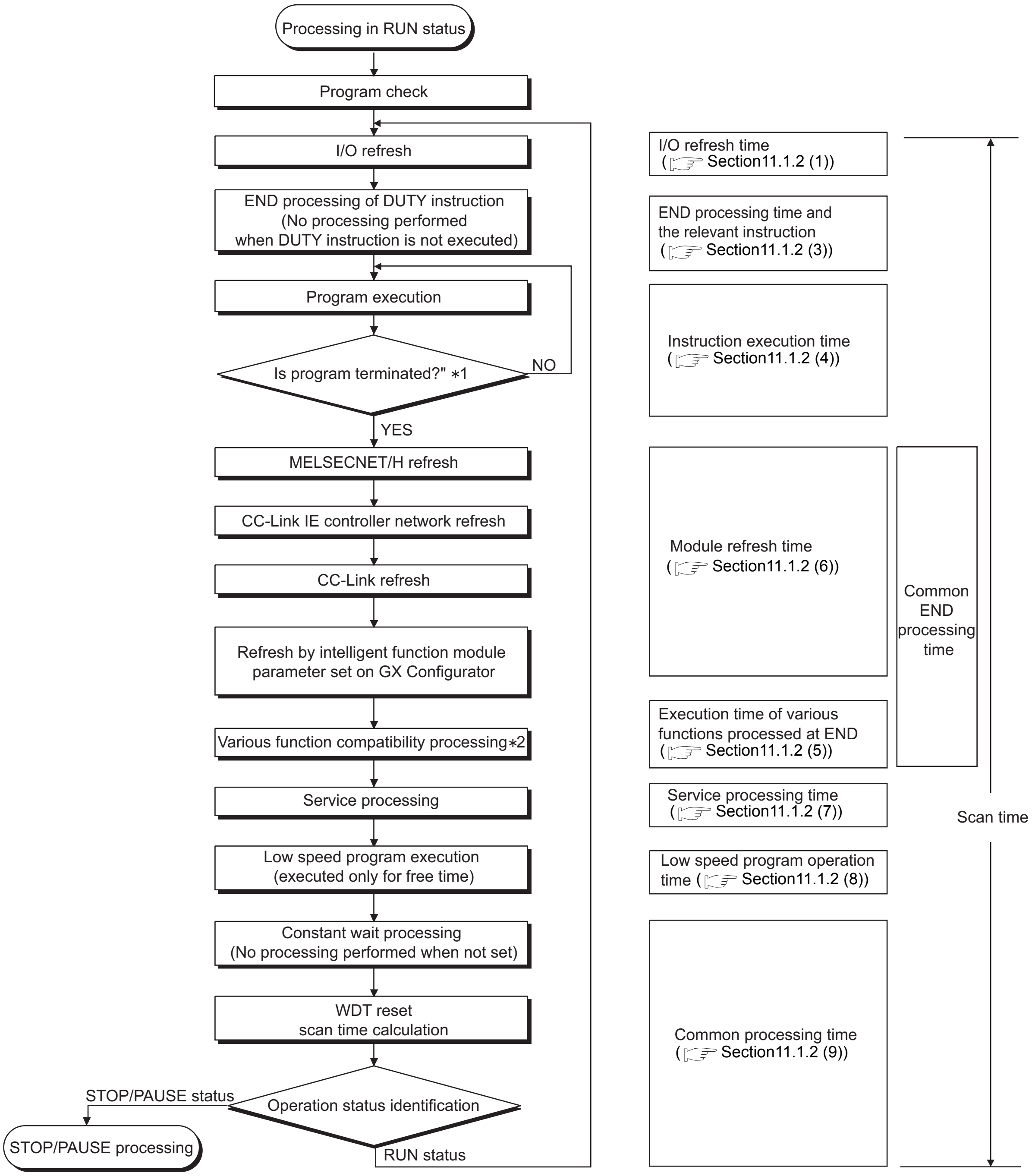
(1) Scan time structure of Basic model QCPU


Processing in RUN status
Program check
I/O refresh
END processing of DUTY instruction (No processing performed when DUTY instruction is not executed)
Program execution
Is program terminated? *1
NO
YES
MELSECNET/H refresh
CC-Link IE controller network refresh
CC-Link refresh
Refresh by intelligent function module parameter set on GX Configurator
Various function compatibility processing *2
Service processing
Constant wait processing (No processing performed when not set)
WDT reset scan time calculation
STOP/PAUSE status
Operation status identification
STOP/PAUSE processing
RUN status
Hardware and system information check (update)
I/O refresh time (Section11.1.2 (1))
END processing time and the relevant instruction (Section11.1.2 (3))
Instruction execution time (Section11.1.2 (4))
Module refresh time (Section11.1.2 (6))
Execution time of various functions processed at END (Section11.1.2 (5))
Service processing time (Section11.1.2 (7))
Common processing time (Section11.1.2 (9))
Scan time

* 1 : Program end indicates the timing when the END, GOEND, FEND or STOP instruction is executed.
* 2 : Indicates a calendar update or error clear.

Figure 11.1 Scan time structure of Basic model QCPU

(2) Scan time structure of High Performance model QCPU

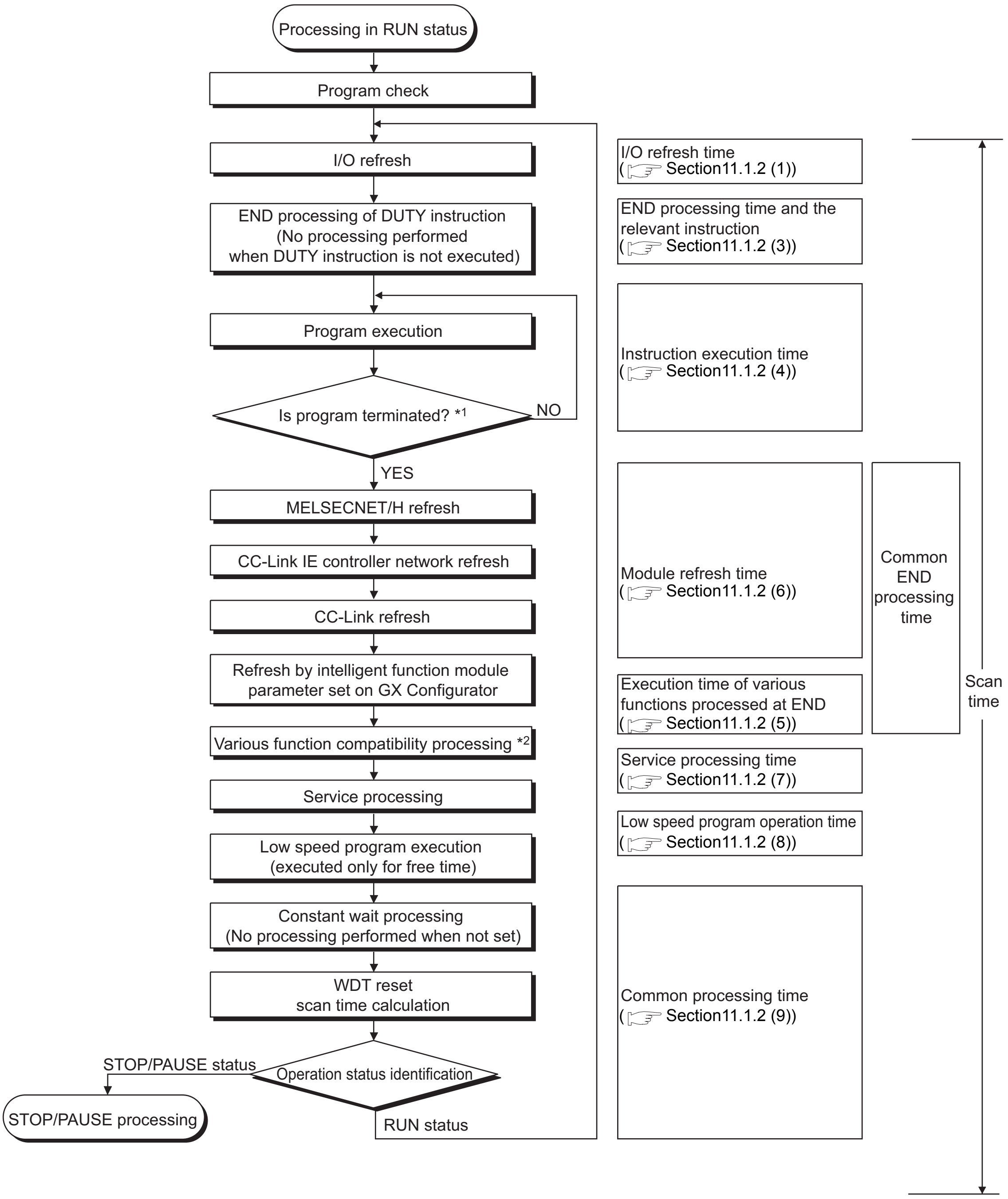

Processing in RUN status
Program check
I/O refresh
END processing of DUTY instruction (No processing performed when DUTY instruction is not executed)
Program execution
Is program terminated?" *1
NO
YES
MELSECNET/H refresh
CC-Link IE controller network refresh
CC-Link refresh
Refresh by intelligent function module parameter set on GX Configurator
Various function compatibility processing*2
Service processing
Low speed program execution (executed only for free time)
Constant wait processing (No processing performed when not set)
WDT reset scan time calculation
STOP/PAUSE status
Operation status identification
STOP/PAUSE processing
RUN status
I/O refresh time (☞ Section11.1.2 (1))
END processing time and the relevant instruction (☞ Section11.1.2 (3))
Instruction execution time (☞ Section11.1.2 (4))
Module refresh time (☞ Section11.1.2 (6))
Execution time of various functions processed at END (☞ Section11.1.2 (5))
Service processing time (☞ Section11.1.2 (7))
Low speed program operation time (☞ Section11.1.2 (8))
Common processing time (☞ Section11.1.2 (9))
Common END processing time
Scan time

* 1 : Program end indicates the timing when the END, GOEND, FEND or STOP instruction is executed.

* 2 : Indicates a calendar update or error clear.

Figure 11.2 Scan time structure of High Performance model QCPU

(3) Scan time structure of Process CPU

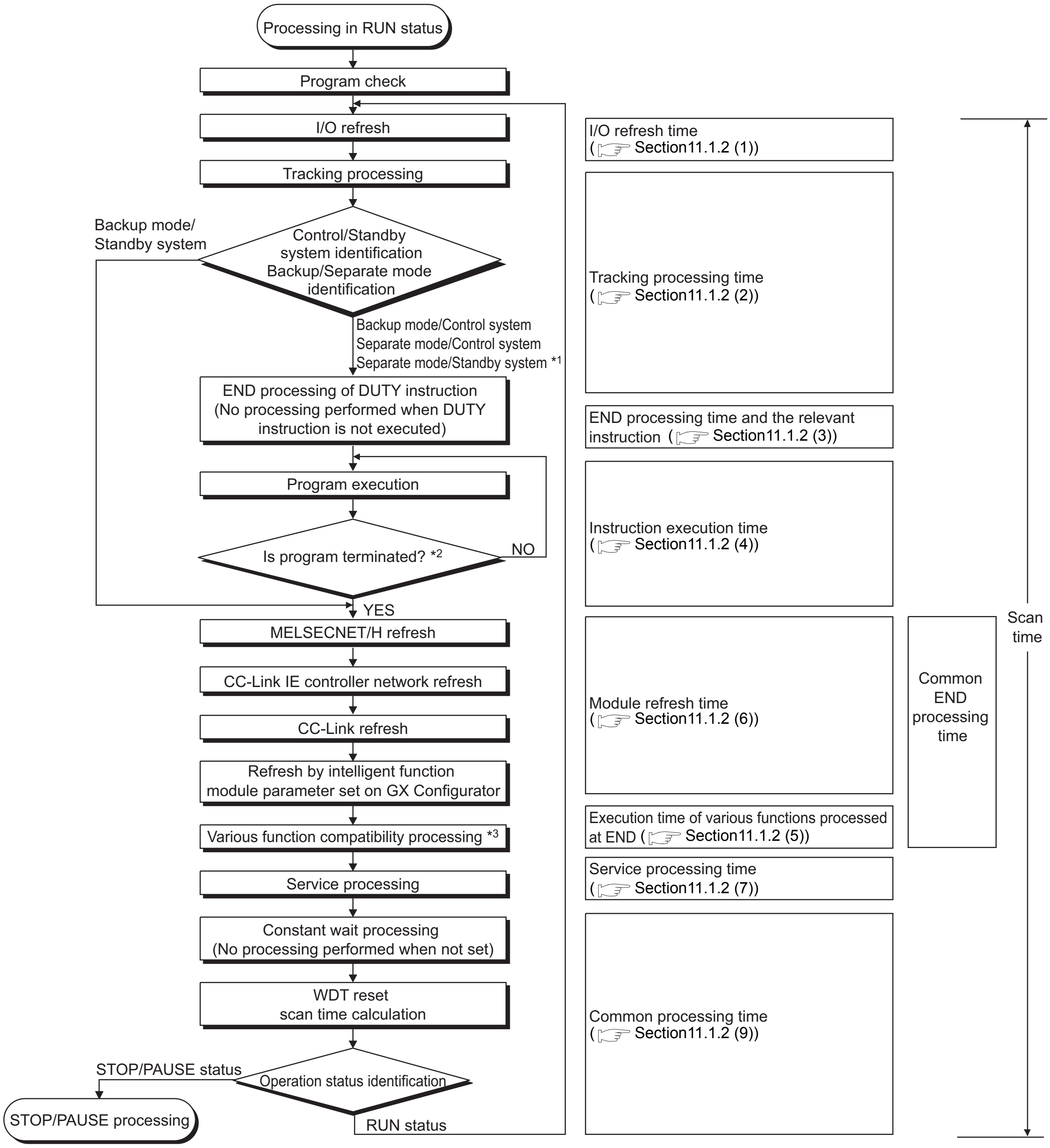

Processing in RUN status
Program check
I/O refresh
END processing of DUTY instruction (No processing performed when DUTY instruction is not executed)
Program execution
Is program terminated? *1
NO
YES
MELSECNET/H refresh
CC-Link IE controller network refresh
CC-Link refresh
Refresh by intelligent function module parameter set on GX Configurator
Various function compatibility processing *2
Service processing
Low speed program execution (executed only for free time)
Constant wait processing (No processing performed when not set)
WDT reset scan time calculation
STOP/PAUSE status
Operation status identification
STOP/PAUSE processing
RUN status
I/O refresh time (☞ Section11.1.2 (1))
END processing time and the relevant instruction (☞ Section11.1.2 (3))
Instruction execution time (☞ Section11.1.2 (4))
Module refresh time (☞ Section11.1.2 (6))
Execution time of various functions processed at END (☞ Section11.1.2 (5))
Service processing time (☞ Section11.1.2 (7))
Low speed program operation time (☞ Section11.1.2 (8))
Common processing time (☞ Section11.1.2 (9))
Common END processing time
Scan time

* 1 : Program end indicates the timing when the END, GOEND, FEND or STOP instruction is executed.

* 2 : Indicates a calendar update or error clear.

Figure 11.3 Scan time structure of Process CPU

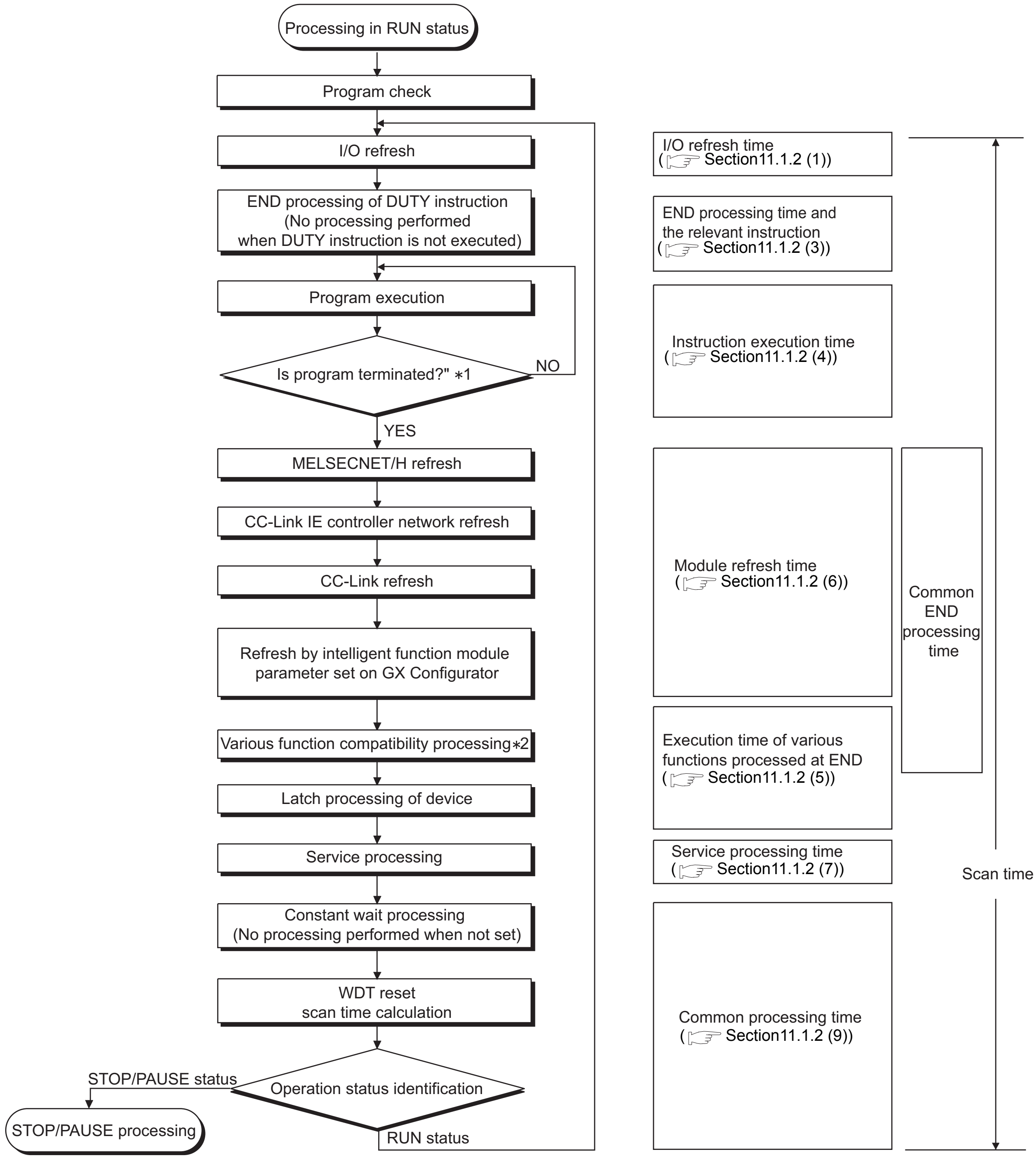
(4) Scan time structure of Redundant CPU



* 1 : Program end indicates the timing when the END, GOEND, FEND or STOP instruction is executed.
* 2 : Indicates a calendar update, program memory check processing or error clear.

Figure 11.4 Scan time structure of Redundant CPU

(5) Scan time structure of Universal model QCPU


Processing in RUN status
Program check
I/O refresh
END processing of DUTY instruction (No processing performed when DUTY instruction is not executed)
Program execution
Is program terminated?" ∗1
NO
YES
MELSECNET/H refresh
CC-Link IE controller network refresh
CC-Link refresh
Refresh by intelligent function module parameter set on GX Configurator
Various function compatibility processing∗2
Latch processing of device
Service processing
Constant wait processing (No processing performed when not set)
WDT reset scan time calculation
STOP/PAUSE status
Operation status identification
STOP/PAUSE processing
RUN status
I/O refresh time (☞ Section11.1.2 (1))
END processing time and the relevant instruction (☞ Section11.1.2 (3))
Instruction execution time (☞ Section11.1.2 (4))
Module refresh time (☞ Section11.1.2 (6))
Execution time of various functions processed at END (☞ Section11.1.2 (5))
Service processing time (☞ Section11.1.2 (7))
Common processing time (☞ Section11.1.2 (9))
Common END processing time
Scan time

* 1 : Program end indicates the timing when the END, GOEND, FEND or STOP instruction is executed.

* 2 : Indicates a calendar update, program memory check processing or error clear.

Figure 11.5 Scan time structure of Universal model QCPU

11.1.2 Time required for each processing included in scan time

The scan time is the sum of the processings and execution times shown in Section 11.1.1. This section explains how to calculate the processing and execution times shown in Section 11.1.1.

(1) I/O refresh time

The I/O refresh time is the time for refreshing I/O data to/from the following modules mounted on the main base unit and extension base units.

Basic Process Redundant
Note11.1 Note11.1 Note11.1

Universal
UD
Note11.1

- Input module
- Output module
- Intelligent function module (special function module)Note11.1

■ 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

Refer to Table11.1 for N1 and N2.

Table11.1 I/O refresh time

CPU module	N1					N2				
	Q3□B, Q3□SB, Q3□RB Q3□DB	Q5□B, Q6□B, Q6□RB	Q6□WRB	QA1S6□B	QA6□B	Q3□B, Q3□SB, Q3□RB Q3□DB	Q5□B, Q6□B, Q6□RB	Q6□WRB	QA1S6□B	QA6□B
Q00JCPU	2.05μs	2.95μs	----	----	----	1.25μs	2.20μs	----	----	----
Q00CPU	2.00μs	2.75μs	----	----	----	1.20μs	2.05μs	----	----	----
Q01CPU	1.95μs	2.70μs	----	----	----	1.15μs	2.00μs	----	----	----
Q02CPU	2.2μs	2.9μs	----	4.3μs	4.3μs	1.3μs	2.1μs	----	3.5μs	3.5μs
Q02HCPU Q06HCPU Q12HCPU Q25HCPU	1.7μs	2.4μs	----	3.7μs	3.7μs	1.3μs	2.1μs	----	3.5μs	3.5μs
Q02PHCPU Q06PHCPU Q12PHCPU Q25PHCPU	1.7μs	2.4μs	----	----	----	1.3μs	2.1μs	----	----	----
Q12PRHCPU Q25PRHCPU	1.7μs	2.4μs	2.4μs	----	----	1.3μs	2.1μs	2.1μs	----	----
Q02UCPU	1.5μs	2.3μs	----	----	----	1.0μs	1.8μs	----	----	----
Q03UDCPU Q04UDHCPU Q06UDHCPU Q13UDHCPU Q26UDHCPU Q03UDECPU Q04UDEHCPU Q06UDEHCPU Q13UDEHCPU Q26UDEHCPU	1.5μs	2.3μs	----	----	----	1.0μs	1.8μs	----	----	----

Basic Process Redundant
Note11.1 Note11.1 Note11.1

Universal
UD
Note11.1

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the AnS/A series corresponding modules.

Remark

The applicable base unit changes depending on the CPU module.

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

(2) Tracking processing time

The tracking processing time indicates the time required when using the tracking function of the Redundant CPU.

Refer to the following manual for the tracking processing time.

☞ QnPRHCPU User's Manual (Redundant System)

(3) END processing time and the relevant instruction

(a) END processing and DUTY instruction execution

Basic Note11.2

During the DUTY instruction execution, the CPU module turns ON/OFF the user dynamic block (SM420 to 424, SM430 to 434[Note11.2]) according to the number of times for END processing, i.e., number of scan times specified in that instruction. In this case, the END processing time changes with the number of times specified in the DUTY instruction.

Table11.2 END processing during DUTY instruction execution

CPU module	END processing time	
	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, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.01ms	0.01ms
Q02UCPU	0.0190ms	0.0200ms
Q03UDCPU, Q03UDECPU	0.0043ms	0.0046ms
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.0041ms	0.0045ms

(4) Instruction execution time

The instruction execution time is the sum of processing times of the instructions used in the program to be executed by the CPU module.

Refer to the following manual for the processing time of each instruction.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Note11.2

The Basic model QCPU cannot use SM430 to 434.

Basic
Note11.3

(a) Overhead time at execution of interrupt program/fixed scan execution type program Note11.3

The interrupt program/fixed scan execution type program includes overhead time. Also, the interrupt program includes pre-start overhead time and program-end overhead time.

When calculating the instruction execution time, add the following overhead time to the above instruction execution time.

Table11.3 Pre-start overhead time of interrupt program (B1)

CPU module	Fixed scan interrupt (I28 to 31) processing		Multiple CPU synchronous interrupt (I45)		Interrupt processing from QI60 (I0 to 15)[*1] /interrupt from the intelligent function module (I50 to 127)	
	Without high-speed start	With high-speed start	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, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	85μs	40μs	----	----	90μs	45μs
Q12PRHCPU, Q25PRHCPU (When extension base unit is unconnected)	85μs	40μs	----	----	90μs	45μs
Q12PRHCPU, Q25PRHCPU (When extension base unit is connected)	85μs	40μs	----	----	1090μs[*2]	1045μs[*2]
Q02UCPU	50μs	19μs	----	----	60μs	31μs
Q03UDCPU, Q03UDECPU	47μs	17μs	46μs	16μs	54μs	22μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	46μs	16μs	44μs	14μs	52μs	22μs

[*1] : Indicates the value when the QI60 is mounted on slot 0 of the main base unit.

[*2] : The QI60 cannot be used when connecting to the extension base unit. The values in table show the overhead time of processing interrupted from the intelligent function module.

Basic
Note11.3

Since the Basic model QCPU cannot use fixed scan execution type programs, it is not necessary to be conscious of the overhead time of the fixed scan execution type program.

Table11.4 Program-end overhead time of interrupt program (B2)

CPU module	Without high-speed start	With high-speed start
Q00JCPU	175 µs	150 µs
Q00CPU	145 µs	125 µs
Q01CPU	135 µs	120 µs
Q02CPU	180 µs	75 µs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	80 µs	35 µs
Q02UCPU	26 µs	7 µs
Q03UDCPU, Q03UDECPU	26 µs	7 µs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	26 µs	7 µs

Basic
Note11.3

Table11.5 Overhead time of fixed scan execution type program Note11.3

CPU module	Without high-speed start	With high-speed start
Q02CPU	380 µs	230 µs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	165 µs	100 µs
Q02UCPU	73 µs	25 µs
Q03UDCPU, Q03UDECPU	73 µs	24 µs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	72 µs	23 µs

Basic
Note11.3

Since the Basic model QCPU cannot use fixed scan execution type programs, it is not necessary to be conscious of the overhead time of the fixed scan execution type program.

Basic
Note11.4

1) Overhead time taken when local devices in interrupt program are made available[Note11.4]

When SM777 (setting of whether local devices in interrupt program are enabled or disabled) is turned ON to make the local devices in the interrupt program available, the following time is added to the overhead time in Table11.6 and Table11.7.

The definition of n, N1, N2, and N3 in the following table is as follows:

- n: The number of local device points (Unit: k word)
- N1: The number of device types that specifed a local device
- N2: The number of word device points that specifed a local device
- N3: The number of bit device points that specified a local device

Table11.6 When local device file in standard RAM is used

CPU module	Value added to pre-start overhead time of interrupt program (Table11.3)	Value added to program-end overhead time of interrupt program (Table11.4)
Q02CPU	$(0.35 \times n + 0.05) \times 10^3$ μs	$(0.35 \times n + 0.05) \times 10^3$ μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	$(0.15 \times n + 0.03) \times 10^3$ μs	$(0.15 \times n + 0.03) \times 10^3$ μs
Q02UCPU	$(12 \times N1) + (0.22 \times (N2 + (N3 \div 16))) + 220$ μs	$(12 \times N1) + (0.22 \times (N2 + (N3 \div 16))) + 60$ μs
Q03UDCPU, Q03UDECPU	$(8 \times N1) + (0.22 \times (N2 + (N3 \div 16))) + 80$ μs	$(8 \times N1) + (0.22 \times (N2 + (N3 \div 16))) + 20$ μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	$(8 \times N1) + (0.10 \times (N2 + (N3 \div 16))) + 80$ μs	$(8 \times N1) + (0.10 \times (N2 + (N3 \div 16))) + 20$ μs

Table11.7 When local device file in SRAM card is used

CPU module	Value added to pre-start overhead time of interrupt program (Table11.3)	Value added to program-end overhead time of interrupt program (Table11.4)
Q02CPU	$(1.15 \times n + 0.30) \times 10^3$ μs	$(1.15 \times n + 0.30) \times 10^3$ μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	$(0.85 \times n + 0.15) \times 10^3$ μs	$(0.85 \times n + 0.15) \times 10^3$ μs
Q02UCPU	$(16 \times N1) + (0.43 \times (N2 + (N3 \div 16))) + 260$ μs	$(16 \times N1) + (0.43 \times (N2 + (N3 \div 16))) + 60$ μs
Q03UDCPU, Q03UDECPU	$(12 \times N1) + (0.43 \times (N2 + (N3 \div 16))) + 100$ μs	$(12 \times N1) + (0.43 \times (N2 + (N3 \div 16))) + 20$ μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	$(12 \times N1) + (0.40 \times (N2 + (N3 \div 16))) + 100$ μs	$(12 \times N1) + (0.40 \times (N2 + (N3 \div 16))) + 20$ μs

Basic
Note11.4

Since the Basic model QCPU cannot execute multiple programs, it is irrelevant to the difference between global devices and local devices (☞Section 10.14.1).
It is not necessary to be conscious of the overhead time of the fixed scan execution type program.

(5) Execution time of various functions processed at END

The execution time of various functions processed at END is the sum of times required for calendar update, program memory check processing and error clear.

(a) Calendar update processing time

This indicates the time taken to change/read the clock data at END processing when the clock data set request (SM210 changes from OFF to ON) or the clock data read request (SM213 turns ON) is issued.

Table11.8 Calendar update processing time

CPU module	END processing time	
	When clock data set request is issued	When clock data read request is issued
Q00JCPU	1.25 ms	0.04 ms
Q00CPU	0.99 ms	0.03 ms
Q01CPU	0.98 ms	0.02 ms
Q02CPU	0.26 ms	0.01 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.11 ms	0.005 ms
Q02UCPU	0.044 ms	0.017 ms
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.011 ms	0.004 ms

(b) Error clear processing

This indicates the time taken to clear the continuation error stored in SD50 when SM50 (error clear) rises (changes from OFF to ON).

Table11.9 Error clear processing time

CPU module	Common processing time	
	Annunciator	Other error
Q00JCPU	2.1 ms	2.0 ms
Q00CPU	1.75 ms	1.7 ms
Q01CPU	1.45 ms	1.35 ms
Q02CPU	1.15 ms	0.41 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.84 ms	0.21 ms
Q02UCPU	0.180 ms	0.175 ms
Q03UDCPU, Q03UDECPU	0.068 ms	0.062 ms
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.065 ms	0.062 ms

Basic Note11.5 High Performance Note11.5

Universal Note11.6

(c) **Program memory check processing time** Note11.5, Note11.6

This indicates the time taken to make a program memory check when it has been set in the PLC parameter dialog box of GX Developer.

Time required for program memory check

$$= \frac{\text{Check capacity per scan (step)}^{*1}}{1024} \times 3.5 \text{ ms}$$

* 1 : Indicates the step set in the PLC RAS setting (2) of the PLC parameter dialog box. (☞ Section 9.1.2(5))

Basic Note11.5 High Performance Note11.5

Since the Basic model QCPU, and High Performance model QCPU do not support the setting for program memory check, it is not necessary to be conscious of this processing time.

Universal Note11.6

Since the Universal model QCPU does not take processing time of memory check, it can be ignored.

(6) Module refresh time

The module refresh time is the sum of refresh times taken by MELSECNET/H, CC-Link and others.

(a) Refresh of CC-Link IE controller network

This indicates the time taken to refresh data between the link devices of the CC-Link IE controller network module and those of the CPU module.

Refer to the following manual for the refresh time of CC-Link IE controller network.

☞ CC-Link IE Controller Network Reference Manual

(b) Refresh of MELSECNET/H

This indicates the time taken to refresh data between the link devices of the MELSECNET/H network and those of the CPU module.

Refer to the following manual for the refresh time of MELSECNET/H.

☞ Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

☞ Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

(c) Auto refresh of CC-Link

This indicates the time taken to refresh data between the CC-Link master/local module and CPU module.

Refer to the following manual for the auto refresh time of CC-Link.

☞ CC-Link System Master/Local Module User's Manual (Details)

(d) Auto refresh of intelligent function module

This indicates the time taken to refresh data between the intelligent function module buffer memory and CPU module devices.
Use the intelligent function module utility package (GX Configurator) to make auto refresh settings.

■ Calculation method

Calculate the intelligent auto refresh time with the following expression.

(Refresh time) = KN1 + KN2 × (number of refresh points)

Use the following values in Table11.10 for KN1 and KN2.

Table11.10 When intelligent function module is mounted on main base unit

CPU module	KN1	KN2
Q00JCPU	115 μs	55 μs
Q00CPU	91 μs	46 μs
Q01CPU	85 μs	41 μs
Q02CPU	53 μs	13 μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	27 μs	6 μs
Q02UCPU	23 μs	6 μs
Q03UDCPU, Q03UDECPU	6 μs	5 μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	4 μs	5 μs

Table11.11 When intelligent function module is mounted on extension base unit

CPU module	KN1	KN2
Q00JCPU	120 μs	56 μs
Q00CPU	92 μs	48 μs
Q01CPU	86 μs	43 μs
Q02CPU	61 μs	15 μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	29 μs	8 μs
Q02UCPU	45 μs	7 μs
Q03UDCPU, Q03UDECPU	7 μs	6 μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	5 μs	6 μs

(Example)

When the number of auto refresh points for the analog-digital conveter module (Q64AD) is 4 points (when the module is mounted on the Q01CPU main base unit)

0.249(ms) = 0.085 + 0.041 × 4

(7) Service processing time

Service processing indicates the processing for communication with GX Developer and external device.

(a) Basic model QCPU

Table 11.12 shows the processing time at monitoring data by GX Developer.

Table11.12 Monitor processing time by GX Developer

Function	When connected to RS-232 of host station CPU module			When connected to other station[*4]		
	Q00JCPU	Q00CPU	Q01CPU	Q00JCPU	Q00CPU	Q01CPU
Read of program from PLC[*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 taken to read an 8k-step program from the program memory.

* 2 : Time taken when 32 points have been set in registration monitor.

* 3 : Time taken when a 100-step ladder is added.

* 4 : Time taken when access is made via the MELSECNET/H, Ethernet, CC-Link or serial communication module.

- Communication with serial communication module or Ethernet interface module

  This indicates the time taken to communicate with the serial communication module or Ethernet module.

  Refer to the following manual for the time taken for communication with each module.

  ☞ Q Corresponding MELSECNET/H Network System Reference Manual

(b) **High Performance model QCPU, Process CPU, Redundant CPU**

Tables 11.13 and 11.14 show the processing time at monitoring data or setting the monitor condition by GX Developer.

Table11.13 When 32 points of data registers have been set in device registration monitor

CPU module	Processing time
Q02CPU	0.017ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.011ms

Table11.14 When monitor condition has been set

CPU module	Processing time	
	Specified step match	Specified device match
Q02CPU	0.05ms	0.01ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.03ms	0.01ms

(c) **Universal model QCPU**

Table 11.15 and 11.16 shows the processing time for monitoring data, reading programs, and setting monitor conditions in GX Developer.

Table11.15 shows the processing time at monitoring data or reading a program by GX Developer.

CPU module	Processing time [*1]	
	Registration monitor (Data register 32 points)	Program read (10K step)
Q02UCPU	0.65ms	1.60ms
Q03UDCPU, Q03UDECPU	0.35ms	1.10ms
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.35ms	1.10ms

* 1: It is the processing time when the service process execution count is set to 1 time.

Table11.16 When monitor condition has been set

CPU module	Processing time	
	Specified step match	Specified device match
Q02UCPU	0.03ms	0.04ms
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.01ms	0.03ms

Basic Redundant Universal

Note11.7

(8) Low speed program operation time[Note11.7]

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.

The processing times are the same as those in (4) of this section.

(9) Common processing time

This indicates the processing time common to the CPU modules.

Table11.17 shows the common processing time for each CPU module model.

Table11.17 Common processing time

CPU module	Common processing time
Q00JCPU	0.66ms
Q00CPU	0.60ms
Q01CPU	0.52ms
Q02CPU	0.40ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	0.16ms
Q12PRHCPU, Q25PRHCPU	0.5ms
Q02UCPU	0.17ms
Q03UDCPU	0.12ms
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU	0.09ms
Q03UDECPU	0.21ms
Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.17ms

(10)Multiple CPU high speed transmission processing time

The Multiple CPU high speed transmission processing time is the required time for Multiple CPU data transmission when the Multiple CPU high speed transmission function is used.

Refer to the following manual for the Multiple CPU high speed transmission processing time.

☞QCPU User's Manual (Multiple CPU System)

Basic Redundant Universal

Note11.7

Since the Basic model QCPU, Redundant CPU and Universal model QCPU cannot use low speed execution type programs, it is not necessary to be conscious of the low speed execution type program operation time.

Basic High Performance Process
Note11.8 Note11.8 Note11.8

Redundant
Note11.8

(11) Latch processing time of device data Note11.8

The scan time will be prolonged by the time shown below when the latch range is set in the PLC parameter device setting.*1*2*3

The definition of N1, N2, and N3 in the following table is as follows:

- N1: The number of device types specified to be latched (The latch range (1) and the latch range (2) are counted as different device types.)
- N2: The number of bit devices specified to be latched
- N3: the number of word devices specified to be latched

Table11.18 Latch processing prolonging time

CPU module	Latch processing prolonging time
Q02UCPU	$(4.0 \times N1) + (0.12 \times (N2 \div 16 + N3))\mu s$
Q03UDCPU, Q03UDECPU	$(3.0 \times N1) + (0.12 \times (N2 \div 16 + N3))\mu s$
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	$(3.0 \times N1) + (0.05 \times (N2 \div 16 + N3))\mu s$

* 1: When setting the latch range of timer (T), retentive timer (ST), and counter (C), word device one point and bit device two points are occupied.
* 2: Number of points setting for latch relay (L) is included.
* 3: The scan time is not prolonged even if the latch range of the file register (R, ZR), extended data register (D), and extended link register (W) are set.

☒POINT

Minimize the number of latch points to reduce the scan time.

The following alternative methods can also reduce the number of latch points.

- Move data to be latched to a file register.
- Store the device data that is updated not so often to the standard ROM using the SP.DEVST instruction. (The device data stored in the standard ROM can be read with the S(P).DEVLD instruction.)

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic High Performance Process
Note11.8 Note11.8 Note11.8

Redundant
Note11.8

Since the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not perform the processing for latch, Section 11.1.2(11) can be ignored.

11.1.3 Factors that increase the scan time

When the following functions or operations are performed, this will increase the scan time of the CPU module.
When executing any of them, make sure to allow for the processing time (the value given in this section to the value calculated in Section 11.1.2).

Basic Note11.9

(1) Sampling trace[Note11.9]

When sampling trace (☞ Section 6.14) is executed, the following processing time shown in Table11.19 will be required.

Table11.19 Processing time taken when 50 points of internal relays (bit devices) and 50 points of data registers (word devices) are set as sampling trace data

	CPU module	Processing time
Standard RAM	Q02CPU	0.16ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.06ms
	Q02UCPU	0.15ms
	Q03UDCPU, Q03UDECPU	0.07ms
	Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.06ms
SRAM card	Q02CPU	0.24ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.12ms
	Q02UCPU	0.16ms
	Q03UDCPU, Q03UDECPU	0.08ms
	Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.06ms

Basic Note11.9

Since the Basic model QCPU cannot use sampling trace, it is not necessary to be conscious of the sampling trace processing time.

Basic
Note11.10

(2) Use of local devicesNote11.10

When local devices are used, the following processing time shown in Table11.19 is required.

The definition of n, N1, N2, and N3 in the following table is as follows:

- n: The number of programs using a local device[*1]
- N1: The number of device types that specified a local device
- N2: The number of word device points that specifed a local device
- N3: The number of bit device points that specified a local device

Table11.20 Local device setting

CPU module		Processing time
Standard RAM	Q02CPU	3.52 × n × 10^3 μs [*2]
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	1.54 × n × 10^3 μs [*2]
	Q02UCPU	((12 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 140) × n μs
	Q03UDCPU, Q03UDECPU	((8 × N1) + (0.22 × (N2 + (N3 ÷ 16))) + 50) × n μs
	Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	((8 × N1) + (0.10 × (N2 + (N3 ÷ 16))) + 50) × n μs
SRAM card	Q02CPU	10.73 × n × 10^3 μs [*2]
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	8.16 × n × 10^3 μs [*2]
	Q02UCPU	((16 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + 160) × n μs
	Q03UDCPU, Q03UDECPU	((12 × N1) + (0.43 × (N2 + (N3 ÷ 16))) + 60) × n μs
	Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	((12 × N1) + (0.40 × (N2 + (N3 ÷ 16))) + 60) × n μs

[*1] For the High performance model QCPU, Process QCPU, Redundant CPU, and the Universal model QCPU whose serial number (first 5 digits) is "10011" or earlier, "n" indicates the number of executing programs.

[*2] It is the processing time when the number of local device points is 10k.

Basic
Note11.10

Since the Basic model QCPU cannot use local devices, it is not necessary to be conscious of the processing time described on this page.

(a) Scan time increased when local devices in subroutine program are made available

When SM776 (setting of whether local devices are usable or not when the subroutine program is called) is turned ON to make the local devices in the subroutine program available, the scan time increases per call by the time indicated in Table11.21 and Table11.22.

The definition of n, N1, N2, and N3 in the following table is as follows:

- n: The number of local device points (Unit: k word)
- N1: The number of device points that specified a local device
- N2: The number of word device points that specified a local device
- N3: The number of bit device points that specified a local device

Table11.21 When local device file in standard RAM is used

CPU module	Increase of scan time when a subroutine in the same file is called	Increase of scan time when a subroutine in another file is called
Q02CPU	$(0.35 \times n + 0.05) \times 10^3$ μs	$(0.70 \times n + 0.10) \times 10^3$ μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	$(0.15 \times n + 0.03) \times 10^3$ μs	$(0.30 \times n + 0.05) \times 10^3$ μs
Q02UCPU	0.00 μs	$(24 \times N1) + (0.44 \times (N2 + (N3 \div 16))) + 280$ μs
Q03UDCPU, Q03UDECPU	0.00 μs	$(16 \times N1) + (0.44 \times (N2 + (N3 \div 16))) + 100$ μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.00 μs	$(16 \times N1) + (0.20 \times (N2 + (N3 \div 16))) + 100$ μs

Table11.22 When local device file in SRAM card is used

CPU module	Increase of scan time when a subroutine in the same file is called	Increase of scan time when a subroutine in another file is called
Q02CPU	$(1.15 \times n + 0.30) \times 10^3$ μs	$(2.30 \times n + 0.60) \times 10^3$ μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	$(0.85 \times n + 0.03) \times 10^3$ μs	$(1.65 \times n + 0.30) \times 10^3$ μs
Q02UCPU	0.00 μs	$(32 \times N1) + (0.86 \times (N2 + (N3 \div 16))) + 320$ μs
Q03UDCPU, Q03UDECPU	0.00 μs	$(24 \times N1) + (0.86 \times (N2 + (N3 \div 16))) + 120$ μs
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.00 μs	$(24 \times N1) + (0.80 \times (N2 + (N3 \div 16))) + 120$ μs

Basic
Note11.11

(3) Overheard time taken to execute multiple programsNote11.11

This indicates the overhead time taken to execute multiple programs by the CPU module.

When multiple programs are executed, the following processing time shown in Table11.23 is required.

Table11.23 Overhead time taken to execute multiple programs

CPU module	Processing time
Q02CPU	0.08 × n ms*1
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.03 × n ms*1
Q02UCPU	0.04 × n ms*1
Q03UDCPU, Q03UDECPU	0.02 × n ms*1
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.02 × n ms*1

* 1 : n indicates the number of program files.

Basic
Note11.11

(4) Removal/setting of memory cardNote11.11

When the memory card is removed/set, the following processing time shown in Table11.24 is required.

This processing time arises during only one scan when the memory card is removed/set.

Table11.24 Processing time taken for removal/setting of memory card

CPU module	Processing time	
	When memory card is inserted	When memory card is removed
Q02CPU	0.16ms	0.10ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.08ms	0.04ms
Q02UCPU	0.7ms	0.2ms
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.6ms	0.1ms

Basic
Note11.11

Since the Basic model QCPU cannot execute multiple programs and use the memory card, it is not necessary to be conscious of the processing time described on this page.

(5) Use of file registers

When a file register is used, the processing time shown in Table 11.25 occurs when [Use the same file name as the program.] is selected in the parameter setting. When [Use the following file.] is selected, the scan time will not be prolonged.

Table11.25 Processing time taken when file registers are used

CPU module		Processing time
Standard RAM	Q02CPU	1.03ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.41ms
	Q02UCPU	0.082ms
	Q03UDCPU, Q03UDECPU	0.043ms
	Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.041ms
SRAM card	Q02CPU	1.14 × n ms[*1]
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	0.50 × n ms[*1]
	Q02UCPU	0.11 × n ms[*1]
	Q03UDCPU, Q03UDECPU	0.06 × n ms[*1]
	Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.06 × n ms[*1]

[* 1] : n indicates the number of program files.

(6) Execution of online change

When online program change is executed, the following processing time is required.

(a) Online change executed in ladder mode

The scan time increases by the value indicated in Table 11.23 after online change.

Table11.26 Increased time when online change is in ladder mode

CPU module model name	Allocate memory for online change	
	No change	Re-setting
Q00JCPU	Max. 2.1ms	Max. 30ms
Q00CPU	Max. 1.7ms	Max. 26ms
Q01CPU	Max. 1.7ms	Max. 36ms
Q02CPU	Max. 4ms (Max. 10ms*1)	Max. 30ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	Max. 2ms (Max. 4ms*1)	Max. 90ms
Q02UCPU	Max. 1.2ms	Max. 1.2ms
Q03UDCPU, Q03UDECPU	Max. 1.0ms	Max. 1.0ms
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	Max. 0.7ms	Max. 0.7ms

* *1: Shows the value when the local pointer is in a writing range for online change.

Basic
Note11.12

(b) Online batch file change Note11.12

Table11.27 Increased time when online batch file change is executed

CPU module model name	When free area is secured in program memory	When free area is secured in memory card (except ATA card)
Q02CPU	Max. 90ms (Max. 200ms*2)	Max. 150ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	Max. 350ms (Max. 650ms*2)	Max. 650ms
Q02UCPU	Max. 5ms	*3
Q03UDCPU, Q03UDECPU	Max. 4ms	*3
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	Max. 3ms	*3

* 2: Shows the value for SFC program.
* 3: The Universal model QCPU can write data to a file during RUN regardless of empty capacities of a program memory and a memory card.

When the ATA card is used, the scan time increases 1.25s at 30k steps.

Basic
Note11.12

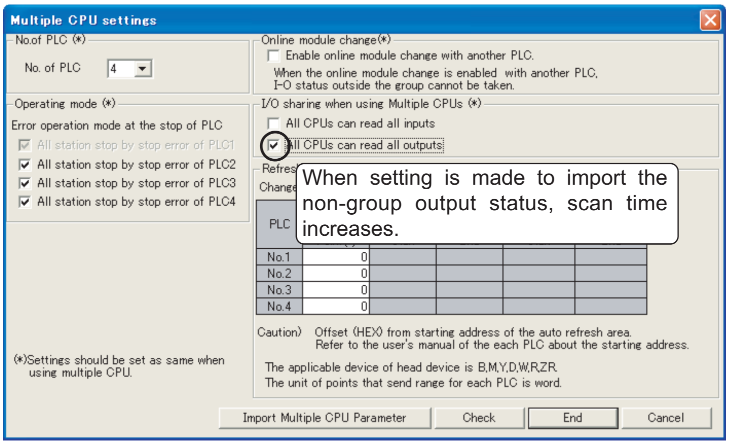
Since the Basic model QCPU cannot execute file-write in RUN. Therefore, it is not necessary to consider the increased time at the file-write in RUN.

Redundant

Note11.13

(7) Importing the non-group output statusNote11.13

In a multiple CPU system configuration, the scan time increases when settings are made to import the non-group output status in the multiple CPU setting of GX Developer.



Multiple CPU settings

No.of PLC (*)

No. of PLC: 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.

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

I/O sharing when using Multiple CPUs (*)

☐ All CPUs can read all inputs

☑ All CPUs can read all outputs

Refres

Change

PLC					
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 that send range for each PLC is word.

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

When setting is made to import the non-group output status, scan time increases.

Figure 11.6 Multiple CPU setting screen

(8) Functions that increase scan time only when Basic model QCPU is used

When the Basic model QCPU is used, the scan time also increases by use of the following functions.

- System monitor
- Battery check
- Fuse blown check
- Module verify
- General data processing

The following table indicates the scan time increased when system monitor is executed with a total of 12 intelligent function modules mounted on one extension base unit.

Table11.28 Scan time increased when system monitor is used (When a total of 12 intelligent function modules are mounted)

CPU module model name	Scan time increase
Q00JCPU	0.036ms
Q00CPU	0.015ms
Q01CPU	0.011ms

Redundant

Note11.13

Since the Redundant CPU is inapplicable to a multiple CPU system, it is not necessary to be conscious of the non-group output status import processing time.

Basic Note11.14 High Performance Note11.14

(9) Execution of the memory check function[Note11.14]

When the memory check function is executed on a CPU module, the scan time increases by the processing time of the memory check function. For the processing time, refer to Section 6.28.

Basic Note11.15

(10)Scan time measurement[Note11.15]

When the scan time measurement of the CPU module is executed by GX Developer (☞Section 6.13), the following processing time shown in Table 11.26 is required.

Table11.29 Processing time taken to execute scan time measurement of CPU by GX Developer

CPU Module	Processing Time
Q02CPU	120.0+16.5×number of branch instructions μs*1
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU	52.0+7.0×number of branch instructions μs*1
Q02UCPU	112.0+4.0×number of branch instructions μs*1
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	40.0+3.0×number of branch instructions μs*1

* 1 : The number of the branch instructions is a total of instructions described below which is 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

Basic Note11.14 High Performance Note11.14

Since the Basic model QCPU and High Performance model QCPU do not have the memory check function, it is not necessary to be conscious of the processing time of the memory check function.

Basic Note11.15

Since the Basic model QCPU does not provide the scan time measurement, it is not necessary to be conscious of the scan time measurement processing time.

Basic High Performance Process
Note11.16 Note11.16 Note11.16
Redundant
Note11.16

(11) Program memory batch transfer[Note11.16]

When executing the program memory batch transfer by GX Developer, the processing time shown in Table11.30 is generated.

Table11.30 Scan time prolonging time when transferring data from program cache memory to program memory

CPU Module	Scan time prolonging time
Q02UCPU	4.2ms
Q03UDCPU, Q03UDECPU	4.0ms
Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	3.8ms

Basic
Note11.17
High Performance Process Redundant
Note11.18 Note11.18 Note11.18
Universal
Note11.19

(12) Redundant power supply system diagnostics

Note11.17,Note11.18,Note11.19

When the "Diagnose redundant power supply system" checkbox is checked on the PLC RAS tab of PLC parameter in GX Developer, processing time shown in Table11.31 is required.

Table11.31 Scan time increased when redundant power supply system diagnostics is set

CPU module	Processing time	
	With power supply error[*1]	Without power supply error[*1]
Q02UCPU	0.09ms	0.09ms
Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.043ms	0.052ms

[*1] : Power supply error indicates any of the following.
- Redundant power supply module failure
- Power supply for the redundant power supply module OFF
- Redundant power supply module not mounted

Basic High Performance Process
Note11.16 Note11.16 Note11.16
Redundant
Note11.16

Since the Basic model QCPU, High Performance model QCPU, Process CPU, and Redundant CPU do not use the program cache memory, processing time required for transferring data from a program cache memory to a program memory can be ignored.

Basic
Note11.17

Since the Basic model QCPU does not execute the redundant power supply system diagnostics, it is not necessary to be conscious of the processing time.

High Performance Process Redundant
Note11.18 Note11.18 Note11.18

It is not necessary to be conscious of the processing time during the redundant power supply system diagnostics in the High Performance model QCPU, Process CPU, and Redundant CPU.

Universal
Note11.19

When setting the redundant power supply system diagnostics in the Universal model QCPU, check the versions of CPU module and GX Developer. (☞ Appendix 4.5)

11.1.4 Factors that can shorten scan time by changing the settings

The scan time can be shortened by changing the PLC parameter settings of GX Developer described in this section.

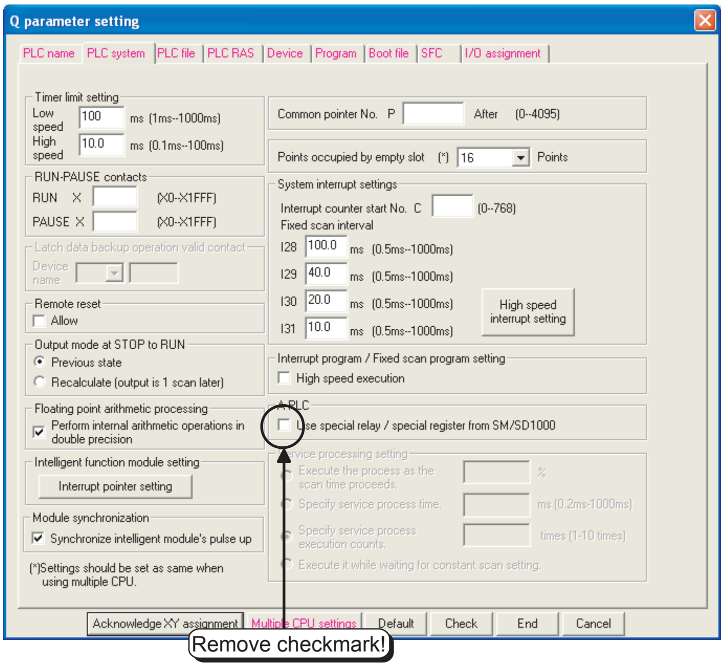
Basic Redundant Universal
Note11.20 Note11.20 Note11.20

(1) A series CPU compatibility setting (☞ Section 9.1.2(2))Note11.20

The scan time can be shortened by the processing time indicated in Table11.32 by selecting "Do not use special relays/special registers SM1000/SD1000 and later" (removing the check mark on the setting screen) in the PLC system setting.

Table11.32 Processing time for each A series CPU module model

CPU module	Processing time
Q02CPU	0.07ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	0.03ms


Q parameter setting
PLC name | PLC system | PLC file | PLC RAS | Device | Program | Boot file | SFC | I/O assignment
Interrupt pointer setting
High speed interrupt setting
Acknowledge XY assignment | Multiple CPU settings | Default | Check | End | Cancel
Remove checkmark!

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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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.

Figure 11.7 A series CPU compatibility setting

☒POINT

After selecting "Do not use special relays/special registers SM1000/SD1000 and later" (removing the check mark on the setting screen), replace the following A series compatible special relays with the Q series dedicated special relays.

- A series compatible special relays (SM1000 to 1299)
  → Q series dedicated special relays (SM0 to 999)
- A series compatible special registers (SD1000 to 1299)
  → Q series dedicated special registers (SD0 to 999)

Basic Redundant Universal
Note11.20 Note11.20 Note11.20

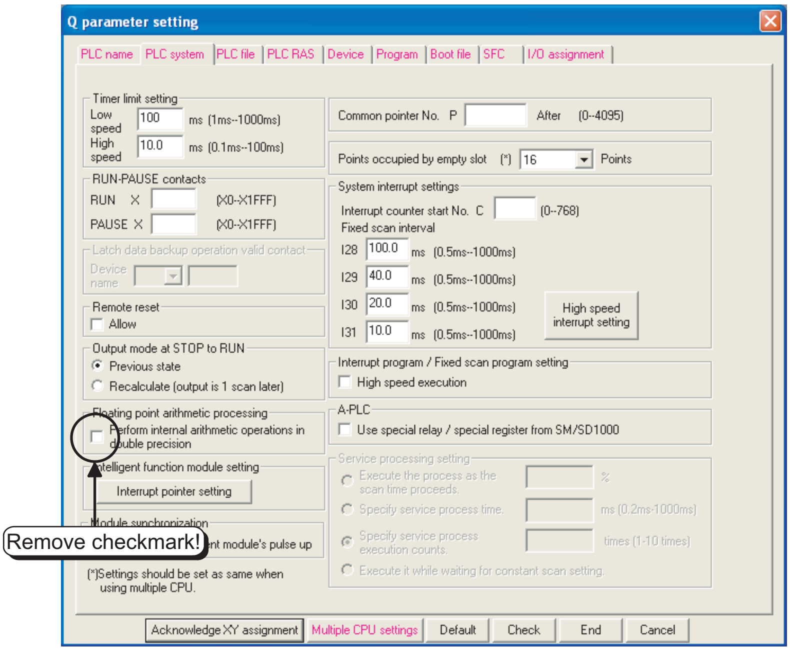
The Basic model QCPU, Redundant CPU and Universal model QCPU do not support the A series CPU compatibility setting.

Basic Process Redundant
Note11.21 Note11.21 Note11.21

Universal
UD
Note11.21

(2) Floating point arithmetic processing (☞ Section 9.1.2(2))Note11.21

The time required for the arthmetic processing of the instruction using a floating point can be shortened by selecting "Do not perform internal operation processing with double precision" (remove the check mark on the setting screen) in the PLC system 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

Module synchronization
…nt 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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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.

Figure 11.8 Floating point arithmetic processing screen

Remark

Refer to the following manual for the arithmetic processing time of the instruction using a floating point.

☞ QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)

Basic Process Redundant
Note11.21 Note11.21 Note11.21

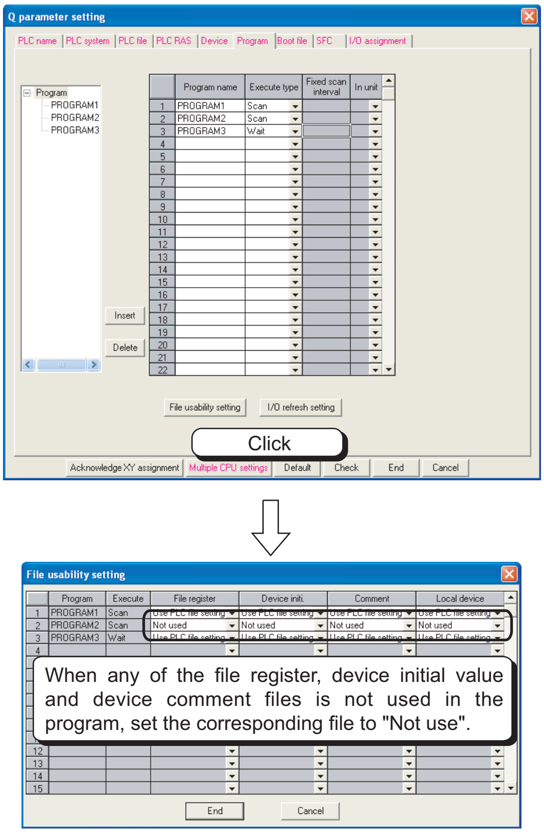
Universal
UD
Note11.21

The Basic model QCPU, Process CPU, Redundant CPU and Universal model QCPU do not support the floating-point arithmetic processing setting.

Basic Note11.22 Universal Note11.22

(3) File usability setting (☞ Section 9.1.2(7))Note11.22

In the program that does not use the file register, device initial value or device comment file, the overhead time of the program can be shortened by selecting "Not used" in the file usability setting.



	Program name	Execute type	Fixed scan interval	In unit
1	PROGRAM1	Scan		
2	PROGRAM2	Scan		
3	PROGRAM3	Wait		

	Program	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

When any of the file register, device initial value and device comment files is not used in the program, set the corresponding file to "Not use".

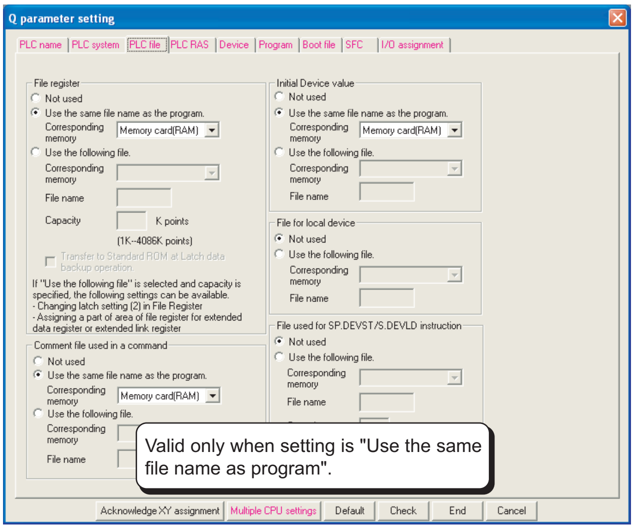
Figure 11.9 File usability setting screen

Basic Note11.22 Universal Note11.22

The Basic model QCPU does not support the file usability setting.
The Universal model QCPU can set the local device only. When using the file usablility setting with the Universal model QCPU, check the versions of the CPU module and GX developer.
(☞ Appendix 4.5)

☒POINT

Time reduction by this setting is enabled only when "Use the same file name as the program" is selected in the 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--4086K 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: Memory card(RAM)
○ Use the following file.
Corresponding memory
File name

Initial Device value
○ Not used
● Use the same file name as the program.
Corresponding memory: Memory card(RAM)
○ 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

Valid only when setting is "Use the same file name as program".

Figure 11.10 PLC file

11.2 Other Processing Times

This section explains the processing times other than those described in Section 11.1.

(1) Constant scan accuracy

Table11.33 indicates the constant scan accuracy.

Table11.33 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 11.1.2 (4) (a))	Sum of the following times 1) Time indicated in the "With monitor, without user interrupt" field on the left 2) Sum of interrupt program execution times
Q00CPU	0.12ms	0.60ms		
Q01CPU	0.10ms	0.50ms		
Q02CPU	0.02ms			
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU, Q12PRHCPU, Q25PRHCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q13UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q13UDEHCPU, Q26UDEHCPU	0.01ms			

With monitor : Indicates the status in which monitor is being executed with GX Developer connected or the status in which communication with the external device is being made by the serial communication function.

Without monitor : Indicates the status in which monitor is not executed by GX Developer or serial communication function.

CHAPTER12 PROCEDURE FOR WRITING PROGRAM TO CPU MODULE

This chapter describes the procedure for writing program created at the GX Developer to the CPU module.
The CPU module startup procedure is not described in this manual.
Refer to the following manuals for the CPU module startup procedure.

☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)
☞ QCPU User's Manual (Multiple CPU System)
☞ QnPRHCPU User's Manual (Redundant System)

12.1 Basic Model QCPU

12.1.1 Items to be examined for program creation

To create a program with the Basic model QCPU, the program capacity, the number of device points used, etc. must be determined in advance.

(1) Program size considerations

Examine whether programs and parameters can be stored within the program capacity executable in the used CPU module. (☞ Section 5.4.3(1))
The program capacities executable in CPU modules are shown in Table12.1.

Table12.1 Program capacity of Basic model QCPU

CPU module	Program capacity
Q00JCPU	8k steps (32k bytes)
Q00CPU	8k steps (32k bytes)
Q01CPU	14k steps (56k bytes)

(2) Applications of devices and setting of their numbers of points

Consider the applications of the devices used in a program and their number of points. (☞ CHAPTER 10)

(3) Device initial value setting

Set the data necessary as initial values to the device memory of the Basic model QCPU and the buffer memory of the intelligent function module.
(☞ Section 6.26)

(4) Boot operation considerations

When performing boot operation, make the boot file setting of PLC parameter.
(☞ Section 5.1.5, Section 12.1.4)

12.1.2 Hardware check

Make a hardware check before writing the created program.

In the following procedure, □ indicates the operation on the GX Developer side, and □ indicates that on the Basic model QCPU side.

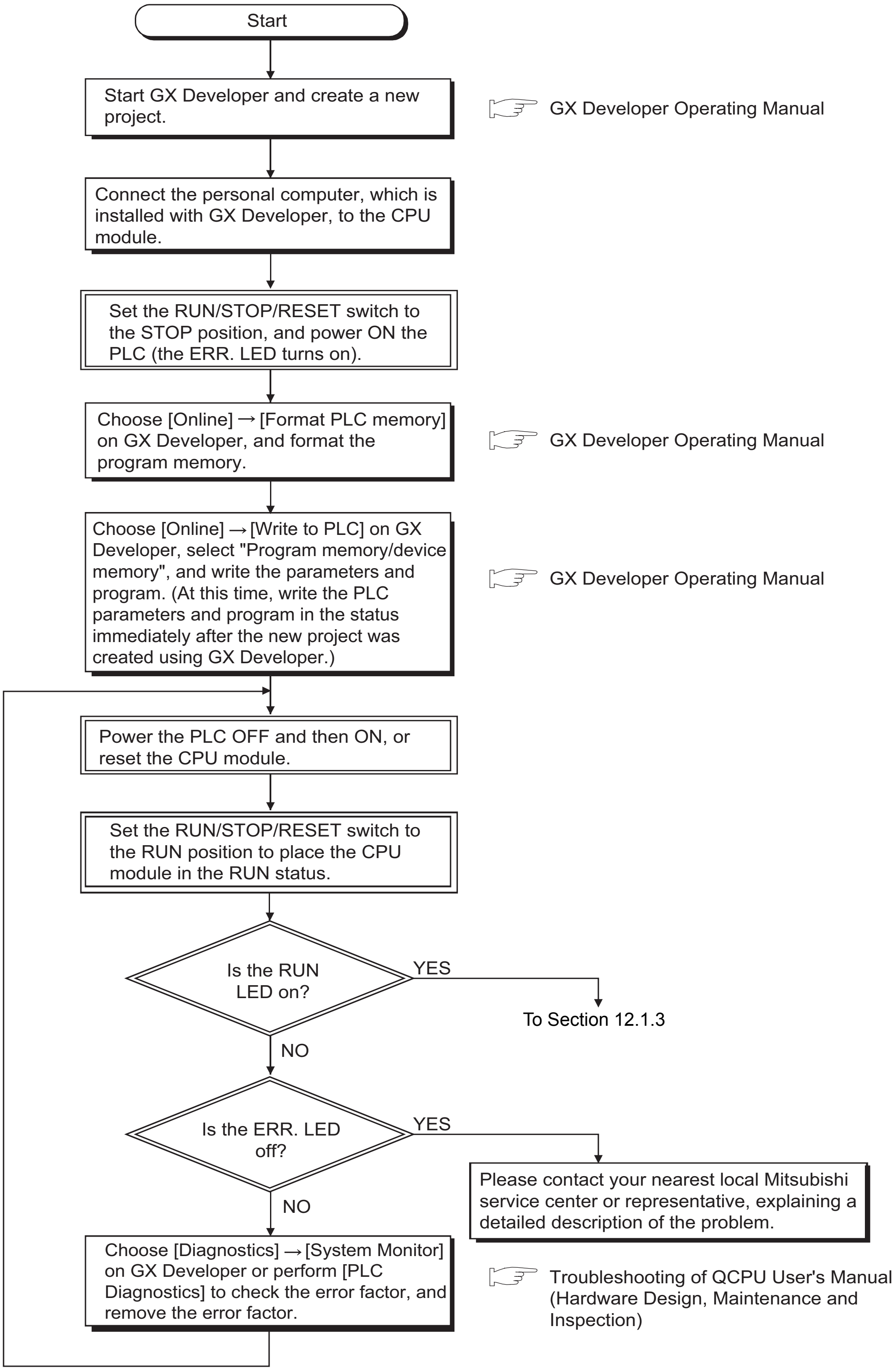

Start
Start GX Developer and create a new project.
GX Developer Operating Manual
Connect the personal computer, which is installed with GX Developer, to the CPU module.
Set the RUN/STOP/RESET switch to the STOP position, and power ON the PLC (the ERR. LED turns on).
Choose [Online] → [Format PLC memory] on GX Developer, and format the program memory.
GX Developer Operating Manual
Choose [Online] → [Write to PLC] on GX Developer, select "Program memory/device memory", and write the parameters and program. (At this time, write the PLC parameters and program in the status immediately after the new project was created using GX Developer.)
GX Developer Operating Manual
Power the PLC OFF and then ON, or reset the CPU module.
Set the RUN/STOP/RESET switch to the RUN position to place the CPU module in the RUN status.
Is the RUN LED on?
YES
To Section 12.1.3
NO
Is the ERR. LED off?
YES
Please contact your nearest local Mitsubishi service center or representative, explaining a detailed description of the problem.
NO
Choose [Diagnostics] → [System Monitor] on GX Developer or perform [PLC Diagnostics] to check the error factor, and remove the error factor.
Troubleshooting of QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Figure 12.1 Hardware check flowchart

Remark

Refer to the following manual for the installation and mounting procedures of the CPU module.

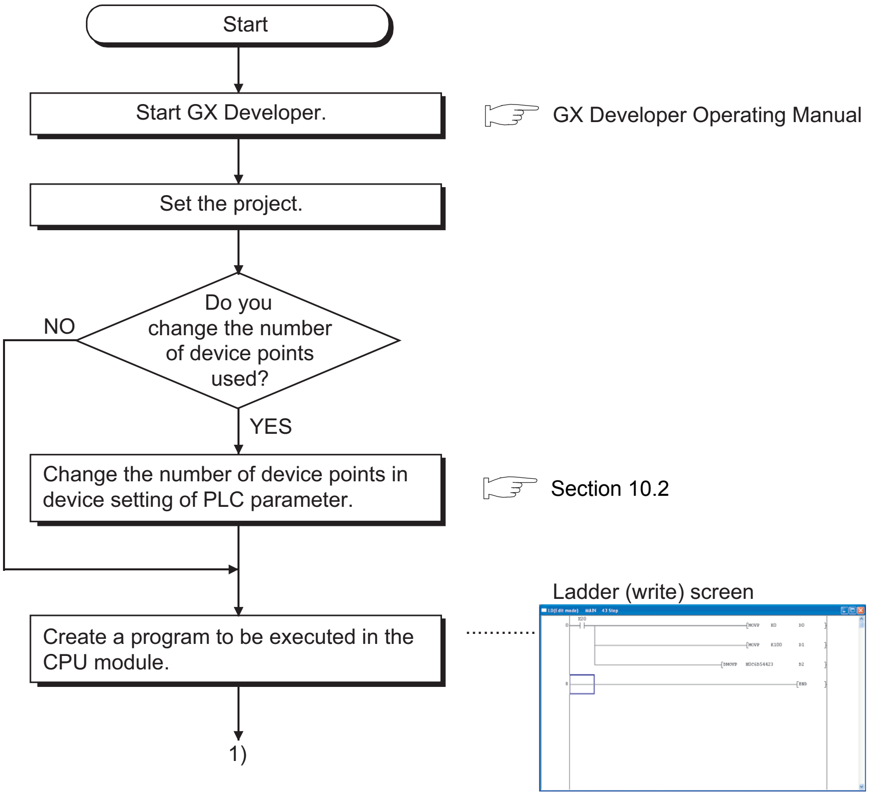
☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)

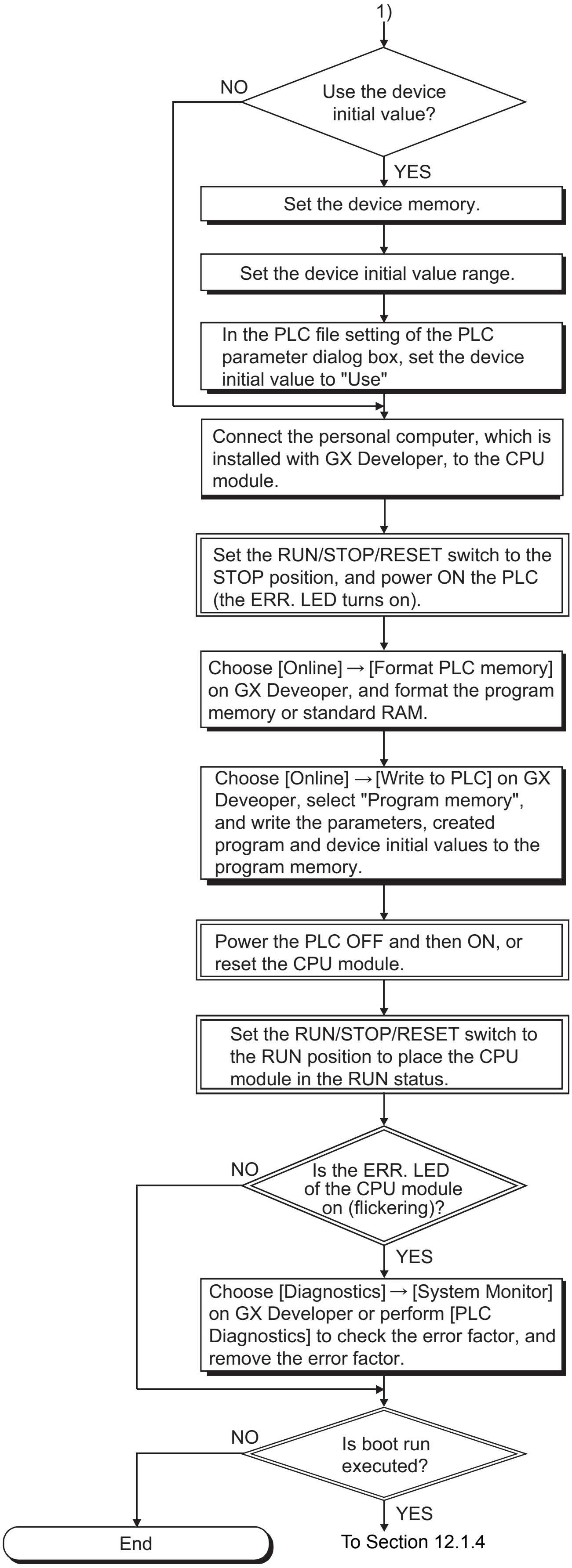
12.1.3 Procedure for writing program

This section explains the procedure for writing the parameters and program created by GX Developer to the Basic model QCPU.

