

# **MELSEC System Q**

Programmable Logic Controllers

User's Manual  
(Functioning and Programming)

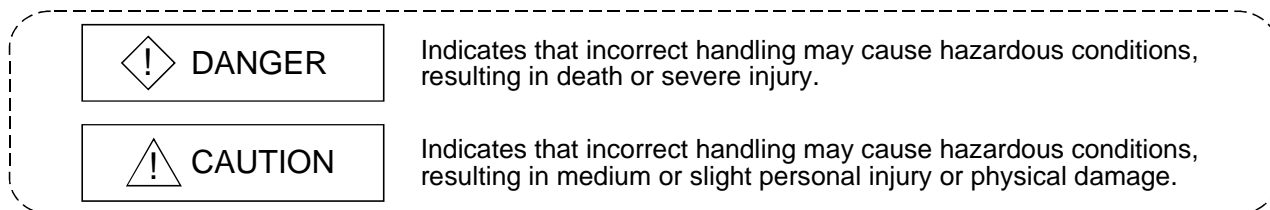
## **CPU Modules Q CPU (Q Mode)**


## • SAFETY INSTRUCTIONS •

(Always read these instructions before using this equipment.)

When using Mitsubishi equipment, thoroughly read this manual and the associated manuals introduced in this manual. Also pay careful attention to safety and handle the module properly.

These SAFETY PRECAUTIONS classify the safety precautions into two categories: "DANGER" and "CAUTION".



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 stop calculation and hold or turn off all output according to the parameter setting.
  - (a) The power supply module has over current protection equipment and over voltage protection equipment.
  - (b) The PLC CPUs self-diagnosis functions, such as the Watch dog timer error, detect problems.

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 High Performance model QCPU (Q Mode) 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 output 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 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 High Performance model QCPU (Q Mode) 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.
- Hold down the module loading lever at the module bottom, and securely insert the module fixing latch into the fixing hole in the base unit. 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 more cables, be sure that the base unit and the 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 external power supply before loading or unloading the module.  
Not doing so could result in electric shock or damage to the product.
- 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 external power supply 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.

## [Wiring Precautions]

### CAUTION

- 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.
- 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.

## [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 all phases of the external power supply off 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 a cellular phone or PHS more than 25cm (9.85 inch) away from the PLC.  
Not doing so can cause a malfunction.
- Switch all phases of the external power supply off before mounting or removing the module.  
If you do not switch off the external power supply, it will cause failure or malfunction of the module.
- 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 5 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
Dec., 1999	SH (NA)-080038-A	First edition
Dec., 2000	SH (NA)-080038-B	<p>Add the Q33B type main base unit and Q63B type extension base unit.            Change Chapter 11 (1) to (3) to Section 11.1 to 11.3.            Unify the name from the software package (GPP function, ladder logic test tool function, GPPW, etc) to the product name (GX Developer, GX Configurator).            Add the explanation of the following functions of which serial number's upper five digits were added in 02092 (02092*****-A)</p> <ul style="list-style-type: none"> <li>• Automatic write to standard ROM</li> <li>• Forced ON/OFF correspondence for external I/O</li> <li>• Remote password setting</li> <li>• Increment of Q12HCPU and Q25HCPU standard RAM capacity</li> <li>• MELSECNET/H remote I/O network correspondence</li> <li>• Interrupt module (QI60) correspondence</li> </ul> <p><b>Correction</b>            Section 1.1, Section 2.1, 2.2, Chapter 3, Section 4.2, 4.2.1, 4.2.2, 4.2.3, 4.6, Section 5.2, 5.3, Section 6.1, 6.9.3, Section 7.3 (3) (4), 7.6.5, 7.8, 7.14, 7.18, 7.19.1, 7.20.1, Chapter 9, Section 10.2, 10.10, Section 11.3, Appendix 1,2</p> <p><b>Addition</b>            Section 2.3, Section 4.1.3, 4.2.5, Section 5.4, 5.5.2, 5.6.2, Section 6.6, 6.6.2, 6.7, Section 7.6.3, 7.7.2, 7.7.3, 7.9.1, 7.9.3, 7.16, 7.17, 7.17.2, 7.20.1, Section 10.10</p>
Jun., 2001	SH (NA)-080038-C	<p>The explanation of the multiple PLC system added to the function version B of the QCPU was added to Chapters 13 to 19.            General name for QCPU was changed to the High Performance model QCPU.            The Q52B and Q55B extension base units and PC CPU module were added.</p> <p><b>Overall correction</b></p>
Apr., 2002	SH (NA)-080038-D	<p>Addition of the high speed interrupt function (Section 7.20) added to the serial No. whose upper 5 digits are 04012 (04012*****) and the description of the Q2MEM-2MBS SRAM card</p> <p><b>Overall correction</b></p>
Mar., 2003	SH (NA)-080038-E	<p><b>Additon model</b>            Q32SB, Q33SB, Q35SB, Q61SP</p> <p><b>Correction</b>            SAFETY PRECAUTIONS, About Manuals, Chapters 1, Section 1.2, 1.3, Section 2.1, 2.2, 2.3, Chapters 3, Section 4.2.2, 4.3.1, 4.4, 4.8.4, Section 5.6.1, 5.8, Chapters 6, Section 6.1, 6.2, 6.3, 6.5, 6.7, 6.8, 6.9.1, 6.9.2, 6.9.3, Section 7.1, 7.12, 7.12.1, 7.13, 7.14, 7.15.2, 7.17, 7.17.2, 7.19.2, 7.21.1, 7.21.2, 7.22.2, Chapters 9, Section 10.1, 10.13.2, Section 11.2, Section 14.1, 14.2.4, Section 15.2, Section 16.2, Section 17.2, Appendix 1, 2, 4.2, 4.3</p> <p><b>Addition</b>            Section 6.9.4, Section 7.8, 7.9</p>

Print Date	* Manual Number	Revision
Jul., 2003	SH (NA)-080038-F	<div data-bbox="587 315 743 344" style="border: 1px solid black; padding: 2px;">Correction</div> SAFETY PRECAUTIONS, Section 2.1, Section 6.2, 6.7, Section 10.2.11, Section 13.1, 13.3, Section 14.1, Chapters 16, Section 16.3.2, Appendix 2  <div data-bbox="587 427 715 456" style="border: 1px solid black; padding: 2px;">Addition</div> Appendix 5, 5.1, 5.2

Japanese Manual Version SH-080020-J

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.

© 1999 MITSUBISHI ELECTRIC CORPORATION



## INTRODUCTION

Thank you for choosing the Mitsubishi MELSEC-Q Series of General Purpose Programmable Controllers. Please read this manual carefully so that equipment is used to its optimum.

## CONTENTS

SAFETY INSTRUCTIONS .....	A- 1
REVISIONS .....	A- 6
CONTENTS.....	A- 8
Manuals.....	A-19
How to Use This Manual.....	A-20
Generic Terms and Abbreviations.....	A-21

<b>1 OVERVIEW</b>	<b>1 – 1 to 1 - 11</b>
-------------------	------------------------

1.1 Features.....	1- 2
1.2 Programs .....	1- 5
1.3 Convenient Programming Devices and Instructions .....	1- 8

<b>2 SYSTEM CONFIGURATION FOR SINGLE CPU SYSTEM</b>	<b>2- 1 to 2- 8</b>
---	---------------------

2.1 System Configuration.....	2- 1
2.2 Precaution on System Configuration.....	2- 6
2.3 Confirming the Serial No. and Function Versions.....	2- 8

<b>3 PERFORMANCE SPECIFICATION</b>	<b>3- 1 to 3- 3</b>
------------------------------------	---------------------

<b>4 SEQUENCE PROGRAM CONFIGURATION AND EXECUTION CONDITIONS</b>	<b>4- 1 to 4-53</b>
--	---------------------

4.1 Sequence Program .....	4- 1
4.1.1 Main routine program .....	4- 3
4.1.2 Sub-routine programs .....	4- 4
4.1.3 Interrupt programs.....	4- 6
4.2 Program Execute Type .....	4-10
4.2.1 Initial execution type program .....	4-16
4.2.2 Scan execution type program .....	4-18
4.2.3 Low speed execution type program .....	4-20
4.2.4 Stand-by type program .....	4-26
4.2.5 Fixed scan execution type program.....	4-32
4.3 Operation Processing.....	4-35
4.3.1 Initial processing.....	4-35
4.3.2 I/O refresh (I/O module refresh processing).....	4-35
4.3.3 Automatic refresh of the intelligent function module .....	4-36
4.3.4 END processing .....	4-36
4.4 RUN, STOP, PAUSE Operation Processing.....	4-37
4.5 Operation Processing during Momentary Power Failure .....	4-38

4.6 Data Clear Processing .....	4-39
4.7 I/O Processing and Response Lag.....	4-40
4.7.1 Refresh mode .....	4-40
4.7.2 Direct mode .....	4-43
4.8 Numeric Values which Can Be Used in Sequence Programs .....	4-45
4.8.1 BIN (Binary code) .....	4-47
4.8.2 HEX (Hexadecimal).....	4-48
4.8.3 BCD (Binary Coded Decimal) .....	4-49
4.8.4 Real numbers (floating decimal point data).....	4-50
4.9 Character String Data.....	4-53

<b>5 ASSIGNMENT OF I/O NUMBERS</b>	<b>5- 1 to 5-19</b>
------------------------------------	---------------------

5.1 Relationship Between the Number of Stages and Slots of the Extension Base Unit .....	5- 1
5.2 Installing Extension Base Units and Setting the Number of Stages .....	5- 2
5.3 Base Unit Assignment (Base Mode) .....	5- 3
5.4 What are I/O Numbers? .....	5- 7
5.5 Concept of I/O Number Assignment.....	5- 8
5.5.1 I/O numbers of main base unit, Slim type base unit and extension base unit .....	5- 8
5.5.2 Remote station I/O number.....	5-10
5.6 I/O Assignment by GX Developer.....	5-11
5.6.1 Purpose of I/O assignment by GX Developer .....	5-11
5.6.2 Concept of I/O assignment using GX Developer .....	5-13
5.7 Examples of I/O Number Assignment.....	5-16
5.8 Checking the I/O Numbers .....	5-19

<b>6 MEMORIES AND FILES HANDLED BY HIGH PERFORMANCE MODEL QCPU</b>	<b>6- 1 to 6-28</b>
--	---------------------

6.1 About the High Performance model QCPU's Memory.....	6- 3
6.2 Program Memory.....	6- 6
6.3 About the Standard ROM .....	6- 8
6.4 About the Standard RAM.....	6- 9
6.5 Memory Card .....	6-11
6.6 Writing Data to the Standard ROM or the Flash Card.....	6-13
6.6.1 Writing Data to the standard ROM or to the Flash card using GX Developer .....	6-13
6.6.2 Automatic write to standard ROM (Auto Download all Data from Memory card to standard ROM) .....	6-15
6.7 Executing Standard ROM/Memory Card Programs (Boot Run).....	6-18
6.8 Program File Configuration.....	6-21
6.9 GX Developer File Operation and File Handling Precautions .....	6-23
6.9.1 File operation.....	6-23
6.9.2 File handling precautions .....	6-24
6.9.3 File capacity.....	6-25
6.9.4 Memory capacity for files .....	6-26

<b>7 FUNCTION</b>	<b>7- 1 to 7-94</b>
-------------------	---------------------

7.1 Function List .....	7- 1
7.2 Constant Scan.....	7- 2
7.3 Latch Functions.....	7- 5

7.4 Setting the Output (Y) Status when Changing from/to STOP Status to/from RUN Status .....	7- 7
7.5 Clock Function .....	7- 9
7.6 Remote Operation .....	7-12
7.6.1 Remote RUN/STOP .....	7-12
7.6.2 Remote PAUSE.....	7-15
7.6.3 Remote RESET .....	7-17
7.6.4 Remote latch clear .....	7-19
7.6.5 Relationship of the remote operation and High Performance model QCPU RUN/STOP switch...	7-20
7.7 Selecting the Input Response Time of the Q Series Module (I/O Response Time) .....	7-21
7.7.1 Selecting the response time of the input module .....	7-21
7.7.2 Selecting the response time of the high speed input module .....	7-22
7.7.3 Selecting the response time of the interrupt module .....	7-23
7.8 Error-time Output Mode Setting .....	7-24
7.9 Hardware Error-time CPU Operation Mode Setting .....	7-25
7.10 Setting the Switches of the Intelligent Function Module.....	7-26
7.11 Monitoring Function.....	7-27
7.11.1 Monitor condition setting .....	7-27
7.11.2 Monitoring test for local device .....	7-31
7.11.3 Enforced ON/OFF for external I/O.....	7-33
7.12 Writing in Program during High Performance model QCPU RUN .....	7-37
7.12.1 Writing data in the circuit mode during RUN .....	7-37
7.12.2 Writing a batch of files during RUN .....	7-40
7.13 Execution Time Measurement.....	7-42
7.13.1 Program monitor list .....	7-42
7.13.2 Interrupt program monitor list .....	7-46
7.13.3 Scan time measurement .....	7-47
7.14 Sampling Trace Function.....	7-49
7.15 Debug Function with Multiple Users.....	7-58
7.15.1 Multiple-user monitoring function.....	7-59
7.15.2 Multiple-user RUN write function .....	7-60
7.16 Watch Dog Timer (WDT).....	7-62
7.17 Self-Diagnosis Function.....	7-64
7.17.1 Interrupt due to error occurrence.....	7-67
7.17.2 LED display when error occurs.....	7-67
7.17.3 Error caucellation .....	7-68
7.18 Failure History .....	7-69
7.19 System Protect.....	7-70
7.19.1 Password registration.....	7-70
7.19.2 Remote password .....	7-72
7.20 Monitoring High Performance model QCPU System Status from GX Developer (System Monitor) .....	7-75
7.21 LED Display.....	7-79
7.21.1 LED display .....	7-79
7.21.2 Priority setting.....	7-81
7.22 High Speed Interrupt Function .....	7-83
7.22.1 High speed interrupt program execution .....	7-85
7.22.2 High speed I/O refresh, high speed buffer transfer.....	7-86
7.22.3 Processing time.....	7-88
7.22.4 Restrictions.....	7-90
7.23 Module Service Interval Time Reading .....	7-93

8 COMMUNICATION WITH INTELLIGENT FUNCTION MODULE/SPECIAL FUNCTION MODULE

8- 1 to 8- 9

8.1 Communication Between High Performance model QCPU and Q-series Intelligent Function Modules..... 8- 1

8.1.1 Initial setting and automatic refresh setting using GX Configurator ..... 8- 2

8.1.2 Communication using device initial value..... 8- 3

8.1.3 Communication using FROM/TO instruction ..... 8- 4

8.1.4 Communication using the intelligent function module device ..... 8- 4

8.1.5 Communication using the instructions dedicated for intelligent function modules..... 8- 5

8.2 Request from Intelligent Function Module to High Performance model QCPU ..... 8- 6

8.2.1 Interrupt from the intelligent function module ..... 8- 6

8.3 Communication Between High Performance model QCPU and AnS Series Special Function Modules..... 8- 7

8.3.1 Communication using device initial value..... 8- 7

8.3.2 Communication using FROM/TO instruction ..... 8- 8

8.3.3 Communication using the intelligent function module device ..... 8- 8

8.3.4 Effects of quicker access to the special function module and countermeasures against them ..... 8- 9

9 PARAMETER LIST

9- 1 to 9-12

10 EXPLANATION OF DEVICES

10- 1 to 10-72

10.1 Device List ..... 10- 1

10.2 Internal User Devices ..... 10- 3

10.2.1 Inputs (X) ..... 10- 5

10.2.2 Outputs (Y) ..... 10- 8

10.2.3 Internal relays (M) ..... 10-10

10.2.4 Latch relays (L) ..... 10-11

10.2.5 Annunciators (F) ..... 10-12

10.2.6 Edge relay (V) ..... 10-16

10.2.7 Link relays (B) ..... 10-17

10.2.8 Link special relays (SB) ..... 10-18

10.2.9 Step relays (S) ..... 10-18

10.2.10 Timers (T) ..... 10-19

10.2.11 Counters (C) ..... 10-24

10.2.12 Data registers (D) ..... 10-29

10.2.13 Link registers (W) ..... 10-30

10.2.14 Link special registers (SW) ..... 10-31

10.3 Internal System Devices ..... 10-32

10.3.1 Function devices (FX, FY, FD) ..... 10-32

10.3.2 Special relays (SM) ..... 10-34

10.3.3 Special registers (SD) ..... 10-35

10.4 Link Direct Devices (J[ ]\[ ]) ..... 10-36

10.5 Intelligent Function Module Devices (U[ ]\G[ ]) ..... 10-39

10.6 Index Registers (Z) .....	10-40
10.6.1 Switching between scan execution type programs and low speed execution type programs ...	10-41
10.6.2 Switching between scan/low speed execution type programs and interrupt/ fixed scan execution type programs .....	10-42
10.7 File Registers (R) .....	10-44
10.7.1 File register capacity .....	10-45
10.7.2 Differences in memory card access method by memory card type .....	10-45
10.7.3 Registering the file registers .....	10-46
10.7.4 File register designation method .....	10-50
10.7.5 Precautions in using file registers .....	10-51
10.8 Nesting (N) .....	10-53
10.9 Pointers (P) .....	10-54
10.9.1 Local pointers .....	10-54
10.9.2 Common pointers .....	10-55
10.10 Interrupt Pointers (I) .....	10-57
10.11 Other Devices .....	10-59
10.11.1 SFC block device (BL) .....	10-59
10.11.2 SFC transition device (TR) .....	10-59
10.11.3 Network No. designation device (J) .....	10-59
10.11.4 I/O No. designation device (U) .....	10-60
10.11.5 Macro instruction argument device (VD) .....	10-61
10.12 Constants .....	10-62
10.12.1 Decimal constants (K) .....	10-62
10.12.2 Hexadecimal constants (H) .....	10-62
10.12.3 Real numbers (E) .....	10-63
10.12.4 Character string ( " " ) .....	10-63
10.13 Convenient Uses for Devices .....	10-64
10.13.1 Global devices and local devices .....	10-64
10.13.2 Device initial values .....	10-70

<b>11 HIGH PERFORMANCE MODEL QCPU PROCESSING TIME</b>	<b>11- 1 to 11- 4</b>
---	-----------------------

11.1 Reading High Performance model QCPU's Scan Time .....	11- 1
11.2 Factors Responsible for Extended Scan Time .....	11- 2
11.3 Factors Responsible for Shortened Scan Time .....	11- 4

<b>12 PROCEDURE FOR WRITING PROGRAMS TO HIGH PERFORMANCE MODEL QCPU</b>	<b>12- 1 to-12- 8</b>
---	-----------------------

12.1 Writing Procedure for 1 Program .....	12- 1
12.1.1 Items to consider when creating one program .....	12- 1
12.1.2 Procedure for writing programs to the High Performance model QCPU .....	12- 2
12.2 Procedure for Multiple Programs .....	12- 5
12.2.1 Items to consider when creating multiple programs .....	12- 5
12.2.2 Procedure for writing programs to the High Performance model QCPU .....	12- 6

<b>13 OUTLINE OF MULTIPLE CPU SYSTEM</b>	<b>13- 1 to 13- 6</b>
13.1 Features.....	13- 1
13.2 Outline of Multiple CPU System .....	13- 3
13.3 Differences with Single CPU System .....	13- 5
<b>14 SYSTEM CONFIGURATION OF MULTIPLE CPU SYSTEM</b>	<b>14- 1 to 14- 21</b>
14.1 System Configuration.....	14- 1
14.2 Precautions For Multiple CPU System Configuration .....	14- 5
14.2.1 Function versions of High Performance model QCPU , motion CPUs and PC CPU module that can be used, and their mounting positions.....	14- 5
14.2.2 Precautions when using Q series corresponding I/O modules and intelligent function modules.....	14- 9
14.2.3 Limitations when mounting AnS series corresponding I/O modules and special function modules.....	14-10
14.2.4 Modules that have mounting restrictions.....	14-12
14.2.5 Compatible GX Developers and GX Configurators .....	14-13
14.2.6 Parameters that enable the use of multiple CPU system.....	14-14
14.2.7 Resetting the multiple CPU system.....	14-18
14.2.8 Processing when High Performance model QCPU stop errors occur .....	14-19
14.2.9 Reducing the time required for multiple CPU system processing.....	14-21
<b>15 ALLOCATING MULTIPLE CPU SYSTEM I/O NUMBERS</b>	<b>15- 1 to 15- 3</b>
15.1 Concept behind Allocating I/O Numbers .....	15- 1
15.1.1 I/O modules and intelligent function module I/O numbers.....	15- 1
15.1.2 I/O number of High Performance model QCPU, Motion CPU and PC CPU module.....	15- 2
15.2 Setting of Control CPUs with GX Developer .....	15- 3
<b>16 COMMUNICATION BETWEEN CPU MODULES IN MULTIPLE CPU SYSTEM</b>	<b>16- 1 to 16-15</b>
16.1 Automatic Refresh of CPU Shared Memory .....	16- 2
16.2 Communication with Multiple CPU Dedicated Instructions and Intelligent Function Module Devices..	16- 9
16.3 Interactive Communications between The High Performance model QCPU and Motion CPU .....	16-11
16.3.1 Control commands from the High Performance model QCPU to the Motion CPU .....	16-11
16.3.2 Reading and writing device data .....	16-12
16.4 CPU Shared Memory.....	16-13
<b>17 COMMUNICATIONS BETWEEN THE MULTIPLE CPU SYSTEM'S I/O MODULES AND INTELLIGENT FUNCTION MODULES</b>	<b>17- 1 to 17- 5</b>
17.1 Range of Control CPU Communications.....	17- 1
17.2 Range of Non-control CPU Communications .....	17- 1

18 PROCESSING TIME FOR MULTIPLE CPU SYSTEM HIGH PERFORMANCE MODEL QCPUs	18- 1 to 18- 3
--	----------------

- 18.1 Concept behind CPU Scanning Time ..... 18- 1
- 18.2 Factor to Prolong the Scan Time ..... 18- 2

19 STARTING UP THE MULTIPLE CPU SYSTEM	19- 1 to 19- 9
--	----------------

- 19.1 Flow-chart for Starting Up the Multiple CPU System ..... 19- 1
- 19.2 Setting Up the Multiple CPU System Parameters (Multiple CPU Settings, Control CPU Settings) ..... 19- 3
  - 19.2.1 System configuration..... 19- 3
  - 19.2.2 Creating new systems..... 19- 4
  - 19.2.3 Using existing preset multiple CPU settings and I/O allocations..... 19- 7

APPENDICES	App- 1 to App-60
------------	------------------

- APPENDIX 1 Special Relay List..... App- 1
- APPENDIX 2 Special Register List ..... App-22
- APPENDIX 3 List of Interrupt Pointer Nos. & Interrupt Factors ..... App-58
- APPENDIX 4 Enhancement of the High Performance Model QCPU Functions ..... App-59
  - APPENDIX 4.1 Specification Comparison .....App-59
  - APPENDIX 4.2 Function Comparison .....App-59
  - APPENDIX 4.3 Usability of Added Functions Depending on GX Developer Version .....App-60
- APPENDIX 5 Transportation Precautions..... App-61
  - APPENDIX 5.1 Controlled models.....App-61
  - APPENDIX 5.2 Transport guidelines .....App-61

INDEX	Index- 1 to Index- 3
-------	----------------------

## CONTENTS

### 1. OVERVIEW

#### 1.1 Features

### 2. SYSTEM CONFIGURATION FOR SINGLE CPU SYSTEM

#### 2.1 System Configuration

#### 2.2 Precaution on System Configuration

#### 2.3 Confirming Serial Number and Function Version

### 3. GENERAL SPECIFICATIONS

### 4. HARDWARE SPECIFICATION OF THE CPU MODULE

#### 4.1 Performance Specification

#### 4.2 Part Names and Settings

#### 4.3 Switch Operation After Writing in Program

#### 4.4 Latch Clear Operation

#### 4.5 Executing automatic Write to standard ROM

### 5. POWER SUPPLY MODULE

#### 5.1 Specification

##### 5.1.1 Power supply module specifications

##### 5.1.2 Selecting the power supply module

##### 5.1.3 Precaution when connecting the uninterruptive power supply

#### 5.2 Names of Parts and Settings

### 6. BASE UNIT AND EXTENSION CABLE

#### 6.1 Base Unit Specification Table

#### 6.2 Extension Cable Specification Table

#### 6.3 Parts Names of Base Unit

#### 6.4 Setting the Extension Base Unit

#### 6.5 Guideline for Use of Extension Base Units (Q5 □ B)



## 7. MEMORY CARD AND BATTERY

- 7.1 Memory Card Specifications
- 7.2 Battery Specifications (For CPU Module and SRAM Card)
- 7.3 Handling the Memory Card
- 7.4 The Names of The Parts of The Memory Card
- 7.5 Memory Card Loading/Unloading Procedures
- 7.6 Installation of Battery (for CPU Module and Memory Card)

## 8. EMC AND LOW VOLTAGE DIRECTIVE

- 8.1 Requirements for Conformance to EMC Directive
  - 8.1.1 Standards applicable to the EMC Directive
  - 8.1.2 Installation instructions for EMC Directive
  - 8.1.3 Cables
  - 8.1.4 Power supply module
  - 8.1.5 When using Q1AS6  B type base unit
  - 8.1.6 Others
- 8.2 Requirement to Conform to the Low Voltage Directive
  - 8.2.1 Standard applied for MELSEC-Q series PLC
  - 8.2.2 MELSEC-Q series PLC selection
  - 8.2.3 Power supply
  - 8.2.4 Control box
  - 8.2.5 Grounding
  - 8.2.6 External wiring

## 9. LOADING AND INSTALLATION

- 9.1 General Safety Requirements
- 9.2 Calculating Heat Generation by PLC
- 9.3 Module Installation
  - 9.3.1 Precaution on installation
  - 9.3.2 Instructions for mounting the base unit
  - 9.3.3 Installation and removal of module
- 9.4 How to set Stage Numbers for the Extension Base Unit
- 9.5 Connection and Disconnection of Extension Cable
- 9.6 Wiring
  - 9.6.1 The precautions on the wiring
  - 9.6.2 Connecting to the power supply module

## 10. MAINTENANCE AND INSPECTION

- 10.1 Daily Inspection
  - 10.2 Periodic Inspection
  - 10.3 Battery Replacement
    - 10.3.1 Battery service life
    - 10.3.2 Battery replacement procedure
  - 10.4 When Resuming Operation after Storage of PLC without Battery
  - 10.5 When Resuming PLC Operation after Storage of PLC with Battery Gone Flat
- A - 16

## 11. TROUBLESHOOTING

### 11.1 Troubleshooting Basics

### 11.2 Troubleshooting

#### 11.2.1 Troubleshooting flowchart

#### 11.2.2 Flowchart for when "MODE" LED is not turned on

#### 11.2.3 Flowchart for when "MODE" LED is flickering

#### 11.2.4 Flowchart for when "POWER" LED is turned off

#### 11.2.5 Flowchart for when the "RUN" LED is turned off

#### 11.2.6 When the "RUN" LED is flickering

#### 11.2.7 Flowchart for when "ERR." LED is on/flickering

#### 11.2.8 When "USER" LED is turned on

#### 11.2.9 When "BAT." LED is turned on

#### 11.2.10 Flowchart for when "BOOT" LED is flickering

#### 11.2.11 Flowchart for when output module LED is not turned on

#### 11.2.12 Flowchart for when output load of output module does not turn on

#### 11.2.13 Flowchart for when unable to read a program

#### 11.2.14 Flowchart for when unable to write a program

#### 11.2.15 Flowchart for when it is unable to perform boot operation from memory card

#### 11.2.16 Flowchart for when UNIT VERIFY ERR. occurs

#### 11.2.17 Flowchart for when CONTROL BUS ERR. occurs

### 11.3 Error Code List

#### 11.3.1 Procedure for reading error codes

#### 11.3.2 Error code list

### 11.4 Canceling of Errors

### 11.5 I/O Module Troubleshooting

#### 11.5.1 Input circuit troubleshooting

#### 11.5.2 Output circuit troubleshooting

### 11.6 Special Relay List

### 11.7 Special Register List

## APPENDICES

### APPENDIX 1 Error Code Return to Origin During General Data Processing

#### APPENDIX 1.1 Error code overall explanation

#### APPENDIX 1.2 Description of the errors of the error codes (4000H to 4FFFH)

### APPENDIX 2 External Dimensions Diagram

#### APPENDIX 2.1 CPU module

#### APPENDIX 2.2 Power supply module

#### APPENDIX 2.3 Main base unit

#### APPENDIX 2.4 Slim type main base unit

#### APPENDIX 2.5 Extension base unit

### APPENDIX 3 Upgraded Functions of High Performance model QCPU

#### APPENDIX 3.1 Specification comparison

#### APPENDIX 3.2 Function comparison

#### APPENDIX 3.3 Usability of added functions depending on GX Developer version

APPENDIX 4 Transportation Precautions

APPENDIX 4.1 Controlled models

APPENDIX 4.2 Transport guidelines

INDEX
-------

## Manuals

The following manuals are also related to this product.

In necessary, order them by quoting the details in the tables below.

### **Related Manuals**

Manual Name	Manual Number (Model Code)
High Performance Model QCPU (Q Mode) 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-080037 (13JL97)
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 (PID Control Instructions) This manual describes the dedicated instructions used to exercise PID control. (Sold separately)	SH-080040 (13JF59)
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)/QnACPU 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)

## How to Use This Manual

This manual is prepared for users to understand memory map, functions, programs and devices of the CPU module when you use MELSEC-Q Series PLCs.

The manual is classified roughly into three sections as shown below.

- |                       |   |
|-----------------------|---|
| (1) Chapters 1 and 2  | Describe the outline of the CPU module and the system configuration. The feature of CPU module and the basics of the system configuration of CPU module are described.                                    |
| (2) Chapters 3 to 6   | Describe the performance specifications, executable program, I/O No. and memory of the CPU module.  |
| (3) Chapter 7         | Describes the functions of the CPU modules.   |
| (4) Chapter 8         | Describes communication with intelligent function modules.  |
| (5) Chapters 9 and 10 | Describe parameters and devices used in the CPU modules.  |
| (6) Chapter 11        | Describes the CPU module processing time.   |
| (7) Chapter 12        | Describes the procedure for writing parameters and programs created at the GX Developer to the CPU module.  |
| (8) Chapters 13 to 19 | Describes an overview of the multiple PLC system, the system configuration, the I/O numbers, communications between CPU modules, and communications between I/O modules and intelligent function 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 these functions, refer to the manual shown below.

- High Performance Model QCPU (Q Mode) User's Manual (Hardware Design, Maintenance and Inspection)

## Generic Terms and Abbreviations

The following abbreviations and general names for Q02CPU, Q02HCPU, Q06HCPU Q12HCPU, and Q25HCPU are used in the manual.

Generic Term/Abbreviation	Description
High Performance model QCPU	General name for Q02CPU, Q02HCPU, Q06HCPU Q12HCPU and Q25HCPU modules.
Q Series	Abbreviation for Mitsubishi MELSEC-Q Series Programmable Logic Controller.
AnS Series	Abbreviation for small types of Mitsubishi MELSEC-A Series Programmable Logic Controller.
GX Developer	General product name for SWnD5C-GPPW-E, SWnD5C-GPPW-A-E, SWnD5C-GPPW-V-E and SWnD5C-GPPW-VA-E. For QCPU, version 4 or later can be used.
Q3□B	General name for Q33B, Q35B, Q38B and Q312B type main base units that accept High Performance model QCPU, Q series power supply module, input/output modules and intelligent function modules.
Q3□SB	General name for Q32SB, Q33SB and Q35SB type slim type main base units that accept High Performance model QCPU, slim type power supply module, input/output modules and intelligent function modules.
Q5□B	General name for Q52B and Q55B that accept the Q Series I/O and intelligent function modules.
Q6□B	General name for Q63B, Q65B, Q68B and Q612B type extension base unit with Q Series power supply module, I/O module, intelligent function module attachable.
QA1S6□B	General name for QA1S65B and QA1S68B type extension base unit with AnS Series power supply module, I/O module, special function module attachable.
QnCPU	General name for Q02CPU.
QnHCPU	General name for Q02HCPU, Q06HCPU Q12HCPU and Q25HCPU
Main base unit	General name for Q33B, Q35B, Q38 and Q312B type main base unit with Q Series power supply module, I/O module, intelligent function module attachable.
Slim type main base unit	General name for Q32SB, Q33SB and Q35SB type slim type main base units that accept High Performance model QCPU, slim type power supply module, input/output modules and intelligent function modules.
Extension base unit	General name for Q5□B, Q6□B and QA1S6□B.
Base unit	General name for main base unit and extension base unit.
Extension cable	General name for QC05B, QC06B, QC12B, QC30B, QC50B and QC100B type extension cable.
Power supply module	General name for Q61P-A1, Q61P-A2, Q62P, Q63P, Q64P, A1S61PN, A1S62P and A1S63P type power supply module.
Slim type power supply module	General name for Q61SP type slim type power supply module.
Battery	General name for battery for Q6BAT and Q7BAT type CPU module and Q2MEM-BAT type SRAM card.
SRAM card	Abbreviation for Q2MEM-1MBS and Q2MEM-2MBS type SRAM card.
Flash card	General name for Q2MEM-2MBF and Q2MEM-4MBF type Flash card.
ATA card	General name for Q2MEM-8MBA, Q2MEM-16MBA and Q2MEM-32MBA type ATA card.
Memory card	General name for SRAM card, Flash card and ATA card.
CPU slot	Slot on the right of the power supply module of the main base unit.
Control CPU	High Performance model QCPU/motion CPU that controls any of the I/O and intelligent function modules mounted on the main or extension base unit. For example, when the module mounted on slot 3 is controlled by the CPU No. 2, the CPU No. 2 is the control CPU of the module on slot 3.
Non-controlled module (Non-group module)	I/O or intelligent function module other than the controlled module. For example, when the module mounted on slot 3 is controlled by the CPU No. 2, the module on slot 3 is the non-controlled module of the CPU Nos. 1, 3 and 4.

Generic Term/Abbreviation	Description
Controlled module	I/O or intelligent function module controlled by the control CPU. For example, when the module mounted on slot 3 is controlled by the CPU No. 2, the module on slot 3 is the controlled module of the CPU No. 2.
CPU numbers	Numbers assigned to differentiate between the High Performance model QCPU and motion CPU mounted in a multi CPU system. The CPU on the CPU slot is the CPU No. 1, the one on slot 0 is the CPU No. 2, the one on slot 1 is the CPU No. 3, and the one on slot 2 is the CPU No. 4.
Single CPU system	System mounted with the High Performance model QCPU on the CPU slot to exercise control.
PC CPU module	MELSEC-Q Series corresponding PC CPU module
Non-control CPUs	High Performance model QCPUs and/or Motion CPUs other than the control CPU. For example, when the module mounted on slot 3 is controlled by the CPU No. 2, the CPU Nos. 1, 3 and 4 are the non-control CPU of the module on slot 3.
Multiple CPU system	System mounted with up to four High Performance model QCPU, Motion CPU and PC CPU module on the main base unit to exercise control.





1. OVERVIEW

1

This manual describes the functions, programs and devices of the High Performance model QCPU.

Refer to the following manual for the specifications, etc. of the power supply modules, base units, extension cables, memory cards and battery.  
 High Performance Model QCPU (Q Mode) User's Manual (Hardware Design, Maintenance and Inspection)

Functions are added when the High Performance model QCPU is updated. The added functions can be discriminated by the function version/serial number of the CPU module.

Table 1.1 gives the added functions and the corresponding GX Developer versions. When using the added function, confirm the function version/serial number and the GX Developer version.

Table 1.1 List of Functions Added to High Performance Model QCPU and Function Versions/Serial Numbers

Update Details of High Performance Model QCPU			Corresponding GX Developer
Function version	Serial No.	Added functions	
A	"02092" or later	<ul style="list-style-type: none"> <li>• Automatic write to standard ROM</li> <li>• Enforced ON/OFF for external I/O</li> <li>• Remote password setting</li> <li>• Increased standard RAM capacity of Q12HCPU, Q25HCPU</li> <li>• Compatibility with MELSECNET/H remote I/O network</li> <li>• Interrupt module (QI60) compatibility</li> </ul>	Version 6 or later
B	—	<ul style="list-style-type: none"> <li>• Compatibility with the multiple PLC system</li> </ul>	Version 6 or later
	"03051" or later	<ul style="list-style-type: none"> <li>• Installation of PC CPU module into the multiple PLC system</li> </ul>	Version 7 or later
	"04012" or later	<ul style="list-style-type: none"> <li>• High speed interrupt function</li> <li>• Compatibility with index modification for module designation of dedicated instruction</li> <li>• Selection of refresh item for COM instruction</li> <li>• Extended life battery of SRAM card</li> <li>• Compatibility with 2Mbyte SRAM card</li> <li>• Increased standard RAM capacity of Q02HCPU, Q06HCPU</li> </ul>	Version 7.10L or later
	"04122" or later	<ul style="list-style-type: none"> <li>• SFC program online batch change</li> <li>• File memory capacity change</li> </ul>	Version 8 or later
	"05032" or later	<ul style="list-style-type: none"> <li>• CC-Link remote network additional mode</li> <li>• Incomplete derivative PID operation function</li> <li>• Floating-point comparison instruction speedup</li> </ul>	Version 8.03D or later

**POINT**

- (1) Refer to Section 2.3 for the serial No. and function version of the High Performance model QCPU.
- (2) Refer to Appendix 4 for details.

## 1.1 Features

High Performance model QCPU has the following new features:

- (1) **Controllable multiple I/O points**  
 All High Performance model QCPUs support 4096 points (X/Y0 to FFF) as the number of actual I/O points capable of getting access to the I/O module installed on the base unit.  
 They also support 8192 points max. (X/Y0 to 1FFF) as the number of I/O devices which can be used in the remote I/O stations such as MELSECNET/H remote I/O NET, CC-Link data link and MELSECNET/MINI-S3 data link.
- (2) **Lineup according to program capacity**  
 The optimum CPU module for the program capacity to be used can be selected.  
 Q02CPU, Q02HCPU : 28k step  
 Q06HCPU : 60k step  
 Q12HCPU : 124k step  
 Q25HCPU : 252k step
- (3) **Realised high speed processing**  
 Depending on the type of the sequencer, high speed processing has been realized.(Example: when LD instruction is used)  
 Q02CPU : 0.079  $\mu$ s  
 Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU : 0.034  $\mu$ s

In addition, an access to the intelligent function module or an increase in speed of the link refresh of the network have been realized by the connection system (System bus connection) of the newly developed base unit.

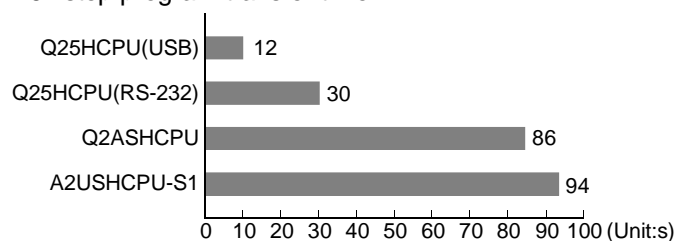
Access to the intelligent function module : 20  $\mu$ s /word (approx. 7 times)\*1

MELSECNET/H link refresh processing : 4.6ms/8k word (approx. 4.3 times)\*1

\*1: Where Q02HCPU is compared with Q2ASHCPU-S1.

- (4) **Increase in debugging efficiency through high speed communication with GX Developer**  
 In the High Performance model QCPU, a time required for writing/reading of a program or monitoring has been reduced through the high speed communication at a speed of 115.2kbps max. by the RS-232, and a communication time efficiency at the time of debugging has been increased.  
 In the Q02HCPU, Q06HCPU, Q12HCPU, and Q25HCPU, a high speed communication at a speed of 12Mbps is allowed through the USB.

26k step program transfer time

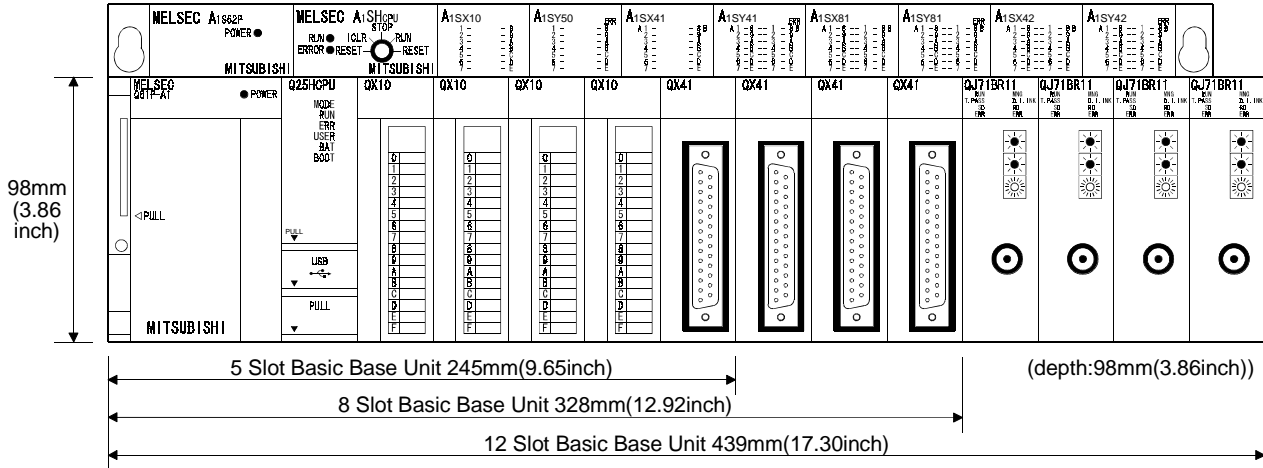


- (5) **AnS series I/O module or special function module are available.**  
 For Q series, if an appropriate module is not available, the AnS series I/O module or special function module can also be used for the High Performance model QCPU through the use of the QA1S65B/QA1S68B extension base unit.

(6) Saved space by a reduction in size

The installation space for Q series has been reduced by approx. 60 % of the space for AnS series.

Comparison of installation space



(7) Connection of up to seven extension base units.

- (a) The High Performance model QCPU can connect to seven extension base units (eight base units including the main) and accept up to 64 modules.
- (b) The overall distance of the extension cables is up to 13.2m to ensure high degree of extension base unit arrangement.

(8) Memory extension by memory card

The High Performance model QCPU is provided with a memory card installation connector to which a memory card of 32 Mbyte max. can be connected (32 Mbyte is available when a ATA card is used). When a memory card of large capacity is installed, a large capacity of file can be controlled, comments to all data devices can be set up, and the programs in the past can be stored in the memory as they are in the form of the corrected histories. If a memory card is not installed, a program can be stored onto the standard ROM built in the CPU module, and file registers can be handled by the standard RAM.

**REMARK**

- The number of file registers that can be handled changes depending on the function version/serial number of the CPU module used.

CPU Module Type		Number of File Registers
Q02HCPU		32k points
Q06HCPU	First 5 digits of serial number are "04011" or earlier	32k points
Q06HCPU	First 5 digits of serial number are "04012" or later	64k points
Q12HCPU	First 5 digits of serial number are "02091" or earlier	32k points
Q25HCPU	First 5 digits of serial number are "02092" or later	128k points

Refer to Section 2.3 to confirm the function version and serial number of the High Performance model QCPU.

- (9) **Data can be written automatically to standard ROM**  
You need not use GX Developer to write parameters/programs on a memory card to the standard ROM of the High Performance model QCPU. When the standard ROM is used to perform ROM operation, you can load a memory card into the High Performance model QCPU and write parameters/programs on the memory card to the standard ROM. Hence, you need not carry GX Developer (personal computer) to rewrite the parameters/programs.
- (10) **External I/O can be turned ON/OFF forcibly**  
If the High Performance model QCPU is in the RUN mode, you can operate GX Developer to turn external inputs/outputs ON/OFF forcibly, independently of the program execution status.  
You need not put the High Performance model QCPU in the STOP mode to perform wiring/operation tests by forced ON/OFF of outputs.
- (11) **Remote password can be set**  
When access to an Ethernet module or serial communication module is made externally, whether access to the High Performance model QCPU can be made or not can be selected with a remote password.
- (12) **Remote I/O network of MELSECNET/H can be configured**  
You can load the remote master station of the MELSECNET/H to configure an MELSECNET/H remote I/O system.
- (13) **PC CPU module compatibility in multiple CPU system configuration**  
The High Performance model QCPU, Motion CPU and PC CPU module can configure a multiple CPU system.
- (14) **Ease of handling CC-Link system**  
When one master module for CC-Link system is used, I/O signals for up to 64 remote I/O stations can be controlled without parameter setting.  
The remote I/O stations can be controlled as if I/O modules on the base unit are controlled.
- (15) **File password for blocking illegal access**  
The file password can be used to set the access level (read disable, write disable) of a program to prevent program file, device initial value file and device comment file changes due to illegal access.

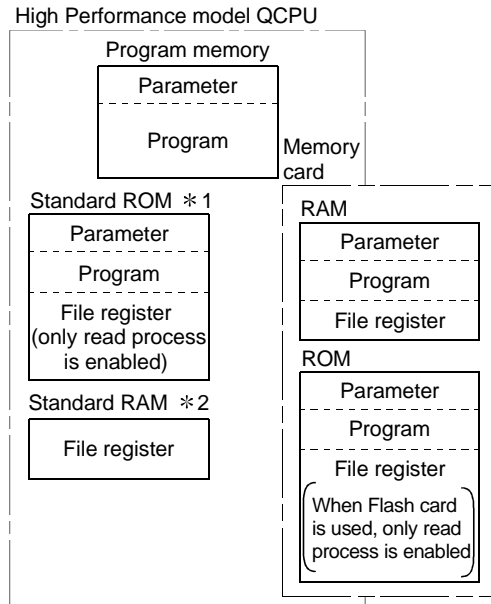
**REMARK**

- 1) Features (9) to (12) are functions added to the High Performance model QCPU whose serial number is "02092" or later in its upper 5 digits.
- 2) The remote password facility can be executed when the Ethernet module or serial communication module of function version B and GX Developer Version 6 or later are used.
- 3) In addition to the remote password, there are the following protection facilities for the High Performance model QCPU.  
Protection of the whole CPU module by making system settings of the High Performance model QCPU  
Protection of the memory card by setting the write protect switch of the memory card  
File-by-file protection using password
- 4) The MELSECNET/H remote I/O network facility can be executed when the MELSECNET/H network module of function version B and GX Developer (Version 6 or later) are used.
- 5) The feature in (13) is the function added to the High Performance model QCPU whose first five digits of serial No. are "03051" or later.

1.2 Programs

(1) Program management by memory card

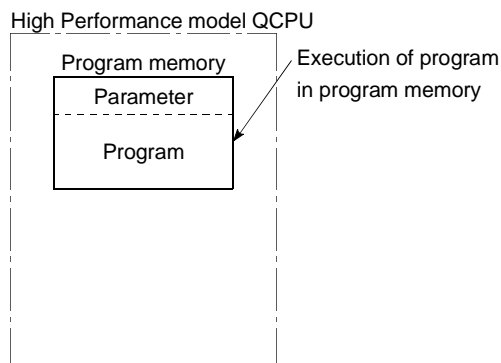
- (a) Programs created with GX Developer can be stored in the QCPU's program memory, standard ROM or memory card.



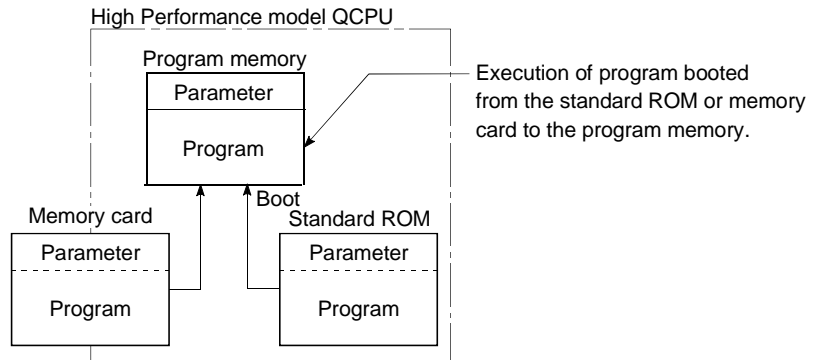
\*1: The standard ROM is used when parameters and programs are written to ROM.

\*2: The standard RAM is used when access to the file register need to speed up.

- (b) The QCPU processes programs stored in the program memory.



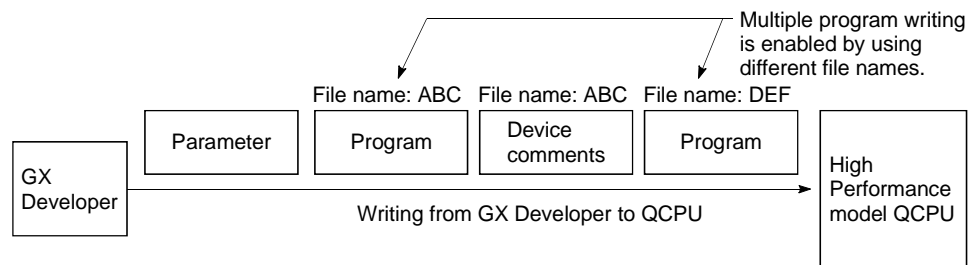
Programs stored in the standard ROM/memory card are executed after they are booted to (read to) the QCPU program memory. (Programs to be booted to the QCPU are designated on the "(PLC) Parameter" dialog box, and the parameter drive is designated by a DIP switch setting at the QCPU.)



(2) Program construction

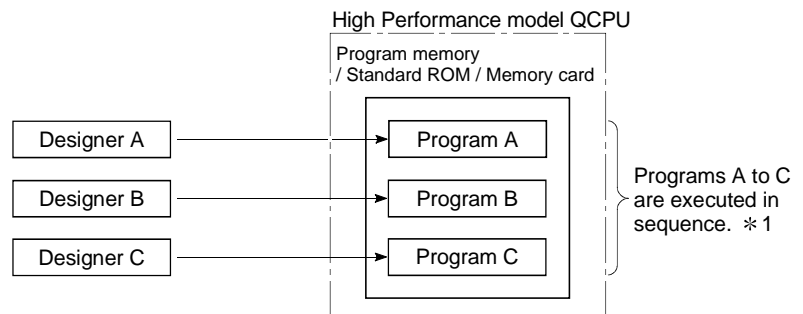
Programs are stored in a file format in the program memory, standard ROM or memory card.

Multiple programs can therefore be stored in the program memory, standard ROM or memory card by using different file names.



Therefore, the program creation can be split among several designers so that they control and manage the programs by process or function. Only the relevant programs should be modified or debugged when the specifications are changed.

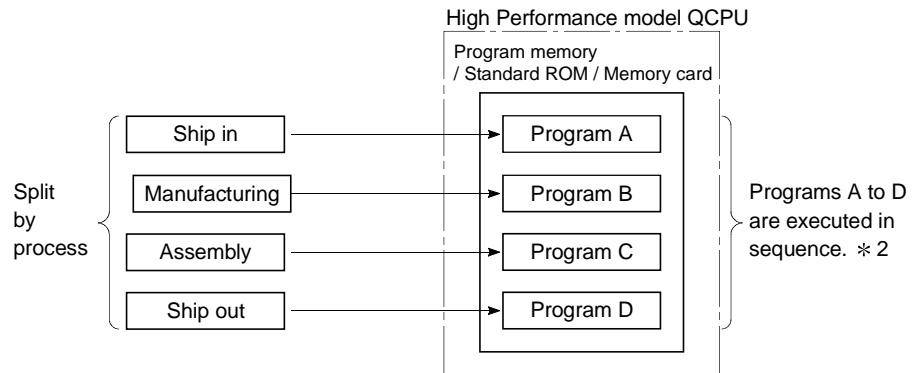
(a) Example of program creation split among several designers:



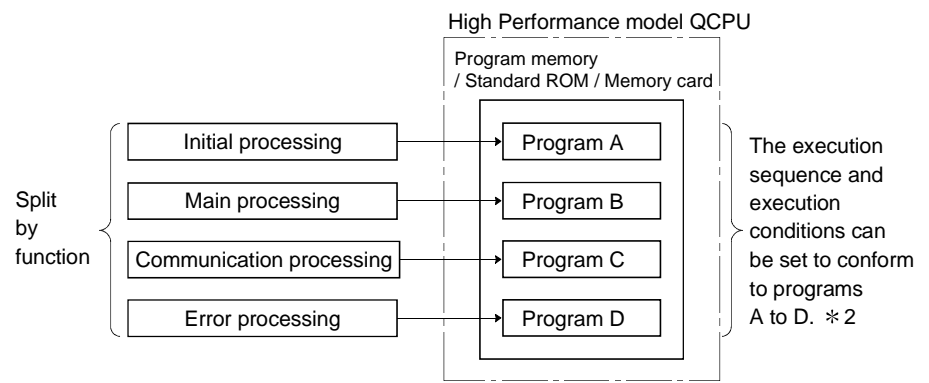
**REMARK**

\*1: See Section 4.2 for details on the execution sequence.

(b) Example of programs split by process: \*1



(c) Example of programs split by function:



**REMARK**

\*1: Programs split by process can be further split by function.

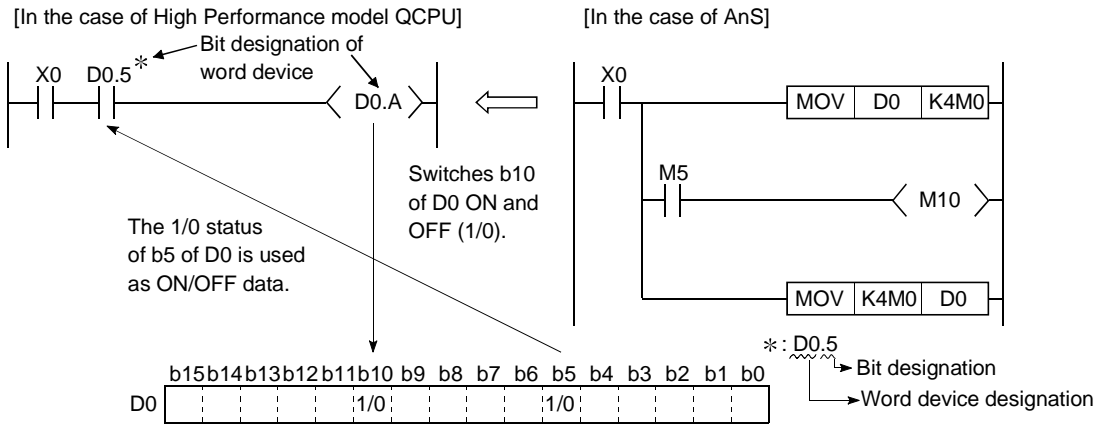
\*2: See Section 4.2 for details on the execution sequence and execution conditions.

### 1.3 Convenient Programming Devices and Instructions

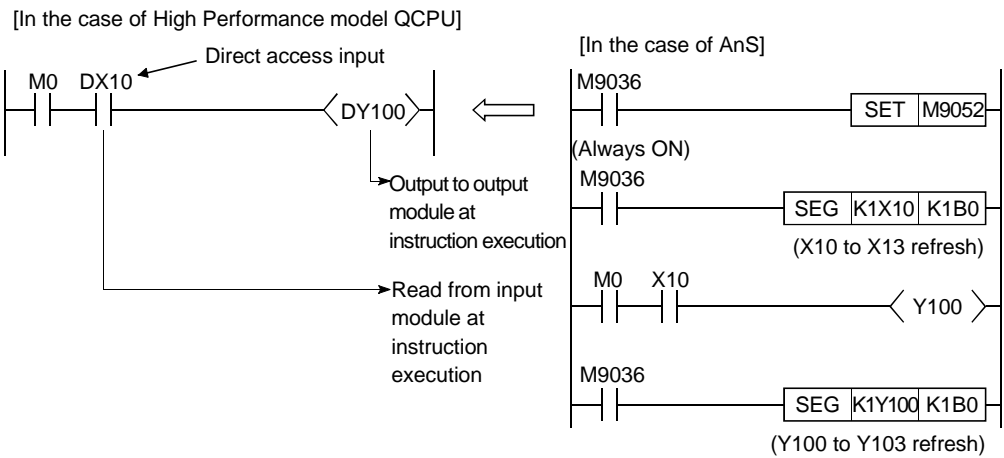
The QCPU features devices and instructions which facilitate program creation. Some of them are described below.

#### (1) Flexible device designation

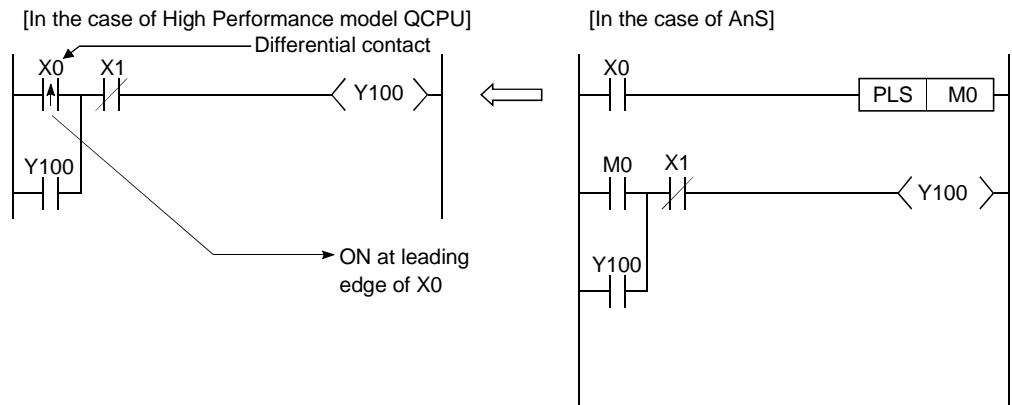
(a) Word device bits can be designated to serve as contacts or coils.



(b) Direct processing in 1-point units is permitted within a program simply by using direct access inputs (DX [ ]) and direct access outputs (DY [ ]).

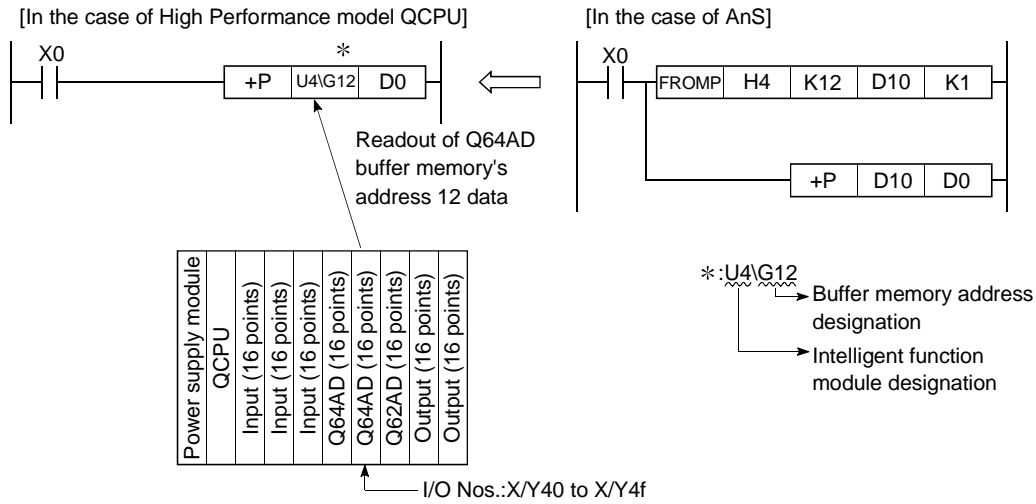


(c) Differential contacts (—|/|—) eliminate the need of converting inputs to pulses.





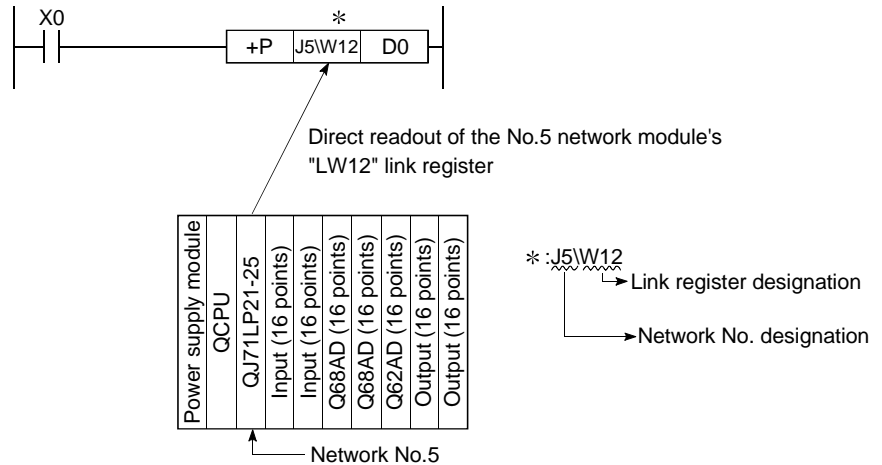
(d) The buffer memory of intelligent function module (e.g. Q64AD, Q62DA) can be used in the same way as devices when programming.



**POINT**

By making the automatic refresh setting of the used intelligent function module using GX Configurator, the corresponding data can be read/written from/to the device memory of the CPU module without access being made directly to the buffer memory.

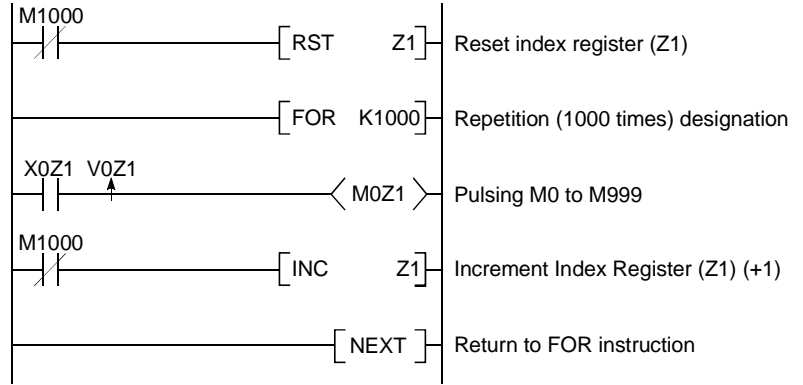
(e) Direct access to link devices (LX, LY, LB, LW, SB, SW) of MELSECNET/H network modules (e.g. QJ71LP21-25) is allowed without refresh settings.



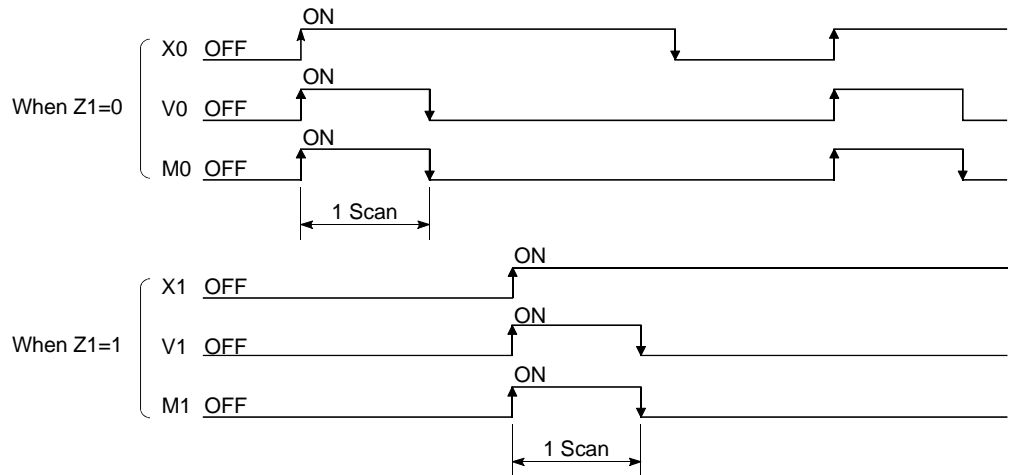
(2) Edge relays simplify pulse conversion processing

- (a) The use of a relay (V) that comes ON at the leading edge of the input condition simplifies pulse processing when a contact index qualification has been made.

[Circuit example]

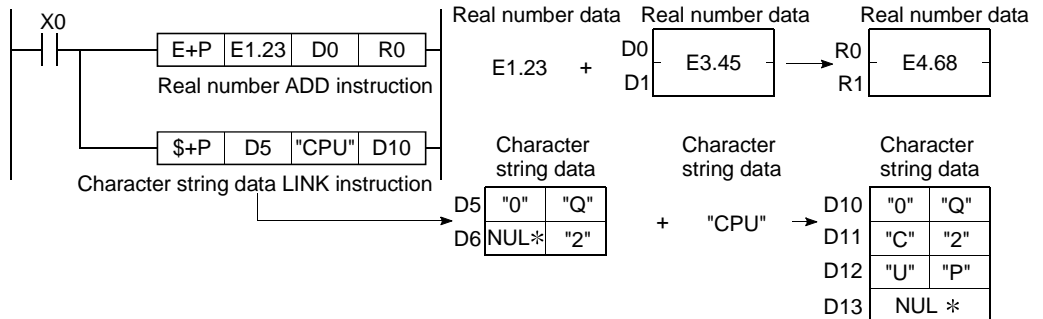


[Timing chart]



(3) Simple data processing

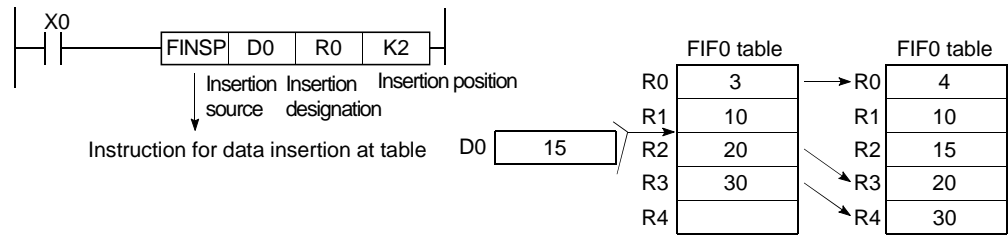
- (a) Real numbers (floating decimal point data) and character string constants can be used in the programming as they are.



**REMARK**

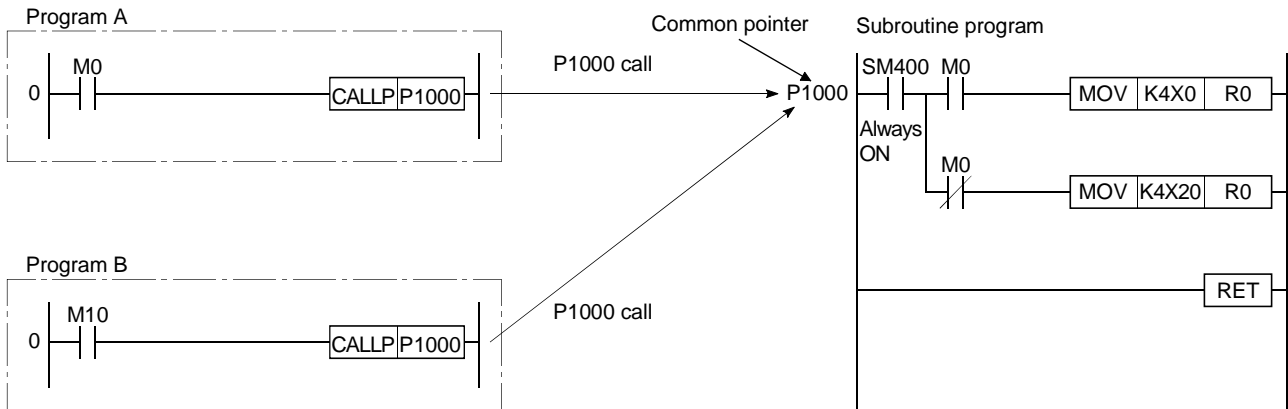
\*: NUL indicates "00H (character string END)".

- (b) Data processing instructions such as table processing instructions, etc., enable high speed processing of large amounts of data.

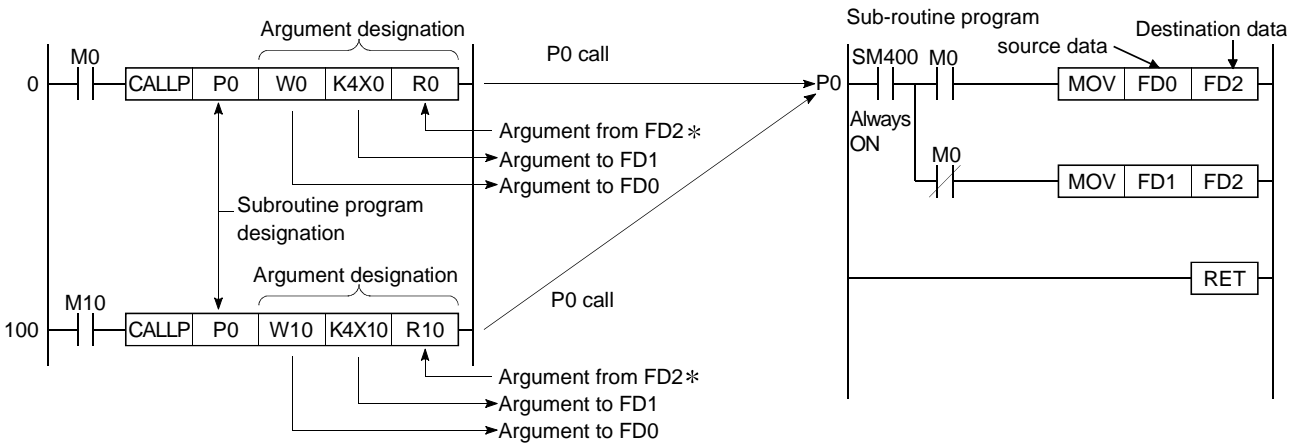


(4) Easy shared use of sub-routine programs

- (a) A common pointer can be used to call the same sub-routine program from all sequence programs being executed.



- (b) The use of sub-routine call instructions with arguments simplifies the creation of sub-routine programs which are called several times.



**REMARK**

\*: For details on the argument I/O condition, see Section 10.3.1.

## 2. SYSTEM CONFIGURATION FOR SINGLE CPU SYSTEM

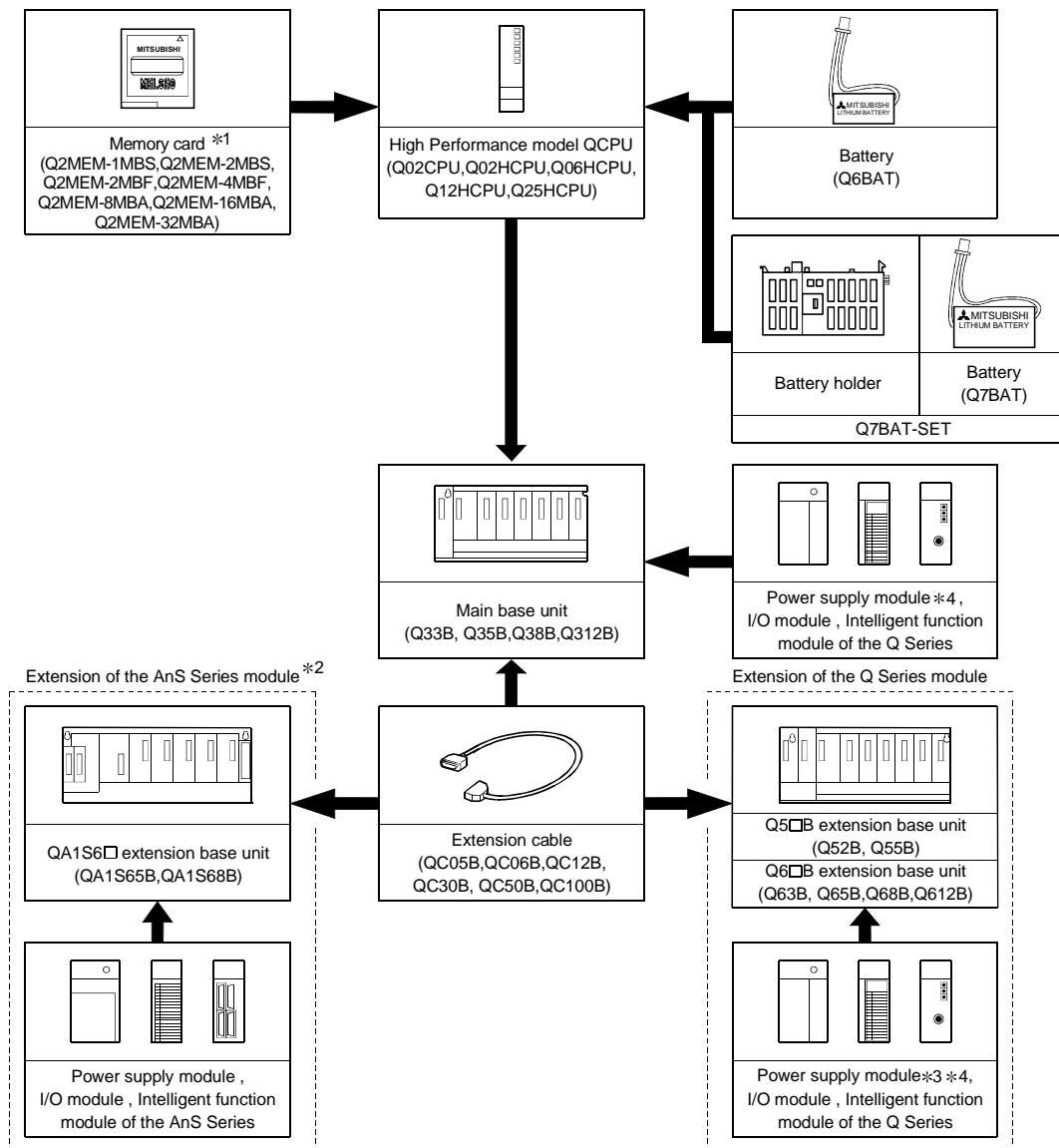
This section describes the system configuration of the High Performance model QCPU, cautions on use of the system, and configured equipment.

### 2.1 System Configuration

The outline of the equipment configuration, configuration with peripheral devices, and system configuration in the High Performance model QCPU system is described below.

#### (1) Equipment configuration

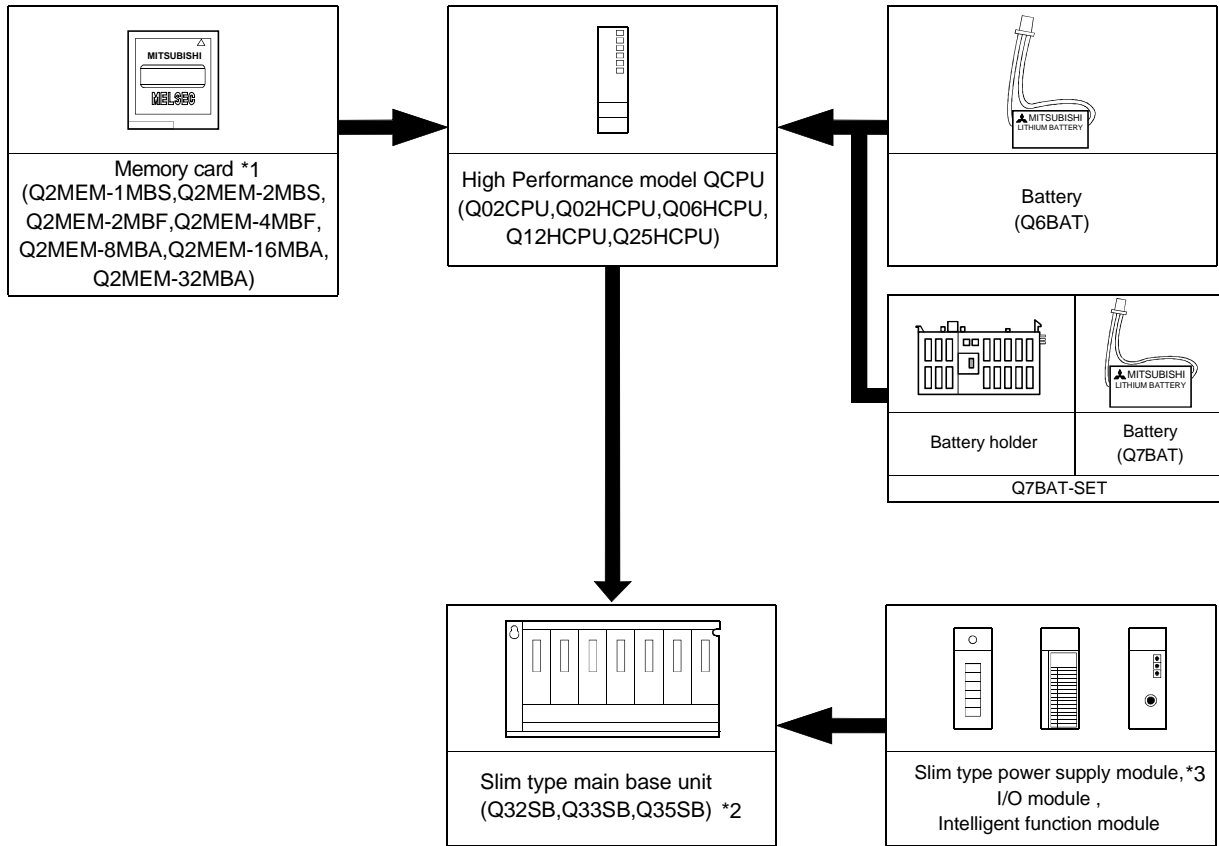
(a) When main base unit (Q3□B) is used



#### POINTS

- \*1: The number of memory cards to be installed is one sheet.  
The memory card must be selected from SRAM, Flash, and ATA according to the application and capacity.  
With commercial memory cards, the Operation is not assured.
- \*2: QA1S65B and QA1S68B extension base units are used for the power supply module, I/O module, and special function module of the AnS series.
- \*3: The Q series power supply module is not required for the Q5□B type extension base unit.
- \*4: As a power supply module, use the Q61P-A1, Q61P-A2, Q62P or Q64P.  
The slim type power supply module (Q61SP) cannot be used.

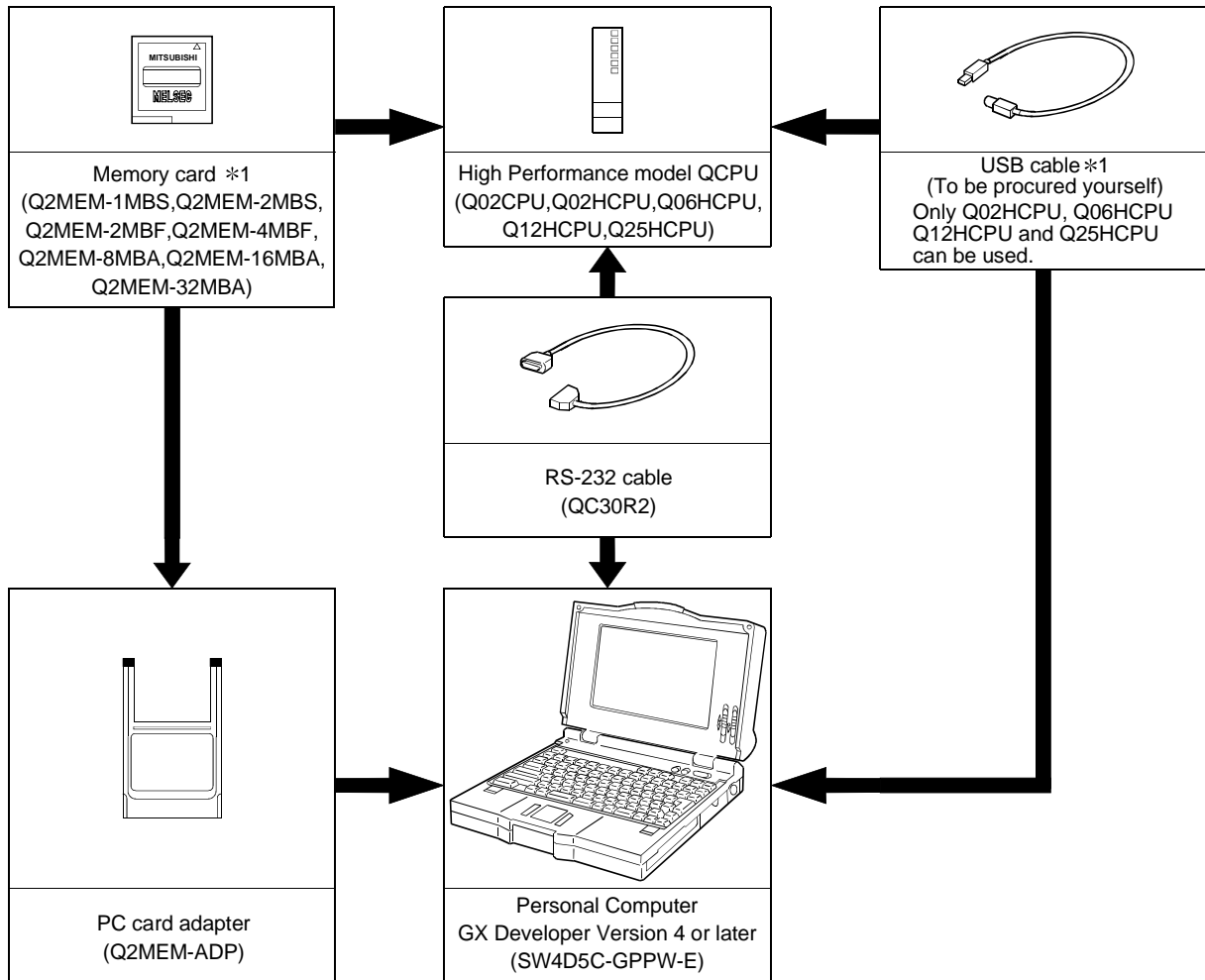
(b) When slim type main base unit (Q3□SB) is used



**POINTS**

- \*1: One memory card is installed.  
Select the memory card from the SRAM card, Flash card and ATA card according to the application and capacity.  
When the memory card available on the market is used, operation is not guaranteed.
- \*2: The slim type main base unit does not have an extension cable connector. The extension base or GOT cannot be connected.
- \*3: As a power supply module, use the slim type power supply module (Q61SP).  
The Q61P-A1, Q61P-A2, Q62P or Q64P cannot be used as a power supply module.

(2) Configuration of peripheral devices



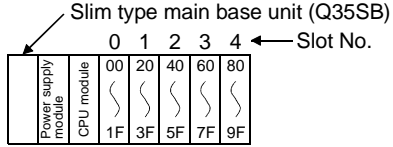
\*1: For writing into memory card on GX Developer, and USB cable, refer to the operating manual of the GX Developer.

(3) Outline of system configuration

(a) When main base unit (Q3□B) is used

<p>System configuration</p>	
<p>Maximum number of Extension Stages</p>	<p>Seven Extension Stages</p>
<p>Maximum number of I/O modules to be installed</p>	<p>64 modules</p>
<p>Maximum number of occupied I/O points</p>	<p>4096</p>
<p>Main base unit</p>	<p>Q33B, Q35B, Q38B, Q312B</p>
<p>Extension base unit</p>	<p>Q52B, Q55B, Q63B, Q65B, Q68B, Q612B, QA1S65B, QA1S68B</p>
<p>Extension cable</p>	<p>QC05B, QC06B, QC12B, QC30B, QC50B, QC100B</p>
<p>Notes</p>	<ol style="list-style-type: none"> <li>(1) Extension bases unit of up to seven stages can be used.</li> <li>(2) Do not use extension cable longer than an overall extension length of 13.2m(43.31ft.).</li> <li>(3) When using an extension cable, do not bind it together with the main circuit (high voltage and heavy current) line or do not lay down them closely to each other.</li> <li>(4) When setting the No. of the expansion stages, set it in the ascending order so that the same No. is not set simultaneously by two extension base units.</li> <li>(5) When Q5□B, Q6□B and QA1S6□B types of extension base units are mixed, first connect the Q5□B, Q6□B type and then connect the QA1S6□B type. When setting the No. of the extension stages, set it from Q5□B/Q6□B in order. Although there are no particular restrictions in the order of the installation of the Q5□B and Q6□B, refer to Section 6.6 for usability.</li> <li>(6) Connect the extension cable from OUT of the extension cable connector of the base unit to IN of the extension base unit on the next stage.</li> <li>(7) If 65 or more modules are installed, an error will occur.</li> <li>(8) When bus-connected, the GOT occupies one extension stage and one slot.</li> <li>(9) The High Performance model QCPU processes the GOT as a 16-point intelligent function module. Hence, connection of one GOT decreases the number of controllable points on base unit by 16 points.</li> <li>(10) As a power supply module, the Q61SP cannot be used. Use the Q61P-A1, Q61P-A2, Q62P or Q64P as a power supply module.</li> </ol>

(b) When slim type main base unit (Q3□SB) is used

System configuration	 <p style="text-align: center;">*The above system assumes that each slot is loading with a 32-point module.</p>
Maximum number of Extension Stages	Extension not allowed
Maximum number of I/O modules to be installed	5 modules
Maximum number of occupied I/O points	4096
Slim type main base unit type	Q32SB, Q33SB, Q35SB
Extension base unit	Cannot be connected.
Extension cable	Cannot be connected.
Notes	<p>(1) The Q61P-A1, Q61P-A2, Q62P or Q64P cannot be used as a power supply module. Use the Q61SP as a power supply module.</p> <p>(2) The slim type main base unit does not have an extension cable connector. The extension base or GOT cannot be connected.</p>



## 2.2 Precaution on System Configuration

This section describes hardware and software packages compatible with QCPU.

### (1) Hardware

(a) The number of modules to be installed and functions are limited depending on the type of the modules.

Applicable Module	Type	Limit of number of modules to be installed
Q Series MELSECNET/10H network module	<ul style="list-style-type: none"> <li>• QJ71LP21</li> <li>• QJ71BR11</li> <li>• QJ71LP21-25</li> <li>• QJ71LP21G</li> <li>• QJ71LP21GE</li> </ul>	Up to 4 units
Q series Ethernet interface module	<ul style="list-style-type: none"> <li>• QJ71E71</li> <li>• QJ71E71-B2</li> <li>• QJ71E71-100</li> </ul>	Up to 4 units
Q series CC-Link system master local module	<ul style="list-style-type: none"> <li>• QJ61BT11</li> <li>• QJ61BT11N</li> </ul>	No limit
MELSECNET/MINI-S3 data link module	<ul style="list-style-type: none"> <li>• A1SJ71PT32-S3</li> <li>• A1SJ71T32-S3</li> </ul>	No limit (setting of automatic refresh function not allowed)
AnS series special function module shown on the right	<ul style="list-style-type: none"> <li>• A1SD51S</li> <li>• A1SD21-S1</li> <li>• A1SJ71J92-S3 (When GET/PUT service is used)</li> </ul>	Total of 6 units
Interrupt module	<ul style="list-style-type: none"> <li>• A1SI61</li> <li>• QI60</li> </ul>	One unit only

※: A maximum of 4 modules if the network parameters for CC-Link are set and controlled by the GX Developer. There is no restriction in the number of modules when the parameters are set by the special-purpose instructions for the CC-Link. For details on the CC-Link System Master Local Unit that can set parameters with the special-purpose instructions, refer to the user's manual for the CC-Link Master Local module.

(b) When the AnS series special-function modules shown below are used, a limitation is given to an accessible device range.

- A1SJ71J92-S3 type JEMANET interface module
- A1SD51S type intelligent communication

Device	Accessible device range
Input (X), Output (Y)	X/Y0 to 7FF
Internal relay (M), Latch relay (L)	M0 to 8191
Link relay (B)	B0 to FFF
Timer (T)	T0 to 2047
Counter (C)	C0 to 1023
Data register (D)	D0 to 6143
Link register (W)	W0 to FFF
Annunciator (F)	F0 to 2047

(c) A graphic operation terminal can be used only for the GOT900 series (Basic OS matching Q mode and communication driver must be installed).

The GOT800 series, A77GOT, and A64GOT cannot be used.

(d) The modules shown below cannot be used.

Module Name	Type
MELSECNET/IO network module	A1SJ71LP21, A1SJ71BR11, A1SJ71QLP21, A1SJ71QLP21S, A1SJ71QLP21GE, A1SJ71QBR11
MELSECNET (II), /B data link module	A1SJ71AP21, A1SJ71AR21, A1SJ71AT21B
Ethernet interface module	A1SJ71QE71-B2-S3(-B5-S3), A1SJ71E71-B2-S3(-B5-S3)
Serial communication module, computer link module	A1SJ71QC24(N), A1SJ71UC24-R2(-R4/-PRF)
CC-Link master-local module	A1SJ61QBT11, A1SJ61BT11
Modem interface module	A1SJ71CMO-S3
ME-NET interface module	A1SJ71ME81

(e) A dedicated instruction for the next module which was present in the QnA/A series program instruction cannot be used for the High Performance model QCPU. Re-writing using FROM/TO instruction is required.

Module Name	Type
High speed counter module	A1SD61, A1SD62, A1SD62D(-S1), A1SD62E
MELSECNET/MINI-S3	A1SJ71PT32-S3, A1SJ71T32-S3
Positioning module	A1SD75P1-S3(P2-S3/P3-S3)
ID module	A1SJ71ID1-R4, A1SJ71ID2-R4

(f) Some system configurations and functions are restricted when writing the parameter of the "High speed interrupt fixed scan interval" setting. Refer to the following manual for the restrictions when the parameter of the "High speed interrupt fixed scan interval" setting has been written.

- High Performance model QCPU (Q mode) User's Manual  
(Function Explanation, Program Fundamentals)

Note that the above restrictions do not apply to the High Performance model QCPU of serial number "04011" or earlier since it ignores the "High speed interrupt fixed scan interval" setting.

(2) Software package

GX Developer that can use the functions added to the High Performance model QCPU changes depending on the function version/serial number of the CPU module.

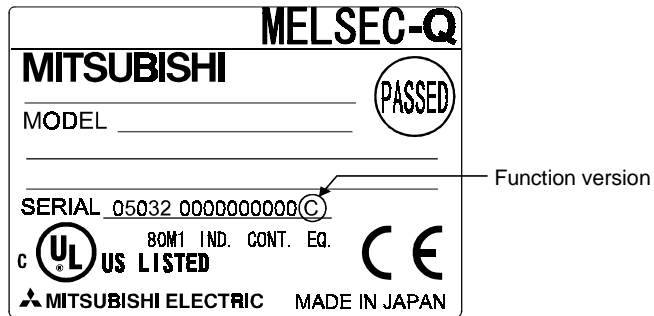
Function Version	Serial Number	GX Developer
A	—	Version 4 (SW4D5C-GPPW-E) or later
—	Functions added to "02092"	Version 6 (SW6D5C-GPPW-E) or later
B	—	
—	Functions added to "03052"	Version 7 (SW7D5C-GPPW-E) or later
—	Functions added to "04012"	Version 7.10L (SW7D5C-GPPW-E) or later
—	Functions added to "04122 "	Version 8 (SW8D5C-GPPW) or later
—	Functions added to "05032 "	Version 8.03D (SW8D5C-GPPW) or later

POINT
(1) Refer to Section 2.3 for the serial No. and function version of the High Performance model QCPU.
(2) Refer to Appendix 4 for details.

2.3 Confirming the Serial Number and Function Version

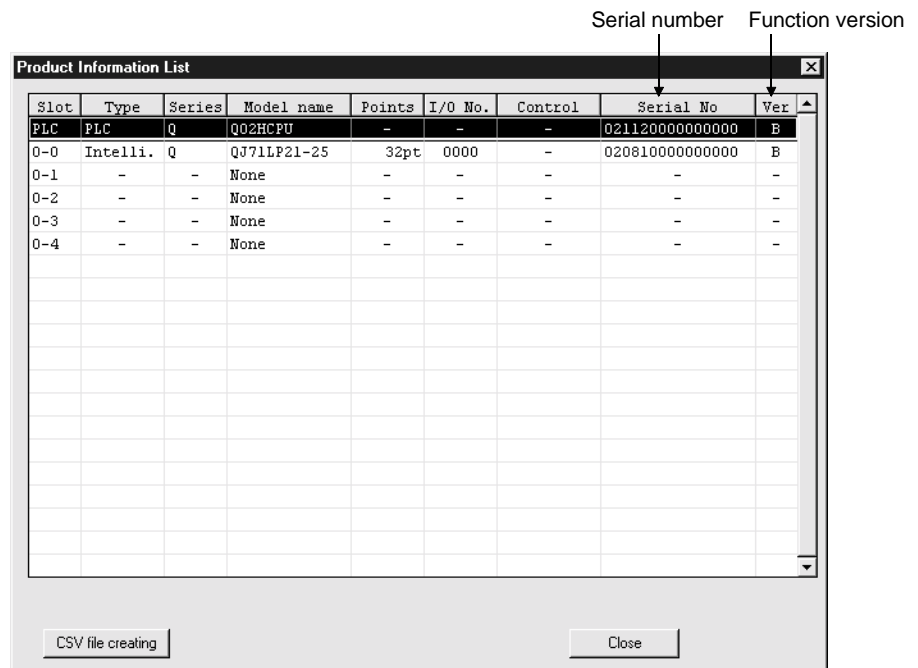
The CPU module serial No. can be confirmed on the rated plate and GX Developer's system monitor.

(1) Confirming the serial No. on the rated plate



(2) Confirming the serial No. on the system monitor (list of product information)

The CPU module serial No. and function version can be confirmed with the list of product information on the GX Developer (Version 6 or later) system monitor. Serial Nos. and function versions of the intelligent function module and CPU module can also be confirmed.



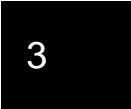
### 3. Performance Specification

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

Performance Specifications

Item	Model					Remark	
	Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU		
Control method	Repetitive operation of stored program					—	
I/O control mode	Refresh mode					Direct I/O is possible by direct I/O specification (DX□, DY□)	
Programming language (Sequence control dedicated language)	Relay symbol language, logic symbolic language, MELSAP3 (SFC), MELSAP-L, Function block, structured text (ST)					—	
Processing speed (Sequence instruction)	LD X0	0.079 μs	0.034 μs			—	
	MOV D0 D1	0.237 μs	0.102 μs			—	
Total number of instructions	381 (excluding intelligent function module dedicated instructions)					—	
Constant scan (ms) (Function for setting the scan timer to fixed settings)	0.5 to 2000 (configurable in increments of 0.5 ms)					Set parameter values to specify	
Program *2 capacity	Program memory (Drive 0)	28k step	60k step	124k step	252k step	See Section 6.2.	
Memory capacity	Memory card (RAM) (Drive 1)	Capacity of loading memory cards (2Mbyte max.)				See Section 6.5.	
	Memory card (ROM) (Drive 2)	Installed memory card capacity (Flash card: 4 Mbyte max., ATA card: 32 Mbyte max.)				See Section 6.5.	
	Standard RAM (Drive 3)	64kbyte	128kbyte *5	256kbyte *3		See Section 6.4.	
	Standard ROM (Drive 4)	112 kbyte	240 kbyte	496 kbyte	1008 kbyte	See Section 6.3.	
	CPU shared memory *3	8 kbyte				See Section 14.2.4.	
Maximum number of stored files	Program memory	28	60	124	252 *1	See Section 6.2.	
	Memory card (RAM)	256				See Section 6.5.	
	Memory card (ROM)	Flash card	288				See Section 6.5.
		ATA card	512				See Section 6.5.
	Standard RAM	2				Only one file register and one local device	
	Standard ROM	28	60	124	252	See Section 6.3.	
Standard ROM number of writings	Max. 100000 times					—	

- \*1: 124 is the maximum number of programs that can be executed on High Performance model QCPU. 125 or more programs cannot be executed.
- \*2: The maximum number of sequence steps (for one program) for which the parameters are stored in another drive and executed with the High Performance model QCPU can be calculated with the following expression.  
(Program size) - (File header size (default: 34 steps))  
Refer to the High Performance model QCPU (Q Mode) User's Manual (Function Explanation, Program Fundamentals) for details on the program size and file.
- \*3: The memory capacity of the Q12HCPU or Q25HCPU whose first five digits of serial No. are "02091" or earlier is 64k bytes.  
(Refer to Section 2.3 for the serial No. confirmation method.)
- \*4: The CPU shared memory is not latched. The CPU shared memory is cleared when the power is turned on to the PLC or when the CPU module is reset.
- \*5: The memory capacity of the Q02HCPU or Q06HCPU whose first five digits of serial No. are "04011" or earlier is 64k bytes.  
(Refer to Section 2.3 for the serial No. confirmation method.)



Performance Specifications (continued)

Item	Model					Remark	
	Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU		
Number of I/O device points	8192 points (X/Y0 to 1FFF)					Number of devices usable on program	
Number of occupied I/O points	4096 points (X/Y0 to FFF)					Number of points accessible to actual I/O modules	
Number of device points	Internal relay [M]	Default 8192 points (M0 to 8191)					Number of use points is set with parameters.
	Latch relay [L]	Default 8192 points (L0 to 8191)					
	Link relay [B]	Default 8192points (B0 to 1FFF)					
	Timer [T]	Default 2048 points (T0 to 2047) (for low / high speed timer) Select between low / high speed timer by instructions. The measurement unit of the low / high speed timer is set with parameters. (Low speed timer : 1 to 1000ms, 1ms/unit, default 100ms) (High speed timer : 0.1 to 100ms, 0.1ms/unit, default 10ms)					
	Retentive timer [ST]	Default 0 point(for low / high speed retentive timer) Switchover between the low / high speed retentive timer is set by instructions. The measurement unit of the low /high speed retentive timer is set with parameters. (Low speed retentive timer : 1 to 1000ms, 1ms/unit, default 100ms) (High speed retentive timer : 0.1 to 100ms, 0.1ms/unit, default 10ms)					
	Counter [C]	<ul style="list-style-type: none"> <li>• Normal counter default 1024 points (C0 to 1023)</li> <li>• Interrupt counter maximum 256 points (default 0 point, set with parameters)</li> </ul>					
	Data register [D]	Default 12288 points (D0 to 12287)					
	Link register [W]	Default 8192 points (W0 to 1FFF)					
	Annunciator [F]	Default 2048 points (F0 to 2047)					
	Edge relay [V]	Default 2048 points (V0 to 2047)					
File register	[R]	<ul style="list-style-type: none"> <li>• When a standard RAM is used: 32768 points (R0 to 32767)</li> <li>• When a SRAM card is used: The number of points of up to 1041408 points can be used by block conversion in increments of 32768 points (R0 to 32767).</li> <li>• When a Flash card (2Mbyte) is used: The number of points of up to 1041408 points can be used by block conversion in increments of 32768 points (R0 to 32767).</li> <li>• When a Flash card (4Mbyte) is used: The number of points of up to 1042432 points can be used by block conversion in increments of 32768 points (R0 to 32767).</li> </ul>					When a Flash card is used, read only is possible. The ATA card cannot be used.
	[ZR]	<ul style="list-style-type: none"> <li>• When a standard RAM is used: 32768 points (ZR0 to 32767)</li> <li>• When a SRAM card is used: 1041408 points (ZR0 to 1041407), No block conversion necessary.</li> <li>• When a Flash card (2Mbyte) is used: 1041408 points (ZR0 to 1041407), No block conversion necessary.</li> <li>• When a Flash card (4Mbyte) is used: 1042432 points (ZR0 to 1042431), No block conversion necessary.</li> </ul>					

3

Performance Specifications (continued)

Item	Model					Remark	
	Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU		
Number of device points	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)					
	Step relay [S] *6	8192 points (S0 to 8191)					
	Index register [Z]	16 points (Z0 to 15)					
	Pointer [P]	4096 points (P0 to 4095), set parameter values to select usable range of in-file pointer / shared pointers.					
	Interrupt pointer [ I ]	256 points (I0 to 255) The specified intervals of the system interrupt pointers I28 to I31 can be set with parameters.(0.5 to 1000ms, 0.5 ms/unit) Default I28 : 100ms I29 : 40ms I30 : 20ms I31 : 10ms					
	Special relay [SM]	2048 points (SM0 to 2047)					
	Special register [SD]	2048 points (SD0 to 2047)					
	Function input [FX]	16 points (FX0 to F)					
	Function output [FY]	16 points (FY0 to F) *7					
	Function register[FD]	5 points (FD0 to 4) *7					
Link direct device	Device having a direct access to link device. MELSECNET/10(H) use only. Specified form : J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□						
Intelligent function module device	Device having a direct access to the buffer memory of the intelligent function module. Specified form : U□□\G□□						
Latch (power failure compensation) range	L0 to 8191 (default) (Latch range can be set for B, F, V, T, ST, C, D, and W.)					Set parameter values to specify	
Remote RUN/PAUSE contact	RUN and PAUSE contacts can be set from among X0 to 1FFF, respectively.						
Clock function	Year, month, day, hour, minute, second, day of the week (leap year automatic distinction) Accuracy -3.18 to +5.25s (TYP. +2.12s) /d at 0°C Accuracy -3.93 to +5.25s(TYP. +1.90s)/d at 25°C Accuracy -14.69 to +3.53s(TYP. -3.67s)/d at 55°C						
Allowable momentary power failure period	Varies according to the type of power supply module.						
5VDC internal current consumption	0.60A	0.64A	0.64A	0.64A	0.64A		
External dimensions	H	98mm (3.86inch)					
	W	27.4mm (1.08inch)					
	D	89.3mm (3.52inch)					
Weight	0.20kg	0.20kg	0.20kg	0.20kg	0.20kg		

\*6: The step relays are devices for the SFC function.  
\*7: In a program, only FX0 to FX4 and FY0 to FY4 can be used.

**REMARK**

For general specifications, refer to the High Performance Model QCPU (Q Mode) User's Manual (Hardware Design, Maintenance and Inspection).

## 4 SEQUENCE PROGRAM CONFIGURATION AND EXECUTION CONDITIONS

Sequence programs and SFC programs can be executed at the High Performance model QCPU.

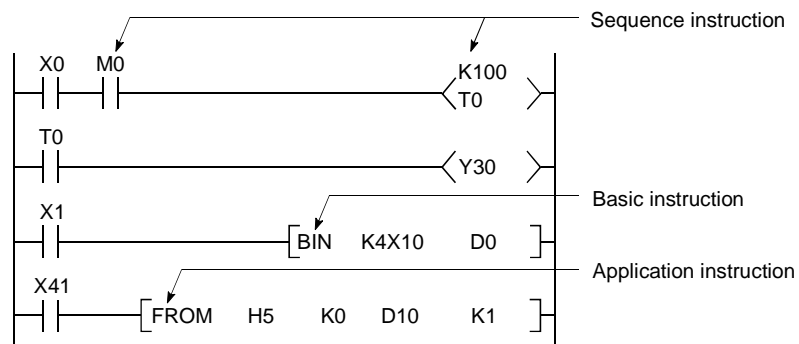
This chapter describes the sequence program configuration and execution conditions. SFC programs are not described in this manual.

For details on SFC programs, refer to the QCPU (Q mode)/QnACPU Programming Manual (SFC).

### 4.1 Sequence Program

#### (1) Definition of sequence program

- (a) A sequence program consists of sequence instructions, basic instructions, and application instructions, etc.

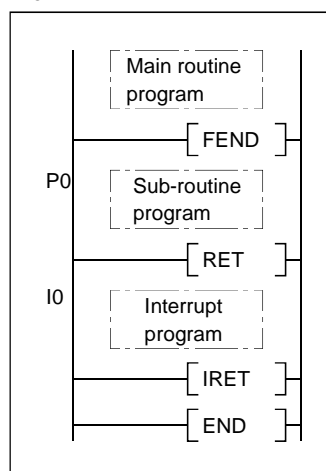


- (b) There are 3 types of sequence program: main routine programs, sub-routine programs, and interrupt programs.

For details on these programs, see the following sections of this manual:

- Main routine programs : Section 4.1.1
- Sub-routine programs : Section 4.1.2
- Interrupt programs : Section 4.1.3

File A



#### REMARK

For details on the sequence instructions, basic instructions, and application instructions, refer to the " QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions)".

(2) Sequence program writing format

Programming for sequence programs is enabled using either ladder mode or list mode.

(a) Ladder mode

- The ladder mode is based on the relay control sequence ladder. Programming expressions are similar to the relay control sequence ladder.
- Relay symbolic language programming occurs in ladder block units. A ladder block is the smallest unit of sequence program operation, with the ladder beginning from the left bus and ending at the right bus.

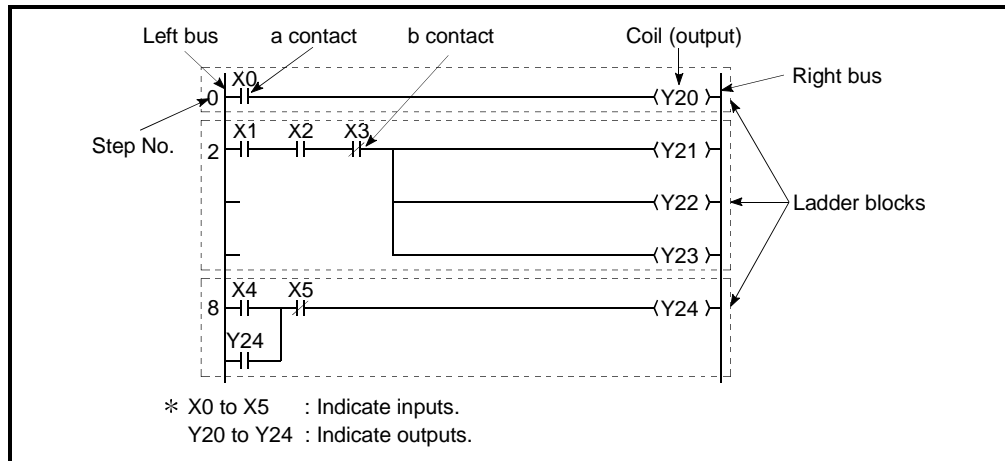


Fig.4.1 Ladder Block

(b) List mode

The list mode uses dedicated instructions instead of the contact symbols, coil symbols, etc., used in the ladder mode.

Contact a, contact b and coil instructions are as follows:

- a contact .....LD, AND, OR
- b contact .....LDI, ANI, ORI
- coil.....OUT

(3) Program operation

In sequence program, the instructions are executed in order beginning from step 0 and ending at the END/FEND.

With ladder mode, the instructions in a ladder block are executed in order beginning from left bus to the right bus. When one ladder block is completed, the next downward ladder block will be executed.

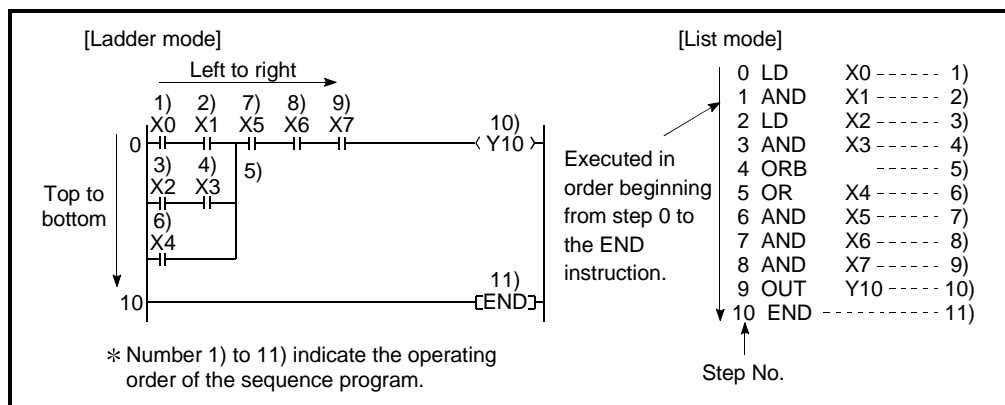


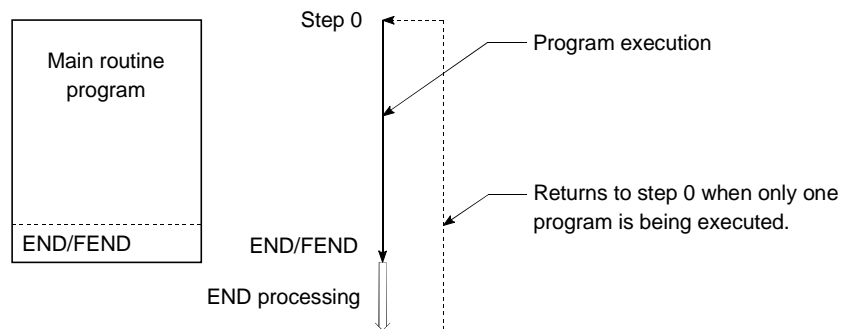
Fig.4.2 Sequence Program Operation



4.1.1 Main routine program

(1) Definition of main routine program

- (a) A main routine program begins from step 0 and ends at the END/FEND instruction. \*1
- (b) In the main routine program, the instructions are executed from step 0 to the END/FEND.
  - 1) If only one program is executed, the program is operated from step 0 again after the END/FEND instruction is executed and the END instruction is processed.



- 2) If multiple programs are being executed, each of them will be operated according to the designated execution conditions after the END/FEND instruction is processed.

(2) Execution conditions for main routine programs \*2

If multiple programs are being executed, the following five types of execution conditions can be designated by the program in the PLC parameters according to the application.

- Initial execution program : See Section 4.2.1.
- Scan execution type program : See Section 4.2.2.
- Low speed execution type program : See Section 4.2.3.
- Stand-by type program : See Section 4.2.4.
- Fixed scan execution type program : See Section 4.2.5.

**REMARK**

\*1: For details on the END/FEND instruction, refer to the "QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)".

\*2: If only one program is executed, it is processed under the "scan execution type program" condition without designation by the program in the PLC parameters.

4.1.2 Sub-routine programs

(1) Definition of sub-routine program

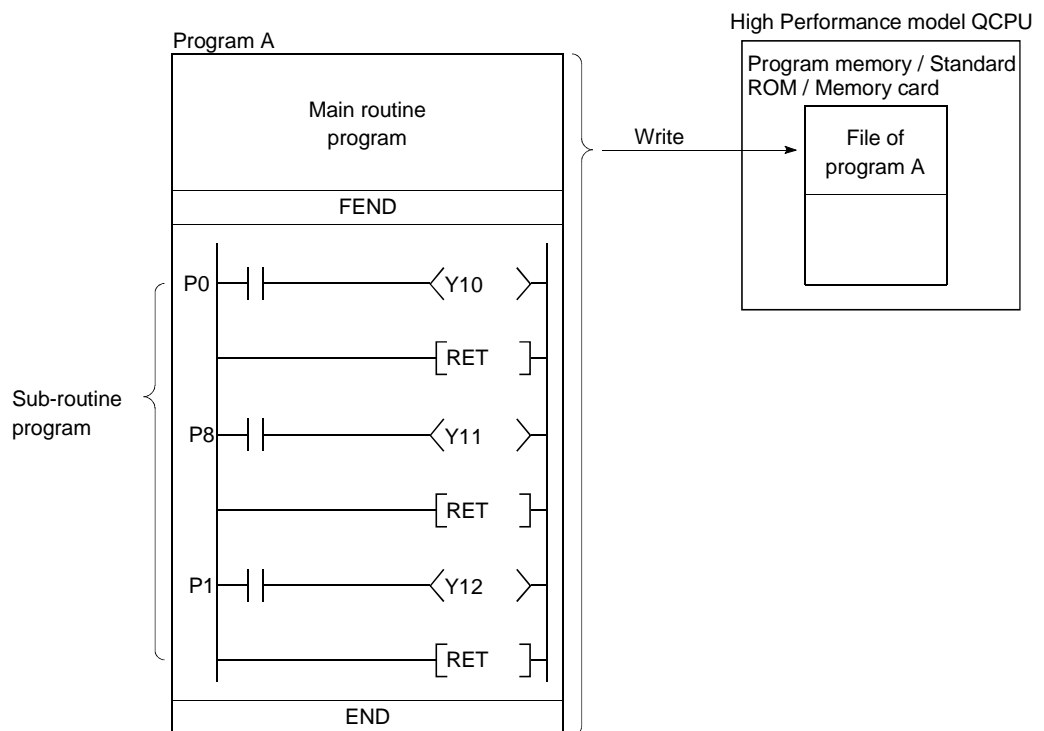
- (a) A sub-routine program begins from a pointer (P[ ]) and ends at a RET instruction.
- (b) A sub-routine program is executed only when called by a CALL instruction (e.g. CALL(P), FCALL(P)) from the main routine program.
- (c) Sub-routine program application
  - 1) The overall step count can be reduced by using a sub-routine program as a program which is executed several times in one scan.
  - 2) The step count of a constantly executed program can be reduced by using a sub-routine program as a program which is executed only when a given condition is satisfied.

(2) Sub-routine program management

Sub-routine programs are created after the main routine program (after FEND instruction), and the combination of main and sub-routine programs can be managed as one program.

(a) When created after the main routine program

- A sub-routine program is created between the main routine program's FEND and END instructions.
- Because there are no restrictions on the order in which sub-routine programs are created, it is not necessary to set the pointers in ascending order when creating multiple sub-routine programs.
- Either a local pointer or a common pointer can be used. \*



**REMARK**

\*: See Section 10.9 for details on local and common pointers.  
See Section 10.8 for details on sub-routine program nesting.

- (b) Using the sub-routine program as a separate program  
Sub-routine programs can also be managed as separate, separate programs (stand-by type programs). (See Section 4.2.4 for details on stand-by type programs).

4.1.3 Interrupt programs

(1) Definition of interrupt program

- (a) An interrupt program begins from the interrupt pointer (I□), and ends at the IRET instruction. \*1
- (b) Interrupt programs are executed only when an interrupt factor occurs. \*1

**POINT**

The interrupt pointers include a pointer designed for only the high speed interrupt function (I49).  
When you have used I49, do not use interrupt programs, which use the other interrupt pointers I0 to I48, I50 to I255, and fixed scan execution type programs. If any fixed scan execution type program or the like is run, the interrupt program using I49 cannot be executed at the preset interrupt cycle intervals.  
See Section 7.20 for details on the high speed interrupt function.

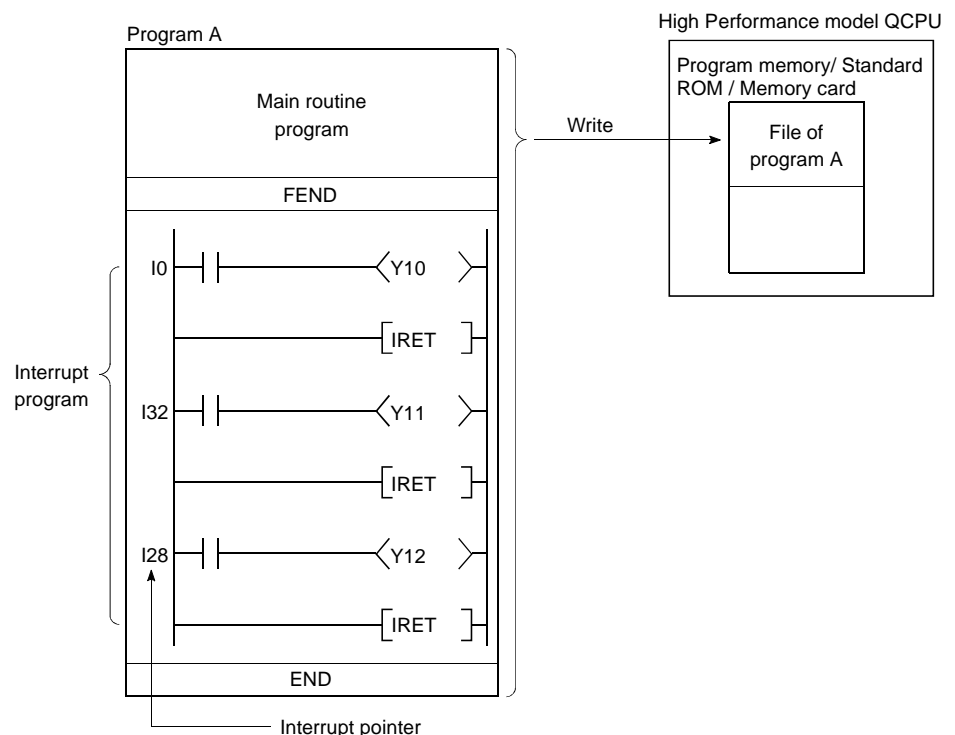
**REMARK**

\*1: See Section 10.10 for details on interrupt factors and interrupt pointers.

(2) Interrupt program management

Interrupt programs are created after the main routine program (after the FEND instruction), and the combination of main and sub-routine programs can be managed as one program.

- (a) When created after the main routine program
  - An interrupt program is created between the main routine program's FEND and END instructions.
  - Because there are no restrictions on the order in which interrupt programs are created, it is not necessary to set the interrupt pointers in ascending order when creating multiple interrupt programs.



- (b) Using the interrupt program as a separate program  
 Interrupt programs can also be managed as separate, discrete programs (stand-by type programs). (See Section 4.2.4 for details on stand-by type programs).  
 However, the same interrupt program pointer number cannot be used more than once in the program being executed by the CPU module.

(3) Executing interrupt programs

- (a) In order to execute an interrupt program with the interrupt pointer I32 to I47, IMASK and EI instructions are required to obtain permission for the interruption. \*1
  - 1) If an interrupt factor occurs prior to an interruption permitted status, the interrupt program corresponding to the factor will be executed when the "interruption permitted" status is established.
  - 2) If an interrupt factor occurs during a STOP/PAUSE, the interrupt program corresponding to the factor will be executed when the "interruption permitted" status is established following a return to the RUN status.

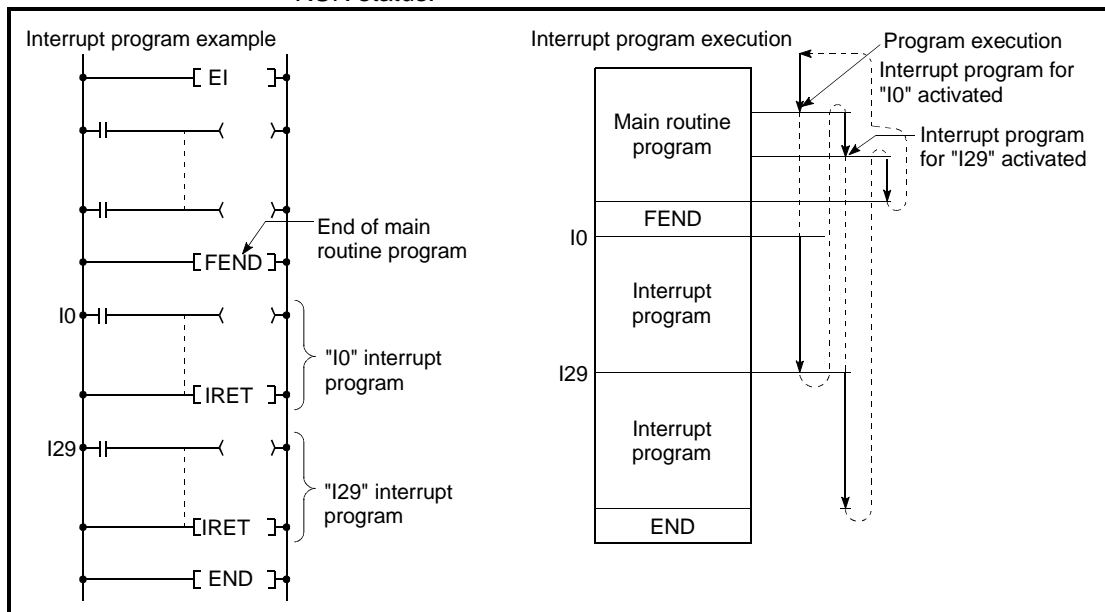


Fig.4.3 Interrupt Program Execution

- (b) When an interrupt factor occurs, the interrupt program with the interrupt pointer number corresponding to that factor is executed. However, interrupt program execution varies according to the condition at that time.
- 1) If two or more interrupts occur at the same time:  
The interrupt programs are executed, starting with the one corresponding to an interrupt pointer number (I □) of the highest priority. \*2  
The remaining interrupt programs remain on stand-by until processing of the higher priority interrupt program is completed.  
If the same interrupt factor as that being executed occurs before the interrupt program is processed, the interrupt factor is stored in the memory and, after the interrupt program has been processed, the same interrupt program is executed again.
  - 2) When an instruction is being executed:  
The interrupt program may be executed by interrupting the execution of an instruction in the main routine program. When the same device is used in both the main routine program and interrupt program, device data may be separated.  
To prevent the separation of the device data, the following measures must be taken.
    - (a) Do not specify the device, to which data will be written in the interrupt program, directly in the main routine program, but use another device by shifting the data with a transfer instruction, etc.
    - (b) If inconvenience is caused when the instruction is interrupted in the main routine program, execute it after disabling the interruption with the DI instruction.

However, since the interrupt program will not interrupt during access to the device of each argument of the instruction, data separation will not occur on an argument basis.
  - 3) Interruption during a network refresh:  
If an interrupt factor occurs during a network refresh operation, the network refresh operation is suspended, and the interrupt program is executed.  
This means that "assurance of blocks in cyclic data at each station" cannot be secured by using a device designated as a destination of link refresh operation on the MELSECNET/H Network System. \*3

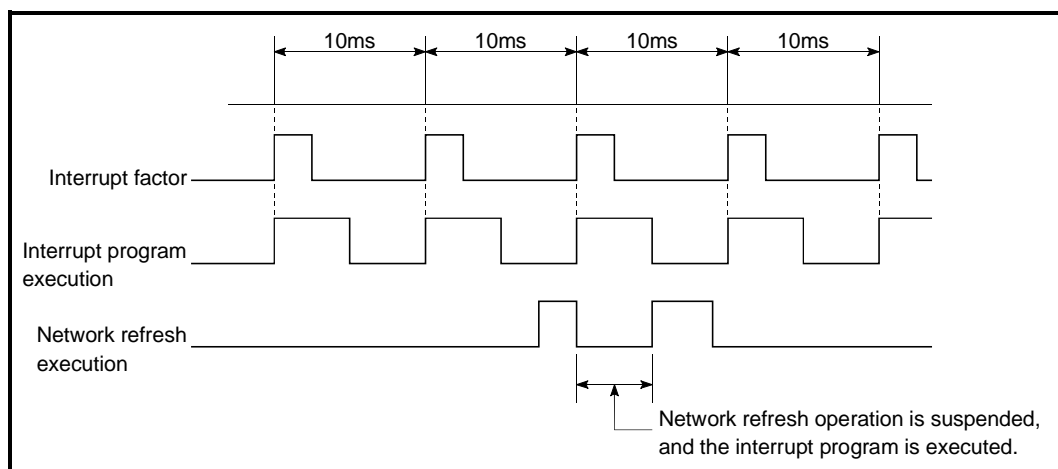


Fig.4.4 Interruption during Network Refresh Operation

- 4) Interruption during END processing:  
If an interrupt factor occurs during an END processing waiting period when constant scan is performed, the interrupt program corresponding to the factor will be executed.

**REMARK**

- \*1: For details on the IMASK and EI instructions, refer to the "QCPU (Q mode)/QnACPU Programming Manual (Common Instructions).  
To execute interrupt programs I0 through I31 and I48 through I255, use an EI instruction to enter the interrupt programs into an interrupt enabled status.
- \*2: See Section 10.10 for details on the priority ranking of interrupt programs.
- \*3: For assurance of station unit blocks in cyclic data, see the "MELECNET/H Network System Reference Manual."

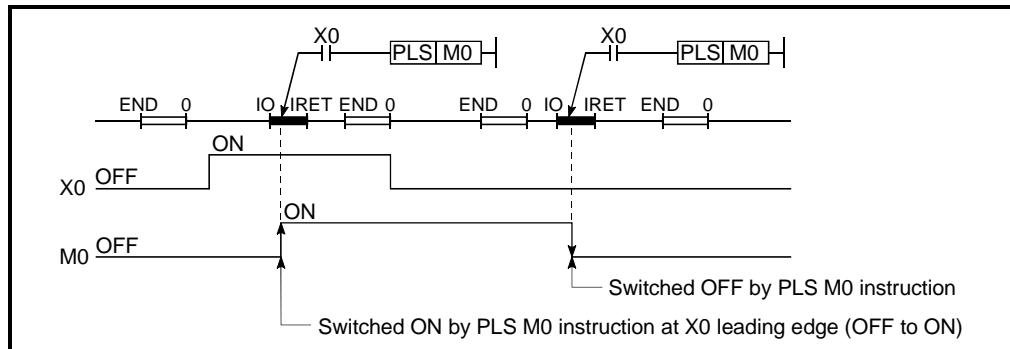
- (c) When the interrupt program is executed in the default setting of the High Performance model QCPU, the save and restoration of the index register value and the save and restoration of the file register block No. are performed at the time of switching between the main routine program and interrupt program.  
Refer to Section 10.6.2 for details..

- (4) High speed execution of an interrupt program and overhead time  
By default, High Performance model QCPU performs the following process when executing an interrupt program:
- To hide and restore an index register. (See section 10.6.2)
  - To hide and restore the file name of a file register in use.
- The above-listed processes are not performed if "Execute at a High Speed" is selected at the "PLC System" tab screen in the "(PLC) Parameter" dialog box.  
This will make it possible to shorten the duration of overhead time required for execution of an interrupt program.

CPU TYPE	OVERHEAD TIME ( μs)	
	High speed execution is not selected	High speed execution is selected
Q02CPU	380	230
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	165	100

(5) Program creation restrictions

- (a) A device which is switched ON by a PLS instruction in an interrupt program will remain ON until that interrupt program is executed again.



- (b) A DI status (interruption prohibited) is established during execution of an interrupt program.  
Do not execute EI/DI instructions in the interrupt program.
- (c) Timers cannot be used in interrupt programs.  
As timers are used at OUT T[ ] instructions to update present values and switch contacts ON and OFF, the use of a timer in the interrupt program would make a normal time count impossible.
- (d) The following commands cannot be used in the interrupt program.
- COM
  - ZCOM
- (e) When the interrupt program/fixed scan execution type program is executed at a measuring time such as the scan time or execution time, the values of the interrupt program/fixed scan execution type program are added to the measured time.  
Thus, if the interrupt program/ fixed scan execution type program is executed, the values stored in the following special registers and GX Developer monitor values will be longer than when the interrupt program/ fixed scan execution type program is not executed.
- 1) 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
    - SD532, SD533: Minimum scan time for low speed
    - SD534, SD535: Maximum scan time for low speed
    - SD540, SD541: END processing time
    - SD542, SD543: Constant scan wait time
    - SD544, SD545: Cumulative execution time for low speed execution type programs
    - SD546, SD547: Execution time for low speed execution type programs
    - SD548, SD549: Scan program execution time
    - SD551, SD552: Service interval time
  - 2) GX Developer monitor values
    - Execution time measurement
    - Scan time measurement
    - Constant scan

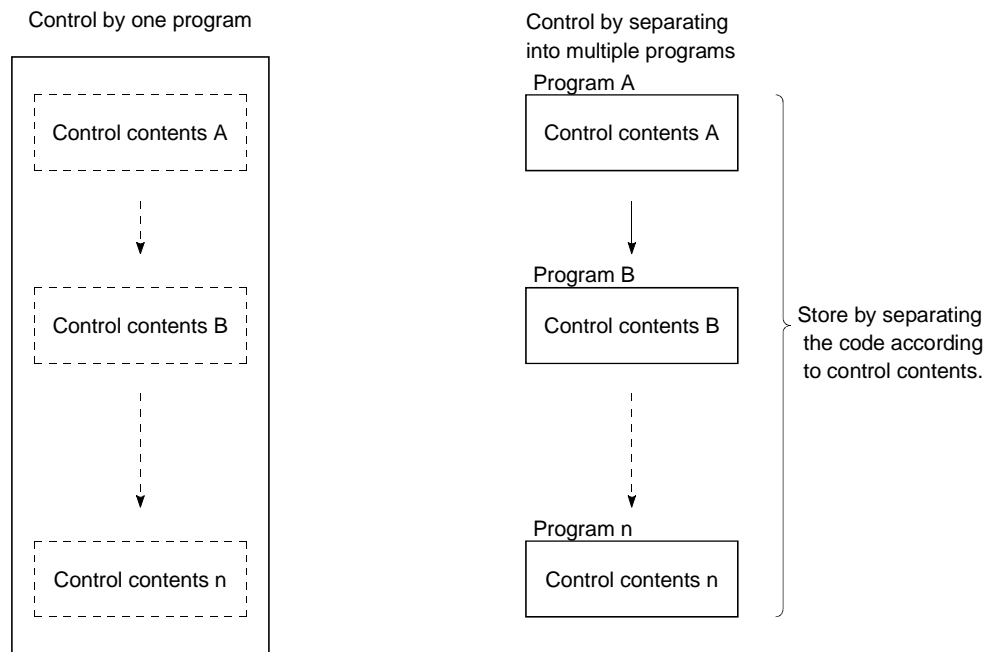


### 4.2 Program Execute Type

Programs executed by High Performance model QCPU can be stored in the High Performance model QCPU's program memory, standard ROM or memory card. Programs can be stored in the standard ROM or memory card as a single program, but also as multiple programs by splitting them into separate programs for each control function.

This permits the programming procedure to be split up among several program designers, who can design separate programs for each operation and can store them in the standard ROM or memory card.

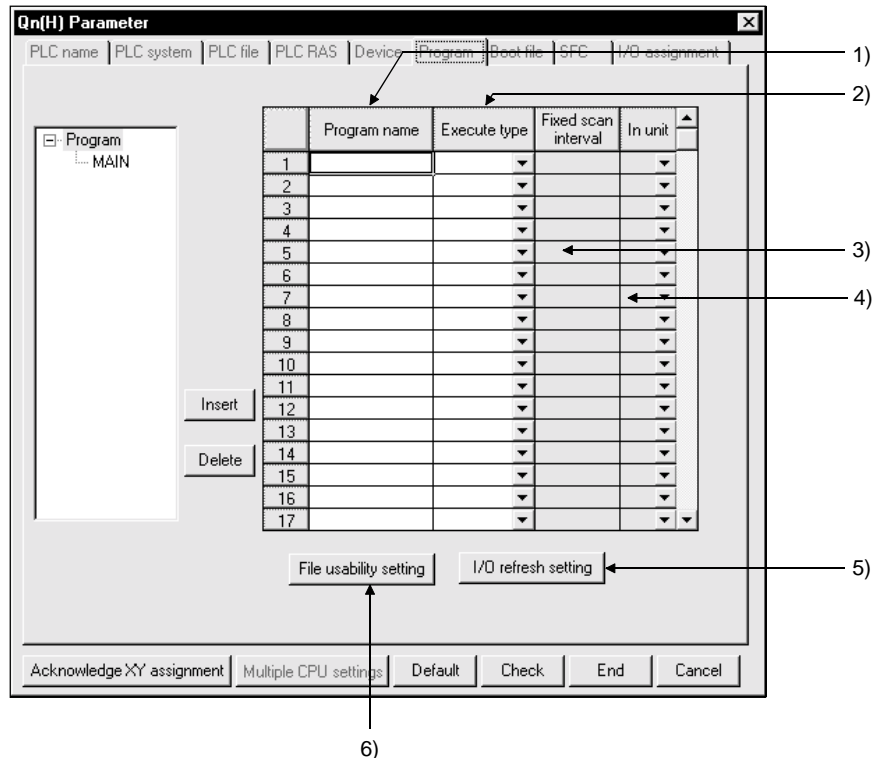
When multiple programs are executed by High Performance model QCPU, "program name (file name)" and "execute type" settings of the programs must be designated.



(1) Execute Type Setting

- (a) To execute several programs, specify a "Program name" and "Execute type" of each program at the "Program" tab screen in the "(PLC) Parameter" dialog box.

High Performance model QCPU will execute the selected programs in the order of specified Execute Type setting.



- 1) Program name  
This column is used to specify a program name (file name) of the program to be executed by High Performance model QCPU.
- 2) Execute type  
This column is used to specify the execute type of the program defined in the "Program name" column. See Section (b).
- 3) Fixed scan  
This column is used to specify time intervals at which to an execution type program.  
The Fixed Scan setting range is determined by the units of time intervals as follows:
  - In the unit "ms": 0.5 to 999.5
  - In the unit "s": 1 to 60
- 4) In units  
This column is used to specify the units (ms/s) of fixed scan intervals.

5) File Use Setting

Sets whether to use the data (file register, initial device value, comment, local device) set at the "PLC file" tab screen in the "(PLC) Parameter" dialog box, per program. The data is set for each program. By default, the option "Use PLC file setting" is selected. If the option "Not used" is selected, the File Use setting is made as listed below in the table.

Setting item	Processing when the option "Not used" is selected
File register	File registers can not be used in the program.
Initial device value	The device initial value is not set when the program file name and the device initial value is the same.
Comment file used in a command	Comments can not be used in the program.
Local device	Local devices are not hidden or restored at the time of program conversion.

6) I/O Refresh Setting

High Performance model QCPU uses the I/O Refresh setting to update output and input from an I/O module and an intelligent function module.

The I/O Refresh Setting button is used to update the range of selected programs. Make the I/O Refresh setting for a scan execution type program if you want to receive an input (X) or produce an output (Y) before executing the fixed scan execution type program.

