MITSUBISHI

Changes for the Better

Mitsubishi Programmable
Logic Controller

Training Manual

Q-series basic course(for GX Developer)

MELSEC

• SAFETY PRECAUTIONS •

(Always read these instructions before using the exercise.)

When designing the system, always read the relevant manuals and give sufficient consideration to safety.
During the exercise, pay full attention to the following points and handle the product correctly.

[EXERCISE PRECAUTIONS]

WARNING

- Do not touch the terminals while the power is on to prevent electric shock.
- When opening the safety cover, turn off the power or conduct a sufficient check of safety before operation.

Caution

- Follow the instructor's direction during the exercise.
- Do not remove the module of the demonstration machine or change wirings without permission.
  Doing so may cause failures, malfunctions, personal injuries and/or a fire.
- Turn off the power before installing or removing the module.
  Failure to do so may result in malfunctions of the module or electric shock.
- When the demonstration machine (X/Y table, etc.) emits abnormal odor/sound, press "Power switch" or "Emergency switch" to turn off.
- When a problem occurs, notify the instructor as soon as possible.

REVISIONS

* The textbook number is given on the bottom left of this textbook.

Print Date	* Textbook number	Revision
Jan., 2006	SH-080617ENG-A	First edition

CONTENTS

INTRODUCTION

This textbook describes the PLC, the program editing methods, the sequence instructions and the application instructions to help you understand the MELSEC-Q series programming.
The multiple CPU system is available for the MELSEC-Q series with multiple CPU modules, but this textbook explains the case in which one CPU module is used.
Refer to the following school textbook for the exercises when the multiple CPU system is used.

: Q Programming School Textbook (Practice) ....SH-080045-D or later

The related manuals are shown below.

(1) QCPU User's Manual
(Hardware Design, Maintenance and Inspection)........................SH(NA)-080483ENG
Explains the hardware.

(2) QCPU User's Manual
(Function Explanation,Program Fundamentals) ....SH(NA)-080484ENG
Explains the functions and programming method.

(3) QCPU(Q Mode)/QnACPU Programming Manual
(Common Instructions).. SH(NA)-080039
Explains details of each instruction.

(4) GX Developer Version 8
Operating Manual (Startup)...SH(NA)-080372E

(5) Ladder Logic Test Function software for Windows
SW5D5C-LLT-E Operation Manual.. SH(NA)-080064

(6) GX Developer Version 8
Operating Manual ....SH(NA)-080373E
Explains the operating method.

In this textbook, the following CPUs are generically named "QCPU (Q mode)".

- Q02CPU
- Q02HCPU
- Q06HCPU
- Q12HCPU
- Q25HCPU

CHAPTER 1 BASICS OF PLC

1.1 Program

Taking PLC as a control ladder, PLC can be described by an input ladder, output ladder, and internal sequential operation.

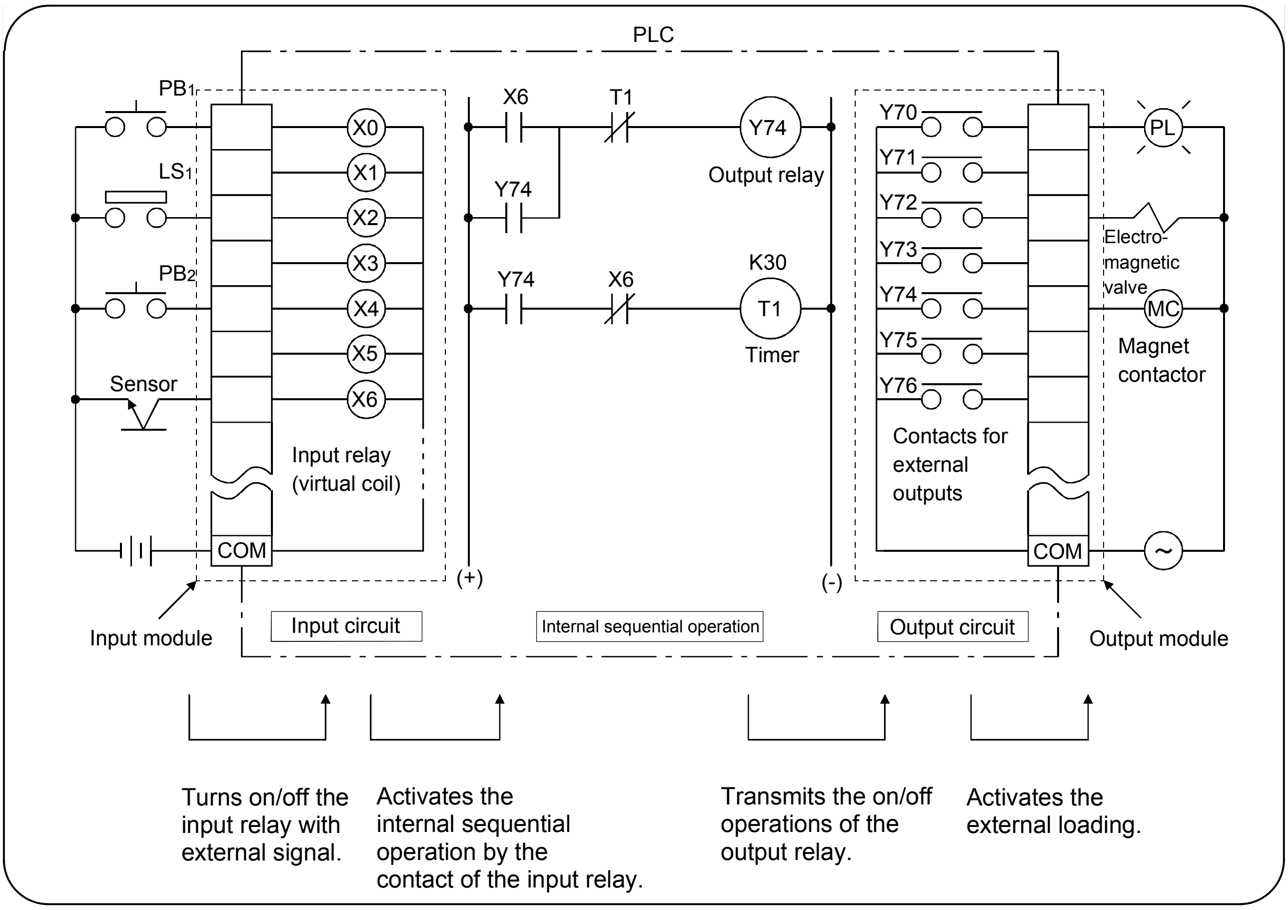


Figure 1.1 PLC Configuration

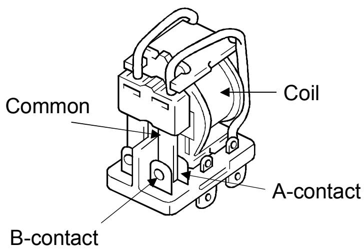
The PLC is an assembly of relays and timers/counters as well as an electronic device centered around a microprocessor.

As in figure 1.1, internal sequential operation is performed by turning on/off the coil by connecting the normally open and normally close serially or in parallel.

"Relay", which is also called an electromagnetic relay, is a switch that passes on signals. Also, it is a key component that makes up a logic ladder.

1) Energizing the coil ⇨ Excitation
   - Closes the normally open (Conducted)
   - Opens the normally close (Not conducted)

2) De-energizing the coil ⇨ Demagnetization
   - Opens the normally open (Not conducted)
   - Closes the normally close (Conducted)



	Coil off (all times)	Coil on (in operation)
Normally open ⊣⊢	Not conducted	Conducted
Normally close ⊣/⊢	Conducted	Not condcuted

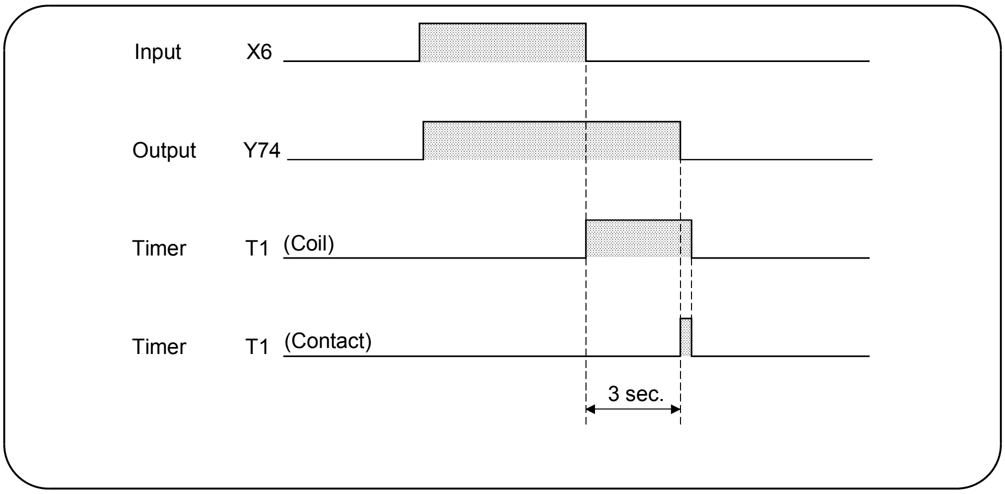
Internal Sequential Operation

The signal flow of figure 1.1 internal sequential operation is described below.