This section explains the procedure for writing a program to the program memory (☞ Section 5.1.2).
When storing the program into the standard ROM and executing boot run, perform the procedure given in this section, and then carry out the procedure given in Section 12.1.4.

Procedural steps shown in ▭ boxes are performed at the GX Developer, and those shown in ▭ boxes are performed in the Basic model QCPU.


Start
Start GX Developer.
☞ GX Developer Operating Manual
Set the project.
Do you change the number of device points used?
NO
YES
Change the number of device points in device setting of PLC parameter.
☞ Section 10.2
Create a program to be executed in the CPU module.
Ladder (write) screen
LD(Edit mode) MAIN 43 Step
0
X20
MOVP K0 D0
MOVP K100 D1
DMOVP H0C6B54423 D2
8
END
1)

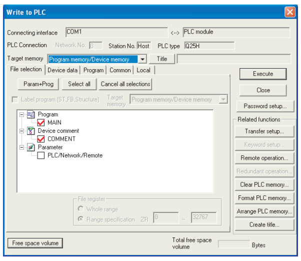

1)
Use the device initial value?
NO
YES
Set the device memory.
Set the device initial value range.
In the PLC file setting of the PLC parameter dialog box, set the device initial value to "Use"
Connect the personal computer, which is installed with GX Developer, to the CPU module.
Set the RUN/STOP/RESET switch to the STOP position, and power ON the PLC (the ERR. LED turns on).
Choose [Online] → [Format PLC memory] on GX Deveoper, and format the program memory or standard RAM.
Choose [Online] → [Write to PLC] on GX Deveoper, select "Program memory", and write the parameters, created program and device initial values to the program memory.
Power the PLC OFF and then ON, or reset the CPU module.
Set the RUN/STOP/RESET switch to the RUN position to place the CPU module in the RUN status.
Is the ERR. LED of the CPU module on (flickering)?
NO
YES
Choose [Diagnostics] → [System Monitor] on GX Developer or perform [PLC Diagnostics] to check the error factor, and remove the error factor.
Is boot run executed?
NO
YES
End
To Section 12.1.4

☞ Section 6.26

☞ Section 6.26

☞ Section 6.26

Write to PLC screen



Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q25H

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 12.2 Flowchart for writing program

12.1.4 Boot run procedure

This section explains a boot run procedure.

In the following procedure, □ indicates the operation on the GX Developer side, and ▭ indicates that on the Basic model QCPU side.

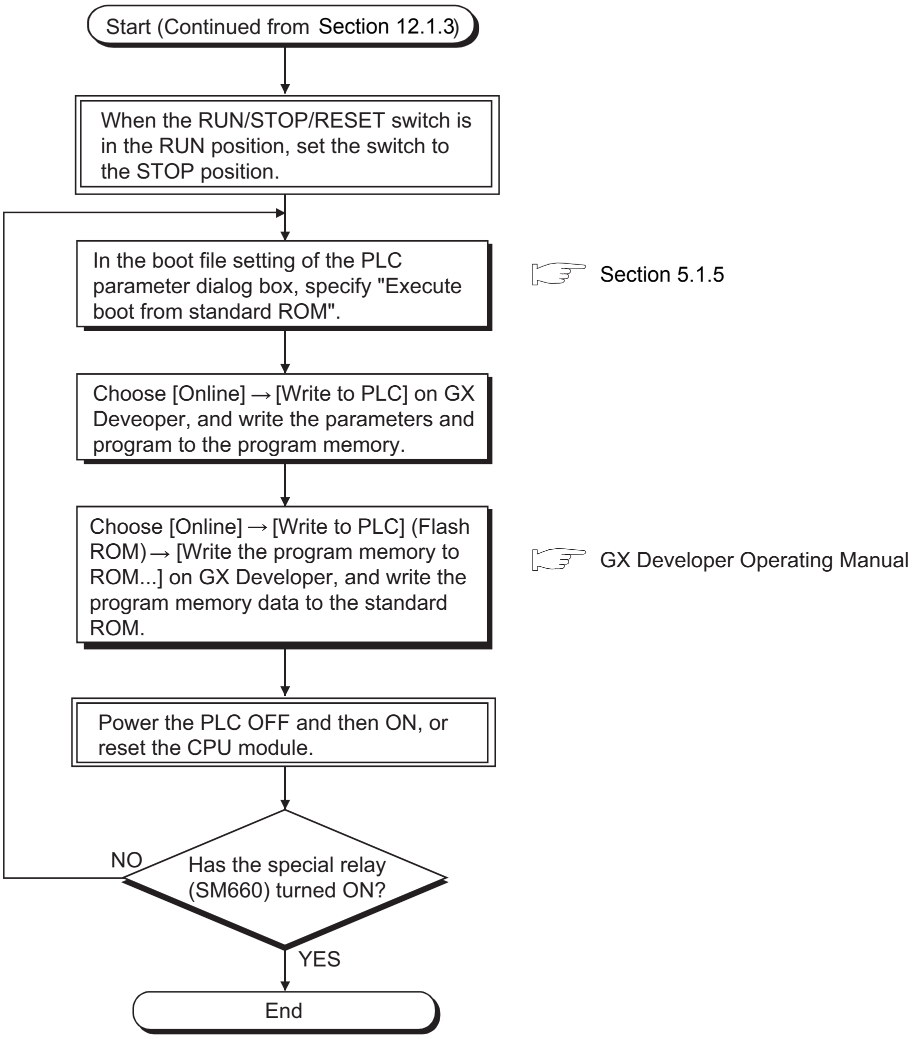

Start (Continued from Section 12.1.3)
When the RUN/STOP/RESET switch is in the RUN position, set the switch to the STOP position.
In the boot file setting of the PLC parameter dialog box, specify "Execute boot from standard ROM".
Section 5.1.5
Choose [Online] → [Write to PLC] on GX Deveoper, and write the parameters and program to the program memory.
Choose [Online] → [Write to PLC] (Flash ROM) → [Write the program memory to ROM...] on GX Developer, and write the program memory data to the standard ROM.
GX Developer Operating Manual
Power the PLC OFF and then ON, or reset the CPU module.
Has the special relay (SM660) turned ON?
NO
YES
End

Figure 12.3 Boot run flowchart

12.2 High Performance Model QCPU, Process CPU, Redundant CPU, Universal Model QCPU

12.2.1 Items to be examined for program creation

When creating programs by the CPU module, it is necessary to predetermine the program capacity, device points, file name and others of each program.

(1) Program size considerations

Check that CPU module's program capacity is adequate for storing the program and parameter data. (☞ Section 5.4.3(2))
The program capacities of the CPUs are shown in Table12.2.

Table12.2 Program capacity of High Performance model QCPU, Process CPU, Redundant CPU

CPU module	Program capacity
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)
Q02UCPU	20k steps (80k bytes)
Q03UDCPU, Q03UDECPU	30k steps (120k bytes)
Q04UDHCPU, Q04UDEHCPU	40k steps (160k bytes)
Q06UDHCPU, Q06UDEHCPU	60k steps (240k bytes)
Q13UDHCPU, Q13UDEHCPU	130k steps (520k bytes)
Q26UDHCPU, Q26UDEHCPU	260k steps (1040k bytes)

Whether the program will be stored into the program memory, standard ROM or memory card can be set in the parameter.
If the CPU module capacity is only adequate for the program, the parameter data should be stored in the standard ROM/memory card.

(2) Determining the program structuring unit

The High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU allows the file name to be changed to store multiple programs.
When creating multiple programs, determine the program structuring unit (by process, by function).

(3) Setting the execution condition of created program

When executing multiple programs, set an execution condition to each program. (☞ Section 3.3.6)
Programs cannot be executed if their file names and execution conditions have not been set.

(4) Usage and point setting of used devices

The usage and the points of devices required for the program must be determined. (☞ CHAPTER 10)

(5) Device initial value setting

Set the data necessary as the initial values to the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU device memory and intelligent function module buffer memory. (☞ Section 6.26)

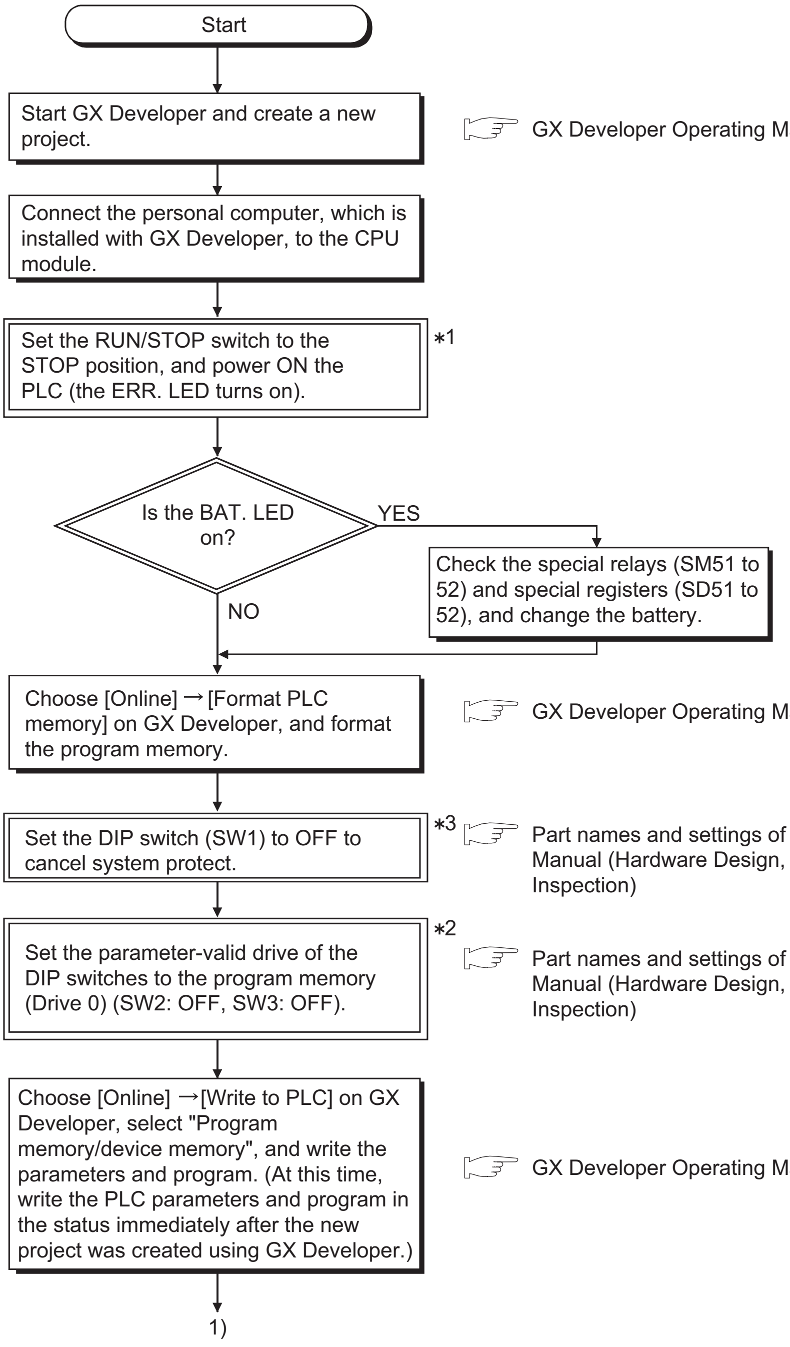
(6) Examination of boot run

When executing boot run, make the boot file setting of the PLC parameter dialog box. (☞Section 5.2.9, Section 12.2.5)

12.2.2 Hardware check

Make a hardware check before writing the created program.

In the following procedure, □ indicates the operation on the GX Developer side, and ▭ indicates that on the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU side.



Start

Start GX Developer and create a new project. ☞ GX Developer Operating Manual

Connect the personal computer, which is installed with GX Developer, to the CPU module.

Set the RUN/STOP switch to the STOP position, and power ON the PLC (the ERR. LED turns on). *1

Is the BAT. LED on?

YES: Check the special relays (SM51 to 52) and special registers (SD51 to 52), and change the battery.

NO

Choose [Online] → [Format PLC memory] on GX Developer, and format the program memory. ☞ GX Developer Operating Manual

Set the DIP switch (SW1) to OFF to cancel system protect. *3 ☞ Part names and settings of QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Set the parameter-valid drive of the DIP switches to the program memory (Drive 0) (SW2: OFF, SW3: OFF). *2 ☞ Part names and settings of QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Choose [Online] →[Write to PLC] on GX Developer, select "Program memory/device memory", and write the parameters and program. (At this time, write the PLC parameters and program in the status immediately after the new project was created using GX Developer.) ☞ GX Developer Operating Manual

1)

* 1 : RUN/STOP/RESET switch for the Universal model QCPU.

* 2 : The parameter-valid drive setting by the DIP switch is not necessary for the Universal model QCPU. (☞ Section 5.2.11)

* 3 : The system protect cannot be cancelled by the DIP switch in the Universal model QCPU.

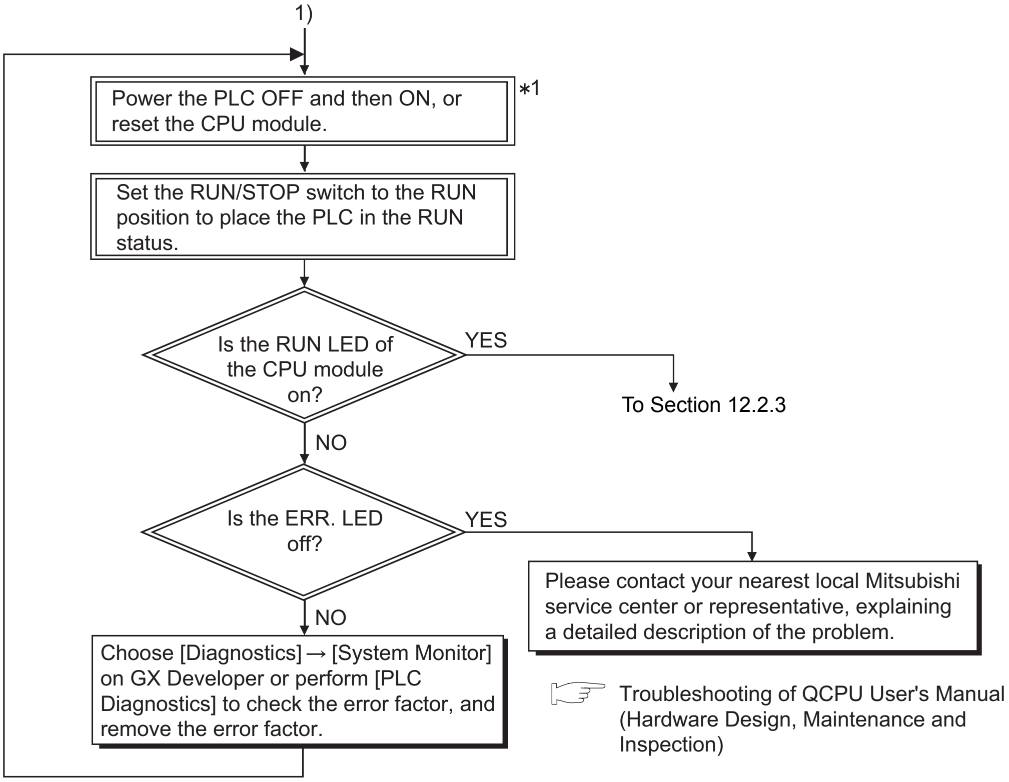

1)
Power the PLC OFF and then ON, or reset the CPU module.
*1
Set the RUN/STOP switch to the RUN position to place the PLC in the RUN status.
Is the RUN LED of the CPU module on?
YES
To Section 12.2.3
NO
Is the ERR. LED off?
YES
Please contact your nearest local Mitsubishi service center or representative, explaining a detailed description of the problem.
NO
Choose [Diagnostics] → [System Monitor] on GX Developer or perform [PLC Diagnostics] to check the error factor, and remove the error factor.
☞ Troubleshooting of QCPU User's Manual (Hardware Design, Maintenance and Inspection)

Figure 12.4 Hardware check flowchart

* 1 : RUN/STOP/RESET switch for the Universal model QCPU.

Remark

Refer to the following manual for the installation and mounting procedures of the CPU module.

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

12.2.3 Procedure for writing one program

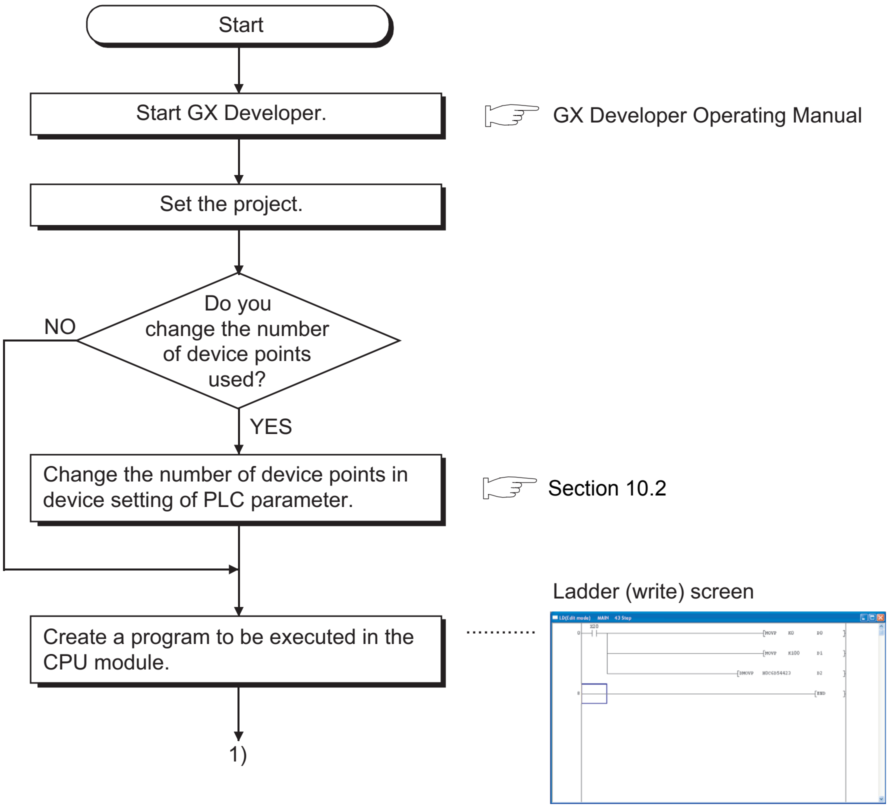
This section explains the procedure for writing the parameters and program created by GX Developer to the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU.

This section explains the procedure for writing a program to the program memory (☞ Section 5.2.2).

Universal UD Note12.1

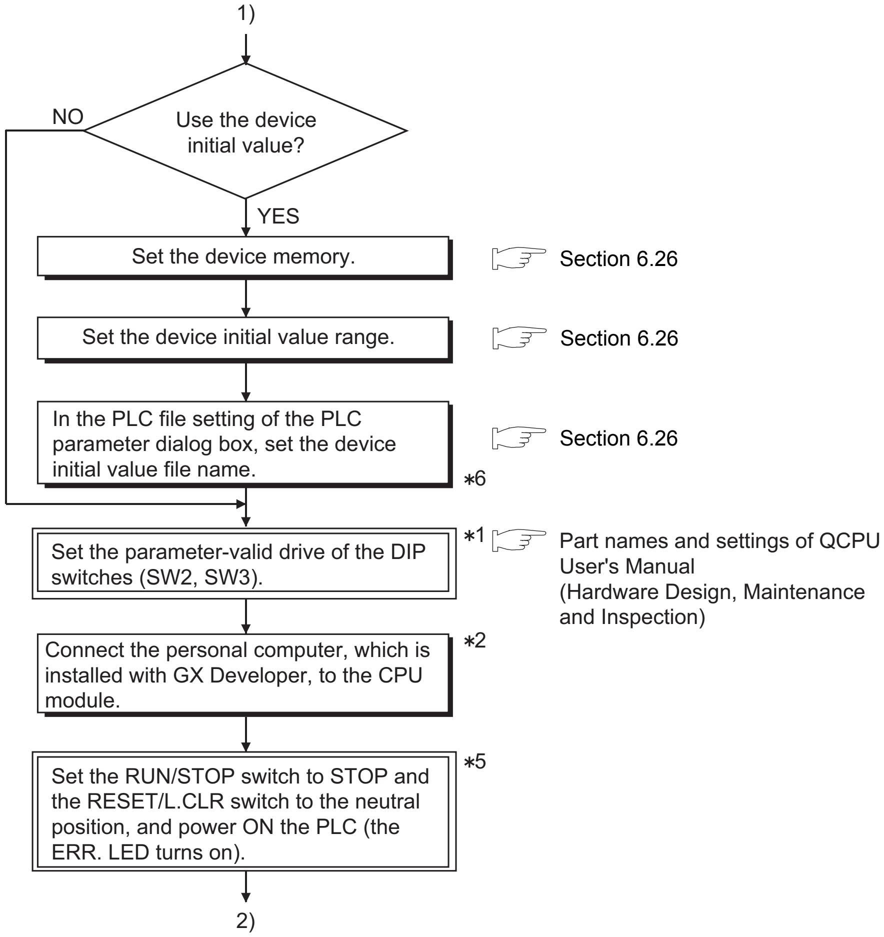
When storing the program into the standard ROM Note12.1 or memory card and executing boot run, perform the procedure given in this section, and then carry out the procedure given in Section 12.2.5.

In the following procedure, □ indicates the operation on the GX Developer side, and □ indicates that on the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU side.


Start
Start GX Developer.
☞ GX Developer Operating Manual
Set the project.
Do you change the number of device points used?
NO
YES
Change the number of device points in device setting of PLC parameter.
☞ Section 10.2
Ladder (write) screen
Create a program to be executed in the CPU module.
1)

Universal UD Note12.1

The Universal model QCPU cannot boot data from the standard ROM to the program memory. (☞ Section 5.2.3)

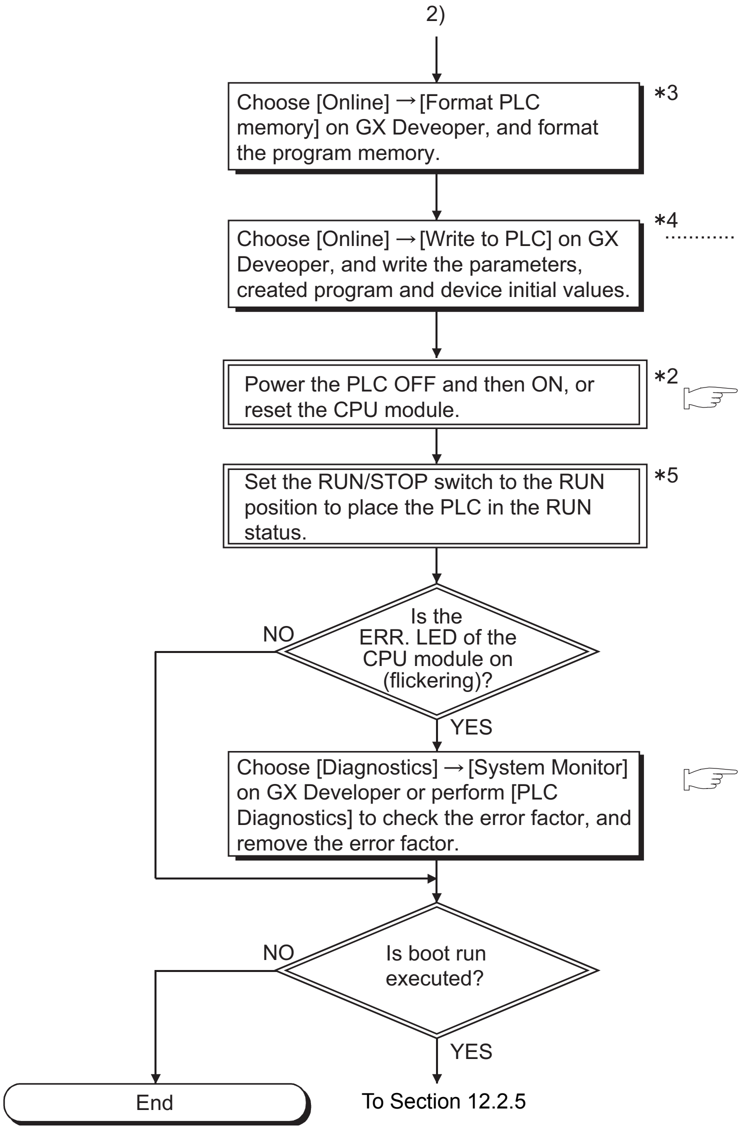

1)
Use the device initial value?
NO
YES
Set the device memory.
Section 6.26
Set the device initial value range.
Section 6.26
In the PLC file setting of the PLC parameter dialog box, set the device initial value file name.
Section 6.26
*6
*1
Set the parameter-valid drive of the DIP switches (SW2, SW3).
Part names and settings of QCPU User's Manual (Hardware Design, Maintenance and Inspection)
*2
Connect the personal computer, which is installed with GX Developer, to the CPU module.
*5
Set the RUN/STOP switch to STOP and the RESET/L.CLR switch to the neutral position, and power ON the PLC (the ERR. LED turns on).
2)

* 1 : By default, the parameter-valid drive is set to the program memory (Drive 0) with the DIP switches. (SW2: OFF, SW3: ON)
When writing the parameters to the standard ROM or memory card, change the parameter-valid drive with the DIP switches.

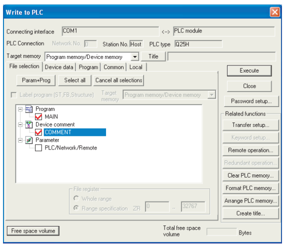
* 2 : Refer to the following manual for the procedure when the Redundant CPU is used.
☞ QnPRHCPU User's Manual (Redundant System)

* 5 : RUN/STOP/RESET switch for the Universal model QCPU.

* 6 : The parameter-valid drive setting by the DIP switch is not necessary for the Universal model QCPU. (☞ Section 5.2.11)


2)
Choose [Online] → [Format PLC memory] on GX Deveoper, and format the program memory.
*3
Choose [Online] → [Write to PLC] on GX Deveoper, and write the parameters, created program and device initial values.
*4
Power the PLC OFF and then ON, or reset the CPU module.
*2
Set the RUN/STOP switch to the RUN position to place the PLC in the RUN status.
*5
Is the ERR. LED of the CPU module on (flickering)?
NO
YES
Choose [Diagnostics] → [System Monitor] on GX Developer or perform [PLC Diagnostics] to check the error factor, and remove the error factor.
Is boot run executed?
NO
YES
End
To Section 12.2.5

Write to PLC screen



Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0 Station No. Host PLC type Q25H

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)

* 2 : Refer to the following manual for the procedure when the Redundant CPU is used.

QnPRHCPU User's Manual (Redundant System)

* 3 : When storing the file register and device initial value data into the standard RAM or memory card (except the Flash card), also format the standard RAM or memory card.

* 4 : Write each data to the following memory.

•Program ..... Program memory

•Parameter ..... Memory set as the parameter-valid drive with the DIP switches

•Device initial value ..... Memory specified in the PLC file setting of the PLC parameter dialog box

* 5 : RUN/STOP/RESET switch for the Universal model QCPU.

Figure 12.5 Flowchart for writing one program

☒POINT

In the MELSECNET/H system where the Universal model QCPU No.1 is the control CPU, the Universal model QCPU can communicate with the QnUCPU where MISSING PARA. has been occurred via the MELSECNET/H. (For the network No.1 only.)

12.2.4 Procedure for writing multiple programs

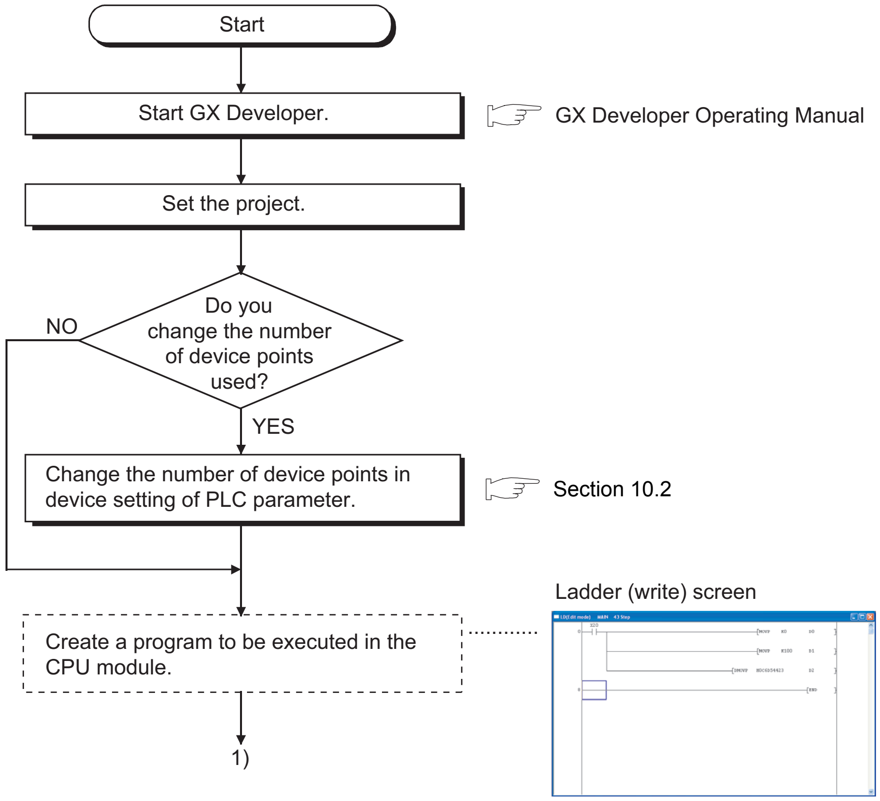
This section explains the procedure for writing the parameters and multiple programs created by GX Developer to the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU.

This section explains the procedure for writing the programs to the program memory (☞ Section 5.2.2).

Universal UD Note12.2

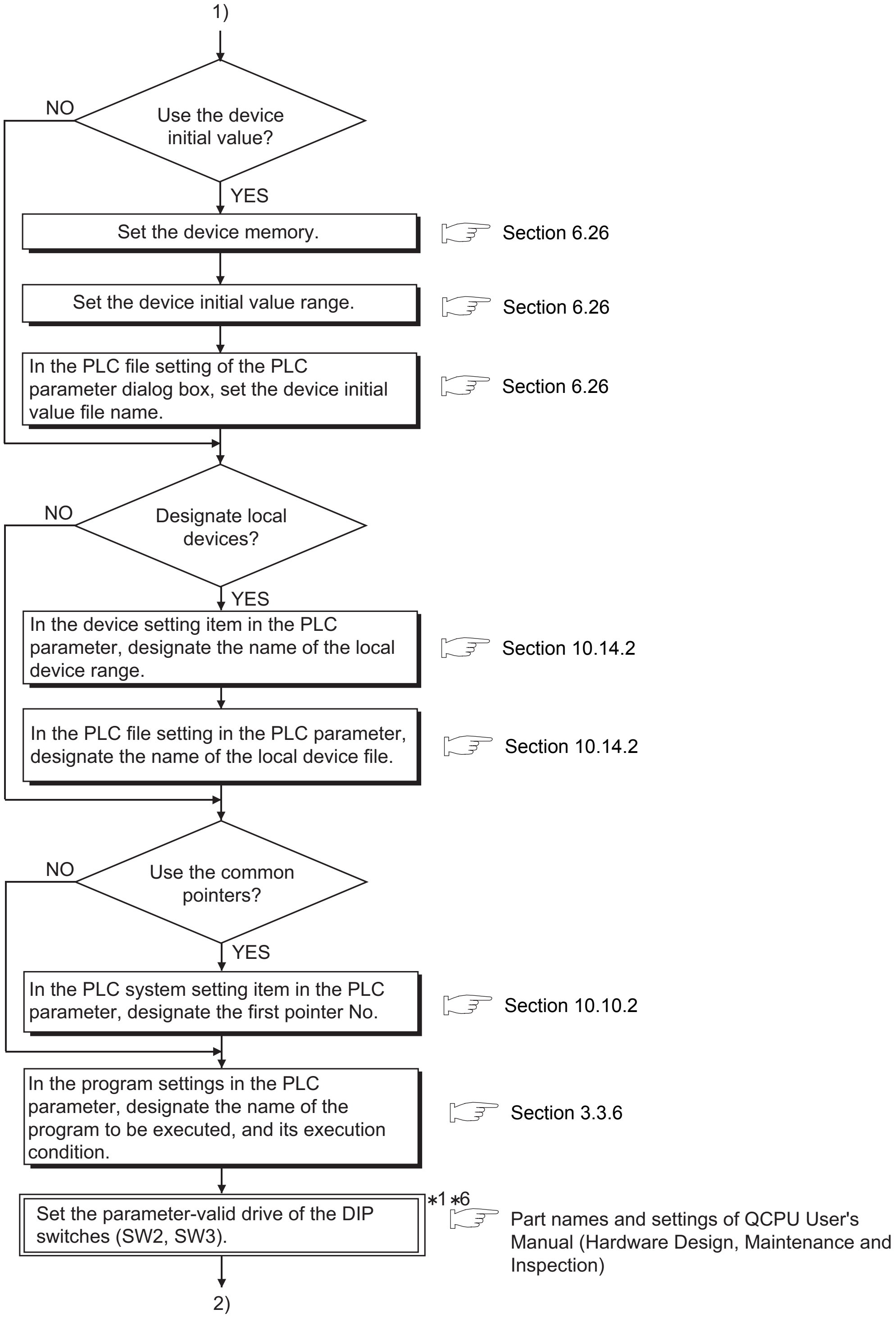
When storing the programs into the standard ROMNote12.2 or memory card and executing boot run, perform the procedure given in this section, and then carry out the procedure given in Section 12.2.5.

In the following procedure, □ indicates the operation on the GX Developer side, and □ indicates that on the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU side.


Start
Start GX Developer.
☞ GX Developer Operating Manual
Set the project.
Do you change the number of device points used?
NO
YES
Change the number of device points in device setting of PLC parameter.
☞ Section 10.2
Create a program to be executed in the CPU module.
Ladder (write) screen
1)

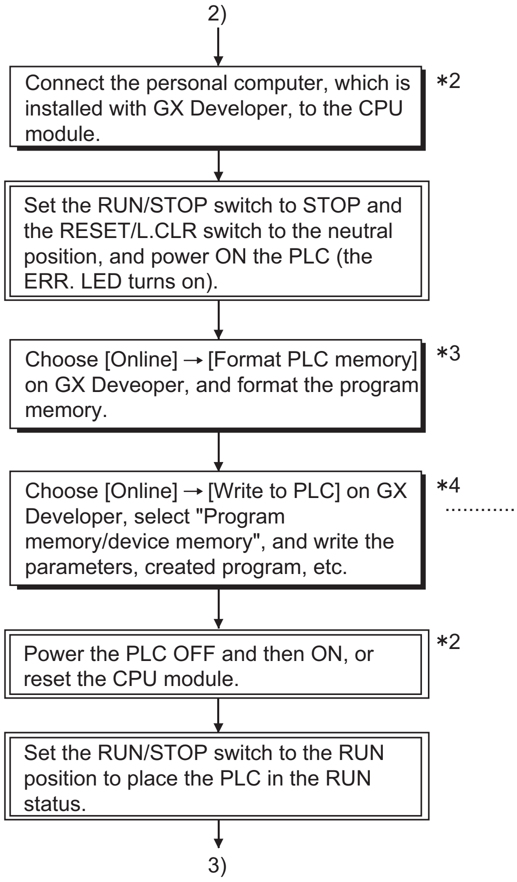
Universal UD Note12.2

The Universal model QCPU cannot boot data from the standard ROM to the program memory. (☞ Section 5.2.3)


1)
Use the device initial value?
NO
YES
Set the device memory.
☞ Section 6.26
Set the device initial value range.
☞ Section 6.26
In the PLC file setting of the PLC parameter dialog box, set the device initial value file name.
☞ Section 6.26
Designate local devices?
NO
YES
In the device setting item in the PLC parameter, designate the name of the local device range.
☞ Section 10.14.2
In the PLC file setting in the PLC parameter, designate the name of the local device file.
☞ Section 10.14.2
Use the common pointers?
NO
YES
In the PLC system setting item in the PLC parameter, designate the first pointer No.
☞ Section 10.10.2
In the program settings in the PLC parameter, designate the name of the program to be executed, and its execution condition.
☞ Section 3.3.6
Set the parameter-valid drive of the DIP switches (SW2, SW3).
*1 *6
☞ Part names and settings of QCPU User's Manual (Hardware Design, Maintenance and Inspection)
2)

* 1 : By default, the parameter-valid drive is set to the program memory (Drive 0)with the DIP switches. (SW2: OFF, SW3: ON)
When writing the parameters to the standard ROM or memory card, change the parameter-valid drive with the DIP switches.

* 6 : The parameter-valid drive setting by the DIP switch is not necessary for the Universal model QCPU. (☞ Section 5.2.11)



2)

Connect the personal computer, which is installed with GX Developer, to the CPU module. *2

Set the RUN/STOP switch to STOP and the RESET/L.CLR switch to the neutral position, and power ON the PLC (the ERR. LED turns on).

Choose [Online] → [Format PLC memory] on GX Deveoper, and format the program memory. *3

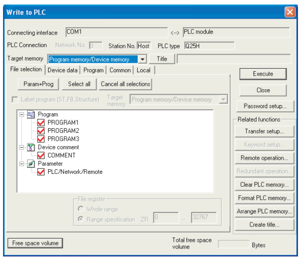
Choose [Online] → [Write to PLC] on GX Developer, select "Program memory/device memory", and write the parameters, created program, etc. *4

Power the PLC OFF and then ON, or reset the CPU module. *2

Set the RUN/STOP switch to the RUN position to place the PLC in the RUN status.

3)

Write to PLC screen



Write to PLC

Connecting interface: COM1 <--> PLC module

PLC Connection: Network No. 0, Station No. Host, PLC type Q25H

Target memory: Program memory/Device memory

File selection:
- Program
  - ☑ PROGRAM1
  - ☑ PROGRAM2
  - ☑ PROGRAM3
- Device comment
  - ☑ COMMENT
- Parameter
  - ☑ PLC/Network/Remote

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

* 2 : Refer to the following manual for the procedure when the Redundant CPU is used.

☞ QnPRHCPU User's Manual (Redundant System)

* 3 : When storing the file register and device initial value data into the standard RAM or memory card (except the Flash card), also format the standard RAM or memory card.

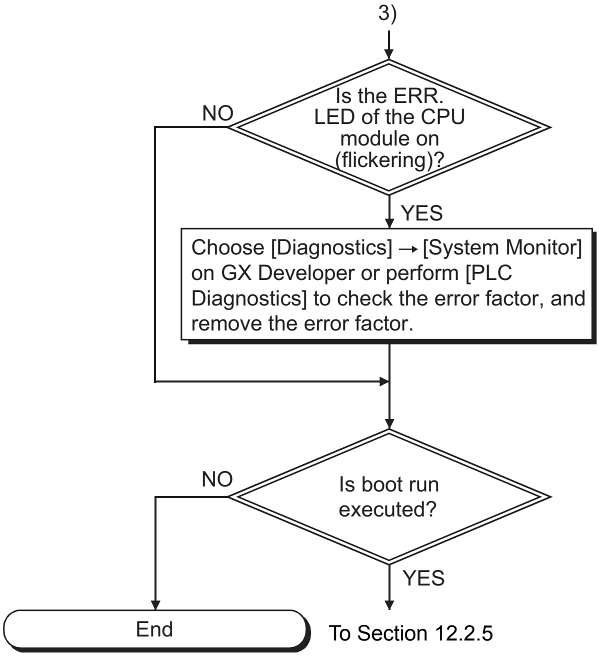
* 4 : Write each data to the following memory.

•Program ..... Program memory

•Parameter ..... Memory set as the parameter-valid drive with the DIP switches

•Device initial value ..... Memory specified in the PLC file setting of the PLC parameter dialog box

* 5 : RUN/STOP/RESET switch for the Universal model QCPU.


3)
Is the ERR. LED of the CPU module on (flickering)?
NO
YES
Choose [Diagnostics] → [System Monitor] on GX Developer or perform [PLC Diagnostics] to check the error factor, and remove the error factor.
Is boot run executed?
NO
YES
End
To Section 12.2.5

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

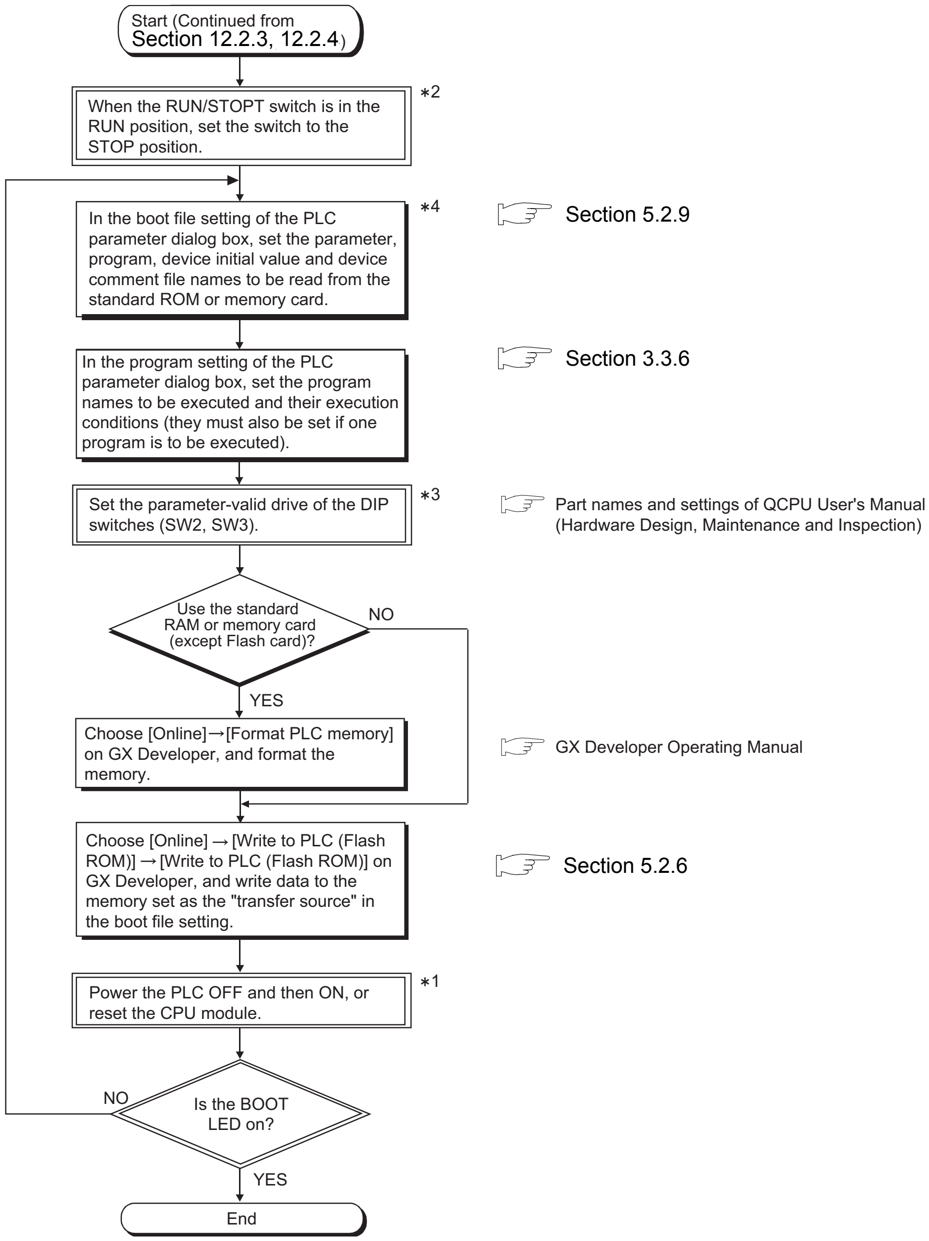
Figure 12.6 Flowchart for writing multiple programs

☒POINT

In the MELSECNET/H system where the Universal model QCPU No.1 is the control CPU, the Universal model QCPU can communicate with the QnUCPU where MISSING PARA. has been occurred via the MELSECNET/H. (For the network No.1 only.)

12.2.5 Boot run procedure

This section explains a boot run procedure.
In the following procedure, □ indicates the operation on the GX Developer side, and ▭ indicates that on the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU side.


Start (Continued from Section 12.2.3, 12.2.4)
When the RUN/STOPT switch is in the RUN position, set the switch to the STOP position.
*2
In the boot file setting of the PLC parameter dialog box, set the parameter, program, device initial value and device comment file names to be read from the standard ROM or memory card.
*4
☞ Section 5.2.9
In the program setting of the PLC parameter dialog box, set the program names to be executed and their execution conditions (they must also be set if one program is to be executed).
☞ Section 3.3.6
Set the parameter-valid drive of the DIP switches (SW2, SW3).
*3
☞ Part names and settings of QCPU User's Manual (Hardware Design, Maintenance and Inspection)
Use the standard RAM or memory card (except Flash card)?
NO
YES
Choose [Online]→[Format PLC memory] on GX Developer, and format the memory.
☞ GX Developer Operating Manual
Choose [Online] → [Write to PLC (Flash ROM)] → [Write to PLC (Flash ROM)] on GX Developer, and write data to the memory set as the "transfer source" in the boot file setting.
☞ Section 5.2.6
Power the PLC OFF and then ON, or reset the CPU module.
*1
NO
Is the BOOT LED on?
YES
End

* 1 : Refer to the following manual for the procedure when the Redundant CPU is used.
☞ QnPRHCPU User's Manual (Redundant System)
* 2 : RUN/STOP/RESET switch for the Universal model QCPU.
* 3 : The parameter-valid drive setting by the DIP switch is not necessary for the Universal model QCPU. (☞ Section 5.2.11)
* 4 : The Universal model QCPU cannot boot data from the standard ROM to program memory. (☞ Section 5.2.3).

Figure 12.7 Boot run flowchart

APPENDICES

Appendix 1 Special Relay List

Special relays, SM, are internal relays whose applications are fixed in the Programmable Controller.
For this reason, they cannot be used by sequence programs in the same way as the normal internal relays.
However, they can be turned ON or OFF as needed in order to control the CPU module and remote I/O modules.

The heading descriptions in the following special relay lists are shown in TableApp.1.

TableApp.1 Explanation of special relay list

Item	Function of Item
Number	• Indicates special register number
Name	• Indicates name of special register
Meaning	• Indicates contents of special register
Explanation	• Discusses contents of special register in more detail
Set by (When set)	• Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed. <Set by> S : Set by system U : Set by user (sequence programs or test operations from GX Developer) S/U : Set by both system and user <When set> Indicated only for registers set by system Each END : Set during each END processing Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN) Status change : Set only when there is a change in status Error : Set when error occurs Instruction execution : Set when instruction is executed Request : Set only when there is a user request (through SM, etc.) System switching : Set when system switching is executed.
Corresponding ACPU M9□ □ □	• Indicates the corresponding special relay (M9□□□) of the ACPU. (When the contents are changed, the special relay is represented M9□□□ format change. Incompatible with the Q00J/Q00/Q01 and QnPRH.) • New indicates the special relay newly added to the QnACPU or Q series CPU module.
Corresponding CPU	Indicates the corresponding CPU module type name. ○ : Indicates all the QnACPUs and QCPU. QCPU : Indicates all the Q series CPU modules. Q00J/Q00/Q01 : Indicates the Basic model QCPU. Qn(H) : Indicates the High Performance model QCPU. QnPH : Indicates the Process CPU. QnPRH : Indicates the Redundant CPU. QnU : Indicates the Universal model QCPU QnA : Indicates the QnA series and Q2ASCPU series. Rem : Indicates the MELSECNET/H remote I/O modules. Each CPU module model name: Indicates the relevant specific CPU module. (Example: Q4AR, Q3A)

For details on the following items, refer to the following manuals:

- Networks → Manual of the corresponding network module
- SFC → QCPU(Q mode)/QnACPU Programming Manual (SFC)

☒POINT

(1) SD1200 to SD1255 are used for QnACPU.
These relays are vacant with QCPU.

(1) Diagnostic Information

TableApp.2 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM0	Diagnostic errors	OFF : No error ON : Error	• Turns ON if an error occurs as a result of diagnosis. (Includes when an annunciator is ON, and when an error is detected with CHK instruction) • Remains ON even if the condition is restored to normal thereafter.	S (Error)	New	QnA Qn(H) QnPH QnPRH Rem
			• Turns ON if an error occurs as a result of diagnosis. (Includes when an annunciator is ON) • Remains ON even if the condition is restored to normal thereafter.	S (Error)	New	Q00J/Q00/Q01 QnU
SM1	Self-diagnostic error	OFF : No self-diagnosis errors ON : Self-diagnosis	• Turns ON if an error occurs as a result of diagnosis. (Does not include when an annunciator is ON or when an error is detected by the CHK instruction) • Remains ON even if the condition is restored to normal thereafter.	S (Error)	M9008	QnA Qn(H) QnPH QnPRH Rem
			• Turns ON if an error occurs as a result of diagnosis. (Does not include when an annunciator is ON) • Remains ON even if the condition is restored to normal thereafter.	S (Error)	New	Q00J/Q00/Q01 QnU
SM5	Error common information	OFF : No error common information ON : Error common information	• When SM0 is ON, turns ON if there is error common information	S (Error)	New	○ Rem
SM16	Error individual information	OFF : No error individual information ON : Error individual information	• When SM0 is ON, turns ON if there is error individual information	S (Error)	New	
SM50	Error reset	OFF→ON: Error reset	• Conducts error reset operation	U	New	
SM51	Battery low latch	OFF : Normal ON : Battery low	• Turns ON if battery voltage at CPU module or memory card drops below rated value. • Remains ON even if the battery voltage returns to normal thereafter. • Synchronizes with the BAT. ALARM/BAT. LED.	S (Error)	M9007	QnA Qn(H) QnPH QnPRH QnU
			• Turns ON if battery voltage at CPU module or memory card drops below rated value. • Remains ON even if the battery voltage returns to normal thereafter. • Synchronous with BAT. LED	S (Error)	New	Q00J/Q00/Q01
SM52	Battery low	OFF : Normal ON : Battery low	• Same as SM51, but turns OFF subsequently when battery voltage returns to normal.	S (Error)	M9006	○
SM53	AC/DC DOWN detection	OFF : AC/DC DOWN not detected ON : AC/DC DOWN detected	• Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module. Reset when the power supply is switched OFF, then ON.	S (Error)	M9005	
			• Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module. Reset when the power supply is switched OFF, then ON.			QCPU
			• Turns ON if an instantaneous power failure of within 1ms occurs during use of the DC power supply module. Reset when the power supply is switched OFF, then ON.			QnA
SM54	MINI link error	OFF : Normal ON : Error	• Turns ON if MINI (S3) link error is detected at even one of the installed AJ71PT32 (S3) modules. • Remains ON even if the condition is restored to normal thereafter.	S (Error)	M9004	

TableApp.2 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM56	Operation error	OFF : Normal ON : Operation error	• ON when operation error is generated • Remains ON if the condition is restored to normal thereafter.	S (Error)	M9011	○
SM60	Blown fuse detection	OFF : Normal ON : Module with blown fuse	• Turns ON if there is at least one output module whose fuse has blown. • Remains ON if the condition is restored to normal thereafter. • Blown fuse status is checked even for remote I/O station output modules.	S (Error)	M9000	○ Rem
SM61	I/O module verify error	OFF : Normal ON : Error	• Turns ON if the I/O module differs from the status registered at power on. • Remains ON if the condition is restored to normal thereafter. • I/O module verification is also conducted for remote I/O station modules.	S (Error)	M9002	
SM62	Annunciator detection	OFF : Not detected ON : Detected	• Goes ON if even one annunciator F goes ON.	S (Instruction execution)	M9009	○
SM80	CHK detection	OFF : Not detected ON : Detected	• Goes ON if error is detected by CHK instruction. • Remains ON if the condition is restored to normal thereafter.	S (Instruction execution)	New	QnA Qn(H) QnPH QnPRH
SM90	Startup of monitoring timer for step transition (Enabled only when SFC program exists)	OFF : Not started(monitoring timer reset) ON : Started(monitoring timer started)	Corresponds to SD90 • Goes ON when measurement of step transition monitoring timer is commenced. • Resets step transition monitoring timer when it goes OFF.	U	M9108	QnA Qn(H) QnPH QnPRH
SM91			Corresponds to SD91		M9109	
SM92			Corresponds to SD92		M9110	
SM93			Corresponds to SD93		M9111	
SM94			Corresponds to SD94		M9112	
SM95			Corresponds to SD95		M9113	
SM96			Corresponds to SD96		M9114	
SM97			Corresponds to SD97		New	
SM98			Corresponds to SD98		New	
SM99			Corresponds to SD99		New	
SM100	Serial communication function using flag	OFF : Serial communication function is not used. ON : Serial communication function is used.	• Stores the setting of whether the serial communication function is used or not in the serial communication setting parameter	S (Power-ON or reset)	New	Q00J/Q00/Q01
SM101	Communication protocol status flag	OFF : GX Developer ON : MC protocol communication device	• Stores whether the device that is communicating via the RS-232 interface is GX Developer or MC protocol communication device	S (RS232 communication)		
SM110	Protocol error	OFF : Normal ON : Abnormal	• Turns ON when an abnormal protocol was used to make communication in the serial communication function. • Remains ON if the condition is restored to normal thereafter	S (Error)		
SM111	Communication status	OFF : Normal ON : Abnormal	• Turns ON when the mode used to make communication was different from the setting in the serial communication function. • Remains ON if the condition is restored to normal thereafter.	S (Error)		
SM112	Error information clear	ON : Cleared	• Turns ON when the error codes stored in SM110, SM111, SD110 and SD111 are cleared. (Activated when turned from OFF to ON)	U		
SM113	Overrun error	OFF : Normal ON : Abnormal	• Turns ON when an overrun error occurred in the serial communication error.	S (Error)		
SM114	Parity error	OFF : Normal ON : Abnormal	• Turns ON when a parity error occurred in the serial communication error.	S (Error)		
SM115	Framing error	OFF : Normal ON : Abnormal	• Turns ON when a framing error occurred in the serial communication error.	S (Error)		

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM165	Program memory batch transfer execution status	OFF : End ON : Not executed or not completed	• Turns ON when writing to the program cache memory. • Turns OFF when the program batch transfer is completed. • Remains ON when the program batch transfer is not executed after writing to the program memory.	S (Status change)	New	QnU*6

*6: This applies to the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU whose first 5 digits of serial number is "10012" or later, and the Q13UDHCPU, Q26UDHCPU.

(2) System information

TableApp.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM202	LED OFF command	OFF→ON : LED OFF	• When this relay goes from OFF to ON, the LEDs corresponding to the individual bits at SD202 go off	U	New	QnA Qn(H) QnPH QnPRH QnU
SM203	STOP contact	STOP status	• Goes ON at STOP status	S (Status change)	M9042	○
SM204	PAUSE contact	PAUSE status	• Goes ON at PAUSE status	S (Status change)	M9041	
SM205	STEP-RUN contact	STEP-RUN status	• Goes ON at STEP-RUN status	S (Status change)	M9054	QnA
SM206	PAUSE enable coil	OFF : PAUSE disabled ON : PAUSE enabled	• PAUSE status is entered if this relay is ON when the PAUSE contact goes ON	U	M9040	○
	Device test request acceptance status	OFF : Device test not yet executed ON : Device test executed	• Comes ON when the device test mode is executed on GX Developer.	S (Request)	New	Rem
SM210	Clock data set request	OFF : Ignored ON : Set request	• When this relay goes from OFF to ON and after END instruction execution of subsequent scan, clock data stored in SD210 to SD213 are written to the CPU module.	U	M9025	○
SM211	Clock data error	OFF : No error ON : Error	• ON when error is generated in clock data (SD210 to SD213) value, and OFF if no error is detected.	S (Request)	M9026	
SM212	Clock data display	OFF : Ignored ON : Display	• Displays clock data as month, day, hour, minute, and second at the LED display at front of CPU module.(Enabled only for Q3ACPU and Q4ACPU)	U	M9027	Q3A Q4A Q4AR
SM213	Clock data read request	OFF : Ignored ON : Read request	• When this relay is ON, clock data is read to SD210 to SD213 as BCD values.	U	M9028	○ Rem
SM220	CPU No.1 preparation completed	OFF : CPU No.1 preparation uncompleted ON : CPU No.1 preparation completed	Turned ON when access can be made to the CPU module No.1 from the other CPU module at power-on or reset operation. SM220 is used as interlock for accessing the CPU module No.1 when the multiple CPU synchronous setting is asynchronous.	S (When status changed)	New	QnU
SM221	CPU No.2 preparation completed	OFF : CPU No.2 preparation uncompleted ON : CPU No.2 preparation completed	Turned ON when access can be made to the CPU module No.2 from the other CPU module at power-on or reset operation. SM221 is used as interlock for accessing the CPU module No.2 when the multiple CPU synchronous setting is asynchronous.			
SM222	CPU No.3 preparation completed	OFF : CPU No.3 preparation uncompleted ON : CPU No.3 preparation completed	Turned ON when access can be made to the CPU module No.3 from the other CPU module at power-on or reset operation. SM222 is used as interlock for accessing the CPU module No.3 when the multiple CPU synchronous setting is asynchronous.			
SM223	CPU No.4 preparation completed	OFF : CPU No.4 preparation uncompleted ON : CPU No.4 preparation completed	Turned ON when access can be made to the CPU module No.4 from the other CPU module at power-on or reset operation. SM223 is used as interlock for accessing the CPU module No.4 when the multiple CPU synchronous setting is asynchronous.			QnU*5
SM235	Online module change flag	OFF : Online module change is not in progress ON : Online module change in progress	• Turns on during online module change. (for host CPU)	S (During online module change)	New	QnPH
SM236	Online module change complete flag	OFF :Online module change incomplete ON : Online module change complete	• Turns ON for one scan after online module change is complete. • This contact point can only be used by the scan program. (for host CPU)	S (When online module change is complete)	New	

*5: The Universal model QCPU except the Q02UCPU.

TableApp.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM237	Device range check inhibit flag	OFF : Check device range ON : Do not check device range	• Selects whether or not to check the device range by the BMOV/FMOV instruction (Only when the sub-set condition is established).	U	New	QnU*6
SM240	No. 1 CPU reset flag	OFF : No. 1 CPU reset cancel ON : No. 1 CPU resetting	• Goes OFF when reset of the No. 1 CPU is canceled. • Comes ON when the No. 1 CPU is resetting (including the case where the CPU module is removed from the base). The other CPUs are also put in reset status.	S (Status change)	New	Q00/Q01*1 Qn(H)*1 QnPH QnU
SM241	No. 2 CPU reset flag	OFF : No. 2 CPU reset cancel ON : No. 2 CPU resetting	• Goes OFF when reset of the No. 2 CPU is canceled. • Comes ON when the No. 2 CPU is resetting (including the case where the CPU module is removed from the base). The other CPUs result in "MULTI CPU DOWN" (error code: 7000).			
SM242	No. 3 CPU reset flag	OFF : No. 3 CPU reset cancel ON : No. 3 CPU resetting	• Goes OFF when reset of the No. 3 CPU is canceled. • Comes ON when the No. 3 CPU is resetting (including the case where the CPU module is removed from the base).The other CPUs result in "MULTI CPU DOWN" (error code: 7000).			
SM243	No. 4 CPU reset flag	OFF : No. 4 CPU reset cancel ON : No. 4 CPU resetting	• Goes OFF when reset of the No. 4 CPU is canceled. • Comes ON when the No. 4 CPU is resetting (including the case where the CPU module is removed from the base).The other CPUs result in "MULTI CPU DOWN" (error code: 7000).			Qn(H)*1 QnPH QnU*5
SM244	No. 1 CPU error flag	OFF : No. 1 CPU normal ON : No. 1 CPU during stop error	• Goes OFF when the No. 1 CPU is normal (including a continuation error). • Comes ON when the No. 1 CPU is during a stop error.			Q00/Q01*1 Qn(H)*1 QnPH QnU
SM245	No. 2 CPU error flag	OFF : No. 2 CPU normal ON : No. 2 CPU during stop error	• Goes OFF when the No. 2 CPU is normal (including a continuation error). • Comes ON when the No. 2 CPU is during a stop error.			
SM246	No. 3 CPU error flag	OFF : No. 3 CPU normal ON : No. 3 CPU during stop error	• Goes OFF when the No. 3 CPU is normal (including a continuation error). • Comes ON when the No. 3 CPU is during a stop error.			
SM247	No. 4 CPU error flag	OFF : No. 4 CPU normal ON : No. 4 CPU during stop error	• Goes OFF when the No. 4 CPU is normal (including a continuation error). • Comes ON when the No. 4 CPU is during a stop error	S (Status change)	New	Qn(H)*1 QnPH QnU*5
SM250	Max. loaded I/O read	OFF : Ignored ON : Read	• When this relay goes from OFF to ON, maximum loaded I/O number is read to SD250.	U	New	QnA Qn(H) QnPH QnPRH Rem
SM251	I/O change flag	OFF : No replacement ON : Replacement	• By turning this relay ON after setting the head I/O number of the replaced I/O module to SD251, the I/O module can be replaced online (with power on). (Only one module can be replaced for each setting.) • Turn this relay ON in the test mode of the program or peripheral device for an I/O module change during RUN, or in the test mode of the peripheral device for an I/O change during STOP. • Do not execute a RUN/STOP mode change until I/O module change is finished.	U	M9094	Q2A(S1) Q3A Q4A Q4AR
SM252	I/O change OK	OFF : Replacement prohibited ON : Replacement enabled	• Goes ON when I/O replacement is OK.	S (END)	New	

*1: This applies to the CPU of function version B or later.

*5: The Universal model QCPU except the Q02UCPU.

*6: This applies to the Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU whose first 5 digits of serial number is "10012" or later, and the Q13UDHCPU, Q26UDHCPU.

TableApp.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM254	All stations refresh command	OFF : Refresh arrival station ON : Refresh all stations	• Effective for the batch refresh (also effective for the low speed cyclic) • Designate whether to receive arrival stations only or to receive all slave stations in the MELSECNET/H.	U	New	Qn(H) QnPH QnPRH
			• Designate whether to receive arrival stations only or to receive all slave stations in the CC-Link IE controller network.			Qn(H)*2 QnPH*6 QnPRH*6
			• Effective for the batch refresh (also effective for the low speed cyclic) • Specify whether to receive only arrival station or all stations in the MELSECNET/H or CC-Link IE controller network.			QnU
SM255	MELSECNET/10, MELSECNET/H module 1 information	OFF : Operative network ON : Standby network	• Goes ON for standby network(If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	QnA Qn(H) QnPH QnPRH
SM256		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM257		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM260	MELSECNET/10, MELSECNET/H module 2 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	
SM261		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM262		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM265	MELSECNET/10, MELSECNET/H module 3 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	
SM266		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM267		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM270	MELSECNET/10, MELSECNET/H module 4 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	
SM271		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM272		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM280	CC-Link error	OFF : Normal ON : Error	• Goes ON when a CC-Link error is detected in any of the installed CC-Link module. Goes OFF when normal operation is restored.	S (Status change)	New	Qn(H) QnPH QnPRH Rem
			• Goes ON when a CC-Link error is detected in any of the installed CC-Link module. Remains ON if the condition is restored to normal thereafter.	S (Error)	New	QnA
SM315	Communication reserved time delay enable/disable flag	OFF : Without delay ON : With delay	• This flag is enabled when the time reserved for communication processing is set in SD315. • Turns ON to delay the END processing by the time set in SD315 in order to perform communication processing. (The scan time increases by the period set in SD315.) • Turns OFF to perform the END processing without a delay of the time set in SD315 when there is no communication processing. (Defaults to OFF)	U	New	Q00J/Q00/Q01
SM320	Presence/absence of SFC program	OFF : SFC program absent ON : SFC program present	• Turns ON when an SFC program is registered. • OFF when an SFC program is not registered.	S (Initial)	M9100	QnA Q00J/Q00/Q01*1 Qn(H) QnPH QnPRH QnU
SM321	Start/stop SFC program	OFF : SFC program not executed (stop) ON : SFC program executed (start)	• Initial value is set at the same value as SM320. (Goes ON automatically if SFC program is present.) • Turn this relay from ON to OFF to stop program execution. • Turn this relay from OFF to ON to resume program execution.	S (Initial)/U	M9101form at change	

*1: This applies to the CPU of function version B or later.

*2: The module whose first 5 digits of serial No. is "09012" or later.

*6: The module whose first 5 digits of serial No. is "10042" or later.

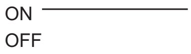
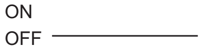
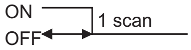
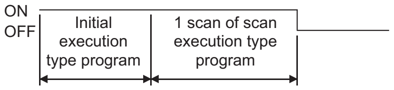
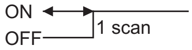
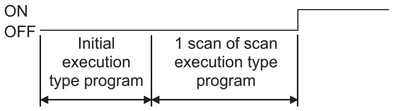
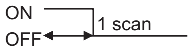
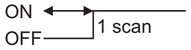
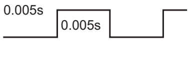
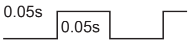
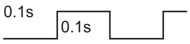
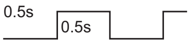
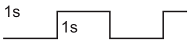
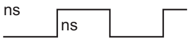
TableApp.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM322	SFC program start status	OFF : Initial start ON : Resume start	• The SFC program starting mode in the SFC setting of the PLC parameter dialog box is set as the initial value. AT initial start: OFF At continued start: ON	S (Initial)/U	M9102format change	QnA Q00J/Q00/Q01[*1] Qn(H) QnPH QnPRH QnU
SM323	Presence/absence of continuous transition for entire block	OFF : Continuous transition not effective ON : Continuous transition effective	Set the presence/absence of continuous transition for the block where "Continuous transition bit" of the SFC data device has not been set.	U	M9103	
SM324	Continuous transition prevention flag	OFF : When transition is executed ON : When no transition	• OFF during operation in the continuous transition mode or during continuous transition, and ON when continuous transition is not executed. • Always ON during operation in the no continuous transition mode.	S (Instruction execution)	M9104	
				S (Status change)	New	
SM325	Output mode at block stop	OFF : OFF ON : Preserves	Select whether the coil outputs of the active steps are held or not at the time of a block stop. • As the initial value, the output mode at a block stop in the parameter is OFF when the coil outputs are OFF, and ON when the coil outputs are held. • All coil outputs go OFF when this relay is OFF. • Coil outputs are preserved when this relay is ON.	S (Initial)/U	M9196	
SM326	SFC device clear mode	OFF : Clear device ON : Preserves device	Selects the device status when the stopped CPU is run after the sequence program or SFC program has been modified when the SFC program exists.	U	New	
SM327	Output during end step execution	OFF : Hold step output turned OFF (cleared) ON : Hold step output held	• Select the device status at the time of switching from STOP to program write to RUN.(All devices except the step relay)	S (Initial)/U	New	QnA Qn(H) QnPH QnPRH QnU
				U		Q00J/Q00/Q01[*1]
SM328	Clear processing mode when end step is reached	OFF : Clear processing is performed. ON : Clear processing is not performed.	Select whether clear processing will be performed or not if active steps other than the ones being held exist in the block when the end step is reached.? • When this relay turns OFF, all active steps are forcibly terminated to terminate the block. • When this relay is ON, the execution of the block is continued as-is. • If active steps other than the ones being held do not exist when the end step is reached, the steps being held are terminated to terminate the block.	U	New	Q00J/Q00/Q01[*1] QnU
SM330	Operation mode for low speed execution type program	OFF : Asynchronous mode ON : Synchronous mode	Select whether the low speed execution type program will be executed in the asynchronous mode or in the synchronous mode. • Asynchronous mode (this relay is turned OFF.) Mode in which the operation of the low speed execution type program is performed continuously within the excess time. • Synchronous mode (this relay is turned ON.) Mode in which the operation of the low speed execution type program is not performed continuously and operation is performed from the next scan if there is excess time.	U	New	QnA Qn(H) QnPH
SM331	Normal SFC program execution status	OFF : Not executed ON : Being executed	• Indicates whether the normal SFC program is being executed or not. • Used as an SFC control instruction execution interlock.	S (Status change)	New	Qn(H)[*3] QnPH[*4] QnPRH
SM332	Program execution management SFC program execution status	OFF : Not executed ON : Being executed	• Indicates whether the program execution management SFC program is being executed or not. • Used as an SFC control instruction execution interlock.			
SM390	Access execution flag	ON indicates completion of intelligent function module access	• The status of the intelligent function module access instruction executed immediately before is stored. (This data is overwritten when the intelligent function module access instruction is executed again.) • Used by the user in a program as a completion bit.	S (Status change)	New	Qn(H) QnPH QnPRH
SM391	GINT instruction execution completion flag	OFF : Not executed ON : Execution completed	Indicates execution status of the S(P).GINT instruction. • Turned OFF before the instruction is executed. • Turned ON after the instruction is completed.	S (Instruction execution)	New	QnU

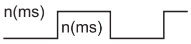
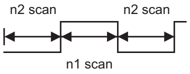
*1: This applies to the CPU of function version B or later.
*3: The module whose first 5 digits of serial No. is "04122" or later.
*4: The module whose first 5 digits of serial No. is "07032" or later.