(b) There are following 5 execute types:

1) Initial execution (Initial)

This program type is executed once only at power ON or when STOP-RUN switching occurs. (See Section 4.2.1)

2) Scan execution (Scan)

This program type is executed once per scan, beginning from the scan which follows execution of the initial execution program. (See Section 4.2.2)

3) Low speed execution (Low speed)

This program type is executed only when a constant scan setting is made or when a time is set for execution of low speed execution type programs.

- When a constant scan setting is made, the program is executed during the surplus time of a scan execution type program.
- When a time for execution of low speed execution type programs is set, the program is executed during this set time. (See Section 4.2.3.)

4) Stand-by (Wait)

This program is executed only when its execution is requested. (See Section 4.2.4.)

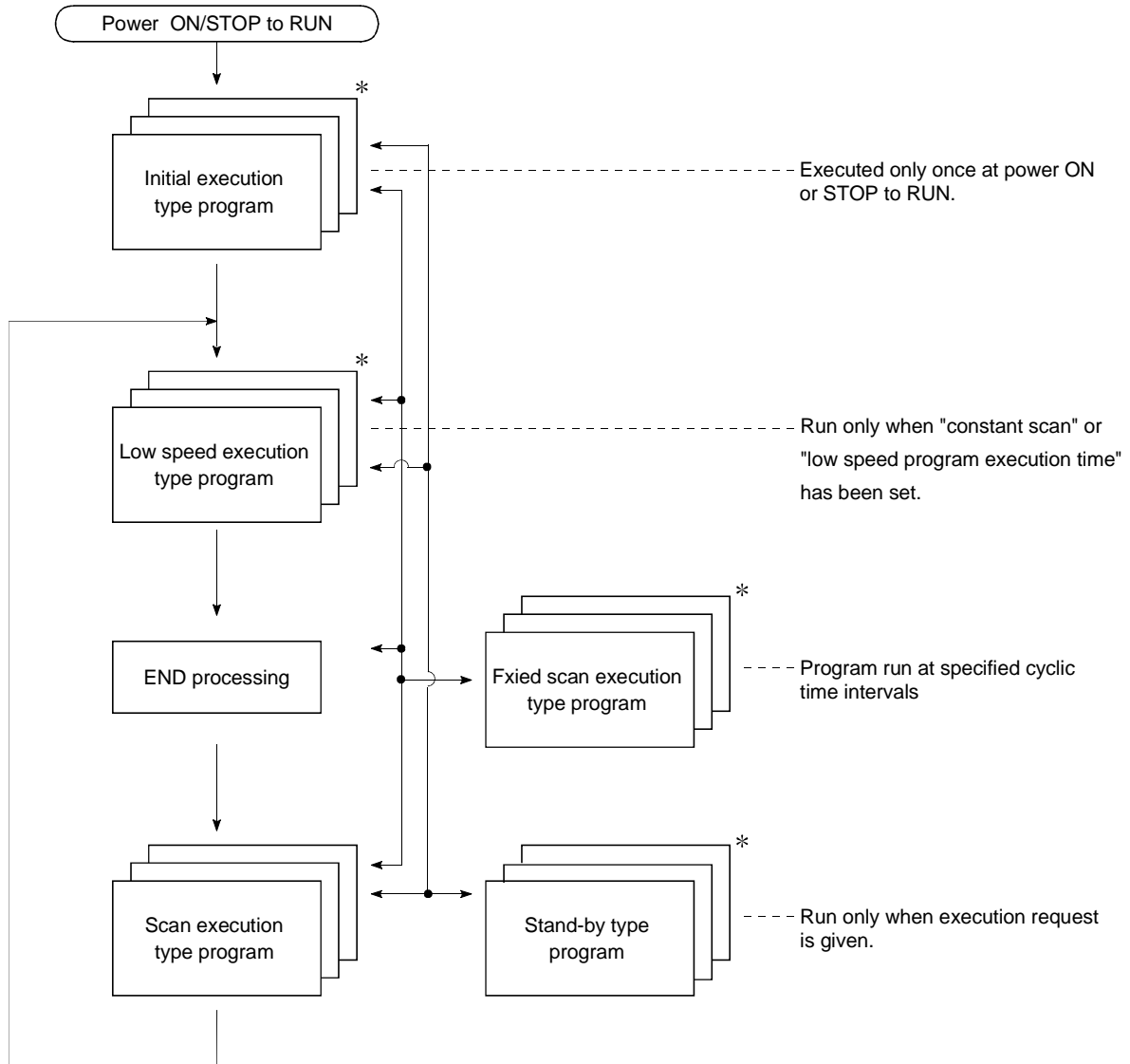
5) Fixed scan execution (Fixed scan)

Program that is executed at time intervals specified in the "Fixed scan" and "In units" columns of the Program Setting sheet of the PLC Parameter dialog box. (See Section 4.2.5.)

(c) Scantimes of programs being executed (except the fixed scan execution type program) can be checked on the monitor of the program list. (See Section 7.11.1.)

(2) Flow of each program of High Performance model QCPU

The flow of each program after power-ON or STOP of the PLC to RUN switching of the CPU module is shown below.

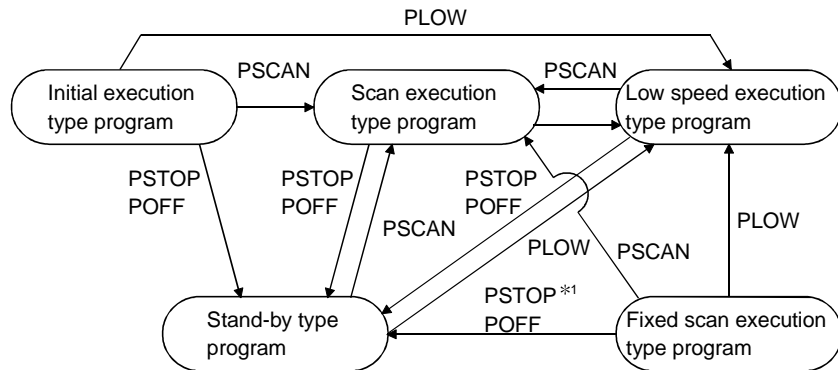


**POINT**

Not all execute types need to be set for the High Performance model QCPU. Use the items marked with "\*" as needed, such as the Initial execution, low speed execution, stand-by and fixed scan execution type programs.

(3) Changing the Execute Type

- (a) The Execute Type setting made at the "Program" tab screen in the "(PLC) Parameter" dialog box can be changed at any time while a sequence program is executed. To change the execute type of a program, use a PSCAN, PLOW, PSTOP or POFF instruction.



- (b) The following table shows the timing of changing the execute type of a program by using a PSCAN, PLOW, PSTOP or POFF instruction.

Executed instruction Execute type before change	PSCAN	PSTOP	POFF	PLOW
Scan execution type	No change - remains scan execution type.	Becomes stand-by type.	Output is turned OFF in the next scan.	Becomes low speed type.
Initial execution type	Becomes scan execution type.		Becomes stand-by type from the next scan after that.	
Stand-by type		No change - remains stand-by type.	No processing.	
Low speed execution type	Low speed execution type execution is stopped: becomes scan executions from the next scan. (Execution from step 0.)	Low speed execution type execution is stopped: becomes stand-by type from the next scan.	Low speed execution type execution is stopped, and output is turned OFF in the next scan. Becomes stand-by type from the next scan after that.	No change - remains low speed executions.
Fixed scan execution type	Becomes scan execution type.	Becomes stand-by type.	Output is turned OFF in the next scan. Becomes stand-by type from the next scan after that.	Becomes low speed type.

**POINT**

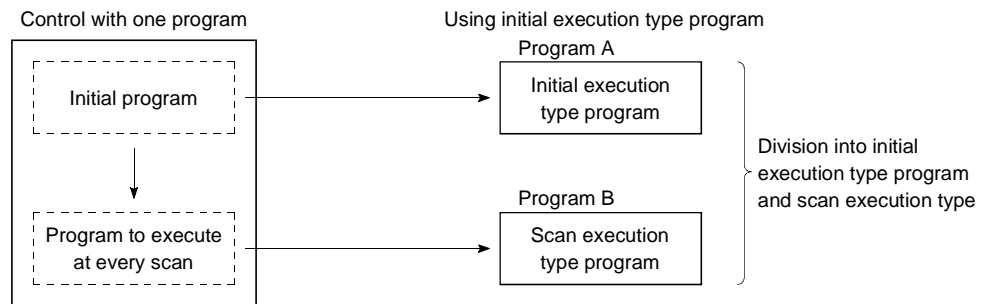
\*1: If the fixed scan execution type program is changed to another execution type, you cannot return to the fixed scan execution type.

4.2.1 Initial execution type program

(1) Definition of initial execution type program

- (a) An initial execution type program is executed once only at power ON, or when STOP to RUN switching occurs.
- (b) This program's execute type is designated as "initial" in the program of the PLC parameters.
- (c) In the same manner as the initial processing for the intelligent function module, the initial execution program is executed only once, and is not required in subsequent scans.

An instruction that contains a complete device cannot be used for an initial execution type program because the complete device needs several scans to complete the execution.

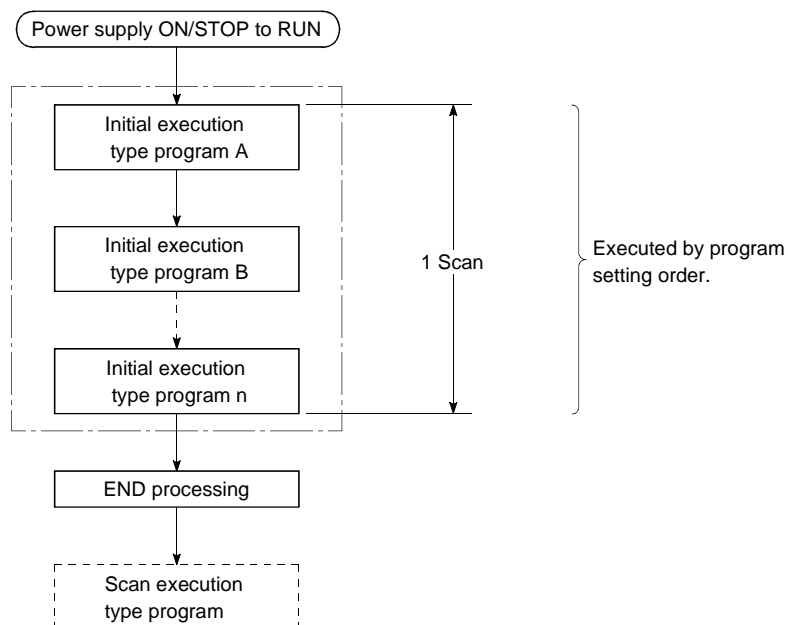


(2) Using multiple initial execution type programs

When multiple initial execution type programs are used, they are executed one by one in ascending order of the program in the PLC parameters.

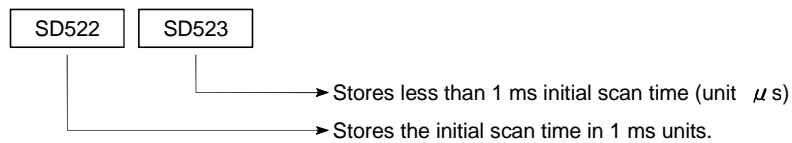
(3) END processing

END processing is performed when all initial execution type programs are completed, and the "scan execution type program" is then executed from the next scan.



(4) Initial scan time

- (a) This is the execution time period for initial execution type programs.  
If multiple initial execution type programs are executed, it is the execution time period in which all those programs are executed.  
When an interrupt program/fixed scan execution type program is executed while an initial execution type program is running, the execution time of the interrupt program/fixed scan execution type program will be added to the initial execution type program.
- (b) The High Performance model QCPU measures the initial scan time and stores the result in special registers (SD522, SD523). \*1  
The initial scan time can be checked by monitoring the SD522 and SD523 special registers.



If the SD522 value is 3, and the SD523 value is 400, the initial scan time is 3.4 ms.

**POINT**

\*1: The accuracy of the initial scan time stored at the special registers is  $\pm 0.1$  ms. The initial scan time count will continue even if a watchdog time reset instruction (WDT) is executed at the sequence program.

(5) Initial execution monitor time

- (a) The execution period of the initial execution type program can be monitored by this timer. The default value is not set.  
When monitoring the execution time of the initial execution type program, designate the initial execution monitor time within the range of 10 to 2000 ms range at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box. (Setting unit: 10 ms)
- (b) The low speed execution type program is executed after the execution of the initial execution type program is completed.  
To use the low speed execution type program, specify the time that is longer than the sum of the initial scan time and the execution time of the low speed execution type program.
- (c) When the initial scan time exceeds the set initial execution monitor time, "WDT ERROR (error code: 5000)" occurs, and High Performance model QCPU operation is stopped.

**POINT**

When the initial execution monitor time is designated, there will be a 10 ms error in the count value.  
Therefore, a monitor time setting (t) of 10 ms will result in a "WDT ERROR" when the initial scan time is in the range  $10 \text{ ms} < t < 20 \text{ ms}$ .

4.2.2 Scan execution type program

(1) Definition of scan execution type program

- (a) Scan execution type programs are executed once per scan, beginning from the scan which follows execution of the initial execution type program.
- (b) Set the execute type to "scan" at the "Program" tab screen in the "(PLC) Parameter" dialog box.

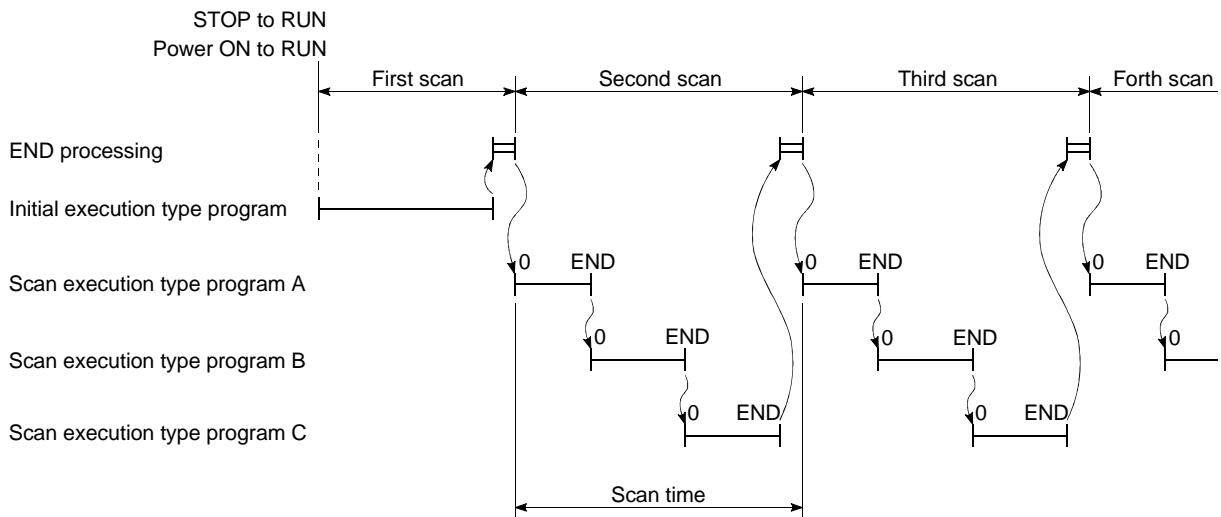
(2) Executing multiple scan execution type programs

When multiple scan execution type programs are used, they are executed one by one in ascending order set at the "Program" tab screen in the "(PLC) Parameter" dialog box.

(3) END processing

The first scan execution type program is executed again, when all scan execution type programs are executed and the END processing is completed.

The END processing (network refresh) can be performed for each program while several scan execution type programs are executed. To do this, include a COM instruction at the end of each scan execution type program.



(4) Constant scan

Constant scan is a function that repeats the execution of a main routing program at give intervals.

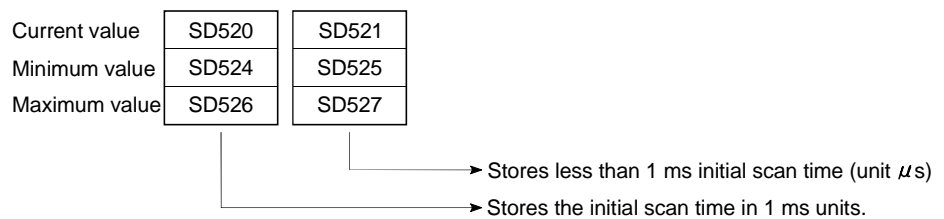
When constant scan is set, the scan execution program is executed at intervals of the preset constant scan time.

Refer to Section 7.2 for details of constant scan.



(5) Scan time

- (a) The "scan time" is a total of following the execution time of the scan execution type program and END processing.  
If multiple scan execution type programs are used, the "scan time" is the total time required to execute all the programs.  
When an interrupt program/fixed scan execution type program is executed, the value added to the interrupt program/fixed scan execution type program's execution time will become the scan time.
- (b) The scan time current value, minimum value, and maximum value are measured at the High Performance model QCPU, and the results are stored in special registers (SD520, SD521, and SD524 to SD527). \*1  
Therefore, the initial scan time can be checked by monitoring the SD520, SD521, and SD524 to SD527 special registers.



If the SD520 value is 3, and the SD521 value is 400, the initial scan time is 3.4 ms.

**POINT**

\*1: The accuracy of the scan time stored at the special registers is  $\pm 0.1$  ms. The scan time count will continue even if a watch dog timer reset instruction (WDT) is executed at the sequence program.

(6) WDT (Watch dog timer)

This is the timer which monitors the scan time, and its default setting is 200 ms. This WDT setting can be designated within the range of 10 to 2000 ms range at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box.

(Setting units: 10 ms)

When using the low speed execution type program, set the WDT greater than the scan time plus the execution time of the low speed execution type program.

If the scan time (execution time for scan execution type program + low speed execution type program) exceeds the WDT setting value, a "WDT ERROR (error code: 5000)" occurs, and High Performance model QCPU operation is stopped.

**POINT**

The WDT measurement error is 10 ms.  
Therefore, a WDT setting (t) of 10 ms will result in a "WDT ERROR" if the scan time is in the following range:  $10 \text{ ms} < t < 20 \text{ ms}$ .

**REMARK**

Use the GX Developer's Program Monitor List to check the execution time of a program being executed. See Section 7.11.1 for details on the GX Developer's Program Monitor List.

4.2.3 Low speed execution type program

(1) Definition of low speed execution type program

- (a) Low speed execution type programs are executed only during "constant scanning surplus time" or during the period designated for "low speed program execution time".
  - 1) For a constant scan time with enhanced control accuracy, designate a constant scan time setting at the "PLC RAS" tab screen in the "(PLC) Parameter".  
(Setting range: 0.5 to 2000 ms, setting unit: 0.5 ms)
  - 2) To secure execution time for low speed execution type programs at each scan, designate a low speed program execution time in the "PLC RAS" tab screen the (PLC) "Parameter".  
(Setting range: 1 to 2000 ms, setting unit: 1 ms)
  - 3) In order to execute a low speed execution type program, set either the constant scan time or low speed program execution time.
- (b) Set the execute type of the low speed program to "low speed" in the program of the PLC parameters.
- (c) The low speed execution type program is used for programs which do not require execution in each scan, for example programs for printer output.

(2) Executing multiple low speed execution type programs

When multiple low speed execution type programs are used, they are executed one by one in ascending number order of the program in the PLC parameters.

(3) Execution time of the Low speed execution type program to be executed per scan

- (a) If all the low speed execution type program operation is completed within one scan and there is surplus time, the processing executed after that depends on the ON/OFF status of special register SM330 and the execution condition for low speed execution type programs.
  - 1) Asynchronous method (SM330 = OFF)  
Method in which low speed execution type program operation is continued in the surplus time.
  - 2) Synchronous method (SM330 = ON)  
Method in which even if there is surplus time, low speed execution type program operation is not continued, and operation starts again from the next scan.

Operation method for low speed execution type programs	SM330 setting status	Execution condition for low speed execution type programs	
		When constant scan time is set	When low speed program execution time is set
Asynchronous method	OFF	The low speed execution type program is re-executed * <sup>1</sup> .	The low speed execution type program is re-executed * <sup>2</sup> .
Synchronous method	ON	Constant scan waiting time is generated * <sup>3</sup> .	Scan execution type program operation is started * <sup>4</sup> .

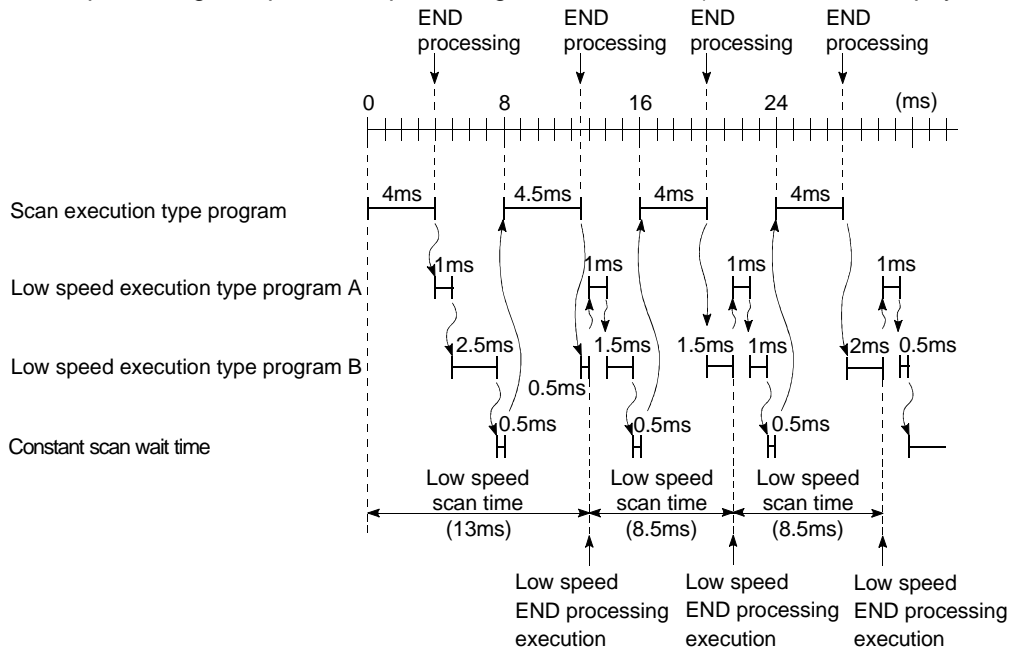
- \*1 If a constant scan time has been designated, the low speed execution type program will be executed repeatedly during the constant scan's surplus time.  
Therefore, the low speed execution type program's execution time varies from scan to scan.  
As the low speed execution type program will not be executed at all if the constant scan's surplus time is 0.5 ms or less, a constant scan time setting should be designated which provides a surplus time of more than 0.5 ms.
  - \*2 If a low speed program execution time has been designated, the low speed execution type program will be executed repeatedly in accordance with that time setting.  
Therefore, the scan time will vary from scan to scan.
  - \*3 If a constant scan time has been designated, the surplus time after completion of low speed END processing is waiting time, and execution of a scan execution type program starts when the constant scan time has elapsed.  
This means that the scan time is constant in each scan.  
However, if the surplus time after the constant scan is less than 0.5 ms, low speed execution type programs cannot be executed. If using a low speed execution type program, set the constant scan time so that the surplus time is 0.5 ms or longer.
  - \*4 If a "low speed program execution time" has been designated, scan execution type program operation is started ignoring the surplus time after completion of low speed END processing.  
This means that the scan time differs in each scan.
- (b) If a low speed execution type program cannot be processed within constant scan surplus time or within the low speed program execution time, program execution is temporarily stopped and the remainder of the program is executed in the next scan.

1 : Asynchronous method

(1) Constant scan time setting

The low speed execution type program is operated under the following conditions as shown below.

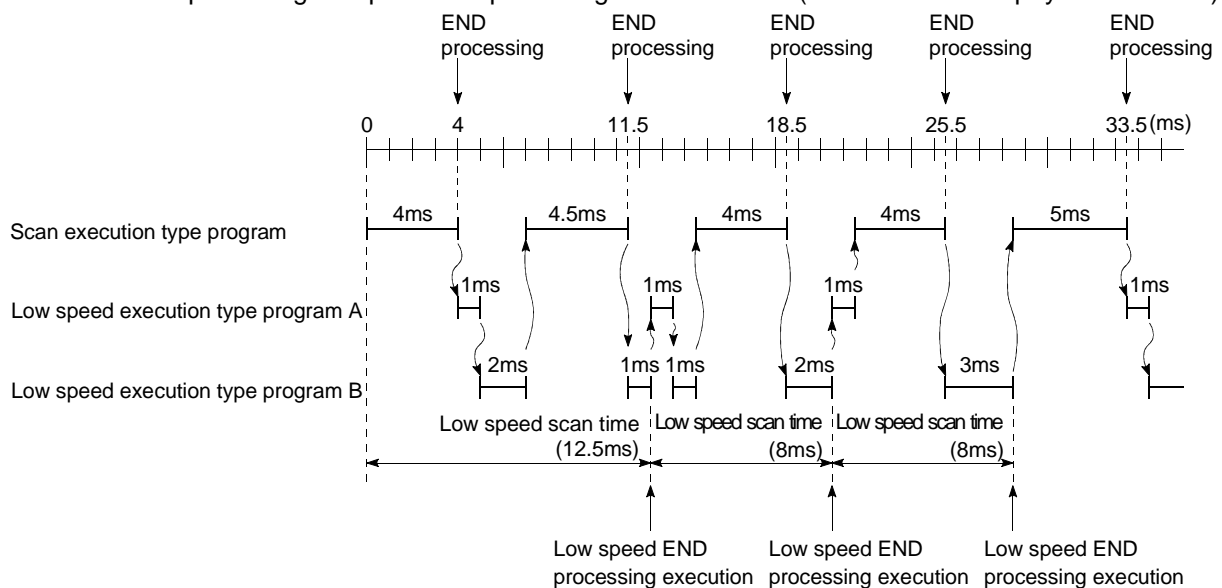
- Constant scan time : 8ms
- Total scan execution type program time : 4ms to 5ms
- Execution time of low speed execution type program A : 1ms
- Execution time of low speed execution type program B : 3ms
- END processing/low speed END processing : 0ms (0 ms is used to simplify the illustration)



(2) Low speed program execution time setting

The low speed execution type program is operated under the following conditions as shown below.

- Low speed program execution time : 3ms
- Total scan execution type program time : 4ms 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 (0 ms is used to simplify the illustration)

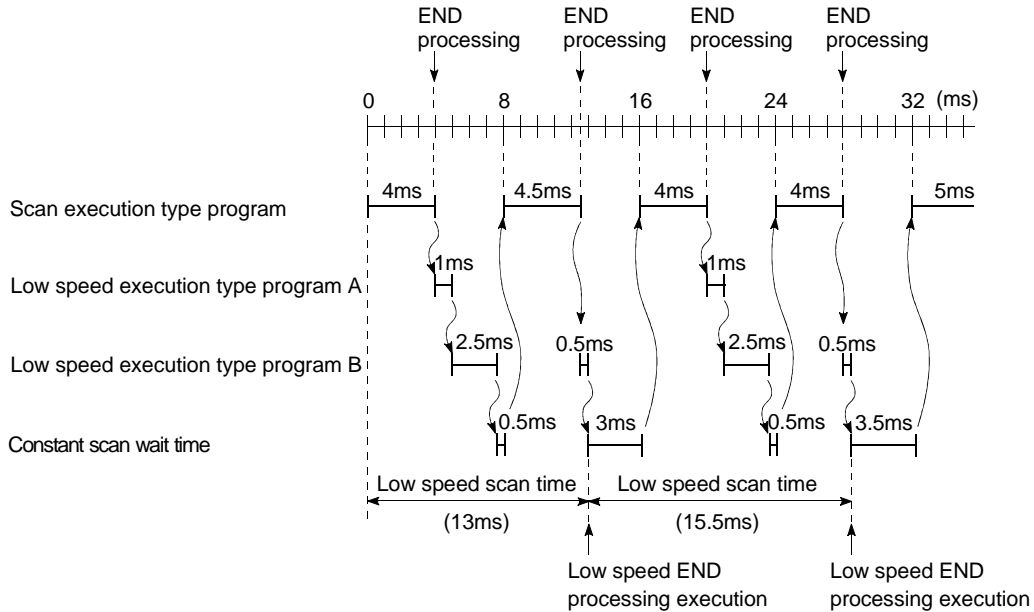


2 : Synchronous method

(1) Constant scan time setting

The low speed execution type program is operated under the following conditions as shown below.

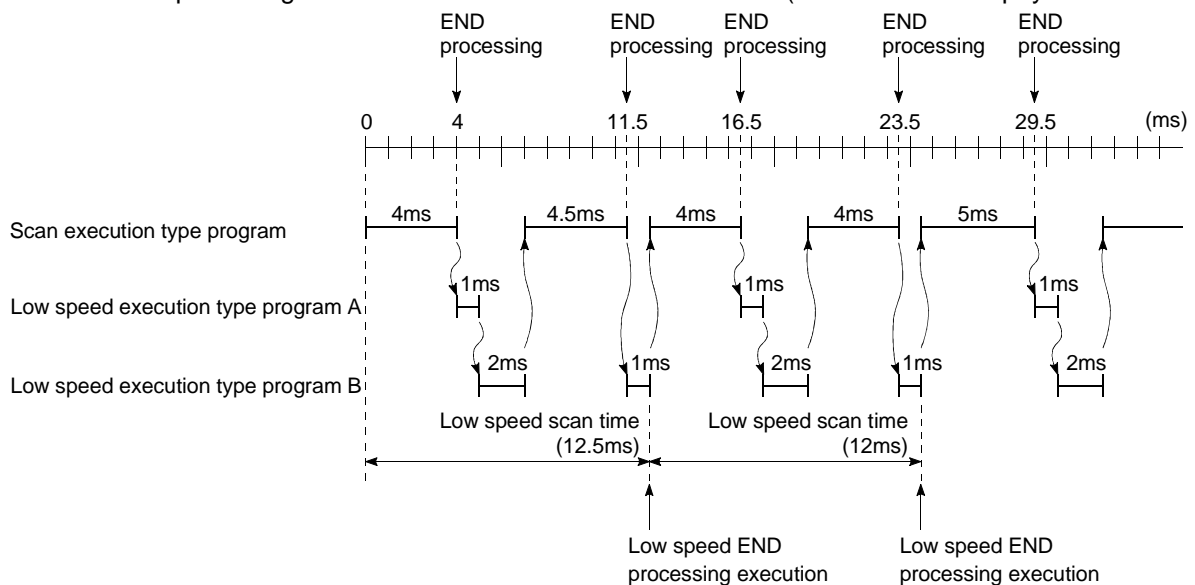
- Constant scan time : 8ms
- Total scan execution type program time : 4ms to 5ms
- Execution time of low speed execution type program A : 1ms
- Execution time of low speed execution type program B : 3ms
- END processing/low speed END processing : 0ms (0 ms is used to simplify the illustration)



(2) Low speed program execution time setting

The low speed execution type program is operated under the following conditions as shown below.

- Low speed program execution time : 3ms
- Total scan execution type program time : 4ms to 5ms
- Execution time of low speed execution type program A : 1ms
- Execution time of low speed execution type program B : 3ms
- END processing : 0ms (0 ms is used to simplify the illustration)



- (4) Precautions for creating Low speed execution type programs
- (a) See Section 10.6.1 for details on index register processing when switching from a scan execution type program to a low speed execution type program.
  - (b) See Section 10.6.2 for details on index register processing when an interrupt program/a fixed scan execution type program is executed during execution of low speed execution type program.
  - (c) The low speed program execution time should be set so that sum of the [scan time] + [low speed program execution time] sum is less than the WDT setting value.
  - (d) The COM instruction can not be used in low speed execution type programs.
  - (e) Low speed execution type programs can also be executed with scans that execute the initial execution type programs.  
Establish an interlock with SM402 and SM403 for the circuit that validates the low speed execution type program's operation after the scan execution type program has been executed.
  - (f) When the "constant scan time" and "low speed program execution time" have been set, "PRO. TIME OVER (error code: 5010)" will occur if (surplus time of constant scan) < (low speed program execution time).

(5) Low speed END processing

The low speed END processing is performed when all the low speed execution type programs are executed.

The following processing is performed for the low speed END processing.

- Low speed program special relay/special register setting
- Low speed execution type program write during RUN
- Low speed scan time measurement
- Low speed execution type program watch dog timer reset

When the low speed END processing is completed, the low speed execution type program is executed from the beginning again.

POINT
(1) During execution of low speed execution type programs, the constant scan time may deviate by the amount of "the maximum instruction processing time + low speed END processing time".



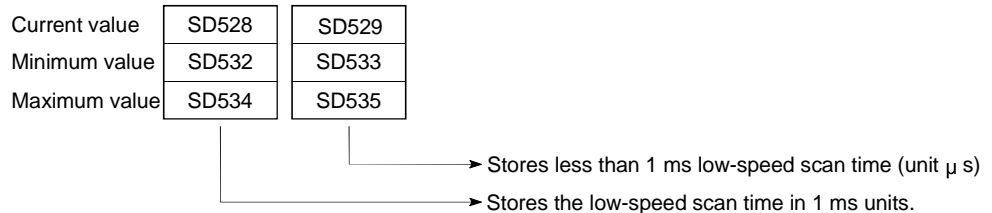
(6) Low speed scan time

- (a) The low speed scan time is the total time required for low speed execution type program execution and low speed END processing.

If multiple low speed execution type programs are executed, the low speed scan time is the total time required to execute all the programs, plus the low speed END processing time.

When an interrupt program/fixed scan execution type program is executed, the value added with the interrupt program/fixed scan execution type program's execution time will become the low speed scan time.

- (b) The low speed scan time is measured by the High Performance model QCPU, and the result is stored in special registers (SD528 to SD535). \*1  
The low speed scan time can therefore be checked by monitoring the SD528 to SD535 special registers.



If the SD528 value is 50, and the SD529 value is 400, the low speed scan time is 50.4 ms.

**POINT**

\*1: The accuracy of the scan time stored at the special registers is  $\pm 0.1$  ms.  
The scan time count will continue even if a watchdog time reset instruction (WDT) is executed in the sequence program.

**(7) Low speed execution monitor time**

The execution time of the low speed execution type program can be monitored by this timer. The default value is not set.

When monitoring the execution time of the low speed execution type program, designate the low speed execution monitor time in a 10 to 2000 ms range at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box. (Setting unit: 10 ms)  
If the execution time of the low speed execution type program exceeds this timer setting, a "PRG TIME OVER" error occurs.

**POINT**

The low speed execution time measurement occurs at low speed END processing. Therefore a "PRG TIME OVER" error will occur if the low speed execution monitor time (t) is designated as 100 ms, and the measured low speed scan time at low speed END processing exceeds 100 ms.

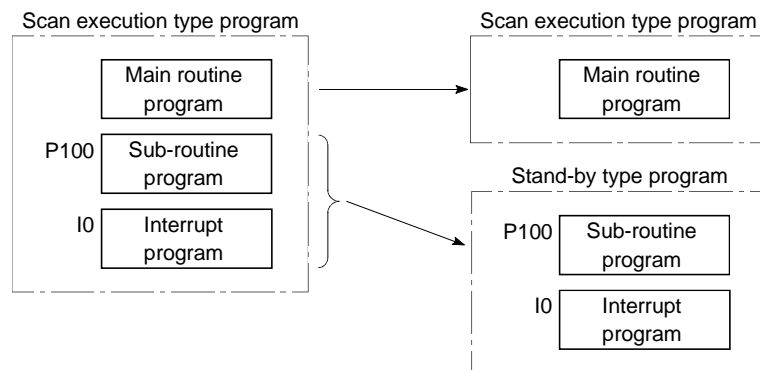
4.2.4 Stand-by type program

(1) Definition of stand-by type program

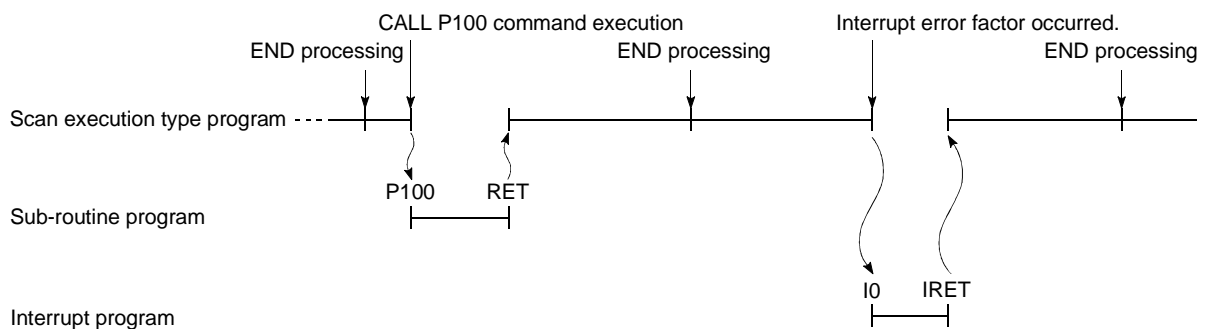
- (a) Stand-by type programs are executed only when requested.
- (b) Stand-by type programs are used for the following applications.
  - 1) Placing programs in the library  
Sub-routine and interrupt programs are converted to stand-by type programs which are managed separately from the main program.
  - 2) Changing the program setup  
Main routing programs registered as stand-by type programs can execute the programs required for control by converting them to scan execution type programs. They will be reconverted to stand-type programs after they completes the program execution.

(2) Stand-by type program applications

- (a) Placing programs in the library
  - 1) This application is used to manage sub-routine and interrupt programs separately from the main routine program. Multiple sub-routine and interrupt programs can be created for a single stand-by type program.



- 2) When stand-by type program execution is completed, the program, which was active before the stand-by type program was executed, is executed. A stand-by type program's sub-routine and interrupt programs are executed as shown below.





(b) Changing the program setup

- 1) Create a program compatible with all programs and use it only to execute necessary programs.

Programs designated as stand-by type programs in the "(PLC) Parameter" dialog box can be converted to scan execution type programs and executed in a sequence program.

Change the execution type in the High Performance model QCPU by using PSCAN, PLOW, PSTOP, and POFF instructions. (See Section 4.2 3.)

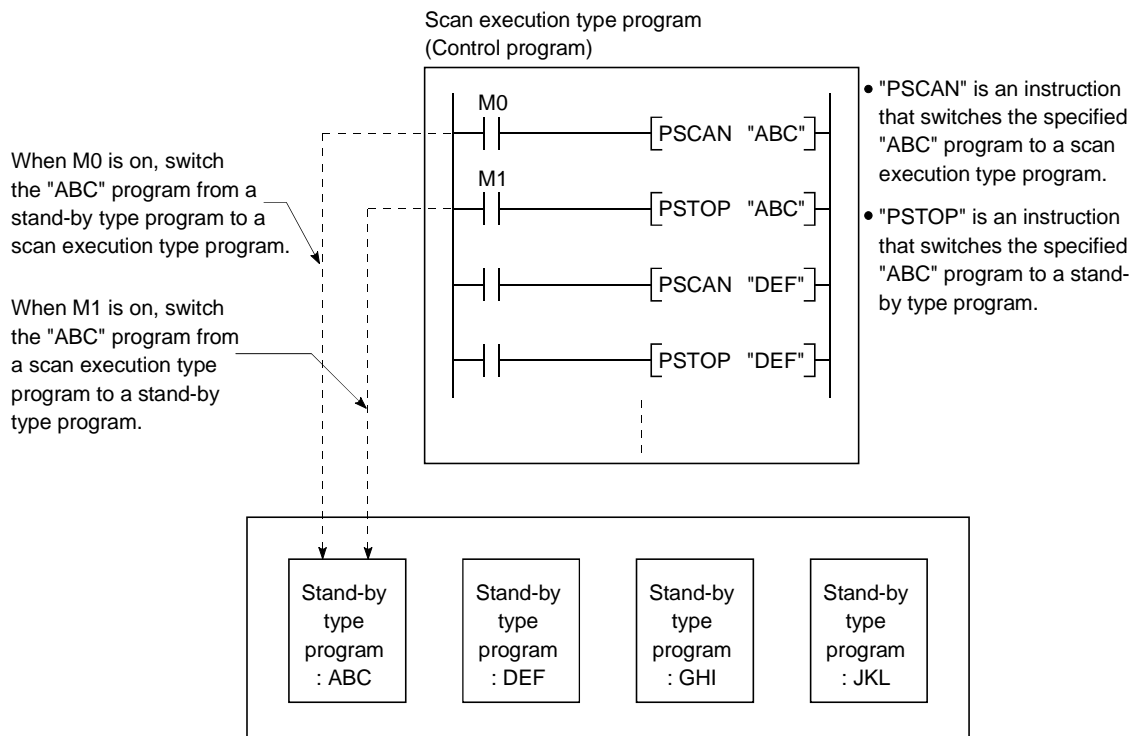
(c) The following methods can be used to convert a program which is to be executed.

- 1) Selecting the program to be executed from a single management program:

- Convert a stand-by type program that meets the designated conditions to a scan execution type program by using a constantly executed scan execution type program as the management program. Then execute the converted program.

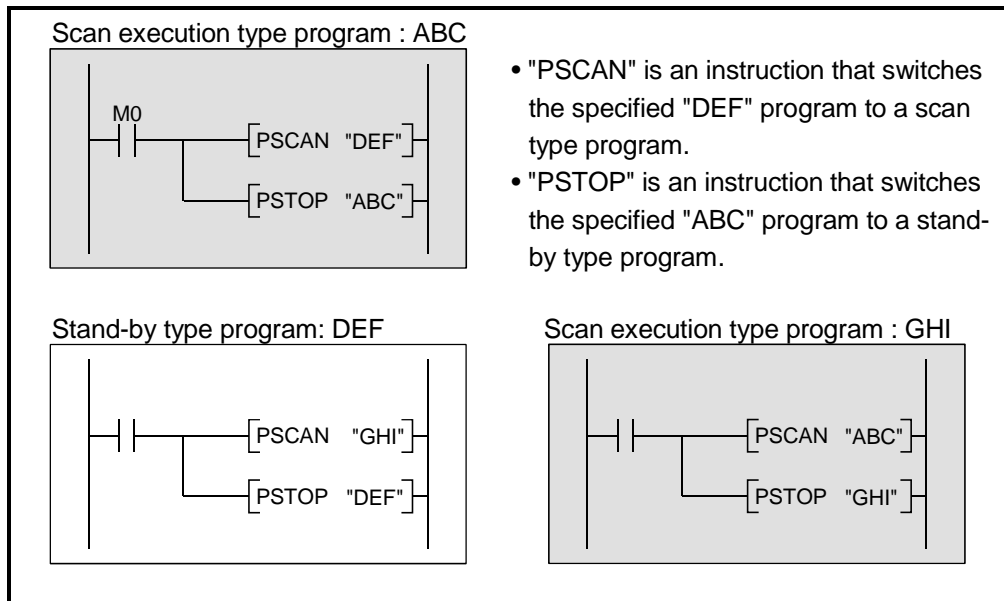
Scan execution type programs which are not required can be converted to stand-by type programs.

- Execute types of "ABC", "DEF", "GHI" and "JKL" stand-by type programs are converted as shown below.



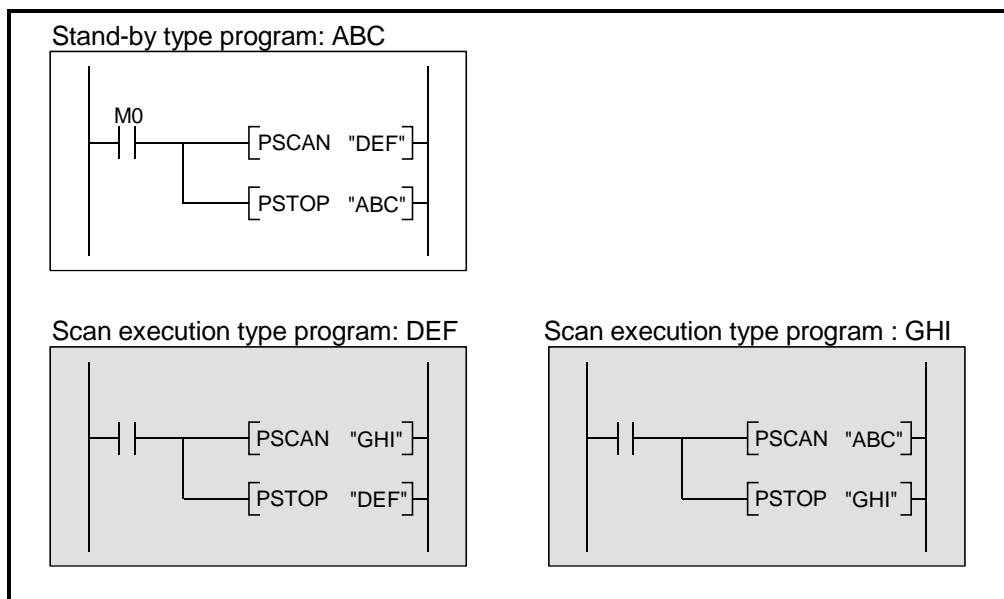
- 2) When changing the execute type of the scan execution type programs and stand-by type programs by the scan execution type programs having the condition for switching the execution type.
- The scan execution type program being executed changes the next program to be executed from a stand-by type program to a scan execution type program.
  - If the condition realizes when "ABC" and "GHI" programs have been set to scan execution type, and "DEF" program to stand-by type, the execute types of "ABC" and "DEF" programs are switched as shown below.

[Before execution of PSCAN and PSTOP instructions]

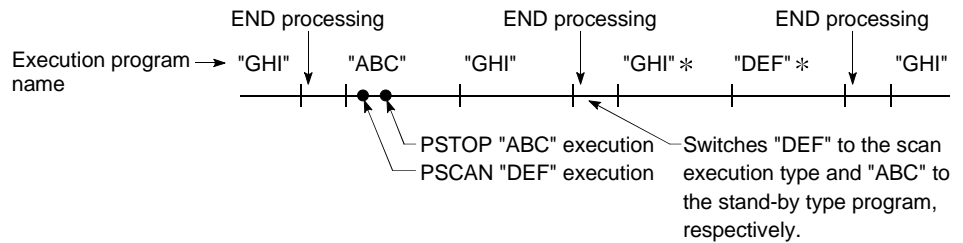


↓ When M0 is on

[After execution of PSCAN and PSTOP instructions]



- (d) The execute type of program is switched at END processing. The program execute type does not change while the program is being executed. If different execute type is specified for a same program in a same scan, the last-specified execute type becomes effective.



**REMARK**

- 1) \*: The "GHI" and "DEF" programs are executed in the order as set at the "Program" tab screen in the "(PLC) Parameter" dialog box.

(3) Precautions for creating stand-by type programs

(a) Because current value is updated and contact ON/OFF is switched when the OUT T[] instruction is executed, timers cannot be used in stand-by type programs.

(b) Gathering sub-routine programs in a single program

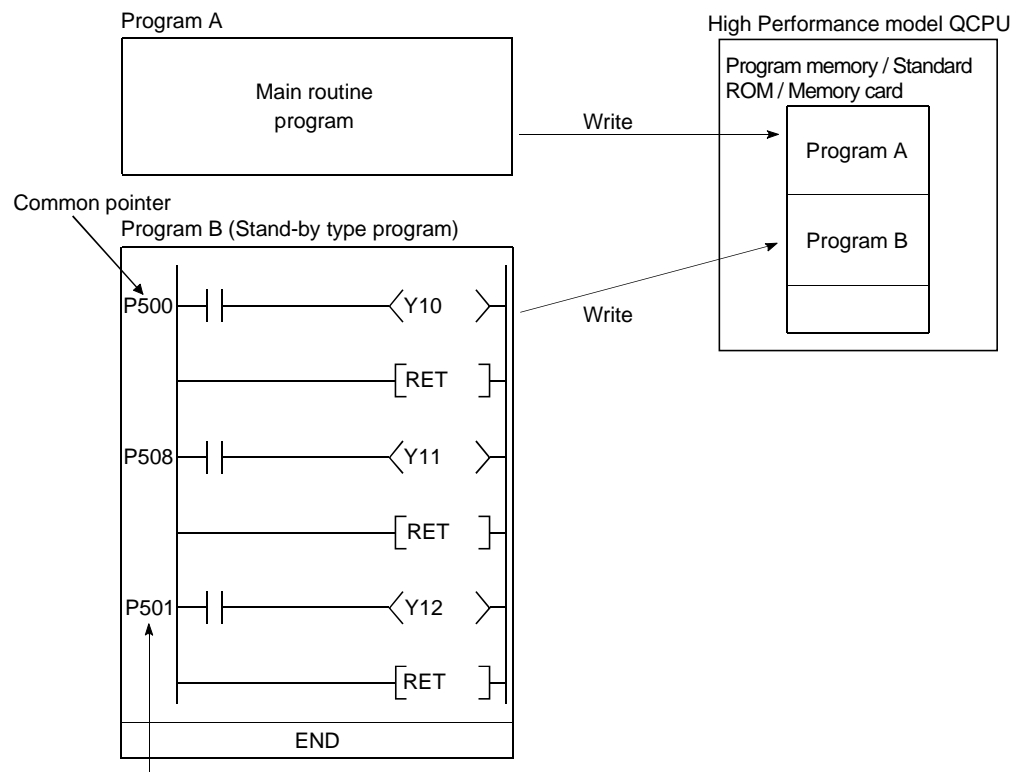
1) Create the sub-routine programs in order beginning from step 0 of the stand-by type program. An END instruction is required at the end of the sub-routine program.

2) Because there are no restrictions on the order of creating sub-routine programs, the pointer numbers need not be assigned in ascending order when creating multiple sub-routine programs.

3) Use common pointers. \*

Sub-routine programs with common pointers can be called from all programs executed by High Performance model QCPU.

(If local pointers are used, the stand-by type program's sub-routine programs will not be executed.)



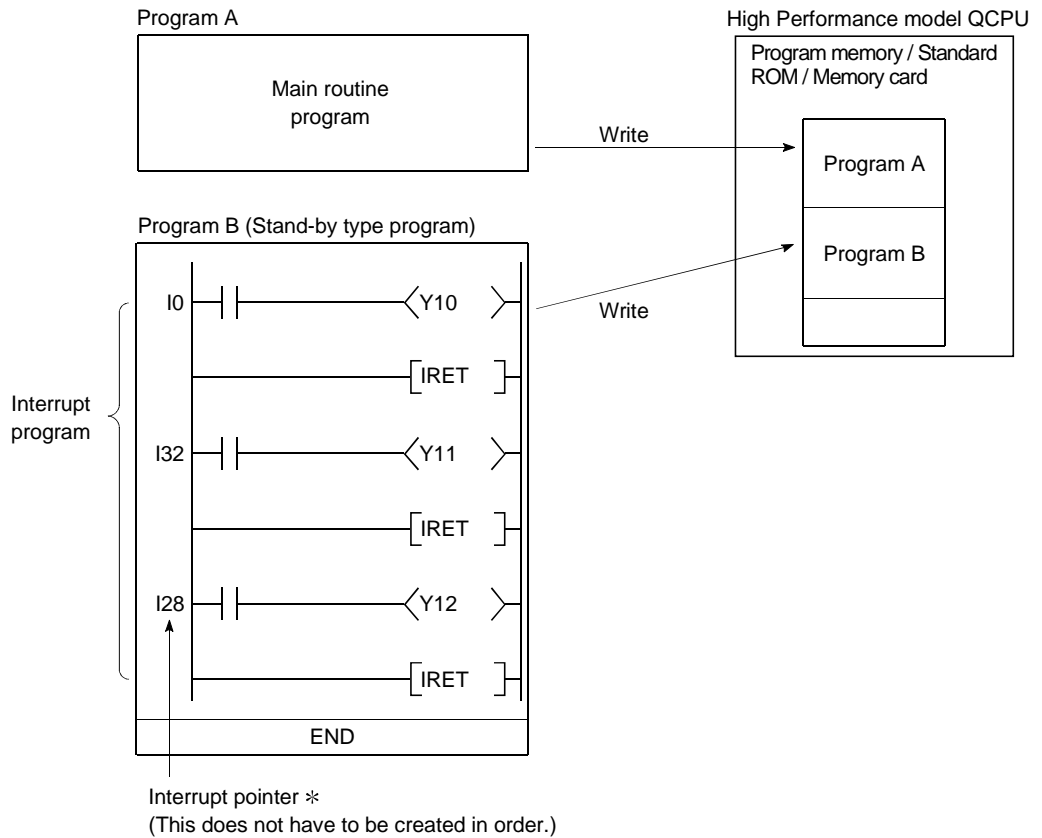
Use a common pointer. \*  
(This does not have to be created in order.)

4) See section 10.13.1 for execution of a sub-routine program that contains local devices.

**REMARK**

\*: See Section 10.9 for details on common pointers and local pointers.

- (c) Gathering interrupt programs in a single program
- 1) Create the interrupt programs in order beginning from step 0 of the stand-by type program.  
An END instruction is required at the end of the interrupt program.
  - 2) Because there are no restrictions on the order of creating interrupt programs, the pointer numbers need not be assigned in ascending order when creating multiple interrupt programs.



**REMARK**

\*: See Section 10.10 for details on interrupt pointers.

4.2.5 Fixed scan execution type program

(1) Definition of fixed scan execution type program

- (a) This program is executed at specified intervals.  
Without describing an interrupt point and IRET instruction, a fixed scan execution can be performed for each file.
- (b) The type of execution is set to "Fixed Scan" at the "Program" tab screen in the (PLC) parameter dialog box.

(2) Execution of fixed scan execution type program

- (a) Fixed scan execution type programs are executed at specified cyclic time intervals.  
When multiple fixed scan execution type programs have reached there simultaneously, they are executed in the ascending order set at the "Program" tab screen in the "(PLC) Parameter" dialog box.
- (b) Set the periodic interval at the "Program" setting tab screen in "(PLC) Parameter" dialog box. The setting range varies with the set unit.
  - When the unit is "ms": 0.5 to 999.5ms
  - When the unit is "s": 1 to 60s
- (c) When the specified times of fixed scan execution type programs and interrupt programs (I28 to I31) have come simultaneously, the priority of execution is given to the interrupt programs.
- (d) Execution during network refreshing  
When the execution conditions of fixed scan execution type programs are established during the network refreshing, the network refresh is suspended, and interrupt programs are executed.  
Therefore, even if the "block assurance of cyclic data for each station" is made in the MELSECNET/H network system, the above operation will not be assured when a device set to be refreshed is used in the interrupt programs. \*1

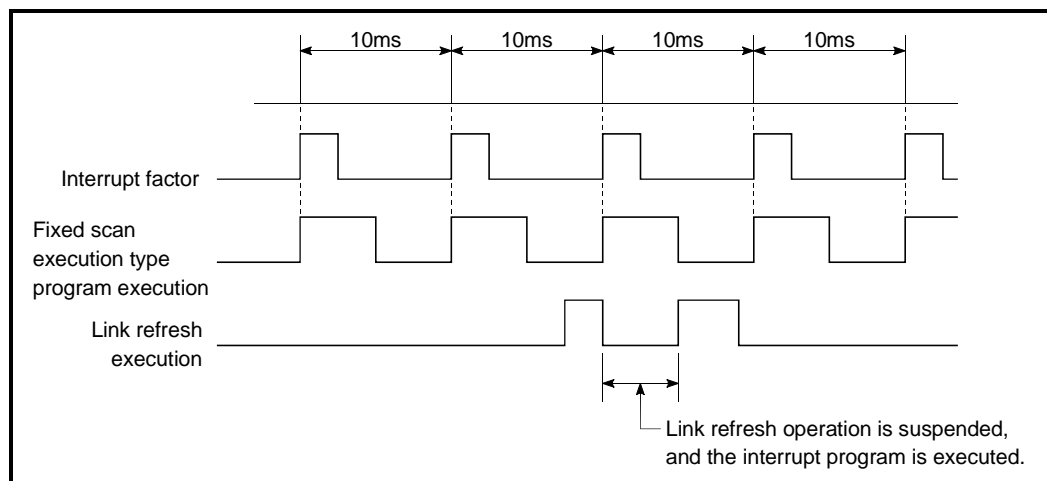


Fig. 4.5 Execution of Fixed Scan Execution Type Programs during Network Refreshing

**REMARK**

- \*1: Refer to the following manual on the block assurance of cyclic data for each station.
  - Q-Corresponding MELSECNET/H Network System Reference Manual

- (e) Execution during END processing:  
When the execution condition of fixed scan execution type programs are established during the wait time of END instruction while the constant scan is executed, the fixed scan execution type programs are executed.
- (f) Perform the processing of the index register when the program is switched from the scan execution type program to the fixed scan execution type program by seeing Section 10.6.2.

(3) Setting of fixed scan execution type program for high speed execution and overhead time

When fixed scan execution type programs are executed, the processing below is performed.

- Save and return of index register
- Save and return of file name of file register in use

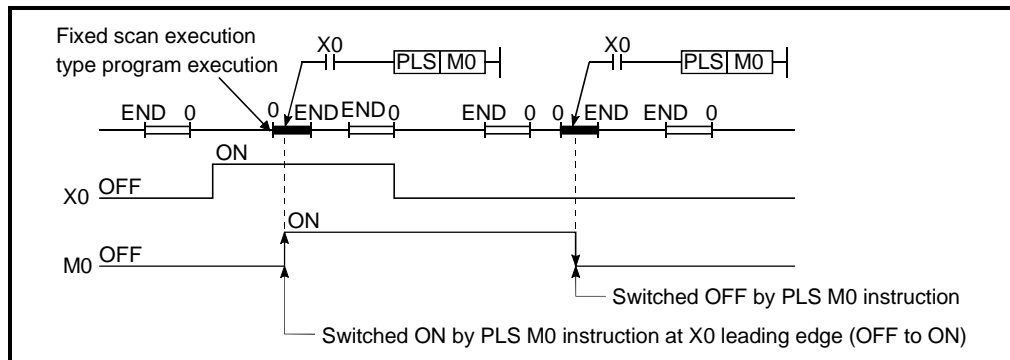
If "High Speed Execution" is selected from the interrupt program/fixed scan execution type program at the "PLC system" tab screen in the "(PLC) Parameter" dialog box, the processing above will not be performed.

As a result, the overhead time for the fixed scan execution type programs can be reduced.

CPU type	Overhead time	
	High speed execution is not selected	High speed execution is selected
Q02CPU	380 μs	230 μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	165 μs	100 μs

(4) Cautions on programming

- (a) If a device is turned ON in a fixed scan execution type program by a PLS instruction, it is kept ON until the same type of the fixed scan execution type program is executed again.



- (b) During the execution of a fixed scan execution type program, interruption is prohibited (DI). Therefore, do not execute EI/DI instructions during the programming of the fixed scan execution type program.
- (c) During the programming of a fixed scan execution type program, a timer cannot be used.  
Because the timer updates the current values and turns ON/OFF at the time of execution of OUT T[ ] instruction, if the timer is used during the programming in the fixed scan execution type program, the current values will be updated only when the fixed scan execution type program is executed, and normal measurement will be disabled.

- (d) The following commands cannot be used in the fixed scan execution type program.
  - COM
  - ZCOM
- (e) When a fixed scan execution type program is executed, an interruption must be allowed by an EI instruction in the initial execution type program/scan execution type program.
- (f) When the interrupt program/fixed scan execution type program is executed at a measuring time such as the scan time or execution time, the values of the interrupt program/fix scan execution type program are added to the measured time.

Thus, if the interrupt program/ fixed scan execution type program is executed, the values stored in the following special registers and GX Developer monitor values will become longer than when the interrupt program/ fixed scan execution type program is not executed.

1) 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
- SD532, SD533: Minimum scan time for low speed
- SD534, SD535: Maximum scan time for low speed
- SD540, SD541: END processing time
- SD542, SD543: Constant scan wait time
- SD544, SD545: Cumulative execution time for low speed execution type programs
- SD546, SD547: Low speed execution time
- SD548, SD549: Scan program execution time
- SD551, SD552: Service interval time

2) GX Developer monitor values

- Execution time measurement
- Scan time measurement
- Constant scan



### 4.3 Operation Processing

#### 4.3.1 Initial processing

This is a preprocessing for sequence operation execution, and is performed only once as shown in the table below.

When the initial processing is completed, the High Performance model QCPU goes in the RUN/STOP switch setting status. (See Section 4.4.)

Initial processing item	High Performance model QCPU 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	○	○	×
PLC parameter check	○	○	○
Multiple CPU system parameter value equality check	○	○	○
Device initialization of the range not latched (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	○	○	○

○ : executed, × : not executed

#### REMARK

\*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 from STOP to RUN (RUN LED flickers) to STOP to RUN.)

Fully note in the above switch operation that normal operation may not be performed since the previous data may not be maintained in the case of the pulse conversion instruction (PLS, \_P) depending on the program change.

#### 4.3.2 I/O refresh (I/O module refresh processing)

In I/O refresh, an input (X) is received from the input module/intelligent function module, and output (Y) of the High Performance model QCPU is sent to the output module/intelligent function module.

The I/O refresh is executed before the sequence program operation starts.

During constant scan execution, the I/O refresh is executed after the constant scan delay time has elapsed.

(The I/O refresh is executed at each constant scan cycle.)

### 4.3.3 Automatic refresh of the intelligent function module

When automatic refresh of intelligent function modules is set, communication with the intelligent function modules of the designated data is performed.

Refer to the manual of the intelligent function modules, for details on the automatic refresh setting of intelligent function modules.

### 4.3.4 END processing

This is a post-processing to return the sequence program execution to step 0 after completing the whole sequence program operation processing once.

- (a) When a refresh request is made from the network module, refresh processing is performed.
- (b) When the trace point of the sampling trace is set at every scan (after END instruction execution), the set device status is stored in the sampling trace area.

POINT
(1) When the constant scan function (See Section 7.2) 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. See Section 4.2.3 for details on the low speed execution type program and low speed END processing.

4.4 RUN, STOP, PAUSE Operation Processing

The High Performance model QCPU has three types of operation status; RUN, STOP and PAUSE status.

The High Performance model QCPU operation processing is explained below:

- (1) **RUN Status Operation Processing**
  - (a) RUN status indicate that the sequence program operation is performed from step 0 to END (FEND) instruction to step 0 repeatedly.
  - (b) When entering the RUN status, the output status saved at STOP status is output by setting "Previous status" as "Output mode at STOP to RUN" at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.
  - (c) It usually takes 1 to 3 seconds to prepare for starting the sequence program operation since STOP status is switched to RUN status, though it might be longer depending on the system.
- (2) **STOP Status Operation Processing**
  - (a) STOP status indicates that the sequence programs are stopped by RUN/STOP switch or remote STOP function. (See 7.6.1 for details on remote STOP function.)  
High Performance model QCPU might enter STOP status when a stopping error occurs.
  - (b) When entering the STOP status, save the output status and turn off all output.  
The device memory of other than the output (Y) is retained.
- (3) **PAUSE Status Operation Processing**
  - (a) The PAUSE status indicates that the sequence program operations are paused by remote PAUSE function while maintaining the output and device memory status. (See Section 7.6.2 for details on remote PAUSE function.)
- (4) **High Performance model QCPU Operation Processing with RUN/STOP Status**

Operation Processing of High Performance model QCPU RUN/STOP status	Sequence program operation processing	External output	Device memory	
			M, L, S, T, C, D	Y
RUN to STOP	Executes up to the END instruction and stops.	OS saves the status immediately before the STOP status and all points turn off.	Maintains the status immediately before the STOP status.	OS saves the status immediately before the STOP status and all points turn off.
STOP to RUN	Starts at step 0.	Determined by the "STOP → RUN-time output mode" in the PLC parameter dialog box.	Starts executing the operation from the status immediately before the STOP status. When the device initial value function is designated, however, the device initial value is set or the local devices are cleared.	Determined by the "STOP → RUN-time output mode" in the PLC parameter dialog box.

POINT
<p>The High Performance model QCPU performs the following in any of RUN, STOP, and PAUSE status:</p> <ul style="list-style-type: none"> <li>• I/O module refresh processing</li> <li>• Data communication with GX Developer and serial communication module</li> <li>• Refresh process of MELSECNET/H and CC-Link</li> </ul> <p>For this reason, I/O monitor and test operation using GX Developer, reading/writing from the serial communication, communication with another station using MELSECNET/H, and communication with a remote station over the CC-Link can be made even in the STOP or PAUSE status.</p>

#### 4.5 Operation Processing during Momentary Power Failure

The High Performance model QCPU detects a momentary power failure when the input power voltage supplied to the power supply module is lower than the regulated ranges.

When the High Performance model QCPU detects a momentary power failure, the following operation processing is performed:

- (1) When momentary power failure occurs for a period shorter than the permitted power failure time
  - (a) The output is maintained when the momentary power failure occurs, and file name of the file accessed and error history are logged. Then the system interrupts the operation processing. (The timer clock continues.)
  - (b) When there is an SFC continue specification, a system saving processing is performed.
  - (c) When a momentary power failure ends, the operation processing is resumed.
  - (d) Even if the operation is interrupted due to momentary power failure, the watch dog timer (WDT) measurement continues. For example, if the GX Developer PLC parameter mode WDT setting is set at 200 ms, when a momentary failure of 15 ms occurs at scan time 190 ms, the watch dog timer error is set.

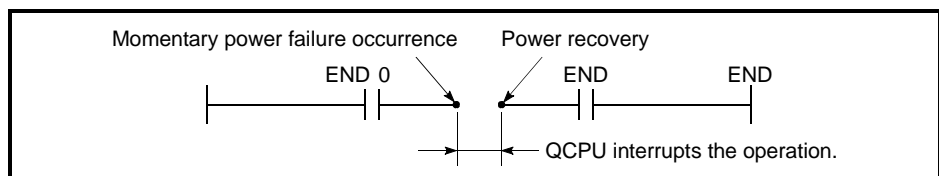


Fig.4.6 Operation Processing When Momentary Power Failure Occurs

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

The High Performance model QCPU starts initially. (PLC power is turned on.)

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

- Power ON
- Resetting using RESET/L. CLR switch
- Remote setting using GX Developer

## 4.6 Data Clear Processing

### (1) Data clear

The High Performance model QCPU clears all data except for the following, when a reset operation is performed with RESET/L.CLR switch, or power ON to OFF to ON.

- (a) Program memory data (except for when "clear program memory" is set at boot specification.)
- (b) Data in the memory card
- (c) Device data with latch specification (latch clear valid)
- (d) Device data with latch specification (latch clear invalid)
- (e) File register data
- (f) Failure history data (when special register SD storage)

Data in (c) is cleared by operating "latch clear" with the RESET/L.CLR switch, or operating "remote latch clear" from GX Developer.

See Section 7.6.4 for details on the remote latch clear.

### (2) Device latch specification

- (a) Specify the device latch (latch range setting) for each device at the "Device" tab screen in the "(PLC) Parameter" dialog box.

There are two types of latch range settings:

- 1) Valid latch clear key  
Sets the latch range that can be cleared by operating "latch clear" with the RESET/L.CLR switch and remote latch clear.
- 2) Invalid latch clear key  
Sets the latch range that can not be cleared even by operating "latch clear" with the RESET/L.CLR switch or operating "remote latch clear" from GX Developer.

- (b) The devices in which RESET/L.CLR switch is set to invalid can only be cleared by an instruction or GX Developer.

- 1) Instruction to clear method  
Reset with the RST instruction or send "0" with the MOV/FMOV instruction.
- 2) GX Developer clear method  
Clear all device memory in the online PLC memory clear (including latch).  
Refer to the GX Developer operating manual for details of the GX Developer operation methods.

<b>POINT</b>
--------------

To clear file registers or local devices, use the RST instruction to perform a reset operation, or use the MOV/FMOV instruction to transmit "0".
--

<b>REMARK</b>
---------------

Refer to following manual for the MOV/FMOV instruction.

- QCPU (Q mode)/QnACPU Programming Manual (Common instructions)

4.7 I/O Processing and Response Lag

In the direct mode, the batch communication with I/O modules is performed before sequence program operation starts.

By creating a sequence program with direct access I/O, I/O processing can be performed in a direct mode to communicate with I/O module at execution of each instruction in the sequence program.

For details on direct access I/O, see Section 10.2.1 and 10.2.2, respectively.

4.7.1 Refresh mode

(1) Definition of refresh mode

With the refresh mode, batch communication with the I/O modules is performed before the sequence program operation starts.

- (a) Batch reading of the input module ON/OFF information is executed in the High Performance model QCPU's internal input device memory when sequence program operation starts. This ON/OFF data (in the input device memory) is then used for processing when a sequence program is executed.
- (b) The processing result of the output (Y) sequence program is output to the High Performance model QCPU's internal output device memory, and batch output of the ON/OFF data (in output device memory) to the output module is executed when sequence program operation starts.

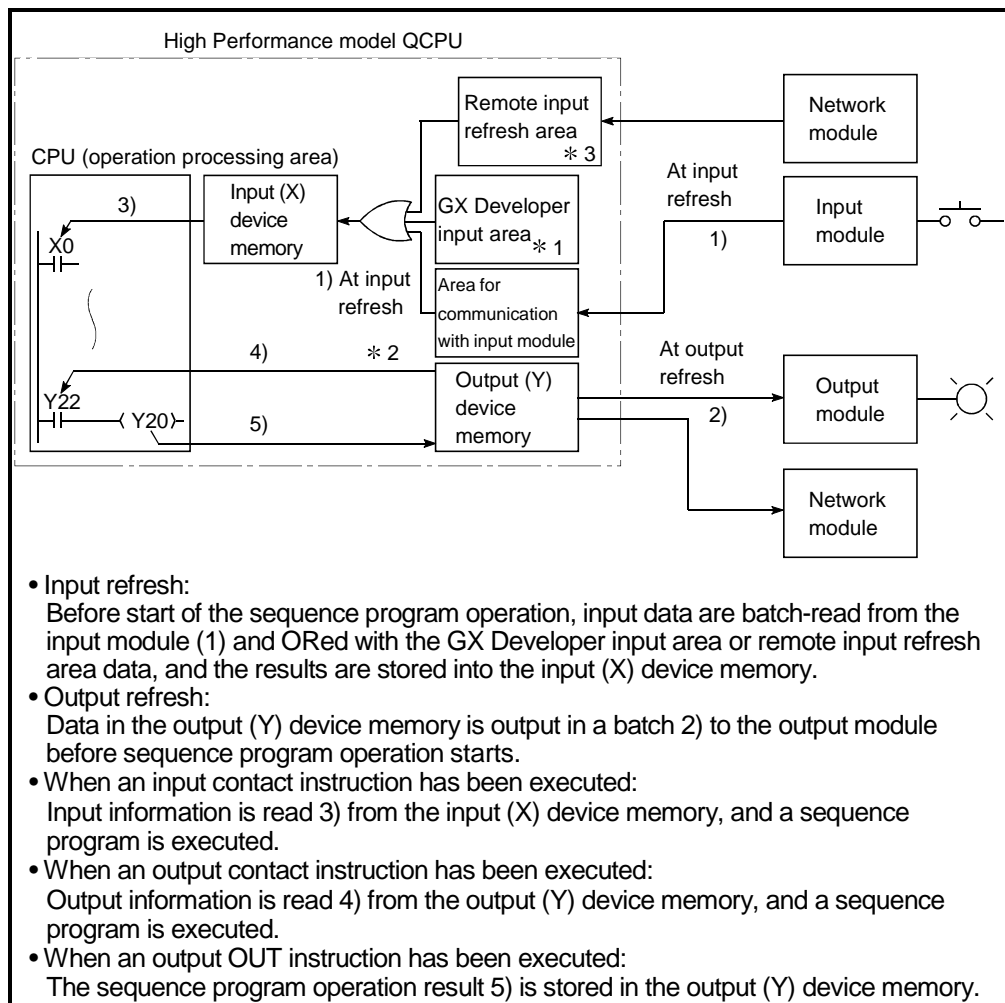


Fig.4.7 I/O Information Flow in Refresh Mode

**REMARK**

- \*1: The peripheral device input area can be switched ON and OFF by the following:
  - Test operation by the GX Developer
  - A network refresh by the MELSECNET/H network system
  - Writing from a serial communication module
  - CC-Link automatic refresh
- \*2: The output (Y) device memory can be switched ON and OFF by the following:
  - Test operation by GX Developer
  - A network refresh by the MELSECNET/H network system
  - Writing from a serial communication module
  - CC-Link automatic refresh
- \*3: The remote I/O refresh area indicates the area used when automatic refresh setting is made to the input (X) with MELSECNET/H and CC-Link. Automatic refresh of the remote input refresh area is executed during END processing.

(2) Response lag

An output module lags max.2 scans behind an input module. (See Fig.4.8)

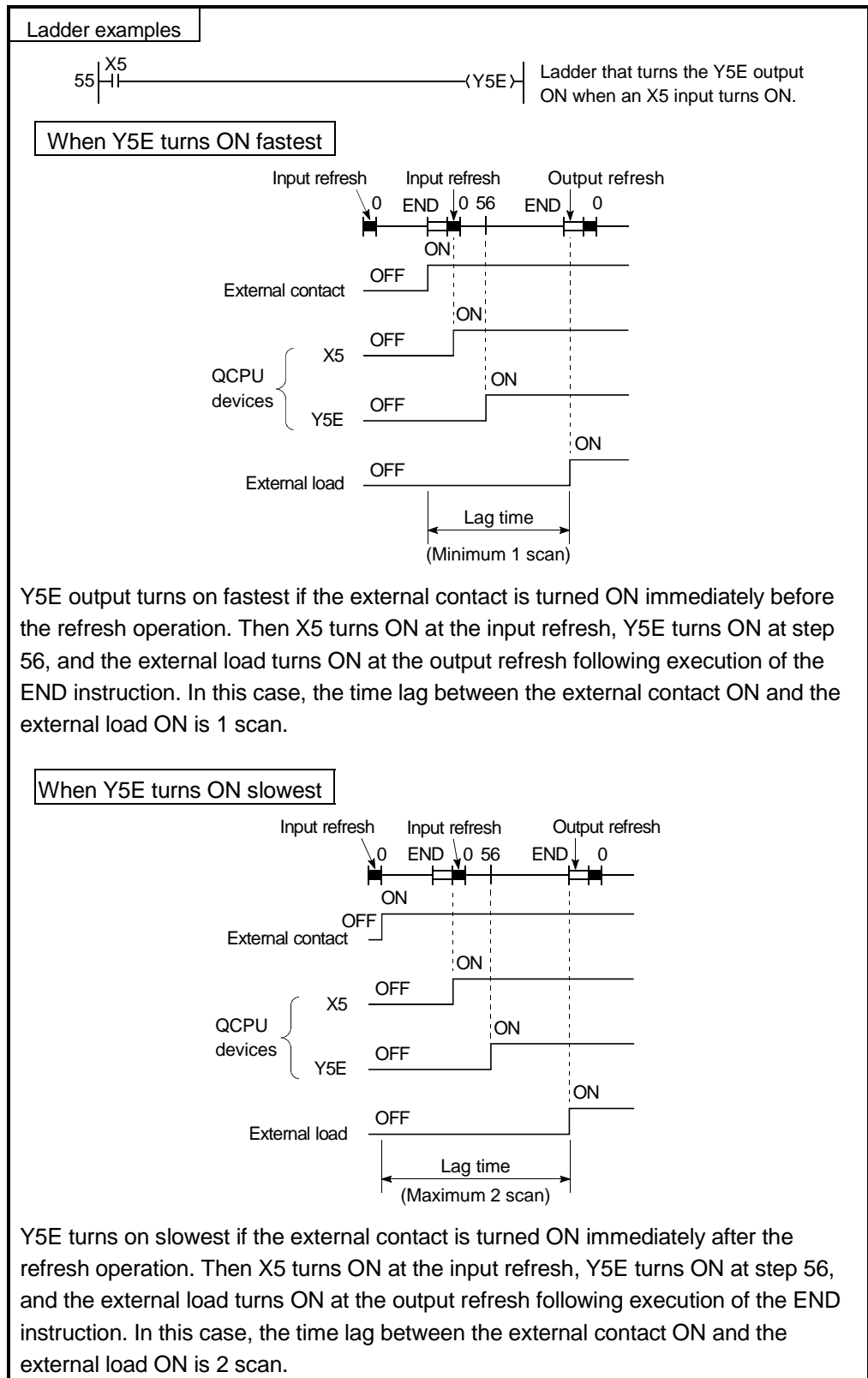


Fig.4.8 Timing chart showing response of Output "Y" when Input "X" turns ON



4.7.2 Direct mode

(1) Definition of direct mode

In the direct mode the communication with the I/O modules is performed when executing sequence program instructions.

With High Performance model QCPU, direct mode I/O processing can be performed by using direct access inputs (DX) and direct access outputs (DY). For details on direct I/O, see Section 10.2.1 and 10.2.2, respectively.

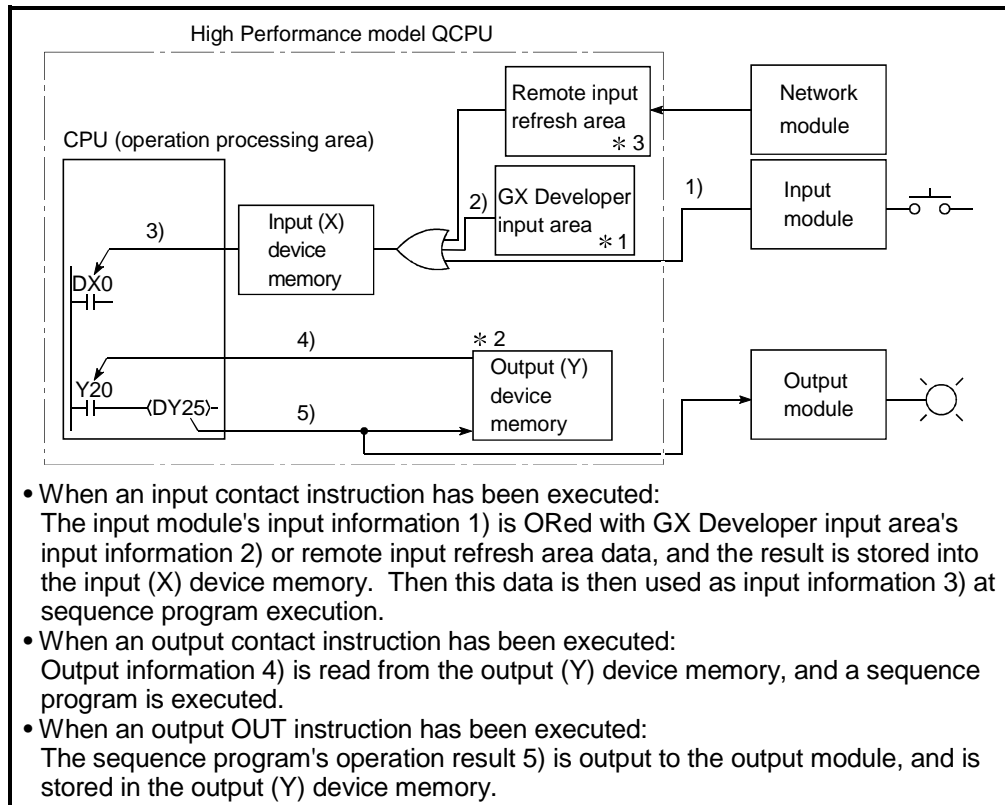


Fig.4.9 I/O Information Flow in Direct Mode

**REMARK**

\*1: The GX Developer input area can be turned ON and OFF by the following:

- Test operation by GX Developer
- Writing from a serial communication module

\*2: The output (Y) device memory can be turned ON and OFF by the following:

- Test operation by the GX Developer
- A network refresh by MELSECNET/H network system
- Writing from a serial communication module
- CC-Link automatic refresh

\*3: The remote input refresh area indicates the area used when automatic refresh setting is made to the input (X) with MELSECNET/H and CC-Link. Automatic refresh of the remote input refresh area is performed during END processing.

(2) Response lag

An output module lags max.1 scan behind an input module. (See Fig.4.10)

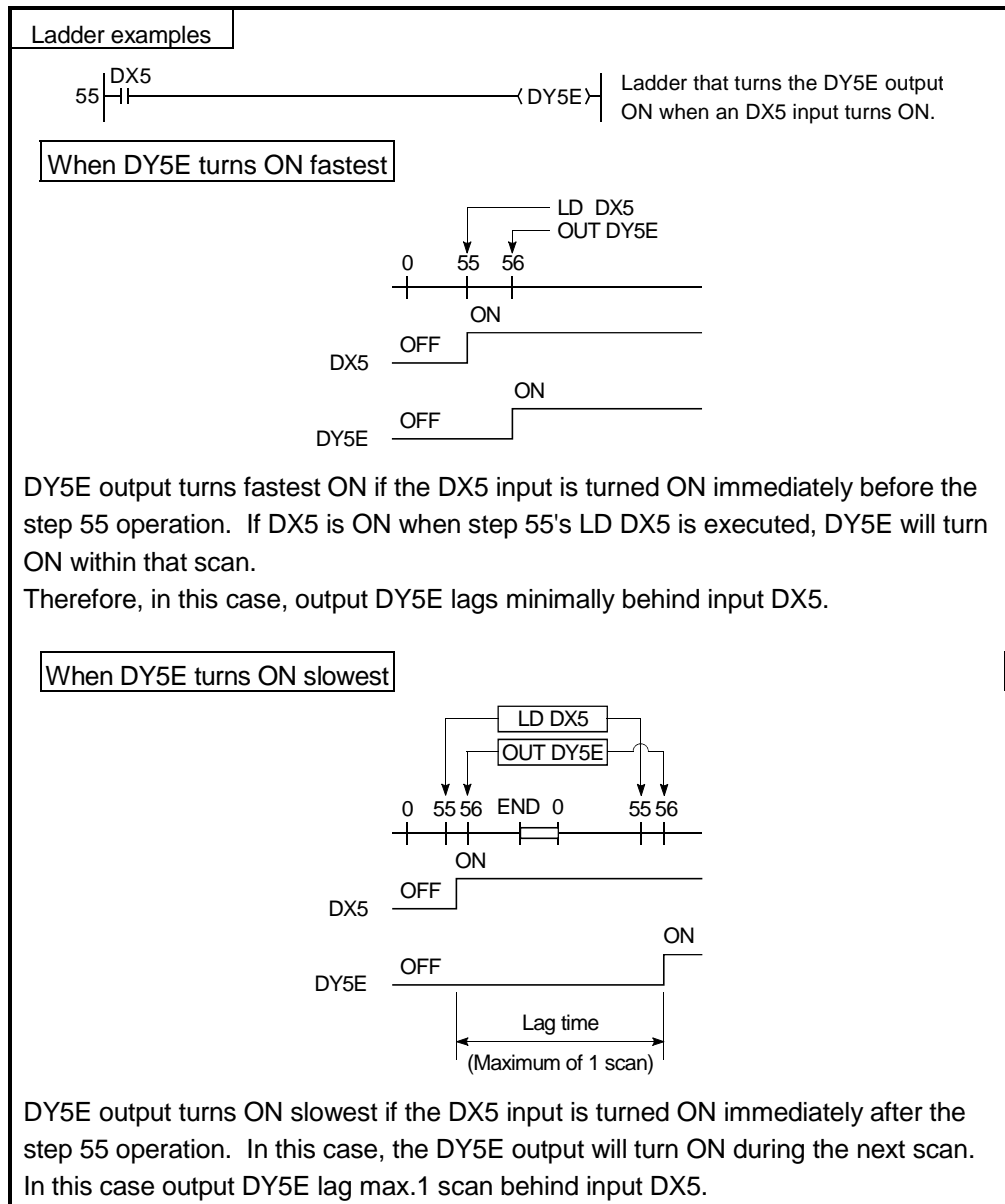


Fig.4.10 Timing chart showing response of Output "Y" when Input "X" turns ON

4.8 Numeric Values which Can Be Used in Sequence Programs

Numeric and alphabetic data are expressed by "0" (OFF) and "1" (ON) numerals in the High Performance model QCPU.

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 High Performance model QCPU.

Real numbers may also be used. (See Section 4.8.4)

The numeric expressions by BIN, HEX, BCD, and Decimal (DEC) notations are shown in Table 4.1 below.

Table 4.1 BIN, HEX, BCD, and Decimal Numeric Expressions

DEC (Decimal)	HEX (Hexadecimal)	BIN (Binary)				BCD (Binary Coded Decimal)			
0	0					0			
1	1					1			
2	2					10			
3	3					11			
•	•					•			
•	•					•			
•	•					•			
9	9					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) External numeric inputs to High Performance model QCPU

When inputting numeric values to the High Performance model QCPU from an external source (such as digital switch), use BCD (binary coded decimal) which allows the same setting as decimal form.

However, the High Performance model QCPU handles the values as BIN data since the operation is based on BIN form, if it uses values set in the BCD form as they are.

Therefore, the High Performance model QCPU operates with numeric values that are different from the set values.

A BIN instruction is provided to convert the BCD input data to the BIN data compatible with the High Performance model QCPU.

A program which converts numeric data to BIN data can be created at the sequence program to allow the settings of numeric values from an external source without being conscious of the corresponding BIN values.

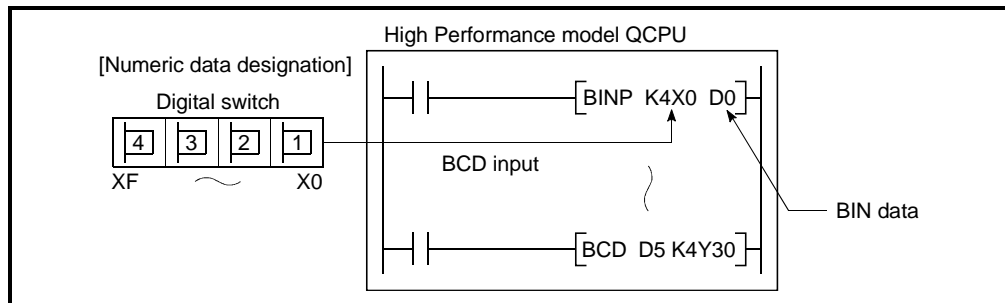


Fig.4.11 Digital Switch Data Input to High Performance model QCPU

(2) External numeric outputs from High Performance model QCPU

A digital display can be used to display numeric data which is output from the High Performance model QCPU.

However, BIN data cannot be displayed at the digital display as it is because the High Performance model QCPU use it for operation.

Therefore BCD instruction is provided for the High Performance model QCPU to convert the BIN data to BCD data. A program which converts BIN data to BCD data can be created in the sequence program in order to display the output data in the same manner as decimal data.

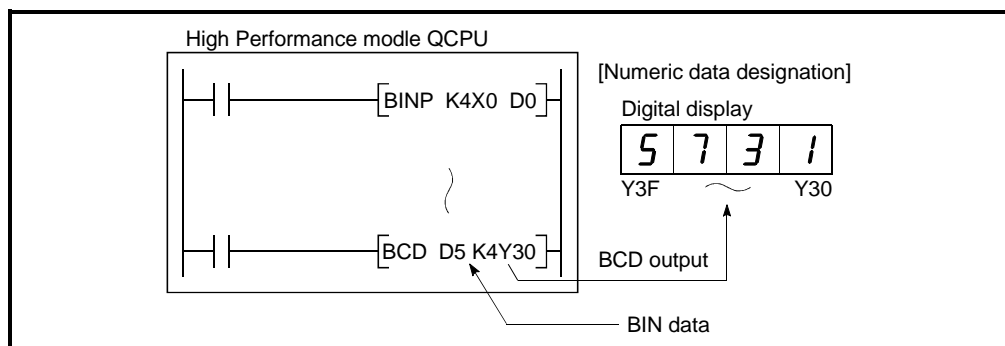


Fig.4.12 Digital Display of Data from High Performance model QCPU

4.8.1 BIN (Binary Code)

(1) Binary code

Binary data 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).

Table 4.2 gives a comparison between binary and decimal notations.

Table 4.2 Comparison between Binary and Decimal Notations

DEC (Decimal)	BIN (Binary)
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011

← Carry  
← Carry  
← Carry

(2) Binary numeric expression

(a) High Performance model QCPU registers (data registers, link registers, etc.) consist of 16 bits, and a "2<sup>n</sup>" value is allocated to each of the register bits. The most significant bit (initial bit) is used to discriminate between "positive" and "negative".

- 1) When most significant bit is "0"...Positive
- 2) When most significant bit is "1"...Negative

The numeric expressions for the High Performance model QCPU registers are shown in Fig.4.13 below.

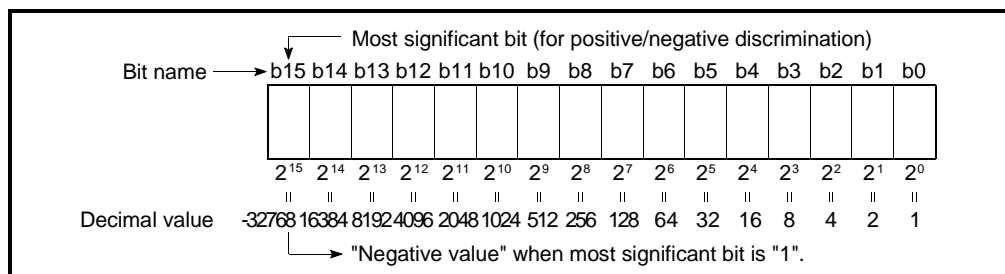


Fig.4.13 Numeric Expressions for High Performance model QCPU Registers

(b) Usable numeric data for High Performance model QCPU

As shown in Fig.4.13, the numeric expression range is -32768 to 32767. Therefore, numeric data within this range can be stored in the High Performance model QCPU registers.

4.8.2 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 A to F are used to represent the numbers 10 to 15.

Then, a carry occurs after F.

Table 4.3 shows numeric expressions by binary, hexadecimal, and decimal notations.

Table 4.3 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

High Performance model QCPU registers (data registers, link registers, etc.) consist of 16 bits.

Therefore, the numeric values expressed in hexadecimal notation can be stored in each register within 0 to FFFF<sub>H</sub> range.

4.8.3 BCD (Binary Coded Decimal)

(1) BCD notation

BCD notation is binary expression with a carry similar to that of the decimal notation.

Though it uses 4-bit representation like hexadecimal notation, it does not use letters A to F.

Table 4.4 gives numeric expressions by binary, BCD, and decimal notations.

Table 4.4 Comparison of BIN, BCD, and DEC Numeric Expressions

DEC (Decimal)	BIN (Binary)	BCD (Binary Coded Decimal)
0	0	0
1	1	1
2	10	10
3	11	11
4	100	100
5	101	101
6	110	110
7	111	111
8	1000	1000
9	1001	1001
10	1010	1 0000
11	1011	1 0001
12	1100	1 0010

   Carry ←

(2) BCD numeric expression

High Performance model QCPU 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.

4.8.4 Real numbers (floating decimal point data)

(1) Real numbers

Real numbers are single precision floating decimal point data.

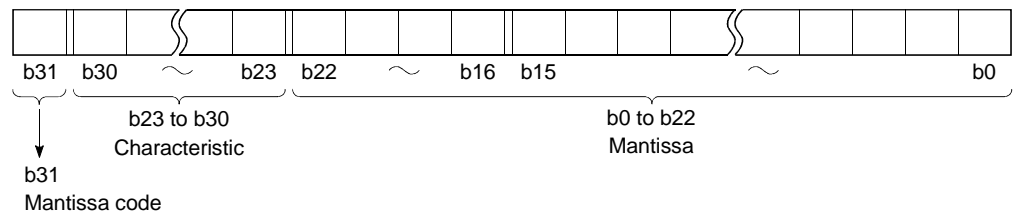
(2) Internal expression of floating decimal point data

The High Performance model QCPU's internal expression of received real number data is explained below.

Real number data is expressed as shown below, using 2 word devices.

1. [Mantissa] × 2<sup>(characteristic)</sup>

The bit configuration used for internal expression of floating decimal point data is shown and explained below.



- Mantissa code: The mantissa code is expressed at b31 as follows.  
0: Positive  
1: Negative

- Characteristic: The "n" of "2<sup>n</sup>" is expressed in various ways at b23 to b30, depending on the b23 to b30 BIN value.

b23 to b30	FFH	FEH	FDH		81H	80H	7FH	7EH		02H	01H	00H
n	Non-numeric	127	126		2	1	0	-1		-125	-126	Non-numeric

- Mantissa: For a binary value of 1.XXXXXX..., the "XXXXXX" portion of the value is expressed at b0 to b22 (23 bits).

(3) Calculation examples

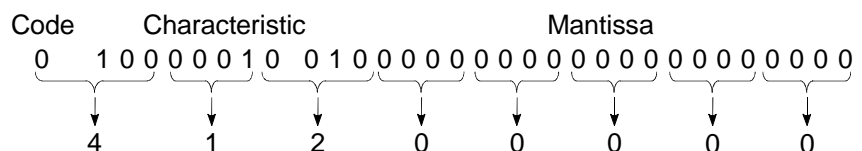
Calculation examples are shown below (the nnnnn "X" indicates an X-system data expression).

(a) Storing "10"

$$(10)_{10} \rightarrow (1010)_2 \rightarrow (1.\underbrace{01000}_{\text{Mantissa}} \times \underbrace{2^3}_{\text{Characteristic}})_2$$

Mantissa code      Positive to 0  
 Characteristic    3 to 82H to (10000010)<sub>2</sub>  
 Mantissa            (010 0000 0000 0000 0000)<sub>2</sub>

Therefore, the data expression will be 41200000H, as shown below.



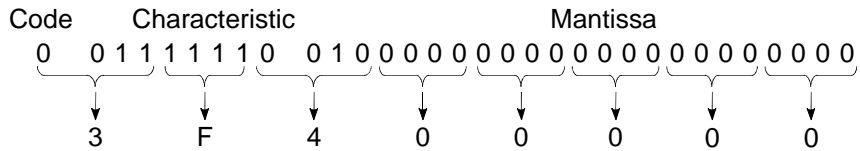


(b) Storing "0.75"

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

Mantissa code            Positive to 0  
Characteristic        -1 to 7EH to (01111110)<sub>2</sub>  
Mantissa                (100 00000 00000 00000 00000)<sub>2</sub>

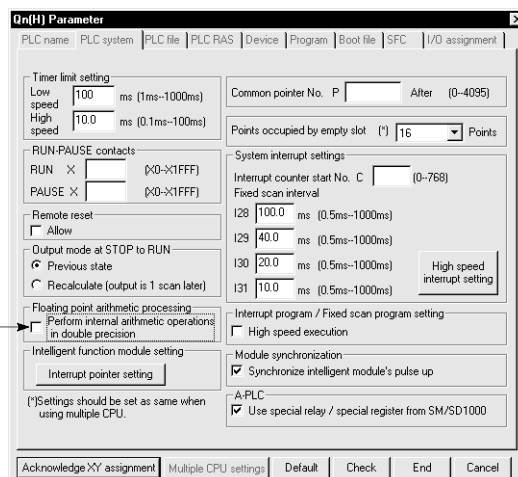
Therefore, the data expression will be 3F400000<sub>H</sub>, as shown below.



**POINT**

- (1) The monitor function for GX Developer permits monitoring the real number data of the High Performance model QCPU. 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.
  - (3) It is possible to select either "Perform internal arithmetic operation in double precision" or "Do not perform internal arithmetic operation in double precision" with the floating point arithmetic processing on the PLC parameter's PLC system settings. (The result of the operation will be short precision regardless of the floating point arithmetic processing setting.)  
It is recommended that "Do not perform internal arithmetic operation in double precision" is selected if increased speed for the real arithmetic operations is required, and "Perform internal arithmetic operation in double precision" is selected if precision is required when applying compatibility with conventional equipment.
- Only internal arithmetic operations will be performed at double precision (64-bits) when "Perform internal arithmetic operation in double precision" is selected (default setting).  
Precision will be increased for commands that use many real arithmetic operations with internal arithmetic operations, such as the SIN command and COS command, when double precision is set.
  - Real arithmetic operations will be performed faster owing to the internal arithmetic operations being performed with short precision (32-bit) when "Do not perform internal arithmetic operation in double precision" is set, and there are also cases where a certain amount of precision will be lost.

To turn off "Perform internal arithmetic operations in double precision", uncheck the check box.



**REMARK**

In binary notation, the portion of the value following the decimal point is calculated as follows:

0.1	1	0	1
↑	↑	↑	↑
This bit expresses $2^{-1}$	This bit expresses $2^{-2}$	This bit expresses $2^{-3}$	This bit expresses $2^{-4}$

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

4.9 Character String Data

(1) Character String Data

The High Performance model QCPU uses ASCII code data.

(2) ASCII code character strings

ASCII code character strings are shown in the Table below.

"00H" (NUL code) is used at the end of a character string.

					0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1				
					0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1				
					0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1				
					0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1				
b8	b7	b6	b5	b4	b3	b2	b1	Column Low	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
									NUL		(SP)	0	@	P	`	p								
									0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
									0	0	1	0	2	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	0	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)	.	>	N	^	n	~				
									1	1	1	1	F	/	?	O	Under line	o	—					

## 5 ASSIGNMENT OF I/O NUMBERS

This section describes the necessary information on the I/O number assignment for the data exchange between High Performance model QCPU and I/O modules or intelligent function modules.

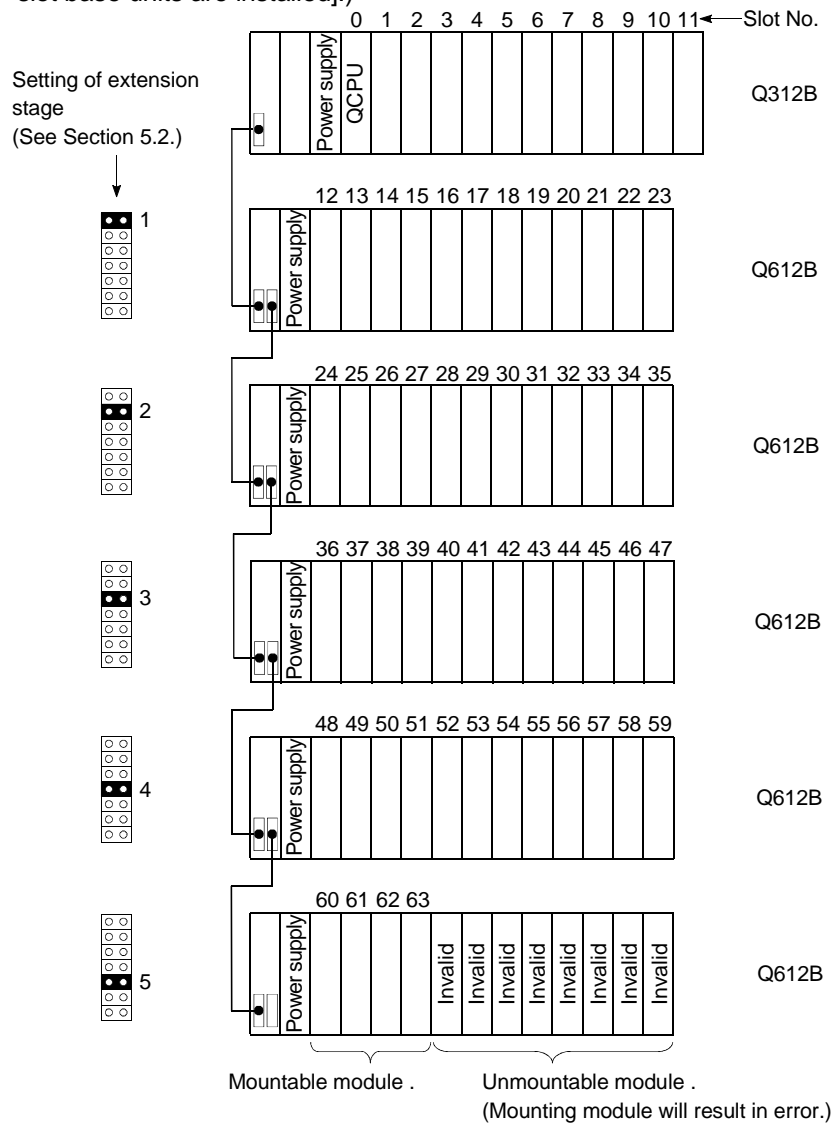
### 5.1 Relationship Between the Number of Stages and Slots of the Extension Base Unit

High Performance model QCPU allows the system configuration using eight base units: one main base unit and seven extension base units.

However, the number of available slots (modules) is limited to 64 slots including empty slots.

An error (SP. UNIT LAY ERR.) occurs when a module (input, output, or intelligent function module) is installed to the 65th or subsequent slots.

Be sure to install modules within the range of 64 slots. (An error does not occur as long as all modules are installed within the range of 64 slots, even if the total number of slots of the main and extension base units results in 65 slots or more [e.g. When 6 12-slot base units are installed].)



5.2 Installing Extension Base Units and Setting the Number of Stages

There are two types of extension base units: Q5□B/Q6□B for mounting of Q-Series modules and QA1S6□B and for mounting AnS Series modules.

(1) Connecting order of extension base units

When using both Q5□B/Q6□B and QA1S6□B, connect all Q5□B/Q6□B modules closer to the main base unit, then connect QA1S6□B modules.

(2) Setting order of the stage numbers for extension base units

Extension base units require the setting of the extension stage numbers (1 to 7) using the stage No. setting connector.

Assign the extension stage numbers starting from 1 to 7 to the extension base units in the connected order starting from the one connected to the main base unit.

(3) Cautions for assigning extension stage numbers to extension base units

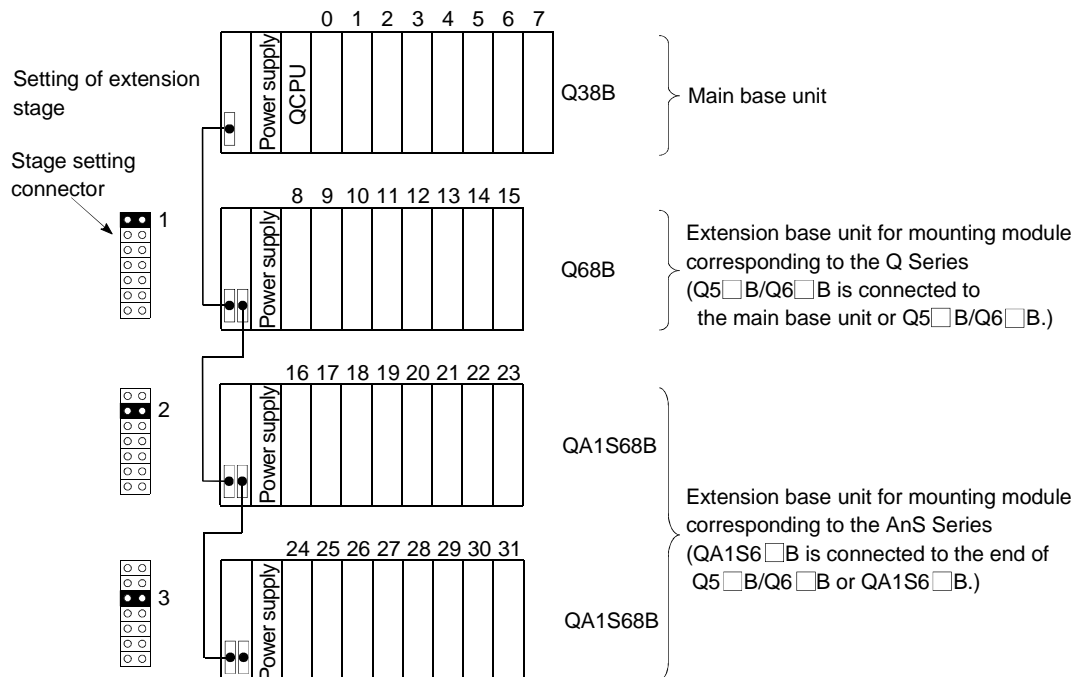
(a) Assign consecutive numbers to extension stages.

If you assign stage numbers to base units in "Auto" mode and assign some stage numbers to no modules, "0" is assigned to the skipped stage as the number of slots. Consequently, the number of empty slots does not increase. The skipped stage is also assigned with "0" of I/O point.

(b) It is impossible to set and use the same extension stage number with two or more extension base units.

(c) You cannot use the system if two or more connector pins are inserted to the stage setting connector.

On the contrary, you cannot use the system if no connector pin is inserted to the stage setting connector.



5.3 Base Unit Assignment (Base Mode)

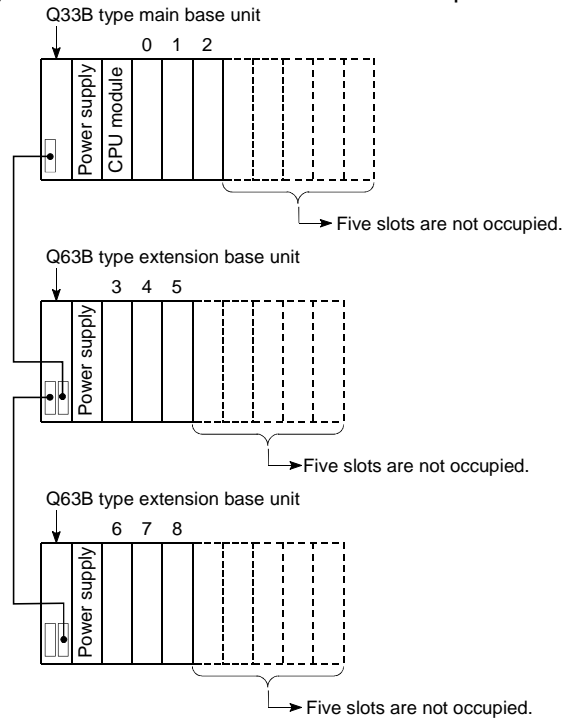
There are "Auto" and "Detail" modes to assign the number of modules can be mounted in the main and extension base units of High Performance model QCPU.

(1) Auto mode

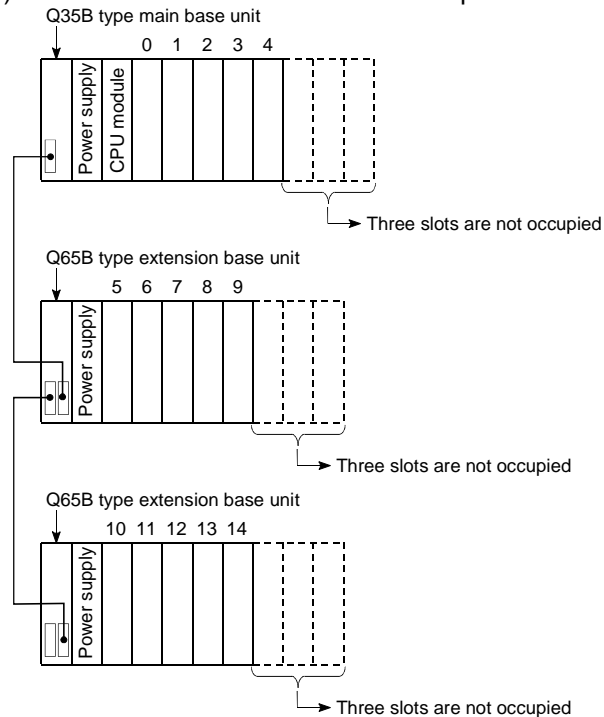
In Auto mode, the slot numbers are assigned to the main and extension base units according to the number of slots than can be occupied.

The I/O numbers are assigned according to the modules which can be mounted to the current base unit.

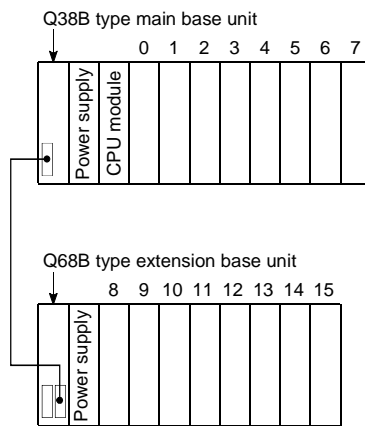
(a) For 3-slot base unit: 3 slots are occupied



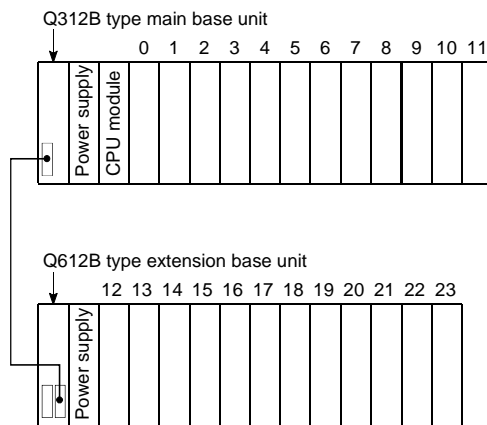
(b) For 5-slot base unit: 5 slots are occupied



(c) For 8-slot base unit: 8 slots are occupied



(d) For 12-slot base unit: 12 slots are occupied



(2) Detail mode

- (a) In Detail mode, the number of mountable modules is assigned to the individual base units (main and extension base units) at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box.

Use this mode to match the number of slots to the one for the AnS Series base units (8 fixation).

- (b) Cautions on setting the number of slots

The number of slots can be set regardless of the number of the module being used.

However, the number of slots must be set for all the base units in use.

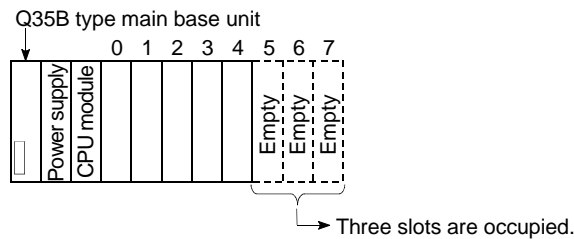
If the number of slot is not set for all the base units, I/O assignment may not work correctly.

The followings result if the preset number of slots differs from that of the installed base units.

- 1) When the designated number of slots is larger than that of the installed base unit:

Among the designated slots, those after the slots occupied by the installed base unit will be empty slots.

For example, when 8 slots are designated for a 5-slot base unit, 3 slots will be empty slots.



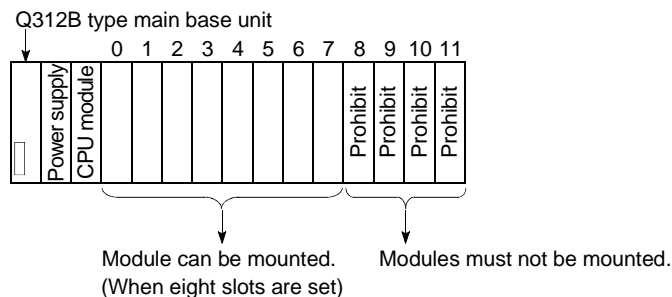
The number of points for the empty slots is the one designated at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box. (Default value is 16 points.)

- 2) When the designated number of slots is smaller than that of the base unit being used:

The slots other than those designated are disabled.

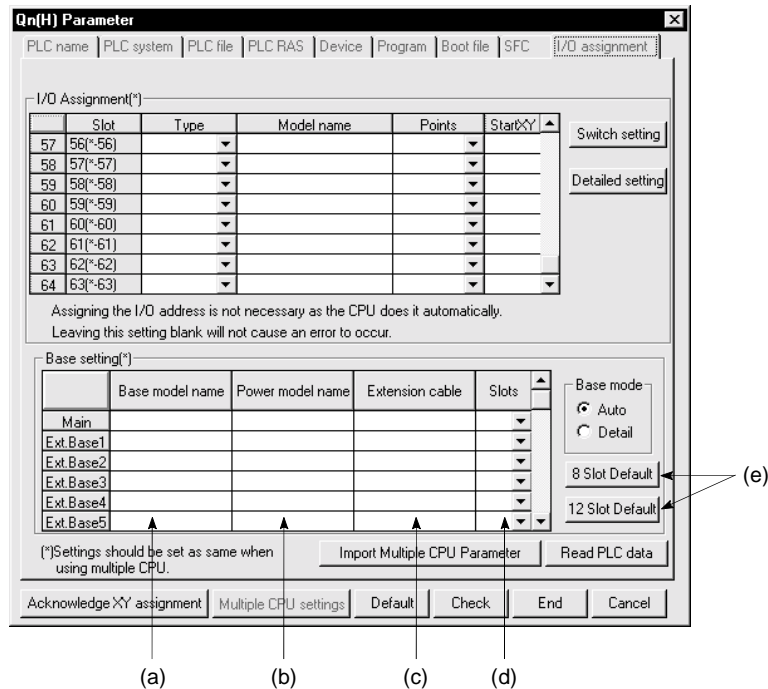
For example, when 8 slots are designated for a 12-slot base unit, the 4 slots on the right of the base unit are disabled.

(If a module is mounted to the prohibited slot, an error [SP. UNIT LAY ERR.] occurs.)





## (3) Setting screen and setting items for Base mode of GX Developer



- (a) **Base model name**  
Designate the model name of the installed base unit with 16 or less characters. High Performance model QCPU does not use the designated model name. (It is used as a user's memo or for parameter printing)
- (b) **Power model name**  
Designate the model name of the installed power supply module with 16 or less characters. High Performance model QCPU does not use the designated model name. (It is used as a user's memo or for parameter printing)
- (c) **Increase cable name**  
Designate the model name of the extension cable being used with 16 or less characters. High Performance model QCPU does not use the designated model name. (It is used as a user's memo or for parameter printing)
- (d) **Points (Used with High Performance model QCPU)**  
Select the number of points for the slot of the base unit being used from the followings:
- 2 (2 slots)
  - 3 (3 slots)
  - 5 (5 slots)
  - 8 (8 slots)
  - 10 (10 slots)
  - 12 (12 slots)
- (e) **8 fixation/12 fixation (Used with High Performance model QCPU)**  
Select either option to designate the number of slots for all base units to the same number.

5.4 What are I/O Numbers?

I/O numbers are used in sequence programs for importing ON/OFF data to High Performance model QCPU from outsides and outputting ON/OFF data from High Performance model QCPU to outsides.

Input (X) is used for the importing of ON/OFF data to High Performance model QCPU.

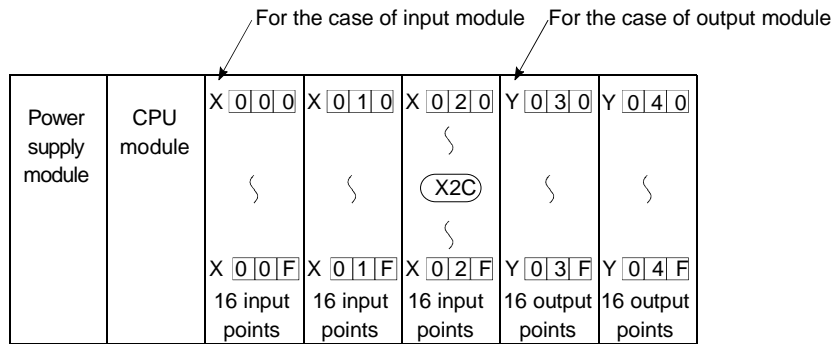
Output (Y) is used for outputting ON/OFF data from High Performance model QCPU.

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 follows.

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.



## 5.5 Concept of I/O Number Assignment

### 5.5.1 I/O numbers of main base unit and extension base unit

High Performance model QCPU assigns I/O numbers at power-on or reset according to the following items.

As a result, High Performance model QCPU can be controlled without performing I/O assignment using GX Developer.

To assign I/O numbers, follow the items below:

#### (1) Number of slots of base units

The numbers of slots of the main and extension base units are set according to the Base mode setting. (For Base mode, see Section 5.3.)

(a) In Auto mode, the number of slots is determined as the available number of modules mounted to each base unit.

For example, 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 is determined as the one designated at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box.

#### (2) Order of I/O number assignment

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 High Performance model QCPU in the main base unit.

#### (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, in the order in which the setting connectors of the extension base unit are set.

#### (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 (special function modules).

When 32-point input module is mounted on the right of High Performance model QCPU, X0 to X1F are assigned as I/O numbers.

#### (5) I/O numbers of empty slots

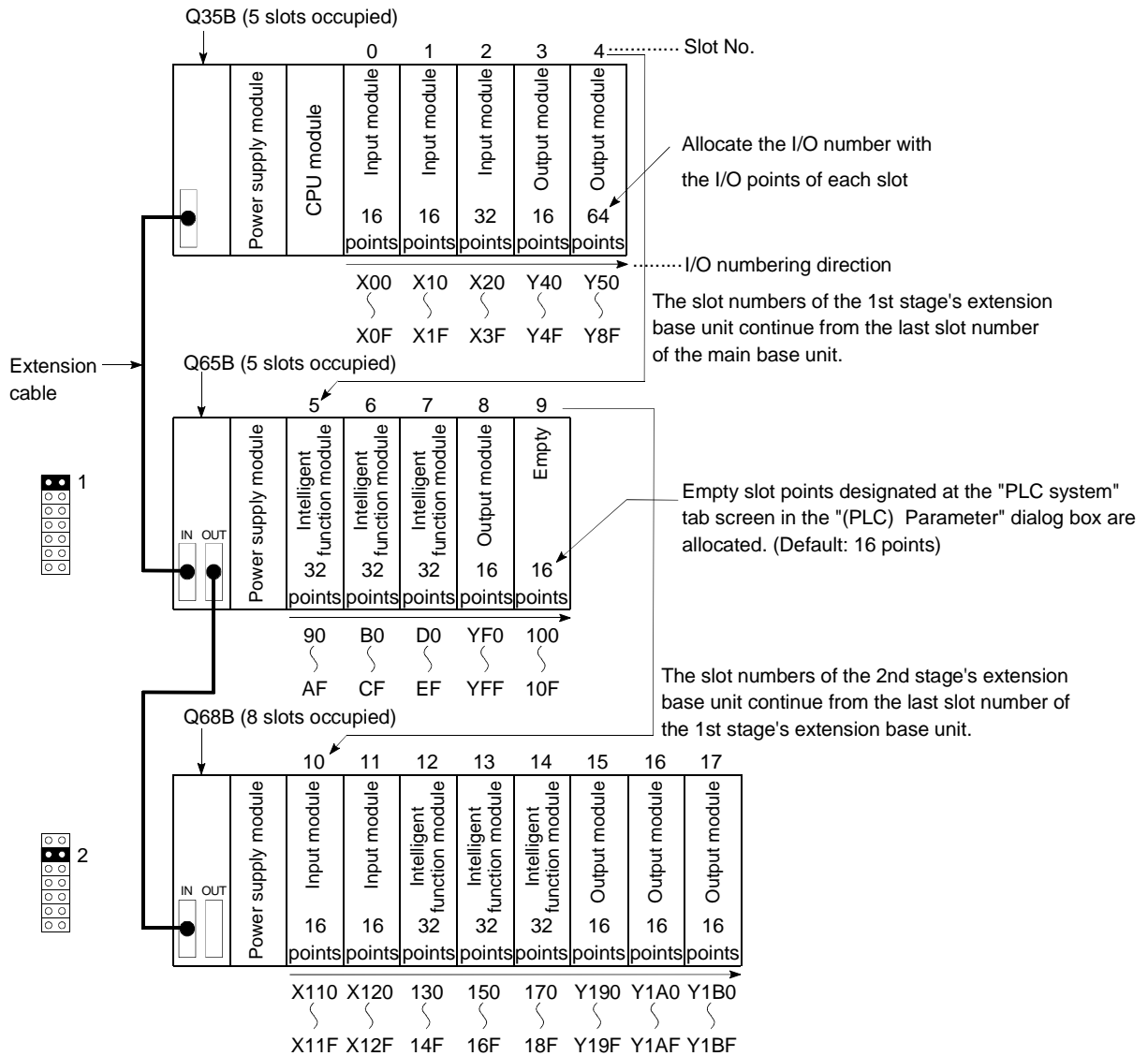
If the base unit has empty slots mounted with no I/O modules or no intelligent function modules (special function modules) are mounted, the points designated at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box are assigned to the empty slots. (Default value is 16 points.)

#### 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. (Lower I/O numbers are assigned first.)

To reserve empty extension stages for future expansion, use the PLC parameter to set the base unit.

The following shows the example of the I/O number assignment when the base unit is set in Auto mode without I/O assignment:

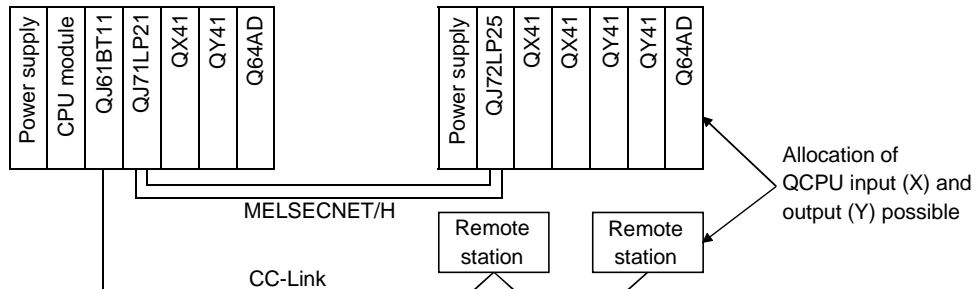


**POINT**

The above example shows the case where the intelligent function module has 32 I/O points. The number of occupied I/O points may vary depending on the intelligent function module. Refer to the manual of the intelligent function module being used and check the number of the I/O points before assigning the I/O numbers.

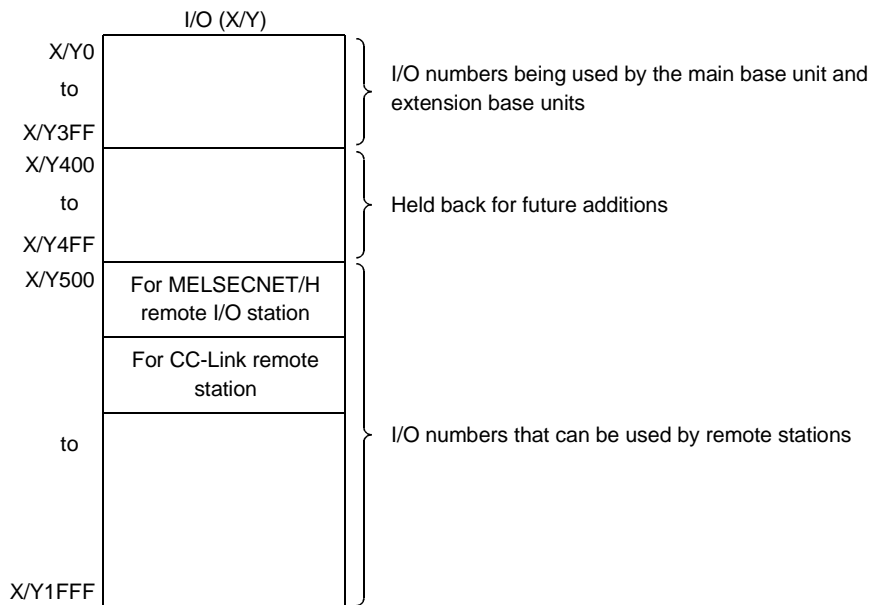
5.5.2 Remote station I/O number

It is possible to allocate High Performance model QCPU 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, the CC-Link and other remote I/O systems.



When using High Performance model QCPU device input (X) and output (Y) in remote stations, I/O numbers that succeed the numbers used by the main base unit and extension base units' I/O modules and intelligent function modules will be allocated. For example, if X/Y0 to X/Y3FF are being used by the main base unit and extension base units' I/O modules and intelligent function modules, then numbers above X/Y400 can be used by the remote station. However, the I/O numbers for remote stations should be set in consideration of additions to the main base unit and extension base units' I/O modules and intelligent function modules.

For example, if 1024 points from X/Y0 to X/Y3FF are being used by the main base unit and extension base units, and 256 points from X/Y400 to X/Y4FF are to be held back for use with future additions, then the situation shown in the diagram below is to be observed.



**POINT**  
 If network parameter setting is not made in the CC-Link system, 2048 points in the range from X/Y1000 to X/Y17FF are assigned to the master local module of the CC-Link having the lowest number.

**REMARK**

There is restriction on the order of allocating I/O numbers for MELSECNET/H remote I/O networks, CC-Link or other networks.

## 5.6 I/O Assignment by GX Developer

This section describes the I/O assignment using GX Developer.

### 5.6.1 Purpose of I/O assignment by GX Developer

I/O assignment by GX Developer is used under the following circumstances.

- (1) **Reserving points when converting to module other than 16-point modules**  
You can reserve the number of points in advance so that you do not have to change the I/O numbers when the current module will be changed to one with a different number of occupied I/O points in the future.  
For example, you can assign a 32-point I/O module to the slot where a 16-point I/O module is mounted at present.
- (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 of input modules and interrupt modules (I/O response time)**  
To match the input response time of the input modules and interrupt modules to the system, select "Type" at the "I/O assignment" tab screen in advance. (For details, see Section 7.7.)
- (5) **Error-time output mode setting**  
The output mode at error can be set to the output module, I/O module or intelligent function module. Make this setting after selecting "Type" in I/O assignment. (For details, see Section 7.8.)
- (6) **Setting the switch of intelligent function modules**  
To set the switch of the intelligent function module, select "Type" at the "I/O assignment" tab screen in advance. (For details, see Section 7.10.)
- (7) **Setting outputs during High Performance model QCPU error**  
To set the output status (retain/clear) of the output modules and intelligent function modules when the High Performance model QCPU stops the operation due to a stop error, select "Type" at the "I/O assignment" tab screen in advance. (For details, see Section 7.8.)
- (8) **Setting High Performance model QCPU operation during a hardware error of intelligent function modules**  
To set the High Performance model QCPU operation (continue/stop) during a hardware error of an intelligent function module, select "Type" at the "I/O assignment" tab screen in advance. (For details, see Section 7.9.)

POINT
<p>(1) The I/O assignment is necessary for changing the response time of the input modules and the switch of intelligent function modules. For the I/O assignment, input module response time setting, intelligent function module switch setting, and error-time output mode setting, the PLC must be powered off then on again (ON to OFF to ON) or the High Performance model QCPU must be reset.</p> <p>(2) If any of the I/O modules other than the 16-point modules fails without I/O assignment 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 using GX Developer.</p>

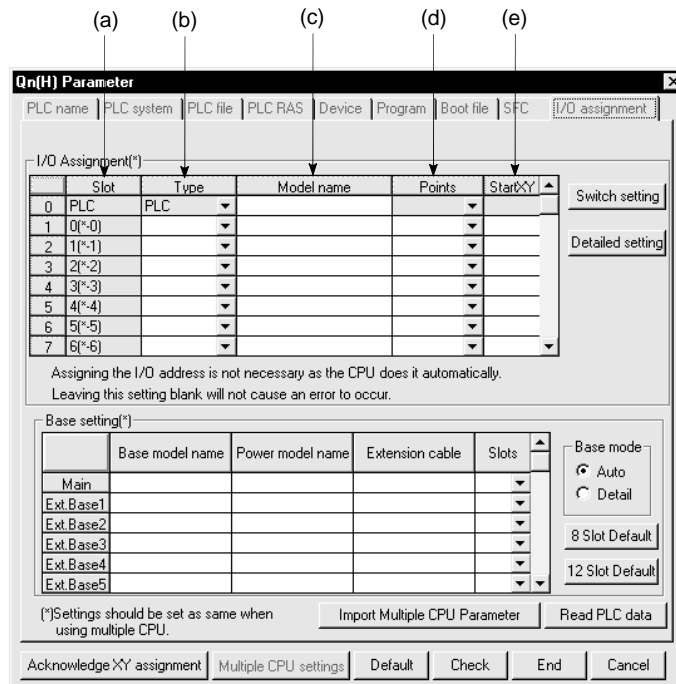
## 5.6.2 Concept of I/O assignment using GX Developer

## (1) I/O assignment for each slot

"Type" (module type), "Points" (number of occupied I/O points), and "Start" (head I/O number) can be designated individually for each slot of the base unit.

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. The I/O assignment is conducted at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box.



## (a) Slot

Displays the slot No. and the ordinal position of the slot in the base unit. If the base unit is not designated in Detail mode, the stage number of the base unit is shown as " \* ", and the ordinal number of a slot is counted from slot 0 of the main base unit.

## (b) Type (For High Performance model QCPU)

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 corresponding special function module)
- Interrupt (Q Series interruption module)\*2

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

**REMARK**

\*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

Designate the model name of the mounted module with 16 or less characters. High Performance model QCPU does not use the designated model name. (It is used as a user's memo)

## (d) Points (Used with High Performance model QCPU)

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)
- 1024 (1024 points)

If the number of occupied I/O points is not designated for a slot, the one of the actually mounted module is used.

## (e) Start XY (Used with High Performance model QCPU)

1) 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.

2) Avoid the I/O number designation of each slot from overlapping the I/O numbers assigned by High Performance model QCPU.

An error (SP. UNIT LAY ERR.) occurs when the I/O numbers overlap.

## (2) Slot status after I/O assignment

When the I/O number is assigned to a slot, the assigned I/O number takes priority regardless of the actual installation of a module.

(a) If the designated number of occupied I/O points is lower than that of the actually mounted I/O module, some I/O points of the mounted module are not used.

For example, if a slot where a 32-point input module is mounted is designated for a 16-point input module, the latter 16 points of the 32-point input module are disabled.

(b) If the designated number of occupied points is less than that of the actually mounted intelligent function module, "SP. UNIT LAY ERR." will occur.

(c) If the designated number of occupied I/O points is higher than that of the actually mounted I/O module, the points exceeding the points of the actually mounted module are set as dummies.

- (d) Be sure to set the same module type for the mounted module and the I/O assignment.

If the module type of the I/O assignment is different from that of the actually mounted module, the module may not work normally.

For the intelligent function module, make sure that the numbers of I/O points are the same.

Actually installed module	I/O assignment	Result
Input module	Output/Empty	Empty
Output module	Input/Empty	Empty
Input module/output module	Intelligent	Error (SP. UNIT LAY ERR.)
Intelligent function module	Empty	Empty
	Input/output	Error (SP. UNIT LAY ERR.)
Empty slot	Intelligent	No error occurs.

- (e) Be sure to assign the I/O numbers so that the last I/O number is within the range of FFF<sub>H</sub> or less. An error (SP. UNIT LAY ERR.) occurs when the last I/O number exceeds FFF<sub>H</sub>. (System monitor of GX Developer shows "\*\*\*" as an I/O address.)

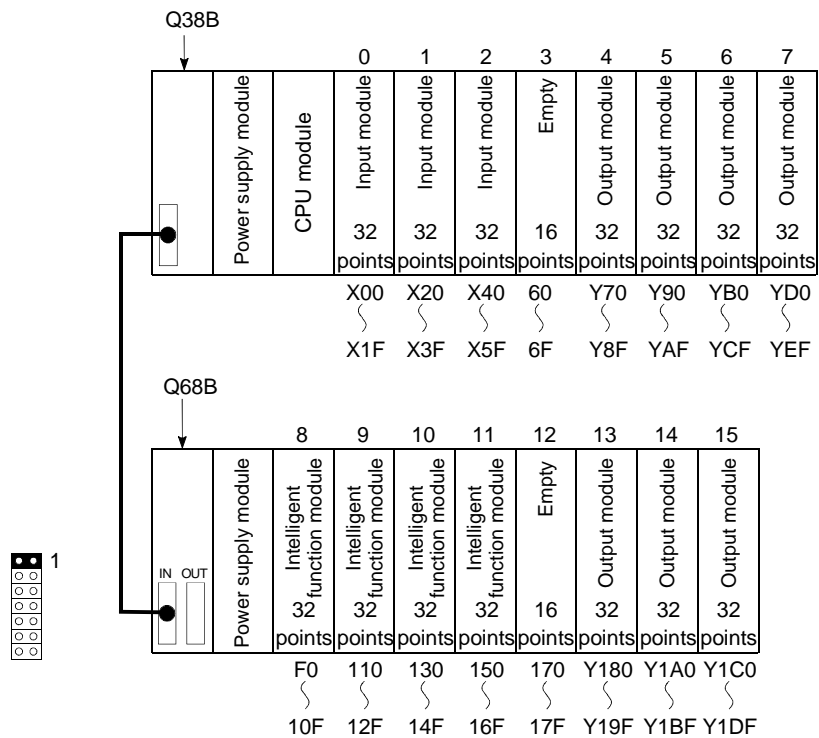
5.7 Examples of I/O Number Assignment

This section shows the examples of the I/O number assignment using GX Developer.