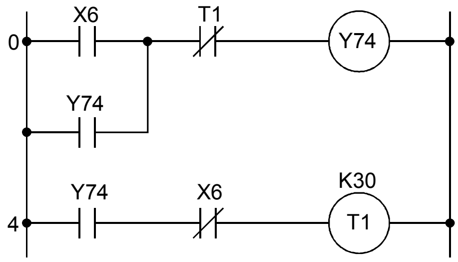
1) When the sensor turns on, the coil of input relay X6 is energized.
2) Energizing the coil of input relay X6 conducts the normally open X6 and energizes the coil of output relay Y74.
   (As the timer is not energized at this moment, the normally close remains conducted.)
3) Once the coil of output relay Y74 is energized, the external connection Y74 is conducted allowing the magnet contactor (MC) to be turned on.
4) Turning off the sensor demagnetizes the coil of input relay X6 and ceases the conduction of normally open X6. As self-maintaining the normally open Y74 is conducted, the coil remains energized. (Self-maintaining operation)
5) When the coil of output relay Y74 is energized (with the normally open Y74 conducted), turning off the sensor (with normally close X6 conducted) energizes the coil of the timer T1 so that the timer starts measuring the time.
   After a lapse of three seconds (K30 indicates 3.0 seconds), the normally open of the timer becomes conducted and the normally close is brought non conductive.
6) As a result, the coil of output relay Y74 demagnetizes and the load magnet contactor drops.
   The output relay self- maintenance is also released.

Time Chart

A time chart that explains the input/output relays and timer operations is shown below.



The internal sequential operation can be described as a program of PLC. The program is saved in the program memory as close to the instruction list described below.



(a) Ladder diagram

Step number	Instruction words	I/O number
0	LD	X6
1	OR	Y74
2	ANI	T1
3	OUT	Y74
4	LD	Y74
5	ANI	X6
6	OUT	T1 K30
7	END	

Repeat operation

(b) Instruction list (program list)

Figure 1.2 Program

- The program consists of a large number of instruction words and I/O numbers.
  (I/O numbers include not only the input (X) and output (Y), but all the contacts and the factor numbers of the coils such as a timer (T) that make up a ladder diagram. Those numbers are also called device numbers.)

- The instructions, which contain instruction words and I/O numbers, are added numbers that represent the order of operations.
  Those numbers are called step numbers.
  (Instruction words are sometimes referred to as instructions.)

- The number of steps varies depending on the types of instructions or the setting method for the numbers to be used for the I/O numbers and operations. (The more complicated the operations become, the more numbers are added to the instructions.)

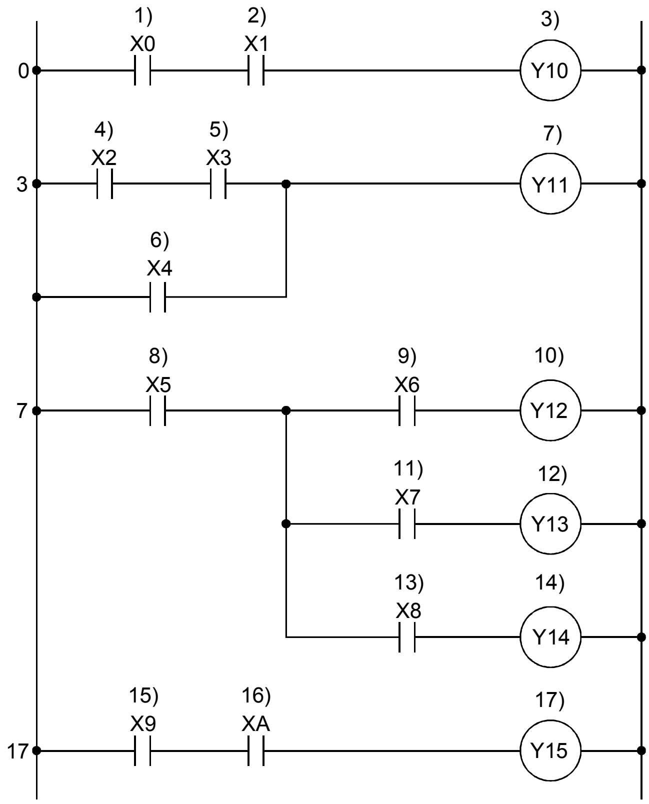
- The instructions repeat from "Step number = 0" to "Instruction words end" endlessly.
  This is called "repeat operation", "cyclic operation" or "scanning".)
  Amount of time required to take once through is called operation cycle (scan time).

- The number of the programming steps from step number = 0 to the END instruction defines the length or the size of the program.

- The program is stored in the program memory within the CPU. The operation is performed in one ladder block unit.
  One ladder block ranges from the operation START instruction (LD, LDI) to the OUT instruction (including the DATA instruction).

1.2 Program Processing Procedure

The operation process is performed serially from the start step of the program memory left to right then top to bottom (in the order corresponding to 1), 2)... 17)) in ladder block unit as shown below.



1.3 MELSEC-Q Module Configuration

(1) High Performance model and Basic model
Q series CPU has two models; High Performance model and Basic model.
As a High Performance model CPU (Q mode) is used as the demonstration in this course, "QCPU" means "High Performance model CPU (Q mode)" in this textbook unless otherwise specified.

* See "Q series data book (L08029E)" for details.

Model CPU	Description	
High Performance model	Target	For the use of focusing on high-speed processing and system extension
	CPU name	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
	Features	• High-speed processing • Large-capacity memory...... the number of programs: 252(Q25H) file register: 128 K(Q12H/Q25H built-in) • Number of I/O device points…8,192 (including devices for remote I/O) • Number of I/O points .... 4,096 (excluding the I/Os for remote I/O) • The SFC program is available. • The Multiple CPU system is configurable. • The Motion CPU is mountable. • The personal computer CPU (manufactured by CONTEC CO., LTD) is applicable. • The real number, trigonometric function, character-string, and PID control operation are available. • The memory cards are compatible • The ROM operation is available by using the CPU built-in standard ROM.
Basic model	Target	For the use for a small-scale system
	CPU name	Q00JCPU, Q00CPU, Q01CPU
	Features	• Number of I/O device points…..2,048 (including the devices for remote I/O) • Number of I/O points (excluding the I/Os for remote I/O)..... 256 (Q00J) 1,024 (Q00/Q01) • All-in-one power supply base is available (Q00J). • Built-in serial communication ready for service (Q00/Q01) • The ROM operation is available by using the CPU built-in standard ROM.

(2) Q mode CPU and A mode CPU
QCPU series has two models; the one is Q mode CPU models that utilize the Q-series original functions and performance. The other is A mode CPU models that especially improve CPU processing ability by using the program hardware of the conventional AnS series.
This textbook deals with the Q mode models.

(3) Basic configuration of the PLC system

The actual configuration of the PCL is explained below using the building-block type.

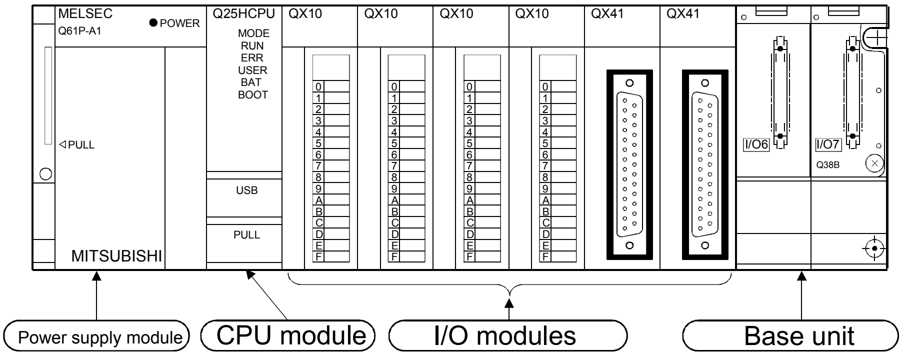
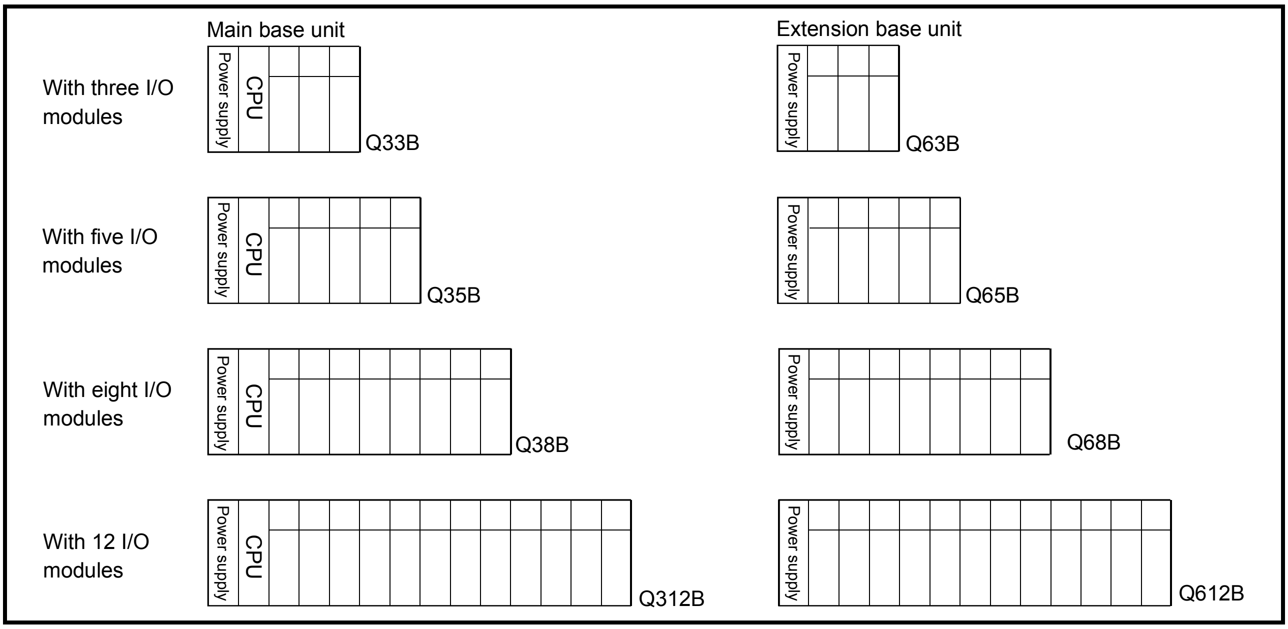


Figure 1.3 MELSEC-Q Module Configuration

Base Unit



• The main roles of the base unit are; to attach the power supply module, CPU module, and I/O modules securely, to supply 5VDC power from the power supply module to the CPU module and I/O modules, and to transmit the control signals to each module.