(3) System clocks/counters

TableApp.4 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM400	Always ON		• Normally is ON	S (Every END processing)	M9036	○
SM401	Always OFF		• Normally is OFF	S (Every END processing)	M9037	
SM402	After RUN, ON for 1 scan only		• After RUN, ON for 1 scan only. • This connection can be used for scan execution type programs only. • When an initial execution type program is used, this relay turns OFF at the END processing of the scan execution type program in the first scan after RUN.  ON OFF Initial execution type program 1 scan of scan execution type program	S (Every END processing)	M9038	QnA Qn(H) QnPH QnPRH QnU
			• After RUN, ON for 1 scan only.	S (Every END processing)	New	Q00J/Q00/Q01
SM403	After RUN, OFF for 1 scan only		• After RUN, OFF for 1 scan only. • This connection can be used for scan execution type programs only. • When an initial execution type program is used, this relay turns OFF at the END processing of the scan execution type program in the first scan after RUN.  ON OFF Initial execution type program 1 scan of scan execution type program	S (Every END processing)	M9039	QnA Qn(H) QnPH QnPRH QnU
			• After RUN, OFF for 1 scan only.	S (Every END processing)	New	Q00J/Q00/Q01
SM404	Low speed execution type programON for 1 scan only after RUN		• After RUN, ON for 1 scan only. • This connection can be used for low speed execution type programs only.	S (Every END processing)	New	QnA Qn(H) QnPH
SM405	Low speed execution type programAfter RUN, OFF for 1 scan only		• After RUN, OFF for 1 scan only. • This connection can be used for low speed execution type programs only.	S (Every END processing)	New	
SM409	0.01 second clock		• Repeatedly changes between ON and OFF at 5-ms interval. • When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SM410	0.1 second clock		• Repeatedly changes between ON and OFF at each designated time interval. • When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)	S (Status change)	M9030	○
SM411	0.2 second clock				M9031	
SM412	1 second clock				M9032	
SM413	2 second clock				M9033	
SM414	2n second clock		• This relay alternates between ON and OFF at intervals of the time (unit: s) specified in SD414. • When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)	S (Status change)	M9034form at change	

TableApp.4 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM415	2n (ms) clock		• This relay alternates between ON and OFF at intervals of the time (unit: ms) specified in SD415. • When Programmable Controller power supply is turned OFF or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SM420	User timing clock No.0		• Relay repeats ON/OFF switching at fixed scan intervals. • When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (For the redundant CPU, however, this relay is always OFF after system switching.) • The ON/OFF intervals are set with the DUTY instruction DUTY n1 n2 SM420 n1: ON scan interval n2: OFF scan interval	S (Every END processing)	M9020	○
SM421	User timing clock No.1				M9021	
SM422	User timing clock No.2				M9022	
SM423	User timing clock No.3				M9023	
SM424	User timing clock No.4				M9024	
SM430	User timing clock No.5		• For use with SM420 to SM424 low speed programs	S (Every END processing)	New	QnA Qn(H) QnPH
SM431	User timing clock No.6					
SM432	User timing clock No.7					
SM433	User timing clock No.8					
SM434	User timing clock No.9					

(4) Scan information

TableApp.5 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM510	Low speed program execution flag	OFF : Completed or not executed ON : Execution under way.	• Goes ON when low speed execution type program is executed.	S (Every END processing)	New	QnA Qn(H) QnPH
SM551	Reads module service interval	OFF : Ignored ON : Read	• When this relay goes from OFF to ON, the module service interval designated by SD550 is read to SD551 to SD552.	U	New	QnA Qn(H) QnPH QnPRH Rem

(5) I/O refresh

TableApp.6 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM580	Program to program I/O refresh	OFF : Not refreshed ON : Refreshed	• When this special relay is turned ON, I/O refresh is performed after execution of the first program, and the next program is then executed. When a sequence program and an SFC program are to be executed, the sequence program is executed, I/O refresh is performed, and the SFC program is then executed.	U	New	Q00J/Q00/Q01[*1]

[*1] This applies to the CPU of function version B or later.

(6) Drive information

TableApp.7 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM600	Memory card (A) usable flags	OFF : Unusable ON : Use enabled	• ON when memory card (A) is ready for use by user	S (Status change)	New	QnA Qn(H) QnPH QnPRH QnU
SM601	Memory card (A) protect flag	OFF : No protect ON : Protect	• Goes ON when memory card (A) protect switch is ON	S (Status change)	New	
SM602	Drive 1 flag	OFF : No drive 1 ON : Drive 1 present	• Turns ON when the mounted memory card (A) is RAM	S (Status change)	New	
SM603	Drive 2 flag	OFF : No drive 2 ON : Drive 2 present	• Turns ON when the mounted memory card (A) is ROM	S (Status change)	New	
SM604	Memory card (A) in-use flag	OFF : Not used ON : In use	• Goes ON when memory card (A) is in use	S (Status change)	New	
SM605	Memory card (A) remove/insert prohibit flag	OFF : Remove/insert enabled ON : Remove/insert prohibited	• Goes ON when memory card (A) cannot be inserted or removed	U	New	
SM609	Memory card remove/insert enable flag	OFF : Remove/insert prohibited ON : Remove/insert enabled	• Turned ON by user to enable the removal/insertion of memory card. • Turned OFF by the system after the memory card is removed. • This contact can be used only when SM604 and SM605 are OFF.	S/U	New	Qn(H) QnPH QnPRH QnU
SM620	Drive 3/4 usable flag	OFF : Unusable ON : Use enabled	• Always ON	S (Initial)	New	QCPU
	Memory card B usable flags	OFF : Unusable ON : Use enabled	• ON when memory card B is ready for use by user	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SM621	Memory card B protect flag	OFF : No protect ON : Protect	• Goes ON when memory card B protect switch is ON	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SM622	Drive 3 flag	OFF : No drive 3 ON : Drive 3 present	• Always ON	S (Initial)	New	QCPU
			• Goes ON when drive 3 (card 2 RAM area) is present	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SM623	Drive 4 flag	OFF : No drive 4 ON : Drive 4 present	• Always ON	S (Initial)	New	QCPU
			• Goes ON when drive 4 (card 2 ROM area) is present	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SM624	Memory card B in-use flag	OFF : Not used ON : In use	• Goes ON when memory card B is in use	S (Status change)	New	Q2A(S1) Q3A Q4A Q4AR
	Drive 3/4 in-use flag	OFF : Not used ON : In use	• Goes ON when the file within Drive 3 (standard RAM) or Drive 4 (standard ROM) is used.	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SM625	Memory card B remove/insert prohibit flag	OFF : Remove/insert enabled ON : Remove/insert prohibited	• Goes ON when memory card B cannot be inserted or removed	U	New	Q2A(S1) Q3A Q4A Q4AR
SM640	File register use	OFF : File register not used ON : File register in use	• Goes ON when file register is in use	S (Status change)	New	○

TableApp.7 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM650	Comment use	OFF : File register not used ON : File register in use	• Goes ON when comment file is in use	S (Status change)	New	QnA Qn(H) QnPH QnPRH QnU
SM660	Boot operation	OFF : Internal memory execution ON : Boot operation in progress	• Goes ON while boot operation is in process • Goes OFF if boot designation switch is OFF	S (Status change)	New	QnA Qn(H) QnPH QnPRH
		OFF : Program memory execution ON : Boot operation in progress	• Goes ON while boot operation is in process	S (Status change)	New	Q00J/Q00/Q01 QnU
SM671	Latch data backup to standard ROM completion flag	OFF : Not completed ON : Completed	• Turned ON when latch data backup to the standard ROM is completed. • Time when the latch data backup to the standard ROM was performed is stored in SD672 or later.	S (Status change)	New	QnU
SM672	Memory card A file register access range flag	OFF : Within access range ON : Outside access range	• Goes ON when access is made to area outside the range of file register R of memory card A(Set within END processing.) • Reset at user program	S/U	New	QnA Qn(H) QnPH QnPRH
SM673	Memory card B file register access range flag	OFF : Within access range ON : Outside access range	• Goes ON when access is made outside the range of file registers, R. of memory card B.(Set within END processing.) • Reset at user program	S/U	New	Q2A(S1) Q3A Q4A Q4AR
SM675	Error completion of latch data backup to standard ROM	OFF : No Error ON : Error	• Turned ON when data cannot be backuped to the standard ROM by the latch data backup normally. • Turned OFF when data is backuped to the standard ROM by the latch data backup normally.	S	New	QnU
SM676	Specification of restore repeated execution	OFF : Not specified ON : Specified	• If latch data backup is performed when SM676 is ON, restore the data every time turning ON from OFF the power supply from the next power-on. • Delete the backuped latch data, or restore the data every time turning ON from OFF the power supply until the latch data backup operation will be executed again.	U	New	QnU
SM680	Program memory write error	OFF : Write error ON : Write not executed/ normal	• Turns ON if a write error is detected at writing to program memory (flash ROM). Turns OFF by the write direction.	S (At write)	New	QnU
SM681	Program memory writing flag	OFF : During writing ON : Write not executed	• Turns ON when writing to the program memory (flash ROM) is in progress, and turns OFF when writing is completed.	S (At write)	New	QnU
SM682	Program memory overwrite count error flag	OFF : Overwrite count is 100,000 or more ON : Overwrite count is less than 100,000	• Turns ON when the overwrite count of program memory (flash ROM) reaches 100,000.	S (At write)	New	QnU
SM685	Standard ROM write error	OFF : Write error ON : Write not executed/ normal	• Turns ON when write error is detected at writing to standard ROM (flash ROM). • Turns OFF by the write direction.	S (At write)	New	QnU
SM686	Standard ROM writing flag	OFF : During overwriting ON : Overwrite not executed	• Turns ON when writing to the standard ROM (flash ROM) is in progress, and turns OFF when writing is completed.	S (At write)	New	QnU
SM687	Standard ROM overwrite count error flag	OFF : Overwrite count is 100,000 or more ON : Overwrite count is less than 100,000	• Turns ON when the overwrite count of standard ROM (flash ROM) reaches 100,000. (It is necessary to change CPU module.)	S (At write)	New	QnU

(7) Instruction-Related Special Relays

TableApp.8 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM700	Carry flag	OFF : Carry OFF ON : Carry ON	• Carry flag used in application instruction	S (Instruction execution)	M9012	○
SM701	Number of output characters selection	Switching the number of output characters and the output pattern	• Used for the PR, PRC, BINDA, DBINDA, BINHA, DBINHA, BCDDA, DBCDDA, or COMRD instruction	U	M9049	QnA Qn(H) QnPH QnPRH QnU
SM702	Search method	OFF : Search next ON : 2-part search	• Designates method to be used by search instruction. • Data must be arranged for 2-part search.	U	New	○
SM703	Sort order	OFF : Ascending order ON : Descending order	• The sort instruction is used to designate whether data should be sorted in ascending order or in descending order.	U	New	
SM704	Block comparison	OFF : Non-match found ON : All match	• Goes ON when all data conditions have been met for the BKCMP instruction.	S (Instruction execution)	New	
SM707	Selection of real number instruction processing type	OFF : Speed oriented ON : Accuracy oriented	• When SM707 is OFF, real number instructions are processed at high speed. • When it is ON, real number instructions are processed with high accuracy.	U	New	Q4AR
SM710	CHK instruction priority ranking flag	OFF : Conditions priority ON : Pattern priority	• Remains as originally set when OFF. • CHK priorities updated when ON.	S (Instruction execution)	New	QnA Qn(H) QnPH QnPRH
SM711	Divided transmission status	OFF : Other than during divided processing ON : During divided processing	• In processing of AD57(S1), goes ON when screen is split for transfer, and goes OFF when split processing is completed	S (Instruction execution)	M9065	QnA
SM712	Transmission processing selection	OFF : Batch processing ON : Divided processing	• In processing of AD57(S1), goes ON when canvas screen is divided for transfer.	S (Instruction execution)	M9066	
SM714	Communication request registration area BUSY signal	OFF : Communication request to remote terminal module enabled ON : Communication request to remote terminal module disabled	• Used to determine whether communications requests to remote terminal modules connected to the AJ71PT32-S3 can be executed or not.	S (Instruction execution)	M9081	
SM715	EI flag	OFF : During DI ON : During EI	• ON when EI instruction is being executed.	S (Instruction execution)	New	○
SM720	Comment read completion flag	OFF : Comment read not completed ON : Comment read completed	• Turns on only during one scan when the processing of the COMRD or PRC instruction is completed.	S (Status change)	New	Qn(H) QnPH
			• Turns on only during one scan when the processing of the COMRD instruction is completed.			QnPRH QnU
SM721	File being accessed	OFF : File not accessed ON : File being accessed	• Switches ON while a file is being accessed by the S. FWRITE, S. FREAD, COMRD, PRC, or LEDC instruction.	S (Status change)	New	Qn(H) QnPH
			• Switches ON while a file is being accessed by the S. FWRITE, S. FREAD, COMRD or LEDC instruction.			Qn(H) QnPH QnPRH
			• Switches ON while a file is being accessed by the S. FWRITE, S. FREAD, COMRD or SP.DEVST instruction.			QnU
			• Turns ON while the ATA card or standard ROM is being accessed.			QnU[*2]
SM722	BIN/DBIN instruction error disabling flag	OFF : Error detection performed ON : Error detection not performed	• Turned ON when "OPERATION ERROR" is suppressed for BIN or DBIN instruction.	U	New	QCPU
SM730	BUSY signal for CC-Link communication request registration area	OFF : Request for communication with intelligent device station enabled ON : Request for communication with intelligent device station disabled	• Used for determination whether to enable or disable the communication request for the intelligent device station connected with CC-Link module.	S (Instruction execution)	New	QnA
SM734	XCALL instruction execution condition designation	OFF : Not executed by execution condition risen ON : Executed by execution condition risen	• During OFF, XCALL instructions will not be executed even if execution condition is risen. • During ON, XCALL instructions will be executed when execution condition is risen.	U	New	Qn(H)[*1]

[*1]: The module whose first 5 digits of serial No. is "06082" or later.

[*2]: The module whose first 5 digits of serial No. is "09072" or later.

TableApp.8 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM735	SFC comment readout instruction in execution flag	OFF : SFC comment readout instruction is inactivated. ON : SFC comment readout instruction is activating.	• Turns on the instructions, (S(P).SFCSCOMR) to read the SFC step comments and (S(P). SFCTCOMR) to read the SFC transition condition comments.	S (status change)	New	Qn(H)*2 QnPH*3 QnPRH*3
SM736	PKEY instruction execution in progress flag	OFF : Instruction not executed ON : Instruction execution	• ON when PKEY instruction is being executed. Goes OFF when CR is input, or when input character string reaches 32 characters.	S (Instruction execution)	New	QnA
SM737	Keyboard input reception flag for PKEY instruction	OFF : Keyboard input reception enabled ON : Keyboard input reception disabled	• Goes ON when keyboard input is being conducted. Goes when keyboard input has been stored at the CPU.	S (Instruction execution)	New	
SM738	MSG instruction reception flag	OFF : Instruction not executed ON : Instruction execution	• Goes ON when MSG instruction is executed	S (Instruction execution)	New	Qn(H) QnPRH QnA
SM774	PID bumpless processing (for complete derivative)	OFF : Matched ON : Not matched	• Specifies whether to match the set value (SV) with the process value (PV) or not in the manual mode.	U	New	QnA Q00J/Q00/Q01*4 Qn(H) QnPRH QnU
SM775	Selection of refresh processing during COM instruction execution	OFF : Performs link refresh ON : Performs no link refresh	• Select whether link refresh processing will be performed or not when only communication with the CPU module is made at the execution of the COM instruction.	U	New	QnA Q00J/Q00/Q01 Qn(H) QnPH
		OFF : Performs all refresh processes ON : Performs refresh set by SD778	• Select whether to perform all refreshes or the refresh processing set by SD778 when the COM instruction is executed.	U	New	Q00J/Q00/Q01*4 Qn(H)*5 QnPH*3 QnPRH QnU
SM776	Enable/disable local device at CALL	OFF : Local device disabled ON : Local device enabled	• Set whether the local device of the subroutine program called at execution of the CALL instruction is valid or invalid.	U	New	QnA Qn(H) QnPH QnPRH QnU
SM777	Enable/disable local device in interrupt program	OFF : Local device disabled ON : Local device enabled	• Set whether the local device at execution of the interrupt program is valid or invalid.	U	New	
SM780	CC-Link dedicated instruction executable	OFF : CC-Link dedicated instruction executable ON : CC-Link dedicated instruction not executable	• Switches ON when the number of the CC-Link dedicated instructions that can be executed simultaneously reaches 32. Switches OFF when the number goes below 32.	U	New	QnA
SM794	PID bumpless processing(for incomplete derivative)	OFF : Matched ON : Not matched	• Specifies whether to match the set value (SV) with the process value (PV) or not in the manual mode.	U	New	Q00J/Q00/Q01*4 Qn(H)*6 QnPRH QnU

*1: The module whose first 5 digits of serial No. is "06082" or later.
*2: The module whose first 5 digits of serial No. is "07012" or later.
*3: The module whose first 5 digits of serial No. is "07032" or later.
*4: This applies to the CPU module of function version B or later.
*5: The module whose first 5 digits of serial No. is "04012" or later.
*6: The module whose first 5 digits of serial No. is "05032" or later.

TableApp.8 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM796	Block information using dedicated instruction of Multiple CPU high speed transmission (for CPU No.1)	OFF : Block is secured ON : Block set by SD796 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.1) is less than the number of blocks specified by SD796. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	QnU
SM797	Block information using dedicated instruction of Multiple CPU high speed transmission (for CPU No.2)	OFF : Block is secured ON : Block set by SD797 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.2) is less than the number of blocks specified by SD797. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	QnU
SM798	Block information using dedicated instruction of Multiple CPU high speed transmission (for CPU No.3)	OFF : Block is secured ON : Block set by SD798 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.3) is less than the number of blocks specified by SD798. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	QnU
SM799	Block information using dedicated instruction of Multiple CPU high speed transmission (for CPU No.4)	OFF : Block is secured ON : Block set by SD799 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.4) is less than the number of blocks specified by SD799. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	QnU*7

*7: The Universal model QCPU except the Q02UCPU.

(8) Debug

TableApp.9 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM800	Trace preparation	OFF : Not ready ON : Ready	• Switches ON when the trace preparation is completed	S (Status change)	New	Qn(H) QnPH QnPRH QnU
	Sampling trace preparation		• Goes ON when sampling trace is ready	S (Status change)	New	QnA
SM801	Trace start	OFF : Suspend ON : Start	• Trace is started when this relay switches ON. • Trace is suspended when this relay switches OFF. (All related special Ms switches OFF.)	U	M9047	Qn(H) QnPH QnPRH QnU
	Sampling trace start		• Sampling trace started when this goes ON • Suspended when OFF (Related special M all OFF)	U	M9047	QnA
SM802	Trace execution in progress	OFF : Suspend ON : Start	• Switches ON during execution of trace.	S (Status change)	M9046	Qn(H) QnPH QnPRH QnU
	Sampling trace execution in progress		• Goes ON during execution of sampling trace	S (Status change)	M9046	QnA
SM803	Trace trigger	OFF → ON: Start	• Trace is triggered when this relay switches from OFF to ON. (Identical to TRACE instruction execution status)	U	M9044	Qn(H) QnPH QnPRH QnU
	Sampling trace trigger		• Sampling trace trigger goes ON when this goes from OFF to ON (Identical to STRA instruction execution status)	U	M9044	QnA
SM804	After trace trigger	OFF : Not after trigger ON : After trigger	• Switches ON after trace is triggered.	S (Status change)	New	Qn(H) QnPH QnPRH QnU
	After sampling trace trigger		• Goes ON after sampling trace trigger	S (Status change)	New	QnA
SM805	Trace completed	OFF : Not completed ON : End	• Switches ON at completion of trace	S (Status change)	M9043	Qn(H) QnPH QnPRH QnU
	Sampling trace completed		• Goes ON at completion of sampling trace	S (Status change)	M9043	QnA
SM806	Status latch preparation	OFF : Not ready ON : Ready	• Goes ON when status latch is ready	S (Status change)	New	
SM807	Status latch command	OFF → ON: Latch	• Runs status latch command	U	New	
SM808	Status latch completion	OFF : Latch not completed ON : Latch completed	• Comes ON when status latch is completed.	S (Status change)	M9055	
SM809	Status latch clear	OFF → ON: Clear	• Enable next status latch	U	New	
SM810	Program trace preparation	OFF : Not ready ON : Ready	• Goes ON when program trace is ready	S (Status change)	New	
SM811	Start program trace	OFF : Suspend ON : Start	• Program trace started when this goes ON • Suspended when OFF (Related special M all OFF)	S (Status change)	New	
SM812	Program trace execution under way	OFF : Suspend ON : Start	• ON when program trace execution is underway	U	New	
SM813	Program trace trigger	OFF → ON: Start	• Program trace trigger goes ON when this goes from OFF to ON (Identical to PTRA instruction execution status)	S (Status change)	New	
SM814	After program trace trigger	OFF : Not after trigger ON : After trigger	• Goes ON after program trace trigger	S (Status change)	New	
SM815	Program trace completion	OFF : Not completed ON : End	• Goes ON at completion of program trace	S (Status change)	New	

TableApp.9 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM820	Step trace preparation	OFF : Not ready ON : Ready	• Goes ON after program trace registration, at ready	S (Status change)	New	QnA
SM821	Step trace starts	OFF : Suspend ON : Start	• Select whether execution of step trace is started or suspended. • When this goes ON, step trace is started • Suspended when OFF (Related special M all OFF)	U	M9182form at change	
SM822	Step trace execution underway	OFF : Suspend ON : Start	• Goes ON when step trace execution is underway • Goes OFF at completion or suspension	S (Status change)	M9181	
SM823	After step trace trigger	OFF : Not after trigger ON : Is after first trigger	• Goes ON if even 1 block within the step trace being executed is triggered. • Goes OFF when step trace is commenced.	S (Status change)	New	
SM824	After Step trace trigger	OFF : Is not after all triggers ON : Is after all triggers	• Goes ON if all blocks within the step trace being executed are triggered. • Goes OFF when step trace is commenced.	S (Status change)	New	
SM825	Step trace completed	OFF : Not completed ON : End	• Goes ON at step trace completion. • Goes OFF when step trace is commenced.	S (Status change)	M9180	
SM826	Trace error	OFF : Normal ON : Errors	• Switches ON if error occurs during execution of trace	S (Status change)	New	Qn(H) QnPH QnPRH QnU
	Sampling trace error		• Goes ON if error occurs during execution of sampling trace.	S (Status change)	New	QnA
SM827	Status latch error	OFF : Normal ON : Errors	• Goes ON if error occurs during execution of status latch.	S (Status change)	New	
SM828	Program trace error	OFF : Normal ON : Errors	• Goes ON if error occurs during execution of program trace.	S (Status change)	New	
SM829	Forced registration specification of trace setting	ON : Forced registration enabled OFF : Forced registration disabled	• Even when the trigger condition is established, the sampling trace setting can be set to the QnUCPU by turning SM829 ON and registering the sampling trace setting by GX Developer.	U	New	QnU

(9) Latch area

TableApp.10 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM900	Power off file	OFF : No power off file ON : Power off file present	• Goes ON if a file is present during access when power is interrupted.	S (Status change)/U	New	QnA
SM910	RKEY registration flag	OFF : Keyboard input notregistered ON : Keyboard input registered	• Goes ON at registration of keyboard input. • OFF if keyboard input is not registered.	S (Instruction execution)	New	

(10)A to Q/QnA conversion correspondences

Special relays SM1000 to SM1255 are the relays which correspond to ACPU special relays M9000 to M9255 after A to Q/QnA conversion.

(However, the Basic model QCPU, Redundant CPU, and Universal model QCPU do not support the A to Q/QnA conversion.)

These special relays are all set by the system, and cannot be set by the user program.

To turn them ON/OFF by the user program, change the special relays in the program into those of QCPU/QnACPU.

However, some of SM1084 and SM1200 to SM1255 (corresponding to M9084 and M9200 to M9255 before conversion) can be turned ON/OFF by the user program, if they could be turned ON/OFF by the user program before conversion.For details on the ACPU special relays, see the user's manuals for the individual CPUs, and MELSECNET or MELSECNET/B Data Link System Reference Manuals

POINT

The processing time may be longer when converted special relays are used with the High Performance model QCPU and Process CPU. Uncheck "Compatibility with A-PLC" within the PLC system setting in GX Developer parameters when converted special relays are not used.

Remark

The following are additional explanations about the Special Relay for Modification column.

① When a special relay for modification is provided, the device number should be changed to the provided QCPU/QnACPU special relay.

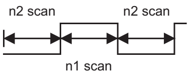
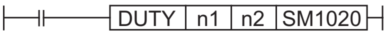
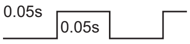
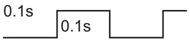
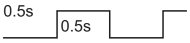
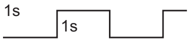
② When ⊟ is provided, the converted special relay can be used for the device number.

③ When ☒ is provided, the device number does not work with QCPU/QnACPU.

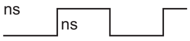
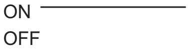
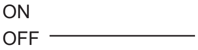
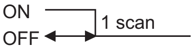
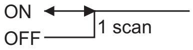
TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9000	SM1000	–	Fuse blown	OFF : Normal ON : Module with blown fuse	• Turned on when there is one or more output modules of which fuse has been blown. • Remains ON if the condition is restored to normal thereafter. • Output modules of remote I/O stations are also checked fore fuse condition.	QnA Qn(H) QnPH
M9002	SM1002	–	I/O module verify error	OFF : Normal ON : Error	• Turned on if the status of I/O module is different form entered status when power is turned on. • Remains ON if the condition is restored to normal thereafter. • I/O module verification is done also to remote I/O station modules. • Reset is enabled only when special registers SD1116 to SD1123 are reset.	
M9004	SM1004	–	NIMI link master module error	OFF : Normal ON : Error	• Goes ON if MINI (S3) link error is detected at even one of the installed AJ71PT32 (S3) modules. • Remains ON if the condition is restored to normal thereafter.	QnA

TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9005	SM1005	–	AC DOWN detection	OFF : AC DOWN not detected ON : AC DOWN detected	• Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module. • Reset when the power supply is switched OFF, then ON.	QnA Qn(H) QnPH
					• Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module. • Reset when the power supply is switched OFF, then ON.	Qn(H) QnPH
					• Turns ON if an instantaneous power failure of within 1ms occurs during use of the DC power supply module. • Reset when the power supply is switched OFF, then ON.	QnA
M9006	SM1006	–	Battery low	OFF : Normal ON : Battery low	• Turns ON when the battery voltage drops to or below the specified. • Turns OFF when the battery voltage returns to normal thereafter.	QnA Qn(H) QnPH
M9007	SM1007	–	Battery low latch	OFF : Normal ON : Battery low	• Turns ON when the battery voltage drops to or below the specified. • Remains ON if the battery voltage returns to normal thereafter.	
M9008	SM1008	SM1	Self-diagnosis error	OFF : No error ON : Error	• Turned on when error is found as a result of self-diagnosis.	
M9009	SM1009	SM62	Annunciator detection	OFF : No F number detected ON : F number detected	• Turned on when OUT F of SET F instruction is executed. • Switched off when SD1124 data is cleared to zero.	
M9011	SM1011	SM56	Operation error flag	OFF : No error ON : Error	• Turned on when operation error occurs during execution of application instruction. • Remains ON if the condition is restored to normal thereafter.	
M9012	SM1012	SM700	Carry flag	OFF : Carry OFF ON : Carry ON	• Carry flag used in application instruction.	
M9016	SM1016	×	Data memory clear flag	OFF : Ignored ON : Output claered	• Clears the data memory including the latch range (other than special relays and special registers) in remote run mode from computer, etc. when SM1016 is on.	Qn(H) QnPH
M9017	SM1017	×	Data memory clear flag	OFF : Ignored ON : Output claered	• Clears the unlatched data memory (other than special relays and special egisters) in remote run mode from computer, etc. when SM1017 is on.	Qn(H) QnPH
M9020	SM1020	–	User timing clock No.0		• Relay which repeats on/off at intervals of predetermined scan. • When power is turned on or reset is per-formed, the clock starts with off. Set the intervals of on/off by DUTY instruction.  n1: ON scan interval n2: OFF scan interval	QnA Qn(H) QnPH
M9021	SM1021	–	User timing clock No.1			
M9022	SM1022	–	User timing clock No.2			
M9023	SM1023	–	User timing clock No.3			
M9024	SM1024	–	User timing clock No.4			
M9025	SM1025	–	Clock data set request	OFF : Ignored ON : Set request present used	• Writes the clock data stored in SD1025 to SD1028 to the CPU module after the END instruction is executed in the scan in which SM1025 turned from OFF to ON.	
M9026	SM1026	–	Clock data error	OFF : No error ON : Error	• Switched on by clock data (SD1025 to SD1028) error	
M9027	SM1027	–	Clock data display	OFF : Ignored ON : Display	• Clock data is read from SD1025 to SD1028 and month, day, hour, minute and minute are indicated on the CPU module front LED display.	Q3A Q4A Q4AR
M9028	SM1028	–	Clock data read request	OFF : Ignored ON : Read request	• Reads clock data to SD1025 to SD1028 in BCD when SD1028 is on.	QnA Qn(H) QnPH
M9029	SM1029	×	Batch processing of data communications requests	OFF : Batch processing not conducted ON : Batch processing conducted	• The SM1029 relay is turned on using a sequence program to process all data communication requests accepted during one scan in the END processing of that scan. • The batch processing of the data communication requests can be turned on and off during running. • The default is OFF (processed one at a time for each END processing in the order in which data communication requests are accepted).	
M9030	SM1030	–	0.1 second clock		• 0.1 second, 0.2 second, 1 second and 2 second, clocks are generated. • Not turned on or off per scan but turned on and off even during scan if corresponding time has elapsed. • Starts with off when Programmable Controller power supply is turned on or CPU module reset is performed.	
M9031	SM1031	–	0.2 second clock			
M9032	SM1032	–	1 second clock			
M9033	SM1033	–	2 second clock			

TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9034	SM1034	–	2n minute clock(1 minute clock)*		• Alternates between ON and OFF according to the seconds specified at SD414. (Default: n = 30) • Not turned on or off per scan but turned on and off even during scan if corresponding time has elapsed. • Starts with off when Programmable Controller power supply is turned on or CPU module reset is performed.	QnA Qn(H) QnPH
M9036	SM1036	–	Always ON		• Used as dummy contacts of initialization and application instruction in sequence program. • SM1038 and SM1037 are turned on and off without regard to position of key switch on CPU module front. SM1038 and SM1039 are under the same condition as RUN status except when the key switch is at STOP position, and turned off and on. Switched off if the key switch is in STOP position. SM1038 is on for one scan only and SM1039 is off for one scan only if the key switch is not in STOP position.	
M9037	SM1037	–	Always OFF			
M9038	SM1038	–	ON for 1 scan only after RUN			
M9039	SM1039	–	RUN flag(After RUN, OFF for 1 scan only)			
M9040	SM1040	SM206	PAUSE enable coil	OFF : PAUSE disabled ON : PAUSE enabled	• When RUN key switch is at PAUSE position or pause contact has turned on and if SM204 is on, PAUSE mode is set and SM206 is turned on.	
M9041	SM1041	SM204	PAUSE status contact	OFF : PAUSE not in effect ON : PAUSE in effect		
M9042	SM1042	SM203	STOP status contact	OFF : STOP not in effect ON : STOP in effect	• Switched on when the RUN key switch or RUN/STOP switch is in STOP position.	
M9043	SM1043	SM805	Sampling trace completed	OFF : Sampling trace in progress ON : Sampling trace completed	• Turned on upon completion of sampling trace performed the number of times preset by parameter after $\boxed{\text{STRA}}$ instruction is executed. Reset when $\boxed{\text{STRAR}}$ instruction is executed.	
M9044	SM1044	SM803	Sampling trace	OFF→ON $\boxed{\text{STRA}}$ Same as execution ON→OFF $\boxed{\text{STRAR}}$ Same as execution	• Turning on/off SM803 can execute $\boxed{\text{STRA}}$ / $\boxed{\text{STRAR}}$ instruction. (SM803 is forcibly turned on/off by a peripheral device.) When switched from OFF to ON: $\boxed{\text{STRA}}$ instruction When switched from ON to OFF: $\boxed{\text{STRAR}}$ instruction The value stored in SD1044 is used as the condition for the sampling trace. At scanning, at time → Time (10 ms unit)	
M9045	SM1045	×	Watchdog timer (WDT) reset	OFF : Does not reset WDT ON : Resets WDT	• The SM1015 relay is turned on to reset the WDT when the ZCOM instruction and data communication request batch processing are executed (used when the scan time exceeds 200 ms).	
M9046	SM1046	SM802	Sampling trace	OFF : Trace not in progress ON : Trace in progress	• Switched on during sampling trace.	
M9047	SM1047	SM801	Sampling trace preparations	OFF : Sampling trace suspended ON : Sampling trace started	• Sampling trace is not executed unless SM801 is turned ON. Sampling trace is suspended when SM801 goes OFF.	
M9049	SM1049	SM701	Switching the number of output characters	OFF : Output until NULL code encountered ON : 16 characters output	• When SM701 is OFF, characters up to NULL (00H) code are output. • When SM701 is ON, ASCII codes of 16 characters are output.	
M9051	SM1051	×	CHG instruction execution disable	OFF : Enabled ON : Disable	• Switched ON to disable the CHG instruction. • Switched ON when program transfer is requested. Automatically switched OFF when transfer is complete.	
M9052	SM1052	×	SEG instruction switch	OFF : 7SEG segment display ON : I/O partial refresh	• When SM1052 is ON, the SEG instruction is executed as an I/O partial refresh instruction. When SM1052 is OFF, the SEG instruction is executed as a 7-SEG display instruction.	
M9054	SM1054	SM205	STEP RUN flag	OFF : STEP RUN not in effect ON : STEP RUN in effect	• Switched on when the RUN key switch is in STEP RUN position.	QnA
M9055	SM1055	SM808	Status latch completion flag	OFF : Not completed ON : Completed	• Turned on when status latch is completed. • Turned off by reset instruction.	

*: 1 minute clock indicates the name of the special relay (M9034) of the ACPU.

TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9056	SM1056	×	Main side P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested	• Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	QnA Qn(H) QnPH
M9057	SM1057	×	Sub side P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested		
M9058	SM1058	×	Main side P, I set completion	Momentarily ON at P, I set completion	• Turned ON once when the P, I set has been completed, and then turned OFF again.	
M9059	SM1059	×	Sub program P, I set completion	Momentarily ON at P, I set completion		
M9060	SM1060	×	Sub program 2 P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested	• Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	
M9061	SM1061	×	Sub program 3 P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested		
M9065	SM1065	SM711	Divided transfer status	OFF : Divided processing not underway ON : During divided processing	• Turned on when canvas screen transfer to AD57(S1)/AD58 is done by divided processing, and turned off at completion of divided processing	QnA
M9066	SM1066	SM712	Transfer processing switch	OFF : Batch transfer ON : Divided transfer	• Turned on when canvas screen transfer to AD57(S1)/AD58 is done by divided processing.	
M9070	SM1070	×	A8UPU/A8PUJrequired search time*	OFF : Read time not shortened ON : Read time shortened	• Turned ON to shorten the search time in the A8UPU/A8PUJ. (In this case, the scan time is extended by 10 %.)	QnA Qn(H) QnPH
M9081	SM1081	SM714	Communication request registration area BUSY signal	OFF : Empty spaces in communication request registration area ON : No empty spaces in communication request registration area	• Indication of communication enable/disable to remote terminal modules connected to the AJ71PT32-S3, A2C or A52G.	QnA
M9084	SM1084	×	Error check	OFF : Error check executed ON : No error check	It is set whether the error checks below are performed or not when the END instruction is processed (to set the END instruction processing time). • Check for fuse blown. • Check of battery • Collation check of I/O module	QnA Qn(H) QnPH
M9091	SM1091	×	Operation error details flag	OFF : No error ON : Error	• Turns ON when the detail factor of the operation error is stored into SD1091. • Remains ON if the condition is restored to normal thereafter.	
M9094	SM1094	SM251	I/O exchange flag	OFF : Exchanged ON : Not exchanged	• The I/O module can be changed online (with power on) when SM251 is turned ON after the head I/O number of the I/O module is set to SD251. (One module only is allowed to be changed by one setting.) • To be switched on in the program or peripheral device test mode to change the module during CPU RUN. To be switched on in peripheral device test mode to change the module during CPU STOP. • RUN/STOP mode must not be changed until I/O module change is complete.	QnA
M9100	SM1100	SM320	Presence/absence of SFC program	OFF : SFC programs not used ON : SFC programs used	• Turned on if the SFC program is registered. • Turned off if the SFC program is not registered.	QnA Qn(H) QnPH
M9101	SM1101	SM321	Start/stop SFC program	OFF : SFC programs stop ON : SFC programs start	• The value in SM320 is set as the initial value. (The relay automatically turns ON when the SFC program is present.) • When this relay turns from ON to OFF, execution of the SFC program stops. • When this relay turns from OFF to ON, execution of the SFC program resumes.	
M9102	SM1102	SM322	SFC program start status	OFF : Initial start ON : Resume start	• The SFC program start mode in the SFC setting of the PLC parameter dialog box is set as the initial value. At initial start: OFF At continue start: ON	

*1: The A8UPU/A8PUJ is not available for the QCPU/QnACPU.

TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9103	SM1103	SM323	Presence/absence of continuous transition	OFF : Continuous transition not effective ON : Continuous transition effective	• Set whether continuous transition will be performed for the block where the "continuous transition bit" of the SFC information device is not set.	QnA Qn(H) QnPH
M9104	SM1104	SM324	Continuous transition suspension flag	OFF : When transition is completed ON : When no transition	• OFF during operation in the continuous transition mode or during continuous transition, and ON when continuous transition is not executed. • Always ON during operation in the no continuous transition mode.	
M9108	SM1108	SM90	Step transition monitoring timer start (equivalent of SD90)	OFF : Monitoring timer reset ON : Monitoring timer reset start	• Turns ON when the measurement of the step transition monitoring timer is started. Turning this relay OFF resets the step transition monitoring timer.	
M9109	SM1109	SM91	Step transition monitoring timer start (equivalent of SD91)			
M9110	SM1110	SM92	Step transition monitoring timer start (equivalent of SD92)			
M9111	SM1111	SM93	Step transition monitoring timer start (equivalent of SD93)			
M9112	SM1112	SM94	Step transition monitoring timer start (equivalent of SD94)			
M9113	SM1113	SM95	Step transition monitoring timer start (equivalent of SD95)			
M9114	SM1114	SM96	Step transition monitoring timer start (equivalent of SD96)			
M9180	SM1180	SM825	Active step sampling trace completion flag	OFF : Trace started ON : Trace completed	• Set when sampling trace of all specified blocks is completed. Reset when sampling trace is started.	QnA
M9181	SM1181	SM822	Active step sampling trace execution flag	OFF : Trace not being executed ON : Trace execution under way	• Set when sampling trace is being executed.Reset when sampling trace is completed or suspended	
M9182	SM1182	SM821	Active step sampling trace permission	OFF : Trace disable/suspend ON : Trace enable	• Selects sampling trace execution enable/disable. ON : Sampling trace execution is enabled. OFF : Sampling trace execution is disabled. If turned off during sampling trace execution, trace is suspended.	
M9196	SM1196	SM325	Operation output at block stop	OFF : Coil output OFF ON : Coil output ON	• Selects the operation output when block stop is executed. ON : Retains the ON/OFF status of the coil being used by using operation output of the step being executed at block stop. OFF : All coil outputs are turned off. (Operation output by the SET instruction is retained regardless of the ON/OFF status of M9196.)	QnA Qn(H) QnPH
M9197	SM1197	×	Switch between blown fuse and I/O verify error display	SM 1197 · SM 1198 · I/O numbers to be displayed OFF · OFF · X/Y0 to 7F0 ON · OFF · X/Y800 to FF0	Switches I/O numbers in the fuse blow module storage registers (SD1100 to SD1107) and I/O module verify error storage registers (SD1116 to SD1123) according to the combination of ON/OFF of the SM1197 and SM1198.	
M9198	SM1198	×		OFF · ON · X/Y1000 to 17F0 ON · ON · X/Y1800 to 1FF0		
M9199	SM1199	×	Data recovery of online sampling trace/status latch	OFF : Data recovery disabled ON : Data recovery enabled	• Recovers the setting data stored in the CPU module at restart when sampling trace/status latch is executed. • SM1199 should be ON to execute again. (Unnecessary when writing the data again from peripheral devices.)	

TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9200	SM1200	–	ZNRD instruction (LRDP instruction for ACPU) reception (for master station)	OFF : Not accepted ON : Accepted	• Depends on whether or not the ZNRD (word device read) instruction has been received. • Used in the program as an interlock for the ZNRD instruction. • Use the RST instruction to reset.	QnA
M9201	SM1201	–	ZNRD instruction (LRDP instruction for ACPU) completion (for master station)	OFF : Not completed ON : End	• Depends on whether or not the ZNRD (word device read) instruction execution is complete. • Used as a condition contact for resetting M9200 and M9201 after the ZNRD instruction is complete. • Use the RST instruction to reset.	
M9202	SM1202	–	ZNWR instruction (LWTP instruction for ACPU) reception (for master station)	OFF : Not accepted ON : Accepted	• Depends on whether or not the ZNWR (word device write) instruction has been received. • Used in the program as an interlock for the ZNWR instruction. • Use the RST instruction to reset.	
M9203	SM1203	–	ZNWR instruction (LWTP instruction for ACPU) completion (for master station)	OFF : Not completed ON : End	• Depends on whether or not the ZNWR (word device write) instruction execution is complete. • Used as a condition contact to reset M9202 and M9203 after the ZNWR instruction is complete. • Use the RST instruction to reset.	
M9204	SM1204	–	ZNRD instruction (LRDP instruction for ACPU) reception (for local station)	OFF : Not completed ON : End	• On indicates that the ZNRD instruction is complete at the local station.	
M9205	SM1205	–	ZNWR instruction (LWTP instruction for ACPU) reception (for local station)	OFF : Not completed ON : End	• On indicates that the ZNWR instruction is complete at the local station.	
M9206	SM1206	–	Host station link parameter error	OFF : Normal ON : Abnormal	• Depends on whether or not the link parameter setting of the host is valid.	
M9207	SM1207	–	Link parameter check results	OFF : Match ON : Mismatch	• Depends on whether or not the link parameter setting of the master station in tier two matches that of the master station in tier three in a three-tier system. (Valid for only the master station in a three-tier system.)	
M9208	SM1208	–	Sets master station B and W transmission range (for lower link master stations only)	OFF : Transmits to tier2 and tier 3 ON : Transmits to tier2 only	• Depends on whether or not the B and W data controlled by higher-link master station (host station) is sent to lower-link local stations (tertiary stations). When SM1208 is OFF............ B and W of host station is sent to tertiary stations. When SM1208 is ON .... B and W of host station is not sent to tertiary stations.	
M9209	SM1209	–	Link parameter check command (for lower link master stations only)	OFF : Executing the check function ON : Check non-execution	• Set to ON not to match B and W of the higher and lower links. • When SM1209 is ON, the link parameters of the higher and lower link are not checked. • When SM1209 is OFF, the link parameters of the higher and lower link are checked.	
M9210	SM1210	–	Link card error (for master station)	OFF : Normal ON : Abnormal	Control is performed depending on whether the link card hardware is faulty or not.	
M9211	SM1211	–	Link module error (for local station use)	OFF : Normal ON : Abnormal	Control is performed depending on whether the link card hardware is faulty or not.	
M9224	SM1224	–	Link status	OFF : Online ON : Offline,station-to-station test, or self-loopback test	Depends on whether the master station is online or offline or is in station-to-station test or self-loopback test mode.	
M9225	SM1225	–	Forward loop error	OFF : Normal ON : Abnormal	Depends on the error condition of the forward loop line.	
M9226	SM1226	–	Reverse loop error	OFF : Normal ON : Abnormal	Depends on the error condition of the reverse loop line.	
M9227	SM1227	–	Loop test status	OFF : Not being executed ON : Forward or reverse loop test execution underway	Depends on whether or not the master station is executing a forward or a reverse loop test.	
M9232	SM1232	–	Local station operation status	OFF : RUN or STEP RUN status ON : STOP or PAUSE status	Control is performed depending on whether a local station is in the STOP or PAUSE mode.	
M9233	SM1233	–	Local station error detect status	OFF : No errors ON : Error detection	Depends on whether or not a local station has detected an error in another station.	

TableApp.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9235	SM1235	–	Local station, remote I/O station parameter error detect status	OFF : No errors ON : Error detection	Depends on whether or not a local or a remote I/O station has detected any link parameter error in the master station	QnA
M9236	SM1236	–	Local station, remote I/O station initial communications status	OFF : No communications ON : Communications underway	Depends on the results of initial communication between a local or remote I/O station and the master station. (Parameter communication, etc.)	
M9237	SM1237	–	Local station, remote I/O station error	OFF : Normal ON : Abnormal	Depends on the error condition of a local or remote I/O station.	
M9238	SM1238	–	Local station, remote I/O station forward or reverse loop error	OFF : Normal ON : Abnormal	Depends on the error condition of the forward and reverse loop lines of a local or a remote I/O station.	
M9240	SM1240	–	Link status	OFF : Online ON : Offline, station-to-stationtest, or self-loopback test	Depends on whether the local station is online or offline, or is in station-to-station test or self-loopback test mode.	
M9241	SM1241	–	Forward loop line error	OFF : Normal ON : Abnormal	Depends on the error condition of the forward loop line.	
M9242	SM1242	–	Reverse loop line error	OFF : Normal ON : Abnormal	Depends on the error condition of the reverse loop line.	
M9243	SM1243	–	Loopback implementation	OFF : Loopback not being conducted ON : Loopback implementation	Depends on whether or not loopback is occurring at the local station.	
M9246	SM1246	–	Data not received	OFF : Reception ON : No reception	Depends on whether or not data has been received from the master station.	
M9247	SM1247	–	Data not received	OFF : Reception ON : No reception	Depends on whether or not a tier three station has received data from its master station in a three-tier system.	
M9250	SM1250	–	Parameters not received	OFF : Reception ON : No reception	Depends on whether or not link parameters have been received from the master station.	
M9251	SM1251	–	Link relay	OFF : Normal ON : Abnormal	Depands on the data link condition at the local station.	
M9252	SM1252	–	Loop test status	OFF : Not being executed ON : Forward or reverse loop test execution underway	Depends on whether or not the local station is executing a forward or a reverse loop test.	
M9253	SM1253	–	Master station operation status	OFF : RUN or STEP RUN status ON : STOP or PAUSE status	Control is performed depending on whether the master station is in the STOP or PAUSE mode.	
M9254	SM1254	–	Local station other than host station operation status	OFF : RUN or STEP RUN status ON : STOP or PAUSE status	Control is performed depending on whether a local station other than the host is in the STOP or PAUSE mode.	
M9255	SM1255	–	Local station other than host station error	OFF : Normal ON : Abnormal	Depends on whether or not a local station other than the host is in error.	

(11) QCPU with built-in Ethernet port

TableApp.12 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1270	Time setting function (SNTP client) execution	OFF : No time setting function (SNTP client) execution ON : Time setting function (SNTP client) execution	Set this to ON when executing the time setting function (SNTP client). (Only when the time setting function is in “Use” with the time setting parameter.)	U	New	QnU*1
SM1273	Remote password mismatch count clear	OFF : Not clear ON : Clear	To clear the acumulated numeber (SD979 to 999) of mismatched remote passwords, the setting SM1273 is executed.	U	New	

*1: This applies to the QnUDE(H)CPU.

(12)Process control instructions

TableApp.13 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1500	Hold mode	OFF : No-hold ON : Hold	• Specifies whether or not to hold the output value when a range over occurs for the S.IN instruction range check.	U	New	Q4AR QnPH QnPRH
SM1501	Hold mode	OFF : No-hold ON : Hold	• Specifies whether or not the output value is held when a range over occurs for the S.OUT instruction range check.	U	New	

(13)For redundant systems (Host system CPU information *1)

SM1510 to SM1599 are only valid for redundant systems.

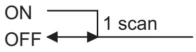
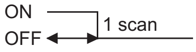
All off for standalone systems.

TableApp.14 Special relay

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU M9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SM1510</td><td>Operation mode</td><td>OFF : Redundant system backup mode, stand-alone system
ON : Redundant system separate mode</td><td>• Turns on when the operating mode is redundant system separate.</td><td>S (Each END)</td><td>New</td><td>Q4AR
QnPRH</td></tr>
<tr><td>SM1511</td><td>Start mode when power supply is on</td><td>OFF : System A fixed mode
ON : Previous control system latch mode</td><td>• Turns on when the start mode for a redundant system when the power is turned on is the previous control system latch mode.</td><td>S (Initial)</td><td>New</td><td rowspan="2">Q4AR</td></tr>
<tr><td>SM1512</td><td>Start mode when CPU is started</td><td>OFF : Initial start
ON : Hot start</td><td>• Turns on when the CPU module operation mode is hot start when the redundant system is started up.</td><td>S (Initial)</td><td>New</td></tr>
<tr><td>SM1511</td><td>System A identification flag</td><td colspan="2" rowspan="2">• Distinguishes between system A and system B.
• The flag status does not change even if the tracking cable is disconnected.
<table>
<tr><th></th><th>System A</th><th>System B</th><th>When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)</th></tr>
<tr><td>SM1511</td><td>ON</td><td>OFF</td><td>OFF</td></tr>
<tr><td>SM1512</td><td>OFF</td><td>ON</td><td>OFF</td></tr>
</table></td><td rowspan="2">S (Initial)</td><td rowspan="2">New</td><td rowspan="2">QnPRH</td></tr>
<tr><td>SM1512</td><td>System B identification flag</td></tr>
<tr><td rowspan="2">SM1513</td><td>Operation status when CPU is started</td><td>OFF : Initial start
ON : Hot start</td><td>• Turns on when the CPU module operation mode is hot start when the redundant system is actually start up.</td><td>S (Initial)</td><td>New</td><td>Q4AR</td></tr>
<tr><td>Debug mode status flag</td><td>OFF : Not in debug mode
ON : Debug mode</td><td>• Turns on when the redundant system operating mode is set to debug mode.</td><td>S (Initial)</td><td>New</td><td>QnPRH</td></tr>
<tr><td>SM1514</td><td>Operation mode at CPU module change</td><td>OFF : Initial start
ON : Hot start</td><td>• Turns on when the operation is hot start when the CPU module operation is switched for a redundant system.</td><td>S (Initial)</td><td>New</td><td rowspan="3">Q4AR</td></tr>
<tr><td>SM1515</td><td>Output hold mode</td><td>OFF : Output reset
ON : Output hold</td><td>• Turns on when the output mode during a stop error is output hold.</td><td>S (Each END)</td><td>New</td></tr>
<tr><td>SM1516</td><td>Operation system status</td><td>OFF : Control system
ON : Standby system</td><td>• Turns on when the CPU module operation system status is the standby system.</td><td>S (Status change)</td><td>New</td></tr>
<tr><td>SM1515</td><td>Control system judgment flag</td><td colspan="2" rowspan="2">• Indicates operation system status.
• The flag status does not change even if the tracking cable is disconnected.
<table>
<tr><th></th><th>Control system</th><th>Standby system</th><th>When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)</th></tr>
<tr><td>SM1515</td><td>ON</td><td>OFF</td><td>OFF</td></tr>
<tr><td>SM1516</td><td>OFF</td><td>ON</td><td>OFF</td></tr>
</table></td><td rowspan="2">S (Status change)</td><td rowspan="2">New</td><td rowspan="2">QnPRH</td></tr>
<tr><td>SM1516</td><td>Standby system judgment flag</td></tr>
</table>

*1: The information of the host CPU module is stored.

TableApp.14 Special relay

Number	Name	Meaning	Explanation			Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1517	CPU module startup status	OFF : Power supply on startup ON : Operation system switch start up	• Turns on when the CPU module is started up by the operation system switch. • Reset using the user program.			S (Status change)/U	New	Q4AR
			• Turns on when the CPU module is started up by the system switching (switching from the standby system to the control system). Remains OFF when the standby system is switched to the control system by a power-ON startup.			S (Status change)	New	QnPRH
SM1518	Tracking execution mode	OFF : Batch transfer mode ON : Carryover mode	• When this relay is turned OFF, the start of tracking is delayed until it is executable if the tracking memory is being used at END. • When this relay is turned ON, the start of tracking is carried over to next END if the tracking memory is being used at END.			U	New	Q4AR
	Standby system to control system switching status flag		• Turns ON once switch between standby system to control system, (ON for 1 scan only) occurs. • This status flag can only be used for scan execution type programs.			S (Each END)	New	QnPRH
SM1519	Previous Control System Identification Flag		• On the last operation Control System was System B,if power supply is supplied, or reset is released on both SYSTEM together,After RUN, ON for 1 scan only by System A side.			S (Each END)	New	
SM1520	Data tracking transfer trigger specification	OFF : No trigger ON : Trigger	SM1520	Block 1	<In the case of Q4AR> • When data is transferred by the data tracking instruction S. TRUCK, the target block is specified as trigger. <In the case of QnPRH> • When data is transferred based on the tracking setting of the redundant parameter dialog box, the target block is specified as trigger. • When "Auto Tracking block No.1" is enabled in the tracking setting, SM1520 is turned ON by the system at power ON/ STOP to RUN. In other cases, SM1520 to SM1583 are turned ON by the user.	<In the case of Q4AR> U <In the case of QnPRH> S (initial)/U	New	Q4AR QnPRH
SM1521			SM1521	Block 2				
SM1522			SM1522	Block 3				
SM1523			SM1523	Block 4				
SM1524			SM1524	Block 5				
SM1525			SM1525	Block 6				
SM1526			SM1526	Block 7				
SM1527			SM1527	Block 8				
SM1528			SM1528	Block 9				
SM1529			SM1529	Block 10				
SM1530			SM1530	Block 11				
SM1531			SM1531	Block 12				
SM1532			SM1532	Block 13				
SM1533			SM1533	Block 14				
SM1534			SM1534	Block 15				
SM1535			SM1535	Block 16				
SM1536			SM1536	Block 17				
SM1537			SM1537	Block 18				
SM1538			SM1538	Block 19				
SM1539			SM1539	Block 20				
SM1540			SM1540	Block 21				
SM1541			SM1541	Block 22				
SM1542			SM1542	Block 23				
SM1543			SM1543	Block 24				
SM1544			SM1544	Block 25				
SM1545			SM1545	Block 26				
SM1546			SM1546	Block 27				
SM1547			SM1547	Block 28				
SM1548			SM1548	Block 29				

TableApp.14 Special relay

Number	Name	Meaning	Explanation			Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1549	Data tracking transfer trigger specification	OFF : No trigger ON : Trigger	SM1549	Block 30	<In the case of Q4AR> When data is transferred by the data tracking instruction S. TRUCK, the target block is specified as trigger. <In the case of QnPRH> • When data is transferred based on the tracking setting of the redundant parameter dialog box, the target block is specified as trigger. • When "Auto tracking block No. 1" is enabled in the tracking setting, SM1520 is turned ON by the system at power ON/ STOP to RUN. In other cases, SM1520 to SM1583 are turned ON by the user.	<In the case of Q4AR> U <In the case of QnPRH> S (initial)/U	New	Q4AR QnPRH
SM1550			SM1550	Block 31				
SM1551			SM1551	Block 32				
SM1552			SM1552	Block 33				
SM1553			SM1553	Block 34				
SM1554			SM1554	Block 35				
SM1555			SM1555	Block 36				
SM1556			SM1556	Block 37				
SM1557			SM1557	Block 38				
SM1558			SM1558	Block 39				
SM1559			SM1559	Block 40				
SM1560			SM1560	Block 41				
SM1561			SM1561	Block 42				
SM1562			SM1562	Block 43				
SM1563			SM1563	Block 44				
SM1564			SM1564	Block 45				
SM1565			SM1565	Block 46				
SM1566			SM1566	Block 47				
SM1567			SM1567	Block 48				
SM1568			SM1568	Block 49				
SM1569			SM1569	Block 50				
SM1570			SM1570	Block 51				
SM1571			SM1571	Block 52				
SM1572			SM1572	Block 53				
SM1573			SM1573	Block 54				
SM1574			SM1574	Block 55				
SM1575			SM1575	Block 56				
SM1576			SM1576	Block 57				
SM1577			SM1577	Block 58				
SM1578			SM1578	Block 59				
SM1579			SM1579	Block 60				
SM1580			SM1580	Block 61				
SM1581			SM1581	Block 62				
SM1582			SM1582	Block 63				
SM1583			SM1583	Block 64				
SM1590	Switching status from the network module	OFF : Normal ON : Switching unsuccessful	• Turns ON when switching could not be executed normally if the network module had detected a network fault and issued a switching request to the host CPU module.			S (Error ocurrs)	New	Q4AR
	System switching enable/disable flag from network module	OFF : System switching request issuing module absent ON : System switching request issuing module present	• Turns ON when a system switching request is issued from the network module. The module No. that issued system switching can be checked by SD1590. • Turns OFF when all bits of SD1590 are OFF.			S (Each END)		QnPRH
SM1591	Standby system error detection disable flag at system switching	ON : Error is not detected by new standby system at system switching OFF : Error is detected by new standby system at system switching	This flag is used to determine if the new standby station detects 6210:STANDBY during system switching. This applies to the following switching methods: • System switching from GX Developer • System switching using dedicated instruction • System switching by the intelligent function module			U	New	
SM1592	Enable/disable user system switching	OFF : Disable user system switching ON : Enable user system switching	• This flag enables system switching by the user from GX Developer or by dedicated instruction. (SP.CONTSW).			U	New	

TableApp.14 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1593	Setting to access extension base unit of standby system CPU	OFF : Error ON : Ignored	Sets the operation for the case accessing buffer memory of the intelligent function module mounted on the extension base unit from the standby system CPU in separate mode. OFF : "OPERATION ERROR" (error code: 4112) will be returned when accessing buffer memory of the intelligent function module on the extension base unit from the standby system CPU. ON : No processing is performed when accessing buffer memory of intelligent function module on the extension base unit from the standby system CPU.	U	New	QnPRH*2
SM1595	Memory copy to other system start flag	OFF : Start memory copy ON : No memory copy initiated	• When SM1595 is turned from OFF to ON, memory copy from control system to standby system starts. Note that when SM1595 is turned from OFF to ON, memory copy dos not start if the I/O No. of the copy destination (standby system CPU module: 3D1H) is not stored in SD1595.			QnPRH
SM1596	Memory copy to other system status flag	OFF : Memory copy not executed ON : Memory copy executed	• Turns on while memory is copied to other system. • Turns off when memory copy execution has completed.	S (Starting to copy/finish)		
SM1597	Memory copy to other system completion flag	OFF : Memory copy not completed ON : Memory copy completed	• Turns on once the memory copying to the other system has completed.	S (finish)/U	New	QnPRH
SM1598	Copy contents of standard ROM during memory copy	OFF : Copy standard ROM data ON : Standard ROM data is not copied	• If set to on by user, the standard ROM data is not copied to the other system while memory copy is executing.	U		

*2: The module whose first 5 digits of serial No. is "09012" or later.

(14)For redundant system (Other system CPU information *1)

SM1600 to SM1650 only valid for the CPU redundant system backup mode, so they cannot be refreshed during the separate mode.

Either the backup mode or the separate mode is valid for the SM4651 to SM1699.

SM1600 to SM1699 are all turned off for stand-alone system.

TableApp.15 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding Host SM□□ *2	Corresponding CPU
SM1600	Diagnosis error	OFF : No error ON : Error	• Turns on if a error occurs in the diagnosis results. (Including external diagnosis) • Remains on even if returns to normal thereafter.	S (Each END)	SM0	Q4AR
	Other system error flag	OFF : No error ON : Error	• Turns on when an error occurs during redundant system. Error check (Turns on single bit of SD1600.) • Is off when no errors are present	S (Each END)	–	QnPRH
SM1601	Self diagnosis error	OFF : No self diagnosis error ON : Self diagnosis error	• Turns on when an error occurs in the self-diagnosis results. • Remains on even if returns to normal thereafter.	S (Each END)	SM1	Q4AR
SM1605	Error common information	OFF : No error common information ON : Error common information	• Turns on when there is error common information and the SM1600 is on.	S (Each END)	SM5	

*1 Stores other system CPU diagnostic information and system information.

*2 This shows the special relay(SM△□) for the host system CPU.

TableApp.15 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding Host SM□ □ *2	Corresponding CPU
SM1610	Other system diagnostics error	OFF : No error ON : Error	• Turns on when a diagnostics error occurs. (Includes error detection when annunciator is ON, and by CHK instruction) • Corresponds to status of SM0 at other system	S (Each END)	SM0	QnPRH
SM1611	Other systems self diagnostics error.	OFF : No self diagnostics error occurred ON : Self diagnostics error occurred	• Turns on when a self diagnostics error occurs. (Dose not include error detection when annunciator is ON, and by CHK instruction) • Corresponds to status of SM1 at other system	S (Each END)	SM1	
SM1615	Other system common error information	OFF : No common error information present ON : Common error information present	• Turns on when there is common error information at other system • Corresponds to status of SM5 at other system	S (Each END)	SM5	
SM1616	Error individual information	OFF : No error individual information ON : Error individual information	• Turns on when there is error individual information and the SM1600 is on.	S (Each END)	SM16	Q4AR
SM1626	Error individual information for other systems	OFF : No individual error information present ON : Individual error information present	• Turns on when there is individual error information at other system • Corresponds to status of SM16 at other system	S (Each END)	SM16	QnPRH
SM1649	Standby system cancel error flag	OFF to ON: Cancels error of standby system	By turning this relay from OFF to ON, the continue error that occurred in the standby system CPU module can be canceled. Use SD1649 to specify the error code of the error to be canceled.	U	–	
SM1653	STOP contact	STOP status	• Turns on when in the STOP status.	S (Each END)	SM203	Q4AR
SM1654	PAUSE contact	PAUSE status	• Turns on when in the PAUSE status.	S (Each END)	SM204	
SM1655	STEP-RUN contact	STEP-RUN status	• Turns on when in the STEP-RUN status.	S (Each END)	SM205	

*2 This shows the special relay(SM□ □) for the host system CPU.

(15)For redundant system (tracking)

Either the backup mode or the separate mode is valid for SM1700 to SM1799.

All is turned off for stand-alone system.

TableApp.16 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1700	Tracking execution flag	OFF : Execution not possible ON : Execution possible	• Turns on when tracking can be normally executed.	S (status change)	New	Q4AR
	Transfer trigger completion flag	OFF : Transfer not completed ON : Transfer completed	• Turns on for one scan, once transfer of block 1 to block 64 is completed.	S (status change)	New	QnPRH
SM1709	Manual system switching disable/ enable setting during online program change redundant tracking	ON : Manual system switching enabled (Disable canceled) OFF : Manual system switching disabled	(1) Turning this relay from OFF to ON enables manual system switching during online program change redundant tracking. After the manual system switching disable status is canceled, the system automatically turns off SM1709. (2) System switching due to any of the following conditions is executed even during online program change redundant tracking, regardless of the status of this relay. •Power off, reset, hardware failure, CPU stop error (3) In either of the following statuses, the system switching disable status can also be canceled by this relay. •Multiple-block online program change redundant tracking execution status •File batch online program change redundant tracking execution status	S (When executed)/U		
SM1710	Transfer tracking data during online program change enable flag	OFF : No device tracking ON : Transfer device memory	(1) Set whether the tracking of the following data will be executed or not during online program change redundant tracking. •Device memory (Including SM/SD that will automatically execute tracking) •PIDINIT information, S.PIDINIT information, SFC information (2) SM1710 can be also used to set whether tracking will be executed or not while online change of multiple program blocks or batch of files is being performed to ensure consistency of both systems. (3) This SM is also transferred form control system CPU module to standby system CPU module by tracking data.	U		

TableApp.16 Special relay

Number	Name	Meaning	Explanation			Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1712	Transfer trigger completion flag	OFF : Transfer uncompleted ON : Transfer completed	SM1712	Block 1	<In the case of Q4AR> Turns ON only during one scan when the transmission of the corresponding data is completed. <In the case of QnPRH> Turns ON only during one scan when the transmission of the corresponding block is completed.	S (status change)	New	Q4AR QnPRH
SM1713			SM1713	Block 2				
SM1714			SM1714	Block 3				
SM1715			SM1715	Block 4				
SM1716			SM1716	Block 5				
SM1717			SM1717	Block 6				
SM1718			SM1718	Block 7				
SM1719			SM1719	Block 8				
SM1720			SM1720	Block 9				
SM1721			SM1721	Block 10				
SM1722			SM1722	Block 11				
SM1723			SM1723	Block 12				
SM1724			SM1724	Block 13				
SM1725			SM1725	Block 14				
SM1726			SM1726	Block 15				
SM1727			SM1727	Block 16				
SM1728			SM1728	Block 17				
SM1729			SM1729	Block 18				
SM1730			SM1730	Block 19				
SM1731			SM1731	Block 20				
SM1732			SM1732	Block 21				
SM1733			SM1733	Block 22				
SM1734			SM1734	Block 23				
SM1735			SM1735	Block 24				
SM1736			SM1736	Block 25				
SM1737			SM1737	Block 26				
SM1738			SM1738	Block 27				
SM1739			SM1739	Block 28				
SM1740			SM1740	Block 29				
SM1741			SM1741	Block 30				
SM1742			SM1742	Block 31				
SM1743			SM1743	Block 32				
SM1744			SM1744	Block 33				
SM1745			SM1745	Block 34				
SM1746			SM1746	Block 35				
SM1747			SM1747	Block 36				
SM1748			SM1748	Block 37				
SM1749			SM1749	Block 38				
SM1750			SM1750	Block 39				
SM1751			SM1751	Block 40				
SM1752			SM1752	Block 41				
SM1753			SM1753	Block 42				
SM1754			SM1754	Block 43				
SM1755			SM1755	Block 44				
SM1756			SM1756	Block 45				
SM1757			SM1757	Block 46				
SM1758			SM1758	Block 47				
SM1759			SM1759	Block 48				

TableApp.16 Special relay

Number	Name	Meaning	Explanation			Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1760	Transfer trigger completion flag	OFF : Transmission uncompleted ON : Transmission end	SM1760	Block 49	<In the case of Q4AR> Turns ON only during one scan when the transmission of the corresponding data is completed. <In the case of QnPRH> Turns ON only during one scan when the transmission of the corresponding block is completed.	S (status change)	New	Q4AR QnPRH
SM1761			SM1761	Block 50				
SM1762			SM1762	Block 51				
SM1763			SM1763	Block 52				
SM1764			SM1764	Block 53				
SM1765			SM1765	Block 54				
SM1766			SM1766	Block 55				
SM1767			SM1767	Block 56				
SM1768			SM1768	Block 57				
SM1769			SM1769	Block 58				
SM1770			SM1770	Block 59				
SM1771			SM1771	Block 60				
SM1772			SM1772	Block 61				
SM1773			SM1773	Block 62				
SM1774			SM1774	Block 63				
SM1775			SM1775	Block 64				

(16)Redundant power supply module information

TableApp.17 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU M9□□□	Corresponding CPU
SM1780	Power supply off detection flag	OFF : No redundant power supply module with input power OFF detected ON : Redundant power supply module with input power OFF detected	• Turns ON when one or more redundant power supply modules with input power OFF are detected. • Turns on if any of SD1780 bits is on. • Turns off if all bits of SD1780 are off. • Turns OFF when the main base unit is not the redundant main base unit (Q38RB). 0: communication with peripheral device is executed 1: communication with peripheral device is nonexecuted	S (Each END)	New	Qn(H)*2 QnPH*2 QnPRH Rem
SM1781	Power supply failure detection flag	OFF : No faulty redundant power supply module detected ON : Faulty redundant power supply module detected	• Turns ON when one or more faulty redundant power supply modules are detected. • Turns on if any of SD1781 bits is on. • Turns off if all bits of SD1781 are off. • Turns OFF when the main base unit is not the redundant main base unit (Q38RB). 0: communication with peripheral device is executed 1: communication with peripheral device is nonexecuted	S (Each END)		
SM1782	Momentary power failure detection flag for power supply 1 *1	OFF : No momentary power failure detected ON : Momentary power failure detected	• Turns ON when a momentary power failure of the input power supply to the power supply 1 or 2 is detected one or more times. After turning ON, remains ON even if the power supply recovers from the momentary power failure. • Turns OFF the flag (SM1782, SM1783) of the power supply 1/2 when the CPU module starts. • When the input power to one of the redundant power supply modules turns OFF the corresponding flag turns OFF. • Turns OFF when the main base unit is not the redundant main base unit (Q38RB). 0: communication with peripheral device is executed 1: communication with peripheral device is nonexecuted	S (Each END)		
SM1783	Momentary power failure detection flag for power supply 2 *1					

*1: The "power supply 1" indicates the redundant power supply module (Q64RP) mounted on the POWER 1 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).
The "power supply 2" indicates the redundant power supply module (Q64RP) mounted on the POWER 2 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).

*2: The module whose first 5 digits of serial No. is "04012" or later.
However, for the multiple CPU system configuration, this applies to all CPU modules whose first 5 digits of serial No. are "07032" or later.

Appendix 2 Special Register List

The special registers, SD, are internal registers with fixed applications in the Programmable Controller.

For this reason, it is not possible to use these registers in sequence programs in the same way that normal registers are used.

However, data can be written as needed in order to control the CPU modules and remote I/O modules.

Data stored in the special registers are stored as BIN values if no special designation has been made to the contrary.

The heading descriptions in the following special register lists are shown in TableApp.18.

TableApp.18 Descriptions of the special register lists headings

Item	Function of Item
Number	• Indicates special register number
Name	• Indicates name of special register
Meaning	• Indicates contents of special register
Explanation	• Discusses contents of special register in more detail
Set by (When set)	• Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed. <Set by> S : Set by system U : Set by user (sequence programs or test operations from GX Developer) S/U : Set by both system and user <When set> Indicated only for registers set by system Each END : Set during each END processing Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN) Status change : Set only when there is a change in status Error : Set when error occurs Instruction execution : Set when instruction is executed Request : Set only when there is a user request (through SM, etc.) System switching : Set when system switching is executed.
Corresponding ACPU M9□□□	• Indicates corresponding special register in ACPU (When the contents are changed, the special register is represented D9□□□ format change. Incompatible with the Q00J/Q00/Q01 and QnPRH.) • New indicates the special register newly added to the QnACPU or Q series CPU module.
Corresponding CPU	Indicates the relevant CPU module. ○ : Indicates all the QnACPU and QCPU. QCPU : Indicates all the Q series CPU modules. Q00J/Q00/Q01 : Indicates the Basic model QCPU. Qn(H) : Indicates the High Performance model QCPU. QnPH : Indicates the Process CPU. QnPRH : Indicates the Redundant CPU. QnU : Indicates the Universal model QCPU QnA : Can be applied to QnA series and Q2ASCPU series Rem : Can be applied to the MELSECNET/H remote I/O modules. Each CPU type name : Can be applied only to the specific CPU. (e.g. Q4AR, Q3A)

For details on the following items, refer to the following manuals:

- Networks → Manual of the corresponding network module
- SFC → QCPU(Q mode)/QnACPU Programming Manual (SFC)

POINT

(1) SD1200 to SD1255 are used for the QnACPU.
These relays are empty for the QCPU.