- (1) When changing the number of points of an empty slot from 16 to 32 points:

Reserve 32 points to the empty slot (slot No. 3) so that the I/O numbers do not change when a 32-point input module is mounted in the future. (The empty slot for slot No. 12 is not changed from 16 points.) \*1

- (a) System configuration and I/O number assignment before the I/O assignment with GX Developer



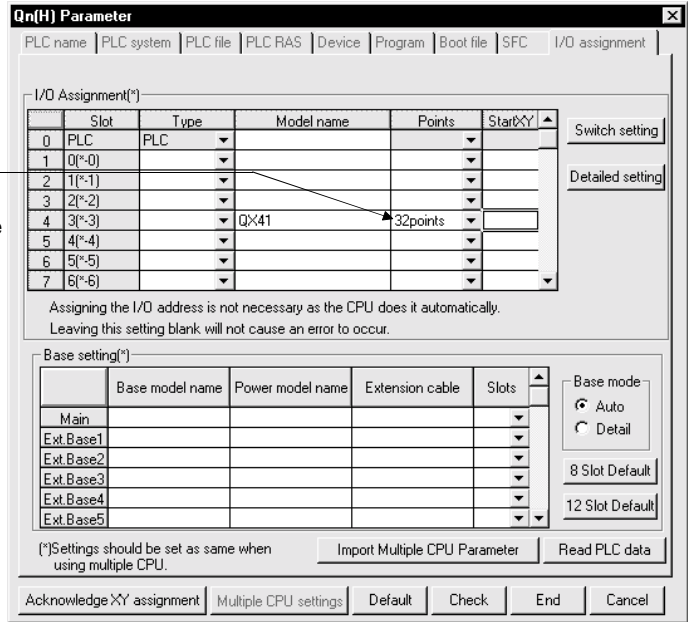
**REMARK**

\*1: This is the case where the number of points for an empty slot is set to 16 at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

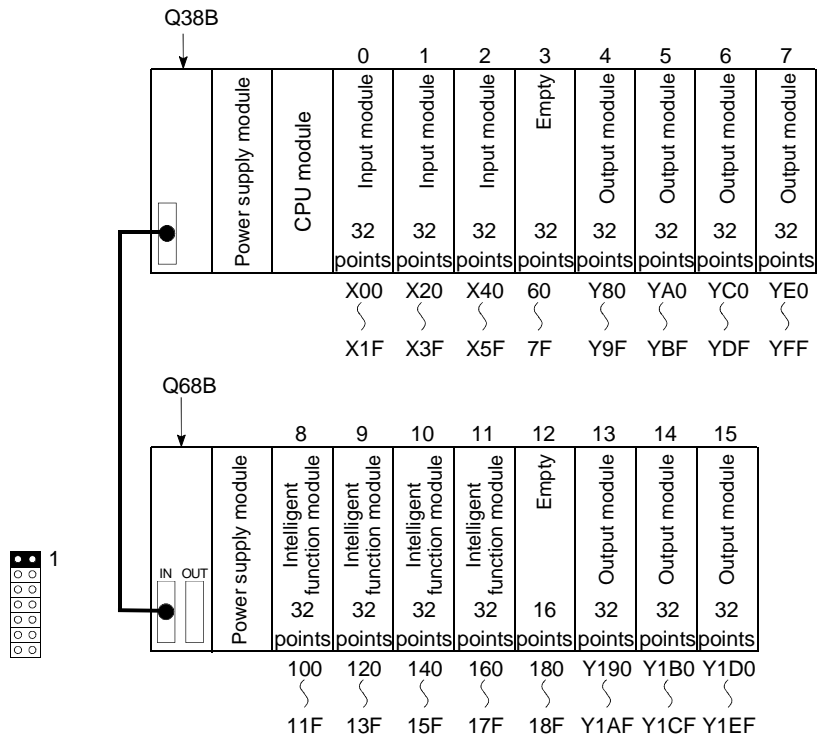
(b) I/O assignment with GX Developer

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.)



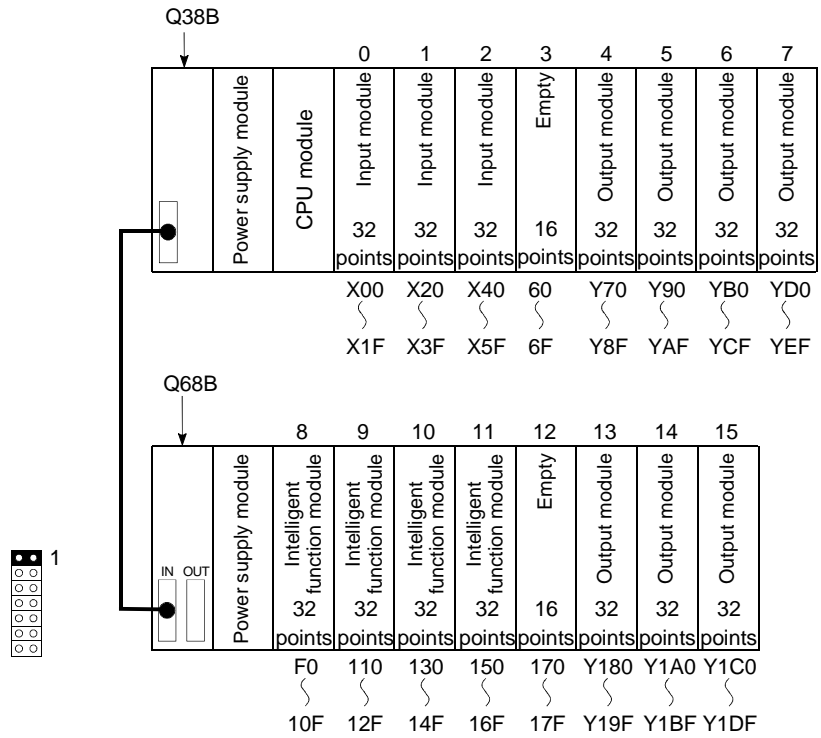
(c) I/O number assignment after the I/O assignment with GX Developer



(2) Changing the I/O number of slots

Change the I/O number of an empty slot (slot No. 3) to X200 through X21F 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).

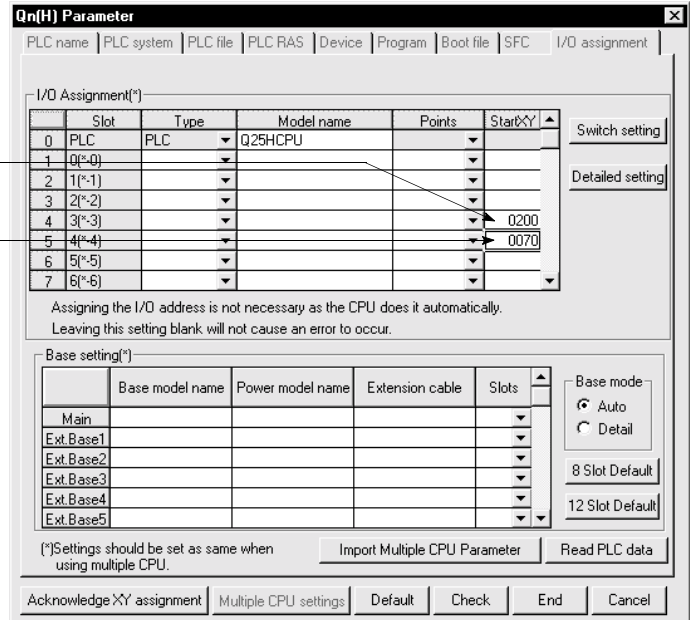
(a) System configuration and I/O number assignment before the I/O assignment with GX Developer



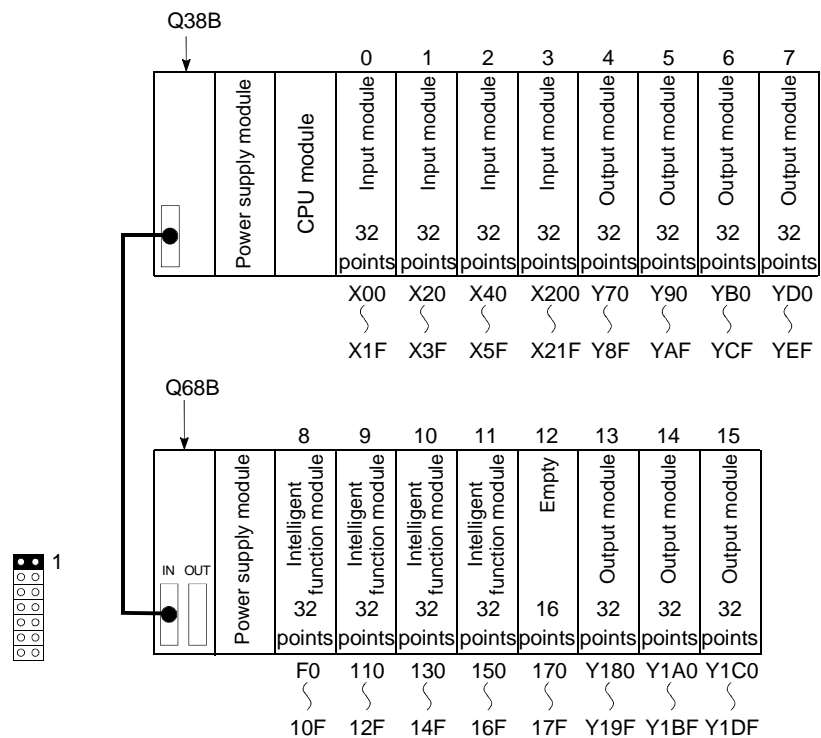
(b) I/O assignment with GX Developer

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.

"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.)



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



5.8 Checking the I/O Numbers

System monitor of GX Developer allows the check of the mounted modules of High Performance model QCPU and their I/O numbers. (For system monitor, refer to Section 7.20.)

## 6 MEMORIES AND FILES HANDLED BY HIGH PERFORMANCE MODEL QCPU

### (1) Files handled by High Performance model QCPU

(a) The High Performance model QCPU parameters, programs, comment data, etc. are assigned "file names" and "extension", and are then stored in the following memories:

- Program memory
- Standard ROM
- Memory card

When reading and writing this data from GX Developer to the High Performance model QCPU, files can be specified by type (parameter, program, comment, etc.) without regard to their extension. (GX Developer automatically assigns the appropriate extension for the file type which has been specified.)

(b) It is impossible to set and use the same extension stage number with two or more extension base units.

### (2) High Performance model QCPU file management

The use of different file and extension names permits multiple files to be stored in the High Performance model QCPU.

Because the High Performance model QCPU can also process a given program as one file, programs created can be managed individually according to their "designer", "process", or "function" by using different program file names.

Moreover, program execution is allowed for multiple programs stored in the High Performance model QCPU.

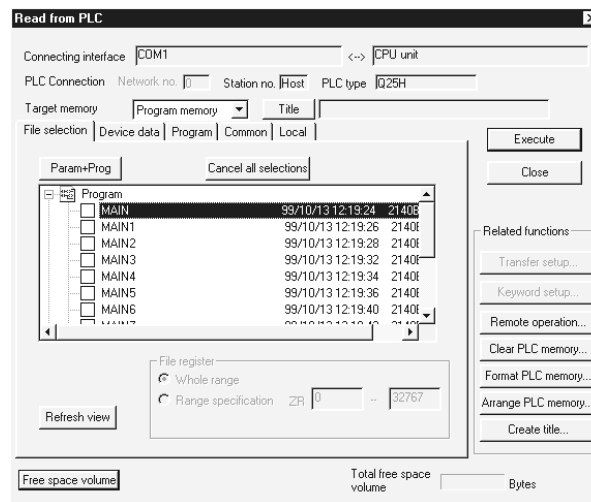
(See Chapter 4 for details on High Performance model QCPU program execution details.)

### (3) File written from GX Developer

The High Performance model QCPU stores files written from GX Developer in the memory (program memory/standard ROM/memory card).

## (4) File details

File name, file size and writing date, which are set when created with GX Developer, are added to every file written into High Performance model QCPU. When monitoring the files by GX Developer, the files are displayed as shown below.



## (a) File name

- 1) The file name consists of the file name (max. 8 characters) and the extension (3 characters).

A file name of any file written from the GX Developer into the High Performance model QCPU will be displaced in uppercase characters on screen. When naming a file by using a sequence program, type a file name in uppercase characters.

An extension which corresponds to the file type designated when the file was written in the High Performance model QCPU by GX Developer is automatically appended to the file name.

- 2) The following Windows-reserved words cannot be used for a file name.
  - COM1 to COM9
  - LPT1 to LPT9
  - AUX
  - CON
  - PRN
  - NUL
  - CLOCK\$

## (b) Date &amp; time

The date & time when the file was written from GX Developer to the High Performance model QCPU is indicated.

The set date and time indicate the GX Developer-side date and time.

## (c) Size

The file size when written from GX Developer to the High Performance model QCPU is indicated in byte units. (To view the latest High Performance model QCPU data, click on the Update button.)

Files are stored in the High Performance model QCPU program file and standard ROM in 4-byte units (1 step), and in the memory card in 1-byte units. When calculating a file's size, at least 64 bytes (136 bytes for programs) will be added to all user created files other than file registers.



## 6.1 About the High Performance model QCPU's Memory

### (1) User Memory

A user memory can be created within the memory of the High Performance model QCPU by using the GX Developer/sequence program.

The High Performance model QCPU has the following built-in memories:

- Program memory
- Standard RAM
- Standard ROM

A memory card can be mounted on the High Performance model QCPU to increase the size of a user memory.

- (a) Programs used for arithmetic operations of the High Performance model QCPU are stored in the program memory. Programs stored in the standard ROM or on a memory card are booted (read) into the program memory for arithmetic operation.
- (b) Parameters and programs are stored in the standard ROM. These data are used for ROM operation of the High Performance model QCPU.
- (c) File register and local device data is stored in the standard RAM. The use of file registers in the standard RAM will enable high speed access as is the case with data registers.

#### (d) Memory card

A memory card can be connected to a memory card interface of the High Performance model QCPU. This allows the read/write of data. The High Performance model QCPU supports three types of memory cards: SRAM card, Flash card, and ATA card.

- 1) The SRAM card allows the write/read of programs through a sequence program in the following cases where:
  - File registers are used in excess of the standard RAM capacity.
  - Sampling trace data is stored.
  - Failure history data is stored.The use of file registers allows the write/read of data at 1017k points in a sequence program.
- 2) The Flash card allows only the read through a sequence program. The Flash card is useful when data written by the High Performance model QCPU is read through a sequence program but no change is made to the data.  
The use of file registers allows a sequence program to read a maximum of 1018 k points of data written by the High Performance model QCPU.
- 3) The ATA card is used for PLC user data (general-purpose data). Access to PLC user data stored on the ATA card can be made in CSV format/binary format by using a file access instruction (e.g. FWRITE) in a sequence program.

(2) Types of Data Stored in the High Performance model QCPU Memory or on the Memory Card

The table below shows the type of data stored in a standard RAM/standard ROM or on a memory card.

Data Name	High Performance model QCPU Built-In			Memory Card (RAM)	Memory Card (ROM)		Remarks
	Program Memory	Standard RAM	Standard ROM	SRAM Card	Flash Card	ATA Card	
Parameter	○	×	○	○	○	○	1 data/drive
Intelligent function module parameter	○	×	○	○	○	○	1 data/drive
Program	⊙	×	○ *1	○ *1	○ *1	○ *1	
Device comment	○ *2	×	○	○	○	○ *3	
Device initial value	○	×	○	○	○	○	
File register	×	○ *5	×	○	○ *4	×	
Local device	×	○	×	○	×	×	1 data/CPU module
Debug data	×	×	×	○	×	×	
Failure history data	×	×	×	○	×	×	
PLC user data	×	×	×	×	×	○ *6	

⊙ : Needed, ○ : Stored, × : Not stored

**REMARK**

- \*1: Boot the program memory to execute a program.
- \*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: A standard RAM hold a single file.
- \*6: Data can be written or read with the following instructions:
  - S.FREAD (allows the batch read from a specified file on a memory card)
  - S.FWRITE (allows the batch write to a specified file on a memory card)

The table below shows file names and extensions of data files stored in the High Performance model QCPU or on a memory card.

Data name	File name
Parameter	PARAM.QPA
Intelligent function module parameter	IPARAM.QPA
Program	***.QPG
Device comment	***.QCD
Device initial value	***.QDI
File register	***.QDR
Local device	***.QDL
Debug data	***.QTD
Failure history data	***.QFD
PLC user data	***.***

The \* portions can be named by the user.

**(3) Drive Number**

- (a) The High Performance model QCPU uses drive numbers to control standard RAMs, standard ROMs, and memory cards. GX Developer specifies a selected memory (standard RAM, standard ROM or memory card) to read/write parameters and program files from to the High Performance model QCPU. There is no need to specify the drive number when using the GX Developer.
- (b) The table below shows the drive numbers used to specify a selected memory (program memory, standard RAM, standard ROM or memory card) when using a sequence program. The drive number must be used to specify a selected memory when the read/write is made through access from a serial communication module.

Memory		Drive Number
High Performance model QCPU built-in	Program memory	0
	Standard RAM	3
	Standard ROM	4
Memory card (RAM)	SRAM card	1
Memory card (ROM)	Flash card	2
	ATA card	2

**(4) Memory Capacity and Formatting**

The following table shows the size of a memory of the High Performance model QCPU and whether to format a memory.

		Q02CPU	Q02HCPU	Q06HCPU	Q12HCPU	Q25HCPU	Whether to Format
Standard RAM		64 kbyte	128 kbyte		256 kbyte		*1
Program memory		28 k steps (112 kbyte)	28 k steps (112 kbyte)	60 k steps (240 kbyte)	124 k steps (496 kbyte)	252 k steps (1008 kbyte)	*1
Standard ROM		112 kbyte	112 kbyte	240 kbyte	496 kbyte	1008 kbyte	*2
Memory card	SRAM card	Q2MEM-1MBS: 1 Mbyte Q2MEM-2MBS: 2 Mbyte					Required. (Use the GX Developer or a personal computer)
	Flash card	Q2MEM-2MBF: 2 Mbyte Q2MEM-4MBF: 4 Mbyte					Not required.
	ATA card	Q2MEM-8MBA: 8 Mbyte Q2MEM-16MBA: 16 Mbyte Q2MEM-32MBA: 32 Mbyte					Required. (Use the GX Developer)

\*1: If the memory is in the initial status or it is unstable due to low voltage of the battery (Q6BAT, Q7BAT), formatting automatically starts upon power-on or resetting of the PLC. However, format with GX Developer before starting operation.

\*2: The standard ROM is used in ROM formation of the program memory and therefore formatting is unnecessary for it.

## 6.2 Program Memory

### (1) What is the Program Memory?

- (a) The High Performance model QCPU's program memory is an internal RAM that stores programs executed by the High Performance model QCPU.
- (b) The data storage in the program memory is backed up by High Performance model QCPU's built-in batteries (Q6BAT, Q7BAT).
- (c) Before using the High Performance model QCPU for the first time, the program memory must be formatted by GX Developer.  
For details on the formatting procedure by GX Developer, refer to GX Developer manuals.

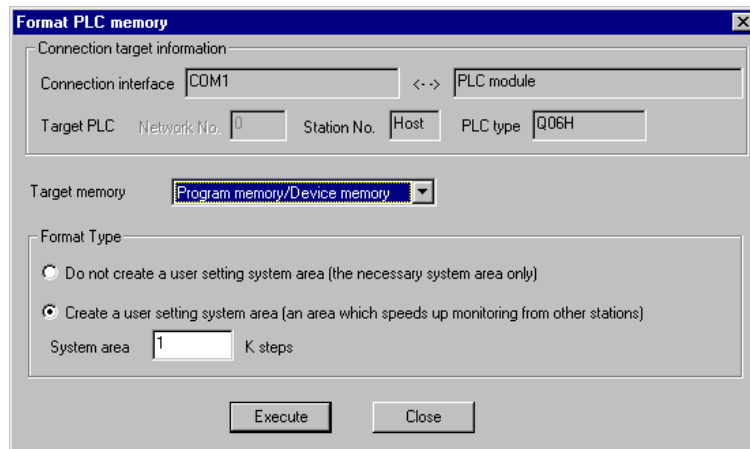
### (2) Data Storage

Data on parameters and programs can be stored in the program memory. For the types of data stored in the program memory, see Section 6.1.

### (3) Format

#### (a) Formatting

Choose "Online" → "Format PLC memory" to open the Format PLC memory dialog box. Select "Program memory/Device memory" from the Target Memory list box.



## (b) Memory capacity after formatting

The memory capacity of the program memory after formatting (when the user setting area of the system area is not set) is indicated in Table 6.1.

When the user setting area of the system area is set, the value in Table 6.1 decreases by the set capacity.

Table 6.1 Memory capacity after formatting \*1

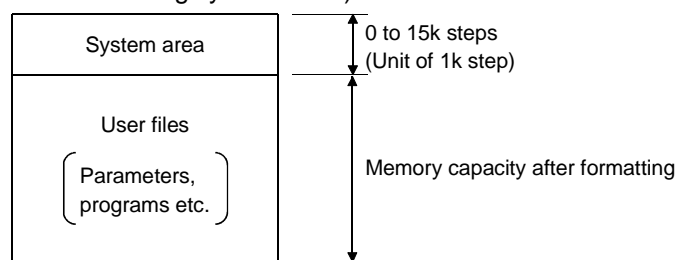
CPU model Name	Memory * <sup>2</sup>	Max. Number of Files Stored
Q02CPU	28 k steps (114688 bytes) to 13 k steps (53248 bytes)	28 files
Q02HCPU	28 k steps (114688 bytes) to 13 k steps (53248 bytes)	28 files
Q06HCPU	60 k steps (245760 bytes) to 45 k steps (184320 bytes)	60 files
Q12HCPU	124 k steps (507904 bytes) to 109 k steps (446464 bytes)	124 files
Q25HCPU	252 k steps (1032192 bytes) to 237 k steps (970752 bytes)	252 files * <sup>3</sup>

## (c) Precautions for formatting

## 1) Formatting of program memory

The High Performance model QCPU program memory can only be used after being formatted by GX Developer.

When formatting the program memory, designate whether or not a system area is to be allocated for user settings and multiple-block online change. (0 to 15 k steps (in 1 k-step units) can be allocated for the user setting system area.)



## 2) System area setting

If RS-232 and USB interfaces are connected to GX Developer, the system area user setting data is used for registering monitor data from the GX Developer connected to serial communication module.

The allocation of space for system and user-defined areas will make it much easier to perform monitoring operation by operating from the GX Developer connected to the serial communications module.

Although the designation of a user setting area speeds up monitoring from the GX Developer connected to serial communication module, it also reduces the amount of space available for user files.

**REMARK**

1)\*1: This table shows an example in which 0 k step is allocated for a system area.

2)\*2: In computing the memory capacity, 1 step is equal to 4 bytes.

3)\*3: Maximum number of executable program is 124. More than 124 programs can not be executed.

### 6.3 About the Standard ROM

#### (1) What is the standard ROM?

- (a) The standard ROM is used for the ROM operation of the High Performance model QCPU.
- (b) Programs stored in the standard ROM can be used after being read (boosted) to the program memory in accordance with the setting made at the "Boot file" tab screen in the "(PLC) Parameter" dialog box.
- (c) The standard ROM does not need formatting.
- (d) Writing into standard ROMs is performed with the GX Developer's on-line "PLC Writing" (flash ROM) of "Create Program Memory ROM" (see Section 6.6.1.) It is also possible to write in a standard ROM from the memory card with "Automatic writing in the standard ROM" without using GX Developer.

#### POINTS

- (1) Before writing data to a standard ROM, all previous data stored in the standard ROM are erased. Therefore, all data stored in a standard ROM must be read out and copied into the program memory at first. There, read through and modify it as necessary. Then, write the modified data back into a standard ROM at a time. Please note that an error may occur if data stored in the standard ROM is used in a sequence program, with data being written in the standard ROM.
- (2) For details on the formatting procedure by GX Developer, refer to GX Developer manuals.

#### (2) Data Storage

A standard ROM stores data such as parameters and programs.  
See Section 6.1 for the data to store in the standard ROM.

#### (3) Memory Capacity

Table 6.2 shows the memory capacity of standard ROMs.

Table 6.2 Memory Capacity

Model Name	Memory Capacity *2	Max. Number of Files Stored
Q02CPU	28 k steps (114688 bytes)	28 files
Q02HCPU	28 k steps (114688 bytes)	28 files
Q06HCPU	60 k steps (245760 bytes)	60 files
Q12HCPU	124 k steps (507904 bytes)	124 files
Q25HCPU	252 k steps (1032192 bytes)	252 files

#### REMARK

In computing a memory capacity, 1 step is equal to 4 bytes.

## 6.4 About the Standard RAM

### (1) What is the standard RAM?

- (a) The standard RAM is used when using file registers or local devices without a memory card being mounted on the High Performance model QCPU.
- (b) The standard ROM must be formatted by using GX Developer when using the High Performance module QCPU for the first time. Refer to the GX Developer manual for details on the formatting method.
- (c) Data can be written into the standard RAM by using the online function: "Write to PLC."

### (2) Stored Data

A standard RAM holds two files: file register file and local device file. Any other files cannot be written into a standard RAM.

(The standard RAM cannot store other files than the file register and local device files.)

### (3) Formatting

#### (a) Formatting

To format a standard RAM, choose "Online" → "Format PLC memory" and then select "Standard RAM" in the "Target memory" list box. See Section 6.2 for the PLC Memory Format dialog box.

#### (b) Memory capacity after formatted

Table 6.3 shows the memory capacity of standard RAM.

Table 6.3 Memory Capacity

CPU Type	Serial No.	Number of Files Stored	Number of Files Stored	
			File Register	Local Device
Q02CPU	—	32 k words (64 kbytes)	1	1
Q02HCPU	"04011" or earlier	32 k words (64 kbytes)	1	1
	"04012" or later	64 k words (128 kbytes)		
Q06HCPU	"04011" or earlier	32 k words (64 kbytes)	1	1
	"04012" or later	64 k words (128 kbytes)		
Q12HCPU	"02091" or earlier	32 k words (64 kbytes)	1	1
	"02092" or later	128 k words (256 kbytes)		
Q25HCPU	"02091" or earlier	32 k words (64 kbytes)	1	1
	"02092" or later	128 k words (256 kbytes)		

## (4) Precautions

When setting file registers and local devices in the standard RAM, memory capacity is secured in 1024 byte units for the Q12HCPU/Q25HCPU that have the serial number whose upper five digits are "02092" or later.

Memory capacity is secured in 512 byte units for the Q12HCPU/Q25HCPU, Q02CPU, Q02HCPU and Q06HCPU that have the serial number whose upper five digits are "02091" or earlier.

**POINT**

When specifying file registers using the serial access method (ZR□) with commands that access the standard RAM on "02092" and subsequent High Performance model QCPUs, the amount of time required for processing each command from "02091" and previous High Performance model QCPUs will be prolonged. (QnCPU: Average 0.65μs, QnHCPU: Average 1.1μs)

The processing time when the MOV command is used is shown below.

Command	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



## 6.5 Memory Card

### (1) Memory card

- (a) A memory card is used to expand the size of an internal memory of the High Performance model QCPU.
- (b) There are three types of memory cards for use in the High Performance model QCPU: SRAM card, Flash card, and ATA card.

POINTS
<p>(1) Before the memory card can be used for the first time, the memory card must be formatted by GX Developer. For details on the formatting procedure by GX Developer, refer to the GX Developer manuals.</p> <p>(2) Before writing data into a Flash card, all previous data stored on the Flash card are erased. For this reason, to write data into the Flash card, you must first read and copy all previous data stored in the Flash card before writing necessary data. Please note that an error may occur if data stored in the Flash card is used in a sequence program, with data being written on the Flash card.</p>



### (2) Stored Data

A memory card holds parameter and program data. See Section 6.1 for the types of data stored in a memory card.

### (3) Format

The SRAM card and ATA card must have all been formatted.

Since the SRAM card and ATA card purchased are not yet formatted, use them after formatting with GX Developer.

(The Flash card need not be formatted.)

#### (a) Execution of formatting

To format the card, choose "Online" \_ "Format PLC memory" and then select "Memory card (RAM)" or "Memory card (ROM)" on GX Developer.

(Refer to Section 6.2 for the Format PLC memory screen.)

POINT
<p>Do not format ATA card using other than GX Developer. (If it is formatted using format function of Windows®, the ATA card may not be usable with set in a CPU module.)</p>

#### (b) Precautions

When the SRAM card or ATA card is formatted, the "memory card information area" is automatically secured, reducing the space by the size of the "memory card information area".

## (4) Memory capacity

Table 6.4 indicates the the memory capacities of the memory cards.

Table 6.4 Memory Capacity

Type	Memory Card Model Name	Memory Capacity	Number of Files Stored
SRAM card	Q2MEM-1MBS	1011.5 kbyte *1	256 files
	Q2MEM-2MBS	2034 kbyte *1	288 files
Flash card	Q2MEM-2MBF	2035 kbyte	288 files
	Q2MEM-4MBF	4079 kbyte	288 files
ATA card	Q2MEM-8MBA	7940 kbyte *1	512 files
	Q2MEM-16MBA	15932 kbyte *1	512 files
	Q2MEM-32MBA	31854 kbyte *1	512 files

\*1: The indicated memory capacities of the SRAM card and ATA card are those after formatting.

## 6.6 Writing Data to the Standard ROM or the Flash Card

### 6.6.1 Writing Data to the standard ROM or to the Flash card using GX Developer

The "write to PLC" function in the GX Developer Online menu does not allow the user to write files into a standard ROM or on a Flash card. For writing files to a standard ROM or to a Flash card by operating from GX Developer, GX Developer Online menu provides two functions: "Write the program memory to ROM" and "Write to PLC (Flash ROM)."

#### (1) Write the program memory to ROM

- (a) The "Write the program memory to ROM" function allows a batch of files stored in a program memory to be written to a standard ROM or a Flash card. This function is used to debug the programs stored in the program memory.
- (b) When the "Write a memory to ROM" function is executed, all files stored in the standard ROM or Flash card are erased before a batch of files stored in a program memory are written. No files can be added to the standard ROM or Flash card.
- (c) The memory capacity of a standard ROM or Flash card is the same as that of a program memory. A memory of a larger size than the memory capacity of a program memory cannot be used.
- (d) To execute the "Write the program memory to ROM" function, set the length of GX developer's time-check to 60 seconds or longer. Shorter time-check may cause a time-out on the GX Developer side. To execute the "Write the program memory to ROM" function via the CC-Link network by operating from a GX Developer at a local station, set the length of CC-Link's CPU monitoring time (SW0A) to 60 seconds or longer. The default is 90 seconds. Use the default value when making the setting.

#### (2) Write to PLC (Flash ROM)

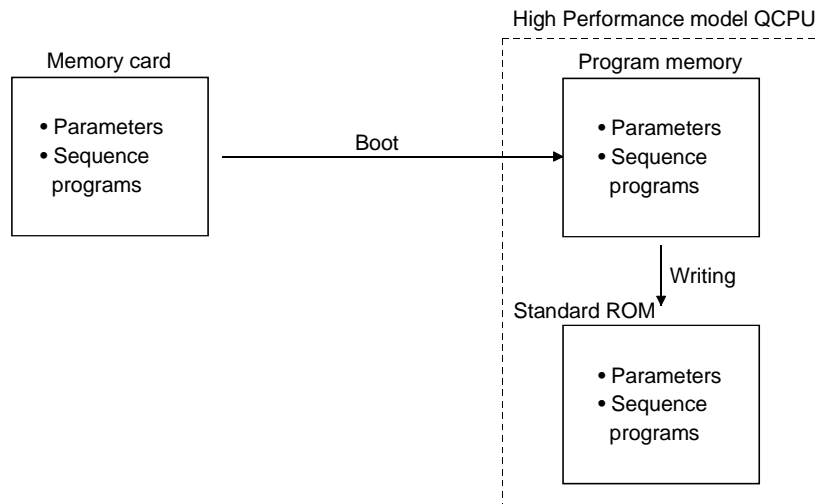
- (a) The "Write to PLC (Flash ROM)" function is used to write a batch of files specified by GX Developer to a standard ROM or Flash card.
- (b) The "Write to PLC (Flash ROM)" function can fill all available space in a standard ROM or Flash card. If a program that contains a small number of steps is written to a Flash card, it will take long to fill the Flash card with programs. When a RS-232 interface is mounted at Q2MEM-4MBF, a baud rate of 115.2k bps takes about 14 minutes. To write data to a Flash card, increase a baud rate or use a USB interface. If the "Write to PLC (Flash ROM)" function is executed from a local station, communication time will be longer.
- (c) To execute the "Write to PLC (Flash ROM)" function, set the length of GX Developer's time-check to 60 seconds or longer. Shorter time-check may cause a time-out on the GX Developer side. To execute the "Write to PLC (Flash ROM)" function via the CC-Link network by operating from GX Developer at a local station, set the length of CC-Link's CPU monitoring time (SW0A) to 60 seconds or longer. The default is 90 seconds. Use the default value when making the setting.

- (d) When the "Write to PLC (Flash ROM)" function is executed, all files stored in the standard ROM or on the Flash card are erased before a batch of files specified by GX Developer are written. No files can be added to the standard ROM or Flash card. To add new files to old files, read all the old files from High Performance model QCPU and write them again into the High Performance model QCPU.
- (e) The "Write to PLC (Flash ROM)" function can be executed when the High Performance model QCPU is in RUN status. However, for the following cases, execute the "Write to PLC (Flash ROM)" function after the High Performance model QCPU enters into STOP status.
  - 1) The file registers of the Flash card is used in a sequence program.
  - 2) The file registers are used in a sequence program by setting the file register to "set not to use" in the PLC parameter.If the "Write to PLC (Flash ROM)" function is executed when the High Performance model QCPU is in RUN status, an error may occur and the High Performance model QCPU may stop running.
- (f) While the "Write to PLC (Flash ROM)" function is executed, the read/write cannot be made from other modules. This may cause a time out on the side of other modules.

POINT
When the High Performance model QCPU is expanded to STOP status and Write to PLC (Flash ROM) is being performed, do not set it in RUN status. RUN cannot be performed normally during Write to PLC (Flash ROM). Perform RUN after Write to PLC (Flash ROM) is completed.

### 6.6.2 Automatic write to standard ROM (Auto Download all Data from Memory card to Standard ROM)

"Automatic write to standard ROM" function writes the parameters and sequence programs stored on the memory card into the High Performance model QCPU's standard ROM without using GX Developer. (The writing of parameters and sequence programs into the memory card is performed by GX Developer (Version 6 or later.) With this function, the parameters and sequence programs are booted from the memory card to the program memory, and the booted parameters and sequence programs are then written from the program memory into the standard ROM, as shown below.



"Automatic write to standard ROM" is used to change the High Performance model QCPU programs that perform ROM operations with the standard ROM. Overwriting in the standard ROM is performed by GX Developer, but using "Automatic write to standard ROM" moves the memory card in which the parameters and the changed programs are written to the High Performance model QCPU, so that they are written into the standard ROM from the memory card.

"Automatic write to standard ROM" is available with a combination of the High Performance model QCPU with the upper five digits of its serial No. as "02092" or later and GX Developer Version 6 or later.

If the memory card, in which "Automatic write to standard ROM" is set, is mounted in the High Performance model QCPU whose the upper five digits of its serial No. is the "02092," or later the boot operations are performed from the standard ROM.

The followings are necessary for "Automatic write to standard ROM".

- Set "Automatic write to standard ROM settings" in the "(PLC) Parameters dialog box".
- Memory card on which the parameters and programs are stored.
- Memory card mounted onto the High Performance model QCPU and the High Performance model QCPU switch settings.

#### POINT

Perform "Automatic write to standard ROM" after the High Performance model QCPU control is suspended.

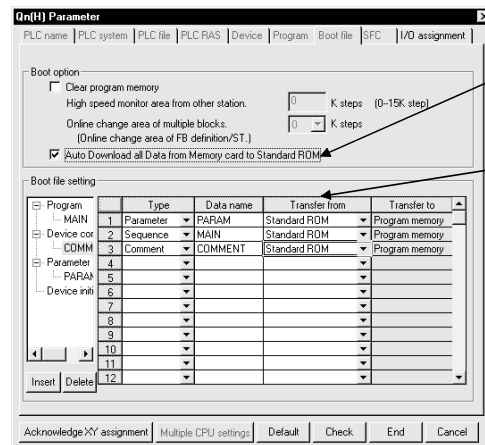
A suspension error (BOOT OK (Error Code: 9020)) occurs when automatic write to standard ROM is completed.

It is necessary to reset the High Performance model QCPU or restart, the power supply to the PLC after "Automatic write to standard ROM" is completed.

(1) Execution procedure for "Automatic write to standard ROM"  
Observe the following procedure for "Automatic write to standard ROM".

(a) Operations with the GX Developer (Settings for "Automatic write to Standard ROM")

- 1) Check "Auto Download all Data from Memory card to Standard ROM" at the "Boot file" tab screen in the "(PLC) Parameter" dialog box. Set the parameters and programs to be booted. (Set the "Transfer from" to "Standard ROM".)



Check "Auto Download all Data from Memory card to Standard ROM"

Set the "Transfer from" to the "Standard ROM"

- 2) Store the setup parameters and the programs to be booted in the memory card.

(b) Operations with High Performance model QCPU (Automatic write to standard ROM)

- 1) Switch off the power supply to the PLC.
- 2) Mount the memory card that contains the parameters and programs to be booted onto the High Performance model QCPU.
- 3) Set the parameter valid drive to the mounted memory card with the CPU's dip switches as follows:
  - When a SRAM card is mounted: SW2 : ON, SW3 : OFF
  - When a Flash/ATA card is mounted: SW2 : OFF, SW3 : ON
- 4) Switch on the power supply to the PLC.  
Boot the file specified with the parameter to the program memory from the memory card. Write the content of the program memory to the standard ROM when the boot is completed.
- 5) "BOOT" LED will flicker when automatic write to standard ROM is completed, and the High Performance model QCPU will assume a suspension error status.
- 6) Switch off the power supply to the PLC.
- 7) Remove the memory card, and then set the parameter valid drive to the standard ROM with the CPU's dip switches as follows:
  - Standard ROM: SW2 : ON, SW3 : ON

- (c) The parameters and programs will be booted from the standard ROM to the program memory to enable actual operations when the PLC is switched on.

## (2) Precautions

This section indicates the precautions for performing "Automatic write to standard ROM"

- (a) If the file to be booted from the memory card shares the same name as a file in the program memory, the memory card data will be overwritten. Also, if the file to be booted from the memory card does not share the same name as a file in the program memory, it will be added to the program memory. The "FILE SET ERROR (Error code: 2401)" will occur at this time if the capacity of the program memory is exceeded.
- (b) It is possible to select whether to perform the boot after the program memory has been cleared, or perform the boot without clearing the program memory when booting from the memory card to the program memory. Performing the boot after the program memory has been cleared when "Automatic write to standard ROM" prevents the program memory from overflowing during the boot.
- (c) The "Auto Download all Data from Memory card to Standard ROM" setting at the "Boot file" tab screen is valid only when the High Performance model QCPU parameter valid drive is to "Memory Card".  
The "Auto Download all Data from Memory card to Standard ROM" setting at the "Boot file" tab screen is disabled if the parameter valid drive is set to "Program Memory" or "Standard ROM".

## 6.7 Executing Standard ROM/Memory Card Programs (Boot Run)

## (1) Executing High Performance model QCPU programs

- (a) The High Performance model QCPU executes programs stored in the program memory.

The High Performance model QCPU does not perform operation of programs stored in the standard ROM or memory cards.

- (b) To execute programs stored in the standard ROM and memory card, designate file names to be booted (read) to program memory at the "Boot file" tab screen in the "(PLC) Parameter" dialog box.

Programs with file names designated on the "Boot file" tab screen are booted from the standard ROM/memory card to the program memory and executed in the order set on the "Boot file" tab screen when the PLC is powered ON or the High Performance model QCPU is reset.

## (2) Preparation for Boot Run

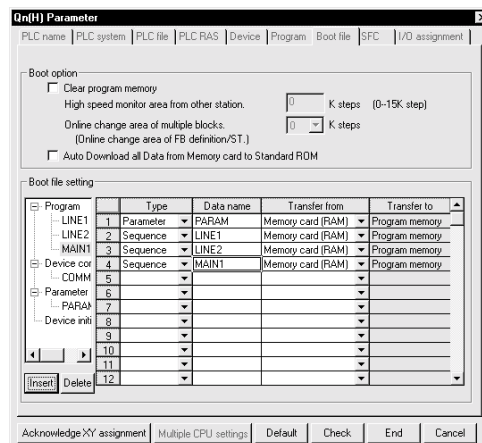
Perform the following steps in preparation for boot run:

- (a) Create a program using GX Developer.

Create a program used for the boot run.

- (b) Select a boot file using GX Developer.

Select a boot file at the "Boot file" tab screen in the "(PLC) Parameter" dialog box.



- (c) Make the High Performance model QCPU hardware setting.

Set High Performance model QCPU Dip switches to specify a parameter-driven drive.

- (d) Insert a memory card.

Insert a memory card in a slot if you want to store parameters or programs on the memory card during the boot run.

- (e) Write parameters and programs using GX Developer.

Write parameters on the parameter-driven drive. Write a program to the memory specified at the "Boot file" tab screen in the "(PLC) Parameter" dialog box.

- (f) Execute a program.

Reset the High Performance model QCPU with the RESET/L.CLR switch. After the boot run is completed in the specified memory, the "BOOT" LED lights up.



(3) Changing Program Files While the High Performance model QCPU is in the Run Status.

- (a) While the High Performance model QCPU is in RUN status, addition/change/deletion of program files from the standard ROM or memory card to the program memory can be made by using any of the following instructions in a sequence program.

- PLOAD (Loading program from memory card)
- PUNLAOD (Unloading program from program memory)
- PSWAP (Load + Unload)

For details on the PLOAD, PUNLAOD and PSWAP instructions, refer to the QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions).

- (b) Even if a program file is changed while the High Performance model QCPU is in RUN status, the settings specified at the "Program" tab screen the "(PLC) Parameter" dialog box will remain unchanged. When the High Performance model QCPU is in STOP status, the settings made at the "Program" tab screen in the "(PLC) Parameter" dialog box must be adjusted to any changes (addition, change or deletion of program names) made when the High Performance model QCPU was in RUN status. If no adjustment is made at the "Program" tab screen in the "(PLC) Parameter" dialog box, an error may occur when the High Performance model QCPU enters into RUN status from STOP status.

(4) Precautions for Executing Programs in the Standard ROM/Memory Card

- (a) For boot run, store parameters (PLC parameters) of the boot file setting in a standard RAM or memory card. If parameters are stored in a program memory and a parameter valid drive is set to "Program Memory", the boot file setting made in the "(PLC) Parameter" dialog box is ignored. As a result, a boot run is not performed when power is turned on or when the PLC is reset.
- (b) If programs are written in the program memory during RUN status while a boot run is performed from a memory card (RAM), any change made will be reflected in the programs stored on the memory card (RAM). For details on the writing of programs during RUN status, see Section 7.10.
- (c) If programs are written in the program memory during RUN status while a boot run is performed from a standard ROM/memory card (ROM), any change made will not be reflected in the programs stored in the standard ROM or the memory card (ROM).
- (d) At the "Boot file" tab screen in the "(PLC) Parameter" dialog box, set the maximum number of boot files to the number of files stored in the program memory. The number of boot files will be decreased by one in the following cases where:
- A header is specified.
  - A PLC parameter in which a boot file setting is made is booted.
- (e) If boot operation is made under the following conditions, it may take maximum 200 ms for each 1k steps (4kbyte) during boot sequence.
- To boot from an ATA card.
  - To boot from standard ROM with an ATA card mounted.

- (f) If the program memory is changed when a sequence program is written in the program memory and PLC is turned on or reset, boot operation mode may be selected.

If the "BOOT" LED is lit on the front panel of the High Performance model QCPU, the boot operation mode is selected.

Cancel the boot operation mode with the following procedure.

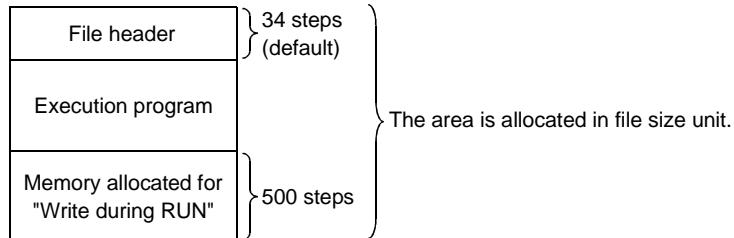
- 1) Write parameters, in which no boot file settings are made, into the program memory.
- 2) Using the DIP switch of the CPU module, set "program memory" for the valid drive setting. (DIP switch setting: SW2: OFF. SW3: OFF)
- 3) Turn off and on the PLC or reset the CPU module.  
(After the procedure, the settings given in steps 1) and 2) become valid.)

6.8 Program File Configuration

(1) Program File Configuration

- (a) Program files consist of a file header, an execution program, and allocate memory allocated for "Write during RUN".

Program file configuration



- (b) As shown below, the size of a program stored in the High Performance model QCPU includes all the above components.

- 1) File header: The file name, file size, and file creation data, etc., are stored in this area.  
The file header size changes between 25 and 35 steps (100 and 140 bytes) depending on the device setting in the PLC parameter dialog box.
- 2) Execution program: The created program is stored in this area.  
1 step is 4 bytes.
- 3) Memory allocated for "Write during RUN": This area is used when write during RUN that write during RUN increases the number of steps is executed from GX Developer.  
Default value is set to 500 steps (2000 bytes).  
The number of memory allocated for "Write during RUN" can be changed using the online write to PLC program.  
The number of memory allocated for "Write during RUN" can be redefined if the number of memory allocated is not sufficient for write during RUN. (See Section 7.12.1.)

(2) The size of the program displayed by GX Developer

During programming by GX Developer, the program size (the total of the file header size and the number of created program steps) is displayed as the number of steps as shown below.

During programming, the size of the program created is displayed.

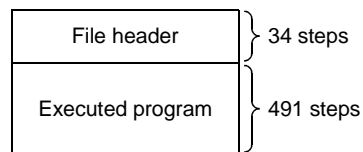
"Number of steps used" display



**PRECAUTIONS**

- 1) The program capacity displayed during programming with GX Developer is the sum of file header and executed program capacities and does not include the capacity of steps secured for write during RUN.

(Example) The capacity of the program whose executed program part has 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

**Status of File on GX Developer**

- 2) Since a file is stored on the program memory in file size unit, the program capacity displayed during programming with GX Developer may differ from the capacity of the program file on the High Performance model QCPU. Refer to Section 6.9.3 for details.

## 6.9 GX Developer File Operation and File Handling Precautions

## 6.9.1 File operation

GX Developer online operation allows the files which are stored in the program memory, standard ROM and memory card, to perform the file operations in the table below.

However the available file operations vary according to the presence or absence of a password (registered by GX Developer), the High Performance model QCPU "write protect" switch setting status, and the High Performance model QCPU RUN/STOP status.

Table 6.5 File Operations from GX Developer

File Operation	Operation Enabled/Disabled				Operation Description
	A*	B*	C	D	
Read from PLC	○	△	○	○	Files are read from concerned memory.
Write to PLC	△	△	×	○	Files are written to the program memory or SRAM card.
Verify with PLC	△	△	○	○	Verify the target memory and the GX Developer's file.
Write the program memory to ROM	○	○	×	○	Write a batch of files from the program memory to the standard ROM or Flash card.
Write to PLC (Flash ROM)	△	△	×	○	Write a batch of files from GX Developer to the standard ROM or Flash card.
Delete PLC data	△	△	×	×	A file stored in memory is deleted.
Format PLC memory	○	○	×	×	Memory formatting is executed.
Arrange PLC memory	○	○	×	×	Memory files which are no longer contiguous are re-organized to make them contiguous.
Write during RUN in the ladder mode	△	△	×	○	Write changes made in the ladder mode into the program memory.

○ : Execution enabled, △ : Execution enabled with some restrictions, × : Execution disabled

**REMARK**

- 1) The codes (A, B, C, D) used at the "operation enabled/disabled" item in the above table are explained below.

Table 6.6 Operation enabled/disabled

Code	Description
A	When "write prohibit" password is registered in a file
B	When "read/write prohibit" password is registered in a file
C	When the High Performance model QCPU's "system protect" switch is ON
D	When High Performance model QCPU RUN status is in effect

- 2)\*: Execution is allowed only when the passwords match.

### 6.9.2 File handling precautions

#### (1) Power OFF (or reset) during program operation

- (a) If power is switched OFF during a file operation which will not cause a file shift, the memory data will not be lost.
- (b) If files and data in the memory of the High Performance model QCPU is backed up using the battery (Q6BAT), the program memory data will not be lost when the power is switched OFF during the following file operations which cause a file shift.
  - File size change
  - PLC memory arrangement
  - New file creation
  - Writing a program file during the RUN status
  - Writing a program in excess of memory allocated for 'Write during RUN'
  - Reading a file with the PLOAD instruction

Files stored in the memory card will not be lost unless the memory card is removed from the High Performance model QCPU while the power is OFF.

<b>POINT</b>
--------------

If the above operations are done, the half-processed data will be stored in the High Performance model QCPU internal memory, and will be restored when power is switched ON again. A battery backup is required in order to save internal memory data for this reason.
--

#### (2) Simultaneous write to the same file from multiple GX Developers

The High Performance model QCPU does not allow access from other GX Developers to the file being written.

It does not allow write from other GX Developers to the file being accessed, either.

Therefore, to perform write from multiple GX Developers to the same file, start the processing of next GX Developer after the processing of current GX Developer is completed.

#### (3) Simultaneous access to different files from multiple GX Developers

The High Performance model QCPU allows simultaneous access from other GX Developers to up to 10 different files of the same CPU module.

<b>REMARK</b>
---------------

For details on the PLOAD instruction, refer to the QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions).

## 6.9.3 File capacity

The file size differs with the types of files used. When a program memory, standard RAM, standard ROM, and memory card are used, calculate the size of a file by referring to the table 6.7 shown below.

Table 6.7 List of File Capacity

Function	Estimated File Size (in bytes)
Drive header	64
Parameter	Default: 564 (increased by the parameter setting) For Reference: Boot setting to $70 + (18 \times (\text{Number of files}))$ With the MELSECNET/H setting to maximum 4096 / units increased With Ethernet setting to maximum 922 / units increased With CC-LINK setting to maximum 251 / units increased With remote password setting to $64 + 20 + (\text{number of target modules} \times 10)$ , increased to up to 164
Sequence program	$136 * 1 + (4 \times ((\text{number of steps}) + (\text{number of steps secured for write during RUN})))$
Device comment	$74 + (\text{Total of comment data size of each device})$ • Comment data size of a device = $10 + 10250 \times a + 40 \times b$ • a: quotient of $(\text{Number of device points}) / 256$ • b: remainder of $(\text{Number of device points}) / 256$
Device init	$66 + 44 \times n + 2 \times (\text{Total number of device points specified in the device initial value setting})$ • n : specified number of device initial values
User setting area	When set at the time of formatting (0 to 15k)
Multiple-block online change setting	When set at the time of formatting (0/1.25k/2.5k)
File register	$2 \times (\text{Number of file register points})$
Sampling trace data	$362 + (20 + 2 \times (\text{Number of word device points}) + (\text{Number of bit device points}) / 8) \times (\text{Number of traces}) + 12 \times (\text{Device range}) * 2$
Failure history data	$72 + 54 \times (\text{Number of failures stored})$
Local device	$70 + 6 (\text{Type of specified device}) + 2 \times ((\text{total number of M and V points}) / 16 + (\text{D points}) + 18 \times (\text{Total points of T, ST and C}) / 16) \times (\text{number of programs}) * 1$ • Symbols "M, V, D, T, ST, and C" stand for the following devices: M: internal relay V: edge relay D: data register T: timer ST: relative timer C: counter

\*1: 136 is the default (increases depending on the parameter setting).

\*2: Round down the fractional portions of  $(\text{bit devices})/8$ ,  $(\text{total number of M and V points})/16$ , and  $(\text{total number of T, ST and C points})/16$ .

### 6.9.4 Memory capacity for files

The memory capacity for files indicates a capacity after the files have been written to the memory area.

For the files transferred from the memory card to the program memory in boot operation, note that the secured memory capacity changes after transfer.

#### (1) File size unit

When files are written to the memory area, the storage capacity unit changes depending on the CPU module and memory area to which the files will be written. This unit is called the file size unit.

##### (a) File size units classified by memory areas

The following table indicates the file size units classified by the CPU module and memory area to which files will be written.

Memory Area CPU Module Type	File Size Unit of Program Memory/Standard ROM/Flash Card * 1
Q02CPU	128 steps/512 bytes * 2
Q02HCPU	
Q06HCPU	
Q12HCPU	256 steps/1024 bytes * 2
Q25HCPU	512 steps/2048 bytes * 2

\*1: The file size unit of the Flash card applies to the case where the program memory contents are written to the Flash card via the CPU module using GX Developer.

\*2: 1024 steps/4096 bytes for the CPU module whose first five digits of serial No. are "04121" or earlier.

##### (b) File size units classified by memory cards

Type	Memory Card Model Name	Memory Capacity
SRAM card	Q2MEM-1MBS	512 bytes
	Q2MEM-2MBS	1024 bytes
Flash card	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 either of the following cases.

- 1) Where files are written to the Flash card via the CPU module using GX Developer.
- 2) Where files bypass the CPU module and are written to the Flash card using GX Developer.



(2) Memory capacity calculation example

When files are written to the memory area, the storage capacity unit changes depending on the CPU module and memory area to which the files will be written. This unit is called the file size unit.

(a) Conditions

- 1) Write target CPU module: Q25HCPU
- 2) Written files

File Name	File Capacity
PARAM. QPA (parameter file)	564 bytes
MAIN. QPG (sequence program)	525 steps/2100 bytes * 1

\* 1: Indicates the program capacity (file header + executed program) displayed on GX Developer. (Refer to Section 6.8.)

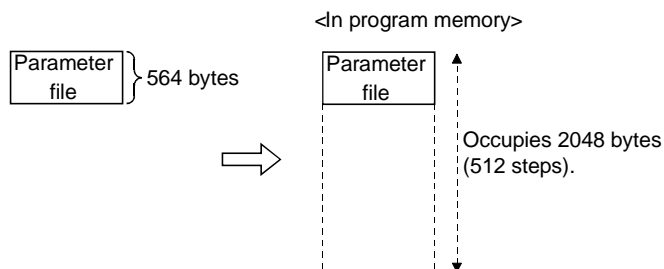
- 3) Steps secured for write during RUN: 500 steps/2000 bytes

(b) Memory capacity calculation

Memory capacity calculation is based on the file size unit of the write target CPU module. In the example, the file size unit of the Q25HCPU is 512 steps/2048 bytes according to Section (1).

1) Parameter file capacity calculation

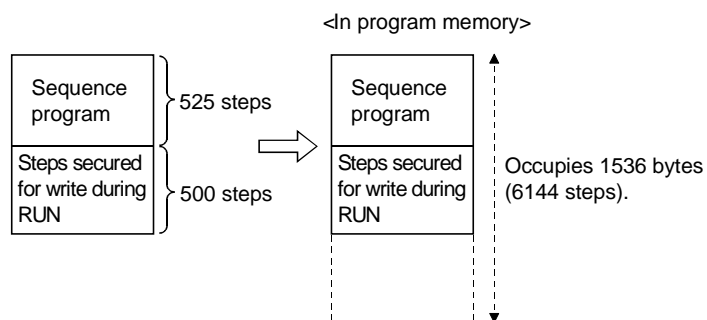
The parameter file capacity is 564 bytes. However, since the parameter file is stored in the file size unit in the program memory, it occupies 512 steps/2048 bytes.



2) Program capacity calculation

The program capacity is the sequence program capacity + steps secured for write during RUN.

In the example, the program capacity is 525 steps + 500 steps = 1025 steps. However, since the program file is stored in the file size unit in the program memory, it occupies 1536 steps/6144 bytes.



3) 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)
	Steps secured for write during RUN	500 steps	
	Total	1025 steps	
Memory capacity total			2048 steps (8192 bytes)

POINT

- (1) The file, which was executed in the CPU whose serial No. is "04122" or later, may not be executed in the CPU whose serial No. is "04121" or earlier.
- (2) About combination of write to PLC and GX Developer  
The following table indicates the combinations of the CPU module and GX Developer versions used when files are read from the CPU module to GX Developer and written to the other CPU module.

Legend ◎: Write to PLC enabled ○: Restrictions on write to PLC

Write Source CPU Write Destination CPU	GX Developer version8		GX Developer version7	
	Files executed in CPU whose serial No. is "04122" or later	Files executed in CPU whose serial No. is "04121" or earlier	Files executed in CPU whose serial No. is "04122" or later	Files executed in CPU whose serial No. is "04121" or earlier
Serial No. "04122" or later	◎	◎	○ * 2	○ * 2
Serial No. "04121" or earlier	○ * 1	◎	○ * 1 * 2	○ * 2

\* 1: Since the file size units differ, the files may not be stored into the CPU depending on the file capacity.  
\* 2: Unless the steps secured for write during RUN is decreased, the files may not be stored into the CPU depending on the file capacity.

## 7 FUNCTION

Function of High Performance model QCPU module is as follows:

## 7.1 Function List

Functions of High Performance model QCPU are listed below:

Item	Description	Reference section
Constant scan	This function executes the program in a set time interval regardless of the program scan time.	Section 7.2
Latch function	This function maintains the device data when performing the reset operation during power off.	Section 7.3
Output status selection function for transition from STOP status to RUN status	This function selects the output Y status (output before STOP/output after the calculation execution) when the CPU module is set from STOP status to RUN status.	Section 7.4
Clock function	This function executes the CPU module internal clock.	Section 7.5
Remote operation	This function operates the CPU module from a remote place.	Section 7.6
Remote RUN/STOP	This function stops and starts operating the CPU module.	Section 7.6.1
Remote PAUSE	This function stops the CPU module operation while retaining the output (Y) of the CPU module.	Section 7.6.2
Remote RESET	This function resets the CPU module when the CPU module is in a STOP status.	Section 7.6.3
Remote latch clear	This function clears the latch data of the CPU module when the CPU module is in a STOP status.	Section 7.6.4
Input response time selection for input module compatible with Q Series	The response time of the input module/ Composite I/O module compatible with Q Series can be selected from 1 ms, 5 ms, 10 ms, 20 ms and 70 ms with this function. (Default: 10 ms)	Section 7.7.1
Input response time selection for high speed input module compatible with Q Series	The response time of the high speed input module compatible with Q Series can be selected from 0.1 ms, 0.2 ms, 0.4 ms, 0.6 ms and 1 ms with this function. (Default: 0.2 ms)	Section 7.7.2
Input response time selection for interrupt module compatible with Q Series	The response time of the interrupt module compatible with Q Series can be selected from 0.1 ms, 0.2 ms, 0.4 ms, 0.6 ms and 1 ms with this function. (Default: 0.2 ms)	Section 7.7.3
Error-time output mode setting	This function sets whether the output to the Q series compatible output module, hybrid I/O module or intelligent function module will be cleared or held when the CPU module results in a stop error.	Section 7.8
Hardware error-time CPU operation mode setting	This function sets whether the operation of the CPU module will be stopped or continued when the hardware error of the intelligent function module occurs.	Section 7.8
Switch setting of intelligent function module compatible with Q Series	Use this function for various settings of the intelligent function module. (Refer to each intelligent function module for the details of the setting.)	Section 7.10
Monitoring function	This function monitors the status of programs and devices on the CPU module by operating from the GX Developer.	Section 7.11
Set monitor conditions	This function monitors using a fine timing of the CPU module.	Section 7.11.1
Monitor/test local Devices	This function monitors and/or tests the local devices of the designated program using the GX Developer.	Section 7.11.2
Turn ON/OFF external I/O	This function forcibly turns the external I/O of the CPU module on or off from the GX Developer.	Section 7.11.3
Write during RUN	This function writes programs when the CPU module is in the RUN status.	Section 7.12
Measure execution time	This function displays the processing time of a program being executed, the number of times to execute an interrupt program, and the execution time of a program.	Section 7.13
Program list monitor	This function displays the processing time of a program being executed.	Section 7.13.1
Interrupt program monitor	This function displays the number of times to execute an interrupt program.	Section 7.13.2
Scan time measurement	This function measures the execution time of a program between selected steps.	Section 7.13.3
Sampling trace function	This function samples specified device data from the CPU module at a specified timing.	Section 7.14
Multiple-user debugging function	This function enables multiple users to debug programs by using several GX Developers.	Section 7.15
Watch dog timer	This function monitors operational delays caused by CPU module's hardware and program errors.	Section 7.16
Self-Diagnosis function	This function enables the CPU module to check for failures.	Section 7.17
Failure history	This function stores a failure history of diagnosis results in the memory.	Section 7.18
System protect	This function prevents the programs from being modified from GX Developer, serial communication module or like.	Section 7.19
Password registration	This function provides read/write protection for files stored in the CPU module against access from the GX Developer.	Section 7.19.1
Remote password	A function to prevent illegal access from external sources with serial communication modules and Ethernet modules.	Section 7.19.2
System display	This function connects to the GX Developer and monitors system configuration.	Section 7.20
LED display	This function enables the front-mounted LEDs to indicate the operating conditions of the CPU module.	Section 7.21
LED display	This function indicates the normal or abnormal operating conditions of the CPU module.	Section 7.21.1
Preference setting	This function sets failure preferences to turn off LED displays.	Section 7.21.2
High speed interrupt function	This function executes an interrupt program by fixed scan interrupt at 0.2ms to 1.0ms intervals using the interrupt pointer I49.	Section 7.22
Module service interval time read	This function monitors the access interval time (time between the access acceptance of the CPU module and the next access acceptance) of the intelligent function module, network module or peripheral device.	Section 7.23

7.2 Constant Scan

(1) What is 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.

Because I/O refresh is made prior to execution of the sequence program, use of the constant scan function helps maintain the I/O refresh interval at a constant rate even if the sequence program execution time varies.

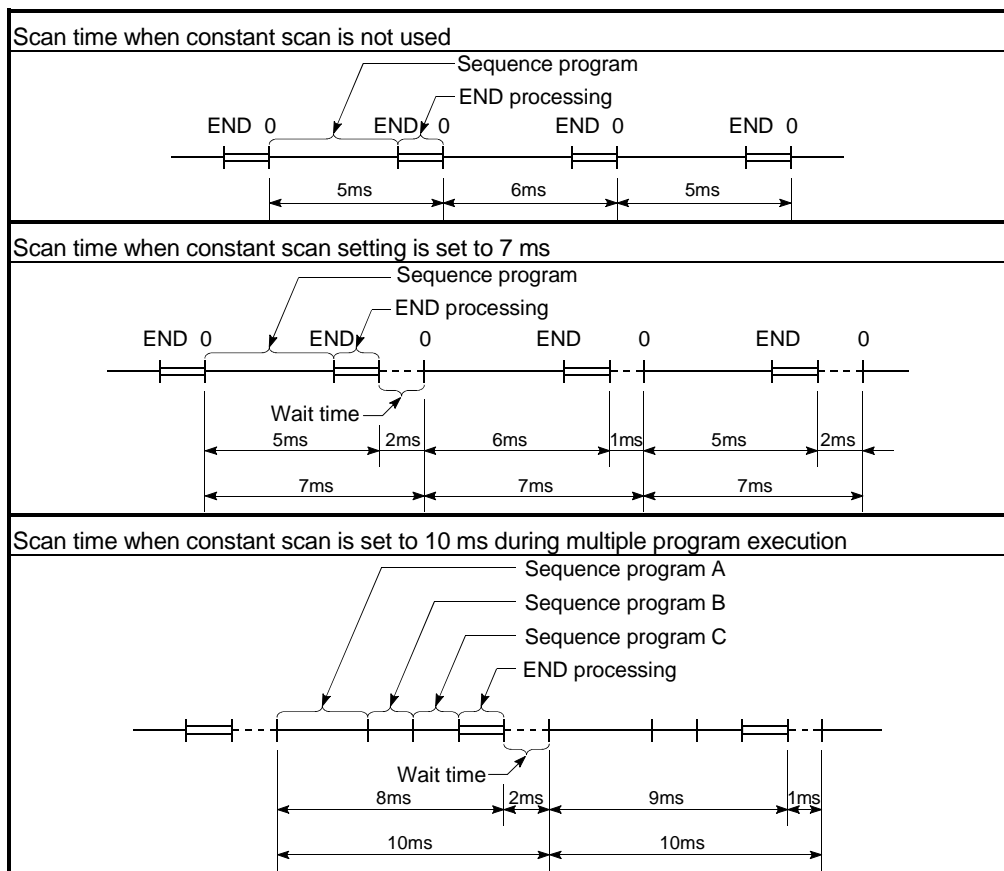


Fig. 7.1 Constant scan operation

**REMARK**

When using a low speed execution type program, the constant scan function setting or low speed execution type program execution time must be set.

(2) Setting the constant scan time

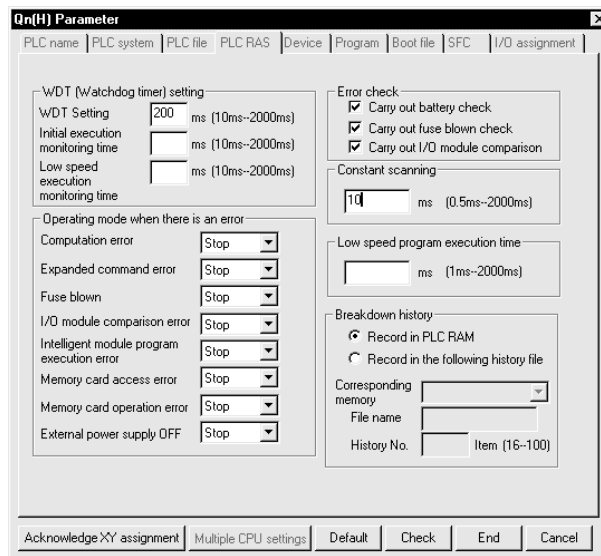
- (a) The constant scan time is set at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box.

The constant scan can be set in the range of 0.5 ms to 2000 ms.

A setting can be made in 0.5 ms units.

- When executing constant scan, set the constant scan time.
- When not executing a constant scan, leave the constant scan time blank.

[Example] When the constant scan is set to 10 ms.



- (b) Set the set time of the constant scan longer than the maximum scan time of the sequence program. Also, set the constant scan set time shorter than the WDT set time.

$$(WDT \text{ Set Time}) > (\text{Constant Scan Set Time}) > (\text{Sequence Program maximum Scan Time})$$

If the sequence program scan time is longer than the constant scan set time, the High Performance model QCPU detects PRG.TIME OVER (an error code: 5010), the sequence program is executed with the scan time by ignoring the constant scan.

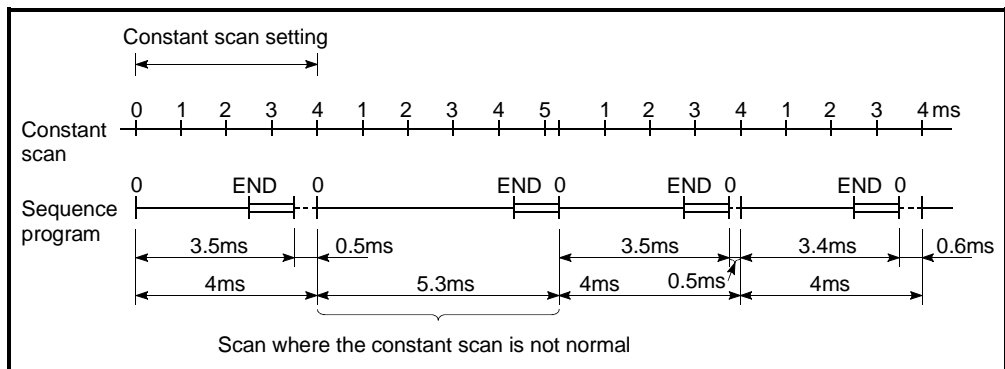


Fig. 7.2 Operation when the Scan Time is longer than the Constant Scan

If the time is longer than the WDT set time, the High Performance model QCPU detects a WDT error and stops the program execution.

- (c) The sequence program processing stops during the wait time from the last END processing execution until the next scan starts.
- 1) If a low speed execution type program is executed, it will be interrupted for - 0.5 ms (a constant scan time setting).
  - 2) If an interrupt factor occurs after the END processing is performed, the interrupt program or fixed scan execution type program is executed.
- (d) Constant Scan Time Accuracy
- This section describes the accuracy of a constant scan time setting.
- 1) The remaining portion (wait time) of a constant scan time setting is 0.02 ms on Q02CPU and 0.01 ms on Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU when the following programs are not executed.
    - low speed execution type program
    - interrupt program
    - fixed scan execution type program
  - 2) Wait time is 0.5 ms when a low speed execution type program is used. If the maximum processing time for one instruction in a low speed execution type program is 0.5 ms, the remaining portion of constant scan time is the same as described in 1). If the maximum processing time exceeds 0.5 ms, constant scan delays for an excessive duration.
  - 3) 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.

**REMARK**

Refer to "QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)" for the command processing time.

### 7.3 Latch Functions

#### (1) What is Latch Functions?

- (a) 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 power is turned on.
  - The reset operation is performed.
  - There is a momentary power failure for more than the permissible amount of time.

Latch function maintains the device information when the above conditions occur.

The availability of latches does not affect the operation performed by a program.

- (b) The latch can be used to continue control by maintaining the production quantity, defect count, and address even when there is a momentary power failure for more than the permissible amount of time.

- (c) The following devices can use the latch function:  
(The default latch range is only the latch relay.)

- 1) Latch relay(L)
- 2) Link relay(B)
- 3) Annunciator(F)
- 4) Edge relay(V)
- 5) Timer(T)
- 6) Retentive timer(ST)
- 7) Counter(C)
- 8) Data register(D)
- 9) Link register(W)

#### (2) Latch Range Setting

The latch range setting is performed at the "Device" tab screen in the "(PLC) Parameter" dialog box.

There are two types of range in which the latch clear key (RESET/L.CLR switch) and remote latch clear operation become valid or invalid in the latch range setting.

**(3) Clearing the Latch Range Device Data**

The status of devices to which "latch clear" is made is shown in the table below.

Latch setting	Clear/retention after "latch clear"
Devices not designated in latch range	Clear
Latch (1) setting (Devices with "latch clear" option)	Clear
Latch (2) setting (Devices without "latch clear" option)	Retain *

\* : Refer to Section 4.6 for the clearing method.

POINT
File registers (R) cannot be cleared with latch clear. (See Section 10.7 for clearing file registers.)

**(4) Precautions**

- (a) Even if the device has been latch-specified, it will not be latched when the local device or the device initialization is specified.
- (b) The device details of the latch range are maintained with the battery (Q6BAT) attached to the High Performance model QCPU.
  - 1) The battery is necessary to latch the device if ROM operation is performed using the sequence program that has been stored on the standard ROM or memory card.
  - 2) Take care that, if the battery connector is disconnected from the connector of the High Performance model QCPU when PLC is turned off, the latch range device memory is not retained but becomes undefined.



## 7.4 Setting the Output (Y) Status when Changing from/to STOP Status to/from RUN Status

## (1) Output (Y) Status when changing from STOP Status to RUN Status

When changing from RUN status to STOP status, the RUN status output (Y) is stored in the sequence and all the outputs (Y) are turned OFF.

The status after transition from STOP to RUN can be selected from the following two options with the High Performance model QCPU.

- The output status prior to STOP is output.
- The output is cleared.

(Default: After transition from STOP to RUN, the output (Y) status prior to STOP is output then the program is executed.)

## (a) Output (Y) status prior to STOP is output

After the output (Y) status before the STOP status is output, the sequence program calculations are performed.

## (b) Output is cleared

Clears all output (Y) and outputs the output (Y) after executing the sequence program calculations.

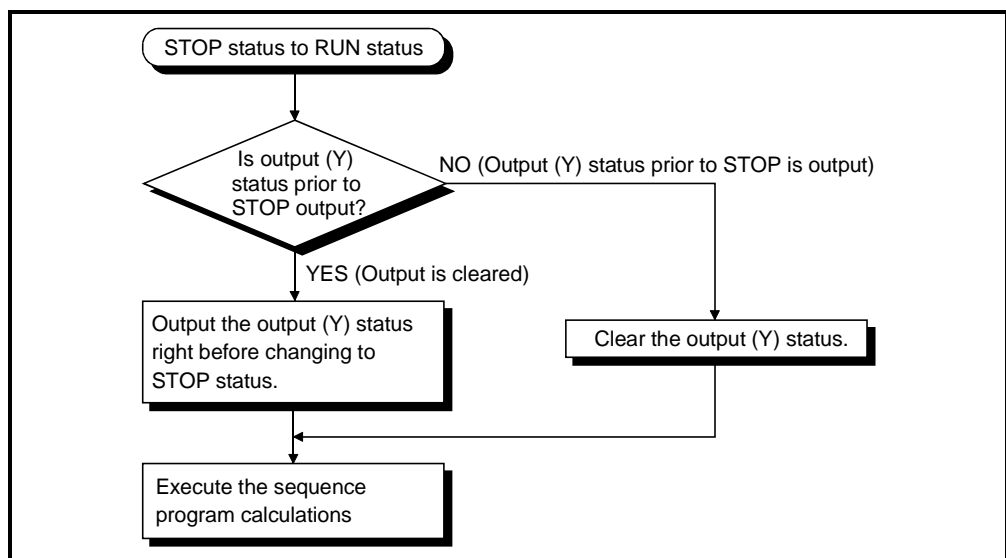
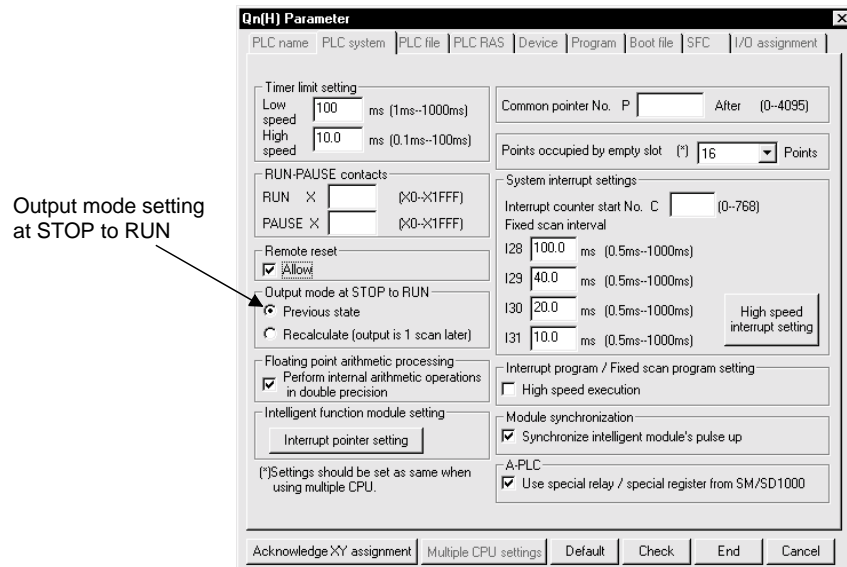


Fig. 7.3 Processing when Change from STOP Status to RUN Status

## (2) Setting the Output (Y) Status when Changing from STOP Status to RUN Status

The output (Y) status before the STOP status when switching from STOP status to Run status can be set at the "PLC System" tab screen in the "(PLC) Parameter" dialog box.



## (3) Precaution

If an output (Y) is forcefully turned ON with the High Performance model QCPU in the STOP status, it will not remain in the ON status even if the STOP status is switched to the RUN status.

The output status is effected as set for "Output mode at STOP to RUN" at the "PLC System" tab screen.

## 7.5 Clock Function

## (1) What is Clock Function?

(a) Clock function reads the clock data inside the High Performance model QCPU with a sequence program and uses the data for clock management. Also, the time data is used for time maintenance for the High Performance model QCPU system functions such as those for failure history. The clock operations are maintained using the battery (Q6BAT) even when the PLC power is off or when a momentary power failure for longer than the permitted time occurs.

## (b) Clock Data

The following table lists the time data, that are used for the High Performance model QCPU clock element.

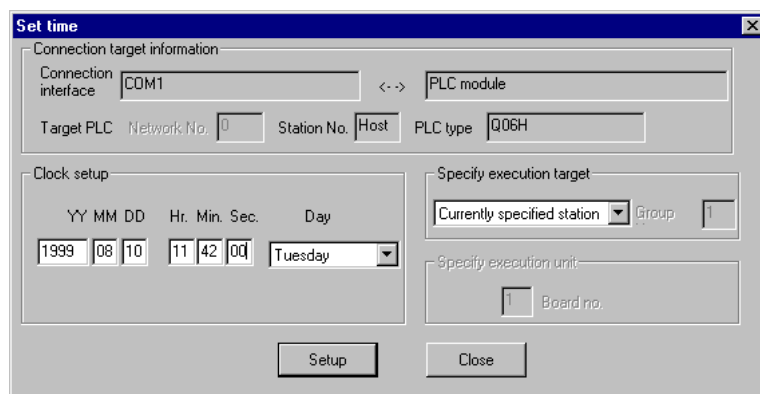
Data Name	Contents	
Year	Four digits in AD (Countable from 1980 to 2079)	
Month	1 to 12	
Day	1 to 31 (Automatic leap year calculation)	
Hour	0 to 23 (24 hours)	
Minute	0 to 59	
Second	0 to 59	
Day of the week	0	Sunday
	1	Monday
	2	Tuesday
	3	Wednesday
	4	Thursday
	5	Friday
	6	Saturday

## (2) Writing/Reading Time Data To/From Clock Element

(a) The following two methods can be used to write to the time data clock element.

## 1) Method to write from GX Developer

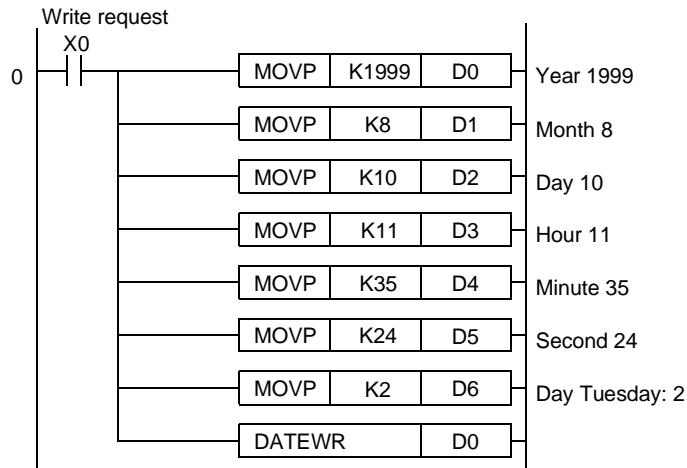
The time data is written to the clock element by displaying "Online" → "Set time" window.



2) Method to Write from the Program

The time data is written to the clock element by using the clock instruction (DATEWR).

A program example to write the time data using the time data write instruction (DATEWR).

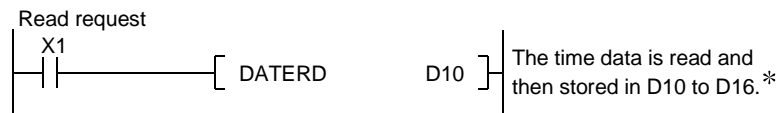


Refer to "QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)" for details on the DATEWR instruction.

(b) Reading Time Data

When reading the time data to the data register, use the time data read instruction (DATERD) from the program.

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.



Refer to the "QCPU (Q mode)/QnACPU Programming Manual (Common instructions) for the details on the DATERD instruction.

**REMARK**

- 1) Time Data can be written to and read from by special relays (SM210 to SM213) and special registers (SD210 to SD213). See Appendix 1 for details on special relay. See Appendix 2 for details on special registers.

- 2) \*: The figure below shows the clock data stored in D10 to D16.

D10	1999	4 digits in AD	} See Section 7.5(1).
D11	8	Month	
D12	10	Date	
D13	11	Hour	
D14	35	Minute	
D15	24	Second	
D16	2	Day of the week	

**(3) Precautions**

- (a) The clock data is not set prior to shipment.

The clock data is used in High Performance model QCPU system and intelligent function module for failure history and other functions. Be sure to set the accurate time when operating the High Performance model QCPU for the first time.

- (b) Even when a part of the time data correcting, all data must be written to the clock element again.

- (c) The data to be written to the clock element is checked in the range described in (1) (b) of Section 7.5.

For this reason, if improbable clock data in the range described in (1) (b) of Section 7.5 is written to the clock element, correct clock operation is unavailable.

**Example**

	Writing to clock element	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

**(4) Accuracy of Clock Data**

The accuracy of the clock function differs with the ambient temperature, as shown below:

Ambient Temperature (°C)	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)

**(5) 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. The year data is read out in 4 digits. It can be compared as it is by using a compare instruction.

## 7.6 Remote Operation

The High Performance model QCPU provides the RUN/STOP switches for switching between the STOP status and the RUN status. The RESET/L.CLR switch also provides the Reset and Latch Clear functions.

The High Performance model QCPU can allow control of the High Performance model QCPU operation status by external operations

(GX Developer function, intelligent function module, and remote contact).

The following four options are available for remote operations:

- Remote RUN/STOP
- Remote PAUSE
- Remote RESET
- Remote LATCH CLEAR

### REMARK

The serial communication module is used as the example to describe the intelligent function module.

### 7.6.1 Remote RUN/STOP

#### (1) What is Remote RUN/STOP?

- (a) The remote RUN/STOP performs RUN/STOP of the High Performance model QCPU externally with the CPU module RUN/STOP switch at RUN.
- (b) Using remote RUN/STOP for the following remote operations are useful:
  - 1) When the High Performance model QCPU is at a position out of reach
  - 2) When performing RUN/STOP of the control board High Performance model QCPU externally

#### (c) Calculations during Remote RUN/STOP

The program calculation that performs remote RUN/STOP is as follows:

- 1) Remote STOP  
Executes the program to the END instruction and enters the STOP status.
- 2) 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.

(2) Method with Remote RUN/STOP

There are two ways to perform remote RUN/STOP:

(a) Method with remote RUN contact

The remote RUN contact is set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box of GX Developer.

The device can be set in the range of input X0 to 1FFF.

By turning the set remote RUN contact ON/OFF, the remote RUN/STOP can be performed.

- 1) When the remote RUN contact is OFF, the QCPU enters the RUN status.
- 2) When the remote RUN contact is ON, the QCPU enters the STOP status.

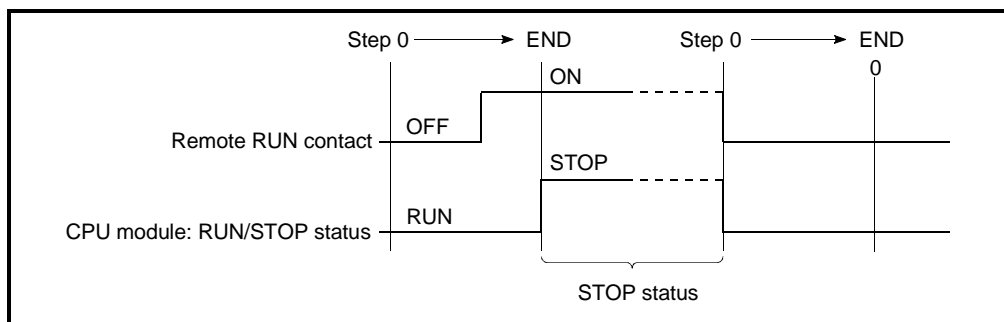


Fig. 7.4 Time Chart for RUN/STOP with Remote RUN Contact

(b) Method with GX Developer, serial communication module, etc.

High Performance model QCPU RUN/STOP can be performed by the remote RUN/STOP operation from the GX Developer, serial communication module, etc.

The GX Developer operation is performed with on-line remote operations.

The serial communication module and Ethernet interface module are controlled by commands complying with the MC protocol.

For details of the MC protocol, refer to the following manual.

- Q Corresponding MELSEC Communication Protocol Reference Manual

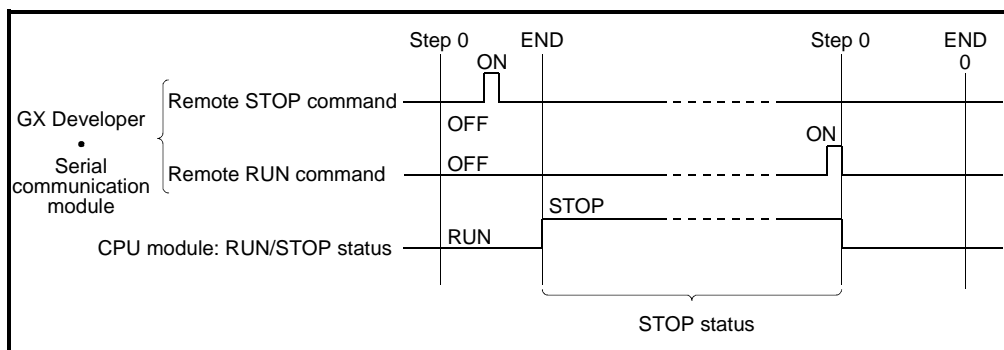


Fig. 7.5 Remote RUN/STOP Time Chart using GX Developer, serial communication module, etc.

### (3) Precautions

- (a) Take note of the following, because STOP has priority in High Performance model QCPU:
- 1) The High Performance model QCPU enters the STOP status when remote STOP is performed from remote RUN contact, GX Developer, or by using serial communication module.
  - 2) To set the High Performance model QCPU to RUN status from STOP status again, perform the remote RUN from the external factor (remote RUN contact, GX Developer, serial communication module, etc.) from which the remote STOP was performed.

REMARK
--------

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.



7.6.2 Remote PAUSE

(1) What is Remote PAUSE?

- (a) The remote PAUSE performs PAUSE of the High Performance model QCPU externally with the CPU module RUN/STOP switch at RUN position. The PAUSE function stops the High Performance model QCPU calculations while maintaining the ON/OFF status of all output (Y).
- (b) This can be used to maintain the output (Y) on even if the High Performance model QCPU is changed to STOP status, in such areas as process control.

<b>POINT</b>
The output (Y) is turned off upon a stopping error. To retain the output even upon a stopping error, use I/O allocation of PC parameters to set output retention.

(2) Method with Remote PAUSE

There are two ways to use remote PAUSE:

(a) Method with remote PAUSE Contact

The remote PAUSE contact is set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box of GX Developer.

The device can be set in the range of input X0 to 1FFF.

- 1) The PAUSE status contact (SM204) is turned on when the END processing is executed for the scan with both remote PAUSE contact and PAUSE permission flag (SM206) on.
- 2) When the remote 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.

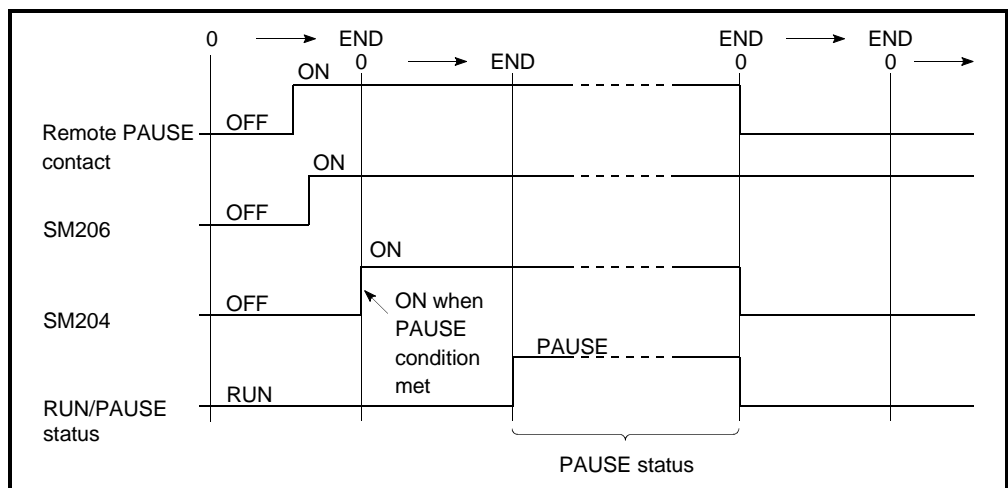


Fig. 7.6 PAUSE Time Chart with Remote PAUSE Contact

(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.

The GX Developer operation is performed by on-line remote operation.

The serial communication module and Ethernet interface module are controlled by commands complying with the MC protocol.

For details of the MC protocol, refer to the following manual.

• Q Corresponding MELSEC Communication Protocol Reference Manual

- 1) 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.

- 2) When the remote RUN command is received, the sequence program calculations are performed again from step 0.

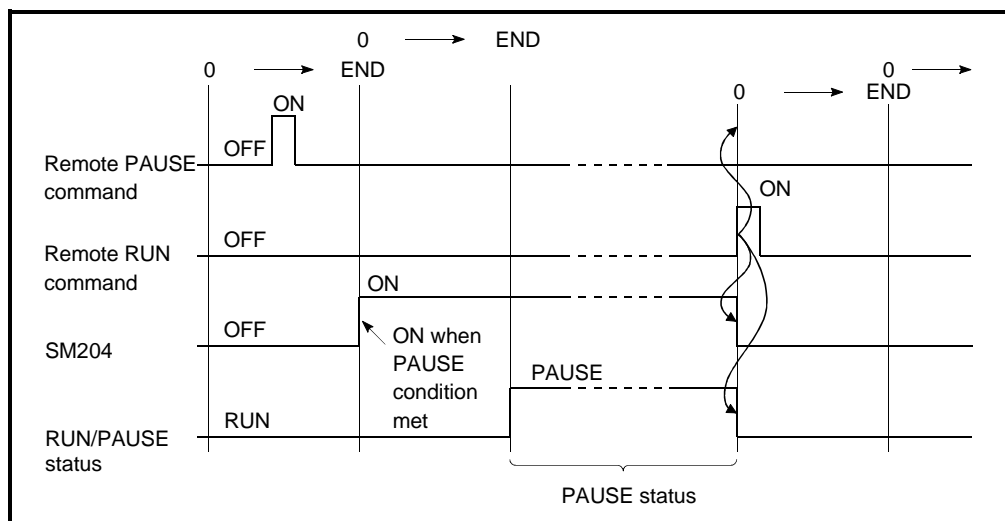
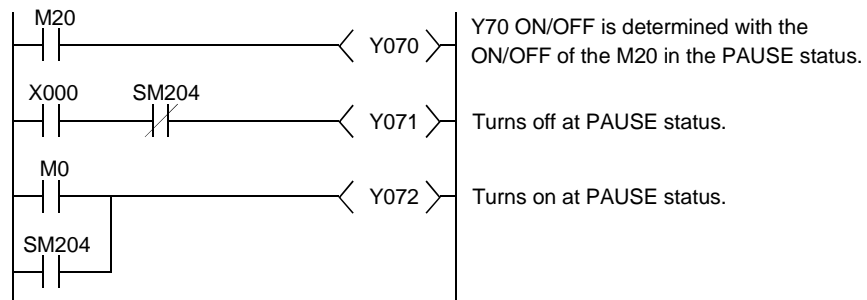


Fig. 7.7 PAUSE Time Chart with GX Developer

(3) Precaution

To set the output (Y) ON/OFF status when change to the PAUSE status, perform an interlock with the PAUSE status contact (SM204).



## 7.6.3 Remote RESET

## (1) What is Remote RESET?

- (a) The remote RESET resets the High Performance model QCPU externally when the High Performance model QCPU is at STOP status. Even if the High Performance model QCPU RUN/STOP switch is at RUN, the reset can be performed when the High Performance model QCPU is stopped and an error that can be detected by the self-diagnosis function occurs.
- (b) Remote RESET can reset the High Performance model QCPU remotely when an error occurs for which the High Performance model QCPU cannot be operated directly. Remote RESET can be executed only in STOP status. When the High Performance model QCPU is in RUN status, use Remote STOP to arrange the STOP status.

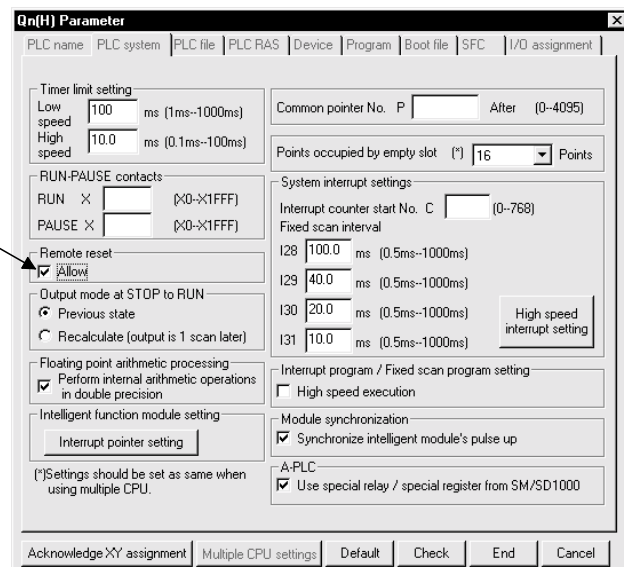
## (2) Remote RESET Method

The remote RESET can only be performed from GX Developer or by operating serial communication module.

To perform the remote RESET, follow the following steps:

- (a) 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 High Performance model QCPU.

Allow the  
"Remote reset"



- (b) When the High Performance model QCPU is in RUN status, use remote STOP to arrange the STOP status.
- (c) Reset High Performance model QCPU by the remote RESET operation.
- 1) For the GX Developer, this is performed by on-line remote operation.
  - 2) The serial communication module and Ethernet interface module are controlled by commands complying with the MC protocol. For details of the MC protocol, refer to the following manual.
    - Q Corresponding MELSEC Communication Protocol Reference Manual

### (3) Precautions

- (a) 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 High Performance model QCPU. If the "Allow" check box is not checked, a remote RESET operation is not performed.
- (b) Remote RESET cannot be performed when the High Performance model QCPU is in RUN status.
- (c) After the reset operation is complete, the High Performance model QCPU will enter operation status set at the RUN/STOP switch.
  - 1) With the RUN/STOP switch in the "STOP" position, the High Performance model QCPU enters into the "STOP" status.
  - 2) With the RUN/STOP switch in the "RUN" position, the High Performance model QCPU enters into the "RUN" status.
- (d) Take care that Remote RESET does not reset High Performance model QCPU if an error occurs in the High Performance model QCPU due to noise.  
If Remote RESET does not reset, use the RESET/L.CLR switch to reset or turn the PLC off.

POINT
(1) If Remote RESET is executed when the High Performance model QCPU is stopped due to an error, the High Performance model QCPU enters the operation status set at the RUN/STOP switch after it is reset.
(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 High Performance model QCPU is not reset since the reset process is not performed in it. If the status of the High Performance model QCPU 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.

### 7.6.4 Remote latch clear

#### (1) What is Remote Latch Clear?

- (a) The remote latch clear resets the device data latched to the High Performance model QCPU using the GX Developer etc., when the High Performance model QCPU is in STOP status.
- (b) Remote latch clear is useful when the High Performance model QCPU is in the following areas. In these cases, the operations are performed in combination with the remote RUN/STOP.
  - When the High Performance model QCPU is at a position out of reach
  - When externally performing latch clear of the High Performance model QCPU inside a control panel.

#### (2) Remote Latch Clear Method

The remote latch clear can only be performed from GX Developer or by using serial communication module.

To perform the remote latch clear, follow the following steps:

- (a) Use the remote STOP to bring the High Performance model QCPU to STOP status.
- (b) Use the Latch Clear to bring the High Performance model QCPU to the Latch Clear status.
  - 1) The GX Developer operations are performed by on-line remote operation.
  - 2) The serial communication module and Ethernet interface module are controlled by commands complying with the MC protocol.  
For details of the MC protocol, refer to the following manual.
    - Q Corresponding MELSEC Communication Protocol Reference Manual
- (c) To return the High Performance model QCPU to RUN status after the remote latch clear, perform a remote RUN operation.

#### (3) Precautions

- (a) Either remote latch clear or latch clear by RESET/L.CLR switch cannot be performed when the High Performance model QCPU is in RUN status.
- (b) The latch range for the device set at the "Device" tab screen in the "(PLC) Parameter" dialog box includes a range that makes latch clear (RESET/L.CLR switch) valid or invalid.  
Remote latch clear operation is reset in the range of latch clear valid setting.
- (c) Devices that are not latched are cleared when the remote latch clear is performed.  
The data in the failure history storage memory of the High Performance model QCPU will also be cleared by a remote latch clear operation.

### 7.6.5 Relationship of the remote operation and High Performance model QCPU RUN/STOP switch

#### (1) Relationship of the Remote Operation and High Performance model QCPU Switch

The High Performance model QCPU operation status is as follows with the combination of remote operations to RUN/STOP switch.

Remote operation RUN/STOP switch	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 remote 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 remote 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 RESET or LATCH CLEAR can be performed if the High Performance model QCPU changed to the STOP status by a remote operation.

\*5 This includes a situation where the High Performance model QCPU 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

While a remote operation is being performed by one GX Developer, another remote operation cannot be performed by another GX Developer.

After a remote operation that is being performed by one GX Developer is cancelled, a new remote operation can be performed by another GX Developer.

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.

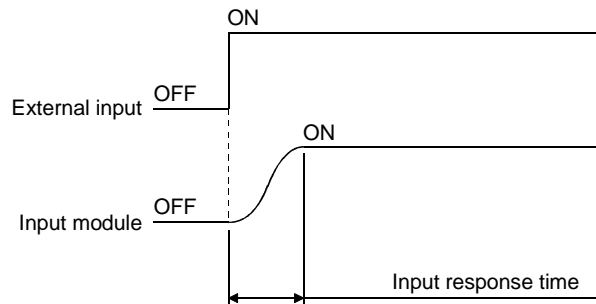
When a remote RUN operation is performed by the GX Developer that is performing a remote PAUSE operation, and then that remote operation is cancelled, a new remote operation can be performed by another GX Developer.

7.7 Selecting the Input Response Time of the Q Series Module (I/O Response Time)

7.7.1 Selecting the response time of the input module

(1) Selecting the response time of the input module

The input response time of a Q Series input module can be set to a desired response time: 1 ms, 5 ms, 10 ms, 20 ms or 70 ms. The input module reads external inputs at the specified response time. The default value of an input response time is 10 ms.



(2) Setting the Input Response Time

At the "I/O Assignment" tab screen in the "(PLC) Parameter" dialog box, specify the desired input response time. Select "Input" in the "Type" column of a slot for which to specify the desired input response time. For the Composite I/O module, choose "I/O mix".

Input module : Select "Input".  
 Composite I/O module: Select "I/O mix".      Select "Detailed setting".      Select "I/O response time".

The image shows two screenshots from a software interface. The left screenshot is the 'Qn(H) Parameter' dialog box, specifically the 'I/O Assignment' tab. It features a table with columns for Slot, PLC, Type, Model name, Points, and Start/End addresses. An arrow points to the 'Type' column where 'Input' is selected for slot 1. The right screenshot is the 'Intelligent functional module detailed setting' dialog box, showing a table with columns for Slot, Type, Model name, Error time output mode, H/W error time PLC operation mode, I/O response time, and Control PLC. An arrow points to the 'I/O response time' column where '10ms' is selected for slot 1.

(3) Reactions

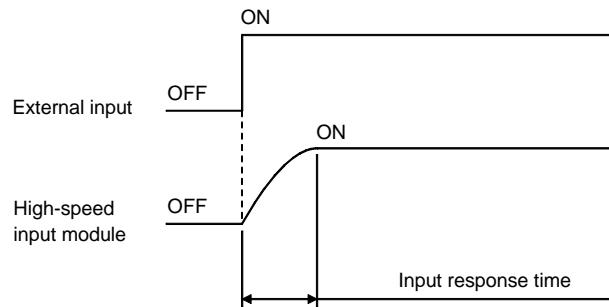
- (a) Higher input response time may result in response to inputs being influenced by noise. Set the desired input response time by taking into consideration the operating environment of an input module in use.
- (b) No change can be made to the input response time of an AnS Series corresponding input module. An input response time setting is not made for a slot of an AnS Series corresponding input module.
- (c) The input response speed setting is valid in the following cases.
  - After the PLC is turned on
  - When the High Performance model QCPU is reset

7.7.2 Selecting the response time of the high speed input module

(1) Selecting the response time of the high speed input module

Changing the response time of the high speed input module means to amend the input response speed for high speed input modules (QX40-S1) that support the Q Series to 0.1 ms, 0.2 ms, 0.4 ms, 0.6 ms and 1 ms.

Input from external devices is accepted at the input response speed set for the high speed input module. The default setting for the input response time is 0.2 ms.



(2) Setting the Input Response Time

Input response time is set at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box.

Select "Hi. input" among the slot types for which the input response time is to be set.

The image shows two screenshots from the GX Developer software. The first screenshot is the 'Qn(H) Parameter' dialog box, with the 'I/O assignment' tab selected. An arrow points to the 'Hi. input' option in the 'Type' column of the 'I/O Assignment(\*)' table. The second screenshot is the 'Intelligent functional module detailed setting' dialog box, with an arrow pointing to the 'I/O response time' column, where a dropdown menu is open showing options like 0.2ms, 0.1ms, 0.4ms, 0.6ms, and 1ms.

(3) Precautions

- (a) The system will be adversely affected by noise, etc., when the input response time is set to high speed. Set the input response time in consideration of the operating environment.
- (b) The GX Developer Version 5 (SW5D5C-GPPW-E) or later is required when changing the response time of the high speed input module. The input response time of the high speed input module cannot be amended with the GX Developer Version 4(SW4D5C-GPPW-E.) (Fixed at 0.2 ms (default setting).)
- (c) The input response speed setting is valid in the following cases.
  - After the PLC is turned on
  - When the High Performance model QCPU is reset

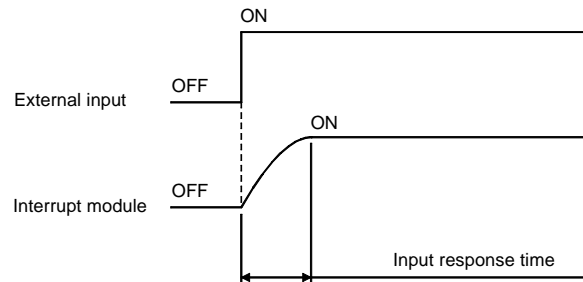


## 7.7.3 Selecting the response time of the interrupt module

## (1) Selecting the response time of the interrupt module

Changing the response time of the interrupt module means to amend the input response speed for interrupt modules (QI60) that support the Q Series to 0.1 ms, 0.2 ms, 0.4 ms, 0.6 ms and 1 ms.

Input from external devices is accepted at the input response speed set for the interrupt module. The default setting for the input response time is 0.2 ms.



## (2) Setting the Input Response Time

Input response time is set up at the "I/O assignment" tab screen in the "(PLC) Parameter" dialog box.

Select "Interrupt" among the slot types for which the input response time is to be set.

The figure shows two screenshots of the software interface. The first screenshot is the 'Qn(H) Parameter' dialog box, showing the 'I/O Assignment' tab. The 'Type' column for slot 1 is set to 'Interrupt'. The second screenshot is the 'Intelligent functional module detailed setting' dialog box, showing the 'I/O response time' column for slot 1 set to 0.2ms.

## (3) Precautions

- The system will be adversely affected by noise, etc., when the input response time is set to high speed. Set the input response time in consideration of the operating environment.
- The input response time cannot be amended for the A1S161 (interruption module that supports the AnS Series.) No processing will be performed with interruption modules that support the AnS Series, even if the input response time is set in the slot.
- The GX Developer Version 6 (SW6D5C-GPPW-E) or later is required when changing the response time of the interrupt module. The input response time of the interrupt module cannot be amended with the SW5D5C-GPPW-E or earlier GX Developers. (Fixed at 0.2 ms (default setting.))
- The input response speed setting is valid in the following cases.
  - After the PLC is turned on
  - When the High Performance model QCPU is reset

7.8 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 compatible output module, hybrid I/O module or intelligent function module will be cleared or held when the CPU module results in a stop error.

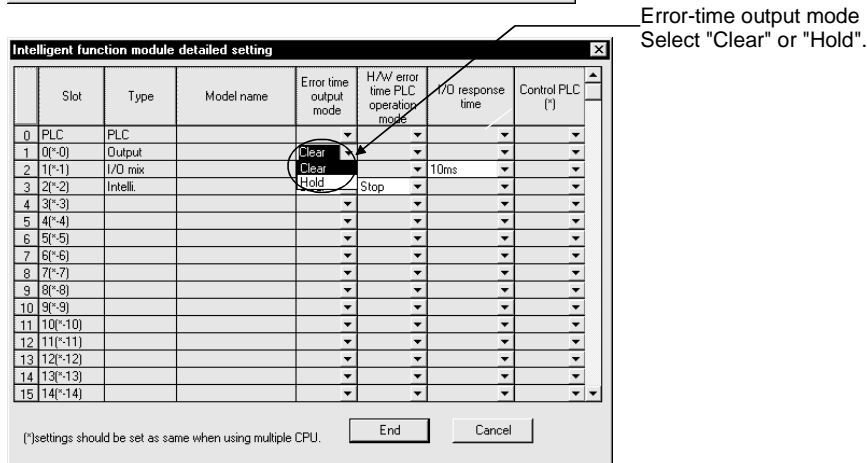
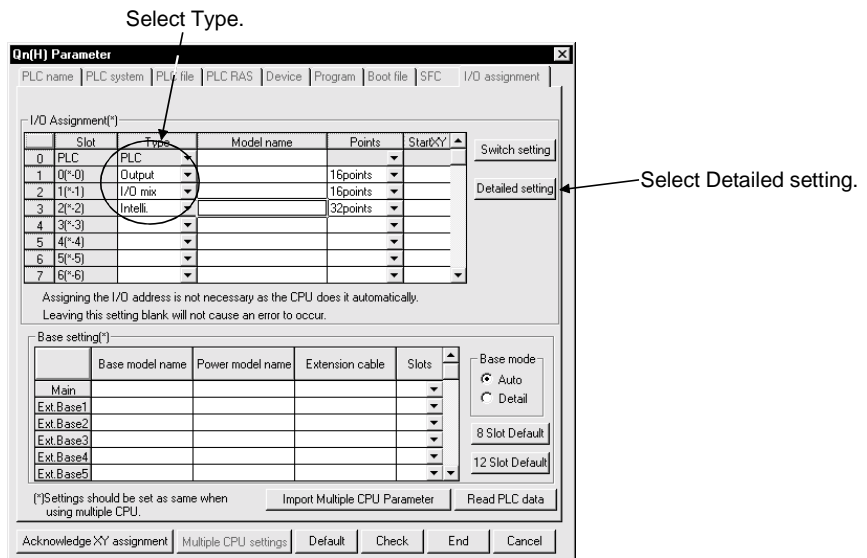
(2) Error-time Output Mode Setting

Make the error-time output mode setting in the I/O assignment of the PLC parameter dialog box.

Choose "Output", "Hybrid I/O" or "Intelligent" as the type of the slot to which the error-time output mode is set.

Then select Detailed setting.

Choose "Clear" or "Hold" for the slot to which the error-time output mode is set. (The default is "Clear".)



(3) Precautions

The error-time output mode setting is made valid when:

- The PLC CPU is powered ON; or
- The CPU module is reset.

Failure to perform either operation after changing the error-time output mode setting will result in "PARAMETER ERROR (error code: 3000)".

7.9 Hardware Error-time CPU Operation Mode Setting

(1) Hardware Error-time CPU Operation Mode Setting

The hardware error-time CPU 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.

(2) Hardware Error-time CPU Operation Mode Setting

Make the hardware error-time CPU operation mode setting in the I/O assignment of the PLC parameter dialog box.

Choose "Intelligent" as the type of the slot to which the hardware error-time CPU operation mode is set.

Then select Detailed setting.

Choose the hardware error-time CPU operation mode of the slot to which the hardware error-time CPU operation mode is set.

(The default is "Stop".)

Select Intelligent.

Select Detailed setting.

Hardware error-time CPU operation mode  
Select "Stop" or "Continue".

(3) Precautions

(a) The hardware error-time CPU operation mode setting is made valid when:

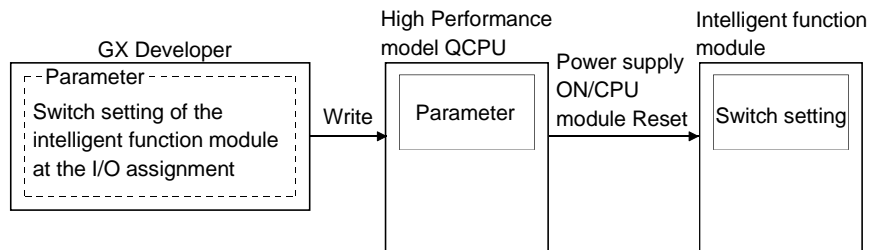
- The PLC CPU is powered ON; or
- The CPU module is reset.

Failure to perform either operation after changing the hardware error-time CPU operation mode setting will result in "PARAMETER ERROR (error code: 3000)".

7.10 Setting the Switches of the Intelligent Function Module

(1) Setting the Switches of the Intelligent Function Module

The switches of the intelligent function module is to set the switches of an Q Series intelligent function module using GX Developer. The settings of the switches set by GX Developer is written from High Performance model QCPU to each intelligent function module at the leading edge or reset of High Performance model QCPU.



(2) Setting the Switches of the Intelligent Function Module

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 module.

Select "Intelli." (pointing to the 'Type' column in the I/O Assignment table)

Select "Switch Setting" (pointing to the 'Switch setting' dropdown in the I/O Assignment dialog)

Designate the contents of the intelligent function module switch. (pointing to the switch selection area in the Switch setting dialog)

Slot	PLC	Type	Model name	Points	StartCV
0	PLC	PLC			
1	0(*-0)	Intelli.		16points	
2	1(*-1)				
3	2(*-2)				
4	3(*-3)				
5	4(*-4)				
6	5(*-5)				
7	6(*-6)				

Slot	Type	Model name	Switch 1	Switch 2	Switch 3	Switch 4	Switch 5
0	PLC	PLC					
1	0(*-0)	Intelli.					
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) Precautions

- (a) Do not apply the switch setting for an intelligent function module to an AnS Series corresponding special function module. If the switch setting for an intelligent function module is specified for an AnS Series corresponding special function module, an error (SP.PARA.ERROR) will occur.
- (b) For details on the switch setting for an intelligent function module, refer to the manual of the intelligent function module in use.
- (c) 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". The switch settings for interruption modules with the SW5D5C-GPPW-E or earlier GX Developers are made by setting the type to "Intelligent". Refer to the following manual for further details on the interruption module's switch settings.
  - Building Block I/O Module User's Manual
- (d) The switch setting of the intelligent function module is valid in the following cases.
  - After the PLC is turned on
  - When the High Performance model QCPU is reset

## 7.11 Monitoring Function

## (1) What is Monitoring Function?

- (a) This is a function to read the program, device and intelligent function module status of the High Performance model QCPU by using GX Developer.

The High Performance model QCPU performs the END processing to handle monitor requests from GX Developer.

The results of High Performance model QCPU's END processing are displayed on the GX Developer side.

- (b) By setting the monitoring conditions with GX Developer, it is possible to monitor the High Performance model QCPU operation status under the specified conditions.

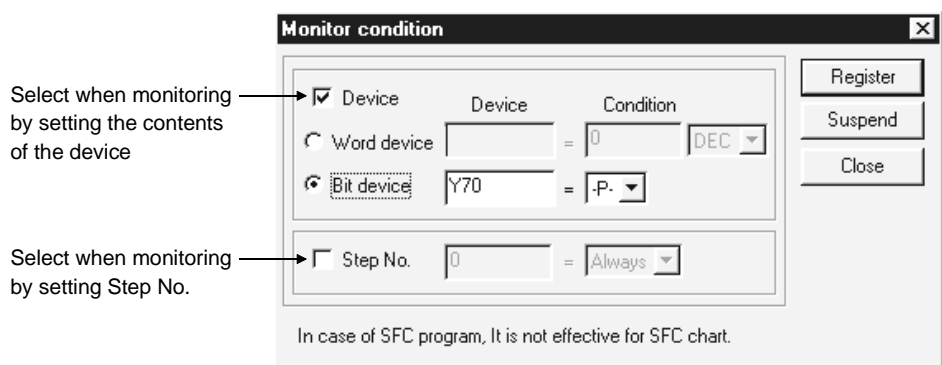
It is also possible to maintain the monitoring status under the specified conditions by setting the monitoring stop conditions.

- (c) The use of local devices for execution of multiple programs makes it possible to monitor local device data.

## 7.11.1 Monitor condition setting

## (1) Setting monitor execution conditions when monitoring circuits

Choose "Online" → "Monitor" → "Monitor condition" 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.



- (a) When only "Step No." is specified:

- 1) The monitor data sampled when the status previous to execution of the specified step becomes "the specified".
- 2) The specification method for the execution status is indicated below:
  - a) When changing from non-execution status to executing status : <-P->
  - b) When changing from executing status to non-execution status : <-F->
  - c) Always when executing only : <ON>
  - d) Always when not executing only : <OFF>
  - e) Always regardless of status : <Always>

**REMARK**

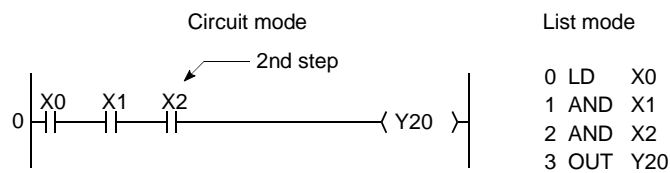
When "Step No. [ 0]" is specified, set the condition to "Always".

**POINT**

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 2nd step is ON (Step No. [ 2] = <ON>).

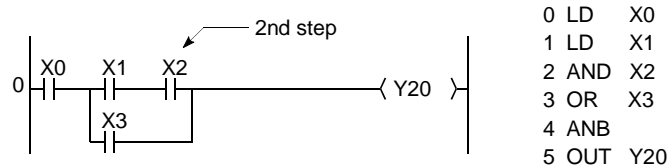
(1) When the 2nd step is connected by the AND instruction:

As shown below, the monitor execution condition is established when both "X0" and "X1" are ON.

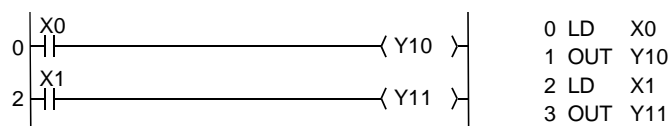


(2) When the 2nd step is connected between the AND/OR blocks:

As shown below, the monitor execution condition is established when "X1" is ON. Whether "X0" is ON or OFF, it does not affect the monitor execution condition.



(3) 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.



(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.

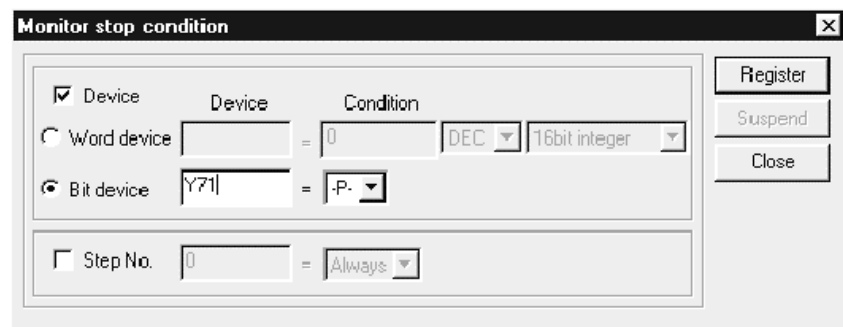
**POINT**

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 100th step where D1=5.

The monitor interval of GX Developer depends on the processing speed of GX Developer. When a monitor condition is established during the monitor interval of GX Developer, the monitor will be performed, even if a monitor condition is established at a shorter interval than the monitor interval.

## (2) Monitor Stop Condition Set Up

Choose "Online" → "Monitor" → "Monitor stop condition" to open the "Monitor Stop Condition" dialog box. The following shows an example of stopping a monitoring operation at the leading edge of Y71.



- (a) When "Step No." is specified:
- 1) Monitoring is stopped when the status at execution of the specified step becomes as specified.
  - 2) The specification method for the execution status is shown below:
    - a) When changing from non-execution status to executing status : <-P->
    - b) When changing from executing status to non-execution status : <-F->
    - c) Always when executing only : <ON>
    - d) Always when not executing only : <OFF>
    - e) Always regardless of status : <Always>
  - 3) When "Step No." is not specified, the monitoring operation is stopped after the High Performance model QCPU 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) When monitoring after setting the monitor condition, the file displayed on GX Developer is monitored. Match the file to be monitored by executing the "New PLC Read" and file name on GX Developer.
- (b) When monitoring the file register which is not specified, 0 is displayed.
- (c) Perform the monitoring by matching the device allocation of the High Performance model QCPU and GX Developer.
- (d) 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) 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.
  - The detailed condition setting for the monitoring can only be set for one user.
- (f) The monitoring detailed condition setting can only be set in circuit monitor.
- (g) When the same device is specified as the monitor condition and monitor stop condition, specify "ON" or "OFF".
- (h) 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.
  - 1) When skipping steps specified with the CJ command, the SCJ command and the JMP command.
  - 2) When the specified step is the END command, the FEND command exists while the program is running, and the END command is not executed.
- (i) Do not reset the High Performance model QCPU while monitoring conditions are being registered.



7.11.2 Monitoring test for local device

(1) Monitoring and Testing Local Devices

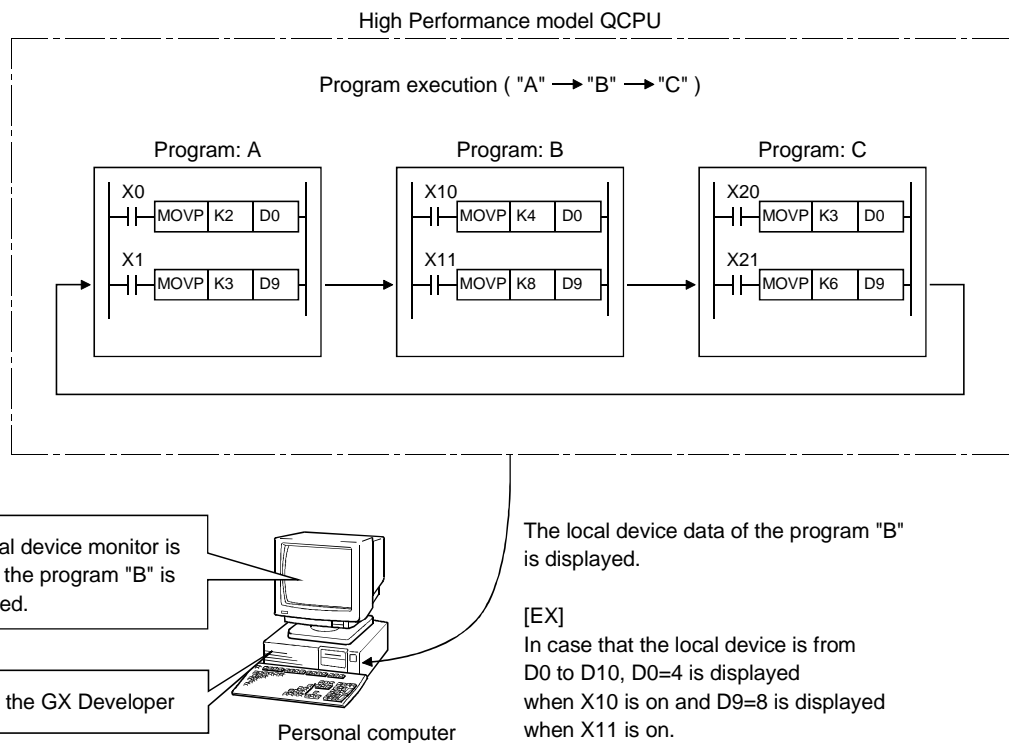
(a) Local devices specified at the "Device" tab screen in the "(PLC) Parameter" dialog box can be monitored or tested by operating from GX Developer. This function is useful when debugging a program and monitoring local devices in a program monitored by GX Developer. See Section 10.13.1 for local devices.

(b) Monitoring the Local Devices

The table below shows the status of three programs "A", "B", and "C" being executed on the High Performance model QCPU, with local devices D0 to D9 specified. It assumes that these three programs are executed in the order of A → B → C → (END processing) → A → B ....

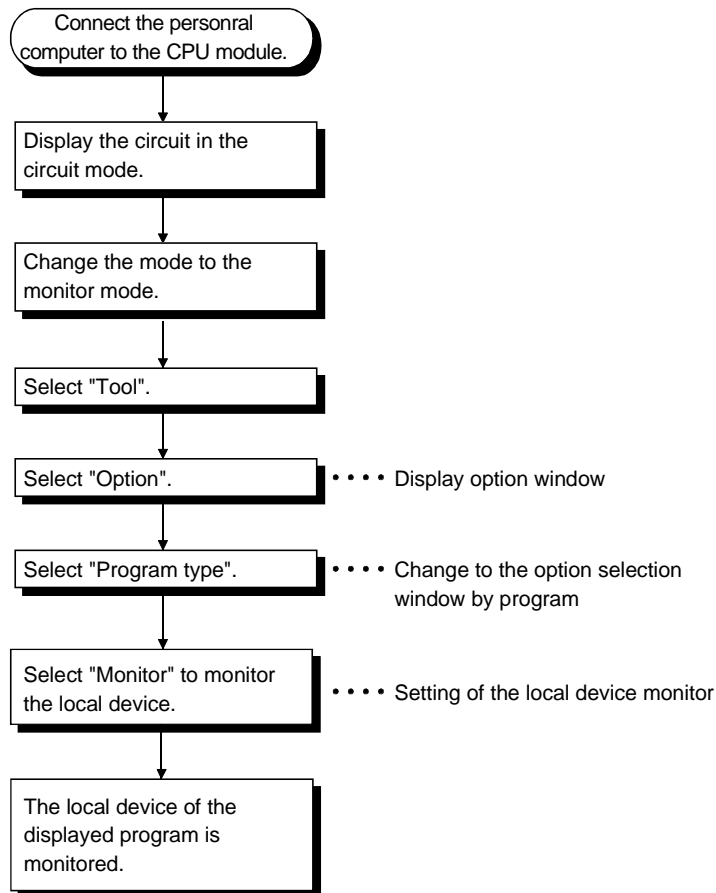
	Monitor Device	
	D0	D100
When local devices are specified	D0 in Program "C" is monitored.	D100 is monitored after Program "C" is executed.
When local devices are not specified	D0 in the displayed program is monitored.	D100 is monitored after the displayed program is executed.

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".



## (2) Monitoring the Local Devices

Monitor local devices in the following steps:



## (3) Precautions

- (a) 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) 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 High Performance model QCPU.
- (c) If local devices in a stand-by type program are monitored, scan time is extended for some time because local device data is read and saved. See Section 10.13.1 for details.
- (d) Local devices in a fix scan execution type program cannot be monitored or tested.

7.11.3 Enforced ON/OFF for external I/O

Enforced ON/OFF operations from GX Developer will forcibly switch the external I/O on and off.

The information registered for ON/OFF will be cancelled with GX Developer operations.

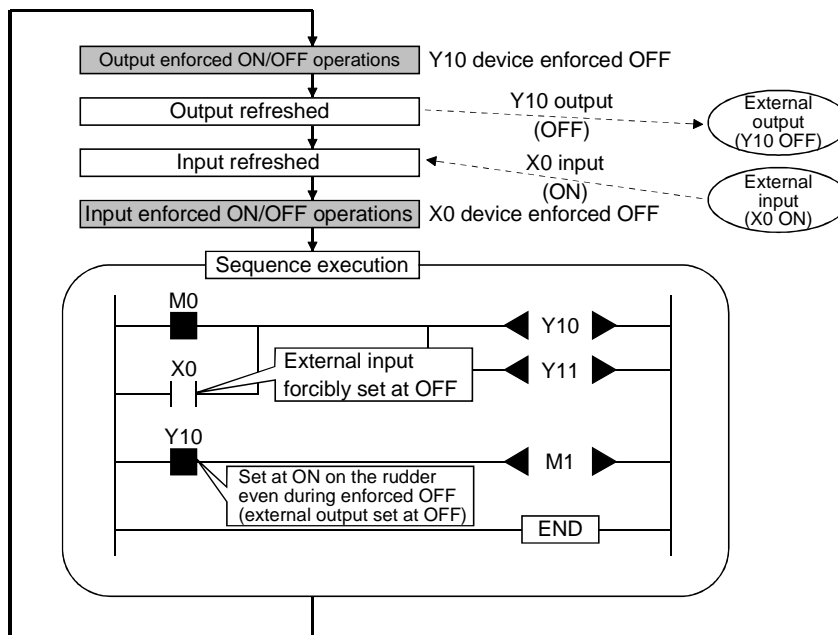
A GX Developer Version 6 or higher is required to use this function.

It is possible to perform enforced ON (enforced ON registration,) enforced OFF (enforced OFF registration) and cancel enforced ON/OFF (cancel registration) with the enforced ON/OFF function.

The operations for performing enforced ON, enforced OFF and canceling enforced ON/OFF are shown in the table below.

Operation	Input (X) operation	Output (Y) operation
During canceling (no operations)	Performs sequence program operations with external input.	Outputs the results of sequence program operations externally.
During enforced ON	Performs sequence program operations in the enforced ON status.	Outputs "ON" externally regardless of the results of sequence program operations.
During enforced OFF	Performs sequence program operations in the enforced OFF status.	Outputs "OFF" externally regardless of the results of sequence program operations.

The operations when enforced ON/OFF is performed are shown in the diagram below.



(1) Explanation of specifications

- (a) Enforced ON/OFF can be performed regardless of the High Performance model QCPU's RUN/STOP status. However, enforced ON/OFF is only allowed for input during stop errors. The output is only performed to device Y.
- (b) Devices can be registered within the ranges: input (X0 to X1FFF), output (Y0 to Y1FFF).

- (c) The input and output eligible for enforced ON/OFF are shown below.
- 1) Input (X) and output (Y) for modules mounted on the base unit.
  - 2) I/O (X/Y) of High Performance model QCPU or I/O (LX/LY) of MELSECNET/H modules to be refreshed High Performance model QCPU.
  - 3) I/O (X/Y) of High Performance model QCPU or I/O (RX/RX) of CC-Link to be refreshed High Performance model QCPU.

When enforced ON/OFF registration is performed for devices outside the above refresh ranges (ex: empty slots) only the High Performance model QCPU device memory is set at ON/OFF, and this is not output externally.

- (d) Canceling ON/OFF registration information
- 1) ON/OFF registration information can be canceled from GX Developer. Devices for which enforced ON/OFF has been performed will assume the following statuses when ON/OFF registered information has been cancelled.

Enforced ON/OFF 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 High Performance model QCPUs to be refreshed from LX of MELSECNET/H module	Assumes the refreshed ON/OFF status from MELSECNET/H.	
	Input of High Performance model QCPUs 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)	Maintains the enforced ON/OFF status.	
Output	Output from modules mounted on the base unit	Outputs the results of the sequence program operations.	OFF is output.
	Output of High Performance model QCPUs to be refreshed from LX of MELSECNET/H module	Outputs the results of the sequence program operations.	OFF is output.
	Output of High Performance model QCPUs to be refreshed from RX of CC-Link	Outputs the results of the sequence program operations.	OFF is output.
	Output other than above (outside of the refresh range)	Assumes the result of the sequence program operations.	Assumes the OFF status.

- 2) The enforced ON/OFF settings are cleared with the following operations.
  - Power supply OFF → ON
  - Reset with the High Performance model QCPU RESET/L. CLR switch
  - Reset with remote reset operations

(e) The timing for external I/O enforced ON/OFF is shown in the table below.

Refresh area	Input	Output
I/O modules on the base unit (X, Y)	<ul style="list-style-type: none"> <li>• During END processing (input refresh)</li> <li>• During the execution of commands that used direct access input (DX) (LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF)</li> </ul>	<ul style="list-style-type: none"> <li>• During END processing (output refresh)</li> <li>• During the execution of commands that used direct access output (DY) (OUT, SET, DELTA, RST, PLS, PLF, FF, LDF, MC)</li> </ul>
I/O of High Performance model QCPU to be refreshed from LX, LY of MELSECNET/H	<ul style="list-style-type: none"> <li>• During END processing (MELSECNET/H refresh)</li> <li>• During execution of the COM command</li> <li>• During execution of the ZCOM command</li> </ul>	
I/O of High Performance model QCPU to be refreshed from RX, RY of CC-Link	<ul style="list-style-type: none"> <li>• During END processing (CC-Link refresh)</li> <li>• During execution of the COM command</li> <li>• During execution of the ZCOM command</li> </ul>	

- (f) A total of thirty-two devices can be registered for enforced ON and OFF.
- (g) Sequence program operations take precedence when used with an output Y contact.
- (h) The enforced ON, OFF and cancelled status (including those that are not set up) can be confirmed with GX Developer. Confirmation is also allowed with the MODE judgment LED when at least one device is registered. (the MODE LED will flicker.)
- (i) It is possible to register enforced ON/OFF for external I/O in the same CPU module from multiple GX Developers connected to the network. However, when enforced ON/OFF is registered in the same device from multiple GX Developers, it will assume the most recent registered ON/OFF status.  
Owing to this, there are cases when the GX Developer executed first will display different ON/OFF information to the CPU module ON/OFF information. When performing enforced ON/OFF from multiple GX Developers, ensure that the most up-to-date information is set with the "Load Registration Status" switch before executing the enforced ON/OFF procedure.

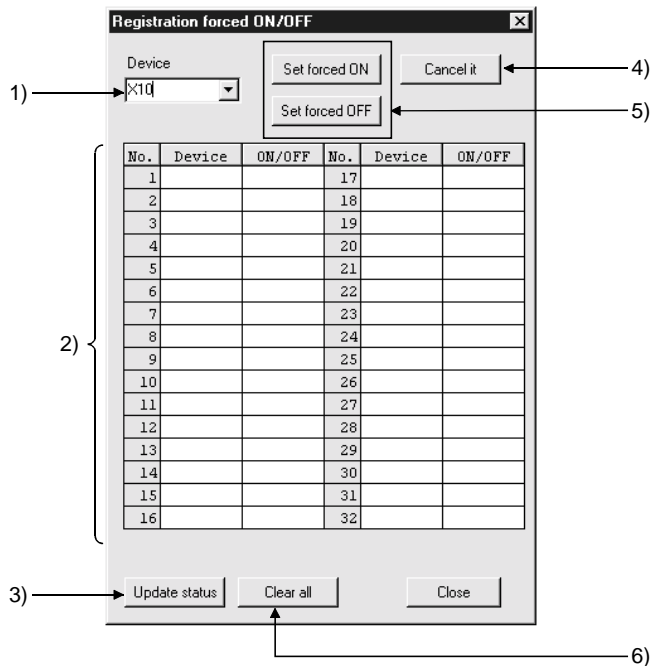
(2) Operation procedure

The operation procedure is explained below.

(a) Register enforced ON/OFF for the specified device.

[Online] → [Debug] → [Enforced I/O Registration/Cancellation]

It is possible to perform enforced ON or enforced OFF for a specified device by selecting [Enforced ON Registration] or [Enforced OFF Registration] after the device has been specified on the [Enforced I/O Registration/Cancellation] setup screen.



(b) Descriptions of the fields to set up are provided below.

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)	Load registration status	Displays the registration status loaded from CPU module.
4)	Enforced ON/OFF registration	Performed enforced ON/OFF registration for specified devices.
5)	Registration cancellation	Cancels the enforced ON/OFF for registered devices.
6)	Bulk registration cancellation	Cancels all registered enforced I/O registrations.

## 7.12 Writing in Program during High Performance model QCPU RUN

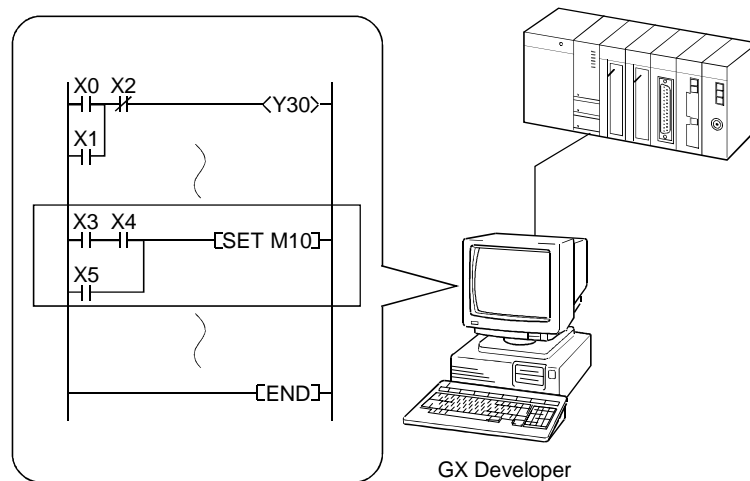
When the High Performance model QCPU is in the RUN status, you can write programs or files in any of the following steps:

- Writing data in the circuit mode during RUN (see Section 7.12.1).
- Writing data by using pointers during RUN (see Section 7.12.2).
- Writing a batch of files during RUN (see Section 7.15.2).

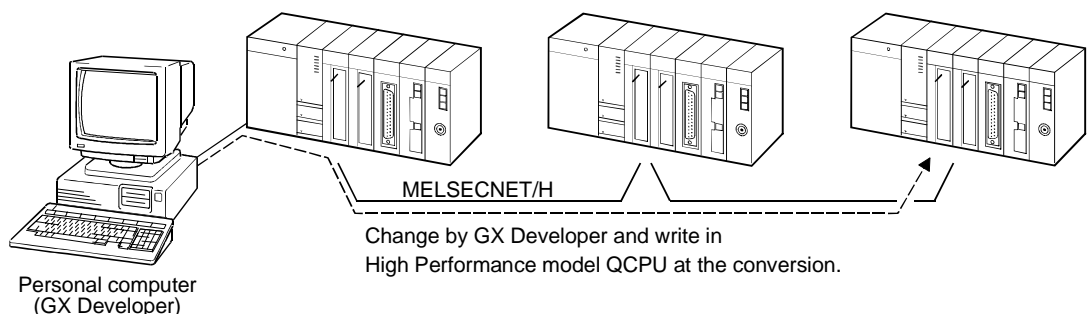
### 7.12.1 Writing data in the circuit mode during RUN

#### (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 High Performance model QCPU RUN status.
- The program can be changed without stopping the process in High Performance model QCPU program by performing writing data in the circuit mode during RUN status.