• Q00JCPU, which is all-in-one power supply base, is available for the basic model CPU.

Power Supply Module

Input	Output	Module model
100V-120V AC	5V DC 6A	Q61P-A1
200-240V AC	5V DC 6A	Q61P-A2
100V-120V AC	5V DC 3A, 24V DC 0.6A	Q62P
24V DC	5V DC 6A	Q63P

CPU Module

Maximum number of I/O connected to the PLC	Program Capacity (Maximum)	Basic instruction processing speed	CPU type
4096 points	28 K steps	79ns	Q02CPU
	28 K steps	34ns	Q02HCPU
	60 K steps		Q06HCPU
	124 K steps		Q12HCPU
	256 K steps		Q25HCPU
256 points	8 K steps	200ns	Q00JCPU
1024 points	14 K steps	160ns	Q00CPU
	14 K steps	100ns	Q01CPU

I/O Module

Format \ No. of I/O		8 points	16 points	32 points	64 points
Input module	120V AC	—	○	—	—
	240V AC	○	—	—	—
	24V DC (Plus common)	—	○	○	○
	24V DC (High-speed input)	○	—	—	—
	24V DC (Minus Common)	—	○	○	—
	5/12V DC	—	○	○	○
Output module	Connection output	—	○	—	—
	Independent contact output	○	—	—	—
	Triac output	—	○	—	—
	Transistor output (Sink)	○	○	○	○
	Transistor output (Source)	—	○	○	—
I/O mixed		○	—	○	—

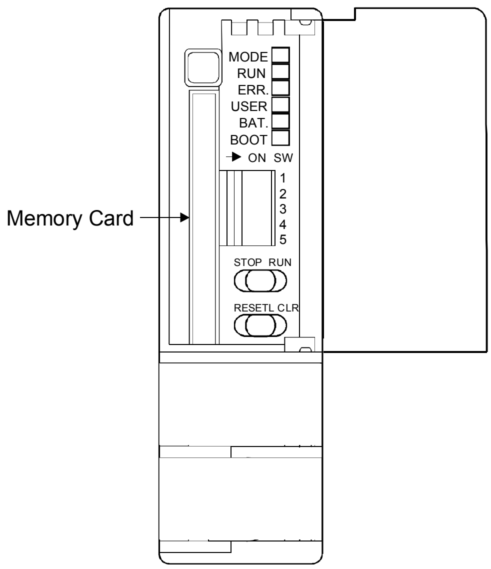
Memory Card

QCPU equips a built-in memory as a standard to store the parameters and the program so that a program can normally execute without a memory card.

High Performance model QCPU is compatible with the memory cards.
However, the Basic model QCPU does not support the memory cards.

The memory cards are required when using the following functions;

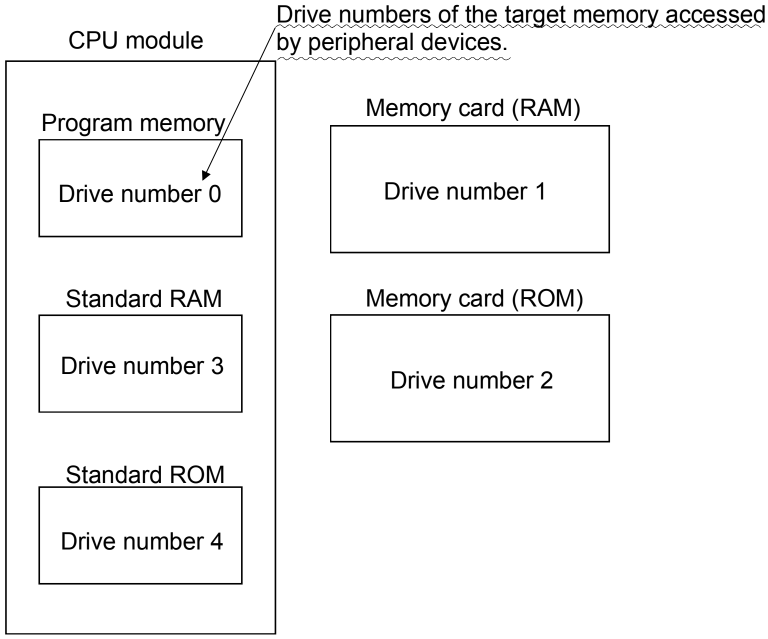
Type	Description
SRAM card	Can be written/changed within the amount of the memory. <Example of the usage> • During boot operation • When using the file register that exceeds 32 K points/128 K points in volume. • For storing the sampling trace data • For storing the SFC trace data • For storing the failure history data
Flash card	Writes the contents of the program memory or the specified file in one pass. The newly written data replaces all the original data. Can only be read by using the sequence program. <Example of the usage> • During boot operation • When no change will be made to the data
ATA card	Can be written/changed within the amount of the program. Using the file access order (such as FWRITE instruction) in the sequence program, accesses the PLC user data of the ATA card in the CSV type/binary type. <Example of the usage> • During boot operation • When used by PLC user data (general-purpose data)



- The memory cards are required when the amount of the built-in program memory, standard RAM, and standard ROM are not enough for storing the data.
- The memory cards should be selected according to the size of the program or the type of the data to be stored.
- The SRAM-type RAM card must install the supplied batteries upon purchase. The SRAM card data cannot be duplicated unless those batteries are installed.
- Format the memory card before using it first.
- The Flash card is writable 100,000 times. The ATA card is writable 1,000,000 times.

<Reference: QCPU System Configuration>

The memory of QCPU consists of the following block configurations.



- Program memory :Stores the program that is used by QCPU for operation. The programs stored in the standard ROM or in the memory card are read into the program memory before execution.
- Standard RAM :Stores the data of the file register and the local device.
- Standard ROM :Stores the data of the parameters and the program when the QCPU is operated with ROM.
- Memory card (RAM) :Stores the local device, debug data, SFC trace data, and failure history data as well as the parameters and program.
- Memory card (ROM) :The Flash card stores the parameters, program, and the file register.
  :The ATA card stores the parameters, program, and the PLC user data (general-purpose file).

1.4 External I/O Signals and I/O Numbers

(1) Wiring of I/O devices
The signals from external input devices are substituted by the input numbers, which is determined by the fixing points and port numbers of the connected input module, and are dealt with in the program.
The operation result output (coil) uses the output numbers that are determined by the fixing points and port numbers of the output module that is connected to the external output devices.