(1) Diagnostic Information

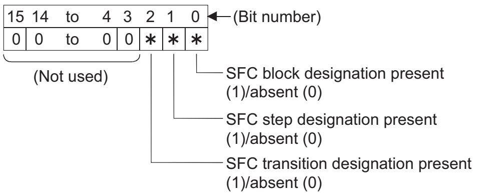
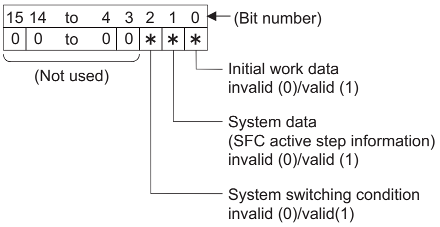
TableApp.19 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD0	Diagnostic errors	Diagnosis error code	• Error codes for errors found by diagnosis are stored as BIN data. • Contents identical to latest fault history information.	S (Error)	D9008 format change	○ Rem
SD1	Clock time for diagnosis error occurrence	Clock time for diagnosis error occurrence	• Year (last two digits) and month that SD0 data was updated is stored as BCD 2-digit code. b15 to b8 b7 to b0 | Year (0 to 99) | Month (1 to 12) | (Example) October, 1995 H9510	S (Error)	New	
SD2			• The day and hour that SD0 was updated is stored as BCD 2-digit code. b15 to b8 b7 to b0 | Day (1 to 31) | Hour (0 to 23) | (Example) 10 a.m. on 25th H2510			
SD3			• The minute and second that SD0 data was updated is stored as BCD 2-digit code. b15 to b8 b7 to b0 | Minutes (0 to 59) | Seconds (0 to 59) | (Example) 35 min. 48 sec. H3548			
SD4	Error information categories	Error information category code	Category codes which help indicate what type of information is being stored in the common information areas (SD5 through SD15) and the individual information areas (SD16 through SD26) are stored here. The category code for judging the error information type is stored. b15 to b8 b7 to b0 | Individual information category codes | Common information category codes | • The common information category codes store the following codes: 0 : No error 1: Unit/module No./ CPU No./Base No.* 2: File name/Drive name 3: Time (value set) 4: Program error location 5: System switching cause (for Q4ARCPU, Redundant CPU only) 6: Reason(s) for tracking capacity excess error (specific to Redundant CPU) 7: Base No./Power supply No. (The first 5 digits of serial number "10042" or higer are chosen for Universal model QCPU.) 8: Tracking transmission data classification (specific to Redundant CPU) *: For a multiple CPU system that consists of the Basic model QCPU, High Performance model QCPU, Process CPU, Universal model QCPU the module number or CPU number is stored depending on the error that occurred. (Refer to the corresponding error code for which number has been stored.) CPU No. 1: 1, CPU No. 2: 2, CPU No. 3: 3, CPU No. 4: 4 • The individual information category codes store the following codes: 0: No error 1: (Empty) 2: File name/Drive name 3: Time (value actually measured) 4: Program error location 5: Parameter number 6: Annunciator number 7: CHK instruction failure No. (except for the Basic model QCPU and the Universal model QCPU) 8: Reason(s) for system switching failure (specific to Redundant CPU) 12: File diagnostic information (specific to the Universal model QCPU) 13: Parameter No./CPU No. (specific to the Universal model QCPU)	S (Error)	New	

TableApp.19 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>Error common information</td><td>Error common information</td><td>
• Common information corresponding to the error codes (SD0) is stored here.
• The following ten types of information are stored here:
• The error common information type can be judged by the "common information category code" in SD4. (The values of the "common information category code" stored in SD4 correspond to following 1) to 8),
1) Slot No.
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>Slot No./CPU No./Base No.∗1, ∗2, ∗3, ∗4</td></tr>
<tr><td>SD6</td><td>I/O No.∗5</td></tr>
<tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(Empty)</td></tr>
</table>
*1: For a multiple CPU system that consists of the Basic model QCPU, High Performance model QCPU, Process CPU, Universal model QCPU, the slot number or CPU number is stored depending on the error that occurred.
Slot 0 in the multiple CPU system is the one on the slot on the right of the rightmost CPU module.
(Refer to the corresponding error code for which number has been stored.)
No. 1 CPU: 1, No. 2 CPU: 2, No. 3 CPU: 3, No. 4 CPU: 4
*2: If a fuse blown or I/O verify error occurred in the module loaded in the MELSECNET/H remote I/O station, the network number is stored into the upper 8 bits and the station number into the lower 8 bits.
Use the I/O No. to check the module where the fuse blown or I/O verify error occurred.
*3: 255 is stored into SD5 of the Basic model QCPU when an instruction, etc. has been executed for the module later than the one on the last slot where a module can be mounted.
*4: Definitions of base No. and slot No.
<Base No.>
Value used to identify the base unit on which the CPU module has been mounted. The following shows the definition of the base No.
<table>
<tr><th>Base No.</th><th>Definition</th></tr>
<tr><td>0</td><td>Indicates the main base unit mounted with the CPU module.</td></tr>
<tr><td>1 to 7</td><td>Indicates the extension base unit. The stage number setting made by the stage number setting connector on the extension base unit is the base No.
When stage number setting is extension 1:
Base No. = 1
when stage number setting is extension 7:
Base No. = 7</td></tr>
</table>
<Slot No.>
Value used to identify the slot of each base unit and the module mounted on that slot.
•The I/O slot 0 (slot on the right side of the CPU slot) of the main base unit is defined as the slot of "Slot No. = 0".
•The slot Nos. are consecutively assigned to the slots of the base units in order of the main base unit and extension base units 1 to 7.
•When the number of base unit slots has been set in the I/O assignment setting of the PLC parameter dialog box, the slot Nos. are assigned for only the number of set slots.
*5: When 0FFFFH is stored into SD6 (I/O No.), the I/O No. cannot be identified due to overlapping I/O No., etc. in the I/O assignment setting of the PLC parameter dialog box. Therefore, identify the error location using SD5.
2) File name/Drive name
<table>
<tr><th>Number</th><th colspan="2">Meaning</th></tr>
<tr><td>SD5</td><td colspan="2">Drive</td></tr>
<tr><td>SD6
SD7
SD8
SD9</td><td colspan="2">File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD10</td><td>Extension ∗6</td><td>2EH(.)</td></tr>
<tr><td>SD11</td><td colspan="2">(ASCII code: 3 characters)</td></tr>
<tr><td>SD12
SD13
SD14
SD15</td><td colspan="2">(Empty)</td></tr>
</table>
(Example) File name = ABCDEFGH. IJK
<table>
<tr><th>b15 to b8</th><th>b7 to b0</th></tr>
<tr><td>42H(B)</td><td>41H(A)</td></tr>
<tr><td>44H(D)</td><td>43H(C)</td></tr>
<tr><td>46H(F)</td><td>45H(E)</td></tr>
<tr><td>48H(H)</td><td>47H(G)</td></tr>
<tr><td>49H(I)</td><td>2EH(.)</td></tr>
<tr><td>4BH(K)</td><td>4AH(J)</td></tr>
</table>
</td><td>S (Error)</td><td>New</td><td>○
Rem</td></tr>
</table>

TableApp.19 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by
(When Set)</th><th>Corres-ponding ACPU
D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD5</td><td rowspan="11">Error common information</td><td rowspan="11">Error common information</td><td rowspan="6">
3) Time (value set)
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>Time : 1μs units (0 to 999μs)</td></tr>
<tr><td>SD6</td><td>Time : 1ms units (0 to 65535ms)</td></tr>
<tr><td>SD7</td><td rowspan="9">(Empty)</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>
4) Program error location
<table>
<tr><th>Number</th><th colspan="2">Meaning</th></tr>
<tr><td>SD5</td><td colspan="2" rowspan="4">File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD6</td></tr>
<tr><td>SD7</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td><td>Extension *6</td><td>2EH(.)</td></tr>
<tr><td>SD10</td><td colspan="2">(ASCII code: 3 characters)</td></tr>
<tr><td>SD11</td><td colspan="2">Pattern *7</td></tr>
<tr><td>SD12</td><td colspan="2">Block No.</td></tr>
<tr><td>SD13</td><td colspan="2">Step No./transition condition</td></tr>
<tr><td>SD14</td><td colspan="2">Sequence step No. (L)</td></tr>
<tr><td>SD15</td><td colspan="2">Sequence step No. (H)</td></tr>
</table>
*7 : Contents of pattern data

15 14 to 4 3 2 1 0
(Bit number)
0 0 to 0 0 * * *
(Not used)
SFC block designation present (1)/absent (0)
SFC step designation present (1)/absent (0)
SFC transition designation present (1)/absent (0)
</td><td rowspan="6">S (Error)</td><td rowspan="6">New</td><td rowspan="6">○
Rem</td></tr>
<tr><td>SD6</td></tr>
<tr><td>SD7</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td><td rowspan="5">
5) Reason(s) for system switching
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>System switching condition
(0: automatic system switching/
1: manual system switching)</td></tr>
<tr><td>SD6</td><td>System switching direction
(0:standby system to control system/
1: control system to standby system)</td></tr>
<tr><td>SD7</td><td>Tracking flag *8</td></tr>
<tr><td>SD8</td><td rowspan="8">(Empty)</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>
*8 : Tracking flag contents
Shows whether or not the tracking data is valid.

15 14 to 4 3 2 1 0
(Bit number)
0 0 to 0 0 * * *
(Not used)
Initial work data invalid (0)/valid (1)
System data (SFC active step information) invalid (0)/valid (1)
System switching condition invalid (0)/valid(1)
</td><td rowspan="5">S (Error)</td><td rowspan="5">New</td><td rowspan="5">Q4AR</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>

TableApp.19 Special register

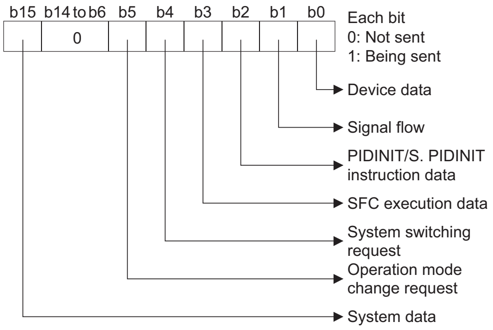
<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td rowspan="2">SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td rowspan="2">Error common information</td><td rowspan="2">Error common information</td><td>
5) Reason(s) for system switching
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>System switching condition *13</td></tr>
<tr><td>SD6</td><td>Control system switching instruction argument</td></tr>
<tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(Empty)</td></tr>
</table>
*13: Details of reason(s) for system switching
0 : No system switching condition (default)
1 : Power-OFF, reset, hardware failure, watchdog timer error
2 : Stop error (except watchdog timer error)
3 : System switching request by network module
16 : Control system switching instruction
17 : Control system switching request from GX Developer

6) Reason(s) for tracking capacity excess error
The block No. when the data amount that can be tracked (100k) is exceeded is indicated by the bit pattern of the corresponding special relay.
<table>
<tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD5</td><td>1
(SM1535)
(Block16)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1
(SM1528)
(Block9)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1
(SM1520)
(Block1)</td></tr>
<tr><td>SD6</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD7</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD8</td><td>1
(SM1583)
(Block64)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1
(SM1568)
(Block49)</td></tr>
<tr><td>SD9</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD15</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>
</td><td>S (Error)</td><td>New</td><td>QnPRH</td></tr>
<tr><td>
7) Power supply No.
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>Base No.</td></tr>
<tr><td>SD6</td><td>Power supply No.</td></tr>
<tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(Empty)</td></tr>
</table>
1: Power supply 1 fault
2: Power supply 2 fault
"Power supply module 1": Redundant power supply module mounted on POWER 1 slot of redundant base unit (Q38RB, Q68RB,Q65WRB)
"Power supply module 2": Redundant power supply module mounted on POWER 2 slot of redundant base unit (Q38RB, Q68RB,Q65WRB)
</td><td>S (Error)</td><td>New</td><td>Qn(H)*1
QnPH*1
QnPRH
QnU*2</td></tr>
</table>

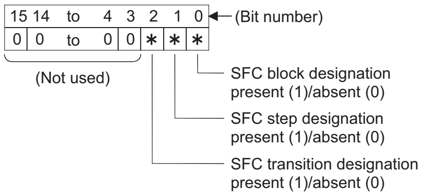
*1: The module whose first 5 digits of serial No. is "07032" or later.

*2: The module whose first 5 digits of serial No. is "10042" or later.

TableApp.19 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD5</td><td rowspan="11">Error common information</td><td rowspan="11">Error common information</td><td rowspan="11">8) Tracking transmission data classification
Stores the data classification during tracking.
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD5</td><td>Data type ∗15</td></tr>
<tr><td>SD6</td><td rowspan="10">(Empty)</td></tr>
<tr><td>SD7</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>
*15: Details of data classification

</td><td rowspan="11">S (Error)</td><td rowspan="11">New</td><td rowspan="11">QnPRH</td></tr>
<tr><td>SD6</td></tr>
<tr><td>SD7</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>

TableApp.19 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD16</td><td rowspan="11">Error individual information</td><td rowspan="11">Error individual information</td><td rowspan="11">
• Individual information corresponding to error codes (SD0) is stored here.
• There are the following eight different types of information are stored.
• The error individual information type can be judged by the "individual information category code" in SD4. (The values of the "individual information category code" stored in SD4 correspond to following 1) to 8),12), and 13),)
1) (Empty)
2) File name/Drive name
<table>
<tr><th>Number</th><th colspan="2">Meaning</th></tr>
<tr><td>SD16</td><td colspan="2">Drive</td></tr>
<tr><td>SD17</td><td colspan="2" rowspan="4">File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td><td>Extension ∗6</td><td>2EH(.)</td></tr>
<tr><td>SD22</td><td colspan="2">(ASCII code: 3 characters)</td></tr>
<tr><td>SD23</td><td colspan="2" rowspan="4">(Vacancy)</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
(Example) File name = ABCDEFGH. IJK
<table>
<tr><th>b15 to b8</th><th>b7 to b0</th></tr>
<tr><td>42H(B)</td><td>41H(A)</td></tr>
<tr><td>44H(D)</td><td>43H(C)</td></tr>
<tr><td>46H(F)</td><td>45H(E)</td></tr>
<tr><td>48H(H)</td><td>47H(G)</td></tr>
<tr><td>49H(I)</td><td>2EH(.)</td></tr>
<tr><td>4BH(K)</td><td>4AH(J)</td></tr>
</table>
3) Time (value actually measured)
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>Time : 1μs units (0 to 999μs)</td></tr>
<tr><td>SD17</td><td>Time : 1ms units (0 to 65535ms)</td></tr>
<tr><td>SD18</td><td rowspan="9">(Vacancy)</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
4) Program error location
<table>
<tr><th>Number</th><th colspan="2">Meaning</th></tr>
<tr><td>SD16</td><td colspan="2" rowspan="4">File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD17</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td><td>Extension ∗6</td><td>2EH(.)</td></tr>
<tr><td>SD21</td><td colspan="2">(ASCII code: 3 characters)</td></tr>
<tr><td>SD22</td><td colspan="2">Pattern ∗7</td></tr>
<tr><td>SD23</td><td colspan="2">Block No.</td></tr>
<tr><td>SD24</td><td colspan="2">Step No./transition No.</td></tr>
<tr><td>SD25</td><td colspan="2">Sequence step No. (L)</td></tr>
<tr><td>SD26</td><td colspan="2">Sequence step No. (H)</td></tr>
</table>
*7 : Contents of pattern data

15 14 to 4 3 2 1 0
(Bit number)
0 0 to 0 0 ∗ ∗ ∗
(Not used)
SFC block designation present (1)/absent (0)
SFC step designation present (1)/absent (0)
SFC transition designation present (1)/absent (0)

5) Parameter No. 6) Annunciator number /
7) CHK instruction malfunction number
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>Parameter No. ∗16</td></tr>
<tr><td>SD17</td><td rowspan="10">(Vacancy)</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>No.</td></tr>
<tr><td>SD17</td><td rowspan="10">(Vacancy)</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
*16: For details of the parameter No., refer to the QCPU User's Manual (Function Explanation・Program Fundamentals) or to the User's Manual of the CPU module used.
</td><td rowspan="11">S (Error)</td><td rowspan="11">New</td><td rowspan="11">○
Rem</td></tr>
<tr><td>SD17</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>

Remark

*6 : Extensions are shown below.

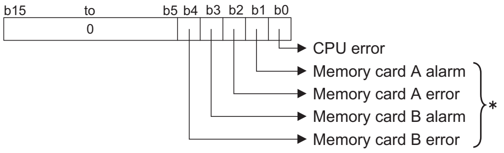
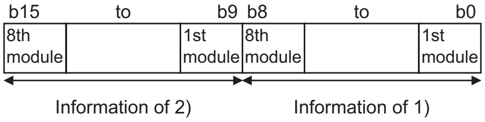
TableApp.20 Extension name

SDn	SDn+1		Extension Name	File Type
Higher 8 bits	Lower 8 bits	Higher 8 bits		
51H	50H	41H	QPA	Parameters
51H	50H	47H	QPG	• Sequence program • SFC program
51H	43H	44H	QCD	Device comment
51H	44H	49H	QDI	Initial device value
51H	44H	52H	QDR	File register
51H	44H	53H	QDS	Simulation data (For QnA)
51H	44H	4CH	QDL	Local device (Other than the Basic model QCPU)
51H	54H	44H	QTD	Sampling trace data (Other than the Basic model QCPU)
51H	54H	4CH	QTL	Status latch data (For QnA)
51H	54H	50H	QTP	Program trace data (For QnA)
51H	54H	52H	QTR	SFC trace file (For QnA)
51H	46H	44H	QFD	Breakdown history data (Other than the Basic model QCPU and the Universal model QCPU)
51H	53H	54H	QST	SP.DEVST/S.DEVLD instruction file (For Universal model QCPU only)

TableApp.19 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td rowspan="2">SD26</td><td rowspan="2">Error individual information</td><td rowspan="2">Error individual information</td><td>8) Reason(s) for system switching failure
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>System switching prohibition condition ∗14</td></tr>
<tr><td>SD17</td><td rowspan="10">(Vacancy)</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table>
*14: Details of reason(s) for system switching failure
0 : Normal switching completion (default)
1 : Tracking cable fault (cable removal, cable fault, internal circuit fault, hardware fault)
2 : Hardware failure, power OFF, reset or watchdog timer error occurring in standby system
3 : Hardware failure, power OFF, reset or watchdog timer error occurring in control system
4 : Preparing for tracking
5 : Time limit exceeded
6 : Standby system is in stop error (except watchdog timer error)
7 : Operation differs between two systems (in backup mode only)
8 : During memory copy from control system to standby system
9 : Online change
10 : Error detected by network module of standby system
11 : System switching being executed</td><td rowspan="2">S (Error)</td><td rowspan="2">New</td><td>QnPRH</td></tr>
<tr><td>12) File diagnostic information
<table>
<tr><td>SD16</td><td>Failuer information (H)</td><td>drive No.(L)</td></tr>
<tr><td>SD17</td><td colspan="2" rowspan="4">File name
(ASCII: 8 characters)</td></tr>
<tr><td>SD18</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td><td>EXtension *6</td><td>2EH(.)</td></tr>
<tr><td>SD22</td><td colspan="2">(ASCII; 3 characters)</td></tr>
<tr><td>SD23</td><td colspan="2" rowspan="2">Failure information 2
(CRC value that is read)</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td><td colspan="2" rowspan="2">Failure information 3
(CRC value that is calculated)</td></tr>
<tr><td>SD26</td></tr>
</table>
13) Parameter No./CPU No.
<table>
<tr><th>Number</th><th>Meaning</th></tr>
<tr><td>SD16</td><td>Parameter No.*16</td></tr>
<tr><td>SD17</td><td>CPU No. (1 to 4)</td></tr>
<tr><td>SD18</td><td rowspan="9">(Empty)</td></tr>
<tr><td>SD19</td></tr>
<tr><td>SD20</td></tr>
<tr><td>SD21</td></tr>
<tr><td>SD22</td></tr>
<tr><td>SD23</td></tr>
<tr><td>SD24</td></tr>
<tr><td>SD25</td></tr>
<tr><td>SD26</td></tr>
</table></td><td>QnU</td></tr>
</table>

TableApp.19 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD50	Error reset	Error number that performs error rese	• Stores error number that performs error reset	U	New	○ Rem
SD51	Battery low latch	Bit pattern indicating where battery voltage drop occurred	• All corresponding bits go 1(ON) when battery voltage drops. • Subsequently, these remain 1(ON) even after battery voltage has been returned to normal.  b15 to b5 b4 b3 b2 b1 b0 0 CPU error Memory card A alarm Memory card A error Memory card B alarm Memory card B error * *: This does not apply to Basic model QCPU. • In the alarm, data can be held within the time specified for battery low. • The error indicates the complete discharge of the battery. • When the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU is used, the corresponding bits always remain OFF since the memory card B is a standard memory.	S (Error)	New	○
SD52	Battery low	Bit pattern indicating where battery voltage drop occurred	• Same configuration as SD51 above • After the alarm is detected (ON), the alarm turns OFF by error detection (ON). (For the Universal model QCPU only) • Turns to 0 (OFF) when the battery voltage returns to normal thereafter. • When the High Performance model QCPU, Process CPU, Redundant CPU or Universal model QCPU is used, the corresponding bits always remain OFF since the memory card B is a standard memory.	S (Error)	New	
SD53	AC/DC DOWN detection	Number of times for AC/DC DOWN detection	• Every time the input voltage falls to or below 85% (AC power)/65% (DC power) of the rating during operation of the CPU module, the value is incremented by 1 and stored in BIN code. • The counter repeats increment and decrement of the value ; 0→32767→-32768→0	S (Error)	D9005	○ Rem
SD54	MINI link errors	Error detection state	1) When any of X(n+0)/X(n+20), X(n+6)/X(n+26), X(n+7)/X(n+27) and X(n+8)/X(n+28) of the mounted MINI(-S3) turns ON, the bit of the corresponding station turns to 1 (ON). 2) Turns to 1 (ON) when communication between the mounted MINI(-S3) and CPU module cannot be made.  b15 to b9 b8 to b0 8th module 1st module 8th module 1st module Information of 2) Information of 1)	S (Error)		QnA
SD60	Number of module with blown fuse	Number of module with blown fuse	• Value stored here is the lowest station I/O number of the module with the blown fuse.	S (Error)		○ Rem
SD61	I/O module verify error number	I/O module verify error module number	• The lowest I/O number of the module where the I/O module verification number took place.	S (Error)		

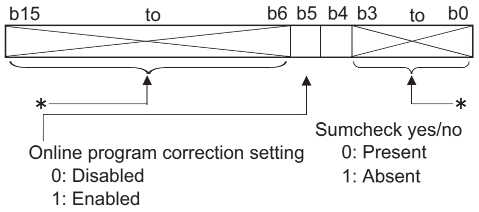
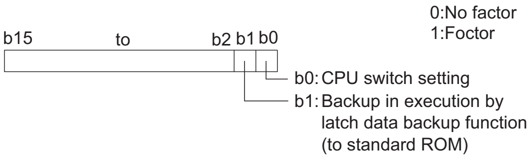
TableApp.19 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD62	Annunciator number	Annunciator number	• The first annunciator number (F number) to be detected is stored here.	S (Instruction execution)	D9009	○
SD63	Number of annunciators	Number of annunciators	• Stores the number of annunciators searched.	S (Instruction execution)	D9124	
SD64	Table of detected annunciator numbers	Annunciator detection number	When F goes ON due to OUT F or SET F, the F numbers which go progressively ON from SD64 through SD79 are registered. The F numbers turned OFF by RST F are deleted from SD64 - SD79, and the F numbers stored after the deleted F numbers are shifted to the preceding registers. Execution of the LEDR instruction shifts the contents of SD64 to SD79 up by one. (This can also be done by using the INDICATOR RESET switch on the of the Q3A/Q4ACPU.) After 16 annunciators have been detected, detection of the 17th will not be stored from SD64 through SD79.	S (Instruction execution)	D9125	
SD65					D9126	
SD66					D9127	
SD67					D9128	
SD68					D9129	
SD69					D9130	
SD70					D9131	
SD71					D9132	
SD72					New	
SD73					New	
SD74					New	
SD75					New	
SD76					New	
SD77					New	
SD78					New	
SD79					New	

		SET F50	SET F25	SET F99	RST F25	SET F15	SET F70	SET F65	SET F38	SET F110	SET F151	SET F210	LEDR	
SD62	0	50	50	50	50	50	50	50	50	50	50	50	99	(Number detected)
SD63	0	1	2	3	2	3	4	5	6	7	8	9	8	(Number of annunciators detected)
SD64	0	50	50	50	50	50	50	50	50	50	50	50	99	(Number detected)
SD65	0	0	25	25	99	99	99	99	99	99	99	99	15	
SD66	0	0	0	99	0	15	15	15	15	15	15	15	70	
SD67	0	0	0	0	0	0	70	70	70	70	70	70	65	
SD68	0	0	0	0	0	0	0	65	65	65	65	65	38	
SD69	0	0	0	0	0	0	0	0	38	38	38	38	110	
SD70	0	0	0	0	0	0	0	0	0	110	110	110	151	
SD71	0	0	0	0	0	0	0	0	0	0	151	151	210	
SD72	0	0	0	0	0	0	0	0	0	0	0	210	0	
SD73	0	0	0	0	0	0	0	0	0	0	0	0	0	
SD74	0	0	0	0	0	0	0	0	0	0	0	0	0	
SD75	0	0	0	0	0	0	0	0	0	0	0	0	0	
SD76	0	0	0	0	0	0	0	0	0	0	0	0	0	
SD77	0	0	0	0	0	0	0	0	0	0	0	0	0	
SD78	0	0	0	0	0	0	0	0	0	0	0	0	0	
SD79	0	0	0	0	0	0	0	0	0	0	0	0	0	

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD80	CHK number	CHK number	• Error codes detected by the CHK instruction are stored as BCD code.	S (Instruction execution)	New	QnA Qn(H) QnPH QnPRH
SD90	Step transition monitoring timer setting value (Enabled only when SFC program exists)	F number for timer set value and time over error	Corresponds to SM90 • Set the annunciator number (F number) that will be turned ON when the step transition monitoring timer setting or monitoring timeout occurs. b15 to b8 b7 to b0 F number setting (0 to 255) Timer time limit setting (1 to 255s: (1s units)) • Turning ON any of SM90 to SM99 during an active step starts the timer, and if the transition condition next to the corresponding step is not met within the timer time limit, the set annunciator (F) turns ON.	U	D9108	QnA Qn(H) QnPH QnPRH
SD91			Corresponds to SM91		D9109	
SD92			Corresponds to SM92		D9110	
SD93			Corresponds to SM93		D9111	
SD94			Corresponds to SM94		D9112	
SD95			Corresponds to SM95		D9113	
SD96			Corresponds to SM96		D9114	
SD97			Corresponds to SM97		New	
SD98			Corresponds to SM98		New	
SD99			Corresponds to SM99		New	

TableApp.19 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD100	Transmission speed storage area	Stores the transmission speed specified in the serial communication setting.	96 : 9.6kbps, 192 : 19.2kbps, 384 : 38.4kbps, 576 : 57.6kbps, 1152 : 115.2kbps	S (Power-ON or reset)	New	Q00/Q01
SD101	Communication setting storage area	Stores the communication setting specified in the serial communication setting.	 b15 to b6 b5 b4 b3 to b0 * * Sumcheck yes/no 0: Present 1: Absent Online program correction setting 0: Disabled 1: Enabled * : Since the data is used by the system, it is undefined.	S (Power-ON or reset)	New	
SD102	Transmission wait time storage area	Stores the transmission wait time specifed in the serial communication setting.	0 : No waiting time 1 to FH : Waiting time (unit: 10ms) Defaults to 0.	S (Power-ON or reset)	New	
SD105	CH1 transmission speed setting (RS-232)	Stores the preset transmission speed when GX Developer is used.	96 : 9600bps, 192 : 19.2kbps, 384 : 38.4kbps, 576 : 57.6kbps, 1152 : 115.2kbps *: Other than RS-232 connection holds the data at RS-232 connection. (When disconnected, the default value is 1152.)	S	New	Qn(H) QnPH QnPRH QnU[*3] Rem
SD110	Data sending result storage area	Stores the data sending result when the serial communication function is used.	Stores the error code at the timeout sending data.	S (Error)	New	Q00/Q01
SD111	Data receiving result storage area	Stores the data receiving result when the serial communication function is used.	Stores the error code at the time of receiving data.	S (Error)		
SD118	Amount of battery consumption	Amount of battery consumption	Displays current amount of battery consumption. Range: 1 to 2 (Q02UCPU, Q03UDCPU, Q04UDHCPU, Q03UDECPU, Q04UDEHCPU) 1 to 3 (Q06UDHCPU, Q06UDEHCPU) 1 to 4 (Q13UDHCPU, Q26UDHCPU, Q13UDEHCPU, Q26UDEHCPU)	S (Status change)	New	QnU [*4]
SD119	Battery life-prolonging factor	Battery life-prolonging factor	Stores the factor which makes the battery life-prolonging function valid. When SD119 is other than 0, the battery life-prolonging function is valid.  0:No factor 1:Foctor b15 to b2 b1 b0 b0: CPU switch setting b1: Backup in execution by latch data backup function (to standard ROM)	S (Status change)	New	QnU

*3: This applies to Universal model QCPUs except for the QnUDE(H)CPU.

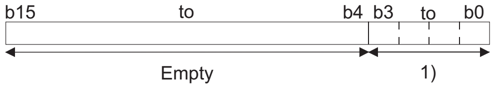
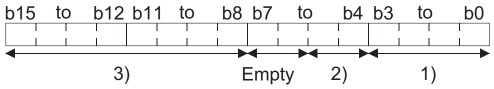
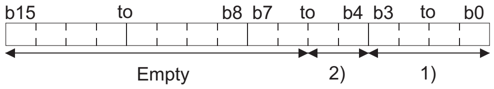
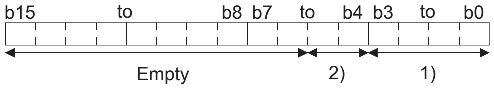
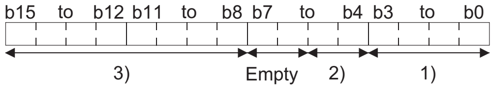
*4: This applies to the following modules.
- Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU that the first 5 digits of the serial No. is "10012" or higher.
- Q13UDHCPU, Q26UDHCPU, QnUDE(H)CPU

TableApp.19 Special register

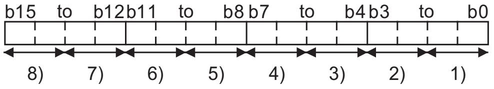
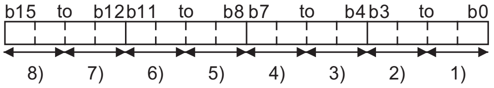
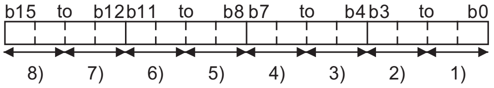
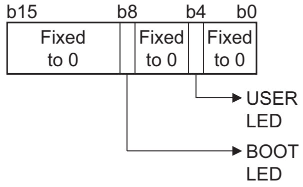
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD130	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown 0: No blown fuse 1: Blown fuse present	• The numbers of output modules whose fuses have blown are input as a bit pattern (in units of 16 points). (If the module numbers are set by parameter, the parameter-set numbers are stored.) b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD130 0 0 0 1 (YC0) 0 0 0 1 (Y80) 0 0 0 0 0 0 0 0 SD131 1 (Y1F0) 0 0 0 0 1 (Y1A0) 0 0 0 0 0 0 0 0 0 0 SD137 0 0 0 0 1 (Y7B0) 0 0 0 0 0 0 0 1 (Y730) 0 0 0 Indicates fuse blow. • Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation	S (Error)	New	Q00J/Q00/Q01
SD131						
SD132						
SD133						
SD134						
SD135						
SD136						
SD137						
SD150	I/O module verify error	Bit pattern, in units of 16 points, indicating the modules with verify errors. 0: No I/O verify errors 1: I/O verify error present	• When I/O modules, of which data are different from those entered at power-ON, have been detected, the I/O module numbers (in units of 16 points) are entered in bit pattern. (Preset I/O module numbers set in parmeters when parameter setting has been performed.) b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD150 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 (X Y 0) SD151 0 0 0 0 0 0 1 (X Y 190) 0 0 0 0 0 0 0 0 0 SD157 0 1 (X Y 7E0) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Indicates an I/O module verify error. • Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.			
SD151						
SD152						
SD153						
SD154						
SD155						
SD156						
SD157						

(2) System information

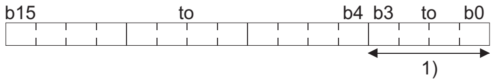
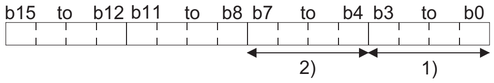
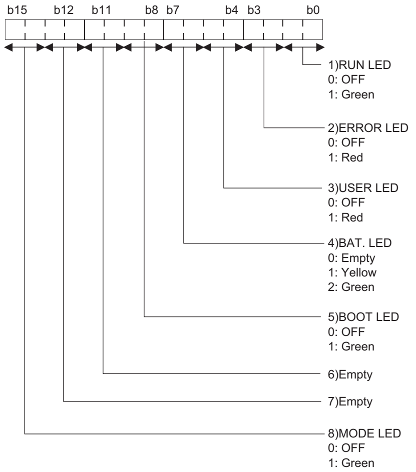
TableApp.21 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD200	Status of switch	Status of CPU switch	• The switch status of the remote I/O module is stored in the following format.  1) Remote I/O module switch statusAlways 1: STOP	S (Always)	New	Rem
			• The CPU switch status is stored in the following format:  1): CPU switch status: 0: RUN 1: STOP 2: L.CLR 2): Memory card switch: Always OFF 3): DIP switch: b8 through b12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON. b13 through b15 are empty.	S (Every END processing)	New	Qn(H) QnPH QnPRH
			• The CPU switch status is stored in the following format:  1): CPU switch status: 0: RUN 1: STOP 2): Memory card switch: Always OFF	S (Every END processing)	New	Q00J/Q00/Q01
			• The CPU switch status is stored in the following format:  1): CPU switch status: 0: RUN 1: STOP 2): Memory card switch: Always OFF	S (when RUN/ STOP/RESET switch changed)	New	QnU
			• The CPU switch status is stored in the following format:  1): CPU switch status: 0: RUN 1: STOP 2: L.CLR 2): Memory card switch: b4 corresponds to memory card A, and b5 corresponds to memory card B. 0: OFF, 1: ON 3): DIP switch: b8 through b12 correspond to SW1 through SW5 of system setting switch 1. b14 and b15 correspond to SW1 and SW2 of system setting switch 2, respectively. OFF at 0; ON at 1	S (Every END processing)	New	QnA

TableApp.21 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD201	LED status	Status of CPU-LED	• The following bit patterns store the status of the LEDs on the CPU module: • 0 is off, 1 is on, and 2 is flicker.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 8) 7) 6) 5) 4) 3) 2) 1) 1): RUN 5): BOOT 2): ERR. 6): Empty Mode bit pattern 3): USER 7): Empty 0: OFF 1: Green 4): BAT. 8): MODE 2: Orange (The Basic model QCPU does not include 3) to 8).)	S (Status change)	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH
			• The following bit patterns store the status of the LEDs on the CPU module: • 0 is off, 1 is on, and 2 is flicker.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 8) 7) 6) 5) 4) 3) 2) 1) 1): RUN 5): BOOT 2): ERROR 6): Empty 3): USER 7): Empty 4): BAT. 8): MODE	S (Status change)	New	QnU
			• The following bit patterns store the status of the LEDs on the CPU module: • 0 is off, 1 is on, and 2 is flicker  b15 to b12 b11 to b8 b7 to b4 b3 to b0 8) 7) 6) 5) 4) 3) 2) 1) 1): RUN 5): BOOT 2): ERROR 6): CARD A (memory card A) 3): USER 7): CARD B (memory card B) 4): BAT.ALARM 8): Empty	S (Status change)	New	QnA
SD202	LED off command	Bit pattern of LED that is turned off	• Stores bit patterns of LEDs turned off (Only USER and BOOT enabled) • Turned off at 1, not turned off at 0	U	New	
			• Specify the LEDs to be turned off using this register, and turn SM202 from OFF to ON to turn off the specified LEDs. USER and BOOT can be specified as the LEDs to be turned off. • Specify the LEDs to be turned off in the following bit pattern. (Turned off at 1, not be turned off at 0.)  b15 b8 b4 b0 Fixed to 0 Fixed to 0 Fixed to 0 USER LED BOOT LED	U	New	Qn(H) QnPH QnPRH QnU

TableApp.21 Special register

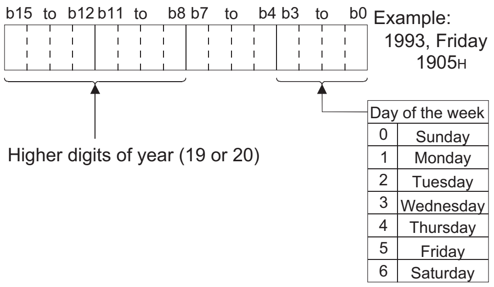
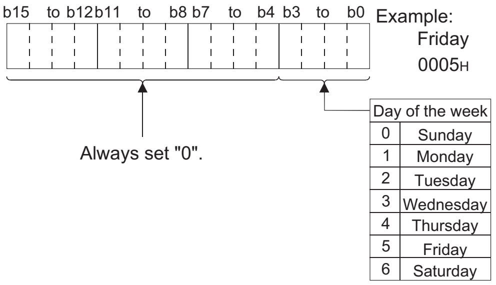
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD203	Operating status of CPU	Operating status of CPU	• The operating status of the remote I/O module is stored in the following format.  1) Remote I/O module operating statusAlways 2: STOP	S (Always)	New	Rem
			• The CPU operating status is stored as indicated in the following figure:  1): Operating status of CPU — 0: RUN 1: STEP-RUN (For the QnACPU only) 2: STOP 3: PAUSE 2): STOP/PAUSE cause — 0: Instruction in remote operation program from RUN/STOP switch ("RUN/STOP/ RESET switch" for Basic model QCPU) 1: Remote contact 2: Remote operation from GX Developer/ serial communication, etc. 3: Internal program instruction Note: Priority is earliest first — 4: Error	S (Every END processing)	D9015 format change	○
SD204	LED display color	CPU-LED display color	• The LED display color of the LED status shown in SD201 1) to 8).  b15 b12 b11 b8 b7 b4 b3 b0 1)RUN LED 0: OFF 1: Green 2)ERROR LED 0: OFF 1: Red 3)USER LED 0: OFF 1: Red 4)BAT. LED 0: Empty 1: Yellow 2: Green 5)BOOT LED 0: OFF 1: Green 6)Empty 7)Empty 8)MODE LED 0: OFF 1: Green	S (status change)	New	QnU

TableApp.21 Special register

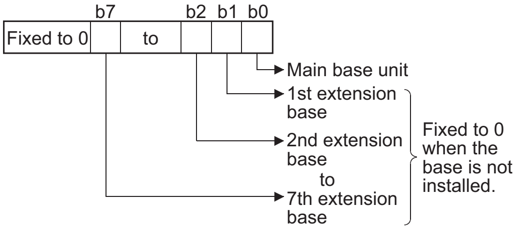
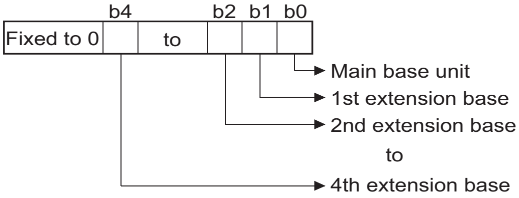
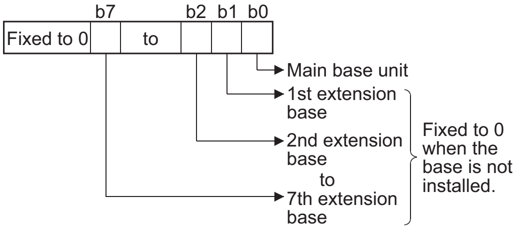
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD206	Device test execution type	0: Test not yet executed 1: During X device test 2: During Y device test 3: During X/Y device test	• Set when the device test mode is executed on GX Developer.	S (Request)	New	Rem
SD207	LED display priority ranking	Priorities 1 to 4	• When error is generated, the LED display (flicker) is made according to the error number setting priorities. (The Basic model QCPU supports only the annunciator (error item No. 7). • The Universal model QCPU sets execution/non-execution of LED display of the error corresponding to the each priority ranking when the error occurs. • The setting areas for priorities are as follows: b15 to b12 b11 to b8 b7 to b4 b3 to b0 SD207 Priority 4 Priority 3 Priority 2 Priority 1 SD208 Priority 8 Priority 7 Priority 6 Priority 5 SD209 Priority 11 Priority 10 Priority 9 (Priority 11 is valid when Redundant CPU is used.) Default Value SD207 = 4321_H(0000_H for Basic model QCPU) SD208 = 8765_H(0700_H for Basic model QCPU) (0765_H for Redundant CPU) SD209 = $00A9_H$(0000_H for Basic model QCPU) ($0B09_H$ for Redundant CPU) • No display is made if "0" is set. • In case of the Basic model QCPU, the ERR. LED turns ON when the annunciator turns ON, if "7" has been set to either of priorities 1 to 11. • In case of the Basic model QCPU, the ERR. LED does not turn ON when the annumciator turns ON, if "7" has not been set to either of priorities 1 to 11. However, even if "0" has been set, information concerning CPU module operation stop (including parameter settings) errors will be indicated by the LEDs without conditions.	U	D9038	QnA Q00J/ Q00/Q01*9 Qn(H) QnPH QnPRH QnU
SD208		Priorities 5 to 8			D9039 format change	
SD209		Priorities 9 to 11			New	
SD210	Clock data	Clock data (year, month)	• The year (last two digits) and month are stored as BCD code as shown below: b15 to b12 b11 to b8 b7 to b4 b3 to b0 Year Month Example: July, 1993 9307_H	S (Request)/U	D9025	○ Rem
SD211	Clock data	Clock data (day, hour)	• The day and hour are stored as BCD code as shown below: b15 to b12 b11 to b8 b7 to b4 b3 to b0 Day Hour Example: 31st, 10 a.m. 3110_H		D9026	
SD212	Clock data	Clock data (minute, second)	• The minutes and seconds (after the hour) are stored as BCD code as shown below: b15 to b12 b11 to b8 b7 to b4 b3 to b0 Minute Second Example: 35 min, 48 s 3548_H		D9027	

*9: Function version is B or later.

TableApp.21 Special register

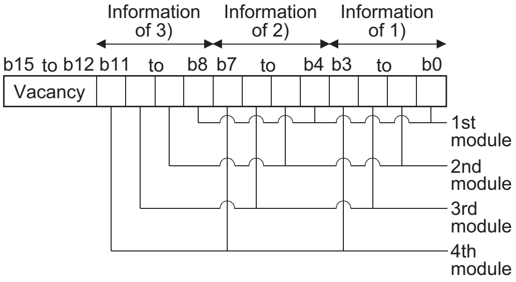
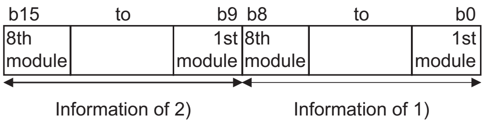
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD213	Clock data	Clock data (higher digits of year, day of week)	• The year (first two digits) and the day of the week are stored as BCD code as shown below.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example: 1993, Friday 1905H Higher digits of year (19 or 20) Day of the week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday	S (Request)/U	D9028	QCPU Rem
		Clock data (day of week)	• The day of the week is stored as BCD code as shown below.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example: Friday 0005H Always set "0". Day of the week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday			QnA
SD220	LED display data	LED display data	• LED display ASCII data (16 characters) stored here. (On the Basic model QCPU, the registers store the message (16 characters of ASCII data) at error occurrence (including annunciator ON). b15 to b8 / b7 to b0 SD220: 15th character from the right / 16th character from the right SD221: 13th character from the right / 14th character from the right SD222: 11th character from the right / 12th character from the right SD223: 9th character from the right / 10th character from the right SD224: 7th character from the right / 8th character from the right SD225: 5th character from the right / 6th character from the right SD226: 3rd character from the right / 4th character from the right SD227: 1st character from the right / 2nd character from the right • The LED display device data at the time of CHK is not stored in the Basic model QCPU and the Universal model QCPU.	S (When changed)	New	○
SD221						
SD222						
SD223						
SD224						
SD225						
SD226						
SD227						
SD235	Module to which online module change is being performed	The header I/O number of the module to which online module change is being performed /10H	• 10H is added to the value of the header I/O number of which the online module change is being performed.	S (During online module change)	New	QnPH QnPRH
SD240	Base mode	0: Automatic mode 1: Detail mode	• Stores the base mode.	S (Initial)	New	QCPU Rem
SD241	Extension stage number	0: Main base only 1 to 7: Extension stage number	• Stores the maximum number of the extension bases being installed.	S (Initial)	New	

TableApp.21 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD242	A/Q base differentiation	Base type differentiation 0: QA**B is installed (A mode) 1: Q**B is installed (Q mode)	 b7 b2 b1 b0 Fixed to 0 to Main base unit 1st extension base 2nd extension base to 7th extension base Fixed to 0 when the base is not installed.	S (Initial)	New	Qn(H) QnPH QnPRH Rem
	Installed Q base presence/ absence	Base type differentiation 0: Base not installed 1: Q**B is installed	 b4 b2 b1 b0 Fixed to 0 to Main base unit 1st extension base 2nd extension base to 4th extension base	S (Initial)	New	Q00J/Q00/Q01
	Installed Q base presence/ absence	Base type differentiation 0: Base not installed 1: Q**B is installed	 b7 b2 b1 b0 Fixed to 0 to Main base unit 1st extension base 2nd extension base to 7th extension base Fixed to 0 when the base is not installed.	S (Initial)	New	QnU
SD243 SD244	No. of base slots	No. of base slots	b15 to b12 / b11 to b8 / b7 to b4 / b3 to b0 SD243: Extension 3 / Extension 2 / Extension 1 / Main SD244: Extension 7 / Extension 6 / Extension 5 / Extension 4 • As shown above, each area stores the number of slots being installed.	S (Initial)	New	Qn(H) QnPH QnPRH QnU Rem
SD243 SD244	No. of base slots (Operation status)	No. of base slots	b15 to b12 / b11 to b8 / b7 to b4 / b3 to b0 SD243: Extension 3 / Extension 2 / Extension 1 / Main SD244: Fixed to 0 / Fixed to 0 / Fixed to 0 / Extension 4 • As shown above, each area stores the number of slots being installed. (Number of set slots when parameter setting has been made)	S (Initial)	New	Q00J/Q00/Q01
SD245 SD246	No. of base slots (Mounting status)	No. of base slots	b15 to b12 / b11 to b8 / b7 to b4 / b3 to b0 SD245: Extension 3 / Extension 2 / Extension 1 / Main SD246: Fixed to 0 / Fixed to 0 / Fixed to 0 / Extension 4 • As shown above, each area stores the number of module-mounted slots of the base unit (actual number of slots of the installed base unit).	S (Initial)		Q00J/Q00/Q01*9
SD250	Loaded maximum I/O	Loaded maximum I/O No.	• When SM250 goes from OFF to ON, the upper 2 digits of the final I/O number plus 1 of the modules loaded are stored as BIN values.	S (Request END)	New	Qn(H) QnPH QnPRH Rem
			• The upper 2 digits of the final I/O number plus 1 of the modules loaded are stored as BIN values.	S (Initial)	New	Q00J/Q00/Q01 QnU Rem
SD251	Head I/O number for replacement	Head I/O No. for module replacement	• Stores the upper two digits of the head I/O number of an I/O module that is removed/replaced in the online status (with power on). (Default value: 100H).	U	D9094	Q2A(S1) Q3A Q4A Q4AR
SD253	RS422 transmission speed	RS422 transmission speed	• Stores transmission speed of RS422. 0 : 9600bps 1 : 19.2kbps 2 : 38.4kbps	S (When changed)	New	QnA

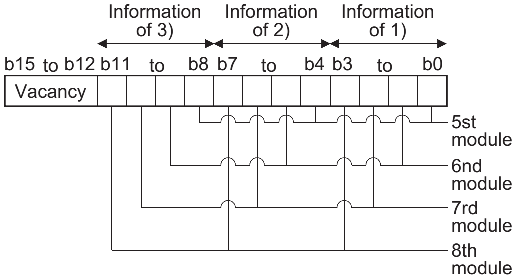
*9: Function version is B or later.

TableApp.21 Special register

Number	Name	Meaning		Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD254	MELSECNET/10. MELSECNET/H information	Number of modules installed		• Indicates the number of mounted MELSECNET/10 modules or MELSECNET/H modules.	S (Initial)	New	○
SD255		Information from 1st module	I/O No.	• Indicates I/O number of mounted MELSECNET/10 module or MELSECNET/H module			
SD256			Network No.	• Indicates network No. of mounted MELSECNET/10 module or MELSECNET/H module			
SD257			Group number	• Indicates group No. of mounted MELSECNET/10 module or MELSECNET/H module			
SD258			Station No.	• Indicates station No. of mounted MELSECNET/10 module or MELSECNET/H module			
SD259			Standby informa-tion	• In the case of standby stations, the module number of the standby station is stored. (1 to 4)			QnA Qn(H) QnPH QnPRH QnU
SD260 to SD264		Information from 2nd module		• Configuration is identical to that for the first module.			
SD265 to SD269		Information from 3rd module		• Configuration is identical to that for the first module.			QnA Qn(H) QnPH QnPRH QnU*10
SD270 to SD274		Information from 4th module		• Configuration is identical to that for the first module.			
SD280	CC-Link error	Error detection status		1) When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON). 2) When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON). 3) Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.  Information of 3) Information of 2) Information of 1) b15 to b12 b11 to b8 b7 to b4 b3 to b0 Vacancy 1st module 2nd module 3rd module 4th module The above module Nos. n are in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)	S (Error)	New	Qn(H) QnPH QnPRH Rem
				• When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON). • When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON). • Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.  b15 to b9 b8 to b0 8th module 1st module 8th module 1st module Information of 2) Information of 1)	S (Error)	New	QnA

*10: The Universal model QCPU except the Q02UCPU.

TableApp.21 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD281	CC-Link error	Error detection status	1) When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON). 2) When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON). 3) Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.  Information of 3) Information of 2) Information of 1) b15 to b12 b11 to b8 b7 to b4 b3 to b0 Vacancy 5st module 6nd module 7rd module 8th module The above module Nos. n are in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)	S (Error)	New	Qn(H)*14 QnPH*14 QnPRH*15
SD286	Device assignment	Points assigned to M (for extension)	• The number of points assigned to M is stored with 32 bits. • Even if the points assigned to M are 32K points or less, the points are stored	S (Initial)	New	QnU*16
SD287						
SD288		Points assigned to B (for extension)	• The number of points assigned to B is stored with 32 bits. • Even if the points assigned to B are 32K points or less, the points are stored			
SD289						
SD290	Device assignment (Same as parameter contents)	Number of points assigned for X	• Stores the number of points currently set for X devices	S (Initial)	New	○ Rem
SD291		Number of points assigned for Y	• Stores the number of points currently set for Y devices			
SD292		Number of points assigned for M	• Stores the number of points currently set for M devices			
SD293		Number of points assigned for L	• Stores the number of points currently set for L devices			○
SD294		Number of points assigned for B	• Stores the number of points currently set for B devices			○ Rem
SD295		Number of points assigned for F	• Stores the number of points currently set for F devices			○
SD296		Number of points assigned for SB	• Stores the number of points currently set for SB devices			○ Rem
SD297	Device assignment (Same as parameter contents)	Number of points assigned for V	• Stores the number of points currently set for V devices	S (Initial)	New	○
SD298		Number of points assigned for S	• Stores the number of points currently set for S devices			
SD299		Number of points assigned for T	• Stores the number of points currently set for T device			
SD300		Number of points assigned for ST	• Stores the number of points currently set for ST devices			
SD301		Number of points assigned for C	• Stores the number of points currently set for C devices			
SD302		Number of points assigned for D	• Stores the number of points currently set for D devices			○ Rem
SD303		Number of points assigned for W	• Stores the number of points currently set for W devices			
SD304		Number of points assigned for SW	• Stores the number of points currently set for SW devices			

*14: The module whose first 5 digits of serial No. is "08032" or later.

*15: The module whose first 5 digits of serial No. is "09102" or later.

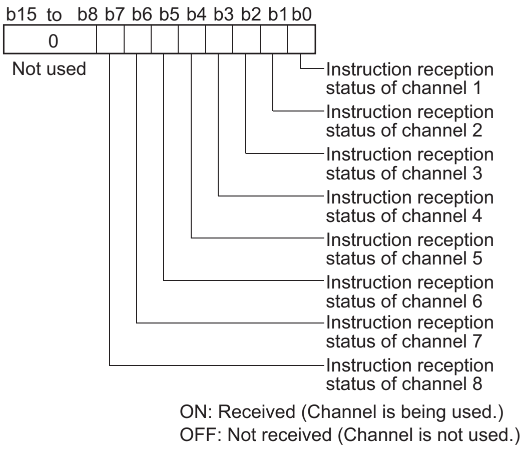
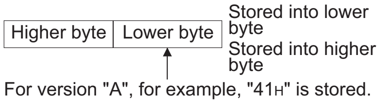
*16: The module whose first 5 digits of serial No. is "10042" or later.

TableApp.21 Special register

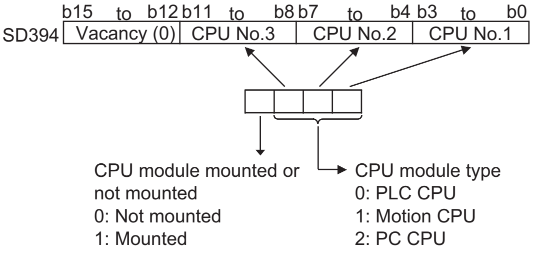
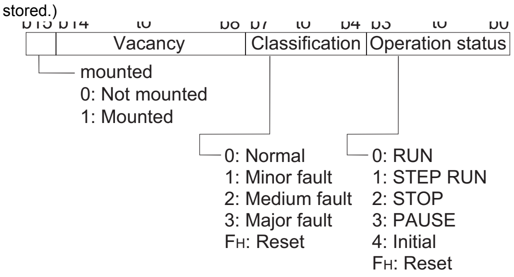
Number	Name	Meaning		Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD305	Device assignment (Index register)	16 bit modification Number of points assigned for Z		• Stores the number of points of index register (Z) to be modified in the range of 16 bits. (The assignment is set by the ZR device index modification setting parameter.)	S (Initial)	New	QnU
SD306	Device assignment (Same as parameter contents)	Number of points assigned for ZR (extension)		• Stores the number of points set for ZR device (excluding the number of points set for the extended data register (D) and extended link register (W)). The number of points is stored in this SD only when 1k or more points are set for the extended data register (D) and extended link register (W).	S (Initial)	New	QnU
SD307							
SD308	Device assignment (Including extended data register (D) and extended link register (W))	Number of points assigned for D (internal + extension)		• Stores the total number of points set for the data register (D), which is an internal user device, and the extended data register (D) (in 32-bit binary data).			
SD309							
SD310		Number of points assigned for W (internal + extension)		• Stores the total number of points set for the link register (W), which is an internal user device, and the extended link register (D) (in 32-bit binary data).			
SD311							
SD315	Time reserved for communication processing	Time reserved for communication processing		• Reserves the designated time for communication processing with GX Developer or other units. • The greater the value is designated, the shorter the response time for communication with other devices (GX Developer, serial communication units) becomes. • If the designated value is out of the range above, it is processed that no setting is made. • Setting range: 1 to 100 ms • Note that the scan time becomes longer by the designated time.	U	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH Rem
SD340	Ethernet information	No. of modules installed		• Indicates the number of mounted Ethernet module.	S (Initial)	New	QCPU Rem
SD341		Information of 1st module	I/O No.	• Indicates I/O No. of mounted Ethernet module			
SD342			Network No.	• Indicates network No. of mounted Ethernet module			
SD343			Group No.	• Indicates group No. of mounted Ethernet module			
SD344			Station No.	• Indicates station No. of mounted Ethernet module			
SD345 to SD346			Empty	• Empty (With QCPU, the Ethernet module IP address of the 1st module is stored in buffer memory.)			Qn(H) QnPH QnPRH QnU Rem
SD347			Empty	• Empty (With QCPU, the Ethernet module error code of the 1st module is read with the ERRRD instruction.)			
SD348 to SD354	Ethernet information	Information from 2nd module		• Configuration is identical to that for the first module.	S (Initial)	New	
SD355 to SD361		Information from 3rd module		• Configuration is identical to that for the first module.			Qn(H) QnPH QnPRH QnU*10 Rem
SD362 to SD368		Information from 4th module		• Configuration is identical to that for the first module.			

*10: The Universal model QCPU except the Q02UCPU.

TableApp.21 Special register

Number	Name	Meaning		Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD340	Ethernet information	No. of modules installed		• Indicates the number of mounted Ethernet module.	S (Initial)	New	QnA
SD341		Information of 1st module	I/O No.	• Indicates I/O No. of mounted Ethernet module			
SD342			Network No.	• Indicates network No. of mounted Ethernet module			
SD343			Group No.	• Indicates group No. of mounted Ethernet module			
SD344			Station No.	• Indicates station No. of mounted Ethernet module			
SD345 to SD346			IP address	• Indicates IP address of mounted Ethernet module			
SD347			Error code	• Indicates error code of mounted Ethernet module			
SD348 to SD354		Information from 2nd module		• Configuration is identical to that for the first module.			
SD355 to SD361		Information from 3rd module		• Configuration is identical to that for the first module.			
SD362 to SD368		Information from 4th module		• Configuration is identical to that for the first module.			
SD380	Ethernet instruction reception status	Instruction reception status of 1st module		 b15 to b8 b7 b6 b5 b4 b3 b2 b1 b0 0 Not used Instruction reception status of channel 1 Instruction reception status of channel 2 Instruction reception status of channel 3 Instruction reception status of channel 4 Instruction reception status of channel 5 Instruction reception status of channel 6 Instruction reception status of channel 7 Instruction reception status of channel 8 ON: Received (Channel is being used.) OFF: Not received (Channel is not used.)	S (Initial) S (Instruction execution)	New	QnA QnPRH
SD381	Ethernet instruction reception status	Instruction reception status of 2nd module		• Configuration is identical to that for the first module.	S (Instruction execution)	New	QnPRH
SD382		Instruction reception status of 3rd module		• Configuration is identical to that for the first module.			
SD383		Instruction reception status of 4th module		• Configuration is identical to that for the first module.			
SD392	Software version	Internal system software version		• Stores the internal system software version in ASCII code.  Higher byte Lower byte Stored into lower byte Stored into higher byte For version "A", for example, "41H" is stored. Note: The internal system software version may differ from the version indicated by the version symbol printed on the case.	S (Initial)	D9060	QnA

TableApp.21 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD393	Multiple CPU system information	Number of multiple CPUs	• The number of CPU modules that comprise the multiple CPU system is stored. (1 to 3, Empty also included)	S (Initial)	New	Q00/Q01*9 QnU
SD394		CPU mounting information	• The CPU module types of No. 1 CPU to 3 and whether the CPU modules are mounted or not are stored.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 SD394 Vacancy (0) CPU No.3 CPU No.2 CPU No.1 CPU module mounted or not mounted 0: Not mounted 1: Mounted CPU module type 0: PLC CPU 1: Motion CPU 2: PC CPU			Q00/Q01*9
SD395		Multiple CPU number	• In a multiple CPU system configuration, the CPU number of the host CPU is stored. CPU No. 1: 1, CPU No. 2: 2, CPU No. 3: 3, CPU No. 4: 4	S (Initial)	New	Q00/Q01*9 Qn(H)*9 QnPH QnU
SD396		No. 1 CPU operation status	The operation information of each CPU No. is stored. (The information on the number of multiple CPUs indicated in SD393 is stored.)  b15 b14 to b8 b7 to b4 b3 to b0 Vacancy Classification Operation status mounted 0: Not mounted 1: Mounted 0: Normal 1: Minor fault 2: Medium fault 3: Major fault FH: Reset 0: RUN 1: STEP RUN 2: STOP 3: PAUSE 4: Initial FH: Reset	S (END processing error)	New	Q00/Q01*9 QnU
SD397		No. 2 CPU operation status				
SD398		No. 3 CPU operation status				
SD399		No. 4 CPU operation statu				QnU

*9: Function version is B or later.

(3) System clocks/counters

TableApp.22 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD412	1 second counter	Number of counts in 1-second units	• Following programmable controller CPU module RUN, 1 is added each second • Count repeats from 0 to 32767 to -32768 to 0	S (Status change)	D9022	○
SD414	2n second clock setting	2n second clock units	• Stores value n of 2n second clock (Default is 30) • Setting can be made between 1 and 32767	U	New	
SD415	2nms clock setting	2nms clock units	• Stores value n of 2nms clock (Default is 30) • Setting can be made between 1 and 32767	U	New	Qn(H) QnPH QnPRH QnU
SD420	Scan counter	Number of counts in each scan	• Incremented by 1 for each scan execution after the CPU module is set to RUN. (Not counted by the scan in an initial execution type program.) • Count repeats from 0 to 32767 to -32768 to 0	S (Every END processing)	New	QnA Qn(H) QnPH QnPRH QnU
			• Incremented by 1 for each scan execution after the CPU module is set to RUN. • Count repeats from 0 to 32767 to -32768 to 0	S (Every END processing)	New	Q00J/Q00/Q01
SD430	Low speed scan counter	Number of counts in each scan	• Incremented by 1 for each scan execution after the CPU module is set to RUN. • Count repeats from 0 to 32767 to -32768 to 0 • Used only for low speed execution type programs	S (Every END processing)	New	QnA Qn(H) QnPH

(4) Scan information

TableApp.23 Special register

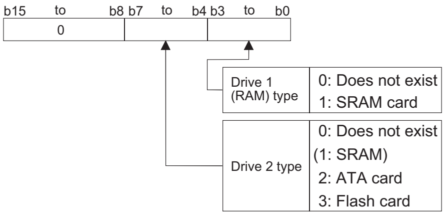
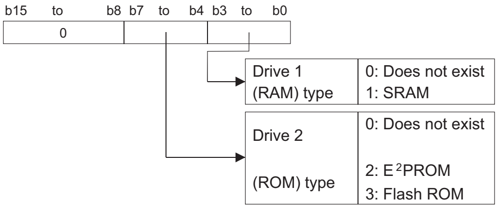
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD500	Execution program No.	Program No. in execution	• Program number of program currently being executed is stored as BIN value.	S (Status change)	New	QnA Qn(H) QnPH QnPRH QnU
SD510	Low speed excution type program No.	Low speed execution type program No. in execution	• Program number of low speed excution type program No. currently being executed is stored as BIN value. • Enabled only when SM510 is ON.	S (Every END processing)	New	QnA Qn(H) QnPH
SD520	Current scan time	Current scan time (in 1 ms units)	• The current scan time is stored into SD520 and SD521. (Measurement is made in 100 μs units. (For the Universal model QCPU, in 1μþs units.)) SD520: Stores the ms place. (Storage range: 0 to 65535) SD521: Stores the μs place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999)) (Example) When the current scan time is 23.6ms, the following values are stored. SD520 = 23 SD521 = 600	S (Every END processing)	D9017 format change	QnA Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU
SD521		Current scan time (in 100 μs units)		S (Every END processing)	New	
SD522	Initial scan time	Initial scan time (in 1 ms units)	• Stores the scan time of an initial execution type program into SD522 and SD523. (Measurement is made in 100 μs units. (For the Universal model QCPU, in 1μs units.)) SD522: Stores the ms place. (Storage range: 0 to 65535) SD523: Stores the μs place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (First END processing)	New	QnA Qn(H) QnPH QnPRH QnU
SD523		Initial scan time (in 100 μs units)				
SD524	Minimum scan time	Minimum scan time (in 1 ms units)	• Stores the minimum value of the scan time except that of an initial execution type program into SD524 and SD525. (Measurement is made in 100 μs units. (For the Universal model QCPU, in 1μs units.)) SD524: Stores the ms place. (Storage range: 0 to 65535) SD525: Stores the μs place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	D9018 format change	
SD525		Minimum scan time (in 100 μs units)		S (Every END processing)	New	
SD526	Maximum scan time	Maximum scan time (in 1 ms units)	• Stores the maximum value of the scan time except that of an initial execution type program into SD526 and SD527. (Measurement is made in 100 μs units. (For the Universal model QCPU, in 1μs units.)) SD526: Stores the ms place. (Storage range: 0 to 65535) SD527: Stores the μs place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	D9019 format change	
SD527		Maximum scan time (in 100 μs units)			New	
SD528	Current scan time for low speed execution type programs	Current scan time (in 1 ms units)	• Stores the current scan time of a low speed execution type program into SD528 and SD529. (Measurement is made in 100 μs units.) SD528: Stores the ms place. (Storage range: 0 to 65535) SD529: Stores the μs place. (Storage range: 0 to 900)	S (Every END processing)	New	QnA Qn(H) QnPH
SD529		Current scan time (in 100 μs units)				
SD532	Minimum scan time for low speed execution type programs	Minimum scan time (in 1 ms units)	• Stores the minimum value of the scan time of a low speed execution type program into SD532 and SD533. (Measurement is made in 100 μs units.) SD532: Stores the ms place. (Storage range: 0 to 65535) SD533: Stores the μs place. (Storage range: 0 to 900)	S (Every END processing)	New	
SD533		Minimum scan time (in 100 μs units)				
SD534	Maximum scan time for low speed execution type programs	Maximum scan time (in 1 ms units)	• Stores the maximum value of the scan time except that of the first scan of a low speed execution type program into SD534 and SD535. (Measurement is made in 100 μs units.) SD534: Stores the ms place. (Storage range: 0 to 65535) SD535: Stores the μs place. (Storage range: 0 to 900)	S (Every END processing)	New	
SD535		Maximum scan time (in 100 μs units)				
SD540	END processing time	END processing time (in 1 ms units)	• Stores the time from the end of a scan execution type program to the start of the next scan into SD540 and SD541. (Measurement is made in 100 μs units.(For the Universal model QCPU, in 1 μs units.)) SD540: Stores the ms place. (Storage range: 0 to 65535) SD541: Stores the μs place. (Storage range: 0 to 900) (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	New	QnA Qn(H) QnPH QnPRH QnU
SD541		END processing time (in 100 μs units)				

TableApp.23 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD524	Minimum scan time	Minimum scan time (in 1 ms units)	• Stores the minimum value of the scan time into SD524 and SD525. (Measurement is made in 100 μs units.) SD524: Stores the ms place. (Storage range: 0 to 65535) SD525: Stores the μs place. (Storage range: 0 to 900)	S (Every END processing)	New	Q00J/Q00/Q01
SD525		Minimum scan time (in 100 μs units)				
SD526	Maximum scan time	Maximum scan time (in 1 ms units)	• Stores the maximum value of the scan time into SD526 and SD527. (Measurement is made in 100 μs units.) SD526: Stores the ms place. (Storage range: 0 to 65535) SD527: Stores the μs place. (Storage range: 0 to 900)	S (Every END processing)		
SD527		Maximum scan time (in 100 μs units)				
SD540	END processing time	END processing time (in 1 ms units)	• Stores the time from when the scan program ends until the next scan starts into SD540 and SD541. (Measurement is made in 100 μs units.) SD540: Stores the ms place. (Storage range: 0 to 65535) SD541: Stores the μs place. (Storage range: 0 to 900)	S (Every END processing)	New	Q00J/Q00/Q01
SD541		END processing time (in 100 μs units)				
SD542	Constant scan wait time	Constant scan wait time (in 1 ms units)	• Stores the wait time for constant scan setting into SD542 and SD543. (Measurement is made in 100 μs units. (For the Universal model QCPU, in 1μs units.)) SD542: Stores the ms place. (Storage range: 0 to 65535) SD543: Stores the μs place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	New	○
SD543		Constant scan wait time (in 100 μs units)				
SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	• Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100 μs units.) SD544: Stores the ms place. (Storage range: 0 to 65535) SD545: Stores the μs place. (Storage range: 0 to 900) • Cleared to 0 after the end of one low speed scan.	S (Every END processing)	New	QnA Qn(H) QnPH
SD545		Cumulative execution time for low speed execution type programs (in 100 μs units)				
SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	• Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 μs units.) SD546: Stores the ms place. (Storage range: 0 to 65535) SD547: Stores the μs place. (Storage range: 0 to 900) • Stored every scan.	S (Every END processing)	New	
SD547		Execution time for low speed execution type programs (in 100 μs units)				
SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	• Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 μs units.) SD548: Stores the ms place. (Storage range: 0 to 65535) SD549: Stores the μs place. (Storage range: 0 to 900) • Stored every scan.	S (Every END processing)	New	QnA Qn(H) QnPH QnPRH
SD549		Scan execution type program execution time (in 100 μs units)				
SD548	Scan program execution time	Scan program execution time (in 1 ms units)	• Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 μs units. (For the Universal model QCPU, in 1μs units.)) SD548: Stores the ms place. (Storage range: 0 to 65535) SD549: Stores the μs place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999)) • Stored every scan.	S (Every END processing)	New	Q00J/Q00/Q01 QnU
SD549		Scan program execution time (in 100 μs units)				
SD550	Service interval measurement module	Unit/module No.	• Sets I/O number for module that measures service interval.	U	New	QnA Qn(H) QnPH QnPRH
SD551	Service interval time	Module service interval (in 1 ms units)	• Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 μs units.) SD551: Stores the ms place. (Storage range: 0 to 65535) SD552: Stores the μs place. (Storage range: 0 to 900)	S (Request)	New	
SD552		Module service interval (in 100 μs units)				

(5) Drive information

TableApp.24 Special register

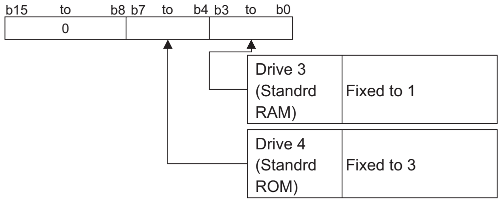
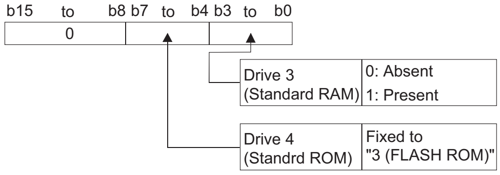
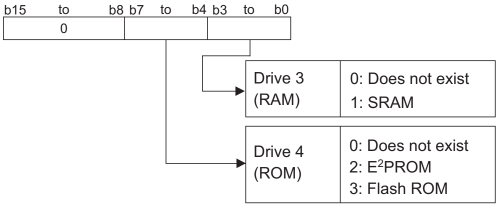
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD600	Memory card typs	Memory card typs	• Indicates the type of the memory card installed.  b15 to b8 b7 to b4 b3 to b0 0 Drive 1 (RAM) type 0: Does not exist 1: SRAM card Drive 2 type 0: Does not exist (1: SRAM) 2: ATA card 3: Flash card	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU
	Memory card A typs	Memory card A typs	• Indicates the type of memory card A installed.  b15 to b8 b7 to b4 b3 to b0 0 Drive 1 (RAM) type 0: Does not exist 1: SRAM Drive 2 (ROM) type 0: Does not exist 2: E²PROM 3: Flash ROM	S (Initial and card removal)	New	QnA
SD602	Drive 1 (Memory card RAM) capacity	Drive 1 capacity	• Drive 1 capacity is stored in 1 k byte units. • (Empty capacity after format is stored.)	S (Initial and card removal)	New	QnA Qn(H) QnPH QnPRH QnU
SD603	Drive 2 (Memory card ROM) capacity	Drive 2 capacity	• Drive 2 capacity is stored in 1 k byte units.[*1] • (Empty capacity after format is stored.)	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU
			• Drive 2 capacity is stored in 1 k byte units.	S (Initial and card removal)	New	QnA

[*1] When the Q2MEM-8MBA is used, value stored in the special register SD603 differs depending on the combination of the serial number of the High Performance model QCPU and the manufacture control number of the ATA card.
For details, refer to QCPU User's Manual (Hardware Design, Maintenance and Inspection).