- Writing program during RUN can be performed from a GX Developer connected to another station in the network.



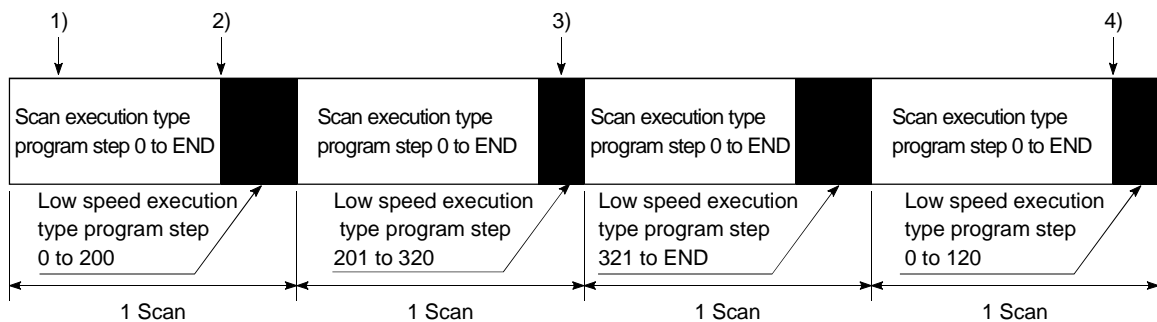
## (2) Precautions

Take a note of the following when writing during RUN:

- (a) The memory that can be written during RUN is only program memory.
- 1) If the write during RUN is performed while booting a program from a memory card (RAM), the program to be booted will be changed. While booting, it takes some time until the write during RUN is completely executed.
  - 2) If the write during RUN is performed while booting a program from the standard ROM or memory card (ROM), the program to be booted will not be changed.

Before turning off the PLC or resetting the High Performance model QCPU, write the program memory into the standard ROM or memory card (ROM).

- (b) A maximum of 512 steps can be written at once during RUN.
- (c) When a low speed execution type program is being executed, the RUN write is started once the low speed execution type program is complete. Also, the low speed execution is stopped temporarily during a RUN write.



- 1): RUN write command of the scan execution type program
- 2): RUN write execution of the 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

- (d) If the write during RUN is executed while the PLOAD, PUNLOAD or PSWAP instruction is executed, the processing will enter into a stand-by status for the write during RUN. If the write during RUN is executed while the PLOAD, PUNLOAD or PSWAP instruction is executed, the execution of the instruction is delayed until the write during RUN is executed.



- (e) The capacity of a High Performance model QCPU's program file is a sum of the capacity of the program created and steps used for the write during RUN. The write during RUN is executed when the capacity of a program file is increased. If the capacity of a program file becomes larger than what it was before, steps can be assigned for the write during RUN. This means that the write during RUN can be executed only when enough space is available in a user memory area. If steps are assigned again while the write during RUN is executed, scan time could be extended for a value shown below in the table. Controls are interrupted for a value shown below in the table.

CPU Type	Step for Write During RUN	
	If Not Changed	If Assigned Again
QnCPU	max. 2 ms	max. 30 ms.
QnHCPU	max. 1 ms	max. 90 ms

- (f) High Performance model QCPU does not work correctly, if the following instructions are written during RUN write.
- 1) Trailing edge instruction
 

If the execution conditions of the following trailing edge instructions are not arranged upon completion of writing, the trailing edge instruction is executed.

    - LDF
    - ANDF
    - ORF
    - MEF
    - PLF
  - 2) Leading edge instruction
 

If the execution conditions for leading edge instructions (PLS instruction and  $\square$ P instruction) are arranged upon completion of writing, leading edge instruction is not executed.

The leading edge instruction is executed when the execution conditions are OFF then ON.
  - 3) SCJ instruction
 

If the execution conditions of the SCJ instruction are arranged upon completion of writing, a jump to the designated pointer occurs even in a scan cycle.

7.12.2 Writing a batch of files during RUN

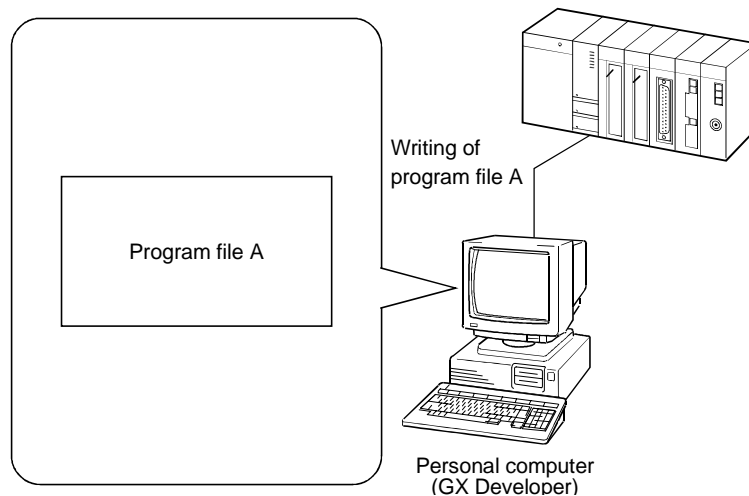
(1) File-Write During RUN function

(a) The file-write during RUN function is used to write a batch of files to the High Performance model QCPU as shown below in the table.

Memory Name	High Performance model QCPU Built-in			Memory Card (RAM)	Memory Card (ROM)	
	Program Memory	Standard RAM	Standard ROM	SRAM Card	Flash Card	ATA Card
Parameter	×	×	×	×	×	×
Intelligent function module parameter	×	×	×	×	×	×
Program	○	×	×	○	×	○
Device comment	○	×	×	△	×	△
Device initial value	×	×	×	×	×	×
File register	×	△	×	△	×	×
Local device	×	×	×	×	×	×
Debug data	×	×	×	×	×	×
Failure history data	×	×	×	×	×	×
PLC user data	×	×	×	×	×	×

○ : Writable data , × : Unwritable data

△ : Writable data if access is not being made in sequence program



<b>POINT</b>
<p>The file-write during RUN allows writing three types of files:</p> <ul style="list-style-type: none"> <li>• Program: program memory, SRAM card, ATA card</li> <li>• Device comment: program memory, SRAM card, ATA card</li> <li>• File register: standard RAM, SRAM card</li> </ul> <p>Any other files cannot be written while the High Performance model QCPU is in the RUN status.</p>

## (2) Precautions

The precautions for file-write during RUN are as follows.

- (a) The file-write during RUN can be executed when any of the following conditions is met.
- 1) Program memory
    - When continuous space is available.
    - When space is available.
  - 2) Memory card
    - When space is available.
- (b) Please note that scan time could be extended as shown below in the table if the file-write during RUN is executed. Controls are stopped for some time as specified by a value in the table.

Event	QnCPU	QnHCPU
When continuous space is available in a program memory	max. 80 ms	max. 300 ms
When space is available in a program memory	max. 80 ms	max. 300 ms
When space is available in a memory card (except ATA card)	max. 120 ms	max. 570 ms

Please note that scan time is extended for 1.25 seconds at 30 k step when an ATA card is in use.

- (c) Please note that no access can be made from an instruction in a sequence program while a batch of files is written, with the High Performance model QCPU in the RUN status. While the file-write during RUN is being executed, an instruction to make access to a file is not executed.
- (d) Do not execute file write during RUN simultaneously from multiple locations. Doing so may erase program file contents.
- (e) If a sequence program file being executed is written when the High Performance model QCPU is in the RUN status, the following will not work properly. After the write is complete, a fall instruction is executed only when its execution condition is OFF.
- LDF
  - ANDF
  - ORF
  - MEF
  - PLF
- (f) If batch write of SFC program files during RUN is executed, an initial start is made after execution.

POINT
<p>The following CPU module and GX Developer versions are required to execute batch write of SFC program files during RUN.</p> <ul style="list-style-type: none"> <li>• CPU module whose serial No. is "04122" or later</li> <li>• GX Developer Version 8 or later</li> </ul>

7.13 Execution Time Measurement

This is a function to display the processing time of the program being executed.

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. The details of each function are indicated in sections 7.13.1 to 7.13.3.

- Program monitor list
- Interrupt program monitor list
- Scan time measurement

7.13.1 Program monitor list

(1) What is Program Monitor List?

- (a) This is a function to display the processing time of the program being executed.
- (b) The scan time, number of times executed, and processing time by item can be displayed for each program.

(2) Using the Program Monitor List

- (a) Choose "Online" → "Monitor" → "Program monitor list". "Program Monitor List" dialog box appears on screen.
- (b) The following shows an example of executing program monitor list.

The screenshot shows the 'Program monitor list' dialog box with the following data:

**Total scan time**

	Monitor time(ms)	Sum of scan time(ms)
Scan	200	2.000
Initial		2.000
Low speed		0.100

**Scan execution part, detailed scan time**

	Program(ms)	
	0.100	
END operation time(ms)	0.500	
Low speed program(ms)	1.100	
Constant waiting(ms)	0.300	

**Each program execution status**

	Program	Execute	Scan time(ms)	Execute count
1	MAIN	Scan	0.000	24995
2	MAIN1	Scan	0.000	24995
3	MAIN5	Initial	0.000	1
4	MAIN4	Low speed	0.000	47167
5	MAIN3	Fixed Scan	.....	0
6	MAIN2	Wait	0.000	0
7				
8				
9				
10				
11				

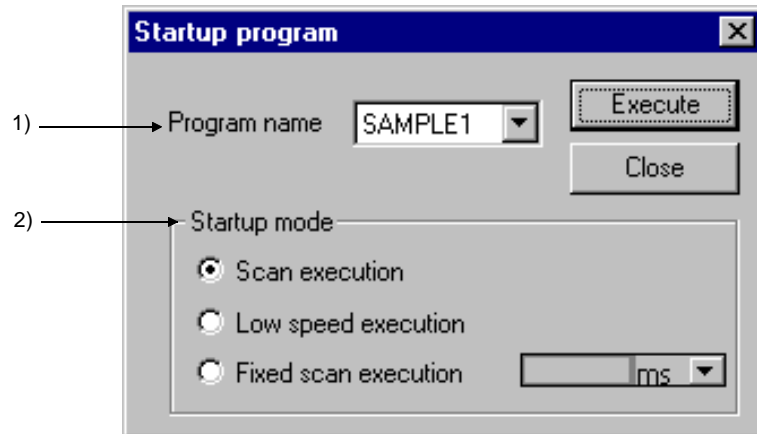
Buttons at the bottom: Start monitor, Stop monitor, Startup program, Stop program, Close.

- a) "Total Scan Time"  
The monitor time set in WDT(the watch dog timer) of "PLC RAS" tab screen in the (PLC) "Parameter" dialog box and total scan time for each program type are displayed.
- "Monitor Time"  
The monitoring time for the scan execution type program, initialization program, and low speed execution type program are displayed.  
If the scan time exceeds this time, the High Performance model QCPU displays the watch dog timer error.
  - "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 Time Details for Scan Execution"  
The details of the scan time are displayed.
- "Program"  
The total execution time of the scan execution type program is displayed.
  - "END operation time"  
The END operation time is displayed.
  - "Low speed program"  
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.
  - "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.
- c) "Each Program Execution Status"  
The execution status of program specified at the "Program" tab screen in the (PLC) "Parameter" dialog box is displayed.
- "Program"  
The program name is displayed in the order set in the parameter.
  - "Execute"  
The program type set in the parameter is displayed.
  - "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.
  - "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.

(3) Program can be started and stopped on the program list monitor screen.

(a) Startup program button

Clicking the startup program button displays the following dialog box.



1) Program name

Only the program, that is set at the "Program" tab screen in "(PLC) Parameter" dialog box, can be selected. It is not allowed to enter a program name freely.

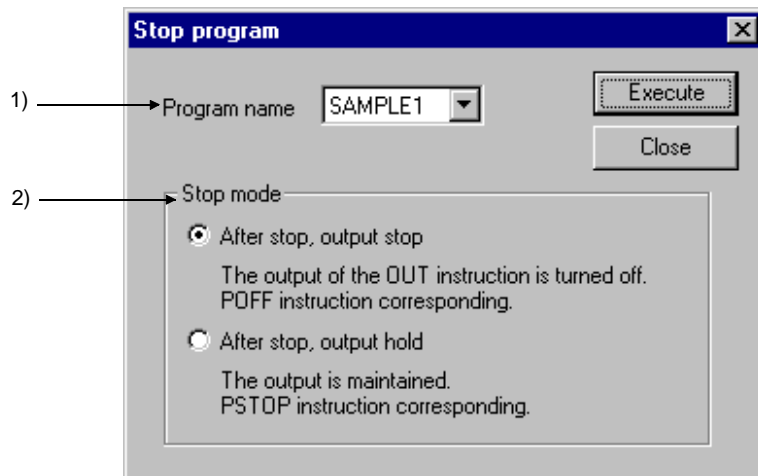
2) Startup mode

A stand-by type program for "Scan execution", "Low speed execution" or "Fixed scan execution" can be set.

Startup mode defaults to the value that was set by choosing [PLC Parameter] - <Program>. ms or s can be selected as the unit.

## (b) Stop program button

Clicking the stop program button displays the following dialog box.



## 1) Program name

Only the program, that is set at the <Program> tab in the "(PLC Parameter)" dialog box, can be selected.  
It is not allowed to enter a program name freely.

## 2) 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.)
- 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. (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".

**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)

## (4) 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.

## 7.13.2 Interrupt program monitor list

## (1) What is Interrupt Program Monitor List?

- (a) This function displays execution count of the interrupt program (I0 to I255).
- (b) 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.

The following shows an execution example of the interrupt program monitor list:

Cut in pointer	Execute count	Common comment
I28	2512	100ms
I29	6280	40ms
I30	12560	20ms
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	

a)

b)

## a) "Execute count"

The number of times the interrupt program was executed is displayed. This function starts counting the number when High Performance model QCPU is in RUN status (When the number reaches 65536 times, it is reset to 0).

## b) "Common Comment"

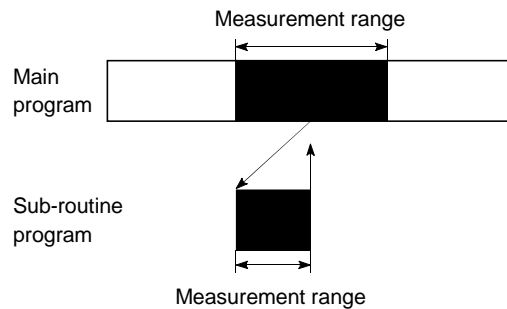
This indicates device comments created on interrupt points (I0 to I255).



## 7.13.3 Scan time measurement

## (1) What is Scan Time Measurement?

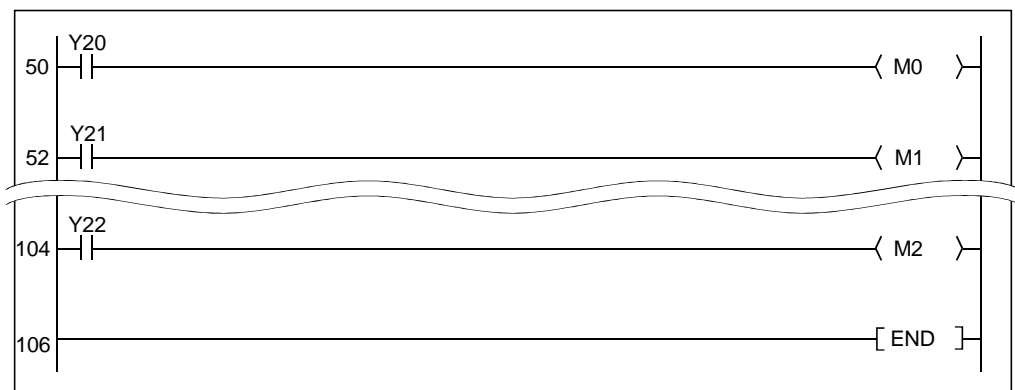
- (a) This function displays the set program interval processing time.
- (b) To specify a scan time measurement range, follow either of the following two steps:
  - Make the setting on the "Ladder monitor" window.
  - Make the setting on the "Scan Time Measurement" dialog box.
- (c) The time for the subroutines and interrupt program can be measured as well.
- (d) The time includes the time required for processing sub-routines, when the sub-routine CALL command is within the range of scan time measurement. The amount of time required for executing interruption programs and fixed scan execution type programs is all added to this.



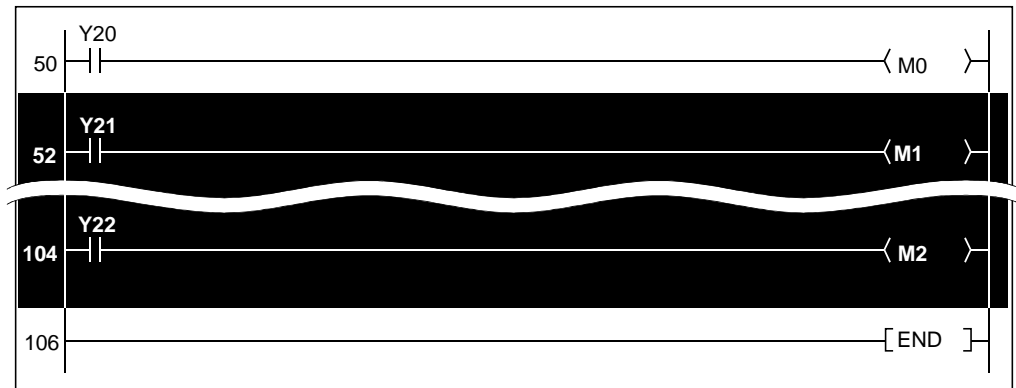
## (2) Measuring Scan Time

To measure scan time, follow the following steps.

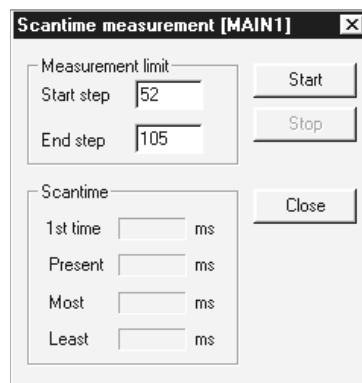
- (a) Display the leading edge of the circuit of which scan time to be measured, and set the monitor mode.



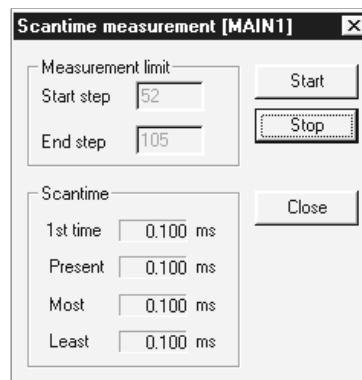
- (b) The scan time measurement range is specified.  
(The specified area is highlighted.)



- (c) Choose "Online" → "Monitor" → "Scantime measurement" to open "Scan time Measurement" dialog box.



- (d) Click on the "Start" button.



### (3) Precautions

- Set the "Measurement limit" so that the value of "Start step" is larger than that of "End step".
- The scan time to skip to another program file cannot be measured.
- If the measurement time is less than 0.100 ms, 0.000 ms is displayed.
- If a measurement range is specified between the FOR instruction and the NEXT instruction, scan time will show the execution time of making a measurement in the measurement range between specified steps.

7.14 Sampling Trace Function

POINT
<p>(1) The SRAM card (Q2MEM-1MBS, Q2MEM-2MBS) is required to store the trace data and trace results. After mounting the SRAM card to the High Performance model QCPU, execute sampling trace.</p> <p>(2) Sampling trace is not executed if the Flash card (Q2MEM-2MBF, Q2MEM-4MBF) or ATA card (Q2MEM-8MBA, Q2MEM-16MBA, Q2MEM-32MBA) is installed, because the cards cannot store the trace data and trace results.</p>

(1) What is Sampling Trace Function?

- (a) This function samples the device continuously on the High Performance model QCPU at specified timings.
- (b) 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.
- (c) 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 in the memory card.
- (d) The sampling trace file stores the trace condition data and trace execution data necessary to perform the sampling trace. When trace is started by using GX Developer, the trace is performed as many times as specified.  
The sampling trace area is 60 kbyte.

The number of traces can be obtained by dividing 60 kbyte by the number of bytes specified as a device. The expression is:  $(\text{Number of Bit Devices}) / 8 + 2 \times (\text{Number of Word Devices})^{*1}$

\*1: Round up result of "(number of Bit Devices)/8" in the expression to the right of the decimal point.

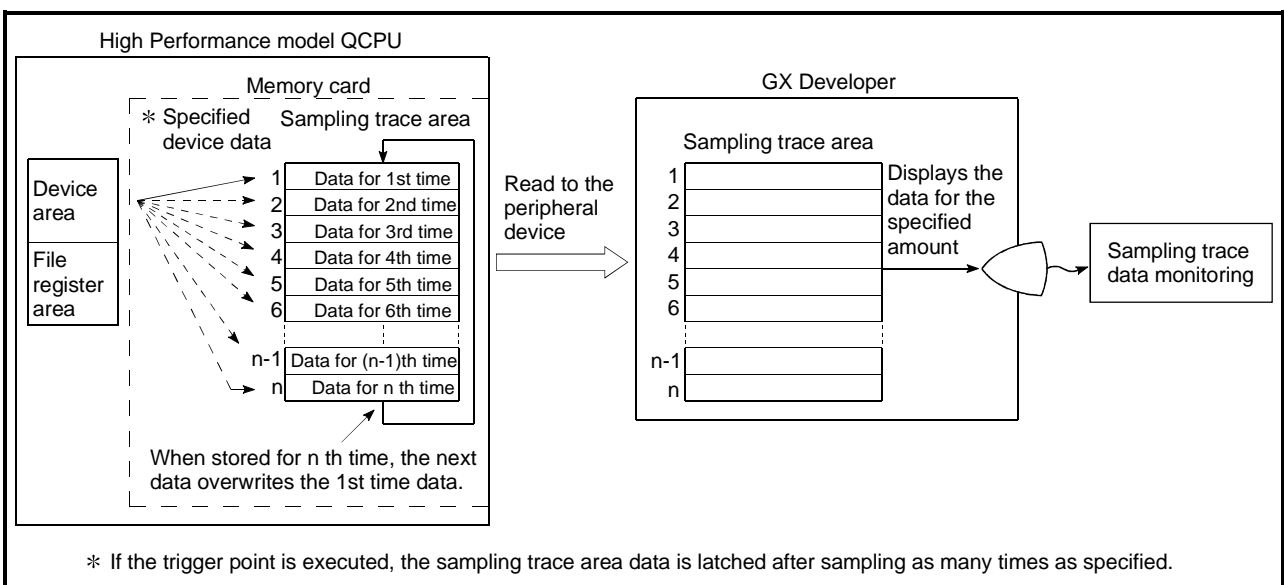
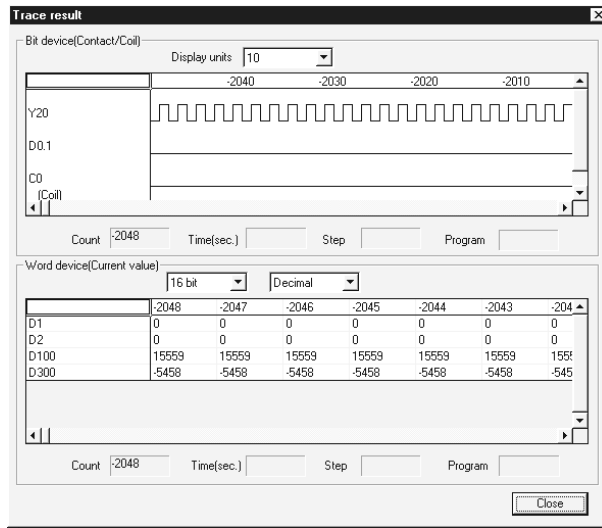


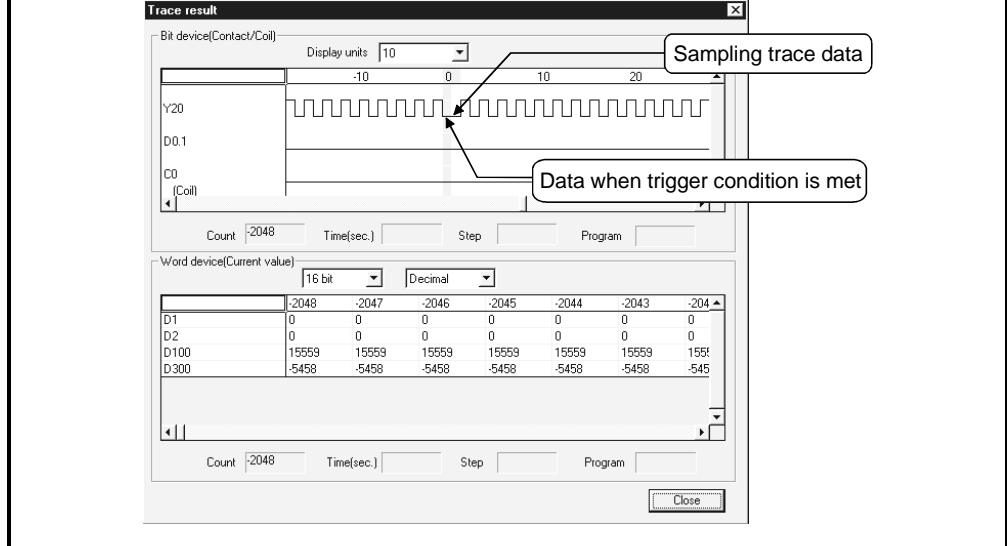
Fig. 7.8 Sampling Trace Operation

- (e) The trace result displays the ON/OFF status of the bit device for the sampling cycle, and the current value of the word device.

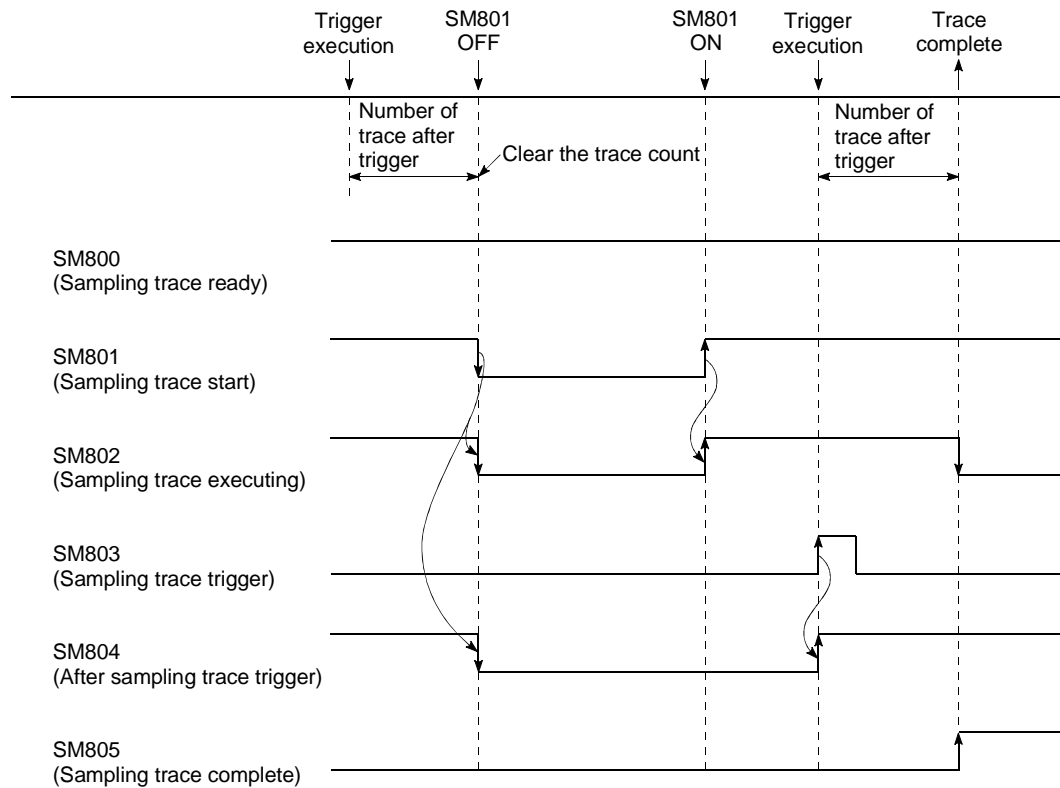


**POINT**

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.



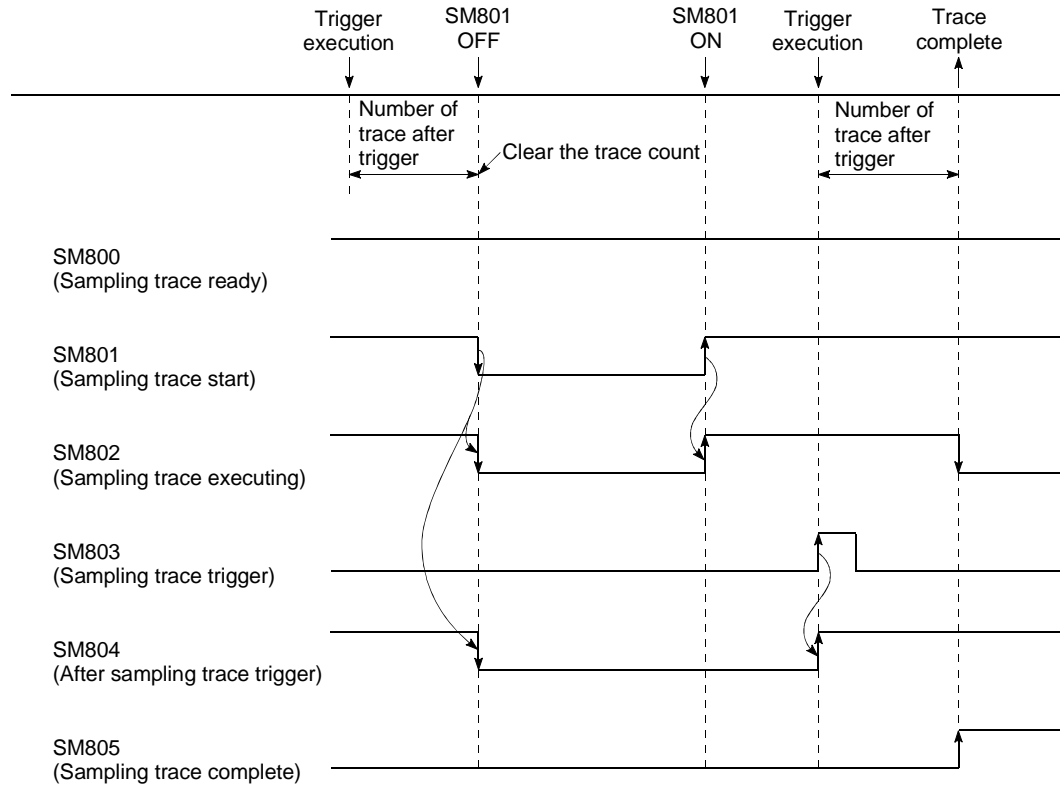
- (f) The execution status of the sampling trace function is stored in the special relay (SM800, SM802, SM804 and SM805).  
 If an error occurs while the sampling trace function is used, SM826 turns on. By using special relays in a sequence program, the execution status of the sampling trace function can be checked.
- 1) After the "trace data" and "trace conditions" set using GX Developer are written to the High Performance model QCPU, SM800 (sampling trace ready) turns on. SM800 indicates whether the sampling trace can be executed or not.
  - 2) 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.
  - 3) 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.
  - 4) After the sampling trace is completed, SM805 (sampling trace complete) is turned on.



When trace is interrupted from GX Developer, the SM800 is also turned off.

(g) Trace interrupt

- 1) When SM801 (sampling trace start) is turned off during sampling trace, the sampling trace is interrupted. In the meantime, the number of traces is cleared.
- 2) When turning on SM801 again, trace is restarted.



\* When trace is interrupted from GX Developer, the SM800 is also turned off.

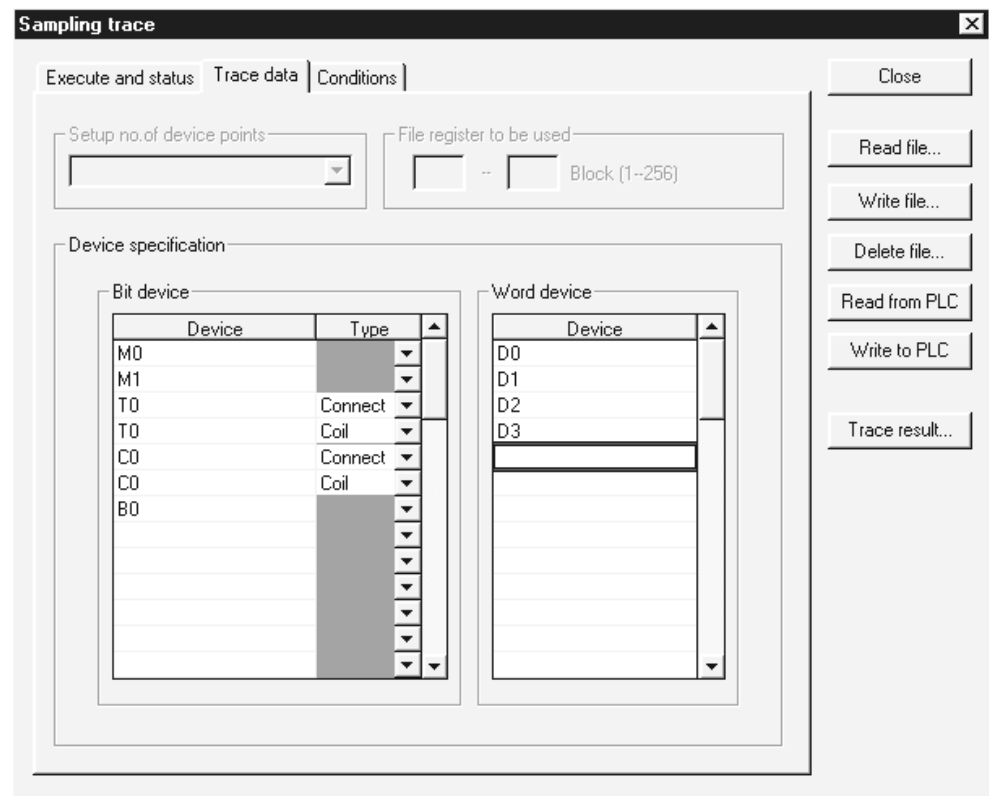
## (2) Operation Procedure

The sampling trace operation is performed in the following procedures:

Each operation is performed on the "Sampling trace" dialog box within the online mode trace menu.

## (a) Trace Device Setting

Set the device to perform sampling trace at the "Trace data" tab screen in the "Sampling trace" dialog box.



## 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

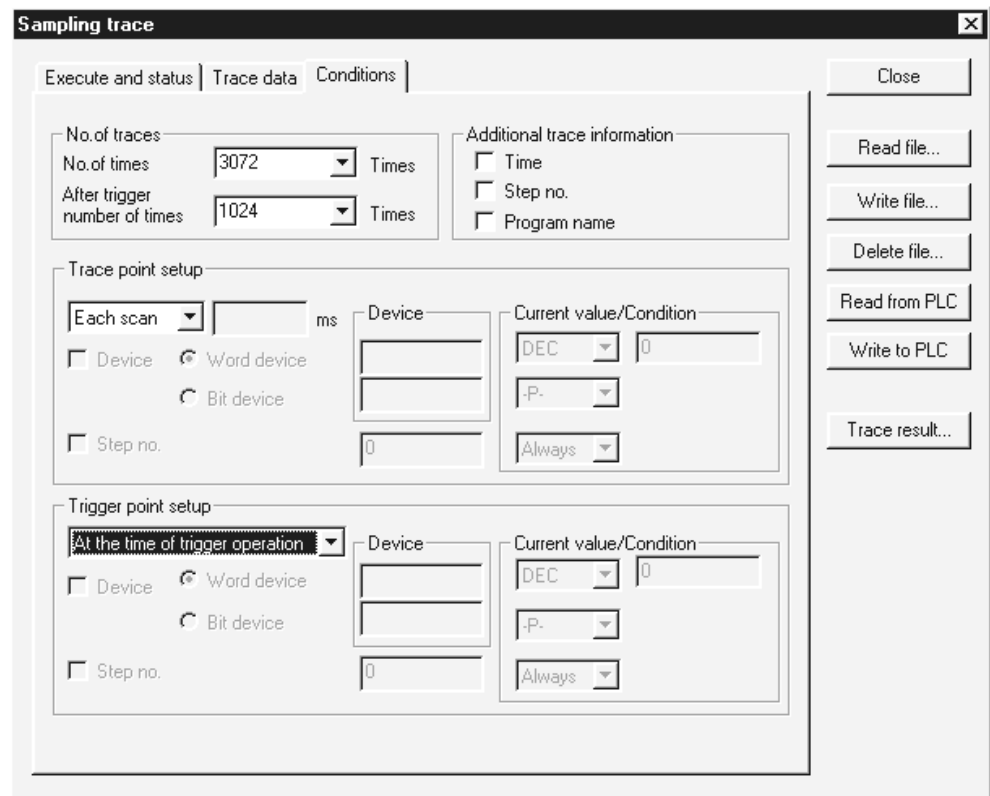
Maximum of 50 word devices can be set as follows.

- T(current value), ST(current value), C(current value), D, SD, W, SW, R, Z, ZR,
- U□\G, J□\W, J□\SW

(b) Setting the Trace Condition

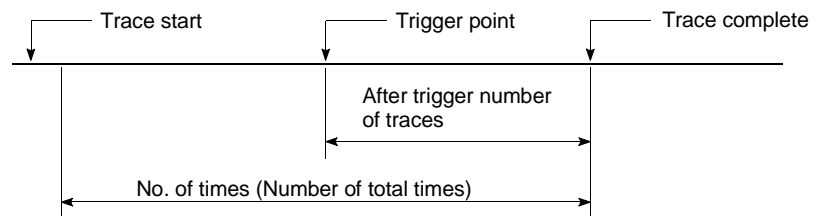
Set the trace condition at the "Conditions" tab screen in the "Sampling trace" dialog box.

The trace condition setting can set to "No. of traces", "Trace point setup", "Trigger point setup", and "Additional trace information".



1) No. of traces

- a) The "No. of times" sets the number of times to execute the sampling trace from trace execution to trace complete.
- b) The "After trigger number of times" sets the number of times to executes the sampling trace from trigger execution to trace complete.



- c) 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:

## a) Each Scan

Collects trace data for every scan (END processing).

## b) Interval

Collects trace data at specified times.

## c) Detailed

Sets the device and step no. The setting method and trace data sampling timing is the same as mentioned in section 7.11.1., (when setting the monitor condition.)

The devices that can be set in the detailed condition are as follows:

- Bit Device : X, Y, M, L, F, SM, V, B, SB, T(contact), ST(contact), C(contact), J  \X, J  \Y, J  \B, J  \SB, BL  \S
- Word Device : T(current value), ST(current value), C(current value), D, SD, W, SW, R, Z, ZR, U  \G, J  \W, J  \SW

The following attributes can be set for the above devices:

- Bit device number of digits specification
- Word device bit number specification

## 3) Trigger Point setup

This sets the point to execute the trigger. Select one from the following:

## a) At the time of TRACE order:

The time of execution of TRACE instruction is set as the trigger.

## b) At the time of trigger operation:

The trigger operation from GX Developer device is set as the trigger.

## c) Detailed setting

The device and step number is set. The setting method and trigger execution timing is the same as mentioned in Section 7.11.1., (the monitor condition setting.)

## 4) Additional trace 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):

## a) Time

Stores the time when the trace was executed.

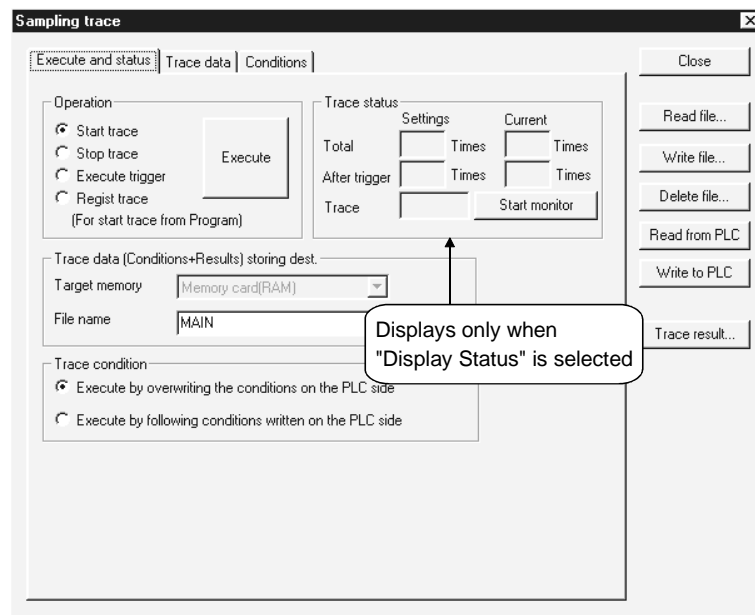
## b) Step No.

Stores the step number when the trace was executed.

## c) Program Name

Stores the program name that executed the trace.

- (c) The created trace data and trace condition is written to the memory card.  
 The trace file is written to the memory card (SRAM card).  
 The trace file is written to the memory card (SRAM card) from the "Write to PLC" dialog box in the "Sampling Trace" dialog box.  
 The files are written in the memory card (SRAM card) with file names, so multiple trace files can be stored.
- (d) Sampling trace is executed.  
 The sampling trace is executed at the "Execute and status" tab screen in the "Sampling Trace" dialog box.



"Operation", "Trace data (Condition + Results) storing dest", and "Trace condition" can be set when "Execute and Status" is displayed.

- 1) At the "Operation" section, select one of the following:
  - Start trace  
The trace is started. Starts to count the trace count.
  - Stop trace  
The trace is interrupted. The total trace count and trace count after trigger are cleared. (When restarting trace, select "Start trace" again.)
  - Execute trigger  
Starts to count the trace count after trigger. The trace is complete when the trace count reaches the specified trace count after trigger.
  - Regist trace  
Registers trace data when a program is executed.
- 2) In the "Trace data (Conditions+Results) storing dest." section, specify a file name of a file in which to store trace data and trace conditions.  
 Trace results are also stored in a selected file with a specified file name.

- 3) At the "Trace Condition" section, select one of the following:
  - Execute by overwriting the conditions on PLC side.  
Overwrites trace condition to the existing trace file.
  - Execute by following conditions written on PLC side  
Executes the program with the conditions in the trace file specified in "Trace Data (Condition + Results) storing dest".
- (e) Read the trace results from the High Performance model QCPU and display the data.
  - 1) "Read from PLC" reads the trace result from the High Performance model QCPU.
  - 2) The read trace results appears in "Trace results" display.

<b>POINT</b>
Sampling trace is executed only once. When re-executing, execute the TRACE instruction, and reset the sampling trace.

### (3) Precautions

- (a) SRAM card is required for sampling trace.  
Set the sampling trace file in the memory card (SRAM).
- (b) 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 High Performance model QCPU.
- (c) The trace information (trace file) registered in the High Performance model QCPU is registered in the SRAM card and latched. As the condition data is stored in the trace file, even if the power is off or the High Performance model QCPU is reset, the sampling trace can be executed under registered trace conditions.  
At power on/reset of High Performance model QCPU, latched trace information is cleared in the cases where:
  - The SRAM card registered in a trace file is not inserted
  - The trace file is corrupted
 This requires registering trace information once again by operating from GX Developer.  
To clear data, perform the latch-clear operation with the RESET/L.CLR switch.  
To perform sampling trace again after latch clear, execute sampling trace after selecting "Regist trace".
- (d) This is performed by connecting the High Performance model QCPU and GX Developer.
- (e) While in STOP status, the High Performance model QCPU cannot read sampling trace results. To enable the High Performance model QCPU to read the sampling trace results, enter the High Performance model QCPU into RUN status.
- (f) When executing the sampling trace, ensure that trigger conditions cannot be satisfied at trigger points. If the trigger conditions are met when executing the sampling trace, they will not be recognized as trigger conditions.

## 7.15 Debug Function with Multiple Users

## (1) What is Debug Function with Multiple Users?

- (a) This function performs debugging from multiple GX Developer connected to High Performance model QCPU or Serial communication module at the same time.
- (b) If debugging tasks are classified by process or by function, this function is used to perform debugging of different files from multiple GX Developer at once.

## (2) Function Description

The debug function combination for multiple users are as follows:

Functions to be executed later Functions being executed	Monitor	Write during RUN	Execution time measurement	Sampling trace
Monitor	○	×	○	○
Write during RUN	×	×	×	×
Execution time measurement	○	×	×	○
Sampling trace	○	×	○	×

○ : Can be performed at the same time. (However, the detailed condition can only be set from one GX Developer. In this case, the detailed condition setting cannot be performed from another GX Developer.)

× : Can only be performed from one GX Developer. (This function cannot be performed by the GX Developer while it is being executed by another GX Developer.)

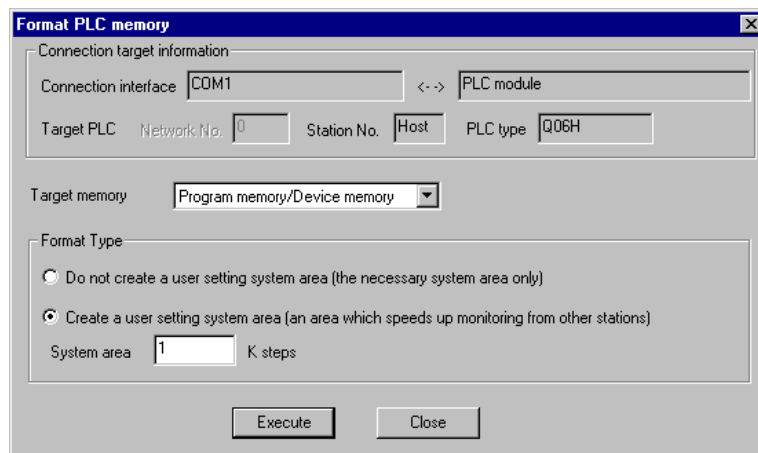
## 7.15.1 Multiple-user monitoring function

## (1) What is Multiple-User Monitoring Function?

- (a) The multiple-user monitoring operation can be performed by operating from multiple GX Developers connected to the High Performance model QCPU or the Serial communications module.
- (b) 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

- (a) For multi-user monitoring operation, create a user-defined system file in the following steps.
  - 1) Choose "Online" → "Format PLC Memory" to open the "Format PLC Memory" dialog box.
  - 2) Select "program memory" from the "Target Memory list box".
  - 3) At the "Format Type" section, select "Create a user setting system area..." so that its radio button is checked.
  - 4) Specify the desired k steps in the "System Area" text box.
- (b) The following figure illustrates an example in which "1k step" is specified in the "System Area" text box.



- 1) A maximum of 15 k steps can be set in 1 k step modules as a system area. Only 1 k step can correspond to one station monitor file. Therefore, a maximum of 15 station monitor files can be set.

## (3) Precautions

- (a) The detailed condition setting of the monitor can only be set from one area.
- (b) Monitoring can be performed even if a station monitor file is not set, but high speed monitoring cannot be performed. 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) When a user-defined system area of 15k steps is created, simultaneous monitoring of a single CPU from 16 areas can be speeded up.

## 7.15.2 Multiple-user RUN write function

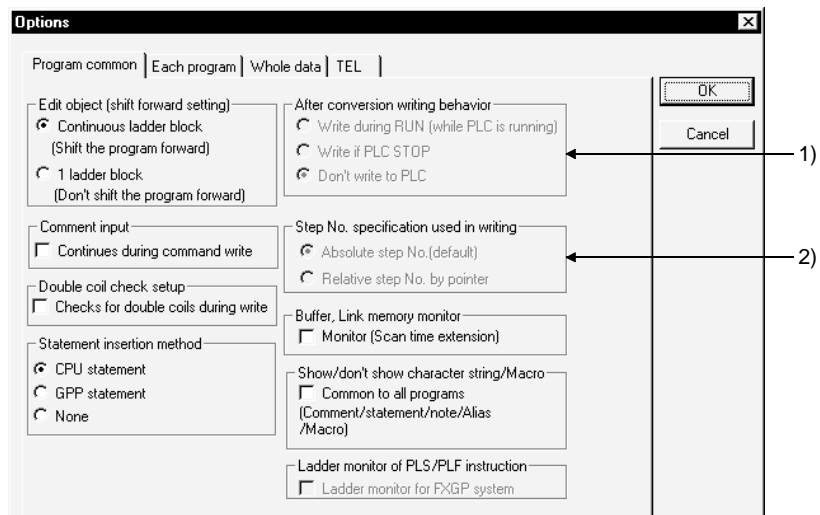
## (1) What is Multiple-User RUN Write Function?

- (a) Multiple users can write to one file or different files during RUN.
- (b) To enable multiple users to write to a single file at the same time during RUN operation, specify the desired pointer for the "write during RUN" in advance, and then select "Relative step No. by pointer" so that its radio button is checked.

## (2) Operation Procedure

The multiple-user RUN write operation is performed in the following procedures:

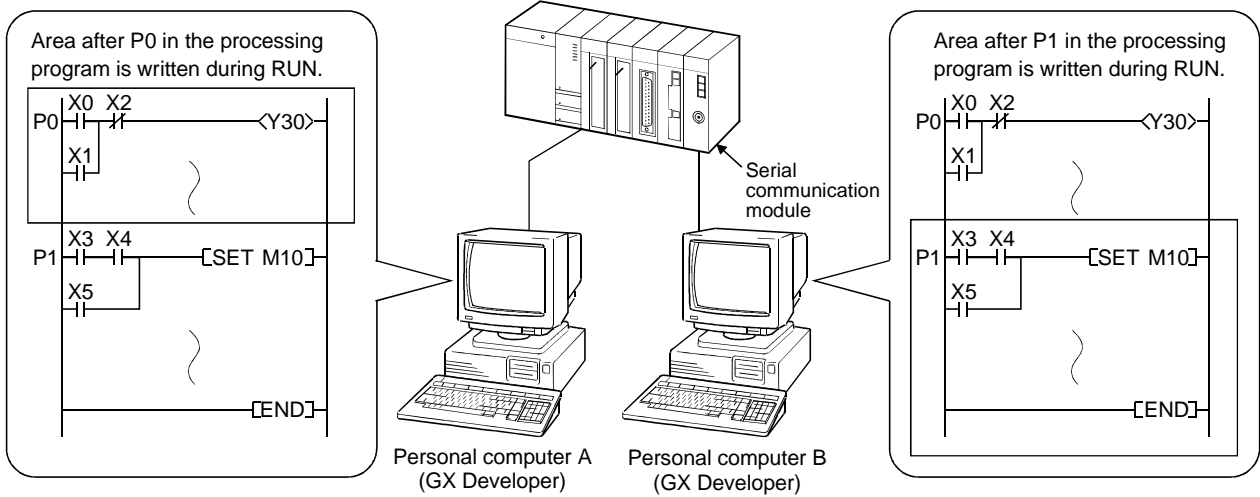
- (a) Select Tool from Option menu and set "After conversion writing behavior" and "Step No. specification used in writing".



- 1) Set "Write during RUN (while PLC is running)" in "After conversion writing behavior".
- 2) Select "Absolute step No. (default)" or "Relative step No. by pointer" in "Step No. specification used in writing".

- (b) The specified circuit of the pointer is displayed to write the circuit after conversion during RUN.

The following is an example of GX Developer A writing during RUN from P0 and GX Developer B writing during RUN from P1. The program area surrounded with  is the area to be written during RUN.



(3) Precautions

Precautions on "write during RUN" is the same as precautions on "write during RUN in the circuit mode" in Section 7.12.1. For further information, see Section 7.12.1.

## 7.16 Watch dog Timer (WDT)

## (1) What is Watch dog Timer (WDT)?

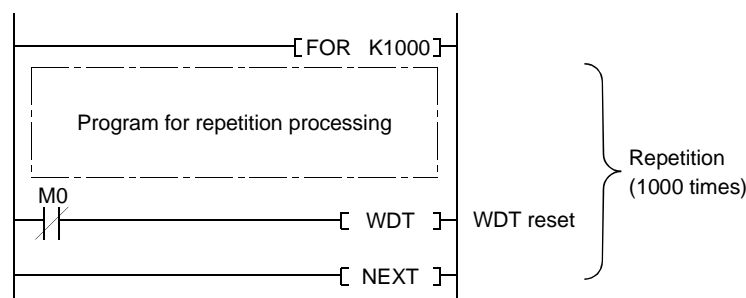
- (a) The watch dog timer is an internal sequence timer to detect High Performance model QCPU hardware and sequence program error.
- (b) When the watch dog timer expires, a watch dog timer error occurs. The High Performance model QCPU responds to the watch dog timer error as follows:
  - 1) The High Performance model QCPU 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 5001 (WDT ERROR) is stored in SD0.
- (c) The default value of the watch dog timer is 200 ms. The setting range is 10 to 2000 ms (in 10ms units).

## (2) Watch dog Timer Setting and Reset

- (a) The watch dog timer setting can be changed at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box.
- (b) High Performance model QCPU resets the watch dog timer during the END processing.
  - 1) When the END/FEND instruction is executed within the set value of the watch dog timer in the sequence program and the High Performance model QCPU is operating correctly, the watch dog timer does not time out.
  - 2) When the scan time of a sequence program is extended due to the High Performance model QCPU hardware error or execution of interrupt program/fixed scan execution type program, and END/FEND instruction cannot be executed within the set watch dog timer value, the watch dog timer times out.

## (3) Precautions

- (a) An error of 0 to 10 ms occurs in the measurement time of the watch dog timer. Set the watch dog timer for a desired value by taking such an error into account.
- (b) The watch dog timer is reset with the WDT instruction in the sequence program. If the watch dog timer expires while the FOR and NEXT instructions are repetitiously executed, reset the watch dog time with the WDT instruction.





- (c) The scan time value is not reset even if the watch dog timer is reset in the sequence program.

The scan time value is measured to the END instruction.

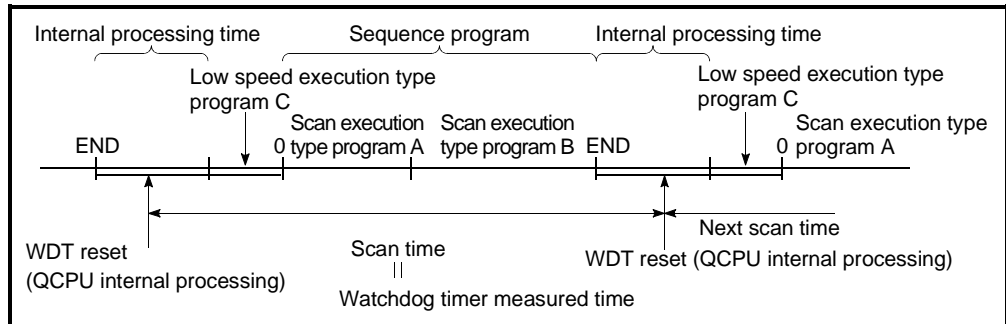


Fig. 7.9 Watch dog Timer Reset

#### REMARK

- 1) Scan time is the time from when the High Performance model QCPU 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 the same at every scan, and differs, depending on
  - Whether the commands used are executed or not executed.
  - Whether to execute or not an interrupt program and a fixed scan execution type program.
- 2) To execute at the same scan time at every scan, use the constant scan function.  
For details of the constant scan function, see section 7.2.

### 7.17 Self-Diagnosis Function

- (1) What is Self-Diagnosis Function
  - (a) The self-diagnosis is a function performed by the High Performance model QCPU itself to diagnose whether there is an error in the High Performance model QCPU.
  - (b) The self-diagnosis function is used to prevent the High Performance model QCPU erroneous operation as well as preventive maintenance.  
The self-diagnosis processing detects and displays the error when an error occurs at the High Performance model QCPU power on or during High Performance model QCPU RUN mode. It also stops High Performance model QCPU calculations.
- (2) Processing for Error Detection
  - (a) When the High Performance model QCPU detects an error, it turns on ERR. LEDs. When an error is detected, special relays (SM0, SM1) are turned ON and an error code of the error is stored in 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.
  - (b) The High Performance model QCPU stores 16 latest error codes. (See Section 7.18.)  
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.
- (3) High Performance model QCPU operation at the time of error detection
  - (a) When an error is detected from the self-diagnosis, there are two types of modes that the High Performance model QCPU operation can change to.
    - 1) High Performance model QCPU calculation stop mode  
The calculation is stopped on detecting an error, and all outputs of modules are set to OFF if the outputs for the modules have been set to "Clear (Default)" at the "Error time output mode" section in the "I/O assignment" tab screen within the "(PLC) Parameter" dialog box. And output (Y) is held.  
(The outputs (Y) on the device memory are retained.)  
However, the outputs of the modules are held if they have been set to "Hold" there. The output (Y) is also held.
    - 2) High Performance model QCPU calculation continue mode  
When an error is detected, the program (Instruction) area where the error occurred is skipped and the rest of the program is executed.  
(The outputs (Y) on the device memory are retained.)
  - (b) The calculation "continue/stop" can be set at the "PLC RAS" tab screen in the (PLC) "Parameter" dialog box if the following error occur.  
(All parameter defaults are set at "Stop".)
    - 1) Computation error
    - 2) Expanded Command error
    - 3) Fuse blown
    - 4) I/O module comparison error
    - 5) Intelligent module program execution error
    - 6) Memory card access error
    - 7) Memory card operation error
    - 8) External power supply OFFFor example, when the I/O module verification error is set to "continues", the calculations are continued in the I/O address before the error occurred.

## (4) Error check selection

The following error checking can be set to "yes/no" at the "PLC RAS" tab screen in the "(PLC) Parameter" dialog box.

(All parameter defaults are set at "Yes".)

- (a) Battery check
- (b) Fuse blown check
- (c) I/O unit comparison

## Self-Diagnosis List

	Diagnosis description	Error message	Diagnostic timing
Hardware failure	CPU error	MAIN CPU DOWN	• Always
	END instruction not executed	END NOT EXECUTE	• When the END instruction is executed
	RAM check	RAM ERROR	• When the power is turned on/when reset
	Calculation circuit check	OPE.CIRCUIT ERR.	• When the power is turned on/when reset
	Fuse blown (default... stop) * 1	FUSE BRAKE OFF	• When the END instruction is executed (Default... Yes) * 2
	I/O interrupt error	I/O INT ERROR	• When an interrupt occurs
	Intelligent function module error	SP.UNIT DOWN	• When the power is turned on/when reset • When the FROM/TO instruction is executed
	Control bus error	CONTROL-BUS ERROR.	• When the power is turned on/when reset • When the END instruction is executed • When the FROM/TO instruction is executed
	Momentary stop occurred	AC/DC DOWN	• Always
	Battery low	BATTERY ERROR	• Always (Default...Yes) * 3
Handling error	I/O module verification (Default... Stop) * 1	UNIT VERIFY ERROR	• When the END instruction is executed (Default... Yes) * 2
	Intelligent function module allocation error	SP.UNIT LAY ERR.	• When the power is turned on/when reset • When switched from STOP to RUN
	Intelligent program execution error (Default... Stop) * 1	SP.UNIT ERROR	• When the FROM/TO instruction is executed
	Intelligent function module version error	SP.UNIT VER.ERR.	• When the power is turned on/when reset
	No parameter	MISSING PARA.	• When the power is turned on/when reset
	Boot error	BOOT ERROR	• When the power is turned on/when reset
	Memory card operation error (Default... Stop) * 1	ICM.OPE.ERROR	• When the memory card is installed/removed
	File setting error	FILE SET ERROR	• When the power is turned on/when reset
	File access error (Default... Stop) * 1	FILE OPE.ERROR	• When an instruction is executed
	Instruction execution not possible	CAN'T EXE.PRG.	• When the power is turned on/when reset
Parameter error	Parameter setting check	PARAMETER ERROR	• When the power is turned on/when reset • When switched from STOP to RUN
	Link parameter error	LINK PARA.ERROR	• When the power is turned on/when reset • When switched from STOP to RUN
	SFC parameter error	SFC PARA.ERROR	• When switched from STOP to RUN
	Intelligent function module parameter error	SP.PARA.ERROR	• When the power is turned on/when reset

\*1:Can be changed to "Continue" in the GX Developer function parameter setting.

\*2:Can be set to "No" in the GX Developer function parameter setting. Also, checking is not performed when SM251 is on.

\*3:Can be set to "No" in the GX Developer function parameter setting.

## Self-Diagnosis List (Continued from the preceding page)

Diagnosis description		Error message	Diagnostic timing
Password error		REMOTE PASS.ERR.	<ul style="list-style-type: none"> <li>When the power is turned on/when reset</li> <li>When switched from STOP to RUN</li> </ul>
Program error	Instruction code check	INSTRUCT CODE ERR.	<ul style="list-style-type: none"> <li>When the power is turned on/when reset</li> <li>When switched from STOP to RUN</li> <li>When an instruction is executed</li> </ul>
	No END instruction	MISSING END INS.	<ul style="list-style-type: none"> <li>When the power is turned on/when reset</li> <li>When switched from STOP to RUN</li> </ul>
	Pointer setting error	CAN'T SET(P)	<ul style="list-style-type: none"> <li>When the power is turned on/when reset</li> <li>When switched from STOP to RUN</li> </ul>
	Pointer setting error	CAN'T SET(I)	<ul style="list-style-type: none"> <li>When the power is turned on/when reset</li> <li>When switched from STOP to RUN</li> </ul>
	Operation check error (Default... Stop) * 1	OPERATION ERROR	When an instruction is executed
	FOR to NEXT instruction structure error	FOR NEXT ERROR	When an instruction is executed
	CALL to RET instruction structure error	CAN'T EXECUTE(P)	When an instruction is executed
	Interrupt program error	CAN'T EXECUTE(I)	When an instruction is executed
	Instruction execution not possible	INST.FORMAT ERR.	When an instruction is executed
	SFC program structure error	SFCP.CODE ERROR	When switched from STOP to RUN
	SFC block structure error	CAN'T SET(BL)	When switched from STOP to RUN
	SFC step structure error	CAN'T SET(S)	When switched from STOP to RUN
	SFC syntax error	SFCP.FORMAT ERR.	When switched from STOP to RUN
	SFC operation check error (Default... Stop) * 1	SFCP.OPE.ERROR	When an instruction is executed
	SFC program execution error	SFCP.EXE.ERROR	When switched from STOP to RUN
	SFC block execution error	BLOCK EXE.ERROR	When an instruction is executed
	SFC step execution error	STEP EXE.ERROR	When an instruction is executed
PLC error	Watch dog error supervision	WDT ERROR	Always
	Program time exceeded	PRG.TIME OVER	Always
Multiple CPU	Other CPU major error	MULTI CPU DOWN	<ul style="list-style-type: none"> <li>Always</li> <li>When the power is turned on/when reset</li> </ul>
	Multiple CPU consistency error	CPU VER.ERR.	When the power is turned on/when reset
	Other CPU minor error	MULTI CPU ERR.	Always
BOOT OK		BOOT OK	When the power is turned on/when reset
Annunciator check		F * * * *	When an instruction is executed
CHK Instruction check		<CHK>ERR * * * - * * *	When an instruction is executed

\* 1: Can be changed to "continues" in the GX Developer function parameter setting.

### 7.17.1 Interrupt due to error occurrence

The High Performance model QCPU can execute the interrupt program of the interrupt pointer that is set as the interrupt object when an error occurs.

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.

Interrupt pointer	Corresponding error message
I32	Stop all errors *
I33	Empty
I34	UNIT VERIFY ERR. FUSE BREAK OFF SP.UNIT ERROR
I35	OPERATION ERROR SFCP OPE.ERROR SFCP EXE.ERROR
I36	ICM.OPE.ERROR FILE OPE.ERROR
I37	EXTEND INS.ERR.
I38	PRG.TIME OVER
I39	CHK instruction Annunciator detect
I40 to I47	Empty

Errors that occur when the system can continue the drive mode, where "continue" is selected from "continues/stops".

POINT
<p>(1) The interrupt pointers I32 to I39 is at an execution disable mode when the power is started or High Performance model QCPU is reset. When using I32 to I39, use the IMASK instruction and EI instruction to enable execution.</p> <p>(2) *: The I32 interrupt program is not executed upon the following serious errors.</p> <ul style="list-style-type: none"> <li>• MAIN CPU DOWN</li> <li>• END NOT EXECUTE</li> <li>• RAM ERROR</li> <li>• OPE CIRCUIT ERR.</li> </ul>

### 7.17.2 LED display when error occurs

When an error occurs, the LED located on the front of the High Performance model QCPU turns on / flickers.

See Section 7.21 for the details on the LED operation.

### 7.17.3 Error cancellation

High Performance model QCPU error cancel operation can be performed only for error that can continue the High Performance model QCPU operation.

#### (1) Error cancellation

##### (a) Procedures for error cancellation

The error cancel is performed as follows:

- 1) Resolve the cause of error.
- 2) Store the error code of the error to be canceled in the special register SD50.
- 3) Switch special relay SM50 from OFF to ON.
- 4) The error is canceled.

##### (b) Status after error cancellation

When the CPU module is recovered by canceling 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 canceling the error, it is logged again in the failure history.

##### (c) Cancellation of annunciator

For the cancellation of the annunciator detected multiple times, only the first detected "F" is canceled.

POINT
<p>(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.</p> <p>[Example]</p> <p>When the error codes 2100 and 2101 occur, canceling the error code 2100 will also cancel the error code 2101.</p> <p>When 2100 and 2111 occur in the error code and error code 2100 is canceled, error code 2111 is canceled as well.</p> <p>(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).</p> <p>[Example]</p> <p>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).</p> <p>Refer to the error code list in the High Performance model QCPU (Q mode) User's Manual (Hardware Design, Maintenance and Inspection), and resolve the error cause.</p>

## 7.18 Failure History

The High Performance model QCPU can store the failure history (results detected from the self-diagnosis function and the time) in the memory.

<b>POINT</b>
The detection time uses the High Performance model QCPU internal clock, so make sure to set the correct time when using the High Performance model QCPU for the first time.

### (1) Storage Area

- (a) The latest 16 failures are stored in the latched High Performance model QCPU failure history storage memory.
- (b) When storing more than 16, the history can be stored in the memory card file using PLC RAS setting in the "(PLC) Parameter" box.
- (c) When the history count set in the parameter and that stored in the memory card are different after the following operation is performed, the contents of the memory card history file is cleared, then the 16 failure data in the High Performance model QCPU failure history storage memory is transferred to the history file.
  - 1) When the history count in the parameter history file is changed in the middle of operation.
  - 2) When a memory card, which has a different history count from that set in the parameter, is mounted.
- (d) The storage area in the failure history file is as follows:

Storage area	File in the set memory card
Amount that can be stored	Max. 100 (can be changed) * 1

\*1: When the number of storage exceed the amount that can be stored, the oldest history is overwritten with the latest history.

<b>POINT</b>
Even if the failure history file set in the parameter does not exist in the memory card, the error will not occur in the High Performance model QCPU. The High Performance model QCPU stores the failure to the failure history storage file.

### (2) Failure History Clearing Method

The failure history storage memory/failure history file are cleared using the failure history clear in the GX Developer PLC diagnosis mode.

Data stored in both the High Performance model QCPU failure history storage memory and failure history file within the memory card can be cleared with a failure history clear.

## 7.19 System Protect

The High Performance model QCPU has a few protection functions (system protect) to prevent the program changes by a third party other than the designer (from GX Developer function or serial communication module).

There are the following methods for system protects.

Item to protect	Protect valid file	Protection description	Method	Valid Timing	Remarks
All of CPU	All files	Prohibits all write/control instructions to the High Performance model QCPU	Set the High Performance model QCPU system setting switch SW1 on.	Always	Valid for devices too
Memory card module	All files	Performs drive protect for the memory card, and write protect.	Set write-protect switch on the memory card on.	Always	—
File module	Programs Device comments Device initial values	Changes the attribute for each file as follows: 1) Read/Write display prohibit 2) Write prohibit	Change the attribute for the file in the Password Registration.	Always	—

\*The control instruction, read/write display, and write mentioned above are as follows:

Item	Description
Control instruction	High Performance model QCPU 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 High Performance model QCPU dip switches are performed even when the High Performance model QCPU system's SW1 setup switch is set to ON and the system protect function is activated.

- Booting from the standard ROM and the memory card
- Automatic write to standard ROM

## 7.19.1 Password registration

Password is used to prohibit reading and writing data of the program and comments in High Performance model QCPU from GX Developer.

The Password Registration is set for the specified memory (program memory/standard memory/memory card) program file, device comment file, and device initial file.

Either of the following two descriptions is to be registered.

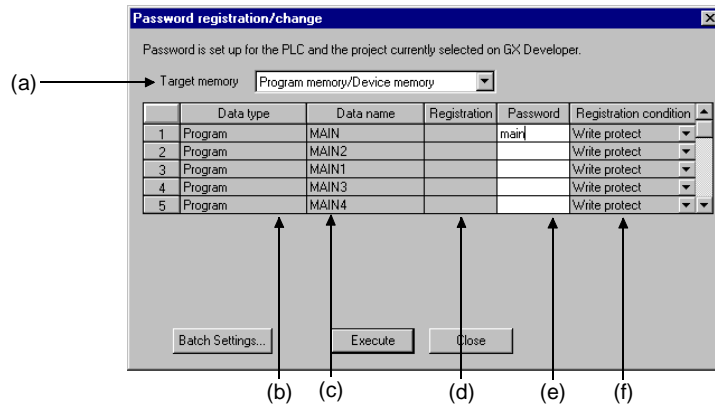
- The file name is not displayed and read/write cannot be performed as well.
- Write cannot be performed to the file. (Read only)

If the password is registered, file operations from GX Developer cannot be performed unless the same password is input.



## (1) Password Registration

To register the password, select GX Developer Online → Password setup/keyword set up for writing to PLC → Register password.



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.
- (f) Registration Condition
- 1) Write Protect ..... Write operation is restricted by a password. (Reading is not allowed.)
  - 2) Read/Write protect..... Read/Write operation is restricted by a password.
  - 3) Clear..... Password is cleared. (Sets password currently registered in "Password".)

**POINT**

- (1) Password-protected files are limited to program files, device comment files, and device initial value files. Other files cannot be password-protected.
- (2) 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.
  - Program memory/Memory card: Format PLC memory
  - Standard ROM: Write to PLC (Flash ROM)
 Take notes of the password registered and keep it on hand.

## 7.19.2 Remote passwords

The remote password function prevents illegal access to the High Performance model QCPU by users in remote locations.

The remote password function is enabled for use by setting it up in the High Performance model QCPU.

When the remote password function has been set, a check will be run on remote passwords when users in remote locations attempt to access the High Performance model QCPU with serial communication modules or Ethernet modules with modem functions.

## (1) Setting up, amending and canceling remote passwords

## (a) Remote password setup

Remote passwords are set up on the GX Developer's remote password setup screen. The GX Developer is then connected to the High Performance model QCPU into which the remote password is to be set, and the password uploaded.

The High Performance model QCPU will transmit the remote password to specified serial communication modules and Ethernet modules when the power supply to the sequence is switched on or the High Performance model QCPU is reset.

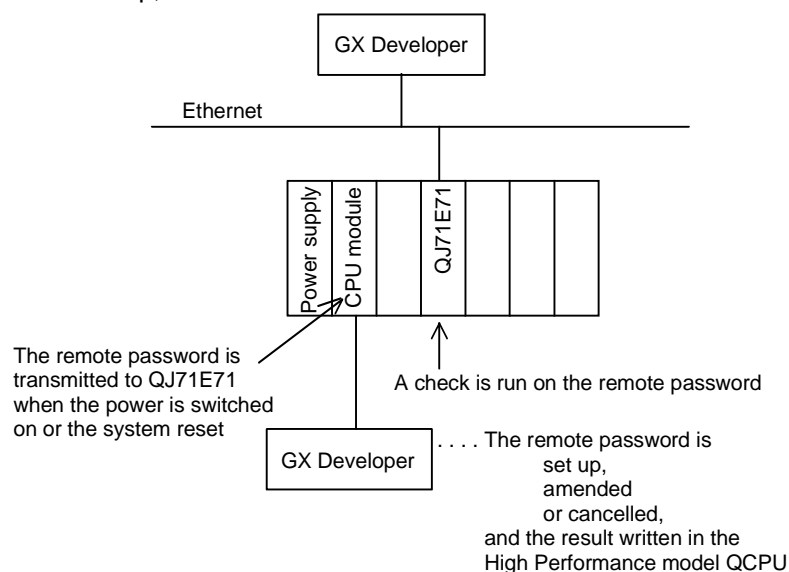
## (b) Amending and canceling remote passwords

It is possible to amend and cancel remote passwords by connecting the GX Developer to the relevant High Performance model QCPU.

Remote passwords set in the High Performance model QCPU can be amended or cancelled by setting up an amended password or canceling a remote password with the GX Developer.

Remote passwords cannot be amended or cancelled from a remote location.

For example, an outline of what will happen when a remote password is set up, amended or cancelled from an Ethernet module is shown below.

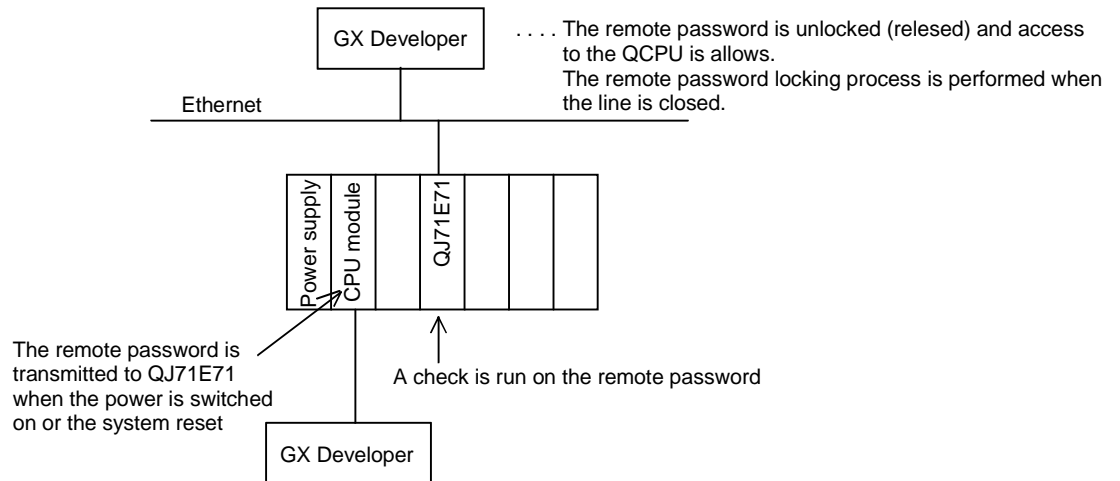


(2) Remote password lock/unlock processing

Unlocks the Ethernet module remote passwords for the access source via modems, serial communication modules and the Ethernet.

Access to the High Performance model QCPU is enabled if the remote password matches up.

For example, an outline of what will happen during remote password lock/unlock processing with an Ethernet module is shown below.



(3) Number of remote password-set modules

The number of remote password-set modules changes depending on the version of used GX Developer.

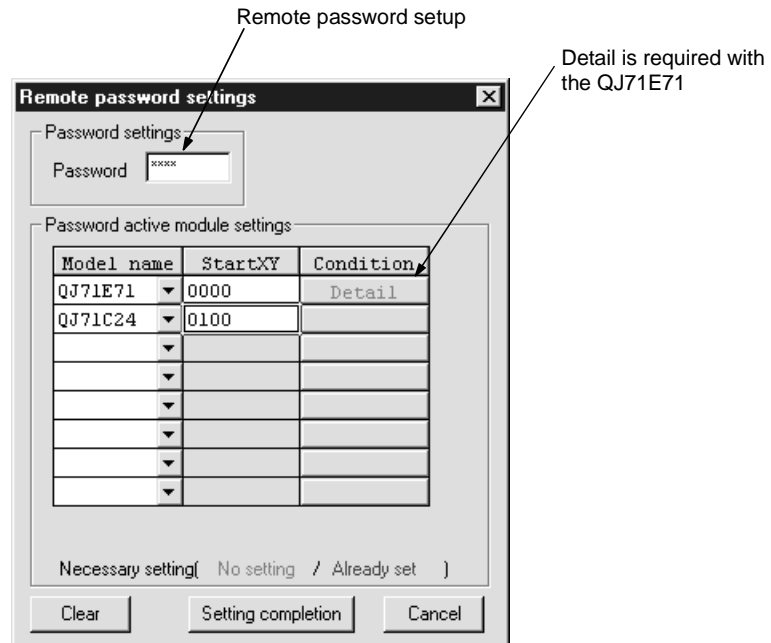
The following table indicates the number of remote password-set modules depending on the GX Developer version.

GX Developer Version	Module Name	Maximum Number of Set Modules	Maximum Number of Set Modules in System
Version6 Version7	Ethernet module	4 modules	8 modules
	Serial communication module	4 modules	
	Modem interface module		
Version8 or later	Ethernet module	4 modules	8 modules
	Serial communication module	8 modules	
	Modem interface module		

(4) Procedure for setting up the remote password

[GX Developer] → [Remote Password] → [Remote Password Setup] screen → [Advanced Remote Password Setup] screen.

(a) Setup screen



(b) Setup fields

Field		Description	Setup range/Selection range
Password settings		Remote password entry	4 bytes, alphanumeric, special characters
Password active module settings	Model name	Model selection	QJ71E71/QJ71C24
	Start XY	Module's head address setup	0000 <sub>H</sub> to 0FE0 <sub>H</sub>
Detail		—	Setup/Not setup
User's connection No.		User's connection No. setup	Connection No. 1 to Connection No. 16
System connection	Automatic open UDP port	Adds a check to the valid remote password port	—
	FTP communication port (TCP/IP)		
	GX Developer communication port (TCP/IP)		
	GX Developer communication port (UDP/IP)		
	HTTP port		

**POINT**

Refer to the following manuals for further details on the remote password function.

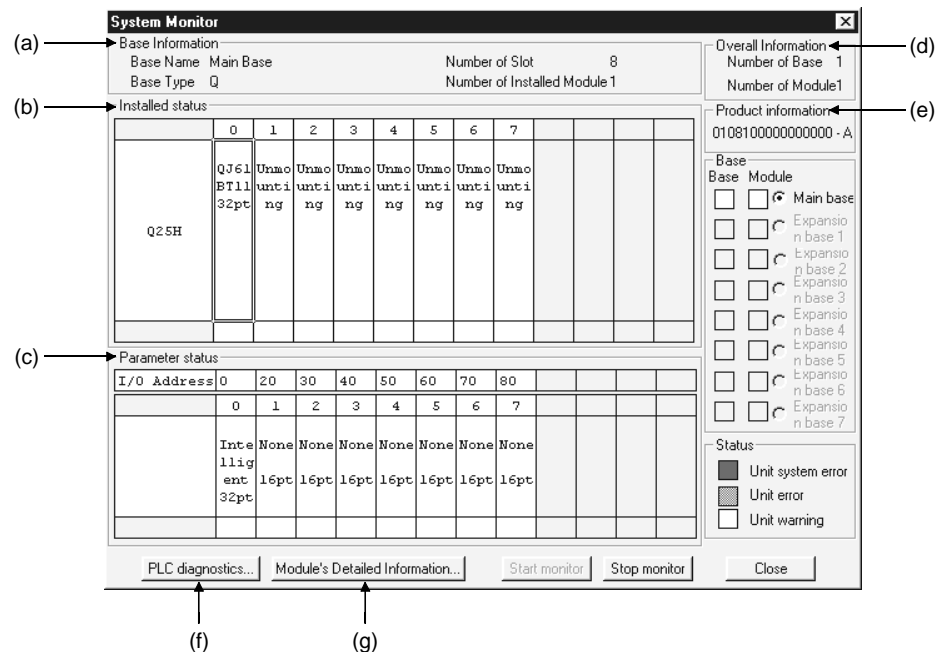
- Using Serial Communication Modules  
Q Corresponding Serial Communication Module Users' Manual (Application)
- Using Ethernet Modules  
Q Corresponding Ethernet Interface Module Users' Manual (Basic)

## 7.20 Monitoring High Performance model QCPU System Status from GX Developer (System Monitor)

### (1) In Case of GX Developer Version 4 (SW4D5C-GPPW-E) or GX Developer Version 5 (SW5D5C-GPPW-E)

The System Monitor window provides the following information about the High Performance model QCPU connected to the personal computer:

- Base information
- Installed status
- Parameter status
- Overall information
- Product information



#### (a) Base Information

The Base Information section shows information about a selected base unit: the name of a base unit (main base or extension base 1 to 7), the number of slots, base type, the number of modules mounted on the base.

- 1) The "Base name" field indicates the main base or extension base (with power source). It indicates the extension base even if a power source module is not installed on the extension base.
- 2) The "Base type" field indicates the following abbreviations:
  - Q for Q33B, Q35B, Q38B, Q312B
  - Q for Q63B, Q65B, Q68B, Q612B
  - QA for QA1S65B, QA1S68B
- 3) The "Number of Installed Module" field indicates the number of modules installed on the base unit.

#### (b) Installed status

The "Installed status" section shows the model name of the selected base unit and the number of points. The "slot" column indicates "Unmounting" if a base unit is not installed in a particular slot.

- (c) **Parameter status**

The "Parameter status" section shows the I/O number of each slot of a base unit, module type, and the number of points. If the number of available points and the installed status are displayed in any column of the Parameter status section, make the setting so that I/O assignments of PLC parameter can match the installed status.
- (d) **Overall Information**

The Overall Information section shows the number of base units used and the number of modules installed on the base.
- (e) **Base**

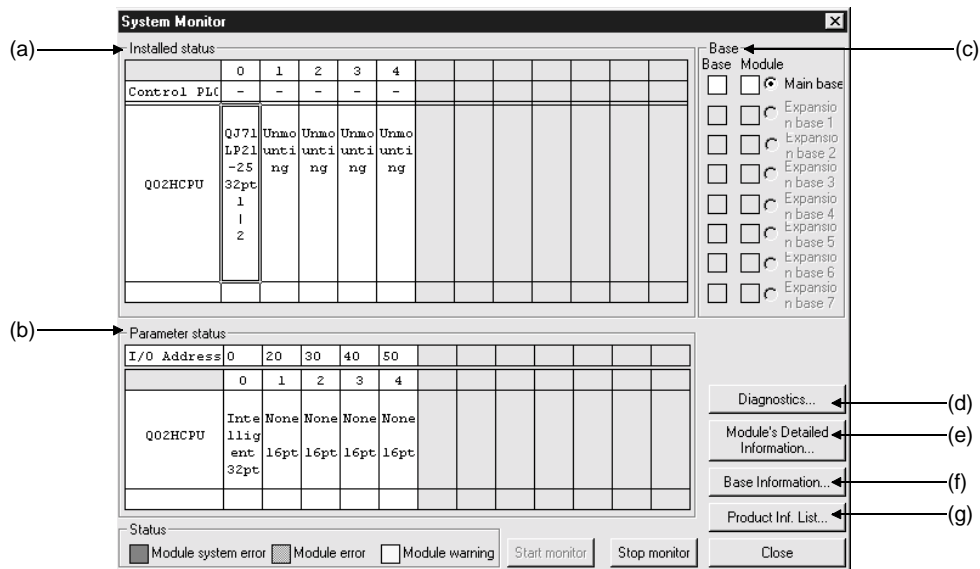
The Base section indicates the status of base units used and modules installed. The Module column indicates the status of a module when the module is in an abnormal condition.
- (f) **PLC diagnostics**

This button is used to monitor the status of the High Performance model QCPU and an error.
- (g) **Module's Detailed Information**

This button is used to view detailed information about a selected module. For details on intelligent function modules, see the manual of an intelligent function module.

- (2) In case of GX Developer Version 6 (SW6D5C-GPPW-E) or later It is possible to confirm the following information for High Performance model QCPUs connected to personal computers with the GX Developer system monitor (see illustration below.)

- Installed status
- Operation status
- Module's detailed information
- Product information



(a) Installed status

Enables the controlling CPU, the model and the number of modules mounted onto the selected base unit to be confirmed.

"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.

(b) 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.

In this event, align the PLC parameter's I/O allocation with the actual status by allocating an I/O.

(c) Base

Enables the status of the modules mounted onto the base unit in use to be confirmed.

The status will be displayed in the unit column when an error has occurred for even one of the modules.

(d) Diagnostics

This function is used to confirm the status of the High Performance model QCPU and errors.

(e) 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.

(f) Base information

Enables the "Overall Information" and "Base Information" to be confirmed.

1) Overall information

Enables the number of base units in use and the number of modules mounted on the base units to be confirmed.

2) 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 (main base unit, additional base units 1 to 7) to be confirmed.

(g) Product Information List

Enables the individual information for mounted CPU modules, I/O modules and intelligent function modules to be confirmed (type, series, model, number, head I/O, control PLC, serial No., function version.)

Serial No.      Function version

↓                      ↓

Slot	Type	Series	Model name	Points	I/O No.	Control	Serial No	Ver
PLC	PLC	Q	Q02HCPU	-	-	-	0211200000000000	B
0-0	Intelli.	Q	QJ71LP21-25	32pt	0000	-	0208100000000000	B
0-1	-	-	None	-	-	-	-	-
0-2	-	-	None	-	-	-	-	-
0-3	-	-	None	-	-	-	-	-
0-4	-	-	None	-	-	-	-	-

Product Information List

CSV file creating      Close



## 7.21 LED Display

The High Performance model QCPU has an LED to indicate the High Performance model QCPU operation status on the front of the High Performance model QCPU. The display details of each LED are described below.

## 7.21.1 LED display

(1) The details of the LED display are shown below:

LED name	Display Description
Mode	Indicates the High Performance model QCPU mode (Q mode/ A mode) ON (green): Q mode (No registration of enforced ON/OFF for external I/O) ON (orange): A mode Flicker (green) : with registration of enforced ON/OFF for external I/O
RUN	Indicates the CPU module operation status. On : When operating with the RUN/STOP switch at "RUN". Off : When stopped with the RUN/STOP switch at "STOP". Or when an error that stops operation is detected. Flicker : When writing parameters and programs during STOP, and when setting the RUN/STOP switch from [STOP] → [RUN]. Perform the following operations in order to illuminate the RUN LED after program writing. • Set the RUN/STOP switch to [RUN] → [STOP] → [RUN]. • Reset the system with the RESET/L. CLR switch. • Switch on the power to the PLC again. Perform the following operations in order to illuminate the RUN LED after parameter writing. • Reset the system with the RESET/L. CLR switch. • Switch on the power to the PLC again. (When the RUN/STOP switch has been set to [RUN] → [STOP] → [RUN] after the parameters have been amended, the parameters related to intelligent function modules and other network parameters will not be reflected back.)
ERR.	Indicates the CPU module error detection status. On : When a self-diagnosis error that does not stop the operation (except for battery error) is detected. (Set the operation error set mode to "continue" in the parameter mode PLC RAS setting.) Off : Normal Flicker : When an error that stops the operation is detected. When automatic write to standard ROM is complete normally (BOOT LED also flickers)
USER	The detection status for the CHK instruction or annunciator F status is indicated. On : When an error is detected with the CHK instruction, or when the annunciator is turned on. Off : Normal Flicker : When the latch clear is executed.
BAT.	Indicates the CPU module and Memory card battery status. On : When a battery error is detected due to low battery voltage. Off : Normal
BOOT	Indicates the execution status of the boot operation. On : When the execution is complete. Off : When not executed. Flicker : When automatic write to standard ROM is complete normally. (ERR. LED also flickers)

(2) Method to turn off the LED

The LED that is on can be turned off by the following operation. (Except for the reset operation.)

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.)	○	○	○	×
Turn off the LED by operating the special relay SM202 and special register SD202.	×	○	×	○

○ : Valid    × : Invalid

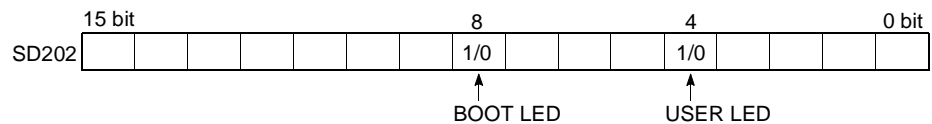
\*1: Special relay and special register contents

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. (For further information on error codes, see the High Performance model QCPU (Q Mode) Users 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. (Only USER LED and BOOT LED can be turned off.)



1 means "turn off" and 0 means "leave on" in the setting. The setting to turn off each LED is as follows. (All in hexadecimal.)

- When turning off both LEDs: SD202=110H
- When turning off only the BOOT LED: SD202=100H
- When turning off only the USER LED: SD202=10H

(3) Method to not display the ERR. LED, USER LED, and BAT. LED

The ERR. LED, USER LED, and BAT. LED have the same priorities explained in Section 7.21.2.

When an error number for an LED is deleted from this priority, the LED will not turn on even if an error with that error number occurs.

(See **POINT** in Section 7.21.2 for the setting method.)

7.21.2 Priority setting

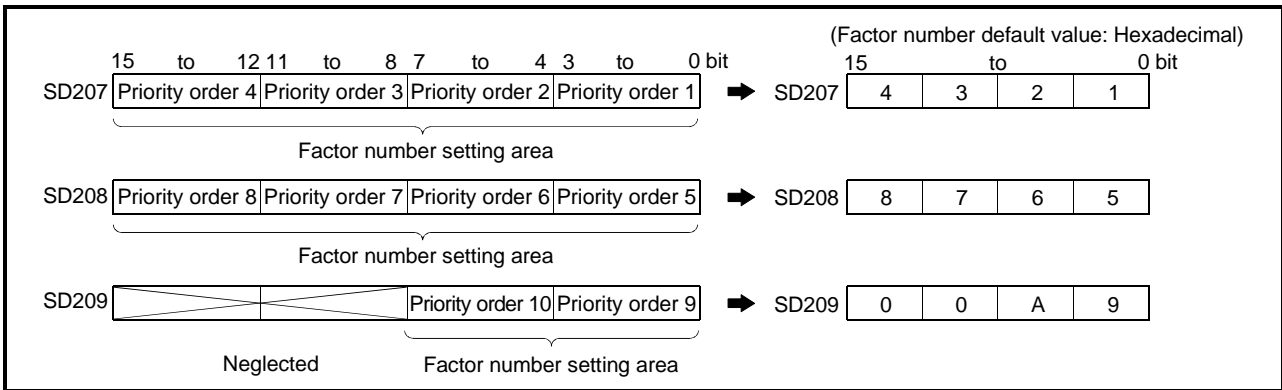
When multiple factors that can be displayed occur, the display is performed with the following conditions:

- 1) A stop error is displayed without condition.
- 2) An operation continue error is displayed according to the priority factor number set as the default.

The priority can be changed. (Set with special registers SD207 to SD209)

- 3) When errors with the same priority level occur, the error detected first is displayed.

The priority is set with the special registers SD207 to SD 209 in the following manner:



The description and default priority for the factor number to be set in the special registers SD207 to SD209 are as follows:

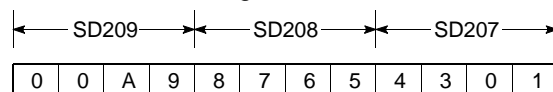
Priority	Factor number (Hexadecimal)	Description	Remarks
1	1	AC/DC DOWN	Power shutoff
2	2	UNIT VERIFY ERR. FUSE BREAK OFF SP.UNIT ERROR	I/O module verification error Fuse shutoff Intelligent function module verify error
3	3	OPERATIN ERROR LINK PARA. ERROR SFCP OPE.ERROR SFCP EXE.ERROR	Calculation error Link parameter error SFC instruction calculation error SFC program execution error
4	4	ICM.OPE.ERROR FILE OPE.ERROR	Memory card operation error File access error
5	5	PRG.TIME OVER	Constant scan setting time time up over Low speed execution monitoring time time up
6	6	CHK instruction	
7	7	Annunciator	
8	8	—	
9	9	BATTERY ERR.	
10	A	Clock data	

### POINT

- (1) When leaving the LED turned off at the error described above, set the factor number setting area (each 4 bits), which stores the factor number corresponding to SD207 to SD209 to "0".

[Example]

To leave the ERR. LED off when a fuse shutoff error is detected, set the factor number setting area to "0" where the error number is "2".



Because the factor number "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 number "2" (I/O module verify error or intelligent function module verify error) is detected, the ERR. LED remains off.

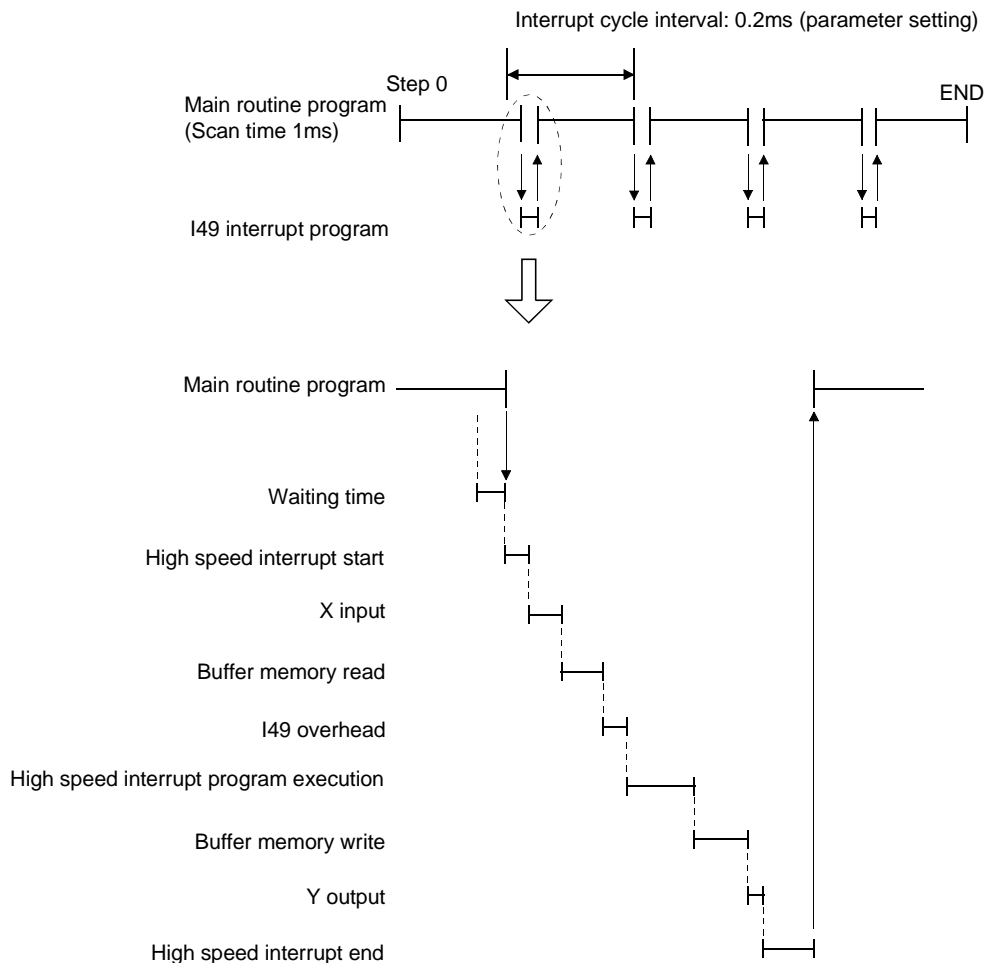
- (2) Even if the LED is set to be turned off, error code storage is performed for SM0 (Diagnostic error flag) on, SM1 (self-diagnosis flag) on, and SD0 (CPU diagnostic error register).

### 7.22 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.

High accuracy control, e.g. accurate positioning detection, is available with the PLC CPU alone.



POINT
<p>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 I0 to I48, I50 to I255, and fixed scan execution type programs.</p> <p>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 7.22.3 for other restrictions.</p>

## (1) Compatible CPUs

CPU Type	Compatibility	Remarks
Q02HCPU, Q06HCPU Q12HCPU, Q25HCPU	Compatible	Upper 5 digits of serial No. is "04012" or later
Q02CPU	Incompatible	—

## (2) 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.

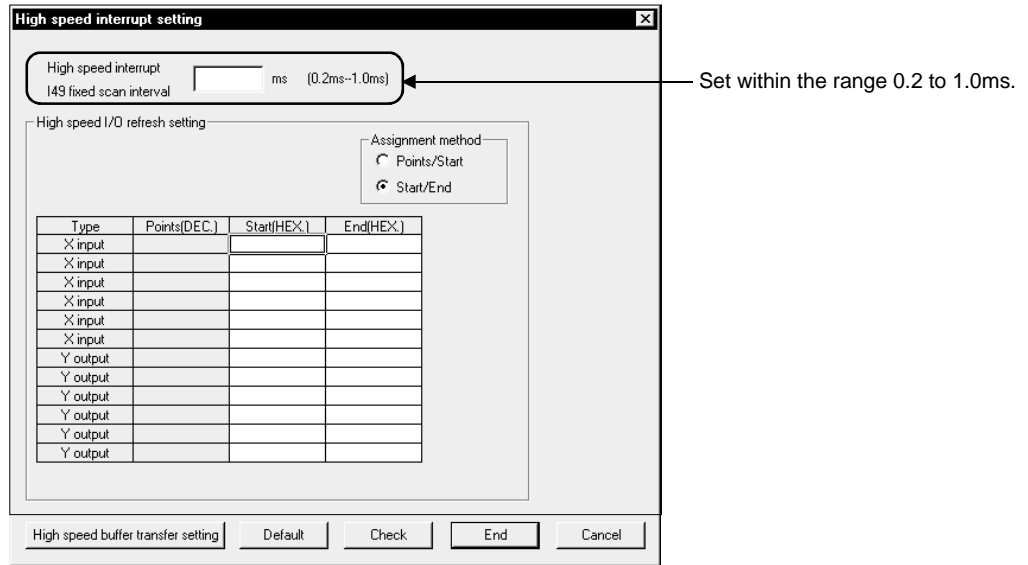
## (3) Detailed items of high speed interrupt function

Item	Description
Interrupt program execution	Runs the interrupt program created with I49.
High speed I/O refresh	Updates I/O signals between the I/O, intelligent function modules and CPU module at interrupt cycle intervals.
High speed buffer transfer	Updates data between the intelligent function module buffer memories and CPU module devices at interrupt cycle intervals.

## 7.22.1 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.

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.



Note the following when executing a high speed interrupt program.

- (1) The high speed interrupt program is only executed as interrupt enable.  
(Refer to Section 7.22.4 (3) for the items that delay high speed interrupt's start from interrupt disable.)
- (2) 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.)

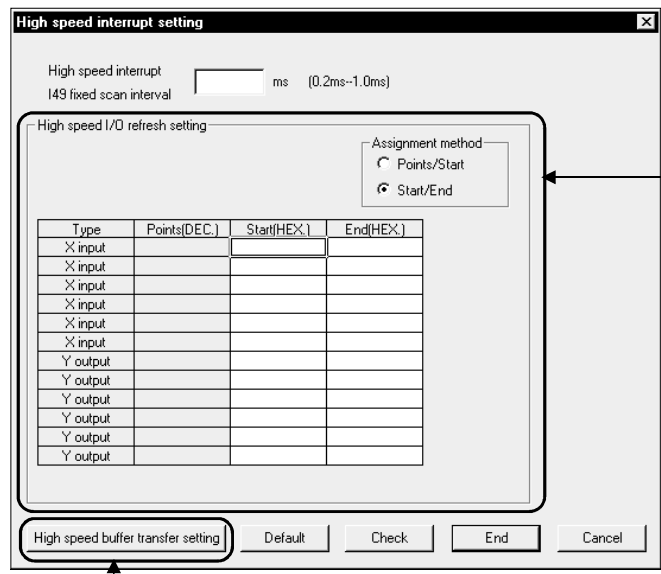
7.22.2 High speed I/O refresh, high speed buffer transfer

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) To execute this function, set "High speed interrupt I49 fixed scan interval" that was set in Section 7.22.1 and "High speed I/O refresh setting" and "High speed buffer transfer setting".

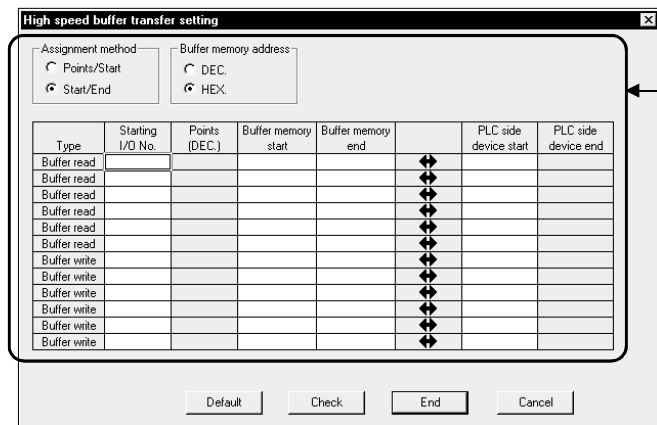
• High speed I/O refresh setting



Set the X/Y refresh ranges.

Choose here when setting the high speed buffer transfer ranges.

• High speed buffer transfer setting



Set the buffer memory transfer ranges.

**POINT**

It is recommended to mount the target modules of this function on the main base. (Access time to modules mounting on the main base is shorter than that on the extension base.)



The settings for high speed I/O refresh and high speed buffer transfer are as follows.

Item	Sub Item	Contents	Restrictions	Number of Settings
High speed I/O refresh setting	Start (X/Y)	Head device No. (X0 to XFF0/Y0 to YFF0)	I/O and intelligent function modules	Up to 6 points for X input and Y output, respectively.
	Points	Number of transferred bits (16 to 4096)	Specify multiples of 16 only. *1	
High speed buffer transfer setting	Starting I/O No.	Head I/O address / 10 <sub>H</sub> (0 to FF <sub>H</sub> )	Intelligent function modules only *2	Up to 6 points for read and write, respectively.
	Buffer memory start	Head address (0 to FFFF <sub>H</sub> )	Intelligent function modules only	
	Points	Number of transferred bits (1, 2 to FFFE <sub>H</sub> )	Specify even addresses and even words only *3	
	PLC side device start	Head device No.	D, W, R, ZR only	

\*1: Only multiples of 16 can be set for both the head device No. and the number of transferred bits.

\*2: Since the QA base cannot be connected, the A/QnA series intelligent function modules are not the target.  
(When the QA base is connected, "PARAMETER ERROR (3006)" is detected.)

"PARAMETER ERROR (3006)" is also detected if an error occurs in intelligent function module mounting check or buffer memory capacity check.

\*3: Only when the specified number of transferred words is 1, an odd address can be specified.

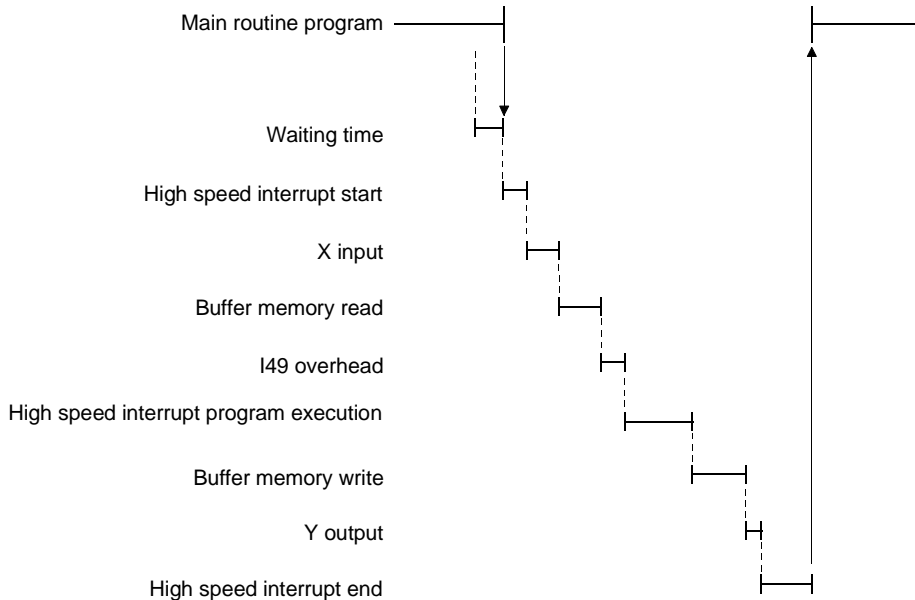
- (2) This function's execution requires both the EI instruction is in execution, and I49 is not masked by the IMASK instruction. By default, I49 is not masked by the IMASK instruction.

Executability of high speed interrupt (○: Executable, ×: Inexecutable)

Conditions		High speed interrupt program	High speed I/O refresh High speed buffer transfer
EI instruction in execution	IMASK instruction (I49 unmasked)	○	○
	IMASK instruction (I49 masked)	×	×
DI instruction in execution	—	×	×

7.22.3 Processing time

The following chart shows the processing times of the high speed interrupt function between a start and an end.



High speed I/O refresh and high speed buffer transfer take the following processing times in their processings.

Processing Item	Processing Time
Waiting time	<ul style="list-style-type: none"> <li>• Max. 37.5 μs or more than 37.5 μs instruction processing time</li> <li>• Max. 40 μs when MELSECNET/H, CC-Link or intelligent function modules are mounted on extension base.</li> </ul>
High speed interrupt start + high speed interrupt end	22 μs
X input	(1) Main base $\text{Time} = 0.14 \times (\text{total number of X points}) + 0.65 \times (\text{number of settings}) + 0.85$ (2) Extension base $\text{Time} = 0.21 \times (\text{total number of X points}) + 0.65 \times (\text{number of settings}) + 0.85$ (Calculation example) 3.74 μs when the module is mounted on the main base, the number of settings is 1, and the number of X points is 16
Buffer memory read	(1) Main base (a) 16 words or less: $\text{Time} = 0.47 \times (\text{total number of transferred words}) + 2.85 \times (\text{number of settings}) + 0.95$ (b) More than 16 words: $\text{Time} = 0.5 \times (\text{total number of transferred words}) + 0.95$ (2) Extension base (a) 16 words or less: $\text{Time} = 1.07 \times (\text{total number of transferred words}) + 2.85 \times (\text{number of settings}) + 0.95$ (b) More than 16 words: $\text{Time} = 1.1 \times (\text{total number of transferred words}) + 0.95$ (Calculation example) 4.74 μs when the module is mounted on the main base, the number of settings is 1, and the number of words is 2
I49 overhead	41 μs
High speed interrupt program execution	Depends on the user-created interrupt program.

Processing Item	Processing Time
Buffer memory write	<p>(1) Main base</p> <p>(a) 16 words or less:  Time = <math>0.47 \times (\text{total number of transferred words}) + 2.65 \times (\text{number of settings}) + 0.95</math></p> <p>(b) More than 16 words:  Time = <math>0.55 \times (\text{total number of transferred words}) + 0.95</math></p> <p>(2) Extension base</p> <p>(a) 16 words or less:  Time = <math>1.07 \times (\text{total number of transferred words}) + 2.65 \times (\text{number of settings}) + 0.95</math></p> <p>(b) More than 16 words:  Time = <math>1.15 \times (\text{total number of transferred words}) + 0.95</math></p> <p>(Calculation example) 4.54 <math>\mu</math>s when the module is mounted on the main base, the number of settings is 1, and the number of words is 2</p>
Y output	<p>(1) Main base  Time = <math>0.13 \times (\text{total number of Y points}) + 1.55</math></p> <p>(2) Extension base  Time = <math>0.2 \times (\text{total number of Y points}) + 1.55</math></p> <p>(Calculation example) 3.63 <math>\mu</math>s when the module is mounted on the main base, the number of settings is 1, and the number of Y points is 16</p>

## 7.22.4 Restrictions

This section explains the items to be noted when the high speed interrupt function is executed.

Depending on the items, a WDT error may occur or high speed interrupt may not be executed at preset cycle intervals if the corresponding restrictions are not satisfied.

The restrictions are roughly classified into the following four different items.

- (1) Items that are all disabled when high speed interrupt setting is made
- (2) Items that are disabled only within high speed interrupt
- (3) Items that hold high speed interrupt by interrupt disable
- (4) Items to be noted in addition to (1) to (3)

The time taken to run an interrupt program once should not exceed the preset time of the interrupt cycle interval. (If that time exceeds the preset time of the interrupt cycle interval, high speed interrupt operation cannot be guaranteed.)

### (1) Items that are all disabled when high speed interrupt setting is made

No.	Item	Restriction	When Used
1	Q02CPU	No function for Q02CPU.	Parameter error is detected.
2	Base unit	QA1S6□B or QA6□B base is not available.	Parameter error is detected.
3	Multiple PLC system	Multiple PLC 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 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. 1) SFC transition monitor check function using SM90 to 99, SD90 to 99 2) 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 available and is ignored. (When trace read is performed, data may not be set.)
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 program correction, online file batch-write	Online program correction is not available. Online file batch-write is not available.	Since interrupt is disabled during online program correction, a high speed interrupt start is delayed during that period and therefore high speed interrupt is not available at preset cycle. (The following time is taken. • Max. 102 μs for online program correction • Max. 300ms for online file batch-write)
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. 410 μs for standard RAM • 400 μs + 100 μs × number of program files for SRAM card)

No.	Item	Restriction	When Used
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.) • $390 \mu s + 170 \mu s \times n$ for standard RAM • $390 \mu s + 950 \mu s \times n$ 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: $(0.07 \times N + 34) \mu s$ N points random read/write: $(0.07 \times N + 101) \mu s$
14	Monitor via other station	Monitor via 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 ( $102 \mu s$ ) 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.)

## (2) 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	Detailed condition monitor	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 hold high speed interrupt by interrupt disable

No.	Item	Restriction
1	Instruction	Interrupt is disabled during instruction execution.
2	Link refresh	Interrupt is disabled during refresh (bus access). For MELSECNET/H, CC-Link or intelligent function module refresh, waiting time is max. $37.5 \mu s$ when modules are mounted on main base or max. $40 \mu s$ 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 $30 \mu s$ . 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. $(0.096 \times \text{number of device points} + 20) \mu s$
5	AC DOWN	Waiting time is max. 20ms for high speed interrupt start.

- (4) Items to be noted in addition to (1) to (3)
- (a) 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) 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) As in the other interrupt programs, there are the following items as precautions for program creation.
- 1) The device turned on by the PLS instruction in the high speed interrupt program remains on until the same interrupt program is executed again.
  - 2) While the high speed interrupt program is executed, DI (interrupt disable) is established. Do not execute the EI/DI instruction in the high speed interrupt program.
  - 3) Timers cannot be used in the high speed interrupt program.
  - 4) If the high speed interrupt program is executed during time measurement such as scan time or execution time measurement, the high speed interrupt program running time is added to the measurement time. Hence, if the high speed interrupt program is executed, the values to be stored into the following special registers and the monitor values of GX Developer become longer than the values when the high speed interrupt program is not 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
  - SD532, SD533: Minimum scan time for low speed
  - SD534, SD535: Maximum scan time for low speed
  - SD540, SD541: END processing time
  - SD542, SD543: Constant scan waiting time
  - SD544, SD545: Low speed program cumulative execution time
  - SD546, SD547: Low speed program execution time
  - SD548, SD549: Scan program execution time
  - SD551, SD552: Service interval time
- (GX Developer monitor values)
- Execution time measurement
  - Scan time measurement
  - Constant scan

### 7.23 Module Service Interval Time Reading

The High Performance QCPU 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.

To read the module service interval time <sup>\*1</sup>, operate the following special relay and special registers.

#### (1) 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

#### (2) 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 FFFF <sub>H</sub> .
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: 100 μs units (range 0 to 900, stored at intervals of 100 μs) (Example) When module service interval time is 123.4ms SD551 = 123, SD552 = 400

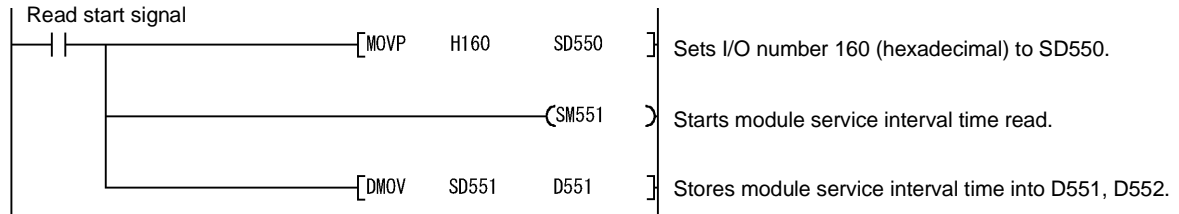
#### REMARK

\*1: The module service interval indicates the time between a transient request such as monitor, test, program read/write.

The access interval in cyclic communication from the network module is not stored.

(Program example)

When reading the module service interval time of the intelligent function module of X/Y160



#### POINT

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.



8 COMMUNICATION WITH INTELLIGENT FUNCTION MODULE/SPECIAL FUNCTION MODULE

- (1) **Description of intelligent function modules/special function modules**  
 High Performance model QCPU allows the use of the Q Series compatible intelligent function modules and the AnS series compatible special function modules.  
 The intelligent function module/special function module is a module that allows High Performance model QCPU 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/special function modules**  
 The intelligent function module/special function module is equipped with memory (buffer memory) to store the data received from or output to external devices.  
 High Performance model QCPU reads/writes the data from/to the buffer memory.

8.1 Communication Between High Performance model QCPU and Intelligent Function Modules

The following methods enable the communication between High Performance model QCPU and intelligent function modules:

- Initial setting or automatic refresh setting using GX Configurator
- Device initial value
- FROM/TO instruction
- Intelligent function module device
- Instructions dedicated for intelligent function modules

The following table shows the communication timing for the communication methods with intelligent function modules described above:

Communication method with intelligent function modules	Communication timing					Storage location * 1	
	Power ON	High Performance model QCPU reset	STOP → RUN	Instruction execution	END processing	High Performance model QCPU * 2	Intelligent * 3
GX Configurator	Initial setting	○	○	○	—	—	○
	Automatic refresh setting	—	—	—	—	○	—
Device initial value	○	○	○	—	—	○	—
FROM/TO instruction * 4	—	—	—	○	—	○	—
Intelligent function module device * 4	—	—	—	○	—	○	—
Instructions dedicated for intelligent function modules * 4	—	—	—	○	—	○	—

○: Can be stored. —: Cannot be stored.

**REMARK**

- \*1: Indicates whether the data (designated by the GX Configurator, of the device initial value, etc.) is stored in High Performance model QCPU or in an intelligent function module.
- \*2: Represents the internal memory of High Performance model QCPU or a memory card.
- \*3: "Intelligent" represents an intelligent function module.
- \*4: Represents the program using the intelligent function module device, the FROM/TO instruction, or the instructions dedicated for intelligent function modules.

8.1.1 Initial setting and automatic refresh setting using GX Configurator

(1) Initial and automatic refresh settings of intelligent function modules  
 Installing the GX Configurator compatible with the intelligent function module enables the initial setting and automatic refresh setting with GX Developer. When the initial setting and automatic refresh setting of the intelligent function module is designated with GX Developer, you can write/read data without creating the program for the communication with the intelligent function module. Moreover, you can conduct the initial setting or automatic refresh setting without designating the buffer memory address of the intelligent function module.

(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.

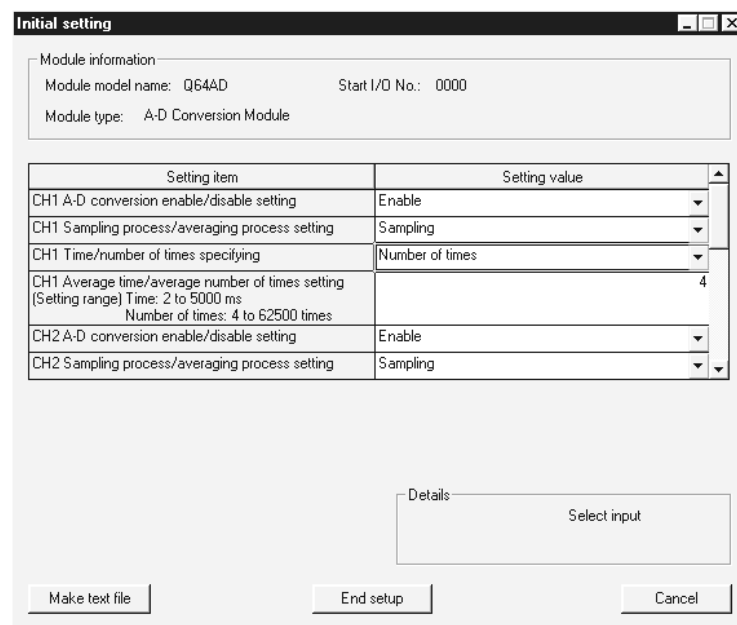
(a) Initial setting

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.

[Initial setting screen]



The designated initial setting data is stored in the intelligent function module.

## (b) Auto refresh setting

For the auto refresh setting, designate the device at High Performance model QCPU 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.

[Auto refresh setting screen]

Setting item	Module side Buffer size	Module side Transfer word count	Transfer direction	PLC side Device
CH1 Digital output value	1	1	->	D11
CH2 Digital output value	1	1	->	D12
CH3 Digital output value	1	1	->	D13
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	->	

The designated auto refresh setting data is stored in the intelligent function parameters of High Performance model QCPU.

**REMARK**

For the details of GX Configurator, refer to the manual of the intelligent function module being used.

## 8.1.2 Communication using device initial value

## (1) Device initial value

The device initial value is used to designate the initial setting of the intelligent function module without using a program.

The designated device initial value is written from High Performance model QCPU to the intelligent function module when High Performance model QCPU is turned ON, is reset, or is switched from STOP to RUN.

## (2) Designation of the device initial value

Using the device memory of GX Developer, designate the data of the intelligent function module to be used as the device initial value.

In the device initial value setting of GX Developer, designate the range to be used with the intelligent function module device as the device of the device initial value.

**REMARK**

1) For the device initial value, see Section 10.13.2.

2) For the intelligent function module device, see Section 10.5.

### 8.1.3 Communication using FROM/TO instruction

#### (1) FROM/TO instruction

At the execution of the FROM/TO instruction, the data stored in the buffer memory of the intelligent function module can be read, or data can be written to the buffer memory of the intelligent function module.

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

- 1) For the details of the FROM/TO instruction, refer to the following manuals.
  - QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)
- 2) 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 using the 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 High Performance model QCPU in High Performance model QCPU programs.

It enables reading data stored in the buffer memory of the intelligent function module, or enables writing data to the buffer memory of the intelligent function module.

#### (2) Difference from the FROM/TO instruction

The intelligent function module device can be handled as a device of High Performance model QCPU, enabling the processing of data read from the intelligent function module with one instruction.

This saves the number of steps in the entire program.

The processing speed is the total of the instruction execution time and the access time to/from the intelligent function modules.

#### POINT

When reading and processing the data of the intelligent function module frequently in the program, use the FROM instruction to read the data at one point in the program and store and process it in a data register, instead of using the intelligent function module device every time.

Otherwise, the intelligent function module device accesses the intelligent function module every time the instruction is executed, resulting in longer scan time for the program.

#### REMARK

For the intelligent function module device, see Section 10.5.

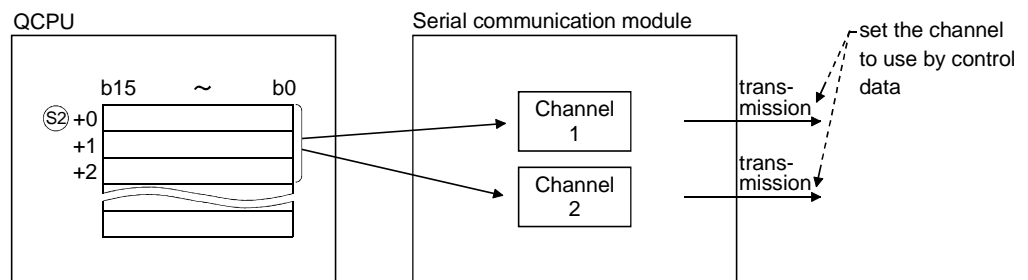
8.1.5 Communication using the instructions dedicated for intelligent function modules

(1) Description of the instructions dedicated for intelligent function modules

- (a) The instructions dedicated for intelligent function modules are the instructions that facilitate programming using the functions of the intelligent function modules.

For example, the OUTPUT instruction, which is the instruction dedicated for serial communication modules, allows data transmission in user-specified message format with no handshaking protocol.

In this case, the communication is possible without considering the buffer memory address of the objective serial communication module.



- (b) A completion device should be designated for the instruction dedicated for intelligent function modules. The designated completion device turns ON for one scan when the execution of the instruction dedicated for intelligent function modules is completed. When the completion device turns ON, another instruction dedicated for intelligent function modules can be executed to the same intelligent function module. 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.

(2) Note

- (a) If the instruction dedicated for intelligent function modules are executed and High Performance model QCPU is switched from RUN to STOP before the completion device turns ON, the completion device turns ON one scan later when High Performance model QCPU is switched to RUN next time.
- (b) The instruction dedicated for intelligent function modules can be executed to the intelligent function modules of the main base unit and extension base unit. The instruction dedicated for intelligent function modules cannot be executed to the intelligent function module installed to the remote I/O station of MELSECNET/H.

**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 Request from Intelligent Function Module to High Performance model QCPU

8.2.1 Interrupt from the intelligent function module

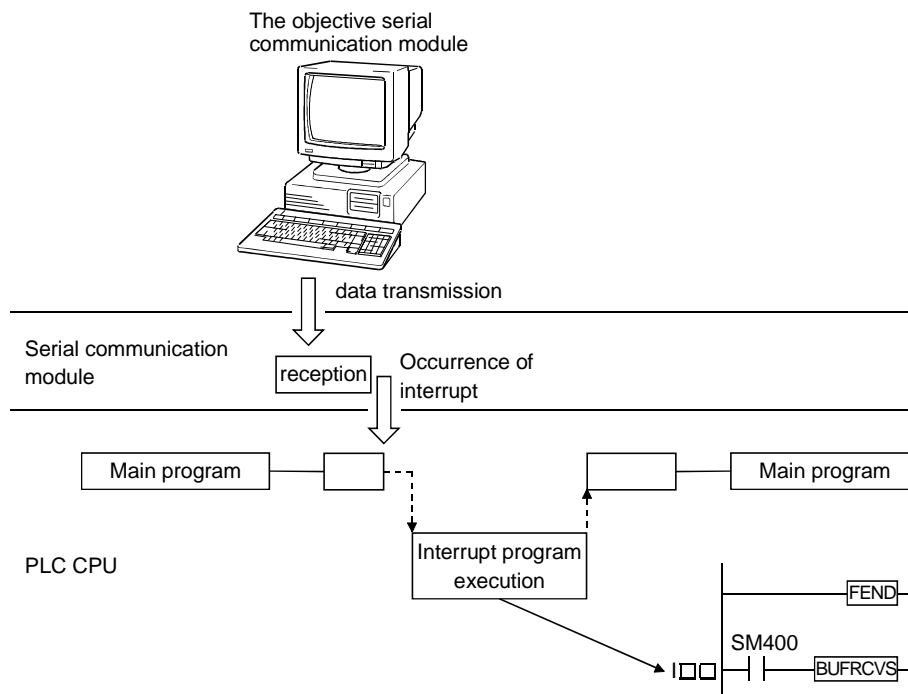
(1) Interrupt from the intelligent function module

High Performance model QCPU executes an interrupt program (I50 to I255) 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 High Performance model QCPU.



(2) 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.

To execute an interrupt program by the interrupt from the intelligent function module, refer to the manual of the intelligent function module being used.

### 8.3 Communication Between High Performance model QCPU and AnS Series Special Function Modules

The following methods enable the communication between High Performance model QCPU and the special function modules compatible with AnS Series:

- Device initial value
- Intelligent function module device
- FROM/TO instruction

The following table shows the communication timing for the communication methods with special function modules described above:

Communication method with special function modules	Communication timing					Storage location * 1	
	Power ON	High Performance model QCPU reset	STOP → RUN	Instruction execution	END processing	High Performance model QCPU * 2	Special * 3
Device initial value	○	○	○	—	—	○	—
FROM/TO instruction * 4	—	—	—	○	—	○	—
Intelligent function module device * 4	—	—	—	○	—	○	—

○: Executable. —: Not executable

#### REMARK

- \*1: Indicates whether the data of the device initial value is stored in High Performance model QCPU or in a special function module.
- \*2: Represents the internal memory of High Performance model QCPU or a memory card.
- \*3: "Special" represents a special function module.
- \*4: Represents the program using the intelligent function module device or the FROM/TO instruction.

#### 8.3.1 Communication using device initial value

##### (1) Device initial value

The device initial value is used to designate the initial setting of the special function module without using a program.

The designated device initial value is written from High Performance model QCPU to the special function module when High Performance model QCPU is turned ON, is reset, or is switched from STOP to RUN.

##### (2) Designation of the device initial value

In the device initial value setting of GX Developer, designate the special function module device as the device of the device initial value.

#### REMARK

- 1) For the device initial value, see Section 10.13.2.
- 2) For the intelligent function module device, see Section 10.5.

### 8.3.2 Communication using FROM/TO instruction

#### (1) FROM/TO instruction

At the execution of the FROM/TO instruction, the data stored in the buffer memory of the special function module can be read, or data can be written to the buffer memory of the special function module.

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

The TO instruction writes the data of the designated device to the buffer memory of the special function module.

#### REMARK

- 1) For the details of the FROM/TO instruction, refer to the following manuals.
  - QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)
- 2) For the details of the buffer memory of the special function module, refer to the manual of the special function module being used.

### 8.3.3 Communication using the intelligent function module device

#### (1) Intelligent function module device

The intelligent function module device is the buffer memory of the special function module represented as a device of High Performance model QCPU in High Performance model QCPU programs.

It enables reading data stored in the buffer memory of the special function module to be read, or writing data to the buffer memory of the special function module.

#### (2) Difference from the FROM/TO instruction

The intelligent function module device can be handled as a device of High Performance model QCPU, enabling the processing of data read from the special function module with one instruction.

This saves the number of steps in the entire program.

The processing speed is the total of the instruction execution time and the access time to/from the special function modules.

#### POINT

When reading and processing the data of the special function module frequently in the program, use the FROM instruction to read the data at one point in the program and store and process it in a data register, instead of using the intelligent function module device every time.

Otherwise, the intelligent function module device accesses to the special function module every time the instruction is executed, resulting in longer scan time for the program.

#### REMARK

For the intelligent function module device, see Section 10.5.



8.3.4 Effects of quicker access to the special function module and countermeasures against them

(1) Effects of quicker access to the special function module

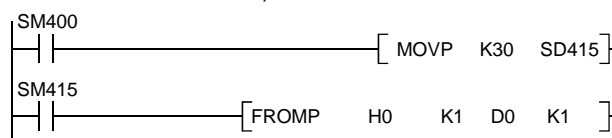
As the scan time of High Performance model QCPU increases, there are some limitations for the execution of the FROM/TO instruction to the AnS-compatible special function module.

The following effects may be produced in the program which reads/writes data from/to a special function module with the FROM/TO instruction at every scan:

- (a) Special function modules which assign priority to the FROM/TO instruction
  - 1) When the FROM/TO instruction is executed, the special function module stops its processing and processes the FROM/TO instruction first. As a result, the processing time of the special function module becomes longer, resulting in watch dog timer error of the special function module.
  - 2) The followings are the special function modules which assign priority to the FROM/TO instruction.
    - A1S64AD, A1S68AD
    - A1S62RD3, A1S62RD4
    - A1S68DAV, A1S68DAI
    - A1S68TD
    - A1SD75P1(-S3), A1SD75P2(-S3), A1SD75P3(-S3)
    - A1SD75M1, A1SD75M2, A1SD75M3
- (b) Special function modules which suspend processing of the FROM/TO instruction
  - 1) Even when the FROM/TO instruction is executed, the special function module continues its processing and suspends processing of the FROM/TO instruction until the processing is completed. As a result, the scan time becomes longer by the period to wait the completion of the processing of the special function module.
  - 2) The following are the special function modules which suspend the FROM/TO instruction.
    - A1S63ADA, A1S66ADA
    - A1SD61, A1SD62, A1SD62D, A1SD62E
    - A1SD70, A1SD71-S2, A1SD71-S7
    - A1SJ71PT32-S3, A1SJ71T32-S3
    - A1SD51S
    - A1SJ71ID1-R4, A1SJ71ID2-R4

(2) Countermeasures against the effects of quicker access to the special function module

To use a special function module with High Performance model QCPU, 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.



**REMARK**

- 1) To change the clock value of SM415, store the new value in SD415.
- 2) For details of SM415, see appendix 1. For details of SD415, see appendix 2.

## 9 PARAMETER

### (1) Parameter types and setting

- (a) The High Performance model QCPU has two different parameter types: the "PLC parameters" set for a system that uses the PLC independently; and the "network parameters" set for use of MELSECNET/H, Ethernet or CC-Link.
- (b) In the High Performance model QCPU of function version B, the following items necessary to configure a multiple CPU system have been added to the PLC parameters. (Refer to Section 14.2.6)
  - Multiple CPU setting
  - Control CPU setting (I/O assignment)
- (c) In this chapter, the PLC parameters and network parameters to be set using GX Developer are listed in a table.  
For details of each setting item, refer to the reference section or reference manual given in the table.  
For the setting operation on GX Developer, refer to the GX Developer Operating Manual.

- (2) When any PLC parameter setting has been changed, power off and then on the PLC (ON → OFF → ON) or reset the QCPU.  
If the High Performance model QCPU is switched from STOP → RUN without any operation after the PLC parameter setting change, the new parameter setting may not become valid or PARAMETER ERROR (error code: 3000) may occur.

Table 9.1 Parameter List

Item		Parameter No.	Description	
PLC name		—	Designate the label and comment for the CPU module to be used. These settings do not affect CPU operation	
Label		0000H	Designates the label setting (name and use).	
Comment		0001H	Designates the comment setting.	
PLC system		—	These are the settings required for using the CPU module. Default values are available for PLC control.	
Timer limit setting	Low Speed	1000H	Designates the low speed/high speed timer settings	
	High Speed			
RUN-PAUSE contacts		1001H	Designates the contact which controls the CPU module's RUN/PAUSE operation.	
Remote reset		1002H	Enables/disables the remote reset operation from the GX Developer.	
Output mode at STOP to RUN		1003H	Designates the output(Y) mode at STOP-RUN switching.	
Floating point arithmetic processing		1004H	Specifies whether to perform floating-point arithmetic processing with double precision.	
Intelligent functional module setting (interrupt pointer setting)		100AH	Specifies the assignment of interrupt pointers (I50 to I255), leading I/O Nos. and leading SI Nos. of an intelligent function module.	
Common pointer		1005H	Designates the common pointer "first No."	
Number of empty slots		1007H	Designates the number of empty slot points in the main/extension base units.	
System interrupt settings	Interrupt counter start No.	1008H	Designates the interrupt counter "first No."	
	Fixed scan interval (n: 28 to31)		Specifies time intervals at which to execute interrupt pointers (I28 to I31).	
	High speed interrupt setting	X input	100FH	Sets the fixed-cycle interval, high speed I/O refresh setting, and high speed buffer transfer setting of the high speed interrupt pointer I49.
		Y output	1010H	
		Buffer read	1011H	
Buffer write	1012H			
Interrupt program/fixed scan program setting		1008H	Specifies whether to perform the high speed execution of an interrupt program.	
Module synchronization		100CH	Specifies whether to bring the start of a CPU module into synchronization with the start of an intelligent function module.	
A series CPU compatibility setting		100DH	Specifies whether to use MELSEC-A Series special relays/special registers (SM1000/SD1000 to SM1299/SD1299).	
PLC file setting		—	Designates the various files used in the CPU module.	
File register		1100H	Designates the file for file registers to be used in the program.	
Comment file used in a command		1101H	Designates the file for comments to be used in the program.	
Initial Device value		1102H	Designates the file for the device initial values to be used in the CPU module.	
File for local device		1103H	Designates the file for local devices to be used in the program.	