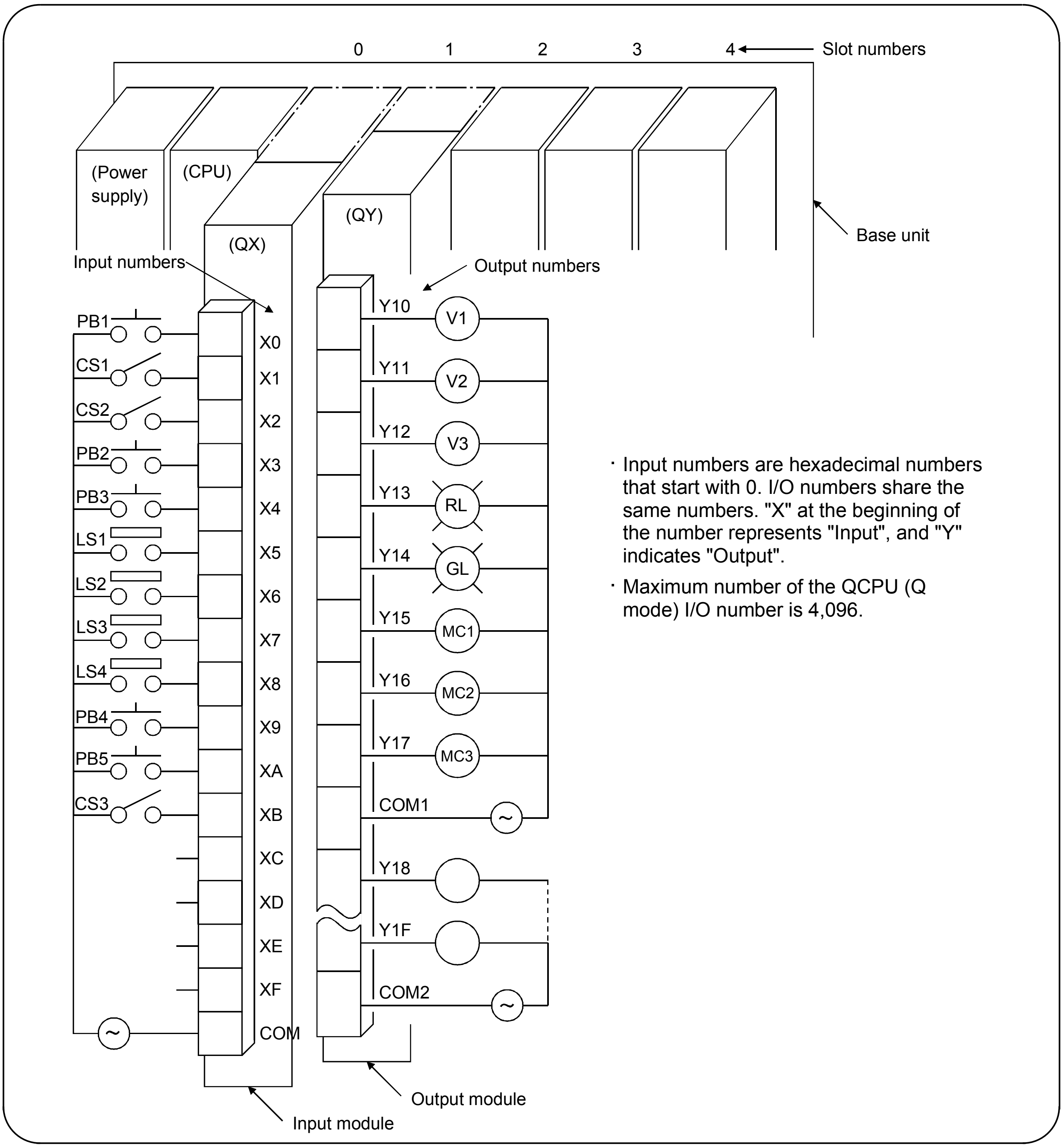
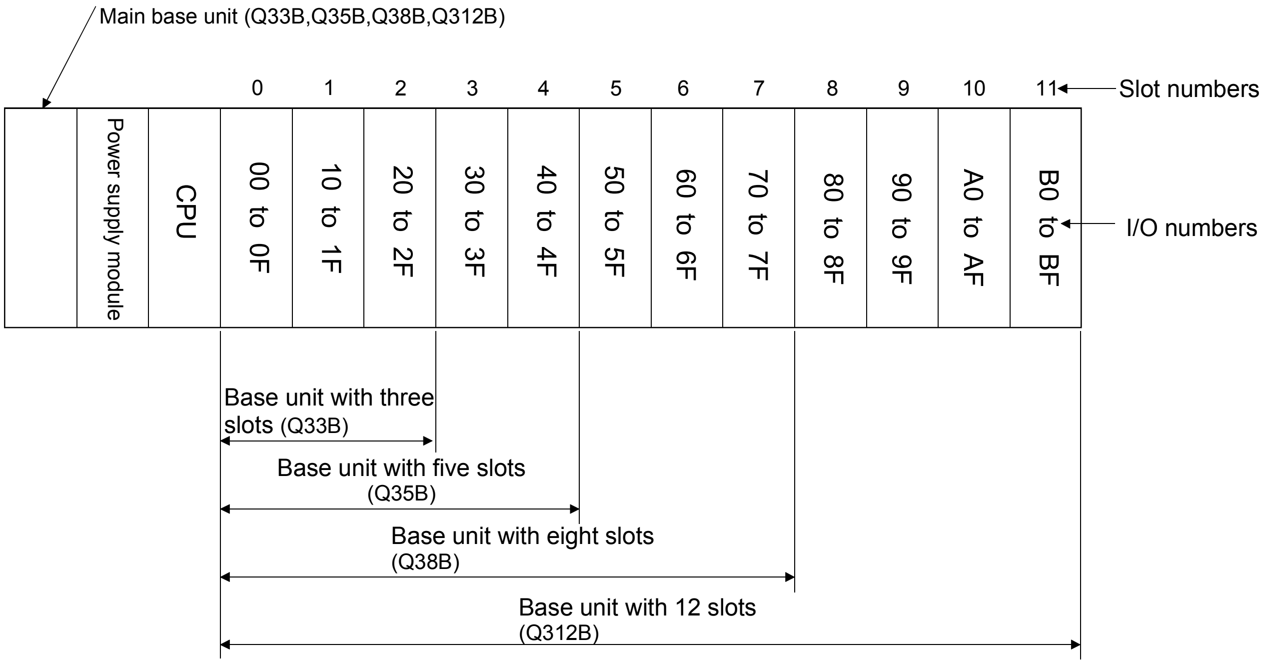


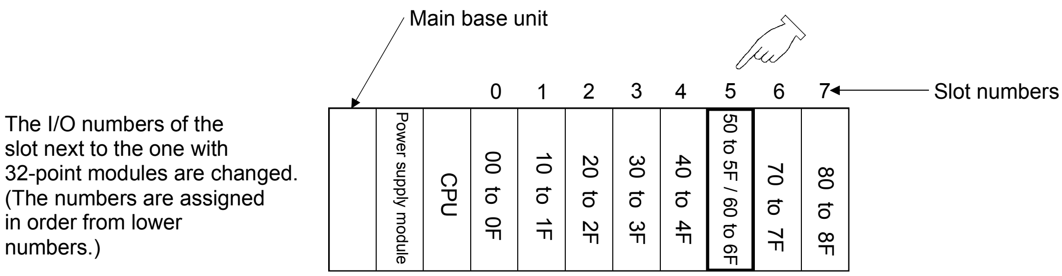
Figure 1.4 Wiring of I/O devices

(2) I/O numbers of the main base unit

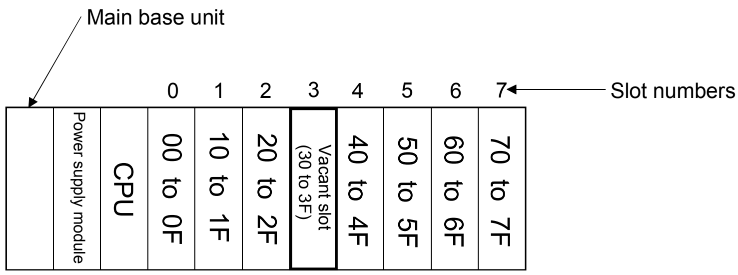
The I/O numbers of the I/O modules that are attached to the main base unit are assigned as follows. This concept applies both to the I/O modules and to the intelligent function module.



• The I/O numbers of one slot (one module) is assigned in ascending order in units of 16 points. (0 to F_H)
  The status that 16-point module is attached to each slot is considered as a standard.
  For example, the I/O numbers when 32-point module is attached to the fifth slot is as shown below.



• The vacant slot (The slot with no I/O modules installed) is also assigned with the I/O numbers.
  For example, if the third slot is vacant, the I/O numbers are assigned as described below. (Default)
  The number of assigned point is changeable depending on the setting.



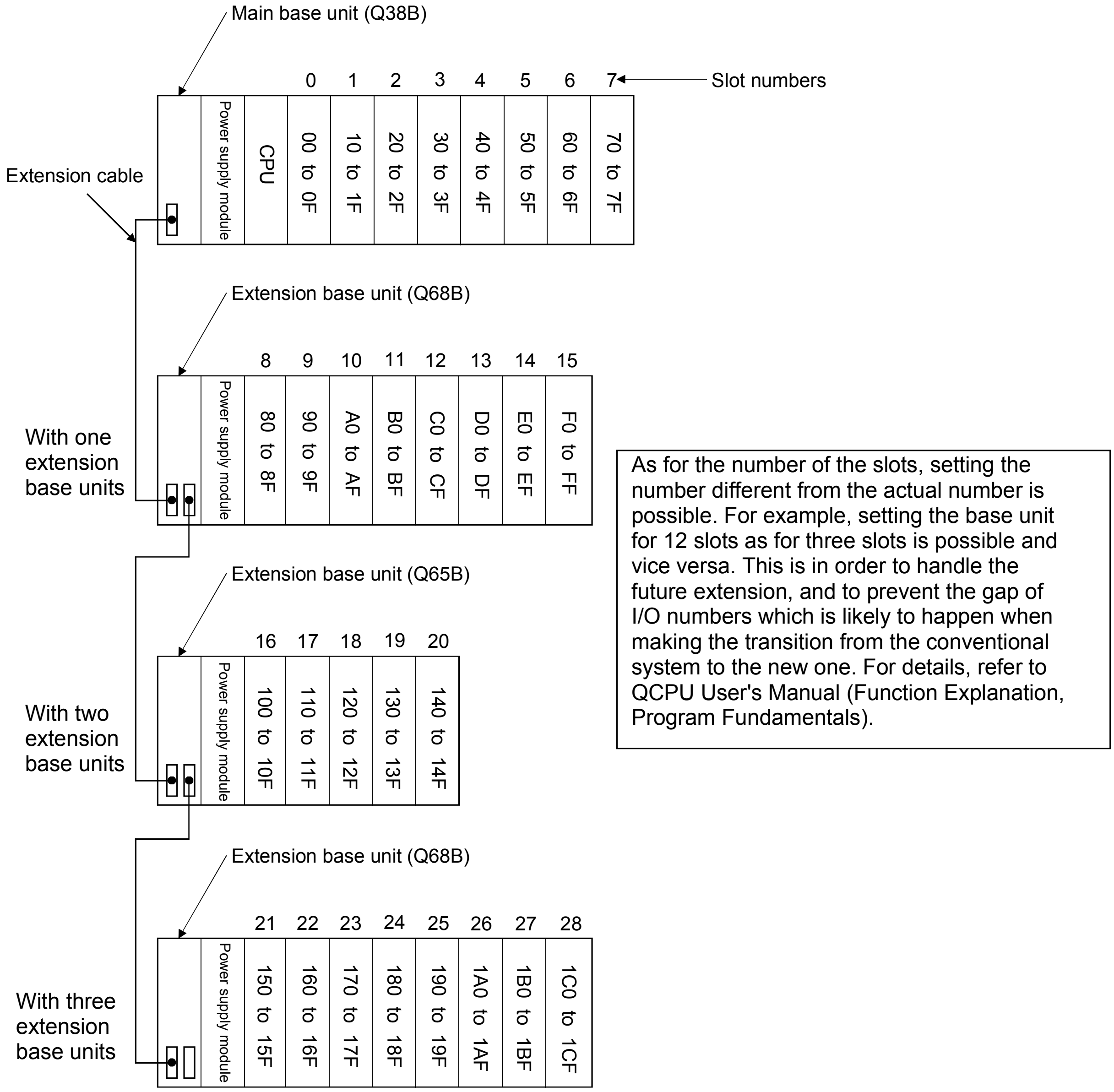
• As for the multiple CPU configuration (Two to four CPUs), the I/O numbers are assigned from the slot next to the one to which a CPU is attached.

(3) I/O numbers of the extension base unit

Connect the extension base unit when the required number of the slots exceeds the number of the slots of the main base unit.

