

Basic Model QCPU(Q Mode)

mitsubishi

User's Manual

(Function Explanation,
Program Fundamentals)

The graphic features the text "Q series series" in a large, stylized, 3D font. The first "Q" is significantly larger and more prominent than the rest of the text. The words "series" and "series" are stacked to the right of the large "Q". The text is rendered in a light gray color with a subtle gradient and a soft shadow, giving it a floating appearance. It is positioned over a background consisting of two overlapping rectangular areas: a solid light gray rectangle on the left and a white rectangle with a fine, repeating geometric pattern on the right.

Mitsubishi Programmable
Logic Controller

MELSEC-Q

The following table indicates differences between the Basic model QCPU.

Item		Q00JCPU	Q00CPU	Q01CPU
CPU module		CPU module, Power supply module, Main base unit (5 slots) Integrated type	Stand-alone CPU module	
Main base unit		Unnecessary	Necessary (Q33B, Q35B, Q38B, Q312B)	
Extension base unit		Connectable (Q52B, Q55B, Q63B, Q65B, Q68B, Q612B)		
Number of extension stages		Up to 2 stages	Up to 4 stages	
Number of input/output modules to be installed		16 modules	24 modules	
Power supply module				
Main base unit		Unnecessary	Necessary (Q61P-A1, Q61P-A2, Q62P, Q63P)	
Extension base unit	Q52B, Q55B	Unnecessary		
	Q63B, Q65B, Q68B, Q612B	Necessary (Q61P-A1, Q61P-A2, Q62P, Q63P)		
Extension cable		QC05B, QC06B, QC12B, QC30B, QC50B, QC100B		
Memory card interface		No		
External interface	RS-232	Yes (transmission rate: 9.6kbps, 19.2kbps, 38.4kbps, 57.6kbps, 115.2 kbps)		
	USB	No		
Processing speed (Sequence instruction)	LD X0	0.20μs	0.16μs	0.10μs
	MOV D0 D1	0.70μs	0.56μs	0.35μs
Program capacity*		8k steps (32 kbyte)	8k steps (32 kbyte)	14k steps (56 kbyte)
Memory capacity	Program memory	58 kbyte	94 kbyte	
	Standard RAM	—	64 kbyte	
	Standard ROM	58 kbyte	94 kbyte	
Device memory capacity		The number of device points can be changed within the range of 16.4 kbyte		
Number of input/output devices points (Remote I/O is contained.)		2048 points		
Number of input/output points		256 points	1024 points	
File register		No	Yes (32k points fixed)	
Serial communication function		No	Yes (using the RS-232 interface of the CPU module)	

*: 1 step of the program capacity is 4 Bytes.

1.1 Features

(1) Many controllable input/output points

As the number of input/output points accessible to the input/output modules loaded on the base units, 256 points (X/Y0 to FF) are supported by the Q00JCPU and 1024 points (X/Y0 to 3FF) by the Q00CPU/Q01CPU.

Up to 2048 points (X/Y0 to 7FF) are supported as the number of input/output device points usable for refreshing the remote input/output of CC-Link and the link inputs and outputs (LX, LY) of MELSECNET/H.

(2) Lineup according to program capacity

The optimum CPU module for the program capacity to be used can be selected.

Q00JCPU, Q00CPU : 8k steps

Q01CPU : 14k steps

(3) Fast processing

The LD instruction processing speeds are the following values.

Q00JCPU : 0.20 μ s

Q00CPU : 0.16 μ s

Q01CPU : 0.10 μ s

In addition, the high-speed system bus of the MELSEC-Q series base unit speeds up access to an intelligent function module and the link refresh of a network.

MELSECNET/H link refresh processing : 2.2ms/2k words *1

*1 This speed only applies when the SB/SW is not used with the Q01CPU and the MELSECNET/H network module is used as the main base unit.

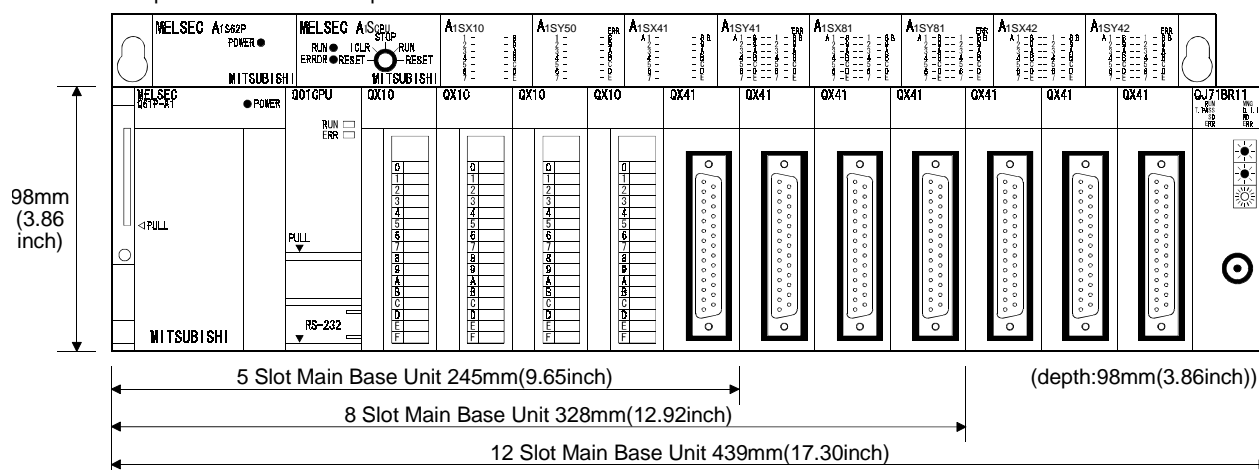
(4) Increase in debugging efficiency through high-speed communication with GX Developer

The RS-232 interface of the Basic model QCPU enables program write/read or monitor at a maximum of 115.2kbps.

(5) Saved space by a reduction in size

The installation area of the Basic model QCPU is about 60% of that of the AnS series.

Comparison of installation space



(6) Connection of up to four/two extension base units

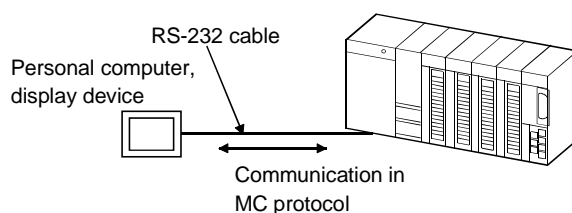
- (a) The Q00JCPU can connect up to two extension base units (three base units including the main) and accepts up to 16 modules.
- (b) The Q00/Q01CPU can connect up to four extension base units (five base units including the main) and accepts up to 24 modules.
- (c) The overall distance of the extension cables is up to 13.2m to ensure high degree of extension base unit arrangement.

POINT

When bus-connecting the GOT, the number of extension base units connected decreases by one since the GOT uses one stage of the above base units.

(7) Serial communication function for communication with personal computer or display device

With the RS-232 interface of the Q00CPU or Q01CPU connected with a personal computer, display device or the like, the MELSEC communication protocol (hereafter referred to as the MC protocol) can be used to make communication.



The serial communication function only allows communication in the MC protocol (QnA-compatible 3C frame (format 4), QnA-compatible 4C frame (format 4, 5)).

The serial communication function does not allow communication in the nonprocedure protocol or bidirectional protocol.

Refer to the following manual for the MC protocol.

- Q Corresponding MELSEC Communication Protocol Reference Manual

(8) Built-in standard ROM

The flash ROM for storing parameters and sequential program is installed as a standard feature for easier protection of important program.

(9) Easy operation of CC-Link system

The I/O signals for up to 32 remote I/O stations can be controlled without parameters when one master module of the CC-Link system is used.

The remote I/O stations can be controlled in a similar manner to controlling the input/output modules installed on the base unit.

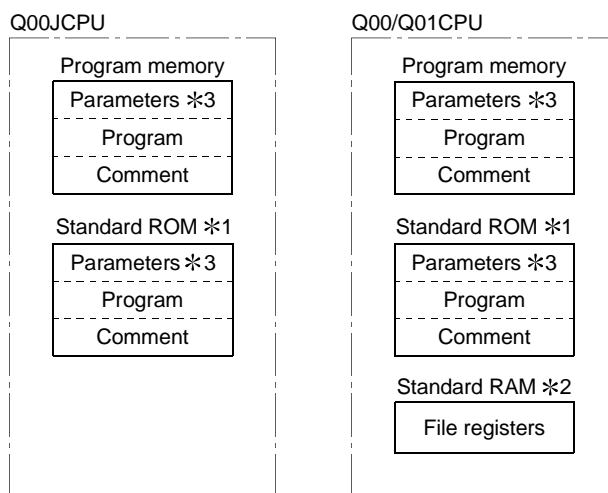
(10) Blocking an invalid access using the file password

Program can be prevented from being altered through invalid access by presetting the access level (reading prohibited, writing prohibited) in the file password.

1.2 Program Storage and Calculation

(1) Program storage

Program created at GX Developer can be stored in Basic model QCPU's program memory or standard ROM.

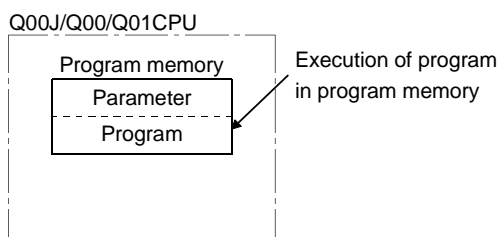


*1: The standard ROM is used when parameters, program and comment are written to ROM.

*2: The standard RAM is used for file registers.

*3: Including the intelligent parameters of the intelligent function module set on GX Configurator.

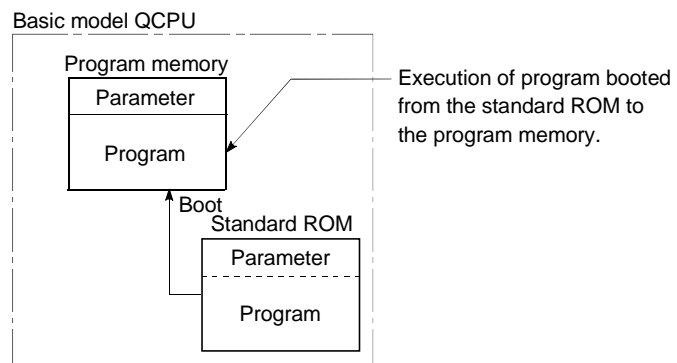
(2) The Basic model QCPU processes program which are stored in the program memory.



(3) Boot operation of program

The program stored on the standard ROM is booted (read) to the program memory of the Basic model QCPU and executed.

Booting a program from the standard ROM to the program memory requires boot file setting in the PLC parameter.

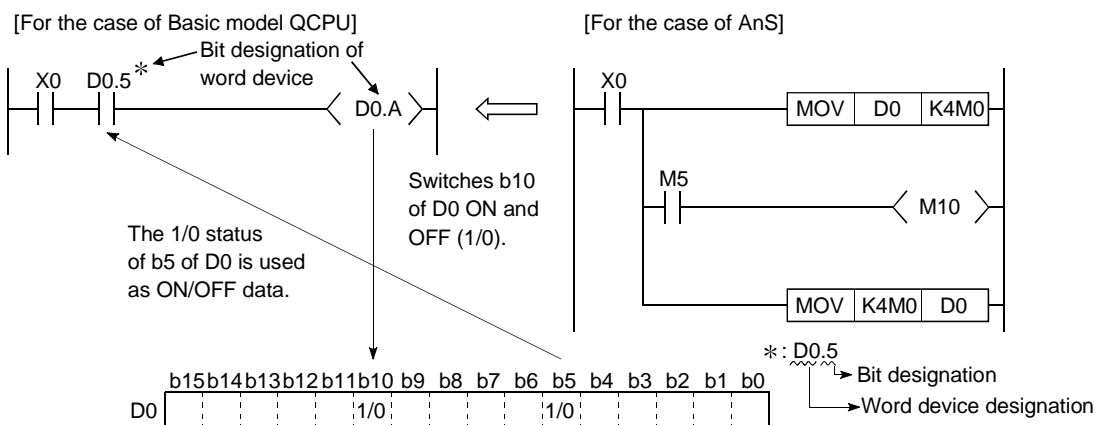


1.3 Convenient Programming Devices and Instructions

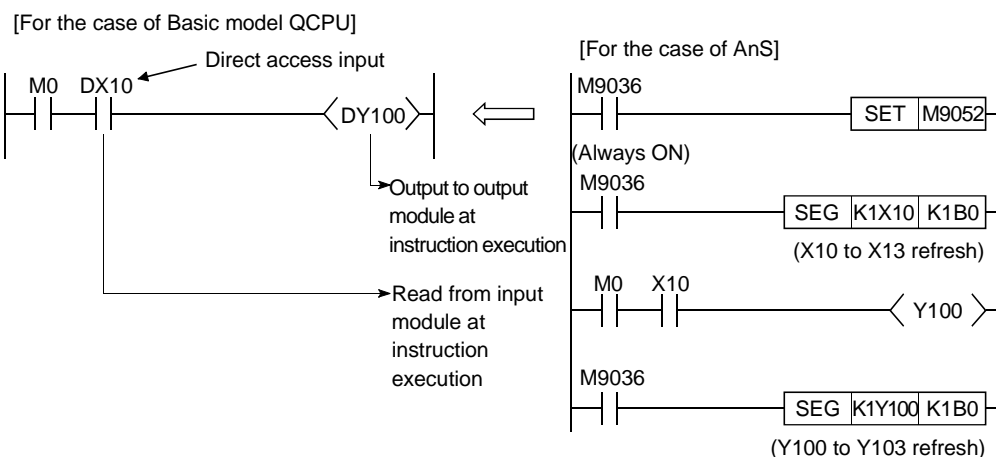
The Q00J/Q00/A01CPU features devices and instructions which facilitate program creation. A few of these are described below.

(1) Flexible device designation

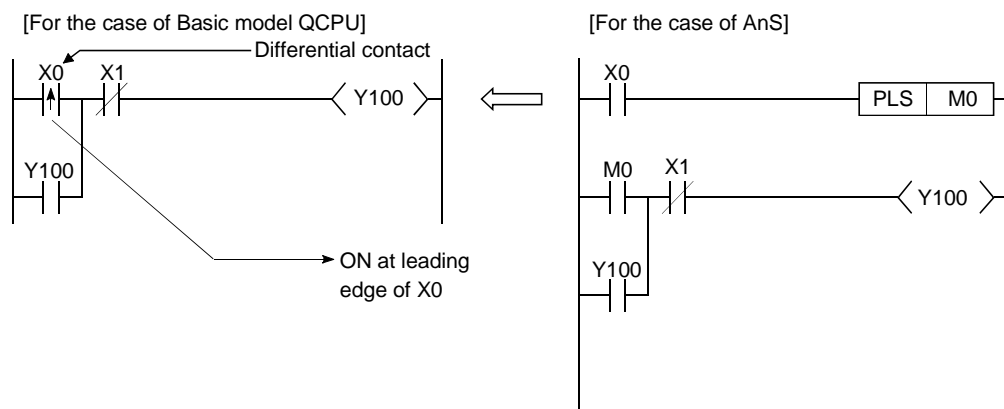
(a) Word device bits can be designated to serve as contacts or coils.



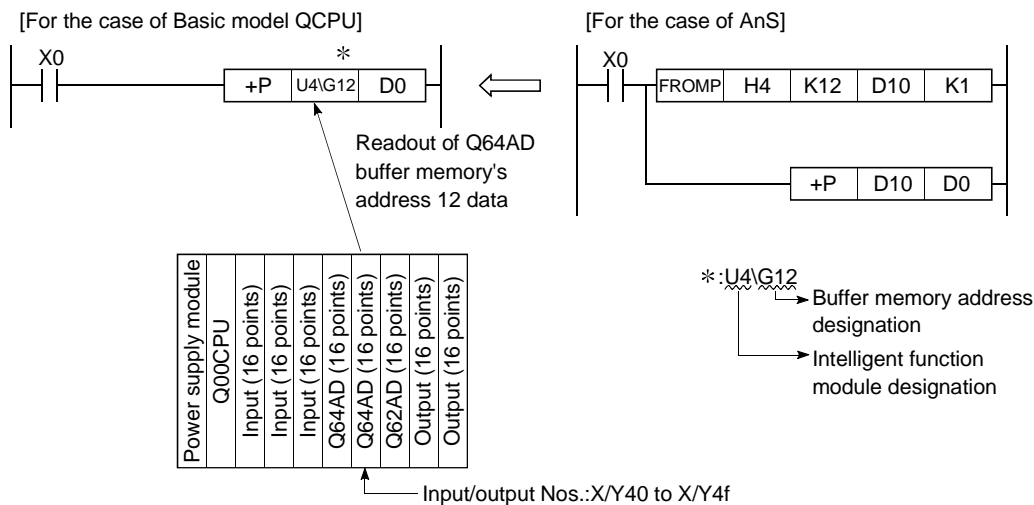
(b) Direct processing in 1-point units is possible within a program simply by using direct access inputs (DX[]) and direct access outputs (DY[]).