	Default Value	Setting Range	Reference Section
	—	—	—
	No setting	Max. of 10 characters	—
	No setting	Max. of 64 characters	—
	—	—	—
	100 ms	1 to 1000 ms(1 ms units)	Section 10.2.10
	10.0 ms	0.1 to 100.0 ms	Section 10.2.10
	No setting	X0 to X1FFF	Section 7.6.1
	Disabled	Enabled/Disabled	Section 7.6.3
	Previous status (produce the status of an output (X) before STOP	Produce the status of an output (X) before STOP/Clear the output (output is 1 scan later)	Section 7.4
	Perform internal arithmetic operation with double precision	Check/Not Checked to perform internal arithmetic operation with double	Section 4.8.4
	No setting	I50 to I255, leading I/O No, leading SI No.	Section 10.10
	No setting	P0 to P4095	Section 10.9.2
	16 points	0/16/32/64/128/256/512/1024 points	Section 5.6.1
	No setting	C0 to C22722 (Counter setting points can be set up to 256.)	Section 10.2.11
	I28: 100.0 ms I29: 40.0 ms I30: 20.0 ms I31: 10.0 ms	0.5 to 1000 ms (0.5 ms units)	Section 10.10
	No setting	I49 : 0.2 to 1.0 ms (0.1ms units)	Section 7.21
	The high speed execution is disabled.	Enable/Disable the high speed execution.	Section 4.1.3 Section 4.2.5
	The start of an intelligent function module is synchronized.	Yes/No to synchronize the start of an intelligent function module.	—
	Special relays/special registers after SM1000/SD1000 are used.	Yes/NO to use the special relays/special registers after SM1000/SD1000.	Section 10.3.2 Section 10.3.3
	—	—	—
	Not used	<ul style="list-style-type: none"> <li>• Not used</li> <li>• Use the same file name as the program</li> <li>• Use the designated file</li> </ul>	Section 10.7
	Not used	<ul style="list-style-type: none"> <li>• Not used</li> <li>• Use the same file name as the program</li> <li>• Use the following file</li> </ul>	—
	Not used	<ul style="list-style-type: none"> <li>• Not used</li> <li>• Use the same file name as the program</li> <li>• Use the following file</li> </ul>	Section 10.13.2
	Not used	<ul style="list-style-type: none"> <li>• Not used</li> <li>• Use the following file</li> </ul>	Section 10.13.1

Table 9.1 Parameter List (continued)

Item	Parameter No.	Description
PLC RAS setting	—	These settings are used for the RAS function.
WDT setting (Watch dog timer)	3000H	WDT setting
		Initial execution watching time
		Low speed execution watching time
Operating mode when there is an error	3002H	Designates the CPU module operation mode to be established when an error is detected.
Error check	3001H	Designates whether or not to detect a specified error .
Constant scanning	3003H	Designates the constant scanning time.
Low speed program execution time	3006H	Designates the time setting for low speed execution program execution at each scanning.
Fault history	3005H	Designates the storage designation for the CPU module fault history.
Device setting	—	These settings designate the number of points for each device, the latch range, and the local device range.
Device point	2000H	Designates the number of device points used.
Latch(1) head/end (Latch clear key enabled)	2001H	Designates the latch range where the latch clear key is enabled.
Latch(2) head/end (Latch clear key disabled)	2002H	Designates the latch range where the latch clear key is disabled.
Local device head/end	2003H	Designates the device range used for local devices.
Program setting		Specifies a program name and execution conditions to write several programs onto the CPU module.
Boot file setting	7000H	Boot option
		Boot file setting
		Automatic refresh to standard ROM
SFC setting	8002H	Designates the SFC program start mode, starting conditions, and the output mode in a block stop for SFC program use.
	8003H	
	8005H	

	Default Value	Setting Range	Reference Section
	—	—	—
	200 ms	10 to 2000 ms (10 ms units)	Section 4.2.2
	No setting	10 to 2000 ms (10 ms units)	Section 4.2.1
	No setting	10 to 2000 ms (10 ms units)	Section 4.2.3
	Stop	Stop/Continue	Section 7.16
	Checked	Checked/Not checked	Section 7.16
	No setting	0.5 to 2000 ms(0.5 ms units)	Section 7.2
	No setting	1 to 2000 ms	Section 4.2.3
	Record in PLC RAM	Record in PLC RAM/Record in the following history file	Section 7.17
	—	—	—
	X: 8 k points Y: 8 k points M: 8 k points L: 8 k points B: 8 k points F: 2 k points SB: 2 k points V: 2 k points S: 8 k points T: 2 k points ST: 0 k point C: 1 k point D: 12 k points W: 8 k points SW: 2 k points	X(8 k points), Y(8 k points), S(8 k points), SB(2k points) and SW(2 k points) are fixed. Including the above points(3.7 k words), a total range of 29 k words is available. • For one device: Max. 32 k points • Total number for the bit devices: Max. 64 k points	Section 10.1 Section 10.2
	No setting	Only 1 range is designated for each device of B, F, V, T, ST, C, D, W.	Section 7.3
	No setting	Only 1 range is designated for each device of L, B, F, V, T, ST, C, D, W.	Section 7.3
	No setting	Only 1 range is designated for each device of M, V, T, ST, C, D.	Section 10.13.1
	No setting	Program name, execution type (fixed scan for fixed scan execution), file use setting, I/O refresh setting	Section 4.2
	Do not clear the program memory during boot.	Do not clear the program memory during boot. / Clear the program memory during boot.	Section 6.6.2
	No setting	Type, data name and source drive (The destination drive is automatically set in the program memory.)	Section 6.6
	Do not execute automatic refresh to the standard ROM.	Do not execute automatic refresh to the standard ROM. / Do not execute automatic refresh to the standard ROM.	Section 6.6.2
	—	See the QCPU (Q mode) / QnACPU Programming Manual (SFC).	—

Table 9.1 Parameter List (continued)

Item		Parameter No.	Description
I/O assignment setting		—	Designates the status of installation of each module of the system.
I/O Assignment	Type	400H	Designates the type of the installed module.
	Model name		Designates the model of the installed module. (Memorandum for users who do not use the CPU module.)
	Points		Designates the number of points of each slot.
	Head XY (Head I/O number)		Designates the first input and output numbers of each slot.
Standard setting	Base model name	401H	Designates the model of the used main base unit and extension base unit. (Memorandum for users who do not use the CPU module.)
	Power module model name		Designates the model of the power supply module installed to the main and extension base units. (Memorandum for users who do not use the CPU module)
	Extension cable model name		Designates the model of the extension cable. (Memorandum for users who do not use the CPU module)
	Number of slots		Designates the number of slots of the main and extension base units. The number of slots is designated for each base unit.
Switch setting		407H	Designates various switches of the intelligent function module.
Detailed setting	Error time output module	403H	Designates whether the output is cleared or retained upon a stopping error of the control PLC.
	H/W error time CPU operation mode	4004H	Designates whether the control PLC continues operation or it is stopped upon a hardware error of the intelligent function module.
	I/O response time	405H	Designates the response time of the input module, high speed input module and I/O mixture module.
	Control CPU	406H	Designates the control PLC of the I/O module and intelligent function module.
Acknowledge XY assignment		—	Contents of I/O allocation, MELSECNET/Ethernet setting and CC-Link setting can be checked.
Multiple CPU setting		—	Defines settings for establishment of a multiple CPU system.
No. of CPU		E00H	Designates the number of CPUs used in the multiple CPU system.
Operating mode		E01H	Designates the operation of the multiple CPU system upon a stopping error of the PLC No.2 to No.4 CPU modules. The multiple CPU system is stopped if a stopping error occurs to the PLC No.1 (Fixed)
Online module change setting		E006H	Designates whether online module change will be enabled or disabled for other CPU modules. (When it is enabled, the out of group I/O status cannot be imported.)
Out of group input setting		E04H	Designates whether the input status of the input module and intelligent function module controlled by other PLCs are acquired or not.
Out of group output setting			Designates whether the output status of the output module controlled by other PLCs are acquired or not.
Refresh settings		E002H E003H	Designates the devices and the number of points for which data write/read will be performed by automatic refresh between CPU modules in the multiple CPU system.

	Default Value	Setting Range	Reference Section
	—	—	—
	No setting	<ul style="list-style-type: none"> <li>• PLC CPU No.2 to No.4: PLC No.n/Empty (Designate "CPU (empty)" for slots where no CPU module is installed.)</li> <li>• Input/output module and intelligent function module</li> <li>• Input, high speed input, output, intelligent, input/output mixture, interrupt</li> </ul>	Section 5.6
	No setting	<ul style="list-style-type: none"> <li>• 16 single-byte characters</li> </ul>	
	No setting	<ul style="list-style-type: none"> <li>• 0 point, 16 points, 32 points, 48 points, 64 points, 128 points, 256 points, 512 points, 1024 points</li> </ul>	
	No setting	<ul style="list-style-type: none"> <li>• 0H to FF0H</li> </ul>	
	No setting	<ul style="list-style-type: none"> <li>• 16 single-byte characters</li> </ul>	Section 5.3
	No setting	<ul style="list-style-type: none"> <li>• 16 single-byte characters</li> </ul>	
	No setting	<ul style="list-style-type: none"> <li>• 16 single-byte characters</li> </ul>	
	No setting	<ul style="list-style-type: none"> <li>• 2,3,5,8,10,12</li> </ul>	
	No setting	<ul style="list-style-type: none"> <li>• See the manual of the intelligent function module to be used.</li> </ul>	Section 7.6
	Clear	<ul style="list-style-type: none"> <li>• Cleared/retained</li> </ul>	Section 7.8
	Stop	<ul style="list-style-type: none"> <li>• Stopped/continue</li> </ul>	—
	Input, I/O mixture: 10 ms High speed input: 0.2 ms	<ul style="list-style-type: none"> <li>• Input, I/O mixture: 1 ms, 5 ms, 10 ms, 20 ms, 70 ms</li> <li>• High speed input: 0.1 ms, 0.2 ms, 0.4 ms, 0.6 ms, 1.0 ms</li> </ul>	Section 7.7
	PLC No.1	<ul style="list-style-type: none"> <li>• PLC No.1, PLC No.2, PLC No.3, No.4</li> </ul>	Section 14.2.1
	—	—	—
	—	—	—
	1 module	<ul style="list-style-type: none"> <li>• 1 to 4 modules</li> </ul>	Section 14.2.1
	Stop all CPUs upon error of PLC No.n	<ul style="list-style-type: none"> <li>• Stop or do not stop all PLCs upon an error of PLC No.n.</li> </ul>	Section 14.2.8
	Online module change is disabled for other CPU modules	<ul style="list-style-type: none"> <li>• Online module change is enabled/disabled for other CPU modules.</li> </ul>	—
	Do not permit inputs from outside group	<ul style="list-style-type: none"> <li>• Permit or do not permit inputs from outside the group.</li> </ul>	Section 17.2
	Do not permit outputs to outside group	<ul style="list-style-type: none"> <li>• Permit or do not permit outputs to outside the group.</li> </ul>	Section 17.2
	No setting	<ul style="list-style-type: none"> <li>• Setting range of each CPU: 0 to 2048 points (in 2-point intervals) / module Max. 4k points (4096 points) / system</li> <li>• Device on CPU side: B, M, Y, D, R, ZR Devices equivalent to the number of points set for the transmission range from the designated device number are occupied.                             <ul style="list-style-type: none"> <li>• 16 points are occupied with B, M and Y for each point of transmission range.</li> <li>• 1 point is occupied with D, W, R and ZR for each point of transmission range.</li> </ul> </li> </ul>	Section 16.1



Table 9.1 Parameter List (continued)

Item	Parameter No.	Description
Network parameter		Designates parameters for MELSECNET/H, Ethernet and CC-Link.
MELSECNET/H setting * 1		Designates network parameters for MELSECNET/H.
No. of module setting	5000H	
Valid module during other station access	5001H	
Interlink transmission parameter	5002H	
Routing parameters	5003H	
Network setting	5NM0H	
Network refresh parameters	5NM1H	
Common parameter	5NM2H	
Station inherent parameter	5NM3H	
Common parameter No.2	5NMAH	
Station inherent parameter No.2	5NMBH	
Interrupt setting		
Ethernet setting * 1		Designates network parameters for Ethernet.
No. of module setting	9000H	
Starting I/O No.	9N00H	
Network No.		
Operational settings		
Initial settings		
Open settings		
Routing information		
MNET/10 routing information		
FTP Parameters		
E-mail setting		
Interrupt setting	9N09H	

	Default Value	Setting Range	Reference Section
	—	—	—
	No setting	<ul style="list-style-type: none"> <li>• Refer to the Q Corresponding MELSECNET/H manual.</li> </ul>	—
	No setting	<ul style="list-style-type: none"> <li>• Refer to the Q Corresponding Ethernet manual.</li> </ul>	—

Table 9.1 Parameter List (continued)

Item	Parameter No.	Description	
Network parameter		Designates parameters for MELSECNET/H, Ethernet and CC-Link.	
CC-Link setting * 2		Designates network parameters for CC-Link.	
No. of module setting	C000H		
Remote input (RX) Refresh device	CNM1H		
Remote output (RY) Refresh device			
Remote register (RWr) Refresh device			
Remote register (RWw) Refresh device			
Ver.2 Remote input (RX) Refresh device			
Ver.2 Remote output (RY) Refresh device			
Ver.2 Remote register (RWr) Refresh device			
Ver.2 Remote register (RWw) Refresh device			
Special relay (SB) Refresh device			
Special register (SW) Refresh device			
Start I/O No.			CNM2H
All connect count			
Retry count			
Automatic reconnection station count			
Stand by master station No.			
CPU down specification			
Scan mode specification			
Delay time setting			
Station information setting			
Remote device station initial setting			
Interrupt setting			
Remote password			
Password setting	—	Enter the remote password.	
Password valid module setting	Model name	—	Select the module model name to be checked for the remote password set to the QCPU.
	Head X/Y	—	Set the head address of the module to be checked for the remote password.
Detailed setting	—	—	
User connection No. valid setting	—	Set the user connection No.	
System connection valid setting	—	Set the remote password valid port of the system connection.	

	Default Value	Setting Range	Reference Section
	—	—	—
	No setting	• Refer to the CC-Link manual.	—
	No	4 bytes, alphanumeric characters, special symbols	
	No	QJ71E71/QJ71C24	
	No	0000 <sub>H</sub> to 0FE0 <sub>H</sub>	
	—	—	
	No	Connection No. 1 to connection No. 16	
	No	Specify the remote password valid port. <ul style="list-style-type: none"> <li>• Automatic open UDP port</li> <li>• FTP communication port (TCP/IP)</li> <li>• GX Developer communication port (TCP/IP)</li> <li>• GX Developer communication port (UDP/IP)</li> <li>• HTTP port</li> </ul>	

\*1: N and M indicate the following.

N: Indicates the module number.

M: Indicates the network type.

M	Network Type
1H	MELSECNET/10 mode (Control station), MELSECNET/H mode (Control station)
2H	MELSECNET/10 mode (Normal station), MELSECNET/H mode (Normal station)
5H	MELSECNET/H (Remote master)
AH	MELSECNET/H Stand by station

\*2: N and M indicate the following.

N: Indicates the module number.

M: Indicates the network type.

M	Network Type
0H	Master station
1H	Local station
2H	Standby master station

## 10 EXPLANTION OF DEVICES

This chapter describes all devices that can be used in the High Performance model QCPU.

### 10.1 Device List

The names and data ranges of devices which can be used in the High Performance model QCPU are shown in Table 10.1 below.

Table 10.1 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 * 3	8192 points	X0 to X1FFF	Changeable within 29k words. * 3	Section 10.2.1
		Output * 3	8192 points	Y0 to Y1FFF		Section 10.2.2
		Internal relay	8192 points	M0 to M8191		Section 10.2.3
		Latch relay	8192 points	L0 to L8191		Section 10.2.4
		Anunciator	2048 points	F0 to F2047		Section 10.2.5
		Edge relay	2048 points	V0 to V2047		Section 10.2.6
		Step relay * 3	8192 points	S0 to S511 / block		Section 10.2.9
		Link special relay * 3	2048 points	SB0 to SB7FF		Section 10.2.8
	Word devices	Link relay	8192 points	B0 to B1FFF		Section 10.2.7
		Timer * 1	2048 points	T0 to T2047		Section 10.2.10
		Retentive timer * 1	0 points	(ST0 to ST2047)		
		Counter * 1	1024 points	C0 to C1023		Section 10.2.11
		Data register	12288 points	D0 to D12287		Section 10.2.12
		Link register	8192 points	W0 to W1FFF		Section 10.2.13
		Link special register * 1	2048 points	SW0 to SW7FF		Section 10.2.14
Internal system devices	Bit devices	Function input	5 points	FX0 to FXF * 4	Unchangeable	Section 10.3.1
		Function output	5 points	FY0 to FYF * 4		Section 10.3.1
		Special relay	2048 points	SM0 to SM2047		Section 10.3.2
	Word devices	Function register	5 points	FD0 to FD4		Section 10.3.1
		Special register	2048 points	SD0 to SD2047		Section 10.3.3
Link direct devices	Bit device	Link input	8192 points	Jn\X0 to Jn\X1FFF	Unchangeable	Section 10.4
		Link output	8192 points	Jn\Y0 to Jn\Y1FFF		
		Link relay	16384 points	Jn\B0 to Jn\B3FFF		
		Link special relay	512 points	Jn\SB0 to Jn\SB1FF		
	Word device	Link register	16384 points	Jn\W0 to Jn\W3FFF		
		Link special register	512 points	Jn\SW0 to Jn\SW1FF		

Class	Type	Device Name	Default Values		Parameter Designated Setting Range	Reference Section
			Number of Points	Range Used		
Intelligent function module device	Word device	Buffer register	65536 points	Un\G0 to Un\G65535 * 2	Unchangeable	Section 10.5
Index register	Word device	Index register	16 points	Z0 to Z15	Unchangeable	Section 10.6
File register	Word device	File register	0 points	—	0 to 1018 k points (1 k units)	Section 10.7
Nesting	—	Nesting	15 points	N0 to N14	Unchangeable	Section 10.8
Pointers	—	Pointer	4096 points	P0 to P4095	Unchangeable	Section 10.9
		Interrupt pointer	256 points	I0 to I255		Section 10.10
Other	Bit devices	SFC block device	320 points	BL0 to BL319	Unchangeable	Section 10.11.1
		SFC transition device	512 points	TR0 to TR511		Section 10.11.2
	—	Network No	256 points	J1 to J255		Section 10.11.3
		I/O No	—	U0 to UFF		Section 10.11.4
		Macro instruction argument device	—	VD0 to VD□		Section 10.11.5
Constants	—	Decimal constants	K-2147483648 to K2147483647		Section 10.12.1	
		Hexadecimal constants	H0 to HFFFFFFF		Section 10.12.2	
		Real number constants	E±1.17550-38 to E±3.40282+38		Section 10.12.3	
		Character string constants	"ABC" and "123"		Section 10.12.4	

**REMARK**

- \*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.  
For details regarding the buffer memory's "number of points", refer to the Intelligent/Special Function Module Manual.
- \*3: Inputs, outputs, step relays, link special relays, link special registers remain at their default values, which cannot be changed.
- \*4: In a program, only FX0 to FX4 and FY0 to FY4 can be used.

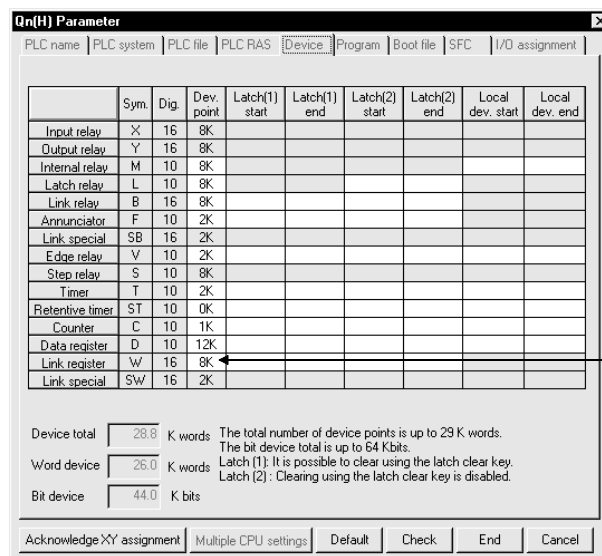
10.2 Internal User Devices

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.

[Device setting screen]



Default value  
 "Dev. point" can be changed for the device where a "Dev. point" value is shown in brackets.

(1) Setting range in the internal user device

The number of points for internal user devices other than the input(X), output(Y), step relay(S), link special relay(SB), and link special registers(SW) can be changed within a 29 k words (including 3.7 k words for an internal user device) range, at the "Device" tab screen in the "(PLC) Parameter" dialog box. The following gives more information.

(a) Setting range

- 1) The number of device points is designated in 16-point units.
- 2) A maximum of 32 k points can be designated for one device.  
 The maximum total number of points for the internal relay, latch relay, annunciator, edge relay, link relay, link special relay, step relay, timer, retentive timer, and counter, is 64 k points.  
 1 point is calculated as 2 points (1 for coil, 1 for contact) for the timer, retentive timer, and counter.

(2) Memory capacity

Use the following expression to obtain the memory capacity of an internal user device.

$$3.7 + (\text{Bit devices capacity}) + (\text{Word devices capacity}) + (\text{Timer, retentive timer and counter capacity}) \leq 29k$$

(a) For bit devices:

For bit devices, 16 points are calculated as 1 word.

$$(\text{Bit device capacity}) = \frac{(M+L+F+V+B \text{ total number of points})}{16} (\text{Word})$$



- (b) For timer (T) retentive timer (ST), and Counter (C):

For the timer, retentive timer, and counter, 16 points are calculated as 18 words.

$$\text{(Timer, retentive, counter capacity)} = \frac{\text{(T, ST, C total number of points)}}{16} \times 18 \text{ (Word)}$$

- (c) For word devices:

For data registers (D) and link registers (W), 16 points are calculated as 16 words.

$$\text{(Word device capacity)} = \frac{\text{(D, W total number of points)}}{16} \times 16 \text{ (Word)}$$

#### POINT

- (1) 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.

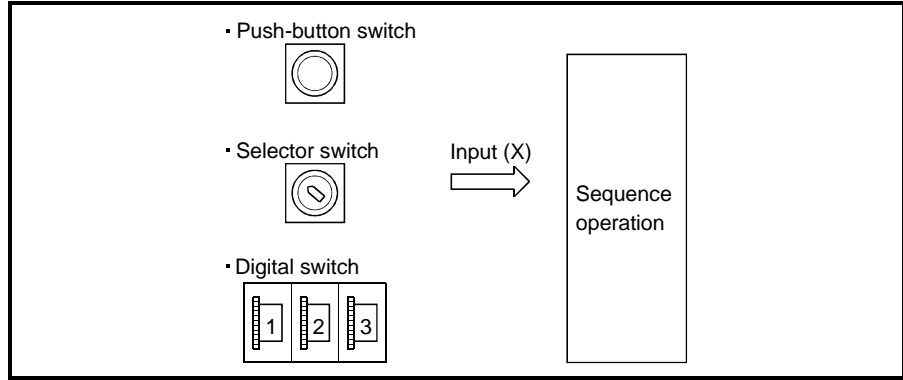
- The sequence program
- The SFC program

After changing the setting, the sequence program and SFC program must be read from the High Performance model QCPU to GX Developer, and then they must be written back to the High Performance model QCPU again.

10.2.1 Inputs (X)

(1) Definition

- (a) 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.



- (b) 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.

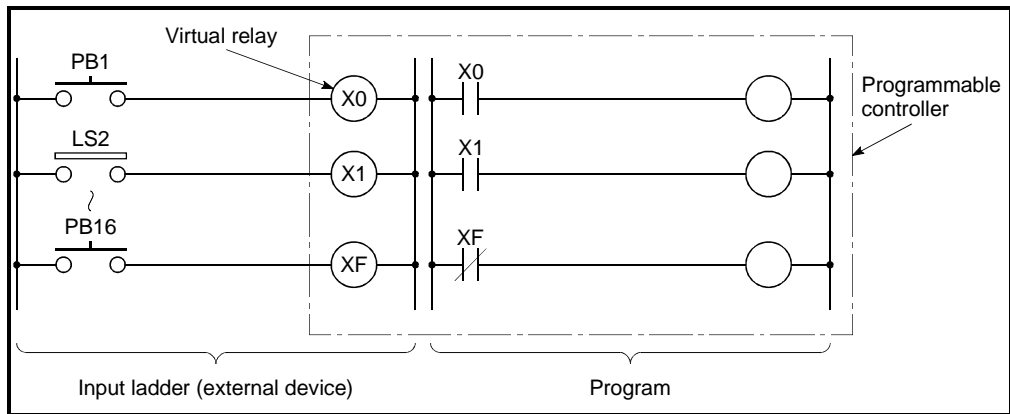


Figure 10.1 Inputs(X)

- (c) 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.

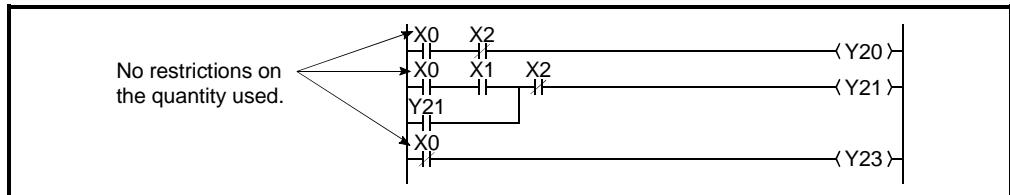
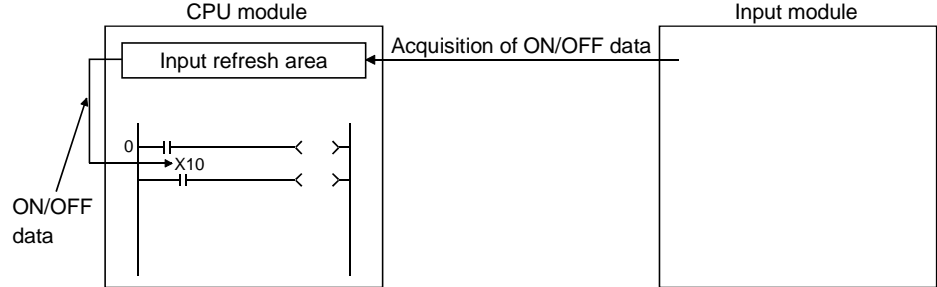


Figure 10.2 Input(X) Used in Program

(2) Reading the inputs

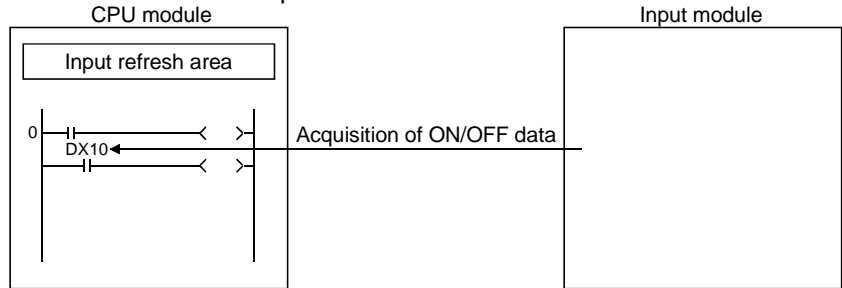
(a) There are 2 types of input: "refresh inputs" and "direct access inputs".

1) The refresh input executes an operation with ON/OFF data. The data is read by performing an input refresh before the sequence programs is executed. \*1



The refresh input is indicated as "X[]" in the sequence program. For example, a "10" input becomes "X10".

2) The direct access input executes an operation with ON/OFF data. The data is read from an input module when the instruction is executed. \*2



The direct access is indicated as "DX[]" in the sequence program. For example, a "10" input becomes "DX10".

The direct access input can be made in a LD/AND/OR instruction that uses an input in units of 1 point.

(b) Differences between refresh input and direct access input

The direct access input accesses an input module directly when an instruction is executed, which results in slower processing speed compared with the refresh input.

The direct access input is used only for inputting to the input module or the intelligent function module/special function module mounted on the main base unit or extension base unit.

The refresh input and direct access input differences are shown in Table 10.2 below.

Table 10.2 Differences Between Refresh

Item		Refresh Input	Direct Access Inputs
Processing speed (LD X/DX)	Q02CPU	0.079 μs	Main base unit : 8.0 μs Extension base unit : 8.0 μs
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.034 μs	Main base unit : 4.0 μs Extension base unit : 4.8 μs
Input module mounted on base/extension base unit		Usable	Usable
Input of intelligent function module mounted on base/extension base unit			
Input of I/O link module mounted on extension base unit			
Input used at MELSECNET/H network system or CC-Link system		Usable	Unusable

**REMARK**

\*1: See Section 4.7.1 for details on the refresh mode.

- (c) The same input number can be designated for a refresh input and a direct access input.  
 If the number is used as a refresh input after being used as a direct access input, the operation is executed with the ON/OFF data read by performing a direct access input.

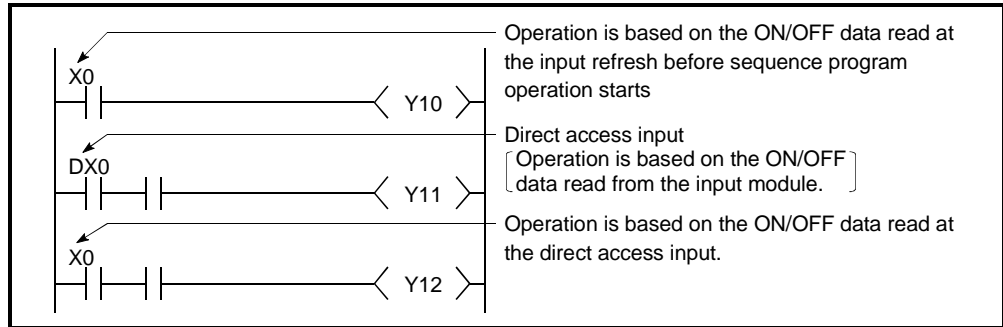


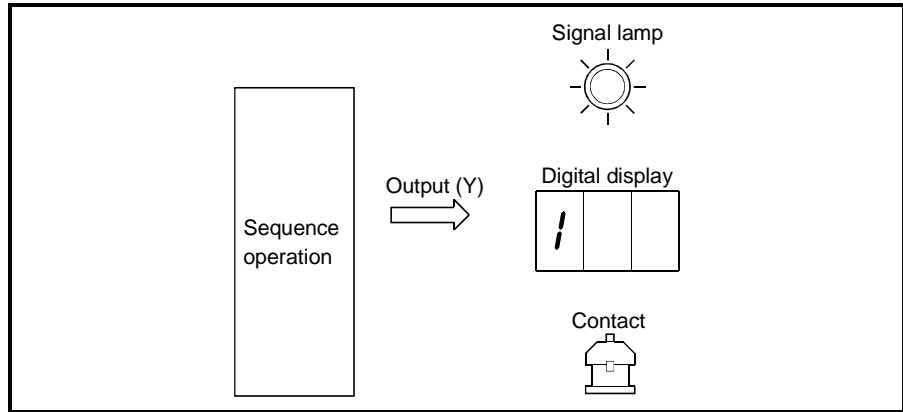
Figure 10.3 Refresh Input & Direct Access Input

POINT
<p>(1) When debugging a program, an input (X) can be set to ON/OFF as described below.</p> <ul style="list-style-type: none"> <li>• OUT Xn instruction</li> </ul> <div style="text-align: center; margin: 10px 0;"> </div> <ul style="list-style-type: none"> <li>• GX Developer test operation</li> </ul> <p>(2) An input (X) can be used in the following cases.</p> <ul style="list-style-type: none"> <li>• Destination device for RX refresh of CC-link</li> <li>• Destination device for refresh of link input of MELSECNET/H</li> </ul>

10.2.2 Outputs (Y)

(1) Definition

- (a) Outputs give out the program control results to the external devices such as solenoid, electromagnetic switch, signal lamp and digital display.



- (b) Outputs give out the result equivalent to one N/O contact.
- (c) 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.

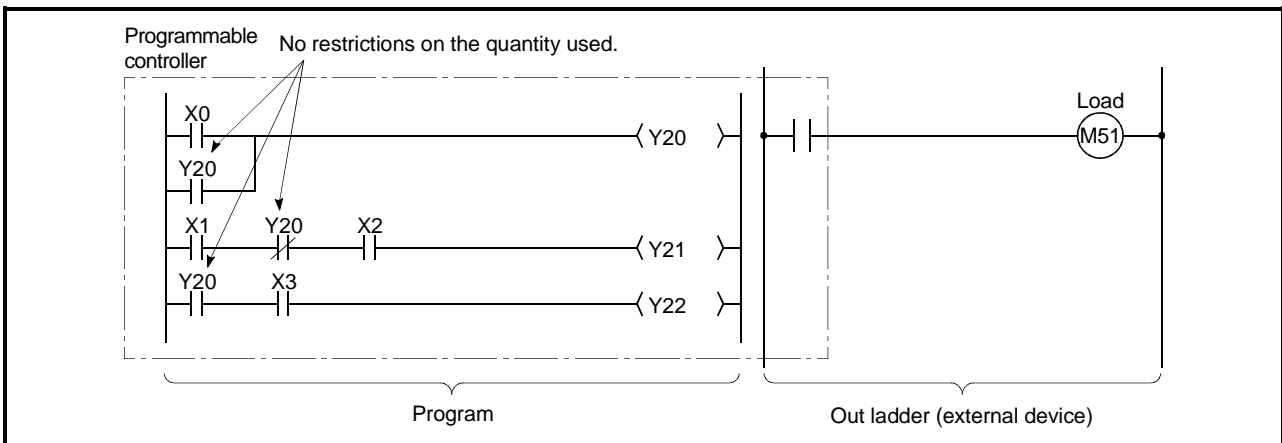
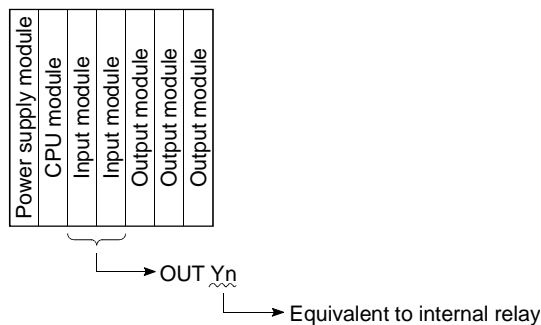


Figure 10.4 Output(Y)

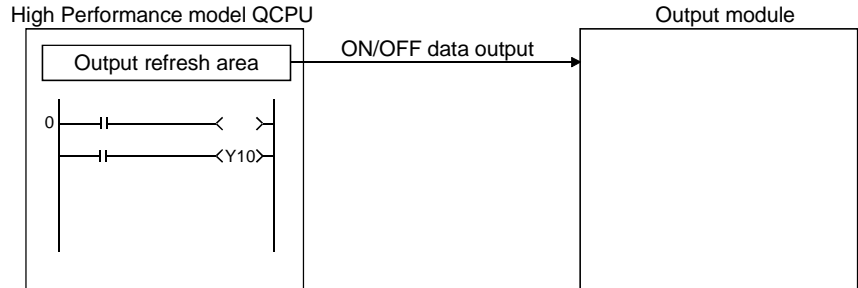
(2) Using outputs as internal relays (M)

"Y" corresponding to the slots installed with input modules and empty slots can serve as internal relays (M).



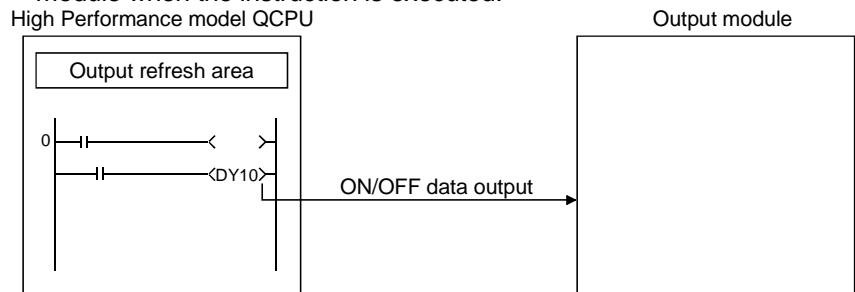
(3) Output method

- (a) There are 2 types of output: "refresh outputs" and "direct access outputs".  
 1) The refresh output gives out the ON/OFF data to an output module by performing an output refresh before the sequence program is executed. \*1



The refresh output is indicated as "Y[]" in the sequence program. For example, a "10" output becomes "Y10".

- 2) The direct access output gives out the ON/OFF data to an output module when the instruction is executed.



The direct access output is indicated as "DY[]" in the sequence program. For example, a "10" output becomes "DY10".

- (b) Differences between refresh output and direct access output  
 The direct access output accesses an output module directly when an instruction is executed, which realizes shorter external output time. However, it processes an instruction slower compared with the refresh output.  
 The direct access output is used only for outputting to the output module or the intelligent function module/special function module mounted on the base unit or extension base unit.  
 The refresh and direct output differences are shown in Table 10.3 below.

Table 10.3 Differences Between Refresh Outputs & Direct Access Outputs

Item		Refresh Input	Direct Access Outputs
Processing speed (OUT Y/DY)	Q02CPU	0.158 μs	Main base unit : 8.0 μs Extension base unit : 8.8 μs
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.068 μs	Main base unit : 4.0 μs Extension base unit : 4.8 μs
Output module installed at base/extension base unit		Usable	Usable
Outputs of intelligent function module installed at base/extension base unit			
Outputs of I/O link module installed at extension base unit			
Outputs used at MELSECNET/H network system or CC-Link system		Usable	Unusable

**REMARK**

\*1: See Section 4.7.1 for details on the refresh mode.

10.2.3 Internal relays (M)

(1) Definition

(a) Internal relays are auxiliary relays which cannot be latched by the programmable controller's internal latch (memory backup).

All internal relays are switched OFF at the following times:

- When power is switched from OFF to ON.
- When reset occurs.
- When latch clear operation is executed.

(b) 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.

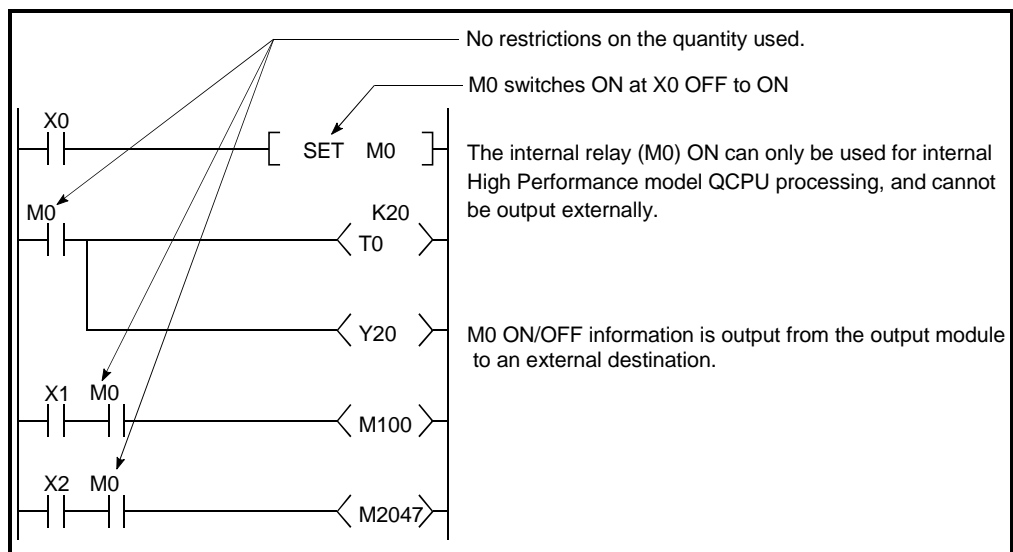


Figure 10.5 Internal Relay

(2) Procedure for external outputs

Outputs (Y) are used to output sequence program operation results to an external destination.

**REMARK**

Latch relays (L) should be used when a latch (memory backup) is required. See Section 10.2.4 for details on latch relays.

10.2.4 Latch relays (L)

(1) Definition

(a) 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 power is switched from OFF to ON.
- When reset occurs.

The latch is backed up by the High Performance model QCPU battery.

(b) Latch relays can be switched OFF by performing latch clear for the High Performance model QCPU. However, the latch relay set as "Latch (2): Cannot clear with Latch Clear key" at the "Device" tab screen in the "(PLC) Parameter" dialog box cannot be turned off, even when the RESET/L.CLR switch/remote latch clear is made for latch clear of it.

(c) 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.

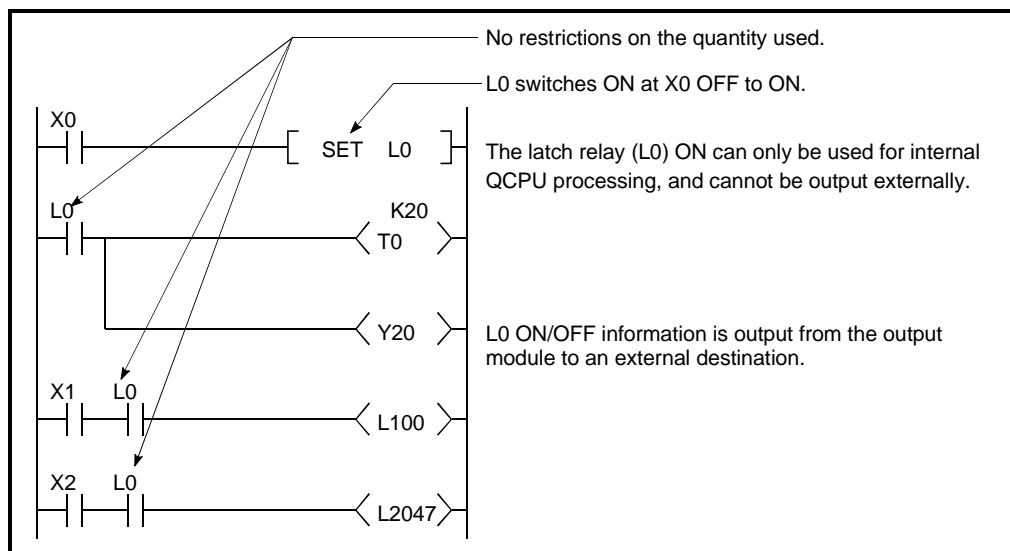


Figure 10.6 Latch Relay

(2) Procedure for external outputs

Outputs (Y) are used to output sequence program operation results to an external destination.

**REMARK**

Internal relays (M) should be used when a latch (memory backup) is not required. See Section 10.2.3 for details on internal relays.



10.2.5 Anunciators (F)

(1) Definition

(a) Anunciators are internal relays used for fault detection programs created by the user.

(b) 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 SD79).

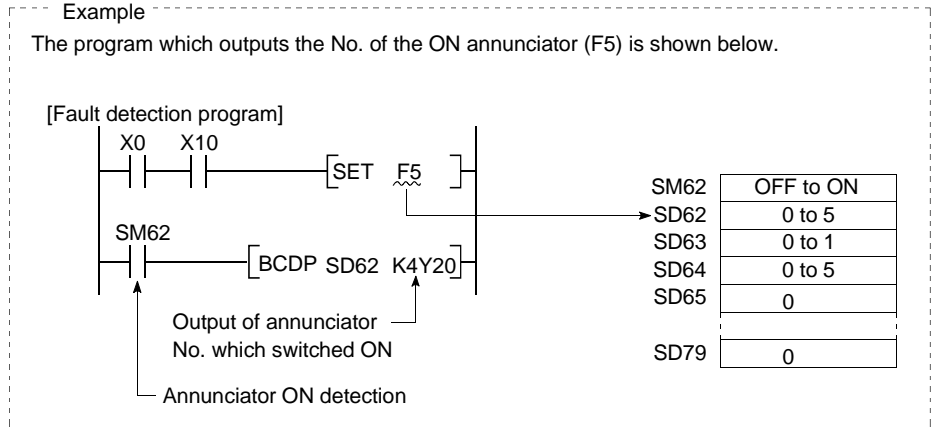
- Special relay :SM62..... :Switches ON if even one annunciator switches ON.
- Special register :SD62 ..... No. of first annunciator which switched ON is stored here.  
SD63 ..... The number (quantity) of annunciators which are ON is stored here.  
SD64 to SD79 ..... Annunciator Nos. are stored in the 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 "fault history area".

(c) 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 SD79) when the special relay (SM62) switches ON.

Example

The program which outputs the No. of the ON annunciator (F5) is shown below.



(2) Anunciator ON procedure

(a) Anunciator ON procedure

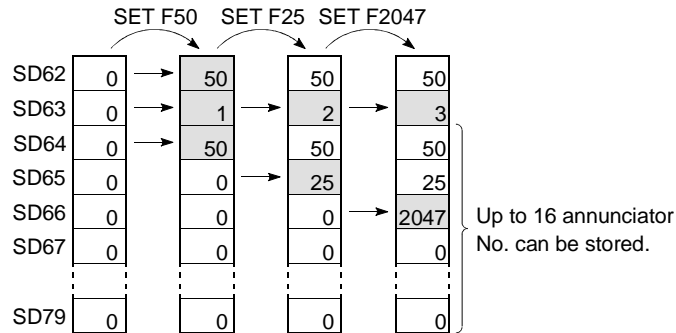
Anunciator operation can be controlled by the SET F□ and OUT F□ instructions.

- 1) The SET F□ instruction switches the annunciator ON only at the leading edge (OFF to ON) of the input condition, and keeps the annunciator ON when the input condition switches OFF.  
In cases where many annunciators are used, the OUT F□ instruction can be used to speed up the scan time.
- 2) The OUT F□ instruction can switch the annunciator ON or OFF. It takes longer to do so than the SET F□ instruction. If the annunciator is switched OFF by using an OUT F□ instruction, this will require the execution of an RST F□ or LEDR instruction. Use a SET F□ instruction to switch the annunciator ON.

<b>POINT</b>	<p>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 SD79.)</p>
--------------	--

(b) Processing at annunciator ON

- 1) Data stored at special registers (SD62 to SD79)
  - a) Nos. of annunciators which switched ON are stored in order at SD64 to SD79.
  - b) The annunciator No. which was stored at SD64 is stored at SD62.
  - c) "1" is added to the SD63 value.



- 2) Processing at CPU
  - "USER" LED at High Performance model QCPU front is ON.
- 3) Setting the display priority at error occurrence to SD207 - SD209 allows you to select whether the ERR. LED is to be ON or OFF when the annunciator turns ON.

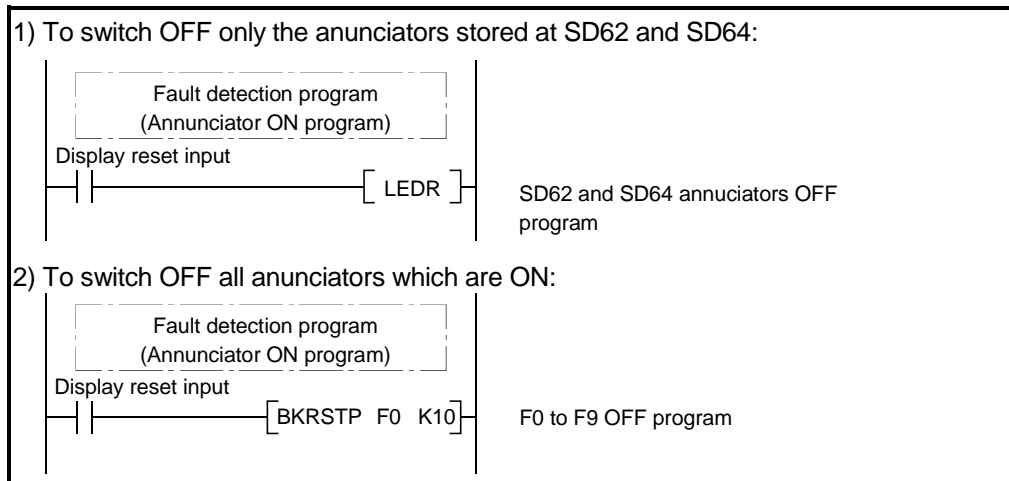
(3) Annunciator OFF procedure and processing content

(a) Annunciator OFF procedure

An annunciator can be switched OFF by the RST F□, LEDR, BKRST, and OUT F□ instructions.

- 1) An annunciator No. which has been switched ON by the SET F□ instruction can be switched OFF by the RST F□ instruction.
- 2) The LEDR instruction is used to switch OFF the annunciator Nos. stored at SD62 and SD64.
- 3) The BKRST instruction is used to switch all the annunciator Nos. within a specified range.
- 4) 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" (item (b) below) is not performed. Execute the RST F□, LEDR or BKRST instructions after the annunciator has been switched OFF by the OUT F□ instruction.



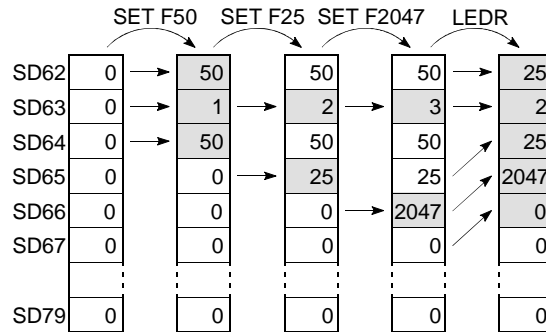
**REMARK**

For details on the LEDR and BKRST instruction, refer to the QCPU(Q mode)/QnACPU Programming Manual(Common Instructions).

(b) Processing at annunciator OFF

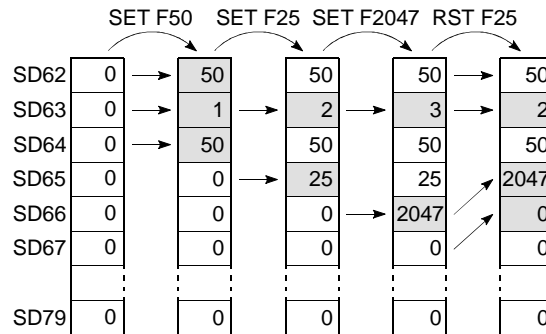
1) Special register (SD62 to SD79) data operation at execution of LEDR instruction

- a) Annunciator No. stored at SD64 is deleted, and annunciator Nos. stored at subsequent registers (SD65 to SD79) are moved up to fill the empty space.
- b) The annunciator No. stored at SD64 is stored at SD62.
- c) "-1" is subtracted from the SD63 value.
- d) If the SD63 value is "0", SM62 is switched OFF.



2) Special register (SD62 to SD79) data operation when an annunciator is switched OFF by the RST F□ instruction

- a) The annunciator No. which was switched OFF is deleted, and all subsequent annunciator Nos. are moved up to fill the empty space.
- b) If the annunciator No. stored at SD64 was switched OFF, the new annunciator No. which is stored at SD64 is stored at SD62.
- c) "-1" is subtracted from the SD63 value.
- d) If the SD63 value is "0", SM62 is switched OFF.



3) Processing by High Performance model QCPU

If all SD64 to SD79 annunciator Nos. are switched OFF, the "USER" LED on the High Performance model QCPU front display is switched OFF.

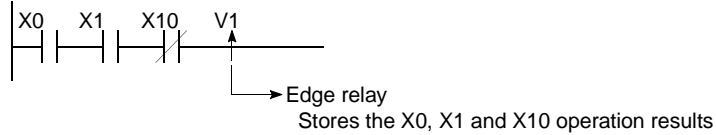
**POINT**

If an error occurs to continue operation with the higher-priority over an annunciator when the annunciator is switched ON, eliminate the error by executing an LEDR instruction. See Section 7.20.2 for priority. In this case, executing an LEDR instruction will not switch the annunciator OFF. To switch the annunciator OFF, you must first eliminate the error before executing the LEDR instruction because the error takes priority over the annunciator.

10.2.6 Edge relay (V)

(1) Definition

- (a) 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.

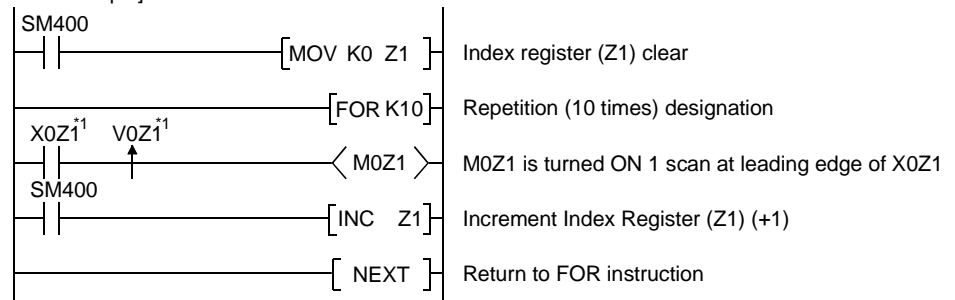


- (b) The same edge relay number cannot be used twice in programs executed by the High Performance model QCPU.

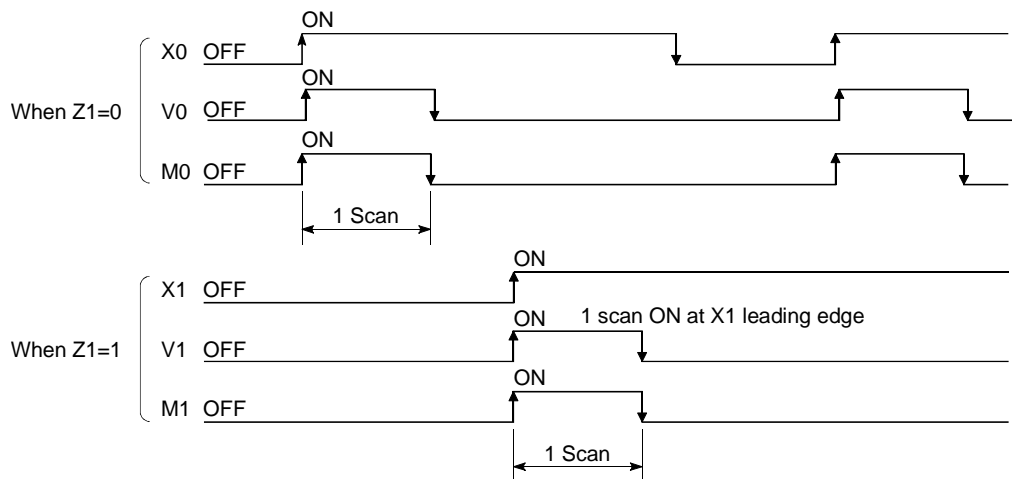
(2) Edge relay applications

Edge relays are used for detecting the leading edge (OFF to ON) in programs configured using index modification.

[Ladder example]



[Timing chart]



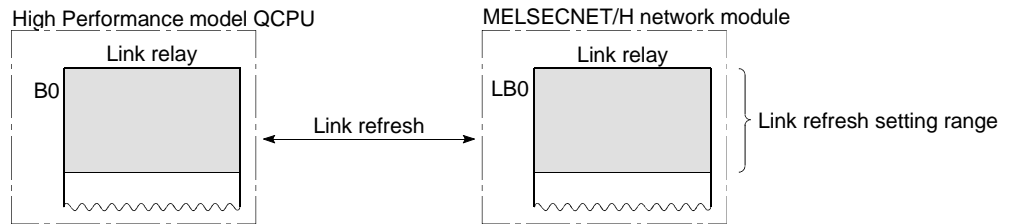
**REMARK**

\*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.

10.2.7 Link relays (B)

(1) Definition

- (a) A link relay is the High Performance model QCPU relay used to refresh the High Performance model QCPU from the MELSECNET/H network module's link relay (LB) and to refresh the MELSECNET/H network module's link relay (LB) from the High Performance model QCPU data.



Internal relays or latch relays can be used for data ranges not used by the MELSECNET/H network system.

- Range where no link relay latch is performed...Internal relay
- Range where link relay latch is performed.....Latch relay

- (b) There are no restrictions on the number of contacts (N/O contacts, N/C contacts) used in the program.

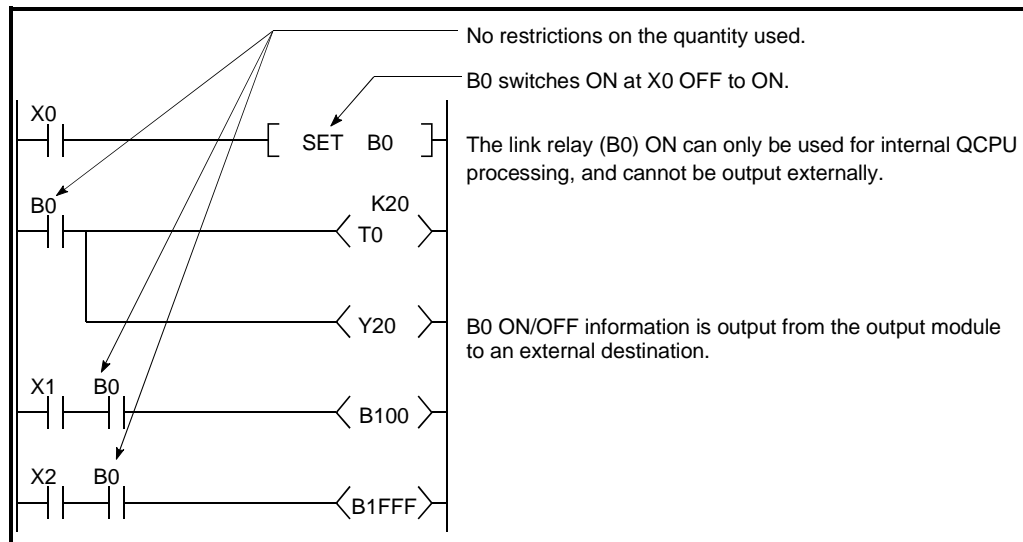


Figure 10.7 Link Relay

(2) Using link relays in the network system

In order to use link relays in the network system, a network parameter setting is required. Link relays for which no network parameter setting has been designated can be used as internal relays or latch relays.

**REMARK**

- 1) For details on the network parameters, refer to the For Q Corresponding MELSECNET/H Network System Reference Manual.
- 2) The MELSECNET/H Network Module has 16384 link relay points assigned. High Performance model QCPU has 8192 link relay points assigned. When using subsequent points after Point 8192, change the number of link relay points at the "Device" tab screen in the "(PLC) Parameter" dialog box.

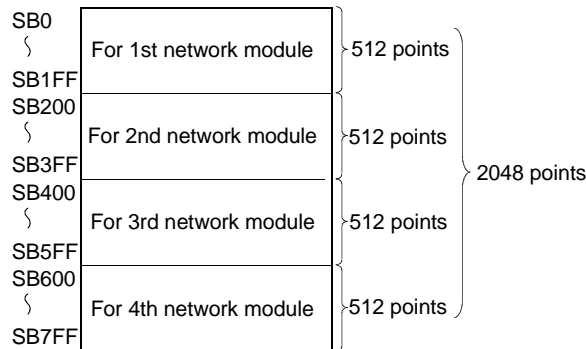
10.2.8 Link special relays (SB)

(1) Definition

- (a) A link special relay indicates the communication status and error detection of an intelligent function module, such as the MELSECNET/H Network Module.
- (b) Because link special relays are switched ON and OFF in accordance with various problems which may occur during a data link, they serve as a tool for identifying data link problems.

(2) Number of link special relay points

There are a total of 2048 link special relay points between SB0 and SB7FF. Link special relays are assigned at a rate of 512 points per each intelligent function module, such as the MELSECNET/H Network Module. Link special relays are assigned as shown below.



**REMARK**

For details on link special relays used at the QCPU, refer to the QCPU (Q mode)/QnACPU Programming Manual (Common Instructions).

10.2.9 Step relays (S)

A step relay is an SFC program device.

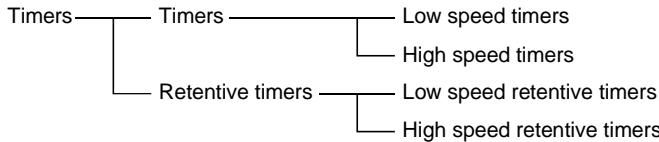
For details regarding procedures for using step relays, refer to the QCPU (Q mode)/QnACPU Programming Manual (SFC).

**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 Timers (T)

Timers are of up-timing, with the time measurement beginning when the coil switches ON, and ending (time out) when the current value exceeds the set value. The current value matches the set value when a "time-out" occurs. 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.

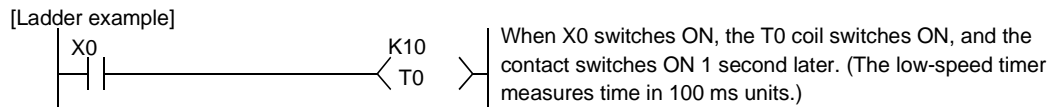


With a timer setting (instruction format), a device is assigned for a low speed timer or high speed timer. The OUT T0 instruction is used to assign a device for a low -speed timer. The OUTH T0 instruction is used to assign a device for a high speed timer. With a timer setting (instruction format), a 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.

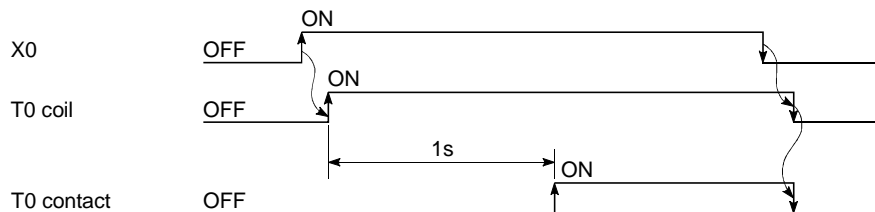
**Low-speed timers**

(1) Definition

- (a) Low speed timers are valid only while the coil is ON.
- (b) 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.



[Time chart]



(2) Measurement units

- (a) The default time measurement units setting for low speed timers is 100 ms.
- (b) 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.

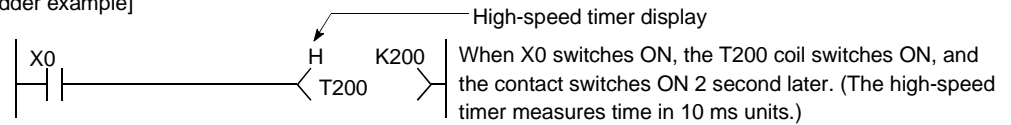


**High-speed timers**

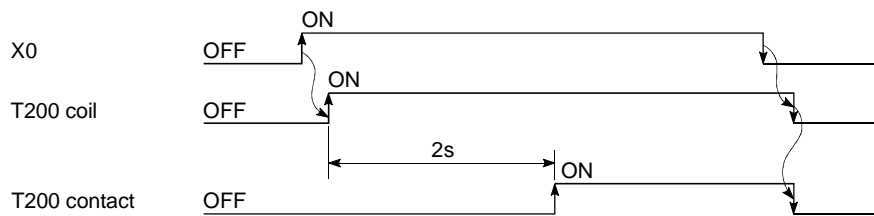
**(1) Definition**

- (a) High speed timers are valid only while the coil is ON. A high speed timer is marked with a symbol "H".
- (b) 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.

[Ladder example]



[Time chart]



**(2) Measurement units**

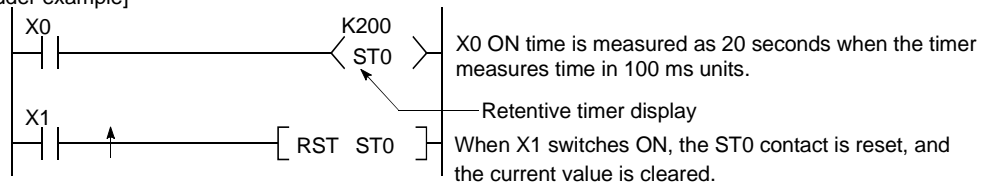
- (a) The default time measurement units setting for high speed timers is 10 ms.
- (b) The time measurement units setting can be designated in 0.1ms units within a 0.1 ms to 100 ms range.  
This setting is designated at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

**Retentive timers**

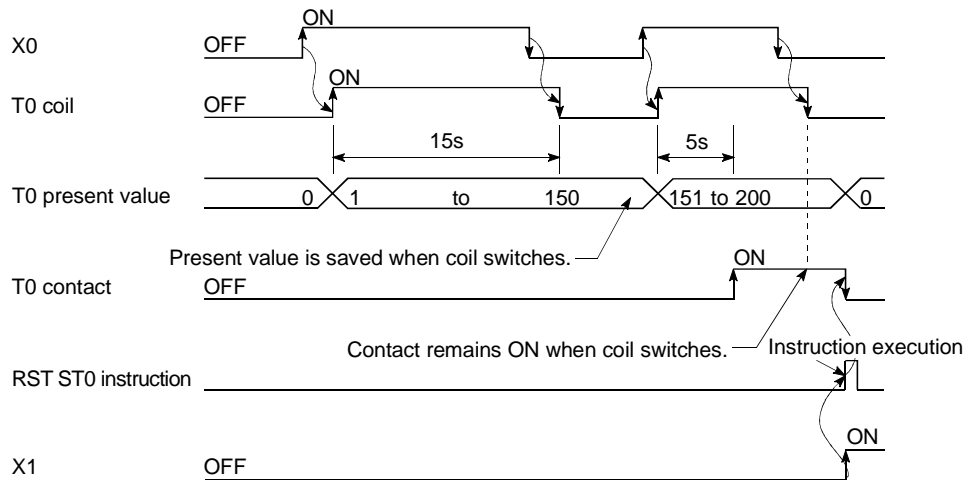
(1) Definition

- (a) Retentive timers measure the "coil ON" time.
- (b) 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.
- (c) There are 2 retentive timer types: low speed retentive timer, and high speed retentive timer.
- (d) The RST T[] instruction is used to clear (reset) the current value and switch the contact OFF.

[Ladder example]



[Time chart]



(2) Measurement units

- (a) 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

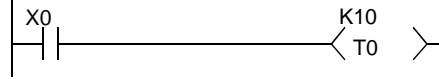
**REMARK**

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.

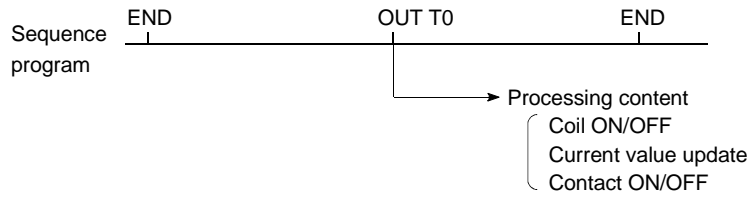
**Timer Processing and accuracy**

- (a) 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.

[Ladder example]

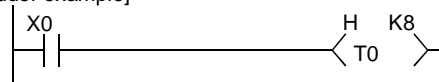


[Processing at execution of OUT T0 instruction]

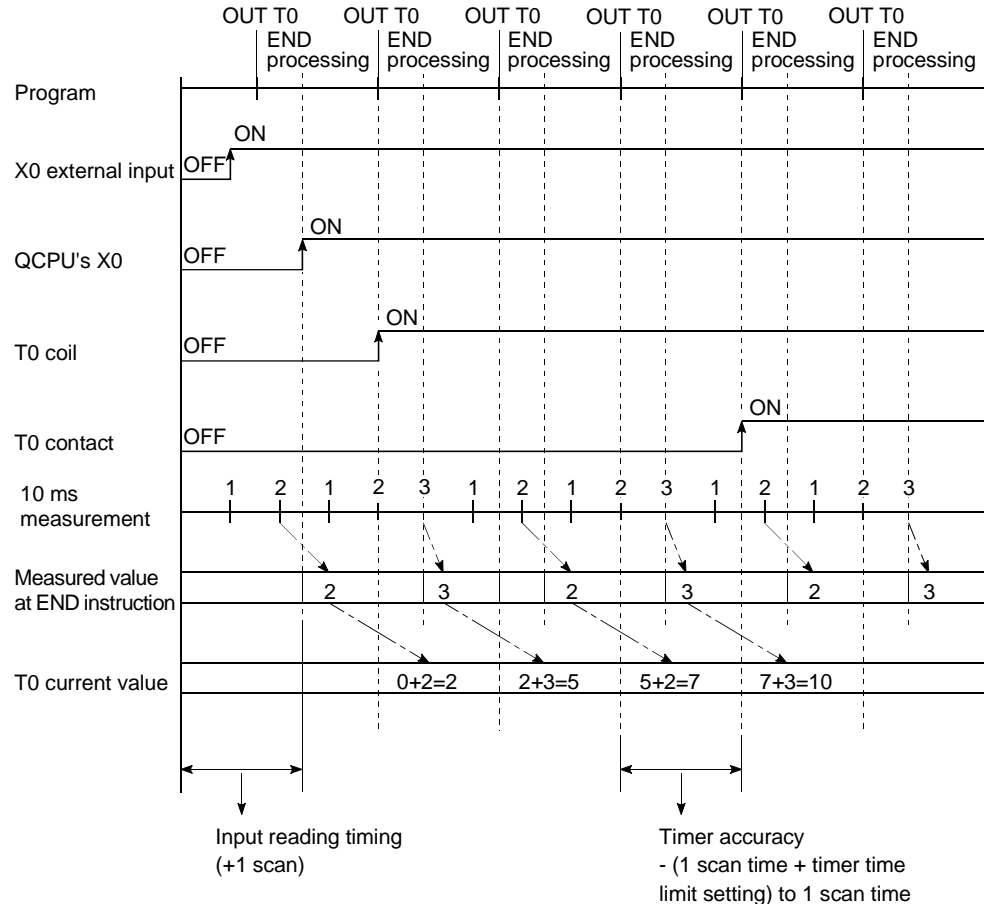


- (b) When the OUT T□ instruction is executed, the current value is added to the scan time measured at the END instruction. If the timer coil is OFF when the OUT T□ instruction is executed, the current value is not updated.

[Ladder example]



[Current value update timing]

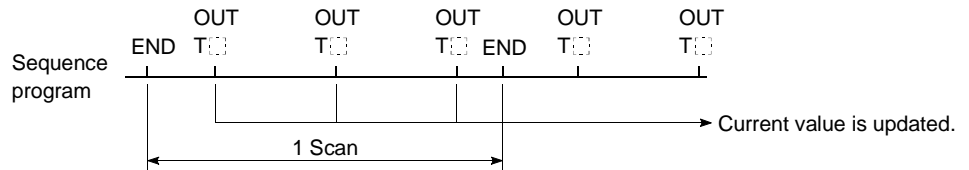


- (c) The timer response accuracy from when reading input (X), until when outputing it is + (2-scan time + timer time limit setting).

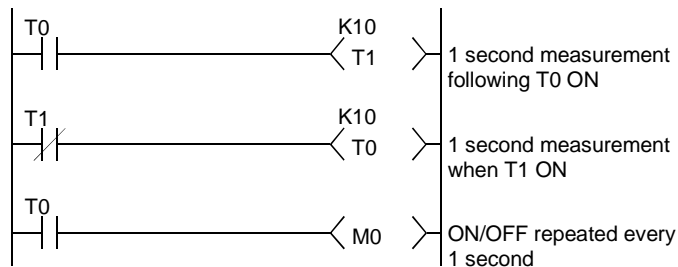
**Precautions for using timers**

The following are a few precautions regarding timer use:

- (a) 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.



- (b) 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.
- (c) Timers cannot be used in interrupt programs and fixed scan execution programs.
- (d) If the timer set value is "0", the contact turns ON when the OUT T[] instruction is executed.
- (e) 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.
- (f) 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.  
See Section 4.3.2 for details on the low speed scan time.
- (g) If two timers are used, the ON/OFF ladders should be created as shown below.



10.2.11 Counters (C)

Counters are "up-timing" types, with the contact being switched ON when the count value equals the set value (count-out condition).

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.

**Counters**

(1) Definition

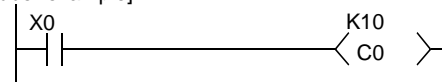
A counter is a device which counts the number of input condition leading edges in sequence programs.

(2) Count processing

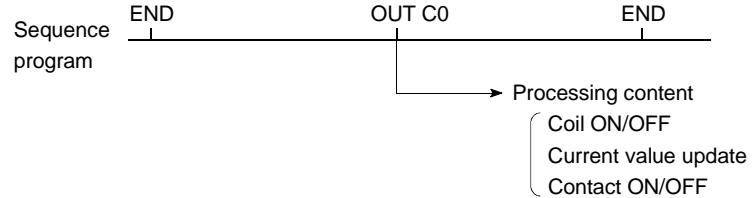
(a) When and OUT C[] instruction is executed, the following counter processing occurs: coil ON/OFF, current value update (count value + 1), and contact ON/OFF.

Counter current value update and contact ON/OFF processing are not performed at END processing.

[Ladder example]



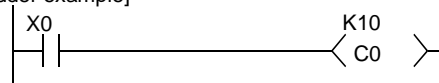
[Processing at OUT C0 Instruction (X0: OFF to ON)]



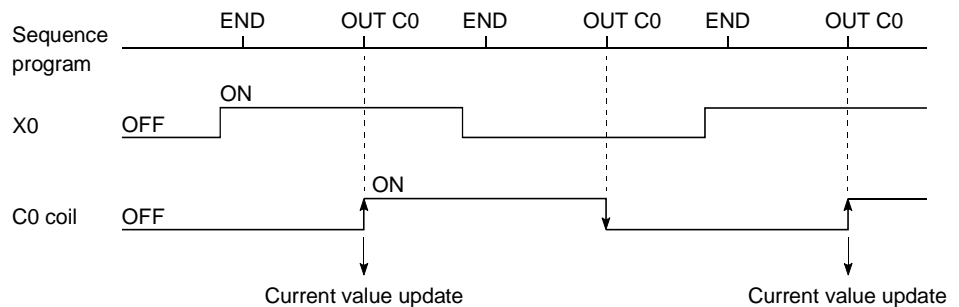
(b) The current value update (count value + 1) is performed at the leading edge (OFF to ON) of the OUT C[] instruction.

The current value is not updated in the following OUT C[] instruction statuses: OFF, ON to ON, ON to OFF

[Ladder example]

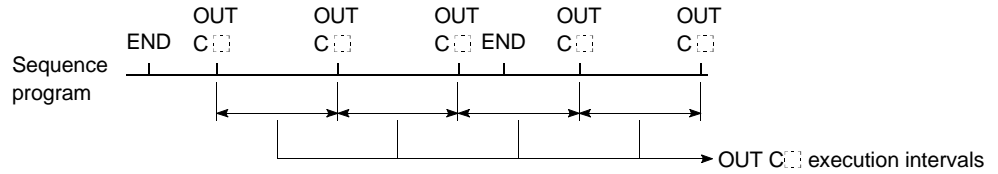


[Current value update timing]



- (c) Multiple counters can be used within a single scan to achieve the maximum counting speed.

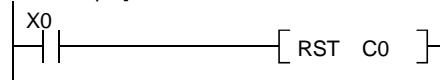
In such cases, the direct access input (DX[]) method should be used for the counter input signals. \*1



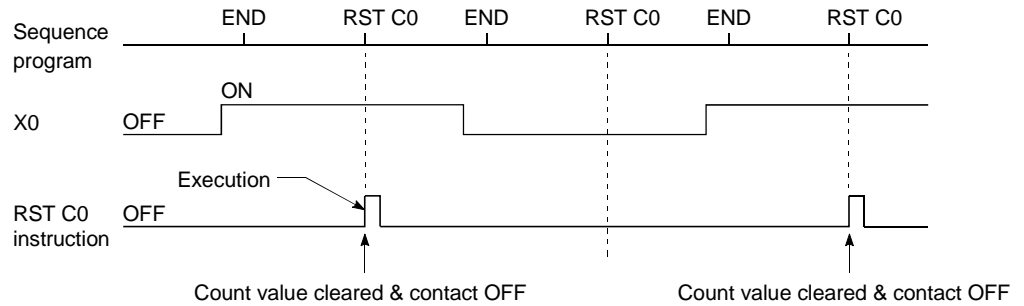
(3) Resetting the counter

- (a) 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.
- (b) The count value is cleared and the contact is switched OFF at execution of when the RST C[] instruction.

[Ladder example]



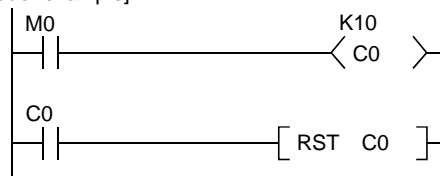
[Counter reset timing]



(4) Precautions for resetting the counter

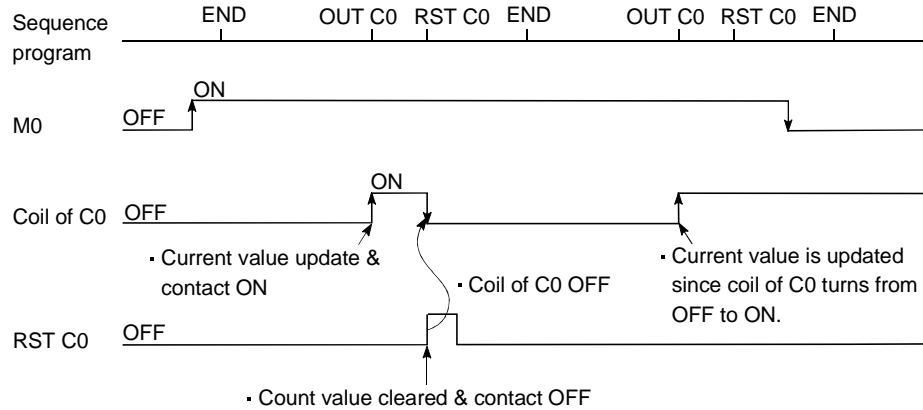
When the RST C[] instruction is executed, the coil of C[] also turns OFF. Hence, if the execution condition of the OUT C[] instruction is still ON after execution of the RST C[] instruction, the coil of C[] is turned ON at the execution of the OUT C[] instruction to update the current value (increment the count value by 1).

[Ladder example]



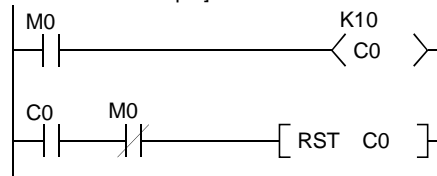
In the above ladder example, when M0 turns from OFF to ON, the coil of C0 turns ON, updating the current value. When C0 reaches the preset value finally, the contact of C0 turns ON, and the execution of the RST C0 instruction clears the current value of C0. At this time, the coil of C0 also turns OFF. 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.)

[Current value update timing]



As a remedy for the above, it is recommended to insert the normally closed contact of the OUT C0 instruction execution condition as the execution condition of the RST C0 instruction as shown in the following modified ladder example so that the coil of C0 will not turn OFF while the OUT C0 instruction execution condition (M0) is ON.

[Modified ladder example]



(5) 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:

$$\text{Maximum counting speed (Cmax)} = \frac{n}{100} \times \frac{1}{T} [\text{times/s}]$$

n: Duty(%) \* 2

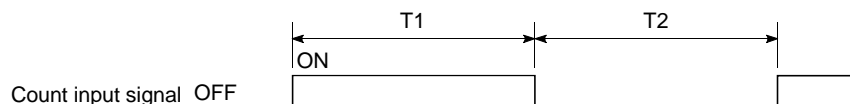
T: Execution interval of the OUT C[] instruction

**REMARK**

- 1) \*1: See Section 10.2.1 for details on direct access inputs.
- 2) \*2: The "duty" is the count input signal's ON-OFF time ratio expressed as a percentage value.

• When  $T1 \geq T2$   $n = \frac{T1}{T1+T2} \times 100\%$

• When  $T1 < T2$   $n = \frac{T2}{T1+T2} \times 100\%$



**Interrupt counters**

**(1) Definition**

Interrupt counters are devices which count the number of interrupt factor occurrences.

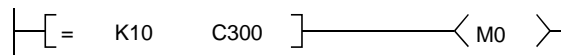
**(2) Count processing**

(a) 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.

(b) 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.

The figure below 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.)

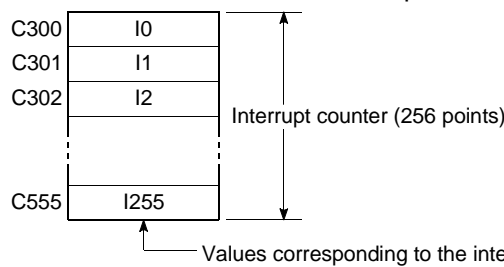


**(3) Setting the interrupt counter**

(a) In order to use interrupt counters, at first interrupt counter No. setting must be designated at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

256 points are then allocated for interrupt counters, beginning from the "first counter No." which is designated.

If C300 is designated as the first interrupt counter No., numbers C300 to C555 will be allocated for interrupt counters.



(b) In order to use an interrupt counter, an "interruption permitted" status must be established by E1 instruction at the main routine program.



#### (4) Precautions

- (a) 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) 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
- (c) 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
- (d) 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) 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) The interrupt counter's count value can be read out by using the sequence program MOV instruction.

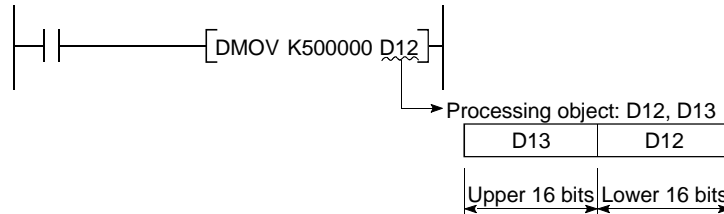
10.2.12 Data registers (D)

(1) Definition

- (a) Data registers are memory devices which store numeric data (-32768 to 32767, or 0000H to FFFFH).
- (b) Data registers, which consist of 16 bits per point, read and write data in 16-bit units.



- (c) 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.



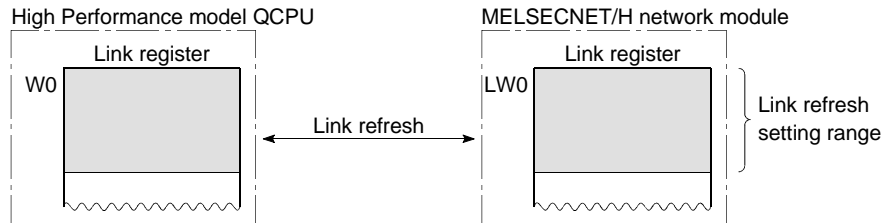
Two data registers can store a range of numeric data from -2147483648 to 2147483647 or from 0H to FFFFFFFFH.

- (d) Data stored by the sequence program is maintained until another data save operation occurs.

10.2.13 Link registers (W)

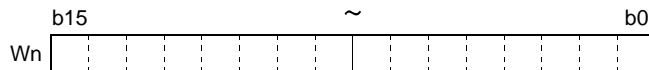
(1) Definition

- (a) A link register is the High Performance model QCPU memory used to refresh the High Performance model QCPU with data from the link registers (LW) of intelligent function modules including MELSECNET/H network module.  
Link registers are used to store numeric data (-32768 to 32767, or 0000H to FFFFH).

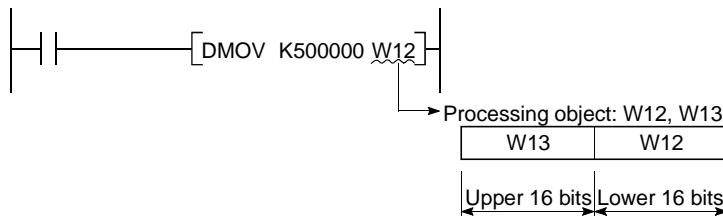


When used outside the MELSECNET/H network system's range, link registers can serve as data registers.

- (b) Link registers, which consist of 16 bits per point, read and write data in 16 bit units.



- (c) If the link registers are used for 32-bit instructions, the data is stored in registers  $W_n$  and  $W_n + 1$ . The lower 16 bits of data are stored in the link register No. ( $W_n$ ) designated in the sequence program, and the higher 16 bits of data are stored in the designated register No. + 1 ( $W_n + 1$ ).  
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.



In two link register points, -2147483648 to 2147483647 or 0H to FFFFFFFFH data can be stored.

- (d) Data stored by the link register is maintained until another data is save.

**REMARK**

The MELSECNET/H network module has 16384 link register points. The High Performance model QCPU has 8192 link register points. When subsequent points after Point 8192 are used for link registers, change a "number of points" setting of link registers at the "Device" tab screen in the "(PLC) Parameter" dialog box.

(2) 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**

For details on network parameters, refer to the Q Corresponding MELSECNET/H Network System Reference Manual.

10.2.14 Link special registers (SW)

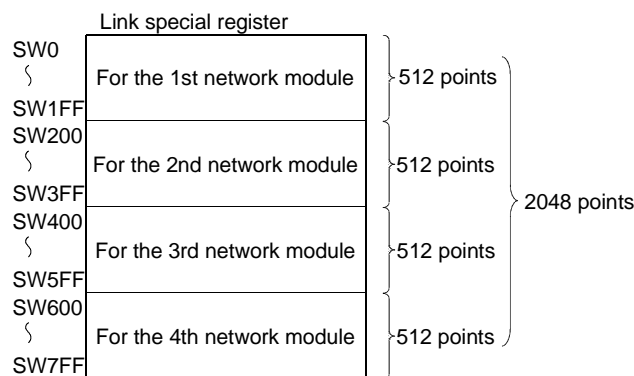
(1) Definition

- (a) Link special registers are used to store data on the communication status and errors of an intelligent function
- (b) 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

There are 2048 link special register points from SW0 to SW7FF. The link special register points are assigned at the rate of 512 points per intelligent function module, such as a MELSECNET/H network module.

By default, the following points are assigned for link registers as shown below.



**REMARK**

For details on link special registers used in the QCPU, refer to the QCPU(Q mode)/QnACPU Programming Manual (Common Instructions).

### 10.3 Internal System Devices

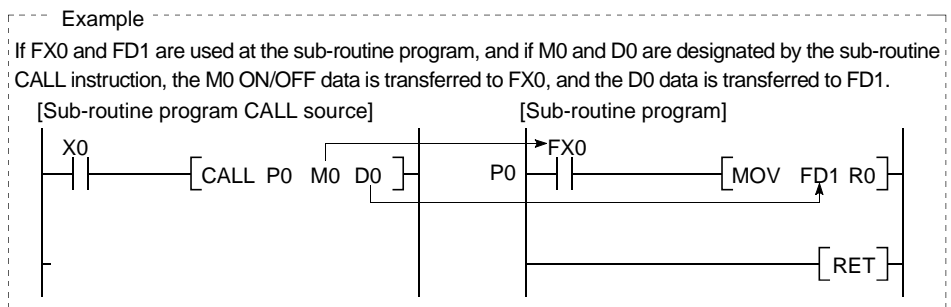
Internal system devices are used for system operations.

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

- (a) Function devices are used in a sub-routine program with arguments to perform write/read of data between the sub-routine call source with argument and the sub-routine program with argument.



- (b) Because the function devices used for each sub-routine program CALL source can be set, the same sub-routine program can be used without regard to other sub-routine CALL sources.

##### (2) 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 sub-routine program.
- In the sub-routine program, these devices are used for reading and processing bit data designated by sub-routine with argument CALL instruction.
- All the QCPU bit data designation devices can be used.

###### (b) Function output devices (FY)

- These devices are used to designate outputs of sub-routine program operation results (ON/OFF data) to the sub-routine program CALL source.
- The operation results are stored at the device designated by using sub-routine programs with arguments.
- All bit data designation devices except High Performance model QCPU inputs (X, DX) can be used.

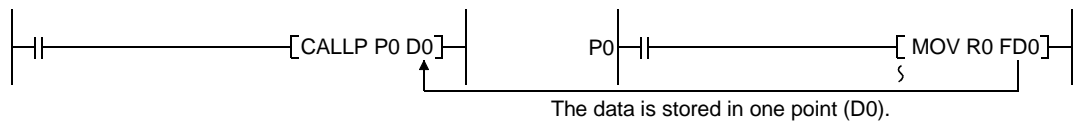
(c) Function registers (FD)

- Function registers are used to perform write/read of data between the sub-routine call source and the sub-routine program.
- The function register I/O condition is automatically determined by the High Performance model QCPU. If the sub-routine program data is the source data, the data is designated as sub-routine input data. If the sub-routine program data is the destination data, the data is designated as sub-routine output data.

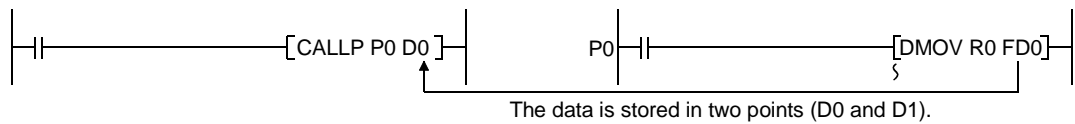
- 1 point occupies 4 words.

The number of words used depends on an instruction in a sub-routine program.

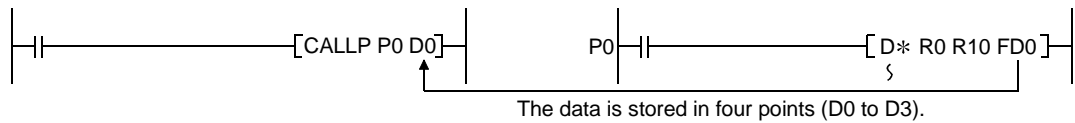
A one-word instruction requires 1 word.



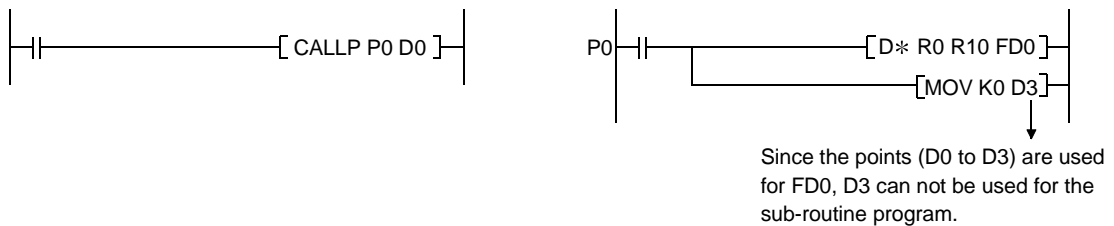
A two-words instruction requires 2 words.



The destination of 32-bit multiplication/division operation requires 4 words.



- Valid devices cannot be used in a sub-routine 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.



- High Performance model QCPU's word data devices can be used.

**REMARK**

For a procedure for using function devices, refer to the QCPU (Q mode)/QnACPU Programming Manual (Common Instructions).

### 10.3.2 Special relays (SM)

#### (1) Definition

A special relay is used to store High Performance model QCPU status data.

#### (2) Special relay classifications

Special relays are classified according to their applications, as shown below.

- (a) For fault diagnosis : SM0 to SM199
- (b) System information : SM200 to SM399
- (c) System clock/system counter : SM400 to SM499
- (d) Scan information : SM500 to SM599
- (e) Memory card information : SM600 to SM699
- (f) Instruction related : SM700 to SM799
- (g) For debugging : SM800 to SM899
- (h) Latch area : SM900 to SM999
- (i) For A-PLC : SM1000 to SM1299\*

#### REMARK

- 1) For details on special relays which can be used by the High Performance model QCPU, refer to Appendix 1.
- 2) \*: This takes effect only after you have turned on the "Use special relay/special register form SM1000/SD1000" check box in the "Compatibility with A-PLC" section at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.

### 10.3.3 Special registers (SD)

#### (1) Definition

A special register is used to store High Performance model QCPU status data (diagnosis and system information).

#### (2) Special register classifications

Special registers are classified according to their applications, as shown below.

- (a) For fault diagnosis : SD0 to SD199
- (b) System information : SD200 to SD399
- (c) System clock/system counter : SD400 to SD499
- (d) Scan information : SD500 to SD599
- (e) Memory card information : SD600 to SD699
- (f) Instruction related : SD700 to SD799
- (g) For debugging : SD800 to SD899
- (h) Latch area : SD900 to SD999
- (i) For A-PLC : SD1000 to SD1299\*
- (j) Fuse-blown module : SD1300 to SD1399
- (k) Check of I/O modules : SD1400 to SD1499

#### REMARK

- 1) For details on special relays which can be used by the High Performance model QCPU, refer to Appendix 2.
- 2) \*: This takes effect only after you have turned on the "Use special relay/special register form SM1000/SD1000" check box in the "Compatibility with A-PLC" section at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.



10.4 Link Direct Devices (J□\□)

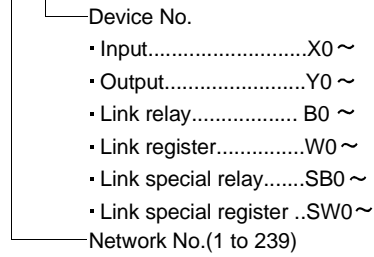
(1) Definition

(a) At END processing of sequence program, a data refresh (data transfer) is performed between the High Performance model QCPU and the MELSECNET/H network modules. Link direct devices are used to directly access the link devices in the MELSECNET/H network modules.

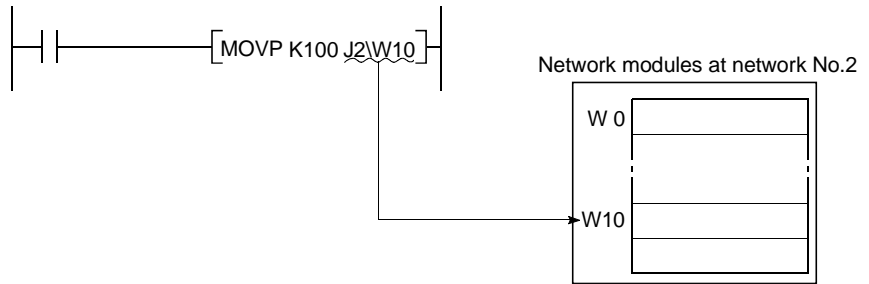
(b) Designation method

- Link direct devices are designated by network No. and device No.

Designation method: J□\□



- For link register 10 (W10) of network No.2, the designation would be "J2W10"



- For a bit device (X, Y, B, SB), digit designation is necessary.

Designation example : J1\K1X0, J10\K4B0

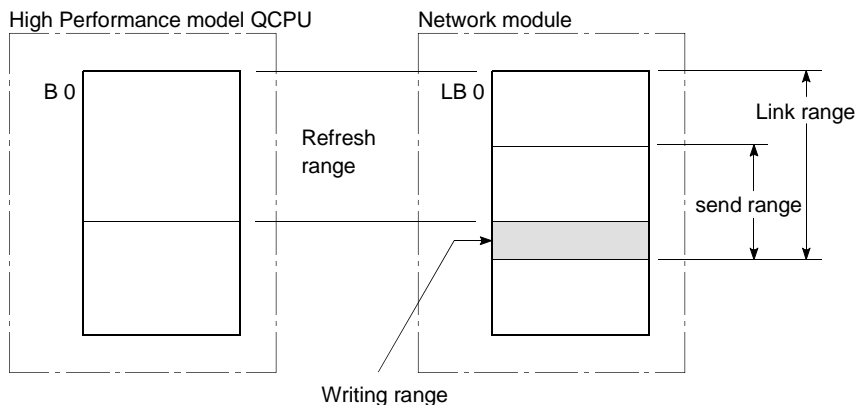
(2) Designation range

Link direct device designations are allowed for all the link devices in network modules.

( Device outside the range specified by the network refresh parameters can also be designated. )

(a) Writing

- 1) 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.

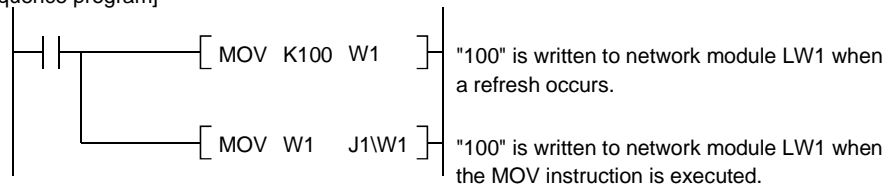


- 2) Although writing is also allowed in the "refresh range" portion of the link device range (specified by refresh parameters), the link module's link device data will be rewritten when a refresh operation occurs. Therefore, when writing by link direct device, the same data should also be written to the High Performance model QCPU related devices designated by refresh parameter.

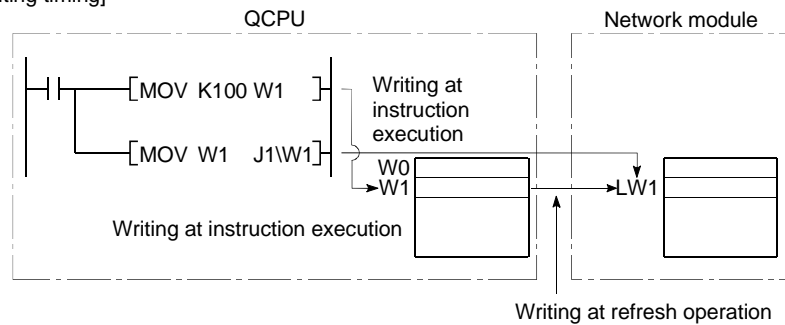
[Refresh parameter settings]

- Network No. : 1
- High Performance model QCPU(W0 to W3F) ↔ Network module (LW0 to LW3F)

[Sequence program]



[Writing timing]



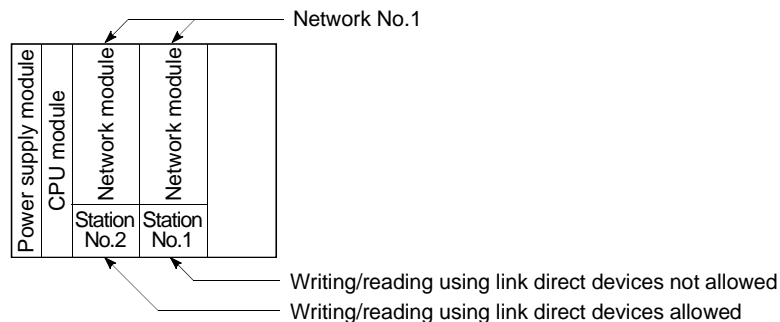
- 3) When data is written to another station's writing range using a link direct device, the data which is received from that station will replace the written data.

(b) Reading

Reading by link direct device is allowed in the entire link device range of network modules.

**POINT**

Only one network module capable of writing/reading link direct devices can be used per network number.  
 If two or more network modules are installed at the same network number, the network module with the lowest first I/O number will be the one that handles writing/reading using link direct devices.  
 For example, if station No.1 and station No.2 network modules are installed in network No.1 as shown in the figure below, the station No.2 network module will handle link direct device operations.



## (3) Differences between "link direct devices" and "link refresh"

The differences between "link direct devices" and "link refresh" are shown in Table 10.4 below.

Table 10.4 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		All network module link devices	Refresh parameter designated range
Access data guarantee range		Word units (16 bits)	

**REMARK**

- 1) For details on the MELSECNET/H network system, refer to the Q Corresponding MELSECNET/H Network System Reference Manual.
- 2) For details on network parameters, common parameters, and network refresh parameters, refer to the following manuals:
  - Detailed information : Q Corresponding MELSECNET/H Network System Reference Manual
  - Setting procedures : GX Developer Operating Manual, Windows Version

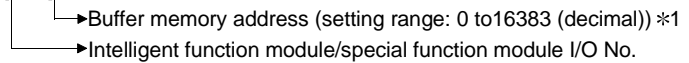
10.5 Intelligent Function Module Devices (U□\G□)

(1) Definition

(a) The intelligent function module devices allow the High Performance model QCPU 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.

(b) 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□



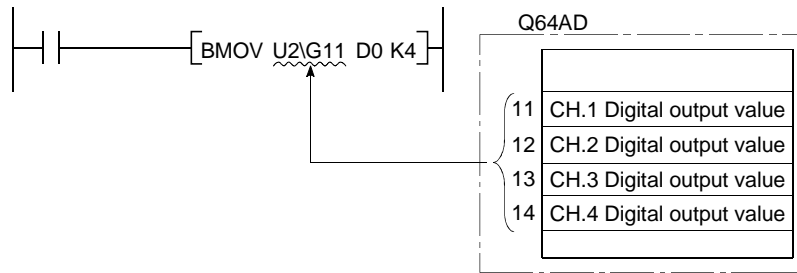
\*Setting: If the I/O No. is a 3-digit value, designate the first 2 digits.

For X/Y1F0....X/Y1F0

Designate "1F"

\*Setting range: 00H to FEH

When digital output values of channels (CH.1 to CH.4) of the Q64AD Type Analog-Digital Conversion Module (X/Y20 to 2F) mounted at Slot 2 of the main base unit are stored in D0 to D3, the I/O number and the buffer memory address are specified as shown below.



(2) Processing speed

The processing speed for intelligent function module devices is;

(a) Reading or writing the buffer memory of the intelligent function module/special function module is rather faster than the "processing speed of FROM/TO instructions." (For example, case of "MOV U2\G11 D0")

(b) To conduct reading the buffer memory of the intelligent function module/special function module 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")

If the same buffer memory of the same intelligent function module/special function module is used two or more times in a sequence program, the processing speed can be increased by using the FROM instruction to read that buffer memory data to a High Performance model QCPU device.

**REMARK**

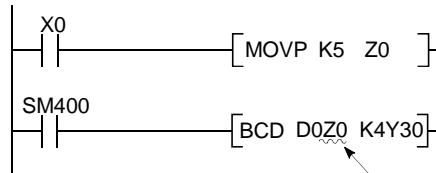
\*1: For details on buffer memory addresses and applications, refer to the intelligent function module/special function module manual.

10.6 Index Registers (Z)

(1) Definition

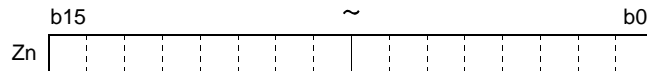
- (a) Index registers are used in the sequence program for indirect setting (index qualification) designations.

An index register point is used for index modification.



Index registers consist of 16 bits per point.

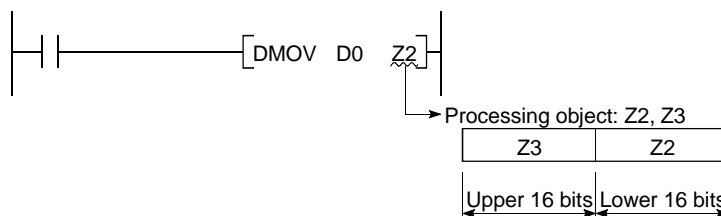
- (b) There are 16 index registers (Z0-Z15).
- (c) Index registers, which consist of 16 bits per point, read and write data in 16bit units.



- (d) 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.



**REMARK**

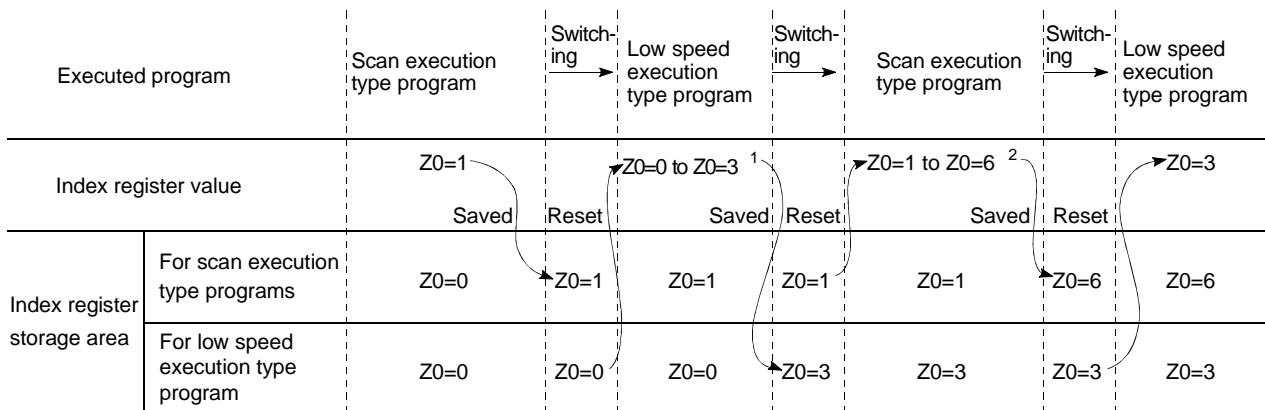
For index modification using the index register, refer to the following manual.  
 QCPU (Q mode) / QnACPU Programming Manual (Common instructions)

### 10.6.1 Switching between scan execution type programs and low speed execution type programs

When switching from a scan execution type programs or low speed execution type program to another program type, the index register (Z0 to Z15) data is saved (protected) and reset.

#### (1) Index register processing at switching between scan execution type programs and low speed execution type programs

- (a) When switching from a scan execution type program to a low speed execution type program occurs, the scan execution type program's index register data is saved, and the low speed execution type program's index register data is restored.
- (b) When switching from a low speed execution program to a scan execution type program occurs, the low speed execution type program's index register data is saved, and the scan execution type program's index register data is restored.



\* 1: For low-speed execution type program, Z0 is changed to 3.  
 \* 2: For scan execution type program, Z0 is changed to 6.

#### (2) 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.

10.6.2 Switching between scan/low speed execution type programs and interrupt/fixed scan execution type programs

The following processing is performed at the time of switching between scan/low-speed execution type program and interrupt program/fixed-cycle execution type program.

Index register (Z0 to Z15) value is saved (protected)/restored.

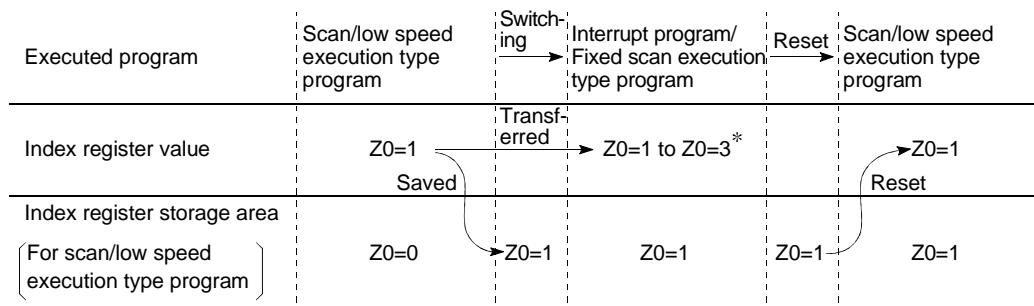
File register block No. is saved (protected)/restored.

For the index register (Z0 to Z15), 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.

(1) Index register processing

(a) When "High-speed execution" is not selected

- 1) When the scan/low speed execution type program is switched to the interrupt/fixed scan execution type program, the scan/low speed execution type program's index register value is first saved, and is then transferred to the interrupt/fixed scan execution type program.
- 2) When the interrupt/fixed scan execution type program is switched to the scan/low speed execution type program, the saved index register value is reset.

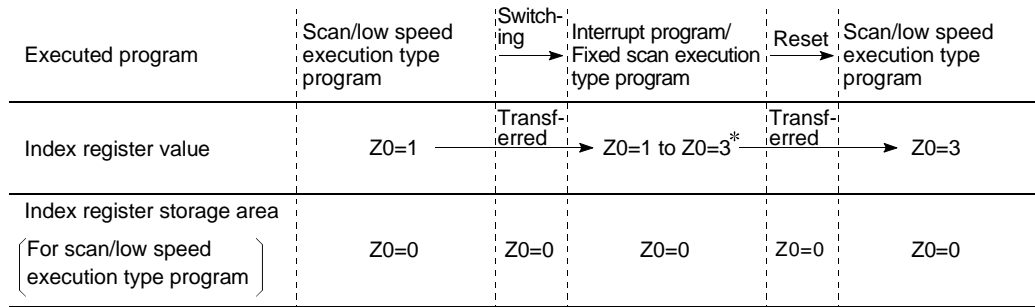


\*: For interrupt program, Z0 is changed to 3.

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.

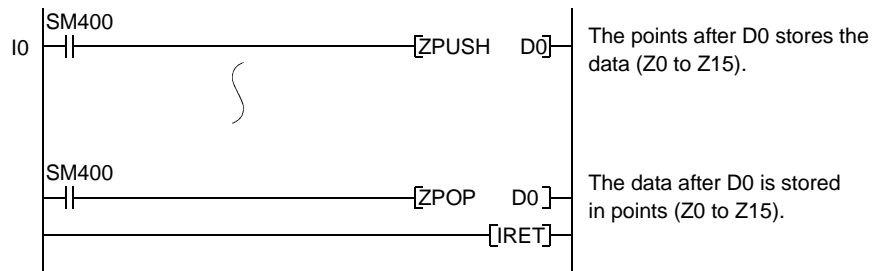
(b) When "High-speed execution" is selected

- 1) If a scan execution type program/low speed execution type program is switched to an interrupt program/fixed scan execution type program, index register data will not be saved/restored.
- 2) 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.



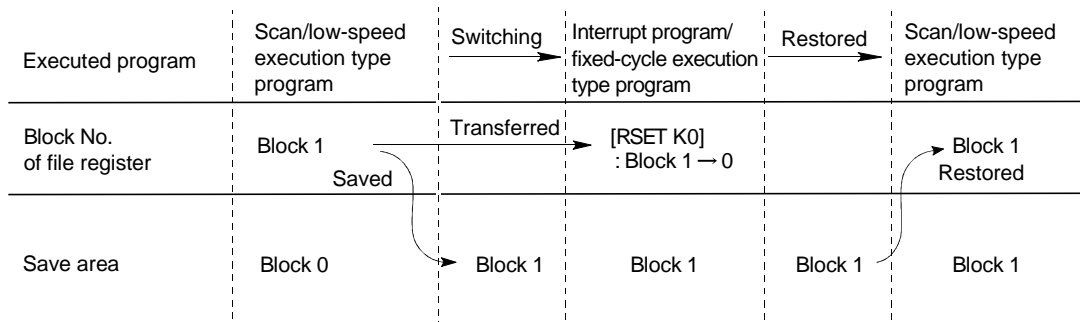
\*: For interrupt program, Z0 is changed to 3.

- 3) 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.



(2) File register block No. processing

- (a) At the time of switching from the scan/low-speed execution type program to the interrupt program/fixed-cycle execution type program, the block No. of the file register in the scan/low-speed execution type program is saved and then transferred to the interrupt program/fixed-cycle execution type program.
- (b) At the time of switching from the interrupt program/fixed-cycle execution type program to the scan/low-speed execution type program, the saved block No. of the file register is restored.





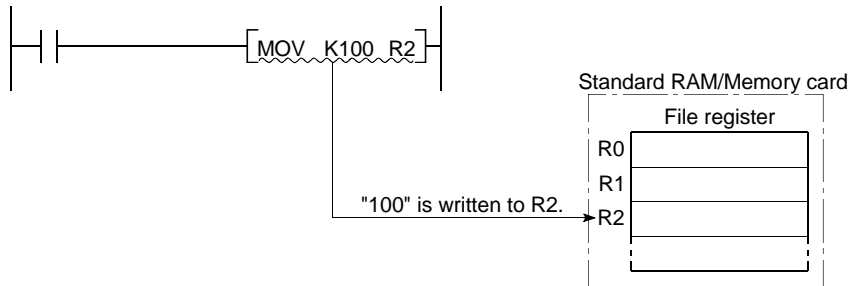
10.7 File Registers (R)

(1) Definition

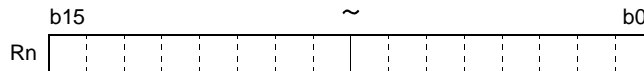
- (a) File registers are expansion devices for data registers.
- (b) File register data is stored in files in the standard RAM, the memory card.  
Refer to the table as follows:

CPU Module Type	Number of File Registers
Q02CPU	32k points
Q02HCPU, Q06HCPU	64k points
Q12HCPU, Q25HCPU	128k points

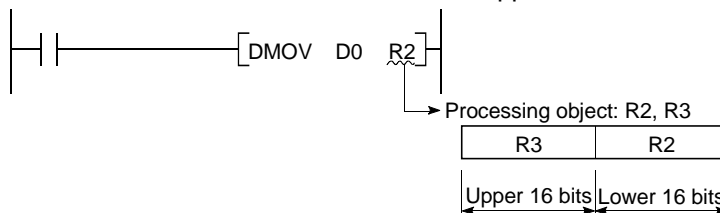
\* Use a memory card to store much more points if necessary.



- (c) File registers, which consist of 16 bits per point, read and write data in 16bit units.



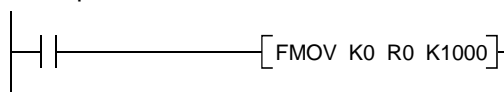
- (d) If the file registers are used for 32-bit instructions, the data will be stored in registers Rn and Rn + 1.  
The lower 16 bits of data are stored in the file register No. (Rn) designated in the sequence program, and the upper 16 bits of data are stored in the designated file register No.+ 1.  
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.



Two file registers can be used to store numeric data from -2147483648 to 2147483647 or from 0H to FFFFFFFFH.

- (e) The content of the file register is retained if power-off or reset operation is performed. (It is not initialized if latch clear is conducted.)  
To initialize the file register contents, perform data clear operation in a sequence program or using GX Developer.

Example: To clear R0 to R999



- When using GX Developer, select file register all clear in the PLC memory clear of the Online dialog box to clear the data.

## 10.7.1 File register capacity

## (1) Using the Standard RAM

The standard RAM can store the following points of file registers.

Q02CPU: 32k points, Q02HCPU/Q06HCPU: 64k points, Q12HCPU/Q25HCPU: 128k points

The standard RAM can store only the file registers and local devices.

Hence, when the local devices are not used, the above number of points can be used for the file registers.

## (2) Using the SRAM Card

The size of a file can be expanded at the rate of 32 k words per block, up to 32 blocks, 1017 k words. The number of expandable blocks depends on the size of programs or device comments stored on a memory card.

## (3) Using the Flash Card

The size of a file can be expanded at the rate of 32 k words per block up to 32 blocks, 1018 k words. The number of expandable blocks depends on the size of programs or device comments stored on a memory card.

**REMARK**

For details regarding the High Performance model QCPU memory cards, see Section 6.1.

## 10.7.2 Differences in memory card access method by memory card type

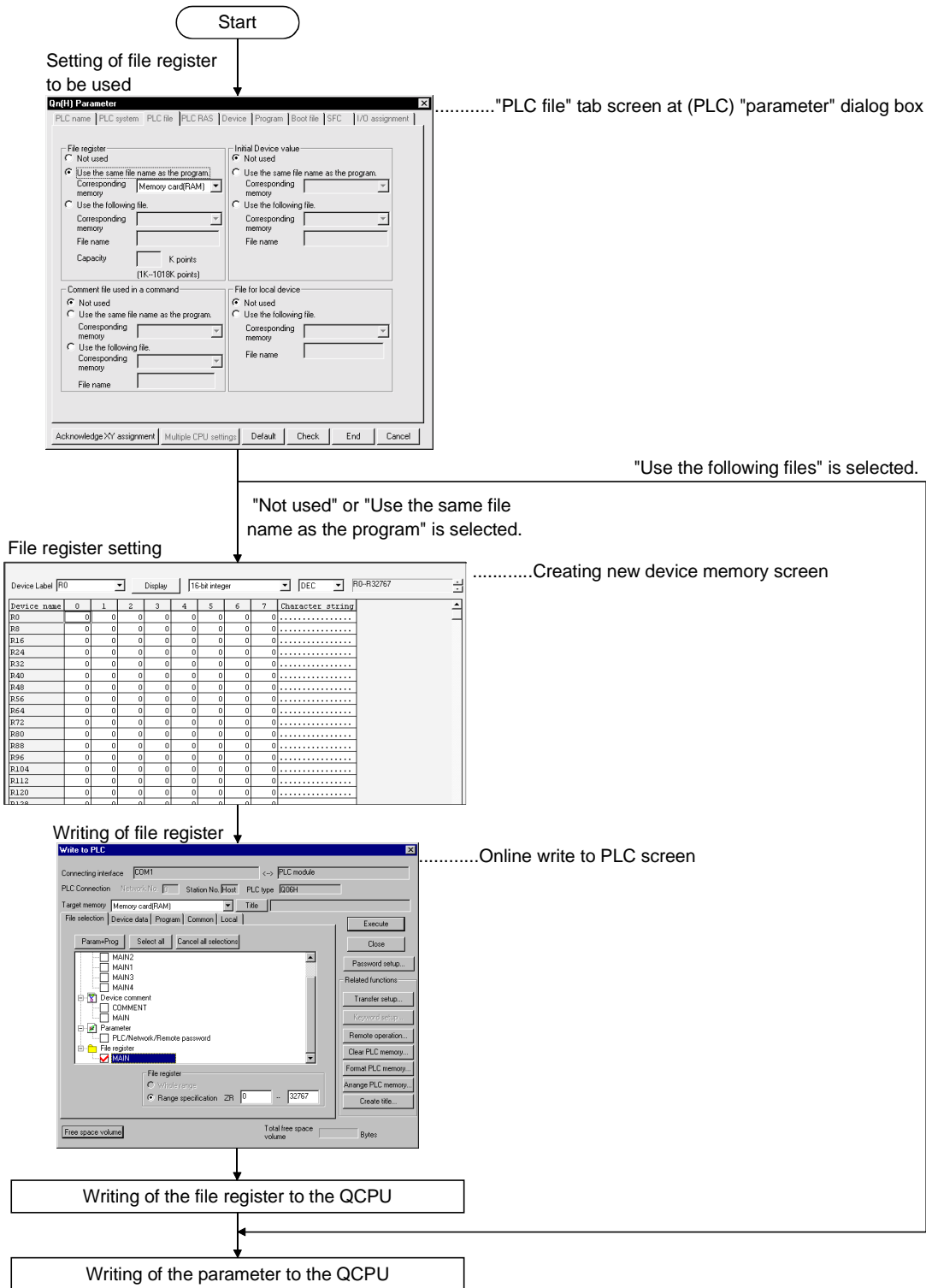
File registers are stored in three types of memories: standard RAM, SRAM card, and Flash card.

Note that the file register access method differs depending on the memory type.

How to Access		Standard RAM	SRAM Card	Flash Card
Read with a user's program		○	○	○
Write with a user's program		○	○	×
PLC read through the device setting		○	○	○
PLC write through the device setting		○	○	×
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 Series	○	○	×
Random write command from GOT 900 Series		○	○	×

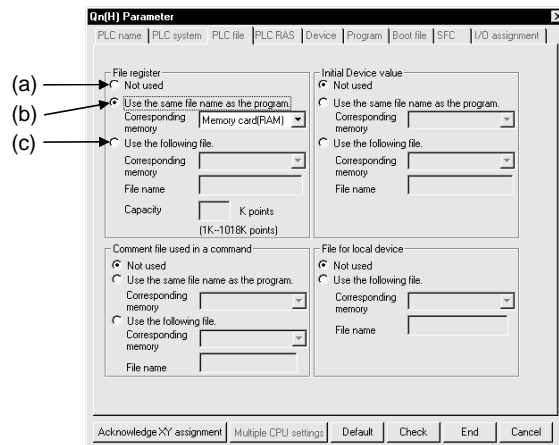
10.7.3 Registering the file registers

To use file registers, register the file registers with the High Performance model QCPU in the following steps.



(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.



(a) When selecting "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) When selecting "Use the same file name as the program"

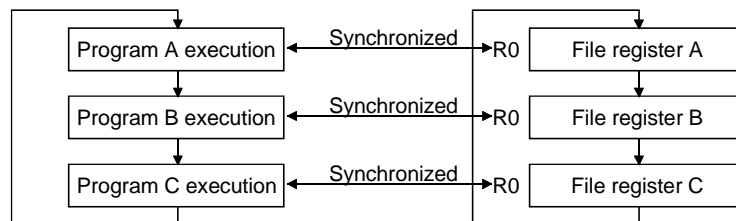
- 1) This setting should be selected when the file registers having the same file name as the sequence program are to be used.
- 2) If the program is changed, the file registers are automatically changed to conform to the new program name. There are also cases where it is convenient to use the file registers as local devices which can only be used with the program currently being executed.

3) The number of file register to use can be set by writing to PLC online.

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 execution --- File register A is accessed.
- At program B execution --- File register B is accessed.
- At program C execution --- File register C is accessed.



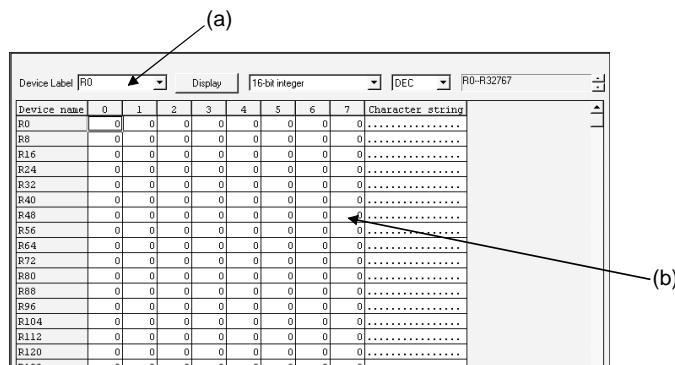
**POINT**  
 File registers dedicated to each program may not be designated with some instructions. Refer to the allowable device in the programming manual of each instruction for details.

(c) When selecting "Use the following file"

- 1) This setting should be selected when a given file register is to be shared by all executed programs.
- 2) 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. 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.



(a) Setting the file registers

Type "Rn" in the list box to view a listing of file registers.

(b) Setting the parameters

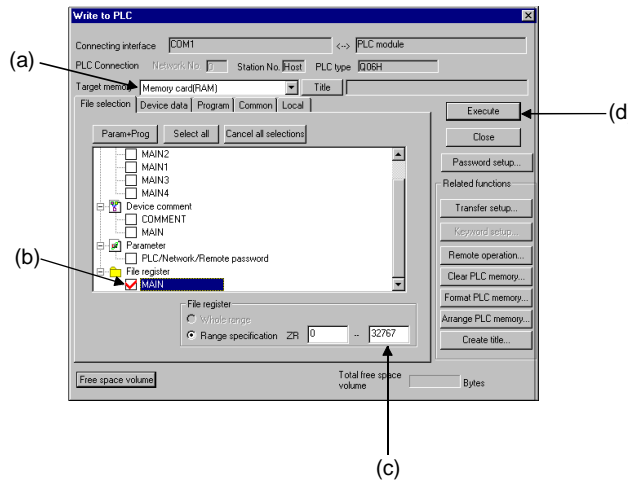
Enter the desired data in columns to specify file registers. This step is not needed when you specify only the capacity of file registers.

(3) Registering the File Register File with the High Performance model QCPU

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 High Performance model QCPU:

- Not used
- Use the same file name as the program

For registration of a file register file, use the "Write to PLC" dialog box.



- (a) Selecting a memory to store file registers  
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 High Performance model QCPU (QCPU-side filename).
  - 1) The capacity of file registers can be specified from ZR0 in the units of 1 point. 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 High Performance model QCPU's memory  
This button is used to store a file register file with the specified number of points in the specified High Performance model QCPU's memory.

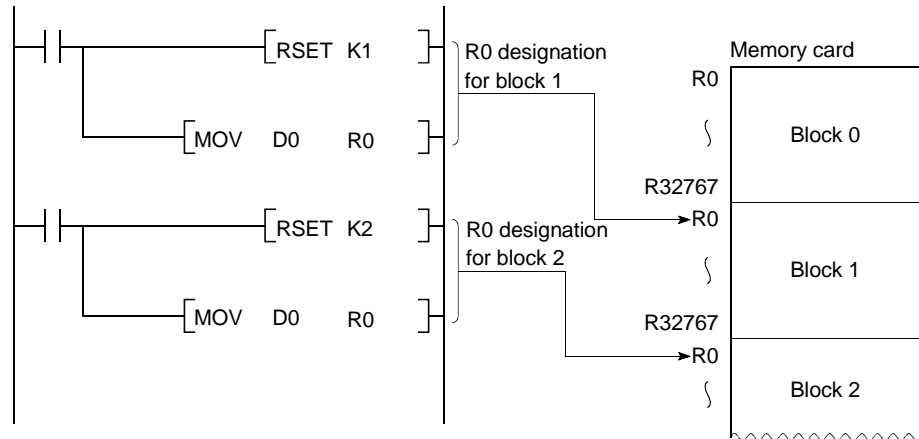
10.7.4 File register designation method

(1) Block switching format

The block switching format designates the number of file register points in 32k point (R0 to R32767) units.

If multiple blocks are used, switch to the block No. to be used in the RSET instruction for further file register settings.

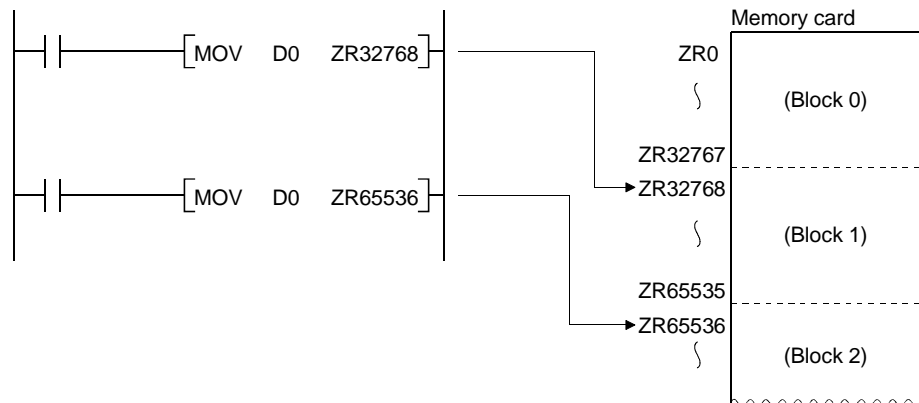
Settings are designated in the R0 to R32767 range for each block.



(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.



## 10.7.5 Precautions in using file registers

## (1) Using file register Nos. not registered or outside the registered range

- (a) When file register files are not registered in the High Performance model QCPU, no error occurs even if reading/writing to file registers

Reading data from a file register results in the following:

- Undefined data is stored in the standard RAM.
- "0H" is stored in a memory card.

- (b) Writing/reading file register Nos. outside the registered range (points)

No error occurs even if reading/writing occurs to these file registers.

Reading data from a file register results in the following:

- Undefined data is stored in the standard RAM.
- "0H" is stored in a memory card.

## (2) File register capacity check

- (a) Perform a file register capacity check in the sequence program, so that reading/writing from/to the file register is performed within the size (number of points) set in the High Performance model QCPU.

- 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.

$$(\text{File register capacity}) > [32\text{k points} \times (\text{switching block No.}) + 1\text{k points}]$$

- (b) The available file register capacity can be checked in the file register capacity storage register (SD 647). \*1

The file register capacity is stored in SD647 in 1k point units.

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.

**REMARK**

\*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.

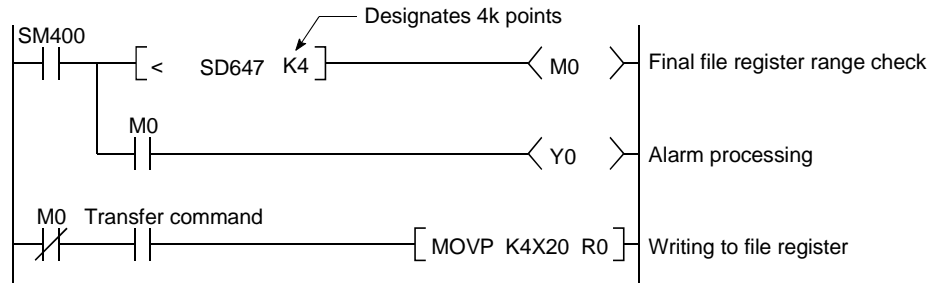


(c) Checking the file register capacity

- 1) Check The file register capacity used for each sequence program.
- 2) 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.

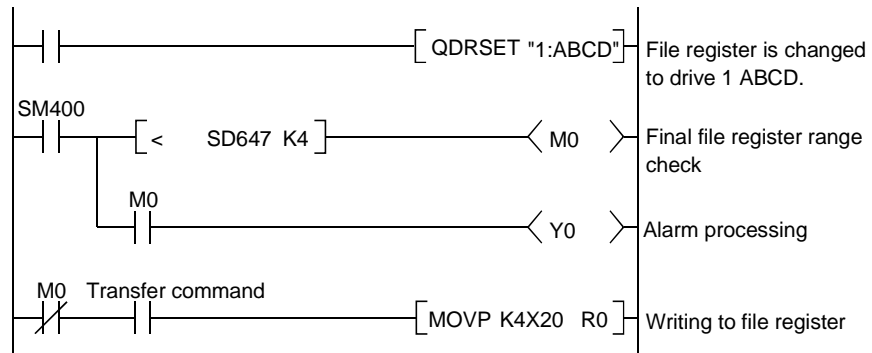
[Program example 1]

The file register "range of use" is checked at the beginning of each program.



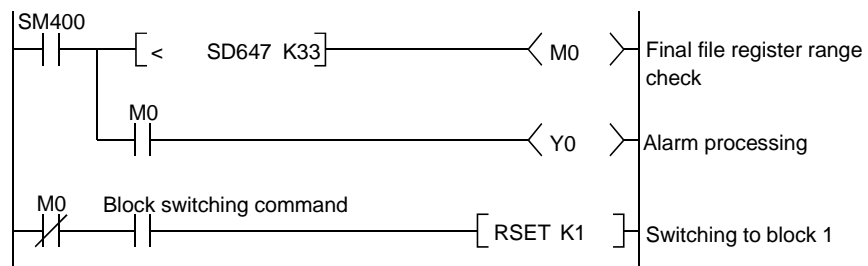
[Program example 2]

The file register "range of use" is checked after executing the QDRSET instruction.



[Program example 3]

For block switching.



(3) Deleting a File Register

To erase unwanted file register files, perform the PLC data deletion online.

10.8 Nesting (N)

(1) Definition

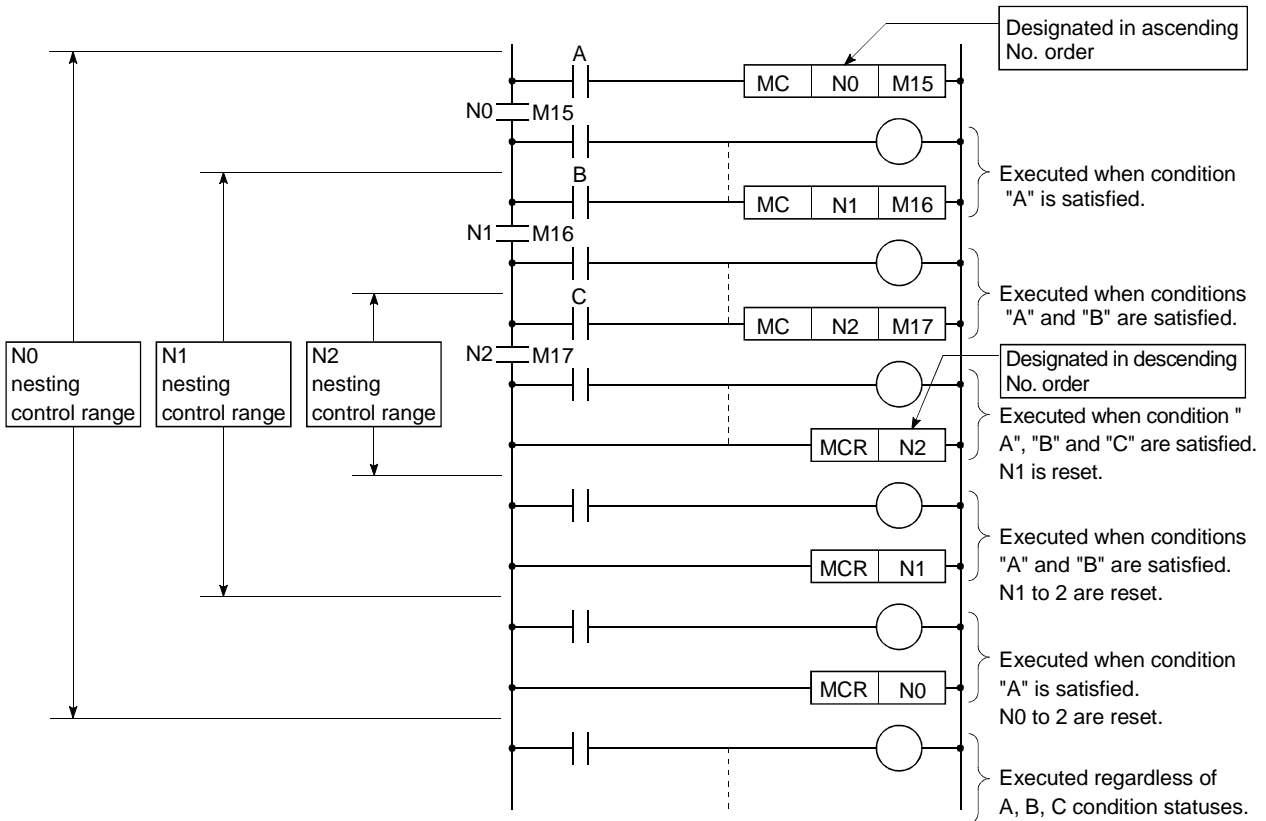
Nesting devices are used to nest MC or MCR master control instructions when programming operating conditions.

(2) Designation method with master control

The master control instructions are used to open and close the ladders' common bus so that switching of ladders may be executed efficiently by the sequence program.

Nesting devices must be numbered in descending order (from N0 to N7) of nested relation.

For details on how to use master control, refer to the QCPU(Q mode)/QnACPU Programming Manual (Common Instructions).



10.9 Pointers (P)

(1) Definition

Pointer devices are used in jump instructions (CJ, SCJ, JUMP) or sub-routine call instructions (CALL, ECALL).

A total of 4096 pointers can be used (total for all programs being executed).

(2) Pointer applications

(a) Pointers are used in jump instructions (CJ, SCJ, JMP) to designate jump destinations and labels (jump destination beginning).

(b) Pointers are used in sub-routine CALL instructions (CALL, CALLP) to designate the CALL destination and label (sub-routine beginning).

(3) Pointer types

There are 2 pointer types: "local pointers (Section 10.9.1)" which are used independently in programs, and "common pointers (Section 10.9.1)" which are used to call sub-routine programs from all programs executed in the CPU.