The I/O numbers are assigned as follows by default.

This concept applies both to the I/O modules and to the intelligent function module.

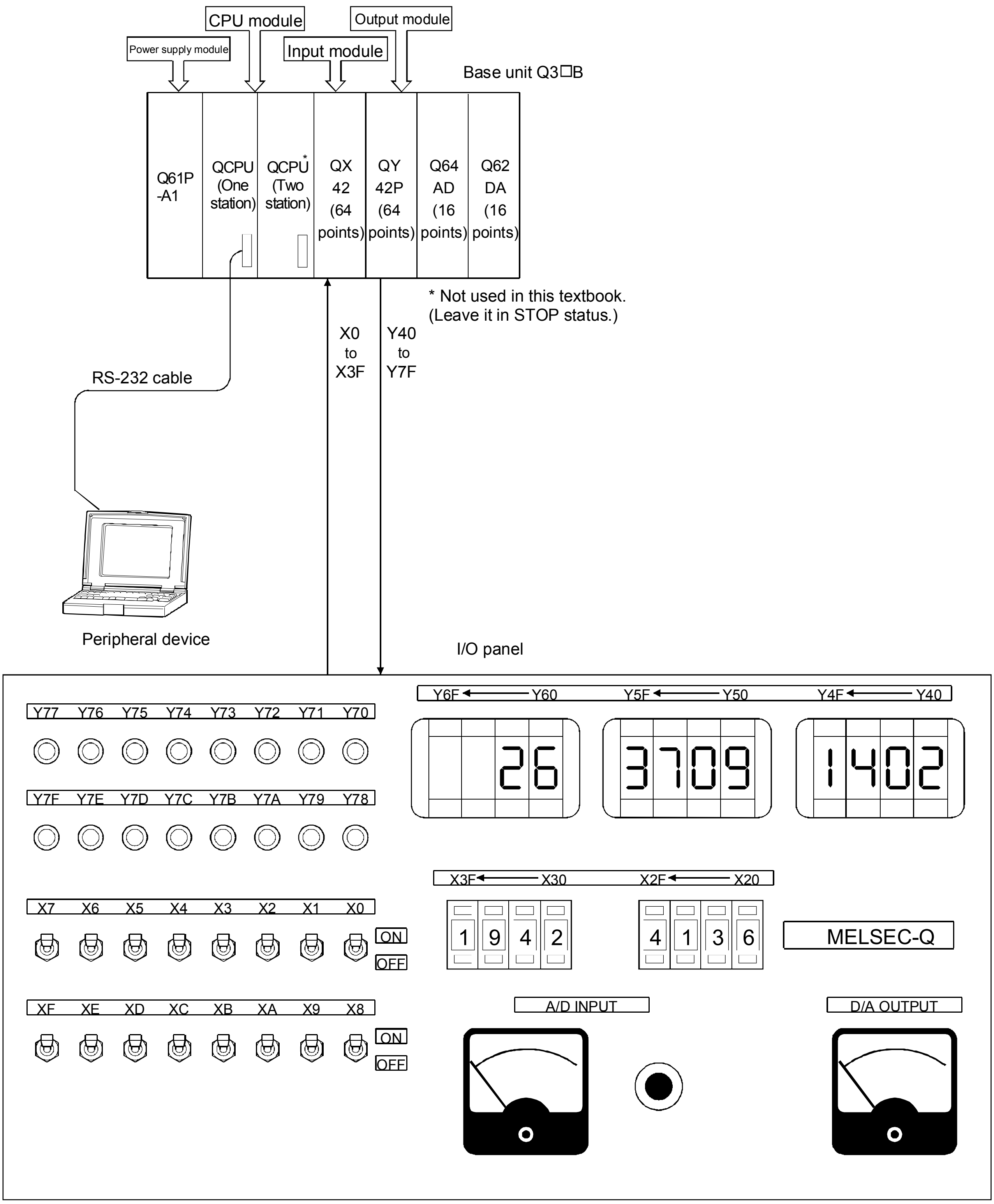


- The slots of the extension base unit are also assigned in ascending order in units of 16 points.
- The head I/O numbers of the extension bases follow the last number of the main base or the one of the previous extension base.
- It is possible to assign "0" as the I/O number to the vacant slot or to the area with no slot.

The number of the extension base units that can be extended is as follows;

CPU type		Number of stages (Including the ones connected via GOT bus)
High Performance model		7
Basic model	Q00JCPU	2
	Q00CPU,Q01CPU	4

1.5 System Configuration and I/O Numbers of Demonstration Machine

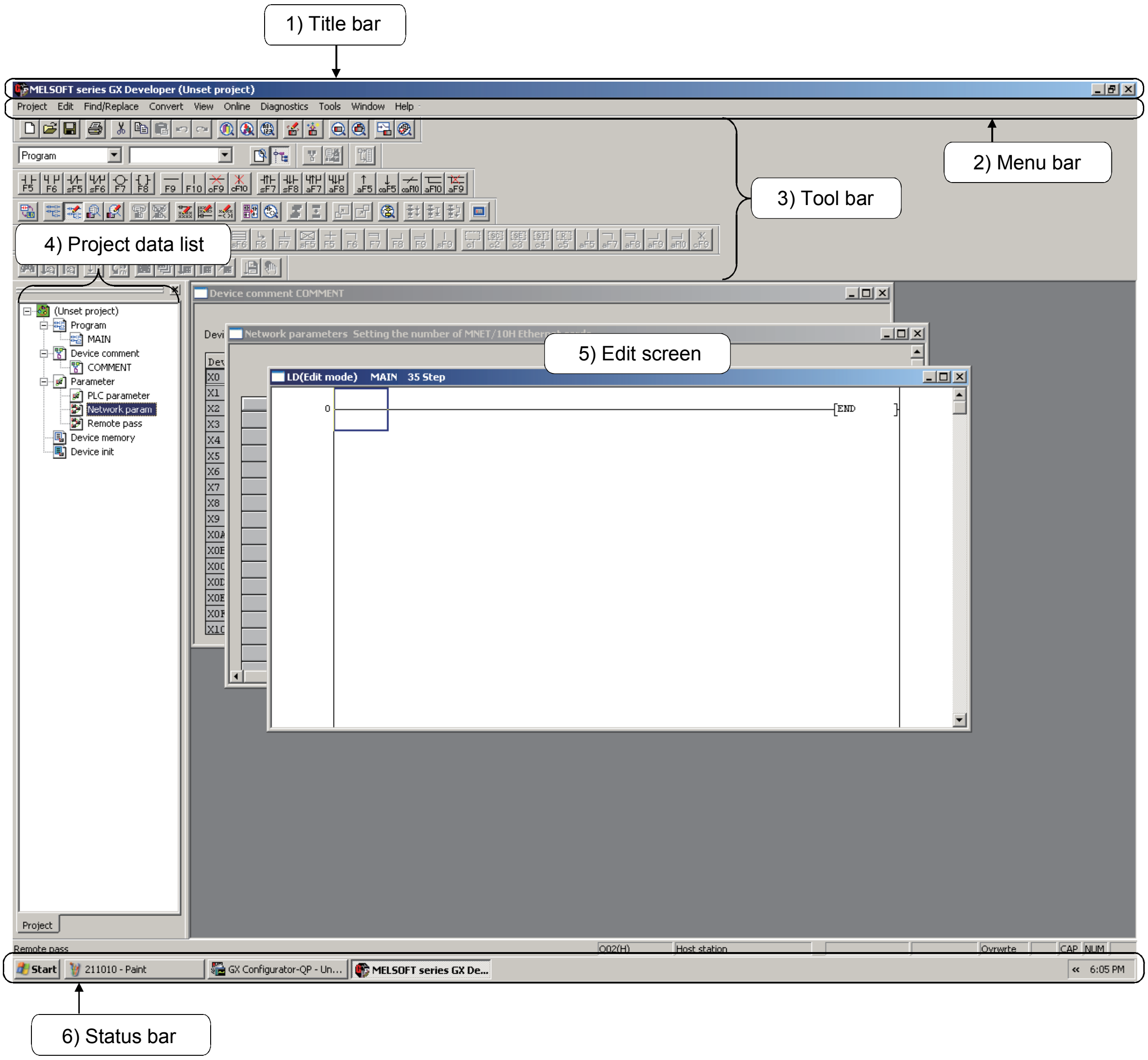


MEMO

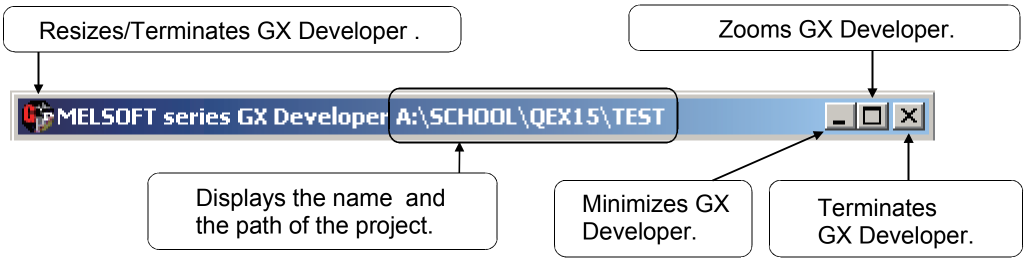
CHAPTER 2 OPERATING GX DEVELOPER

2.1 Basic Knowledge Required for Operating GX Developer

2.1.1 GX Developer Screen

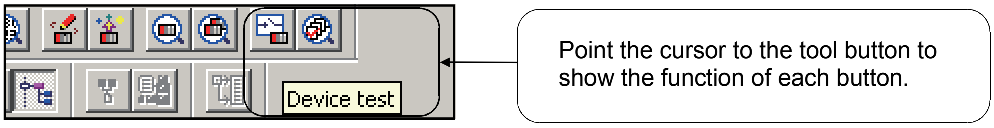


1) Title bar
Shows the name of the active project.