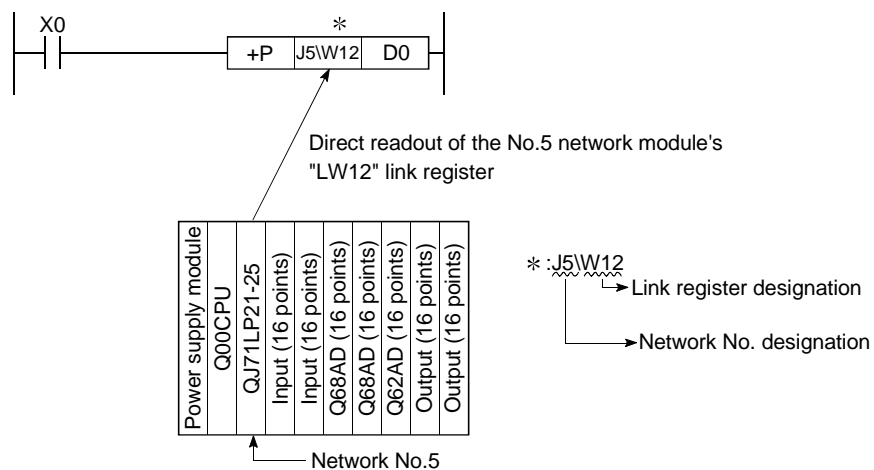
(c) Differential contacts (—|/|—) eliminate the need for converting inputs to pulses.



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



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

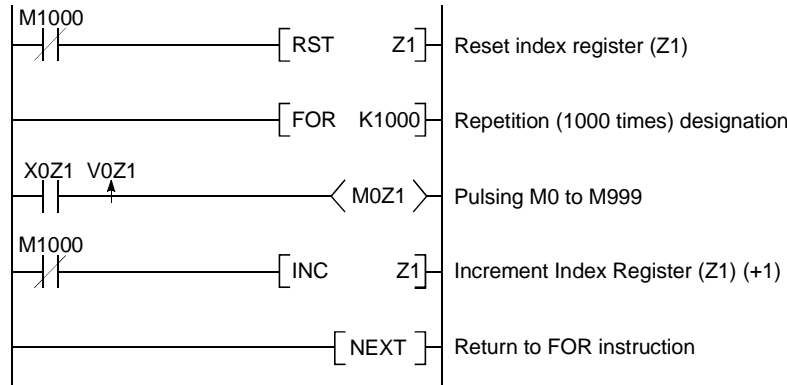


- (f) If index-qualified, each instruction of the Basic model QCPU does not increase in processing time, facilitating writing of a structured program.

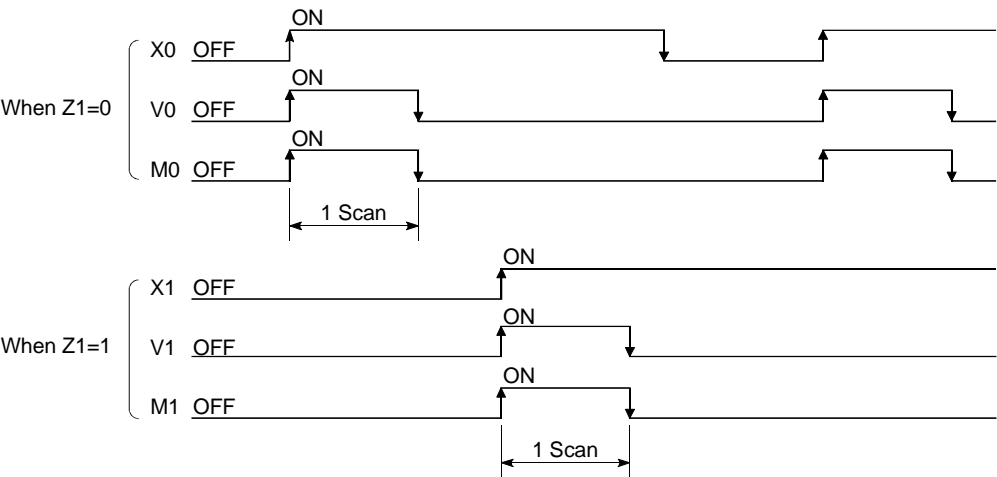
(2) Edge relays simplify pulse conversion processing

- (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]



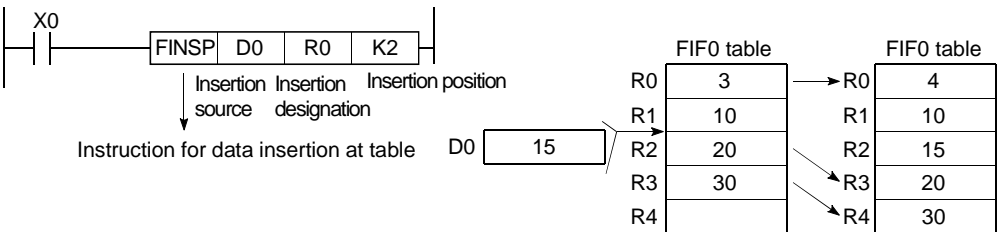
[Timing chart]



REMARK

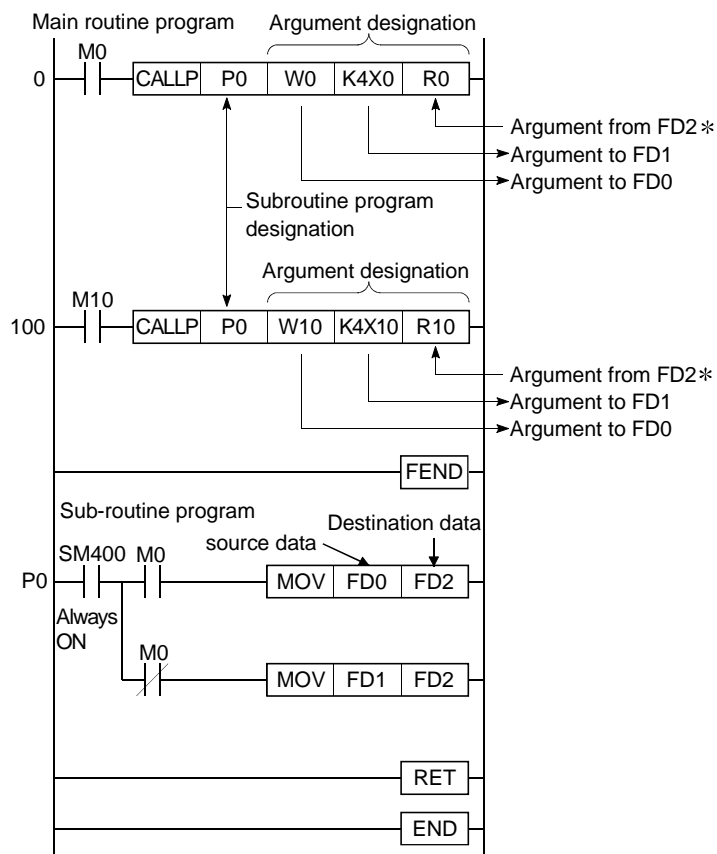
*: NUL indicates "00H (character string END)".

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

Subroutine call instructions with arguments will make it easier to create a subroutine programs that makes several calls.

**REMARK**

* For details regarding the argument input/output condition, refer to Section 10.3.1.

2. SYSTEM CONFIGURATION

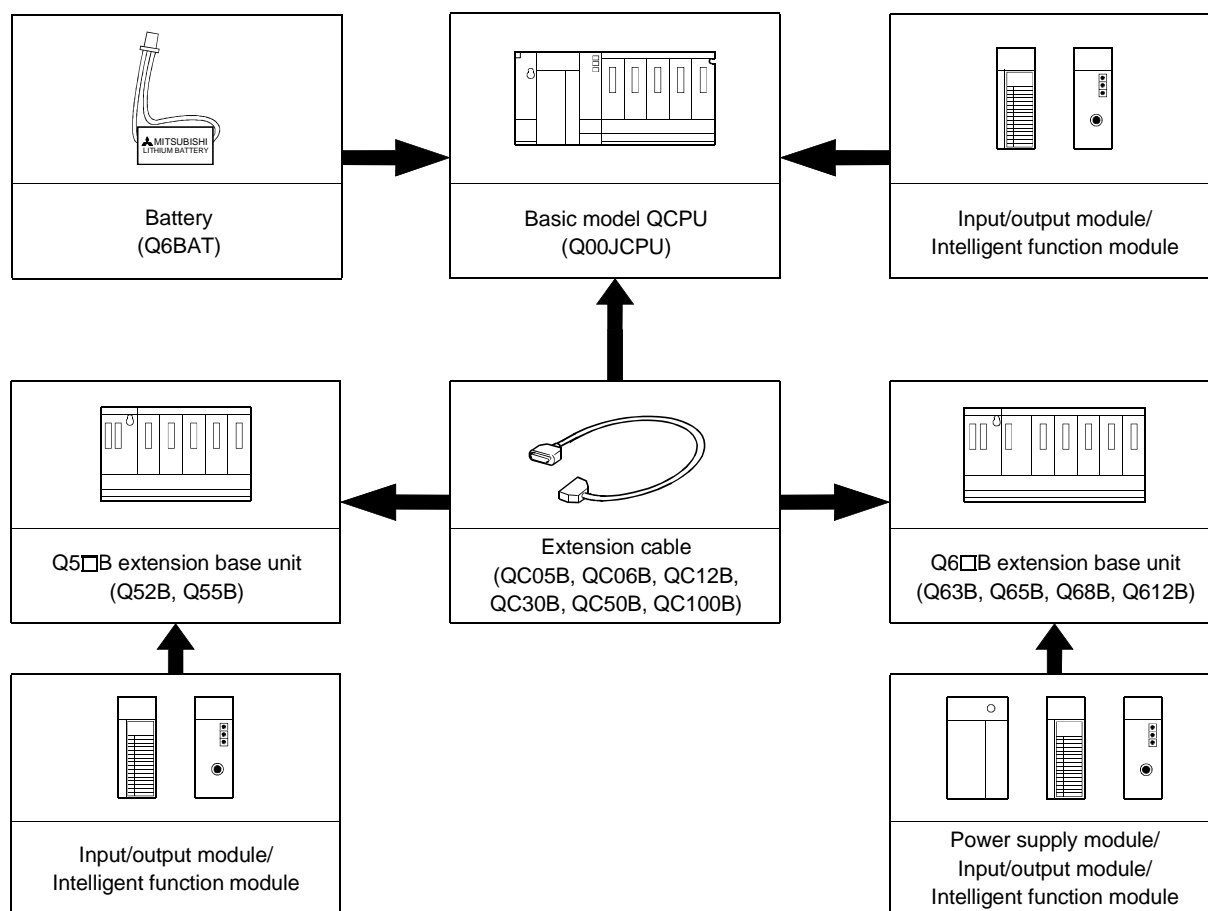
This section describes the system configuration of the Basic model QCPU, cautions on use of the system, and configured equipment.

2.1 System Configuration

2.1.1 Q00JCPU

This section explains the equipment configuration of a Q00JCPU system and the outline of the system configuration.

(1) Equipment configuration



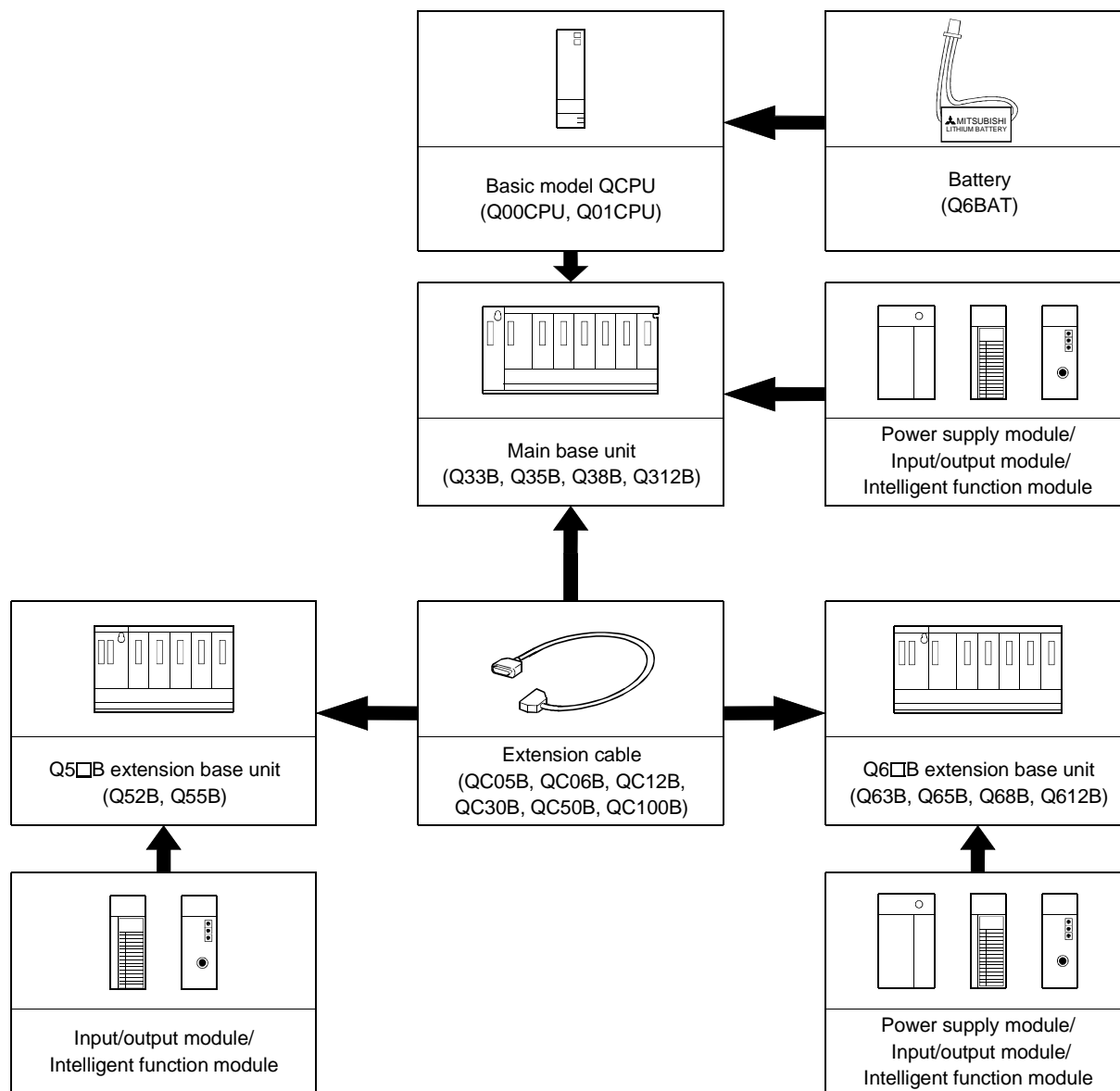
(2) Outline of system configuration

System configuration	<div style="display: flex; justify-content: space-around;"> <div style="width: 45%;"> <p>(a) System including extension base units</p> </div> <div style="width: 45%;"> <p>(b) System including extension base unit and GOT</p> </div> </div> <p style="text-align: center;">Loading will cause error</p> <p style="text-align: center;">Extension cable connector</p> <p style="text-align: center;">*Both of the above systems assume that each slot of the main and extension base units is loaded with a 16-point module.</p>
Maximum number of Extension Stages	Two Extension Stages
Maximum number of input/output modules to be installed	16 modules
Maximum number of input/output points	256
Main base unit	Unnecessary
Extension base unit	Q52B, Q55B, Q63B, Q65B, Q68B, Q612B
Extension cable	QC05B, QC06B, QC12B, QC30B, QC50B, QC100B
Notes	<p>(1) Do not use an extension cable longer than an overall extension length of 13.2m(43.31ft.).</p> <p>(2) 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.</p> <p>(3) 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.</p> <p>(4) The QA1S6□B/QA65B cannot be connected as an extension base unit.</p> <p>(5) 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.</p> <p>(6) If 17 or more modules are installed, an error will occur.</p> <p>(7) When bus-connected, the GOT occupies one extension stage and one slot.</p> <p>(8) The Q00JCPU processes the GOT as a 16-point intelligent function module. Hence, connection of one GOT decreases the number of controllable points on base units by 16 points.</p> <p>(9) The bus extension connector box (A9GT-QCNB) cannot be connected to the Q00JCPU. It should be connected to the extension base unit.</p>

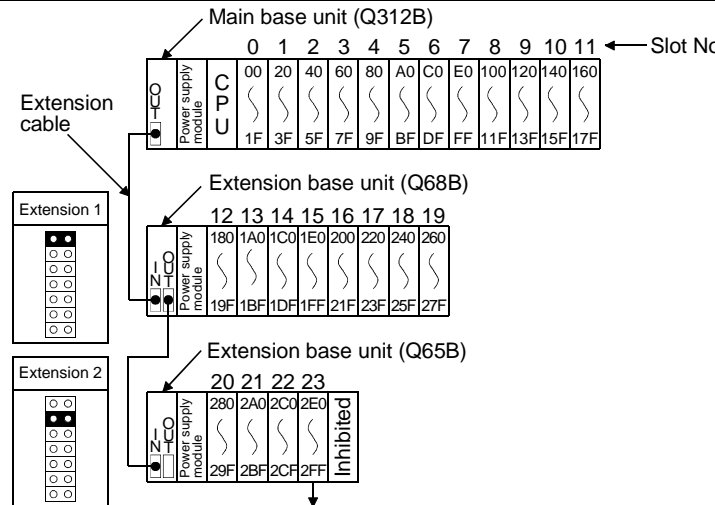
2.1.2 Q00/Q01CPU

This section explains the equipment configuration of a Q00/Q01CPU system and the outline of the system configuration.