10.9.1 Local pointers

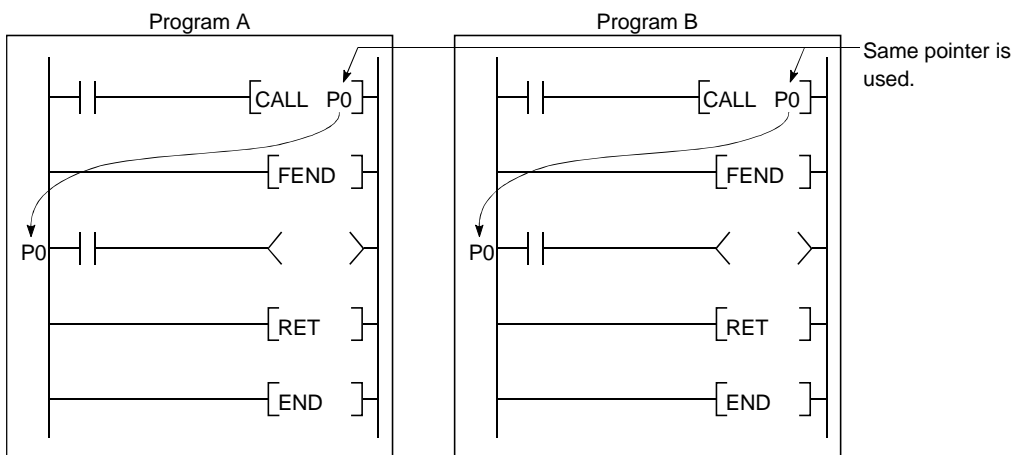
(1) Definition

(a) Local pointers are pointers which can be used independently in program jump instructions and sub-routine call instructions.

Local pointers cannot be used from other program jump instructions and sub-routine CALL instructions.

Use an ECALL instruction to call a sub-routine subprogram in a program file that contains local pointers.

(b) The same pointer No. can be used in each of the programs.



**REMARK**

For further information on jump instructions and sub-routine call instructions, see the QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions).

(2) Number of local pointer points

Local pointers can be divided among all the programs stored in the program memory.

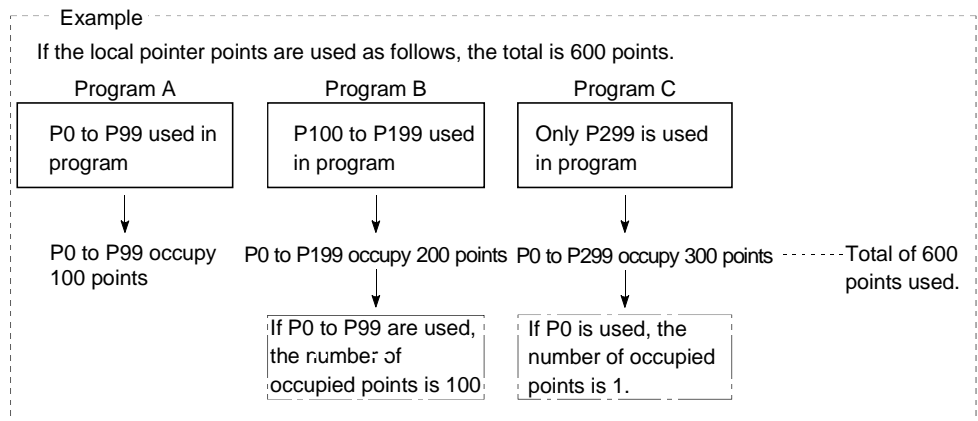
The local pointer No. ranges from P0 to the highest No. of the local pointer in use.

(The High Performance model QCPU's 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.

Therefore, when local pointers are used at several programs, the pointer settings should begin from P0.

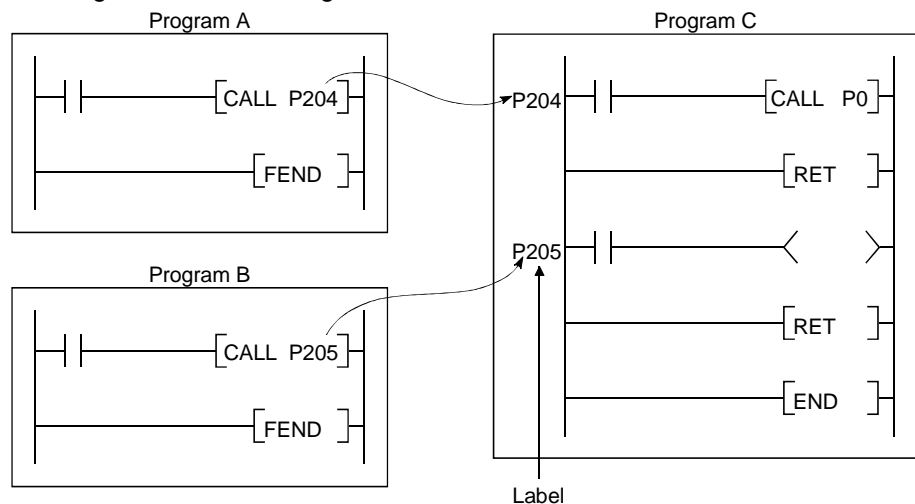
If the total number of pointers (total for all programs) exceeds 4096 points, a pointer configuration error (error code:4020) occurs.



10.9.2 Common pointers

(1) Definition

(a) Common pointers are used to call sub-routine programs from all programs being executed in the High Performance model QCPU.



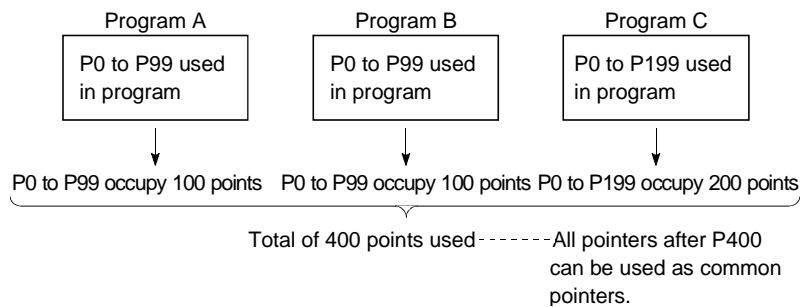
(b) The same pointer No. cannot be used again as a label. Such use will result in a pointer configuration error (error code:4021).

(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.

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.

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.



[Common pointer settings screen]

Qn(H) Parameter dialog box showing various settings. The 'Common pointer No.' field is highlighted with an arrow and the text: "Set the head number of the common pointers here." The field contains "P 400" and "After (0-4095)".

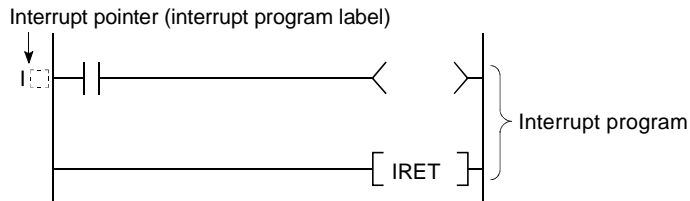
**POINT**

(1) In the jump instruction, jumping to common pointers in other programs is not allowed.  
 Common pointers should be used only with sub-routine call instructions.

10.10 Interrupt Pointers (I)

(1) Definition

(a) Interrupt pointers are used as labels at the beginning of interrupt programs.



(b) A total of 256 interrupt points (I0 to I255) can be used (total for all programs being executed).

(2) Interrupt pointer No. & interrupt factor

(a) As shown below, there are four types of interrupt factor.

- QI60/A1SI61 factor..... Interruption input from the QI60/A1SI61 interruption module.
- Sequence start generator.... Interruption input from special function module factor modules which can dictate an interrupt start to the High Performance model QCPU (QI60, A1SI61 excluded).
- Internal time factor..... Fixed scan interruption by High Performance model QCPU's internal timer.
- Error interruption..... Interruption by an error that does not stop sequence program operation.
- Intelligent function..... Interruption by an intelligent function module interrupt\*1

**REMARK**

\*1: To use the intelligent function module interrupt, the intelligent function module setting (interrupt points setting) is required at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.  
(For the interrupts from the intelligent function module, see Section 8.2.1.)

(b) A list of interrupt pointer Nos. and interrupt factors is given in Table 10.5 below.

Table 10.5 List of Interrupt Pointer Nos. and Interrupt Factors

I No.	Interrupt Factors	Priority Ranking	I No.	Interrupt factors	Priority Ranking				
I10	QI60/A1SI61 interrupt module factor	1st point	I32 * 3	Error factor * 4	Errors that stop operation	1			
I11		2nd point	I33		Empty	—			
I12		3rd point	I34		UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	2			
I13		4th point							
I14		5th point	I35		OPERATION ERROR SFCP OPE. ERROR SFCP ECE. ERROR EX. POWER OFF	3			
I15		6th point							
I16		7th point							
I17		8th point							
I18		9th point	I36		ICM. OPE ERROR FILE OPE. ERROR	4			
I19		10th point							
I20		11th point	I37		Empty	—			
I21		12th point							
I22		13th point	I38		PRG. TIME OVER	5			
I23		14th point							
I24		15th point	I39		CHK instruction execution Anunciator detection	6			
I25		16th point							
I26	Sequence start generator module factor * 1	1st point	I40 to I48	—	Empty	—			
I27		2nd point							
I28		3rd point	I49				Internal timer factor	0, 2 to 1.0mg * 5	* 7
I29		4th point							
I30		5th point							
I31		6th point							
I32		7th point							
I33		8th point							
I34		9th point							
I35		10th point							
I36		11th point							
I37		12th point							
I38	100ms	I50 to I255	Intelligent function module factor * 6	Specifies which intelligent function module is used with parameters.	18 to 223				
I39	40ms								
I40	20ms								
I41	10ms								

**REMARK**

- \*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 internal times shown are the default setting times.  
These times can be designated in 0.5 ms units through a 0.5 to 1000 ms range set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.
- \*3 : When an error interruption with "I32 (error that stops operation)" occurs, the High Performance model QCPU is not stopped until I32 processing is completed.
- \*4 : Execution of error interruptions is prohibited for the interrupt pointer Nos. I32 to I39 when the power is turned on and during a High Performance model QCPU reset. When using interrupt pointer Nos. I32 to I39, set the interruption permitted status by using the IMASK instruction.
- \*5 : 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 0.2 to 1.0ms in 0.1ms increments.
- \*6 : To use the intelligent function module interrupt, the intelligent function module setting (interrupt points setting) is required at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.  
(For the interrupts from the intelligent function module, see Section 8.2.1.)

POINT
*7: 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.

## 10.11 Other Devices

### 10.11.1 SFC block device (BL)

This device is used for checking if the block designated by the SFC program is valid. For details on the use of SFC block devices, refer to the QCPU(Q mode)/QnACPU Programming Manual (SFC).

### 10.11.2 SFC transition device (TR)

This device is used for checking if a forced transition is designated for a specified transition condition in a specified SFC program block.

For details regarding the use of SFC transition devices, refer to the QCPU(Q mode)/QnACPU Programming Manual (SFC).

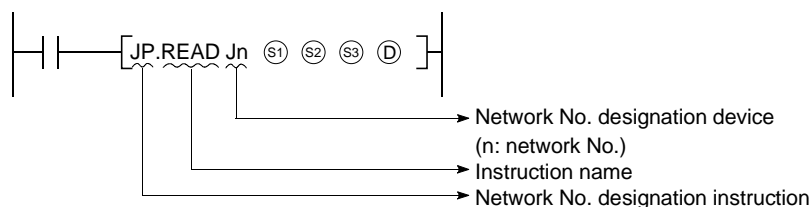
### 10.11.3 Network No. designation device (J)

#### (1) Definition

The network No. designation device is used to designate the network No. in data link instructions.

#### (2) Designating network No. designation device

The network No. designation device is designated in the data link instruction as shown below.



#### REMARK

For details on data link instructions, refer to the Q Corresponding MELSECNET/H Network System Reference Manual.



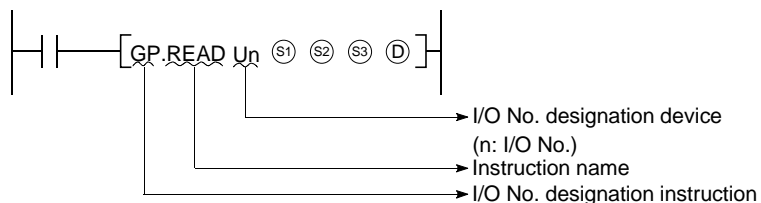
10.11.4 I/O No. designation device (U)

(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 below.



**REMARK**

For details on intelligent function module instructions, refer to the corresponding manual for the intelligent function module to be used.

10.11.5 Macro instruction argument device (VD)

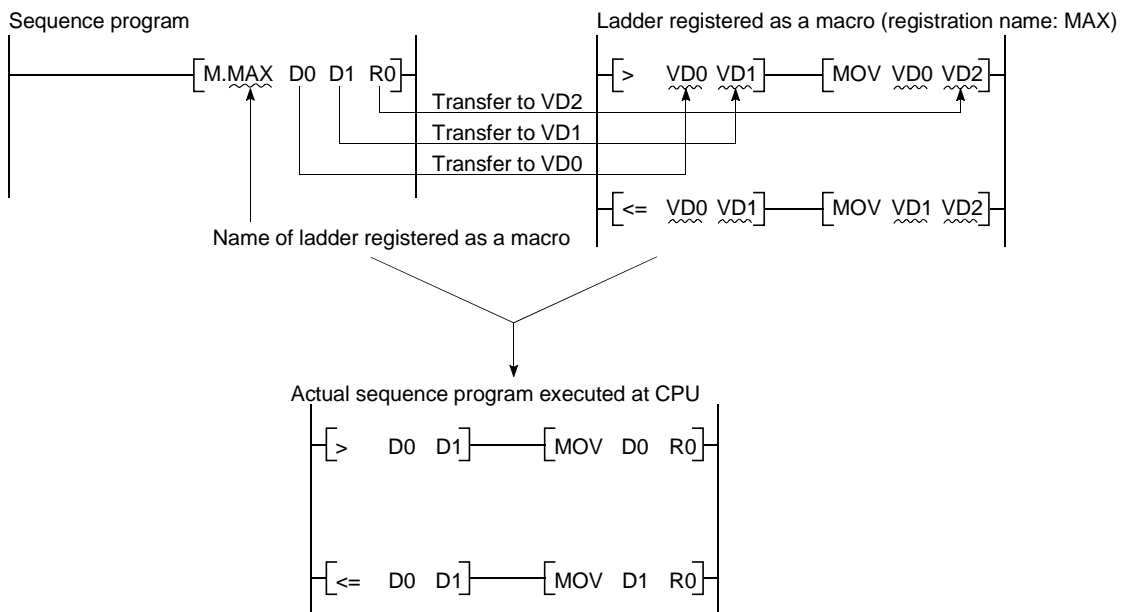
(1) Definition

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

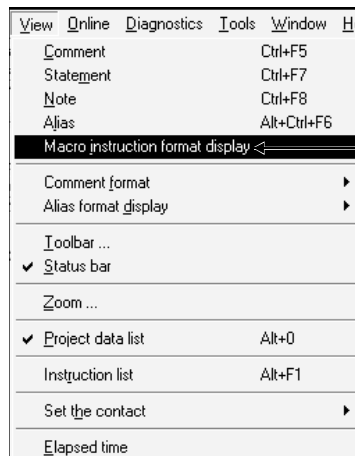
Specify the devices transferred from sequence programs to macro registration ladders as macro instruction argument devices among the devices used in the ladders registered as macro with GX Developer. \*1

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.



**REMARK**

- 1) \*1 : With the macro instruction argument device, VD0 to VD9 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.(Choose "View" - "Macro Instruction format display" to view macro instructions.)



Change of macro instruction display

## 10.12 Constants

### 10.12.1 Decimal constants (K)

#### (1) Definition

Decimal constants are devices which designate decimal data in sequence programs.

They are designated as "K□" settings (e.g. K1234), and are stored in the High Performance model QCPU in binary (BIN) code.

See Section 4.8.1 for details on binary code.

#### (2) Designation range

The designation ranges for decimal constants are as follows:

- For word data (16 bits).....K-32768 to K32767
- For 2-word data (32 bits) .....K-2147483648 to K2147483647

### 10.12.2 Hexadecimal constants (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).

See Section 4.8.3 for details on hexadecimal code.

#### (2) Designation range

The setting ranges for hexadecimal constants are as follows:

- For word data (16 bits)..... H0 to HFFFF (H0 to H9999 for BCD)
- For 2-word data (32 bits) ..... H0 to HFFFFFFFF (H0 to H99999999 for BCD)

## 10.12.3 Real numbers (E)

## (1) Definition

Real numbers are devices which designate real numbers in the sequence program.

Real numbers are designated as "E[]" settings (e.g. E1.234).



See Section 4.8.4 for details on real numbers.

## (2) Designation range

The setting range for real numbers is  $-1.0 \times 2^{128}$  to  $-1.0 \times 2^{-126}$ , 0 and  $1.0 \times 2^{-126}$  to  $1.0 \times 2^{128}$ .

## (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 " $\times 10^n$ " exponent.  
For example, 1234 becomes E1.234 + 3. \*1

**REMARK**

\*1: The "+3" in the above example represents a  $10^3$  value ( $10^3$ ).

## 10.12.4 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.

10.13 Convenient Uses for Devices

When executing multiple programs in the High Performance model QCPU, local devices among the internal user devices can be designated to execute each of the programs in an independent manner.

Moreover, the device initial settings allow the data setting for devices and intelligent function modules/special function modules without using a program.

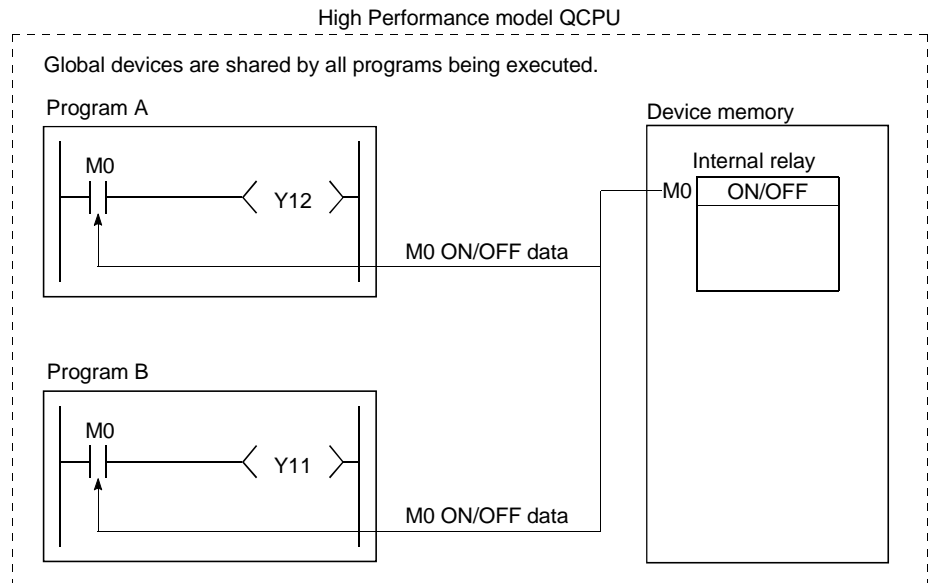
10.13.1 Global devices and local devices

A number of programs can be stored and executed in the High Performance model QCPU.

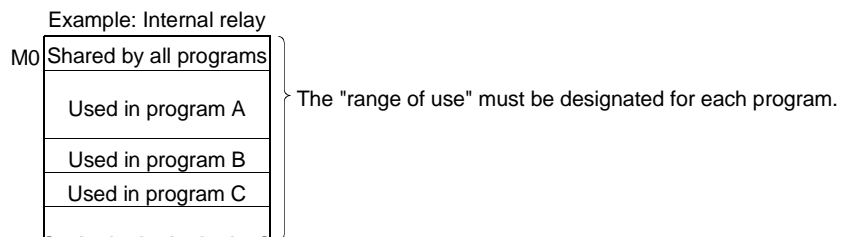
High Performance model QCPU devices are classified into "global devices" shared by all the programs being executed and "local devices" used independently by each of the programs.

(1) Global devices

- (a) Global devices can be shared by all the programs being executed in the High Performance model QCPU. Global device data are stored in the High Performance model QCPU's device memory, and can be used by all programs.

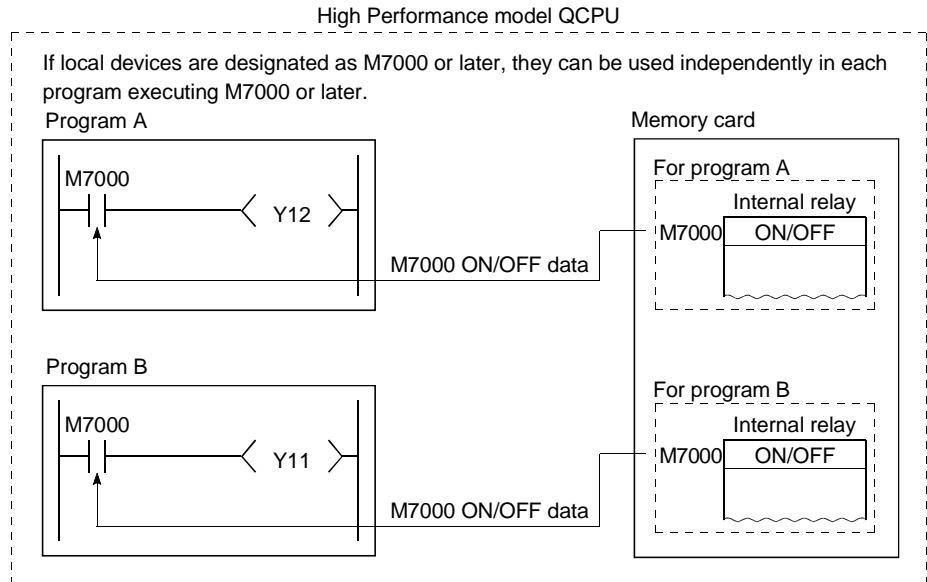


- (b) When executing multiple programs, the "shared range" for all programs, and the "independent range" for each program must be designated in advance.

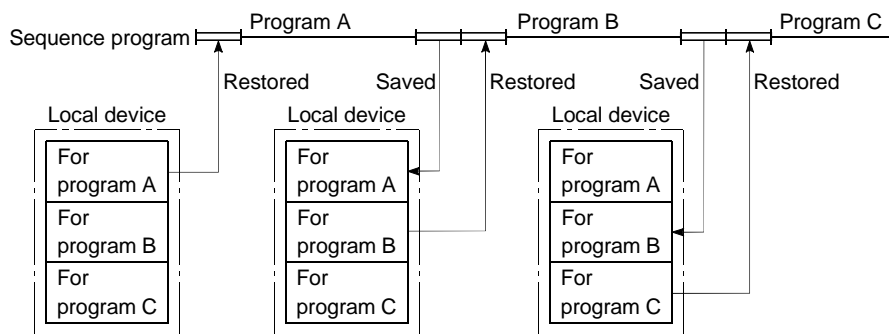


(2) Local devices

- (a) 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.  
 Local devices data can be stored in the standard RAM and the memory card.



- (b) Five device types can be used as local devices: internal relays (M), edge relays (V), timers (T, ST), counters (C), and data registers (D).
- (c) Programs used as local devices exchange the local device file data stored in the memory card with the data in the device memory of High Performance model QCPU.  
 Therefore the scan time is extended by this data exchange time.



**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**

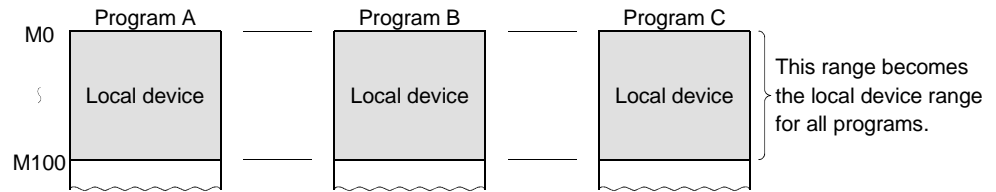
Refer to Section 10.2 for the concept of the number of words of the devices used as local devices.

(d) Local device designation

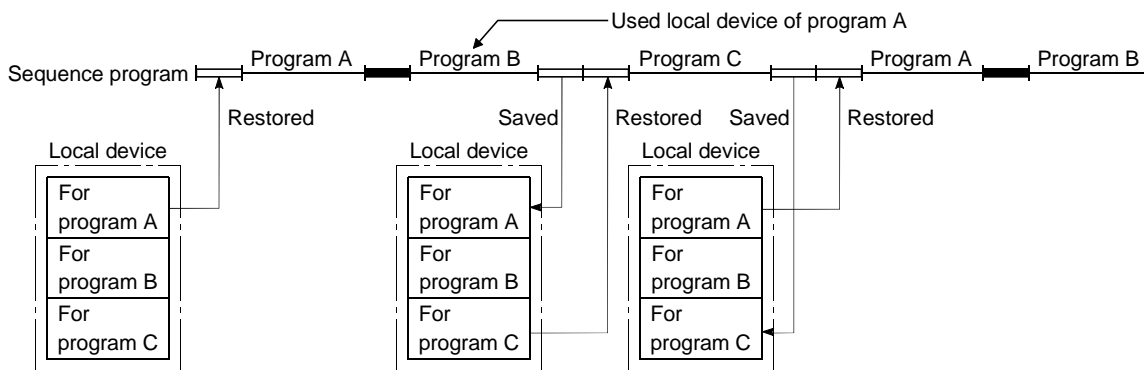
- 1) In order to use the above devices as local devices, the usable local device range must be designated at the "Device" tab screen in the "(PLC) Parameter" dialog box.

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.



- 2) When local device settings are designated, the drive and file name where the local device data is to be stored must be designated at the "PLC file" tab screen in the "(PLC) Parameter" dialog box.
- 3) To write data from the GX Developer onto the High Performance model QCPU, specify whether to use a local device at the "PLC file" tab screen in the "(PLC) Parameter" dialog box. If a local device is not specified, the local devices used for previously-executed programs are selected. This does not require replacing local devices in a memory card with the device memory of the High Performance model QCPU. If local devices are not used for Program B while executing Programs A, B, and C, the local devices are used as shown below.



**POINT**

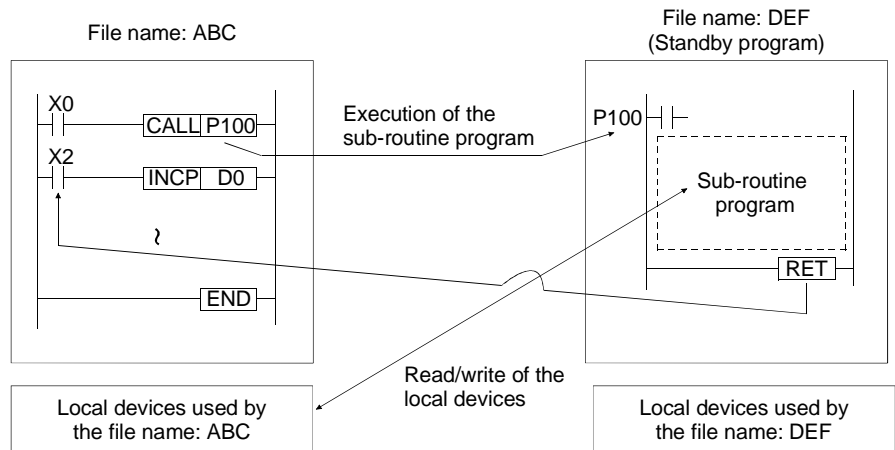
Unless designated as "local devices", all devices are global devices.

- (e) Using local devices used by the file where a sub-routine program is stored  
 It is possible to use local devices that are used by the file where a sub-routine program is stored when executing a sub-routine program.  
 Whether or not such local devices are used is set by special relay "SM776" ON/OFF setting.

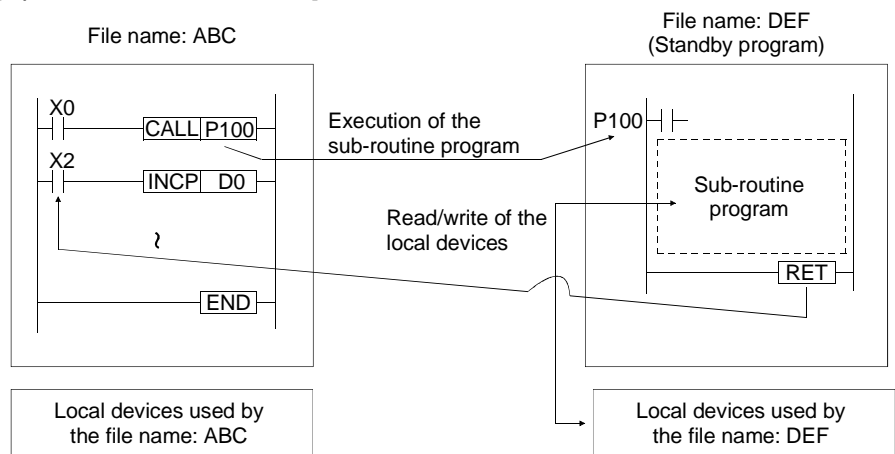
1) Switching over local devices by setting ON/OFF for a special relay (SM776)

	SM776
OFF	Executes calculation by the local devices that are used by the file where the sub-routine program was called
ON	Executes calculation by the local devices that are used by the file where the sub-routine program is stored.

[Operation at "SM776 : OFF"]



[Operation at "SM776 : ON"]



2) Cautions

- If SM776 is ON, the local device data is read when the sub-routine 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 sub-routine program is executed once with the setting of "SM776: ON". (See Section 10.13.1)
- 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, see Appendix 1.



(f) Using local devices when executing an interrupt/fixed scan execution type program

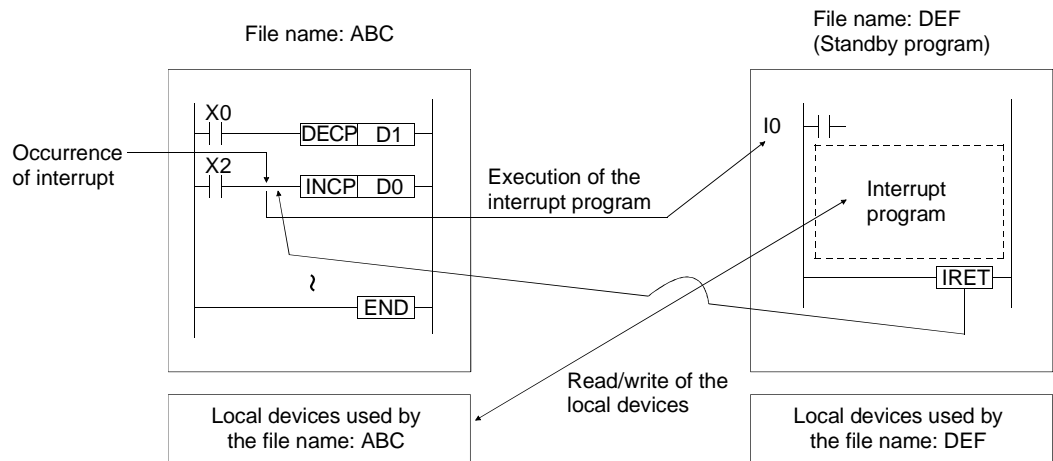
It is possible to use local devices in the file where an interrupt/fixed scan execution type program is stored when executing an interrupt/fixed scan execution type program.

The local devices can be set available/unavailable by special relay "SM777" ON/OFF setting.

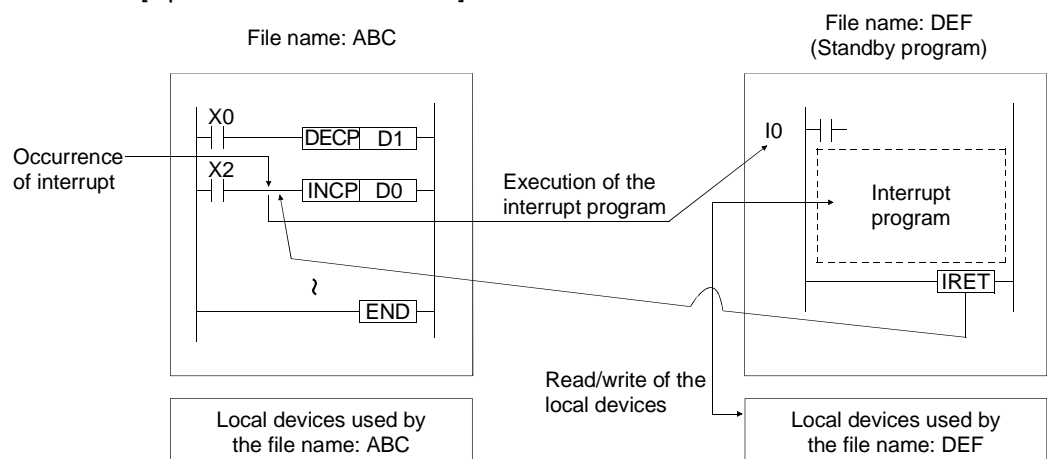
- 1) Switching over local devices by setting ON/OFF for a special relay (SM777)

SM777	
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.

[Operation at "SM777 : OFF"]



[Operation at "SM777 : ON"]



**REMARK**

For details on SM777, see Appendix 1.

## 2) Cautions

- 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". (See Section 10.13.1)
- 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.

## (g) Clearing the Local Device Data

The local device data is cleared in the following cases where:

- 1) The PLC is powered on or the CPU module is reset.
- 2) 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. To clear the local device data, follow the above-listed steps 1) and 2).

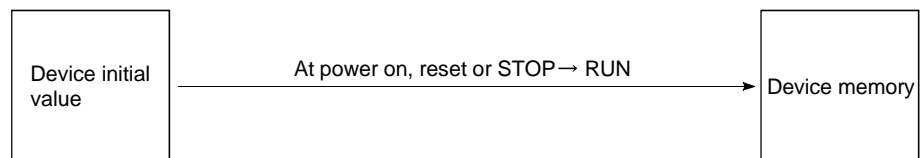
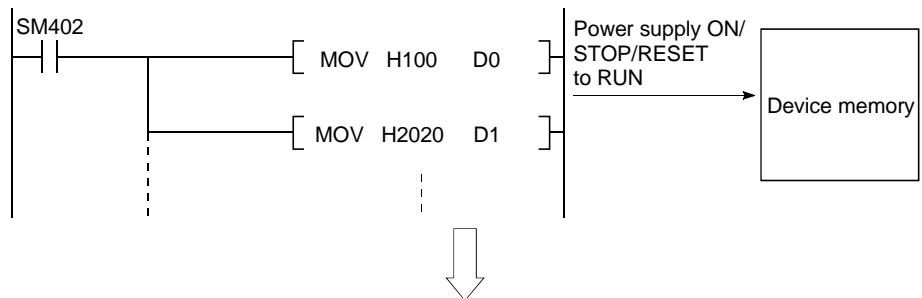
10.13.2 Device initial values

(1) Definition

- (a) Using device initial value registers, the data used for a program in device or intelligent function module buffer memories without using a data setting program.

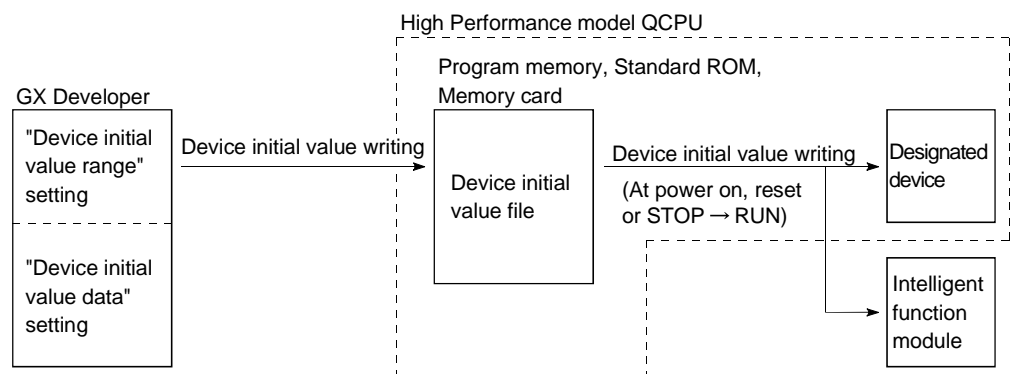
The use of device initial values provides a shortcut to specify device data in a program without using a device data setting program (initial program).

[Data setting by initial processing program]



- (b) 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 High Performance model QCPU's program memory, standard RAM or memory card.

At power ON, or on switching from STOP to RUN, the High Performance model QCPU writes the data from the device initial value file to the specified device or intelligent function module buffer memory.



(c) Device initial values can be used by the following devices:

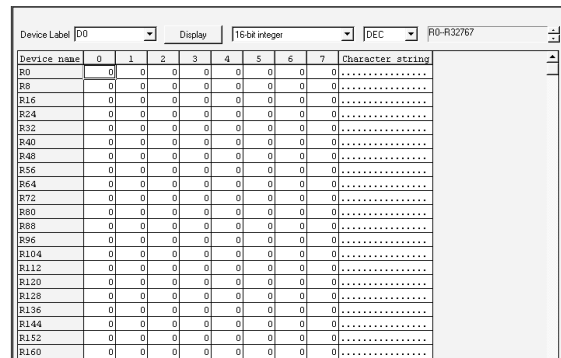
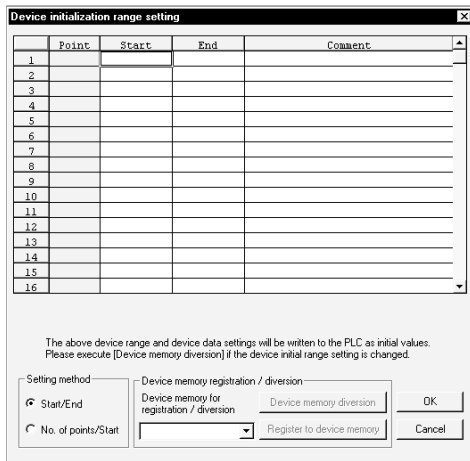
- 1) Timer present value (T)
- 2) Retentive timer present value (ST)
- 3) Counter present value (C)
- 4) Data register (D)
- 5) Special register (SD)
- 6) Link register (W)
- 7) Link special register (SW)
- 8) File register (R0 to R32767)
- 9) File register (ZR0 to ZR1042431)
- 10) Intelligent function module device (U[ ][G ])
- 11) Link direct device (J[ ][W ], J[ ][SW ])

(2) Procedure for using device initial values

- (a) Designate the device initial value range settings in the device mode, in the "Device initial value setting" screen.
- (b) Designate the device initial value data settings in the "device mode" screen.

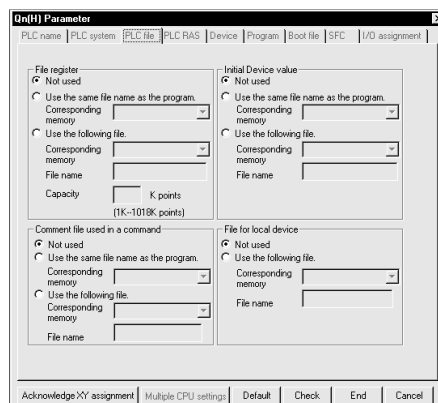
[Device initialization range setting screen]

[Device mode screen]



(c) 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.

[PLC file screen]



(d) Write the device initial value data and parameter settings to the High Performance model QCPU.

### (3) Precautions for the use of device initial values

- (a) In cases where both device initial value data and latch range data are overlapped, the device initial value data takes priority.  
Therefore, the latch range data is rewritten by device initial value data at power ON.
- (b) Device initial values cannot be used in areas where no setting is made for switching from STOP to RUN (for data that is changed by a program at power ON). Create a program to specify a device by using the MOV instruction in the main routine program. Use the TO instruction to write data to the buffer memory of the intelligent function module.

#### REMARK

For details on the setting procedures for the "device initial value range", "device initial value data" items, and for writing the device initial values to the High Performance model QCPU, refer to the GX Developer Operating Manual.

## 11 HIGH PERFORMANCE MODEL QCPU PROCESSING TIME

This chapter describes how to estimate the length of High Performance model QCPU processing time.

### 11.1 Reading High Performance model QCPU's Scan Time

The length of scan time is the total of the following times:

- I/O refresh time
- Instruction execution time
- END processing time

#### (1) I/O refresh time

(a) I/O refresh time is the total time required for refreshing I/O data of the following modules:

- Input module
- Output module
- Intelligent function module (special function module)

(b) I/O refresh time is given in the formula:

$$(I/O \text{ refresh time}) = (\text{Number of inputs point}/16) \times N1 + (\text{Number of outputs point}/16) \times N2$$

(c) The table below shows N1 and N2.

CPU type	N1			N2		
	Q3□B	Q5□B /Q6□B	QA1S□B	Q3□B	Q5□B /Q6□B	QA1S□B
Q02CPU	2.2 μs	2.9 μs	4.3 μs	1.3 μs	2.1 μs	3.5 μs
Q02HCPU Q06HCPU Q12HCPU Q25HCPU	1.7 μs	2.4 μs	3.7 μs	1.3 μs	2.1 μs	3.5 μs

#### (2) Instruction execution time

(a) Instruction execution time is the total processing time required to execute an instruction in a program on the High Performance model QCPU.

For details on the execution time of each instruction, see the QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions).

(b) Overhead time is required for an interrupt program/fixed scan execution type program. Add overhead time to execution time.

#### (3) END processing time

(a) END processing time is the High Performance model QCPU processing time common to the above-listed items (1) and (2).

(b) The table below shows the length of END processing time.

CPU Type	END Processing Time
Q02CPU	0.38 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.15 ms

## 11.2 Factors Responsible for Extended Scan Time

The following functions increase the length of scan time. When using any of the following functions, add a value of extended time to values obtained from Section 11.1.

- MELSECNET/H refresh
- CC-Link automatic refresh
- Sampling trace
- GX Developer monitoring
- Local devices
- Execution of multiple programs
- Installation/removal of a memory card
- File register with the same filename as a program.

### (1) MELSECNET/H refresh

MELSECNET/H refresh requires additional processing time to refresh data between the High Performance model QCPU and the MELSECNET/H network module. For details on MELSECNET/H refresh time, see Q Corresponding MELSECNET/H Network System Reference Manual.

### (2) CC-Link automatic refresh

Refresh time between the High Performance model QCPU and CC-Link master/local module.

#### (a) When module is QJ61BT11N

Refer to the following manual for CC-Link automatic refresh processing time.

- QJ61BT11N CC-Link System Master/Local Module User's Manual (Details)

#### (b) When module is QJ61BT11

Refer to the following manual for CC-Link automatic refresh processing time.

- QJ61BT11 CC-Link System Master/Local Module User's Manual (Details)

(3) Sampling trace time

Sampling trace requires additional processing time. When sampling trace data is specified to execute the sampling trace function, add the sampling trace time to the total processing time.

The table below shows the length of processing time required when sampling trace data is specified to assign 50 internal relay points for bit devices and 50 data register points for word devices.

CPU Type	Processing Time
Q02CPU	0.24 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.12 ms

(4) GX Developer Monitoring

GX Developer monitoring requires additional processing time. Add the GX Developer monitoring time to the total processing time.

(a) The table below shows the processing time required when 64 data register points are assigned by the registered monitor.

CPU Type	Processing Time
Q02CPU	0.10 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.06 ms

(b) The table below shows the processing time required when monitoring conditions are specified.

CPU Type	Processing Time	
	When steps are in match	When devices are in match
Q02CPU	0.05 ms	0.01 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.03 ms	0.01 ms

(5) Local devices

Local devices require additional processing time. Add the processing time of local devices to the total processing time.

CPU Type		Processing Time
Standard RAM	Q02CPU	$0.94+0.40 \times n$ ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$0.39+0.17 \times n$ ms
SRAM Card	Q02CPU	$0.94+1.38 \times n$ ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$0.39+0.95 \times n$ ms

Conditions Local devices: 1k points, n: number of program files

(6) Execution of multiple programs

Execution of multiple programs requires overhead time of each program being executed. Add overhead time to the total processing time.

CPU Type	Processing Time
Q02CPU	$0.08 \times n$ ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$0.03 \times n$ ms

Conditions n: Number of program files



## (7) Installation/Removal of a memory card

Installation/removal of a memory card requires additional processing time. If a memory card is installed or removed, add 1-scan time to the total processing time.

CPU Type	Processing Time	
	Memory Card Inserted	Memory Card Removed
Q02CPU	0.16 ms	0.10 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.08 ms	0.04 ms

## (8) File register

File register requires additional processing time. Add the processing time of file registers to the total processing time.

CPU Type		Processing Time
Standard RAM	Q02CPU	1.03 ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.41 ms
SRAM Card	Q02CPU	$0.94+0.2 \times n$ ms
	Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	$0.40+0.1 \times n$ ms

Conditions n: number of program files

## 11.3 Factors Responsible for Shortened Scan Time

The length of scan time can be shortened by making changes to the PLC Parameter setting as follows:

- A series CPU compatibility
- Arithmetic operation of floating-point

## (1) A series CPU compatibility

When "Use special relay/special register after SM1000/SD1000" is set in the PLC system settings of the PLC parameter, the scan time can be reduced by the value in the following table by setting to "Do not use special relay/special register after SM1000/SD1000."

In this case, the A-series compatible special relays/special registers SM1000/SD1000 to SM1299/SD1999 must be replaced with the Q-series dedicated special relays/special registers SM0/SD0 to SM999/SD999.

CPU Type	Processing Time
Q02CPU	0.07 ms
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	0.03 ms

## (2) Arithmetic operation of floating-points

By default, the "Perform internal arithmetic operations in double precision" check box in the PLC System sheet of the PLC Parameter dialog box is on. Turning off the check box will increase the processing speed of executing an instruction that includes a floating-point.

For details on floating-point arithmetic operation, see the QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions).

## 12 PROCEDURE FOR WRITING PROGRAMS TO HIGH PERFORMANCE MODEL QCPU

This chapter describes the procedure for writing programs created at the GX Developer to the High Performance model QCPU.

### 12.1 Writing Procedure for 1 Program

This section describes the procedure for writing one program to the High Performance model QCPU and executing it.

#### 12.1.1 Items to consider when creating one program

In order to create a program, the program size, number of device points used, and the program file name, etc., must be set in advance.

##### (1) Program size considerations

Check that CPU module's program capacity is adequate for storing the program and parameter data.

The program capacities of the CPUs are shown below:

- Q02CPU : 28 k steps
- Q02HCPU : 28 k steps
- Q06HCPU : 60 k steps
- Q12HCPU : 124 k steps
- Q25HCPU : 252 k steps

If the CPU module capacity is only adequate for the program, the parameter data should be stored in the standard ROM/memory card.

##### (2) Designating a program file name

The file name of the program to be stored in the High Performance model QCPU must be designated.

This file name is used when writing the program and parameters from GX Developer to the High Performance model QCPU, and when designating the program to be executed in the High Performance model QCPU.

See Chapter 6 for details regarding file names.

##### (3) Designating devices

The number of devices required for the program must be determined.

See Chapter 10 for details regarding devices which can be used in the High Performance model QCPU.

##### (4) Device initial value setting

Designate whether or not the device initial value settings are to be used in the High Performance model QCPU devices and intelligent function modules.

See Section 10.13.2 for details regarding device initial values.

12.1.2 Procedure for writing programs to the High Performance model QCPU

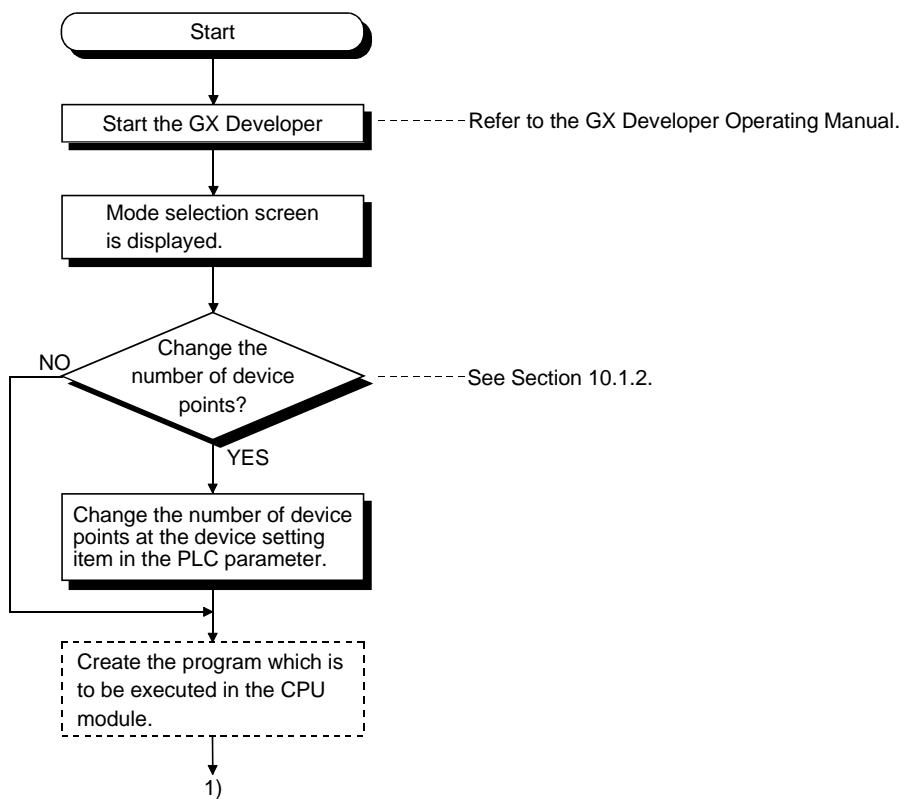
The procedure for writing programs and parameters created with GX Developer to the High Performance model QCPU standard ROM is shown below.

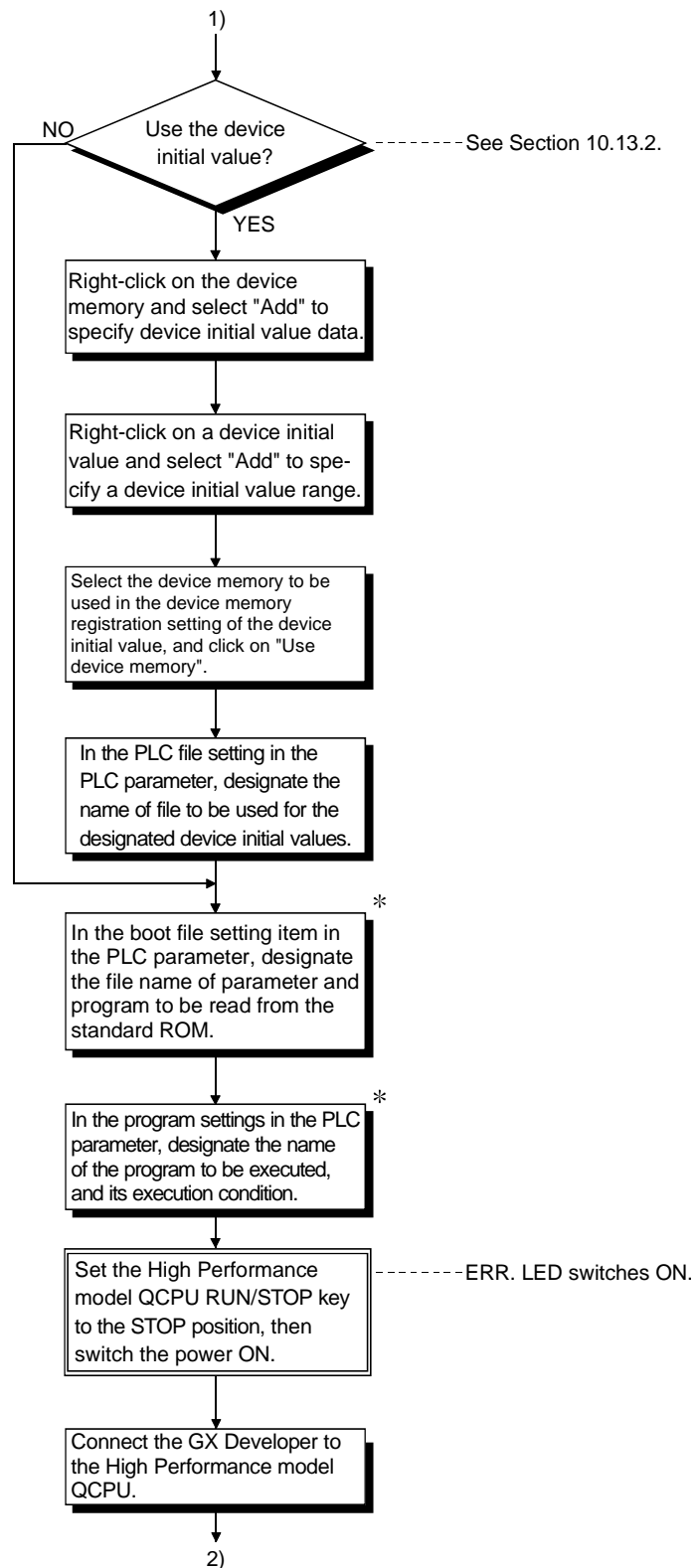
In order to write programs and parameters to the High Performance model QCPU standard ROM, the valid parameters settings must be designated by the High Performance model QCPU DIP switches (SW2, SW3), and the boot settings must be designated in the PLC parameter mode.

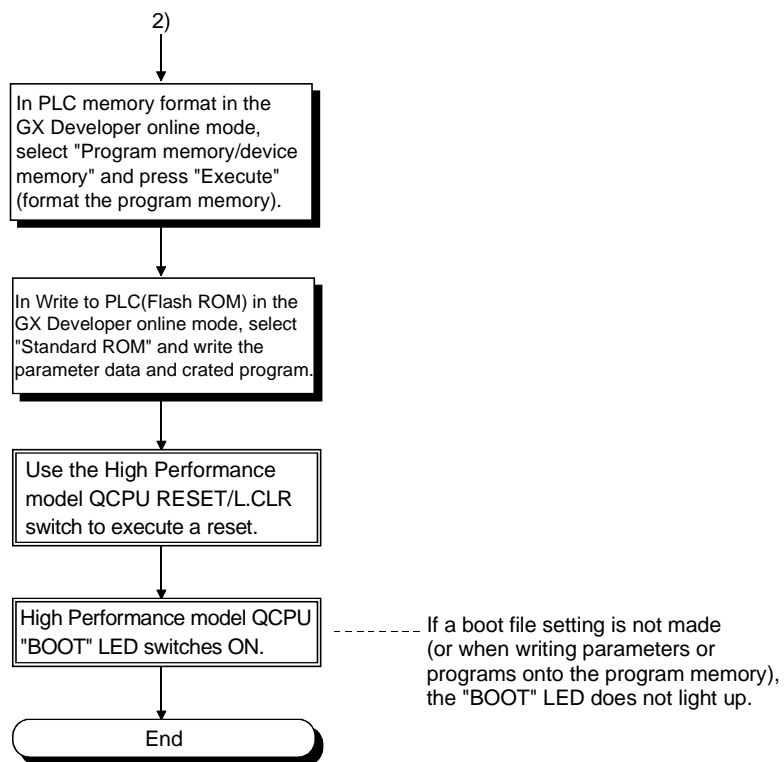
For details regarding High Performance model QCPU DIP switches, refer to the High Performance model QCPU(Q mode) User's Manual(Hardware Design, Maintenance and Inspection).

When writing programs and parameters to the High Performance model QCPU program memory, the steps indicated by asterisks (\*) below are not required.

Procedural steps shown in □ boxes are performed at the GX Developer, and those shown in ▭ boxes are performed in the High Performance model QCPU.







## 12.2 Procedure for Multiple Programs

This section describes the procedure for writing multiple programs split up according to function, process, designer to the High Performance model QCPU and executing them.

### 12.2.1 Items to consider when creating multiple programs

To create multiple programs, it is necessary to set in advance the size of each program, the device used, and the program file name, etc.

#### (1) Program size considerations

Set the program capacity within the range of the High Performance model QCPU program capacity.

The program capacities of the High Performance model QCPU's are shown below:

- Q02CPU : 28 k steps
- Q02HCPU : 28 k steps
- Q06HCPU : 60 k steps
- Q12HCPU : 124 k steps
- Q25HCPU : 252 k steps

Decide whether the parameters are to be stored in the program memory, the standard ROM, or the memory card.

If they are to be stored in the program memory/the standard ROM, the area available for the program will be the capacity shown above, minus the parameter data size.

#### (2) Designating a program file name

Designate the file name of the program to be stored in the High Performance model QCPU.

This file name is used when writing the program and parameters from GX Developer to the High Performance model QCPU, and when designating the program to be executed in the High Performance model QCPU.

See Chapter 6 for details regarding file names.

#### (3) Designating the program execution conditions

In order to execute multiple programs in the High Performance model QCPU, execution conditions must be designated for each program.

Execution is impossible for programs without file name and execution condition settings.

See Section 4.2 for details regarding execution conditions.

#### (4) Designating devices

(a) Designate the number of device points used in each program, and the number of device points which are shared by all programs.

See Chapter 10 for details regarding devices which can be used in the High Performance model QCPU.

(b) Designate whether or not the internal relays, edge relays, timers, counters, and data registers of each program are to be designated as local devices. See Section 10.13.1 for details regarding local devices.

(c) When creating sub-routine programs, designate whether or not common pointers are to be used.

See Section 10.9.2 for details regarding common pointers.

#### (5) Device initial value setting

Designate whether or not the device initial value settings are to be used for the High Performance model QCPU devices and intelligent function modules.

See Section 10.13.2 for details regarding device initial values.

12.2.2 Procedure for writing programs to the High Performance model QCPU

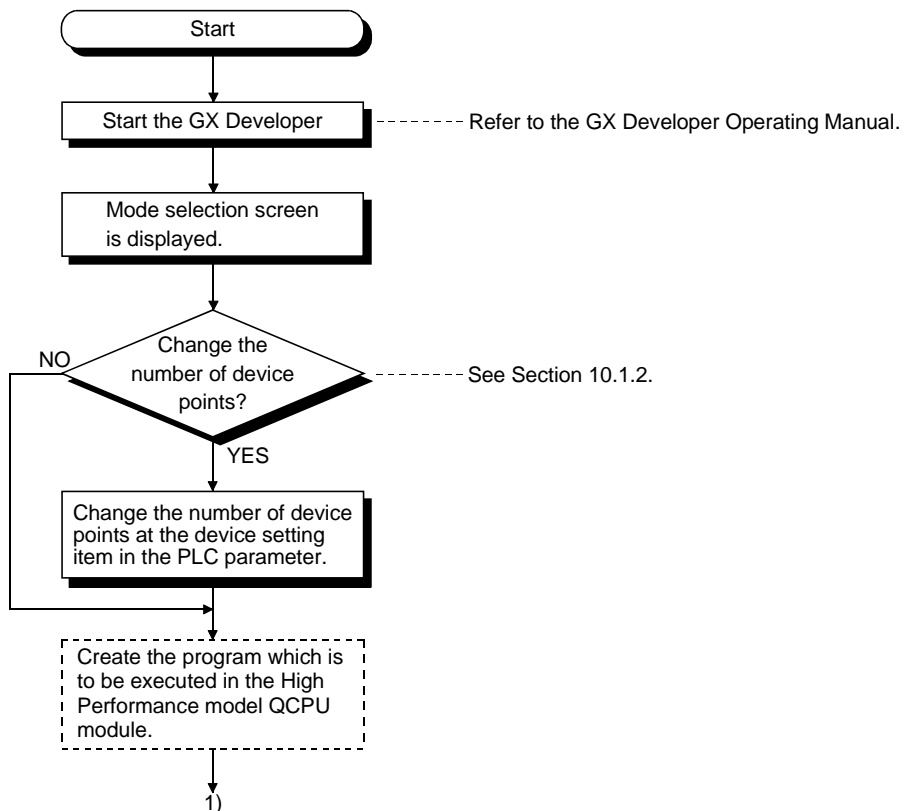
The procedure for writing programs and parameters created by GX Developer to the memory card mounted in the High Performance model QCPU memory card interface is shown below.

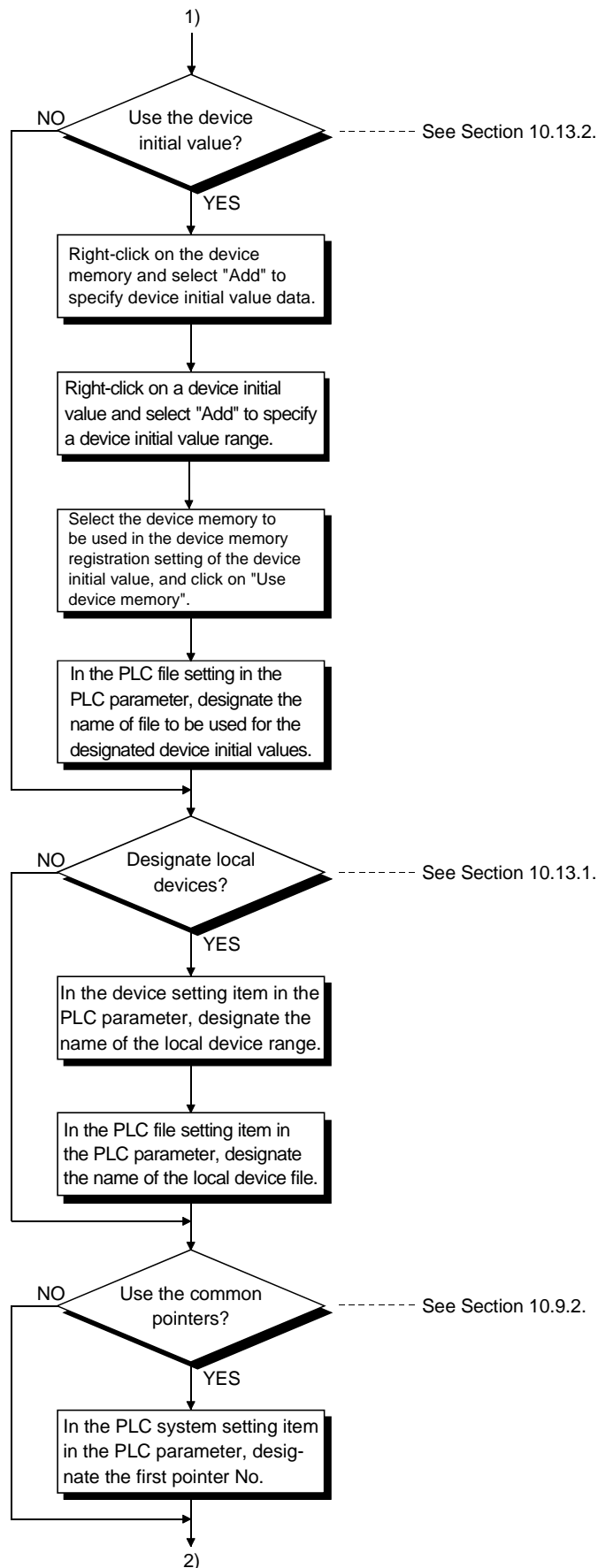
In order to write programs and parameters to the High Performance model QCPU memory card, the memory card must be mounted, the valid parameters drive settings must be designated by the High Performance model QCPU DIP switches (SW 2, SW 3), and the boot settings for the PLC parameters must be designated by GX Developer.

For details regarding High Performance model QCPU DIP switches, refer to the High Performance model QCPU (Q mode) User's Manual (Hardware Design, Maintenance and Inspection).

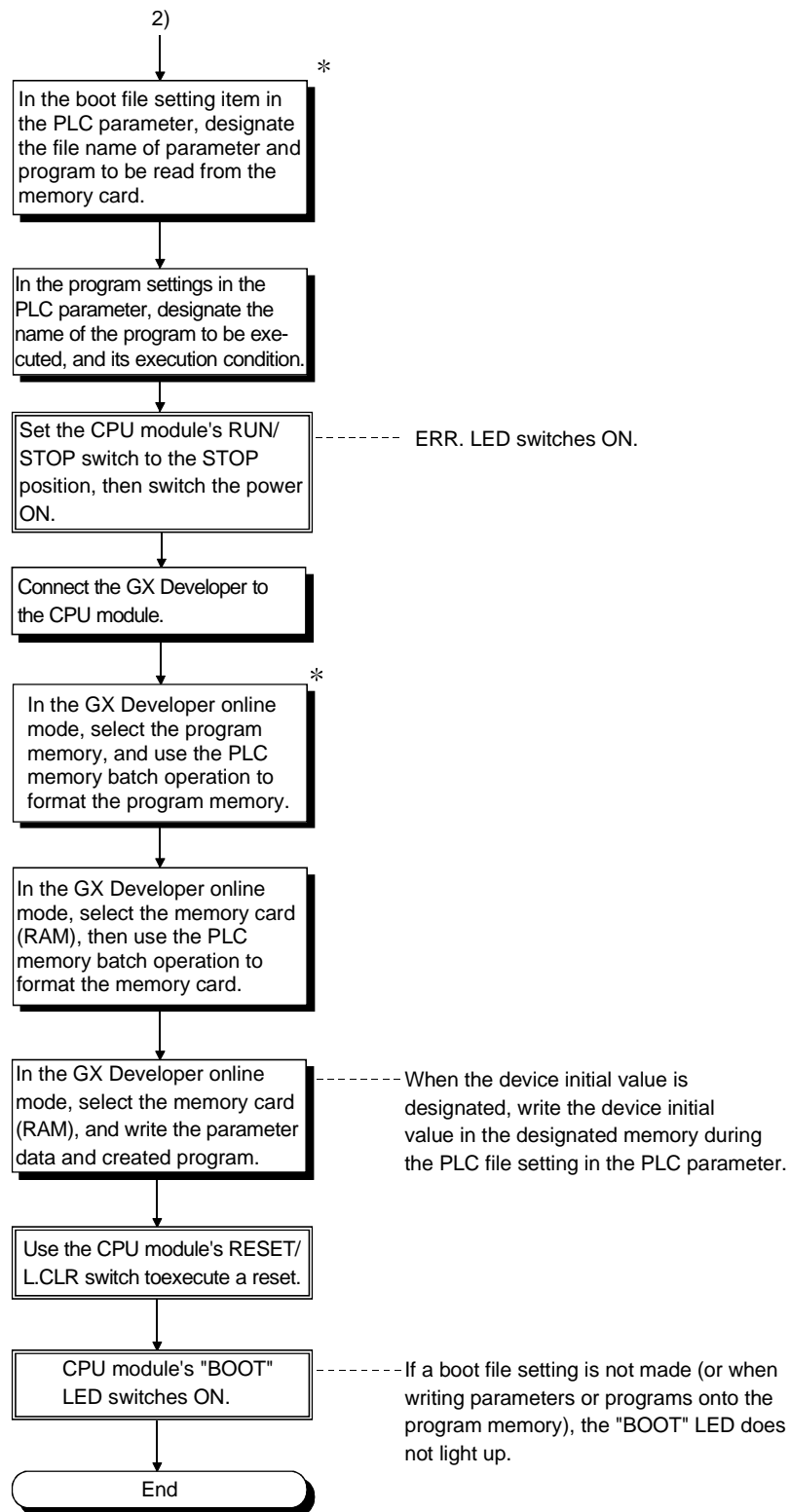
When writing programs and parameters to the High Performance model QCPU program memory, the steps indicated by asterisks (\*) below are not required.

Procedural steps shown in □ boxes are performed at GX Developer side, and those shown in ▭ boxes are performed at the High Performance model QCPU side.









13 OUTLINE OF MULTIPLE CPU SYSTEM

13.1 Features

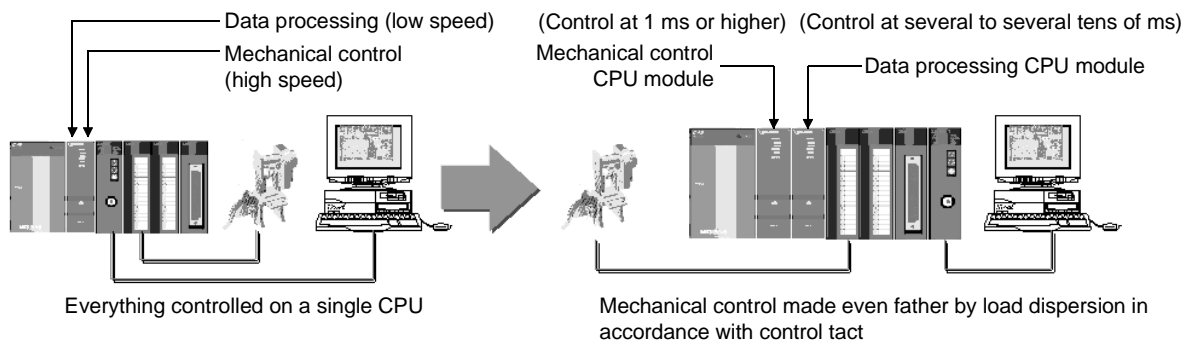
(1) Multi control

- (a) Since each system is not configured on one High Performance model QCPU but on the High Performance QCPU, Motion CPU, and PC CPU module according to the system, the development efficiency and ease of maintenance of the system can be enhanced.
- (b) Each CPU module in the multiple CPU system controls the I/O module and intelligent function module of the base unit slot-by-slot. GX Developer groups the I/O modules and intelligent function modules controlled by each CPU module in the multiple CPU system.

13

(2) Enables system configuration through load dispersion

- (a) By dispersing the high-load processing performed on a single High Performance model QCPU between several High Performance model QCPU, it is possible to reduce the overall system scan time.



- (b) It is possible to increase the amount of memory used throughout the entire system by spreading the memory used between several High Performance model QCPU.



(3) Enables system configuration through function dispersion

By dispersing the functions so that control for production line A and control for production line B is performed on different High Performance model QCPU, it is possible to debug each function individually.

(4) Communication can be made between CPU modules in the multiple CPU system

The following data transfer can be made between CPU modules in the multiple CPU system.

- (a) Automatic refresh setting at GX Developer enables the data transfer between CPU modules.
- (b) The High Performance model QCPU can use the FROM/S.TO instruction to read data from other CPU as necessary.
- (c) Instructions dedicated to Motion can be used to issue control commands from the High Performance model QCPU to the Motion CPU. \*1
- (d) The High Performance model QCPU can issue instructions dedicated to communication between multiple CPUs, to read or write device data from/to the Motion CPU.  
The High Performance model QCPU can issue events to the PC CPU module. \*2

**REMARK**

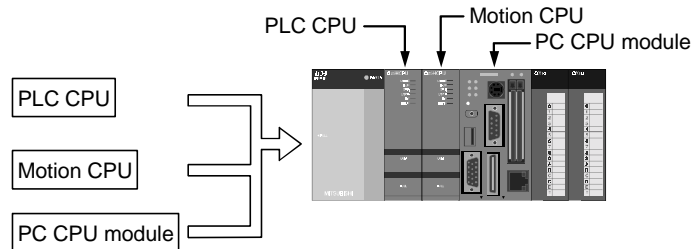
\*1: Refer to the manual of the Motion CPU for instructions dedicated to Motion.

\*2: Refer to the manuals of Motion CPU and PC CPU module for instructions dedicated to the communication between multiple CPUs.

13.2 Outline of Multiple CPU System

(1) What is a multiple CPU system?

- (a) A multiple CPU system is a system in which main base units are mounted on several (maximum four) High Performance model QCPUs and motion CPU in order to control the I/O modules and intelligent function modules.

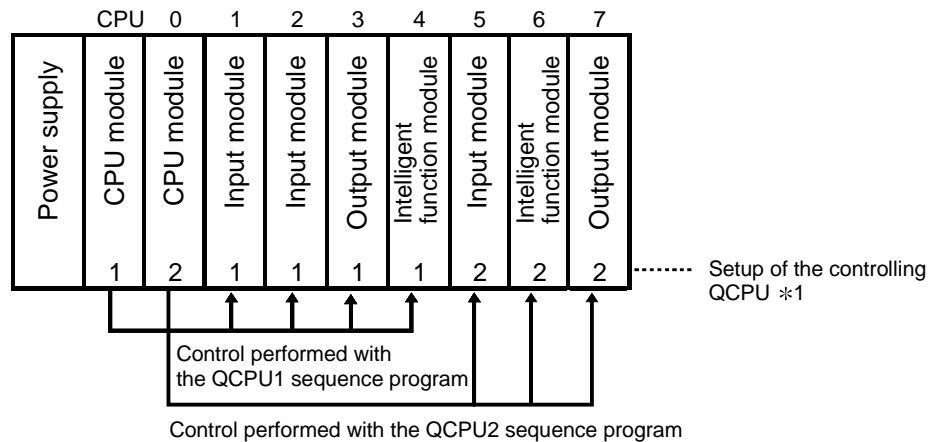


The allowable CPU modules are shown in the table below.

PLC CPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU,
Motion CPU	Q172CPU, Q173CPU, Q172CPUN, Q173CPUN
PC CPU module	CONTEC Co., Ltd.

Choose the CPU modules suitable to the system size and application to configure the system.

It is necessary to set (control CPU setup) which High Performance model QCPU and Motion CPUs are to control which I/O modules and intelligent function modules with a multiple CPU system.



- (b) The CPU module that controls the I/O modules and intelligent function modules is known as the "Control CPU". The I/O modules and intelligent function modules controlled by the control PLC are known "controlled modules". Other modules not controlled by the control CPU are known as "non-controlled modules".

**REMARK**

\*1: For further information on PC CPU module, consult CONTEC Co.,Ltd.

Tel:+81-6-6472-7130

\*2: Indicates the grouping configuration on the GX Developer.

CPU module1 indicates the "CPU No.1," and "1" on the I/O module and intelligent function module indicates that their control CPU is the CPU No.1.

(2) Multiple CPU system setup

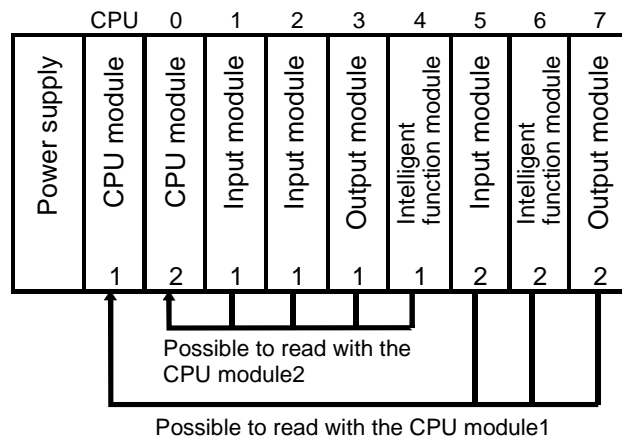
It is necessary to set up the "Number of mounted CPU modules" and the "Control CPU" with PLC parameters in all CPU modules onto which main base units are mounted in order to control a multiple CPU system (see Chapter 9.)

(3) Multiple CPU system access range

- (a) It is possible for a multiple CPU system's control CPU to perform the I/O refresh procedure on control modules and write in the buffer memory of intelligent function modules in the same way as a single CPU system.
- (b) It is possible to access non-control modules in the following ways.
  - Refresh the input for I/O modules and intelligent function modules (the PLC parameter multiple CPU setup is necessary.)
  - Read the intelligent function module's buffer memory.
  - Download the output data for the output module, the I/O combination module and the intelligent function modules. (the PLC parameter multiple CPU setup is necessary.)

However, it is not possible to access non-control modules in the following ways.

- Outputting data to the output module and intelligent function module.
- Writing data into the intelligent function module's buffer memory.



(4) GX Developer access range

- (a) It is possible to write parameter programs and perform monitoring and tests in High Performance model QCPUs connected to personal computers, To access High Performance model QCPUs that are not connected to personal computers, specify the High Performance model QCPU to be accessed (connection destination specification) with the GX Developer.
- (b) It is possible for the GX Developer to access the High Performance model QCPU regardless of the control modules and non-control modules. By connecting a single High Performance model QCPU to a personal computer, it is possible to perform monitoring and tests on all modules being controlled by the multiple PLC system's High Performance model QCPU in the same way as with a single CPU system. Other station High Performance model QCPUs on the same MELSECNET/H, Ethernet or other network can also be accessed.
- (c) It is possible for all High Performance model QCPUs on a multiple CPU system to be accessed from a GX Developer that is connected to other stations on the same network.

### 13.3 Differences with Single CPU System

The differences between single CPU system and multiple CPU system are explained below.

- (1) **Function versions (see Sections 14.2.1 to 14.2.5)**
  - (a) Function version B High Performance model QCPUs are supported by multiple CPU system. Function version A High Performance model QCPUs cannot be used on a multiple CPU system.
  - (b) All I/O modules can be used on a multiple CPU system.
  - (c) Use function version B intelligent function modules on the multiple CPU system.  
Function version A intelligent function modules can be used if set up as control CPU by the CPU No.1.
- (2) **Mounting position of CPU module (see Section 14.2.1)**

High Performance model QCPUs can be mounted on the CPU slot from the right-hand side of the power unit sequentially. The motion CPUs are mounted together on the slot to the right of the High Performance model QCPUs. The PC CPU module is installed on the extreme right side in the multiple CPU system.

The total number of High Performance model QCPU, Motion CPU and PC CPU module must be up to four.
- (3) **Multiple CPU system parameters (see Section 14.2.6)**

In comparison with independent CPU systems, there are more PLC parameter items on a multiple CPU system.

Of the PC parameters that have been added to the multiple CPU system, the parameters that must be set are listed below.

  - Number of CPUs: Sets the number of mounted High Performance model QCPUs, motion CPUs and PC CPU module that are in use.
  - Control CPU settings: Sets which High Performance model QCPU, Motion CPU and PC CPU module controls while modules.
- (4) **Sameness check (see Section 14.2.6)**

A setting exists to indicate that the QCPU / motion CPU / PC CPU module used are the same in the number of CPUs, control CPU settings and other multiple CPU system parameters.

The High Performance model QCPU / Motion CPU / PC CPU module run a check (sameness check) to ascertain that the multiple CPU system parameters are the same when the PLC power is set at ON, the High Performance model QCPU is reset, and the STOP status is changed to the RUN status.

The multiple PLC system will not start up if an error is triggered during the sameness check.
- (5) **Concept of the I/O number (see Section 15.1)**

The right side of the installed CPU module is I/O number "00H" in the multiple CPU system. For this reason, the position of I/O number "00H" varies according to the number of installed CPUs. However, because each PC CPU module occupies two slots (one slot for CPU and one empty slot), the I/O number deviates by the number of points set to the empty slot. (Default: empty 16 points)

(6) Interactive transmission with non-control CPUs (see Chapter 17)

(a) It is possible to control I/O modules and intelligent function modules controlled by the host CPU in the same way as on a single CPU system.

(b) It is not possible to output ON/OFF data to modules that are not controlled by the host CPU or write in the buffer memory of intelligent function modules.

It is possible to read I/O data from non-control modules with PLC parameter settings.

It is possible to confirm the status of modules controlled by other CPUs, the control status of other CPUs, and control the host CPU.

(7) Interactive transmission between each CPU modules on a multiple CPU system (see Chapter 16)

It is possible to perform the following interactive transmission between each CPU modules with a multiple CPU system.

(a) Automatically refreshing the device data between each CPU modules with multiple CPU system parameter settings.

(b) Data transfer between other High Performance model QCPU and PC CPU module via CPU shared memory using multiple CPU instructions (FROM, S.TO instructions)

(c) Reading CPU shared memory of Motion CPU from High Performance model QCPU with multiple CPU instructions (FROM instruction).

(d) Control instruction from the High Performance model QCPU to the Motion CPU with Motion dedicated CPU instructions.

(e) Writing and reading of the device data from the High Performance model QCPU to the Motion CPU with communication dedicated instructions between multiple CPUs.

(f) Event issuance from High Performance model QCPU to PC CPU module using instructions dedicated to multiple CPU communication

(8) Processing during resets and errors (see Sections 14.2.7 and 14.2.8)

The processing performed when resets and errors occur are different for the multiple CPU system's CPU No.1 and the CPU No.2 to CPU No.4.

(a) High Performance model QCPU for CPU No.1 can be reset with a multiple CPU system.

CPU modules for CPU No.2 to No.4 and Motion CPU cannot be reset.

(b) Multiple CPU system operations will be suspended when a stop error occurs with the CPU No.1.

It is possible to select whether to suspend or continue with multiple CPU system operations when a stop error occurs with CPU Nos. 2 to 4 and Motion CPU.

(9) Clock function

An intelligent function module with which the error code and time of occurrence is stored in the buffer memory when an error is triggered is available (time data read from the High Performance model QCPU.)

The CPU No.1 time data will be stored as the time that the error occurred regardless of whether the module concerned is a control CPU or a non-control CPU.

## 14 SYSTEM CONFIGURATION OF MULTIPLE CPU SYSTEM

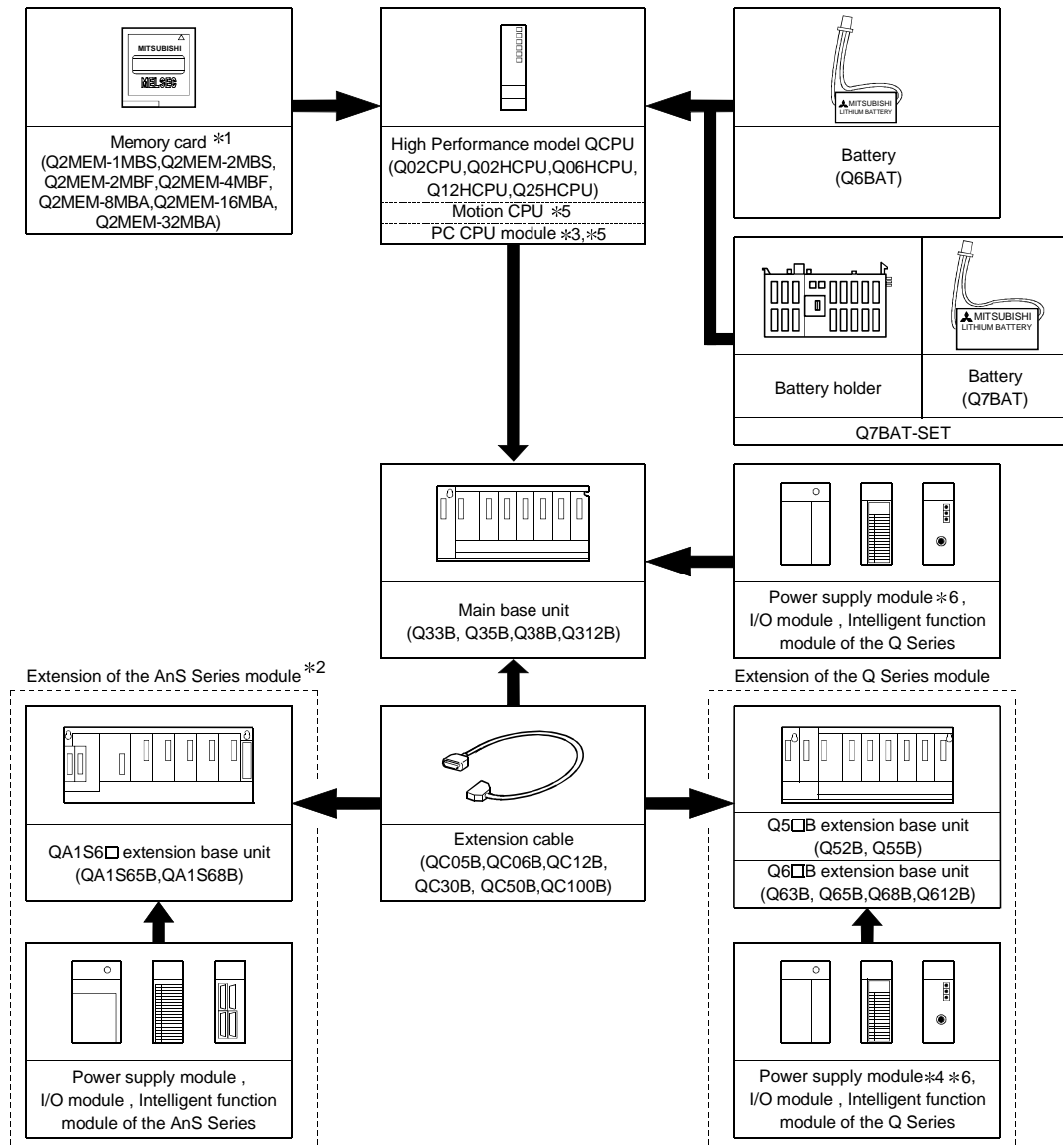
This chapter explains the system configuration of Multiple CPU System, and the precautions for Multiple CPU System configuration.

### 14.1 System Configuration

This section explains the equipment configuration of Multiple CPU System, the connections with peripheral device, and an output of the system's configuration.

#### (1) Equipment configuration of Multiple CPU System

(a) When main base unit (Q3□B) is used

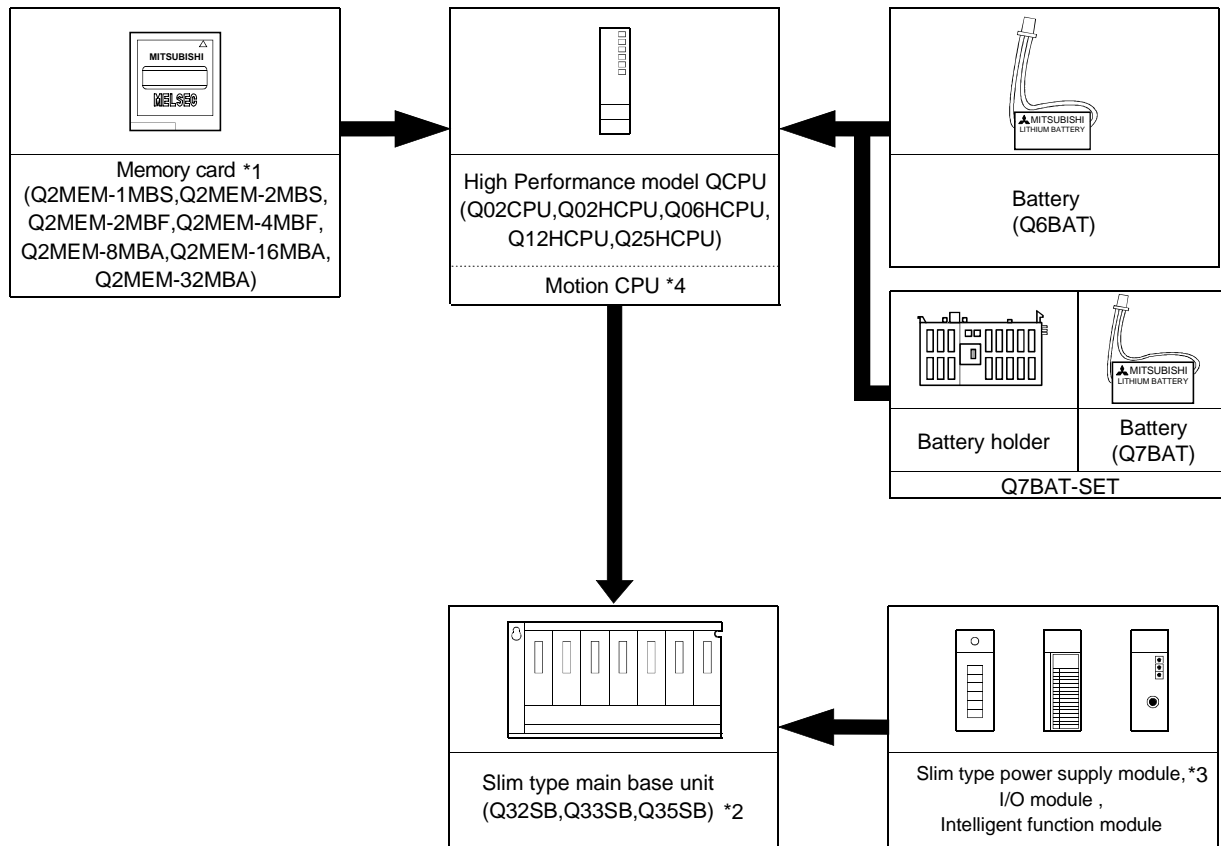


#### POINT

- \* 1: Only one memory card can be mounted.  
Select the memory card SRAM, Flash and ATA in accordance with application and capacity.  
When commercial memory cards, the operation is not assured.
- \* 2: The additional QA1S65B and QA1S68B base units are used as the AnS Series power supply module, the I/O module and the special function module.
- \* 3: For further information on PC CPU module, consult CONTEC Co., Ltd Tel: +81-6-6472-7130
- \* 4: The Q Series power supply module is not required for the Q5□B extension base unit.
- \* 5: The motion CPU and PC CPU module do not accept a battery..
- \* 6: As a power supply module, use the Q61P-A1, Q61P-A2, Q62P or Q64P. The slim type power supply module (Q61SP) cannot be used.



(b) When slim type main base unit (Q3□SB) is used

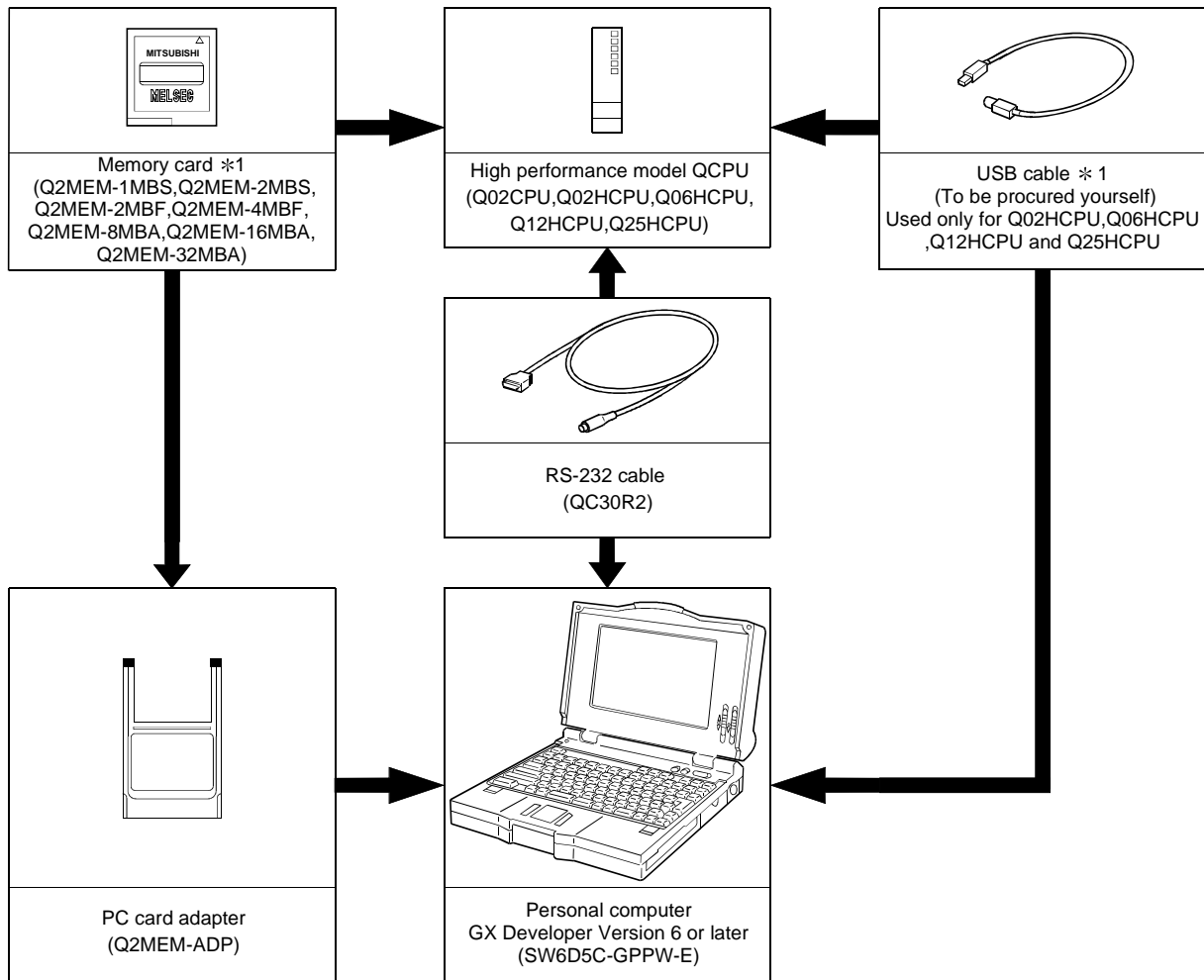


14

**POINT**

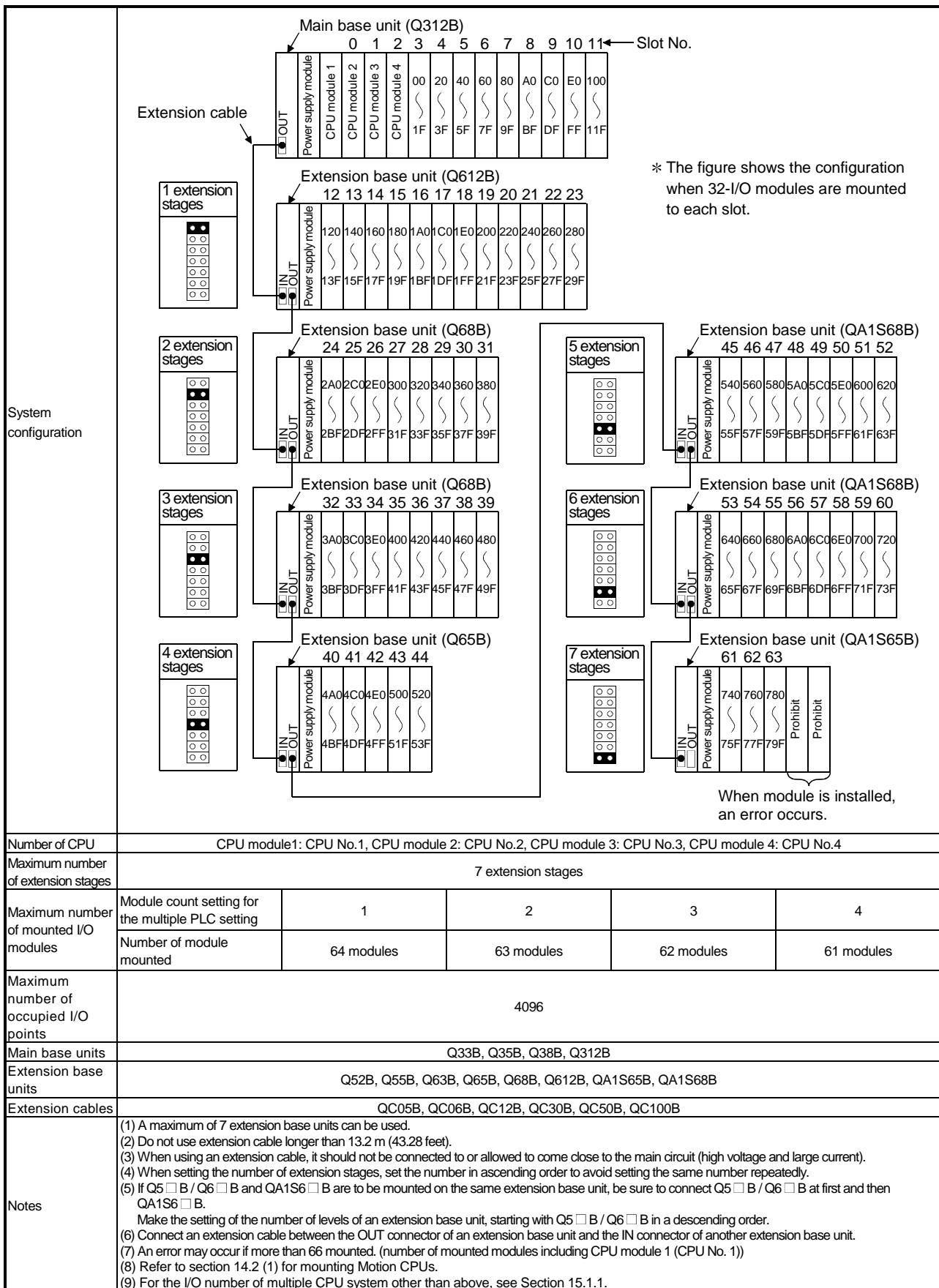
- \*1: One memory card is installed.  
Select the memory card from the SRAM card, Flash card and ATA card according to the application and capacity.  
When the memory card available on the market is used, operation is not guaranteed.
- \*2: The slim type main base unit does not have an extension cable connector. The extension base or GOT cannot be connected.
- \*3: As a power supply module, use the slim type power supply module (Q61SP).  
The Q61P-A1, Q61P-A2, Q62P or Q64P cannot be used as a power supply module.
- \*4: The Motion CPU do not accept a battery.

(2) Configuration of peripheral devices for High Performance model QCPU



\*1: For writing into memory card on GX Developer, and USB cable, refer to the operating manual of the GX Developer.

POINT
1) Refer to the Motion Controller User's Manual for connection between the Motion CPU and peripheral modules.
2) The GX Developer installed in a Personal computer connected to the Motion CPU is not used to communicate with the High Performance model QCPU.
3) You cannot install GX Developer and Motion CPU software package in a single PC.
4) Refer to the manual of the PC CPU module for the connection between the PC CPU module and peripheral modules.



## 14.2 Precautions For Multiple CPU System Configuration

### 14.2.1 Function versions of High Performance model QCPU, Motion CPUs and PC CPU module that can be used, and their mounting positions

#### (1) Function versions

##### (a) Allowable function versions and version confirmation method

- 1) To configure a multiple CPU system, use the High Performance model QCPU and Motion CPU of function version B.
- 2) To use the PC CPU module, use the following High Performance model QCPU and Motion CPU.
  - High Performance model QCPU: First five digits of the serial No. is "03051" or later and function version is B
  - Motion CPU: The first digits of the serial No. must be as follows.  
Q172CPU ..... "H" or later  
Q173CPU ..... "G" or later
- 3) The function version of the High Performance model QCPU and Motion CPU can be checked at the following places.
  - Ratings nameplate of High Performance model QCPU and Motion CPU
  - Product data list of system monitor of GX Developer (See Section 2.3 for details.)

##### (b) Operations with different function versions

Function version A High Performance model QCPUs cannot be used on a multiple CPU system.

Errors shown in table 14.1 will occur and the multiple CPU system will not start up if function version A High Performance model QCPUs are used on a multiple CPU system.

If any of the errors shown in table 14.1 are displayed after executing GX Developer Version 6 or later PLC diagnosis function, replace the function version A High Performance model QCPU with a function version B High Performance model QCPU.

Table 14.1 List of operations with differing function versions

PLC No.1	PLC Nos. 2 to 4	Status of PLC No.1	Status of PLC Nos. 2 to 4
Function version A	Function version A	UNIT VERIFY ERROR (Error code: 2000)	SP. UNIT LAY ERROR (Error code: 2125)
Function version A	Function version B	UNIT VERIFY ERROR (Error code: 2000)	MULTI EXE. ERROR * 1 (Error code: 7010)
Function version B	Function version A	MULTI EXE. ERROR * 1 (Error code: 7010)	SP. UNIT LAY ERROR (Error code: 2125)
Function version B	Function version B	No error	No error

\*1: The following errors may occur except "MULTI EXE. ERROR" when the PLC is turned on or the High Performance model QCPU for CPU No.1 is reset.

- CONTROL-BUS ERR. (error code: 1413, 1414)
- MULTIPLE PLC DOWN (error code: 7000, 7002)

(2) High Performance model QCPU, Motion CPU and PC CPU module mounting positions

- (a) Up to four modules of High Performance model QCPU can be mounted in the CPU slots (starting from the slot on the right side of power supply module closely) and the neighboring slots up to slot 2. There must be no empty slot between CPU modules.

Mount the Motion CPU or PC CPU module in the following way.

- Mount the Motion CPU on the right side of the High Performance model QCPU.
- Mount only one PC CPU module at the right end of CPU modules. (No CPU module can be mounted on the right side of the PC CPU module.)

Table 14.2 Installation positions of CPU modules

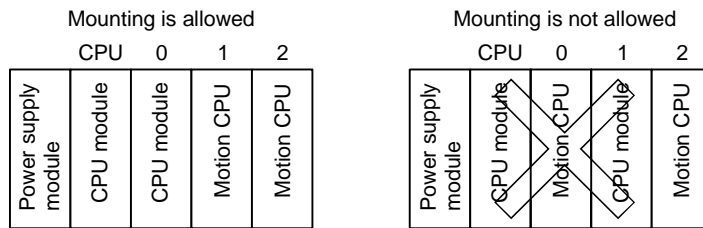
Number of CPUs	Mounting positions of CPU modules				
1	CPU 0 1 2				
	Power supply module	CPU module			
2	CPU 0 1 2				
	Power supply module	CPU module	CPU module		
3	CPU 0 1 2				
	Power supply module	CPU module	CPU module	CPU module	
4	CPU 0 1 2				
	Power supply module	CPU module	CPU module	CPU module	CPU module

\* 1: The PC CPU module occupies two slots.

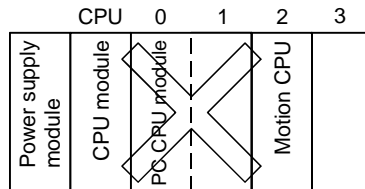
Number of CPUs	Mounting positions of CPU modules																
	CPU 0 1 2					CPU 0 1 2 3					CPU 0 1 2 3						
	Power supply module	CPU module	Motion CPU	Motion CPU	Motion CPU	Power supply module	CPU module	CPU module	CPU module	PC CPU module *1		Power supply module	CPU module	CPU module	Motion CPU	PC CPU module *1	
	Power supply module	CPU module	Motion CPU	Motion CPU	PC CPU module *1												

\*1: The PC CPU module occupies two slots.

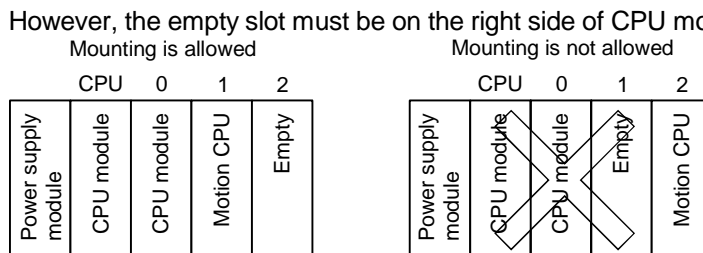
- (b) Motion CPUs are mounted together on the slot to the right of the High Performance model QCPU. High Performance model QCPU's cannot be mounted to the right of Motion CPUs.



- (c) Mount the PC CPU module at the right end in the multiple PLC system. No CPU module can be mounted on the right side of the PC CPU module.



- (d) An empty slot is secured for future addition of a CPU module. The number of CPU module including empty slots are set with the No. of CPU setting, and the type is set in the "CPU (empty)" setting from the slot immediately to the right of the number of CPU modules set at the "I/O Assignment" tab screen in the (PLC) "Parameter" dialog box. For example, when four CPU modules have been set with the multiple CPU setting and two High Performance model QCPU's and one Motion CPU have been mounted, the High Performance model QCPU's are mounted in the CPU module slot and slot 0, the Motion CPU is mounted in slot 1, and slot 2 is left empty.

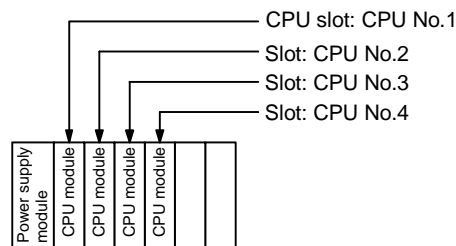


**POINT**

To add a High Performance model QCPU or Motion CPU to a system where the PC CPU module is used, shift the PC CPU module to the right because no CPU module is allowed on the right side of the PC CPU module.

(2) High Performance model QCPU, Motion CPU and PC CPU module CPU numbers

- (a) CPU numbers are allocated for identifying the High Performance model QCPUs, Motion CPUs and PC CPU module mounted on the main base unit in the multiple CPU system. The CPU No.1 is allocated to the CPU slot, and the CPU No.2, No.3 and No.4 are allocated to the right of the CPU No.1.



These PLC numbers are used for the following purposes of

- To access the High Performance model QCPU when the GX Developer (personal computer) is not connected in the multiple CPU system.
- To set up control CPUs with the I/O Assignment in the multiple CPU system.

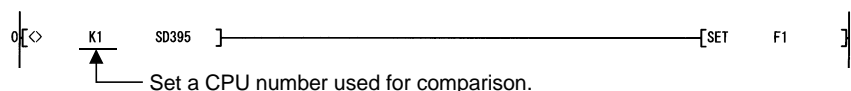
- (b) The High Performance model QCPU stores the host number in the special register (SD395).

It is recommended to build a program for checking the host number using the High Performance model QCPU.

This will enable easy verification when High Performance model QCPUs are not mounted correctly and when programs are written into other CPUs with the GX Developer.

In the program shown below, the annunciator (F1) is set to ON when the High Performance model QCPU writing programs is a CPU other than the CPU No.1 (SD395 = 1.)

The "USER" LED on the front of the High Performance model QCPU is illuminated when the annunciator (F1) is set to ON. The number of the annunciator that has been set at ON will also be stored in a special register (SD62).



**REMARK**

For the own number confirmation method for the Motion CPU and PC CPU module, refer to the manual of the Motion CPU and PC CPU module.

14.2.2 Precautions when using Q Series I/O modules and intelligent function modules

(1) Compatible I/O modules

All I/O modules (QX□, QY□) are compatible with to multiple CPU system. They can be used by setting any of CPU No.1 to No.4 as a control CPU.

(2) Compatible intelligent function modules

- (a) The intelligent function modules compatible with the multiple CPU system are those of function version B or later. They can be used by setting any of CPU No.1 to No.4 as a control CPU.
- (b) Q Series high speed count modules (QD62, QD62D, QD62E) are compatible with by multiple CPU system are those of function version A or later. They can be used by setting any of CPU No.1 to No.4 as a control CPU.
- (c) Q Series interruption modules (QI60) do not have a function version, but are supported by multiple CPU system. CPUs No.1 to No.4 can be set up as control CPUs.
- (d) Intelligent function modules of function version A can be used in multiple CPU system by setting CPU No.1 as a control CPU. However, only control CPU can be accessed from serial communication modules and other external modules. (MELSECNET/H, serial communication modules and other external modules cannot access non-control CPUs.)  
The "SP. UNIT VER. ERR (error code: 2150)" occurs if any of CPU No.2 to No.4 has been set as a control CPU, and the multiple CPU system will not be a started up.

(3) Ranges of access to control and non-control modules

In a multiple CPU system, non-control modules can be accessed by setting "I/O setting outside of the group" at the "Multiple CPU settings" dialog box in "PLC Parameter". The following table indicates accessibility to the control and non-control modules in the multiple CPU system.

Access target		Accessibility		
		Control module	Non-control module (I/O setting outside of the group)	
			Disabled	Enabled
Input (X)		○	×	○
Output (Y)	Read	○	×	○
	Write	○	×	×
Buffer memory	Read	○	○	○
	Write	○	×	×

○: Accessible      ×: Inaccessible

**REMARK**

- The function version of intelligent function modules can be confirmed at the rated name plate of the intelligent function module and with the GX Developer's "System monitor product information list window" (see Section 2.3.)
- See Section 14.2.4 for details on restrictions on the number that can be used with intelligent function modules.



14.2.3 Limitations when mounting AnS Series corresponding I/O modules and special function modules

(1) Compatible I/O modules and special function modules

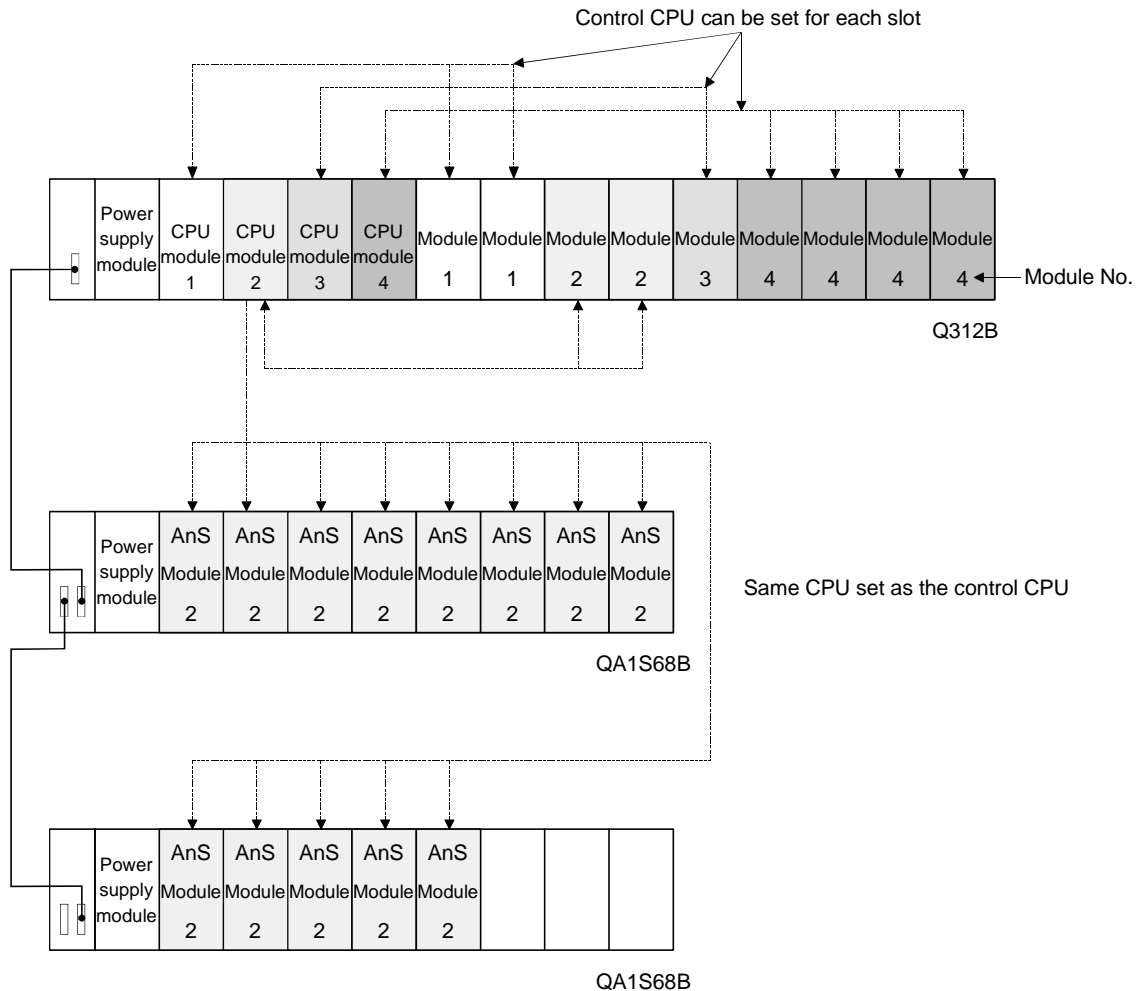
AnS Series corresponding I/O modules and special function modules (compact types) can be used with the High Performance model QCPU.

(2) Control CPU

The AnS Series corresponding I/O modules or special function module can be controlled by only one CPU (control CPU) of the CPUs No.1 to No.4 when configuring a multiple CPU system. For example.

If the control CPU is set up as the CPU No.2 as shown in the illustration below, every slot's control CPU on which I/O modules and special function modules compatible with the AnS Series are mounted are set as the CPU No.2.

The "PARAMETER ERROR (error code: 3009)" occurs even if only one AnS Series corresponding I/O module or special function module is set up, the multiple CPU system will not be started up.



The Module No. shown in the illustration represents the following:  
 CPU 1 to 4 : CPU's CPU number  
 Modules 1 to 4 : Control CPU's CPU number

## (3) Ranges of access to control and non-control modules

The following table indicates accessibility to the control and non-control modules in the multiple CPU system.

Access target		Accessibility		
		Control module	Non-control module (I/O setting outside of the group)	
			Disabled	Enabled
Input (X)		○	×	×
Output (Y)	Read	○	×	×
	Write	○	×	×
Buffer memory	Read	○	×	×
	Write	○	×	×

○: Accessible      ×: Inaccessible

## 14.2.4 Modules that have mounting restrictions

The following table indicates restrictions on the number of modules that can be mounted in multiple CPU system. Ensure that the number of modules mounted is within these ranges.

Product	Model	Number of modules that can be mounted per system	Number of modules that can be mounted per PLC
Q series MELSECNET/H network modules	<ul style="list-style-type: none"> <li>• QJ71LP21</li> <li>• QJ71BR11</li> <li>• QJ71LP21-25</li> <li>• QJ71LP21G</li> <li>• QJ71LP21GE</li> </ul>	Maximum of four PLC to PLC networks and remote I/O networks	Maximum of four PLC to PLC networks and remote I/O networks
Q series Ethernet interface modules	<ul style="list-style-type: none"> <li>• QJ71E71</li> <li>• QJ71E71-B2</li> <li>• QJ71E71-100</li> </ul>	Maximum of four	Maximum of four
Q series CC-Link system master/local modules	<ul style="list-style-type: none"> <li>• QJ61BT11</li> <li>• QJ61BT11N</li> </ul>	No limit *	No limit *
MELSECNET/MINI-S3 data link modules	<ul style="list-style-type: none"> <li>• A1SJ71PT32-S3</li> <li>• A1SJ71T32-S3</li> </ul>	No restrictions (however, the automatic refresh function cannot be set up)	No restrictions (however, the automatic refresh function cannot be set up)
AnS series corresponding Special function modules	<ul style="list-style-type: none"> <li>• A1SD51S</li> <li>• A1SD21-S1</li> <li>• A1SJ71J92-S3</li> </ul> (When the GET and PUT services are used)	Maximum of six	Maximum of six
Interruption modules	• A1SI61	Only one	Only one
	• QI60	Maximum of four (maximum of three when the A1SI61 is in use)	

\*: A maximum of 4 modules per PLC (16 modules per system) can be controlled if the network parameters for CC-Link are set and controlled by GX Developer. There is no restriction in the number of modules when the parameters are set by the instructions dedicated to the CC-Link.

### 14.2.5 Compatible GX Developers and GX Configurators

(1) Compatible GX Developers

GX Developer Version 6 (SW6D5C-GPPW-E) or later are compatible with on multiple CPU system.

GX Developer Version 5 (SW5D5C-GPPW-E) or earlier are not compatible with multiple CPU system.

(2) Compatible GX Configurators

The GX Configurators listed can be used on multiple CPU system without modification.

14.2.6 Parameters that enable the use of multiple CPU system

- (1) Parameters that enable the use of multiple CPU system  
 Compared with the single CPU system, the multiple CPU system includes the "No. of CPU", "control CPU", "refresh setting (automatic refresh setting)" of PLC parameters.  
 The PLC parameters must be set to the same for all the CPU modules used in the multiple CPU system, except some modules.  
 Make similar settings to the PC CPU module, if one is included, using the PC CPU module setting utility.  
 For the setting method, refer to the manual of the PC CPU module.
- (2) The PLC parameter settings for use with multiple CPU system  
 The PLC parameters, necessity of setup, and descriptions that are required for using multiple CPU system are listed in table 14.3.

Table 14.3 Setting list for the multiple CPU and I/O Assignment

PC parameter		Necessity of setup *1	Description *2
I/O Assignment	I/O Assignment		
	Type	—	○
	Model name	—	—
	Points	—	○
	Start	—	○
	Standard setting		
	Base model name	—	—
	Power model name	—	—
	Extension cable	—	—
	Points	—	○
	Switch settings	—	—
	Detailed settings		
	Error time output made	—	—
	H/W error time PLC operation made	—	—
I/O response time	—	—	
Control PLC	○	○	
PLC system settings	Number of empty slots	—	○
Multiple CPU settings	No. of CPU	○	○
	Operation mode	△	○
	Online module change	○	○
	The input condition outside of the group is taken	△	△
	The output condition outside of the group is taken	△	△
	Refresh setting		
	Send range for each CPU	△	○
CPU side devices	△	—	

\*1: Necessity of setup column

○: Items that must be set up for multiple CPU system (operations not possible if not set up.)

△: Items that may be set up when required for multiple CPU system (Operations carried out with the default values when not set up.)

—: Items that are the same as single CPU system.

\*2: Descriptions

○: Items that have the same settings for all CPU modules on the multiple CPU system.

△: Items that have the same settings for all High Performance model QCPUs and PC CPU module on the multiple CPU system (items that do not have settings for Motion CPUs).

—: Items that can be setup up individually for each CPU modules on the multiple CPU system.

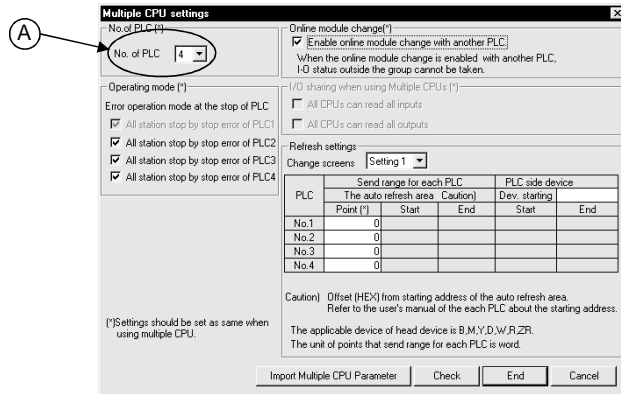
After parameters such as multiple CPU settings are changed, reflect the changes to keep uniformity among all CPUs in the multiple CPU system, then reset the CPU No.1.

It is possible to transfer across and use the CPU settings and I/O Assignments set up for other projects with GX Developer.

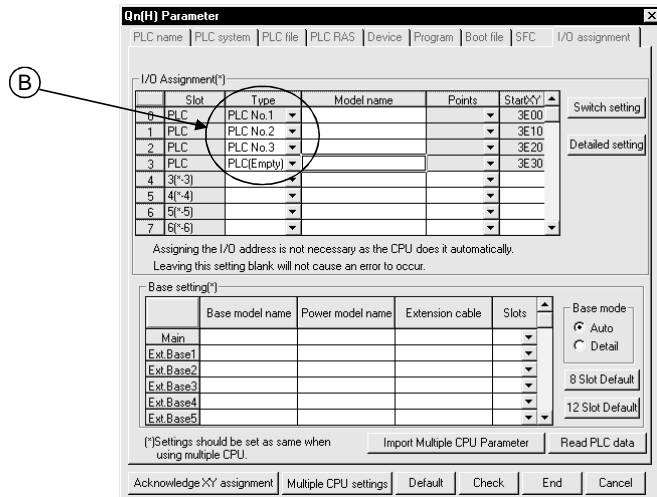
(See section 19.2.3 for details on transferring and using multiple CPU settings and I/O Assignments.)

(a) Number of CPUs setting (setup necessary)

- 1) The number of CPU modules to be used on a multiple CPU system are set at the PLC parameter's "Multiple CPU settings" screen in the (PLC) Parameter dialog (indicated with the "A" arrow.)

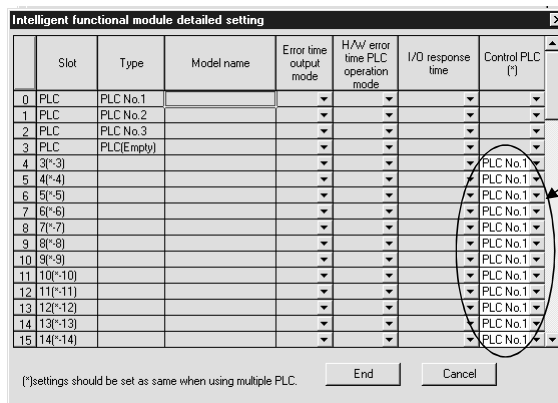


- 2) Ensure that the No. of CPU set for the multiple CPU system is the same as the number of CPUs actually mounted. When an empty slot is secured for the purpose of mounting additional CPU modules in the future, set "PLC (Empty)" at the "I/O assignment" tab screen in the (PLC) "Parameter" dialog box. For example, when setting "4" as "No. of CPU" in the "Multiple CPU settings" screen and securing one of them for future use, set slot 3 to "CPU (Empty)" (indicated with the "B" arrow.)



- 3) A "PARAMETER ERROR (error code: 3010)" occurs to all mounted CPU modules in the following cases.
  - The number of mounted CPU modules exceeds the number set with the No. of CPU setting.
  - No CPU module is installed in slots set for CPU No.1 to No.4.

- (b) Operation mode setting (optional)  
 This is set to continue operation of other CPUs where a stopping error has not occurred when an error occurs at any of CPUs No.2 to No.4.  
 The operation mode for the CPU No.1 cannot be changed (all CPUs will suspend operations when a stop error is triggered for the CPU No.1.)  
 See Section 14.2.8 for further details.
- (c) I/O settings outside of the group (optional)  
 This is set when the input and output (X, Y) for I/O modules and intelligent function modules being controlled by other CPUs is to be downloaded to the host CPU.  
 See Section 17.2 for further details.
- (d) Refresh setting (optional)  
 This is set up to automatically refresh the device data on the multiple CPU system.  
 See Section 16.1 for further details.
- (e) Control CPU settings (optional)  
 Sets up the control CPUs for the I/O modules and intelligent function modules mounted on the base units on the multiple CPU system (indicated by the "C" arrow.)  
 All default settings are set for the CPU No.1.



(3) Multiple CPU setting and I/O Assignment checks

Checks, as shown in table 14.4, will be run to ascertain that all CPU modules have the same settings (sameness check) when the description column in table 14.3 has been set with the "O" symbol and the power to the sequence is switched on, the High Performance model QCPU is reset, or the status is changed from STOP to RUN.

- (a) The multiple CPU system will be started up if all CPUs have the same settings.
- (b) The operations described in table 14.4 will be performed when all CPUs do not have the same settings.  
 In this event, check the multiple CPU settings and I/O Assignment, and set all CPUs with the same settings.  
 To start the multiple CPU system, reset the High Performance model QCPU for CPU No.1 or turn off and on the CPU (power ON → OFF → ON).  
 (For the action after the High Performance model QCPU for CPU No.1 is reset, see Section 14.2.7.)

Table 14.4. List of sameness check contents

Item	CPU No.1	CPU No.1 to 4
When the power to the CPU is switched on When CPU No.1 is reset	No sameness check will be run	<ul style="list-style-type: none"> <li>• A comparison check will be run on the multiple CPU settings and I/O Assignments for CPU No.1.</li> <li>• A "PARAMETER ERROR (error code: 3012)" will occur in the host CPU if they do not match.</li> </ul>
<ul style="list-style-type: none"> <li>• When the RUN/STOP switch has been changed from STOP to RUN.</li> <li>• When parameters are written with the GX Developer</li> </ul>	When CPUs in the RUN mode exist	<ul style="list-style-type: none"> <li>• A comparison check will be run on the multiple CPU settings and I/O Assignments for the CPU in the RUN mode with the lowest number.</li> <li>• A "PARAMETER ERROR (error code: 3012)" will occur in the host CPU if they do not match.</li> </ul>
	When CPUs in the RUN mode do not exist	<ul style="list-style-type: none"> <li>• A comparison check will be run on the multiple CPU settings and I/O Assignments for CPU No.2.</li> <li>• A "PARAMETER ERROR (error code: 3012)" will occur in the host CPU if they do not match.</li> </ul>
	When a stop error occurs at CPU No.1	—

**POINT**

After multiple CPU system parameters unavailable with the Motion CPU are changed for the High Performance model QCPU or PC CPU module in a multiple CPU system including a Motion CPU, be sure to reset the High Performance model QCPU for CPU No.1 or turn off and on the CPU. (Otherwise the High Performance model QCPU or PC CPU module checks consistency with multiple CPU system parameters of the Motion CPU, causing a "PARAMETER ERROR (error code: 3012)."

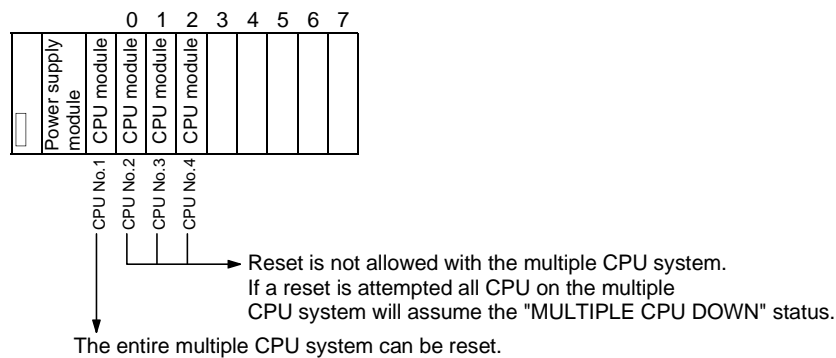


14.2.7 Resetting the multiple CPU system

It is possible to reset the entire multiple CPU system by resetting the CPU No.1. The CPU modules for CPU No.2 to No.4, I/O modules and intelligent function modules will be reset when the CPU No.1 is reset.

If a stop error occurs for any of the CPUs on the multiple CPU system, either reset the CPU No.1 or restart the sequencer (power supply ON → OFF → ON) after the problem has been recovered.

(Recovery is not allowed by resetting the CPU modules for CPU No.2 to No.4 for which stop errors have occurred.)



**POINT**

- (1) It is not possible to reset the CPU modules for CPU No.2 to No.4 individually in the multiple CPU system. If an attempt to reset any of the CPU modules for CPU No.2 to No.4 during operation of the multiple CPU system, a "MULTIPLE CPU DOWN (error code: 7000)" error will occur for the other CPUs, and the entire multiple CPU system will be halted. However, depending on the timing in which the CPU modules have been reset, there are cases where errors other than the "MULTIPLE CPU DOWN" error will halt the other CPUs.
- (2) A "MULTIPLE CPU DOWN (error code: 7000)" error will occur regardless of the operation mode set at the "Multiple CPU settings" screen within the (PLC) Parameter dialog box. (stop/continue all other CPUs on the CPU modules for CPU No.2 to No.4 error) when the CPU modules for CPU No.2 to No.4 are reset (See Section 14.2.8 for details on the multiple CPU setting operation modes.)

14.2.8 Processing when High Performance model QCPU stop errors occur

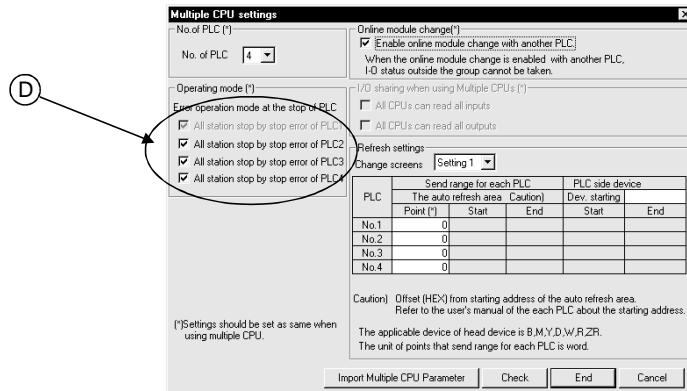
The operations for the entire system will differ when a CPU No.1 stop error occurs and when any of CPU No.2 to No.4 stop error occurs in the multiple CPU system.

(1) When a stop error occurs at the CPU No.1

- (a) A "MULTIPLE CPU DOWN (error code: 7000)" error occurs for the CPU modules for CPU No.2 to No.4 and motion CPUS and the multiple CPU system will be halted when a stop error occurs at the CPU No.1 (See point on the next page for details)
- (b) Observe the following procedures to restore the system.
  - 1) Confirm the cause of the CPU No.1 error with the PC diagnosis function.
  - 2) Remove the cause of the error.
  - 3) Either reset the CPU No.1 or restart the power to the CPU.  
All CPUs on the entire multiple CPU system will be reset and the system restored when the CPU No.1 is reset or the power to the CPU is restarted.

(2) When a stop error occurs at the CPU No. other than No.1

Whether the entire system is halted or not is determined by the multiple CPU setting's "Operating Mode" setting when a stop error occurs in the CPU modules for CPU No.2 to No.4. The default setting is for all CPUs to be stopped with a stop error. When you do not want to stop all CPUs at occurrence of a stop error in any of the CPU modules, click the check box that corresponds to the CPU No. whose error will not stop all CPUs. (Arrow D)



- (a) A "MULTIPLE CPU DOWN (error code: 7000)" error occurs for the CPU modules and the multiple CPU system will be halted when a stop error is occurs in CPU modules for which the "All station stop by stop error of CPU 'n' " has been set. (See POINT on the next page for details.)
- (b) A "MULTIPLE CPU ERROR (error code: 7010)" error occurs for all other CPUs but operations will continue when a stop error occurs in CPU modules for which the " All station stop by stop error of CPU 'n' " has not been set.

**POINT**

A "MULTIPLE CPU DOWN" stop error will occur for the CPU on which the error was detected when a stop error occurs.

There are cases where the timing of error detection will search for the CPU on which the stop error that has caused the "MULTIPLE CPU DOWN" error occurs, not the first CPU on which a stop error occurs, and the entire system will assume the "MULTIPLE CPU DOWN" status.

For example, if a stop error occurs in the CPU No.2 and the CPU No.3 is halted as a direct consequence of this, there are cases where the CPU No.1 will be halted because of the stop error on CPU No.3 depending on the timing of error detection.

Halted with an "OPERATION ERROR"

Halted with stop error detection on CPU No.2 (becomes "MULTIPLE CPU DOWN")

There are cases where the system will be halted with stop error detected on the CPU No.3 depending on the timing of error detection. (becomes "MULTIPLE CPU DOWN")

Owing to this, there are cases where a different CPU No. to the CPU that initially caused the stop error will be stored in the error data's common information category.

In this event, remove the reason for the error on the CPU that caused the stop error in addition to the "MULTIPLE CPU DOWN" error when restoring the system.

In the illustration shown below, the cause of the CPU No.2 error that did not cause the "MULTIPLE CPU DOWN" error is removed.

PLC Module	No.	Present Error	Year/M
PLC1	7000	MULTI CPU DOWN	20/
PLC2	4100	OPERATION ERROR	20/
PLC3	7000	MULTI CPU DOWN	20/
PLC4			

No.	Error message	Year/Month/Day
4100	OPERATION ERROR	2000/11/6

- (c) Observe the following procedures to restore the system.
- 1) Confirm the cause of the CPU No.1 error with the PLC diagnostics function.
  - 2) Remove the cause of the error.
  - 3) Either reset the CPU No.1 or restart the power to the PLC.
- All CPUs on the entire multiple CPU system will be reset and the system restored when the CPU No.1 is reset or the power to the CPU is restarted.

### 14.2.9 Reducing the time required for multiple CPU system processing

#### (1) Multiple CPU system processing

A bus (base unit pattern, extension cable) is used by the CPU module when accessing the I/O module and intelligent function module, and this bus cannot be used by plural CPU module at the same time.

The CPU modules that attempted bus access afterwards when plural CPU module use the bus simultaneously will assume the "Standby" status until processing for the CPU module that executed the procedure first has been completed.

This "Standby" status (the amount of time the CPU module must wait) will cause delays in input and output on the multiple CPU system, and result in extended scan times.

See Chapter 18 for details on extended scan times.

#### (2) Maximum standby time

The host CPU will reach the maximum standby time in the following cases with a multiple CPU system.

- When four CPU modules are used on the multiple CPU system.
- When additional base units are in use.
- When intelligent function modules that possess vast quantities of data are mounted onto additional base units.
- When four CPU modules simultaneously access modules mounted onto additional base units.

#### (3) Reducing the time required for multiple CPU system processing

The following methods are available for reducing the amount of time required for multiple CPU system processing.

- Combine modules with many access points, such as MELSECNET/H and CC-LINK refresh, etc., together into a main base unit.
- Set modules with many access points, such as MELSECNET/H and CC-LINK refresh, etc., as control module on a single CPU module, and ensure that simultaneous access does not occur.
- Reduce the number of MELSECNET/H and CC-LINK refresh access points.
- Reduce the number of automatic refresh points between CPU modules.

POINT
<p>It is possible to reduce scan time by changing the following PLC parameter settings:</p> <ul style="list-style-type: none"> <li>• A Series CPU compatibility setting</li> <li>• Floating point arithmetic processing</li> </ul> <p>See Section 18.3 for details.</p>

## 15 ALLOCATING MULTIPLE CPU SYSTEM I/O NUMBERS

### 15.1 Concept behind Allocating I/O Numbers

Multiple CPU system possess I/O numbers to enable interactive transmission between the CPU modules and the I/O modules and intelligent function modules, and I/O numbers to enable interactive transmission between the CPU modules.

#### 15.1.1 I/O modules and intelligent function module I/O numbers

The difference with single CPU systems is the 00H position (slot) of the I/O number with multiple CPU system.

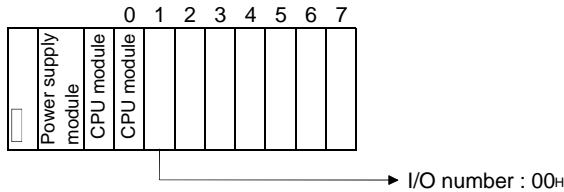
However, the concept behind the sequence for allocating I/O numbers, the I/O numbers for each slot, and the I/O numbers for empty slots is the same for both types of system.

See Chapter 5 (Allocating I/O Numbers) for details on the concept behind the sequence for allocating I/O numbers, the I/O numbers for each slot, and the I/O numbers for empty slots.