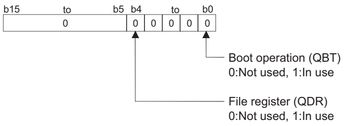
TableApp.24 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD604	Memory card use conditions	Memory card use conditions	• The use conditions for memory card (A) are stored as bit patterns . (In use when ON) • The significance of these bit patterns is indicated below: b0 : Boot operation (QBT) b1 : Parameters (QPA) b2 : Device comments (QCD) b3 : Device initial value (QDI) b4 : File register (QDR) b5 : Sampling trace (QTD) b6 : Not used b7 : Not used b8 : Not used b9 : CPU fault history (QFD) b10 : Not used b11 : Local device (QDL) b12 : Not used b13 : Not used b14 : Not used b15 : Not used	S (Status change)	New	Qn(H) QnPH QnPRH
	Memory card A use conditions	Memory card A use conditions	• The use conditions for memory card A are stored as bit patterns. (In use when ON) • The significance of these bit patterns is indicated below: b0 : Boot operation (QBT) b1 : Parameters (QPA) b2 : Device comments (QCD) b3 : Device initial value (QDI) b4 : File R (QDR) b5 : Sampling trace (QTD) b6 : Status latch (QTL) b7 : Program trace (QTP) b8 : Simulation data (QDS) b9 : CPU fault history (QFD) b10 : SFC trace (QTS) b11 : Local device (QDL) b12 : Not used b13 : Not used b14 : Not used b15 : Not used	S (Status change)	New	QnA
	Memory card use conditions	Memory card use conditions	• The use conditions for memory card (A) are stored as bit patterns . (In use when ON) • The significance of these bit patterns is indicated below: b0 : Boot operation (QBT)*1 b1 : Parameters (QPA) b2 : Device comments (QCD) b3 : Device initial value (QDI)*2 b4 : File register (QDR) b5 : Sampling trace (QTD) b6 : Not used b7 : Not used b8 : Not used b9 : Not used b10 : Not used b11 : Local device (QDL) b12 : Not used b13 : Not used b14 : Not used b15 : Not used *1: Turned ON at boot start and OFF at boot completion. *2: Turned ON when reflection of device initial value is started and OFF when reflection of device initial value is completed.	S (Status change)	New	QnU

TableApp.24 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD620	Drive 3/4 typs	Drive 3/4 typs	• Indicates the drive 3/4 type.  b15 to b8 b7 to b4 b3 to b0 0 Drive 3 (Standrd RAM) Fixed to 1 Drive 4 (Standrd ROM) Fixed to 3	S (Initial)	New	Qn(H) QnPH QnPRH QnU
			• Indicates the drive 3/4 type.  b15 to b8 b7 to b4 b3 to b0 0 Drive 3 (Standard RAM) 0: Absent 1: Present Drive 4 (Standrd ROM) Fixed to "3 (FLASH ROM)"	S (Initial)	New	Q00J/Q00/Q01
	Memory card B typs	Memory card B typs	• Indicates memory card B type installed  b15 to b8 b7 to b4 b3 to b0 0 Drive 3 (RAM) 0: Does not exist 1: SRAM Drive 4 (ROM) 0: Does not exist 2: E2PROM 3: Flash ROM	S (Initial/Card installation and removal)	New	Q2A(S1) Q3A Q4A Q4AR
SD622	Drive 3 (Standard RAM) capacity	Drive 3 capacity	• Drive 3 capacity is stored in 1 k byte units. (Empty capacity after format is stored.)	S (Initial)	New	Qn(H) QnPH QnPRH QnU
				S (Initial/Card installation and removal)	New	Q2A(S1) Q3A Q4A Q4AR
SD623	Drive 4 (Standard ROM) capacity	Drive 4 capacity	• Drive 4 capacity is stored in 1 k byte units. (Empty capacity after format is stored.)	S (Initial)	New	Qn(H) QnPH QnPRH QnU
				S (Initial/Card installation and removal)	New	Q2A(S1) Q3A Q4A Q4AR

TableApp.24 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD624	Drive 3/4 use conditions	Drive 3/4 use conditions	• The conditions for usage for drive 3/4 are stored as bit patterns. (In use when ON) • The significance of these bit patterns is indicated below: b0 : Boot operation (QBT) b1 : Parameters (QPA) b2 : Device comments (QCD) b3 : Device initial value (QDI) b4 : File register (QDR) b5 : Sampling trace (QTD) b6 : Not used b7 : Not used b8 : Not used b9 : CPU fault history (QFD) b10 : SFC trace (QTS) b11 : Local device (QDL) b12 : Not used b13 : Not used b14 : Not used b15 : Not used	S (Status change)	New	Qn(H) QnPH QnPRH
	Memory card B use conditions	Memory card B use conditions	• The use conditions for memory card B are stored as bit patterns. (In use when ON) • The significance of these bit patterns is indicated below: b0 : Boot operation (QBT) b1 : Parameters (QPA) b2 : Device comments (QCD) b3 : Device initial value (QDI) b4 : File register (QDR) b5 : Sampling trace (QTD) b6 : Status latch (QTL) b7 : Program trace (QTP) b8 : Simulation data (QDS) b9 : CPU fault history (QFD) b10 : SFC trace (QTS) b11 : Local device (QDL) b12 : Local device (QDL) b13 : Not used b14 : Not used b15 : Not used	S (Status change)	New	Q2A(S1) Q3A Q4A Q4AR
	Drive 3/4 use conditions	Drive 3/4 use conditions	• The conditions for usage for drive 3/4 are stored as bit patterns. (In use when ON) • The significance of these bit patterns is indicated below: b0 : Not used b1 : Parameters (QPA) b2 : Device comments (QCD) b3 : Device initial value (QDI)[*1] b4 : File register (QDR) b5 : Sampling trace (QTD) b6 : Not used b7 : Not used b8 : Not used b9 : Not used b10 : Not used b11 : Local device (QDL) b12 : Not used b13 : Not used b14 : Not used b15 : Not used *1: Turned ON at boot start and OFF at boot completion.	S (Status change)	New	QnU
SD622	Drive 3 (Standard RAM) capacity	Drive 3 capacity	• Drive 3 capacity is stored in 1k byte units.	S (Initial)	New	Q00J/Q00/Q01
SD623	Drive 4 (Standard ROM) capacity	Drive 4 capacity	• Drive 4 capacity is stored in 1k byte units.	S (Initial)		
SD624	Drive 3/4 use conditions	Drive 3/4 use conditions	• The conditions for usage for drive 3/4 are stored as bit patterns.  b15 to b5 b4 to b0 0 0 0 0 0 0 Boot operation (QBT) 0:Not used, 1:In use File register (QDR) 0:Not used, 1:In use	S (Status change)		
SD640	File register drive	Drive number:	• Stores drive number being used by file register	S (Status change) *10	New	○

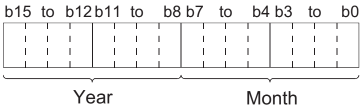
*10: On the Basic model QCPU, data is set at STOP to RUN or RSET instruction execution after parameter execution.

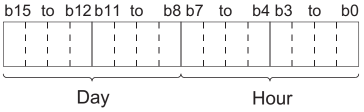
TableApp.24 Special register

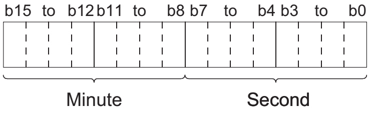
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD641 SD642 SD643 SD644 SD645 SD646	File register file name	File register file name	• Stores file register file name (with extension) selected at parameters or by use of QDRSET instruction as ASCII code. b15 to b8 / b7 to b0 SD641: 2nd character / 1st character SD642: 4th character / 3rd character SD643: 6th character / 5th character SD644: 8th character / 7th character SD645: 1st character of extension / 2EH(.) SD646: 3rd character of the extension / 2nd character of the extension	S (Status change)	New	QnA Qn(H) QnPH QnPRH QnU
			• Stores file register file name (MAIN.QDR) selected at parameters as ASCII code. b15 to b8 / b7 to b0 SD641: 2nd character (A) / 1st character (M) SD642: 4th character (N) / 3rd character (I) SD643: 6th character ( ) / 5th character ( ) SD644: 8th character ( ) / 7th character ( ) SD645: 1st character of the extension (Q) / 2EH(.) SD646: 3rd character of the extension (R) / 2nd character of the extension (D)	S (Initial)	New	Q00J/Q00/Q01
SD647	File register capacity	File register capacity	• Stores the data capacity of the currently selected file register in 1 k word units.	S (Status change)	New	QnA Qn(H) QnPH QnPRH QnU
				S (Initial)		Q00J/Q00/Q01
SD648	File register block number	File register block number	• Stores the currently selected file register block number.	S (Status change) *10	D9035	○
SD650	Comment drive	Comment drive number	• Stores the comment drive number selected at the parameters or by the QCDSET instruction.	S (Status change)	New	QnA Qn(H) QnPH QnPRH QnU
SD651 SD652 SD653 SD654 SD655 SD656	Comment file name	Comment file name	• Stores the comment file name (with extension) selected at the parameters or by the QCDSET instruction in ASCII code. b15 to b8 / b7 to b0 SD651: 2nd character / 1st character SD652: 4th character / 3rd character SD653: 6th character / 5th character SD654: 8th character / 7th character SD655: 1st character of the extension / 2EH(.) SD656: 3rd character of the extension / 2nd character of the extension	S (Status change)	New	
SD660	Boot operation designation file	Boot designation file drive number	• Stores the drive number where the boot designation file (*.QBT) is being stored.	S (Initial)	New	QnA Qn(H) QnPH QnPRH QnU
SD661 SD662 SD663 SD664 SD665 SD666		File name of boot designation file	• Stores the file name of the boot designation file (*.QBT). b15 to b8 / b7 to b0 SD661: 2nd character / 1st character SD662: 4th character / 3rd character SD663: 6th character / 5th character SD664: 8th character / 7th character SD665: 1st character of the extension / 2EH(.) SD666: 3rd character of the extension / 2nd character of the extension	S (Initial)	New	

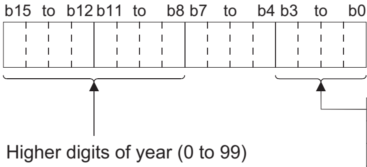
*10: On the Basic model QCPU, data is set at STOP to RUN or RSET instruction execution after parameter execution.

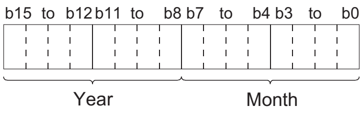
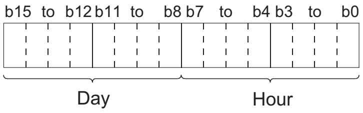
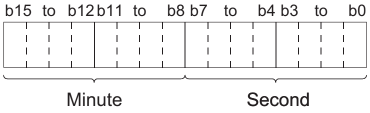
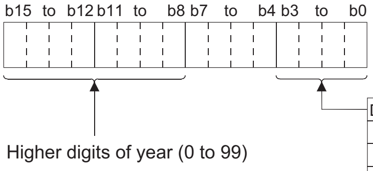
TableApp.24 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD670</td><td>Parameter enable drive information</td><td>Parameter enable drive No.</td><td>• Stores information of parameter storage destination drive which is enabled.
0: Drive 0 (Program memory)
1: Drive 1 (SRAM card)
2: Drive 2 (Flash card/ATA card)
4: Drive 4 (Standard ROM)</td><td>S (Initial)</td><td>New</td><td>QnU</td></tr>
<tr><td>SD671</td><td>Status of latch data backup function</td><td>Status display</td><td>Indicates the status of the latch data backup function.
<table>
<tr><th colspan="2">Status</th><th>Presence/ absence of backup data</th><th>Restore operation at turning power supply ON from OFF</th></tr>
<tr><td>0</td><td>No backup data</td><td>Absent</td><td>Restoring not executed</td></tr>
<tr><td>1</td><td>Restore ready completion</td><td rowspan="4">Present</td><td>Restoring executed when turning power supply ON from OFF the following time</td></tr>
<tr><td>2</td><td>Restore execution completion</td><td>Restoring not executed</td></tr>
<tr><td>3</td><td>Backup execution wait</td><td>Restoring not executed</td></tr>
<tr><td>4</td><td>Restore repeated execution ready completion</td><td>Restoring executed when turning power supply ON from OFF</td></tr>
</table>
• "2 Restore ready completion" is a status immediately after restoring data.
"3 Backup execution wait" is a status after turning power supply ON from OFF at "2 Restore ready completion".</td><td>S (Status change)</td><td>New</td><td>QnU</td></tr>
<tr><td>SD672</td><td rowspan="4">Backup information</td><td>Backup time (Year and month)</td><td>• Stores the last 2 digits of year and month when backup is performed in 2-digit BCD code.


Example:
July, 1993
9307H</td><td rowspan="4">S (At write)</td><td rowspan="4">New</td><td rowspan="4">QnU</td></tr>
<tr><td>SD673</td><td>Backup time (Day and hour)</td><td>• Stores the day and hour when backup is performed in 2-digit BCD code.


Example:
31st, 10 a.m.
3110H</td></tr>
<tr><td>SD674</td><td>Backup time (Minute and second)</td><td>• Stores the minute and second when backup is performed in 2-digit BCD code.


Example:
35 min., 48 sec.
3548H</td></tr>
<tr><td>SD675</td><td>Backup time (Year and day of week)</td><td>• Stores the first 2 digits of year and day of week when backup is performed in BCD code.

b15 to b12 b11 to b8 b7 to b4 b3 to b0
Higher digits of year (0 to 99)
Example:
1993, Friday
1905H
<table>
<tr><th colspan="2">Day of the week</th></tr>
<tr><td>0</td><td>Sunday</td></tr>
<tr><td>1</td><td>Monday</td></tr>
<tr><td>2</td><td>Tuesday</td></tr>
<tr><td>3</td><td>Wednesday</td></tr>
<tr><td>4</td><td>Thursday</td></tr>
<tr><td>5</td><td>Friday</td></tr>
<tr><td>6</td><td>Saturday</td></tr>
</table></td></tr>
</table>

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD676	Backup data restore information	Restore time (Year and month)	• Stores the last 2 digits of year and month when data is restored in 2-digit BCD code.  Example: July, 1993 9307H	S (Initial)	New	QnU
SD677		Restore time (Day and time)	• Stores the day and time when data is restored in 2-digit BCD code.  Example: 31st, 10 a.m. 3110H			
SD678		Restore time (Minute and second)	• Stores the minute and second when data is restored in 2-digit BCD code.  Example: 35 min., 48 sec. 3548H			
SD679		Restore time (Year and day of week)	• Stores the first 2 digits of year and day of week when data is restored in BCD code.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 Higher digits of year (0 to 99) Example: 1993, Friday 1905H Day of the week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday			
SD681	Program memory write (transfer) status	Write (transfer) status display (percentage)	Displays the status of writing (transferring) the program memory (flash ROM) in percentage. (0 to 100%) "0" is set when the write direction is set.	S (At write)	New	QnU
SD682	Program memory write count index	Write count index up to present	• Stores the index value for the number of write operations to the program memory (flash ROM) up to the present in BIN 32-bit value. When the index value exceeds 100 thousand times, "FLASH ROM ERROR" (error code: 1610) occurs. (The index value is calculated even when exceeding 100 thousand times.) Note) The write count does not equal to the index value.(Since a flash ROM write life is prolonged by the system, 1 is added to the write count index when writing is performed twice or so.).	S (At write)	New	QnU
SD683						
SD686	Standard ROM write (transfer) status	Write (transfer) status display (percentage)	Displays the status of writing (transferring) the standard ROM (flash ROM) in percentage. (0 to 100%) "0" is set when the write direction is set.	S (At write)	New	QnU
SD687	Standard ROM write count index	Write count index up to present	• Stores the index value for the number of write operations to the standard ROM (flash ROM) up to the present in BIN 32-bit value. When the index value exceeds 100 thousand times, "FLASH ROM ERROR" (error code: 1610) occurs. (The index value is calculated even when exceeding 100 thousand times.) Note) The write count does not equal to the index value. (Since a flash ROM write life is prolonged by the system, 1 is added to the write count index when the total write capacity after the previous count up reaches about 1M byte.)	S (At write)	New	QnU
SD688						
SD695	Specification of writing to standard ROM instruction count	Specification of writing to standard ROM instruction count	• Specifies the maximum number of executing the instruction (SP.DEVST) to write to the standard ROM per day. • When the number of execution of writing to the standard ROM instruction exceeds the number of times set by SD695, [OPERATION ERROR] (error code: 4113) occurs. • The setting range for SD695 is 0 to 32767. If 0 or value outside the range is set, [OPERATION ERROR] (error code: 4113) occurs when executing the writing to standard ROM instruction.	U	New	QnU

(6) Instruction-Related Registers

TableApp.25 Special register

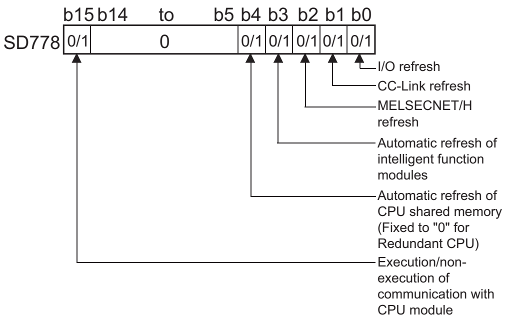
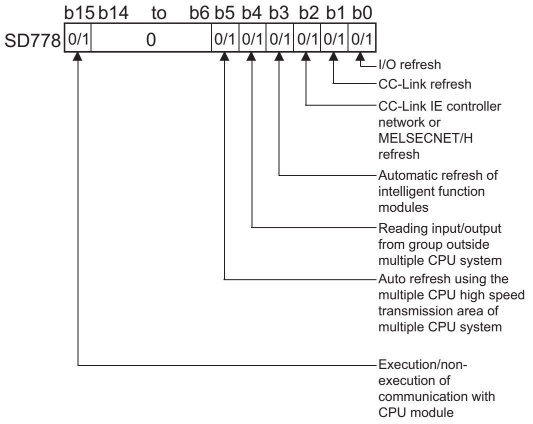
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD705	Mask pattern	Mask pattern	• During block operations, turning SM705 ON makes it possible to use the mask pattern being stored at SD705 (or at SD705 and SD706 if double words are being used) to operate on all data in the block with the masked values.	U	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnA
SD706						
SD714	Number of empty communication request registration areas	0 to 32	• Stores the number of empty blocks in the communications request area for remote terminal modules connected to the MELSECNET/MINI-S3.	S (During execution)	D9081	QnA
SD715	IMASK instruction mask pattern	Mask pattern	• Patterns masked by use of the IMASK instruction are stored in the following manner: b15 b1 b0 SD715 I15 to I1 I0 SD716 I31 to I17 I16 SD717 I47 to I33 I32	S (During execution)	New	○
SD716						
SD717						
SD718	Accumulator	Accumulator	• For use as replacement for accumulators used in A series programs.	S/U	New	
SD719						
SD720	Program No. designation for PLOADP instruction	Program No. designation for PLOADP instruction	Stores the program number of the program to be loaded by the PLOADP instruction when designated. Designation range: 1 to 124	U	New	Qn(H) QnPH
SD730	No. of empty areas for CC-Link communication reguest register area	0 to 32	• Stores the number of empty registration area for the request for communication with the intelligent device station connected to A(1S)J61QBT61.	S (During execution)	New	QnA
SD736	PKEY input	PKEY input	• Special register that temporarily stores keyboard data input by means of the PKEY instruction.	S (During execution)	New	

TableApp.25 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD738	Message storage	Message storage	• Stores the message designated by the MSG instruction. b15 to b8 b7 to b0 SD738 2nd character 1st character SD739 4th character 3rd character SD740 6th character 5th character SD741 8th character 7th character SD742 10th character 9th character SD743 12th character 11th character SD744 14th character 13th character SD745 16th character 15th character SD746 18th character 17th character SD747 20th character 19th character SD748 22nd character 21st character SD749 24th character 23rd character SD750 26th character 25th character SD751 28th character 27th character SD752 30th character 29th character SD753 32nd character 31st character SD754 34th character 33rd character SD755 36th character 35th character SD756 38th character 37th character SD757 40th character 39th character SD758 42nd character 41st character SD759 44th character 43rd character SD760 46th character 45th character SD761 48th character 47th character SD762 50th character 49th character SD763 52nd character 51st character SD764 54th character 53rd character SD765 56th character 55th character SD766 58th character 57th character SD767 60th character 59th character SD768 62nd character 61st character SD769 64th character 63rd character	S (During execution)	New	○
SD739						
SD740						
SD741						
SD742						
SD743						
SD744						
SD745						
SD746						
SD747						
SD748						
SD749						
SD750						
SD751						
SD752						
SD753						
SD754						
SD755						
SD756						
SD757						
SD758						
SD759						
SD760						
SD761						
SD762						
SD763						
SD764						
SD765						
SD766						
SD767						
SD768						
SD769						
SD774 to SD775	PID limit setting (for complete derivative)	0: With limit 1: Without limit	• Specify the limit of each PID loop as shown below. b15 b1 b0 SD774 Loop16 to Loop2 Loop1 SD775 Loop32 to Loop18 Loop17	U	New	Qn(H) QnPRH QnU
SD774	PID limit setting (for complete derivative)	0: With limit 1: Without limit	• Specify the limit of each PID loop as shown below. b15 to b8 b7 to b1 b0 SD774 Loop8 to Loop2 Loop1	U	New	Q00J/Q00/Q01[*9]

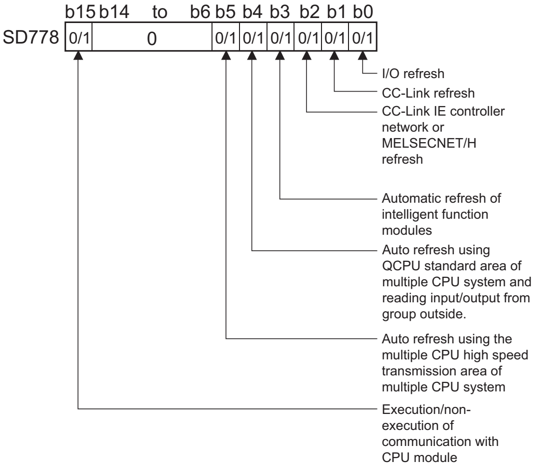
*9: Function version is B or later.

TableApp.25 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD778	Refresh processing selection when the COM instruction is executed	b0 to b14: 0: Do not refresh 1: Refresh b15 bit 0: Communication with CPU module is executed 1: Communication withCPU module is nonexecuted	• Selects whether or not the data is refreshed when the COM instruction is executed. • Designation of SD778 is made valid when SM775 turns ON.  b15 b14 to b5 b4 b3 b2 b1 b0 SD778 0/1 0 0/1 0/1 0/1 0/1 0/1 I/O refresh CC-Link refresh MELSECNET/H refresh Automatic refresh of intelligent function modules Automatic refresh of CPU shared memory (Fixed to "0" for Redundant CPU) Execution/non-execution of communication with CPU module • Refresh between multiple CPUs by COM instruction is performed under the following occasion. Receiving operation from other device: b4 of SD778(refresh in the CPU shared memory) is turned to 1. Sending operation from host CPU : b15 of SD778(communication with peripheral device is executed/nonexecuted) is turned to 0.	U	New	Q00J/Q00/Q01*9 Qn(H)*11
			• Selects whether or not the data is refreshed when the COM instruction is executed. • Designation of SD778 is made valid when SM775 turns ON.  b15 b14 to b6 b5 b4 b3 b2 b1 b0 SD778 0/1 0 0/1 0/1 0/1 0/1 0/1 0/1 I/O refresh CC-Link refresh CC-Link IE controller network or MELSECNET/H refresh Automatic refresh of intelligent function modules Reading input/output from group outside multiple CPU system Auto refresh using the multiple CPU high speed transmission area of multiple CPU system Execution/non-execution of communication with CPU module • Refresh between multiple CPUs by COM instruction is performed under the following occasion. Receiving operation from other device: b4 of SD778(refresh in the CPU shared memory) is turned to 1. Sending operation from host CPU : b15 of SD778(communication with peripheral device is executed/nonexecuted) is turned to 0. • When b2 (refresh of the CC-Link IE controller network and MELSECNET/H) of SD778 is 1, the CC-Link IE controller network and MELSECNET/H perform refresh. Therefore, if there are many refresh points, processing time for the COM instruction will be extended.	U	New	Qn(H)*13 QnPH*12 QnPRH

*9: Function version is B or later.
*11: The module whose first 5 digits of serial No. is "04012" or later.
*12: The module whose first 5 digits of serial No. is "07032" or later.
*13: The module whose first 5 digits of serial No. is "09012" or later.

TableApp.25 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD778	Refresh processing selection when the COM instruction is executed	b0 to b14: 0: Do not refresh 1: Refresh b15 bit 0: communication with peripheral device is executed 1: communication with peripheral device is nonexecuted	• Selects whether or not the data is refreshed when the COM instruction is executed. • Designation of SD778 is made valid when SM775 turns ON.  b15 b14 to b6 b5 b4 b3 b2 b1 b0 SD778 0/1 0 0/1 0/1 0/1 0/1 0/1 0/1 I/O refresh CC-Link refresh CC-Link IE controller network or MELSECNET/H refresh Automatic refresh of intelligent function modules Auto refresh using QCPU standard area of multiple CPU system and reading input/output from group outside. Auto refresh using the multiple CPU high speed transmission area of multiple CPU system Execution/non-execution of communication with CPU module	U	New	QnU
SD780	Remaining No. of simultaneous execution of CC-Link dedicated instruction	0 to 32	• Stores the remaining number of simultaneous execution of the CC-Link dedicated instructions.	U	New	QnA
SD781 to SD793	Mask pattern of IMASK instruction	Mask pattern	• Stores the mask patterns masked by the IMASK instruction as follows: b15 b1 b0 SD781 I63 to I49 I48 SD782 I79 to I65 I64 to SD793 I255 to I241 I240	S (During execution)	New	Qn(H) QnPH QnPRH QnU
SD781 to SD785	Mask pattern of IMASK instruction	Mask pattern	• Stores the mask patterns masked by the IMASK instruction as follows: b15 b1 b0 SD781 I63 to I49 I48 SD782 I79 to I65 I64 to to SD785 I127 to I113 I112	S (During execution)	New	Q00J/Q00/Q01
SD794 to SD795	PID limit setting (for incomplete derivative)	0: With limit 1: Without limit	• Specify the limit of each PID loop as shown below. b15 b1 b0 SD794 Loop16 to Loop2 Loop1 SD795 Loop32 to Loop18 Loop17	U	New	Qn(H)[*13] QnPRH QnU
SD794	PID limit setting (for incomplete derivative)	0: With limit 1: Without limit	• Specify the limit of each PID loop as shown below. b15 to b8 b7 b1 b0 SD794 Loop8 to Loop2 Loop1	U	New	Q00J/Q00/Q01[*9]

*9: Function version is B or later.

*13: The module whose first 5 digits of serial No. is "09012" or later.

TableApp.25 Special register

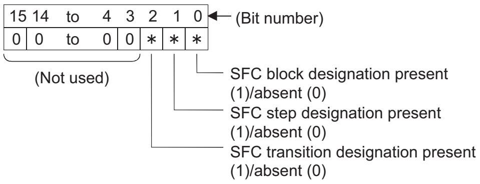
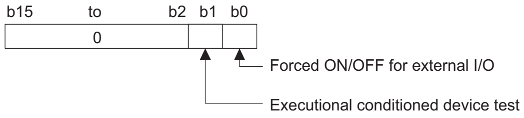
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD796	Maximum number of blocks used for dedicated instruction of Multiple CPU high speed transmission setting (for CPU No.1)	Maximum number of blocks used for dedicated instruction range 1 to 7 (Default: 2) * When value other than 1 to 7 is set, operation is performed as if 7 is set.	• Specifies the maximum number of blocks used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.1). When the dedicated instruction of Multiple CPU high speed transmission is executed to the CPU No.1, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM796 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instrucion of Multiple CPU high speed transmission.	U (At 1 scan after RUN)	New	QnU*14*15
SD797	Maximum number of blocks used for dedicated instruction of Multiple CPU high speed transmission setting (for CPU No.2)		• Specifies the maximum number of blocks used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.2). When the dedicated instruction of Multiple CPU high speed transmission is executed to the CPU No.2, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM797 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instrucion of Multiple CPU high speed transmission.	U (At 1 scan after RUN)	New	QnU*14*15
SD798	Maximum number of blocks used for dedicated instruction of Multiple CPU high speed transmission setting (for CPU No.3)		• Specifies the maximum number of blocks used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.3). When the dedicated instruction of Multiple CPU high speed transmission is executed to the CPU No.3, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM798 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instrucion of Multiple CPU high speed transmission.	U (At 1 scan after RUN)	New	QnU*14*15
SD799	Maximum number of blocks used for dedicated instruction of Multiple CPU high speed transmission setting (for CPU No.4)		• Specifies the maximum number of blocks used for the dedicated instruction of Multiple CPU high speed transmission (target CPU = CPU No.4). When the dedicated instruction of Multiple CPU high speed transmission is executed to the CPU No.4, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM799 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instrucion of Multiple CPU high speed transmission.	U (At 1 scan after RUN)	New	QnU*14*15

*14: The Universal model QCPU except the Q02UCPU.

*15: For the Q03UDCPU, Q04UDHCPU, Q06UDHCPU whose first 5 digits of the serial number are "10011" or earlier, the range is 1 to 9 (Default: 2) and when a value other than 1 to 9 is set, operation is performed as if 9 is set.

(7) Debug

TableApp.26 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD806	Status latch file name	Status latch file name	• Stores file name (with extension) from point in time when status latch was conducted as ASCII code. b15 to b8 / b7 to b0 SD806: 2nd character / 1st character SD807: 4th character / 3rd character SD808: 6th character / 5th character SD809: 8th character / 7th character SD810: 1st character of the extension / 2EH(.) SD811: 3rd character of the extension / 2nd character of the extension	S (During execution)	New	QnA
SD807						
SD808						
SD809						
SD810						
SD811						
SD812	Status latch step	Status latch step	• Stores step number from point in time when status latch was conducted. SD812: Pattern∗ SD813: Block No. SD814: Step No./transition condition No. SD815: Sequence step No. (L) SD816: Sequence step No. (H) *: Contents of pattern data  15 14 to 4 3 2 1 0 0 0 to 0 0 ∗ ∗ ∗ (Bit number) (Not used) SFC block designation present (1)/absent (0) SFC step designation present (1)/absent (0) SFC transition designation present (1)/absent (0)	S (During execution)	D9055 format change	
SD813						
SD814						
SD815						
SD816						
SD840	Debug function usage	Debug function usage	Stores the status of the debug function usage as shown below. 0 : Forced ON/OFF for external I/O 1 : Executional conditioned device test 2 to 15 :Absent (0 fix)  b15 to b2 b1 b0 0 Forced ON/OFF for external I/O Executional conditioned device test (0: Not used, 1: Used)	S (Status change)	New	QnU[*1]

*1: The module whose first 5 digits of serial No. is "10042" or later.

(8) Latch area

TableApp.27 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD900	Drive where power was interrupted	Access file drive number during power loss	• Stores drive number if file was being accessed during power loss.	S (Status change)	New	QnA
SD901	File name active during power loss	Access file name during power loss	• Stores file name (with extension) in ASCII code if file was being accessed during power loss. b15 to b8 b7 to b0 SD901 2nd character 1st character SD902 4th character 3rd character SD903 6th character 5th character SD904 8th character 7th character SD905 1st character of the extension 2EH(.) SD906 3rd character of the extension 2nd character of the extension	S (Status change)	New	
SD902						
SD903						
SD904						
SD905						
SD906						
SD910	RKEY input	RKEY input	• Stored in sequence that PU key code was entered. b15 to b8 b7 to b0 SD910 2nd character 1st character SD911 4th character 3rd character SD912 6th character 5th character SD913 8th character 7th character SD914 10th character 9th character SD915 12th character 11th character SD916 14th character 13th character SD917 16th character 15th character SD918 18th character 17th character SD919 20th character 19th character SD920 22nd character 21st character SD921 24th character 23rd character SD922 26th character 25th character SD923 28th character 27th character SD924 30th character 29th character SD925 32nd character 31st character	S (During execution)	New	
SD911						
SD912						
SD913						
SD914						
SD915						
SD916						
SD917						
SD918						
SD919						
SD920						
SD921						
SD922						
SD923						
SD924						
SD925						

(9) Redundant CPU information (host system CPU information*1)

TableApp.28 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD952	History of memory copy from control system to standby system	Latest status of memory copy from control system to standby system	Stores the completion status of the memory copy from control system to standby system executed last. 1) Stores the same value as stored into SD1596 at normal completion/ abnormal completion of the memory copy from control system to standby system. 2) Backed up for a power failure, this special register holds the status of memory copy from control system to standby system executed last. 3) Cleared to 0 by latch clear operation.	S (Status change)	New	QnPRH

*1: The host system CPU information is stored.

(10)Remote password count

TableApp.29 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD979	Direct MELSOFT connection	Count of unlock processing failures	Stores the count of unlock processing failures. Range: 0 to FFFEH (FFFFH when the limit is exceeded)	S (Status change)	New	QnU[*1]
SD980 to SD995	Connection 1 to 16					
SD997	MELSOFT connection using UDP port					
SD998	MELSOFT connection using TCP port					
SD999	FTP communication port					

*1: This applies to the QnUDE(H)CPU.

(11) A to Q/QnA conversion

ACPU special registers D9000 to D9255 correspond to Q/QnA special registers SD1000 to SD1255 after A to Q/QnA conversion.
(However, the Basic model QCPU, Redundant CPU, and Universal model QCPU do not support the A to Q/QnA conversion.)
These special registers are all set by the system, and cannot be set by the user program.
To set data by the user program, correct the program for use of the Q/QnACPU special registers.
However, some of SD1200 to SD1255 (corresponding to D9200 to 9255 before conversion) can be set by the user program if they could be set by the user program before conversion.
For details on the ACPU special registers, refer to the user's manual for the corresponding CPU, and MELSECNET or MELSECNET/B Data Link System Reference Manuals.

Remark

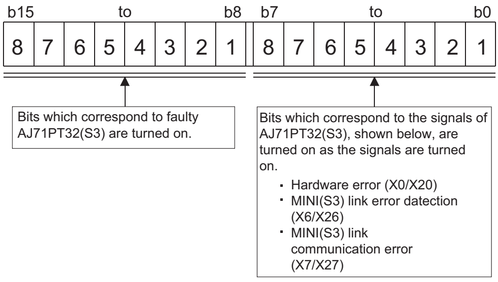
Supplemental explanation on "Special Register for Modification" column

① For the device numbers for which a special register for modification is specified, modify it to the special register for QCPU/QnACPU.

② For the device numbers for which ⊟ is specified, special register after conversion can be used.

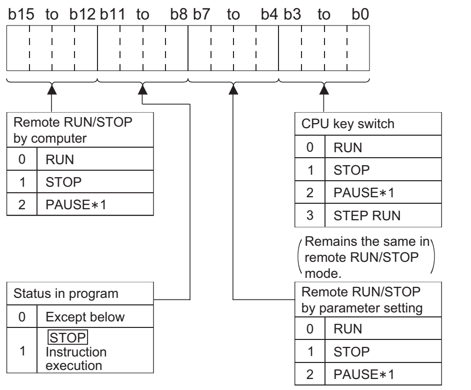
③ Device numbers for which ☒ is specified do not function for QCPU/QnACPU.

TableApp.28 Special register

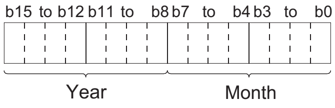
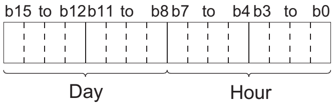
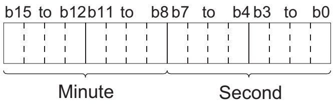
<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Corresponding CPU</th></tr>
<tr><td>D9000</td><td>SD1000</td><td>–</td><td>Fuse blown</td><td>Number of module with blown fuse</td><td>• When fuse blown modules are detected, the first I/O number of the lowest number of the detected modules is stored in hexadecimal. (Example: When fuses of Y50 to 6F output modules have blown, "50" is stored in hexadecimal)
To monitor the number by peripheral devices, perform monitor operation given in hexadecimal.
(Cleared when all contents of SD1100 to SD1107 are reset to 0.)
• Fuse blow check is executed also to the output modules of remote I/O stations.</td><td rowspan="2">QnA
Qn(H)
QnPH</td></tr>
<tr><td>D9001</td><td>SD1001</td><td>–</td><td>Fuse blown</td><td>Number of module with blown fuse</td><td>• Stores the module numbers corresponding to setting switch numbers or base slot numbers when fuse blow occurred.
<table>
<tr><th colspan="2">I/O module for A0J2</th><th colspan="2">Extension base unit</th></tr>
<tr><th>Setting switch</th><th>Stored data</th><th>Base unit slot No.</th><th>Stored data</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>4</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>5</td></tr>
<tr><td>2</td><td>2</td><td>2</td><td>6</td></tr>
<tr><td>3</td><td>3</td><td>3</td><td>7</td></tr>
<tr><td>4</td><td>4</td><td></td><td></td></tr>
<tr><td>5</td><td>5</td><td></td><td></td></tr>
<tr><td>6</td><td>6</td><td></td><td></td></tr>
<tr><td>7</td><td>7</td><td></td><td></td></tr>
</table>
• For the remote I/O station, the value of (module I/O No./10H) + 1 is stored.</td></tr>
<tr><td>D9002</td><td>SD1002</td><td>–</td><td>I/O module verify error</td><td>I/O module verify error module number</td><td>• If I/O modules, of which data are different from data entered, are detected when the power is turned on, the first I/O number of the lowest number unit among the detected units is stored in hexadecimal. (Storing method is the same as that of SD1000.)
To monitor the number by peripheral devices, perform monitor operation given in hexadecimal.
(Cleared when all contents of SD1116 to SD1123 are reset to 0.)
• I/O module verify check is executed also to the modules of remote I/O terminals.</td><td>QnA
Qn(H)
QnPH</td></tr>
<tr><td>D9004</td><td>SD1004</td><td>–</td><td>MINI link master module errors</td><td>Stores setting status made at parameters</td><td>• Error status of the MINI(S3) link detected on loaded AJ71PT32(S3) is stored.


Bits which correspond to faulty AJ71PT32(S3) are turned on.
Bits which correspond to the signals of AJ71PT32(S3), shown below, are turned on as the signals are turned on.
• Hardware error (X0/X20)
• MINI(S3) link error datection (X6/X26)
• MINI(S3) link communication error (X7/X27)</td><td>QnA</td></tr>
<tr><td rowspan="3">D9005</td><td rowspan="3">SD1005</td><td rowspan="3">–</td><td rowspan="3">AC DOWN counter</td><td rowspan="3">Number of times for AC DOWN</td><td>• When the AC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 20ms. (The value is stored in BIN code.) It is reset when the power supply is switched from OFF to ON.</td><td>QnA
Qn(H)
QnPH</td></tr>
<tr><td>• When the DC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 10ms. (The value is stored in BIN code.) It is reset when the power supply is switched from OFF to ON.</td><td>Qn(H)
QnPH</td></tr>
<tr><td>• When the DC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 1ms. (The value is stored in BIN code.) It is reset when the power supply is switched from OFF to ON.</td><td>QnA</td></tr>
<tr><td>D9008</td><td>SD1008</td><td>SD0</td><td>Self-diagnostic error</td><td>Self-diagnostic error number</td><td>• When error is found as a result of self-diagnosis, error number is stored in BIN code.</td><td>QnA
Qn(H)
QnPH</td></tr>
</table>

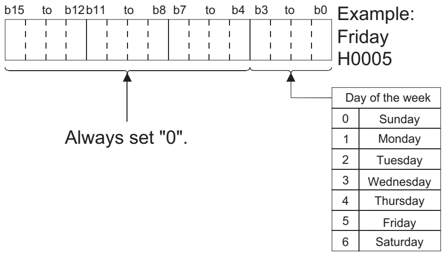
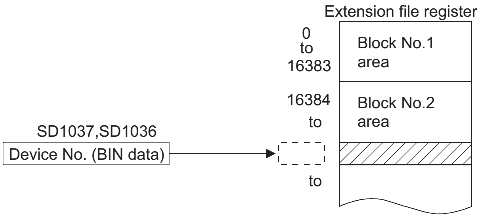
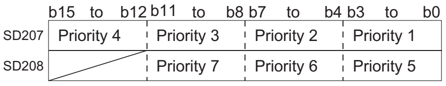
TableApp.28 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9009	SD1009	SD62	Annunciator detection	F number at which external failure has occurred	• When one of F0 to 2047 is turned on by OUT F or SET F , the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code. • SD62 can be cleared by RST F or LEDR instruction. If another F number has been detected, the clearing of SD62 causes the next number to be stored in SD62.	Qn(H) QnPH Q2AS Q2A
					• When one of F0 to 2047 is turned on by OUT F or SET F , the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code. • SD62 can be cleared by executing RST F or LEDR instruction or moving INDICATOR RESET switch on CPU module front to ON position. If another F number has been detected, clearing of SD62 stores the next F number into SD62.	Q3A Q4A Q4AR
D9010	SD1010	×	Error step	Step number at which operation error has occurred.	• When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Thereafter, each time operation error occurs, the contents of SD1010 are renewed.	Qn(H) QnPH
D9011	SD1011	×	Error step	Step number at which operation error has occurred.	• When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Since the step number is stored into SD1011 when SM1011 turns from OFF to ON, the data of SD1011 is not updated unless SM1011 is cleared by a user program.	
D9014	SD1014	×	I/O control mode	I/O control mode number	• The I/O control mode set is returned in any of the following numbers: 0: Both input and output in direct mode 1: Input in refresh mode, output in direct mode 3: Both input and output in refresh mode	
D9015	SD1015	SD203	Operating status of CPU	Operating status of CPU	• The operation status of CPU as shown below are stored in SD203.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 Remote RUN/STOP by computer 0 RUN 1 STOP 2 PAUSE∗1 CPU key switch 0 RUN 1 STOP 2 PAUSE∗1 3 STEP RUN (Remains the same in remote RUN/STOP mode.) Status in program 0 Except below 1 STOP Instruction execution Remote RUN/STOP by parameter setting 0 RUN 1 STOP 2 PAUSE∗1 *1: When the CPU mdoule is in RUN mode and SM1040 is off, the CPU module remains in RUN mode if changed to PAUSE mode.	QnA Qn(H) QnPH

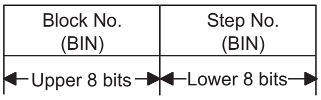
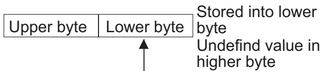
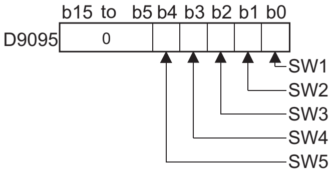
TableApp.28 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9016	SD1016	×	Program number	0: Main program (ROM) 1: Main program (RAM) 2: Subprogram 1 (RAM) 3: Subprogram 2 (RAM) 4: Subprogram 3 (RAM) 5: Subprogram 1 (ROM) 6: Subprogram 2 (ROM) 7: Subprogram 3 (ROM) 8: Main program (E^2PROM) 9: Subprogram 1 (E^2PROM) A: Subprogram 2 (E^2PROM) B: Subprogram 3 (E^2PROM)	• Indicates which sequence program is run presently. One value of 0 to B is stored in BIN code.	QnA Qn(H) QnPH
D9017	SD1017	SD520	Scan time	Minimum scan time (10 ms units)	• If scan time is smaller than the content of SD520, the value is newly stored at each END. Namely, the minimum value of scan time is stored into SD520 in BIN code.	
D9018	SD1018	SD524	Scan time	Scan time (10 ms units)	• At every END, the scan time is stored in BIN code and always rewritten.	
D9019	SD1019	SD526	Scan time	Maximum scan time (10 ms units)	• If scan time is larger than the content of SD526, the value is newly stored at each END. Namely, the maximum value of scan time is stored into SD526 in BIN code.	
D9020	SD1020	×	Constant scan	Constant scan time (User sets in 10 ms units)	• Sets the interval between consecutive program starts in multiples of 10 ms. 0 : No setting 1 to 200 : Set. Program is executed at intervals of (set value) × 10 ms.	
D9021	SD1021	–	Scan time	Scan time (1 ms units)	• At every END, the scan time is stored in BIN code and always rewritten.	
D9022	SD1022	SD412	1 second counter	Count in units of 1s.	• When the PC CPU starts running, it starts counting 1 every second. • It starts counting up from 0 to 32767, then down to -32768 and then again up to 0. Counting repeats this routine.	
D9025	SD1025	–	Clock data	Clock data (year, month)	• The year (last two digits) and month are stored as BCD code as shown below.  Example: 1987, July H8707	
D9026	SD1026	–	Clock data	Clock data (day, hour)	• The day and hour are stored as BCD code as shown below.  Example: 31st, 10 a.m. H3110	
D9027	SD1027	–	Clock data	Clock data (minute, second)	• The minute and second are stored as BCD code as shown below.  Example: 35 min, 48 sec. H3548	

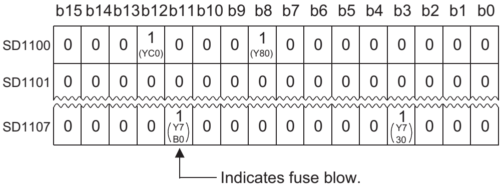
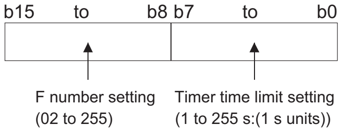
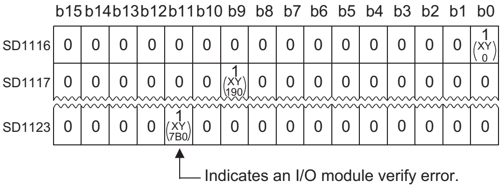
TableApp.28 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9028	SD1028	–	Clock data	Clock data (day of week)	• The day of the week is stored as BCD code as shown below. b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example: Friday H0005 Always set "0". Day of the week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday	QnA Qn(H) QnPH
D9035	SD1035	SD648	Extension file register	Use block No.	• Stores the block No. of the extension file register being used in BCD code.	
D9036	SD1036	×	Extension file registerfor designation of device number	Device number when individual devices from extension file register are directly accessed	• Designate the device number for the extension file register for direct read and write in 2 words at SD1036 and SD1037 in BIN data. Use consecutive numbers beginning with R0 of block No. 1 to designate device numbers. Extension file register 0 to 16383 Block No.1 area 16384 to Block No.2 area SD1037,SD1036 Device No. (BIN data) to	
D9037	SD1037	×				
D9038	SD1038	SD207	LED display priority ranking	Priorities 1 to 4	• Sets priority of ERROR LEDs which illuminate (or flicker) to indicate errors with error code numbers. • Configuration of the priority setting areas is as shown below. b15 to b12 b11 to b8 b7 to b4 b3 to b0 SD207 Priority 4 Priority 3 Priority 2 Priority 1 SD208 Priority 7 Priority 6 Priority 5 • For details, refer to the applicable CPUs User's Manual and the ACPU Programming manual (Fundamentals).	
D9039	SD1039	SD208		Priorities 5 to 7		
D9044	SD1044	×	For sampling trace	Step or time during sampling trace	• Turned on/off with a peripheral device. When STRA or STRAR is executed, the value stored in SD1044 is used as the sampling trace condition. At scanning--------0 At time--------------Time (10 msec unit) The value is stored into SD1044 in BIN code.	
D9049	SD1049	×	Work area for SFC	Block number of extension file register	• Stores the block number of the expansion file register which is used as the work area for the execution of a SFC program in a binary value. • Stores "0" if an empty area of 16K bytes or smaller, which cannot be expansion file register No. 1, is used or if SM320 is OFF.	
D9050	SD1050	×	SFC program error number	Error code generated by SFC program	• Stores error code of errors occurred in the SFC program in BIN code. 0 : No error 80: SFC program parameter error 81: SFC code error 82: Number of steps of simultaneous execution exceeded 83: Block start error 84: SFC program operation error	
D9051	SD1051	×	Error block	Block number where error occurred	• Stores the block number in which an error occurred in the SFC program in BIN code. In the case of error 83 the starting block number is stored.	
D9052	SD1052	×	Error step	Step number where error occurred	• Stores the step number, where error code 84 occurred in an SFC program, in BIN value. • Stores "0" when error code 80, 81 or 82 occurred. • Stores the block stating step number when error code 83 occurs.	

TableApp.28 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9053	SD1053	×	Error transition	Transition condition number where error occurred	• Stores the transition condition number, where error code 84 occurred in an SFC program, in BIN value. Stores "0" when error code 80, 81, 82 or 83 occurred.	QnA Qn(H) QnPH
D9054	SD1054	×	Error sequence step	Sequence step number where error occurred	• Stores the sequence step number of transfer condition and operation output in which error 84 occurred in the SFC program in BIN code.	
D9055	SD1055	SD812	Status latch execution step number	Status latch step	• Stores the step number when status latch is executed. • Stores the step number in a binary value if status latch is executed in a main sequence program. • Stores the block number and the step number if status latch is executed in a SFC program.  Block No. (BIN) Step No. (BIN) Upper 8 bits Lower 8 bits	
D9060	SD1060	SD392	Software version	Software version of internal software	• Stores the software version of the internal system in ASCII code.  Upper byte Lower byte Stored into lower byte Undefind value in higher byte For version "A", for example, "41H" is stored. Note: The software version of the initial system may differ from the version indicated by the version information printed on the rear of the case.	QnA
D9072	SD1072	×	PLC communication check	Data check of serial communication module	• In the self-loopback test of the serial communication module, the serial communication module writes/reads data automatically to make communication checks.	QnA Qn(H) QnPH
D9081	SD1081	SD714	Number of empty blocks in communications request registrtion area	Number of empty blocks in communications request registration area	• Stores the number of empty blocks in the communication request registration area to the remote terminal module connected to the MELSECNET/MINI-S3 master unit, A2CCPU or A52GCPU.	QnA
D9085	SD1085	×	Register for setting time check value	1 s to 65535 s	• Sets the time check time of the data link instructions (ZNRD, ZNWR) for the MELSECNET/10. • Setting range : 1 s to 65535 s (1 to 65535) • Setting unit : 1 s • Default value : 10 s (If 0 has been set, default 10 s is applied)	QnA Qn(H) QnPH
D9090	SD1090	×	Number of special functions modules over	Number of special functions modules over	• For details, refer to the manual of each microcomputer program package.	
D9091	SD1091	×	Detailed error code	Self-diagnosis detailed error code	• Stores the detail code of cause of an instruction error.	
D9094	SD1094	SD251	Head I/O number of I/O module to be replaced	Head I/O number of I/O module to be replaced	• Stores the first two digits of the head I/O number of the I/O module, which will be dismounted/mounted online (with power on), in BIN value. Example) Input module X2F0 → H2F	
D9095	SD1095	SD200	DIP switch information	DIP switch information	• The DIP switch information of the CPU module is stored in the following format. 0: OFF 1: ON  b15 to b5 b4 b3 b2 b1 b0 D9095 0 SW1 SW2 SW3 SW4 SW5	Qn(H) QnPH

TableApp.28 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9100	SD1100	–	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown	• Output module numbers (in units of 16 points), of which fuses have blown, are entered in bit pattern. (Preset output module numbers when parameter setting has been performed.)  b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1100 0 0 0 1 (YC0) 0 0 0 1 (Y80) 0 0 0 0 0 0 0 0 SD1101 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 SD1107 0 0 0 0 1 (Y7B0) 0 0 0 0 0 0 0 1 (Y730) 0 0 0 Indicates fuse blow. • Fuse blow check is executed also to the output module of remote I/O station. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)	QnA Qn(H) QnPH
D9101	SD1101					
D9102	SD1102					
D9103	SD1103					
D9104	SD1104					
D9105	SD1105					
D9106	SD1106					
D9107	SD1107					
D9108	SD1108	–	Step transfer monitoring timer setting	Timer setting valve and the F number at time out	• Set the value of the step transition monitoring timer and the annunciator number (F number) that will be turned ON when the monitoring timer times out.  b15 to b8 b7 to b0 F number setting (02 to 255) Timer time limit setting (1 to 255 s:(1 s units)) • By turning ON any of SM1108 to SM1114, the monitoring timer starts. If the transition condition following a step which corresponds to the timer is not established within set time, set annunciator (F) is turned on.)	
D9109	SD1109					
D9110	SD1110					
D9111	SD1111					
D9112	SD1112					
D9113	SD1113					
D9114	SD1114					
D9116	SD1116	–	I/O module verification error	Bit pattern, in units of 16 points, indicating the modules with verification errors.	• When I/O modules, of which data are different from those entered at power-ON, have been detected, the I/O module numbers (in units of 16 points) are entered in bit pattern. (Preset I/O module numbers set in parmeters when parameter setting has been performed.)  b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1116 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 (XY0) SD1117 0 0 0 0 0 0 1 (XY190) 0 0 0 0 0 0 0 0 0 SD1123 0 0 0 0 1 (XY7B0) 0 0 0 0 0 0 0 0 0 0 0 Indicates an I/O module verify error. • I/O module verify check is executed also to remote I/O station modules. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)	
D9117	SD1117					
D9118	SD1118					
D9119	SD1119					
D9120	SD1120					
D9121	SD1121					
D9122	SD1122					
D9123	SD1123					
D9124	SD1124	SD63	Number of annunciator detections	Number of annunciator detections	• When one of F0 to 255 (F0 to 2047 for AuA and AnU) is turned on by SET F 1 is added to the contents of SD63. When RST F or LEDR instruction is executed, 1 is subtracted from the contents of SD63. (If the INDICATOR RESET switch is provided to the CPU module, pressing the switch can execute the same processing.) • Quantity, which has been turned on by SET F is stored into SD63 in BIN code. The value of SD63 is maximum 8.	

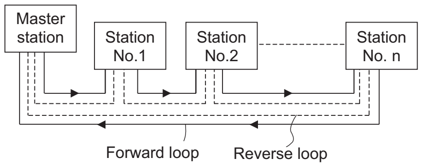
TableApp.28 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9125	SD1125	SD64	Annunciator detection number	Annunciator detection number	• When any of F0 to 2047 is turned on by SET F , the annunciator numbers (F numbers) that are turned on in order are registered into D9125 to D9132. • The F number turned off by RST F is erased from any of D9125 to D9132, and the F numbers stored after the erased F number are shifted to the preceding registerers. By executing LEDR instruction, the contents of SD64 to SD71 are shifted upward by one. (For A3N, A3HCPU, it can be performed by use of INDICATOR RESET switch on front of CPU module.) When there are 8 annunciator detections, the 9th one is not stored into SD64 to SD71 even if detected.	QnA Qn(H) QnPH
D9126	SD1126	SD65				
D9127	SD1127	SD66				
D9128	SD1128	SD67				
D9129	SD1129	SD68				
D9130	SD1130	SD69				
D9131	SD1131	SD70				
D9132	SD1132	SD71				

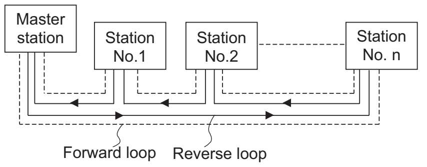
		SET F50	SET F25	SET F99	RST F25	SET F15	SET F70	SET F65	SET F38	SET F110	SET F151	SET F210	LEDR
SD62	0	50	50	50	50	50	50	50	50	50	50	50	99
SD63	0	1	2	3	2	3	4	5	6	7	8	8	8
SD64	0	50	50	50	50	50	50	50	50	50	50	50	99
SD65	0	0	25	25	99	99	99	99	99	99	99	99	15
SD66	0	0	0	99	0	15	15	15	15	15	15	15	70
SD67	0	0	0	0	0	0	70	70	70	70	70	70	65
SD68	0	0	0	0	0	0	0	65	65	65	65	65	38
SD69	0	0	0	0	0	0	0	0	38	38	38	38	110
SD70	0	0	0	0	0	0	0	0	0	110	110	110	151
SD71	0	0	0	0	0	0	0	0	0	0	151	151	210

(12)Special register list dedicated for QnA

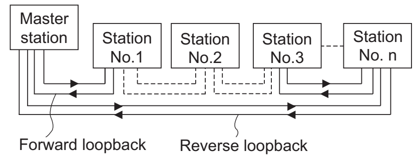
TableApp.30 Special register

<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Corresponding CPU</th></tr>
<tr><td>D9200</td><td>SD1200</td><td>–</td><td>ZNRD instruction processing result (LRDP for ACPU)</td><td>0 : Normal end
2 : ZNRD instruction setting fault
3 : Error at relevant station
4 : Relevant station ZNRD execution disabled</td><td>Stores the execution result of the ZNRD (word device read) instruction
• ZNRD instruction setting fault.... Faulty setting of the ZNRD instruction constant, source, and/ or destination.
• Corresponding station error....... One of the stations is not communicating.
• ZNRD cannot be executed in the corresponding station ..... The specified station is a remote I/ O station.</td><td rowspan="5">QnA</td></tr>
<tr><td>D9201</td><td>SD1201</td><td>–</td><td>ZNWR instruction processing result (LWTP for ACPU)</td><td>0 : Normal end
2 : ZNWR instruction setting fault
3 : Error at relevant station
4 : Relevant station ZNWR execution disabled</td><td>Stores the execution result of the ZNWR (word device write) instruction.
• ZNWR instruction setting fault ... Faulty setting of the ZNWR instruction constant, source, and/ or destination.
• Corresponding station error....... One of the stations is not communicating.
• ZNWR cannot be executed in the corresponding station ..... The specified station is a remote I/ O station.</td></tr>
<tr><td>D9202</td><td>SD1202</td><td>–</td><td rowspan="2">Local station link type</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="2">Stores whether the slave station corresponds to MELSECNET or MELSECNET II.
• Bits corresponding to the MELSECNET II stations become "1."
• Bits corresponding to the MELSECNET stations or unconnected become "0."
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1202</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr>
<tr><td>SD1203</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr>
<tr><td>SD1241</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr>
<tr><td>SD1242</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr>
</table>
• If a local station goes down during the operation, the contents before going down are retained.
Contents of SD1224 to SD1227 and SD1228 to SD1231 are ORed.
If the corresponding bit is "0", the corresponding bit of the special register above becomes valid.
• If the host (master) station goes down, the contents before going down are also retained.</td></tr>
<tr><td>D9203</td><td>SD1203</td><td>–</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9204</td><td>SD1204</td><td>–</td><td>Link status</td><td>0 : Forward loop, during data link
1 : Reverse loop, during data link
2 : Loopback implemented in forward/reverse directions
3 : Loopback implemented only in forward direction
4 : Loopback implemented only in reverse direction
5 : Data link disabled</td><td>Stores the present path status of the data link.
• Data link in forward loop


Master station
Station No.1
Station No.2
Station No. n
Forward loop
Reverse loop

• Data link in reverse loop


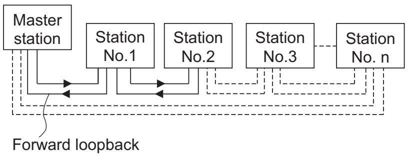
Master station
Station No.1
Station No.2
Station No. n
Forward loop
Reverse loop

• Loopback implemented in forward/reverse directions


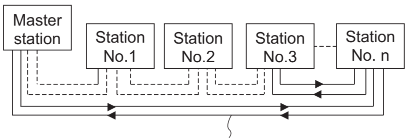
Master station
Station No.1
Station No.2
Station No.3
Station No. n
Forward loopback
Reverse loopback

</td></tr>
</table>

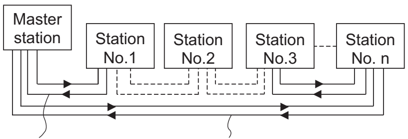
TableApp.30 Special register

<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Corresponding CPU</th></tr>
<tr><td>D9204</td><td>SD1204</td><td>–</td><td>Link status</td><td>0 : Forward loop, during data link
1 : Reverse loop, during data link
2 : Loopback implemented in forward/reverse directions
3 : Loopback implemented only in forward direction
4 : Loopback implemented only in reverse direction
5 : Data link disabled</td><td>• Loopback in forward loop only


Master station
Station No.1
Station No.2
Station No.3
Station No. n

Forward loopback
• Loopback in reverse loop only


Master station
Station No.1
Station No.2
Station No.3
Station No. n

Reverse loopback</td><td rowspan="13">QnA</td></tr>
<tr><td>D9205</td><td>SD1205</td><td>–</td><td>Station implementing loopback</td><td>Station that implemented forward loopback</td><td rowspan="2">Stores the local or remote I/O station number at which loopback is being executed.


Master station
Station No.1
Station No.2
Station No.3
Station No. n

Forward loopback Reverse loopback
In the above example, 1 is stored into D9205 and 3 into D9206.
If data link returns to normal status (data link in forward loop), values in D9205 and D9206 remain 1 and 3. To return them to "0", therefore, use a sequence program or perform reset operation.</td></tr>
<tr><td>D9206</td><td>SD1206</td><td>–</td><td>Station implementing loopback</td><td>Station that implemented reverse loopback</td></tr>
<tr><td>D9210</td><td>SD1210</td><td>–</td><td>Number of retries</td><td>Stored as cumulative value</td><td>Stores the number of retry times due to transmission error.
Count stops at maximum of "FFFF$_{H}$" .
To return the value to "0", perform reset operation.</td></tr>
<tr><td>D9211</td><td>SD1211</td><td>–</td><td>Number of times loop selected</td><td>Stored as cumulative value</td><td>Stores the number of times the loop line has been switched to reverse loop or loopback.
Count stops at maximum of "FFFF$_{H}$".
To return the value to "0", perform reset operation.</td></tr>
<tr><td>D9212</td><td>SD1212</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local station numbers which are in STOP or PAUSE mode.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1212</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr>
<tr><td>SD1213</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr>
<tr><td>SD1214</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr>
<tr><td>SD1215</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr>
</table>
When a local station is switched to STOP or PAUSE mode, the bit corresponding to the station number in the register becomes "1".
Example: When station 7 switches to STOP mode, b6 in SD1212 becomes "1" , and when SD1212 is monitored, its value is "64 (40$_{H}$)".</td></tr>
<tr><td>D9213</td><td>SD1213</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9214</td><td>SD1214</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9215</td><td>SD1215</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
<tr><td>D9216</td><td>SD1216</td><td>–</td><td>Local station error detect status</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local station numbers which are in error.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1216</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr>
<tr><td>SD1217</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr>
<tr><td>SD1218</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr>
<tr><td>SD1219</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr>
</table>
If a local station detects an error, the bit corresponding to the station number becomes "1".
Example: When station 6 and 12 detect an error, b5 and 11 in SD1216 become "1", and when SD1216 is monitored, its value is "2080 (820$_{H}$)".</td></tr>
<tr><td>D9217</td><td>SD1217</td><td>–</td><td>Local station error detect status</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9218</td><td>SD1218</td><td>–</td><td>Local station error detect status</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9219</td><td>SD1219</td><td>–</td><td>Local station error detect status</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
</table>

TableApp.30 Special register

<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Corresponding CPU</th></tr>
<tr><td>D9220</td><td>SD1220</td><td>–</td><td>Local station parameters non-conforming; remote I/O station I/O assignment error</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local station numbers which contain mismatched parameters or of remote station numbers for which incorrect I/O assignment has been made.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1220</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr>
<tr><td>SD1221</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr>
<tr><td>SD1222</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr>
<tr><td>SD1223</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr>
</table>
If a local station acting as the master station of tier three detects a parameter error or a remote I/O station whose I/O assignment is abnormal, the bit of the device number corresponding to the station number of that local station or remote I/O station turns to "1".
Example: When local station 5 and remote I/O station 14 detect an error, b4 and b13 in SD1220 become "1" , and when SD1220 is monitored, its value is "8208 (2010H) " .</td><td rowspan="12">QnA</td></tr>
<tr><td>D9221</td><td>SD1221</td><td>–</td><td>Local station parameters non-conforming; remote I/O station I/O assignment error</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9222</td><td>SD1222</td><td>–</td><td>Local station parameters non-conforming; remote I/O station I/O assignment error</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9223</td><td>SD1223</td><td>–</td><td>Local station parameters non-conforming; remote I/O station I/O assignment error</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
<tr><td>D9224</td><td>SD1224</td><td>–</td><td>Local station and remote I/O station initial communications underway</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local or remote station numbers while they are communicating the initial data with their relevant master station.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1224</td><td>L/R 16</td><td>L/R 15</td><td>L/R 14</td><td>L/R 13</td><td>L/R 12</td><td>L/R 11</td><td>L/R 10</td><td>L/R 9</td><td>L/R 8</td><td>L/R 7</td><td>L/R 6</td><td>L/R 5</td><td>L/R 4</td><td>L/R 3</td><td>L/R 2</td><td>L/R 1</td></tr>
<tr><td>SD1225</td><td>L/R 32</td><td>L/R 31</td><td>L/R 30</td><td>L/R 29</td><td>L/R 28</td><td>L/R 27</td><td>L/R 26</td><td>L/R 25</td><td>L/R 24</td><td>L/R 23</td><td>L/R 22</td><td>L/R 21</td><td>L/R 20</td><td>L/R 19</td><td>L/R 18</td><td>L/R 17</td></tr>
<tr><td>SD1226</td><td>L/R 48</td><td>L/R 47</td><td>L/R 46</td><td>L/R 45</td><td>L/R 44</td><td>L/R 43</td><td>L/R 42</td><td>L/R 41</td><td>L/R 40</td><td>L/R 39</td><td>L/R 38</td><td>L/R 37</td><td>L/R 36</td><td>L/R 35</td><td>L/R 34</td><td>L/R 33</td></tr>
<tr><td>SD1227</td><td>L/R 64</td><td>L/R 63</td><td>L/R 62</td><td>L/R 61</td><td>L/R 60</td><td>L/R 59</td><td>L/R 58</td><td>L/R 57</td><td>L/R 56</td><td>L/R 55</td><td>L/R 54</td><td>L/R 53</td><td>L/R 52</td><td>L/R 51</td><td>L/R 50</td><td>L/R 49</td></tr>
</table>
The bit corresponding to the station number which is currently communicating the initial settings becomes "1" .
Example: When stations 23 and 45 are communicating, b6 of SD1225 and b12 of SD1226 become "1", and when SD1225 is monitored, its value is "64 (40H)", and when SD1226 is monitored, its value is "4096 (1000H) " .</td></tr>
<tr><td>D9225</td><td>SD1225</td><td>–</td><td>Local station and remote I/O station initial communications underway</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9226</td><td>SD1226</td><td>–</td><td>Local station and remote I/O station initial communications underway</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9227</td><td>SD1227</td><td>–</td><td>Local station and remote I/O station initial communications underway</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
<tr><td>D9228</td><td>SD1228</td><td>–</td><td>Local station and remote I/O station error</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local or remote station numbers which are in error.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1228</td><td>L/R 16</td><td>L/R 15</td><td>L/R 14</td><td>L/R 13</td><td>L/R 12</td><td>L/R 11</td><td>L/R 10</td><td>L/R 9</td><td>L/R 8</td><td>L/R 7</td><td>L/R 6</td><td>L/R 5</td><td>L/R 4</td><td>L/R 3</td><td>L/R 2</td><td>L/R 1</td></tr>
<tr><td>SD1229</td><td>L/R 32</td><td>L/R 31</td><td>L/R 30</td><td>L/R 29</td><td>L/R 28</td><td>L/R 27</td><td>L/R 26</td><td>L/R 25</td><td>L/R 24</td><td>L/R 23</td><td>L/R 22</td><td>L/R 21</td><td>L/R 20</td><td>L/R 19</td><td>L/R 18</td><td>L/R 17</td></tr>
<tr><td>SD1230</td><td>L/R 48</td><td>L/R 47</td><td>L/R 46</td><td>L/R 45</td><td>L/R 44</td><td>L/R 43</td><td>L/R 42</td><td>L/R 41</td><td>L/R 40</td><td>L/R 39</td><td>L/R 38</td><td>L/R 37</td><td>L/R 36</td><td>L/R 35</td><td>L/R 34</td><td>L/R 33</td></tr>
<tr><td>SD1231</td><td>L/R 64</td><td>L/R 63</td><td>L/R 62</td><td>L/R 61</td><td>L/R 60</td><td>L/R 59</td><td>L/R 58</td><td>L/R 57</td><td>L/R 56</td><td>L/R 55</td><td>L/R 54</td><td>L/R 53</td><td>L/R 52</td><td>L/R 51</td><td>L/R 50</td><td>L/R 49</td></tr>
</table>
The bit corresponding to the station number with the error becomes "1" .
Example: When local station 3 and remote I/O station 14 have an error, b2 and b13 of SD1228 become "1", and when SD1228 is monitored, its value is "8196 (2004H)".</td></tr>
<tr><td>D9229</td><td>SD1229</td><td>–</td><td>Local station and remote I/O station error</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9230</td><td>SD1230</td><td>–</td><td>Local station and remote I/O station error</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9231</td><td>SD1231</td><td>–</td><td>Local station and remote I/O station error</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
</table>

TableApp.30 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9232	SD1232	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 1 to 8	Stores the local or remote station number at which a forward or reverse loop error has occurred "F" in the above table indicates a forward loop line, and "R" a reverse loop line. The bit of the device number corresponding to the station number of the local station or remote I/O station that has a forward loop line or reverse loop line error. Example: When the forward loop line of station 5 has an error, b8 of SD1232 become "1", and when SD1232 is monitored, its value is "256 (100H)".	QnA
D9233	SD1233	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 9 to 16		
D9234	SD1234	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 17 to 24		
D9235	SD1235	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 25 to 32		
D9236	SD1236	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 33 to 40		
D9237	SD1237	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 41 to 48		
D9238	SD1238	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 49 to 56		
D9239	SD1239	–	Local station and remote I/O station loop error	Stores conditions for up to numbers 57 to 64		
D9240	SD1240	–	Number of times communications errors detected	Stores cumulative total of receive errors	Stores the number of times the following transmission errors have been detected: CRC, OVER, AB. IF Count is made to a maximum of FFFFH. To return the value to "0", perform reset operation.	
D9241	SD1241	–	Local station link type	Stores conditions for up to numbers 33 to 48	Stores whether the slave station corresponds to MELSECNET or MELSECNET II. • Bits corresponding to the MELSECNET II stations become "1." • Bits corresponding to the MELSECNET stations or unconnected become "0." • If a local station goes down during the operation, the contents before going down are retained. Contents of SD1224 to SD1227 and SD1228 to SD1231 are ORed. If the corresponding bit is "0", the corresponding bit of the special register above becomes valid. • If the host (master) station goes down, the contents before going down are also retained.	
D9242	SD1242			Stores conditions for up to numbers 49 to 64		
D9243	SD1243	–	Station number information for host station	Stores station number (0 to 64)	Allows a local station to confirm its own station number	
D9244	SD1244	–	Number of link device stations	Stores number of slave stations	Indicates the number of slave stations in one loop.	
D9245	SD1245	–	Receive error detection count	Stores cumulative total of receive errors	Stores the number of times the following transmission errors have been detected: CRC, OVER, AB. IF Count is made to a maximum of FFFFH. To return the value to "0", perform reset operation.	