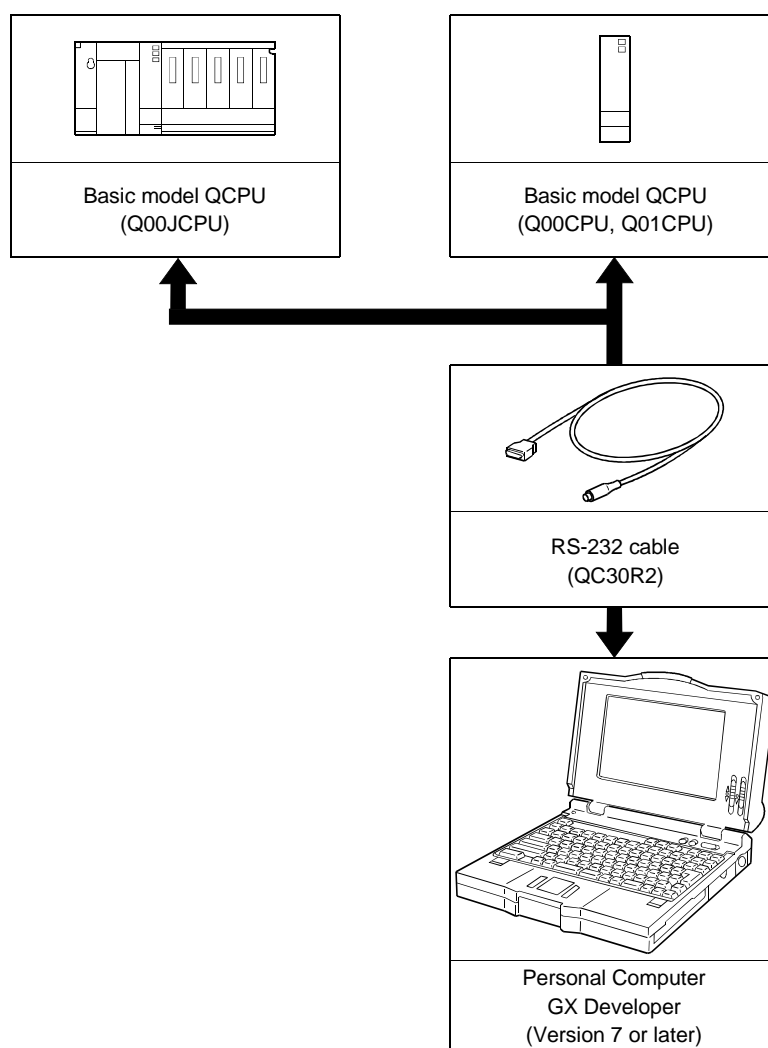
(1) Equipment configuration



(2) Outline of system configuration

System configuration	 <p>Main base unit (Q312B)</p> <p>Slot No.</p> <p>Extension cable</p> <p>Extension 1</p> <p>Extension 2</p> <p>Extension base unit (Q68B)</p> <p>Extension base unit (Q65B)</p> <p>Loading will cause error</p> <p>*The above system assumes that each slot is loading with a 32-point module.</p>
Maximum number of Extension Stages	Four Extension Stages
Maximum number of input/output modules to be installed	24 modules
Maximum number of input/output points	1024
Main base unit	Q33B, Q35B, Q38B, Q312B
Extension base unit	Q52B, Q55B, Q63B, Q65B, Q68B, Q612B
Extension cable	QC05B, QC06B, QC12B, QC30B, QC50B, QC100B
Notes	<p>(1) Do not use an extension cable longer than an overall extension length of 13.2m(43.31ft.).</p> <p>(2) 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.</p> <p>(3) 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.</p> <p>(4) The QA1S6□B/QA65B cannot be connected as an extension base unit.</p> <p>(5) 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.</p> <p>(6) If 25 or more modules are installed, an error will occur.</p> <p>(7) When bus-connected, the GOT occupies one extension stage and one slot.</p> <p>(8) The Q00/Q01CPU processes the GOT as a 16-point intelligent function module. Hence, connection of one GOT decreases the number of controllable points on base units by 16 points.</p>

2.1.3 Configuration of GX Developer



2.2 Precaution on System Configuration

This section describes hardware and software packages compatible with Basic model 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/H network module	QJ71LP21, QJ71BR11, QJ71LP21-25, QJ71LP21G, QJ71LP21GE	One module only
Q series Ethernet interface module	QJ71E71, QJ71E71-B2, QJ71E71-100	One module only
Q series CC-Link system master local module	QJ61BT11	Up to 2 modules function version B or later
Interrupt module	QI60	One module only

- (b) A graphic operation terminal can be used only for the GOT900 series and F900 series (Basic OS matching Q-mode and communication driver must be installed).

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

- (c) A DeviceNet Master-Slave module (QJ71DN91) whose function version is B or later can be used.

(2) Software package

GX Developer and GX Configurator of the versions or later in the following table are usable with the Basic model QCPU.

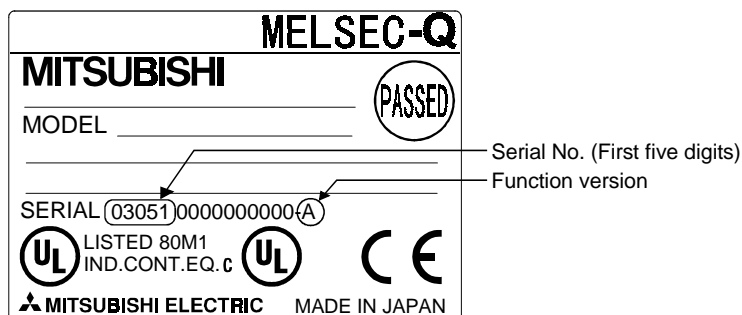
Product Name	Type	Version
GX Developer	SW7D5C-GPPW-E	Ver. 7
GX Simulator	SW6D5C-LLT-E	Ver. 6
GX Configurator-AD	SW0D5C-QADU-E	Ver. 1.10L
GX Configurator-DA	SW0D5C-QDAU-E	Ver. 1.10L
GX Configurator-SC	SW0D5C-QSCU-E	Ver. 1.10L
GX Configurator-CT	SW0D5C-QCTU-E	Ver. 1.10L
GX Configurator-TC	SW0D5C-QTCU-E	Ver. 1.10L
GX Configurator-FL	SW0D5C-QFLU-E	Ver. 1.10L
GX Configurator-DN	SW0D5C-QDNU-E	Ver. 1.10L
GX Configurator-TI	SW0D5C-QTIU-E	Ver. 1.10L
GX Configurator-PT	SW1D5C-QPTU-E	Ver. 1.10L
GX Configurator-QP	SW2D5C-QD75P-E	Ver. 2.10L

2.3 Confirming the function version

The Basic model QCPU function version can be confirmed on the rating nameplate and GX Developer's system monitor.

(1) Confirming the function version on the rating nameplate

The function version is indicated on the rating nameplate.



(2) Confirming the function version on the system monitor (product information List)

The product information list in the system monitor of GX Developer allows you to confirm the function version of the Basic model QCPU.

The product information list of the system monitor also allows you to confirm the function versions of the intelligent function modules.

Serial No. Function version

↓ ↓

Slot	Type	Series	Model name	Points	I/O No.	Master PLC	Serial No	Ver
PLC	PLC	Q	Q01CPU	-	-	-	0305100000000000	A
0-0	Intelli.	Q	QJ71E71	32pt	0000	-	0208100000000000	B
0-1	-	-	None	-	-	-	-	-
0-2	-	-	None	-	-	-	-	-
0-3	-	-	None	-	-	-	-	-
0-4	-	-	None	-	-	-	-	-
0-5	-	-	None	-	-	-	-	-
0-6	-	-	None	-	-	-	-	-
0-7	-	-	None	-	-	-	-	-

CSV file creating Close

3. PERFORMANCE SPECIFICATION

The table below shows the performance specifications of the Basic model QCPU.

Performance Specifications

Item		Model			Remark
		Q00JCPU	Q00CPU	Q01CPU	
Control method		Repetitive operation of stored program			
I/O control method		Refresh mode			Direct input/output is possible by direct input/output specification (DX□, DY□)
Programming language (Sequence control dedicated language)		Relay symbol language, logic symbolic language			The SFC function is not applicable.
Processing speed (Sequence instruction)	LD X0	0.20μs	0.16μs	0.10μs	
	MOV D0 D1	0.70μs	0.56μs	0.35μs	
Total number of instructions		249 (excluding intelligent function module dedicated instructions)			
Constant scan (Function to make the scan time constant)		2 to 2000 ms (configurable in increments of 1 ms)			Set parameter values to specify
Program *1 *2 capacity		8k steps (32 kbyte)	8k steps (32 kbyte)	14k steps (56 kbyte)	
Memory capacity	Program memory (Drive 0)	58 kbyte	94 kbyte	94 kbyte	
	Standard RAM (Drive 3)	0	64 kbyte	64 kbyte	
	Standard ROM (Drive 4)	58 kbyte	94 kbyte	94 kbyte	
Number of stored programs	Program memory	1	1	1	
	Standard ROM	1	1	1	
Number of stored file registers	Standard RAM	—	1	1	
Number of I/O devices points		2048 points (X/Y0 to 7FF)			Number of devices usable on program
Number of I/O points		256 points (X/Y0 to FF)	1024 points (X/Y0 to 3FF)		Number of points accesible to input/output modules

*1: "1 step" in program capacity equals 4 bytes.

*2: The maximum number of steps that can be executed can be obtained as follows:
(Program capacity) - (File header size (Default: 34 steps))

Performance Specifications (continued)

Item		Model		Remark		
		Q00JCPU	Q00CPU		Q01CPU	
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 2048 points (L0 to 2047)			
	Link relay [B]		Default 2048 points (B0 to 7FF)			
	Timer [T]		Default 512 points (T0 to 511) (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 speed retentive timer and 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]		• Normal counter default 512 points (C0 to 511) • Interrupt counter maximum 128 points (default 0 point, set with parameters)			
	Data register [D]		Default 11136 points (D0 to 11135)			
	Link register [W]		Default 2048 points (W0 to 7FF)			
	Annunciator [F]		Default 1024 points (F0 to 1023)			
	Edge relay [V]		Default 1024 points (V0 to 1023)			
	File Register	[R]	None	32768 points (R0 to 32767)		
		[ZR]	None	32768 points (ZR0 to 32767)		

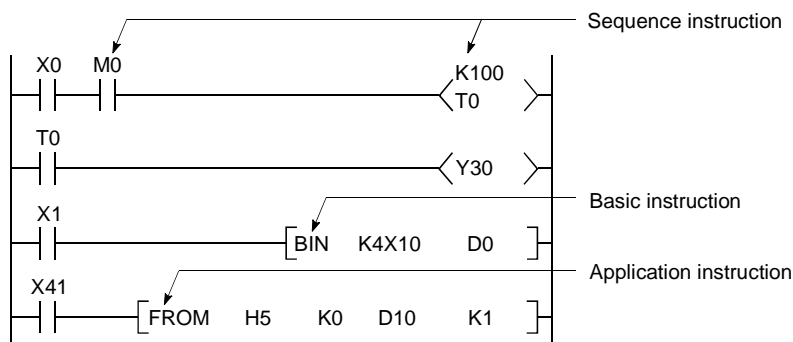
4 SEQUENCE PROGRAM CONFIGURATION & EXECUTION CONDITIONS

Programs that can be executed by the Basic model QCPU are sequence programs only.
This chapter describes the sequence program configuration and execution conditions.

4.1 Sequence Program

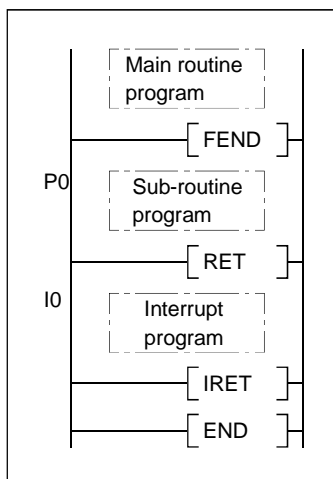
(1) Definition of sequence program

- (a) A sequence program is created using 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 regarding these programs, refer to 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

MAIN



REMARK

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

Performance Specifications (continued)

Item		Model			Remark
		Q00JCPU	Q00CPU	Q01CPU	
Number of device points	Special link relay [SB]	1024 points (SB0 to 3FF)			The number of device points is fixed.
	Special link register [SW]	1024 points (SW0 to 3FF)			
	Step relay [S] *3	2048 points (S0 to 2047)			
	Index register [Z]	10 points (Z0 to 9)			
	Pointer [P]	300 points (P0 to 299)			
	Interrupt pointer [I]	128 points (I0 to 127) The specified intervals of the system interrupt pointers I28 to I31 can be set with parameters.(0.5 to 1000ms, Cunit in 0.5 ms) Default I28 : 100ms I29 : 40ms I30 : 20ms I31 : 10ms			
	Special relay [SM]	1024 points (SM0 to 1023)			
	Special register [SD]	1024 points (SD0 to 1023)			
	Function input [FX]	16 points (FX0 to F)			
	Function output [FY]	16 points (FY0 to F)			
	Function register[FD]	5 points (FD0 to 4)			
Link direct device		Device for direct access to link device. MELSECNET/H use only. Specified form at : J□□\X□□, J□□\Y□□, J□□\W□□, J□□\B□□, J□□\SW□□, J□□\SB□□			
Intelligent function module device		Device for direct access to the buffer memory of the intelligent function module. Specified form at : U□□\G□□			
Latch (power failure compensation) range		L0 to 2047 (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 7FF, respectively.			
Clock function		Year, month, day, hour, minute, second, day of the week (leap year automatic distinction) Accuracy -3.2 to +5.27s (TYP. +1.98s) /d at 0°C Accuracy -2.57 to +5.27s(TYP. +2.22s)/d at 25°C Accuracy -11.68 to +3.65s(TYP. -2.64s)/d at 55°C			
Allowable momentary stop time		Max. 20ms (Min. 100VAC)	Varies according to the type of power supply module.		
5VDC internal current consumption		0.22A*4	0.25A	0.27A	
Weight		0.66kg*5	0.13kg	0.13kg	
External dimensions		H	98mm (3.86in.)		
		W	245mm (9.65in.)		
		D	97mm (3.82in.)		

*3: The "step relay" is a device for the SFC function.

This cannot be used as the SFC function is not applicable to the Basic model QCPU.

*4: This value includes the CPU module and base unit.

*5: This value includes the CPU module, base unit, and power supply module.

(2) Sequence program writing format

Programming for sequence programs is possible 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 processing, with the ladder beginning from the left bus and ending at the right bus.

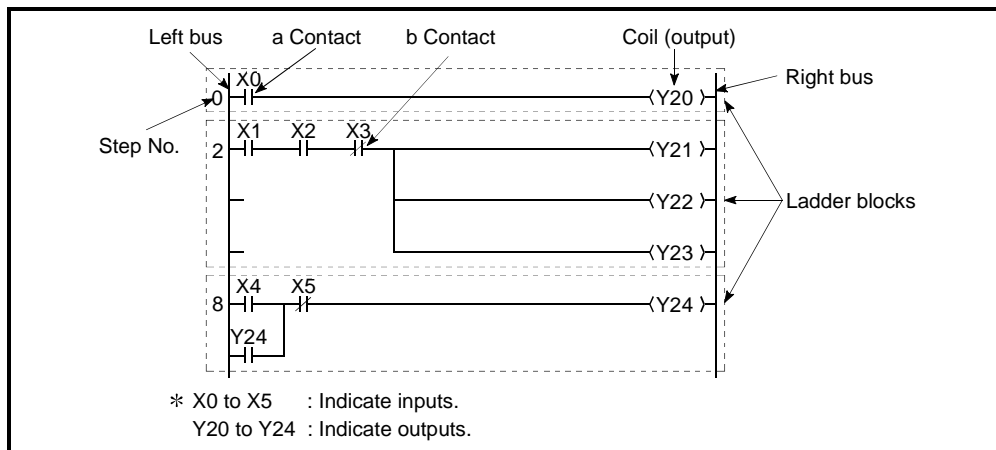


Fig.4.1 Ladder Block

(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 contactLD, AND, OR
- b contactLDI, ANI, ORI
- coil.....OUT

(2) Program processing

Sequence programs are processed in order, beginning from step 0 and ending at the END/FEND instruction.

Processing of ladder mode ladder blocks begins from the left bus, and proceeds from left to right. When one ladder block is completed, processing proceeds downward to the next ladder block.

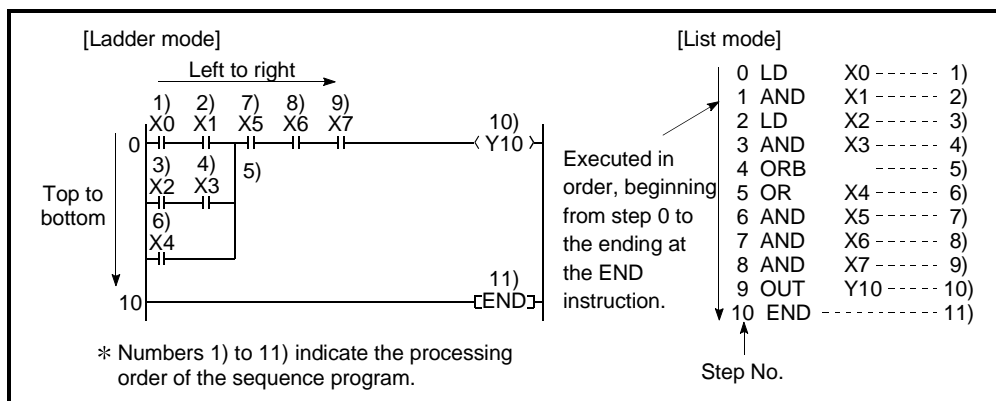


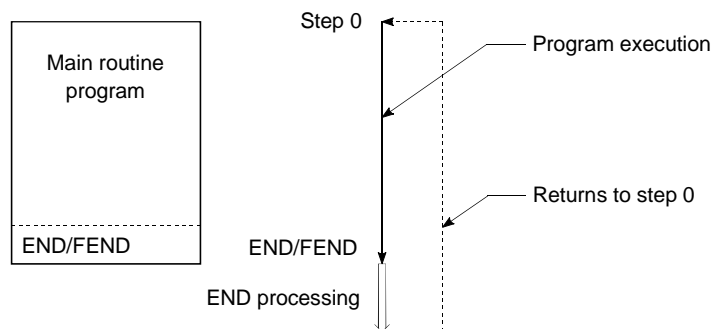
Fig.4.2 Sequence Program Processing

4.1.1 Main routine program

(1) Definition of main routine program

- (a) A main routine program is a program which begins from step 0 and ends at the END/FEND instruction. *1
- (b) The main routine program execution begins from step 0 and ends at the END/FEND instruction.