2) Menu bar
Most frequently used item when operating GX Developer.
Click the menu bar to select a variety of functions from the drop-down menu under the menu bar.

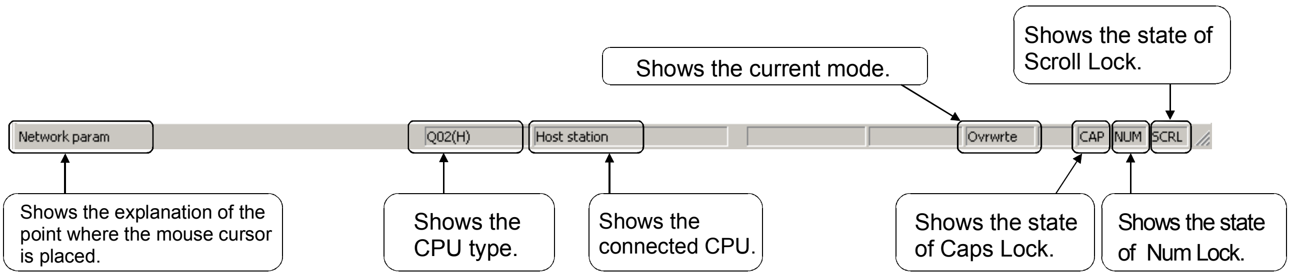
3) Tool bar
Equips buttons to easily access the commonly-used functions. This enables quicker and speedier operation.



4) Project data list
A ladder program creation screen or dialogue boxes can be read out from the list.
This list shows the project data sorted in classified categories.

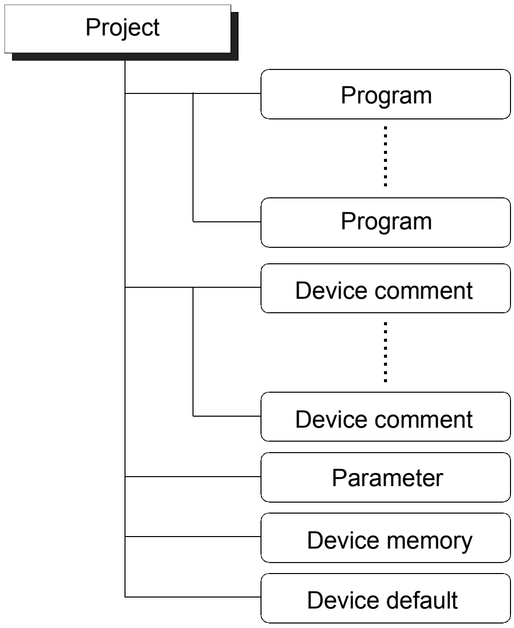
5) Edit screen
Displays a ladder program creation screen or a comment creation screen. Various screens are displayed so that you can edit ladder diagrams, comments, or parameters.

6) Status bar
Shows the status information of the GX Developer.



2.1.2 Project

A project consists of programs, device comments, parameters, device memory, and device default.



Item	Description
Program	A sequential program that is required to operate the PLC CPU.
Device comment	A comment for the sequential program device, which is classified into two types. One is "common comment" that is common to all the projects, and the other is "comment by program", which varies by projects.
Parameter	Used for specifying the range of various settings such as the network related setting and device spectrum.
Device memory	Displays the current device capacity. The device capacity can be changed by entering numeric value.
Device default	Sets the device default for the PLC CPU.

1) One project per GX Developer
One GX Developer can edit only one project unit.
To edit two or more projects at a time, run as many GX Developers as the number of projects.

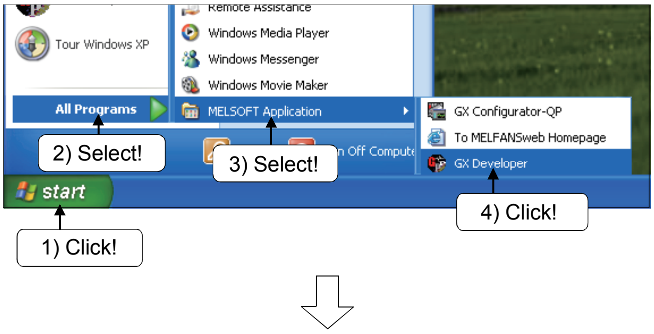
2) Device Comments
The device comments of GX Developer are categorized into common comment and comments by program.

Comment type	Number of comments	Description
Common comment	1	A device comment that is common to all the programs within the project.
Comments by program	Equals the number of the programs	A device comment that is set by each program. The name of the comment must be the same as of the program.

If the contents of the two device comments overlap with one another, select [Tools] → [Option] → [By program] to set priorities of them.

2.2 Operation Before Creating Ladder Programs

2.2.1 Starting up the GX Developer



1) Click the $\boxed{\text{Start}}$ button.

2) Select the [All Programs] menu.

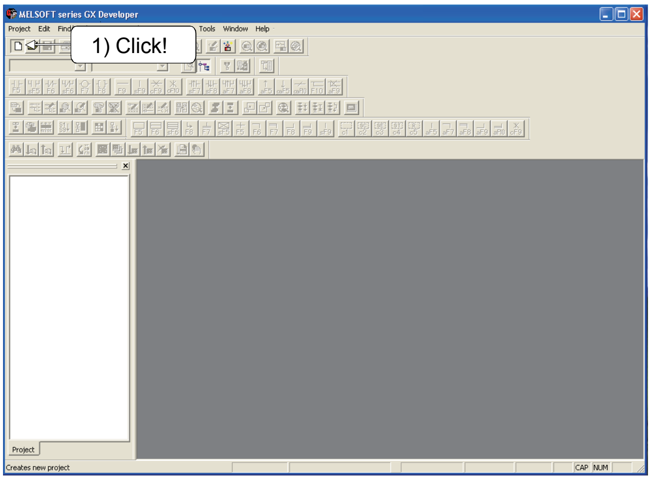
3) Select the [MELSOFT Application] menu.

Put a mouse cursor over the items to select the menu.
(Clicking or double-clicking the mouse is not required.)

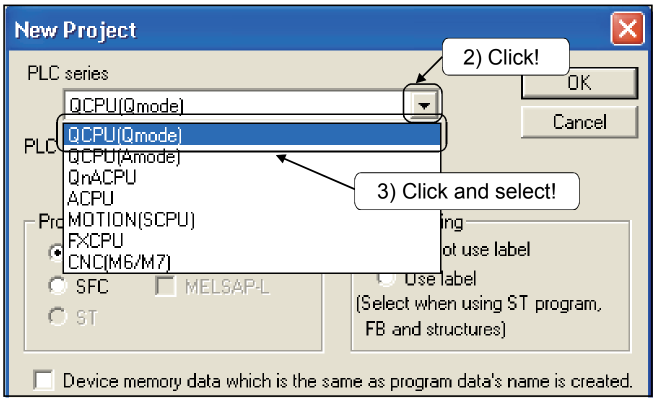
4) Click the [GX Developer] menu.

5) GX Developer will start up.

2.2.2 Creating a new project

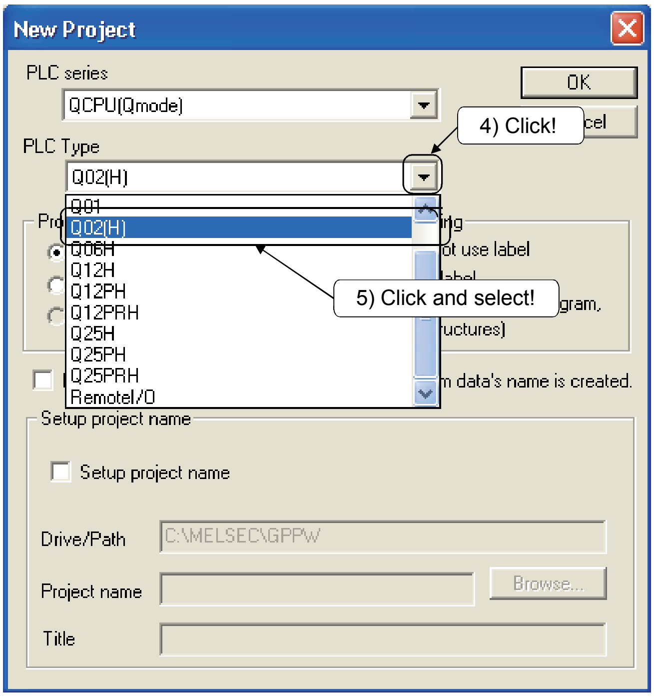


1) Click on the tool bar or select [Project] → [New Project] menu (Ctrl + N).



2) Click the button with an inverted delta symbol on the [PLC series] column.

3) Click on "QCPU(Qmode)" to select it from the drop-down menu.

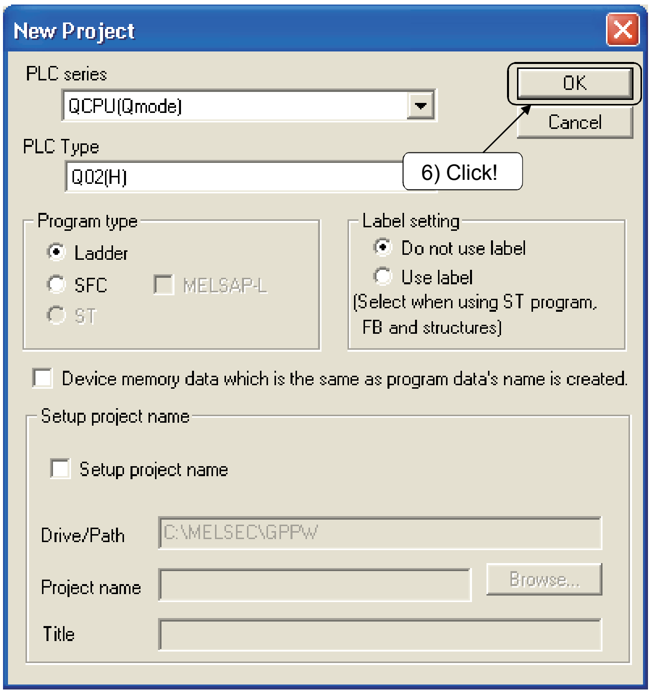


4) Click the button with an inverted delta symbol on the [PLC Type] column.