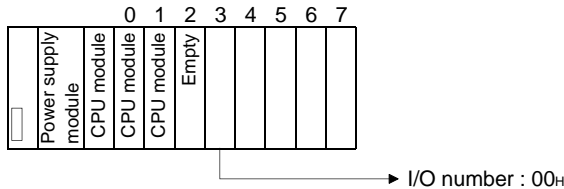
#### (1) "00H" position for I/O numbers

- (a) The number of slots set with the PLC parameters' multiple CPU settings are occupied by the CPU modules on the multiple CPU system.
- (b) The I/O modules and intelligent function modules are mounted from the right of the slots occupied by the CPU modules.
- (c) I/O number of a system without PC CPU module  
The I/O number for the I/O modules and intelligent function modules mounted from the right of the slots occupied by the CPU modules is set as "00H" and consecutive numbers are then allocated sequentially to the right.

##### 1) Example: Two modules are mounted



##### 2) Example: Three modules are mounted and one empty slot exists



- (d) Input/output number of a system with PC CPU module  
The PC CPU module occupies two slots. The one on the right side among the two slots is handled as an empty slot. (16 empty points are occupied with default setting.) Therefore the I/O number of the slot on the right side of the PC CPU module is "10H." (Set the empty slot at zero point using I/O allocation of PLC Parameters dialog box, to assign "00H" to the first I/O number.)



#### REMARK

- If the number of CPU modules mounted on the main base unit is smaller than the number set at the "Multiple CPU setting" of "(PLC) Parameter" dialog box, the slot(s) on the right of the actually mounted CPU modules is (are) set as "CPU (Empty)".
- The I/O number for the multiple CPU system can be confirmed with the system monitor.

15.1.2 I/O number of High Performance model QCPU, Motion CPU and PC CPU module

I/O numbers are allocated to the CPU modules with the multiple CPU system in order to allow interactive communications between the CPU modules with the following commands.

- Multiple CPU commands
- Motion dedicated commands
- Dedicated communication commands between multiple CPUs

The I/O numbers for the CPU modules are fixed for the slots on which they are mounted and cannot be amended.

The table below shows the I/O number allocated to each CPU module when the multiple CPU system is composed.

CPU module mounting position	CPU slot	Slot 0	Slot 1	Slot 2
First I/O number	3E00H	3E10H	3E20H	3E30H

The CPU modules I/O numbers are used in the following cases.

- When writing data in the host CPU's CPU shared memory with the S.TO instruction. \*1
- When reading data from other CPU's CPU shared memory with the FROM instruction. \*1
- When reading data from other CPU's CPU shared memory with the intelligent function module device (U\\_G\\_). \*1
- When specifying the High Performance model QCPU to be accessed with the Ethernet module. \*2
- When specifying the High Performance model QCPU to be accessed with the serial communication module. \*3

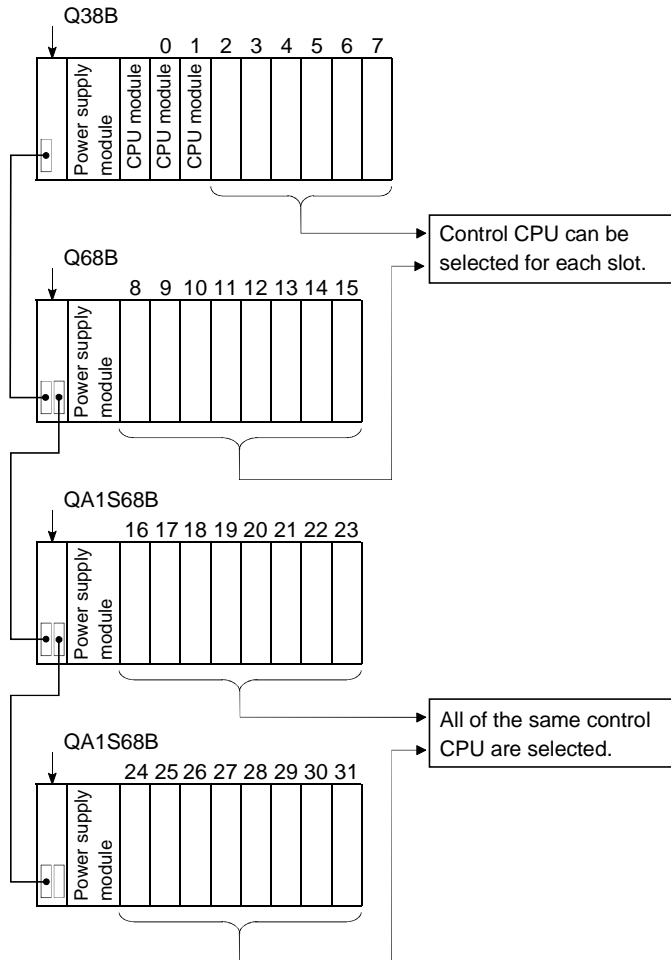
**REMARK**

- \*1: See Chapter 16 for details on among High Performance model QCPU, Motion CPU and PC CPU module.
- \*2: Refer to the Ethernet module's manual for details on accessing the High Performance model QCPU with the Ethernet module.
- \*3: Refer to the serial communication module's manual for details on accessing the High Performance model QCPU with the serial communication module.

15.2 Purpose of PC Parameter I/O Allocations with GX Developer

Sets up the High Performance model QCPU/Motion CPU/PC CPU module that are to control the multiple CPU system's I/O modules and intelligent function modules.

- (a) Q Series I/O modules and intelligent function modules can be selected as control CPUs for each slot.
- (b) AnS Series I/O modules and intelligent function modules are set as control CPUs on the same CPU modules.



16 COMMUNICATION BETWEEN CPU MODULES IN MULTIPLE CPU SYSTEM

It is possible to perform the following interactive transmission between each CPU modules with a multiple CPU system.

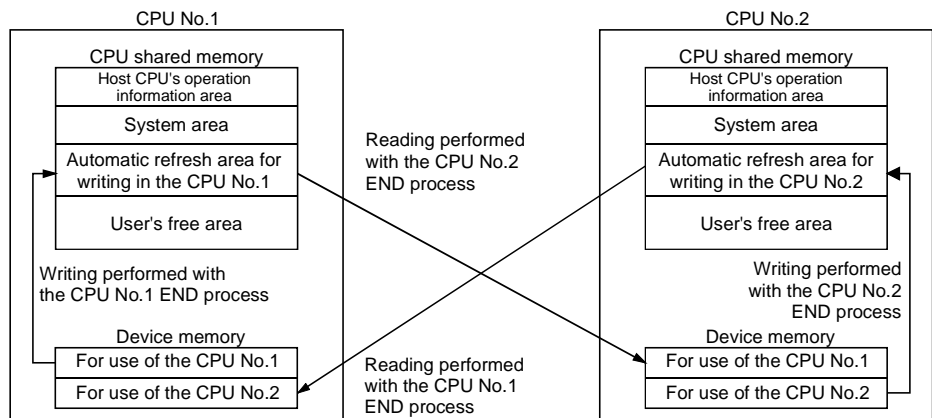
- Automatically refreshing the device data between each CPU modules with multiple CPU system parameter settings.
- Data transfer between other High Performance model QCPU and PC CPU module via CPU shared memory using multiple CPU instructions  
Also, data reading of High Performance model QCPU from CPU shared memory of Motion CPU using multiple CPU instructions
- Control command from the High Performance model QCPU to the Motion CPU with Motion dedicated CPU instructions.
- Writing and reading of the device data from the High Performance model QCPU to the Motion CPU with communication dedicated instructions between multiple CPUs.

Also, event issuance from High Performance model QCPU to PC CPU module using instructions dedicated to multiple CPU communication

(1) Automatic refresh of device data

Automatic refresh of the CPU shared memory is a function of automatic data transfer between CPU modules in END processing of the CPU.

As the device memory data of other CPUs is automatically read when the automatic refresh function is used, is possible for the host CPU to use the device data of other CPUs.

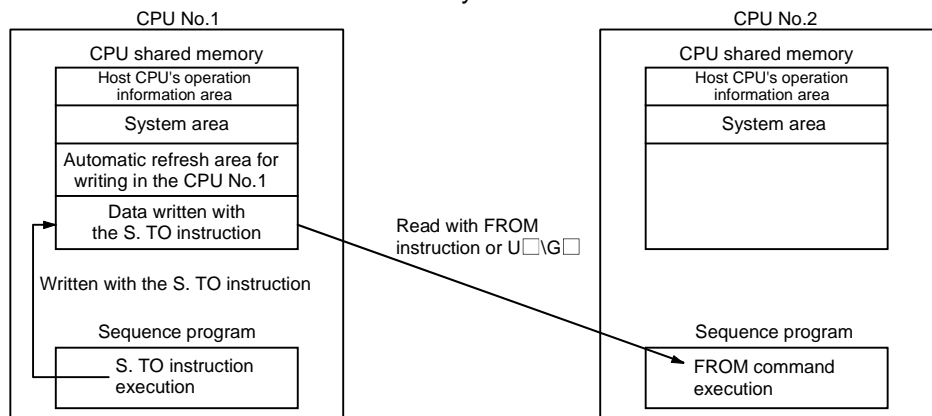


(2) Exchanging data with multiple CPU instructions and instructions that use Intelligent function module device (U□\G□)

The CPUs on the multiple CPU system write data into the host CPU's CPU shared memory with the use of the S. TO instruction/FROM instruction.

The data written to the CPU shared memory of the host CPU with the S. TO instruction is read by High Performance model QCPU of other CPUs with the use of the FROM instruction and U□\G□.

Non-linked device data also read directly when the command is executed.





16.1 Automatic Refresh of CPU Shared Memory

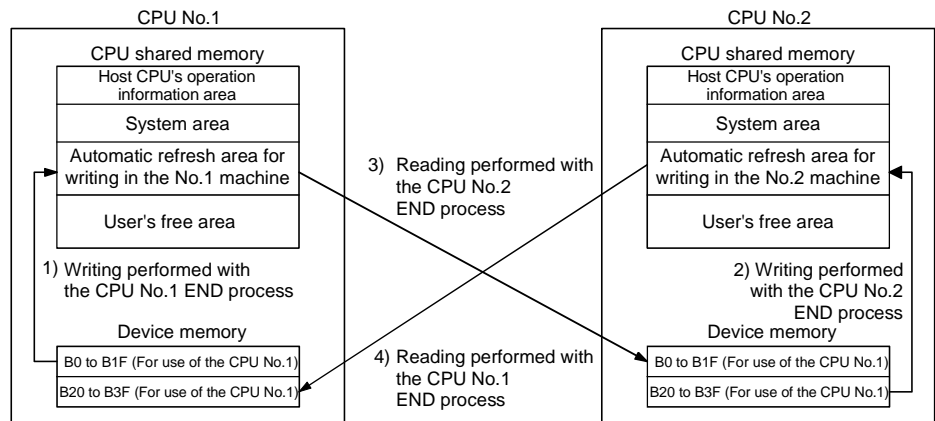
(1) Automatic refresh of CPU shared memory

(a) Automatic refresh of the CPU shared memory is a function of automatic data transfer between CPUs in END processing of the CPU. As the device memory data of other CPUs is automatically read when the automatic refresh function is used, is possible for the host CPU to use the device data of other CPUs.

Data is transmitted between the following parties during automatic refresh of CPU shared memory.

- Between High Performance model QCPU and High Performance model QCPU
- Between High Performance model QCPU and Motion CPU
- Between Motion CPU and Motion CPU
- Between High Performance model QCPU and PC CPU module
- Between Motion CPU and PC CPU module

An outline of operations when the CPU No.1 performs automatic refresh on the 32 points between B0 and B1F, and when the CPU No.2 performs automatic refresh on the 32 points between B20 and B3F.



The processes performed during the CPU No.1 END process.

- 1): The B0 to B1F transmission device data for the CPU No.1 is transferred across to the host CPU shared memory automatic refresh area.
- 4): The data in the CPU No.2 CPU shared memory automatic refresh area is transferred across to B20 to B3F in the host CPU.

The processes performed during the CPU No.2 END process.

- 2): The B20 to B3F transmission device data for the CPU No.2 is transferred across to the host CPU shared memory automatic refresh area.
- 3): The data in the CPU No.1 CPU shared memory automatic refresh area is transferred across to B0 to B1F in the host CPU.

(b) Executing automatic refresh

Automatic refresh is executed when the CPU module is in RUN status, STOP status or PAUSE status. Automatic refresh cannot be performed when a stop error has been triggered in the CPU module.

If a stop error occurs on one module, the other modules for which an error has not occurred will save the data prior to the stop error being triggered. For example, if a stop error occurs in the CPU No.2 when B20 is ON, the B20 in the CPU No.1 will remain at ON, as shown in the operation outline in fig. (a).

(c) When automatic refresh is carried out, it is necessary to set the points to be transmitted by each CPU and the device in which the data is to be stored (the device that will perform automatic refresh) with the PLC parameter multiple CPU settings.

(2) Automatic refresh settings

Set the points to be transmitted by each CPU and the device in which the data is to be stored with the PLC parameter multiple CPU settings for when automatic refresh is to be performed.

Range of transmission setting for each CPU module

Change screens: Setting 1

PLC	Send range for each PLC			PLC side device	
	Point (*)	Start	End	Start	End
No.1	2	0800	0801	B0	B1F
No.2	2	0800	0801	B20	B3F
No.3	0				
No.4	0				

Switching between setting numbers

Sets the header number of the device for which automatic refresh is to be performed (uses consecutive numbers from the setup device number to the number of specified points)

(a) Setting switch/range of transmission for each CPU (refresh range)

- 1): It is possible to set four ranges from Setting 1 to Setting 4 for the refresh setting with the setting switch. For example, it is possible to set the refresh function to divide ON/OFF data into bit devices with Setting 1 and other data into word devices with Setting 2.
- 2): The transmission range for each CPU is set in units of two CPU shared memory points (two words.) (Becomes 2 points when specifying the word device with the CPU device, and 32 points when specifying the bit device.)  
Data for which the point is set at "0" with the range of transmission for each CPU will not be refreshed.  
As the bit device becomes 16 points at one point of the CPU shared memory when refreshing is performed with 32 points between B0 and B1F on the CPU No.1 and with 32 points between B20 and B3F on the CPU No.2, the number of transmission points is two for the CPU No.1 and two for the CPU No.2.
- 3): The number of transmission points is a maximum of 2 k points (2 k words) with a total of four ranges for each CPU module, making a total of 8 k points (8 k words) for all CPUs.

- 2 k points (2 k words) per CPU.
- 8 k points (8 k words) for all CPUs.
- Setting is in units of 2 points (2 words).

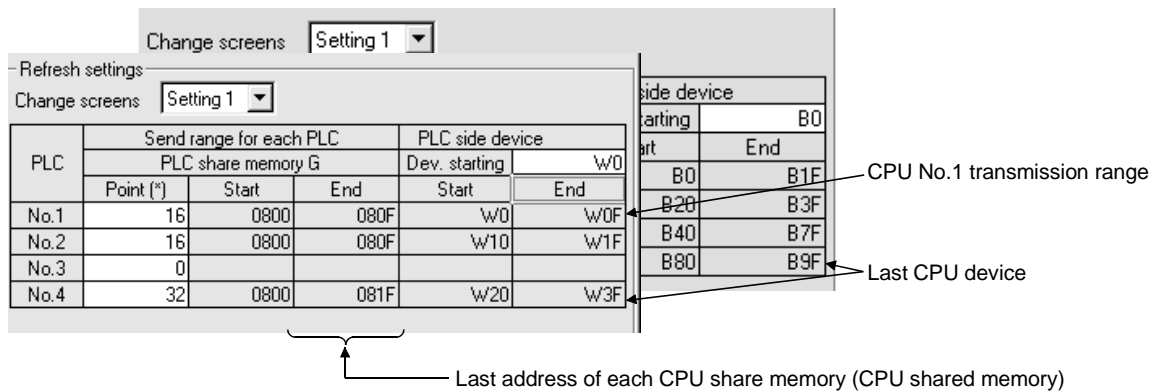
Change screens: Setting 1

PLC	Send range for each PLC			PLC side device	
	Point (*)	Start	End	Start	End
No.1	2	0800	0801	B0	B1F
No.2	2	0800	0801	B20	B3F
No.3	4	0800	0803	B40	B7F
No.4	2	0800	0801	B80	B9F

The CPU shared memory (CPU share memory) is set in two points, and the bit device becomes 32 points when bit device is specified on the CPU device.

Not refreshed as the number of points for CPU No.3 and CPU No.4 is 0

- 4): The CPU shared memory occupied with automatic refresh refreshing becomes the total of setting 1 to setting 4. The first and last addresses of the CPU shared memory being used will be displayed in hexadecimal when the number of transmission points are set. The CPU for which the transmission points have been set in setting 1 and setting 2 will become the last address of the setting 2 CPU shared memory. (Up until 811H is used for CPU No.1 and CPU No.2, and up until 821H is used for the CPU No.4 in the illustration shown below.) The CPUs that transmits only setting 1 will become the last address of the setting 1 CPU shared memory. (CPU No.3 is up to the setting 1 address in the illustration shown below.)



- 5): The same number of transmission points must be set for all CPUs on the multiple CPU system. A "PARAMETER ERROR" occurs if the number of transmission points for one CPU is different.

(b) CPU devices

The following devices can be used for automatic refresh purposes (other devices cannot be set up with the GX Developer.)

Settable devices	Caution
Data register (D) Link register (W) File register (R, ZR)	<ul style="list-style-type: none"> <li>The device in the left column occupies one point for every transmission point</li> </ul>
Link relay (B) Internal relay (M) Output (Y)	<ul style="list-style-type: none"> <li>Multiples of 0 or 16 are specified for the first number.</li> <li>The device in the left column occupies one point for every transmission point.</li> </ul>

- 1) CPU devices use the total amount of transmission point devices consecutively from the specified device number to the CPU No.1 to No.4 in the first set range. Set a device number so that the amount of transmission point devices can be secured. Sixteen times the number of transmission points will be set if a bit device is specified in the CPU device. For example, If the total number of transmission points for CPU No.1 to No.4 is ten, then 160 points will be set between B0 and B9F when the B0 link relay is specified.

2) The CPU devices are set as follows.

- It is possible to change the device and set up settings 1 to 4.  
The same devices can also be specified as long as the device range for settings 1 to 4 are not duplicated.

Setting 1: In the case of link relays

Change screens:

PLC	Send range for each PLC			PLC side device	
	PLC share memory G			Dev. starting	B0
	Point (*)	Start	End	Start	End
No.1	2	0800	0801	B0	B1F
No.2	2	0800	0801	B20	B3F
No.3	4	0800	0803	B40	B7F
No.4	2	0800	0801	B80	B9F

Setting 2: In the case of link registers

Change screens:

PLC	Send range for each PLC			PLC side device	
	PLC share memory G			Dev. starting	W0
	Point (*)	Start	End	Start	End
No.1	16	0802	0811	W0	W0F
No.2	16	0802	0811	W10	W1F
No.3	0				
No.4	0				

Setting 3: In the case of link relays

Change screens:

PLC	Send range for each PLC			PLC side device	
	PLC share memory G			Dev. starting	B100
	Point (*)	Start	End	Start	End
No.1	2	0812	0813	B100	B11F
No.2	2	0812	0813	B120	B13F
No.3	4	0804	0807	B140	B17F
No.4	4	0802	0805	B180	B1BF

• It is possible to change the device and set up settings 1 to 4.

• The same devices can be specified for settings 1 to 4. However, as setting 1 in the illustration on the left uses 160 points between B0 and B9F, BA0 and higher can be used for setting 3. No part of a device number can be duplicated, as shown with B0 to B9F on setting 1 and B90 to B10F on setting 3.

The first and last will be calculated automatically with the GX Developer

- Each of the setting 1 to setting 4 devices can be set up independently.  
For example, the CPU No.1 can be set up as a link relay, and the CPU No.2 can be set up as an internal relay.

Refresh setting for CPU No.1

Change screens Setting 1

Change screens Setting 2

PLC	Send range for each PLC			PLC side device	
	Point (*)	Start	End	Start	End
No.1	16	0802	0811	w0	w0F
No.2	16	0802	0811	w10	w1F
No.3	0				
No.4	0				

PLC side device

End	B0
	B1F
	B3F
	B7F
	B9F

• When the CPU No.1 and CPU No.2 devices have been set up with different devices

Set the same point for all the CPU.

Refresh setting for CPU No.2

Change screens Setting 1

Change screens Setting 2

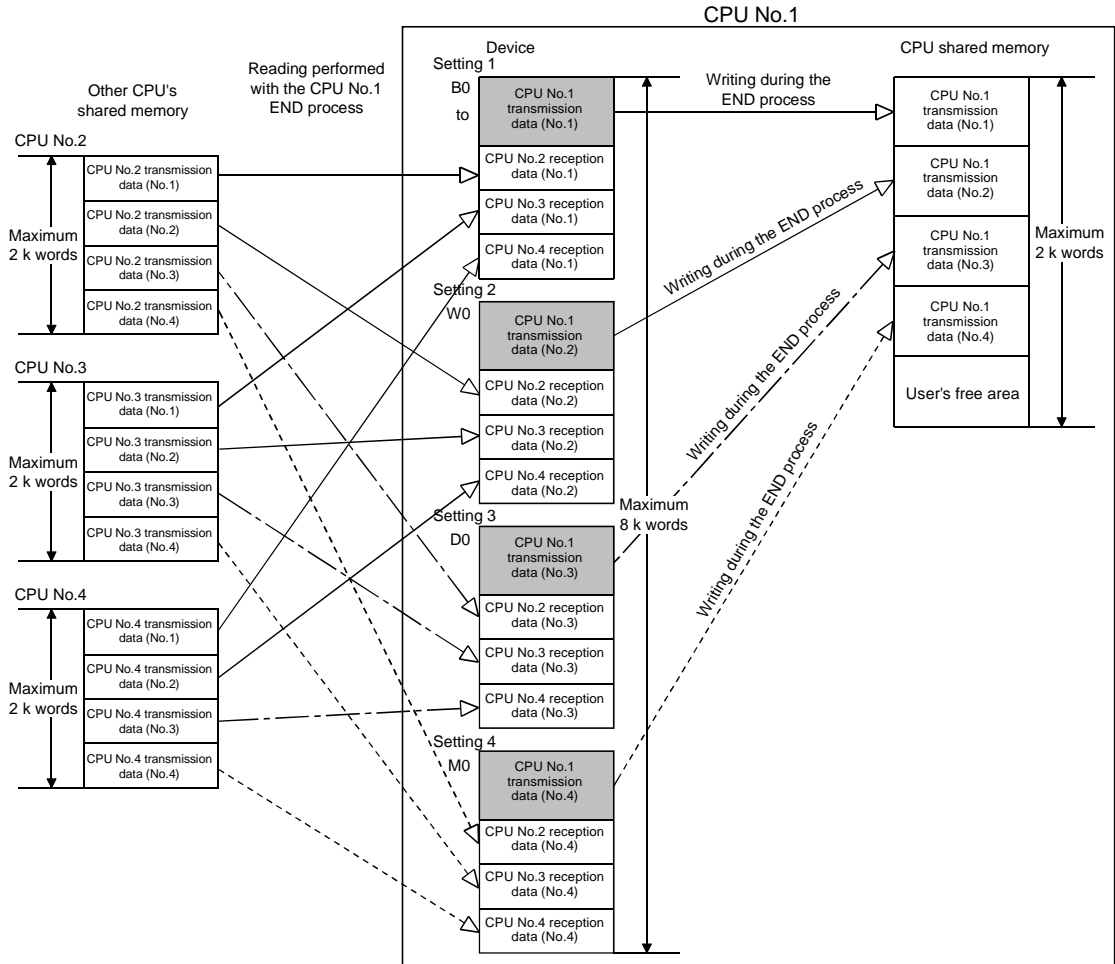
PLC	Send range for each PLC			PLC side device	
	Point (*)	Start	End	Start	End
No.1	16	0802	0811	w0	w0F
No.2	16	0802	0811	w10	w1F
No.3	0				
No.4	0				

PLC side device

End	B0
	B1F
	B3F
	B7F
	B9F

• When the CPU No.1 and CPU No.2 devices have been set up with the same device

- 3) An outline of the operations when the automatic refresh function is divided into four ranges (Setting 1: Link relay (B), Setting 2: Link register (W), Setting 3: Data register (D), Setting 4: Internal relay (M)) and then performed is shown in the illustration below.



(3) Precautions

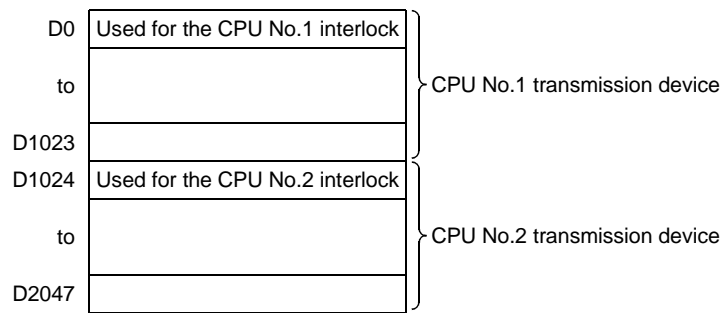
- (a) Device ranges set for the use of the automatic refresh function cannot be set in local devices.  
If the device ranges set for the use of the automatic refresh function are set in local devices, the settings will not be reflected back onto the refresh data.
- (b) Do not set devices for the use of the automatic refresh function in the file register of all programs.  
If devices for the use of the automatic refresh function are set in the file register of all programs, automatic refresh will be performed on the file register that corresponds with the last scan execution type program executed.

- (c) There are cases where old data and new data will become mixed up for each CPU depending on the timing of refreshing the host CPU and reading data from other CPUs.

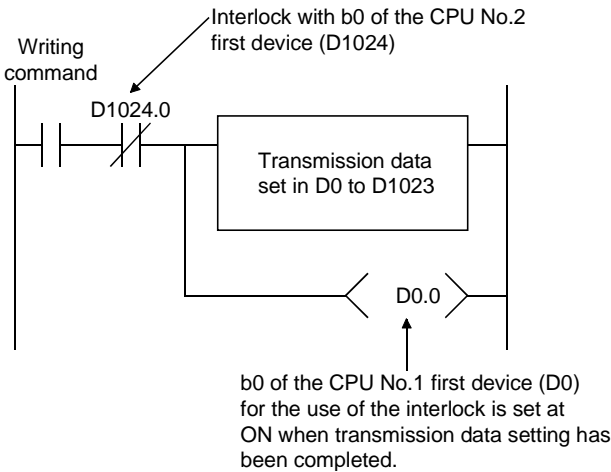
When performing the automatic refresh function, create an interlock program similar to the one shown below that uses the first device to be refreshed for each CPU, and do not use the data from other CPUs when old data does get mixed up with new data.

An example of a program set up with the following multiple CPU setting refresh settings is shown below.

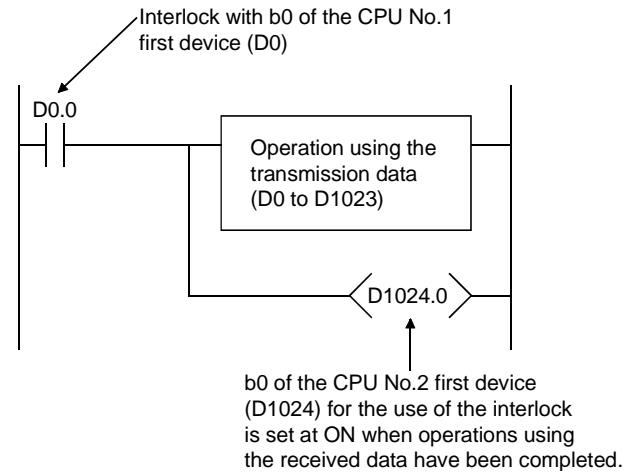
- CPU device: D0
- CPU No.1 transmission points: 1024 points (D0 to D1023)
- CPU vNo.2 transmission points: 1024 points (D1024 to D2047)



Example of a program on the transmission side



Example of a program on the reception side



16.2 Communication with Multiple CPU Dedicated Instructions and Intelligent Function Module Devices

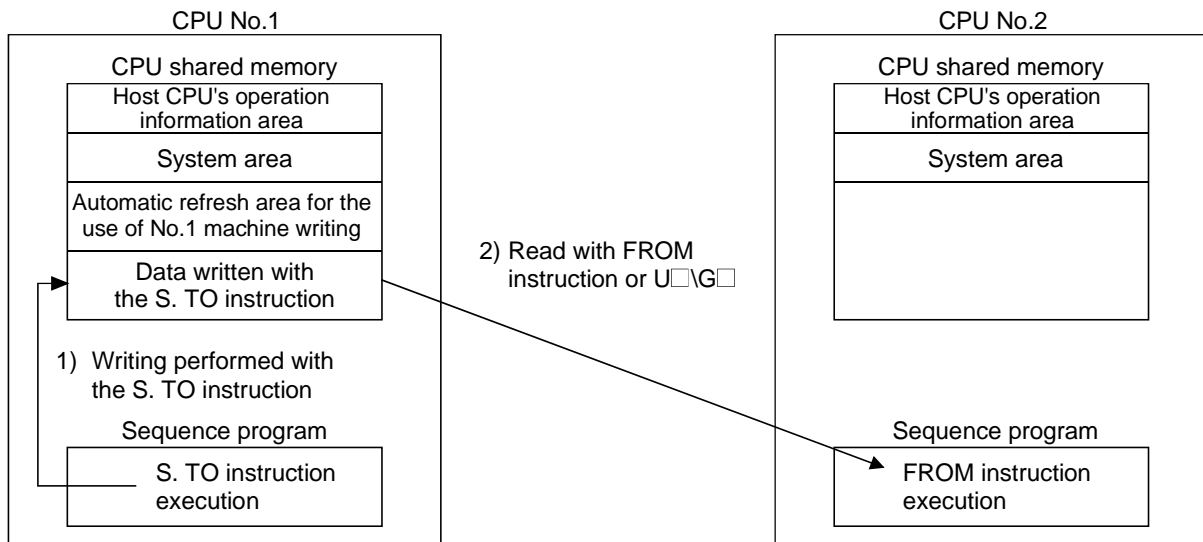
(1) Communication with multiple CPU dedicated instructions (S. TO instruction / FROM instruction) and intelligent function module device (U□\G□)

The High Performance model QCPU of a multiple CPU system can use an S. TO instruction, FROM instruction and intelligent function module device (U□\G□) to access the CPU shared memory of the High Performance model QCPU, Motion CPU and PC CPU module.

The data written in the CPU shared memory of the host CPU with a S. TO instruction can be read by another CPU using a FROM instruction or intelligent function module device (U□\G□).

Contrary to the automatic refresh function for the CPU shared memory, it is possible to read data directly when this instruction is executed.

An outline of a process where data written in the CPU shared memory of CPU No.1 with an S. TO instruction is read by the CPU No.2 using an FROM instruction or intelligent function module device (U□\G□) is shown in the figure below.



CPU No.1 processing

1): Data is written into the user's free area on the CPU No.1 with the S. TO instruction.

CPU No.2 processing

2): An FROM instruction or the intelligent function module device (U□\G□) is used to read data from the free user area of the CPU No.1 to the designated device.

Refer to the following manual for further details on the S. TO and FROM instructions.

QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)

<b>POINT</b>
<p>The Motion CPU cannot use the S. TO instruction, FROM instruction or intelligent function module device.</p> <p>Use "automatic refresh of the CPU shared memory" or "communication dedicated instructions between multiple CPUs" to communicate between the High Performance model QCPU and Motion CPU.</p> <p>For the accessing method from the PC CPU module to the CPU shared memory, refer to the manual of the PC CPU module command between multiple CPUs.</p>



(2) Precautions

- (a) The following values are set in the CPU module's first I/O number with the FROM instruction, the S. TO instruction and instructions that use U□\G□.

PLC No.	PLC No.1	PLC No.2	PLC No.3	PLC No.4
Value set in the first I/O number	3E0H	3E1H	3E2H	3E3H

- (b) Do not perform writing as reading in the system area or automatic refresh area for the CPU shared memory (see Section 16.4).
- (c) An error will not occur when CPUs accessed with the FROM instruction, the S. TO instruction and instructions that use U□\G□ are reset. However, access execution flag (SM390) will remain OFF when instruction execution has been completed.
- (d) Establish an interlock to prevent simultaneous access during interactive data communication with the FROM instruction, the S. TO instruction and instructions that use U□\G□. There are cases where old data and new data will be mixed together if simultaneous access is carried out.
- (e) The instruction that uses the S. TO instruction/U□\G□ cannot be used to write data to the CPU shared memory of other PLCs. "SP. UNIT ERROR (error code: 2115)" occurs if data is written to the CPU shared memory of other CPUs with the instruction that uses U□\G□. "SP. UNIT ERROR (error code: 2117)" occurs if data is written to the CPU shared memory of other CPUs with the instruction that uses the S. TO instruction.
- (f) "SP. UNIT ERROR (error code: 2114)" also occurs if data is written into the CPU shared memory of the host CPU with instructions that use U□\G□.
- (g) "SP. UNIT ERROR (error code: 2114)" occurs if data is read from the CPU shared memory of the host CPU with the FROM instruction and instructions that use U□\G□.
- (h) "SP. UNIT ERROR (error code: 2110)" also occurs if access is attempted on a non-mounted CPU with instructions that use U□\G□.

16.3 Interactive Communications between The High Performance model QCPU and Motion CPU

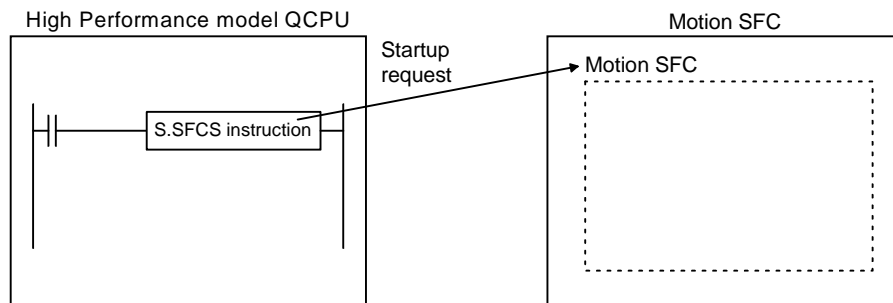
16.3.1 Control commands from the High Performance model QCPU to the Motion CPU

It is possible to issue control commands from the High Performance model QCPU to the Motion CPU, and read and write device data with the Motion dedicated CPU instructions listed below.

(Control commands from Motion CPU to Motion CPU can not be used.)

Instruction name	Description
S.SFCS SP.SFCS	Requests startup of the motion SFC program.
S.SVST SP.SVST	Requests the start of operations for the servo program.
S.CHGV SP.CHGV	Changes the speed of the axes during positioning and JOG operations.
S.CHGT SP.CHGT	Changes the torque control value during operation and suspension when in the real mode.
S.CHGA SP.CHGA	Changes the current values of the halted axes, the synchronized encoder, and the cam axes.

For example, it is possible to start up the Motion CPU's motion SFC from the High Performance model QCPU with the S (P).SFCS instruction.



**POINT**

One High Performance model QCPU module can operate up to 32 " Motion dedicated CPU instructions " and "communication dedicated commands between multiple CPUs (omitting the S (P).GINT instruction)" at one time. However, if the Motion dedicated CPU instructions and communication dedicated instructions between multiple CPUs (omitting S (P).GINT instruction) are made at the same time, the instructions will be executed in order from the first instruction accepted. If there are 33 or more unexecuted instructions, an "OPERATION ERROR (error code: 4107)" will be triggered.

**REMARK**

Refer to the Motion CPU Programming manual for details on and the necessity of use of the motion only instructions.

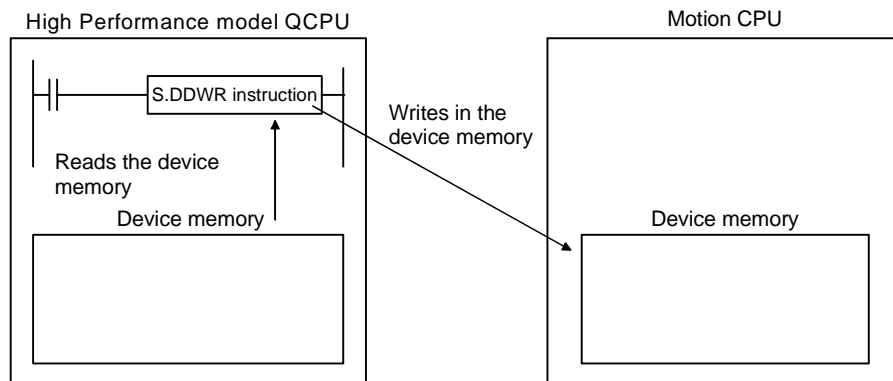
16.3.2 Reading and writing device data

It is possible to read and write device data into the Motion CPU from the High Performance model QCPU with the communication dedicated instructions between multiple CPUs listed in the table below.

(Reading or writing can not take place from the High Performance model QCPU to the High Performance model QCPU, the High Performance model QCPU to PC CPU module, Motion CPU to the High Performance model QCPU, or Motion CPU to Motion CPU.)

Instruction name	Description	CPU module	
		Motion CPU	PC CPU module
S.DDWR SP.DDWR	Writes host CPU device data into other CPU devices.	○	×
S.DDRD SP.DDRD	Reads other CPU device data into the host CPU.	○	×
S.GINT SP.GINT	Requests start up of other CPU interruption programs.	○	○

For example, High Performance model QCPU device data can be written into the Motion CPU's device data with the S.DDWR instruction of the communication dedicated instruction between multiple CPUs.



**POINT**

One High Performance model QCPU can operate up to 32 "Motion dedicated CPU instructions" and "communication dedicated commands between multiple CPUs (omitting the S (P).GINT instruction)" at one time. However, if the Motion dedicated CPU instructions and communication dedicated instructions between multiple CPUs (omitting S (P).GINT instruction) are made at the same time, the instructions will be executed in order from the first instruction accepted. If there are 33 or more unexecuted instructions, an "OPERATION ERROR (error code: 4107)" will be triggered.

**REMARK**

Refer to the Motion CPU Programming Manual for details on and the necessity of use of the communication dedicated instructions between multiple CPUs.

16.4 CPU Shared Memory

The CPU shared memory is for exchanging data between CPU modules, and consists of 4,096 words between 0H and FFFH.

The CPU shared memory consists of four areas; the host CPU operation information area, the system area, the automatic refresh area, and the user's free area. An area consisting of the number of automatic refresh points from 800H is used as the automatic refresh area when the automatic refresh of device data is set up. The beginning of the user's free area starts from the address immediately after the end of the automatic refresh area. 800H to 811H becomes the automatic refresh area if the number of automatic refresh points is 18 (11H points,) and the area after 812H becomes the user's free area. The configuration of the CPU shared memory and the necessity of accessing sequence programs are shown in the illustration below.

CPU shared memory		Host CPU		Other CPUs	
		Writing*1	Reading	Writing	Reading*2
0H	Host CPU operation information area	Disable	Disable	Disable	Enable
to 1FFH					
200H	System area	Disable	Disable	Disable	Disable
to 7FFH					
800H	Automatic refresh area	Disable	Disable	Disable	Disable
to FFFH	User's free area	Enable	Disable	Disable	Enable

**REMARK**

- \*1: Use the S. TO instruction to write the free user area of the host CPU from the High Performance model QCPU. The Motion CPU is not provided with a S. TO instruction, so that it cannot write in the free user area of the host CPU. For the writing method from the PC CPU module to the free user area of the host PLC, refer to the manual of the PC CPU module.
- \*2: To read from the High Performance model QCPU, use the FROM instruction or intelligent function module device (U□G□). Because the Motion CPU is not provided with the FROM instruction or intelligent function module device, data cannot be read from the Motion CPU. For reading from the PC CPU module, refer to the manual of the PC CPU module.

(1) Host CPU operation information area (0H to 1FH)

- (a) The following information is stored in the host CPU with multiple CPU system. These will all remain as 0 and will not change in the case of single CPU system. \*1

Table 16.1 List of host CPU operation information area

CPU shared memory address	Name	Detail	Description * 2	Corresponding special register
0H	Information availability	Information availability flag	The area to confirm if information is stored in the host CPU's operation information area (1H to 1FH,) or not. <ul style="list-style-type: none"> <li>• 0: Information not stored in the host CPU's operation information area</li> <li>• 1: Information stored in the host CPU's operation information area</li> </ul>	—
1H	Diagnostic error	Diagnostic error number	The numbers of errors during diagnostics is stored with BIN.	SD0
2H	Time the diagnosis error occurred	Time the diagnosis error occurred	The year and month that the error number was stored in the CPU shared memory's 1H address is stored with two digits of the BCD code.	SD1
3H			The day and time that the error number was stored in the CPU shared memory's 1H address is stored with two digits of the BCD code.	SD2
4H			The minutes and seconds that the error number was stored in the CPU shared memory's 1H address is stored with two digits of the BCD code.	SD3
5H	Error information identification code	Error information identification code	Stores an identification code to determine what error information has been stored in the common error information and individual error information.	SD4
6H to 10H	Common error information	Common error information	The common information corresponding with the number of the error during diagnostic is stored.	SD5 to SD15
11H to 1BH	Individual error information	Individual error information	The individual information corresponding with the number of the error during diagnostic is stored.	SD16 to SD26
1CH	Empty	—	Cannot be used	—
1DH	Switch status	CPU switch status	Stores the CPU module switch status.	SD200
1EH	LED status	CPU-LED status	Stores the CPU module's LED bit pattern.	SD201
1FH	CPU operation status	CPU operation status	Stores the CPU module's operation status.	SD203

- (b) The host CPU's operation information area is updated when the contents of the corresponding register change. However, there are times when changes in the corresponding register are relayed by a maximum of 200ms when the High Performance model QCPU's scan time is 200ms or less. There are times when changes in the corresponding register are delayed by 200ms or more if the High Performance model QCPU's scan time exceeds 200ms.
- (c) The High Performance model QCPU of another CPU can use FROM instruction or intelligent function module device to read data from the action data area of the host CPU. However, because there is a delay in data updating, use the read data for monitoring purposes.

**REMARK**

- \*1: For the Motion CPU, 5H to 1CH of the host CPU's operation information area is not used. If 5H to 1CH of the host CPU's operation information area is read from the Motion CPU, it will be read as "0."
- \*2: Refer to the corresponding special registers for further details.

(2) System area (200H to 7FFH)

The area used by the High Performance model QCPU, Motion CPU and PC CPU module systems (OS.) This is used by the OS when communication dedicated instructions between multiple PLCs are executed.

(3) Automatic refresh area

The area used when the multiple CPU system is automatically refreshed. Writing is not enabled with the S. TO instruction, and reading is not enabled with the FROM instruction or intelligent function module device (U□\G□).

(4) User's free area

The area for performing communication between CPU modules with the multiple CPU system's S. TO instruction, FROM instruction and intelligent function module device (U□\G□).

The area used after the number of points set for automatic refresh is used. (An area between 800H and FFH can be used as the user's free area when automatic refresh is not being performed.)

## 17 COMMUNICATIONS BETWEEN THE MULTIPLE CPU SYSTEM'S I/O MODULES AND INTELLIGENT FUNCTION MODULES

### 17.1 Range of Control CPU Communications

The relationship between control CPUs and control modules (I/O modules, intelligent function modules, special function modules) is the same as with independent CPU systems.

There is no restriction to control the control module with the control CPU.

### 17.2 Range of Non-control CPU Communications

It is possible for non-control CPUs to read the contents of the intelligent function module's buffer memory.

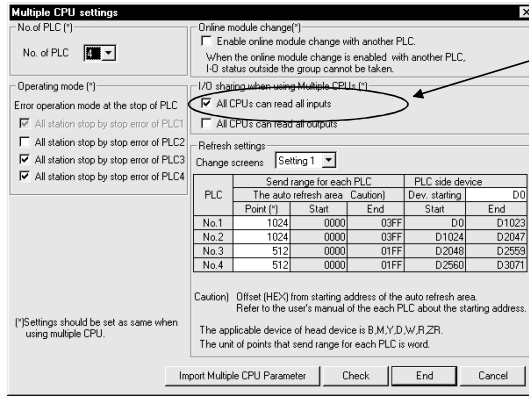
It is also possible to load non-control module input (X) ON/OFF data and another CPU module output (Y) ON/OFF data with the CPU parameters.

Input modules, composite I/O module controlled by other CPUs can be used as interlocks for the host CPU, and the output status to external equipment being controlled by other CPUs can be confirmed.

However, it is not possible for non-control CPUs to output ON/OFF data to output modules, composite I/O module or intelligent function modules, or write in the buffer memory of intelligent function modules.

(1) Loading input (X) from input modules and intelligent function modules

The "Out of group input/output settings" setting in the PLC parameter's multiple CPU settings determines whether input can be loaded from input modules and intelligent function modules being controlled by other CPUs.



I/O sharing when using Multiple CPUs  
 All CPUs can read all inputs  
 All CPUs can read all outputs

(a) When "Load input condition outside of group" has been set

- 1) Loads ON/OFF data from the input and intelligent function modules being controlled by the other CPUs by performing input refresh before a sequence program calculation starts.
- 2) Input (X) loading is performed for the modules mounted onto the following additional base unit slots.

I/O allocation type	Mounted module	Remarks
None	Input module	—
	Intelligent function module	—
Input	Input module	—
	Output module	Loads OFF data
Intelli.	Intelligent function module	—

- 3) It is possible to load ON/OFF data from input modules and intelligent function modules with direct access input.
- 4) Remote station input, such as empty slots, MELSECNET/H and CC-Link, cannot be loaded.  
Use automatic refresh of device data to use the ON/OFF input data for MELSECNET/H, CC-Link and other remote stations in other CPUs.

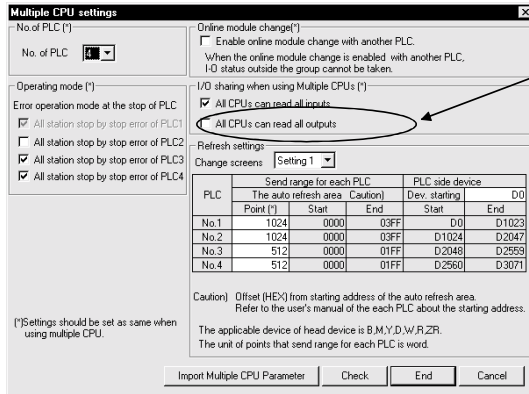
(b) When "Do not load input condition outside of group" has been set

It is not possible to loads ON/OFF data from input modules and intelligent function modules being controlled by other CPUs (remains at OFF.)



(2) Loading output (Y)

The "Out of group input/output settings" setting in the PLC parameter's multiple CPU settings determines whether output can be loaded from output modules and intelligent function modules being controlled by other CPUs.



I/O sharing when using Multiple CPUs  
 All CPUs can read all inputs  
 All CPUs can read all outputs

(a) When "Load output condition outside of group" has been set

- 1) Loads to the host CPU's output (Y) the ON/OFF data that is output to the output and intelligent function modules by the other CPUs by performing output refresh before a sequence program calculation starts.
- 2) Output (Y) loading is performed for the modules mounted onto the following additional base unit slots.

I/O allocation type	Mounted module
None	Output module
	Intelligent function module
Output	Input module
	Output module
	Composite I/O module
Intelli.	Intelligent function module

- 3) It is possible to load output ON/OFF data being controlled by other CPUs with direct access output.
  - 4) Remote station output, such as empty slots, MELSECNET/H and CC-Link, cannot be loaded.  
 Use automatic refresh of CPU shared memory and send the ON/OFF output data for remote stations to use the ON/OFF output data for MELSECNET/H, CC-Link and other remote stations in other CPUs.
- (b) When "Do not load output condition outside of group" has been set  
 It is not possible to load ON/OFF data output to output modules and intelligent function modules by other PLCs into the host CPU's output (Y) (remains at OFF.)

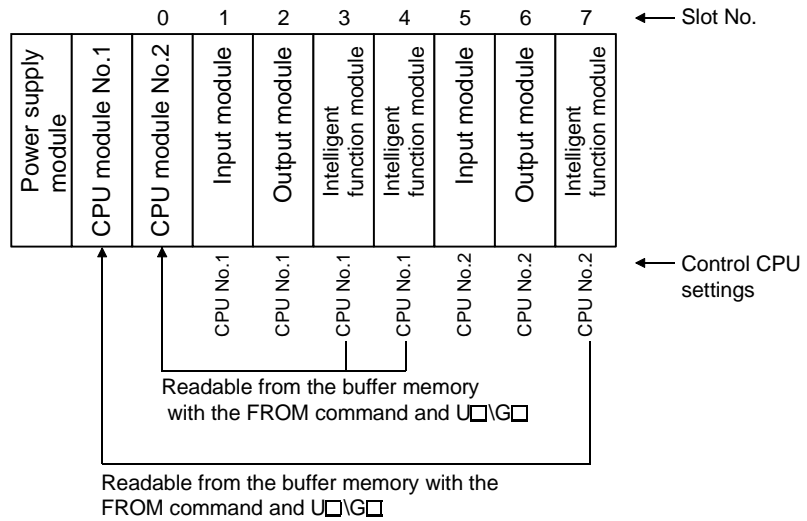
(3) Output to output modules and intelligent function modules

It is not possible to output ON/OFF data to non-control modules. ON/OFF will be performed within the High Performance model QCPU when the output from output modules and intelligent function modules controlled by other PLCs, such as sequence programs, have been set to ON/OFF, but this will not be output to output modules or intelligent function modules.

(4) Accessing the intelligent function module buffer memory

(a) It is possible to read data from the buffer memory of intelligent function modules being controlled by other CPUs with the commands listed below.

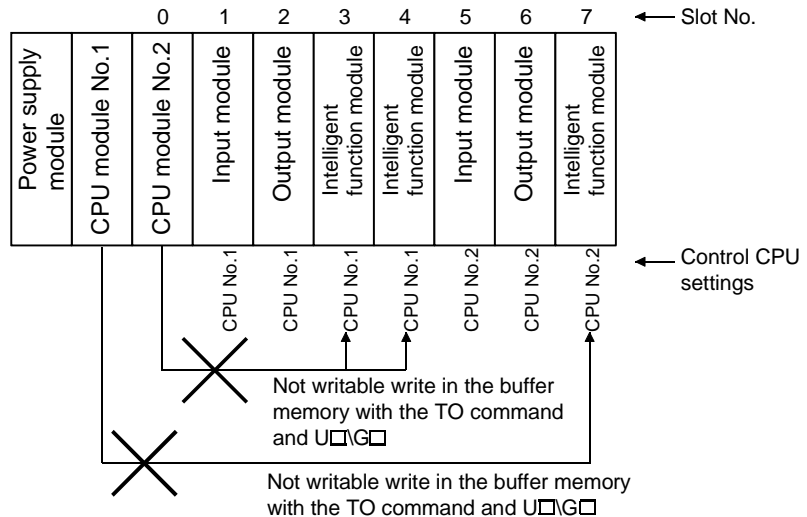
- FROM command
- Commands that use intelligent function module devices (U□\G□)



(b) It is not possible to write in the buffer memory of intelligent function modules being controlled by other CPUs.

- TO instruction
- Intelligent function module devices (U□\G□)
- Intelligent function modules dedicated commands

An "SP UNIT ERROR (error code: 2116)" will be triggered if an attempt to write in the intelligent function module controlled by other CPUs is carried out.



(5) Accessing MELSECNET/H modules

Only control PLCs can access MELSECNET/H modules.

Link direct devices cannot be used in MELSECNET/H modules being controlled by other CPUs.

"OPERATION ERROR (error code: 4102)" occurs if a program that uses link direct devices is used in MELSECNET/H modules being controlled by other CPUs.

## 18 PROCESSING TIME FOR MULTIPLE CPU SYSTEM HIGH PERFORMANCE MODEL QCPUs

### 18.1 Concept behind CPU Scanning Time

The concept behind multiple CPU system scanning time is the same as the single CPU system.

See Section 11.1 for details of the scan time concept.

This chapter provides explanations on the factors to be added to the scan time calculated as explained in Section 11.1 and the method of calculating processing time when configuring multiple CPU system.

#### (1) I/O refresh time

Input refresh time is calculated in accordance with the equation explained in Section 11.1.

The I/O refresh time for the following values only are prolonged when bus access overlaps with other CPUs.

(Prolonged time) =  $\frac{(\text{input points} + \text{output points})}{16} \times N3 \times (\text{number of other CPUs}) (\mu s)$

Use the following values for N3

CPU type	N3	
	Systems with only a main base unit	Systems that include additional base units
Q02CPU	8.7 $\mu s$	21 $\mu s$
Q02HCPU, Q06HCPU		
Q12HCPU, Q25HCPU		

#### (2) Total value of command execution time

Refer to the following manual for details on the processing time of special multiple CPU commands, and the processing time for commands that have different processing times with multiple CPU system.

- QCPU (Q mode)/QnACPU Programming Manual (Common Instructions)

#### (3) END process

The following tables shows the END processing time.

CPU type	END processing time
Q02CPU	0.42 ms
Q02HCPU, Q06HHCPU, Q12HCPU, Q25HCPU	0.19 ms

18.2 Factor to Prolong the Scan Time

The processing time for multiple CPU system is prolonged in comparison with single CPU system when the following functions are used.

Add the following values to the values calculated in Sections 11.1 and 18.a to acquire the amount of time used by these functions.

- Multiple CPU system automatic refresh
- MELSECNET/H refreshing
- CC-Link automatic refresh

(1) Automatic refresh of CPU shared memory

- (a) The amount of time required to perform the refresh function set up with the multiple CPU settings.

This value is the total amount of time required for writing into the host CPU's CPU shared memory, and the amount of time required to read from other CPUs' CPU shared memories.

These values are added when setting up the refresh settings with the PLC parameter multiple CPU settings.

- (b) The automatic refresh period of the CPU shared memory is calculated in the following equation.

(Automatic refresh time) = (N1 + (received word points) × N2) × (number of other CPUs) + (N3 + (transmitted word points) × N4) (μs)

- The received word points must equal the word points transmitted by other CPUs.  
For example, if the host CPU is the CPU No.1, then this value must equal the number of points transmitted for the CPU No.2 to CPU No.4.
- Use the following values for N1 to N4.

CPU type	N1	N2	N3	N4
Q02CPU	82 μs	0.52 μs	106 μs	0.17 μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	27 μs	0.44 μs	27 μs	0.08 μs

- (c) The amount of time required for the automatic refresh process will be prolonged by the following amount of time when processing is duplicated with the automatic refresh function on other PLCs.

(Prolonged time) = (transmitted/received word point) × N5 × (number of other CPUs) (μs)

Use the following values for N5

CPU type	N5	
	Systems with only a main base unit	Systems that include additional base units
Q02CPU	0.54 μs	1.3 μs
Q02HCPU, Q06HCPU		
Q12HCPU, Q25HCPU		

(2) MELSECNET/H refresh

- (a) The amount of time required for performing the refresh process between High Performance model QCPU and MELSECNET/H network modules. Refer to the following manual for details on the refresh time for MELSECNET/H.
  - Q Corresponding MELSECNET/H Network System Refresh Manual
- (b) The amount of time required for the automatic refresh process will be prolonged only by the following amount of time when requests for refreshing are issued by other MELSECNET/H modules at the same time on a multiple CPU system.

(Prolonged time) = (transmitted/received word point) × N5  
 × (number of other CPUs) (μs)

The number of words transmitted/received is the total value of the following transferal data.

- Link refresh data :  $\frac{(LB + LX + LY + SB)}{16} + LW$
- Data transferred to the memory card's file register :  $\frac{(LB + LX + LY + SB)}{16} + LW$
- Transferal between data links :  $(\frac{LB}{16} + LW) \times 2$

Refer to the following table for N5

CPU type	N5	
	Systems with only a main base unit	Systems that include additional base units
Q02CPU	0.54 μs	1.30 μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU		

(3) CC-Link automatic refresh

- (a) The amount of time required for performing the refresh process between High Performance model QCPU and CC-Link master local modules. Refer to the following manual for details on the automatic refresh time for CC-Link.
  - QJ61BT11 CC-Link System Master Local Module User's Manual
- (c) The amount of time required for the automatic refresh process will be prolonged only by the following amount of time when requests for refreshing are issued by other CC-Link modules at the same time on a multiple CPU system.

(Prolonged time) = (transmitted/received word point) × N5  
 × (number of other CPUs) (μs)

The amount of data transmitted/received is the following transferal data.

- Link refresh data :  $\frac{(RX + RY + SB)}{16} + SW$

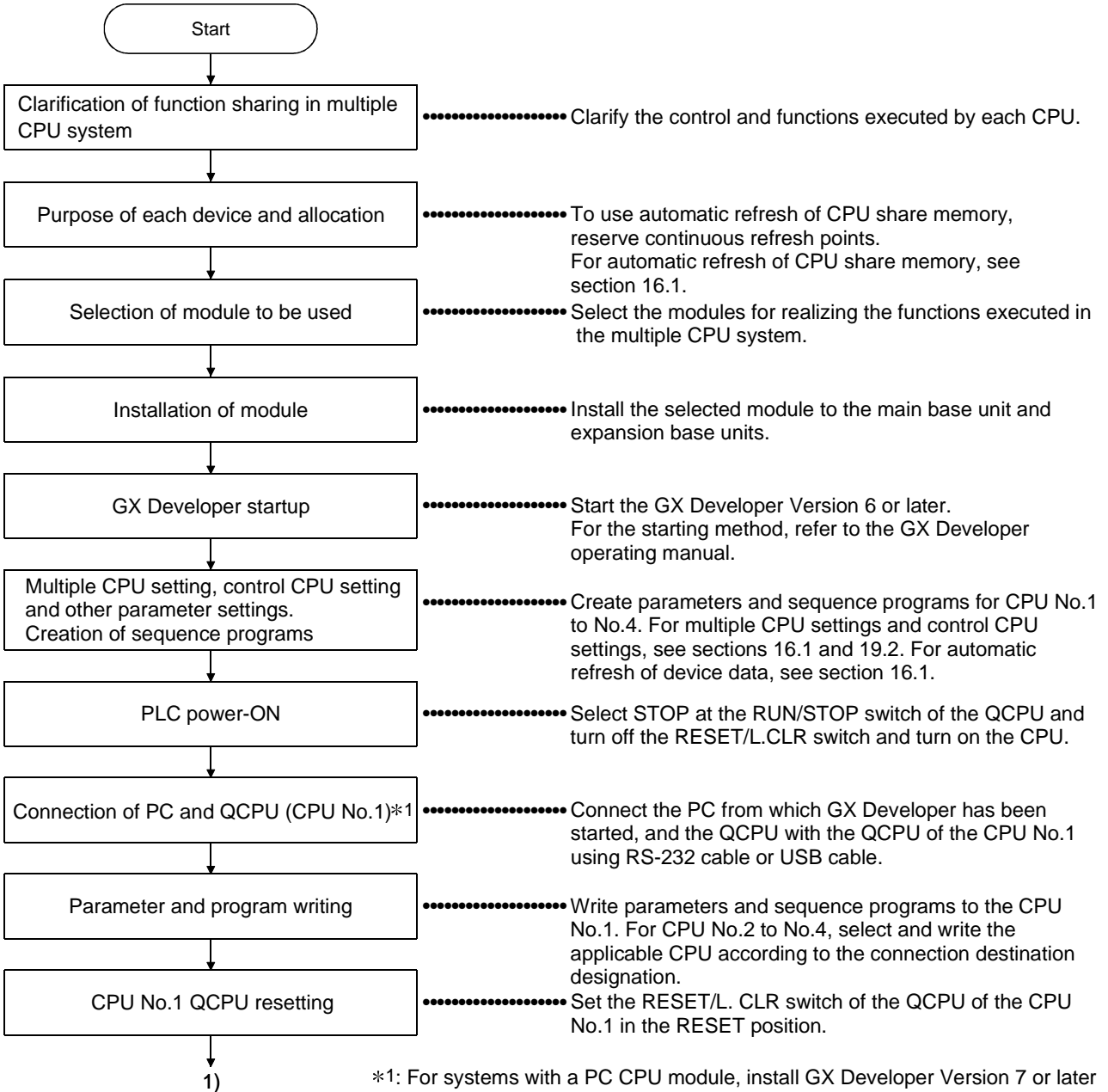
Refer to the following table for N5

CPU type	N5	
	Systems with only a main base unit	Systems that include additional base units
Q02CPU	0.54 μs	1.30 μs
Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU		

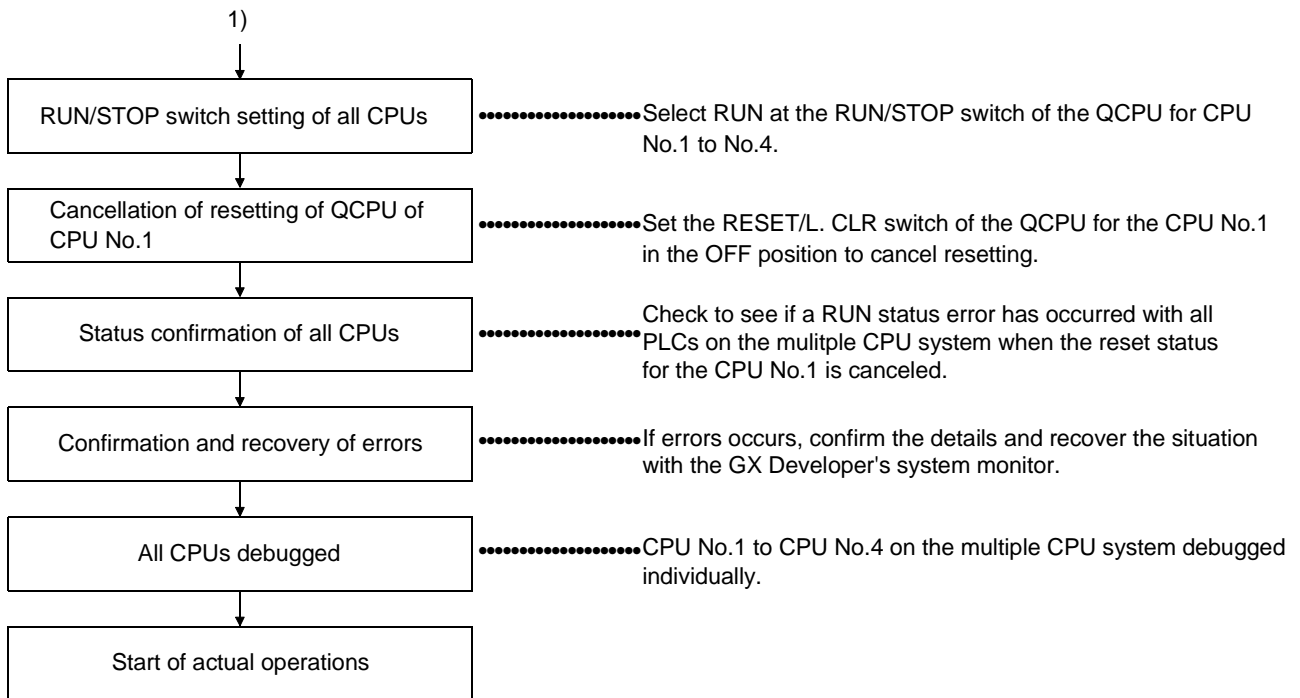
## 19 STARTING UP THE MULTIPLE CPU SYSTEM

This Chapter explains the standard procedures for starting up the multiple CPU system.

### 19.1 Flow-chart for Starting Up the Multiple CPU System



\*1: For systems with a PC CPU module, install GX Developer Version 7 or later to the PC CPU module to connect the High Performance QCPU with GX Developer via the bus. Refer to the GX Developer Version 7 or later operating manual for details.





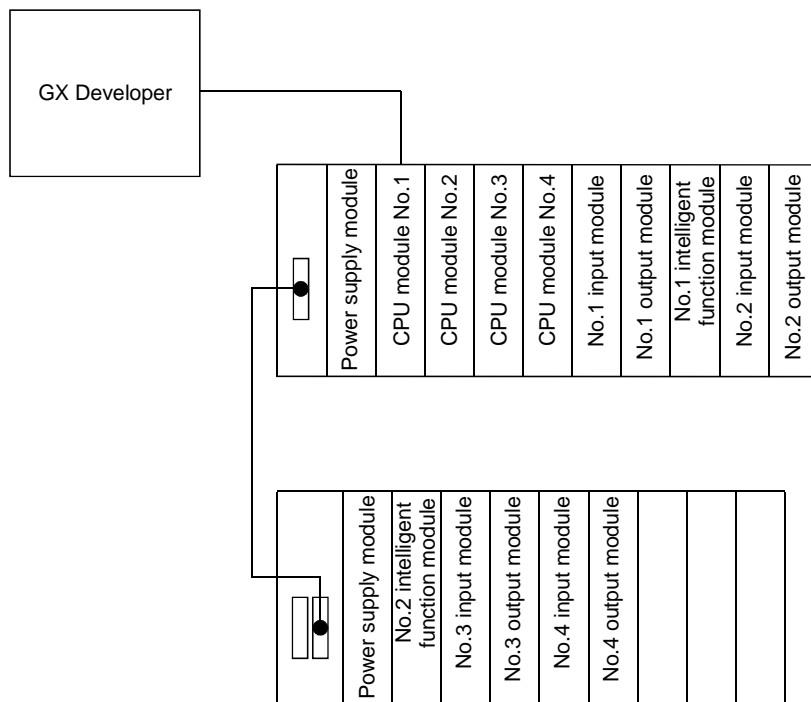
### 19.2 Setting Up the Multiple CPU System Parameters (Multiple CPU Settings, Control CPU Settings)

This section explains the procedures for setting up the multiple CPU system parameters with GX Developer.

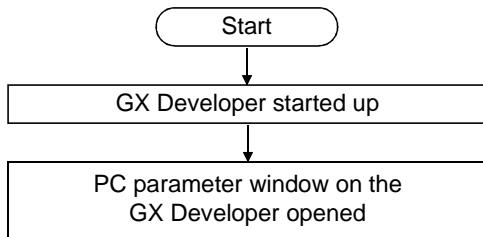
Refer to the GX Developer's operation manual for details on setting up all other parameters.

#### 19.2.1 System configuration

The following shows an example procedures for setting up the multiple CPU system parameters.

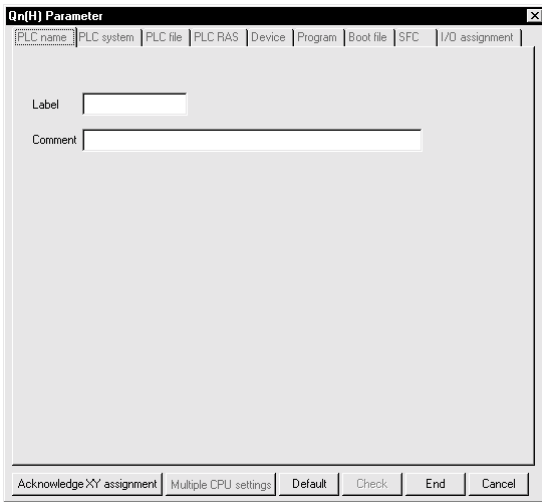


19.2.2 Creating new systems

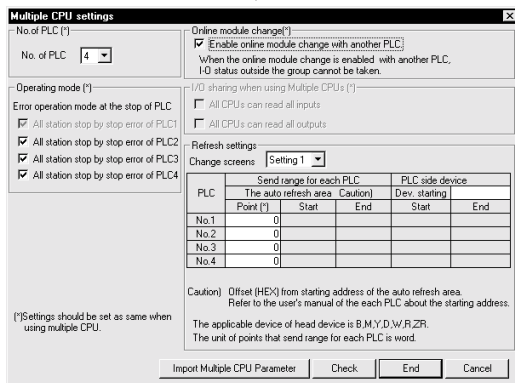


Refer to GX Developer operating manual

Refer to GX Developer operating manual



Select "Multiple CPU Settings" to display the multiple CPU setup window.

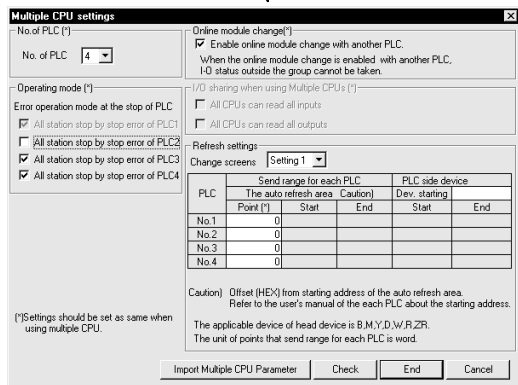


Setting the number of CPUs (required item)

- Sets the number of CPU modules to be mounted onto the main base unit with the multiple CPU system.

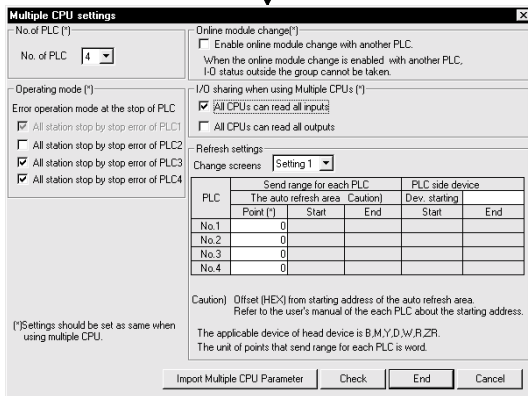
1)

1)



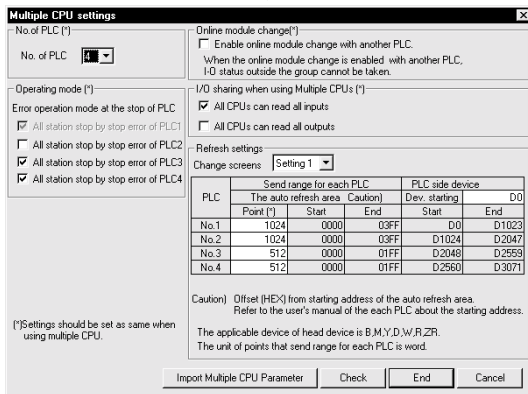
- Setting the operating mode (optional)
- Selects whether to halt operations for all CPUs or continue with operations when a stop error occurs.  
Default: Stop all PLCs upon a stopping error at CPU No.2, No.3 or No.4. (No check).
  - For example, if the tick beside the "All station stop by stop error of CPU2" is removed, the operations for all other CPUs will continue even when an error occurs in the CPU No.2.
  - The operation mode for the CPU No.1 cannot be changed.

↓



- Out of the group input/output settings (optional)
- Sets whether or not the I/O status of non-control PLCs are acquired.  
Default: Do not acquire. (No check)

↓

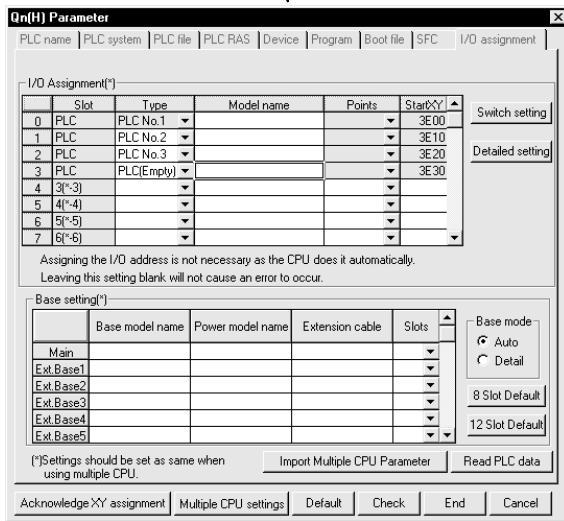


- Multiple CPU system (optional)
- Sets the device and number of CPU share memory G points to perform data communications with the automatic refresh process between CPU modules.
  - This is linked from the device number set with the first device to the number of CPU share memory G points and used. The 1st CPU share memory G point occupies the points shown in the table below.

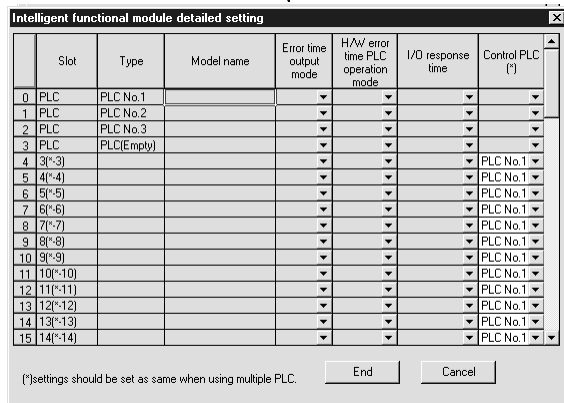
Device	Points occupied
B, M, Y	16 points
D, R, ZR	1 point

2)

2)

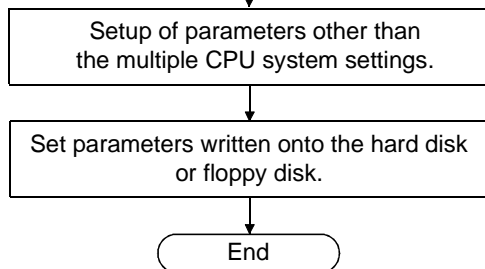


Selects "CPU (Empty)" for the slots on which CPU modules are not to be mounted by type. Select "Detailed Settings" on the I/O assignment window to display the detail settings window.

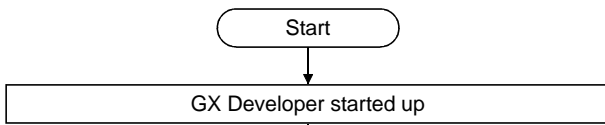


Control CPU settings (required item)

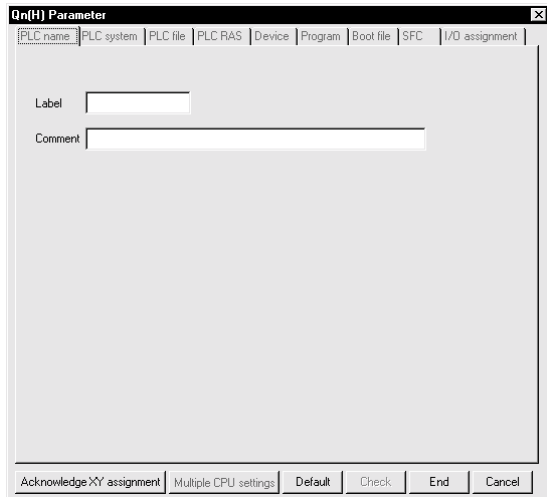
- Selects the control CPUs (CPU No.1 to No.4) for each slot.
- Function version A intelligent function modules set the control CPU No.1.
- Output modules and special function modules that support the AnS series set a single CPU in all slots.



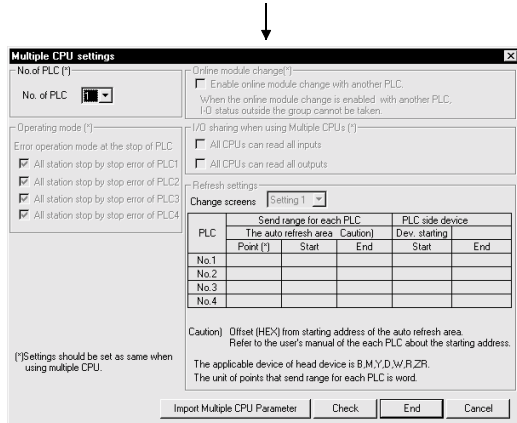
19.2.3 Using existing preset multiple CPU settings and I/O allocations



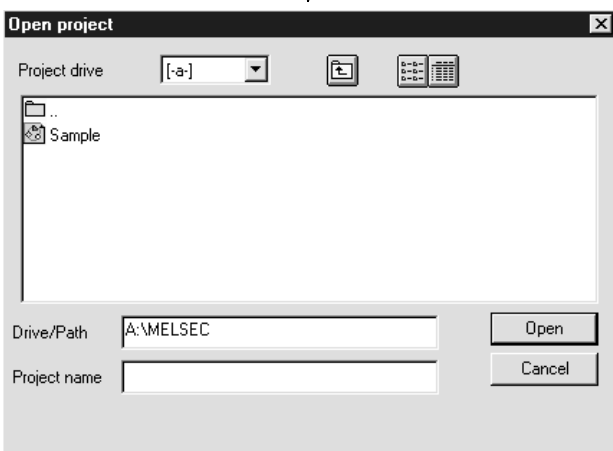
Refer to the GX Developer's operation manual



Opens the GX Developer's PC parameter setup window. Select "Multiple CPU settings" to display the multiple CPU setup window.



Transferring multiple CPU settings  
Click on "Diversion multiple CPU parameters."

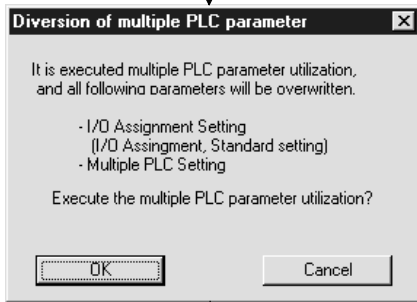


Setting up transferred projects

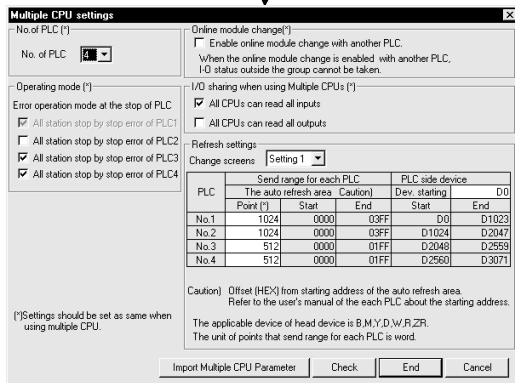
- Select the project into which existing multiple CPU settings and I/O allocations are to be transferred.
- Click on "Open."

1)

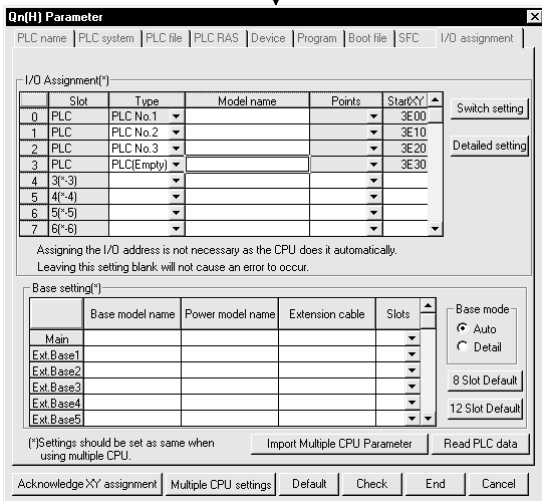
1)



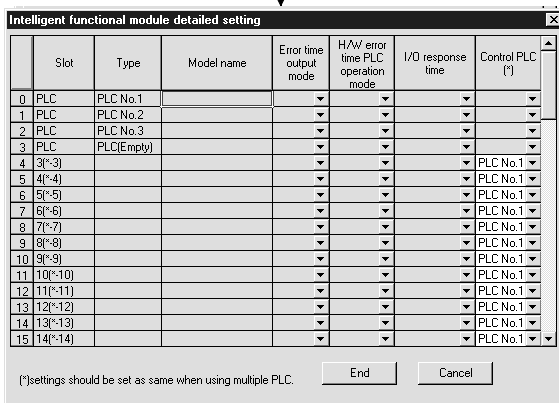
The multiple CPU settings and I/O Assignment Setting data are read and written into the specified project when "OK" is selected.



Confirming the multiple CPU settings  
When changing the CPU devices in the "Refresh settings", enter the device number after it has been changed. ("\*" indicates the item can be changed.)

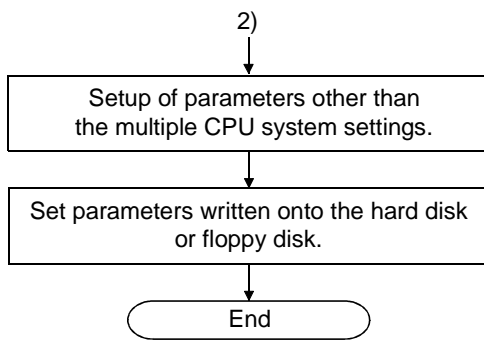


Confirm the I/O Assignment and standard settings on the I/O Allocation Window.  
Select "Detailed Settings" to display the detailed setting window.



Confirm the control CPU settings.

2)



APPENDICES

APPENDIX 1 Special Relay List



Special relays, SM, are internal relays whose applications are fixed in the PLC. 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 headings in the table that follows have the following meanings.

Item	Function of Item
Number	• Indicates the number of the special relay.
Name	• Indicates the name of the special relay.
Meaning	• Indicates the nature of the special relay.
Explanation	• Contains detailed information about the nature of the special relay.
Set by (When set)	<ul style="list-style-type: none"> <li>• Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed.</li> <li>&lt;Set by&gt; <ul style="list-style-type: none"> <li>S : Set by system</li> <li>U : Set by user (in sequence program or test operation at a GX Developer)</li> <li>S/U : Set by both system and user</li> </ul> </li> <li>&lt;When set&gt; → indicated only if setting is done by system. <ul style="list-style-type: none"> <li>Each END : Set during each END processing</li> <li>Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN)</li> <li>Status change : Set only when there is a change in status</li> <li>Error : Set when error is generated</li> <li>Instruction execution : Set when instruction is executed</li> <li>Request : Set only when there is a user request (through SM, etc.)</li> </ul> </li> </ul>
Corresponding ACPU M9 □ □ □	<ul style="list-style-type: none"> <li>• Indicates special relay M9 □ □ □ corresponding to the ACPU. (Change and notation when there has been a change in contents)</li> <li>• Items indicated as "New" have been newly added for High performance model QCPU/QnACPU.</li> </ul>
Corresponding CPU	<ul style="list-style-type: none"> <li>• Indicates the corresponding CPU type name.</li> <li>○+Rem: Can be applied to all CPU types and MELSECNET/H remote I/O modules.</li> <li>○: Can be applied to all types of CPU</li> <li>QCPU: Can be applied to High Performance model QCPU</li> <li>QnA: Can be applied to QnA series and Q2ASCPU Series</li> <li>Remote: Can be applied to the MELSECNET/H remote I/O modules.</li> <li>Each CPU type name: Can be applied only to the specific CPU. (e.g. Q4ARCPU, Q3ACPU)</li> </ul>

For details on the following items, refer to the following manuals:

- Networks → • For Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
  - For Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)
  - For QnA/Q4AR MELSECNET/10 Network System Reference Manual
- SFC → QCPU(Q Mode)/QnACPU Programming Manual (SFC)

<b>POINT</b>
(1) SM1200 to SM1255 are used for QnACPU. These relays are vacant with QCPU.
(2) Special relays SM1500 and later are dedicated for Q4ARCPU.



Special Relay List

(1) Diagnostic Information

APP

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM0	Diagnostic errors	OFF: No error ON : Error	<ul style="list-style-type: none"> <li>ON if diagnosis results show error occurrence (Includes when an annunciator is ON, and when an error is detected with CHK instruction)</li> <li>Stays ON subsequently even if normal operations restored</li> </ul>	S (Error)	New	○+Rem
SM1	Self-diagnosis error	OFF: No self-diagnosis errors ON : Self-diagnosis	<ul style="list-style-type: none"> <li>Comes ON when an error occurs as a result of self-diagnosis.</li> <li>Stays ON subsequently even if normal operations restored</li> </ul>	S (Error)	M9008	
SM5	Error common information	OFF: No error common information ON : Error common information	<ul style="list-style-type: none"> <li>When SM0 is ON, ON if there is error common information</li> </ul>	S (Error)	New	
SM16	Error individual information	OFF: No error common information ON : Error common information	<ul style="list-style-type: none"> <li>When SM0 is ON, ON if there is error individual information</li> </ul>	S (Error)	New	
SM50	Error reset	OFF →ON : Error reset	<ul style="list-style-type: none"> <li>Conducts error reset operation</li> <li>See section 11.3 for further information</li> </ul>	U	New	
SM51	Battery low latch	OFF: Normal ON : Battery low	<ul style="list-style-type: none"> <li>ON if battery voltage at CPU module or memory card drops below rated value. Stays ON subsequently even after normal operation is restored</li> <li>Synchronous with BAT. ALA/BAT. LED</li> </ul>	S (Error)	M9007	○
SM52	Battery low	OFF: Normal ON : Battery low	<ul style="list-style-type: none"> <li>Same as SM51, but goes OFF subsequently when battery voltage returns to normal.</li> </ul>	S (Error)	M9006	
SM53	AC/DC DOWN detection	OFF: AC/DC DOWN not detected ON : AC/DC DOWN detected	<ul style="list-style-type: none"> <li>Comes ON if a momentary power interruption of less than 20ms occurred during use of the AC power supply module, and reset by turning the power OFF, then ON.</li> </ul>	S (Error)	M9005	QCPU
			<ul style="list-style-type: none"> <li>Comes ON if a momentary power interruption of less than 10ms occurred during use of the DC power supply module, and reset by turning power OFF, then ON.</li> </ul>			QnA
			<ul style="list-style-type: none"> <li>Comes ON if a momentary power interruption of less than 1ms occurred during use of the DC power supply module, and reset by turning power OFF, then ON.</li> </ul>			
SM54	MINI link errors	OFF: Normal ON : Error	<ul style="list-style-type: none"> <li>Goes ON if MINI (S3) link error is detected at even one of the installed AJ71PT32 (S3) modules.</li> <li>Stays ON subsequently even after normal operation is restored.</li> </ul>	S (Error)	M9004	QnA
SM56	Operation Errors	OFF: Normal ON : Operation error	<ul style="list-style-type: none"> <li>ON when operation error is generated</li> <li>Stays ON subsequently even if normal operations restored</li> </ul>	S (Error)	M9011	○
SM60	Blown fuse detection	OFF: Normal ON : Module with blown fuse	<ul style="list-style-type: none"> <li>Comes ON even if there is only one output module with a blown fuse, and remains ON even after return to normal</li> <li>Blown fuse status is checked even for remote I/O station output modules.</li> </ul>	S (Error)	M9000	○+Rem
SM61	I/O module verification error	OFF: Normal ON : Error	<ul style="list-style-type: none"> <li>Comes ON if there is a discrepancy between the actual I/O modules and the registered information when the power is turned on</li> <li>I/O module verification is also conducted for remote I/O station modules.</li> </ul>	S (Error)	M9002	
SM62	Annunciator detection	OFF: Not detected ON : Detected	<ul style="list-style-type: none"> <li>Goes ON if even one annunciator F goes ON.</li> </ul>	S (Instruction execution)	M9009	
SM80	CHK detection	OFF: Not detected ON : Detected	<ul style="list-style-type: none"> <li>Goes ON if error is detected by CHK instruction.</li> <li>Stays ON subsequently even after normal operation is restored.</li> </ul>	S (Instruction execution)	New	
SM90	Startup of watchdog timer for step transition (Enabled only when SFC program exists)	OFF: Not started (watchdog timer reset) ON : Started (watchdog timer started)	Corresponds to SD90	<ul style="list-style-type: none"> <li>Goes ON when measurement of step transition watchdog timer is commenced.</li> <li>Resets watchdog timer when it goes OFF.</li> </ul>	U	M9108
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

Special Relay List

(2) System information

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable 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	○
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	
SM206	PAUSE enable coil	OFF : PAUSE disabled ON : PAUSE enabled	• PAUSE status is entered if this relay is ON when the remote PAUSE contact goes ON	U	M9040	Remote
	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	
SM210	Clock data set request	OFF : Ignored ON : Set request	• When this relay goes from OFF to ON, clock data being stored from SD210 to SD213 after execution of END instruction for changed scan is written to the clock device.	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	Q3A Q4A Q4AR
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	
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
SM240	No. 1 CPU reset flag	OFF : PLC No. 1 reset cancel ON : PLC No. 1 resetting	• Goes OFF when reset of the PLC No. 1 is canceled. • Comes ON when the PLC No. 1 is resetting (including the case where the PLC is removed from the base). The other PLCs are also put in reset status.	S (Status change)	New	QCPU function Ver. B
SM241	No. 2 CPU reset flag	OFF : PLC No. 2 reset cancel ON : PLC No. 2 resetting	• Goes OFF when reset of the PLC No. 2 is canceled. • Comes ON when the PLC No. 2 is resetting (including the case where the PLC is removed from the base). The other PLCs result in "MULTI CPU DOWN" (error code: 7000).			
SM242	No. 3 CPU reset flag	OFF : PLC No. 3 reset cancel ON : PLC No. 3 resetting	• Goes OFF when reset of the PLC No. 3 is canceled. • Comes ON when the PLC No. 3 is resetting (including the case where the PLC is removed from the base). The other PLCs result in "MULTI CPU DOWN" (error code: 7000).			
SM243	No. 4 CPU reset flag	OFF : PLC No. 4 reset cancel ON : PLC No. 4 resetting	• Goes OFF when reset of the PLC No. 4 is canceled. • Comes ON when the PLC No. 4 is resetting (including the case where the PLC is removed from the base). The other PLCs result in "MULTI CPU DOWN" (error code: 7000).			
SM244	No. 1 CPU error flag	OFF : PLC No. 1 normal ON : PLC No. 1 during stop error	• Goes OFF when the PLC No. 1 is normal (including a continuation error). • Comes ON when the PLC No. 1 is during a stop error.			
SM245	No. 2 CPU error flag	OFF : PLC No. 2 normal ON : PLC No. 2 during stop error	• Goes OFF when the PLC No. 2 is normal (including a continuation error). • Comes ON when the PLC No. 2 is during a stop error.			
SM246	No. 3 CPU error flag	OFF : PLC No. 3 normal ON : PLC No. 3 during stop error	• Goes OFF when the PLC No. 3 is normal (including a continuation error). • Comes ON when the PLC No. 3 is during a stop error.			
SM247	No. 4 CPU error flag	OFF : PLC No. 4 normal ON : PLC No. 4 during stop error	• Goes OFF when the PLC No. 4 is normal (including a continuation error). • Comes ON when the PLC No. 4 is during a stop error.			

Special Relay List (Continued)

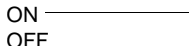
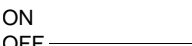
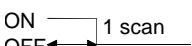
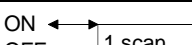

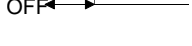







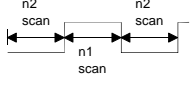
Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM250	Max. loaded I/O read	OFF: Ignored ON : Read	• When this relay goes from OFF to ON, maximum loaded I/O number is read to SD250.	U	New	○+Rem
SM251	I/O change flag	OFF: No replacement ON : Replacement	• After the head I/O number of the I/O module being replaced is set in SD251 is set, on-line I/O module replacement is enabled when this relay is ON. (Only one module can be replaced at each setting.) • To replace an I/O module in the RUN status, use the program or a peripheral device to turn this relay ON; to replace an I/O module in the STOP status, turn this relay ON in the test mode of a peripheral device. • Do not switch between RUN and STOP status until I/O module replacement is completed.	U (END)	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	
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.	U (Every END)	New	QCPU
SM255	MELSECNET/10 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	○
SM256		OFF: Reads ON : Does not read	• For refresh from link to CPU (B, W, etc.) indicate whether to read from the link module.	U	New	
SM257		OFF: Writes ON : Does not write	• For refresh from CPU to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM260	MELSECNET/10 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 (B, W, etc.) indicate whether to read from the link module.	U	New	
SM262		OFF: Writes ON : Does not write	• For refresh from CPU to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM265	MELSECNET/10 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 (B, W, etc.) indicate whether to read from the link module.	U	New	
SM267		OFF: Writes ON : Does not write	• For refresh from CPU to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM270	MELSECNET/10 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 (B, W, etc.) indicate whether to read from the link module.	U	New	
SM272		OFF: Writes ON : Does not write	• For refresh from CPU 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 QJ61QBT11. Goes OFF when normal operation is restored. • Goes ON when a CC-Link error is detected in any of the installed A(1S)J61QBT11. Stays ON even after normal operation is restored.	S (Status change) S (Error)	New New	QCPU Remote QnA
SM320	Presence/absence of SFC program	OFF: SFC program absent ON : SFC program present	• ON if SFC program is correctly registered, and OFF if not registered. • Goes OFF if SFC dedicated instruction is not correct.	S (Initial)	M9100	
SM321	Start/stop SFC program	OFF: SFC program stop ON : SFC program start	• Initial value is set at the same value as SM320. (Goes ON automatically if SFC program is present.) • SFC program will not execute if this goes OFF prior to SFC program processing • Starts SFC program when this relay goes from OFF to ON. • Stops SFC program when this relay goes from ON to OFF.	S (Initial) U	M9101 format change	○

Special Relay List (Continued)

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM322	SFC program start status	OFF: Initial start ON : Restart	<ul style="list-style-type: none"> <li>Initial value is set at ON or OFF depending on parameters.</li> <li>When this relay is OFF, all execution status are cleared from time SFC program was stopped; starts from the initial step of block where the start request was made.</li> <li>When this relay is ON, starts from execution block and execution step active at time SFC program was stopped. (ON is enabled only when resumptive start has been designated at parameters.)</li> <li>SM902 is not automatically designated for latch.</li> </ul>	S (Initial) U	M9102 format change	
SM323	Presence/absence of continuous transition for entire block	OFF: Continuous transition not effective ON : Continuous transition effective	<ul style="list-style-type: none"> <li>When this relay is OFF, transition occurs at one scan/one step, for all blocks.</li> <li>When this relay is ON, transition occurs continuously for all blocks in one scan.</li> <li>In designation of individual blocks, priority is given to the continuous transition bit of the block. (Designation is checked when block starts.)</li> </ul>	U	M9103	
SM324	Continuous transition prevention flag	OFF: When transition is executed ON : When no transition	<ul style="list-style-type: none"> <li>When continuous transition is effective, goes ON when continuous transition is not being executed; goes OFF when continuous transition is being executed.</li> <li>Normally ON when continuous transition is not effective.</li> </ul>	S (Instruction execution)	M9104	○
SM325	Output mode at block stop	OFF: OFF ON : Preserves	When block stops, selects active step operation output. <ul style="list-style-type: none"> <li>All coil outputs go OFF when this relay is OFF.</li> <li>Coil outputs are preserved when this relay is ON.</li> </ul>	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: OFF ON : Preserves	Selects the output action of the step being held when a block is ended by executing the end step. <ul style="list-style-type: none"> <li>All coil outputs go OFF when this relay is OFF.</li> <li>Coil outputs are preserved when this relay is ON.</li> </ul>	S (Initial) U	New	
SM330	Operation mode for low speed execution type program	OFF: Asynchronous mode ON : Synchronous mode	<ul style="list-style-type: none"> <li>Asynchronous mode Mode where the operations for the low speed execution type program are continued during the excess time.</li> <li>Synchronous mode Mode where the operations for the low speed execution type program are started from the next scan even when there is the excess time.</li> </ul>	U (END)	New	
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	QCPU serial No. 04122 or later
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	<ul style="list-style-type: none"> <li>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.)</li> <li>Used by the user in a program as a completion bit.</li> </ul>	S (Status change)	New	QCPU

Special Relay List

(3) System clocks/counters

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU	
SM400	Always ON	ON  OFF	• Normally is ON	S (Every END processing)	M9036	○	
SM401	Always OFF	ON  OFF	• Normally is OFF	S (Every END processing)	M9037		
SM402	ON for 1 scan only after RUN	ON  OFF	• After RUN, ON for 1 scan only. • This connection can be used for scan execution type programs only.	S (Every END processing)	M9038		
SM403	After RUN, OFF for 1 scan only	ON  OFF	• After RUN, OFF for 1 scan only. • This connection can be used for scan execution type programs only.	S (Every END processing)	M9039		
SM404	Low speed execution type program ON for 1 scan only after RUN	ON  OFF	• After RUN, ON for 1 scan only. • This connection can be used for low speed execution type programs only.	S (Every END processing)	New		
SM405	Low speed execution type program After RUN, OFF for 1 scan only	ON  OFF	• 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 PLC 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		QCPU
SM410	0.1 second clock		• Repeatedly changes between ON and OFF at each designated time interval. • When PLC 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)	M9030		○
SM411	0.2 second clock				M9031		
SM412	1 second clock				M9032		
SM413	2 second clock				M9033		
SM414	2n second clock		• Goes between ON and OFF in accordance with the number of seconds designated by SD414. • When PLC 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)	M9034 format change	○	
SM415	2n (ms) clock		• Switches between ON and OFF in accordance with the number of milliseconds designated by SD415. • When PLC 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		QCPU
SM420	User timing clock No.0		• Relay repeats ON/OFF switching at fixed scan intervals. • When PLC power supply is turned OFF or a CPU module reset is performed, goes from OFF to start. • The ON/OFF intervals are set with the DUTY instruction.	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	○	
SM431	User timing clock No.6						
SM432	User timing clock No.7						
SM433	User timing clock No.8						
SM434	User timing clock No.9						

Special Relay List

(4) Scan information

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable 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	○
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	○+Rem

(5) Memory cards

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM600	Memory card A usable flags	OFF: Unusable ON : Use enabled	• ON when memory card A is ready for use by user	S (Initial)	New	○
SM601	Memory card A protect flag	OFF: No protect ON : Protect	• Goes ON when memory card A protect switch is ON	S (Initial)	New	
SM602	Drive 1 flag	OFF: No drive 1 ON : Drive 1 present	• Goes ON when drive 1 (card 1 RAM area) is present	S (Initial)	New	
SM603	Drive 2 flag	OFF: No drive 2 ON : Drive 2 present	• Goes ON when drive 2 (card 1 ROM area) is present	S (Initial)	New	
SM604	Memory card A in-use flag	OFF: Not used ON : In use	• Goes ON when memory card A is in use	S (Initial)	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.	U/S	New	QCPU
SM620	Memory card B usable flags	OFF: Unusable ON : Use enabled	• Always ON • 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	• Always ON • Goes ON when memory card B protect switch is ON	S (Initial)	New	QCPU Q2A (S1) Q3A Q4A Q4AR
SM622	Drive 3 flag	OFF: No drive 3 ON : Drive 3 present	• Always ON • Goes ON when drive 3 (card 2 RAM area) is present	S (Initial)	New	QCPU Q2A (S1) Q3A Q4A Q4AR
SM623	Drive 4 flag	OFF: No drive 4 ON : Drive 4 present	• Always ON • Goes ON when drive 4 (card 2 ROM area) is present	S (Initial)	New	QCPU
SM624	Memory card B in-use flag	OFF: Not used ON : In use	• Goes ON when memory card B is in use	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR
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	
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	○
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	
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	
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	

## Special Relay List (Continued)

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM673	Memory card B file register access range flag	OFF: Within access range ON : Outside access range	<ul style="list-style-type: none"> <li>Goes ON when access is made outside the range of file registers, R. of memory card B. (Set within END processing.)</li> <li>Reset at user program</li> </ul>	S/U	New	Q2A (S1) Q3A Q4A Q4AR

## (6) Instruction-Related Special Relays

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM700	Carry flag	OFF: Carry OFF ON : Carry ON	<ul style="list-style-type: none"> <li>Carry flag used in application instruction</li> </ul>	S (Instruction execution)	M9012	○
SM701	Number of output characters selection	OFF: 16 characters output ON : Outputs until NUL	<ul style="list-style-type: none"> <li>When SM701 is OFF, 16 characters of ASCII code are output.</li> <li>When SM701 is ON, output conducted until NUL (00H) code of ASCII code is encountered.</li> </ul>	U	M9049	
SM702	Search method	OFF: Search next ON : 2-part search	<ul style="list-style-type: none"> <li>Designates method to be used by search instruction.</li> <li>Data must be arranged for 2-part search.</li> </ul>	U	New	
SM703	Sort order	OFF: Ascending order ON : Descending order	<ul style="list-style-type: none"> <li>The sort instruction is used to designate whether data should be sorted in ascending order or in descending order.</li> </ul>	U	New	
SM704	Block comparison	OFF: Non-match found ON : All match	<ul style="list-style-type: none"> <li>Goes ON when all data conditions have been met for the BKCOMP instruction.</li> </ul>	S (Instruction execution)	New	Q4AR
SM707	Selection of real number instruction processing type	OFF: Speed oriented ON : Accuracy oriented	<ul style="list-style-type: none"> <li>When SM707 is OFF, real number instructions are processed at high speed.</li> <li>When it is ON, real number instructions are processed with high accuracy.</li> </ul>	U	New	
SM710	CHK instruction priority ranking flag	OFF: Conditions priority ON : Pattern priority	<ul style="list-style-type: none"> <li>Remains as originally set when OFF.</li> <li>CHK priorities updated when ON.</li> </ul>	S (Instruction execution)	New	○
SM711	Divided transmission status	OFF: Other than during divided processing ON : During divided processing	<ul style="list-style-type: none"> <li>In processing of AD57(S1), goes ON when screen is split for transfer, and goes OFF when split processing is completed</li> </ul>	S (Instruction execution)	M9065	QnA
SM712	Transmission processing selection	OFF: Batch processing ON : Divided processing	<ul style="list-style-type: none"> <li>In processing of AD57(S1), goes ON when canvas screen is divided for transfer.</li> </ul>	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	<ul style="list-style-type: none"> <li>Used to determine whether communications requests to remote terminal modules connected to the AJ71PT32-S3 can be executed or not.</li> </ul>	S (Instruction execution)	M9081	○
SM715	EI flag	OFF: During DI ON : During EI	<ul style="list-style-type: none"> <li>ON when EI instruction is being executed.</li> </ul>	S (Instruction execution)	New	
SM720	Comment read completion flag	OFF: Comment read not completed ON : Comment read completed	<ul style="list-style-type: none"> <li>Switches ON for only one scan when COMRD or PRC instruction is completed.</li> </ul>	S (Status change)	New	QCPU
SM721	File being accessed	OFF: File not accessed ON : File being accessed	<ul style="list-style-type: none"> <li>Switches ON while a file is being accessed by the S.FWRITE, S.FREAD, COMRD, PRC, or LEDC instruction.</li> </ul>	S (Status change)	New	
SM722	BIN/DBIN instruction error disabling flag	OFF: Error detection performed ON : Error detection not performed	<ul style="list-style-type: none"> <li>Turned ON when "OPERATION ERROR" is suppressed for BIN or DBIN instruction.</li> </ul>	U	New	
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	<ul style="list-style-type: none"> <li>Used for determination whether to enable or disable the communication request for the intelligent device station connected with A(1S)J61QBT11.</li> </ul>	S (Instruction execution)	New	QnA

Special Relay List (Continued)

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
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	○
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	
SM774	PID bumpless processing (for complete derivative)	OFF: Forces match ON : Does not force match	• In manual mode, designates whether or not to force the SV value to match the PV value.	U	New	
SM775	Selection of link refresh processing during COM instruction execution	OFF: Performs link refresh ON : Performs no link refresh	• Selects whether only the general data process is performed for the execution of the COM instruction or the link refresh process is also performed.	U	New	QCPU serial number 04012 or later
		OFF: Performs all refresh processes ON : Performs the refresh set the SD778	• Selects whether all refresh process or the refresh set with SD778 is performed when COM instruction is executed.	U	New	
SM776	Enable/disable local device at CALL	OFF: Local device disabled ON : Local device enabled	• Determines whether to enable/disable the local device in the program CALLED at CALL.	U (Status change)	New	○
SM777	Enable/disable local device in interrupt program	OFF: Local device disabled ON : Local device enabled	• Determines whether to enable/disable the local device at the execution of interrupt programs.	U (Status change)	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 (Status change)	New	QnA
SM794	PID bumpless processing (for incomplete derivative)	OFF: Matched ON : Not matched	• Specifies whether the set value (SV) will be matched with the process value (PV) in the manual mode.	U	New	QCPU serial No. 05032 or later

(7) Debug

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM800	Trace preparation	OFF: Not ready ON : Ready	• Switches ON when the trace preparation is completed.	S (Status change)	New	QCPU
	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	QCPU
	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	QCPU
	Sampling trace execution in progress		• Goes ON during execution of sampling trace	S (Status change)	M9046	QnA
SM803	Trace trigger		• Trace is triggered when this relay switches from OFF to ON. (Identical to TRACE instruction execution status)	U	M9044	QCPU
	Sampling trace trigger	OFF → ON: Start	• 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	QCPU
	After sampling trace trigger		• Goes ON after sampling trace trigger	S (Status change)	New	QnA



Special Relay List (Continued)

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM805	Trace completed	OFF: Not completed ON : End	• Switches ON at completion of trace.	S (Status change)	9043	QCPU
	Sampling trace completed		• Goes ON at completion of sampling trace	S (Status change)	9043	QnA
SM806	Status latch preparation	OFF: Not ready ON : Ready	• Goes ON when status latch is ready	S (Status change)	New	QnA
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)	9055	
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	QnA
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 PTR 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	
SM820	Step trace preparation	OFF: Not ready ON : Ready	• Goes ON after program trace registration, at ready.	U	New	○
SM821	Step trace starts	OFF: Suspend ON : Start	• When this goes ON, step trace is started • Suspended when OFF (Related special M all OFF)	S (Status change)	M9182 format 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	QCPU
	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	

(8) Latch area

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM900	Power cut file	OFF: No power cut file ON : Power cut file present	• Goes ON if a file is present during access when power is interrupted.	S/U (Status change)	New	QnA
SM910	RKEY registration flag	OFF: Keyboard input not registered ON : Keyboard input registered	• Goes ON at registration of keyboard input. OFF if keyboard input is not registered.	S (Instruction execution)	New	

(9) 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.

All of these special relays are controlled by the system so that users cannot turn them ON/OFF in the program.

If users want to turn these relays ON/OFF, the program should be modified to use QCPU/QnACPU special relays.

For SM1084 and SM1200 through SM1255, however, if a user can turn ON/OFF some of special relays M9084 and M9200 through M9255 before conversion, the user can also turn ON/OFF the corresponding relays among SM1084 and SM1200 through SM1255 after the 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 QCPU. Uncheck "A-series CPU compatibility setting" within the PC system setting in GX Developer parameters when converted special relays are not used.

**REMARK**



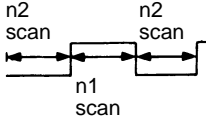
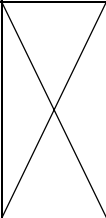
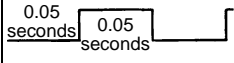
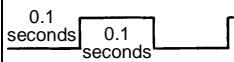
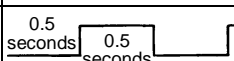
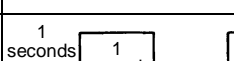
The following are additional explanations about the Special Relay for Modification column.

- 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.

Special Relay List

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU
M9000	SM1000	—	Fuse blown	OFF: Normal ON : Module with blown fuse	<ul style="list-style-type: none"> <li>• Turned on when there is one or more output units of which fuse has been blown.</li> <li>• Remains ON if the condition is restored to normal thereafter.</li> <li>• Output modules of remote I/O stations are also checked fore fuse condition.</li> </ul>	
M9002	SM1002	—	I/O module verification error	OFF: Normal ON : Error	<ul style="list-style-type: none"> <li>• Turned on if the status of I/O module is different form entered status when power is turned on.</li> <li>• Remains ON if the condition is restored to normal thereafter.</li> <li>• I/O module verification is done also to remote I/O station modules.</li> <li>• Reset is enabled only when special registers SD1116 to SD1123 are reset.</li> </ul>	○
M9004	SM1004	—	NIMI link error	OFF: Normal ON : Error	<ul style="list-style-type: none"> <li>• Goes ON if MINI (S3) link error is detected at even one of the installed AJ71PT32 (S3) modules.</li> <li>• Remains ON if the condition is restored to normal thereafter.</li> </ul>	QnA
M9005	SM1005	—	AC DOWN detection	OFF: AC DOWN not detected ON : AC DOWN detected	<ul style="list-style-type: none"> <li>• Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module.</li> <li>• Reset when power is switched OFF, then ON.</li> </ul>	○
					<ul style="list-style-type: none"> <li>• Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module.</li> <li>• Reset when power is switched OFF, then ON.</li> </ul>	
					<ul style="list-style-type: none"> <li>• Turns ON if an instantaneous power failure of within 1ms occurs during use of the DC power supply module.</li> <li>• Reset when power is switched OFF, then ON.</li> </ul>	

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU
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.	
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 <b>[OUT] F</b> or <b>[SET] F</b> instruction is executed. • Switched off when SD1124 data is zeroed.	
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 cleared	• 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.	○
M9017	SM1017		Data memory clear flag	OFF: Ignored ON : Output cleared	• Clears the unlatched data memory (other than special relays and special registers) in remote run mode from computer, etc. when SM1017 is on.	
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 performed, the clock starts with off. • Set the intervals of on/off by <b>[DUTY]</b> instruction.	
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 clock data from SD1025 to SD1028 to the clock element after the <b>[END]</b> instruction is executed during the scan in which SM1025 has changed 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.	
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 PLC 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			

\*: 1 minute clock indicates the name of the special relay (M9034) of the ACPU.

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU
M9034	SM1034	—	2n minute clock (1 minute clock) *		<ul style="list-style-type: none"> <li>Alternates between ON and OFF according to the seconds specified at SD414. (Default: n = 30)</li> <li>Not turned on or off per scan but turned on and off even during scan if corresponding time has elapsed.</li> <li>Starts with off when PLC power supply is turned on or CPU module reset is performed..</li> </ul>	
M9036	SM1036	—	Always ON	ON	<ul style="list-style-type: none"> <li>Used as dummy contacts of initialization and application instruction in sequence program.</li> </ul>	
M9037	SM1037	—	Always OFF	OFF	<ul style="list-style-type: none"> <li>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.</li> </ul>	
M9038	SM1038	—	ON for 1 scan only after RUN	OFF		
M9039	SM1039	—	RUN flag(After RUN, OFF for 1 scan only)	ON		
M9040	SM1040	SM206	PAUSE enable coil	OFF: PAUSE disabled ON : PAUSE enabled	<ul style="list-style-type: none"> <li>When RUN key switch is at PAUSE position or remote pause contact has turned on and if SM204 is on, PAUSE mode is set and SM206 is turned on.</li> </ul>	
M9041	SM1041	SM204	USE 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	<ul style="list-style-type: none"> <li>Switched on when the RUN key switch or RUN/STOP switch is in STOP position.</li> </ul>	
M9043	SM1043	SM805	Sampling trace completed	OFF: Sampling trace in progress ON : Sampling trace completed	<ul style="list-style-type: none"> <li>Turned on upon completion of sampling trace performed the number of times preset by parameter after [STRA] instruction is executed. Reset when [STRAR] instruction is executed.</li> </ul>	
M9044	SM1044	SM803	Sampling trace	OFF → ON [STRA] Same as execution ON → OFF [STRAR] Same as execution	<ul style="list-style-type: none"> <li>Turning on/off SM803 can execute [STRA] / [STRAR] instruction. (SM803 is forcibly turned on/off by a peripheral device.)</li> <li>When switched from OFF to ON: [STRA] instruction</li> <li>When switched from ON to OFF: [STRAR] instruction</li> <li>The value stored in SD1044 is used as the condition for the sampling trace.</li> <li>At scanning, at time → Time (10 ms unit)</li> </ul>	○
M9045	SM1045		Watchdog timer (WDT) reset	OFF: Does not reset WDT ON : Resets WDT	<ul style="list-style-type: none"> <li>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).</li> </ul>	
M9046	SM1046	SM802	Sampling trace	OFF: Trace not in progress ON : Trace in progress	<ul style="list-style-type: none"> <li>Switched on during sampling trace.</li> </ul>	
M9047	SM1047	SM801	Sampling trace preparations	OFF: Sampling trace suspended ON : Sampling trace started	<ul style="list-style-type: none"> <li>Sampling trace is not executed unless SM801 is turned ON.</li> <li>Sampling trace is suspended when SM801 goes OFF.</li> </ul>	
M9049	SM1049	SM701	Selection of number of characters output	OFF: Output until NULL code encountered ON : 16 characters output	<ul style="list-style-type: none"> <li>When SM701 is OFF, characters up to NULL (00H) code are output.</li> <li>When SM701 is ON, ASCII codes of 16 characters are output.</li> </ul>	
M9051	SM1051		CHG instruction execution disable	OFF: Enabled ON : Disable	<ul style="list-style-type: none"> <li>Switched ON to disable the CHG instruction.</li> <li>Switched ON when program transfer is requested. Automatically switched OFF when transfer is complete.</li> </ul>	
M9052	SM1052		SEG instruction switch	OFF: 7SEG segment display ON : I/O partial refresh	<ul style="list-style-type: none"> <li>When SM1052 is ON, the SEG instruction is executed as an I/O partial refresh instruction.</li> <li>When SM1052 is OFF, the SEG instruction is executed as a 7-SEG display instruction.</li> </ul>	
M9054	SM1054	SM205	STEP RUN flag	OFF: STEP RUN not in effect ON : STEP RUN in effect	<ul style="list-style-type: none"> <li>Switched on when the RUN key switch is in STEP RUN position.</li> </ul>	QnA
M9055	SM1055	SM808	Status latch completion flag	OFF: Not completed ON : Completed	<ul style="list-style-type: none"> <li>Turned on when status latch is completed.</li> <li>Turned off by reset instruction.</li> </ul>	
M9056	SM1056		Main side P, I set request	OFF: Other than when P, I set being requested ON : P, I set being requested	<ul style="list-style-type: none"> <li>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.</li> </ul>	○
M9057	SM1057		Sub side P, I set request	OFF: Other than when P, I set being requested ON : P, I set being requested		

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU
M9058	SM1058	<del>SM1058</del>	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	<del>SM1059</del>	Sub program P, I set completion	Momentarily ON at P, I set completion		
M9060	SM1060	<del>SM1060</del>	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	<del>SM1061</del>	Sub program 3 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.	
M9065	SM1065	SM711	Divided processing execution detection	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	Divided processing request flag	OFF: Batch processing ON : Divided processing	• Turned on when canvas screen transfer to AD57(S1)/AD58 is done by divided processing.	
M9070	SM1070	<del>SM1070</del>	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 %.) * The A8UPU/A8PUJ cannot be used in the QCPU/QnACPU special relays.	○
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	<del>SM1084</del>	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 breakage of fuse. • Check of battery • Collation check of I/O module	○
M9091	SM1091	<del>SM1091</del>	Instruction error 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 change flag	OFF: Replacement ON : No replacement	• The I/O module can be changed online when SM251 is turned ON after the head I/O number of the I/O module to be changed is set to SD251. (One module is only 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, and turned off if it is not.	
M9101	SM1101	SM321	Start/stop SFC program	OFF: SFC programs stop ON : SFC programs start	• Should be turned on by the program if the SFC program is to be started. If turned off, operation output of the execution step is turned off and the SFC program is stopped.	
M9102	SM1102	SM322	SFC program start status	OFF: Initial Start ON : Continue	• Selects the starting step when the SFC program is restarted using SM322. ON: All execution conditions when the SFC program stopped are cleared, and the program is started with the initial step of block 0. OFF: Started with the step of the block being executed when the program stopped. • Once turned on, the program is latched in the system and remains on even if the power is turned off. Should be turned off by the sequence program when turning on the power, or when starting with the initial step of block 0.	○

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU	
M9103	SM1103	SM323	Presence/absence of continuous transition	OFF: Continuous transition not effective ON : Continuous transition effective	<ul style="list-style-type: none"> <li>Selects consecutive or step-by-step transfer of steps of which transfer conditions are established when all of the transfer conditions of consecutive steps are established.</li> <li>ON: Consecutive transfer is executed.</li> <li>OFF: One step per one scan is transferred.</li> </ul>	○	
M9104	SM1104	SM324	Continuous transition suspension flag	OFF: When transition is completed ON : When no transition	<ul style="list-style-type: none"> <li>OFF during operation in the continuous transition mode or during continuous transition, and ON when continuous transition is not executed.</li> <li>Always ON during operation in the no continuous transition mode.</li> </ul>		
M9108	SM1108	SM90	Step transition watchdog timer start (equivalent of D9108)	OFF: Watchdog timer reset ON : Watchdog timer reset start	<ul style="list-style-type: none"> <li>Turns ON when the measurement of the step transition watchdog timer is started.</li> <li>Turning this relay OFF resets the step transition watchdog timer.</li> </ul>		
M9109	SM1109	SM91	Step transition watchdog timer start (equivalent of D9109)				
M9110	SM1110	SM92	Step transition watchdog timer start (equivalent of D9110)				
M9111	SM1111	SM93	Step transition watchdog timer start (equivalent of D9111)				
M9112	SM1112	SM94	Step transition watchdog timer start (equivalent of D9112)				
M9113	SM1113	SM95	Step transition watchdog timer start (equivalent of D9113)				
M9114	SM1114	SM96	Step transition watchdog timer start (equivalent of D9114)				
M9180	SM1180	SM825	Active step sampling trace completion flag	OFF: Trace started ON : Trace completed	<ul style="list-style-type: none"> <li>Set when sampling trace of all specified blocks is completed. Reset when sampling trace is started.</li> </ul>		
M9181	SM1181	SM822	Active step sampling trace execution flag	OFF: Trace not being executed ON : Trace execution under way	<ul style="list-style-type: none"> <li>Set when sampling trace is being executed. Reset when sampling trace is completed or suspended.</li> </ul>		
M9182	SM1182	SM821	Active step sampling trace permission	OFF: Trace disable/suspend ON : Trace enable	<ul style="list-style-type: none"> <li>Selects sampling trace execution enable/disable.</li> <li>ON: Sampling trace execution is enabled.</li> <li>OFF: Sampling trace execution is disabled.</li> <li>If turned off during sampling trace execution, trace is suspended.</li> </ul>		
M9196	SM1196	SM325	Operation output at block stop	OFF: Coil output OFF ON : Coil output ON	<ul style="list-style-type: none"> <li>Selects the operation output when block stop is executed.</li> <li>ON: Retains the ON/OFF status of the coil being used by using operation output of the step being executed at block stop.</li> <li>OFF: All coil outputs are turned off. (Operation output by the SET instruction is retained regardless of the ON/OFF status of M9196.)</li> </ul>		
M9197	SM1197	X	Switch between blown fuse and I/O verification error display	SM9197	SM1198		I/O numbers to be displayed
				OFF	OFF	X/Y 0 to 7F0	
ON	OFF	X/Y 800 to FF0					
M9198	SM1198	X		OFF	ON	X/Y 1000 to 17F0	
			ON	ON	X/Y 1800 to 1FF0		

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU
M9199	SM1199	X	Data recovery of online sampling trace/status latch	OFF: Data recovery disabled ON : Data recovery enabled	<ul style="list-style-type: none"> <li>Recovers the setting data stored in the CPU module at restart when sampling trace/status latch is executed.</li> <li>SM1199 should be ON to execute again. (Unnecessary when writing the data again from peripheral devices.)</li> </ul>	○
M9200	SM1200	—	<b>ZNRD</b> instruction (LRDP instruction for ACPUs) reception (for master station)	OFF: Not accepted ON : Accepted	<ul style="list-style-type: none"> <li>Depends on whether or not the <b>ZNRD</b> (word device read) instruction has been received.</li> <li>Used in the program as an interlock for the <b>ZNRD</b> instruction.</li> <li>Use the <b>RST</b> instruction to reset.</li> </ul>	QnA
M9201	SM1201	—	<b>ZNRD</b> instruction (LRDP instruction for ACPUs) completion (for master station)	OFF: Not completed ON : End	<ul style="list-style-type: none"> <li>Depends on whether or not the <b>ZNRD</b> (word device read) instruction execution is complete.</li> <li>Used as a condition contact for resetting M9200 and M9201 after the <b>ZNRD</b> instruction is complete.</li> <li>Use the <b>RST</b> instruction to reset.</li> </ul>	
M9202	SM1202	—	<b>ZNWR</b> instruction (LWTP instruction for ACPUs) reception (for master station)	OFF: Not accepted ON : Accepted	<ul style="list-style-type: none"> <li>Depends on whether or not the <b>ZNWR</b> (word device write) instruction has been received.</li> <li>Used in the program as an interlock for the <b>ZNWR</b> instruction.</li> <li>Use the <b>RST</b> instruction to reset.</li> </ul>	
M9203	SM1203	—	<b>ZNWR</b> instruction (LWTP instruction for ACPUs) completion (for master station)	OFF: Not completed ON : End	<ul style="list-style-type: none"> <li>Depends on whether or not the <b>ZNWR</b> (word device write) instruction execution is complete.</li> <li>Used as a condition contact to reset M9202 and M9203 after the <b>ZNWR</b> instruction is complete.</li> <li>Use the <b>RST</b> instruction to reset.</li> </ul>	
M9204	SM1204	—	<b>ZNRD</b> instruction (LRDP instruction for ACPUs) reception (for local station)	OFF: Not completed ON : End	On indicates that the <b>ZNRD</b> instruction is complete at the local station.	
M9205	SM1205	—	<b>ZNWR</b> instruction (LWTP instruction for ACPUs) reception (for local station)	OFF: Not completed ON : End	On indicates that the <b>ZNWR</b> 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: YES ON : NO	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	<ul style="list-style-type: none"> <li>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).</li> <li>When SM1208 is OFF.....B and W of host station is sent to tertiary stations.</li> <li>When SM1208 is ON.....B and W of host station is not sent to tertiary stations.</li> </ul>	
M9209	SM1209	—	Link parameter check command (for lower link master stations only)	OFF: Executing the check function ON : Check non-execution	<ul style="list-style-type: none"> <li>Set to ON not to match B and W of the higher and lower links.</li> <li>When SM1209 is ON, the link parameters of the higher and lower link are not checked.</li> <li>When SM1209 is OFF, the link parameters of the higher and lower link are checked.</li> </ul>	
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.	

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Applicable CPU
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.	QnA
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.	
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	
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-station test, 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	Depends 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.	



Special Relay List

(10) For redundant systems (Host system CPU information \*1) for Q4AR only  
 SM1510 to SM1599 are only valid for redundant systems. All off for standalone systems.

Number	Name	Meaning	Explanation	Set by (When Set)	ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable 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
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	
SM1510	Operation mode	OFF: Redundant system backup mode, independent system ON : Redundant system separate mode	• Turns on when the operating mode is redundant system separate.	S (Each END)	New	
SM1511	Start mode when power supply is on	OFF: System A fixed mode ON : Previous control system latch mode	• Turns on when the start mode for a redundant system when the power is turned on is the previous control system latch mode.	S (Initial)	New	
SM1512	Start mode when CPU is started	OFF: Initial start ON : Hot start	• Turns on when the CPU module operation mode is hot start when the redundant system is started up.	S (Initial)	New	
SM1513	Operation status when CPU is started	OFF: Initial start ON : Hot start	• Turns on when the CPU module operation mode is hot start when the redundant system is actually start up.	S (Initial)	New	
SM1514	Operation mode when CPU is switched	OFF: Initial start ON : Hot start	• Turns on when the operation is hot start when the CPU module operation is switched for a redundant system.	S (Initial)	New	
SM1515	Output hold mode	OFF: Output reset ON : Output hold	• Turns on when the output mode during a stop error is output hold.	S (Each END)	New	
SM1516	Operation system status	OFF: Control system ON : Standby system	• Turns on when the CPU module operation system status is the standby system.	S (Status change)	New	
SM1517	CPU startup status	OFF: Power supply on startup ON : Operation system switch startup	• Turns on when the CPU module is started up by the operation system switch. • Reset using the user program.	S (Status change) /U	New	
SM1518	Tracking execution mode	OFF: Batch operation 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	
SM1520	Data tracking transmission link specification	OFF: No trigger ON : Trigger	SM1520 Block 1	U	New	
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					

Special Relay List (Continued)

Number	Name	Meaning	Explanation		Set by (When Set)	ACPU	Applicable CPU	
						M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>		
SM1546	Data tracking transmission link specification	OFF: No trigger ON : Trigger	SM1546	Block 27	• Specified the blocks to trigger when the data is transmitted by the data tracking instruction S. TRUCK.	U	New	Q4AR
SM1547			SM1547	Block 28				
SM1548			SM1548	Block 29				
SM1549			SM1549	Block 30				
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 occurs)			

Special Relay List

(11) For redundant system (Other system CPU information \*1) for Q4AR only  
 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 standalone system.

Number	Name	Meaning	Explanation	Set by (When Set)	ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> *2	Applicable 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)	New	Q4AR
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)	New	
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)	New	
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)	New	
SM1653	STOP contact	STOP status	• Turns on when in the STOP status.	S (Each END)	New	
SM1654	PAUSE contact	PAUSE status	• Turns on when in the PAUSE status.	S (Each END)	New	
SM1655	STEP-RUN contact	STEP-RUN status	• Turns on when in the STEP-RUN status.	S (Each END)	New	

\* 1 Stores other system CPU diagnostic information and system information.  
 \* 2 This shows the special relay(SM ) for the host system CPU.

(12) For redundant system (tracking) for Q4AR only

Either the backup mode or the second mode is valid for SM1700 to SM1799. All is turned off for standalone system.

Number	Name	Meaning	Explanation	Set by (When Set)	ACPU M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Applicable CPU
SM1700	Tracking execution flag	OFF: Execution not possible ON : Execution possible	• Turns on when tracking is executed normally.	S (status change)	New	Q4AR
SM1712	Transmission trigger end flag	OFF: Transmission uncompleted ON : Transmission end	SM1712 Block 1	S (status change)	New	
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			

Special Relay List (Continued)

Number	Name	Meaning	Explanation		Set by (When Set)	ACPU	Applicable CPU	
						M9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>		
SM1734	Transmission trigger end flag	OFF: Transmission uncompleted ON : Transmission end	SM1734	Block 23	• One scan turns on when the corresponding data transmission has been completed.	S (status change)	New	Q4AR
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				
SM1760			SM1760	Block 49				
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				

## APPENDIX 2 Special Register

The special registers, SD, are internal registers with fixed applications in the PLC.

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 headings in the table that follows have the following meanings.

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)	<ul style="list-style-type: none"> <li>• Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed.</li> <li>&lt;Set by&gt; <ul style="list-style-type: none"> <li>S : Set by system</li> <li>U : Set by user (sequence programs or test operations from GX Developer)</li> <li>S/U : Set by both system and user</li> </ul> </li> <li>&lt;When set&gt; → Indicated only for registers set by system <ul style="list-style-type: none"> <li>Each END : Set during each END processing</li> <li>Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN)</li> <li>Status change : Set only when there is a change in status</li> <li>Error : Set when error occurs</li> <li>Instruction execution : Set when instruction is executed</li> <li>Request : Set only when there is a user request (through SM, etc.)</li> </ul> </li> </ul>
Corresponding ACPU M9 □ □ □	<ul style="list-style-type: none"> <li>• Indicates corresponding special register in ACPU (D9 □ □ □)(Change and notation when there has been a change in contents)</li> <li>• "New" indicates the item that has been newly added to the High Performance model QCPU/QnACPU.</li> </ul>
Corresponding CPU	<ul style="list-style-type: none"> <li>• Indicates the corresponding CPU module type name.</li> <li>○+Rem: Can be applied to all CPU types and MELSECNET/H remote I/O modules.</li> <li>○: Can be applied to all types of CPU module</li> <li>QCPU: Can be applied to High Performance model QCPU.</li> <li>QnA: Can be applied to QnA series and Q2ASCPU series</li> <li>Remote: Can be applied to the MELSECNET/H remote I/O modules.</li> <li>Each CPU type name: Can be applied only to the specific CPU. (e.g. Q4ARCPU, Q3ACPU)</li> </ul>

For details on the following items, refer to the following manuals:

- Networks → • For Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
  - For Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)
  - For QnA/Q4AR MELSECNET/10 Network System Reference Manual
- SFC → QCPU(Q mode)/QnACPU Programming Manual (SFC)

POINT
(1) SD1200 to SD1255 are used for QnACPU. These relays are vacant with High Performance model QCPU.
(2) Special register SD1500 and later are dedicated for Q4ARCPU.

Special Register List

(1) Diagnostic Information

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU		
SD0	Diagnostic errors	Diagnosis error code	<ul style="list-style-type: none"> <li>• Error codes for errors found by diagnosis are stored as BIN data.</li> <li>• Contents identical to latest fault history information.</li> </ul>	S (Error)	D9008 format change			
SD1	Clock time for diagnosis error occurrence	Clock time for diagnosis error occurrence	<ul style="list-style-type: none"> <li>• Year (last two digits) and month that SD0 data was updated is stored as BCD 2-digit code.</li> </ul> <p>(Example)</p> <p>b15 to b8 b7 to b0 : October, 1995</p> <p>Year (0 to 99)   Month (1 to 12) : 9510H</p>	S (Error)	New			
SD2			<ul style="list-style-type: none"> <li>• The day and hour that SD0 was updated is stored as BCD 2-digit code.</li> </ul> <p>(Example)</p> <p>b15 to b8 b7 to b0 : 10 p.m. on 25th</p> <p>Day (1 to 31)   Hour (0 to 23) : 2510H</p>					
SD3			<ul style="list-style-type: none"> <li>• The minute and second that SD0 data was updated is stored as BCD 2-digit code.</li> </ul> <p>(Example)</p> <p>b15 to b8 b7 to b0 : 35 min. 48 sec. (past the hour)</p> <p>Minutes (0 to 59)   Seconds (0 to 59) : 3548H</p>					
SD4	Error information categories	Error information category code	<ul style="list-style-type: none"> <li>• 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.</li> </ul> <p>b15 to b8 b7 to b0</p> <table border="1" style="margin-left: 40px;"> <tr> <td style="width: 50px;">Individual information category codes</td> <td style="width: 50px;">Common information category codes</td> </tr> </table> <ul style="list-style-type: none"> <li>• The common information category codes store the following codes:                             <ul style="list-style-type: none"> <li>0 : No error</li> <li>1 : Unit/module No./ PLC No./Base No. *</li> <li>2 : File name/Drive name</li> <li>3 : Time (value set)</li> <li>4 : Program error location</li> <li>5 : Switch cause (for Q4AR only)                                     <ul style="list-style-type: none"> <li>* : For a multiple PLC system, the module number or PLC number is stored depending on the error that occurred. (Refer to the corresponding error code for which number has been stored.)</li> <li>PLC No. 1: 1, PLC No. 2: 2, PLC No. 3: 3, PLC No. 4: 4</li> </ul> </li> </ul> </li> <li>• The individual information category codes store the following codes:                             <ul style="list-style-type: none"> <li>0 : No error</li> <li>1 : (Open)</li> <li>2 : File name/Drive name</li> <li>3 : Time (value actually measured)</li> <li>4 : Program error location</li> <li>5 : Parameter number</li> <li>6 : Annunciator number</li> <li>7 : Check instruction malfunction number</li> </ul> </li> </ul>	Individual information category codes	Common information category codes	S (Error)	New	○+Rem
Individual information category codes	Common information category codes							

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU																																											
SD5 SD6 SD7 SD8 SD9 SD10 SD11 SD12 SD13 SD14 SD15	Error common information	Error common information	<ul style="list-style-type: none"> <li>Common information corresponding to the error codes (SD0) is stored here.</li> <li>The following four types of information are stored here:                             <ul style="list-style-type: none"> <li>① Slot No.                                     <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Slot No./PLC No./Base No. *1 *2</td> </tr> <tr> <td>SD6</td> <td>I/O No. (Not used for base No.)</td> </tr> <tr> <td>SD7</td> <td rowspan="8">(Vacant)</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> </tbody> </table> </li> <li>*1: For a multiple CPU system, 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</li> <li>*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.</li> <li>② File name/Drive name                                     <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Number</th> <th>Meaning</th> <th>(Example)</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Drive</td> <td>File name= ABCDEFGH. IJK b15 to b8 b7 to b0</td> </tr> <tr> <td>SD6</td> <td rowspan="4">File name (ASCII code: 8 characters)</td> <td>42H(B) 41H(A)</td> </tr> <tr> <td>SD7</td> <td>44H(D) 43H(C)</td> </tr> <tr> <td>SD8</td> <td>46H(F) 45H(E)</td> </tr> <tr> <td>SD9</td> <td>48H(H) 47H(G)</td> </tr> <tr> <td>SD10</td> <td>Extension*3 2EH(.)</td> <td>49H(I) 2EH(.)</td> </tr> <tr> <td>SD11</td> <td>(ASCII code: 3 characters)</td> <td>4BH(K) 4AH(J)</td> </tr> <tr> <td>SD12</td> <td rowspan="4">(Vacant)</td> <td></td> </tr> <tr><td>SD13</td></tr> <tr><td>SD14</td></tr> <tr><td>SD15</td></tr> </tbody> </table> </li> </ul> </li> </ul>	Number	Meaning	SD5	Slot No./PLC No./Base No. *1 *2	SD6	I/O No. (Not used for base No.)	SD7	(Vacant)	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	Number	Meaning	(Example)	SD5	Drive	File name= ABCDEFGH. IJK b15 to b8 b7 to b0	SD6	File name (ASCII code: 8 characters)	42H(B) 41H(A)	SD7	44H(D) 43H(C)	SD8	46H(F) 45H(E)	SD9	48H(H) 47H(G)	SD10	Extension*3 2EH(.)	49H(I) 2EH(.)	SD11	(ASCII code: 3 characters)	4BH(K) 4AH(J)	SD12	(Vacant)		SD13	SD14	SD15	S (Error)	New	○+Rem
Number			Meaning																																														
SD5			Slot No./PLC No./Base No. *1 *2																																														
SD6			I/O No. (Not used for base No.)																																														
SD7			(Vacant)																																														
SD8																																																	
SD9																																																	
SD10																																																	
SD11																																																	
SD12																																																	
SD13																																																	
SD14																																																	
SD15																																																	
Number			Meaning	(Example)																																													
SD5			Drive	File name= ABCDEFGH. IJK b15 to b8 b7 to b0																																													
SD6	File name (ASCII code: 8 characters)	42H(B) 41H(A)																																															
SD7		44H(D) 43H(C)																																															
SD8		46H(F) 45H(E)																																															
SD9		48H(H) 47H(G)																																															
SD10	Extension*3 2EH(.)	49H(I) 2EH(.)																																															
SD11	(ASCII code: 3 characters)	4BH(K) 4AH(J)																																															
SD12	(Vacant)																																																
SD13																																																	
SD14																																																	
SD15																																																	

\* 3: Refer to REMARK.