When the END/FEND instruction is executed in the main routine program, END processing is performed and operation is then restarted from step 0.



(2) Execution of main routine program

The main routine program is executed every scan.

REMARK

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

4.1.2 Sub-routine programs

(1) Definition of sub-routine program

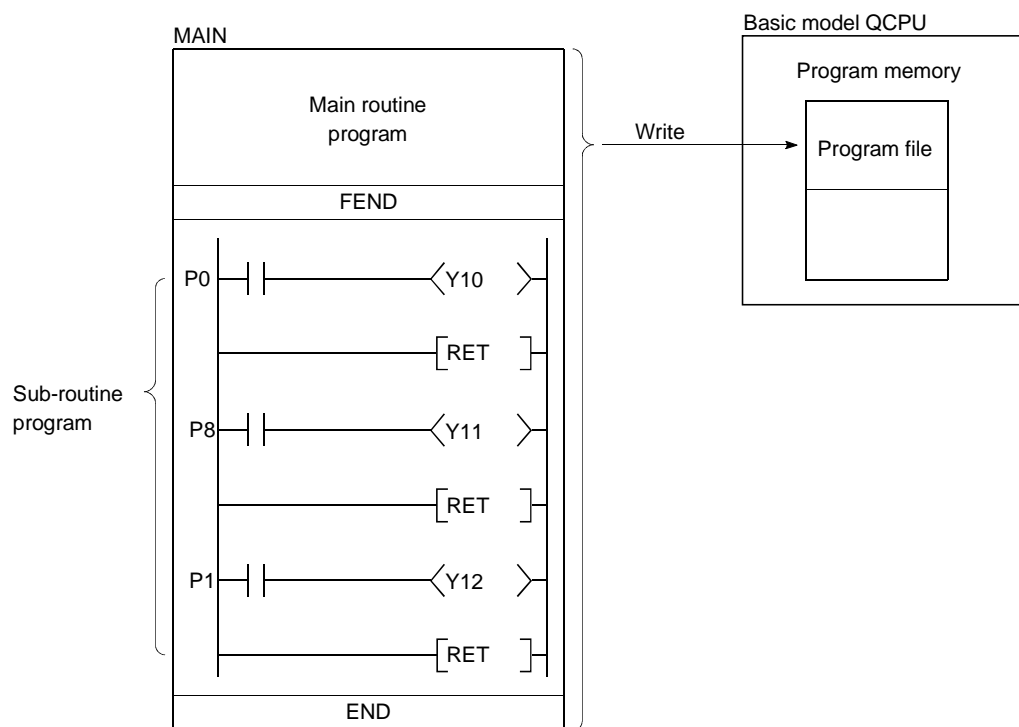
- (a) A sub-routine program is a program which begins from a pointer (P□) and ends at a RET instruction.
- (b) A sub-routine program is executed only when called by a CALL instruction (e.g. CALL(P), FCALL(P)) from the main routine program.
- (c) Sub-routine program application
 - 1) The overall step 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 is managed as one program.

Create a sub-routine program as described below.

- A sub-routine program is created between the main routine program's FEND and END instructions.
- Because there are no restrictions regarding the order in which sub-routine programs are created, there is no need to set the pointers in ascending order when creating multiple sub-routine programs.



4.1.3 Interrupt programs

(1) Definition of interrupt program

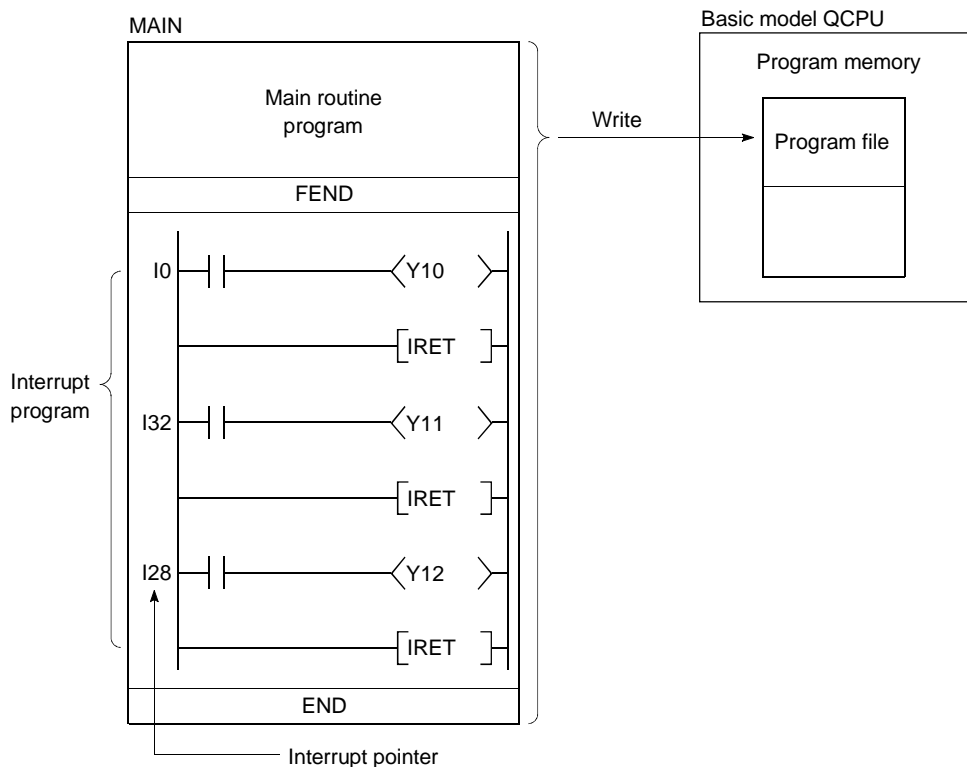
- (a) An interrupt program is a program which begins at the interrupt pointer (I□), and ends at the IRET instruction. *1
- (b) Interrupt programs are executed only when an interrupt factor occurs. *1

(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 is managed as one program.

Create an interrupt program as described below.

- An interrupt program is created between the main routine program's FEND and END instructions.
- Because there are no restrictions regarding the order in which interrupt programs are created, there is no need to set the interrupt pointers in ascending order when creating multiple interrupt programs.



REMARK

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

(3) Executing interrupt programs

- (a) To run an interrupt program, interrupts must have been enabled by the EI instruction. *1
 - 1) If interrupt factors occur before interrupts are enabled, the interrupt factors that occurred are stored, and the interrupt programs corresponding to the stored interrupt factors are executed as soon as interrupts are enabled.
 - 2) If the same interrupt factor occurs more than once, the interrupt factors that occurred are stored or discarded.

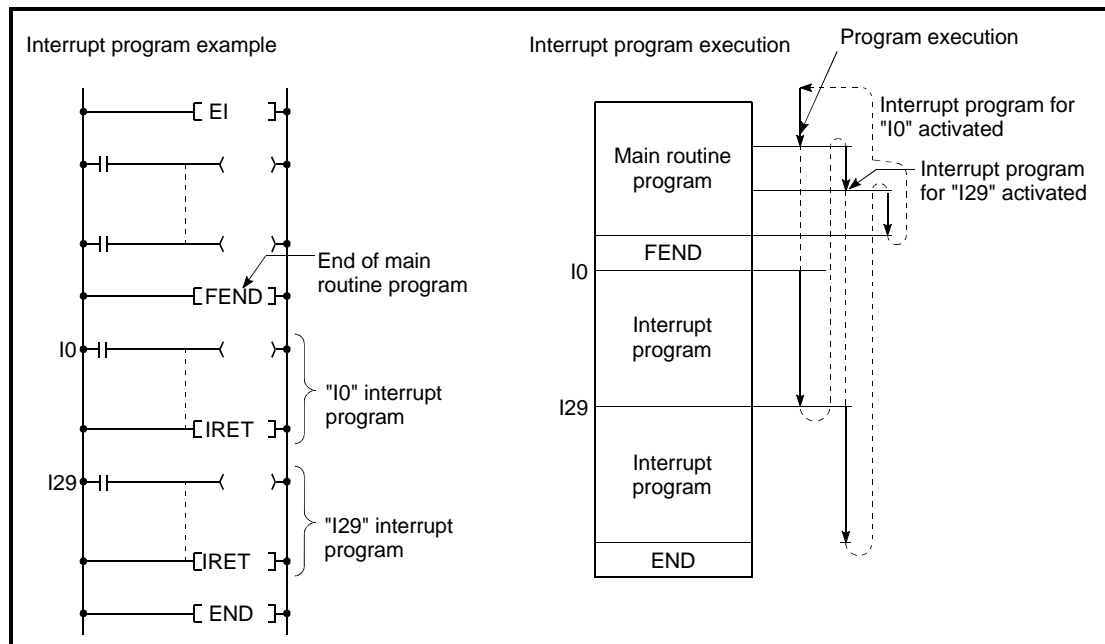


Fig.4.3 Interrupt Program Execution

- (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) : When multiple interruptions occur simultaneously
 When multiple interrupt programs are activated simultaneously, the programs will be executed in order, beginning from the interrupt program with the highest priority interrupt pointer number. *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:
 Interruptions are prohibited during execution of instructions.
 If an interrupt factor occurs during execution of an instruction, the interrupt program will be executed after processing of the instruction is completed.

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 MESSENET/H Network System. *3

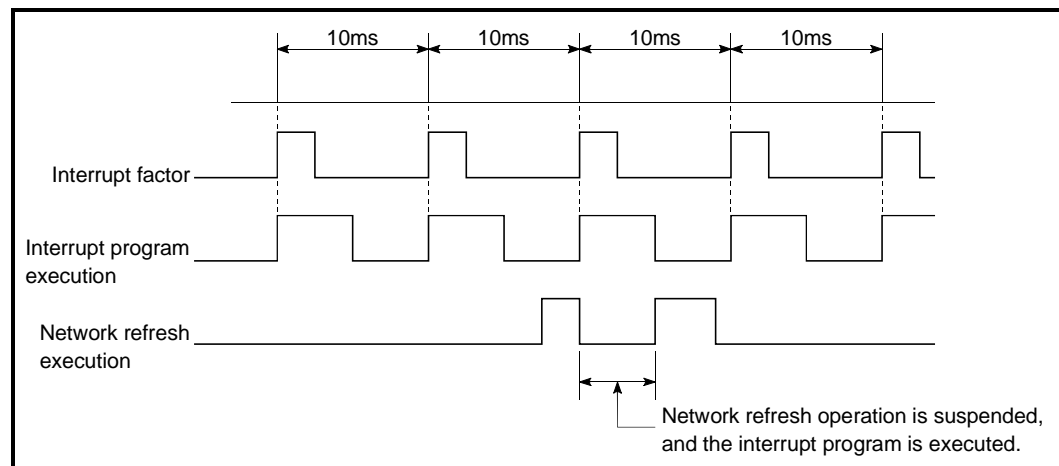


Fig.4.4 Interruption during Network Refresh Operation

4) Interruption during END processing:

If an interrupt factor occurs during an END processing waiting period during constant scanning, the interrupt program corresponding to that factor will be executed.

- (c) See Section 10.6.1 for details regarding index register processing when switching to an interrupt program from a main routine program or sub routine program.

(4) High-speed execution of an interrupt program and overhead time

By default, the Basic model QCPU "hides and restores an index register" when executing an interrupt program.

The above-listed processes are not performed if an option to "Execute at a High Speed" is selected in the PLC System Setting sheet of the PLC Parameter dialog box. This will make it possible to shorten the duration of overhead time required for execution of an interrupt program.

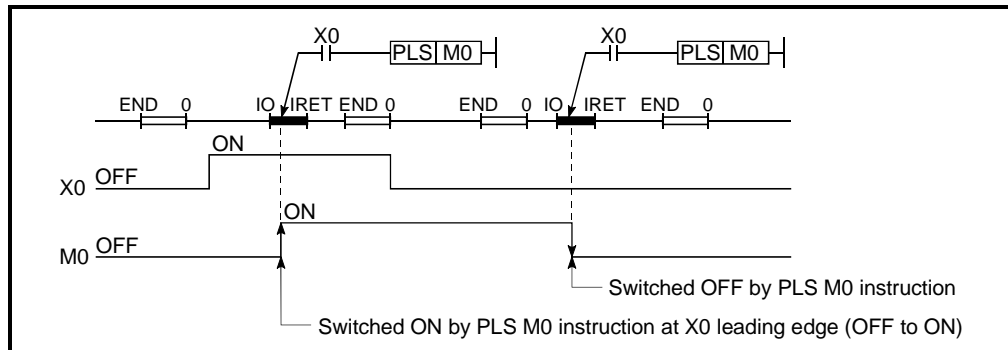
Refer to Section 11.2 for the overhead time of an interrupt program.

REMARK

- *1: For details regarding the IMASK and EI instructions, refer to the "QCPU (Q mode)/ QnACPU Programming Manual (Common Instructions).
- *2: See Section 10.10 for details regarding the priority ranking of interrupt programs.
- *3: For assurance of station unit blocks in cyclic data, see the "MELENET/H Network System Reference Manual."

(5) Program creation restrictions

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



- (b) During execution of the interrupt program, interrupts are disabled (DI) so that other interrupt processing is not performed.
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 instructions cannot be used in interrupt programs.
- COM
 - ZCOM
- (e) When the interrupt program is executed when measuring time such as the scan time or execution time, the measured time will become the value obtained by adding the interrupt program/constant cycle execution type program.
Thus, if the interrupt program is executed, the values stored in the following special registers and GX Developer monitor values will become longer than when the interrupt program is not executed.
- 1) Special registers
 - SD520, SD521: Current scan time
 - SD524, SD525: Minimum scan time
 - SD526, SD527: Maximum scan time
 - SD540, SD541: END processing time
 - SD542, SD543: Constant scan wait time
 - 2) GX Developer monitor values
 - Execution time measurement
 - Scan time measurement
 - Constant scan

4.2 Concept of Scan Time

(1) Scan time

- (a) The "scan time" is a total of following the execution time of program and END processing.
When an interrupt program is executed, the value including the execution time of the interrupt program will be the scan time.
- (b) The scan time present value, minimum value, and maximum value are measured at the Basic model QCPU, and the results are stored in special registers (SD520, SD521, and SD524 to SD527). *1
The scan time can therefore 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 ± 0.1 ms.
The scan time count will continue even if a watchdog time reset instruction (WDT) is executed at the sequence program.

(2) Constant scan setting: *2

When constant scanning is designated, the main routine program is executed at each designated constant scan period.

(3) WDT (Watchdog timer)

This is the timer which monitors the scan time, and its default setting is 200 ms.
This WDT setting can be designated in a 10 ms to 2000 ms range in the PLC RAS settings of the PLC parameter.
(Setting units: 10 ms)

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

*1: The "constant scan" function executes the scan type program repeatedly at regular intervals.
For details regarding of the constant scan, refer to Section 7.2.

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 Basic model QCPU goes in the RUN/STOP/RESET switch setting status. (See Section 4.4.)

Initial processing item	Basic model QCPU status		
	When the power is turned on.	When reset is executed.	When STOP to RUN *1
The Input/Output module initialization	○	○	×
Boot from the standard ROM	○	○	×
Device initialization of the range not latched (bit device: OFF, word device: 0)	○	○	×
Execution of self-diagnosis in the QCPU CPU modules	○	○	×
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	○	○	×
Setting of CC-Link information	○	○	×
Setting of Ethernet information	○	○	×
Setting of serial communication function	○	○	×

○ : executed, × : not executed

REMARK

*1: When parameters or programs are changed in the STOP status, reset by the RUN/STOP/RESET switch.

When the RUN/STOP/RESET switch is turned from STOP to RUN without the reset, RUN LED flashes.

When the RUN/STOP/RESET switch is turned from RUN to STOP to RUN again, the Basic model QCPU goes in the RUN status, and the "When STOP to RUN" status becomes effective.

However, fully note that the pulsing instruction (PLS, □P) may not operate properly since the previous information is not continued depending on program modifications.

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 Basic model QCPU is produced 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 for the intelligent function modules to use for details regarding of 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.

- MELSECNET/H or CC-Link refresh processing
- Automatic refresh of intelligent function module
- Self-diagnostics
- Communication with external device such as GX Developer
- Processing of intelligent function module dedicated instruction

4.4 RUN, STOP, PAUSE Operation Processing

The Basic model QCPU has three types of operation states; RUN, STOP and PAUSE states.

The Basic model QCPU operation processing is explained below:

(1) RUN Status Operation Processing

- (a) RUN status is when the sequence program operation is performed from step 0 to END (FEND) instruction to step 0 repeatedly.
- (b) When entering the RUN state, the output state saved at STOP by the parameter output-mode setting during STOP to RUN.
- (c) The processing time of switching from STOP to RUN until the beginning of sequence program operation changes with system configurations, but usually is 1 to 3 seconds.
However, this time may be longer depending on the conditions.