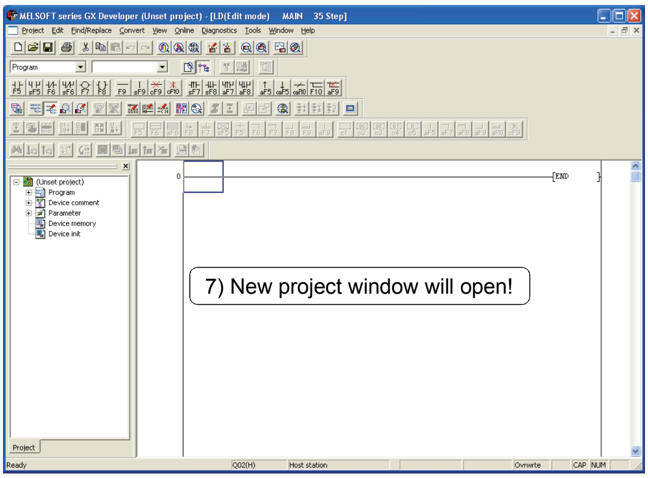
5) Click on "Q02(H)" to select it from the drop-down menu.

(Continue on the next page)

(Continued from the previous page)

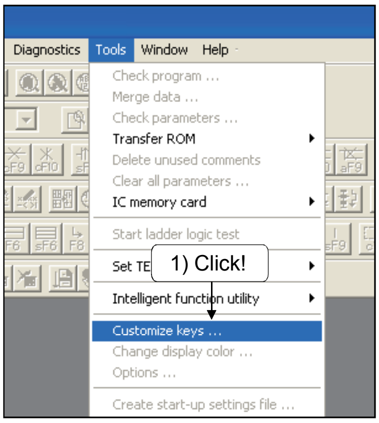


6) Click the OK button.

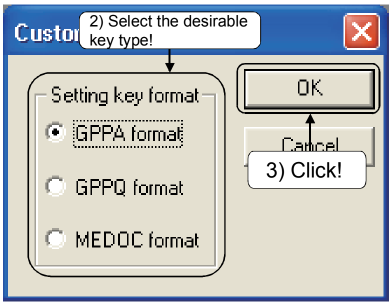


7) New project window will open.

2.2.3 Changing the assignment of the function keys



1) Click [Tooles] → [Customize keys] menu.

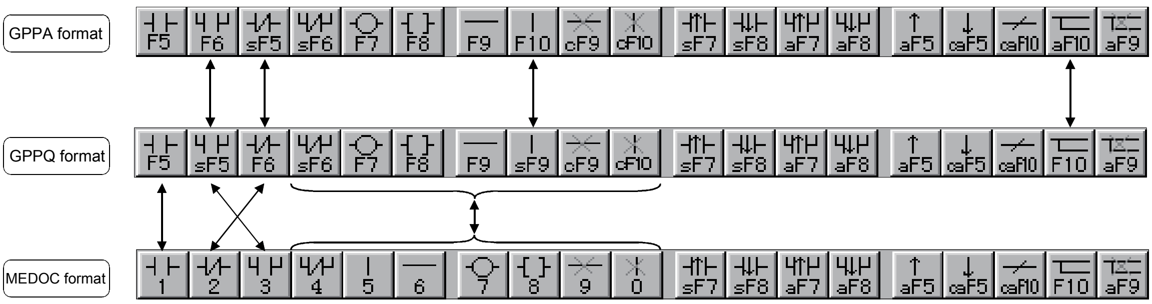


2) Click on the desirable key type to select it.
As an example, the window on the left shows the case when "GPPA type" is selected.

3) Click the $\boxed{\text{OK}}$ button.

REFERENCE

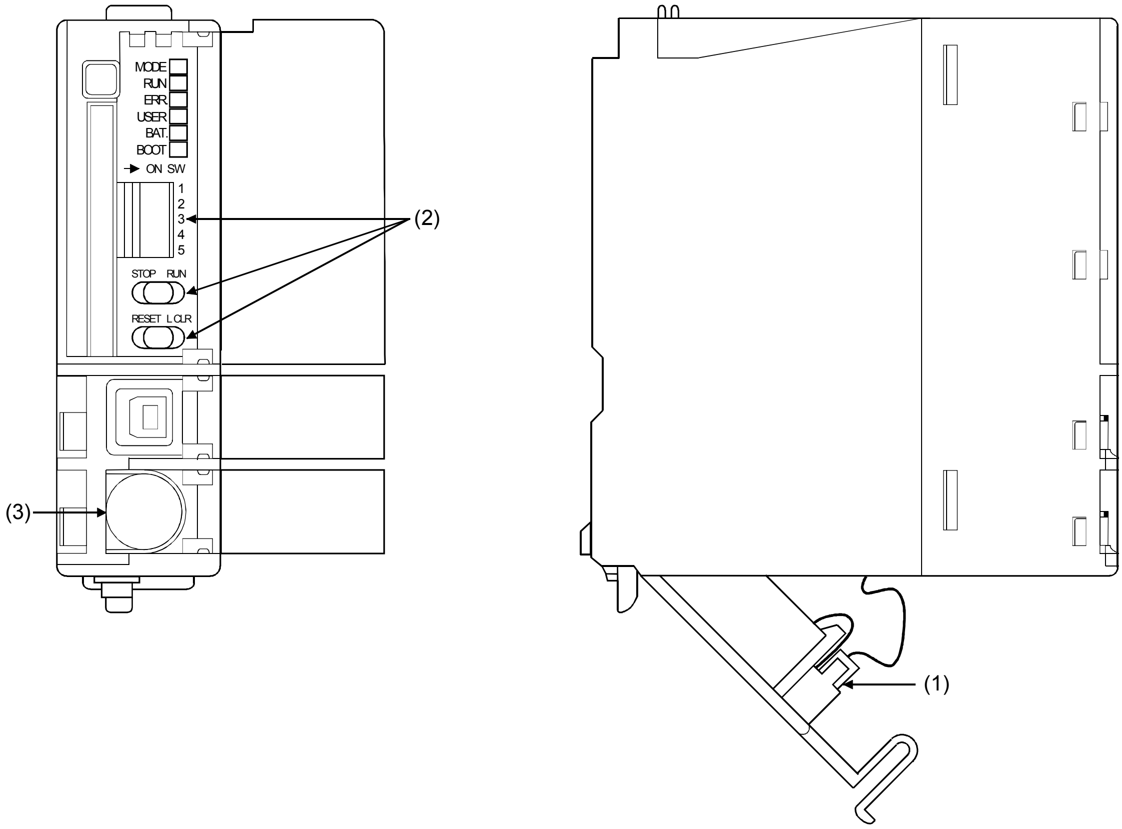
Display of the tool bar changes depending on the selected key type. Assignment of the shortcut keys alters accordingly.



2.3 Preparations for Starting Up CPU

Setting up switches and formatting the built-in memory are required in advance to writing a program to the CPU.

Connect or set up the connectors and the switches indicated as (1) to (3) shown below.
(The figures below are illustrations of Q02HCPU. The locations of the switches and connectors are the same with other CPUs.)

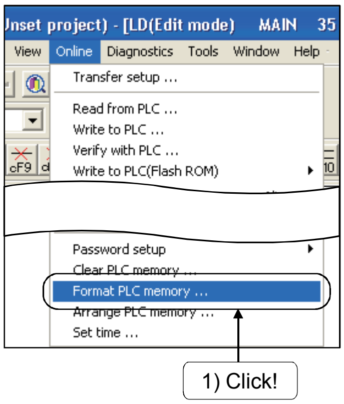


(1) Connect a battery
Connect the battery connector to the CPU as it is not connected with shipment.

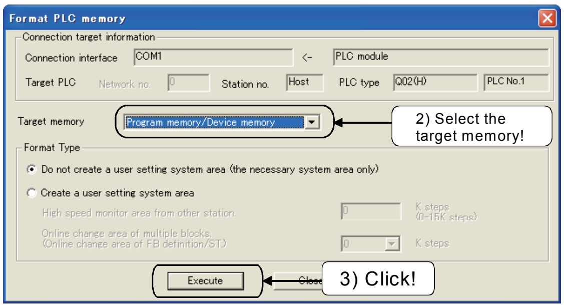
(2) Set up the switches
Set up the DIP switches for the system setting and the RUN/STOP switch.
1) Setting the DIP switches for the system setting
Set all the switches to Off.
(Set switch 1 to Off to remove its write and control protection restrictions.)
2) Setting the RUN/STOP switch
Move the switch to the STOP position.
3) Setting the RESET/L.CLR switch
Move the switch to the central position.

(3) Connect the RS-232 cable

(4) Format the built-in memory of the CPU
The program memory of the QCPU is formatted in the following procedure.

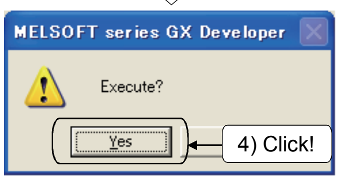


1) Click the [Online] → [Format PLC memory] menu.

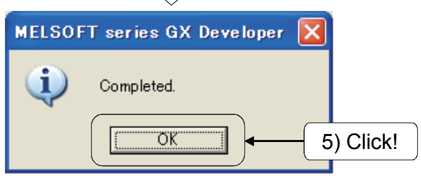


2) The dialogue box as shown on the left will appear. Select "Program memory / Device memory" from the Target memory drop-down menu.