**REMARK**

1) Extensions are shown below.

SD10 Higher8 bits	SD11		Extension name	File type
	Lower8 bits	Higher8 bits		
51H	50H	41H	QPA	Parameters
51H	50H	47H	QPG	Sequence program/SFC program
51H	43H	44H	QCD	Device comment
51H	44H	49H	QDI	Device initial value
51H	44H	52H	QDR	File register
51H	44H	53H	QDS	Simulation data
51H	44H	4CH	QDL	Local device
51H	54H	53H	QTS	Sampling trace data (For QnA)
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
51H	46H	44H	QFD	Trouble history data

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9	Corresponding CPU																																												
SD5	Error common information	Error common information	<p>③ Time (value set)</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Time : 1 μs units (0 to 999 μs)</td> </tr> <tr> <td>SD6</td> <td>Time : 1 ms units (0 to 65535 ms)</td> </tr> <tr> <td>SD7</td> <td rowspan="9">(Vacant)</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> </tbody> </table>	Number	Meaning	SD5	Time : 1 μs units (0 to 999 μs)	SD6	Time : 1 ms units (0 to 65535 ms)	SD7	(Vacant)	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	S (Error)	New	○+Rem																												
Number			Meaning																																															
SD5			Time : 1 μs units (0 to 999 μs)																																															
SD6			Time : 1 ms units (0 to 65535 ms)																																															
SD7			(Vacant)																																															
SD8																																																		
SD9																																																		
SD10																																																		
SD11																																																		
SD12																																																		
SD13																																																		
SD14																																																		
SD15																																																		
SD6																																																		
SD7																																																		
SD8																																																		
SD9																																																		
SD10																																																		
SD11																																																		
SD12																																																		
SD13																																																		
SD14																																																		
SD15	<p>④ Program error location</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td 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</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 *4</td> </tr> <tr> <td>SD12</td> <td colspan="2">Block No.</td> </tr> <tr> <td>SD13</td> <td colspan="2">Step No./transition No.</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> </tbody> </table> <p>* 4 Contents of pattern data</p> <table border="1"> <tr> <td>15</td><td>14</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td>← (Bit number)</td> </tr> <tr> <td>0</td><td>0</td><td>to</td><td>0</td><td>0</td><td>*</td><td>*</td><td>*</td><td></td> </tr> </table> <p>(Not used)</p> <ul style="list-style-type: none"> <li>SFC block designation present (1)/absent (0)</li> <li>SFC step designation present (1)/absent (0)</li> <li>SFC transition designation present (1)/absent (0)</li> </ul>	Number	Meaning	SD5	File name (ASCII code: 8 characters)	SD6	SD7	SD8	SD9	Extension	2EH(.)	SD10	(ASCII code: 3 characters)		SD11	Pattern *4		SD12	Block No.		SD13	Step No./transition No.		SD14	Sequence step No. (L)		SD15	Sequence step No. (H)		15	14	to	4	3	2	1	0	← (Bit number)	0	0	to	0	0	*	*	*		S (Error)	New	Q4AR
Number	Meaning																																																	
SD5	File name (ASCII code: 8 characters)																																																	
SD6																																																		
SD7																																																		
SD8																																																		
SD9	Extension	2EH(.)																																																
SD10	(ASCII code: 3 characters)																																																	
SD11	Pattern *4																																																	
SD12	Block No.																																																	
SD13	Step No./transition No.																																																	
SD14	Sequence step No. (L)																																																	
SD15	Sequence step No. (H)																																																	
15	14	to	4	3	2	1	0	← (Bit number)																																										
0	0	to	0	0	*	*	*																																											
SD5	<p>⑤ Switch cause</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Switch cause (0: automatic switch/ 1: manual switch)</td> </tr> <tr> <td>SD6</td> <td>Switch direction (0:standby system to control system/ 1: control system to standby system)</td> </tr> <tr> <td>SD7</td> <td rowspan="5">Tracking flag *5</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> <td rowspan="4">(Vacant)</td> </tr> <tr> <td>SD13</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table> <p>* 5 Tracking flag contents</p> <p>Shows whether or not the tracking data is valid.</p> <table border="1"> <tr> <td>15</td><td>14</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td>← (Bit number)</td> </tr> <tr> <td>0</td><td>0</td><td>to</td><td>0</td><td>0</td><td>*</td><td>*</td><td>*</td><td></td> </tr> </table> <p>(Not used)</p> <ul style="list-style-type: none"> <li>Invalid work data invalid (0) /valid (1)</li> <li>System data (SFC active step information) invalid (0)/ valid (1)</li> <li>Switching cause invalid (0)/ valid (1)</li> </ul>	Number	Meaning	SD5	Switch cause (0: automatic switch/ 1: manual switch)	SD6	Switch direction (0:standby system to control system/ 1: control system to standby system)	SD7	Tracking flag *5	SD8	SD9	SD10	SD11	SD12	(Vacant)	SD13	SD14	SD15	15	14	to	4	3	2	1	0	← (Bit number)	0	0	to	0	0	*	*	*		S (Error)	New	Q4AR											
Number	Meaning																																																	
SD5	Switch cause (0: automatic switch/ 1: manual switch)																																																	
SD6	Switch direction (0:standby system to control system/ 1: control system to standby system)																																																	
SD7	Tracking flag *5																																																	
SD8																																																		
SD9																																																		
SD10																																																		
SD11																																																		
SD12	(Vacant)																																																	
SD13																																																		
SD14																																																		
SD15																																																		
15	14	to	4	3	2	1	0	← (Bit number)																																										
0	0	to	0	0	*	*	*																																											



Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU																														
SD16	Error individual information	Error individual information	<ul style="list-style-type: none"> <li>Individual information corresponding to error codes (SD0) is stored here.</li> <li>① File name/Drive name                             <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td>Drive</td> </tr> <tr> <td>SD17</td> <td 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</td> </tr> <tr> <td>SD22</td> <td>(ASCII code: 3 characters)</td> </tr> <tr> <td>SD23</td> <td rowspan="4">(Vacant)</td> </tr> <tr> <td>SD24</td> </tr> <tr> <td>SD25</td> </tr> <tr> <td>SD26</td> </tr> </tbody> </table> <div style="margin-left: 20px;">                     (Example)                      File name= ABCDEFGH. IJK                      b15 to b8 b7 to b0  <table border="1" style="margin-left: 20px;"> <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> </div> </li></ul>	Number	Meaning	SD16	Drive	SD17	File name (ASCII code: 8 characters)	SD18	SD19	SD20	SD21	Extension	SD22	(ASCII code: 3 characters)	SD23	(Vacant)	SD24	SD25	SD26	42H(B)	41H(A)	44H(D)	43H(C)	46H(F)	45H(E)	48H(H)	47H(G)	49H(I)	2EH(.)	4BH(K)	4AH(J)	S (Error)	New	○+Rem
Number			Meaning																																	
SD16			Drive																																	
SD17			File name (ASCII code: 8 characters)																																	
SD18																																				
SD19																																				
SD20																																				
SD21			Extension																																	
SD22			(ASCII code: 3 characters)																																	
SD23			(Vacant)																																	
SD24																																				
SD25																																				
SD26																																				
42H(B)			41H(A)																																	
44H(D)			43H(C)																																	
46H(F)			45H(E)																																	
48H(H)			47H(G)																																	
49H(I)			2EH(.)																																	
4BH(K)			4AH(J)																																	
SD17																																				
SD18																																				
SD19																																				
SD20																																				
SD21																																				
SD22																																				
SD23																																				
SD24																																				
SD25																																				
SD26																																				
SD16	Time (value actually measured)																																			
SD17	Time : 1 μs units (0 to 999 μs)																																			
SD18	Time : 1 ms units (0 to 65535 ms)																																			
SD19	(Vacant)																																			
SD20																																				
SD21																																				
SD22																																				
SD23																																				
SD24																																				
SD25																																				
SD26																																				
SD16	Program error location																																			
SD17	File name (ASCII code: 8 characters)																																			
SD18	Extension																																			
SD19																																				
SD20	2EH(.)																																			
SD21	(ASCII code: 3 characters)																																			
SD22	Pattern*																																			
SD23	Block No.																																			
SD24	Step No./transition No.																																			
SD25	Sequence step No. (L)																																			
SD26	Sequence step No. (H)																																			
SD16	* Contents of pattern data																																			
SD17	15 14 to 4 3 2 1 0 ← (Bit number)																																			
SD18	0 0 to 0 0 * * *																																			
SD19	(Not used)																																			
SD20	SFC block designation present (1)/absent (0)																																			
SD21	SFC step designation present (1)/absent (0)																																			
SD22	SFC transition designation present (1)/absent (0)																																			
SD16	④ Parameter number																																			
SD17	⑤ Annunciator number / CHK instruction malfunction number																																			
SD18	⑥ Intelligent function module parameter error (for QCPU only)																																			
SD16	Parameter No. *6																																			
SD17	No.																																			
SD18	(Vacant)																																			
SD19																																				
SD20																																				
SD21																																				
SD22																																				
SD23																																				
SD24																																				
SD25																																				
SD26																																				
SD16		Parameter No. *6																																		
SD17	Error code for intelligent function module																																			
SD18	(Vacant)																																			
SD19																																				
SD20																																				
SD21																																				
SD22																																				
SD23																																				
SD24																																				
SD25																																				
SD26																																				

\* 6 For details of the parameter numbers, refer to the user's manual of the CPU module used.

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD50	Error reset	Error number that performs error reset	<ul style="list-style-type: none"> <li>Stores error number that performs error reset</li> </ul>	U	New	○+Rem
SD51	Battery low latch	Bit pattern indicating where battery voltage drop occurred	<ul style="list-style-type: none"> <li>All corresponding bits go 1(ON) when battery voltage drops.</li> <li>Subsequently, these remain 1(ON) even after battery voltage has been returned to normal.</li> </ul> <ul style="list-style-type: none"> <li>The alarm data can be held within the specified time when batter low occurs.</li> <li>The error indicates the complete discharge of the battery.</li> <li>When the QCPU is used, the memory card B is standard and therefore the corresponding bits always remain OFF.</li> </ul>	S (Error)	New	○
SD52	Battery low	Bit pattern indicating where battery voltage drop occurred	<ul style="list-style-type: none"> <li>Same configuration as SD51 above</li> <li>Turns to 0 (OFF) when the battery voltage returns to normal thereafter.</li> <li>When the QCPU is used, the memory card B is standard and therefore the corresponding bits always remain OFF.</li> </ul>	S (Error)	New	
SD53	AC/DC DOWN detection	Number of times for AC/DC DOWN	<ul style="list-style-type: none"> <li>Every time the input voltage falls to or below 85% (AC power)/65% (DC power) of the rating during calculation of the CPU module, the value is incremented by 1 and stored in BIN code.</li> </ul>	S (Error)	D9005	○+Rem
SD54	MINI link errors	Error detection state	<ul style="list-style-type: none"> <li>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).</li> <li>② Turns to 1 (ON) when communication between the mounted MINI(-S3) and CPU module cannot be made.</li> </ul>	S (Error)	D9004 format change	QnA
SD60	Blown fuse number	Number of module with blown fuse	<ul style="list-style-type: none"> <li>Value stored here is the lowest station I/O number of the module with the blown fuse.</li> </ul>	S (Error)	D9000	○+Rem
SD61	I/O module verification error number	I/O module verification error module number	<ul style="list-style-type: none"> <li>The lowest I/O number of the module where the I/O module verification number took place.</li> </ul>	S (Error)	D9002	
SD62	Annunciator number	Annunciator number	<ul style="list-style-type: none"> <li>The first annunciator number (F number) to be detected is stored here.</li> </ul>	S (Instruction execution)	D9009	○
SD63	Number of annunciators	Number of annunciators	<ul style="list-style-type: none"> <li>Stores the number of annunciators searched.</li> </ul>	S (Instruction execution)	D9124	



Special Register List

(2) System information

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU						
SD200	Status of switch	Status of CPU switch	<ul style="list-style-type: none"> <li>The switch status of the remote I/O module is stored in the following format.</li> </ul> <p>① Remote I/O module switch status Always 1: STOP</p>	S (Always)	New	Remote						
			<ul style="list-style-type: none"> <li>The CPU switch status is stored in the following format:</li> </ul> <table border="1"> <tr> <td>①: CPU switch status</td> <td>0: RUN 1: STOP 2: L.CLR</td> </tr> <tr> <td>②: Memory card switch</td> <td>Always OFF</td> </tr> <tr> <td>③: DIP switch</td> <td>b8 through b12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON b13 through b15 are vacant.</td> </tr> </table>	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 vacant.	S(Every END processing)	New	QCPU
			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 vacant.											
<ul style="list-style-type: none"> <li>The CPU switch status is stored in the following format:</li> </ul> <table border="1"> <tr> <td>①: CPU key Status of switch</td> <td>0 : RUN 1 : STOP 2 : L.CLR</td> </tr> <tr> <td>②: Memory cards switch</td> <td>b4 corresponds to memory card A, and b5 corresponds to memory card B OFF at 0; ON at 1</td> </tr> <tr> <td>③: DIP switch</td> <td>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</td> </tr> </table>	①: CPU key Status of switch	0 : RUN 1 : STOP 2 : L.CLR	②: Memory cards switch	b4 corresponds to memory card A, and b5 corresponds to memory card B OFF at 0; ON at 1	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			
①: CPU key Status of switch	0 : RUN 1 : STOP 2 : L.CLR											
②: Memory cards switch	b4 corresponds to memory card A, and b5 corresponds to memory card B OFF at 0; ON at 1											
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											

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU															
SD201	LED status	Status of CPU-LED	<ul style="list-style-type: none"> <li>The following bit patterns are used to store the statuses of the LEDs on the CPU module:</li> </ul> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">①: RUN</td> <td style="width: 50%;">⑤: BOOT</td> </tr> <tr> <td>②: ERR.</td> <td>⑥: Vacant</td> </tr> <tr> <td>③: USER</td> <td>⑦: Vacant</td> </tr> <tr> <td>④: BAT.</td> <td>⑧: MODE</td> </tr> </table> <p style="margin-top: 5px;">Bit patterns for MODE 0: OFF, 1: Green, 2: Orange</p> </div>	①: RUN	⑤: BOOT	②: ERR.	⑥: Vacant	③: USER	⑦: Vacant	④: BAT.	⑧: MODE	S (Status change)	New	QCPU							
			①: RUN	⑤: BOOT																	
②: ERR.	⑥: Vacant																				
③: USER	⑦: Vacant																				
④: BAT.	⑧: MODE																				
<ul style="list-style-type: none"> <li>Information concerning which of the following status the LEDs on the CPU module are in is stored in the following bit patterns:</li> <li>0 is off, 1 is on, and 2 is flicker</li> </ul> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">①: RUN</td> <td style="width: 50%;">⑤: BOOT</td> </tr> <tr> <td>②: ERROR</td> <td>⑥: CARD A (Memory card A)</td> </tr> <tr> <td>③: USER</td> <td>⑦: CARD B (Memory card B)</td> </tr> <tr> <td>④: BAT.ALARM</td> <td>⑧: Vacant</td> </tr> </table> </div>	①: RUN	⑤: BOOT	②: ERROR	⑥: CARD A (Memory card A)	③: USER	⑦: CARD B (Memory card B)	④: BAT.ALARM	⑧: Vacant	S (Status change)	New	QnA										
①: RUN	⑤: BOOT																				
②: ERROR	⑥: CARD A (Memory card A)																				
③: USER	⑦: CARD B (Memory card B)																				
④: BAT.ALARM	⑧: Vacant																				
SD202	LED off	Bit pattern of LED that is turned off	<ul style="list-style-type: none"> <li>Stores bit patterns of LEDs turned off (Only USER and BOOT enabled)</li> <li>Turned off at 1, not turned off at 0</li> </ul>	U	New	QnA															
SD203	Operating status of CPU	Operating status of CPU	<ul style="list-style-type: none"> <li>The operating status of the remote I/O module is stored in the following format.</li> </ul> <p>① Remote I/O module operating status Always 2: STOP</p>	S (Always)	New	Remote															
			<ul style="list-style-type: none"> <li>The CPU operating status is stored as indicated in the following figure:</li> </ul> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">①: Operating status of CPU</td> <td style="width: 50%;">0:RUN</td> </tr> <tr> <td></td> <td>1:STEP-RUN</td> </tr> <tr> <td></td> <td>2:STOP</td> </tr> <tr> <td></td> <td>3:PAUSE</td> </tr> <tr> <td>②: STOP/PAUSE cause</td> <td>0:RUN/STOP switch</td> </tr> <tr> <td></td> <td>1:Remote contact</td> </tr> <tr> <td></td> <td>2: Remote operation from the GX Developer or Serial Communication.</td> </tr> <tr> <td></td> <td>3:Internal program instruction</td> </tr> <tr> <td>Note: Priority is earliest first</td> <td>4:Errors</td> </tr> </table> </div>	①: Operating status of CPU	0:RUN		1:STEP-RUN		2:STOP		3:PAUSE	②: STOP/PAUSE cause	0:RUN/STOP switch		1:Remote contact		2: Remote operation from the GX Developer or Serial Communication.		3:Internal program instruction	Note: Priority is earliest first	4:Errors
①: Operating status of CPU	0:RUN																				
	1:STEP-RUN																				
	2:STOP																				
	3:PAUSE																				
②: STOP/PAUSE cause	0:RUN/STOP switch																				
	1:Remote contact																				
	2: Remote operation from the GX Developer or Serial Communication.																				
	3:Internal program instruction																				
Note: Priority is earliest first	4:Errors																				
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	<ul style="list-style-type: none"> <li>Set when the device test mode is executed on GX Developer.</li> </ul>	S (Request)	New	Remote															

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td> </td><td> </td><td> </td></tr></table>				Corresponding CPU																				
SD207	LED display priority ranking	Priorities 1 to 4	<ul style="list-style-type: none"> <li>When error is generated, the LED display (flicker) is made according to the error number setting priorities.</li> <li>The setting areas for priorities are as follows:  <table border="1" style="margin-left: 20px;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td>SD207</td> <td>Priority 4</td> <td>Priority 3</td> <td>Priority 2</td> <td>Priority 1</td> </tr> <tr> <td>SD208</td> <td>Priority 8</td> <td>Priority 7</td> <td>Priority 6</td> <td>Priority 5</td> </tr> <tr> <td>SD209</td> <td colspan="2"></td> <td>Priority 10</td> <td>Priority 9</td> </tr> </table> </li> </ul> <p>Default Value SD207=4321H SD208=8765H SD209=00A9H</p> <ul style="list-style-type: none"> <li>No display is made if "0" is set. 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. See Section 7.9.5 REMARK for the priority order.</li> </ul>	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 10	Priority 9	U	D9038	○				
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 10	Priority 9																									
SD208	Priorities 5 to 8	D3039 format change																											
SD209	Priorities 9 to 10	New																											
SD210	Clock data	Clock data (year, month)	<ul style="list-style-type: none"> <li>The year (last two digits) and month are stored as BCD code at SD210 as shown below:  <table border="1" style="margin-left: 20px;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Year</td> <td colspan="2">Month</td> </tr> </table> </li> </ul> <p>Example : July 1993 9307H</p>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Year		Month		S/U (Request)	D9025	○+Rem															
b15 to b12	b11 to b8	b7 to b4	b3 to b0																										
Year		Month																											
SD211	Clock data	Clock data (day, hour)	<ul style="list-style-type: none"> <li>The day and hour are stored as BCD code at SD211 as shown below:  <table border="1" style="margin-left: 20px;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Day</td> <td colspan="2">Hour</td> </tr> </table> </li> </ul> <p>Example : 31st, 10 a.m. 3110H</p>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Day		Hour		D9026																	
b15 to b12	b11 to b8	b7 to b4	b3 to b0																										
Day		Hour																											
SD212	Clock data	Clock data (minute, second)	<ul style="list-style-type: none"> <li>The minutes and seconds (after the hour) are stored as BCD code at SD212 as shown below:  <table border="1" style="margin-left: 20px;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Minute</td> <td colspan="2">Second</td> </tr> </table> </li> </ul> <p>Example : 35 min., 48 sec. 3548H</p>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Minute		Second		D9027																	
b15 to b12	b11 to b8	b7 to b4	b3 to b0																										
Minute		Second																											
SD213	Clock data	Clock data (day of week)	<ul style="list-style-type: none"> <li>Stores the year (two digits) and the day of the week in SD213 in the BCD code format as shown below.  <table border="1" style="margin-left: 20px;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Higher digits of year (0 to 99)</td> <td colspan="2">Day of week</td> </tr> </table> </li> </ul> <p>Example : 1993, Friday 1905H</p> <table border="1" style="margin-left: 20px;"> <tr><th>Day of 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>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Higher digits of year (0 to 99)		Day of week		Day of week	0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday	5	Friday	6	Saturday	S/U (Request)	D9028	QCPU Remote
			b15 to b12	b11 to b8	b7 to b4	b3 to b0																							
Higher digits of year (0 to 99)		Day of week																											
Day of week																													
0	Sunday																												
1	Monday																												
2	Tuesday																												
3	Wednesday																												
4	Thursday																												
5	Friday																												
6	Saturday																												
<ul style="list-style-type: none"> <li>The day of the week is stored as BCD code at SD213 as shown below:  <table border="1" style="margin-left: 20px;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="4">Always set "0"</td> </tr> </table> </li> </ul> <p>Example : Friday 0005H</p> <table border="1" style="margin-left: 20px;"> <tr><th>Day of 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>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Always set "0"				Day of week	0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday	5	Friday	6	Saturday	D9028	QnA				
b15 to b12	b11 to b8	b7 to b4	b3 to b0																										
Always set "0"																													
Day of week																													
0	Sunday																												
1	Monday																												
2	Tuesday																												
3	Wednesday																												
4	Thursday																												
5	Friday																												
6	Saturday																												

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU	
SD220	LED display data	Display indicator data	<ul style="list-style-type: none"> <li>LED display ASCII data (16 characters) stored here.</li> </ul>	S (When changed)	New	○	
SD221			b15 to b8				b7 to b0
SD222			15th character from the right				16th character from the right
SD223			13th character from the right				14th character from the right
SD224			11th character from the right				12th character from the right
SD225			9th character from the right				10th character from the right
SD226			7th character from the right				8th character from the right
SD227			5th character from the right				6th character from the right
SD240	Base mode	0: Automatic mode 1: Detail mode	<ul style="list-style-type: none"> <li>Stores the base mode.</li> </ul>	S (Initial)	New	QCPU Remote	
SD241	No. of extension bases	0: Main base only 1 to 7: No. of extension bases	<ul style="list-style-type: none"> <li>Stores the maximum number of the extension bases being installed.</li> </ul>	S (Initial)	New		
SD242	A/Q base differentiation	Base type differentiation 0: QA * * B is installed (A mode) 1: Q * * B is installed (Q mode)		S (Initial)	New		
SD243	No. of base slots	No. of base slots		S (Initial)	New		
SD244			<ul style="list-style-type: none"> <li>As shown above, each area stores the number of slots being installed.</li> </ul>				
SD250	Loaded maximum I/O	Loaded maximum I/O No.	<ul style="list-style-type: none"> <li>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.</li> </ul>	S (Request END)	New	○+Rem	
SD251	Head I/O number for replacement	Head I/O No. for module replacement	<ul style="list-style-type: none"> <li>Stores the upper two digits of the first I/O number of an I/O module that is removed/replaced in the online status.(default value : 100<sub>+</sub>)</li> </ul>	U	D9094	Q2A (S1) Q3A Q4A Q4AR	
SD253	RS422 transmission speed	RS422 transmission speed	<ul style="list-style-type: none"> <li>Stores transmission speed of RS422.</li> <li>0: 9600bps    1: 19.2kbps    2: 38.4kbps</li> </ul>	S (When changed)	New	QnA	
SD254	MELSECNET /10 (H) information	Number of modules installed	<ul style="list-style-type: none"> <li>Indicates the number of mounted MELSECNET/10 modules or MELSECNET/H modules.</li> </ul>	S (Initial)	New	○	
SD255		I/O No.	<ul style="list-style-type: none"> <li>I/O number of mounted MELSECNET/10 module or MELSECNET/H module</li> </ul>				
SD256		Network No.	<ul style="list-style-type: none"> <li>Network No. of mounted MELSECNET/10 module or MELSECNET/H module</li> </ul>				
SD257		Group number	<ul style="list-style-type: none"> <li>Group No. of mounted MELSECNET/10 module or MELSECNET/H module</li> </ul>				
SD258		Station No.	<ul style="list-style-type: none"> <li>Station No. of mounted MELSECNET/10 module or MELSECNET/H module</li> </ul>				
SD259		Standby information	<ul style="list-style-type: none"> <li>In the case of standby stations, the module number of the standby station is stored. (1 to 4)</li> </ul>				
SD260 to SD264		Information from 2nd module	<ul style="list-style-type: none"> <li>Configuration is identical to that for the 2nd module.</li> </ul>				
SD265 to SD269		Information from 3rd module	<ul style="list-style-type: none"> <li>Configuration is identical to that for the 3rd module.</li> </ul>				
SD270 to SD274	Information from 4th module	<ul style="list-style-type: none"> <li>Configuration is identical to that for the 4th module.</li> </ul>					

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9	Corresponding CPU
SD280	CC-Link error	Error detection status	<ul style="list-style-type: none"> <li>When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON).</li> <li>When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON).</li> <li>Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.</li> </ul> <p>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.)</p>	S (Error)	New	QCPU Remote
			<ul style="list-style-type: none"> <li>When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON).</li> <li>When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON).</li> <li>Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.</li> </ul>	S (Error)	New	QnA
SD290	Device allocation (Same as parameter contents)	Number of points allocated for X	<ul style="list-style-type: none"> <li>Stores the number of points currently set for X devices</li> </ul>	S (Initial)	New	○+Rem
SD291		Number of points allocated for Y	<ul style="list-style-type: none"> <li>Stores the number of points currently set for Y devices</li> </ul>			
SD292		Number of points allocated for M	<ul style="list-style-type: none"> <li>Stores the number of points currently set for M devices</li> </ul>			
SD293		Number of points allocated for L	<ul style="list-style-type: none"> <li>Stores the number of points currently set for L devices</li> </ul>			
SD294		Number of points allocated for B	<ul style="list-style-type: none"> <li>Stores the number of points currently set for B devices</li> </ul>			
SD295		Number of points allocated for F	<ul style="list-style-type: none"> <li>Stores the number of points currently set for F devices</li> </ul>			
SD296		Number of points allocated for SB	<ul style="list-style-type: none"> <li>Stores the number of points currently set for SB devices</li> </ul>			
SD297		Number of points allocated for V	<ul style="list-style-type: none"> <li>Stores the number of points currently set for V devices</li> </ul>			
SD298		Number of points allocated for S	<ul style="list-style-type: none"> <li>Stores the number of points currently set for S devices</li> </ul>			
SD299		Number of points allocated for T	<ul style="list-style-type: none"> <li>Stores the number of points currently set for T device</li> </ul>			
SD300		Number of points allocated for ST	<ul style="list-style-type: none"> <li>Stores the number of points currently set for ST devices</li> </ul>			
SD301		Number of points allocated for C	<ul style="list-style-type: none"> <li>Stores the number of points currently set for C devices</li> </ul>			



Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU			
SD302	Device allocation (Same as parameter contents)	Number of points allocated for D	• Stores the number of points currently set for D devices	S (Initial)	New	○+Rem			
SD303		Number of points allocated for W	• Stores the number of points currently set for W devices						
SD304		Number of points allocated for SW	• Stores the number of points currently set for SW devices						
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. Setting range: 1 to 100 ms If the designated value is out of the range above, it is assumed to no setting. The scan time becomes longer by the designated time.	END processing	New	QCPU			
SD340	Ethernet information	No. of modules installed	• Indicates the number of mounted Ethernet module.	S (Initial)	New	QCPU Remote			
SD341		Information of 1st module	I/O No.				• I/O number of mounted Ethernet module		
SD342			Network No.				• Network No. of mounted Ethernet module		
SD343			Group No.				• Group No. of mounted Ethernet module		
SD344			Station No.				• Station No. of mounted Ethernet module		
SD345 to SD346			Vacant				• Vacant (With QCPU, the Ethernet IP address of the 1st module is stored in buffer memory.)		
SD347			Vacant				• Vacant (With QCPU, the Ethernet error code of the 1st module is read with the ERRORRD instruction.)		
SD348 to SD354		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.						
SD362 to SD368		Information from 4th module	• Configuration is identical to that for the first module.						
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.				• I/O number of mounted Ethernet module		
SD342			Network No.				• Network No. of mounted Ethernet module		
SD343			Group No.				• Group No. of mounted Ethernet module		
SD344			Station No.				• Station No. of mounted Ethernet module		
SD345 to SD346			IP address				• IP address of mounted Ethernet module		
SD347			Error code				• Error code of mounted Ethernet module		
SD348 to SD354		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.						
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	<p>b15 to b8 b7 b6 b5 b4 b3 b2 b1 b0                  0 to 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</p> <p>ON: Received (Channel is used.)                  OFF: Not received (Channel is not used.)</p>	S (Initial)	New				

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD381	Ethernet	Information from 2nd module	• Configuration is identical to that for the first module.	S (Initial)	New	QnA
SD382	instruction reception	Information from 3rd module	• Configuration is identical to that for the first module.			
SD383	status	Information from 4th module	• Configuration is identical to that for the first module.			
SD392	Software version	Internal system software version	<p>• Stores the internal system software version in ASCII code.</p> <p>The data in the lower byte position is indefinite. The software version is stored in the higher byte position.</p> <p>Higher byte   Lower byte</p> <p style="text-align: center;">↑</p> <p>For version "A", for example, "41H" is stored.</p> <p>Note: The internal system software version may differ from the version indicated by the version symbol printed on the case.</p>	S (Initial)	D9060	QnA
SD395	Multiple PLC number	Multiple PLC number	<p>• In a multiple PLC system configuration, the PLC number of the host CPU is stored.</p> <p>PLC No. 1: 1, PLC No. 2: 2, PLC No. 3: 3, PLC No. 4: 4</p>	S (Initial)	New	QCPU function Ver. B or later

(3) System clocks/counters

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD412	1 second counter	Number of counts in 1-second units	<p>• Following programmable controller CPU module RUN, 1 is added each second</p> <p>• Count repeats from 0 to 32767 to -32768 to 0</p>	S (Status change)	D9022	○
SD414	2n second clock setting	2n second clock units	<p>• Stores value n of 2n second clock (Default is 30)</p> <p>• Setting can be made between 1 and 32767</p>	U	New	
SD415	2nms clock setting	2nms clock units	<p>• Stores value n of 2nms clock (Default is 30)</p> <p>• Setting can be made between 1 and 32767</p>	U	New	QCPU
SD420	Scan counter	Number of counts in each scan	<p>• Incremented by 1 for each scan execution after the CPU module is set to RUN. *</p> <p>• Count repeats from 0 to 32767 to -32768 to 0</p>	S (Every END processing)	New	○
SD430	Low speed scan counter	Number of counts in each scan	<p>• Incremented by 1 for each scan execution after the CPU module is set to RUN.</p> <p>• Count repeats from 0 to 32767 to -32768 to 0</p> <p>• Used only for low speed execution type programs</p>	S (Every END processing)	New	

\*: Not counted by the scan in an initial execution type program.

Special Register List

(4) Scan information

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding 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	
SD510	Low speed execution type program No.	Low speed execution type program No. in execution	• Program number of low speed execution type program No. currently being executed is stored as BIN value. • Enabled only when SM510 is ON.	S (Every END processing)	New	
SD520	Current scan time	Current scan time (in 1 ms units)	• Stores current scan time (in 1 ms units) Range from 0 to 65535	S (Every END processing)	D9017 format change	
SD521		Current scan time (in 100 μs units)	• Stores current scan time (in 100 μs units) Range from 00000 to 900 (Example) A current scan of 23.6 ms would be stored as follows: D520=23 D521=600	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 100μs units.) SD522: Stores the ms place. (Storage range: 0 to 65535) SD523: Stores the μs place. (Storage range: 0 to 900)	S (First END processing)	New	
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 100μs 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)	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 100μs 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)	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 100μs 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	
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 100μs 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 100μs 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 100μs 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	
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 100μs units.) SD542: Stores the ms place. (Storage range: 0 to 65535) SD543: Stores the μs place. (Storage range: 0 to 900)	S (First END processing)	New	
SD543		Constant scan wait time (in 100 μs units)				

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100μs units.)</li> <li>SD544: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD545: Stores the μs place. (Storage range: 0 to 900)</li> <li>Cleared to 0 after the end of one low speed scan.</li> </ul>	S (Every END processing)	New	
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)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100μs units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the μs place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	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)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100μs units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the μs place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	
SD549		Scan execution type program execution time (in 100 μs units)				
SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval</li> </ul>	U	New	
SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100μs units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the μs place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	○+Rem
SD552		Module service interval (in 100 μs units)				

Special Register List

(5) Memory card

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU																
SD600	Memory card models	Memory card models	<ul style="list-style-type: none"> <li>Indicates memory card model installed</li> </ul> <table border="1"> <tr> <td>Drive 1 (RAM) model</td> <td>0: Does not exist 1: SRAM</td> </tr> <tr> <td>Drive 2 (ROM) model</td> <td>0: Does not exist (1: SRAM) 2: ATA FLASH 3: Flash ROM</td> </tr> </table>	Drive 1 (RAM) model	0: Does not exist 1: SRAM	Drive 2 (ROM) model	0: Does not exist (1: SRAM) 2: ATA FLASH 3: Flash ROM	S (Initial and card removal)	New	QCPU												
	Drive 1 (RAM) model	0: Does not exist 1: SRAM																				
Drive 2 (ROM) model	0: Does not exist (1: SRAM) 2: ATA FLASH 3: Flash ROM																					
	Memory card A models	Memory card A models	<ul style="list-style-type: none"> <li>Indicates memory card A model installed</li> </ul> <table border="1"> <tr> <td>Drive 1 (RAM) model</td> <td>0: Does not exist 1: SRAM</td> </tr> <tr> <td>Drive 2 (ROM) model</td> <td>0: Does not exist 2: EEPROM 3: Flash ROM</td> </tr> </table>	Drive 1 (RAM) model	0: Does not exist 1: SRAM	Drive 2 (ROM) model	0: Does not exist 2: EEPROM 3: Flash ROM	S (Initial and card removal)	New	QnA												
Drive 1 (RAM) model	0: Does not exist 1: SRAM																					
Drive 2 (ROM) model	0: Does not exist 2: EEPROM 3: Flash ROM																					
SD602	Drive 1 (RAM) capacity	Drive 1 capacity	<ul style="list-style-type: none"> <li>Drive 1 capacity is stored in 1 kbyte units</li> </ul>	S (Initial and card removal)	New	QCPU																
				S (Initial and card removal)	New	QnA																
SD603	Drive 2 (ROM) capacity	Drive 2 capacity	<ul style="list-style-type: none"> <li>Drive 2 capacity is stored in 1 kbyte units</li> </ul>	S (Initial and card removal)	New	QCPU																
				S (Initial and card removal)	New	QnA																
SD604	Memory card use conditions	Memory card use conditions	<ul style="list-style-type: none"> <li>The use conditions for memory card (A) are stored as bit patterns (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr> <td>b0: Boot operation (QBT)</td> <td>b8: Not used</td> </tr> <tr> <td>b1: Parameters (QPA)</td> <td>b9: CPU fault history (QFD)</td> </tr> <tr> <td>b2: Device comments (QCD)</td> <td>b10: Not used</td> </tr> <tr> <td>b3: Device initial value (QDI)</td> <td>b11: Local device (QDL)</td> </tr> <tr> <td>b4: File register R (QDR)</td> <td>b12: Not used</td> </tr> <tr> <td>b5: Trace (QTS)</td> <td>b13: Not used</td> </tr> <tr> <td>b6: Not used</td> <td>b14: Not used</td> </tr> <tr> <td>b7: Not used</td> <td>b15: Not used</td> </tr> </table>	b0: Boot operation (QBT)	b8: Not used	b1: Parameters (QPA)	b9: CPU fault history (QFD)	b2: Device comments (QCD)	b10: Not used	b3: Device initial value (QDI)	b11: Local device (QDL)	b4: File register R (QDR)	b12: Not used	b5: Trace (QTS)	b13: Not used	b6: Not used	b14: Not used	b7: Not used	b15: Not used	S (Status change)	New	QCPU
	b0: Boot operation (QBT)	b8: Not used																				
b1: Parameters (QPA)	b9: CPU fault history (QFD)																					
b2: Device comments (QCD)	b10: Not used																					
b3: Device initial value (QDI)	b11: Local device (QDL)																					
b4: File register R (QDR)	b12: Not used																					
b5: Trace (QTS)	b13: Not used																					
b6: Not used	b14: Not used																					
b7: Not used	b15: Not used																					
	Memory card A use conditions	Memory card A use conditions	<ul style="list-style-type: none"> <li>The use conditions for memory card A are stored as bit patterns (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr> <td>b0: Boot operation (QBT)</td> <td>b8: Simulation data (QDS)</td> </tr> <tr> <td>b1: Parameters (QPA)</td> <td>b9: CPU fault history (QFD)</td> </tr> <tr> <td>b2: Device comments (QCD)</td> <td>b10: SFC trace (QTR)</td> </tr> <tr> <td>b3: Device initial value (QDI)</td> <td>b11: Local device (QDL)</td> </tr> <tr> <td>b4: File register R (QDR)</td> <td>b12: Not used</td> </tr> <tr> <td>b5: Sampling trace (QTS)</td> <td>b13: Not used</td> </tr> <tr> <td>b6: Status latch (QTL)</td> <td>b14: Not used</td> </tr> <tr> <td>b7: Program trace (QTP)</td> <td>b15: Not used</td> </tr> </table>	b0: Boot operation (QBT)	b8: Simulation data (QDS)	b1: Parameters (QPA)	b9: CPU fault history (QFD)	b2: Device comments (QCD)	b10: SFC trace (QTR)	b3: Device initial value (QDI)	b11: Local device (QDL)	b4: File register R (QDR)	b12: Not used	b5: Sampling trace (QTS)	b13: Not used	b6: Status latch (QTL)	b14: Not used	b7: Program trace (QTP)	b15: Not used	S (Status change)	New	QnA
b0: Boot operation (QBT)	b8: Simulation data (QDS)																					
b1: Parameters (QPA)	b9: CPU fault history (QFD)																					
b2: Device comments (QCD)	b10: SFC trace (QTR)																					
b3: Device initial value (QDI)	b11: Local device (QDL)																					
b4: File register R (QDR)	b12: Not used																					
b5: Sampling trace (QTS)	b13: Not used																					
b6: Status latch (QTL)	b14: Not used																					
b7: Program trace (QTP)	b15: Not used																					
SD620	Drive 3/4 models	Drive 3/4 models	<ul style="list-style-type: none"> <li>Indicates the drive 3/4 models.</li> </ul> <table border="1"> <tr> <td>Drive 3 (Standrd RAM)</td> <td>Fixed at "1".</td> </tr> <tr> <td>Drive 4 (Standrd ROM)</td> <td>Fixed at "3".</td> </tr> </table>	Drive 3 (Standrd RAM)	Fixed at "1".	Drive 4 (Standrd ROM)	Fixed at "3".	S (Initial)	New	QCPU												
Drive 3 (Standrd RAM)	Fixed at "1".																					
Drive 4 (Standrd ROM)	Fixed at "3".																					

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU																												
SD620	Memory card B models	Memory card B models	<ul style="list-style-type: none"> <li>Indicates memory card B models installed</li> </ul>	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR																												
SD622	Drive 3 (RAM) capacity	Drive 3 capacity	• Drive 3 capacity is stored in 1 kbyte units.	S (Initial)	New	QCPU																												
			• Drive 3 capacity is stored in 1 kbyte units	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR																												
SD623	Drive 4 (ROM) capacity	Drive 4 capacity	• Drive 4 capacity is stored in 1 kbyte units.	S (Initial)	New	QCPU																												
			• Drive 4 capacity is stored in 1 kbyte units	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR																												
SD624	Drive 3/4 use conditions	Drive 3/4 use conditions	<ul style="list-style-type: none"> <li>The conditions for usage for drive 3/4 are stored as bit patterns. (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr><td>b0: Boot operation (QBT)</td><td>b8: Not used</td></tr> <tr><td>b1: Parameters (QPA)</td><td>b9: CPU fault history (QFD)</td></tr> <tr><td>b2: Device comments (QCD)</td><td>b10: Not used</td></tr> <tr><td>b3: Device initial value (QDI)</td><td>b11: Local device (QDL)</td></tr> <tr><td>b4: File R (QDR)</td><td>b12: Not used</td></tr> <tr><td>b5: Trace (QTS)</td><td>b13: Not used</td></tr> <tr><td>b6: Not used</td><td>b14: Not used</td></tr> <tr><td>b7: Not used</td><td>b15: Not used</td></tr> </table>	b0: Boot operation (QBT)	b8: Not used	b1: Parameters (QPA)	b9: CPU fault history (QFD)	b2: Device comments (QCD)	b10: Not used	b3: Device initial value (QDI)	b11: Local device (QDL)	b4: File R (QDR)	b12: Not used	b5: Trace (QTS)	b13: Not used	b6: Not used	b14: Not used	b7: Not used	b15: Not used	S (Status change)	New	QCPU												
	b0: Boot operation (QBT)	b8: Not used																																
b1: Parameters (QPA)	b9: CPU fault history (QFD)																																	
b2: Device comments (QCD)	b10: Not used																																	
b3: Device initial value (QDI)	b11: Local device (QDL)																																	
b4: File R (QDR)	b12: Not used																																	
b5: Trace (QTS)	b13: Not used																																	
b6: Not used	b14: Not used																																	
b7: Not used	b15: Not used																																	
	Memory card B use conditions	Memory card B use conditions	<ul style="list-style-type: none"> <li>The use conditions for memory card B are stored as bit patterns (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr><td>b0: Boot operation (QBT)</td><td>b8: Simulation data (QDS)</td></tr> <tr><td>b1: Parameters (QPA)</td><td>b9: CPU fault history (QFD)</td></tr> <tr><td>b2: Device comments (QCD)</td><td>b10: SFC trace (QTR)</td></tr> <tr><td>b3: Device initial value (QDI)</td><td>b11: Local device (QDL)</td></tr> <tr><td>b4: File R (QDR)</td><td>b12: Not used</td></tr> <tr><td>b5: Sampling trace (QTS)</td><td>b13: Not used</td></tr> <tr><td>b6: Status latch (QTL)</td><td>b14: Not used</td></tr> <tr><td>b7: Program trace (QTP)</td><td>b15: Not used</td></tr> </table>	b0: Boot operation (QBT)	b8: Simulation data (QDS)	b1: Parameters (QPA)	b9: CPU fault history (QFD)	b2: Device comments (QCD)	b10: SFC trace (QTR)	b3: Device initial value (QDI)	b11: Local device (QDL)	b4: File R (QDR)	b12: Not used	b5: Sampling trace (QTS)	b13: Not used	b6: Status latch (QTL)	b14: Not used	b7: Program trace (QTP)	b15: Not used	S (Status change)	New	Q2A (S1) Q3A Q4A Q4AR												
b0: Boot operation (QBT)	b8: Simulation data (QDS)																																	
b1: Parameters (QPA)	b9: CPU fault history (QFD)																																	
b2: Device comments (QCD)	b10: SFC trace (QTR)																																	
b3: Device initial value (QDI)	b11: Local device (QDL)																																	
b4: File R (QDR)	b12: Not used																																	
b5: Sampling trace (QTS)	b13: Not used																																	
b6: Status latch (QTL)	b14: Not used																																	
b7: Program trace (QTP)	b15: Not used																																	
SD640	File register drive	Drive number:	• Stores drive number being used by file register	S (Initial)	New	○																												
SD641	File register file name	File register file name	<ul style="list-style-type: none"> <li>Stores file register file name (with extension) selected at parameters or by use of QDRSET instruction as ASCII code.</li> </ul> <table border="1"> <tr><td colspan="2">b15 to b8</td><td colspan="2">b7 to b0</td></tr> <tr><td>SD641</td><td>Second character</td><td colspan="2">First character</td></tr> <tr><td>SD642</td><td>Fourth character</td><td colspan="2">Third character</td></tr> <tr><td>SD643</td><td>Sixth character</td><td colspan="2">Fifth character</td></tr> <tr><td>SD644</td><td>Eighth character</td><td colspan="2">Seventh character</td></tr> <tr><td>SD645</td><td>First character of extension</td><td colspan="2">2EH(.)</td></tr> <tr><td>SD646</td><td>Third character of extension</td><td colspan="2">Second character of extension</td></tr> </table>	b15 to b8			b7 to b0		SD641	Second character	First character		SD642	Fourth character	Third character		SD643	Sixth character	Fifth character		SD644	Eighth character	Seventh character		SD645	First character of extension	2EH(.)		SD646	Third character of extension	Second character of extension		S (Initial)	New
b15 to b8				b7 to b0																														
SD641				Second character	First character																													
SD642				Fourth character	Third character																													
SD643				Sixth character	Fifth character																													
SD644				Eighth character	Seventh character																													
SD645	First character of extension	2EH(.)																																
SD646	Third character of extension	Second character of extension																																
SD642																																		
SD643																																		
SD644																																		
SD645																																		
SD646																																		
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																													
SD648	File register block number	File register block number	• Stores the currently selected file register block number.	S (Status change)	D9035																													
SD650	Comment drive	Comment drive number	• Stores the comment drive number selected at the parameters or by the QCDSSET instruction.	S (Status change)	New																													

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU																					
SD651	Comment file name	Comment file name	<ul style="list-style-type: none"> <li>Stores the comment file name (with extension) selected at the parameters or by the QCDSET instruction in ASCII code.</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD651</td> <td>Second character</td> <td>First character</td> </tr> <tr> <td>SD652</td> <td>Fourth character</td> <td>Third character</td> </tr> <tr> <td>SD653</td> <td>Sixth character</td> <td>Fifth character</td> </tr> <tr> <td>SD654</td> <td>Eighth character</td> <td>Seventh character</td> </tr> <tr> <td>SD655</td> <td>First character of extension</td> <td>2EH(.)</td> </tr> <tr> <td>SD656</td> <td>Third character of extension</td> <td>Second character of extension</td> </tr> </table>		b15 to b8	b7 to b0	SD651	Second character	First character	SD652	Fourth character	Third character	SD653	Sixth character	Fifth character	SD654	Eighth character	Seventh character	SD655	First character of extension	2EH(.)	SD656	Third character of extension	Second character of extension	S (Status change)	New	
				b15 to b8	b7 to b0																						
SD651				Second character	First character																						
SD652				Fourth character	Third character																						
SD653				Sixth character	Fifth character																						
SD654				Eighth character	Seventh character																						
SD655				First character of extension	2EH(.)																						
SD656	Third character of extension	Second character of extension																									
SD660	Boot operation designation file	Boot designation file drive number	<ul style="list-style-type: none"> <li>Stores the drive number where the boot designation file ( * .QBT) is being stored.</li> </ul>	S (Initial)	New	○																					
SD661		File name of boot designation file	<ul style="list-style-type: none"> <li>Stores the file name of the boot designation file ( * .QBT).</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD661</td> <td>Second character</td> <td>First character</td> </tr> <tr> <td>SD662</td> <td>Fourth character</td> <td>Third character</td> </tr> <tr> <td>SD663</td> <td>Sixth character</td> <td>Fifth character</td> </tr> <tr> <td>SD664</td> <td>Eighth character</td> <td>Seventh character</td> </tr> <tr> <td>SD665</td> <td>First character of extension</td> <td>2EH(.)</td> </tr> <tr> <td>SD666</td> <td>Third character of extension</td> <td>Second character of extension</td> </tr> </table>		b15 to b8	b7 to b0	SD661	Second character	First character	SD662	Fourth character	Third character	SD663	Sixth character	Fifth character	SD664	Eighth character	Seventh character	SD665	First character of extension	2EH(.)	SD666	Third character of extension	Second character of extension	S (Initial)	New	
				b15 to b8	b7 to b0																						
SD661				Second character	First character																						
SD662				Fourth character	Third character																						
SD663				Sixth character	Fifth character																						
SD664				Eighth character	Seventh character																						
SD665	First character of extension	2EH(.)																									
SD666	Third character of extension	Second character of extension																									

(6) Instruction-Related Registers

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU												
SD705	Mask pattern	Mask pattern	<ul style="list-style-type: none"> <li>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.</li> </ul>	U	New	○												
SD706																		
SD714	Number of vacant communication request registration areas	0 to 32	<ul style="list-style-type: none"> <li>Stores the number of vacant blocks in the communications request area for remote terminal modules connected to the MELSECNET/MINI-S3.</li> </ul>	S (During execution)	M9081	QnA												
SD715	IMASK instruction mask pattern	Mask pattern	<ul style="list-style-type: none"> <li>Patterns masked by use of the IMASK instruction are stored in the following manner:</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b1</td> <td>b0</td> </tr> <tr> <td>SD715</td> <td> 15</td> <td> 1  10</td> </tr> <tr> <td>SD716</td> <td> 31 to</td> <td> 17  16</td> </tr> <tr> <td>SD717</td> <td> 47 to</td> <td> 33  32</td> </tr> </table>		b15 to b1	b0	SD715	15	1  10	SD716	31 to	17  16	SD717	47 to	33  32	S (During execution)	New	○
				b15 to b1	b0													
SD715				15	1  10													
SD716	31 to	17  16																
SD717	47 to	33  32																
SD716																		
SD717																		
SD718	Accumulator	Accumulator	<ul style="list-style-type: none"> <li>For use as replacement for accumulators used in A-series programs.</li> </ul>	S/U	New													
SD719																		
SD720	Program No. designation for PLOAD instruction	Program No. designation for PLOAD instruction	<ul style="list-style-type: none"> <li>Stores the program number of the program to be loaded by the PLOAD instruction when designated.</li> <li>Designation range: 1 to 124</li> </ul>	U	New	QCPU												
SD730	No. of vacant registration area for CC-Link communication request	0 to 32	<ul style="list-style-type: none"> <li>Stores the number of vacant registration area for the request for communication with the intelligent device station connected to A(1S)J61QBT61.</li> </ul>	S (During execution)	New	QnA												
SD736	PKEY input	PKEY input	<ul style="list-style-type: none"> <li>Special register that temporarily stores keyboard data input by means of the PKEY instruction.</li> </ul>	S (During execution)	New	QnA												

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	Corresponding CPU																																																																																																																																				
SD738	Message storage	Message storage	<ul style="list-style-type: none"> <li>Stores the message designated by the MSG instruction.</li> </ul> <table border="1"> <tr> <td colspan="2">b15 to b8</td> <td colspan="2">b7 to b0</td> </tr> <tr> <td>SD738</td> <td>2nd character</td> <td colspan="2">1st character</td> </tr> <tr> <td>SD739</td> <td>4th character</td> <td colspan="2">3rd character</td> </tr> <tr> <td>SD740</td> <td>6th character</td> <td colspan="2">5th character</td> </tr> <tr> <td>SD741</td> <td>8th character</td> <td colspan="2">7th character</td> </tr> <tr> <td>SD742</td> <td>10th character</td> <td colspan="2">9th character</td> </tr> <tr> <td>SD743</td> <td>12th character</td> <td colspan="2">11th character</td> </tr> <tr> <td>SD744</td> <td>14th character</td> <td colspan="2">13th character</td> </tr> <tr> <td>SD745</td> <td>16th character</td> <td colspan="2">15th character</td> </tr> <tr> <td>SD746</td> <td>18th character</td> <td colspan="2">17th character</td> </tr> <tr> <td>SD747</td> <td>20th character</td> <td colspan="2">19th character</td> </tr> <tr> <td>SD748</td> <td>22nd character</td> <td colspan="2">21st character</td> </tr> <tr> <td>SD749</td> <td>24th character</td> <td colspan="2">23rd character</td> </tr> <tr> <td>SD750</td> <td>26th character</td> <td colspan="2">25th character</td> </tr> <tr> <td>SD751</td> <td>28th character</td> <td colspan="2">27th character</td> </tr> <tr> <td>SD752</td> <td>30th character</td> <td colspan="2">29th character</td> </tr> <tr> <td>SD753</td> <td>32nd character</td> <td colspan="2">31st character</td> </tr> <tr> <td>SD754</td> <td>34th character</td> <td colspan="2">33rd character</td> </tr> <tr> <td>SD755</td> <td>36th character</td> <td colspan="2">35th character</td> </tr> <tr> <td>SD756</td> <td>38th character</td> <td colspan="2">37th character</td> </tr> <tr> <td>SD757</td> <td>40th character</td> <td colspan="2">39th character</td> </tr> <tr> <td>SD758</td> <td>42nd character</td> <td colspan="2">41st character</td> </tr> <tr> <td>SD759</td> <td>44th character</td> <td colspan="2">43rd character</td> </tr> <tr> <td>SD760</td> <td>46th character</td> <td colspan="2">45th character</td> </tr> <tr> <td>SD761</td> <td>48th character</td> <td colspan="2">47th character</td> </tr> <tr> <td>SD762</td> <td>50th character</td> <td colspan="2">49th character</td> </tr> <tr> <td>SD763</td> <td>52nd character</td> <td colspan="2">51st character</td> </tr> <tr> <td>SD764</td> <td>54th character</td> <td colspan="2">53rd character</td> </tr> <tr> <td>SD765</td> <td>56th character</td> <td colspan="2">55th character</td> </tr> <tr> <td>SD766</td> <td>58th character</td> <td colspan="2">57th character</td> </tr> <tr> <td>SD767</td> <td>60th character</td> <td colspan="2">59th character</td> </tr> <tr> <td>SD768</td> <td>62nd character</td> <td colspan="2">61st character</td> </tr> <tr> <td>SD769</td> <td>64th character</td> <td colspan="2">63rd character</td> </tr> </table>	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	○
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																																																																																																																																								
SD774 TO SD775	PID limit setting (for complete derivative)	0: Limit set 1: Limit not set	<ul style="list-style-type: none"> <li>Designate the limit for each PID loop as follows:</li> </ul> <table border="1"> <tr> <td colspan="2">b15 to b1</td> <td colspan="2">b0</td> </tr> <tr> <td>SD774</td> <td>Loop16</td> <td>to</td> <td>Loop2</td> </tr> <tr> <td>SD775</td> <td>Loop32</td> <td>to</td> <td>Loop18</td> </tr> <tr> <td></td> <td></td> <td></td> <td>Loop17</td> </tr> </table>	b15 to b1		b0		SD774	Loop16	to	Loop2	SD775	Loop32	to	Loop18				Loop17	U	New	QCPU																																																																																																																				
b15 to b1		b0																																																																																																																																								
SD774	Loop16	to	Loop2																																																																																																																																							
SD775	Loop32	to	Loop18																																																																																																																																							
			Loop17																																																																																																																																							
SD778	Refresh processing selection when the COM instruction is executed	b0 to b14: 0: Do not refresh 1: Refresh  b15 bit: 0: General data processing executed 1: General data processing not executed	<ul style="list-style-type: none"> <li>Selects whether or not the data is refreshed when the COM instruction is executed.</li> <li>Designation of SD778 is made valid when SM775 turns ON.</li> </ul> <table border="1"> <tr> <td>b15</td> <td>b14</td> <td>to</td> <td>b5</td> <td>b4</td> <td>b3</td> <td>b2</td> <td>b1</td> <td>b0</td> </tr> <tr> <td>1/0</td> <td>0</td> <td></td> <td>1/0</td> <td>1/0</td> <td>1/0</td> <td>1/0</td> <td>1/0</td> <td>1/0</td> </tr> </table> <ul style="list-style-type: none"> <li>I/O refresh</li> <li>CC-Link refresh</li> <li>MELSECNET/H refresh</li> <li>Automatic refresh of intelligent function modules</li> <li>Automatic refresh of CPU shared memory</li> <li>Execution/non-execution of general data processing</li> </ul>	b15	b14	to	b5	b4	b3	b2	b1	b0	1/0	0		1/0	1/0	1/0	1/0	1/0	1/0	U	New	QCPU Serial number 04012 or later																																																																																																																		
b15	b14	to	b5	b4	b3	b2	b1	b0																																																																																																																																		
1/0	0		1/0	1/0	1/0	1/0	1/0	1/0																																																																																																																																		





Special Register List

(8) Latch area

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding 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 extension 2EH(.) SD906 3rd character of extension 2nd character of extension	S (Status change)	New	
SD902						
SD903						
SD904						
SD905						
SD906						
SD910						
SD911						
SD912						
SD913						
SD914						
SD915						
SD916						
SD917						
SD918						
SD919						
SD920						
SD921						
SD922						
SD923						
SD924						
SD925						

(9) A to Q/QnA conversion correspondences

ACPU special registers D9000 to D9255 correspond to the special registers SD1000 to SD1255 after A-series to the Q/QnA-series conversion.

These special registers are all set by the system, and users cannot use them to set program data.

Users who need to set data with these registers should edit the special registers for the Q/QnA.

However, before conversion users could set data at special registers D9200 to D9255 only, and after conversion users can also set data at registers 1200 to 1255.

For more detailed information concerning the contents of the ACPU special registers, see the individual CPU users manual, and the MELSECNET and MELSECNET/B data link system reference manual.

**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.

Special Register List

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																								
D9000	SD1000	—	Fuse blown	Number of module with blown fuse	<ul style="list-style-type: none"> <li>• 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.)</li> <li>• Fuse blow check is executed also to the output modules of remote I/O stations.</li> </ul>																																									
D9001	SD1001	—	Fuse blown	Number of module with blown fuse	<ul style="list-style-type: none"> <li>• Stores the module numbers corresponding to setting switch numbers or base slot numbers when fuse blow occurred.</li> </ul> <table border="1" style="margin-left: 20px;"> <thead> <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> </thead> <tbody> <tr><td>0</td><td>1</td><td>0</td><td>5</td></tr> <tr><td>1</td><td>2</td><td>1</td><td>6</td></tr> <tr><td>2</td><td>3</td><td>2</td><td>7</td></tr> <tr><td>3</td><td>4</td><td>3</td><td>8</td></tr> <tr><td>4</td><td>5</td><td></td><td></td></tr> <tr><td>5</td><td>6</td><td></td><td></td></tr> <tr><td>6</td><td>7</td><td></td><td></td></tr> <tr><td>7</td><td>8</td><td></td><td></td></tr> </tbody> </table>	I/O module for A0J2		Extension base unit		Setting switch	Stored data	Base unit slot No.	Stored data	0	1	0	5	1	2	1	6	2	3	2	7	3	4	3	8	4	5			5	6			6	7			7	8			○
I/O module for A0J2		Extension base unit																																												
Setting switch	Stored data	Base unit slot No.	Stored data																																											
0	1	0	5																																											
1	2	1	6																																											
2	3	2	7																																											
3	4	3	8																																											
4	5																																													
5	6																																													
6	7																																													
7	8																																													

Special Register List (Continued)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU														
D9002	SD1002	—	I/O module verification error	I/O module verification error module number	<ul style="list-style-type: none"> <li>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.)</li> <li>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.)</li> <li>I/O module verify check is executed also to the modules of remote I/O terminals.</li> </ul>	○														
D9004	SD1004	—	MINI link errors	Stores setting status made at parameters (modules 1 to 8)	<ul style="list-style-type: none"> <li>Error status of the MINI(S3) link detected on loaded AJ71PT32(S3) is stored.</li> </ul> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">b15</td><td style="text-align: center;">to</td><td style="text-align: center;">b8</td><td style="text-align: center;">b7</td><td style="text-align: center;">to</td><td style="text-align: center;">b0</td> </tr> <tr> <td style="text-align: center;">8</td><td style="text-align: center;">7</td><td style="text-align: center;">6</td><td style="text-align: center;">5</td><td style="text-align: center;">4</td><td style="text-align: center;">3</td><td style="text-align: center;">2</td><td style="text-align: center;">1</td> </tr> </table> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="border: 1px solid black; padding: 5px; font-size: small;">             Bits which correspond to faulty AJ71PT32(S3) are turned on.         </div> <div style="border: 1px solid black; padding: 5px; font-size: small;">             Bits which correspond to the signals of AJ71PT32(S3), shown below, are turned on as the signals are turned on.             <ul style="list-style-type: none"> <li>• Hardware error (X0/X20)</li> <li>• MINI(S3) link error detection (X6/X26)</li> <li>• MINI(S3) link communication error (X7/X27)</li> </ul> </div> </div>	b15	to	b8	b7	to	b0	8	7	6	5	4	3	2	1	QnA
b15	to	b8	b7	to	b0															
8	7	6	5	4	3	2	1													
D9005	SD1005	—	AC DOWN counter	Number of times for AC DOWN	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 power is switched from OFF to ON.	○														
					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 power is switched from OFF to ON.															
					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 power is switched from OFF to ON.	QnA														
D9008	SD1008	SD0	Self-diagnosis error	Self-diagnosis error number	<ul style="list-style-type: none"> <li>When error is found as a result of self-diagnosis, error number is stored in BIN code.</li> </ul>	○														
D9009	SD1009	SD62	Annunciator detection	F number at which external failure has occurred	<ul style="list-style-type: none"> <li>When one of F0 to 255 is turned on by <b>OUT F</b> or <b>SET F</b>, the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code.</li> <li>SD62 can be cleared by <b>RST F</b> or <b>LEDR</b> instruction. If another F number has been detected, the clearing of SD62 causes the next number to be stored in SD62.</li> </ul>	QCPU Q2AS Q2A														
					<ul style="list-style-type: none"> <li>When one of F0 to 255 is turned on by <b>OUT F</b> or <b>SET F</b>, the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code.</li> <li>SD62 can be cleared by executing <b>RST F</b> or <b>LEDR</b> 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.</li> </ul>	Q3A Q4A Q4AR														
D9010	SD1010	X	Error step	Step number at which operation error has occurred.	<ul style="list-style-type: none"> <li>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.</li> </ul>	○														

Special Register List (Continued)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9011	SD1011		Error step	Step number at which operation error has occurred.	<ul style="list-style-type: none"> <li>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.</li> </ul>	○
D9014	SD1014		I/O control mode	I/O control mode number	<ul style="list-style-type: none"> <li>The I/O control mode set is returned in any of the following numbers:                             <ol style="list-style-type: none"> <li>Both input and output in direct mode</li> <li>Input in refresh mode, output in direct mode</li> <li>Both input and output in refresh mode</li> </ol> </li> </ul>	
D9015	SD1015	SD203	Operating status of CPU	Operating status of CPU	<ul style="list-style-type: none"> <li>The operation status of CPU as shown below are stored in SD203.</li> </ul>	
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 <sup>2</sup> PROM) 9: Subprogram 1 (E <sup>2</sup> PROM) A: Subprogram 2 (E <sup>2</sup> PROM) B: Subprogram 3 (E <sup>2</sup> PROM)	<ul style="list-style-type: none"> <li>Indicates which sequence program is run presently. One value of 0 to B is stored in BIN code.</li> </ul>	
D9017	SD1017	SD520	Scan time	Minimum scan time (10 ms units)	<ul style="list-style-type: none"> <li>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.</li> </ul>	
D9018	SD1018	SD524	Scan time	Scan time (10 ms units)	<ul style="list-style-type: none"> <li>Scan time is stored in BIN code at each END and always rewritten.</li> </ul>	

Special Register List (Continued)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																								
D9019	SD1019	SD526	Scan time	Maximum scan time (10 ms units)	<ul style="list-style-type: none"> <li>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.</li> </ul>																									
D9020	SD1020	X	Constant scan	Constant scan time (User sets in 10 ms units)	<ul style="list-style-type: none"> <li>Sets the interval between consecutive program starts in multiples of 10 ms.                             <ul style="list-style-type: none"> <li>0: No setting</li> <li>1 to 200: Set. Program is executed at intervals of (set value) × 10 ms.</li> </ul> </li> </ul>																									
D9021	SD1021	—	Scan time	Scan time (1 ms units)	<ul style="list-style-type: none"> <li>Scan time is stored and updated in BIN code after every END.</li> </ul>																									
D9022	SD1022	SD412	1 second counter	Count in units of 1s.	<ul style="list-style-type: none"> <li>When the PC CPU starts running, it starts counting 1 every second.</li> <li>It starts counting up from 0 to 32767, then down to -32768 and then again up to 0. Counting repeats this routine.</li> </ul>																									
D9025	SD1025	SD210	Clock data	Clock data (year, month)	<ul style="list-style-type: none"> <li>Stores the year (2 lower digits) and month in BCD.                             <div style="display: flex; justify-content: space-around; align-items: center;"> <table border="1" style="font-size: small;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Year</td> <td colspan="2">Month</td> </tr> </table> <div style="margin-left: 10px;">                     Example: 1987, July H8707                 </div> </div> </li> </ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Year		Month																		
b15 to b12	b11 to b8	b7 to b4	b3 to b0																											
Year		Month																												
D9026	SD1026	SD211	Clock data	Clock data (day, hour)	<ul style="list-style-type: none"> <li>Stores the day and hour in BCD.                             <div style="display: flex; justify-content: space-around; align-items: center;"> <table border="1" style="font-size: small;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Day</td> <td colspan="2">Hour</td> </tr> </table> <div style="margin-left: 10px;">                     Example 31th, 10 o'clock H3110                 </div> </div> </li> </ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Day		Hour																		
b15 to b12	b11 to b8	b7 to b4	b3 to b0																											
Day		Hour																												
D9027	SD1027	SD212	Clock data	Clock data (minute, second)	<ul style="list-style-type: none"> <li>Stores the Minute and second in BCD.                             <div style="display: flex; justify-content: space-around; align-items: center;"> <table border="1" style="font-size: small;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td colspan="2">Minute</td> <td colspan="2">Second</td> </tr> </table> <div style="margin-left: 10px;">                     Example: 35 minutes, 48 seconds H3548                 </div> </div> </li> </ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Minute		Second																		
b15 to b12	b11 to b8	b7 to b4	b3 to b0																											
Minute		Second																												
D9028	SD1028	SD213	Clock data	Clock data (day of week)	<ul style="list-style-type: none"> <li>Stores the day of the week in BCD.                             <div style="display: flex; justify-content: space-around; align-items: center;"> <table border="1" style="font-size: small;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> </table> <div style="margin-left: 10px;">                     Example: Friday H0005                 </div> </div> <div style="margin-top: 10px;"> <table border="1" style="font-size: x-small; border-collapse: collapse;"> <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> <p style="margin-left: 40px;">0 must be set.</p> </div> </li></ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Day of the week		0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday	5	Friday	6	Saturday					
b15 to b12	b11 to b8	b7 to b4	b3 to b0																											
Day of the week																														
0	Sunday																													
1	Monday																													
2	Tuesday																													
3	Wednesday																													
4	Thursday																													
5	Friday																													
6	Saturday																													
D9035	SD1035	SD648	Extension file register	Use block No.	<ul style="list-style-type: none"> <li>Stores the block No. of the extension file register being used in BCD code.</li> </ul>																									
D9036	SD1036	X	Extension file register for designation of device number	Device number when individual devices from extension file register are directly accessed	<ul style="list-style-type: none"> <li>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.</li> </ul> <div style="margin-top: 10px;"> <table style="font-size: x-small; border-collapse: collapse;"> <tr> <td style="border: none;">Extension file register</td> <td style="border: none;">0</td> <td style="border: none;">to</td> <td style="border: none;">Block No.1 area</td> </tr> <tr> <td style="border: none;"></td> <td style="border: none;">16383</td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> <tr> <td style="border: none;"></td> <td style="border: none;">16384</td> <td style="border: none;">to</td> <td style="border: none;">Block No.2 area</td> </tr> <tr> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> <tr> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> <tr> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> </table> </div>	Extension file register	0	to	Block No.1 area		16383				16384	to	Block No.2 area													
Extension file register	0	to				Block No.1 area																								
	16383																													
	16384	to	Block No.2 area																											
D9037	SD1037	X			<div style="margin-top: 10px;"> <table style="font-size: x-small; border-collapse: collapse;"> <tr> <td style="border: none;">SD1037, SD1036</td> <td style="border: none;">Device No. (BIN data)</td> <td style="border: none;">→</td> <td style="border: none;">[ ]</td> <td style="border: none;">to</td> <td style="border: none;">[ ]</td> </tr> </table> </div>	SD1037, SD1036	Device No. (BIN data)	→	[ ]	to	[ ]																			
SD1037, SD1036	Device No. (BIN data)	→	[ ]	to	[ ]																									
D9038	SD1038	SD207	LED display priority ranking	Priorities 1 to 4	<ul style="list-style-type: none"> <li>Sets priority of ERROR LEDs which illuminate (or flicker) to indicate errors with error code numbers.</li> <li>Configuration of the priority setting areas is as shown below.                             <div style="margin-top: 10px;"> <table style="font-size: x-small; border-collapse: collapse;"> <tr> <td style="border: none;">SD207</td> <td style="border: none;">Priority 4</td> <td style="border: none;">Priority 3</td> <td style="border: none;">Priority 2</td> <td style="border: none;">Priority 1</td> </tr> <tr> <td style="border: none;">SD208</td> <td style="border: none;">Priority 7</td> <td style="border: none;">Priority 6</td> <td style="border: none;">Priority 5</td> <td style="border: none;"></td> </tr> </table> </div> </li> <li>For details, refer to the applicable CPUs User's Manual and the ACPU Programming manual (Fundamentals).</li> </ul>	SD207	Priority 4	Priority 3	Priority 2	Priority 1	SD208	Priority 7	Priority 6	Priority 5																
SD207	Priority 4	Priority 3		Priority 2		Priority 1																								
SD208	Priority 7	Priority 6	Priority 5																											
D9039	SD1039	SD208		Priorities 5 to 7																										

Special Register List (Continued)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU				
D9044	SD1044		For sampling trace	Step or time during sampling trace	<ul style="list-style-type: none"> <li>Turned on/off with a peripheral device.</li> <li>At scanning -----0</li> <li>At time -----Time (10 msec unit)</li> <li>Stores the value in BIN code.</li> </ul>					
D9049	SD1049		Work area for SFC	Block number of extension file register	<ul style="list-style-type: none"> <li>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.</li> <li>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.</li> </ul>					
D9050	SD1050		SFC program error number	Error code generated by SFC program	<ul style="list-style-type: none"> <li>Stores error code of errors occurred in the SFC program in BIN code.</li> <li>0: No error</li> <li>80: SFC program parameter error</li> <li>81: SFC code error</li> <li>82: Number of steps of simultaneous execution exceeded</li> <li>83: Block start error</li> <li>84: SFC program operation error</li> </ul>					
D9051	SD1051		Error block	Block number where error occurred	<ul style="list-style-type: none"> <li>Stores the block number in which an error occurred in the SFC program in BIN code.</li> <li>In the case of error 83 the starting block number is stored.</li> </ul>	○				
D9052	SD1052			Step number where error occurred	<ul style="list-style-type: none"> <li>Stores the step number, where error code 84 occurred in an SFC program, in BIN value.</li> <li>Stores "0" when error code 80, 81 or 82 occurred.</li> <li>Stores the block stating step number when error code 83 occurs.</li> </ul>					
D9053	SD1053		Error transition	Transition condition number where error occurred	<ul style="list-style-type: none"> <li>Stores the transition condition number, where error code 84 occurred in an SFC program, in BIN value.</li> <li>Stores "0" when error code 80, 81, 82 or 83 occurred.</li> </ul>					
D9054	SD1054		Error sequence step	Sequence step number where error occurred	<ul style="list-style-type: none"> <li>Stores the sequence step number of transfer condition and operation output in which error 84 occurred in the SFC program in BIN code.</li> </ul>					
D9055	SD1055	SD812	Status latch	Status latch step	<ul style="list-style-type: none"> <li>Stores the step number when status latch is executed.</li> <li>Stores the step number in a binary value if status latch is executed in a main sequence program.</li> <li>Stores the block number and the step number if status latch is executed in a SFC program.</li> </ul> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 10px auto;"> <table style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">Block No. (BIN)</td> <td style="border: 1px solid black; padding: 2px;">Step No. (BIN)</td> </tr> <tr> <td style="text-align: center;">← Higher 8 bits →</td> <td style="text-align: center;">← Lower 8 bits →</td> </tr> </table> </div>	Block No. (BIN)	Step No. (BIN)	← Higher 8 bits →	← Lower 8 bits →	
Block No. (BIN)	Step No. (BIN)									
← Higher 8 bits →	← Lower 8 bits →									
D9060	SD1060	SD392	Software version	Software version of internal software	<ul style="list-style-type: none"> <li>Stores the software version of the internal system in ASCII code.</li> </ul> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 10px auto;"> <table style="border-collapse: collapse;"> <tr> <td style="border: 1px solid black; padding: 2px;">Higher byte</td> <td style="border: 1px solid black; padding: 2px;">Lower byte</td> </tr> </table> <p style="text-align: center; margin: 0;">↑ The data in the lower byte position is indefinite. The software version is stored in the higher byte position.</p> </div> <p>For version "A", for example, "41H" is stored.</p> <p>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.</p>	Higher byte	Lower byte	QnA		
Higher byte	Lower byte									
D9072	SD1072		PLC communications check	Computer link data check	<ul style="list-style-type: none"> <li>In the self-loopback test of the serial communication module, the serial communication module writes/reads data automatically to make communication checks.</li> </ul>	○				
D9081	SD1081	SD714	Number of empty blocks in communications request registration area	Number of empty blocks in communications request registration area	<ul style="list-style-type: none"> <li>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.</li> </ul>	QnA				

Special Register List (Continued)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																				
D9085	SD1085		Register for setting time check value	Default value 10s	<ul style="list-style-type: none"> <li>• Sets the time check time of the data link instructions (ZNRD, ZNWR) for the MELSECNET/10.</li> <li>• Setting range: 1 s to 65535 s (1 to 65535)</li> <li>• Setting unit: 1 s</li> <li>• Default value: 10 s (If 0 has been set, default 10 s is applied)</li> </ul>	○																																																																				
D9090	SD1090		Number of special functions modules over	Number of special functions modules over	<ul style="list-style-type: none"> <li>• For details, refer to the manual of each microcomputer program package.</li> </ul>																																																																					
D9091	SD1091		Detailed error code	Self-diagnosis detailed error code	<ul style="list-style-type: none"> <li>• Stores the detail code of cause of an instruction error.</li> </ul>																																																																					
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	<ul style="list-style-type: none"> <li>• Stores the first two digits of the head I/O number of the I/O module, which will be dismantled/mounted online, in BIN value.</li> <li>Example) Input module X2F0 → H2F</li> </ul>	QCPU																																																																				
D9095	SD1095	SD200	DIP switch information	DIP switch information	<ul style="list-style-type: none"> <li>• The DIP switch information of the CPU module is stored in the following format.</li> <li>0: OFF</li> <li>1: ON</li> </ul>																																																																					
D9100	SD1100	—	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown	<ul style="list-style-type: none"> <li>• Output module numbers (in units of 16 points), of which fuses have blown, are entered in bit pattern. (Preset output unit numbers when parameter setting has been performed.)</li> </ul> <table border="1"> <thead> <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> </thead> <tbody> <tr> <td>SD1100</td> <td>0</td><td>0</td><td>0</td><td>1 (1C0)</td><td>0</td><td>0</td><td>0</td><td>1 (1B0)</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>SD1101</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>SD1107</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>1 (77)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (7B)</td><td>0</td><td>0</td><td>0</td> </tr> </tbody> </table> <p>↑ Indicates fuse blow.</p>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1100	0	0	0	1 (1C0)	0	0	0	1 (1B0)	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 (77)	0	0	0	0	0	0	0	1 (7B)	0	0	0	○
	b15				b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																							
SD1100	0				0	0	1 (1C0)	0	0	0	1 (1B0)	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 (77)	0	0	0	0	0	0	0	1 (7B)	0	0	0																																																							
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	<ul style="list-style-type: none"> <li>• Set the set value of the step transition watchdog timer and the annunciator number (F number) that will turn on when the watchdog timer times out.</li> </ul> <p>b15 to b8 b7 to b0</p>	○																																																																				
D9109	SD1109																																																																									
D9110	SD1110																																																																									
D9111	SD1111																																																																									
D9112	SD1112																																																																									
D9113	SD1113																																																																									
D9114	SD1114																																																																									

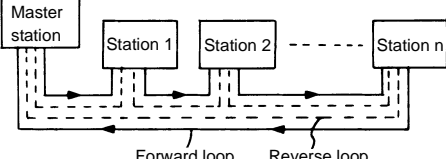
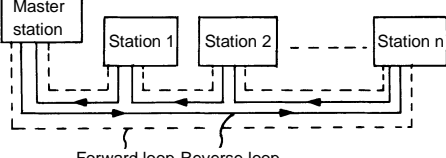


Special Register List (Continued)

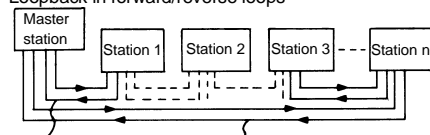
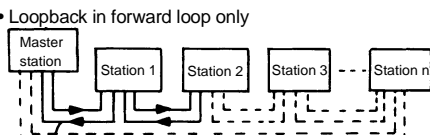
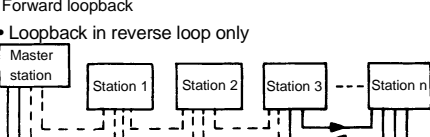
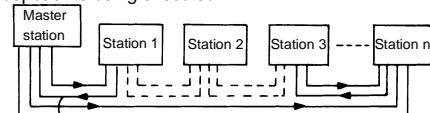
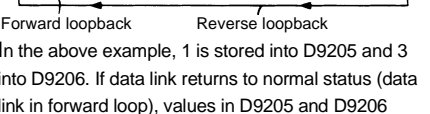
ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																																																																																																																																			
D9116	SD1116	-	I/O module verification error	Bit pattern, in units of 16 points, indicating the modules with verification errors.	<ul style="list-style-type: none"> <li>When I/O modules, of which data are different from those entered at power-on, have been detected, the I/O unit numbers (in units of 16 points) are entered in bit pattern. (Preset I/O unit numbers when parameter setting has been performed.)</li> </ul> <table border="1"> <tr> <td></td> <td>b15</td><td>b14</td><td>b13</td><td>b12</td><td>b11</td><td>b10</td><td>b9</td><td>b8</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td> </tr> <tr> <td>SD1116</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>(XY) 0</td> </tr> <tr> <td>SD1117</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>SD1123</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> <p style="text-align: center;">↑ Indicates I/O module verify error.</p> <ul style="list-style-type: none"> <li>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.)</li> </ul>		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	(XY) 0	SD1117	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SD1123	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																																
	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	(XY) 0																																																																																																																																																																					
SD1117	0					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																																																																																					
SD1123	0					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																																																																																					
D9117	SD1117																																																																																																																																																																																								
D9118	SD1118																																																																																																																																																																																								
D9119	SD1119																																																																																																																																																																																								
D9120	SD1120																																																																																																																																																																																								
D9121	SD1121																																																																																																																																																																																								
D9122	SD1122																																																																																																																																																																																								
D9123	SD1123																																																																																																																																																																																								
D9124	SD1124	SD63	Annunciator detection quantity	Annunciator detection quantity	<ul style="list-style-type: none"> <li>When one of F0 to 255 (F0 to 2047 for AuA and AnU) is turned on by <b>[SET F]</b> 1 is added to the contents of SD63. When <b>[RST F]</b> or <b>[LEDR]</b> 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.)</li> <li>Quantity, which has been turned on by <b>[SET F]</b> is stored into SD63 in BIN code. The value of SD63 is maximum 8.</li> </ul>																																																																																																																																																																																				
D9125	SD1125	SD64	Annunciator detection number	Annunciator detection number	<ul style="list-style-type: none"> <li>When any of F0 to 2047 is turned on by <b>[SET F]</b>, the annunciator numbers (F numbers) that are turned on in order are registered into D9125 to D9132.</li> <li>The F number turned off by <b>[RST F]</b> is erased from any of D9125 to D9132, and the F numbers stored after the erased F number are shifted to the preceding registerers.</li> </ul> <p>By executing <b>[LEDR]</b> 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.)</p> <p>When there are 8 annunciator detections, the 9th one is not stored into SD64 to SD71 even if detected.</p> <table border="1"> <tr> <td></td> <td>SET</td><td>SET</td><td>SET</td><td>RST</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td> </tr> <tr> <td></td> <td>F50</td><td>F25</td><td>F99</td><td>F25</td><td>F15</td><td>F70</td><td>F65</td><td>F38</td><td>F110</td><td>F151</td><td>F210</td><td>LEDR</td><td></td><td></td><td></td><td></td> </tr> </table> <table border="1"> <tr> <td>SD62</td> <td>0</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>99</td> </tr> <tr> <td>SD63</td> <td>0</td><td>1</td><td>2</td><td>3</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>8</td><td>8</td><td>8</td> </tr> <tr> <td>SD64</td> <td>0</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>99</td> </tr> <tr> <td>SD65</td> <td>0</td><td>0</td><td>25</td><td>25</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>15</td> </tr> <tr> <td>SD66</td> <td>0</td><td>0</td><td>0</td><td>99</td><td>0</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>70</td> </tr> <tr> <td>SD67</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>65</td> </tr> <tr> <td>SD68</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>65</td><td>65</td><td>65</td><td>65</td><td>65</td><td>38</td> </tr> <tr> <td>SD69</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>38</td><td>38</td><td>38</td><td>38</td><td>110</td> </tr> <tr> <td>SD70</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>110</td><td>110</td><td>110</td><td>151</td> </tr> <tr> <td>SD71</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>151</td><td>151</td><td>210</td> </tr> </table>		SET	SET	SET	RST	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET		F50	F25	F99	F25	F15	F70	F65	F38	F110	F151	F210	LEDR					SD62	0	50	50	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	8	SD64	0	50	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	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	
	SET	SET				SET	RST	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET																																																																																																																																																																						
	F50	F25				F99	F25	F15	F70	F65	F38	F110	F151	F210	LEDR																																																																																																																																																																										
SD62	0	50				50	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	8																																																																																																																																																																								
SD64	0	50				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	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																																																																																																																																																																												
D9126	SD1126	SD65																																																																																																																																																																																							
D9127	SD1127	SD66																																																																																																																																																																																							
D9128	SD1128	SD67																																																																																																																																																																																							
D9129	SD1129	SD68																																																																																																																																																																																							
D9130	SD1130	SD69																																																																																																																																																																																							
D9131	SD1131	SD70																																																																																																																																																																																							
D9132	SD1132	SD71																																																																																																																																																																																							

Special Register List

(10) Special register list dedicated for QnA

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																																																				
D9200	SD1200	—	ZNRD (LRDP for ACPU) processing results	0: Normal end 2: ZNRD instruction setting fault 3: Error at relevant station 4: Relevant station ZNRD execution disabled	Stores the execution result of the <b>ZNRD</b> (word device read) instruction <ul style="list-style-type: none"> <li><b>ZNRD</b> instruction setting fault: Faulty setting of the <b>ZNRD</b> instruction constant, source, and/or destination</li> <li>Corresponding station error: One of the stations is not communicating.</li> <li><b>ZNRD</b> cannot be executed in the corresponding station: The specified station is a remote I/O station.</li> </ul>	QnA																																																																																																				
D9201	SD1201	—	ZNWR (LWTP for ACPU) processing results	0: Normal end 2: ZNWR instruction setting fault 3: Error at relevant station 4: Relevant station ZNWR execution disabled	Stores the execution result of the <b>ZNWR</b> (word device write) instruction. <ul style="list-style-type: none"> <li><b>ZNWR</b> instruction setting fault: Faulty setting of the <b>ZNWR</b> instruction constant, source, and/or destination.</li> <li>Corresponding station error: One of the stations is not communicating.</li> <li><b>ZNWR</b> cannot be executed in the corresponding station: The specified station is a remote I/O station.</li> </ul>																																																																																																					
D9202	SD1202	—	Local station link type	Stores conditions for up to numbers 1 to 16	Stores whether the slave station corresponds to MELSECNET or MELSECNET11. <ul style="list-style-type: none"> <li>Bits corresponding to the MELSECNET 11 stations become "1."</li> <li>Bits corresponding to the MELSECNET stations or unconnected become "0."</li> </ul> <table border="1"> <thead> <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> </thead> <tbody> <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> </tbody> </table>		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
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																																																																																										
D9203	SD1203	—		Stores conditions for up to numbers 17 to 32	<ul style="list-style-type: none"> <li>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.</li> <li>If the own (master) station goes down, the contents before going down are also retained.</li> </ul>																																																																																																					
D9204	SD1204	—	Link status	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	Stores the present path status of the data link. <ul style="list-style-type: none"> <li>Data link in forward loop</li> </ul>  <ul style="list-style-type: none"> <li>Data link in reverse loop</li> </ul> 	QnA																																																																																																				

Special Register List (Continue)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																																																				
D9204	SD1204	—	Link status	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	<ul style="list-style-type: none"> <li>Loopback in forward/reverse loops</li> </ul>  <ul style="list-style-type: none"> <li>Loopback in forward loop only</li> </ul>  <ul style="list-style-type: none"> <li>Loopback in reverse loop only</li> </ul> 	QnA																																																																																																				
D9205	SD1205	—	Station implementing forward loopback	Station that implemented forward loopback	Stores the local or remote I/O station number at which loopback is being executed. 																																																																																																					
D9206	SD1206	—	Station implementing reverse loopback	Station that implemented reverse loopback	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. 																																																																																																					
D9210	SD1210	—	Number of retries	Stored as cumulative value	Stores the number of retry times due to transmission error. Count stops at maximum of "FFFFH". To return the value to "0", perform reset operation.	QnA																																																																																																				
D9211	SD1211	—	Number of times loop selected	Stored as cumulative value	Stores the number of times the loop line has been switched to reverse loop or loopback. Count stops at maximum of "FFFFH". To return the value to "0", perform reset operation.																																																																																																					
D9212	SD1212	—	Local station operation status	Stores conditions for up to numbers 1 to 16	Stores the local station numbers which are in STOP or PAUSE mode. <table border="1" data-bbox="861 1500 1292 1612"> <thead> <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> </thead> <tbody> <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> </tbody> </table>		Device number	Bit																b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1212	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	SD1213	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17	SD1214	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33	SD1215	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50
Device number	Bit																																																																																																									
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																										
SD1212	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1																																																																																										
SD1213	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17																																																																																										
SD1214	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33																																																																																										
SD1215	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49																																																																																										
D9213	SD1213	—	Local station operation status	Stores conditions for up to numbers 17 to 32																																																																																																						
D9214	SD1214	—	Local station operation status	Stores conditions for up to numbers 33 to 48	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 (40H)".																																																																																																					
D9215	SD1215	—	Local station operation status	Stores conditions for up to numbers 49 to 64																																																																																																						

Special Register List (Continue)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																																																						
D9216	SD1216	—	Local station error detect status	Stores conditions for up to numbers 1 to 16	Stores the local station numbers which are in error.  <table border="1"> <thead> <tr> <th>Device number</th> <th colspan="16">Bit</th> </tr> <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> </thead> <tbody> <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> </tbody> </table>	Device number	Bit																	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1216	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	SD1217	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17	SD1218	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33	SD1219	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49	QnA
Device number	Bit																																																																																																											
	b15	b14		b13		b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																										
SD1216	L16	L15		L14		L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1																																																																																										
SD1217	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17																																																																																												
SD1218	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33																																																																																												
SD1219	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49																																																																																												
D9217	SD1217	—	Stores conditions for up to numbers 17 to 32																																																																																																									
D9218	SD1218	—	Stores conditions for up to numbers 33 to 48	If a local station detects an error, the bit corresponding to the station number becomes "1".																																																																																																								
D9219	SD1219	—	Stores conditions for up to numbers 49 to 64	Example: When station 6 and 12 detect an error, b5 and b11 in SD1216 become "1", and when SD1216 is monitored, its value is "2080 (820H)".																																																																																																								
D9220	SD1220	—	Local station parameters non-conforming; remote I/O station I/O allocation error	Stores conditions for up to numbers 1 to 16	Stores the local station numbers which contain mismatched parameters or of remote station numbers for which incorrect I/O assignment has been made.  <table border="1"> <thead> <tr> <th>Device number</th> <th colspan="16">Bit</th> </tr> <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> </thead> <tbody> <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> </tbody> </table>	Device number	Bit																	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1220	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	SD1221	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17	SD1222	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33	SD1223	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49	QnA
Device number	Bit																																																																																																											
	b15	b14		b13		b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																										
SD1220	L16	L15		L14		L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1																																																																																										
SD1221	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17																																																																																												
SD1222	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33																																																																																												
SD1223	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49																																																																																												
D9221	SD1221	—	Stores conditions for up to numbers 17 to 32																																																																																																									
D9222	SD1222	—	Stores conditions for up to numbers 33 to 48	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".																																																																																																								
D9223	SD1223	—	Stores conditions for up to numbers 49 to 64	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)".																																																																																																								
D9224	SD1224	—	Local station and remote I/O station initial communications underway	Stores conditions for up to numbers 1 to 16	Stores the local or remote station numbers while they are communicating the initial data with their relevant master station.  <table border="1"> <thead> <tr> <th>Device number</th> <th colspan="16">Bit</th> </tr> <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> </thead> <tbody> <tr> <td>SD1224</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>SD1225</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>SD1226</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>SD1227</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> </tbody> </table>	Device number	Bit																	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1224	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	SD1225	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17	SD1226	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33	SD1227	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49	QnA
Device number	Bit																																																																																																											
	b15	b14		b13		b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																										
SD1224	L16	L15		L14		L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1																																																																																										
SD1225	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17																																																																																												
SD1226	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33																																																																																												
SD1227	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49																																																																																												
D9225	SD1225	—	Stores conditions for up to numbers 17 to 32																																																																																																									
D9226	SD1226	—	Stores conditions for up to numbers 33 to 48	The bit corresponding to the station number which is currently communicating the initial settings becomes "1".																																																																																																								
D9227	SD1227	—	Stores conditions for up to numbers 49 to 64	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)".																																																																																																								
D9228	SD1228	—	Local station and remote I/O station error	Stores conditions for up to numbers 1 to 16	Stores the local or remote station numbers which are in error.  <table border="1"> <thead> <tr> <th>Device number</th> <th colspan="16">Bit</th> </tr> <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> </thead> <tbody> <tr> <td>SD1228</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>SD1229</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>SD1230</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>SD1231</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> </tbody> </table>	Device number	Bit																	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1228	L16	L15	L14	L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1	SD1229	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17	SD1230	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33	SD1231	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49	QnA
Device number	Bit																																																																																																											
	b15	b14		b13		b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																										
SD1228	L16	L15		L14		L13	L12	L11	L10	L9	L8	L7	L6	L5	L4	L3	L2	L1																																																																																										
SD1229	L32	L31	L30	L29	L28	L27	L26	L25	L24	L23	L22	L21	L20	L19	L18	L17																																																																																												
SD1230	L48	L47	L46	L45	L44	L43	L42	L41	L40	L39	L38	L37	L36	L35	L34	L33																																																																																												
SD1231	L64	L63	L62	L61	L60	L59	L58	L57	L56	L55	L54	L53	L52	L51	L50	L49																																																																																												
D9229	SD1229	—	Stores conditions for up to numbers 17 to 32																																																																																																									
D9230	SD1230	—	Stores conditions for up to numbers 33 to 48	The bit corresponding to the station number with the error becomes "1".																																																																																																								
D9231	SD1231	—	Stores conditions for up to numbers 49 to 64	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)".																																																																																																								

Special Register List (Continue)

ACPU Special Conversion	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  <table border="1"> <thead> <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> </thead> <tbody> <tr> <td>SD1232</td> <td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td> </tr> <tr> <td>SD1233</td> <td>L/R16</td><td>L/R15</td><td>L/R14</td><td>L/R13</td><td>L/R12</td><td>L/R11</td><td>L/R10</td><td>L/R9</td><td>L/R8</td><td>L/R7</td><td>L/R6</td><td>L/R5</td><td>L/R4</td><td>L/R3</td><td>L/R2</td><td>L/R1</td> </tr> <tr> <td>SD1234</td> <td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td> </tr> <tr> <td>SD1235</td> <td>L/R24</td><td>L/R23</td><td>L/R22</td><td>L/R21</td><td>L/R20</td><td>L/R19</td><td>L/R18</td><td>L/R17</td><td>L/R16</td><td>L/R15</td><td>L/R14</td><td>L/R13</td><td>L/R12</td><td>L/R11</td><td>L/R10</td><td>L/R9</td> </tr> <tr> <td>SD1236</td> <td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td> </tr> <tr> <td>SD1237</td> <td>L/R32</td><td>L/R31</td><td>L/R30</td><td>L/R29</td><td>L/R28</td><td>L/R27</td><td>L/R26</td><td>L/R25</td><td>L/R24</td><td>L/R23</td><td>L/R22</td><td>L/R21</td><td>L/R20</td><td>L/R19</td><td>L/R18</td><td>L/R17</td> </tr> <tr> <td>SD1238</td> <td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td><td>R</td><td>F</td> </tr> <tr> <td>SD1239</td> <td>L/R40</td><td>L/R39</td><td>L/R38</td><td>L/R37</td><td>L/R36</td><td>L/R35</td><td>L/R34</td><td>L/R33</td><td>L/R32</td><td>L/R31</td><td>L/R30</td><td>L/R29</td><td>L/R28</td><td>L/R27</td><td>L/R26</td><td>L/R25</td> </tr> </tbody> </table>	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	SD1233	L/R16	L/R15	L/R14	L/R13	L/R12	L/R11	L/R10	L/R9	L/R8	L/R7	L/R6	L/R5	L/R4	L/R3	L/R2	L/R1	SD1234	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F	SD1235	L/R24	L/R23	L/R22	L/R21	L/R20	L/R19	L/R18	L/R17	L/R16	L/R15	L/R14	L/R13	L/R12	L/R11	L/R10	L/R9	SD1236	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F	SD1237	L/R32	L/R31	L/R30	L/R29	L/R28	L/R27	L/R26	L/R25	L/R24	L/R23	L/R22	L/R21	L/R20	L/R19	L/R18	L/R17	SD1238	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F	SD1239	L/R40	L/R39	L/R38	L/R37	L/R36	L/R35	L/R34	L/R33	L/R32	L/R31	L/R30	L/R29	L/R28	L/R27	L/R26	L/R25	QnA
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																																																																																																																																																													
SD1233	L/R16	L/R15		L/R14		L/R13	L/R12	L/R11	L/R10	L/R9	L/R8	L/R7	L/R6	L/R5	L/R4	L/R3	L/R2	L/R1																																																																																																																																																													
SD1234	R	F		R		F	R	F	R	F	R	F	R	F	R	F	R	F																																																																																																																																																													
SD1235	L/R24	L/R23		L/R22		L/R21	L/R20	L/R19	L/R18	L/R17	L/R16	L/R15	L/R14	L/R13	L/R12	L/R11	L/R10	L/R9																																																																																																																																																													
SD1236	R	F		R		F	R	F	R	F	R	F	R	F	R	F	R	F																																																																																																																																																													
SD1237	L/R32	L/R31		L/R30		L/R29	L/R28	L/R27	L/R26	L/R25	L/R24	L/R23	L/R22	L/R21	L/R20	L/R19	L/R18	L/R17																																																																																																																																																													
SD1238	R	F	R	F	R	F	R	F	R	F	R	F	R	F	R	F																																																																																																																																																															
SD1239	L/R40	L/R39	L/R38	L/R37	L/R36	L/R35	L/R34	L/R33	L/R32	L/R31	L/R30	L/R29	L/R28	L/R27	L/R26	L/R25																																																																																																																																																															
D9233	SD1233	—	Stores conditions for up to numbers 9 to 16																																																																																																																																																																												
D9234	SD1234	—	Stores conditions for up to numbers 17 to 24																																																																																																																																																																												
D9235	SD1235	—	Stores conditions for up to numbers 25 to 32																																																																																																																																																																												
D9236	SD1236	—	Stores conditions for up to numbers 33 to 40																																																																																																																																																																												
D9237	SD1237	—	Stores conditions for up to numbers 41 to 48																																																																																																																																																																												
D9238	SD1238	—	Stores conditions for up to numbers 49 to 56																																																																																																																																																																												
D9239	SD1239	—	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.	QnA																																																																																																																																																																									
D9241	SD1241	—	Local station link type	Stores conditions for up to numbers 33 to 48	Stores whether the slave station corresponds to MELSECNET or MELSECNET11. <ul style="list-style-type: none"> <li>• Bits corresponding to the MELSECNET 11 stations become "1."</li> <li>• Bits corresponding to the MELSECNET stations or unconnected become "0."</li> </ul> <table border="1"> <thead> <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> </thead> <tbody> <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> </tbody> </table>		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																																																																				
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																																																																																																																																																															
D9242	SD1242	—	Stores conditions for up to numbers 49 to 64	Stores conditions for up to numbers 49 to 64	<ul style="list-style-type: none"> <li>• 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.</li> <li>• If the own (master) station goes down, the contents before going down are also retained.</li> </ul>																																																																																																																																																																										
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	—	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.																																																																																																																																																																										

Special Register List (Continue)

ACPU Special Conversion	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9248	SD1248	—	Local station operation status	Stores conditions for up to numbers 1 to 16	Stores the local station number which is in STOP or PAUSE mode. 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)".	QnA
D9249	SD1249	Stores conditions for up to numbers 17 to 32				
D9250	SD1250	Stores conditions for up to numbers 33 to 48				
D9251	SD1251	Stores conditions for up to numbers 49 to 64				
D9252	SD1252	—	Local station error conditions	Stores conditions for up to numbers 1 to 16	Stores the local station number other than the host, which is in error. 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)".	QnA
D9253	SD1253	Stores conditions for up to numbers 17 to 32				
D9254	SD1254	Stores conditions for up to numbers 33 to 48				
D9255	SD1255	Stores conditions for up to numbers 49 to 64				

(11) Fuse blown module

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU	Corresponding CPU
SD1300 SD1301 SD1302 SD1303 SD1304 SD1305 SD1306 SD1307 SD1308 SD1309 to SD1330 SD1331	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	<ul style="list-style-type: none"> <li>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.)</li> <li>Also detects blown fuse condition at remote station output modules</li> </ul> <p>Indicates a blown fuse</p> <ul style="list-style-type: none"> <li>Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation</li> </ul>	S (Error)	D9100 D9101 D9102 D9103 D9104 D9105 D9106 D9107 New New to New New	○+Rem
SD1350 to SD1381	External power supply disconnected module (For future expansion)	Bit pattern in units of 16 points, indicating the modules whose external power supply has been disconnected 0: External power supply disconnected 1: External power supply is not disconnected	<ul style="list-style-type: none"> <li>The module number (in units of 16 points) whose external power supply has been disconnected is input as a bit pattern. (If the module numbers are set by parameter, the parameter-set numbers are used.)</li> </ul> <p>Indicates a blown fuse</p>	S (Error)	New	QCPU Remote

Special Register List (Continue)

(12) I/O module verification

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU																																																																				
SD1400 SD1401 SD1402 SD1403 SD1404 SD1405 SD1406 SD1407 SD1408 SD1409 to SD1430 SD1431	I/O module verification 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	<ul style="list-style-type: none"> <li>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.)</li> <li>Also detects I/O module information</li> </ul> <div style="text-align: center;"> <table border="1" style="border-collapse: collapse; margin: auto;"> <tr> <td></td> <td>b15</td><td>b14</td><td>b13</td><td>b12</td><td>b11</td><td>b10</td><td>b9</td><td>b8</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td> </tr> <tr> <td>D9116</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>1 (1/16)</td> </tr> <tr> <td>D9117</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (1/8)</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>D9123</td> <td>0</td><td>1 (1/16)</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> <p style="text-align: center;">↑ Indicates an I/O module verification error</p> </div> <ul style="list-style-type: none"> <li>Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation</li> </ul>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	D9116	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1 (1/16)	D9117	0	0	0	0	0	0	1 (1/8)	0	0	0	0	0	0	0	0	0	D9123	0	1 (1/16)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	S (Error)	D9116 D9117 D9118 D9119 D9120 D9121 D9122 D9123 New New to New New	○+Rem
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																										
D9116	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1 (1/16)																																																										
D9117	0	0	0	0	0	0	1 (1/8)	0	0	0	0	0	0	0	0	0																																																										
D9123	0	1 (1/16)	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																										

(13) For redundant systems (Host system CPU information \*1) for Q4AR only

SD1510 to SD1599 are only valid for redundant systems. They are all set to 0 for standalone systems.

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1500 SD1501	Basic period	Basic period tome	<ul style="list-style-type: none"> <li>Set the basic period (1 second units) use for the process control instruction using floating point data. Floating points data = <input type="text" value="SD1501"/> <input type="text" value="SD1500"/></li> </ul>	U	New	Q4AR
SD1502	Process control instruction detail error code	Process control instruction detail error code	<ul style="list-style-type: none"> <li>Shows the detailed error contents for the error that occurred in the process control instruction</li> </ul>	S (Error occurrence)	New	
SD1503	Process control instruction generated error location	Process control instruction generated error location	<ul style="list-style-type: none"> <li>Shows the error process block that occurred in the process control instruction.</li> </ul>	S (Error occurrence)	New	
SD1512	Operation mode during CPU module start up	Hot start switch power out time	<ul style="list-style-type: none"> <li>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.</li> </ul>	S (Initial)	New	
SD1590	Switch request network No.	Request origin network No.	<ul style="list-style-type: none"> <li>Stores the request origin at work No. when the SM1590 is turned on.</li> </ul>	S (Error occurrence)	New	

\* 1 Stores the host system CPU information.

(14) For redundant systems (Other system CPU information \*1) for Q4AR only  
 SD1600 to SD1659 is only valid during the back up mode for redundant systems, and refresh cannot be done when in the separate mode. When a standalone system SD1600 to SD1699 are all 0.

Special Register List

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU SD□□*2	Corresponding CPU
SD1600	Diagnosis error	Diagnosis error No.	<ul style="list-style-type: none"> <li>Stores as BIN code the error No. of the error that occurred during the other system CPU module diagnosis.</li> <li>Stores the latest error currently occurring.</li> </ul>	S (Each END)	New	Q4AR
SD1601 SD1602 SD1603	Diagnosis error occurrence time	Diagnosis error occurrence time	<ul style="list-style-type: none"> <li>SD1600 stores the updated date and time.</li> <li>Stores each of the BCD two digits.</li> <li>Refer to SD1 to SD3 for the storage status. (SD1 →SD1601, SD2 →SD1602, SD3 →SD1603)</li> </ul>	S (Each END)	New	
SD1604	Error information classification	Error information classification	<ul style="list-style-type: none"> <li>Stores the error comment information/individual information classification code.</li> <li>Refer to SD4 for the storage status.</li> </ul>	S (Each END)	New	
SD1605 SD1606 SD1607 SD1608 SD1609 SD1610 SD1611 SD1612 SD1613 SD1614 SD1615	Error common information	Error common information	<ul style="list-style-type: none"> <li>Stores the common information for the error code.</li> <li>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)</li> </ul>	S (Each END)	New	
SD1616 SD1617 SD1618 SD1619 SD1620 SD1621 SD1622 SD1623 SD1624 SD1625 SD1626	Error individual information	Error individual information	<ul style="list-style-type: none"> <li>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)</li> </ul>	S (Each END)	New	
SD1650	Switch status	CPU module switch status	<ul style="list-style-type: none"> <li>Stores the CPU module switch status.</li> <li>Refer to SD200 for the storage status. (SD1650 →SD200)</li> </ul>	S (Each END)	New	
SD1651	LED status	CPU module - LED status	<ul style="list-style-type: none"> <li>Stores the CPU module's LED status.</li> <li>Shows 0 when turned off, 1 when turned on, and 2 when flicking.</li> <li>Refer to SD201 for the storage status. (SD1651 →SD201)</li> </ul>	S (Each END)	New	
SD1653	CPU module operation status	CPU module operation status	<ul style="list-style-type: none"> <li>Stores the CPU module operation status. Refer to SD203 for the storage status. (SD1653 →SD203)</li> </ul>	S (Each END)	New	

\* 1 Stores other system CPU module self-diagnosis information and system information.

\* 2 Shows the special register (SD□□) for the host system CPU module.

(15) For redundant systems (Trucking) for Q4AR only  
 SD1700 to SD1799 is valid only for redundant systems.  
 These are all 0 for standalone systems.

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU SD□□*2	Corresponding CPU
SD1700	Trucking error detection count	Trucking error detection count	<ul style="list-style-type: none"> <li>Make it the trucking error detection +1.</li> </ul>	Error occurrence	New	Q4AR

\* 2 Shows the special register (SD□□) for the host system CPU.



APPENDIX 3 List of Interrupt Pointer Nos. and Interrupt Factors

I No.	Interrupt Factors	Priority Ranking	I No.	Interrupt factors	Priority Ranking		
I0	QI60/A1SI61 interrupt module factor	1st point	237	I32 * 3	Errors that stop operation	1	
I1		2nd point	238	I33	Empty	—	
I2		3rd point	239	I34	UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	2	
I3		4th point	240				
I4		5th point	241	I35	Error factor * 4	3	
I5		6th point	242				
I6		7th point	243				
I7		8th point	244				
I8		9th point	245	I36	OPERATION ERROR SFCE OPE. ERROR SFCE ECE. ERROR EX. POWER OFF	4	
I9		10th point	246				
I10		11th point	247	I37	ICM. OPE ERROR FILE OPE. ERROR	5	
I11		12th point	248				
I12		13th point	249	I38	Empty	—	
I13		14th point	250				
I14		15th point	251	I39	PRG. TIME OVER CHK instruction execution Anunciator detection	6	
I15		16th point	252				
I16	Sequence start generator module factor * 1	1st point	224	I40 to I48	Empty	—	
I17		2nd point	225	I49	Internal timer factor	0, 2 to 1.0mg * 5	* 7
I18		3rd point	226				
I19		4th point	227				
I20		5th point	228				
I21		6th point	229				
I22		7th point	230				
I23		8th point	231				
I24		9th point	232				
I25		10th point	234				
I26		11th point	235				
I27		12th point	236				
I28	Internal timer factor * 2	100ms	256				
I29		40ms	255				
I30		20ms	254				
I31		10ms	253				

**REMARK**

- \*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 internal times shown are the default setting times.  
These times can be designated in 0.5 ms units through a 0.5 to 1000 ms range set at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.
- \*3 : When an error interruption with "I32 (error that stops operation)" occurs, the High Performance model QCPU is not stopped until I32 processing is completed.
- \*4 : Execution of error interruptions is prohibited for the interrupt pointer Nos. I32 to I39 when the power is turned on and during a High Performance model QCPU reset. When using interrupt pointer Nos. I32 to I39, set the interruption permitted status by using the IMASK instruction.
- \*5 : 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 0.2 to 1.0ms in 0.1ms increments.
- \*6 : To use the intelligent function module interrupt, the intelligent function module setting (interrupt points setting) is required at the "PLC system" tab screen in the "(PLC) Parameter" dialog box.  
(For the interrupts from the intelligent function module, see Section 8.2.1.)

**POINT**

\*7: When you have set I49 in the PLC parameters, do not execute other interrupt programs (I0 to I48, I50 to I255) and fixed-cycle programs. If any fixed-cycle program or like is run, the interrupt program using I49 cannot be executed at the preset interrupt cycle intervals.

APPENDIX 4 Enhancement of the High Performance Model QCPU Functions

The High Performance model QCPU has been updated to add functions and change the specifications.

The functions and specifications that can be used with the High Performance model QCPU change depending on the function version and serial No.

APPENDIX 4.1 Specification Comparison

Serial No. of CPU Module		Function Version A		Function Version B		
		02091 or earlier	02092 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		×	×	×	×	○

○: Available, ×: N/A

APPENDIX 4.2 Function comparison

Serial No. of CPU Module		Function Version A		Function Version B				
		02091 or earlier	02092 or later	—	03051 or later	04012 or later	04122 or later	05032 or later
Automatic write to standard ROM		×	○	○	○	○	○	○
Enforced ON/OFF for external I/O		×	○	○	○	○	○	○
Remote password setting		×	○	○	○	○	○	○
Compatibility with MELSECNET/H remote I/O network		×	○	○	○	○	○	○
Interrupt module (QI60) compatibility		×	○	○	○	○	○	○
Compatibility with the multiple CPU system		×	×	○	○	○	○	○
Installation of PC CPU module into the multiple CPU system		×	×	×	○	○	○	○
High speed interrupt		×	×	×	×	○	○	○
Compatibility with index modification for module designation of dedicated instruction		×	×	×	×	○	○	○
Selection of refresh item for COM instruction		×	×	×	×	○	○	○
SFC program online batch change		×	×	×	×	×	○	○
File memory capacity change		×	×	×	×	×	○	○
CC-Link remote network additional mode		×	×	×	×	×	×	○
Incomplete derivative PID operation function		×	×	×	×	×	×	○
Floating-point comparison instruction speedup		×	×	×	×	×	×	○

○: Available, ×: N/A

For function details, refer to the High Performance model QCPU (Q mode) User's Manual (Function Explanation, Program Fundamentals).

## APPENDIX 4.3 Added functions and the corresponding GX Developer versions

Added Function	SW4D5C-GPPW-E SW5D5C-GPPW-E	GX Developer				
		Version 6	Version 7	Version 7.10L	Version 8	Version 8.03D
Automatic write to standard ROM	×	○	○	○	○	○
External I/O can be turned ON/OFF forcibly	×	○	○	○	○	○
Remote password setting	×	○	○	○	○	○
Compatibility with MELSECNET/H remote I/O network	×	○	○	○	○	○
Interrupt module (QI60) compatibility	×	○	○	○	○	○
Compatibility with the multiple CPU system	×	○	○	○	○	○
Installation of PC CPU module into the multiple CPU system	×	×	○	○	○	○
High speed interrupt	×	×	×	○	○	○
Compatibility with index modification for module designation of dedicated instruction	—	—	—	—	—	—
Selection of refresh item for COM instruction	—	—	—	—	—	—
SFC program online batch change	×	×	×	×	○	○
File memory capacity change	×	×	×	×	○	○
CC-Link remote network additional mode	×	×	×	×	×	○
Incomplete derivative PID operation function	×	×	×	×	×	○
Floating-point comparison instruction speedup	—	—	—	—	—	—

○: Available, ×: N/A, — : Function not related to GX Developer

## APPENDIX 5 Transportation Precautions

When transporting lithium batteries, make sure to treat them based on the transport regulations.

## APPENDIX 5.1 Controlled models

The batteries for the High Performance model QCPU (including memory cards) are classified as follows:

Product name	Model	Product supply status	Classification for transportation
Q series battery	Q7BAT	Lithium battery	Dangerous goods
Q series battery	Q7BAT-SET	Lithium battery with holder	
Q series battery	Q6BAT	Lithium battery	Non-dangerous goods
Q series memory card battery	Q2MEM-BAT	Lithium coin battery	
Q series memory card	Q2MEM-1MBS Q2MEM-2MBS	Packed with lithium coin battery (Q2MEM-BAT)	

## APPENDIX 5.2 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.

## INDEX

- [A]
- Accuracy of initial scan time ..... 4-16
  - Accuracy of scan time ..... 4-18
  - Annunciator (F) ..... 10-12
  - ASCII code ..... 4-51
  - ATA card ..... 6-11
  - Auto mode ..... 5- 3
  - Automatic write to standard ROM ..... 6-14
- [B]
- B (Link relay) ..... 10-17
  - Base mode ..... 5- 3
  - BCD (Binary coded decimal) ..... 4-47
  - BIN (Binary code) ..... 4-45
  - BL (SFC block device) ..... 10-58
  - Boot Run ..... 6-17
- [C]
- C (Counter) ..... 10-24
  - Catalog PLC memory ..... 6-18
  - Character string ..... 4-51
  - Clock function ..... 7- 9
    - Precision ..... 7-11
  - Common pointer ..... 10-54
  - Concept of I/O assignment ..... 5- 8
  - Concept of I/O assignment using
    - GX Developer ..... 5-12
  - Constant scan ..... 7- 2
  - Constants ..... 10-61
  - Counter (C) ..... 10-24
    - Count processing ..... 10-24
    - Maximum counting speed ..... 10-25
- [D]
- D (Data register) ..... 10-28
  - Data register (D) ..... 10-28
  - Data stored on the memory card ..... 6- 4
  - Decimal constants (K) ..... 10-61
  - Device initial value ..... 10- 69
  - Device list ..... 10- 1
  - Direct access input ..... 10- 6
  - Direct access output ..... 10- 9
  - Direct mode ..... 4-41
  - Drive Number. .... 6- 5
- Duty ..... 10-25
  - DX (Direct access input) ..... 10- 6
  - DY (Direct access output) ..... 10- 9
- [E]
- E (Real numbers) ..... 10-62
  - Edge relay(V) ..... 10-16
  - END processing ..... 4-34
  - Enforced ON/OFF ..... 7-31
  - Enforced ON/OFF for external I/O ..... 1- 4
  - Execute type ..... 4-10
  - Execution time measurement ..... 7-39
  - Execution time of the low speed execution type program ..... 4-19
  - Extension ..... 6- 2, 6- 4
- [F]
- F (Annunciator) ..... 10-12
  - Failure history ..... 7-64
  - FD (Function register) ..... 10-31
  - File register ..... 10-43
    - Access method ..... 10-44
    - Designation method ..... 10-49
    - Registering ..... 10-45
  - File size ..... 6-18
  - Fixed scan execution type program ..... 4-31
  - Flash card ..... 6-12
  - Floating decimal point data ..... 4-48
  - Function device (FX, FY, FD) ..... 10-31
  - Function version ..... 13- 5, 14- 4
  - FX (Function input) ..... 10-31
  - FY (Function output) ..... 10-31
- [G]
- Global device ..... 10-63
  - GX Configurator ..... 8- 2, 14-12
  - GX Developer ..... A-18
- [H]
- H (Hexadecimal constants) ..... 10-61
  - HEX (Hexadecimal) ..... 4-46
  - Hexadecimal constants (H) ..... 10-61
  - high speed interrupt function ..... 7-81
  - High speed retentive timer (ST) ..... 10-21

High speed timer (T) ..... 10-20

[I]

I (Interrupt pointer)..... 10-56

I/O No. designation device (Un)..... 10-59

Index register (Z) ..... 10-39

Initial execution monitor time ..... 4-16

Initial execution type program ..... 4-15

Initial scan time..... 4-16

Input response time..... 7-21

Intelligent function module device (U □ \G □ ). 10-38

Internal relay (M) ..... 10-10

Internal system device ..... 10-31

Internal user device ..... 10- 3

Interrupt module ..... 5-11

Interrupt pointer (I)..... 10-56

Interrupt program..... 4- 6

[J]

J (Network designation device)..... 10-58

J □ \B □ (Link relay) ..... 10-35

J □ \SB □ (Link special relay) ..... 10-35

J □ \SW □ (Link special register) ..... 10-35

J □ \W □ (Link register) ..... 10-35

J □ \X □ (Link input) ..... 10-35

J □ \Y □ (Link output) ..... 10-35

[K]

K (Decimal constants) ..... 10-61

[L]

L (Latch relay)..... 10-11

Latch function ..... 7- 5

Latch relay (L)..... 10-11

LED display ..... 7-74

Link direct device..... 10-35

Link register (W) ..... 10-29

Link relay (B) ..... 10-17

List of Interrupt factors ..... 10-57

App-55

Local device..... 10-63

Low speed END processing ..... 4-23

Low speed execution monitor time ..... 4-24

Low speed execution type program ..... 4-19

Low speed retentive timer (ST)..... 10-21

Low speed scan timer ..... 4-23

Low speed timer (T) ..... 10-19

[M]

M (Internal relay)..... 10-10

Macro instruction argument device (VD)..... 10-60

Main routine program..... 4- 3

Memory card ..... 6-11

Monitor condition setting..... 7-25

Monitoring the local devices ..... 7-30

[N]

N (Nesting) ..... 10-52

[O]

Output (Y)..... 10- 8

[P]

P (Pointer) ..... 10-53

Password..... 7-65

PLOW instruction ..... 4-14

POFF instruction ..... 4-14

Pointer (P) ..... 10-53

Precautions for the use of device initial values .. 10-71

Precautions when using timers ..... 10-23

Priority of LED ..... 7-76

Procedure for using device initial values..... 10-70

Processing at annunciator OFF..... 10-14

Processing at annunciator ON ..... 10-12

Program construction..... 1- 6

Program execute type..... 4-10

Program memory ..... 6- 6

Program monitor list..... 7-39

PSCAN instruction ..... 4-14

PSTOP instruction ..... 4-14

Purpose of I/O assignment..... 5-11

Purpose of I/O assignment using

GX Developer ..... 5-11

[Q]

QCPU ..... A-18

QI60 ..... 2- 4, 7-23

QnCPU ..... A-18

QnHCPU ..... A-18

[R]

R (File register) ..... 10-43

Reading from the time data ..... 7- 9

Real numbers (E)..... 4-48

10-62

Refresh input..... 10- 6

Refresh mode.....	4-38
Refresh output.....	10- 9
Remote latch clear .....	7-19
Remote operation.....	7-12
Remote PAUSE.....	7-15
Remote password .....	7- 1, 7-67
Remote RESET.....	7-17
Remote RUN/STOP .....	7-12
Remote station I/O number.....	5-10
Retentive timer (OUT ST [□]).....	10-21
RUN status .....	4-35

## [S]

S (Step relay).....	10-18
SB (Link special relay) .....	10-18
Scan execution type program.....	4-17
Scan time.....	4-18
SD (Special register) .....	10-34
SD415 (2n-ms clock setting).....	8- 9
SD520, SD521 (Scan time: present value) ..	4-18
SD522, SD523 (Initial scan time) .....	4-16
SD524, SD525 (Scan time: Maximum value).....	4-18
SD526, SD527 (Scan time: Minimum value).....	4-18
SD528, SD529 (Low speed scan time: Present value) .....	4-24
SD530, SD531 (Low speed scan time: Initial value) .....	4-24
SD532, SD533 (Low speed scan time: Minimum value) .....	4-24
SD534, SD535 (Low speed scan time: Maximum value).....	4-24
Self-diagnosis function .....	7-59
Sequence program.....	4- 1
Serial No. ....	1- 1
Setting range in the internal user device ....	10- 3
Setting the number of stages.....	5- 2
SFC block device (BL) .....	10-58
SFC transition device (TR).....	10-58
Single precision floating decimal point data .....	4-48
Size (File capacity) .....	6- 2
SM (Special relay).....	10-33
SM415 (2n-ms clock) .....	8- 9
Link special register (SW).....	10-30
Link special relay (SB) .....	10-18
Special register (SD) .....	10-34
Special relay (SM).....	10-33
SRAM card .....	6-11

ST (Retentive timer: OUT ST [□]) .....	10-21
Stand-by type program .....	4-25
Standard RAM .....	6- 9
Standard RAM memory capacity .....	6- 9
Standard ROM.....	6- 8
Step relay (S) .....	10-18
Sub-routine program.....	4- 4
SW (Link special register).....	10-30
Switch setting of intelligent function module .	7-21
System protect .....	7-65

## [T]

T (Timer).....	10-19
Timer (T).....	10-19
Accuracy.....	10-22
Processing.....	10-22
TR (SFC transition device) .....	10-58

## [U]

U (I/O No. designation device) .....	10-59
U [□] \G [□] (Intelligent function module device)...	10-38
User memory.....	6- 3

## [V]

V (Edge relay) .....	10-16
VD (Macro instruction argument device).....	10-60

## [W]

W (Link register).....	10-29
Watch dog timer.....	7-57
WDT (Watch dog timer).....	7-57
Write during RUN.....	7-35, 7-37, 7-55
Writing to the time data .....	7- 9

## [X]

X (Input).....	10- 5
----------------	-------

## [Y]

Y (Output).....	10- 8
-----------------	-------

## [Z]

Z (Index register).....	10-39
ZR (Serial number access format of file register) .....	10-49

# WARRANTY

Please confirm the following product warranty details before starting use.

## 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 dealer or Mitsubishi Service Company. Note that if repairs are required at a site overseas, on a detached island or remote place, expenses to dispatch an engineer shall be charged for.

### [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 status, 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 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 possible 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 chance loss and secondary loss from warranty liability

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation to damages caused by any cause found not to be the responsibility of Mitsubishi, chance losses, lost profits incurred to the user by Failures of Mitsubishi products, damages and secondary damages caused from special reasons regardless of Mitsubishi's expectations, compensation for accidents, and compensation for damages to products other than Mitsubishi products and other duties.

## 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 logic 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 logic controller device, and that backup and fail-safe functions are systematically provided outside of the device for any problem or fault.
- (2) The Mitsubishi general-purpose programmable logic 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 National Defense purposes shall be excluded from the programmable logic controller applications.

Note that even with these applications, if the user approves that the application is to be limited and a special quality is not required, application shall be possible.

When considering use in aircraft, medical applications, railways, incineration and fuel devices, manned transport devices, equipment for recreation and amusement, and safety devices, in which human life or assets could be greatly affected and for which a particularly high reliability is required in terms of safety and control system, please consult with Mitsubishi and discuss the required specifications.



Microsoft Windows, Microsoft Windows NT are registered trademarks of Microsoft Corporation in the United States and other countries.

Pentium is a registered trademark of Intel Corporation in the United States and other countries.

Ethernet is a registered trademark of Xerox. Co., Ltd in the United States.

Other company and product names herein are either trademarks or registered trademarks of their respective owners.

**HEADQUARTERS**

MITSUBISHI ELECTRIC EUROPE  
EUROPE B.V.  
German Branch  
Gothaer Straße 8  
**D-40880 Ratingen**  
Phone: +49 (0) 2102 / 486-0  
Fax: +49 (0) 2102 / 486-1120  
e mail: megfamail@meg.mee.com

MITSUBISHI ELECTRIC FRANCE  
EUROPE B.V.  
French Branch  
25, Boulevard des Bouvets  
**F-92741 Nanterre Cedex**  
Phone: +33 1 55 68 55 68  
Fax: +33 1 55 68 56 85  
e mail: factory.automation@fra.mee.com

MITSUBISHI ELECTRIC IRELAND  
EUROPE B.V.  
Irish Branch  
Westgate Business Park, Ballymount  
**IRL-Dublin 24**  
Phone: +353 (0) 1 / 419 88 00  
Fax: +353 (0) 1 / 419 88 90  
e mail: sales.info@meir.mee.com

MITSUBISHI ELECTRIC ITALY  
EUROPE B.V.  
Italian Branch  
Via Paracelso 12  
**I-20041 Agrate Brianza (MI)**  
Phone: +39 039 6053 1  
Fax: +39 039 6053 312  
e mail: factory.automation@it.mee.com

MITSUBISHI ELECTRIC SPAIN  
EUROPE B.V.  
Spanish Branch  
Carretera de Rubí 76-80  
**E-08190 Sant Cugat del Vallés**  
Phone: +34 9 3 / 565 3131  
Fax: +34 9 3 / 589 2948  
e mail: industrial@sp.mee.com

MITSUBISHI ELECTRIC UK  
EUROPE B.V.  
UK Branch  
Travellers Lane  
**GB-Hatfield Herts. AL10 8 XB**  
Phone: +44 (0) 1707 / 27 61 00  
Fax: +44 (0) 1707 / 27 86 95  
e mail: automation@meuk.mee.com

MITSUBISHI ELECTRIC JAPAN  
CORPORATION  
Office Tower "Z" 14 F  
8-12,1 chome, Harumi Chuo-Ku  
**Tokyo 104-6212**  
Phone: +81 3 6221 6060  
Fax: +81 3 6221 6075

MITSUBISHI ELECTRIC AUTOMATION USA  
500 Corporate Woods Parkway  
**Vernon Hills, IL 60061**  
Phone: +1 847 / 478 21 00  
Fax: +1 847 / 478 22 83

**MIDDLE EAST REPRESENTATIVE**

TEXEL Electronics Ltd. ISRAEL  
Box 6272  
**IL-42160 Netanya**  
Phone: +972 (0) 9 / 863 08 91  
Fax: +972 (0) 9 / 885 24 30  
e mail: texel\_me@netvision.net.il

**EUROPEAN REPRESENTATIVES**

GEVA AUSTRIA  
Wiener Straße 89  
**AT-2500 Baden**  
Phone: +43 (0) 2252 / 85 55 20  
Fax: +43 (0) 2252 / 488 60  
e mail: office@geva.at

TEHNIKON BELARUS  
Oktjabskaya 16/5, Ap 704  
**BY-220030 Minsk**  
Phone: +375 (0)17 / 22 75 704  
Fax: +375 (0)17 / 22 76 669  
e mail: tehnikon@belsonet.net

Getronics b.v. BELGIUM  
Control Systems  
Pontbeeklaan 43  
**B-1731 Asse-Zellik**  
Phone: +32 (0) 2 / 467 17 51  
Fax: +32 (0) 2 / 467 17 45  
e mail: infoautomation@getronics.com

TELECON CO. BULGARIA  
4, A. Ljapchev Blvd.  
**BG-1756 Sofia**  
Phone: +359 (0) 2 / 97 44 05 8  
Fax: +359 (0) 2 / 97 44 06 1  
e mail: —

INEA CR d.o.o. CROATIA  
Drvinje 63  
**HR-10000 Zagreb**  
Phone: +385 (0) 1 / 36 67 140  
Fax: +385 (0) 1 / 36 67 140  
e mail: —

AutoCont CZECH REPUBLIC  
Control Systems s.r.o.  
Nemocnicni 12  
**CZ-702 00 Ostrava 2**  
Phone: +420 59 / 6152 111  
Fax: +420 59 / 6152 562  
e mail: consys@autocont.cz

louis poulsen DENMARK  
industri & automation  
Geminivej 32  
**DK-2670 Greve**  
Phone: +45 (0) 70 / 10 15 35  
Fax: +45 (0) 43 / 95 95 91  
e mail: lpia@lpmail.com

UTU Elektrotehnika AS ESTONIA  
Pärnu mnt.160i  
**EE-11317 Tallinn**  
Phone: +372 (0) 6 / 51 72 80  
Fax: +372 (0) 6 / 51 72 88  
e mail: utu@utu.ee

Beijer Electronics OY FINLAND  
Ansatie 6a  
**FIN-01740 Vantaa**  
Phone: +358 (0) 9 / 886 77 500  
Fax: +358 (0) 9 / 886 77 555  
e mail: info@beijer.fi

UTECO A.B.E.E. GREECE  
5, Mavrogenous Str.  
**GR-18542 Piraeus**  
Phone: +302 (0) 10 / 42 10 050  
Fax: +302 (0) 10 / 42 12 033  
e mail: sales@uteco.gr

Meltrade Automatika Kft. HUNGARY  
55, Harmat St.  
**HU-1105 Budapest**  
Phone: +36 (0)1 / 2605 602  
Fax: +36 (0)1 / 2605 602  
e mail: office@meltrade.hu

SIA POWEL LATVIA  
Lienes iela 28  
**LV-1009 Riga**  
Phone: +371 784 / 22 80  
Fax: +371 784 / 22 81  
e mail: utu@utu.lv

**EUROPEAN REPRESENTATIVES**

UAB UTU POWEL LITHUANIA  
Savanoriu pr. 187  
**LT-2053 Vilnius**  
Phone: +370 (0) 52323-101  
Fax: +370 (0) 52322-980  
e mail: powel@utu.lt

INTEHSIS SRL MOLDOVA  
Cuza-Voda 36/1-81  
**MD-2061 Chisinau**  
Phone: +373 (0)2 / 562 263  
Fax: +373 (0)2 / 562 263  
e mail: intehsis@mdl.net

Getronics b.v. NETHERLANDS  
Control Systems  
Donauweg 2 B  
**NL-1043 AJ Amsterdam**  
Phone: +31 (0) 20 / 587 67 00  
Fax: +31 (0) 20 / 587 68 39  
e mail: info.gia@getronics.com

Beijer Electronics AS NORWAY  
Teglverksveien 1  
**N-3002 Drammen**  
Phone: +47 (0) 32 / 24 30 00  
Fax: +47 (0) 32 / 84 85 77  
e mail: info@beijer.no

MPL Technology Sp. z o.o. POLAND  
ul. Sliczna 36  
**PL-31-444 Kraków**  
Phone: +48 (0) 12 / 632 28 85  
Fax: +48 (0) 12 / 632 47 82  
e mail: krakow@mpl.pl

Sirius Trading & Services srl ROMANIA  
Str. Biharia No. 67-77  
**RO-013981 Bucuresti 1**  
Phone: +40 (0) 21 / 201 1146  
Fax: +40 (0) 21 / 201 1148  
e mail: sirius@siriustrading.ro

INEA d.o.o. SLOVENIA  
Stegne 11  
**SI-1000 Ljubljana**  
Phone: +386 (0) 1-513 8100  
Fax: +386 (0) 1-513 8170  
e mail: inea@inea.si

Beijer Electronics AB SWEDEN  
Box 426  
**S-20124 Malmö**  
Phone: +46 (0) 40 / 35 86 00  
Fax: +46 (0) 40 / 35 86 02  
e mail: info@beijer.se

ECONOTEC AG SWITZERLAND  
Postfach 282  
**CH-8309 Nürensdorf**  
Phone: +41 (0) 1 / 838 48 11  
Fax: +41 (0) 1 / 838 48 12  
e mail: info@econotec.ch

GTS TURKEY  
Darülaceze Cad. No. 43 Kat. 2  
**TR-80270 Okmeydani-Istanbul**  
Phone: +90 (0) 212 / 320 1640  
Fax: +90 (0) 212 / 320 1649  
e mail: gts@turk.net

CSC Automation Ltd. UKRAINE  
15, M. Raskova St., Fl. 10, Office 1010  
**UA-02002 Kiev**  
Phone: +380 (0) 44 / 238-83-16  
Fax: +380 (0) 44 / 238-83-17  
e mail: csc-a@csc-a.kiev.ua

**EUROPEAN REPRESENTATIVES**

Avtomatika Sever Ltd. RUSSIA  
Lva Tolstogo St. 7, Off. 311  
**RU-197376 St Petersburg**  
Phone: +7 812 / 11 83 238  
Fax: +7 812 / 11 83 239  
e mail: as@avtsev.spb.ru

CONSYS RUSSIA  
Promyshlennaya St. 42  
**RU-198099 St Petersburg**  
Phone: +7 812 / 325 36 53  
Fax: +7 812 / 147 20 55  
e mail: consys@consys.spb.ru

Electrotechnical RUSSIA  
Systems Siberia  
Partizanskaya St. 27, Office 306  
**RU-121355 Moscow**  
Phone: +7 095 / 416-4321  
Fax: +7 095 / 416-4321  
e mail: info@eltechsystems.ru

Electrotechnical RUSSIA  
Systems Siberia  
Shetinkina St. 33, Office 116  
**RU-630088 Novosibirsk**  
Phone: +7 3832 / 22-03-05  
Fax: +7 3832 / 22-03-05  
e mail: info@eltechsystems.ru

Elektrostyle RUSSIA  
ul. Garschina 11  
**RU-140070 Moscow**  
Phone: +7 095 / 514 9316  
Fax: +7 095 / 514 9317  
e mail: info@estl.ru

Elektrostyle RUSSIA  
Krasnij Prospekt 220-1  
Office No. 312  
**RU-630049 Novosibirsk**  
Phone: +7 3832 / 10 66 18  
Fax: +7 3832 / 10 66 26  
e mail: info@estl.ru

ICOS RUSSIA  
Industrial Computer Systems Zao  
Ryazanskij Prospekt 8a, Office 100  
**RU-109428 Moscow**  
Phone: +7 095 / 232 - 0207  
Fax: +7 095 / 232 - 0327  
e mail: mail@icos.ru

NPP Uralelektra RUSSIA  
ul. Sverdlova 11a  
**RU-620027 Ekaterinburg**  
Phone: +7 34 32 / 53 27 45  
Fax: +7 34 32 / 53 27 45  
e mail: elektra@etel.ru

SSMP Rosgidromontazh Ltd. RUSSIA  
23, Lesoparkovaya Str.  
**RU-344041 Rostov On Don**  
Phone: +7 8632 / 36 00 22  
Fax: +7 8632 / 36 00 26  
e mail: —

STC Drive Technique RUSSIA  
ul. Bajkalskaja 239, Office 2 - 23  
**RU-664075 Irkutsk**  
Phone: +7 3952 / 24 38 16  
Fax: +7 3952 / 23 02 98  
e mail: privod@irk.ru

STC Drive Technique RUSSIA  
Poslannikov Per. 9, str.1  
**RU-107005 Moscow**  
Phone: +7 095 / 790-72-10  
Fax: +7 095 / 790-72-12  
e mail: info@privod.ru

**AFRICAN REPRESENTATIVE**

CBI Ltd. SOUTH AFRICA  
Private Bag 2016  
**ZA-1600 Isando**  
Phone: +27 (0) 11 / 928 2000  
Fax: +27 (0) 11 / 392 2354  
e mail: cbi@cbi.co.za