(2) STOP Status Operation Processing

- (a) STOP status is when the sequence program operations are stopped with the RUN/STOP/RESET switch or remote STOP is performed. (Refer to Section 7.6.1 for details regarding of remote STOP function.)
The STOP status is also caused by a stopping error.
- (b) When entering the STOP state, save the output state and turn off all output.
The device memory of other than the output (Y) is retained.

(3) PAUSE Status Operation Processing

- (a) The PAUSE state is when the sequence program operations are paused by remote PAUSE function while maintaining the output and device memory status. (Refer to Section 7.6.2 for details regarding of remote PAUSE function.)

(4) Basic model QCPU Operation Processing with RUN/STOP state

Operation processing RUN/STOP state	Sequence program operation processing	External output	Device memory (Y, M, L, S, T, C, D)
RUN to STOP	Executes up to the END instruction and stops.	OS saves the output state and all output are off.	Maintains the status immediately before the STOP state.
STOP to RUN	Starts at step 0.	Determined by the output mode of the PLC parameter at STOP to RUN.	Starts executing the operation from the status immediately before the STOP state. When a device initial value is designated, however, the value is set. Local devices are cleared.

POINT

The Basic model QCPU performs the following in any of RUN, STOP, and Pause state:

- I/O module refresh processing
- Data communication with the GX Developer and serial communication module
- Refresh process of MELSECNET/H and CC-Link

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.

4.5 Operation Processing during Momentary Power Failure

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

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

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

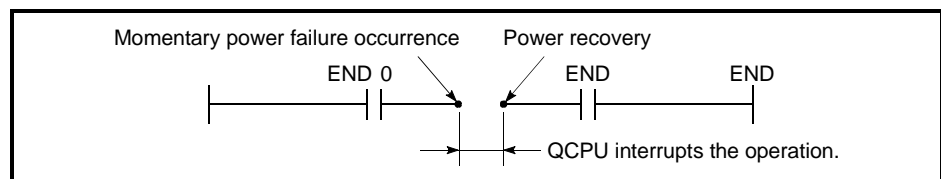


Fig.4.5 Operation Processing When Momentary Power Failure Occurs

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

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

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

- Power ON
- Resetting using RUN/STOP/RESET switch
- Remote setting using GX Developer

4.6 Data Clear Processing

(1) Data clear

The Basic 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
- (b) Device data with latch specification (latch clear valid)
- (c) Device data with latch specification (latch clear invalid)
- (d) File register data
- (e) Failure history data (when special register SD storage)

Data in (b) is cleared using the remote latch clear from the GX Developer function.

Refer to Section 7.6.4 for details regarding of the remote latch clear.

(2) Device latch specification

- (a) Specify the device latch (latch range setting) for each device in the device setting of the PLC parameter.

There are two types of latch range settings:

- 1) Valid latch clear key
Sets the latch range that can be cleared with latch-clear operation using the remote latch clear.
- 2) Invalid latch clear key
Sets the latch range that can not be cleared even with latch-clear operation using the remote latch clear.

- (b) The devices that were set to invalid RESET/L.CLR switch can only be cleared by an instruction or GX Developer clear operation.

- 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.

POINT

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".

REMARK

See following manual for the MOV/FMOV instruction.

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

4.7 Input/Output Processing and Response Lag

The Basic model QCPU features a refresh type input/output processing format in which a batch communication with the input/output module occurs at END processing. A direct communication format is also possible by using direct access inputs/outputs at the sequence program to enable direct communication with the input/output module when the sequence program instructions are executed. For details regarding direct inputs and direct outputs, refer to Sections 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 input/output modules occurs at END processing.

- (a) Batch reading of the input module ON/OFF information is executed in the Basic model QCPU's internal input device memory when END processing occurs. This ON/OFF data (in the input device memory) is then used for processing which occurs when a sequence program is executed.
- (b) The processing result of the output (Y) sequence program is output to the Basic 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 END processing occurs.

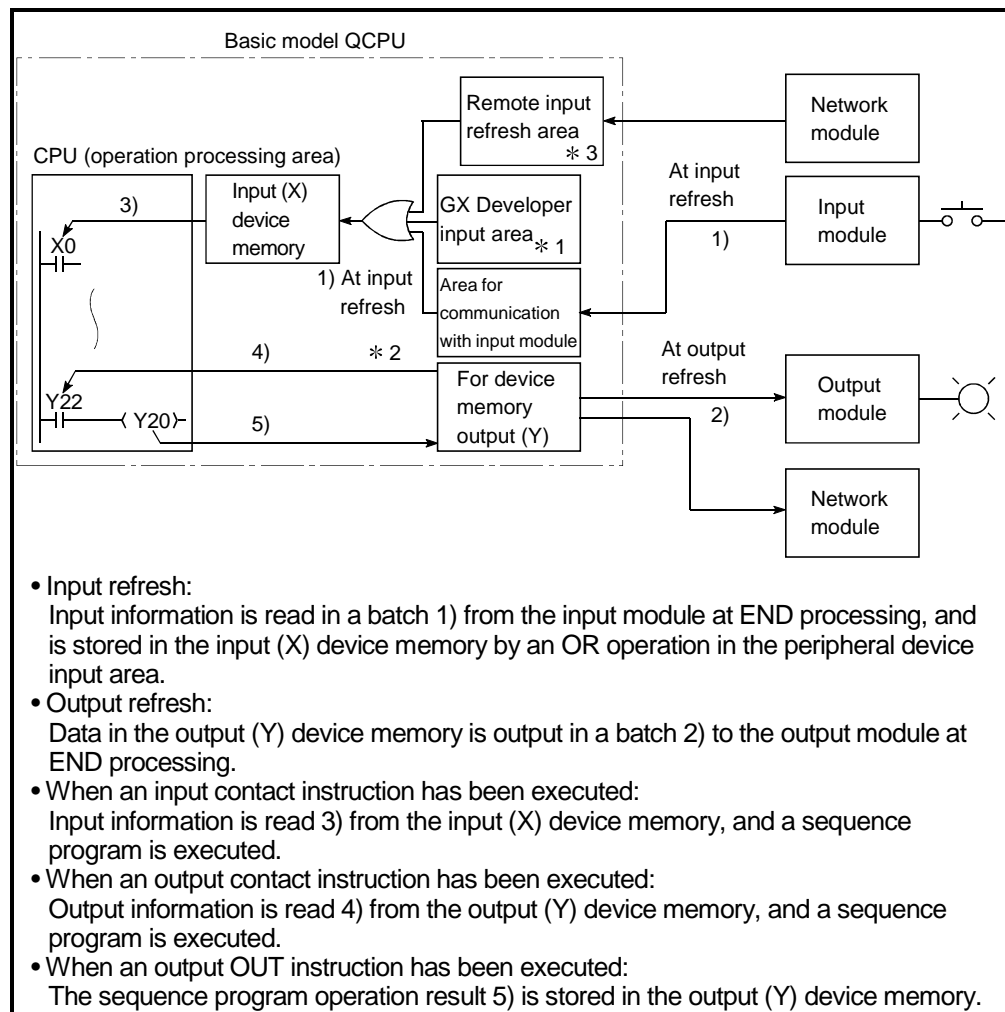


Fig.4.7 Input/Output Information Flow at Refresh Mode

REMARK

- *1: The GX Developer 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
 - Writhing 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 the GX Developer
 - A network refresh by the MELSECNET/H network system
 - Writhing from a serial communication module
 - CC-Link automatic refresh
- *3: The remote input/output 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

Output response lags of up to 2 scans can result from input module changes.
(See Fig.4.7)

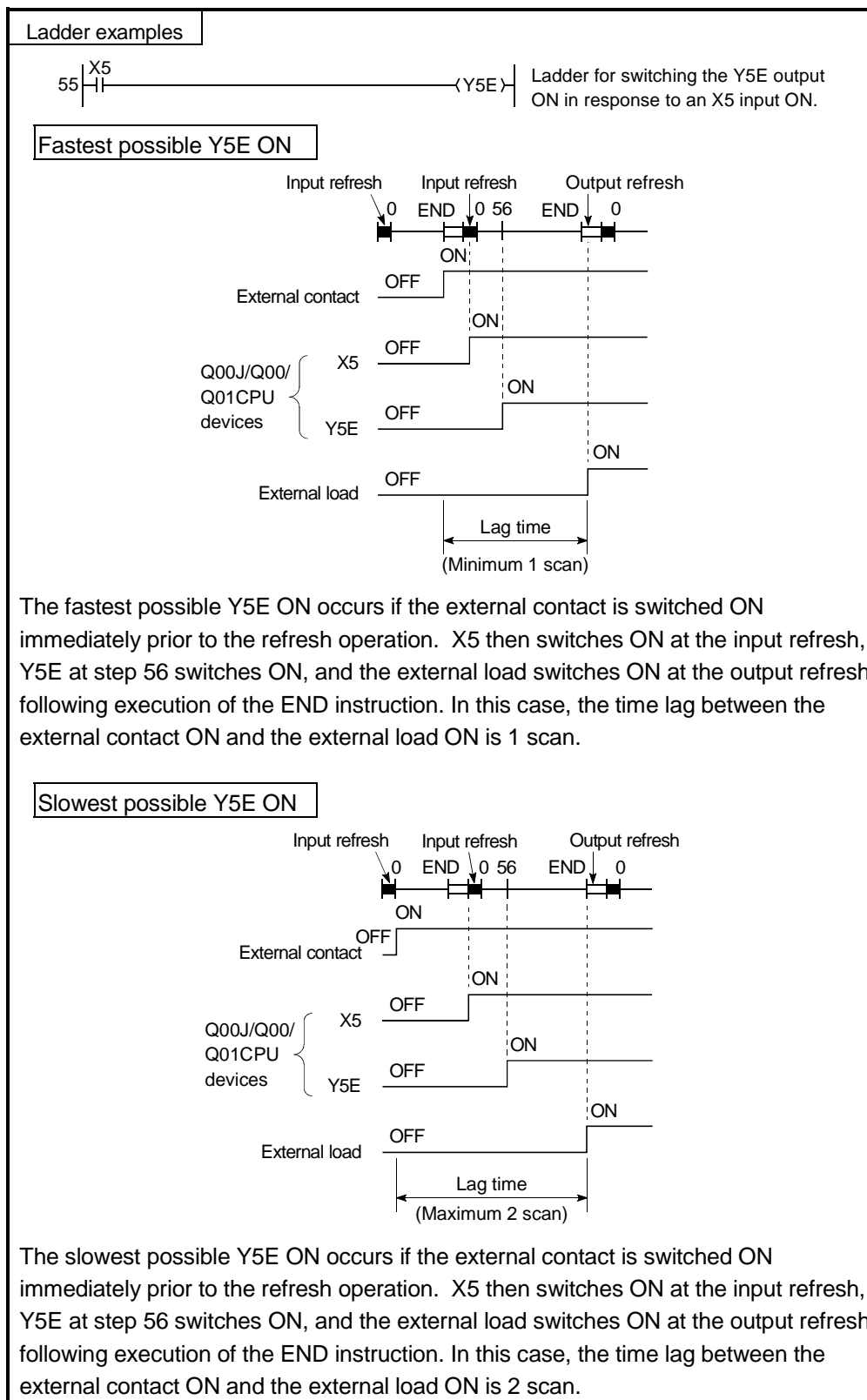


Fig.4.6 Output "Y" change in response to input "X" change

4.7.2 Direct mode

(1) Definition of direct mode

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

With Basic model QCPU, direct mode I/O processing can be executed by using direct access inputs (DX) and direct access outputs (DY).

See 10.2.1 for direct access inputs. See 10.2.2 for direct access outputs.

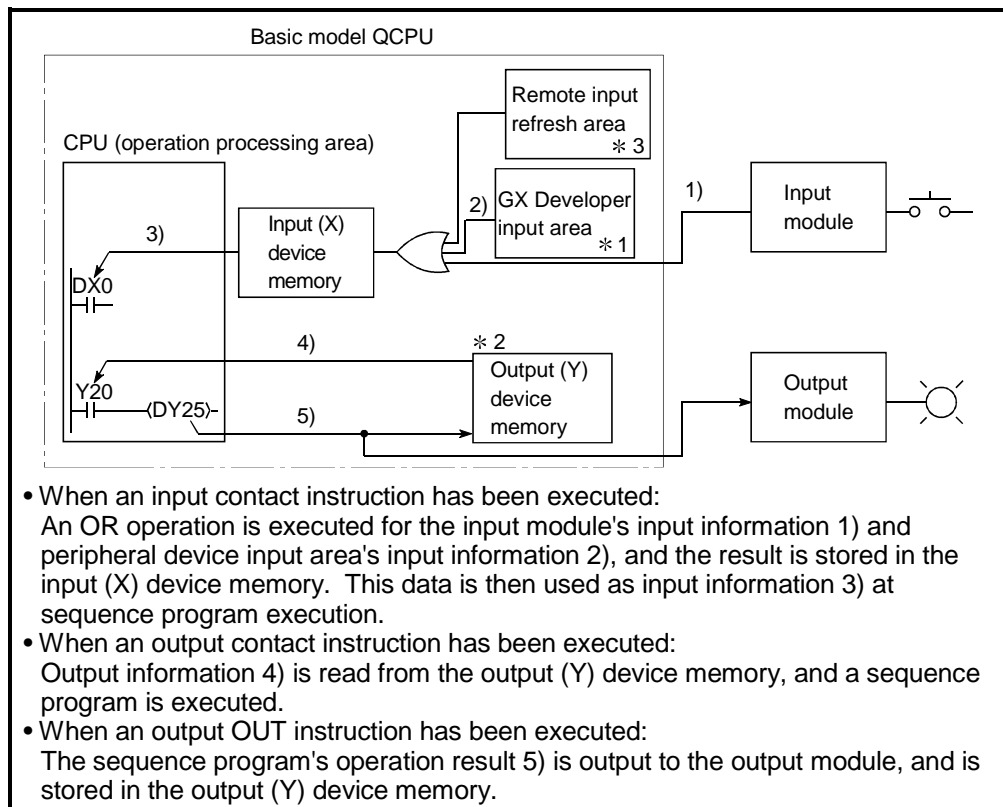


Fig.4.8 Input/Output Information Flow at Direct Mode

REMARK

*1: The GX Developer 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 the GX Developer
- A network refresh by the MELSECNET/H network system
- Writing from a serial communication module
- CC-Link automatic refresh

(2) Response lag

Output response lags of up to 1 scans can result from input module changes.

(See Fig.4.10)

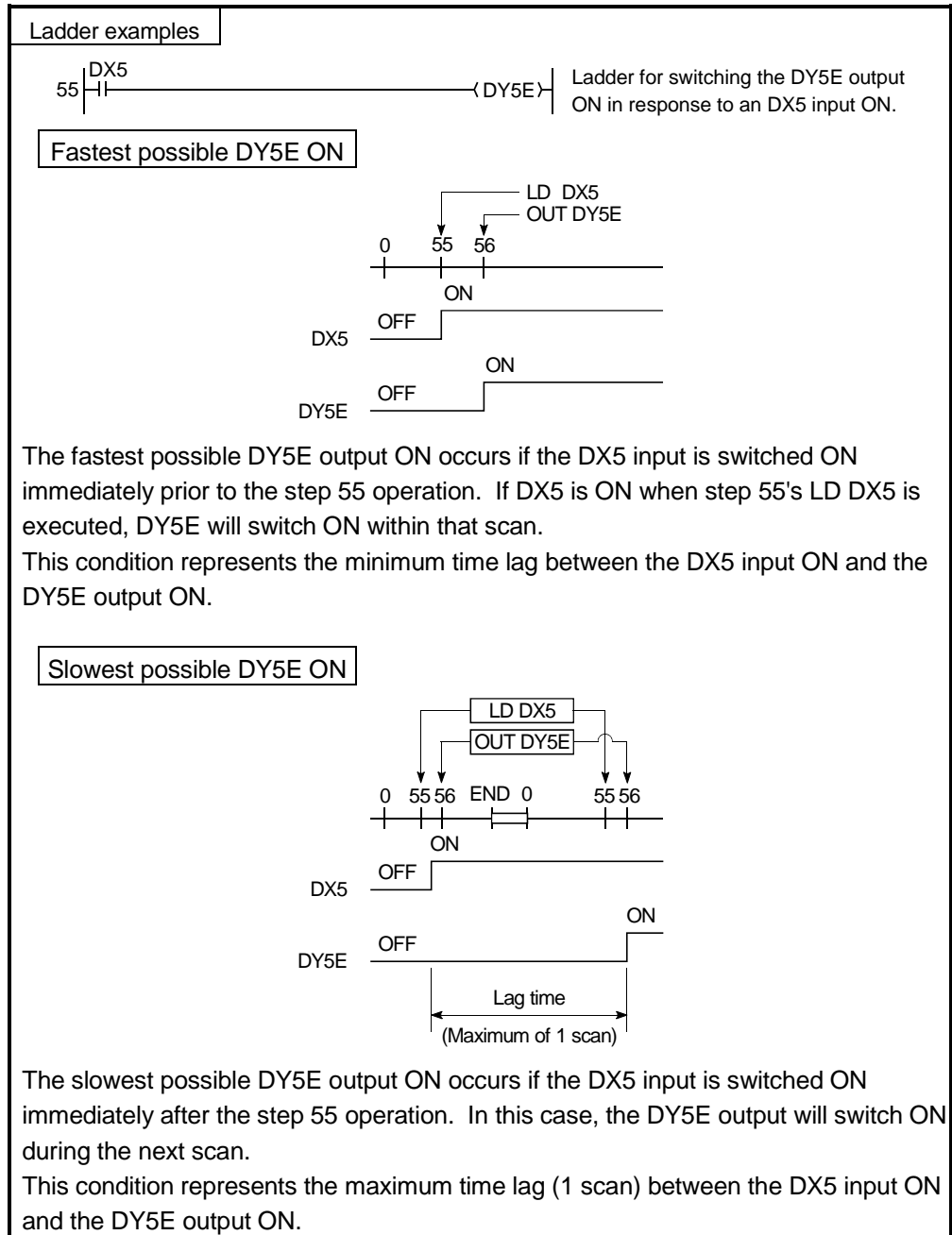


Fig.4.9 Output "Y" Change in Response to Input "X" Change

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 Basic model QCPU.