Device number	Bit															
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SD1232	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R8		L/R7		L/R6		L/R5		L/R4		L/R3		L/R2		L/R1	
SD1233	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R16		L/R15		L/R14		L/R13		L/R12		L/R11		L/R10		L/R9	
SD1234	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R24		L/R23		L/R22		L/R21		L/R20		L/R19		L/R18		L/R17	
SD1235	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R32		L/R31		L/R30		L/R29		L/R28		L/R27		L/R26		L/R25	
SD1236	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R40		L/R39		L/R38		L/R37		L/R36		L/R35		L/R34		L/R33	
SD1237	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R48		L/R47		L/R46		L/R45		L/R44		L/R43		L/R42		L/R41	
SD1238	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R56		L/R55		L/R54		L/R53		L/R52		L/R51		L/R50		L/R49	
SD1239	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F
	L/R64		L/R63		L/R62		L/R61		L/R60		L/R59		L/R58		L/R57	

Device number	Bit															
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SD1202	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1
SD1203	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17
SD1241	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33
SD1242	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49

TableApp.30 Special register

<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Details</th><th>Corresponding CPU</th></tr>
<tr><td>D9248</td><td>SD1248</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local station number which is in STOP or PAUSE mode.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1248</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr>
<tr><td>SD1249</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr>
<tr><td>SD1250</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr>
<tr><td>SD1251</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr>
</table>
The bit corresponding to the station number which is in STOP or PAUSE mode, becomes "1".
Example: When local stations 7 and 15 are in STOP mode, b6 and b14 of SD1248 become "1", and when SD1248 is monitored, its value is "16448 (4040H)".</td><td rowspan="8">QnA</td></tr>
<tr><td>D9249</td><td>SD1249</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9250</td><td>SD1250</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9251</td><td>SD1251</td><td>–</td><td>Local station operation status</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
<tr><td>D9252</td><td>SD1252</td><td>–</td><td>Local station error conditions</td><td>Stores conditions for up to numbers 1 to 16</td><td rowspan="4">Stores the local station number other than the host, which is in error.
<table>
<tr><th rowspan="2">Device number</th><th colspan="16">Bit</th></tr>
<tr><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1252</td><td>L16</td><td>L15</td><td>L14</td><td>L13</td><td>L12</td><td>L11</td><td>L10</td><td>L9</td><td>L8</td><td>L7</td><td>L6</td><td>L5</td><td>L4</td><td>L3</td><td>L2</td><td>L1</td></tr>
<tr><td>SD1253</td><td>L32</td><td>L31</td><td>L30</td><td>L29</td><td>L28</td><td>L27</td><td>L26</td><td>L25</td><td>L24</td><td>L23</td><td>L22</td><td>L21</td><td>L20</td><td>L19</td><td>L18</td><td>L17</td></tr>
<tr><td>SD1254</td><td>L48</td><td>L47</td><td>L46</td><td>L45</td><td>L44</td><td>L43</td><td>L42</td><td>L41</td><td>L40</td><td>L39</td><td>L38</td><td>L37</td><td>L36</td><td>L35</td><td>L34</td><td>L33</td></tr>
<tr><td>SD1255</td><td>L64</td><td>L63</td><td>L62</td><td>L61</td><td>L60</td><td>L59</td><td>L58</td><td>L57</td><td>L56</td><td>L55</td><td>L54</td><td>L53</td><td>L52</td><td>L51</td><td>L50</td><td>L49</td></tr>
</table>
The bit corresponding to the station number which is in error, becomes "1".
Example: When local station 12 is in error, b11 of SD1252 becomes "1", and when SD1252 is monitored, its value is "2048 (800H) ".</td></tr>
<tr><td>D9253</td><td>SD1253</td><td>–</td><td>Local station error conditions</td><td>Stores conditions for up to numbers 17 to 32</td></tr>
<tr><td>D9254</td><td>SD1254</td><td>–</td><td>Local station error conditions</td><td>Stores conditions for up to numbers 33 to 48</td></tr>
<tr><td>D9255</td><td>SD1255</td><td>–</td><td>Local station error conditions</td><td>Stores conditions for up to numbers 49 to 64</td></tr>
</table>

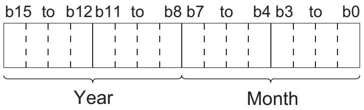
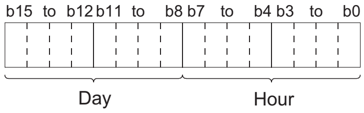
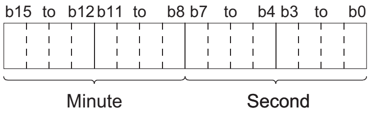
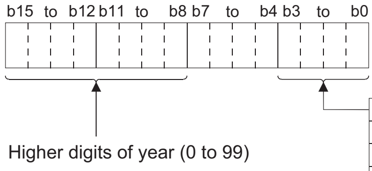
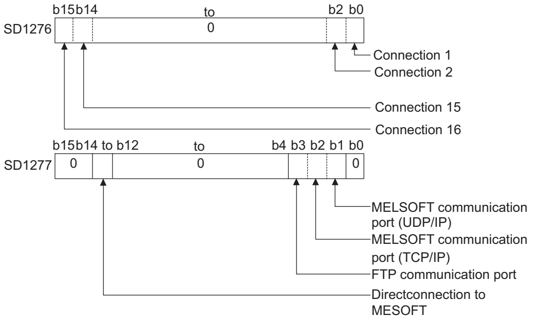
(13)Fuse blown module

TableApp.31 Special register

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When Set)</th><th>Corres-ponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD1300</td><td rowspan="11">Fuse blown module</td><td rowspan="11">Bit pattern in units of 16 points, indicating the modules whose fuses have blown
0 : No blown fuse
1 : Blown fuse present</td><td rowspan="11">• The numbers of output modules whose fuses have blown are input as a bit pattern (in units of 16 points).
(If the module numbers are set by parameter, the parameter-set numbers are stored.)
• Also detects blown fuse condition at remote station output modules
<table>
<tr><th></th><th>b15</th><th>b14</th><th>b13</th><th>b12</th><th>b11</th><th>b10</th><th>b9</th><th>b8</th><th>b7</th><th>b6</th><th>b5</th><th>b4</th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>SD1300</td><td>0</td><td>0</td><td>0</td><td>1 (YC0)</td><td>0</td><td>0</td><td>0</td><td>1 (Y80)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD1301</td><td>1 (Y1F0)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (Y1A0)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD1331</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (Y1F B0)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (Y1F 30)</td><td>0</td><td>0</td><td>0</td></tr>
</table>
Indicates fuse blow.
• Not cleared even if the blown fuse is replaced with a new one.
This flag is cleared by error resetting operation.</td><td rowspan="11">S (Error)</td><td>D9100</td><td rowspan="11">QnA
Qn(H)
QnPH
QnPRH
QnU
Rem</td></tr>
<tr><td>SD1301</td><td>D9101</td></tr>
<tr><td>SD1302</td><td>D9102</td></tr>
<tr><td>SD1303</td><td>D9103</td></tr>
<tr><td>SD1304</td><td>D9104</td></tr>
<tr><td>SD1305</td><td>D9105</td></tr>
<tr><td>SD1306</td><td>D9106</td></tr>
<tr><td>SD1307</td><td>D9107</td></tr>
<tr><td>SD1308</td><td>New</td></tr>
<tr><td>SD1309 to SD1330</td><td>New</td></tr>
<tr><td>SD1331</td><td>New</td></tr>
</table>

(14)QCPU with built-in Ethernet port

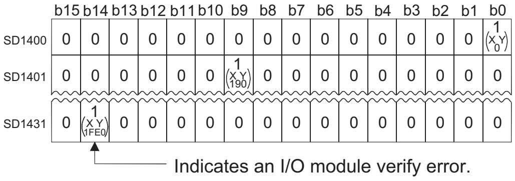
TableApp.32 Special register

Number	Name		Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD1270	Time setting function	Operation result	Stores operationresult.	Stores the operation result of the time setting function. 0: Not executed 1: Success FFFFH: Failure	S (status change)	New	QnU*1
SD1271		Execution time	Stores time acquired with time setting function.	Stores years (last two digits of the Christian Era) and monthes by two digits of BCD code.  Example: July, 1993 9307H			
SD1272				Stores dates and hours acquired with time setting function by two digits of BCD code.  Example: 31st, 10 a.m. 3110H			
SD1273				Stores minutes and seconds acquired with time setting function by two digits of BCD code.  Example: 35 min., 48 sec. 3548H			
SD1274				Stores years (first two digits of the Christian Era) and days acquierd with time setting function.  b15 to b12 b11 to b8 b7 to b4 b3 to b0 Higher digits of year (0 to 99) Example: 1993, Friday 1905H Day of the week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday			
SD1275		Required response time	Stores time required for clock time aquisition.	Stores time taken from transmission to SNTP server until time is set up at CPU. Range: 0 to FFFEH (Unit: ms) FFFFH when the above limit is exceeded.			
SD1276 SD1277	Forced connection invalidation		Specifies forced connection invalidation.	Specify this when a connection is to be invalidated forcibly on the user program. If invalidation is specified for a connection, it stops communication and does not respond. (When a remote password is used and frequent unlock processing errors have occurred on a connection, this is useful for temporarily inhibiting access to the connection.)  b15b14 to b2 b0 SD1276 0 Connection 1 Connection 2 Connection 15 Connection 16 b15b14 to b12 to b4 b3 b2 b1 b0 SD1277 0 0 0 MELSOFT communication port (UDP/IP) MELSOFT communication port (TCP/IP) FTP communication port Directconnection to MESOFT 0: Valid (default) 1: Invalid	U		

*1: This applies to the QnUDE(H)CPU.

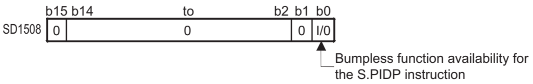
(15)I/O module verification

TableApp.33 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD1400	I/O module verify error	Bit pattern, in units of 16 points, indicating the modules with verification errors. 0 : No I/O verification errors 1 : I/O verification error present	• When the I/O modules whose I/O module information differs from that registered at power-ON are detected, the numbers of those I/O modules are entered in bit pattern. (If the I/O numbers are set by parameter, the parameter-set numbers are stored.) • Also detects I/O module information.  b15 b14 b13 b12 b11 b10 b9 b8 b7 b6 b5 b4 b3 b2 b1 b0 SD1400 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 (X Y 0) SD1401 0 0 0 0 0 0 1 (X Y 190) 0 0 0 0 0 0 0 0 0 SD1431 0 1 (X Y 1FE0) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Indicates an I/O module verify error. • Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.	S (Error)	D9116	QnA Qn(H) QnPH QnPRH QnU Rem
SD1401					D9117	
SD1402					D9118	
SD1403					D9119	
SD1404					D9120	
SD1405					D9121	
SD1406					D9122	
SD1407					D9123	
SD1408					New	
SD1409 to SD1430					New	
SD1431					New	

(16)Process control instructions

TableApp.34 Special register

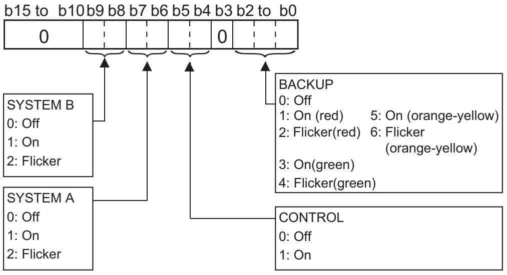
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD1500 SD1501	Basic period	Basic period tome	• Set the basic period (1 second units) use for the process control instruction using floating point data. Floating point data = SD1501 SD1500	U	New	Q4AR QnPH
SD1502	Process control instruction detail error code	Process control instruction detail error code	• Shows the detailed error contents for the error that occurred in the process control instruction.	S (Error)	New	
SD1503	Process control instruction generated error location	Process control instruction generated error location	• Shows the error process block that occurred in the process control instruction.	S (Error)	New	
SD1506 SD1507	Dummy device	Dummy device	• Used to specify dummy devices by a process control instruction.	U	New	QnPH QnPRH
SD1508	Function availability selection for process control instruction	b0 Bumpless function availability setting for the S.PIDP instruction 0: Enabled 1: Disabled (Default: 0)	• Selects the availability (enabled/disabled) of the function for process control instructions.  b15 b14 to b2 b1 b0 SD1508 0 0 0 I/O Bumpless function availability for the S.PIDP instruction	U	New	QnPH QnPRH

(17)For redundant systems (Host system CPU information [*1])

SD1510 to SD1599 are only valid for redundant systems.

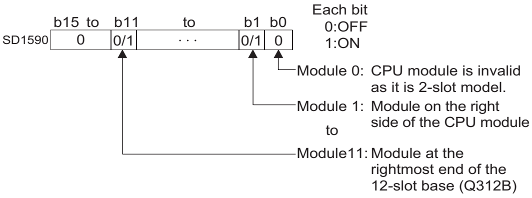
They are all set to 0 for stand-alone systems.

TableApp.35 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD1512	Operation mode during CPU module start up	Hot start switch power out time	• Shows the power out time (S) during the automatic switch from hot start to initial start in the operation mode when the CPU module is started up.	S (Initial)	New	Q4AR
SD1585	Redundant system LED status	4 LED states • BACKUP • CONTROL • SYSTEM A • SYSTEM B	The LED status for BACKUP, CONTROL, SYSTEM A, SYSTEM B is stored in the following format:  b15 to b10 b9 b8 b7 b6 b5 b4 b3 b2 to b0 0 0 SYSTEM B 0: Off 1: On 2: Flicker SYSTEM A 0: Off 1: On 2: Flicker BACKUP 0: Off 1: On (red) 2: Flicker(red) 3: On(green) 4: Flicker(green) 5: On (orange-yellow) 6: Flicker (orange-yellow) CONTROL 0: Off 1: On	S (status change)	New	QnPRH
SD1588	Reason(s) for system switching	Reason(s) for system switching that occurred in host station	Stores the reason(s) for system switching on the host system. The following values are stored corresponding to the methods for system switching: Initialized to 0 when the power supply is switched off and then on or the RESET switch is set to the RESET position and then to the neutral position. 0: Initial value (control system has not been switched) 1: Power off, Reset, H/W failure, WDT error, 2: CPU stop error (except WDT) 3: System switching request from network module 16: System switching dedicated instruction 17: System switching request from GX Developer	S (when condition occurs)	○	

[*1] The information of the host CPU module is stored.

TableApp.35 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU
SD1589	Reason(s) for system switching failure conditions	Reason(s) for system switching failure No.	• Stores the reason(s) for system switching failure. 0: System switching normal (default) 1: Tracking cable is not connected , tracking cable error, FPGA circuit failure. 2: H/W failure, power-OFF, Reset, WDT error on the standby system 3: H/W failure, power-OFF, Reset, WDT error on the Control system 4: Tracking data transfer initialization 5: Communication timeout 6: Serious error(except WDT error) on the Standby system 7: There is difference between both systems (detected as Backup mode only) 8: During memory copy from control system to standby system 9: During online program change 10: During detection of intelligent function module failure on the standby system 11: System switching being executed • Resets to "0" when host system is powered on. • Resets to "0" once system has been switched successfully.	S(when system is switched)	○	QnPRH
SD1590	Switch request network No.	Request source network No.	• Stores the request source at work No. when the SM1590 is turned on.	S (Error)	New	Q4AR
	Network module head address, which requested system switching	Network module head address, which requested system switching	• Stores head address of network module which a system switch request was initiated. • Turns off automatically by system, after network error is reset by user.  Each bit 0:OFF 1:ON b15 to b11 to b1 b0 SD1590 0 0/1 · · · 0/1 0 Module 0: CPU module is invalid as it is 2-slot model. Module 1: Module on the right side of the CPU module to Module11: Module at the rightmost end of the 12-slot base (Q312B) • Please refer to SD1690 which stores the corresponding head address of network module on other system.	S (Error/Status change)	New	QnPRH
SD1595	Memory copy target I/O number	Memory copy target I/O number	• Stores the memory copy target I/O No.(Standby system CPU module: 3D1H) of before SM1595 is turned from OFF to ON.	U	New	
SD1596	Memory copy status	Memory copy status	• Stores the execution result of Memory copy function. 0 : Memory copy successfully completed 4241H : Standby system power supply off 4242H : Tracking cable is disconnected or is damaged 4247H : Memory copy function is being executed 4248H : Unsupported memory copy destination I/O Number	S (Status change)	New	

(18)For redundant systems (Other system CPU information [*1])

SD1600 to SD1659 is only valid during the back up mode for redundant systems, and refresh cannot be done when in the separate mode.

SD1651 to SD1699 are valid in either the backup mode or separate mode.

When a stand-alone system SD1600 to SD1699 are all 0.

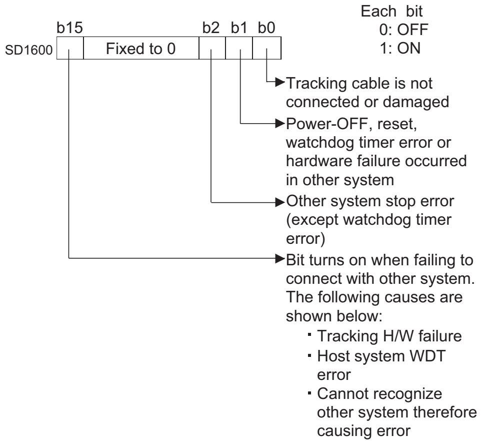
TableApp.36 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU SD□□*2	Corresponding CPU
SD1600	Diagnosis error	Diagnosis error No.	• Stores as BIN code the error No. of the error that occurred during the other system CPU module diagnosis. • Stores the latest error currently occurring.	S (Each END)	SD0	Q4AR
SD1601	Diagnosis error occurrence time	Diagnosis error occurrence time	• SD1600 stores the updated date and time. • Stores each of the BCD two digits. • Refer to SD1 to SD3 for the storage status. (SD1 → SD1601, SD2 → SD1602, SD3 → SD1603)	S (Each END)	SD1 to SD3	
SD1602						
SD1603						
SD1604	Error information classification	Error information classification	• Stores the error comment information/individual information classification code. • Refer to SD4 for the storage status.	S (Each END)	SD4	
SD1605	Error common information	Error common information	• Stores the common information for the error code. • Refer to SD5 to SD15 for the storage status. (SD5 → SD1605, SD6 → SD1606, SD7 → SD1607, SD8 → SD1608, SD9 → SD1609, SD10 → SD1610, SD11 → SD1611, SD12 → SD1612, SD13 → SD1613, SD14 → SD1614, SD15 → SD1615)	S (Each END)	SD5 to SD15	
SD1606						
SD1607						
SD1608						
SD1609						
SD1610						
SD1611						
SD1612						
SD1613						
SD1614						
SD1615						
SD1616	Error individual information	Error individual information	• Stores the individual information for the error code. • Refer to SD16 to SD26 for the storage status. (SD16 → SD1616, SD17 → SD1617, SD18 → SD1618, SD19 → SD1619, SD20 → SD1620, SD21 → SD1621, SD22 → SD1622, SD23 → SD1623, SD24 → SD1624, SD25 → SD1625, SD26 → SD1626)	S (Each END)	SD16 to SD26	
SD1617						
SD1618						
SD1619						
SD1620						
SD1621						
SD1622						
SD1623						
SD1624						
SD1625						
SD1626						
SD1650	Switch status	CPU module switch status	• Stores the CPU module switch status. • Refer to SD200 for the storage status. (SD1650 → SD200)	S (Each END)	SD200	
SD1651	LED status	CPU module -LED status	• Stores the CPU module's LED status. • Shows 0 when turned off, 1 when turned on, and 2 when flicking. • Refer to SD201 for the storage status. (SD1651 → SD201)	S (Each END)	SD201	
SD1653	CPU module operation status	CPU module operation status	• Stores the CPU module operation status. • Refer to SD203 for the storage status. (SD1653 → SD203)	S (Each END)	SD203	

[*1] : Stores other system CPU module diagnostics information and system information.

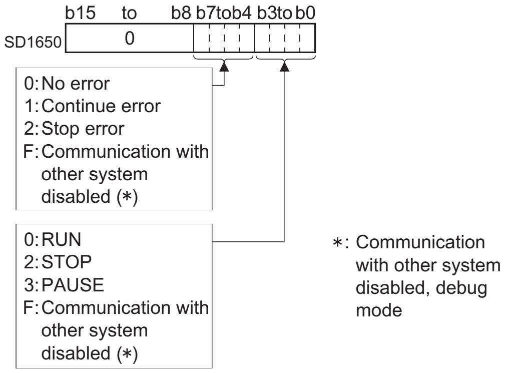
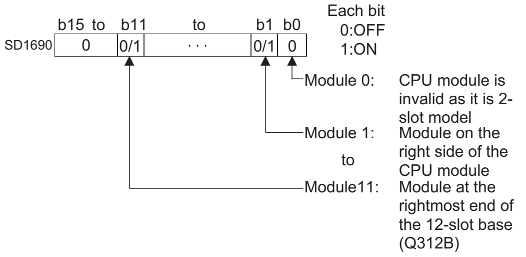
[*2] : Shows the special register (SD□□) for the host system CPU module.

TableApp.36 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU SD□□*2	Corresponding CPU
SD1600	System error information	System error information	• If an error is detected by the error check for redundant system, the corresponding bit shown below turns ON. That bit turns OFF when the error is cleared after that.  Each bit 0: OFF 1: ON b15 b2 b1 b0 SD1600 Fixed to 0 Tracking cable is not connected or damaged Power-OFF, reset, watchdog timer error or hardware failure occurred in other system Other system stop error (except watchdog timer error) Bit turns on when failing to connect with other system. The following causes are shown below: ・Tracking H/W failure ・Host system WDT error ・Cannot recognize other system therefore causing error • If any of b0, b1, b2 and b15 is ON, the other bits are all OFF. • In the debug mode, b0, b1, b2 and b15 are all OFF.	S(Every END)	–	QnPRH
SD1601	System switching results	System switching results	Stores the reasons for system switching. • Stores the reasons for system switching into SD1601 of both systems when system switching occarred. • Initialized to 0 at power OFF to ON/reset to unreset. • The following shows values stored into this register. 0: Initial value (System switching has not occurred) 1: Power-OFF, Reset, H/W failure, WDT error,(*) 2: CPU stop error (except WDT) 3: System switching request by network module 16: System switching dedicated instruction 17: System switching request from GX Developer *: When the system is switched by the power OFF/reset of the control system, "1" is not stored into SD1601 of the new standby system.	S(when system is switched)		
SD1602	System switching dedicated instruction parameter	System switching dedicated instruction parameter	• Stores the parameters for system switching dedicated instruction SP.CONTSW. (The parameters (SD1602) for SP.CONTSW are stored in both systems A&B) • SD1602 is only valid when "16" is stored in SD1601. • This SD1602 is updated once system switch instruction SP.CONTSW is activated.	S(when system is switched)		
SD1610	Other system diagnostic error	Diagnostic error code	• The error value sorted in BIN code. • Stores SD0 of the other system CPU module	S(Every END)	SD0	
SD1611 SD1612 SD1613	Other system diagnostic error occurrence time	Diagnostic error occurrence time	• Stores the date and time when diagnostics error occurred corresponding to error code stored in SD1610. • Data format is the same as SD1 to SD3. • Also, stores the value to SD1 to SD3.	S(Every END)	SD1 to SD3	
SD1614	Other system error information category	Error information category code	• Stores the category code corresponding to the error comment information/individual information code. • Data format is the same as SD4. • Also, stores the value to SD4.	S(Every END)	SD4	
SD1615 to SD1625	Other system error common information	Error common information	• Stores the common information corresponding to the error code stored in this system CPU. • Data composition is the same as SD5 to SD15. • Also, stores the value to SD5 to SD15.	S(Every END)	SD5 to SD15	
SD1626 to SD1636	Other system error individual information	Error individual information	• Stores the individual information corresponding to the error code stored in this system CPU. • Data composition is the same as SD16 to SD26. • Also, stores the value to SD16 to SD26.	S(Every END)	SD16 to SD26	
SD1649	Standby system error cancel command	Error code of error to be cleared	• Stores the error code of the error to be cleared by clearing a standby system error. • Stores the error code of the error to be cleared into this register and turn SM1649 from OFF to ON to clear the standby system error. • The value in the lowest digit (1 place) of the error code is ignored when stored into this register. (By storing 4100 in this register and resetting the error, errors 4100 to 4109 can be cleared.)	S(Every END)	–	

*2: Shows the special register (SD□ □) for the host system CPU module.

TableApp.36 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU SD□□*2	Corresponding CPU
SD1650	Other system operating information	Other system operating information	Stores the operation information of the other system CPU module in the following format. "00FFH" I stored when a communication error occurs, or when in debug mode.  b15 to b8 b7tob4 b3tob0 SD1650 0 0: No error 1: Continue error 2: Stop error F: Communication with other system disabled (*) 0: RUN 2: STOP 3: PAUSE F: Communication with other system disabled (*) *: Communication with other system disabled, debug mode Note : A communication error is caused by the following:. • When the power supply is switched off, or when the other system is reset. • H/W error occurs on either of system A or B. • WDT error occurs. • Tracking cable is not connected. • Tracking cable is disconnected or damaged.	S(Every END)	–	QnPRH
SD1690	Network module head address, which requested system switching on host (control) system	Network module head address, which requested system switching on host (control) system	• Stores head address of network module which a system switch request was initiated, using the following format. • Turns off automatically by system, after network error is reset by user.  b15 to b11 to b1 b0 SD1690 0 0/1 · · · 0/1 0 Each bit 0:OFF 1:ON Module 0: CPU module is invalid as it is 2-slot model Module 1: Module on the right side of the CPU module to Module11: Module at the rightmost end of the 12-slot base (Q312B) • Please refer to SD1590 which stores the corresponding head address of network module on host system.	S(Every END)		

*2 : Shows the special register (SD□□) for the host system CPU.

(19)For redundant systems (Trucking)

SD1700 to SD1779 is valid only for redundant systems.

These are all 0 for stand-alone systems.

TableApp.37 Special register

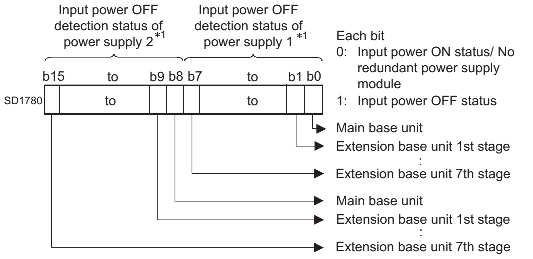
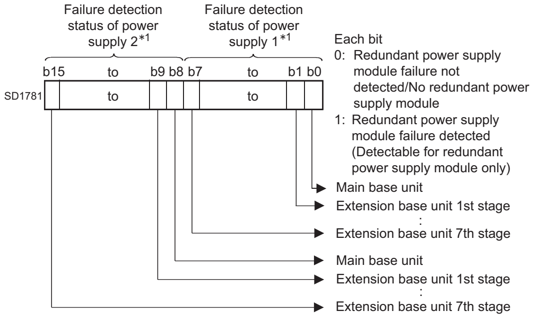
Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD1700	Tracking error detection count	Tracking error detection count	• When the tracking error is detected, count is added by one. • The counter repeats an increment and decrement of the value; 0→32767→-32768→0	S(Error)	New	Q4AR QnPRH
SD1710	Waiting time for online program change (standby system)	Waiting time for online program change (standby system)	• Set in seconds the waiting time of the standby system CPU module from when online program change to the control system CPU module is completed by the online program change for redundancy function until the online program change to the standby system CPU module starts. • If no online program change request is issued to the standby system CPU module within the preset time after completion of the online program change to the control system CPU module, both system CPU modules judge it as the failure of the online program change for redundancy. In this case, both system CPU modules resume the consistency check between system A & B suspended during the online program change. Also, the control system CPU module is set to accept a new request of online program change for redundancy. • When both systems are powered on, 90 seconds are set to SD1710 as the default value. • Set the value within the range 90 to 3600 seconds. When the setting is 0 to 89 seconds, it is regarded as 90 seconds for operation. If the setting is outside the allowed range, it is regarded other than 0 to 3600 seconds for operation. • The waiting time for a start of online program change to the standby system CPU module is checked according to the SD1710 setting during online change of multiple blocks and online change of batch of files for redundancy.	U/ S (Initial)		QnPRH

(20)Redundant power supply module information

SD1780 to SD1789 are valid only for a redundant power supply system.

The bits are all 0 for a singular power supply system.

TableApp.38 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-ponding ACPU D9□□□	Corresponding CPU
SD1780	Power supply off detection status	Power supply off detection status	• Stores the status of the redundant power supply module with input power OFF in the following bit pattern. • Stores 0 when the main base unit is not the redundant power main base unit (Q38RB).  Input power OFF detection status of power supply 2*1 Input power OFF detection status of power supply 1*1 b15 to b9 b8 b7 to b1 b0 SD1780 to to Each bit 0: Input power ON status/ No redundant power supply module 1: Input power OFF status Main base unit Extension base unit 1st stage : Extension base unit 7th stage Main base unit Extension base unit 1st stage : Extension base unit 7th stage • When configuring multiple CPU, the status is stored to 1st CPU module.	S(Every END)	New	Qn(H)*2 QnPH*2 QnPRH QnU*3 Rem
SD1781	Power supply failure detection status	Power supply failure detection status	• Stores the failure detection status of the redundant power supply module in the following bit pattern. (The corresponding bit is cleared to 0 when the input power to the faulty redundant power supply module is switched OFF after detection of the redundant power supply module failure.) • Stores 0 when the main base unit is not the redundant power main base unit (Q38RB).  Failure detection status of power supply 2*1 Failure detection status of power supply 1*1 b15 to b9 b8 b7 to b1 b0 SD1781 to to Each bit 0: Redundant power supply module failure not detected/No redundant power supply module 1: Redundant power supply module failure detected (Detectable for redundant power supply module only) Main base unit Extension base unit 1st stage : Extension base unit 7th stage Main base unit Extension base unit 1st stage : Extension base unit 7th stage • When configuring multiple CPU, the status is stored to 1st CPU module.	S(Every END)	New	
SD1782	Momentary power failure detection counter for power supply 1*1	Momentary power failure detection count for power supply 1	• Counts the number of times of momentary power failure of the power supply 1/2. • Monitors the status of the power supply 1/ 2 mounted on the redundant power main base unit (Q38RB) and counts the number of times of momentary power failure. Status of power supply 1/power supply 2 mounted on the redundant power extension base unit is not monitored. • When the CPU module starts, the counter of the power supply 1/ 2 is cleared to 0. • If the input power to one of the redundant power supply modules is turned OFF, the corresponding counter is cleared to 0. • The counter is incremented by 1 every time the momentary power failure of the power supply 1/ 2 is detected. (The counter repeats increment and decrement of the value; 0→ 32767→ -32768→ 0 (The system monitor of GX Developer shows the counter within the range between 0 and 65535.)) • Stores 0 when the main base unit is not the redundant power main base unit (Q38RB). • When configuring multiple CPU, the status is stored to 1st CPU module.	S(Every END)	New	
SD1783	Momentary power failure detection counter for power supply 2*1	Momentary power failure detection count for power supply 2		S(Every END)	New	

*1: The "power supply 1" indicates the redundant power supply module (Q64RP) mounted on the POWER 1 slot of the redundant base unit (Q38RB/68RB/ Q65WRB).
The "power supply 2" indicates the redundant power supply module (Q64RP) mounted on the POWER 2 slot of the redundant base unit (Q38RB/68RB/ Q65WRB).

*2: The module whose first 5 digits of serial No. is "04012" or later.
However, for the multiple CPU system configuration, this applies to all CPU modules whose first 5 digits of serial No. are "07032" or later.

*3: The module whose first 5 digits of serial No. is "10042" or later.

Appendix 3 List of Parameter No.

The parameter No. is stored into the special register (SD16 to 26), when an error occurs in the parameter settings.
The table for the parameter No. and parameter setting area is shown in this section.
Refer to Section 9.3 for the explanation of mn, **, M, N in the "Parameter No." column.

TableApp.39List of parameter No.

Item		Parameter No.	Referance
Label		0000_H	Section 9.1.1(1), Section 9.1.2(1)
Comment		0001_H	
I/O assignment	Type	0400_H	Section 4.7 Section 9.1.1(8) Section 9.1.2(10)
	Model name		
	points		
	Start XY (Start I/O No.)		
Basic setting	Base model name	0401_H	Section 4.4 Section 9.1.1(8) Section 9.1.2(10)
	Power model name		
	Extension cable		
	Slots		
Detail setting	Error time output mode	0403_H	Section 6.8 Section 9.1.1(8) Section 9.1.2(10)
	I/O response time	0405_H	Section 6.7 Section 9.1.1(8) Section 9.1.2(10)
	Control PLC	0406_H	Section 9.1.1(8) Section 9.1.2(10) QCPU User's Manual (Muitiple CPU System)
Switch setting		0407_H	Section 6.10 Section 9.1.1(8) Section 9.1.2(10)
Group No.		$05mn_H$	Section 9.3(2)
		$0Amn_H$	Section 9.3(2)
Redundant Parameter		$0D00_H$	Section 9.2 QnPRCPU User's Manual (Redundant System)

(Continued on next page)

TableApp.39 List of parameter No.

Item		Parameter No.	Referance
No. of PLC		0E00$_H$	Section 9.1.1(11) Section 9.1.2(12) QCPU User's Manual (Muitiple 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 9.1.1(2) Section 9.1.2(2) Section 10.2.10
	High speed		
RUN-PAUSE contact	RUN	1001$_H$	Section 6.6.1 Section 9.1.1(2) Section 9.1.2(2)
	PAUSE		Section 6.6.2 Section 9.1.1(2) Section 9.1.2(2)
Remote reset		1002$_H$	Section 6.6.3 Section 9.1.1(2) Section 9.1.2(2)
Output mode at STOP to RUN		1003$_H$	Section 6.4 Section 9.1.1(2) Section 9.1.2(2)
Floating point arithmetic processing		1004$_H$	Section 3.9.4 Section 9.1.2(2)
Common pointer No.		1005$_H$	Section 10.10.2 Section 9.1.2(2)
Points occupied by empty slot		1007$_H$	Section 4.6.1(5) Section 9.1.1(2) Section 9.1.2(2)
Interrupt program/fixed scan program setting		1008$_H$	Section 3.1.3 Section 9.1.1(2) Section 9.1.2(2)
System interrupt setting	Interrupt counter start No.		Section 9.1.1(2) Section 9.1.2(2) Section 10.2.11(4)
	Fixed scan interval(n : 28 to 31)		Section 9.1.1(2) Section 9.1.2(2) Section 10.11

(Continued on next page)

TableApp.39 List of parameter No.

Item			Parameter No.	Referance
Intelligent function module setting (interrupt pointer setting)			100A$_H$	Section 10.11, Section 9.1.2(2)
Module synchronization			100C$_H$	Section 9.1.1(2), Section 9.1.2(2)
A-PLC			100D$_H$	Section 9.1.2(2), Section 10.3.2, Section 10.3.3
Use serial communication			100E$_H$	Section 6.24, Section 9.1.1(9)
Transmission speed				
Sum check				
Transmission wait time				
RUN write setting				
System interrupt setting	High-speed interrupt setting	X Input	100F$_H$	Section 3.1.3, Section 6.22, Section 9.1.2(2)
		Y Output	1010$_H$	
		Buffer read	1011$_H$	
		Buffer write	1012$_H$	
Service processing setting			1013$_H$	Section 6.25.2
Latch data backup operation enabled contact			1014$_H$	Section 6.29
File register			1100$_H$	Section 9.1.2(3), Section 10.7
Comment file used in a command			1101$_H$	Section 9.1.2(3)
Initial Device value			1102$_H$	Section 6.26, Section 9.1.1(3), Section 9.1.2(3)
File for local device			1103$_H$	Section 10.14.2, Section 9.1.2(3)
Transfer data to the standard ROM at the time of latch data backup operation			1104$_H$	Section 6.29
File used for the SP.DEVST/S.DEVLD instruction			1105$_H$	Section 6.30

(Continued on next page)

TableApp.39 List of parameter No.

Item		Parameter No.	Referance
Device points		2000_H	Section 9.1.1(5), Section 9.1.2(5), Section 10.1, Section 10.2
Latch (1) start/end		2001_H	Section 3.7, Section 6.3, Section 9.1.1(5), Section 9.1.2(5)
Latch (2) start/end		2002_H	
Local device start/end		2003_H	Section 10.14.2, Section 9.1.2(5)
WDT (watchdog timer) setting	WDT setting	3000_H	Section 3.2, Section 9.1.1(4), Section 9.1.2(4)
	Initial execution monitoring time		Section 3.3.1, Section 9.1.2(4)
	Low speed execution monitoring time		Section 3.3.3 Section 9.1.2(4)
Error check	Carry out battery check	3001_H	Section 6.17, Section 9.1.1(4), Section 9.1.2(4)
	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	3002_H	Section 6.17, Section 9.1.1(4), Section 9.1.2(4)
	Extended instruction error		
	Fuse blown		
	Module comparison error		
	Intelligent module program execution error		
	File access error		
	Memory card operation error		
	External power supply OFF		

(Continued on next page)

TableApp.39 List of parameter No.

Item		Parameter No.	Referance
Constant scanning		3003$_{H}$	Section 6.2 Section 9.1.1(4) Section 9.1.2(4)
Breakdown history		3005$_{H}$	Section 6.18 Section 9.1.2(4)
Low speed program execution time		3006$_{H}$	Section 3.3.3 Section 9.1.2(4)
File register extended setting	Device point	2000$_{H}$	Section 9.1.2(6), Section 10.8
	Latch (1) start/end	2004$_{H}$	
	Latch (2) start/end	2005$_{H}$	
Indexing setting of ZR device		2000$_{H}$	Section 9.1.2(6)
Memory check		3008$_{H}$	Section 9.1.2(5)
Detail setting	Hardware error time PLC operation mode	4004$_{H}$	Section 6.9 Section 9.1.1(8)
Nunber of modules on MELSECNET/H		5000$_{H}$	Section 9.3(2)
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$_{H}$	
Refresh parameters		5NM1$_{H}$	
Setup Common parameters		5NM2$_{H}$	
Station inherent parameters		5NM3$_{H}$	
Sub-master parameters		5NM5$_{H}$	
Setup Common parameters 2		5NMA$_{H}$	
Station inherent parameters 2		5NMB$_{H}$	
Interrupt settings			

(Continued on next page)

TableApp.39 List of parameter No.

Item		Parameter No.	Referance
Program		7000$_H$	Section 3.3.6
Boot option	Clear program memory	7000$_H$	Section 5.1.5, Section 5.2.8, Section 5.2.9, Section 9.1.1(6), Section 9.1.2(8)
	Auto Download all Data from Memory card to Standard ROM		
Boot file setting			
SFC program start mode		8002$_H$	Section 9.1.1(7), Section 9.1.2(9)
Start condition		8003$_H$	
Output mode when the block is stopped		8005$_H$	
Nunber of Ethernet		9000$_H$	Section 9.3(3)
Starting I/O No.		9N00$_H$	
Network No.			
Group No.			
Station No.			
Operational settings			
Initial settings		9N01$_H$	Section 9.3(3)
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 9.3(1)
Interlink transmission parameters		A002$_H$	
Routing parameters		A003$_H$	
Starting I/O No.		ANM0$_H$	
Network No.			
Total stations			
Station No.			
Mode		ANM0$_H$	
Refresh parameters		ANM1$_H$	
Setup Common parameters		ANM2$_H$	
Station inherent parameters		ANM3$_H$	

(Continued on next page)

TableApp.39 List of parameter No.

Item	Parameter No.	Referance
Number of CC-Link	C000H	Section 9.3(4)
Remote input (RX)	CNM1H	
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)		
Operational settings	CNM2H	
All connect count		
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 settings		

(Continued on next page)

TableApp.39 List of parameter No.

Item		Parameter No.	Referance
Start mode setting		D001$_H$	Section 9.2(1), QnPRHCPU User’s Manual (Redundant System)
Standby system watch setting			
Debug mode setting			
Backup mode setting			
Tracking characteristics settings		D002$_H$	Section 9.2(2), Refer to [5.5.3 Tracking data] in QnPRHCPU User’s Manual (Redundant System)
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 setting		D004$_H$	Section 9.3(1), Section 9.3(2), Section 9.3(3)
Redundant settings		D5**$_H$	Section 9.3(2)
		D9**$_H$	Section 9.3(3)
		DA**$_H$	Section 9.3(1)
Communication area setting (refresh setting)		E002$_H$ E003$_H$	Section 9.1.2(11), Section 9.1.2(12), QCPU User’s Manual (Muitiple CPU System)
Online module change (hot swapping) setting		E006$_H$	Section 9.1.2(12), QCPU User’s Manual (Muitiple CPU System)
Refresh parameter detailed device specification		E007$_H$	
CPU specific send range		E008$_H$	
Auto refresh setting		E009$_H$	
Auto refresh device setting		E00A$_H$	
Multiple CPU synchronous startup setting		E00B$_H$	
Host CPU number		E00C$_H$	

(Continued on next page)

Appendix 4 Functions Added or Changed Due to Version Upgrade

The Q series CPU module has been upgraded with new functions and specifications. The functions and specifications that can be used with the CPU module change depending on the function version/serial No.

Appendix 4.1 Basic model QCPU Upgrade

(1) Specification comparison

TableApp.40 Specification comparison

Specifications		First 5 digits of serial No. of CPU Module	
		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	×	○

○ : Usable/compatible, × : Unusable/incompatible

(2) Additional functions and availability of the functions according to the version of GX Developer

TableApp.41 Additional functions and availability of the functions according to the version of GX Developer

New Function	Compatible function version	Compatible serial No.	Compatible GX Developer
Function block (☞ GX Developer Operating Manual (Function Block))	A	04121 or earlier	Version 8.00A or later
Structured text (ST) language (☞ QCPU (Q Mode) Programming Manual (Structured Text))			
MELSAP3 (☞ QCPU (Q Mode)/QnACPU Programming Manual (SFC))	B	04122 or later	
PID operation function[*1] (☞ QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions))			
Real number operation[*1] (☞ Section 10.13.3)			
Intelligent function module event interruption (☞ Section 6.23)			
Device initial value automatic setting function (☞ Section 6.26)			
Remote password setting function (☞ Section 6.19.2)			
E-mail parameter (☞ Manual of the module that includes e-mail function)			
Online change using pointer (☞ Section 6.15.2)			
Increased file register R capacity (32k points to 64k points)[*2] (☞ Section 10.7)			–
Multiple CPU System compatibility (☞ QCPU User's Manual (Multiple CPU System))			Version 8.00A or later
Multiple-block online change (☞ Section 5.1.2(3)(b), Section 5.2.2(3)(b))			
CC-Link Remote network additional mode (☞ CC-Link System Master/Local Module User's Manual)		06112 or later	Version 8.03D or later
Compatibility with CC-Link IE controller network (☞ CC-Link IE Controller Network Reference Manual)	–	–	Version 8.68W or later

– : Not related to function version, serial No., and GX Developer

*1 : When the CPU instruction installed by GX Developer Version 8 is read by GX Developer of Version 7 or earlier, it is processed as an "instruction code error" by GX Developer.

*2 : Unsupported by the Q00JCPU.

(3) Differences among Basic model QCPU models

TableApp.42 Differences among Basic model QCPU models

Item			Q00JCPU	Q00CPU	Q01CPU
CPU module			CPU module, Power supply module, Main base unit (5 slots) Integrated type	Stand-alone CPU module	
Main base unit/slim type main base unit			Unnecessary	Necessary (Q33B, Q35B, Q38B, Q312B)	
Extension base unit			Connectable However, an extension base unit cannot be connected when the slim type main base unit is used.		
Number of extension stages			Up to 2 stages	Up to 4 stages (Cannot be connected when the slim type main base unit is used)	
Number of input/output modules to be installed			16 modules	24 modules	
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			No		
External interface	RS-232		Yes (transmission rate: 9.6kbps, 19.2kbps, 38.4kbps, 57.6kbps, 115.2 kbps)		
	USB		No		
Processing speed (Sequence instruction)	LD X0		0.20μs	0.16μs	0.10μs
	MOV D0 D1		0.70μs	0.56μs	0.35μs
Program capacity[*1]			8k steps (32k byte)	8k steps (32k byte)	14k steps (56k byte)
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 bytes (user free area 320 words)	
Number of times for write to standard ROM			Maximum 100,000 times		
Device memory capacity			The number of device points can be changed within the range of 16.4k words.		
Number of input/output devices points (Remote I/O is contained.)			2048 points		
Number of input/output points			256 points	1024 points	
File register			No	Yes	
Serial communication function			No	Yes (using the RS-232 interface of the CPU module)	

*1 : 1 step of the program capacity is 4 bytes.

*2 : 64k bytes for the function version A.

*3 : Memory added to the function version B.
The CPU shared memory is not latched.
The CPU shared memory is cleared when the PLC is powered OFF and then ON or the CPU module is reset.

Appendix 4.2 High Performance model QCPU Upgrade

(1) Specification comparison

TableApp.43 Specification comparison

Specifications		First 5 digits of Serial No. of CPU Module				
		Function Version A		Function Version B		
		02091 or earlier	02092 or later	"02112" or later	03051 or later	04012 or later
Standard RAM capacity	Q02CPU	64kbyte				
	Q02HCPU	64kbyte				128kbyte
	Q06HCPU	64kbyte				128kbyte
	Q12HCPU	64kbyte	256kbyte			
	Q25HCPU	64kbyte	256kbyte			
CPU shared memory		×	×	○	○	○
Extended life battery SRAM card		×	×	×	×	○
Compatibility with 2Mbyte SRAM card		×	×	×	×	○

○ : Usable/compatible, × : Unusable/incompatible

(2) Additional functions and availability of the functions according to the version of GX Developer

TableApp.44 Additional functions and availability of the functions according to the version of GX Developer

New Function	Compatible function version	Compatible serial No.	Compatible GX Developer
Automatic write to standard ROM (☞ Section 5.2.8)	A	02092 or later	Version 6 or later
Enforced ON/OFF for external I/O (☞ Section 6.11.3)			
Remote password setting (☞ Section 6.19.2)			
Compatibility with MELSECNET/H remote I/O network (☞ Section 4.6.2)			
Interrupt module (QI60) compatibility (☞ Section 10.11)			
Programming module compatibility			–
Compatibility with the multiple CPU system (☞ QCPU User's Manual (Multiple CPU System))	B	02122 or later	Version 7 or later
Installation of PC CPU module into the multiple CPU system (☞ 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
Compatibility with index modification for module designation of dedicated instruction			–
Selection of refresh item for COM instruction			–
SFC program online batch change (☞ Section 6.12.2)		04122 or later	Version 8 or later
File memory capacity change (☞ Section 5.4.4)			
CC-Link remote network additional mode (☞ CC-Link Manual)		05032 or later	Version 8.03D or later
Incomplete derivative PID operation function (☞ QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions))			
Floating-point comparison instruction speedup			–
Read of the SFC activity step comment compatible(☞ QCPU (Q mode)/ QnACPU programming manual (SFC))		07012 or later	–
Error detection in the redundant power supply system		07032 or later	ersion 8.23Z or later
Clock data in 1/1000 sec. units compatible (☞ Section 6.5)			–
Storage of sampling trace files in standard RAM (☞ Section 6.14)			Version 8.23Z or later
Individual setting of refresh device on multiple CPU system (☞ QCPU User’s Manual (Multiple CPU System))			
Selection of trailing edge instruction execution/non-execution at online change(☞ Section 6.12.3)		07092 or later	Version 8.27D or later

(Continued on next page)

TableApp.44 Additional functions and availability of the functions according to the version of GX Developer

New Function	Compatible function version	Compatible serial No.	Compatible GX Developer
Selection of trailing edge instruction execution/non-execution at online change(☞ Section 6.12.3)	B	07092 or later	Version 8.27D or later
Block guarantee function of CC-Link cyclic data per station (☞ CC-Link System Master/Local Module User's Manual (Details))		08032 or later	Version 8.32J or later
CC-Link 8 setting-compatible parameters (☞ CC-Link System Master/Local Module User's Manual (Details))			
Compatibility with CC-Link IE controller network (☞ CC-Link IE Controller Network Reference Manual)		09012 or later	Version 8.45X or later
Compatibility with ATA card (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection))			

— : Not related to GX Developer

Appendix 4.3 Process CPU Upgreade

(1) Additional functions and availability of the functions according to the version of GX Developer

TableApp.45 Additional functions and availability of the functions according to the version of GX Developer

New Function	Compatible function version	Compatible serial No.	Compatible GX Developer
Compatibility with index modification for module designation of dedicated instruction	C	07032 or later	–
Selection of refresh item for COM instruction			
SFC program online batch change (☞ Section 6.12.2)			Version 8 (Version 8.22Y or earlier)
File memory capacity change (☞ Section 5.4.4)			
CC-Link remote network additional mode (☞ CC-Link Manual)			
Program memory check function (☞ Section 9.1.2(5))			Version 8.23Z or later
Read of the SFC activity step comment compatible (☞ QCPU (Q mode)/ QnACPU programming manual (SFC Manual))			–
Error detection in the redundant power supply system			Version 8.23Z or later
Clock data in 1/1000 sec. units compatible (☞ Section 6.5)			–
Storage of sampling trace files in standard RAM (☞ Section 6.14)			Version 8.23Z or later
Individual setting of refresh device on multiple CPU system (☞ QCPU User's Manual (Multiple CPU System))			
Selection of trailing edge instruction execution/non-execution at online change(☞ Section 6.12.3)		07092 or later	Version 8.27D or later
Block guarantee function of CC-Link cyclic data per station (☞ CC-Link System Master/Local Module User's Manual (Details))		08032 or later	Version 8.32J or later
CC-Link 8 setting-compatible parameters (☞ CC-Link System Master/Local Module User's Manual (Details))			
Compatibility with CC-Link IE controller network (☞ CC-Link IE Controller Network Reference Manual)		10042 or later	Version 8.68W or later

– : Not related to GX Developer

Appendix 4.4 Redundant CPU Upgrade

(1) Additional functions and availability of the functions according to the version of GX Developer

TableApp.46 Additional functions and availability of the functions according to the version of GX Developer

New Function	Compatible function version	Compatible serial No.	Compatible GX Developer
Read of the SFC activity step comment compatible (☞ QCPU (Q mode)/ QnACPU programming manual (SFC))	D	07032 or later	–
Clock data in 1/1000 sec. units compatible (☞ Section 6.5)			
Storage of sampling trace files in standard RAM (☞ Section 6.14)			Version 8.23Z or later
Selection of trailing edge instruction execution/non-execution at online change(☞ Section 6.12.3)		07092 or later	Version 8.27D or later
Extension base unit compatible (☞ QnPRHCPU User's Manual (Redundant System))		09012 or later	Version 8.45X or later
CC-Link 8 setting-compatible parameters (☞ CC-Link System Master/Local Module User's Manual (Details))		09102 or later	Version 8.58L or later
Compatibility with CC-Link IE controller network (☞ CC-Link IE Controller Network Reference Manual)		10042 or later	Version 8.68W or later

– : Not related to GX Developer

Appendix 4.5 Universal model QCPU Upgrade

(1) Additional functions and availability of the functions according to the version of GX Developer

TableApp.47 Additional functions and availability of the functions according to the version of GX Developer

New Function	Compatible function version	Compatible serial No.	Compatible GX Developer
PC CPU module compatible (☞ QCPU User's manual (Multiple CPU System))	B	09072 or later	–
Setting of whether to use local device for each program (☞ Section 10.14.2)		10012 or later	Version 8.62Q or later
Program batch transfer status confirmation function (SM165) compatible (☞ Section 5.2.3)			
Multiple CPU high speed transmission dedicated instruction compatible[*1] (☞ QCPU(Q Mode)/QnACPU Programming Manual (Common Instructions))			–
Amount of battery consumption display (☞ QCPU User's Manual (Hardware Design, Maintenance and Inspection)			
Bit device extension (☞ Section 10.2)		10042 or later	Version 8.68W or later
Executional conditioned device test (☞ Section 6.11.4)			
Sampling trace auto start function (☞ Section 6.14)			
CC-Link IE group cyclic communication function (☞ CC-Link IE Controller Network Reference Manual)			
Scan time measurement (☞ Section 6.13.3)			
Forced ON/OFF of external I/O (☞ Section 6.11.3)			
Monitor condition setting (☞ Section 6.11.1)			
Redundant power supply system compatible (☞ Section 6.20)			
32-bit indexing with "ZZ" specification (☞ QCPU(Q Mode)/QnACPU Programming Manual (Common Instructions))			
Extended data register (D) and extended link register (W)(☞ Section 10.8)		09042 or later[*2]	Version 8.70Y or later

– : Not related to GX Developer

[*1] This function is not supported in the Q02UCPU.

[*2] If the data of the extended data register (D) and extended link register (W) are retained in the standard ROM by using the latch data backup function (☞ Section 6.29) to the standard ROM, use the Universal model QCPU whose serial number (first five digits) is "10042" or later.

Appendix 5 Method of replacing High Performance model QCPU with Universal model QCPU

Appendix 5.1 PRECAUTIONS FOR REPLACEMENT

This chapter describes the precautions for replacing the High Performance model QCPU with the Universal model QCPU and the replacement methods.

(1) System configuration

TableApp.48 Precautions for replacement and replacement methods (System configuration)

Item	Precautions	Replacement method	Reference
Use of AnS/A series module	AnS/A series modules are not supported.	Use Q series modules.	---
GOT	GOT900 series cannot be connected.	Use GOT1000 series.	---
Programming tool connection	Applicable USB cables are different. • High Performance model QCPU ... A-B type • Universal model QCPU ... A-miniB type	Use USB cables of A-miniB type. Or, use USB conversion adapters of B-miniB type.	---
Applicable device and software	Devices and softwares compatible with the Universal model QCPU shall be used.	Devices need to be replaced for the compatibility with the Universal model QCPU and software need to be upgraded for the communication with the Universal model QCPU are described in Appendix 5.2.	Appendix 5.2
Multiple CPU system	To configure a multiple CPU system, CPU modules compatible with the Universal model QCPU shall be used.	CPU modules compatible with the Universal model QCPU are described in Appendix 5.2.	Appendix 5.2
	In a multiple CPU system using the Motion CPU, an existing auto refresh area and user setting area cannot be used for data communication with the Motion CPU.	For data communication with the Motion CPU, use an auto refresh area and user setting area in the multiple CPU high-speed transmission area.	Chapter 4 in the "QCPU User's Manual (Multiple CPU System)"
Redundant power supply system*1	In a redundant power supply system using the Q38RB redundant power supply main base unit and the Q68RB redundant power supply extension base unit, the status of the power supply module cannot be stored in the special relays and special registers (SM1780 to SM1783/SD1780 to SD1783). The status neither can be displayed on the system monitor.	Check the status of the power supply module by the LED on the front of the module.	Section 5.3 in the "QCPU User's Manual (Hardware Design, Maintenance and Inspection)"
MELSECNET/H	The simple dual-structured network function is not supported.	---	Section 7.7 in the "Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)"

* 1: The Universal model QCPU whose serial number (first five digits) is "10041" or earlier is targeted.

(2) Program

TableApp.49 Precautions for replacement and replacement methods (Program)

Item	Precautions	Replacement method	Reference
Language and instruction	Some instructions are not supported.	Replace the instructions not supported in the Universal model QCPU are described in Appendix 5.3.	Appendix 5.3
Floating-point operation	The Universal model QCPU operates floating-point data in single precision.	Instructions for floating-point double-precision operation are added for the Universal model QCPU.Replace the instructions if floating-point double-precision operation is required, as described in Appendix 5.4.	Section 3.9.4, Appendix 5.4, Appendix 5.4
	When using the floating-point data comparison instructions, LDE□, ANDE□, ORE□, LDED□, ANDED□, and ORED□, if the comparison source data are -0, nonnumeric, unnormalized number, and ±∞, an OPERATION ERROR (error code: 4101) is detected. (□indicates one of the followings; =,<>,<=,>=,<,>.)	When the floating-point data comparison instructions are used, modify the program as described in Appendix 5.4.	
Device range check at index modification	When a device number exceeds a setting range due to index modification, an OPERATION ERROR (error code: 4101) is detected.	Uncheck the checkbox for "Check device range at indexing" on the PLC RAS setting tab of PLC parameter so that checking is not performed.	Section 6.17
Program execution type	Low-speed execution type programs are not supported.	Use scan execution type programs or fixed scan execution type programs.	Section 3.3.3
	A program execution type cannot be changed by remote operation.	Use instructions for switching program execution types, such as PSTOP, POFF, and PSCAN.	Section 3.3.6
Latch setting	If latch ranges of internal user devices are specified, the processing time is increased in proportion to the device points set for latching. (For example, if 8k points are latched for a latch relay (L), the scan time will be 28.6 μs.)	The latch function of the Universal model QCPU is enhanced. (1) Large-capacity file registers (R, ZR) (2) Writing/reading device data to the standard ROM (SP.DEVST/ S(P).DEVLD instructions) (3) Specifying internal devices as latch ranges Change the latch method to the method described in (1) to (3) above according to the application.	Section 6.3, Appendix 5.4
Interrupt program	The interrupt pointer (I49) for the high-speed interrupt function is not supported.	Consider the use of interrupt pointers for fixed scan interrupt (I28 to I31).	Section 6.22, Section 6.13.2, and Section 10.2.11
	Interrupt counters are not supported.	Check the numbers of executions for interrupt programs on the Interrupt program monitor list screen of GX Developer.	
	The interrupt pointers due to error occurrence (I32 to I40) are not supported.	---	Section 10.11

(Continued on next page)

TableApp.46 Precautions for replacement and replacement methods (Program) (Continued)

Item	Precautions	Replacement method	Reference
SCJ instruction	When the SCJ instruction is used in the Universal model QCPU, the AND SM400 (or NOP instruction) needs to be inserted immediately before the SCJ instruction.	Insert the AND SM400 (or NOP instruction) immediately before the SCJ instruction when the SCJ instruction is used.	Section 6.5.1 in the "QCPU(Q Mode)/ QnACPU Programming Manual (Common Instructions)"
ZPUSH instruction	The number of index registers is increased to 20 for the Universal model QCPU. The areas for saving index registers with the ZPUSH instruction need to be increased as well.	Increase the save areas used for the ZPUSH instruction as needed.	Section 7.18.8 in the "QCPU(Q Mode)/ QnACPU Programming Manual (Common Instructions)"
File usability setting for each program	The following file usability setting for each program is not available. [*1] • File register • Device initial value • Comment	When file usability is set, modify the program as described in Appendix 5.4.	Section 3.3.6, Appendix 5.4 and Appendix 5.4
I/O refresh setting for each program	I/O refresh setting for each program is not available.	Use the RFS instruction if I/O refresh setting for each program is required.	Section 3.3.6 in the "QCPU(Q Mode)/ QnACPU Programming Manual (Common Instructions)"
SM/SD	Usage of some of the special relays and special registers is different.	Replace corresponding special relays and special registers as described in Appendix 5.5.	Appendix 5.5
	A-compatible special relays and special registers are not supported.(SM1000 to SM1255/SD1000 to SD1255)	Using GX Developer, A-compatible special relays and special registers can be replaced with the Universal model QCPU-compatible special relays and special registers. Note that the ones which are not compatible with the Universal model QCPU are replaced with SM1255 and SD1255. Modify programs as needed.	Appendix 1 and Appendix 2
Processing time	Scan time and other processing times are different.	Modify programs as needed, checking the processing timing.	---

* 1: The local device file usability setting is not available for the Universal model QCPU when the first five digits of the serial number are "10011" or earlier.

(3) Drive and file

TableApp.50 Precautions for replacement and replacement methods (Drive and file)

Item	Precautions	Replacement method	Reference
Boot file setting	Files in the standard ROM cannot be booted to the program memory.	Since the Universal model QCPU retains the data in the program memory even the battery voltage drops, the boot file setting is not necessary.Move the files, which are booted from the standard ROM to the program memory, to the program memory.	Section 5.2.9 and Section 5.2.11, Appendix 5.4
	Booting operation is different.	Replacement method when the parameter-valid drive and the boot file setting are set in the High Performance model QCPU is described in Appendix 5.4.	
Automatic all data write from memory card to standard ROM	The setting method of this function is different.	On the Boot file setting tab of PLC parameter, select "standard ROM" for the transfer destination. Note, however, that the transfer destination of "program" is fixed to "program memory". (Setting by DIP switches is not necessary.)	Section 5.2.8 and Section 5.2.9

(4) External communication

TableApp.51 Precautions for replacement and replacement methods (External communication)

Item	Precautions	Replacement method	Reference
Module service interval time read	The module service interval time cannot be read.	---	Section 6.25.1
MC protocol	The following frame types cannot be used when accessing the Universal model QCPU. • A-compatible 1C frame • A-compatible 1E frame	Use the frame types below. • QnA-compatible 2C/3C/4C frame • QnA-compatible 3E frame • 4E frame	"Q Corresponding MELSEC Communication Protocol Reference Manual"

(5) Diagnostic function

TableApp.52 Precautions for replacement and replacement methods (Diagnostic function)

Item	Precautions	Replacement method	Reference
Error history	Error history data cannot be stored in the memory card.	The Universal model QCPU stores all storable data (maximum 100) in the built-in memory.	Section 6.18.2
LED indication priority setting	LED indication priority cannot be set. Only LED indication setting at error occurrence is supported.	---	Section 6.21.2

(6) Debugging

TableApp.53 Precautions for replacement and replacement methods (Debugging)

Item	Precautions	Replacement method	Reference
Monitor[*3]	The monitoring condition cannot be set.	Use the sampling trace function for checking device data whose monitoring condition is specified.With this function, changes of the specified device data can be recorded at the following timings: • at the execution of the specified step • at the rising/falling edge of bit devices • when the value of word devices coincide with the setting value • at every specified time(settable range: 1ms to 5000ms)	Section 6.11.1 and Section 6.14
Scan time measurement with GX Developer[*3]	Time required for executing a part of the program cannot be measured using the scan time measurement function. [*1]	Calculate the time using instruction processing time described in the manual.	• Section 6.13.3 • Appendix 1.4 in the "QCPU(Q Mode) / QnACPU Programming Manual (Common Instructions)"
Forced ON/OFF of external I/O[*3]	The forced ON/OFF function of external I/O is not supported. [*2]	Programs can be replaced as described in Appendix 5.4. Note, however, that replacement method described does not apply in the following cases: • Input and output targeted for forced ON/OFF are referred or changed using the direct input device (DX)/direct output device (DY). • Input and output targeted for forced ON/OFF are referred or changed within an interrupt program.	Section 6.11.3, Appendix 5.4

* 1: Scan time in units of programs can be checked on the Program monitor list screen.
* 2: Device test using GX Developer can be conducted.
* 3: The Universal model QCPU whose serial number (first five digits) is "10041" or earlier is targeted.

(7) CPU front switch

TableApp.54 Precautions for replacement and replacement methods (CPU front switch)

Item	Precautions	Replacement method	Reference
CPU module front switch	The operation method with the RESET/RUN/ STOP switch is modified.	The RESET/STOP/RUN switch of the Universal model QCPU can be used for the reset operation of the CPU module and switching between STOP and RUN status.	Section 4.4 in the "QCPU User's Manual (Hardware Design, Maintenance and Inspection)"
	Latch clear cannot be performed by the switch.	Latch clear can be performed by the remote latch clear operation of GX Developer.	Section 4.4 in the "QCPU User's Manual (Hardware Design, Maintenance and Inspection)"
	The system protect cannot be set by the switch.	Data in the files can be protected by setting a password for each file. Password for each file can be registered with GX Developer.	Section 6.19
	The parameter-valid drive setting is not necessary.	The Universal model QCPU automatically determines the parameter-valid drive.Change the setting as described in Appendix 5.4 when the parameter-valid drive is set to other than the program memory in the High Performance model QCPU.	• Appendix 5.4 • Section 4.4 in the "QCPU User's Manual (Hardware Design, Maintenance and Inspection)"

(8) SFC

TableApp.55 Precautions for replacement and replacement methods (SFC)

Item	Precautions	Replacement method	Reference
Step transition monitoring timer	The step transition monitoring timer is not supported.	Change the program as described in Appendix 3.1 in the manual in the Reference column.	Section 4.6 and Appendix 3.1 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
SFC operation mode setting	The periodic execution block setting is not supported.	Change the program as described in Appendix 3.2 in the manual in the Reference column.	• Section 4.7.4 and • Appendix 3.2 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
	An operation mode at double block START cannot be selected. (Fixed to “WAIT”.)	---	Section 4.7.5 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
	An operation mode at transition to active step cannot be selected. (Fixed to “TRANSFER”.)	---	Section 4.7.6 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
SFC program for program execution management	SFC programs for program execution management are not supported.	---	Section 5.2.3 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
SFC control instruction	Some SFC control instructions are not supported.	SFC control instructions not supported in the Universal model QCPU and alternative methods are described in Appendix 5.3.	• Appendix 5.3 • Section 4.4 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
SFC comment readout instruction	The following SFC comment readout instructions are not supported. • S(P).SFCSCOMR (SFC step comment readout instruction) • S(P).SFCTCOMR (SFC transition condition comment readout instruction)	---	Section 4.8 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"
Method of SFC program change	SFC program files cannot be written during RUN.(Programs in SFC Figure can be changed by writing files during RUN.)	Execute the Write to PLC function after changing the Universal model QCPU into the STOP status.	Section 6.6 in the "QCPU (Q Mode)/ QnACPU Programming Manual (SFC)"

Appendix 5.2 APPLICABLE DEVICES AND SOFTWARE

(1) Devices need to be replaced for the compatibility with the Universal model QCPU

The following tables show devices need to be replaced for the compatibility with the Universal model QCPU. (As for devices not listed in the tables below, replacement is not required.)

TableApp.56 Devices need to be replaced (Communication module)

Product	Model	Universal model QCPU-compatible version*2		
		Used with Q02UCPU, Q03UDCPU, Q04UDHCPU or Q06UDCPU	Used with Q13UDHCPU or Q26UDCPU	Used with Built-in Ethernet port QCPU
Web server module *1	• QJ71WS96	The serial number (first five digits) is "09042" or later.	The serial number (first five digits) is "10012" or later.	The serial number (first five digits) is "10012" or later.
MES interface module	• QJ71MES96	The serial number (first five digits) is "09042" or later.	The serial number (first five digits) is "10012" or later.	The serial number (first five digits) is "10012" or later.

* 1: The Universal model QCPU does not operate normally when the Web server module on which GX RemoteService-I is installed is used.

* 2: The Universal model QCPU does not operate normally when a device whose version is not compatible is used.

TableApp.57 Devices need to be replaced (Personal computer board)

Product		Model	Universal model QCPU-compatible dedicated software package version*1		
			Used with Q02UCPU, Q03UDCPU, Q04UDHCPU or Q06UDCPU	Used with Q13UDHCPU or Q26UDCPU	Used with Built-in Ethernet port QCPU
CC-Link IE controller network interface board		• Q80BD-J71GP21-SX • Q80BD-J71GP21S-SX	No restrictions	Version 1.03D or later	Version 1.03D or later
MELSECNET/ H(10) board	SI/QSI/H-PCF optical cable	• Q80BD-J71LP21-25 • Q80BD-J71LP21S-25	Version 15R or later	Version 18U or later	Version 18U or later
	GI optical cable	• Q80BD-J71LP21G			
	Coaxial cable	• Q80BD-J71BR11			
CC-Link Ver.2 board		• Q80BD-J61BT11N	Version 02C or later	Version 1.05F or later	Version 1.05F or later

* 1: No restrictions on the board itself.

TableApp.58 Devices need to be replaced (GOT)

Product	Model	Universal model QCPU-compatible operating system version provided with GT Designer2*1		
		Used with Q02UCPU, Q03UDCPU, Q04UDHCPU or Q06UDCPU	Used with Q13UDHCPU or Q26UDCPU	Used with Built-in Ethernet port QCPU
GOT1000	GT15□-□ GT11□-□	Version 2.60N or later	Version 2.76E or later	Version 2.81K or later
	GT10□-□	Version 2.76E or later	Version 2.76E or later	---

* 1: No restrictions on GOT itself.

TableApp.59 Devices need to be replaced (Network module and serial communication module)

Product	Model	Universal model QCPU-compatible module version		
		Used with Q02UCPU, Q03UDCPU, Q04UDHCPU or Q06UDCPU	Used with Q13UDHCPU or Q26UDCPU	Used with Built-in Ethernet port QCPU
MELSECNET/H model	• QJ71LP21-25 • QJ71LP21S-25 • QJ71LP21G • QJ71BR11	No restrictions	No restrictions	Some restrictions depending on use conditions[1]
Serial communication module	• QJ71C24N • QJ71C24N-R2 • QJ71C24N-R4			
Modem interface module	• QJ71CMON			The serial number (first five digits) is "10042" or later.

[1] Use the MELSECNET/H module whose serial number (first five digits) is "10042" or later if all conditions 1) to 4) described below are satisfied.

1) A multiple CPU system including Built-in Ethernet port QCPU is configured.
2) GX Developer or GOT is connected to an Ethernet port of Built-in Ethernet port QCPU.
3) GX Developer or GOT accesses the CPU module on another station via the MELSECNET/H module controlled by another CPU.
4) The access target on another station is A/QnA series CPU module.

(2) CPU modules that can configure a multiple CPU system with the Universal model QCPU

CPU modules that can configure a multiple CPU system with the Universal model QCPU are shown below.

(a) CPU modules that can configure a multiple CPU system with the Universal model QCPU (except Q02UCPU)

TableApp.60 CPU modules that can comfiture a multiple CPU system with the Universal model QCPU (except Q02UCPU)

CPU module	Model	Applicable version		Restrictions
		Configured with Q03UDCPU, Q04UDHCPU or Q06UDCPU	Configured with Q13UDHCPU, Q26UDCPU or Built-in Ethernet port QCPU	
Motion CPU	• Q172DCPU • Q173DCPU	No restrictions	No restrictions	Use the multiple CPU high-speed main base unit (Q3□DB) for a main base unit.
PC CPU module	• PPC-CPU852(MS)	Driver S/W (PPC-DRV-02) version 1.01 or later	Driver S/W (PPC-DRV-02) version 1.02C or later	---
High Performance model QCPU	• Q02CPU • Q02HCPU • Q06HCPU • Q12HCPU • Q25HCPU	Function version B or later	Function version B or later	---
Process CPU	• Q02PHCPU • Q06PHCPU • Q12PHCPU • Q25PHCPU	No restrictions	No restrictions	---

(b) CPU modules that can configure a multiple CPU system with the Q02UCPU

TableApp.61 CPU modules that can configure a multiple CPU system with the Q02UCPU

CPU module	Model	Applicable version	Restrictions
Motion CPU	• Q172CPUN(-T) • Q173CPUN(-T) • Q172HCPU(-T) • Q173HCPU(-T)	No restrictions	The multiple CPU high-speed main base unit (Q3□DB) cannot be used for a main base unit.
PC CPU module	• PPC-CPU852(MS)	Driver S/W (PPC-DRV-02) version 1.01 or later	---

(3) Software need to be upgraded for the compatibility with the Universal model QCPU

The following table shows software need to be upgraded for the communication with the Universal model QCPU. (As for software not listed in the table below, version upgrade is not required.)

The latest version can be downloaded from the MELFANSweb.

TableApp.62 Software need to be upgraded

Product	Model	Universal model QCPU-compatible version		
		Used with Q02UCPU, Q03UDCPU, Q04UDHCPU or Q06UDCPU	Used with Q13UDHCPU or Q26UDCPU	Used with Built-in Ethernet port QCPU
GX Developer	SW8D5C-GPPW-E	Version 8.48A or later	Version 8.62Q or later	Version 8.68W or later
GX Configurator-AD	SW2D5C-QADU-E	Version 2.05F or later[*1]	Version 2.05F or later[*2]	Version 2.05F or later[*3]
GX Configurator-DA	SW2D5C-QDAU-E	Version 2.06G or later[*1]	Version 2.06G or later[*2]	Version 2.06G or later[*3]
GX Configurator-SC	SW2D5C-QSCU-E	Version 2.12N or later[*1]	Version 2.12N or later[*2]	Version 2.17T or later[*3]
GX Configurator-CT	SW0D5C-QCTU-E	Version 1.25AB or later[*1]	Version 1.25AB or later[*2]	Version 1.25AB or later[*3]
GX Configurator-TI	SW1D5C-QTIU-E	Version 1.24AA or later[*1]	Version 1.24AA or later[*2]	Version 1.24AA or later[*3]
GX Configurator-TC	SW0D5C-QTCU-E	Version 1.23Z or later[*1]	Version 1.23Z or later[*2]	Version 1.23Z or later[*3]
GX Configurator-FL	SW0D5C-QFLU-E	Version 1.23Z or later[*1]	Version 1.23Z or later[*2]	Version 1.23Z or later[*3]
GX Configurator-QP	SW2D5C-QD75P-E	Version 2.24A or later	Version 2.28E or later	Version 2.29F or later[*4]
GX Configurator-PT	SW1D5C-QPTU-E	Version 1.23Z or later[*1]	Version 1.23Z or later[*2]	Version 1.23Z or later[*3]
GX Configurator-AS	SW1D5C-QASU-E	Version 1.21X or later[*1]	Version 1.21X or later[*2]	Version 1.21X or later[*3]
GX Configurator-MB	SW1D5C-QMBU-E	Version 1.08J or later[*1]	Version 1.08J or later[*2]	Version 1.08J or later[*3]
GX Configurator-DN	SW1D5C-QDNU-E	Version 1.23Z or later[*1]	Version 1.23Z or later[*2]	Version 1.24AA or later[*3]
MX Component	SW3D5C-ACT-E	Version 3.09K or later[*1]	Not available	Not available

* 1: The software can be used by installing GX Developer Version 8.48A or later.
* 2: The software can be used by installing GX Developer Version 8.62Q or later.
* 3: The software can be used by installing GX Developer Version 8.68W or later.
* 4: The software can be used only by connecting USB.

(4) Software not supported in the Universal model QCPU

The following table shows software not supported in the Universal model QCPU.

TableApp.63 Software not supported in the Universal model QCPU

Product	Model
GX Simulator	SW□D5C-LLT-E
GX Explorer	SW□D5C-EXP-E
GX Converter	SW□D5C-CNVW-E

Appendix 5.3 INSTRUCTIONS

Appendix 5.3.1 Instructions not Supported in Universal Model QCPU and Alternative Methods

The Universal model QCPU does not support instructions listed in the TableApp.64 and TableApp.65. Instructions need to be replaced using alternative methods described in the tables. (As for instructions not listed in the tables below, replacement is not required.)