This method of expression is called "binary code" (BIN).

The hexadecimal (HEX) expression method in which BIN data are expressed in 4-bit units, and the BCD (binary coded decimal) expression method are also possible for the Basic model QCPU.

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

The numeric expressions for the 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				0
1	1				1				1
2	2				10				10
3	3				11				11
•	•				•				•
•	•				•				•
•	•				•				•
9	9				1001				1001
10	A				1010			1	0000
11	B				1011			1	0001
12	C				1100			1	0010
13	D				1101			1	0011
14	E				1110			1	0100
15	F				1111			1	0101
16	10			1	0000			1	0110
17	11			1	0001			1	0111
•	•				•				•
•	•				•				•
•	•				•				•
47	2F			10	1111			100	0111
•	•								
•	•								
•	•								
32766	7FFE	0111	1111	1111	1110			—	
32767	7FFF	0111	1111	1111	1111			—	
-32768	8000	1000	0000	0000	0000	1000	0000	0000	0000
-32767	8001	1000	0000	0000	0001	1000	0000	0000	0001
•	•								
•	•								
•	•								
-2	FFFE	1111	1111	1111	1110			—	
-1	FFFF	1111	1111	1111	1111			—	

(1) External numeric inputs to Basic model QCPU

When designating numeric settings for the Basic model QCPU from an external source (digital switch, etc.), a BCD (binary coded decimal) setting can be designated which is the same as a decimal setting.

However, because the Basic model QCPU operation is based on BIN, if the Basic model QCPU uses values designated in the BCD method as they are, it handles the values as BIN.

The Basic model QCPU operation based on such values will be different from the operation specified by the designated values.

A BIN instruction is therefore provided for the Basic model QCPU to convert BCD input data to the BIN data which is used by the Basic model QCPU.

A program which converts numeric data to BIN data can be created at the sequence program in order to allow numeric settings to be designated from an external source without regard to the corresponding BIN values.

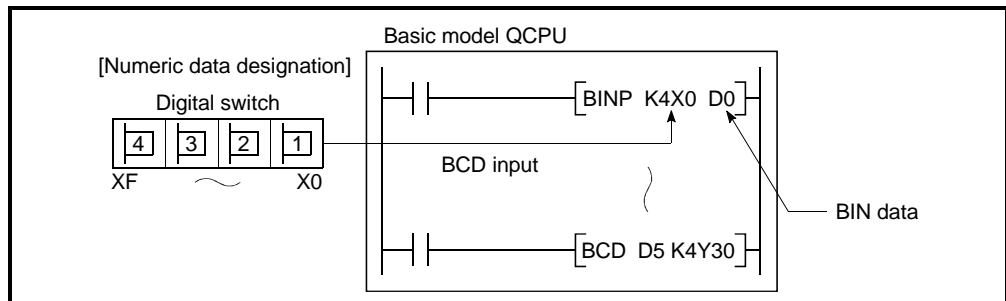


Fig.4.10 Digital Switch Data Input to Basic model QCPU

(2) External numeric outputs from Basic model QCPU

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

However, because the Basic model QCPU uses BIN data, it cannot be displayed at the digital display as is.

A BCD instruction is therefore provided for the Basic model QCPU to convert the BIN data to BCD data. A program which converts BIN data to BCD data can be created at the sequence program in order to display the output data in a manner identical to decimal data.

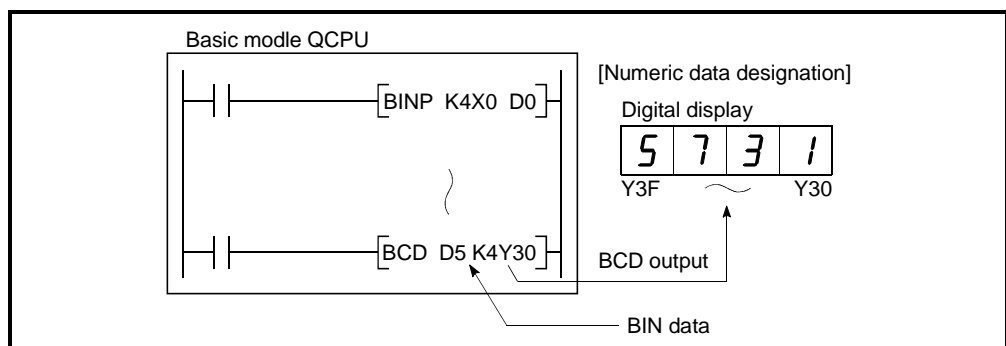


Fig.4.11 Digital Display of Data from Basic model QCPU

4.8.1 BIN (Binary Code)

(1) Binary code

In binary code, numeric values are expressed by numerals "0" (OFF) and "1" (ON) numerals.

When counting in the decimal system, a carry to the "tens" column occurs following 9 (8 to 9 to 10).

In the binary system, this carry occurs following 1 (0 to 1 to 10). The binary "10" therefore represents the decimal "2".

Binary values and their respective decimal values are shown in Fig.4.2 below.

Table 4.2 Binary and Decimal Numeric Value Comparison

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

(2) Binary numeric expression

- (a) Basic model QCPU registers (data registers, link registers, etc.) consist of 16 bits, with a "2ⁿ" 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 Basic model QCPU registers are shown in Fig.4.12 below.

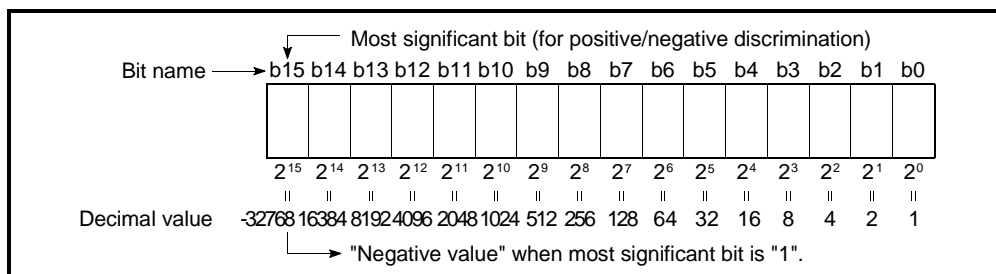


Fig.4.12 Numeric Expressions for Basic model QCPU Registers

(b) Usable numeric data for Basic model QCPU

As shown in Fig.4.11, the numeric expression range is -32768 to 32767. Therefore, numeric data within this range can be stored in the Basic model QCPU registers.

4.8.2 HEX (Hexadecimal)

(1) Hexadecimal notation

In the hexadecimal system, 4 bits of binary data are expressed by 1 digit.

4 bits of binary data can express 16 values (0 to 15).

In the hexadecimal system, values from 0 to 15 are expressed by 1 digit.

This is accomplished by using alphabetic characters following "9", with a carry occurring after "F", as follows:

A comparison of binary, hexadecimal, and decimal numeric expressions is shown in Table 4.3 below.

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

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

Therefore, as expressed in hexadecimal code, the numeric value range which can be stored is 0 to FFFF_H.

4.8.3 BCD (Binary Coded Decimal)

(1) BCD notation

BCD numeric expressions are binary expressions with a carry format identical to that of the decimal system.

As with the hexadecimal system, BCD expressions are the equivalent of 4 binary bits, although the BCD system does not use the A to F alphabetic characters.

A comparison of binary, BCD, and decimal numeric expressions is shown in Table 4.4 below.

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

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

Therefore, as expressed in BCD code, the range of numeric values to be stored is 0 to 9999.

4.9 Character String Data

(1) Character String Data

The Basic model QCPU uses ASCII code data.

(2) ASCII code character strings

ASCII code character strings are shown in the Table below.

"00_H" (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		
									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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	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	
									0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
									0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
									0	1	0	1													

POINT	
Character strings are available for the \$MOV instruction only.	

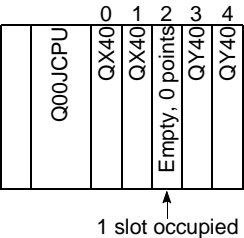
5 ASSIGNMENT OF I/O NUMBERS

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

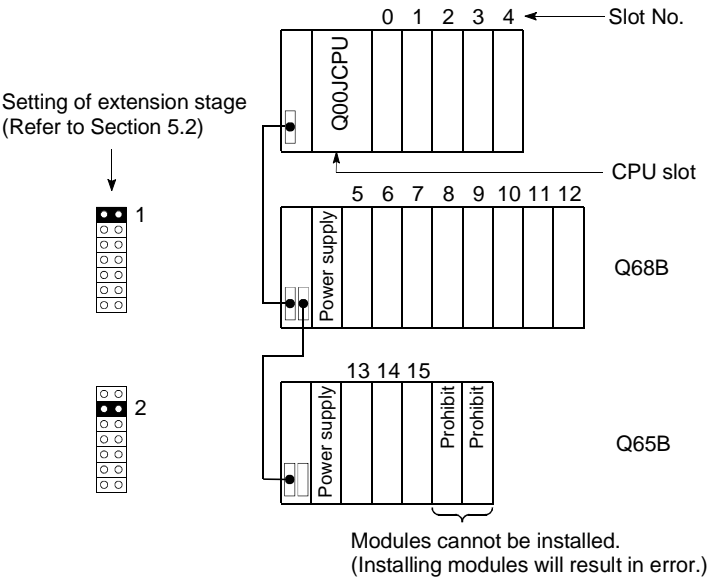
5.1 Relationship Between the Number of Stages and Slots of the Extension Base Unit

5.1.1 Q00JCPU

The Q00JCPU can configure a system with a total of three base units: one main base unit and two extension base units.
Note that the number of usable slots (modules) is 16 slots including vacant slots.
For example, if you set slot 2 for "vacant, zero points" as shown below, it occupies one slot.
Hence, the following system uses five slots, slot 0 - slot 4.



Install modules to slots 0 - 15.
Installing any module to slot 16 or later will result in an error (SP. UNIT LAY ERR.).



When the GOT has been bus-connected, one slot of extension base 1 is used.
Also one GOT occupies 16 I/O points.
When using the GOT, consider the number of slots and the number of I/O points.
Refer to the GOT Manual for details of bus-connecting the GOT.

7.12 Self-Diagnosis Function

(1) What is Self-Diagnosis Function?

- (a) The self-diagnosis is a function performed by Basic model QCPU itself to diagnose whether there is an error in the Basic model QCPU.
- (b) The self-diagnosis function's objective is to prevent Basic model QCPU erroneous operation and as preventive maintenance.
The self-diagnosis processing detects and displays the error when an error occurs when the Basic model QCPU power is turned on or during Basic model QCPU RUN mode. It also stops Basic model QCPU calculations.

(2) Processing for Error Detection

- (a) When Basic 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 program so that these devices can interlock with sequencers and mechanical systems.
- (b) Basic model QCPU stores 16 latest error codes. (Refer to Section 7.13.)
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) Basic 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 Basic model QCPU operation can change to.
 - 1) Basic model QCPU calculation stop mode
Stops the calculation at the point when the error is detected, and turns off all output (Y).
 - 2) Basic 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.
- (b) The following errors can set the calculation "continue/stop" in the parameter mode PLC RAS.
(All parameter defaults are set at "Stop".)
 - 1) Operation error
 - 2) Expanded command error
 - 3) Fuse blown
 - 4) I/O unit comparison
 - 5) Intelligent function module program execution errorFor 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 error checking can be set to "yes/no" in the following error checking in the parameter mode PLC RAS setting.

(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 short (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
	I/O module verification (Default... Stop) * 1	UNIT VERIFY ERROR	• When the END instruction is executed (Default... Yes) * 2
Handling error	Intelligent function module error	SP. UNIT ERROR	• When an instruction is executed
	Intelligent function module allocation error	SP.UNIT LAY ERR.	• When the power is turned on/when rest • When switched from STOP to RUN
	No parameter	MISSING PARA.	• When the power is turned on/when rest
	Boot error	BOOT ERROR	• When the power is turned on/when rest
	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
	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.

*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
Program error	Instruction code check	INSTRUCT CODE.ERR.	<ul style="list-style-type: none"> • When the power is turned on/when reset • When switched from STOP to RUN
	No END instruction	MISSING END INS.	<ul style="list-style-type: none"> • When the power is turned on/when reset • When switched from STOP to RUN
	Pointer setting error	CAN'T SET(P)	<ul style="list-style-type: none"> • When the power is turned on/when reset • When switched from STOP to RUN
	Pointer setting error	CAN'T SET(I)	<ul style="list-style-type: none"> • When the power is turned on/when reset • When switched from STOP to RUN
	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
PLC error	Watchdog error supervision	WDT ERROR	Always
	Program time exceeded	PRG.TIME OVER	Always
Annunciator check		F * * * *	When an instruction is executed

* 1: Can be changed to "continues" in the GX Developer function parameter setting.

7.12.1 LED display when error occurs

When an error occurs, the ERR. LED located on the front of Basic model QCPU turns on.

Refer to Section 7.19 for the details of the ERR. LED operation.

7.12.2 Cancel error

Basic model QCPU error cancel operation can be performed only for error that can continue the Basic model QCPU operation.

(1) Cancellation of error

(a) Procedures for cancellation of error

The error cancel is performed in the following manner:

- 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 cancellation of error

When the CPU is recovered from canceling the error, the special relay, special register, and LED affected by the error are set to the state 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
When error cancellation is performed by storing the code of the error to cancel is stored in SD50, the lower 1 digits of the code number is ignored.

7.13 Failure History

The Basic model QCPU can store the failure history (results detected from the self-diagnosis function and the time) in the memory.

POINT	
The detection time uses the Basic model QCPU internal clock, so make sure to set the correct time when first using Basic model QCPU.	

(1) **Storage Area**

The latest 16 failures are stored in the latched Basic model QCPU failure history storage memory.

(2) **Stored data**

If the same error occurs more than once during PLC power-on, it is stored into the failure history storage memory only once.

(3) **Failure History Clearing Method**

The failure history storage memory are cleared using the failure history clear in the GX Developer PLC diagnosis mode.

Data files stored in Basic model QCPU failure history storage memory can be cleared with a failure history clear.

7.14 System Protect

Basic model QCPU has a few protection functions (system protect) for the program changes to processing of general data obtained from a third party other than the designer (access processing 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
File unit	Program Device comments Device initial values	The attributes for a file is changed to the following: 1) Read/Write display prohibit 2) Write prohibit	Change the attributes for the file in the Password Registration.	Always	

*The control instruction, read/write display, and write are mentioned above are as follows:

Item	Description
Read/Write display	Program read/write operations.
Write	Operation that writes the program and tests.

7.14.1 Password registration

Password is used to prohibit the data read and write of the program and comments in Basic model QCPU from a GX Developer peripheral device.

The Password Registration is set for the specified memory (program memory/standard memory) program file, device comment file, and device initial file.

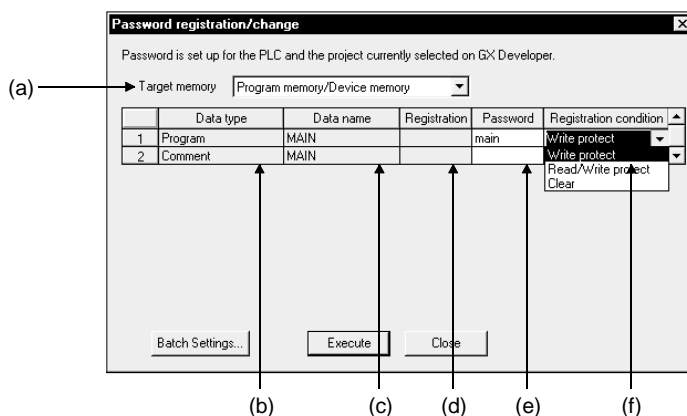
There are two descriptions of items to be registered.

- 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 perform the password registration, select GX Developer Online → Password setup/keyword set up for writing to PLC → Register password.



Each item is described below:

- (a) Target memorySet the memory storing the file whose password is to be registered or changed.
- (b) Data type.....Specifies the type of a file stored in the target memory.
- (c) Data nameDisplays a filename of a file stored in the target memory.
- (d) Registration.....Displays an asterisks " * " that indicates a password-protected file.
- (e) PasswordDefines or changes a password.
Setting a password allows you to set the registration condition.
- (f) Registration Condition
- 1) Write ProtectWrite operation is restricted by the password.
(Reading is possible.)
 - 2) Read/Write protect.....Read/Write operation is restricted by the password.
 - 3) Clear.....Password is cleared.
(Sets password currently registered in the Password.)

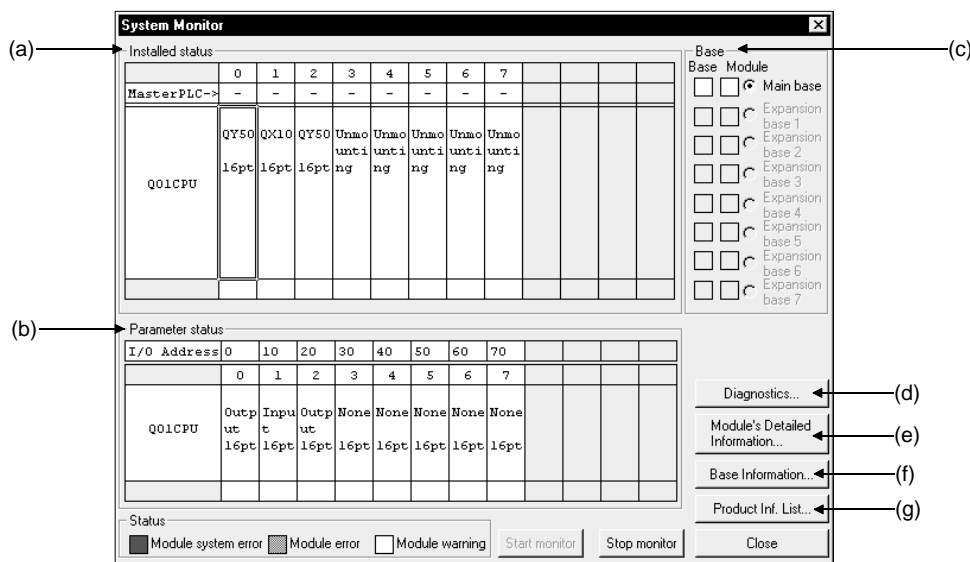
POINT

- (1) Password-protected files are limited to program files, and device comment 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: PLC memory format
 - Standard ROM: storage of program memory data onto ROM
 Take notes of the password registered and keep it on hand.

7.15 GX Developer system monitor

It is possible to confirm the following information for Basic model QCPU 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

You can confirm the types and points of the modules loaded on the selected base unit.

"Not installed" will be displayed for slots in which modules have not been mounted.

When slots have been set as "Empty" with the PLC parameter's I/O allocation setting, the module's model will not be displayed when if a module has been mounted.

(b) Operation 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 PC 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 module column displays an error or warning status if even one module is faulty.

(d) Diagnostics

This function is used to confirm the status of Basic 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 the relevant intelligent function module for details on the detailed information for 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) List of product information

Enables the individual information for mounted CPU module, input/output modules and intelligent function modules to be confirmed (Type, Series, Model name, Points, I/O No., Serial No., function version.)

Serial No. Function version

Slot	Type	Series	Model name	Points	I/O No.	Master PLC	Serial No	Ver
PLC	PLC	Q	Q01CPU	-	-	-	0305100000000000	A
0-0	Intelli.	Q	QJ71E71	32pt	0000	-	0208100000000000	B
0-1	-	-	None	-	-	-	-	-
0-2	-	-	None	-	-	-	-	-
0-3	-	-	None	-	-	-	-	-
0-4	-	-	None	-	-	-	-	-
0-5	-	-	None	-	-	-	-	-
0-6	-	-	None	-	-	-	-	-
0-7	-	-	None	-	-	-	-	-

CSV file creating Close

7.16 LED Display

The LEDs that indicate the operation statuses of the Basic model QCPU are provided on the front panel of the Basic model QCPU. The indications of the LEDs are described below.

(1) The details of the LED display are shown below:

LED name	Display Description
RUN	<p>Indicates Basic model QCPU operation status.</p> <p>On : When operating with the RUN/STOP/RESET switch at "RUN".</p> <p>Off : When stopped with the RUN/STOP/RESET switch at "STOP".</p> <p>Or when an error that stops operation is detected.</p> <p>Flicker : When writing parameters and program during STOP, and when setting the RUN/STOP/RESET switch from [STOP] → [RUN]. Perform the following operations in order to illuminate the RUN LED after program writing.</p> <ul style="list-style-type: none"> • Set the RUN/STOP/RESET switch to [RUN] → [STOP] → [RUN]. • Reset the system with the RUN/STOP/RESET switch. • Switch on the power to the PLC again. <p>Perform the following operations in order to illuminate the RUN LED after parameter writing.</p> <ul style="list-style-type: none"> • Reset the system with the RUN/STOP/RESET switch. • Switch on the power to the PLC again. <p>(When the RUN/STOP/RESET 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.)</p>
ERR.	<p>Indicates Basic model QCPU error detection status.</p> <p>On : When a self-diagnosis error that does not stop the operation is detected. (Set the operation error set mode to "continue" in the parameter mode PLC RAS setting.)</p> <p>Off : Normal</p> <p>Flicker : When an error that stops the operation is detected.</p> <p>When the CPU is reset with the RUN/STOP/RESET switch. Goes off on completion of reset.</p>
POWER	<p>Indicates the 5VDC output status of the power supply built in the Q00JCPU.</p> <p>On : Normal output of 5VDC</p> <p>Off : PLC power off or 5VDC output error</p>

(2) How to turn off the ERR. LED

To turn off the ERR. LED that is on, remove the cause of the error and then operate the special relay SM50 and special register SD50 to cancel the error.
(This does not apply to reset operation.)

REMARK

Refer to Section 7.12.2 for canceling the error.

7.17 Serial Communication Function (Usable with the Q00CPU or Q01CPU)

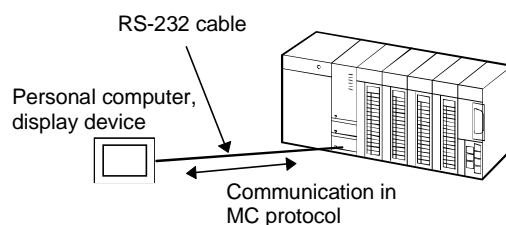
The serial communication function is designed to make communication in the MC protocol (*1) by connecting the RS-232 interface of the CPU module and personal computer, display device or the like by an RS-232 cable.

The serial communication function is not used for connection of GX Developer or GX Configurator and the CPU module.

Communication using the serial communication function can be made by the Q00CPU or Q01CPU.

(The Q00JCPU does not have the serial communication function.)

The following explains the specifications, functions and various settings needed to make communication with a personal computer, display device or the like using the serial communication function.



- *1 The MC protocol is the abbreviation of the MELSEC communication protocol. The MELSEC communication protocol is a name of the communication method to access from the mating equipment to the QCPU in accordance with the communication procedure of the Q series PLC (e.g. serial communication module, Ethernet interface module).
- Serial communication function enables the communication of data in ASCII format and Binary format.

POINT

The CPU that can make communication with a personal computer, Display device or the like using the serial communication function is only the Q00CPU/Q01CPU that is connected with the personal computer, Display device or like. Communication cannot be made with the other station of MELSECNET/H, Ethernet or CC-Link via the Q00CPU/Q01CPU that is connected with the personal computer, Display device or the like.

(1) Specifications

(a) Transmission specifications

The following table indicates the transmission specifications of RS-232C used for the serial communication function of the CPU module.

Use the serial communication function after making sure that the specifications of the personal computer, Display device or the like match those of the following table.

Item	Default	Setting Range
Communication system	Full duplex communication	—
Synchronization system	Asynchronous system	—
Transmission speed *1	19.2kbps	9.6kbps, 19.2kbps, 38.4kbps, 57.6kbps, 115.2kbps
Data format	Start bit: 1 Data bit: 8 Parity bit: Odd Stop bit: 1	—
MC protocol format *2 (Automatic judgment)	Format 4 (ASCII) Format 5 (binary)	—
Frame *2	QnA-compatible 3C frame QnA-compatible 4C frame	—
Transmission control	DTR/DSR control	—
Sum check *1	No	Yes, No
Transmission wait time *1	No wait	No wait, 10ms to 150ms (10ms increments)
Write during RUN setting *1	Not enabled	Enabled, Not enabled
Extension distance	15m	—

*1: Can be set in the PLC parameter setting of GX Developer.

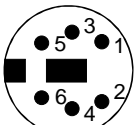
*2: The relationships between the MC protocol formats and frames are indicated in the following table.

Function		Format 4	Format 5
Communication in ASCII code	QnA-compatible 3C frame	○	×
	QnA-compatible 4C frame	○	×
Communication in binary code	QnA-compatible 4C frame	×	○

○: Usable, ×: Unusable

(b) RS-232 connector specifications

The following table indicates the applications of the RS-232 connector of the Q00CPU/Q01CPU.

Appearance	Pin No.	Signal Symbol	Signal Name
 Mini-Din 6 pins (female)	1	RD (RXD)	Receive data
	2	SD (TXD)	Send data
	3	SG	Signal ground
	4	—	—
	5	DSR (DR)	Data set ready
	6	DTR (ER)	Data terminal ready

(c) RS-232 cable

The following RS-232 cable can be used for connection of the Q00CPU/Q01CPU with the personal computer, GOT or the like.

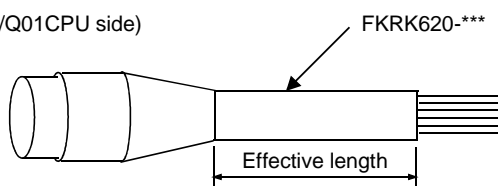
- QC30R2 (cable length: 3m)

- FKRK620-*** (KURAMO ELECTRIC) manufactured

Cable with a mini-DIN connector on one side and without connector on the other side

*** indicates the cable length, which can be specified up to 15m in 0.1m increments

(Q00/Q01CPU side)



Signal layout of Q00/Q01CPU side
connector of FKRK620-***

Pin No.	1	2	3	4	5	6	Metal
Signal name	RD	SD	SG	—	DR	ER	shell
Wire core	Yellow	Brown	Black	Red	Blue	Green	Shield

(2) Functions

The serial communication function allows the MC protocol commands in the following table to be executed.

Refer to the following manual for details of the MC protocol.

• Q-Compatible MELSEC Communication Protocol Reference Manual

Function			Command	Processing	Processing Points
Device memory	Batch read	in bits	0401 (00□1)	Reads bit devices by 1 point.	ASCII: 3584 points BIN: 7168 points
		in words	0401 (00□0)	Reads bit devices by 16 points. Reads word devices by 1 point.	480 words (7680 points) 480 words
	Batch write *1	in bits	1401 (00□1)	Writes to bit devices by 1 point.	ASCII: 3584 points BIN: 7168 points
		in words	1401 (00□0)	Writes to bit devices by 16 points Writes to word devices by 1 point.	480 words (7680 points) 480 words
	Random read	in words	0403 (00□0)	Reads bit devices by 16 points or 32 points by designating the devices at random. Reads word devices by 1 point or 2 points by designating the devices at random.	96 points
	Test *1 (Random write)	in bits	1402 (00□1)	Sets/resets bit devices by 1 point by designating the devices at random.	94 points
		in words	1402 (00□0)	Sets/resets bit devices by 16 points or 32 points by designating the units at random. Writes to word devices by 1 point or 2 points by designating the devices at random.	*2
	Monitor registration	in words	0801 (00□0)	Registers the bit devices to be monitored by 16 points or 32 points.	96 points
				Registers the word devices to be monitored by 1 point or 2 points.	96 points
	Monitor	in words	0802 (00□0)	Monitors the devices registered for monitoring.	Number of points registered for monitor

*1: When performing write during RUN of the CPU module, set write during RUN setting to "Enable".

*2: Set the number of processing points within the range of the following expression.

(Number of word access points) × 12 + (number of double word access points) × 14 ≤ 960

- One point of a bit device corresponds to 16 bits for word access or to 32 bits for double word access.
- One point of a word device corresponds to one word for word access or to two words for double word access.

(3) Accessible devices

Class	Device		Device Code	Device Number Range *1 (Default Value)	Write	Read
Internal system device	Function input		FX	000000 to 00000F	×	×
	Function output		FY	000000 to 00000F		
	Function register		FD	000000 to 000004		
	Special relay		SM	000000 to 001023	○	○
	Special register		SD	000000 to 001023		
Internal user device	Input		X	000000 to 0007FF		
	Output		Y	000000 to 0007FF		
	Internal relay		M	000000 to 008191		
	Latch relay		L	000000 to 002047		
	Annunciator		F	000000 to 001023		
	Edge relay		V	000000 to 001023		
	Link relay		B	000000 to 0007FF		
	Data register		D	000000 to 011135		
	Link register		W	000000 to 0007FF		
	Timer	Contact	TS	000000 to 000511		
		Coil	TC			
		Current value	TN			
	Retentive timer	Contact	SS	—		
		Coil	SC			
		Current value	SN			
	Counter	Contact	CS	000000 to 000511		
		Coil	CC			
		Current value	CN			
	Special link relay		SB	000000 to 0003FF		
	Special link register		SW	000000 to 0003FF		
	Step relay *2		S	000000 to 002047		
	Direct input		DX	000000 to 0007FF	○	
	Direct output		DY	000000 to 0007FF		
	Index register		Z	000000 to 000009		
	File register		R	000000 to 032767		
			ZR	000000 to 007FFF		

○: Read/write enabled, ×: Write disabled

*1: The device number ranges given in the above table are default values.

When you have changed the number of device points on the Q00CPU or Q01CPU, use the new device number range.

*2: Since Basic model QCPU is not compatible with the SFC function, the contents of the step relays, if read, cannot be used as data.

(4) Setting of transmission specifications

Use the serial communication setting PLC parameters to set the transmission speed, sum check, transmission wait time and write during RUN setting of the serial communication function.

- (a) When using the serial communication function to make communication with the personal computer, Display device or the like, specify "Use serial communication".
- (b) The default values of the transmission speed, sum check, transmission wait time and write during RUN setting are displayed.
You can change the transmission speed, sum check, transmission wait time and write during RUN setting according to the specifications of the external device.

Click here to use the serial communication function.

Selecting "Use serial communication" allows you to change the settings.

(5) Instructions

- (a) Connection can be switched to GX Developer during communication with the personal computer, Display device or the like using the serial communication function.
However, the personal computer, Display device or the like that was making communication using the serial communication function results in a communication error.
Refer to the manual of the used device for the way to start the personal computer, Display device or the like when the CPU module is reconnected with the personal computer, Display device or the like.
- (b) When you specify "Use serial communication", the transmission speed changed using GX Developer is not made valid.

POINT

The data set in serial communication setting is made valid when:

- The PLC is powered on; or
- The Q00/Q01CPU is reset.

(6) Error codes for communication made using serial communication function

The following table indicates the error codes, error definitions and corrective actions that are sent from the Q00CPU or Q01CPU to the external device when errors occur during communication made using the serial communication function.

Error Code (Hexadecimal)	Error Item	Error Definition	Corrective Action
4000H to 4FFFH	—	(CPU detected error) * Error that occurred in other than the serial communication function	• Refer to the Appendices of the Basic Model QCPU (Q Mode) User's Manual (Hardware Design, Maintenance and Inspection), and take corrective action.
7153H	Frame length error	• The length of the received message is outside the permissible range.	• Reconsider the sent message. • The number of access points of the message should be within the permissible range.
7155H	Unregistered monitor error	• A monitor request was given before monitor registration was made.	• Give a monitor request after registering the device to be monitored.
7164H	Requested data error	• The requested data or device specifying method is in error.	• Check and correct the sent message/requested data of the device on the other end, and restart communication.
7167H	Disabled during RUN	• A write command was specified for the setting of write during RUN disable.	• Change the setting to write during RUN enable and restart communication.
7168H		• The command specified cannot be executed during RUN.	• Set the CPU module to STOP and restart communication.
716DH	Monitor registration error	• The QnA-compatible 3C/4C frame was not used for monitor registration.	• Perform monitor registration again.
7E40H	Command error	• The command or sub-command specified does not exist.	• Check and correct the sent message of the device on the other end and restart communication.
7E41H	Data length error	• The number of points specified for random read/write exceeds the number of points enabled for communication.	• Check and correct the sent message of the device on the other end and restart communication.
7E42H	Data count error	• The requested number of points exceeds the range of the command.	• Check and correct the sent message of the device on the other end and restart communication.
7E43H	Device error	• The device specified does not exist. • The device specified cannot be specified for the corresponding command.	• Check and correct the sent message of the device on the other end and restart communication.
7E47H	Continuous request error	• The next request was received before the reply message was returned.	• Do not give continuous requests from the device on the other end. • Match the monitoring time of timer 1 with the time-out period of the device on the other end.
7E4FH	Device point count error	• The number of access points is incorrect.	• Check and correct the sent message of the device on the other end and restart communication.
7E5FH	Request destination module I/O number error	• The request destination module I/O number is in error.	• Correct the module I/O number of the data send destination.
7E64H	Registered point count range error	• The number of registered points (word/bit) is outside the range.	• Correct the set value of the registered points (word/bit).
7F01H	Buffer full error	• The next data was received before completion of received data processing.	• Perform handshake with the device on the other end, for example, to increase the sending intervals.
7F21H	Receive header section error	• The command (frame) section specified is in error.	• Check and correct the sent message of the device on the other end and restart communication.
		• The ASCII code received cannot be converted into binary.	