TableApp.64 Instructions not supported in the Universal model QCPU and alternative methods

Symbol	Instruction	Alternative method	Reference
IX	Index modification of entire ladder	Instructions can be replaced using a replacement program.	Appendix 5.3 (1)
IXEND			
IXDEV	Modification value specification in index modification of entire ladder	Change the program so that the device offset values specified by the IXSET instruction are directly set to the index modification table using the MOV instruction.	Appendix 5.3 (2)
IXSET			
PR	Print ASCII code instruction	• It is recommended to use GOT as an ASCII code display device. ASCII codes stored in devices are directly displayed as characters on GOT. • Instructions can be replaced using a replacement program.	Appendix 5.3 (3)
PRC	Print comment instruction	• It is recommended to use GOT as an ASCII code display device.Device comments can be displayed on GOT. • Comment data can be output to a display device in the replacement program of the PR instruction after reading data using the reading device comment data instruction (COMRD(P)).	
CHKST	Specific format failure check instruction	Instructions can be replaced using a replacement program.	Appendix 5.3 (4)
CHK			
CHKCIR	Format change instruction for CHK instruction	Failure detection ladder patens can be changed in a replacement program.	
CHKEND			
PLOW	Program low-speed execution registration instruction	• Use the PSCAN instruction instead of this instruction when low-speed execution type programs are replaced with scan execution type programs. • No instruction can be used if low-speed execution type programs are replaced with fixed scan execution type programs.	---
PCHK	Program execution status check instruction	Check a program execution status on the Program monitor list screen of GX Developer. For details, refer to Section 6.13.1 in this manual.	---
KEY	Numerical key input instruction	• It is recommended to use GOT as a numeral input device. • Instructions can be replaced using a replacement program.	Appendix 5.3 (5)
PLOADP	Load program from memory card	Store all programs to be executed in the program memory. The Universal model QCPU can neither add programs to the program memory nor change them with other programs during RUN.If the capacity of the program memory is not enough, store parameters, device comments, and device initial values in the program memory into the standard ROM or memory card instead.	---
PUNLOADP	Unload program from memory card		
PSWAPP	Load + Unload		

TableApp.65 SFC control instructions not supported in the Universal model QCPU and alternative methods

Symbol	Instruction	Alternative method
LD TRn	Forced transition check instruction	When the programmable controller type is changed, these instructions are converted into SM1255.Modify programs as needed.
AND TRn		
OR TRn		
LDI TRn		
ANDI TRn		
ORI TRn		
LD BLm TRn		
AND BLm TRn		
OR BLm TRn		
LDI BLm TRn		
ANDI BLm TRn		
ORI BLm TRn		
SCHG(D)	Active step change instruction	Refer to Appendix 3 "Restrictions and alternative method of the Basic model QCPU and Universal model QCPU" in the "QCPU (Q Mode)/QnACPU Programming Manual (SFC)".
SET TRn	Transition control instruction	Refer to Appendix 3 "Restrictions and alternative method of the Basic model QCPU and Universal model QCPU" in the "QCPU (Q Mode)/QnACPU Programming Manual (SFC)".
SET BLm TRn		
RST TRn		
RST BLm TRn		
BRSET(S)	Block switching instruction	When the programmable controller type is changed, these instructions are converted into SM1255.Modify programs as needed.

Appendix 5.3.2 Replacing Programs Using Multiple CPU Transmission Dedicated Instructions

(1) Replacing the High Performance model QCPU with the Universal model QCPU (except Q02UCPU)

TableApp.66 shows instructions need to be replaced and corresponding alternative instructions. For the specifications of each instruction, refer to the manuals of Motion CPU.

TableApp.66 Instructions need to be replaced

Symbol	Instruction description	Symbol of alternative instruction
S(P).DDWR	Write other CPU device data into host CPU	D(P).DDWR
S(P).DDRD	Read other CPU device data into host CPU	D(P).DDRD
S(P).SFCS	Request of motion SFC program startup	D(P).SFCS
S(P).SVST	Request of servo program startup	D(P).SVST
S(P).CHGA	Current value change of halted axis/synchronized encoder/cam axis	D(P).CHGA
S(P).CHGV	Axis speed change during positioning and JOG operation	D(P).CHGV
S(P).CHGT	Torque control value change during operation and suspension in real mode	D(P).CHGT
S(P).GINT	Request of other CPU interrupt program startup	D(P).GINT

(2) Replacing the High Performance model QCPU with the Q02UCPU

The Q02UCPU supports the same multiple CPU transmission dedicated instructions used in the High Performance model QCPU. Replacing instructions is not necessary. The Q02UCPU does not support alternative instructions listed in TableApp.66.

9 Parameters
10 Device Explanation
11 CPU Module Processing Time
12 Procedure for Writing Program to CPU Module
Appendices
Index

Appendix 5.3.3 Program Replacement Examples

This section shows program replacement examples for the instructions of which replacement programs are available in Appendix 5.3. (If instructions listed in Appendix 5.3 are not used in the program, it is not necessary to be conscious of the description in this section.)

(1) Replacement example of the IX and IXEND instructions

A replacement example of program using the IX and IXEND instructions is shown below.

Since index registers are saved using the ZPUSH instruction, a 23-word index register save area is required.

(a) Example of device assignment

TableApp.67 Example of device assignment

Before replacement	
Application	Device
Index modification table	D100 to 115

After replacement	
Application	Device
Index modification table	D100 to D115
Index register save area	D200 to D222

If the device numbers in the example above are used for other applications, assign unused device numbers instead.

(b) Program before replacement

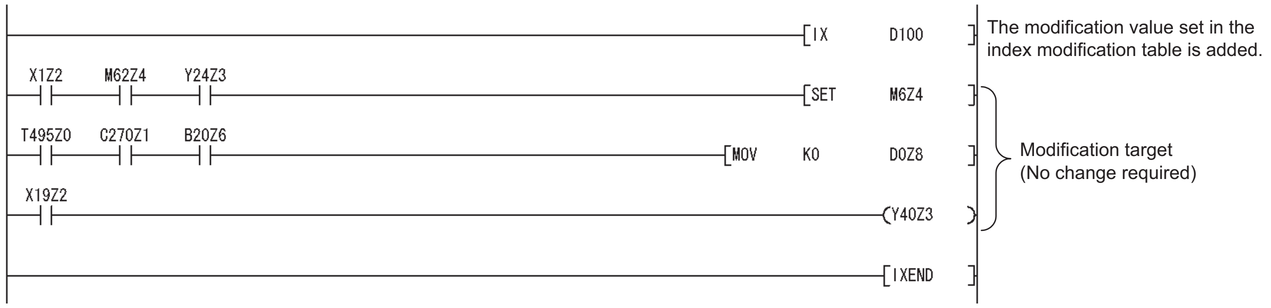


Figure App.1 Sample program

(c) Program after replacement

- Replace the IX instruction with the ZPUSH instruction and the processing for setting the contents of index modification table to index registers.
- Replace the IXEND instruction with the ZPOP instruction.

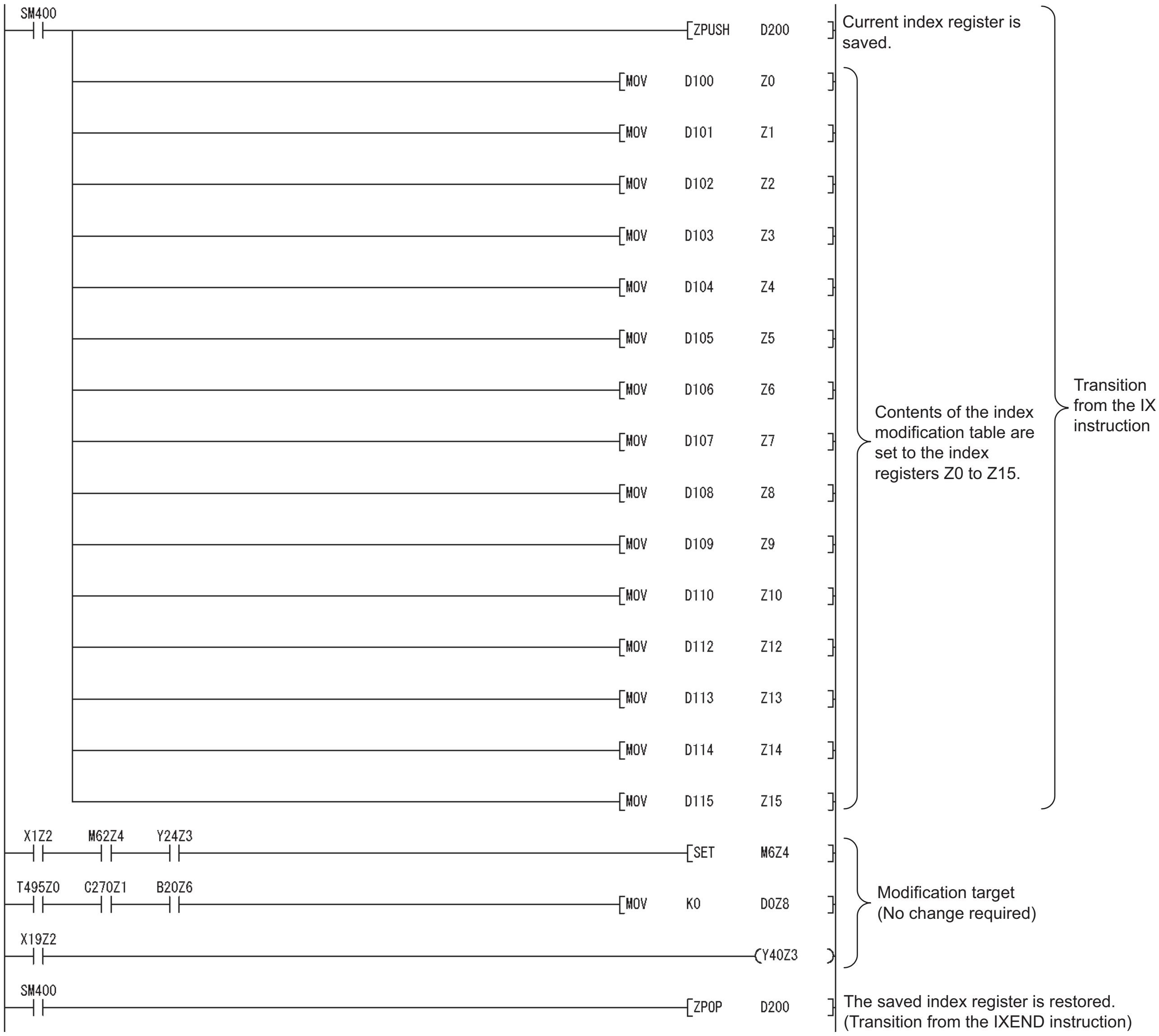

SM400
ZPUSH D200
MOV D100 Z0
MOV D101 Z1
MOV D102 Z2
MOV D103 Z3
MOV D104 Z4
MOV D105 Z5
MOV D106 Z6
MOV D107 Z7
MOV D108 Z8
MOV D109 Z9
MOV D110 Z10
MOV D112 Z12
MOV D113 Z13
MOV D114 Z14
MOV D115 Z15
X1Z2
M62Z4
Y24Z3
SET M6Z4
T495Z0
C270Z1
B20Z6
MOV K0 D0Z8
X19Z2
Y40Z3
SM400
ZPOP D200
Current index register is saved.
Contents of the index modification table are set to the index registers Z0 to Z15.
Transition from the IX instruction
Modification target (No change required)
The saved index register is restored. (Transition from the IXEND instruction)

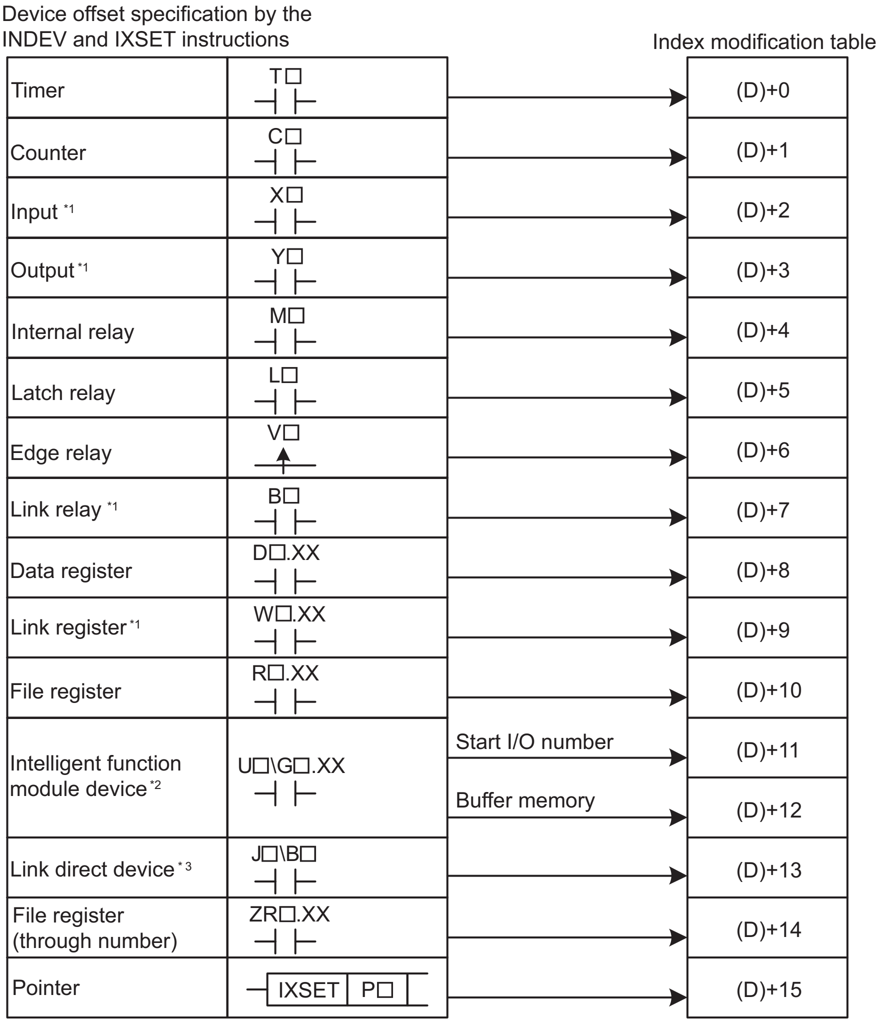
Figure App.2 Sample program

(2) Replacement example of the IXDEV and IXSET instructions

A replacement example of program using the IXDEV and IXSET instructions is shown below.

Change the program so that the device offset value specified by the contacts between the IXDEV and the IXSET instructions are directly set to the index modification table using the MOV instruction.

For the devices whose device offset value is not specified by the IXDEV and IXSET instructions, set the device offset value to 0 in the program after replacement. The following shows how the device offset value is set in the program before and after replacement.



Device offset specification by the INDEV and IXSET instructions			Index modification table
Timer	T□ ┤├	→	(D)+0
Counter	C□ ┤├	→	(D)+1
Input *1	X□ ┤├	→	(D)+2
Output *1	Y□ ┤├	→	(D)+3
Internal relay	M□ ┤├	→	(D)+4
Latch relay	L□ ┤├	→	(D)+5
Edge relay	V□ ─↑─	→	(D)+6
Link relay *1	B□ ┤├	→	(D)+7
Data register	D□.XX ┤├	→	(D)+8
Link register *1	W□.XX ┤├	→	(D)+9
File register	R□.XX ┤├	→	(D)+10
Intelligent function module device *2	U□\G□.XX ┤├	Start I/O number →	(D)+11
		Buffer memory →	(D)+12
Link direct device *3	J□\B□ ┤├	→	(D)+13
File register (through number)	ZR□.XX ┤├	→	(D)+14
Pointer	─[IXSET P□]	→	(D)+15

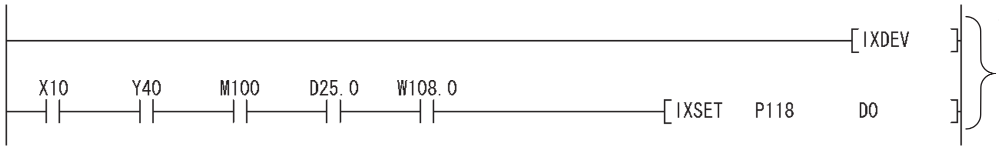
Figure App.3 The device offset value, the index modification table

* 1: Device numbers are expressed in hexadecimal. Use hexadecimal constants (H□) when setting values in the index modification table.

* 2: Start I/O numbers (U□) are expressed in hexadecimal. Use hexadecimal constants (H□) when setting values in the index modification table.

* 3: Devices B, W, X or Y can be specified following J\.□Set device numbers for B, W, X, and Y as device offset values of each device in the index modification table.
For example, if "J10\Y220" is specified by the IXDEV and IXSET instructions, set "K10" in (D)+13 and "H220" in (D)+3 in the replacement program. ((D) indicates the start device in the index modification table.)

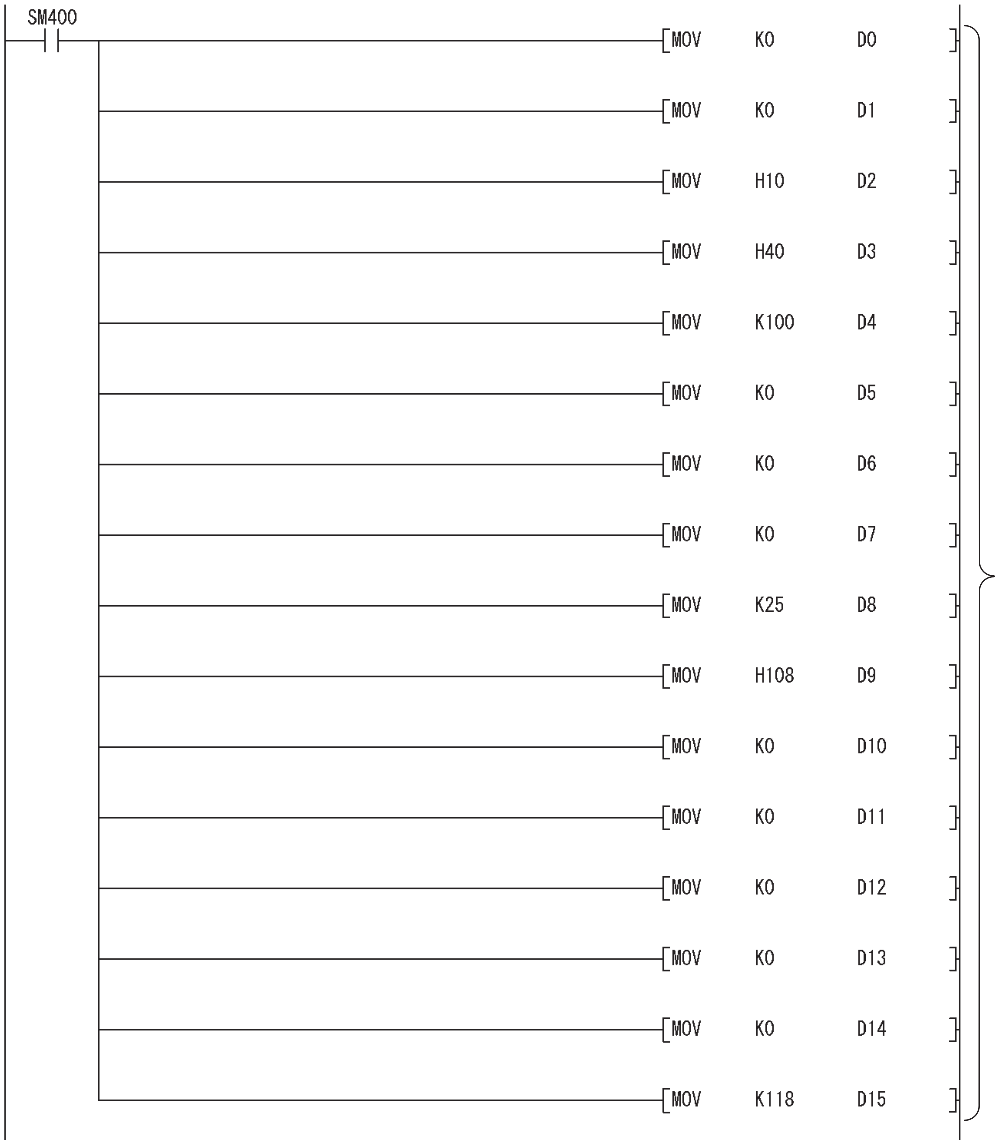
(a) Program before replacement



The device offset values for input (X), output (Y), internal relay (M), data register (D), link register (W), and pointer (P) are set to the index modification table starting from D0.

Figure App.4 Sample program

(b) Program after replacement



The device offset values specified by the IXDEV and IXSET instructions are set to the index modification table starting from D0.

Figure App.5 Sample program

(3) Replacement example of the PR instruction

A replacement example of program using the PR instruction is shown below.
The number of output characters can be switched by the ON/OFF status of SM701.

(a) Example of device assignment

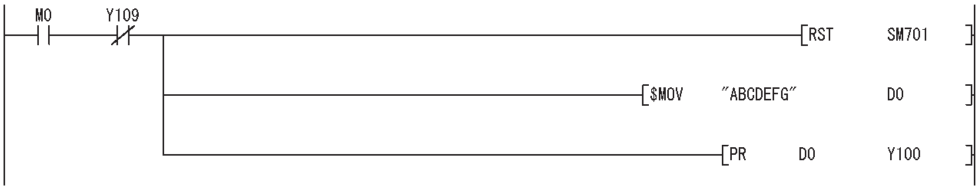
TableApp.68 Example of device assignment

Before replacement	
Application	**Device**
Output string	D0 to D3
ASCII code output signal	Y100 to Y107
Strobe signal	Y108
In-execution flag	Y109

After replacement	
Application	**Device**
Output string	D0 to D3
ASCII code output signal	Y100 to Y107
Strobe signal	Y108
In-execution flag	Y109
Output string storage address (BIN32)	D20 to D21
Output string storage address (BIN32)(Used for sub-routine programs and interrupt programs)	D200 to D201
Number of output characters	D202
Output module start Y number	D203
Character extraction position	D204
Number of extracted characters	D205
String output status value	D206
Result of string extraction by the MIDR instruction	D207
String output in-execution flag	M200
For index modification	Z0

If the device numbers in the example above are used for other applications, assign unused device numbers instead.

(b) Program before replacement



The number of output strings is set to variable. (Output untill ASCII code 00H appears.)

The strings stored in D0 or later are output to Y100 to Y108.

Figure App.6 Sample program

(c) Program after replacement

In the sequence program after replacement, three programs are required as shown below.

A sample program can be downloaded from the MELFANSweb.

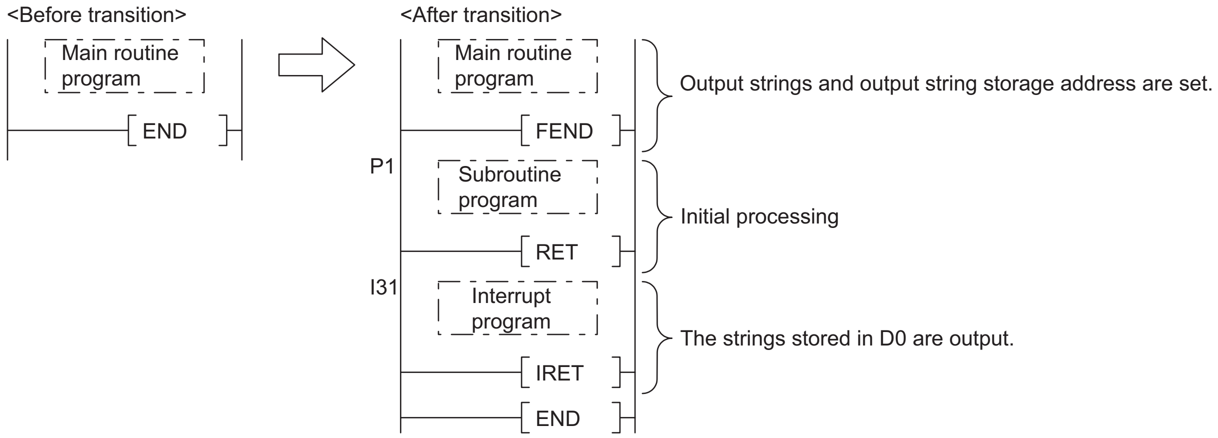


Figure App.7 Program execution

1) Main routine program

- Replace the PR instruction with the CALL instruction so that a subroutine program is called.
- Output string storage device ("D0" in the program below) cannot be specified directly with the CALL instruction.
- Use the ADRSET instruction to acquire the indirect address for the CALL instruction.
- Y device ("Y100" in the program before replacement shown in (b)) cannot be specified directly as output Y number with the CALL instruction. Specify the output Y number in integer.
- An interrupt program is used to output character codes via the output module. Enable the execution of interrupt program using the EI instruction.

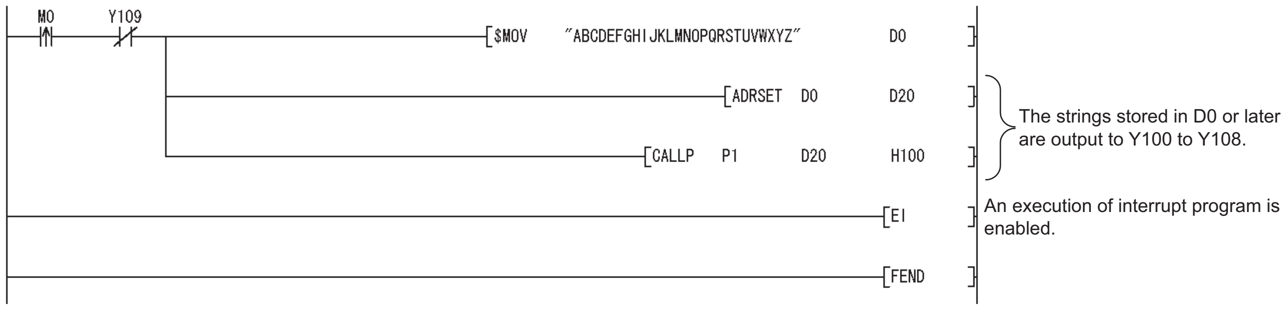
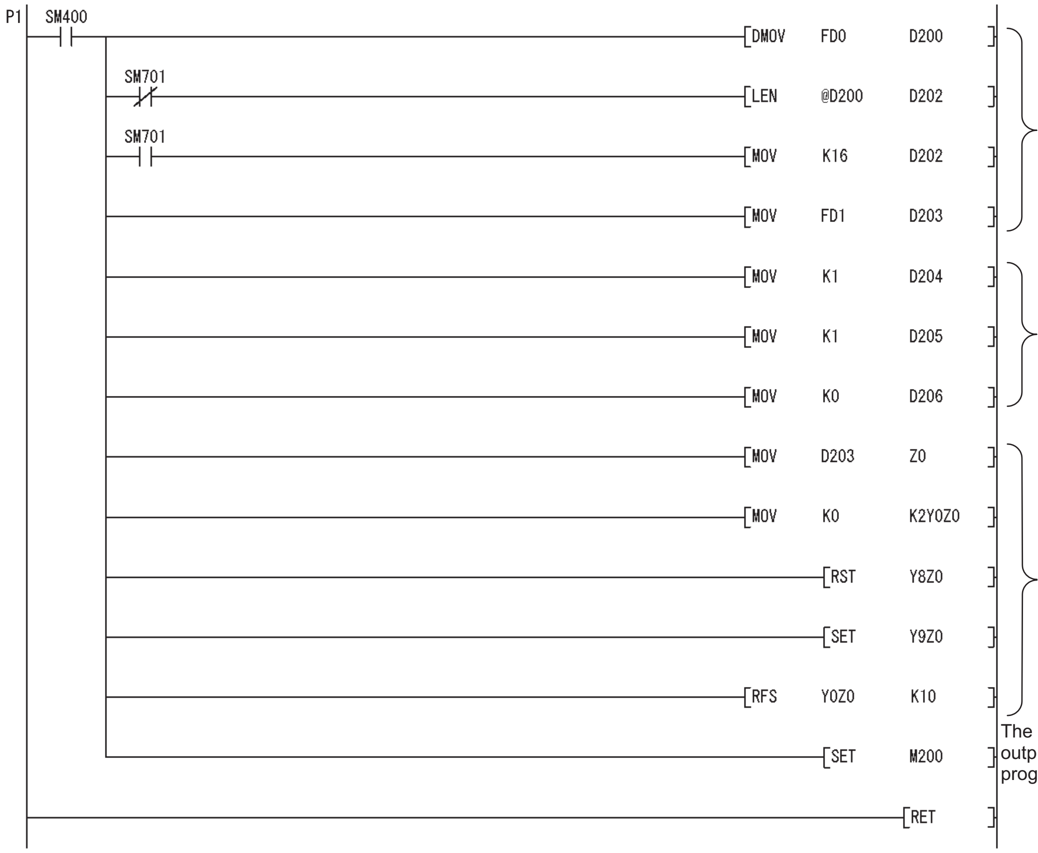

M0
Y109
$MOV "ABCDEFGHIJKLMNOPQRSTUVWXYZ" D0
ADRSET D0 D20
CALLP P1 D20 H100
EI
FEND
The strings stored in D0 or later are output to Y100 to Y108.
An execution of interrupt program is enabled.

Figure App.8 Sample program

2) **Subroutine program**

- In the subroutine program, the data for outputting ASCII codes using a fixed scan interrupt program (10ms) are set to work devices. Also, the flag for activating the processing in the fixed scan interrupt program is turned ON.
- Specify the following arguments for the subroutine program.

First argument	Output string storage address	(Input)
Second argument	Output module start Y number	(Input)



Data specified by the CALL(P) arguments are saved.
- Output string storage address
- Number of output strings
- Output module start number

Devices used for the string output processing of the interrupt program I31 are initialized.

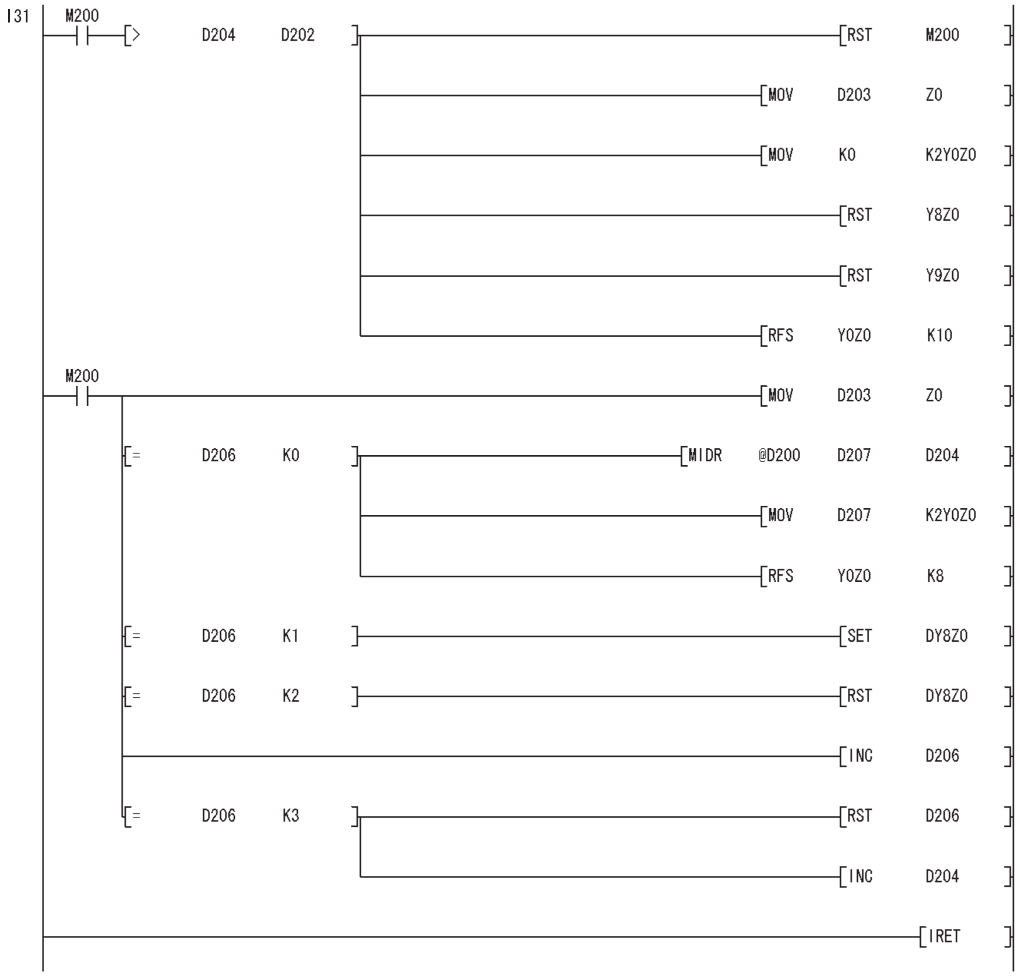
Yn0 to Yn7 (ASCII code), Yn8 (strobe signal), and Yn9 (in-execution flag) are all turned OFF.

The flag to activate the string output processing of the interrupt program is turned ON.

Figure App.9 Sample program

3) **Interrupt program**

The following processing is added to a fixed scan interrupt program (10ms). The fixed scan interrupt program outputs ASCII codes from the output module and controls the strobe signal.



The following signals are all turned OFF when all strings are output.
- Yn0 to Yn7 (ASCII code)
- Yn8 (strobe signal)
- Yn9 (in-execution flag)

Status 0:
One character is extracted from the output string using the MIDR instruction and output to the Y module.The strobe signal is turned OFF for 10ms.

Status 1:
The strobe signal is turned ON for 10ms.

Status 2:
The strobe signal is turned OFF for 10ms.

The status value is incremented by one.

Status 3:
The status value is returned to 0 since the output processing of one character is completed. The next character is extracted.

Figure App.10 Sample program

(4) Replacement example of the CHKST and CHK instructions

A replacement example of program using the CHKST and CHK instructions is shown below.

In the example below, if the replacement program for the CHKST and CHK instructions detects a failure, a failure number (contact number + coil number) is stored in D200 and the annunciator F200 is turned ON.

(a) Example of device assignment

TableApp.69 Example of device assignment

Before replacement		After replacement	
Application	Device	Application	Device
Advance end detection sensor input 1	X100	Advance end detection sensor input 1	X100
Retract end detection sensor input 1	X101	Retract end detection sensor input 1	X101
Advance end detection sensor input 2	X102	Advance end detection sensor input 2	X102
Retract end detection sensor input 2	X103	Retract end detection sensor input 2	X103
Advance end detection sensor input 3	X104	Advance end detection sensor input 3	X104
Retract end detection sensor input 3	X105	Retract end detection sensor input 3	X105
Advance end detection sensor input 4	X106	Advance end detection sensor input 4	X106
Retract end detection sensor input 4	X107	Retract end detection sensor input 4	X107
Failure detection output 1	Y100	Failure detection output 1	Y100
Failure detection output 2	Y102	Failure detection output 2	Y102
Failure detection output 3	Y104	Failure detection output 3	Y104
Failure detection output 4	Y106	Failure detection output 4	Y106
		Coil number (failure type detected)	D100
		Contact number	D101
		Failure number	D200
		Failure detection display	F200
		For index modification	Z0

If the device numbers in the example above are used for other applications, assign unused device numbers instead.

When the advance end detection sensor input performs a failure detection of Xn, assign device numbers for the retract end detection sensor input and the failure detection output as described below.

Advance end detection sensor input	Xn
Retract end detection sensor input	Xn+1
Failure detection output	Yn

(b) **Program before replacement**

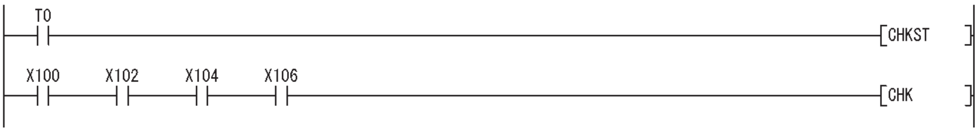


Figure App.11 Sample program

(c) **Program after replacement**

In the sequence program after replacement, two programs are required as shown below.

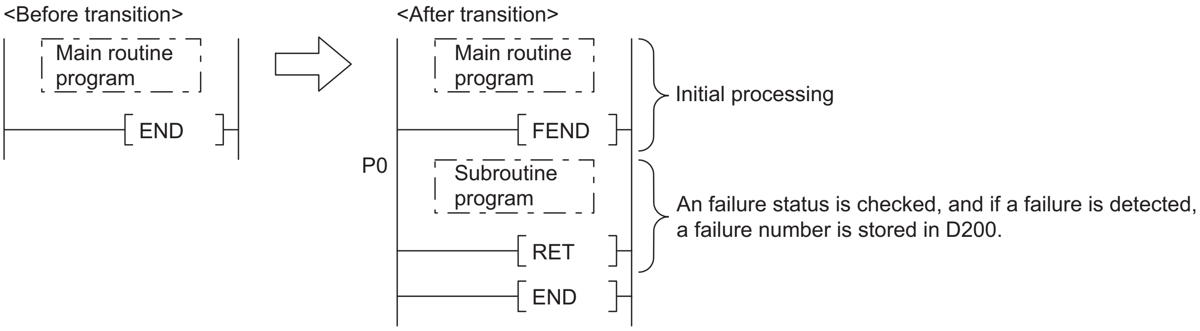


Figure App.12 Program execution

1) **Main routine program**

- Replace the CHKST and CHK instructions with the CALL instructions so that a subroutine program is called.
- One CALL instruction is required for each device specified as check condition in front of the CHK instruction. (In the program before replacement shown in (b), four CALL instructions shall be added since there are four check conditions in front of the CHK instruction.)
- Device number and contact number of X devices (check condition) are specified in each CALL instruction.
- Contact number is used to display failure number when a failure is detected.

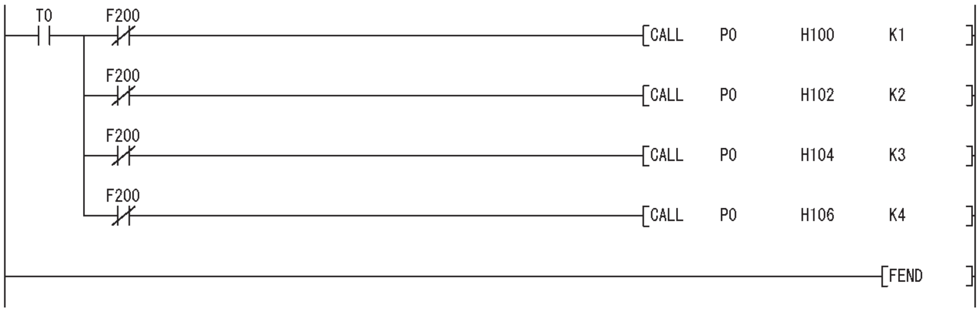
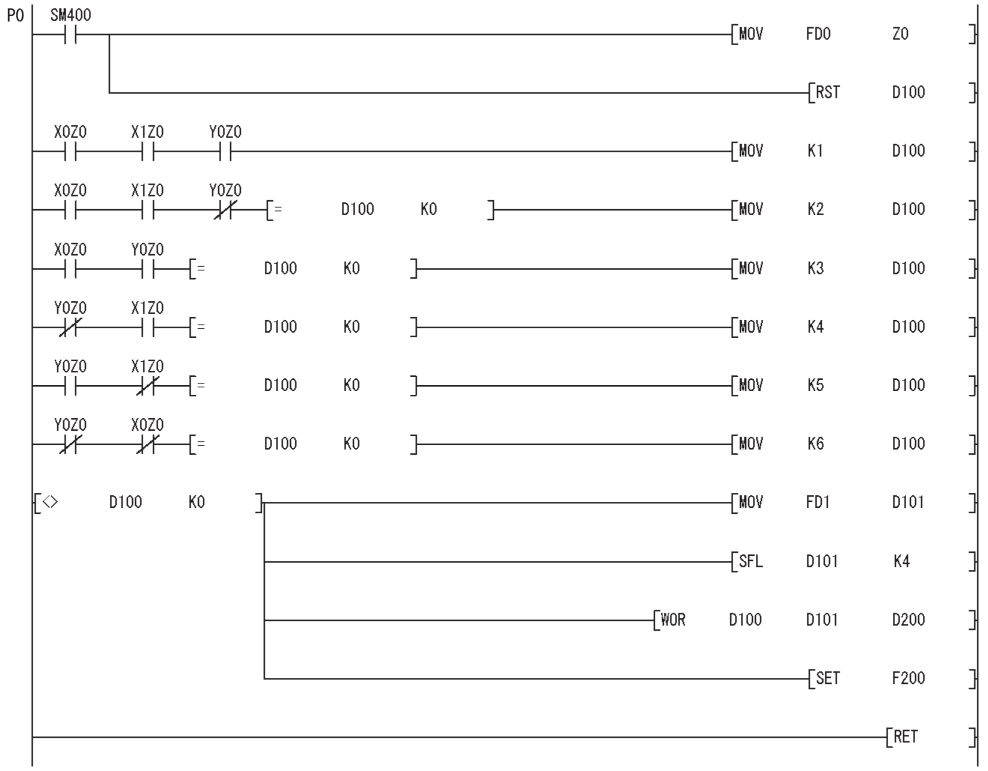


Figure App.13 Sample program

2) **Subroutine program**

- In the subroutine program, an failure status is checked using a failure detection ladder pattern.
  If a failure is detected, a failure number is stored in D200 and the annunciator F200 is turned ON.
- Specify the following arguments for the subroutine program.

First argument	Device number of X device targeted for failure check	(Input)
Second argument	Contact number of X device targeted for failure check	(Input)



<Failure detection target>
If a failure is detected, the coil number corresponding to the failure type is set to D100.

If a failure is detected, a failure number is created by combining the coil number corresponding to the failure type and the contact number.
The annunciator is turned ON.

Figure App.14 Sample program

(d) Replacement method when failure detection ladder patterns are changed by the CHKCIR and CHKEND instructions

Failure detection ladder patterns can be changed in the subroutine program described in (C).

(5) Replacement example of the KEY instruction

A replacement example of program using the KEY instruction is shown below.

(a) Example of device assignment

TableApp.70 Example of device assignment

Before replacement	
Application	**Device**
Numeric input execution instruction	M0
Input complete flag	M1
Input data area	D200 to D203
ASCII code input signal	X100 to X107
Strobe signal	X108

After replacement	
Application	**Device**
Numeric input execution instruction	M0
Input complete flag	M1
Input data area	D200 to D203
ASCII code input signal	X100 to X107
Strobe signal	X108
Input data area address (BIN32)	D210 to D211
(Input data area + 0) address (BIN32)	D212 to D213
(Input data area + 1) address (BIN32)	D214 to D215
(Input data area + 2) address (BIN32)	D216 to D217
For shifting input data	D218
For converting input data	D219 to D220

If the device numbers in the example above are used for other applications, assign unused device numbers instead.

(b) Program before replacement

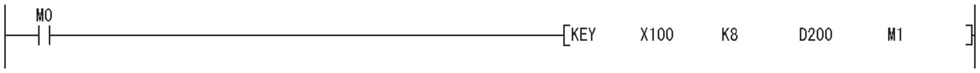


Figure App.15 Sample program

(c) Program after replacement

In the sequence program after replacement, two programs are required as shown below.

A sample program can be downloaded from the MELFANSweb.

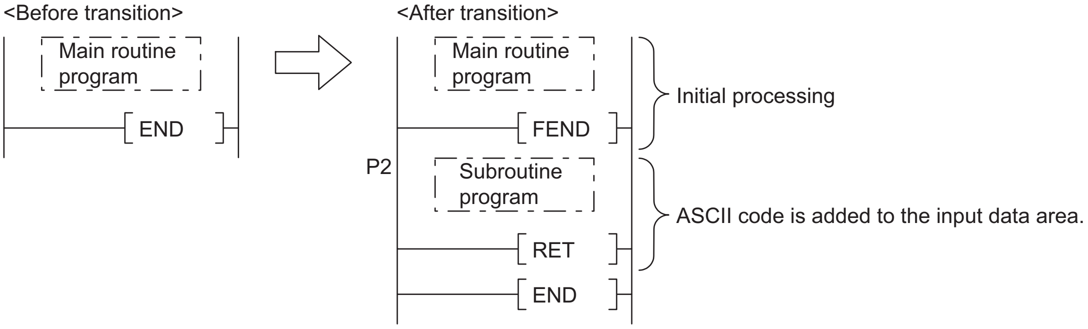


Figure App.16 Program execution

1) Main routing program

- Set "0" in the input data area at the rising edge of the execution instruction ("M0" in the program below) and initialize the program.
- Execute the CALL instruction at every rising edge of the strobe signal ("X108" in the program below) so that a subroutine program is called.
- In the subroutine program, input codes are added to the input data area and the completion status is checked.
- Transfer the following data to the subroutine program at the execution of the CALL instruction.
- ASCII code input value from the input module (Xn0 to Xn7)
- Number of digits to be input.
- Indirect address of the input data area (Use the ADRSET instruction to acquire the indirect address for the input data area.)
- Bit devices to be turned ON when input is completed.

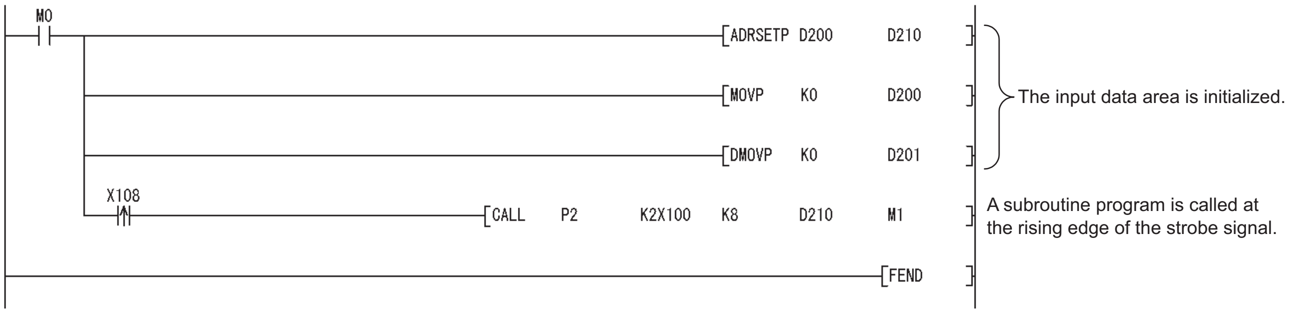
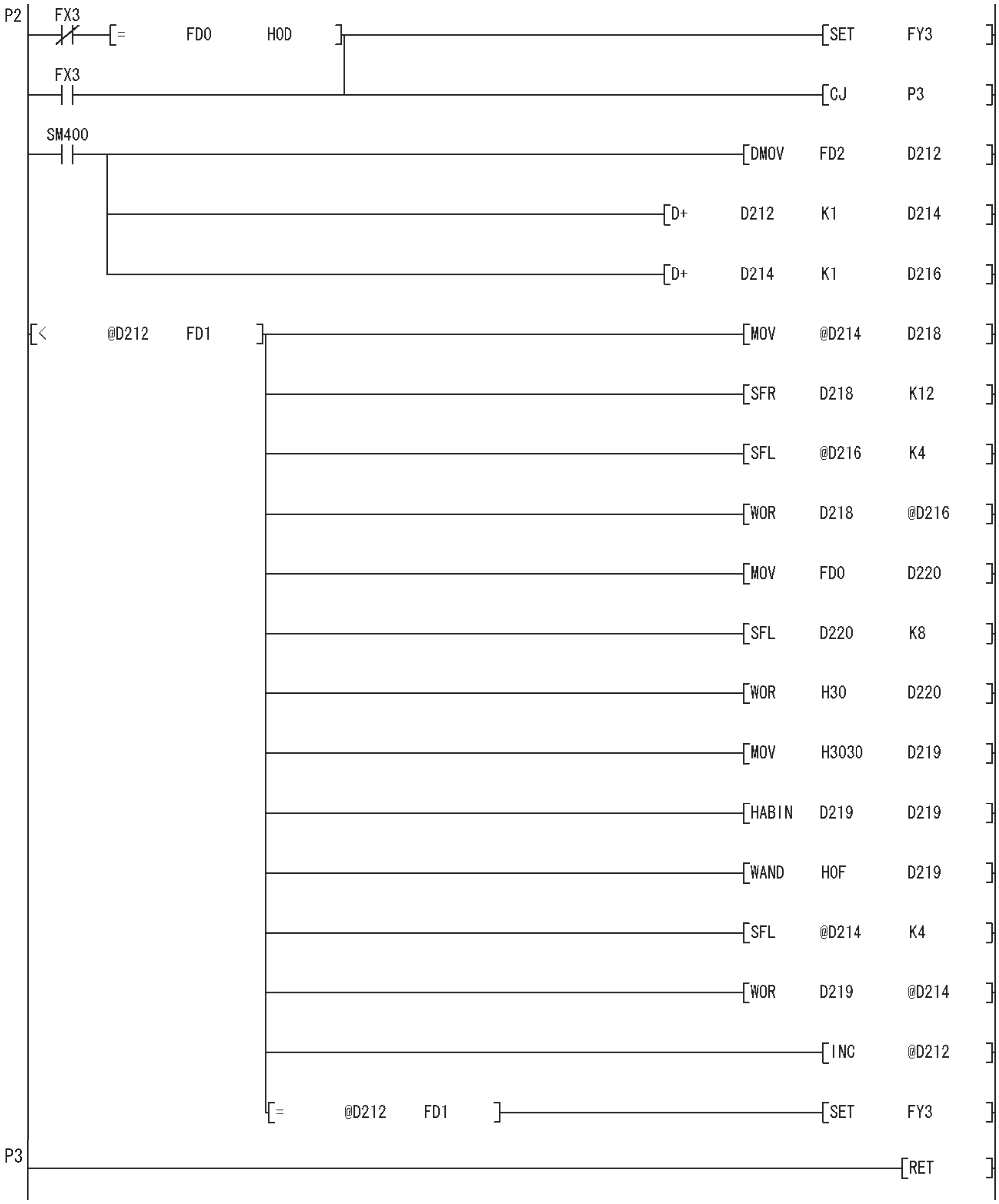

M0
ADRSETP D200 D210
MOVP K0 D200
DMOVP K0 D201
The input data area is initialized.
X108
CALL P2 K2X100 K8 D210 M1
A subroutine program is called at the rising edge of the strobe signal.
FEND

Figure App.17 Sample program

2) **Subroutine program**

- In the subroutine program, ASCII codes specified by an argument are added to the input data area and the completion status is checked.
- Specify the following arguments for the subroutine program.

First argument	ASCII code input from the input module (K2Xn)	(Input)
Second argument	Number of digits to be input	(Input)
Third argument	Indirect address of the input data area	(Input)
Fourth argument	Bit device turned ON when input is completed	(Output)



Numeric entry is ended when the at-completion ON signal is ON or 0DH is input.

Addresses of the input data area are saved in the work devices.

The 1st to 4th digit numerals in (input data area +2) are shifted for one digit to the left.

Numeral entered in ASCII code is converted into one numeral in BIN data using the HABIN instruction.

The 5th to 8th digit numerals in (input data area +1) are shifted for one digit to the left and the converted numeral is set to the 8th digit.

The number of digits to be input in (input data area +0) is incremented by one.

The at-completion ON signal is turned ON when the input processing for specified digits is completed.

Figure App.18 Sample program

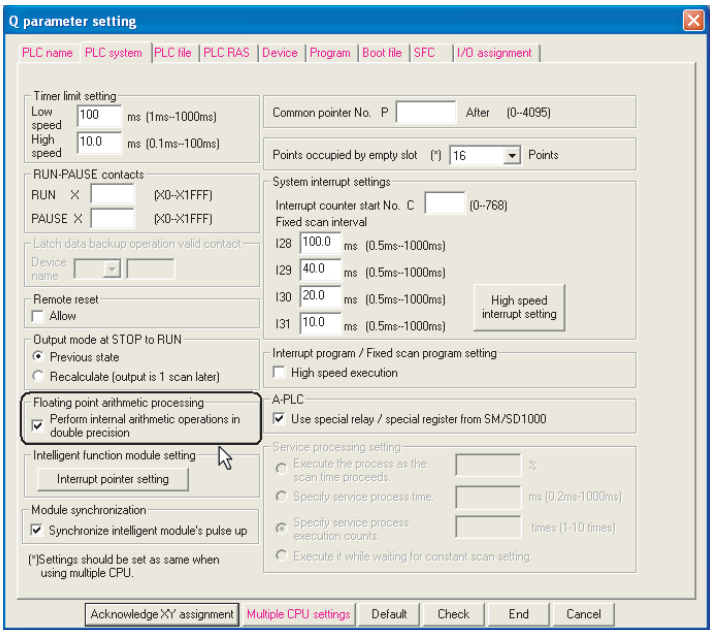
Appendix 5.4 FUNCTIONS

Appendix 5.4.1 Floating-point Operation Instructions

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

The High Performance model QCPU can perform only the single-precision floating-point operation instructions. Note, however, that internal operation processing can be performed in double precision by checking the item on the PLC system setting tab of PLC parameter as shown below. (Default: Checked)



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
I28 100.0 ms (0.5ms--1000ms)
I29 40.0 ms (0.5ms--1000ms)
I30 20.0 ms (0.5ms--1000ms)
I31 10.0 ms (0.5ms--1000ms)

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.

Figure App.19 PLC system setting tab

(b) Universal model QCPU

The Universal model QCPU supports the double-precision floating-point operation instructions.

The operation can be performed either in single precision or double precision depending on the data. Therefore, "Perform internal arithmetic operations in double precision" on the PLC system setting tab of PLC parameter cannot be set in the Universal model QCPU.

Because of this new function, operation results (both in single precision and double precision) slightly differ between the High Performance model QCPU and the Universal model QCPU if "Perform internal arithmetic operations in double precision" is set in the High Performance model QCPU.

If higher accuracy is required in floating-point operations, replace the floating-point operation instructions as described in (4). However, if six or less digits are used as significant digits for the floating-point operation instructions, replacement is not necessary. The single-precision floating-point operation results in the Universal model QCPU can be used as they are in the system. When not replacing instructions, make sure that it does not cause any problems in the actual system.

(2) Floating-point operation instructions for the Universal model QCPU

TableApp.71 lists floating-point operation instructions for the Universal model QCPU. Specifications of the single-precision floating-point operation instructions are compatible with those for the High Performance model QCPU.

TableApp.71 List of floating-point operation instructions supported in the Universal model QCPU

Instruction name		Instruction symbol		Remarks
		Single-precision floating-point data	Double-precision floating-point data	
Comparison	Floating-point data comparison	LDE□	LDED□	□ indicates one of the followings; <>,=,<,>,<=,>=
		ANDE□	ANDED□	
		ORE□	ORED□	
Data transfer	Floating-point data transfer	EMOV(P)	EDMOV(P)	---
Four arithmetic operation	Floating-point data addition	E+(P)	ED+(P)	---
	Floating-point data subtraction	E-(P)	ED-(P)	
	Floating-point data multiplication	E*(P)	ED*(P)	
	Floating-point data division	E/(P)	ED/(P)	
Data conversion	Conversion from BIN 16-bit data to floating-point data	FLT(P)	FLTD(P)	---
	Conversion from BIN 32-bit data to floating-point data	DFLT(P)	DFLTD(P)	
	Conversion from floating-point data to BIN 16-bit data	INT(P)	INTD(P)	
	Conversion from floating-point data to BIN 32-bit data	DINT(P)	DINTD(P)	
	Floating-point sign inversion	ENEG(P)	EDNEG(P)	
Special function	SIN operation	SIN(P)	SIND(P)	---
	COS operation	COS(P)	COSD(P)	
	TAN operation	TAN(P)	TAND(P)	
	SIN-1 operation	ASIN(P)	ASIND(P)	
	COS-1 operation	ACOS(P)	ACOSD(P)	
	TAN-1operation	ATAN(P)	ATAND(P)	
	Conversion from angle to radian	RAD(P)	RADD(P)	
	Conversion from radian to angle	DEG(P)	DEGD(P)	
	Square root	SQR(P)	SQRD(P)	
	Exponential operation	EXP(P)	EXPD(P)	
	Natural logarithm operation	LOG(P)	LOGD(P)	

Floating-point data can be converted mutually between single precision and double precision using instructions in TableApp.72.

TableApp.72 Floating-point data conversion instructions (single precision   double precision)

Instruction name	Instruction symbol
Single precision to double precision conversion	ECON(P)
Double precision to single precision conversion	EDCON(P)

(3) Advantages and disadvantages when using the double-precision floating-point data of the Universal model QCPU

TableApp.73 shows the advantages and disadvantages when performing the double-precision floating-point operation instructions in the Universal model QCPU.
If higher accuracy is required in floating-point operations, it is recommended to replace the instructions with the double-precision floating-point operation instructions.

TableApp.73 Advantages and disadvantages when using the double-precision floating-point operation instructions

Advantage	Disadvantage
The results are more accurate than those of the single-precision floating-point operation instructions.	The instruction processing speed is slower than that of the single-precision floating-point operation instructions. *1 Double-precision floating-operation data use twice as many word device points as single-precision floating-operation data.

* 1: The processing speed of the double-precision floating-point operation instructions in the Universal model QCPU is higher than that of floating-point operation instructions using internal double-precision operations in the High Performance model QCPU.

TableApp.74 shows the comparison between single-precision and double precision floating-point data.

TableApp.74 Comparison between single-precision and double precision floating-point data

Item		Single-precision floating-point data	Double-precision floating-point data
Word point required for data retention		2 words	4 words
Setting range		$-2^{128}<N \leqq -2^{-126}$, 0, $2^{-126} \leqq N<2^{128}$	$-2^{1024}<N \leqq -2^{-1022}$, 0, $2^{-1022} \leqq N<2^{1024}$
Precision (number of bits)	Mantissa part	23 bits	52 bits
	Exponent part	8 bits	11 bits
	Sign part	1 bits	1 bits
Instruction processing speed (Q04UDHCPU/ Q06UDHCPU)(Maximum)	Data comparison (Conductive status)(LDE>= / LDED>=)	5.5 μs	9.0 μs
	Data transfer(EMOV/ EDMOV)	0.019 μs	5.0 μs
	Addition (3 devices)(E+ / ED+)	0.0665 μs	9.2 μs
	SIN operation(SIN/SIND)	5.7 μs	13.8 μs

(4) Method of replacing High Performance model QCPU with Universal model QCPU

(a) Replacing all single-precision floating-point operation instructions with double-precision floating-point operation instructions

Single-precision floating-point data occupy two points of word device per data. On the other hand, four points are required per double-precision floating-point data. Therefore, all device numbers for storing floating-point data need to be reassigned.

Example) Replacing the floating-point operation [A to B+C] (Changing all floating-point data into double precision.)

1) Device assignment

TableApp.75 Device assignment

Before replacement			After replacement		
Application	Device	Data type	Application	Device	Data type
Data A	D0 to D1	Floating-point data (single precision)	Data A	D0 to D3	Floating-point data (double precision)
Data B	D2 to D3		Data B	D4 to D7	
Data C	D4 to D5		Data C	D8 to D11	
Data D	D6 to D7		Data D	D12 to D15	

2) Program before replacement

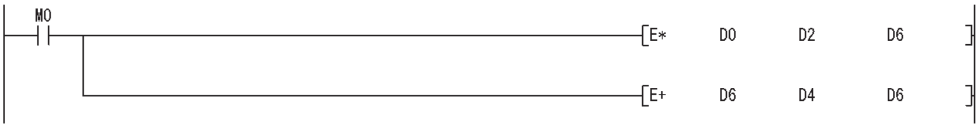
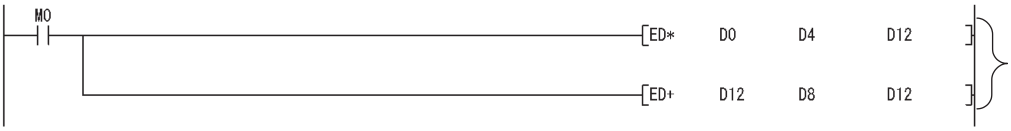


Figure App.20 Sample program

3) Program after replacement



Operation is performed using double-precision floating-point data.

Figure App.21 Sample program

(b) Replacing a part of floating-point operation instructions with double-precision floating-point operation instructions

Only operations require high accuracy are replaced with double-precision floating-point operation instructions. Using the ECON and EDCON instructions, convert floating-point data mutually between single precision and double-precision. The flow of a replacement program is as follows:

- Data required for operations are converted from single precision to double precision using the ECON instruction.
- Operations are performed in double precision using the double-precision floating-point operation instructions.
- Operation results are converted from double precision to single precision using the EDCON instruction.

A program example that floating-point data are converted mutually between single precision and double precision before and after operations is shown below.

Example) Replacing the floating-point operation [A × B+C] (Using the ECON and EDCON instructions)

1) Device assignment

TableApp.76 Device assignment

Before replacement		
Application	Device	Data type
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Operation result	D6 to D7	

After replacement		
Application	Device	Data type
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Operation result	D6 to D7	
Data A (D)	D10 to D13	Floating-point data (double precision)
Data B (D)	D14 to D17	
Data C (D)	D18 to D21	
Operation result (D)	D22 to D25	

2) Program before replacement

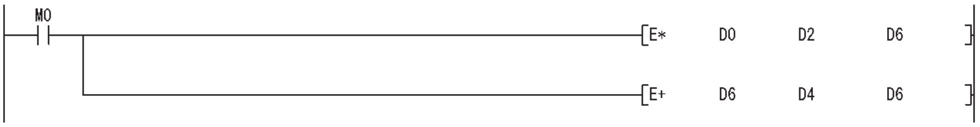
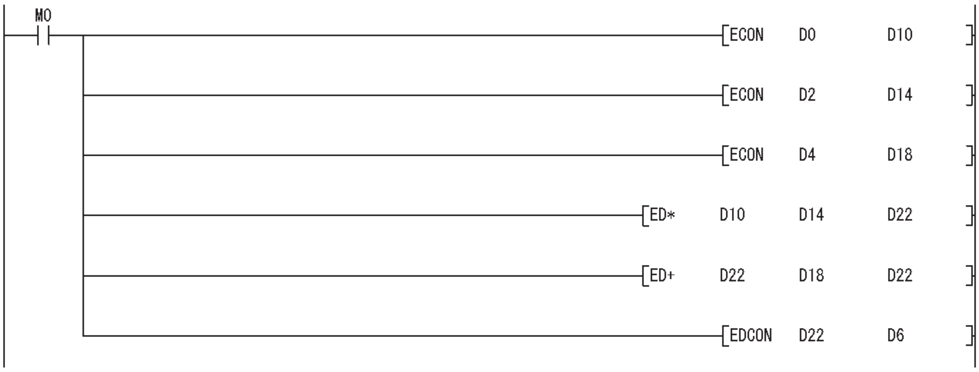


Figure App.22 Sample program

3) Program after replacement



Floating-point data are converted from single precision to double precision.

Operation is performed using double-precision floating-point data.

The floating-point operation result data are converted from double precision to singe precision.

Figure App.23 Sample program

(c) Replacing a part of floating-point operation instructions with double-precision floating-point operation instructions using subroutine programs

The flow of a replacement program described in (b) can be regarded as one subroutine program. Create subroutine programs for each floating-point operation instruction and then replace the original floating-point operation instructions with the CALL(P) instructions so that subroutine programs are called.

With this method, changes in the program are minimized, but the processing for calling subroutine programs increases the scan time. In addition, since conversions from double precision to single precision are performed for each instruction, rounding-off errors generated during operations are larger than those in the replacement program described in (b).

Example) Replacing the floating-point operation [A × B+C] (Using a subroutine program)

1) Device assignment

TableApp.77 Device assignment

Before replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Operation result	D6 to D7	

After replacement		
Application	**Device**	**Data type**
Data A	D0 to D1	Floating-point data (single precision)
Data B	D2 to D3	
Data C	D4 to D5	
Operation result	D6 to D7	
Data A (D)	D900 to D903	Floating-point data (double precision)
Data B (D)	D904 to D907	
Data C (D)	D908 to D911	

2) Program before replacement

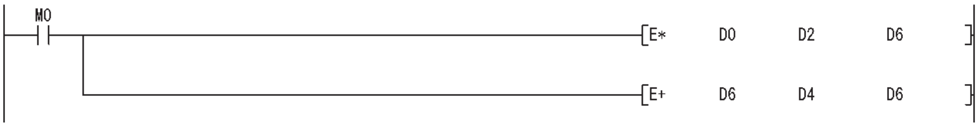
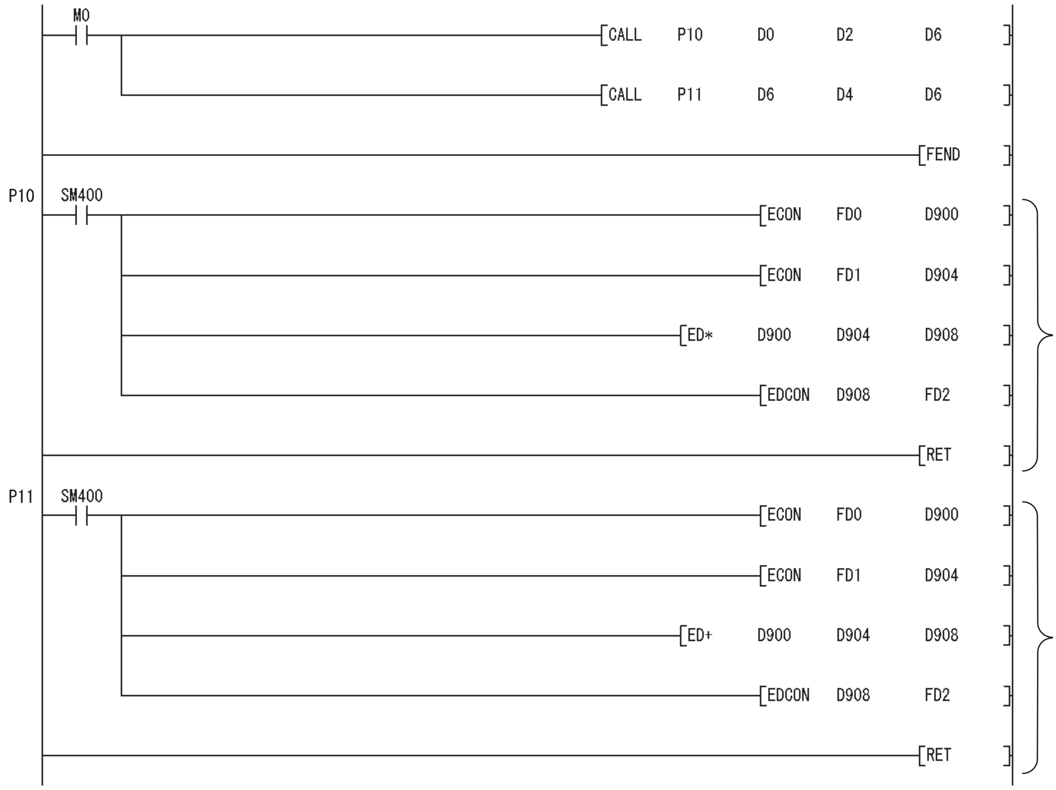


Figure App.24 Sample program

3) Program after replacement



A subroutine program for multiplication using the double-precision floating-point operation instruction

A subroutine program for addition using the double-precision floating-point operation instruction

Figure App.25 Sample program

Appendix 5.4.2 Error Check Processing for Floating-point Data Comparison Instructions

(1) Input data check

An error check processing for floating-point data comparison instructions performed in the Universal model QCPU are enhanced. Input of a "special value" (-0, nonnumeric, unnormalized number, and ±∞) is checked, and if those special values are input, an OPERATION ERROR (error code: 4140) is detected.

When the LDE□, ANDE□, ORE□, LDED□, ANDED□, and/or ORED□ instructions (□indicates one of the followings; =,<>,<,>,<=,>=) are used in the program, an OPERATION ERROR (error code: 4140) might be detected if invalid floating-point data exist. This occurs even when interlocks are provided using the valid data flag (the signal which shows the floating-point validity).

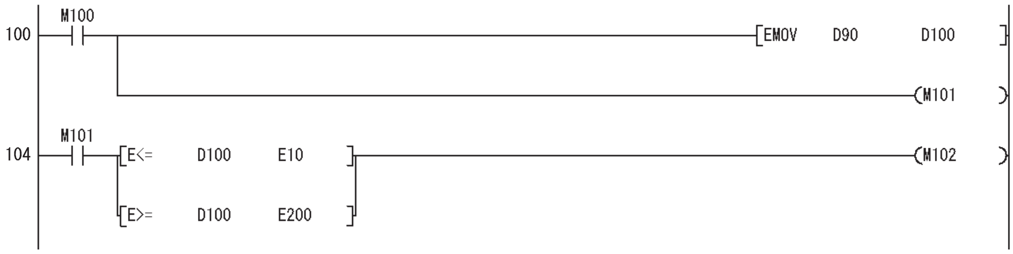
Invalid floating-point data are not stored in the result of operations performed in the Universal model QCPU. Those invalid data are considered to be stored in the following cases:

- The same device is used for storing floating-point data and other data, such as binary values, BCD values, and strings.
  → Use different devices for storing floating-point data and data other than floating-point data
- Floating-point data externally written are invalid.
  → Take measures on the external-source side so that valid data are written.

If an error occurs in the floating-point data comparison instructions, take appropriate measures to remove error causes described above.

Example 1) Detecting an OPERATION ERROR (error code: 4140) in the LDE□ instruction

[Ladder mode]



[List mode]

```
100  LD      M100
101  EMOV    D90     D100
103  OUT     M101
104  LD      M101
105  LDE<=   D100    E10
109  ORE>=   D100    E200
113  ANB
114  OUT     M102
```

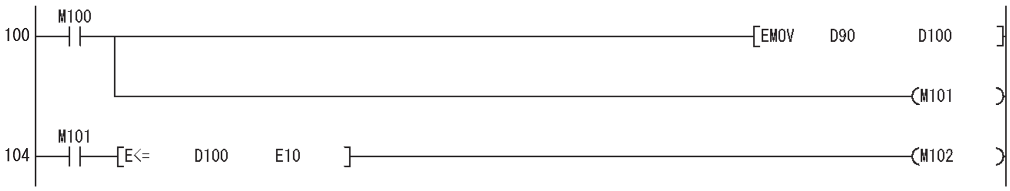
In the ladder block starting from the step 104, the floating-point data comparison instructions of the step 105 and 109 shall not be executed when the M101 (valid data flag) is OFF.

However, the LDE<= instruction of the step 105 and the ORE>= instruction of the step 109 are executed regardless of the execution result of the LD instruction of the step 104 in the program above. Therefore, even when the M101 is OFF, an OPERATION ERROR (error code: 4140) is detected in the LDE<= instruction of the step 105 if a "special value" is stored in D100.

Method of avoiding an OPERATION ERROR occurrence is described in (2).

Example 2) Not detecting an OPERATION ERROR (error code: 4140) in the ANDE□ instruction

[Ladder mode] **[List mode]**

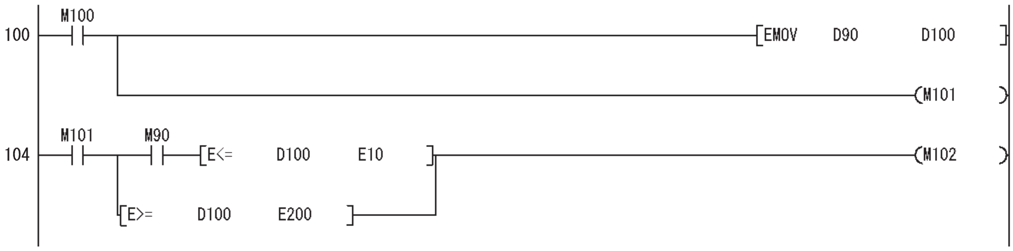


```
100  LD      M100
101  EMOV    D90     D100
103  OUT     M101
104  LD      M101
105  ANDE<=  D100    E10
109  OUT     M102
```

In the ladder block starting from the step 104, the ANDE<= instruction of the step 105 shall not be executed when the M101 (valid data flag) is OFF.
The ANDE<= instruction of the step 105 is not executed when the M101 is OFF in the LD instruction of the step 104 in the program above. Therefore, when the M101 is OFF, an OPERATION ERROR (error code: 4140) is not detected even if a "special value" is stored in D100.

Example 3) Detecting an OPERATION ERROR (error code: 4140) in the ANDE□ instruction

[Ladder mode] **[List mode]**



```
100  LD      M100
101  EMOV    D90     D100
103  OUT     M101
104  LD      M101
105  LD      M90
106  ANDE<=  D100    E10
110  ORE>=   D100    E200
114  ANB
115  OUT     M102
```

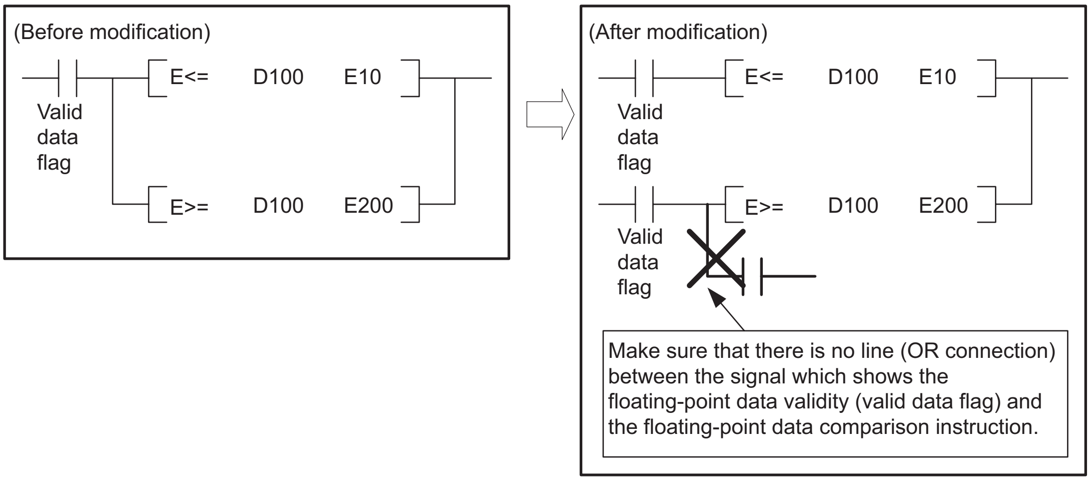
In the ladder block starting from the step 104, the ANDE<= instruction of the step 106 and the OR>= instruction of the step 110 shall not be executed when the M101 (valid data flag) is OFF.
However, if the M90 is ON in the LD instruction of the step 105, the ANDE<= instruction of the step 106 is executed. Therefore, even when the M101 is OFF, an OPERATION ERROR (error code: 4140) is detected in the ANDE<= instruction of the step 106 if the M90 is ON and a "special value" is stored in D100.
Method of avoiding an OPERATION ERROR occurrence is described in (2).