Error Code (Hexadecimal)	Error Item	Error Definition	Corrective Action
7F22H	Command error	<ul style="list-style-type: none"> The command or device specified does not exist. The remote password length is in error. 	<ul style="list-style-type: none"> Check and correct the sent message of the device on the other end and restart communication.
7F23H	MC protocol message error	<ul style="list-style-type: none"> The data (e.g. ETX, CR-LF) specified after the character part does not exist or in error. 	<ul style="list-style-type: none"> Check and correct the sent message of the device on the other end and restart communication.
7F24H	Sum check error	<ul style="list-style-type: none"> The calculated sum check does not match the received sum check. 	<ul style="list-style-type: none"> Reconsider the sum check of the device on the other end.
7F67H	Overrun error	<ul style="list-style-type: none"> The next data was received before the Q00CPU or Q01CPU completed receive processing. 	<ul style="list-style-type: none"> Reduce the communication speed and restart communication. Check the Q00CPU or Q01CPU for occurrence of an instantaneous power failure. (For the Q00CPU or Q01CPU, use the special register SD53 to check.) When an instantaneous power failure has occurred, remove its cause.
7F69H	Parity error	<ul style="list-style-type: none"> The parity bit setting does not match. 	<ul style="list-style-type: none"> Match the setting of the Q00CPU or Q01CPU with that of the device on the other end.
7F6AH	Buffer full error	<ul style="list-style-type: none"> The receive buffer of the OS overflowed, resulting in skipped receive data. 	<ul style="list-style-type: none"> Exercise DTR control to make communication, preventing a buffer full error.
F000H	—	<ul style="list-style-type: none"> Error detected by the MELSECNET/H network system. 	<ul style="list-style-type: none"> Refer to the Q Corresponding MELSECNET/H Reference Manual (PLC to PLC Network) and take corrective action.

8 COMMUNICATION WITH INTELLIGENT FUNCTION MODULE

(1) Description of intelligent function modules

Basic model QCPU allows the use of the Q series-compatible intelligent function modules.

The intelligent function module is a module that allows Basic 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

The intelligent function module is equipped with memory (buffer memory) to store the data received from or output to external devices.

Basic model QCPU reads/writes the data from/to the buffer memory.

8.1 Communication Between Basic model QCPU and Q-series Intelligent Function Modules

The following methods enable the communication between Basic model QCPU and intelligent function modules:

- Initial setting or automatic refresh setting using the GX Configurator
- Intelligent function module device
- Instructions dedicated for intelligent function modules
- FROM/TO instruction

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	Basic model QCPU reset	STOP → RUN	Instruction execution	END processing	Basic model QCPU *2	Intelligent *3
GX Configurator	Initial setting	○	○	○	—	—	○	—
	Automatic refresh setting	—	—	—	—	○	○	—
Intelligent function module device *4		—	—	—	○	—	○	—
Instructions dedicated for intelligent function modules *4		—	—	—	○	—	○	—
FROM/TO instruction *4		—	—	—	○	—	○	—

Communication timing○: Executed —: Not executed

Storage location○: 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 Basic model QCPU or in an intelligent function module.

*2: Represents the internal memory of Basic model QCPU.

*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.

(6) Service processing time

(a) Monitoring using GX Developer

Processing time (unit: ms) for monitoring using GX Developer.
Added when monitoring is performed on GX Developer.

Function	When Connected to RS-232 of Host CPU Module			When Connected to Other Station *4		
	Q00JCPU	Q00CPU	Q01CPU	Q00JCPU	Q00CPU	Q01CPU
Read of program from PLC *1	1.6	1.3	1.2	2.3	1.9	1.8
Device monitor *2	1.2	1.0	0.9	2.4	2.0	1.9
Online program correction *3	1.0	1.0	1.0	1.9	1.6	1.5

*1: Time taken to read an 8k-step program from program memory

*2: Time taken when 32 points have been set in registration monitor

*3: Time taken when a 100-step ladder has been added

*4: Indicates that access is made via MELSECNET/H, Ethernet, CC-Link or serial communication module.

(b) Communication with serial communication module or Ethernet interface module

Time to make communication with the serial communication module or Ethernet interface module.

Refer to the following manual for communication time with the corresponding module.

- Q-Corresponding MELSEC Communication Protocol Reference Manual

(7) Common processing time

Common processing of the CPU module processed in the system.

The common processing times are the values in the following table.

	CPU Type		
	Q00JCPU	Q00CPU	Q01CPU
Common processing time (ms)	0.70	0.55	0.50

The processing times in the above table assume that the constant scan function is not used.

When the constant scan function is used, wait processing is performed for the period of constant scan setting shortage.

11.3 Other Processing Times

(1) Constant scan accuracy

CPU Type	Without Monitor, Without User Interrupt	With Monitor, Without User Interrupt	Without Monitor, With User Interrupt	With Monitor, With User Interrupt
Q00JCPU	0.20	0.90	Interrupt program execution time (Refer to (b) in Section 11.2 (3).)	Sum of the following times 1) Time indicated in "With Monitor, Without User Interrupt" field on the left 2) Sum of interrupt program execution times
Q00CPU	0.12	0.60		
Q01CPU	0.10	0.50		

Unit: ms

With monitor: Indicates the status in which monitor is being performed with GX Developer connected or communication with the external device is being made using the serial communication function.

Without monitor: Indicates the status in which communication using GX Developer or the serial communication function is not being made.

12 PROCEDURE FOR WRITING PROGRAM TO BASIC MODEL QCPU

This chapter describes the procedure for writing program created at the GX Developer to the Basic model QCPU.

12.1 Items to Consider when Creating Program

12

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's program capacity is adequate for storing the program and parameter data.

The program capacities of the CPUs are shown below:

- Q00JCPU : 8 k steps
- Q00CPU : 8 k steps
- Q01CPU : 14 k steps

(2) Applications of devices and setting of their numbers of points

Consider the applications of the devices used in a program and their number of points.

Refer to Chapter 10 for the devices usable with the Basic model QCPU.

(3) ROM operation considerations

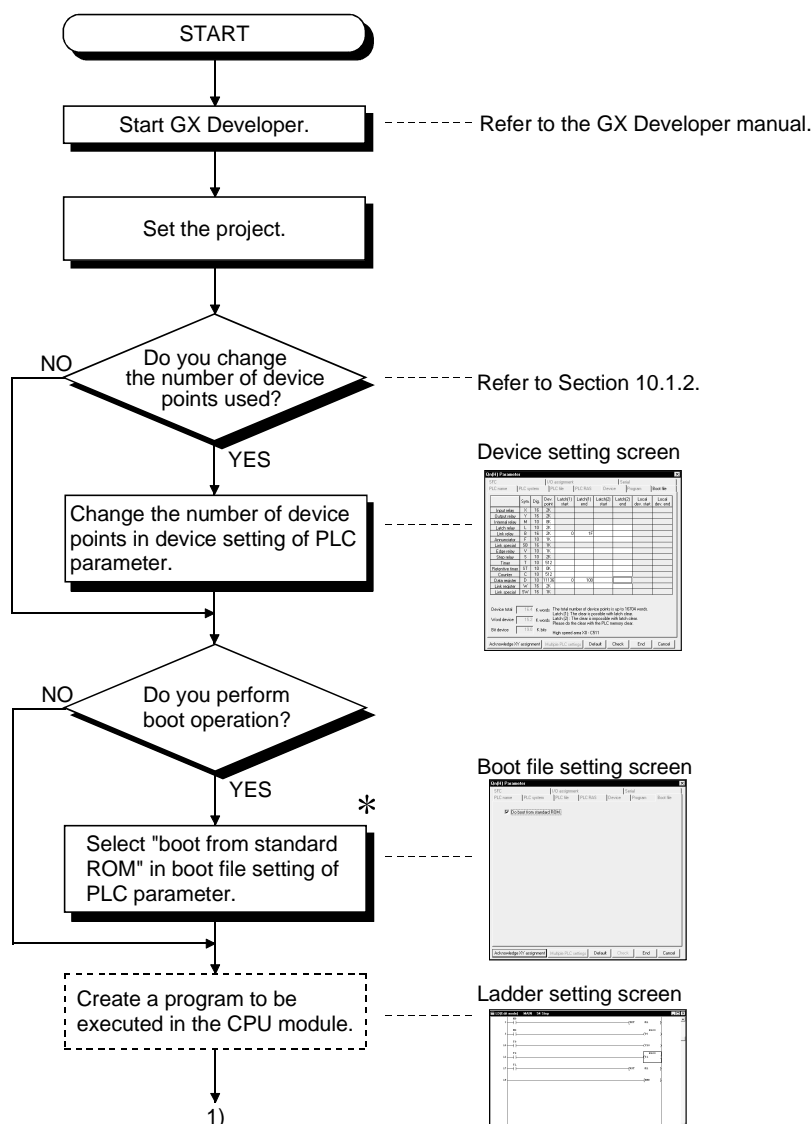
When performing ROM operation, make the boot file setting of PLC parameter.

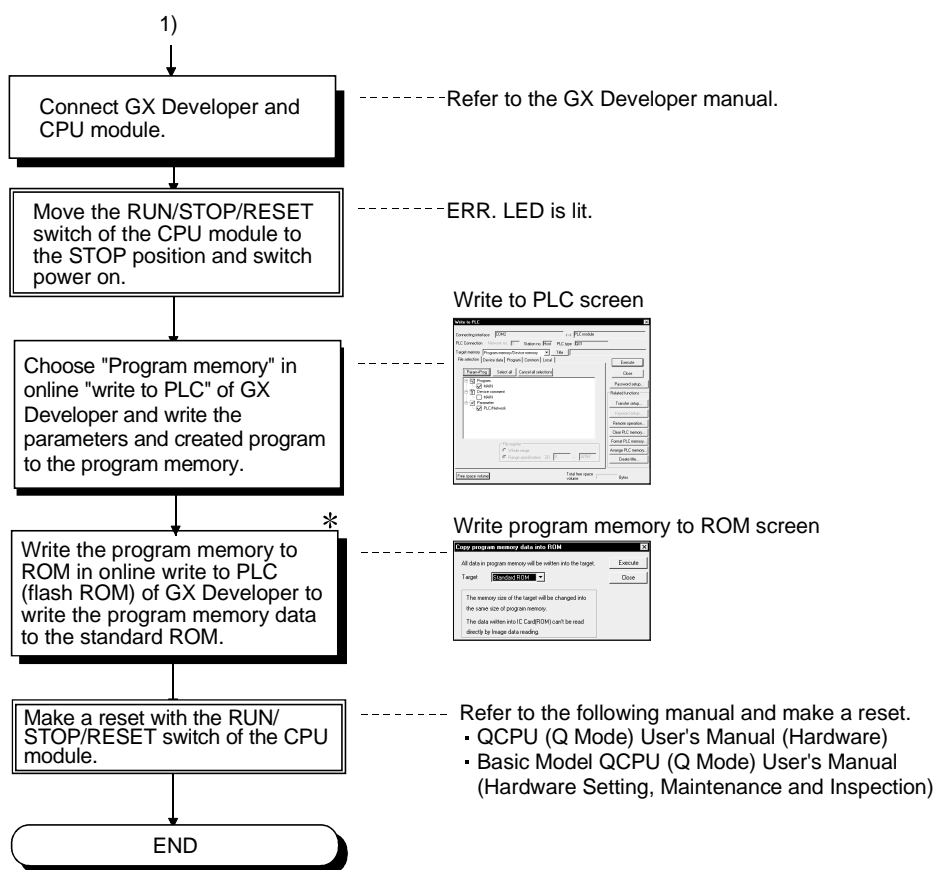
12.2 Procedure for writing program to the Basic model QCPU

The procedure for writing program and parameter created at the GX Developer to the Basic model QCPU standard ROM is shown below.

When writing program and parameter to the Basic 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 Basic model QCPU.





APPENDICES

APP

APPENDIX 1 Special Relay List

Special relays, SM, are internal relays whose applications are fixed in the programmable controller.

For this reason, they cannot be used by sequence programs in the same way as the normal internal relays.

However, they can be turned ON or OFF as needed in order to control the CPU 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"> • Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed. <Set by> <ul style="list-style-type: none"> S : Set by system U : Set by user (in sequence program or test operation at a peripheral device) S/U : Set by both system and user <When set> → indicated only if setting is done by system. <ul style="list-style-type: none"> Each END : Set during each END processing Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN) Status change : Set only when there is a change in status Error : Set when error is generated Instruction execution : Set when instruction is executed Request : Set only when there is a user request (through SM, etc.)

For details on the following items, see these manuals:

- Networks → • Far Q MELSECNET/H Network System Reference Manual (PLC to PLC network)

- 3) As KN1 and KN2, use the values in the following table.

When the intelligent module is installed on the main base unit

CPU Type	KN1 ($\times 10^{-3}$ ms)	KN2 ($\times 10^{-3}$ ms)
Q00JCPU	111	55
Q00CPU	91	46
Q01CPU	85	41

When the intelligent module is installed on the expansion base unit

CPU Type	KN1 ($\times 10^{-3}$ ms)	KN2 ($\times 10^{-3}$ ms)
Q00JCPU	113	56
Q00CPU	92	48
Q01CPU	86	43

(Example)

When the number of automatic refresh points is 4 for the analog-digital converter module (Q64AD) (when installed on the main base unit of Q01CPU)

$$0.249 \text{ (ms)} = 0.085 + 0.041 \times 4$$

(5) Execution times of various functions processed at END

(a) Calendar update processing time

- 1) Time to write the clock data stored in SD210 to SD213 to the clock element at END processing when a clock data set request is given (SM210 turns from OFF to ON).
- 2) Time to read the clock data to SD210 to SD213 at END processing when a clock data read request is given (SM213 turns ON).

CPU Type	END Processing Time (ms)	
	At clock data set request	At clock data read request
Q00JCPU	0.12	0.04
Q00CPU	0.11	0.03
Q01CPU	0.10	0.02

(b) Error cancel processing

Time to cancel the continuation error stored in SD50 on the leading edge of SM50 (error cancel) (when it turns from OFF to ON).

CPU Type	Common Processing Time (ms)	
	Annunciator	Other error
Q00JCPU	0.17	0.10
Q00CPU	0.14	0.09
Q01CPU	0.13	0.08

Special Relay List

(1) Diagnostic Information

Number	Name	Meaning	Explanation	Set by (When Set)
SM0	Diagnostic errors	OFF: No error ON : Error	<ul style="list-style-type: none"> • ON if diagnosis results show error occurrence (Includes external diagnosis) • Stays ON subsequently even if normal operations restored 	S (Error)
SM1	Self-diagnostic error	OFF: No self-diagnosis errors ON : Self-diagnosis	<ul style="list-style-type: none"> • Comes ON when an error occurs as a result of self-diagnosis. • Stays ON subsequently even if normal operations restored 	S (Error)
SM5	Error common information	OFF: No error common information ON : Error common information	<ul style="list-style-type: none"> • When SM0 is ON, ON if there is error common information 	S (Error)
SM16	Error individual information	OFF: No error common information ON : Error common information	<ul style="list-style-type: none"> • When SM0 is ON, ON if there is error individual information 	S (Error)
SM50	Error reset	OFF → ON : Error reset	<ul style="list-style-type: none"> • Conducts error reset operation 	U
SM51	Battery low latch	OFF: Normal ON : Battery low	<ul style="list-style-type: none"> • ON if battery voltage at CPU drops below rated value. • Stays ON subsequently even after normal operation is restored 	S (Error)
SM52	Battery low	OFF: Normal ON : Battery low	<ul style="list-style-type: none"> • Same as SM51, but goes OFF subsequently when battery voltage returns to normal. 	S (Error)
SM53	AC/DC DOWN detection	OFF: AC/DC DOWN not detected ON : AC/DC DOWN detected	<ul style="list-style-type: none"> • 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. • 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. 	S (Error)
SM56	Operation Errors	OFF: Normal ON : Operation error	<ul style="list-style-type: none"> • ON when operation error is generated • Stays ON subsequently even if normal operations restored 	S (Error)
SM60	Blown fuse detection	OFF: Normal ON : Module with blown fuse	<ul style="list-style-type: none"> • Comes ON even if there is only one output module with a blown fuse, and remains ON even after return to normal • Blown fuse state is checked even for remote I/O station output modules. 	S (Error)
SM61	I/O module verification error	OFF: Normal ON : Error	<ul style="list-style-type: none"> • Comes ON if there is a discrepancy between the actual I/O modules and the registered information when the power is turned on 	S (Error)
SM62	Annunciator detection	OFF: Not detected ON : Detected	<ul style="list-style-type: none"> • Goes ON if even one annunciator F goes ON. 	S (Instruction execution)
SM100	Serial communication function using flag	OFF: Serial communication function is not used. ON : Serial communication function is used.	<ul style="list-style-type: none"> • Stores whether the serial communication function in the serial communication setting parameter is used or not. 	S (Power-on or reset)
SM101	Communication protocol status flag	OFF: GX Developer ON : MC protocol communication device	<ul style="list-style-type: none"> • Stores whether the device that is communicating via the RS-232 interface is GX Developer or MC protocol communication device. 	S (RS232 communication)
SM110	Protocol error	OFF: Normal ON : Abnormal	<ul style="list-style-type: none"> • Turns ON when an abnormal protocol was used to make communication in the serial communication function. • Remains ON if the protocol is restored to normal thereafter. 	S (Error)
SM111	Communication status	OFF: Normal ON : Abnormal	<ul style="list-style-type: none"> • Turns ON when the mode used to make communication was different from the setting in the serial communication function. • Remains ON if the mode is restored to normal thereafter. 	S (Error)
SM112	Error information clear	ON : Cleared	<ul style="list-style-type: none"> • Turns ON when the error codes stored in SM110, SM111, SD110 and SD111 are cleared. (Activated when turned from OFF to ON) 	U
SM113	Overrun error	OFF: Normal ON : Abnormal	<ul style="list-style-type: none"> • Turns ON when an overrun error occurred in the serial communication error. 	S (Error)
SM114	Parity error	OFF: Normal ON : Abnormal	<ul style="list-style-type: none"> • Turns ON when a parity error occurred in the serial communication error. 	S (Error)
SM115	Framing error	OFF: Normal ON : Abnormal	<ul style="list-style-type: none"> • Turns ON when a framing error occurred in the serial communication error. 	S (Error)