(2) Method of avoiding an occurrence of OPERATION ERROR (error code: 4140) in the floating-point data comparison instructions

As shown in the modification examples below, connect the contacts of valid data flag in series for each floating-point data comparison instruction. (Use AND connection for connecting the contact of the valid data flag and the floating-point data comparison instruction.)

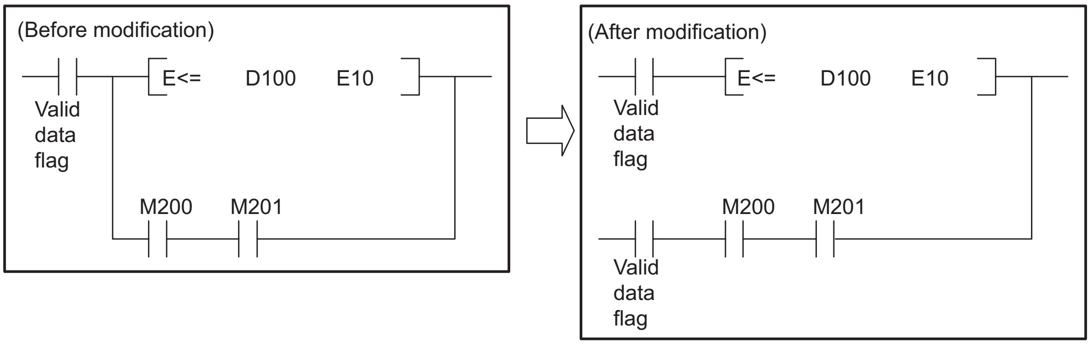
Make sure that there is no line (OR connection) between the valid data flag and the floating-point data comparison instruction.

<Modification example 1>



Make sure that there is no line (OR connection) between the signal which shows the floating-point data validity (valid data flag) and the floating-point data comparison instruction.

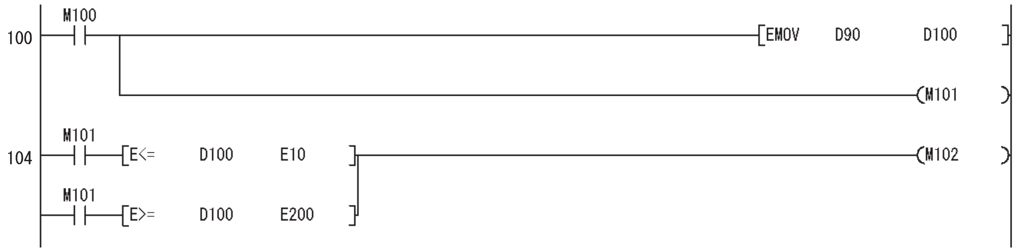
<Modification example 2>



Program examples after modification for Example 1) and 3) in (1) are shown below.

Example 4) Program after modification for Example 1) (An OPERATION ERROR (error code: 4140) is no longer detected.)

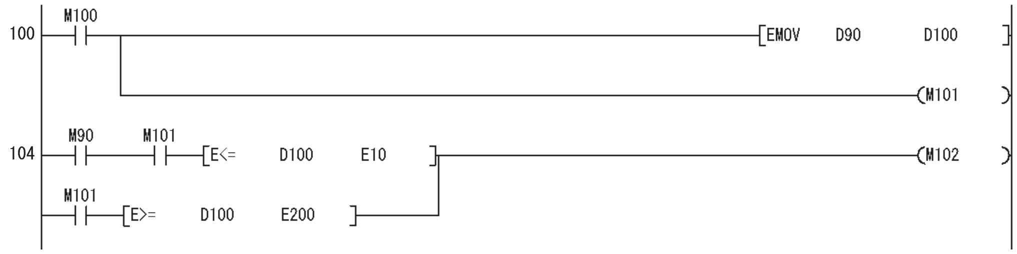
[Ladder mode] **[List mode]**



```
100  LD        M100
101  EMOV      D90      D100
103  OUT       M101
104  LD        M101
105  ANDE<=    D100     E10
109  LD        M101
110  ANDE>=    D100     E200
114  ORB
115  OUT       M102
```

Example 5) Program after modification for Example 3) (An OPERATION ERROR (error code: 4140) is no longer detected.)

[Ladder mode] **[List mode]**



```
100  LD        M100
101  EMOV      D90      D100
103  OUT       M101
104  LD        M90
105  AND       M101
106  ANDE<=    D100     E10
110  LD        M101
111  ANDE>=    D100     E200
115  ORB
116  OUT       M102
```

Appendix 5.4.3 Device Latch Function

(1) Overview

This section explains how to use the latch function *1 of the Universal model QCPU, which has been enhanced compared with the High Performance model QCPU.

* 1: The latch function is a function that allows device data retention even if power is turned OFF or the CPU module is reset.

(2) Methods for the latch function

Device data of the Universal model QCPU can be latched by:

- Large-capacity file registers (R, ZR)
- Writing/reading device data to the standard ROM (SP.DEVST/S(P).DEVLD instructions)
- Specifying internal user devices as latch ranges

(3) Latch function details

(a) Large-capacity file registers (R, ZR)

File registers are the devices whose data can be latched by batteries.
The Universal model QCPU has file registers of larger capacities than the High Performance model QCPU, offering a high-speed file registers processing. If many device points are to be set for latching, it is advisable to use the file registers.

TableApp.78 shows capacities of file registers for each CPU module.

TableApp.78 Capacities of file registers for each CPU module

Model	Q02UCPU	Q03UDCPU, Q03UDECPU	Q04UDHCPU, Q04UDEHCPU	Q06UDHCPU, Q06UDEHCPU	Q13UDHCPU, Q13UDEHCPU	Q26UDHCPU, Q26UDEHCPU
Capacity of file register (R, ZR) in standard RAM *1	64k points	96k points	128k points	384k points	512k points	640k points

* 1: Use of a memory card can increase the number of points.

(b) Writing/reading device data to the standard ROM (SP.DEVST/S(P).DEVLD instructions)

Device data of the Universal model QCPU can be latched using the instructions for writing/reading them to/from the standard ROM (SP.DEVST/S (P).DEVLD).
Utilizing the standard ROM allows data backup without batteries. This method is effective for latching data that will be updated less frequently.

(c) Specifying internal user device as latch ranges

In the same way as for the High Performance model QCPU, latching is available by specifying internal user devices as latch ranges for the Universal model QCPU. The ranges can be set on the Device setting tab of PLC parameter.

Internal user devices that can be latched are as follows:

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

☒POINT

1. If latch ranges of internal user devices are specified in the Universal model QCPU, the processing time is increased in proportion to the device points set for latching. (For example, if 8k points are latched for a latch relay (L), the scan time will be 28.6 µs.)If a shorter scan time is needed, remove unnecessary latch device points to minimize the latch range.
2. Latch range specification of a file register (R, ZR) does not increase the scan time.

(4) How to shorten the scan time

When data to be latched are stored in a file register (R or ZR), the processing time is shorter than that for latching internal user device.

Example) Reducing the number of latch points for a data register (D) from 8k points to 2k points, and using a file register (ZR) instead (When using the Q06UDHCPU)

TableApp.79 Differences between before and after moving latch points of data register (D) to file register (ZR)

Item		Before	After
Latch points for data register (D)		8192 (8k) points	2048 (2k) points (6k points are moved to file register.)
Number of devices in the program	Data register (D) (Latch range)	400	100
	File register (ZR) (Standard RAM)	0	300
Additional scan time		0.41ms	0.13ms *1
Number of steps increased		---	300 steps

[1] Indicates the time added when file register data are stored in the standard RAM.

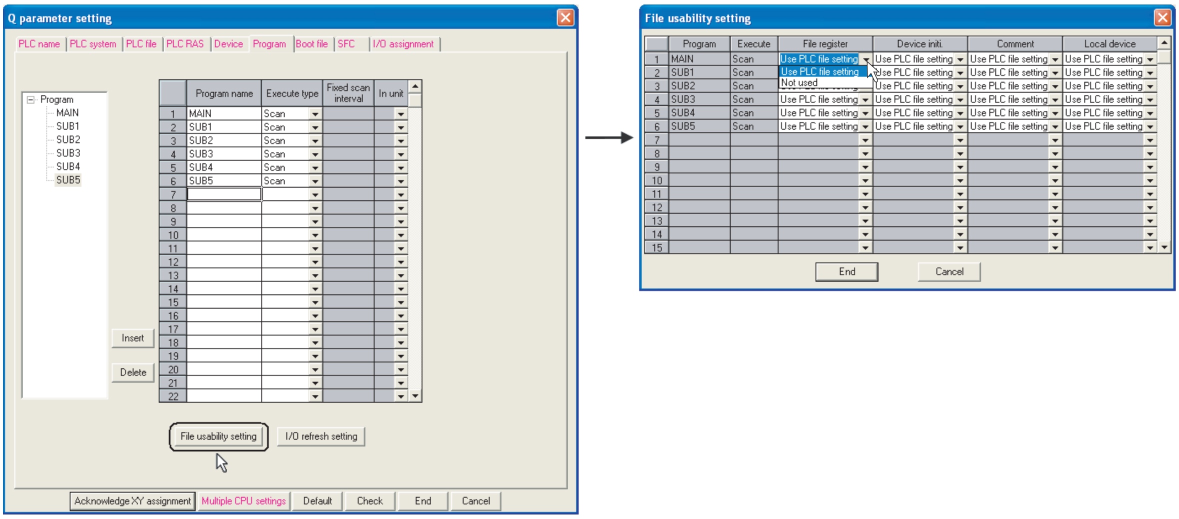
Appendix 5.4.4 File Usability Setting

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

In the High Performance model QCPU, file usability for each program can be set to either "Use PLC file setting" or "Not used" on the screen displayed by clicking the "File usability setting" button on the Program setting tab of PLC parameter.

- File register
- Device initial value
- Comment
- Local device



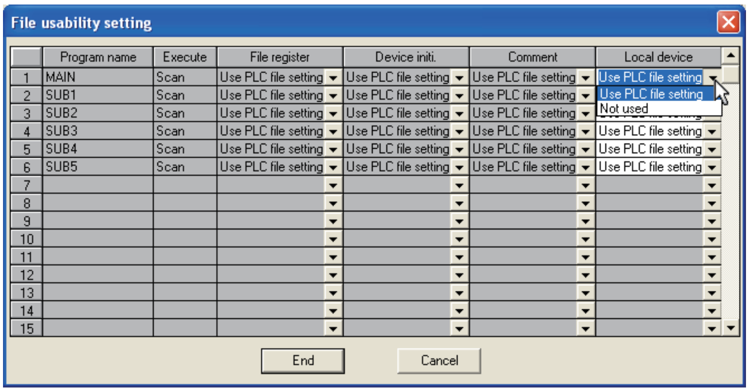
	Program name	Execute type	Fixed scan interval	In unit
1	MAIN	Scan		
2	SUB1	Scan		
3	SUB2	Scan		
4	SUB3	Scan		
5	SUB4	Scan		
6	SUB5	Scan		
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				

	Program	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	SUB1	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
3	SUB2	Scan	Not used	Use PLC file setting	Use PLC file setting	Use PLC file setting
4	SUB3	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
5	SUB4	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
6	SUB5	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
7						
8						
9						
10						
11						
12						
13						
14						
15						

Figure App.26 Program setting tab

(b) Universal model QCPU

In the Universal model QCPU, the following file usability for each program cannot be set on the Program setting tab of PLC parameter. [*1]



	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	SUB1	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
3	SUB2	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Not used
4	SUB3	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
5	SUB4	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
6	SUB5	Scan	Use PLC file setting	Use PLC file setting	Use PLC file setting	Use PLC file setting
7						
8						
9						
10						
11						
12						
13						
14						
15						

Figure App.27 File usability setting

If the file usability is set in the High Performance model QCPU, change the setting as described below.

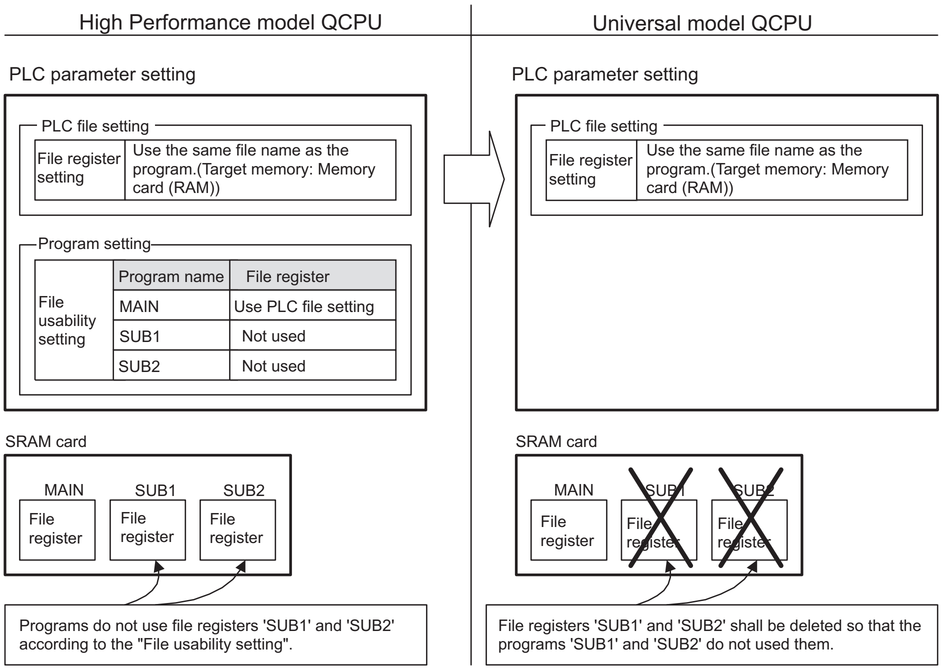
[*1] : The local device file usability setting is not available for the Univerasl model QCPU when the first five digits of the serial number are "10011" or earlier. If the local device is set to be used on the PLC file tab of PLC parameter in the High Performance model QCPU, all the programs use the local device after replacement with the Universal model QCPU.

(2) Method of replacing High Performance model QCPU with Universal model QCPU

(a) When file usability is set for file register, device initial value, and/or comment

Replacement method depends on the contents on the PLC file setting tab of PLC parameter.

TableApp.80 Replacement method

Contents in PLC file setting	Setting in Universal model QCPU
"Not used."h is selected.	No change in parameter setting is required.Operation of the Universal model QCPU is the same regardless of the file usability setting in the High Performance model QCPU.
"Use the same file name as the program." is selected.	When file usability is set to "Not used." in the High Performance model QCPU, delete the corresponding program file (file register, device initial value or comment), which uses the same name as the program, from the target memory. The Universal model QCPU executes a program without using a program file if no program file that uses the same name as the program exists in the target memory.  High Performance model QCPU Universal model QCPU PLC parameter setting PLC file setting File register setting Use the same file name as the program.(Target memory: Memory card (RAM)) Program setting File usability setting Program name File register MAIN Use PLC file setting SUB1 Not used SUB2 Not used PLC parameter setting PLC file setting File register setting Use the same file name as the program.(Target memory: Memory card (RAM)) SRAM card MAIN SUB1 SUB2 File register File register File register SRAM card MAIN SUB1 SUB2 File register File register File register Programs do not use file registers 'SUB1' and 'SUB2' according to the "File usability setting". File registers 'SUB1' and 'SUB2' shall be deleted so that the programs 'SUB1' and 'SUB2' do not used them.
"Use the following file." is selected.	No change in parameter setting is required.Operation of the Universal model QCPU is the same regardless of the file usability setting in the High Performance model QCPU.

(b) When file usability is set for local device

When the local device to be used is set on the PLC file setting tab of PLC parameter, all programs use the set local device.

Appendix 5.4.5 Parameter-valid Drive and Boot File Setting

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

The parameter-valid drive is specified at the switches on the front panel of the High Performance model QCPU.

(b) Universal model QCPU

The Universal model QCPU automatically determines the parameter-valid drive, depending on the existence of parameters in the drive (program memory, memory card or standard ROM). Therefore, when replacing the High Performance model QCPU with the Universal model QCPU, changing the boot file setting for parameter and/or moving files to another drive might be required.

When replacing the module, change the setting as described below.

(2) Method of replacing High Performance model QCPU with Universal model QCPU

(a) When the parameter-valid drive is set to the standard ROM in the High Performance model QCPU

TableApp.81 When the parameter-valid drive is set to the standard ROM

<table>
<tr><th>Setting in High Performance model QCPU</th><th rowspan="2">Setting in Universal model QCPU</th></tr>
<tr><th>Boot file setting of PLC parameter</th></tr>
<tr><td>No boot file setting</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the standard ROM.
• Changes in parameter setting are not required.
• Delete parameters that exist in the program memory and memory card. *2</td></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that programs are stored in the program memory in the first place, instead of booting from the standard ROM.
• Delete all settings for parameter on the Boot file setting tab of PLC parameter.
• Delete parameters that exist in the program memory and memory card. *2
• Move the programs with boot setting into the program memory from the standard ROM. *1</td></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Standard ROM</td><td>Program memory</td></tr></table></td><td>Change the setting so that programs and parameters are stored in the program memory in the first place, instead of booting from the standard ROM.
• Move the programs and parameters with boot setting into the program memory from the standard ROM. *1
• Delete all settings for parameter on the Boot file setting tab of PLC parameter.</td></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card and programs shall be booted from the memory card to the program memory.
• Move the parameters in the standard ROM into the memory card.
• Make setting so that programs are booted from the memory card to the program memory on the Boot file setting tab of PLC parameter. *3</td></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td>Delete all settings for data other than program and parameter in the boot file setting. Since these data can be used even not stored in the program memory, it is not necessary to transfer them to the program memory. Or, change the setting so that they are stored in the program memory in the first place.
• Delete all settings for data other than program and parameter on the Boot file setting tab of PLC parameter.
• Move the data other than programs and parameters into the program memory as needed.</td></tr>
</table>

(Continued on next page)

TableApp.77 When the parameter-valid drive is set to the standard ROM(Continue)

<table>
<tr><th>Setting in High Performance model QCPU</th><th rowspan="2">Setting in Universal model QCPU</th></tr>
<tr><th>Boot file setting of PLC parameter</th></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>(Data other than program and parameter)</td><td>Memory card</td><td>Program memory</td></tr></table></td><td rowspan="2">Delete all settings for data other than program and parameter in the boot file setting. Since these data can be used even not stored in the program memory, it is not necessary to transfer them to the program memory. Or, change the setting so that they are stored in the program memory in the first place.
• Delete all settings for data other than program and parameter on the Boot file setting tab of PLC parameter.
• Move the data other than programs and parameters into the program memory as needed.</td></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>(Data other than program and parameter)</td><td>Standard ROM</td><td>Program memory</td></tr></table>(Data other than program and parameter indicate initial device value, device comment, and label program.)</td></tr>
</table>

* 1: Since the Universal model QCPU retains the data in the program memory even the battery voltage drops, the boot file setting is not necessary.

* 2: The Universal model QCPU searches for parameters in order of the program memory, memory card, and the standard ROM, and then uses the parameters found first. If parameters exist in the program memory or the memory card, the Universal model QCPU cannot use the parameters in the standard ROM.

* 3: The Universal model QCPU ignores the boot file setting for parameters in the standard ROM.

(b) When the the parameter-valid drive is set to the memory card (RAM) or memory card (ROM) in the High Performance model QCPU

TableApp.82 When the parameter-valid drive is set to the memory card (RAM) or memory card (ROM)

<table>
<tr><th>Setting in High Performance model QCPU</th><th rowspan="2">Setting in Universal model QCPU</th></tr>
<tr><th>Boot file setting of PLC parameter</th></tr>
<tr><td>No boot file setting</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card.
• Changes in parameter setting are not required.
• Delete parameters that exist in the program memory. *2</td></tr>
<tr><td>Contents of boot file setting<table><tr><td>Type</td><td>Transfer from</td><td>Transfer to</td></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that the Universal model QCPU can refer to the parameters in the memory card.
• Changes in parameter setting are not required.
• Delete parameters that exist in the program memory. *2</td></tr>
<tr><td>Contents of boot file setting<table><tr><td>Type</td><td>Transfer from</td><td>Transfer to</td></tr><tr><td>Program</td><td>Memory card</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td>No changes are required.</td></tr>
<tr><td>Contents of boot file setting<table><tr><td>Type</td><td>Transfer from</td><td>Transfer to</td></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr></table>(No boot file setting for parameters)</td><td>Change the setting so that programs are stored in the program memory in the first place, instead of booting from the standard ROM.
• Move the programs targeted for booting from the standard ROM into the program memory. *1
• Delete all settings for program on the Boot file setting tab of PLC parameter.
• Delete parameters that exist in the program memory. *2</td></tr>
<tr><td>Contents of boot file setting<table><tr><td>Type</td><td>Transfer from</td><td>Transfer to</td></tr><tr><td>Program</td><td>Standard ROM</td><td>Program memory</td></tr><tr><td>Parameter</td><td>Memory card</td><td>Program memory</td></tr></table></td><td>Change the setting so that programs are stored in the program memory in the first place, instead of booting from the standard ROM.
• Move the programs targeted for booting from the standard ROM into the program memory. *1
• Delete all settings for program on the Boot file setting tab of PLC parameter.</td></tr>
</table>

(Continued on next page)

TableApp.78 When the parameter-valid drive is set to the memory card (RAM) or memory card (ROM)(Continue)

<table>
<tr><th>Setting in High Performance model QCPU</th><th rowspan="2">Setting in Universal model QCPU</th></tr>
<tr><th>Boot file setting of PLC parameter</th></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>(Data other than program and parameter)</td><td>Memory card</td><td>Program memory</td></tr></table></td><td rowspan="2">Delete all settings for data other than program and parameter in the boot file setting. Since these data can be used even not stored in the program memory, it is not necessary to transfer them to the program memory. Or, change the setting so that they are stored in the program memory in the first place.
• Delete all settings for data other than program and parameter on the Boot file setting tab of PLC parameter.
• Move the data other than program and parameter into the program memory as needed.</td></tr>
<tr><td>Contents of boot file setting<table><tr><th>Type</th><th>Transfer from</th><th>Transfer to</th></tr><tr><td>(Data other than program and parameter)</td><td>Standard ROM</td><td>Program memory</td></tr></table>(Data other than program and parameter indicate initial device value, device comment, and label program.)</td></tr>
</table>

* 1: Since the Universal model QCPU retains the data in the program memory even the battery voltage drops, the boot file setting is not necessary.

* 2: The Universal model QCPU searches for parameters in order of the program memory, memory card, and the standard ROM, and then uses the parameters found first. If parameters exist in the program memory or the memory card, the Universal model QCPU cannot use the parameters in the standard ROM.

Appendix 5.4.6 Forced ON/OFF Function of External I/O

(1) Differences between High Performance model QCPU and Universal model QCPU

(a) High Performance model QCPU

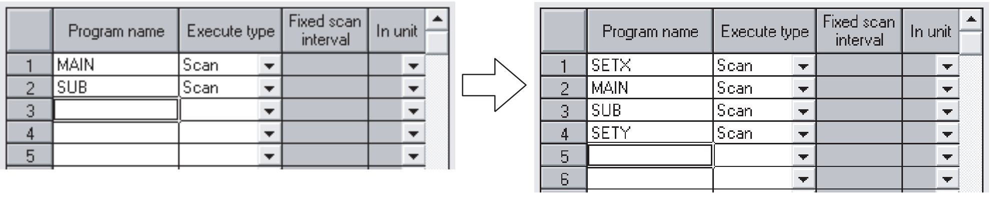
External I/O can be forcibly turned ON/OFF on the screen displayed by selecting [Online] → [Debug] → [Forced input output registration/cancellation] using GX Developer in the High Performance model QCPU.

(b) Universal model QCPU

The Universal model QCPU whose serial number (first five digits) is "10041" or earlier does not support the forced ON/OFF function of external I/O.
External I/O can be forcibly turned ON/OFF using the replacement program described below.

(2) Method of replacing High Performance model QCPU with Universal model QCPU

As shown in Figure App.28, add programs, "SETX" and "SETY", on the Program setting tab of PLC parameter.



	Program name	Execute type	Fixed scan interval	In unit
1	MAIN	Scan		
2	SUB	Scan		
3				
4				
5				

	Program name	Execute type	Fixed scan interval	In unit
1	SETX	Scan		
2	MAIN	Scan		
3	SUB	Scan		
4	SETY	Scan		
5				
6				

Figure App.28 Modification on the Program setting tab of PLC parameter

The following table shows the program setting of the "SETX" and "SETY".

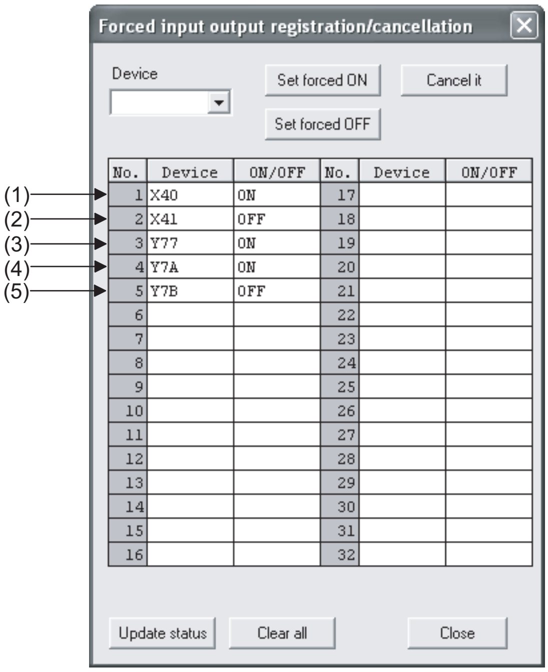
TableApp.83 Program setting of "SETX" and "SETY"

Program name	Execution type	Position where program is added
SETX	Scan	Start of Program setting (No.1)
SETY	Scan	End of Program setting

Example) Forcibly turning X40, X77, and X7A ON, and X41 and Y7B OFF

The programs, "SETX" and "SETY", turns ON or OFF the X and Y devices, which have been registered for forced ON/OFF using the forced ON/OFF function of external I/O, at each scan using the SET and RST instructions.

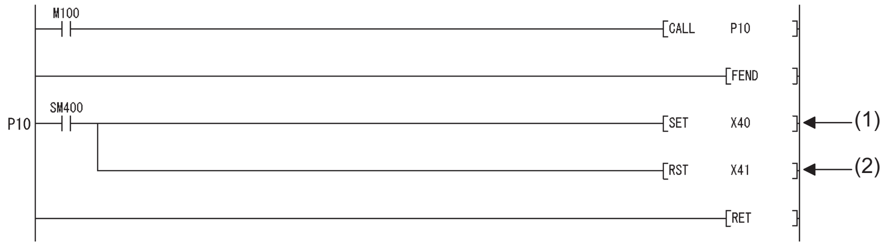
High Performance model QCPU



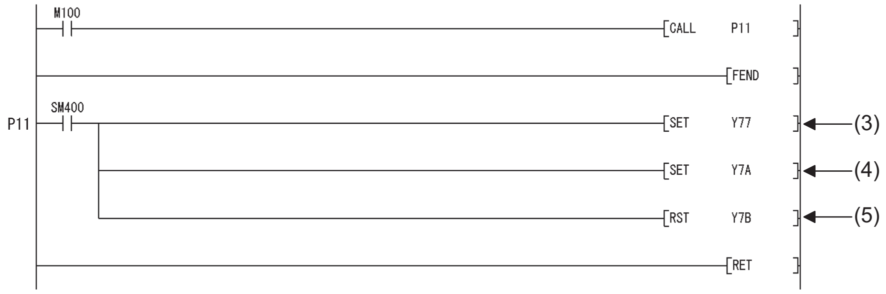
No.	Device	ON/OFF	No.	Device	ON/OFF
1	X40	ON	17		
2	X41	OFF	18		
3	Y77	ON	19		
4	Y7A	ON	20		
5	Y7B	OFF	21		
6			22		
7			23		
8			24		
9			25		
10			26		
11			27		
12			28		
13			29		
14			30		
15			31		
16			32		

Universal model QCPU

- Program example of “SETX”



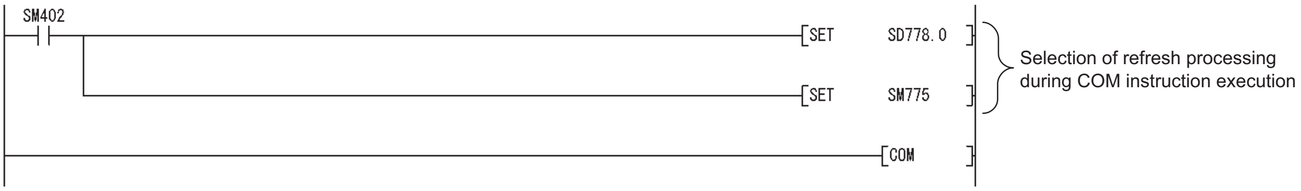
- Program example of “SETY”



(3) Replacing the COM instruction

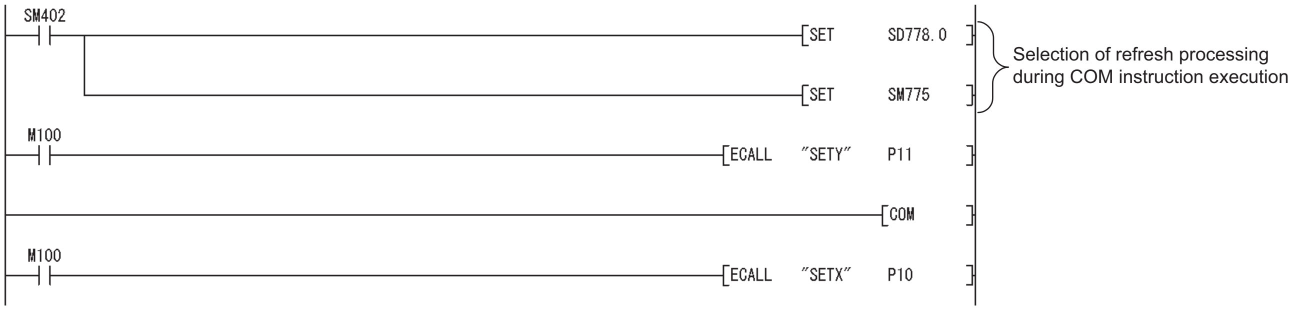
If the COM instruction is used, add subroutine calls for P10 and P11 before and after the COM instruction. (P10 and P11 are pointers shown in the program examples in (2).) When SM775 is ON (Executes refresh set by SD778) and also the 0 bit of SD778 is OFF (Do not execute I/O refresh), replacement of the instruction is not necessary.

(a) Program before replacement



Selection of refresh processing during COM instruction execution

(b) Program after replacement



Selection of refresh processing during COM instruction execution

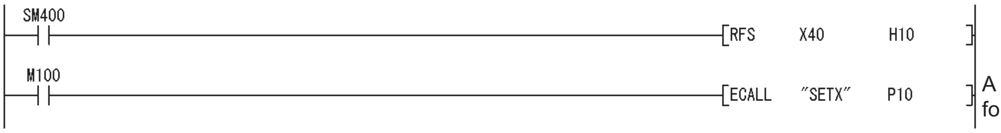
(4) Replacing the RFS instruction

If any I/O numbers targeted for forced ON/OFF are included in the partial refresh range specified by the RFS instruction, add subroutine calls for P10 and P11 before and after the RFS instruction. (P10 and P11 are pointers shown in the program examples in (2).)

If no I/O number targeted for forced ON/OFF is included, addition of subroutine calls for P10 and P11 is not necessary.

(a) When partial refresh for input (X) is executed by the RFS instruction

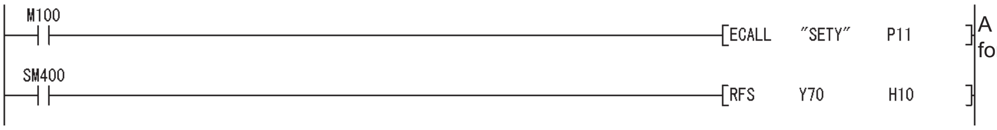
Add a subroutine call that executes forced input after the RFS instruction.



A subroutine call that executes forced input is added.

(b) When partial refresh for output (Y) is executed by the RFS instruction

Add a subroutine call that executes forced output before the RFS instruction.



A subroutine call that executes forced output is added.

(5) Restrictions

Replacements described in (2) to (4) do not apply in the following cases:

- Input and output targeted for forced ON/OFF are referred or changed using the direct input device (DX)/direct output device (DY).
- Input and output targeted for forced ON/OFF are referred or changed within an interrupt program.

Appendix 5.5 SPECIAL RELAYS AND SPECIAL REGISTERS

The Universal model QCPU does not support the special relays and special registers described in Appendix 5.5 and Appendix 5.5.
Replace them using the method described in the table or delete the corresponding parts.

Appendix 5.5.1 Special Relay List

TableApp.84 lists special relays not supported in the Universal model QCPU and measures to be taken.

TableApp.84 Special relays not supported in the Universal model QCPU and measures to be taken

Number	Name/Description		Measures to be taken
SM80	CHK detection		The Universal model QCPU does not support the CHK instruction.For the alternative method of the CHK instruction, refer to Appendix 5.3.
SM91	Step transition monitoring timer start		The Universal model QCPU does not support the step transition monitoring timer function.For the alternative method of this function, refer to "Appendix 3. Restrictions and alternative method of the Basic model QCPU and Universal model QCPU" in the "QCPU (Q Mode)/QnACPU Programming Manual (SFC)".
SM92			
SM93			
SM94			
SM95			
SM96			
SM97			
SM98			
SM99			
SM250	Largest mounted I/O number read		The Universal model QCPU always stores the largest mounted I/O number in SD250. The operation of this relay is not necessary.Delete the corresponding parts.
SM255	MELSECNET/H module 1 information	Indicates operative network or standby network	These are special relays for the simple dual-structured network function.Since the Universal model QCPU does not support this function, there is no application for these special relays. Delete the corresponding parts.
SM256		At refresh from link module to CPU, selects whether to read data from the link module.	
SM257		At refresh from CPU to link module, selects whether to write data to the link module.	
SM260	MELSECNET/H module 2 information	Indicates operative network or standby network	
SM261		At refresh from link module to CPU, selects whether to read data from the link module.	
SM262		At refresh from CPU to link module, selects whether to write data to the link module.	
SM265	MELSECNET/H module 3 information	Indicates operative network or standby network	
SM266		At refresh from link module to CPU, selects whether to read data from the link module.	
SM267		At refresh from CPU to link module, selects whether to write data to the link module.	
SM270	MELSECNET/H module 4 information	Indicates operative network or standby network	
SM271		At refresh from link module to CPU, selects whether to read data from the link module.	
SM272		At refresh from CPU to link module, selects whether to write data to the link module.	

(Continued on next page)

TableApp.80 Special relays not supported in the Universal model QCPU and measures to be taken(Continue)

Number	Name/Description	Measures to be taken
SM280	CC-Link error	Replace the relay with the I/O signals (Xn0, Xn1, XnF) of the mounted CC-Link module.
SM330	Operation mode for low-speed execution type program	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts.
SM331	Normal SFC program execution status	The Universal model QCPU supports only normal SFC programs. Delete SM331 and SM332 which are used as interlocks or replace them with SM321.
SM332	Program execution management SFC program execution status	
SM390	Access execution flag	According to sample programs described in the manual of each module, modify the program so that the module READY signal (Xn) is used as an interlock.
SM404	ON for only 1 scan after RUN of low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts or replace them with the special relays for scan execution type programs (SM402 and SM403).
SM405	OFF for only 1 scan after RUN of low-speed execution type programs	
SM430	User timing clock No.5 (for low-speed execution type programs)	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts or replace them with the special relays for scan execution type programs (SM420 and SM424).
SM431	User timing clock No.6 (for low-speed execution type programs)	
SM432	User timing clock No.7 (for low-speed execution type programs)	
SM433	User timing clock No.8 (for low-speed execution type programs)	
SM434	User timing clock No.9 (for low-speed execution type programs)	
SM510	Low-speed execution type program executing flag	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts.
SM551	Module service interval time read	The Universal model QCPU does not support the service interval measurement function.Delete the corresponding parts.
SM672	Memory card file register access range flag	When the Universal model QCPU accesses outside the range of file registers in the memory card, an OPERATION ERROR (error code: 4101) is detected.There is no application for this special relay.Delete the corresponding parts.
SM710	CHK instruction priority flag	The Universal model QCPU does not support the CHK instruction.For the alternative method of the CHK instruction, refer to Appendix 5.3.
SM734	XCALL instruction execution condition specification	The Universal model QCPU executes the XCALL instruction even at the rising edge of execution condition, too. There is no application for this special relay.Delete the corresponding parts.
SM735	SFC comment readout instruction in-execution flag	The Universal model QCPU does not support the following instructions: • SFC step comment readout instruction (S(P).SFCSCOMR) • SFC transition condition comment readout instruction (S(P).SFCTCOMR) Delete the corresponding parts.
SM1780*1	Power supply OFF detection flag	The Universal model QCPU does not store redundant power supply system information in SM1780 to SM1783.Delete the corresponding parts.(SM1780 to SM1783 are always OFF.)
SM1781*1	Power supply failure detection flag	
SM1782*1	Momentary power failure detection flag for power supply 1	
SM1783*1	Momentary power failure detection flag for power supply 2	

* 1: The special relay can be used in the Universal model QCPU whose serial number (first five digits) is "10042" or later.

Appendix 5.5.2 Special Register List

TableApp.85 lists special registers not supported in the Universal model QCPU and measures to be taken.

TableApp.85 Special registers not supported in the Universal model QCPU and measures to be taken

Number	Name/Description	Measures to be taken
SD80	CHK number	The Universal model QCPU does not support the CHK instruction.For the alternative method of the CHK instruction, refer to Appendix 5.3.
SD90	Step transition monitoring timer setting value	The Universal model QCPU does not support the step transition monitoring timer function.For the alternative method of this function, refer to "Appendix 3. Restrictions and alternative method of the Basic model QCPU and Universal model QCPU" in the "QCPU (Q Mode)/QnACPU Programming Manual (SFC)".
SD91		
SD92		
SD93		
SD94		
SD95		
SD96		
SD97		
SD98		
SD99		
SD280	CC-Link error	Replace these registers with the I/O signals (Xn0, Xn1, XnF) of the mounted CC-Link module.
SD281		
SD315	Time reserved for communication processing	Service processing setting is available for the Universal model QCPU on the PLC system setting tab of PLC parameter. Select "Specify service process time." in the service processing setting and set the service processing time. Other setting methods can be selected as well.
SD430	Low-speed scan counter	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding part or replace it with the special register for scan execution type programs (SD420).
SD510	Low-speed execution type program number	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding part or replace it with the special register for scan execution type programs (SD500).
SD528	Current scan time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts or replace them with the special registers for scan execution type programs (SD520 and SD521).
SD529		
SD532	Minimum scan time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts or replace them with the special registers for scan execution type programs (SD524 to SD527).
SD533		
SD534	Maximum scan time for low-speed execution type programs	
SD535		
SD544	Cumulative execution time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts.
SD545		
SD546	Execution time for low-speed execution type programs	The Universal model QCPU does not support low-speed execution type programs.Delete the corresponding parts.
SD547		
SD550	Service interval measurement module	The Universal model QCPU does not support the service interval measurement function.Delete the corresponding parts.
SD551	Service interval time	
SD552		

(Continued on next page)

TableApp.81 Special registers not supported in the Universal model QCPU and measures to be taken(Continue)

Number	Name/Description	Measures to be taken
SD720	Program No. specification for PLAODP instruction	The Universal model QCPU does not support the PLAODP instruction.Delete the corresponding part.
SD1780[*1]	Power supply OFF detection status	The Universal model QCPU does not store redundant power supply system information in SD1780 to SD1783.Delete the corresponding parts.(SD1780 to SD1783 are always OFF.)
SD1781[*1]	Power supply failure detection status	
SD1782[*1]	Momentary power failure detection counter for power supply 1	
SD1783[*1]	Momentary power failure detection counter for power supply 1	

* 1: The special register can be used in the Universal model QCPU whose serial number (first five digits) is "10042" or later.

Appendix 6 Specifications Comparison Between Ethernet Port of Built-in Ethernet Port QCPU and Ethernet Module

TableApp.86 shows the specifications comparison between the Ethernet port of the Built-in Ethernet port QCPU (QnUDE(H)CPU) and the Ethernet module (QJ71E71-100).

TableApp.86 Specifications Comparison between Built-in Ethernet port QCPU and Ethernet module

Item			Description	Availability	
				QnUDE(H)CPU	QJ71E71-100
Communication using MC protocol	4E frame		Reads/writes data in the CPU module from/to an external device. A frame format that can receive multiple request messages at a time.	×	○
	QnA-compatible 3E frame	Reading/ Writing data in device memory	Reads/writes data (device) in the CPU module from/to an external device.	○[*1]	○
		Other	Reads/writes data (file) in the CPU module from/to an external device.	×	○
	A-compatible 1E frame		• Reads/writes data in the CPU module from/to an external device. • A frame format that is compatible with A series E71.	×	○
Communication using fixed buffer	With procedure ("Procedure exist")		Sends/receives arbitrary data between the CPU module and an external device using the fixed buffer of the Ethernet module.	×	○
	Without procedure ("No procedure")			×	○
Communication using random access buffer			Reads/writes data from in the random access buffer of the Ethernet module from/to multiple external devices.	×	○
Data communication (sending/receiving) by e-mail			Sends/receives data by e-mail. • Sending/receiving e-mail by the CPU module • Sending/receiving e-mail using the CPU module monitoring function (the automatic notification function) of the Ethernet module	×	○
Communication using data link instructions			Reads/writes data in the CPU module on another station via Ethernet using data link instructions.	×	○
File transfer (FTP server function)			Reads/writes data in the CPU module in file units from/to an external device using FTP commands.	○[*2]	○
Communication using Web function			Accesses the CPU module from a personal computer in remote locations via the Internet using a commercially available Web browser.	×	○
CC-Link IE controller network, MELSECNET/H, MELSECNET/10 relay communication			Communicates data in the Ethernet, regarding the system as equivalent to CC-Link IE controller network, MELSECNET/H or MELSECNET/10.	×	○
Router relay communication (Router relay function)			Communicates data via a router or gateway. (The router relay function is not a function by which the Ethernet module works as a router.)	○[*3]	○
Send frame setting	Ethernet (V2.0)		Sends data using the frame format selected for the Ethernet header of the data link layer.	○	○
	IEEE802.3			×	○

○ : Function available, × : Function not available

(To the next page)

TableApp.86 Specifications Comparison between Built-in Ethernet port QCPU and Ethernet module (Continued)

Item		Description	Availability	
			QnUDE(H)CPU	QJ71E71-100
Existence check of external device (Existence check function)	Check with Ping ("Use the Ping")	Checks the connection status of an external device by sending a Ping message (ICMP Echo) to an external device. Closes the corresponding connection if no response message is received.	×	○
	Check with KeepAlive ("Use the KeepAlive")	Checks the connection status of an external device by sending an ACK message, which notifies an open status of the connection opened using the TCP protocol to an external device.	○[*4]	○
Communication using pairing open method		Pairs the receiving and sending connections and opens as one connection. (For communication using the fixed buffer.)	×	○
Communication using automatic open UDP port setting		Enables communication after the station in which an Ethernet module is mounted has been powered ON. (Open/close processing for communication by a sequence program is not required.)	×	○
Remote password check		Prevents unauthorized access to the CPU module by users in remote locations.	○	○
Simultaneous broadcast		Sends/receives data for all external devices on the same Ethernet where the Ethernet module is mounted in data communication using the UDP/IP protocol.	×	○
MELSOFT product [*8] (such as GX Developer) and/or GOT connection		Connects a MELSOFT product and/or GOT using the TCP/IP or UDP/IP protocol. Multiple MELSOFT products and/or GOTs can be connected simultaneously.	○	○
Find CPU function		Finds the CPU modules connected to the same hub as GX Developer, and displays a list.	○	×
Time setting function		Collects time information from the time information server and sets time in the CPU module automatically.	○	×
User connection		Connection which is used by user during communications using the MC protocol or the fixed buffer. Can be used as MELSOFT communication port of system connection by setting parameters. Up to 16 connections can be used.	○	○
System connection	Auto open UDP port	Connection which is used by the system only.	×	○
	FTP transmission port		○	○
	MELSOFT transmission port (UDP/IP)		○[*5]	○[*6]
	MELSOFT transmission port (TCP/IP)		○[*5]	○[*6][*7]
	HTTP port		×	○
	MELSOFT direct connection		○	×

○ : Function available, × : Function not available

[1] Available commands are restricted. (☞ Section 7.4)

[2] The "quote cpuchg" command is not supported. (☞ Section 7.6)

[3] Only a default router can be specified.

[4] Settings are fixed to the following: Interval timer: 5 seconds, Resend timer: 8 times.

[5] Up to 16 MELSOFT products can be connected by setting "MELSOFT connection" for user connections in PLC parameter.

[6] "MELSOFT transmission port" corresponds to "GX Developer transmission port" of the QJ71E71-100.

[7] Up to 17 MELSOFT products can be connected by setting "MELSOFT connection" for user connections (including one system connection) in Network parameter.

[8] The Built-in Ethernet port QCPU does not support the connection with MX Component.

Remark

For details on Ethernet modules, refer to the following manuals.

☞ Q Corresponding Ethernet Interface Module User's Manual (Basic)

☞ Q Corresponding Ethernet Interface Module User's Manual (Application)

Appendix 7 Access Range Using Ethernet Port of Built-in Ethernet Port QCPU

The following figure shows the access range and the access availability of GX Developer and GOT when the Ethernet port of the Built-in Ethernet port QCPU is used.

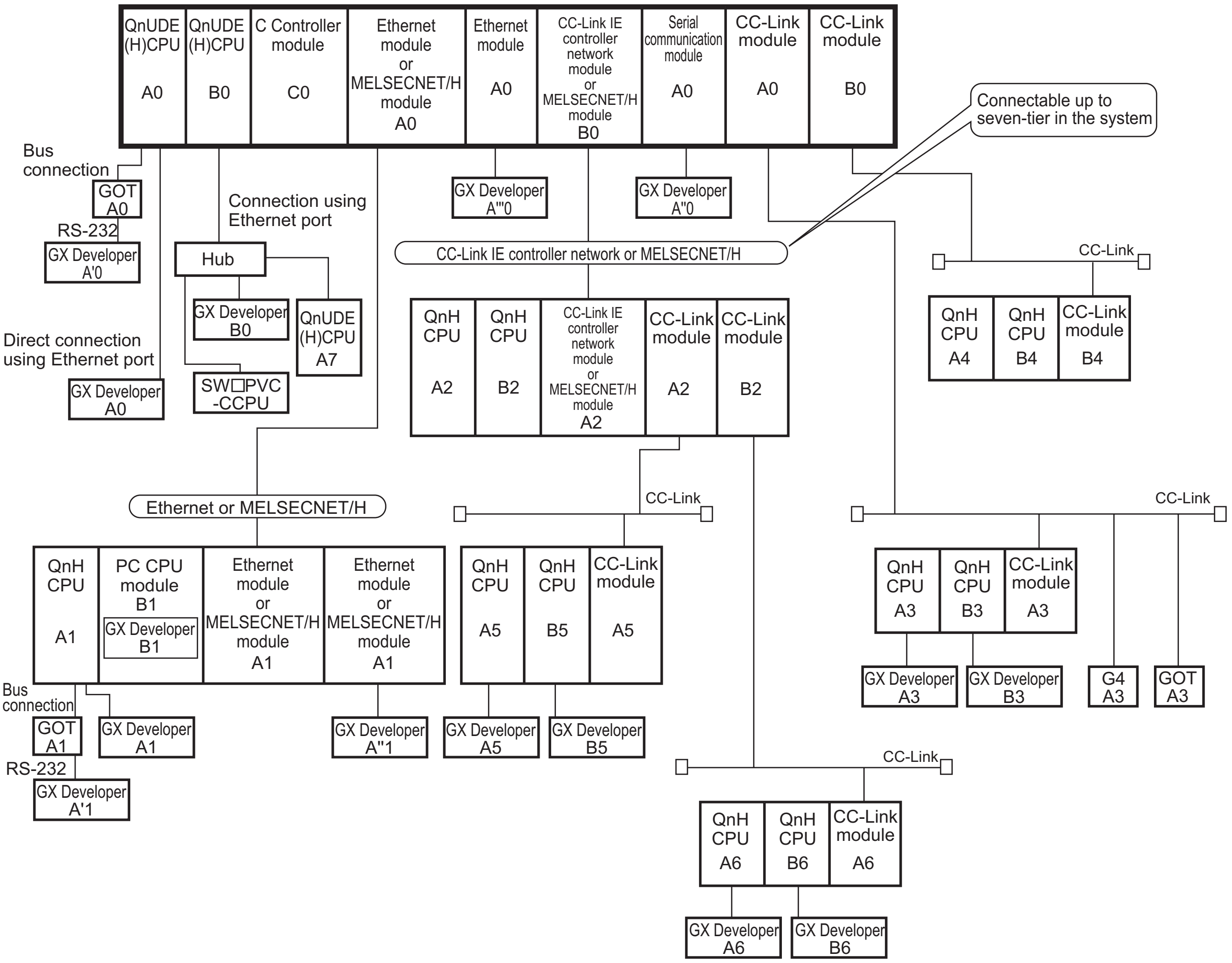


Figure App.29 Access range

TableApp.87 Access availability

Access source	Access target															
	QnUDE(H)CPU A0	QnUDE(H)CPU B0	QnUCPU A1	PC CPU module B1[*3]	QnHCPU A2	QnHCPU B2	QnHCPU A3	QnHCPU B3	QnHCPU A4	QnHCPU B4	QnHCPU A5	QnHCPU B5	QnHCPU A6	QnHCPU B6	QnUDE(H)CPU A7	C Controller module C0
GX Developer A0	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×
GOT A0	○	○	○	○	○	○	×	×	×	×	×	×	×	×	×	×
GX Developer A'0	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×
GX Developer A"0	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×
GX Developer A'"0	○	○	○	○	○	○	○	○	○	○	○[*1]	○[*1]	○[*1]	○[*1]	×	×
GX Developer B0	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×
GX Developer A1	○	○	○	○	○[*1]	○[*1]	○	○	○	○	○[*1]	○[*1]	○[*1]	○[*1]	×	×
GOT A1	○	○	○	○	○[*1]	○[*1]	×	×	×	×	×	×	×	×	×	×
GX Developer A'1	○	○	○	○	○[*1]	○[*1]	○	○	○	○	○[*1]	○[*1]	○[*1]	○[*1]	×	×
GX Developer B1 [*2]	×	×	○	○	×	×	×	×	×	×	×	×	×	×	×	×
GX Developer A"1	○	○	○	○	○[*1]	○[*1]	○	○	○	○	○[*1]	○[*1]	○[*1]	○[*1]	×	×
GX Developer A3	○	○	○	○	○	○	○	○	×	×	×	×	×	×	×	×
GX Developer B3	○	○	○	○	○	○	○	○	×	×	×	×	×	×	×	×
G4 A3	○	○	○	○	○	○	○	○	×	×	×	×	×	×	×	×
GOT A3	○	○	×	×	×	×	○	○	×	×	×	×	×	×	×	×
GX Developer A5	○	○	○[*1]	○[*1]	○	○	×	×	×	×	○	○	×	×	×	×
GX Developer B5	○	○	○[*1]	○[*1]	○	○	×	×	×	×	○	○	×	×	×	×
GX Developer A6	○	○	○[*1]	○[*1]	○	○	×	×	×	×	×	×	○	○	×	×
GX Developer B6	○	○	○[*1]	○[*1]	○	○	×	×	×	×	×	×	○	○	×	×
SW□PVC-CCPU	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×

○ : Available, × : Not available

* 1 : An access becomes available by setting routing parameters.

* 2 : "GX Developer B1" indicates GX Developer activated in PC CPU modules.

* 3 : "PC CPU module B1" indicates SX Controller (Q25SS) activated in PC CPU modules.

POINT

When using a MELSOFT product by connecting it to the Ethernet port of the Built-in Ethernet port CPU, serial communication or CC-Link cannot be specified as a connection target if the access target is QnA or A series CPU.

Remark

For GOT accessible to the CPU module, refer to the following manual.

☞ GT Designer2 Version2 Screen Design Manual

Appendix 8 Precautions for Battery Transport

When transporting lithium batteries, make sure to treat them based on the transport regulations.

(1) Controlled models

The batteries for the Q series CPU module (including memory cards) are classified as shown in TableApp.88.

TableApp.88 List of models for restricted tranportation

Product name	Model	Product supply status	Classification for transportation
Battery	Q8BAT	Lithium battery	Dangerous goods
Battery	Q8BAT-SET	Lithium battery with Q8BAT connection cable	
Battery	Q7BAT	Lithium battery	
Battery	Q7BAT-SET	Lithium battery with holder	
Battery	Q6BAT	Lithium battery	Non-dangerous goods
SRAM card battery	Q2MEMBAT Q3MEMBAT	Lithium coin battery	
Memory card	Q2MEM-1MBS Q2MEM-2MBS	Packed with lithium coin battery (Q2MEM-BAT)	
	Q3MEM-4MBS Q3MEM-8MBS	Packed with lithium coin battery (Q3MEM-BAT)	
	Q3MEM-4MBS-SET Q3MEM-8MBS-SET	Packed with lithium coin battery (Q3MEM-BAT) + Memory card protective cover	

(2) Transport guidelines

Comply with IATA Dangerous Goods Regulations, IMDG code and the local transport regulations when transporting products after unpacking or repacking, while Mitsubishi ships products with packages to comply with the transport regulations.

Also, contact the transporters.

Appendix 9 Device Point Assignment Sheet

(1) For Basic model QCPU

TableApp.89 Device Point Assignmene Sheet (For Basic model QCPU)

Device name	Symbol	Numeric notation	Number of device points*1*2		Restriction check	
			Number of points	Number	Capacity (Word)*3	Number of bit points*2
Input relay	X	16	2k (2048)points	X0000 to 07FF	÷ 16 128 words	× 1 2048 points
Output relay	Y	16	2k (2048)points	Y0000 to 07FF	÷ 16 128 words	× 1 2048 points
Internal relay	M	10	k( )points	M0 to	÷ 16 words	× 1 points
Latch relay	L	10	k( )points	L0 to	÷ 16 words	× 1 points
Link relay	B	16	k( )points	B0000 to	÷ 16 words	× 1 points
Annunciator	F	10	k( )points	F0 to	÷ 16 words	× 1 points
Link special relay	SB	16	1k (1024)points	SB0000 to 03FF	÷ 16 64 words	× 1 1024 points
Edge relay	V	10	k( )points	V0 to	÷ 16 words	× 1 points
Step relay	S	10	2k (2048)points	S0 to 2047	÷ 16 128 words	× 1 2048 points
Timer	T	10	k( )points	T0 to	$\times\frac{18}{16}$ words	× 2 points
Retentive timer	ST	10	k( )points	ST0 to	$\times\frac{18}{16}$ words	× 2 points
Counter	C	10	k( )points	C0 to	$\times\frac{18}{16}$ words	× 2 points
Data register	D	10	k( )points	D0 to	× 1 words	--
Link register	W	16	k( )points	W0000 to	× 1 words	--
Link special register	SW	16	1k (1024)points	SW0000 to 03FF	× 1 1024 words	--
Device total					words (16704 words or less)	points

*1 : ▨ The hatched number of points is fixed. (Unchangeable)

*2 : The maximum number of points of one device is 32k points.

*3 : Enter the value that is obtained by multiplying (or dividing) the number of device points by the numeral indicated in the capacity (word) field.

(2) For High Performance model QCPU, Process CPU, Redundant CPU

TableApp.90 Device Point Assignmene sheet (For High Performance model QCPU, Process CPU, Redundant CPU)

Device name	Symbol	Numeric notation	Number of device points*1*2		Restriction check	
			Number of points	Number	Capacity (words) *3	Number of bit points*2
Input relay	X	16	8k (8192)points	X0000 to 1FFF	÷16 512 words	× 1 8192points
Output relay	Y	16	8k (8192)points	Y0000 to 1FFF	÷16 512 words	× 1 8192points
Internal relay	M	10	k( )points	M0 to	÷16 words	× 1 points
Latch relay	L	10	k( )points	L0 to	÷16 words	× 1 points
Link relay	B	16	k( )points	B0000 to	÷16 words	× 1 points
Annunciator	F	10	k( )points	F0 to	÷16 words	× 1 points
Link special relay	SB	16	2k (2048)points	SB0000 to 07FF	÷16 128 words	× 1 2048points
Edge relay	V	10	k( )points	V0 to	÷16 words	× 1 points
Step relay	S	10	8k (8192)points	S0 to 8191	÷16 512 words	× 1 8192points
Timer	T	10	k( )points	T0 to	$\times\frac{18}{16}$ words	× 2 points
Retentive timer	ST	10	k( )points	ST0 to	$\times\frac{18}{16}$ words	× 2 points
Counter	C	10	k( )points	C0 to	$\times\frac{18}{16}$ words	× 2 points
Data register	D	10	k( )points	D0 to	× 1 words	--
Link register	W	16	k( )points	W0000 to	× 1 words	--
Link special register	SW	16	2k (2048)points	SW0000 to 07FF	× 1 2048 words	--
Device total					words (29696 words or less)	points (65536 points or less)

*1 : □ The hatched number of points is fixed. (Unchangeable)

*2 : The maximum number of points of one device is 32k points.

*3 : Enter the value that is obtained by multiplying (or dividing) the number of device points by the numeral indicated in the capacity (word) field.

(3) For Universal model QCPU

TableApp.91 Device Point Assignmene sheet (For Universal model QCPU)

Device name	Symbol	Numeric notation	Number of device points*1*2		Restriction check	
			Number of points	Number	Capacity (words) *3	Number of bit points*2
Input relay	X	16	8k (8192)points	X0000 to 1FFF	÷ 16 512 words	× 1 8192points
Output relay	Y	16	8k (8192)points	Y0000 to 1FFF	÷ 16 512 words	× 1 8192points
Internal relay	M	10	k( )points	M0 to	÷ 16 words	× 1 points
Latch relay	L	10	k( )points	L0 to	÷ 16 words	× 1 points
Link relay	B	16	k( )points	B0000 to	÷ 16 words	× 1 points
Annunciator	F	10	k( )points	F0 to	÷ 16 words	× 1 points
Link special relay	SB	16	k( )points	SB0000 to	÷ 16 words	× 1 points
Edge relay	V	10	k( )points	V0 to	÷ 16 words	× 1 points
Step relay	S	10	8k (8192)points	S0 to 8191	÷ 16 512 words	× 1 8192points
Timer	T	10	k( )points	T0 to	$\times\frac{18}{16}$ words	× 2 points
Retentive timer	ST	10	k( )points	ST0 to	$\times\frac{18}{16}$ words	× 2 points
Counter	C	10	k( )points	C0 to	$\times\frac{18}{16}$ words	× 2 points
Data register	D	10	k( )points	D0 to	× 1 words	--
Link register	W	16	k( )points	W0000 to	× 1 words	--
Link special register	SW	16	k( )points	SW0000 to	× 1 words	--
Device total					words (29696 words or less)	points (65536 points or less)

*1: The number of points in the □ area is fixed by the system. (Unchangeable)
Note, however, that the Universal model QCPU whose serial number (first five digits) is "10042" or later can set 0 point for step relays (S).

*2: The maximum number of points of one device is 32k points.
Note, however, that the Universal model QCPU whose serial number (first five digits) is "10042" or later can set up to 60k points for internal relays (M) and link relays (B).

*3: Enter the value that is obtained by multiplying (or dividing) the number of device points by the numeral indicated in the capacity (word) field.

INDEX

Memo

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 of damages caused by any cause found not to be the responsibility of Mitsubishi, loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products, special damages and secondary damages whether foreseeable or not , compensation for accidents, and compensation for damages to products other than Mitsubishi products, 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.

6. Product application

(1) In using the Mitsubishi MELSEC programmable controller, the usage conditions shall be that the application will not lead to a major accident even if any problem or fault should occur in the programmable controller device, and that backup and fail-safe functions are systematically provided outside of the device for any problem or fault.

(2) The Mitsubishi programmable controller has been designed and manufactured for applications in general industries, etc. Thus, applications in which the public could be affected such as in nuclear power plants and other power plants operated by respective power companies, and applications in which a special quality assurance system is required, such as for Railway companies or Public service purposes shall be excluded from the programmable controller applications.
In addition, applications in which human life or property that could be greatly affected, such as in aircraft, medical applications, incineration and fuel devices, manned transportation, equipment for recreation and amusement, and safety devices, shall also be excluded from the programmable controller range of applications.
However, in certain cases, some applications may be possible, providing the user consults their local Mitsubishi representative outlining the special requirements of the project, and providing that all parties concerned agree to the special circumstances, solely at the users discretion.

QCPU

User's Manual (Function Explanation, Program Fundamentals)

MODEL	QCPU-U-KP-E
MODEL CODE	13JR74
SH(NA)-080484ENG-M(0807)MEE	

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.

MITSUBISHI ELECTRIC

HEADQUARTERS

MITSUBISHI ELECTRIC EUROPE B.V. **EUROPE**
German Branch
Gothaer Straße 8
D-40880 Ratingen
Phone: +49 (0)2102 / 486-0
Fax: +49 (0)2102 / 486-1120

MITSUBISHI ELECTRIC EUROPE B.V. **CZECH REPUBLIC**
Czech Branch
Radlická 714/113a
CZ-158 00 Praha 5
Phone: +420 (0)251 551 470
Fax: +420 (0)251-551-471

MITSUBISHI ELECTRIC EUROPE B.V. **FRANCE**
French Branch
25, Boulevard des Bouvets
F-92741 Nanterre Cedex
Phone: +33 (0)1 / 55 68 55 68
Fax: +33 (0)1 / 55 68 57 57

MITSUBISHI ELECTRIC EUROPE B.V. **IRELAND**
Irish Branch
Westgate Business Park, Ballymount
IRL-Dublin 24
Phone: +353 (0)1 4198800
Fax: +353 (0)1 4198890

MITSUBISHI ELECTRIC EUROPE B.V. **ITALY**
Italian Branch
Viale Colleoni 7
I-20041 Agrate Brianza (MI)
Phone: +39 039 / 60 53 1
Fax: +39 039 / 60 53 312

MITSUBISHI ELECTRIC EUROPE B.V. **SPAIN**
Spanish Branch
Carretera de Rubí 76-80
E-08190 Sant Cugat del Vallés (Barcelona)
Phone: 902 131121 // +34 935653131
Fax: +34 935891579

MITSUBISHI ELECTRIC EUROPE B.V. **UK**
UK Branch
Travellers Lane
UK-Hatfield, Herts. AL10 8XB
Phone: +44 (0)1707 / 27 61 00
Fax: +44 (0)1707 / 27 86 95

MITSUBISHI ELECTRIC CORPORATION **JAPAN**
Office Tower "Z" 14 F
8-12,1 chome, Harumi Chuo-Ku
Tokyo 104-6212
Phone: +81 3 622 160 60
Fax: +81 3 622 160 75

MITSUBISHI ELECTRIC AUTOMATION, Inc. **USA**
500 Corporate Woods Parkway
Vernon Hills, IL 60061
Phone: +1 847 478 21 00
Fax: +1 847 478 22 53

EUROPEAN REPRESENTATIVES

GEVA **AUSTRIA**
Wiener Straße 89
AT-2500 Baden
Phone: +43 (0)2252 / 85 55 20
Fax: +43 (0)2252 / 488 60

TEHNIKON **BELARUS**
Oktyabrskaya 16/5, Off. 703-711
BY-220030 Minsk
Phone: +375 (0)17 / 210 46 26
Fax: +375 (0)17 / 210 46 26

Koning & Hartman b.v. **BELGIUM**
Woluwelaan 31
BE-1800 Vilvoorde
Phone: +32 (0)2 / 257 02 40
Fax: +32 (0)2 / 257 02 49

INEA BH d.o.o. **BOSNIA AND HERZEGOVINA**
Aleja Lipa 56
BA-71000 Sarajevo
Phone: +387 (0)33 / 921 164
Fax: +387 (0)33/ 524 539

AKHNATON **BULGARIA**
4 Andrej Ljapchev Blvd. Pb 21
BG-1756 Sofia
Phone: +359 (0)2 / 817 6004
Fax: +359 (0)2 / 97 44 06 1

INEA CR d.o.o. **CROATIA**
Losinjska 4 a
HR-10000 Zagreb
Phone: +385 (0)1 / 36 940 - 01/ -02/ -03
Fax: +385 (0)1 / 36 940 - 03

AutoCont C.S. s.r.o. **CZECH REPUBLIC**
Technologická 374/6
CZ-708 00 Ostrava-Pustkovec
Phone: +420 595 691 150
Fax: +420 595 691 199

B:TECH A.S. **CZECH REPUBLIC**
U Borové 69
CZ-58001 Havlíčkův Brod
Phone: +420 (0)569 777 777
Fax: +420 (0)569-777 778

Beijer Electronics A/S **DENMARK**
Lykkegårdsvej 17, 1.
DK-4000 Roskilde
Phone: +45 (0)46/ 75 76 66
Fax: +45 (0)46 / 75 56 26

Beijer Electronics Eesti OÜ **ESTONIA**
Pärnu mnt.160i
EE-11317 Tallinn
Phone: +372 (0)6 / 51 81 40
Fax: +372 (0)6 / 51 81 49

Beijer Electronics OY **FINLAND**
Jaakonkatu 2
FIN-01620 Vantaa
Phone: +358 (0)207 / 463 500
Fax: +358 (0)207 / 463 501

UTECO A.B.E.E. **GREECE**
5, Mavrogenous Str.
GR-18542 Piraeus
Phone: +30 211 / 1206 900
Fax: +30 211 / 1206 999

MELTRADE Ltd. **HUNGARY**
Fertő utca 14.
HU-1107 Budapest
Phone: +36 (0)1 / 431-9726
Fax: +36 (0)1 / 431-9727

Beijer Electronics SIA **LATVIA**
Vestienas iela 2
LV-1035 Riga
Phone: +371 (0)784 / 2280
Fax: +371 (0)784 / 2281

Beijer Electronics UAB **LITHUANIA**
Savanoriu Pr. 187
LT-02300 Vilnius
Phone: +370 (0)5 / 232 3101
Fax: +370 (0)5 / 232 2980

EUROPEAN REPRESENTATIVES

INTEHSIS srl **MOLDOVA**
bld. Traian 23/1
MD-2060 Kishinev
Phone: +373 (0)22 / 66 4242
Fax: +373 (0)22 / 66 4280

Koning & Hartman b.v. **NETHERLANDS**
Haarlerbergweg 21-23
NL-1101 CH Amsterdam
Phone: +31 (0)20 / 587 76 00
Fax: +31 (0)20 / 587 76 05

Beijer Electronics AS **NORWAY**
Postboks 487
NO-3002 Drammen
Phone: +47 (0)32 / 24 30 00
Fax: +47 (0)32 / 84 85 77

MPL Technology Sp. z o.o. **POLAND**
Ul. Krakowska 50
PL-32-083 Balice
Phone: +48 (0)12 / 630 47 00
Fax: +48 (0)12 / 630 47 01

Sirius Trading & Services srl **ROMANIA**
Aleea Lacul Morii Nr. 3
RO-060841 Bucuresti, Sector 6
Phone: +40 (0)21 / 430 40 06
Fax: +40 (0)21 / 430 40 02

Craft Con. & Engineering d.o.o. **SERBIA**
Bulevar Svetog Cara Konstantina 80-86
SER-18106 Nis
Phone: +381 (0)18 / 292-24-4/5
Fax: +381 (0)18 / 292-24-4/5

INEA SR d.o.o. **SERBIA**
Izletnicka 10
SER-113000 Smederevo
Phone: +381 (0)26 / 617 163
Fax: +381 (0)26 / 617 163

AutoCont Control s.r.o. **SLOVAKIA**
Radlinského 47
SK-02601 Dolny Kubin
Phone: +421 (0)43 / 5868210
Fax: +421 (0)43 / 5868210

CS MTrade Slovensko, s.r.o. **SLOVAKIA**
Vajanskeho 58
SK-92101 Piestany
Phone: +421 (0)33 / 7742 760
Fax: +421 (0)33 / 7735 144

INEA d.o.o. **SLOVENIA**
Stegne 11
SI-1000 Ljubljana
Phone: +386 (0)1 / 513 8100
Fax: +386 (0)1 / 513 8170

Beijer Electronics AB **SWEDEN**
Box 426
SE-20124 Malmö
Phone: +46 (0)40 / 35 86 00
Fax: +46 (0)40 / 35 86 02

Econotec AG **SWITZERLAND**
Hinterdorfstr. 12
CH-8309 Nürensdorf
Phone: +41 (0)44 / 838 48 11
Fax: +41 (0)44 / 838 48 12

GTS **TURKEY**
Darülaceze Cad. No. 43 KAT. 2
TR-34384 Okmeydanı-Istanbul
Phone: +90 (0)212 / 320 1640
Fax: +90 (0)212 / 320 1649

CSC Automation Ltd. **UKRAINE**
15, M. Raskova St., Fl. 10, Office 1010
UA-02002 Kiev
Phone: +380 (0)44 / 494 33 55
Fax: +380 (0)44 / 494-33-66

EURASIAN REPRESENTATIVES

Kazpromautomatics Ltd. **KAZAKHSTAN**
Mustafina Str. 7/2
KAZ-470046 Karaganda
Phone: +7 7212 / 50 11 50
Fax: +7 7212 / 50 11 50

CONSYS **RUSSIA**
Promyshlennaya st. 42
RU-198099 St. Petersburg
Phone: +7 812 / 325 36 53
Fax: +7 812 / 325 36 53

ELECTROTECHNICAL SYSTEMS **RUSSIA**
Derbenevskaya st. 11A, Office 69
RU-115114 Moscow
Phone: +7 495 / 744 55 54
Fax: +7 495 / 744 55 54

ELEKTROSTILY **RUSSIA**
Rubzowskaja nab. 4-3, No. 8
RU-105082 Moscow
Phone: +7 495 / 545 3419
Fax: +7 495 / 545 3419

NPP "URALELEKTRA" **RUSSIA**
Sverdlova 11A
RU-620027 Ekaterinburg
Phone: +7 343 / 353 2745
Fax: +7 343 / 353 2461

MIDDLE EAST REPRESENTATIVES

ILAN & GAVISH Ltd. **ISRAEL**
24 Shenkar St., Kiryat Arie
IL-49001 Petah-Tiqva
Phone: +972 (0)3 / 922 18 24
Fax: +972 (0)3 / 924 0761

TEXEL ELECTRONICS Ltd. **ISRAEL**
2 Ha´umanut, P.O.B. 6272
IL-42160 Netanya
Phone: +972 (0)9 / 863 39 80
Fax: +972 (0)9 / 885 24 30

AFRICAN REPRESENTATIVE

CBI Ltd. **SOUTH AFRICA**
Private Bag 2016
ZA-1600 Isando
Phone: + 27 (0)11 / 928 2000
Fax: + 27 (0)11 / 392 2354

MITSUBISHI ELECTRIC
FACTORY AUTOMATION