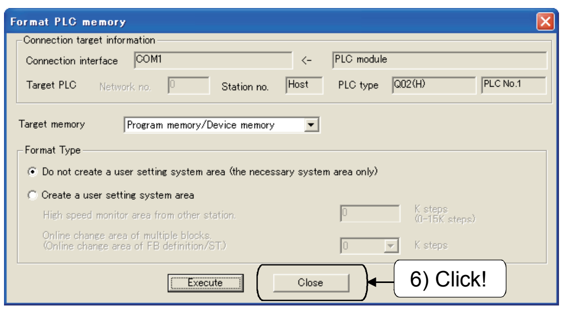
3) Click the Execute button.



4) Click the Yes button to begin format.

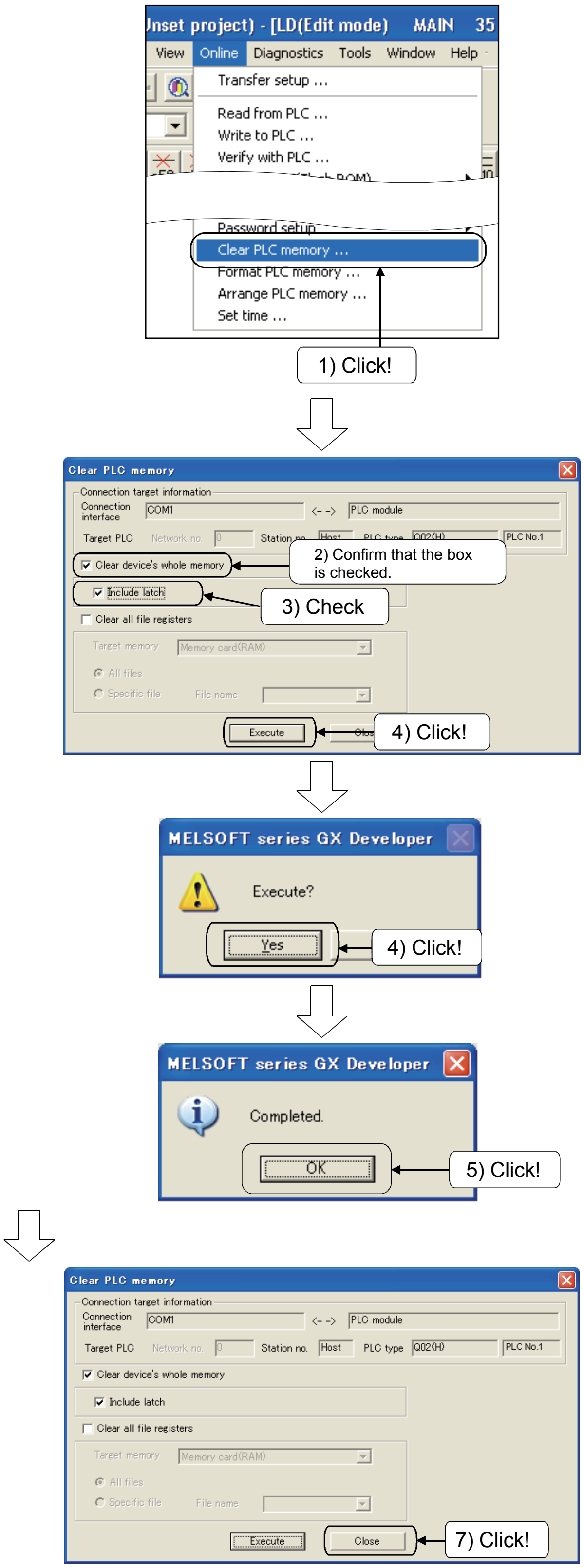


5) When format is complete, the dialogue box shown left will appear. Click the OK button.



6) Click the Close button to close the dialog box.

(5) Clear all the device memory from the CPU
Clears the device memory from the QCPU.



1) Click the [Online] → [Clear PLC memory] menu.

2) The dialogue box as shown on the left will appear. Confirm that the "Clear devices whole memory" checkbox is checked.

3) Check the "Include latch" checkbox.

4) Click the [Execute] button.

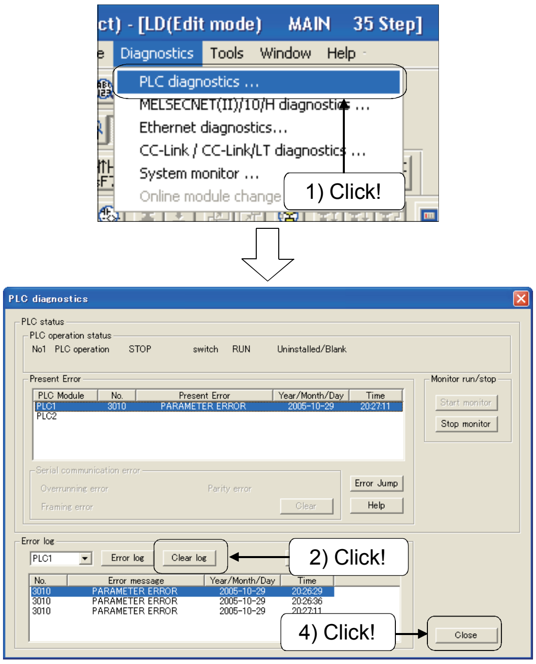
5) Click the [Yes] button to clear the latch device.

6) When completed, the dialogue box as shown on the left will appear. Click the [OK] button.

7) Click the [Close] button to close the dialog box.

(6) Clear the fault history in the CPU

Clear the fault history data stored in the QCPU.



1) Click the [Diagnostics] → [PLC diagnostics] menu.

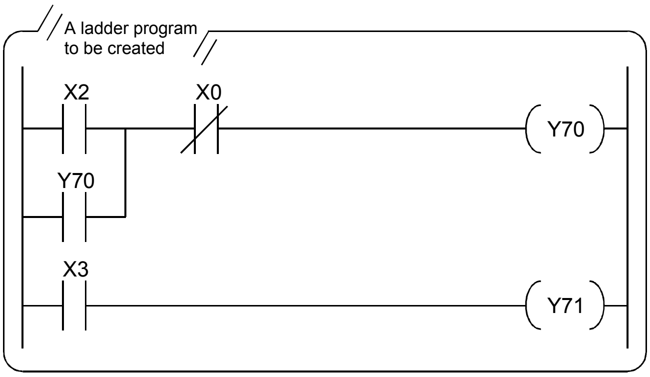
2) The dialogue box as shown on the left will appear. Click the Clear log button.

3) The dialogue box for confirmation will appear. Click the Yes button.

4) Click the Close button to close the dialogue box.

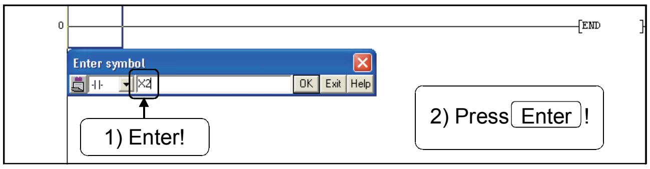
2.4 Creating a Ladder Program

2.4.1 Creating a ladder program using the function keys



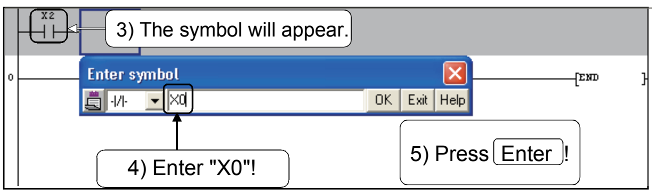
Follow the steps below to create the ladder program as shown on the left.

Make sure that the write mode is active before creating the ladder program.



1) Press the F5 key to open the Enter symbol screen. Enter "X2".
   If you pressed other key(s) by mistake, press Esc and retype.
2) Press Enter to confirm the entry.

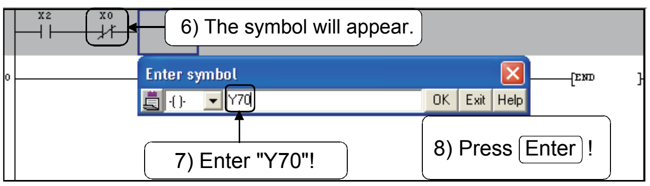
You can also use the OK or Cancel button to confirm or cancel your entry.



3) The symbol you entered ($\overset{X2}{\dashv\vdash}$) will appear.

4) Press the Shift + F5 keys, and enter "X0".

5) Press Enter to confirm the entry.



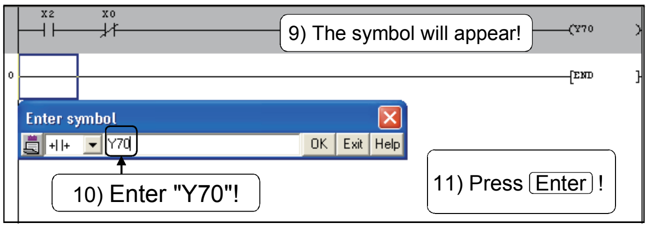
6) The symbol you entered ($\overset{X0}{\dashv/\vdash}$) will appear.

7) Press the F7 key, and enter "Y70".

8) Press Enter to confirm the entry.

(Continued on the next page)

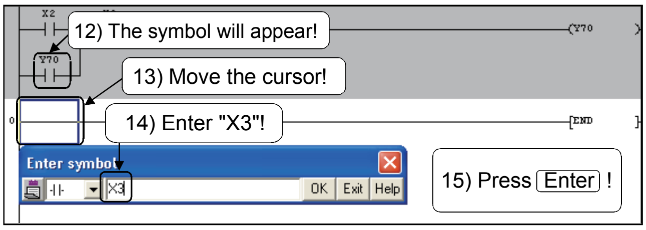
(Continued from the previous page)



9) The symbol you entered (–(Y70)–) will appear.

10) Press the F6 key, and enter "Y70".

11) Press Enter to confirm the entry.

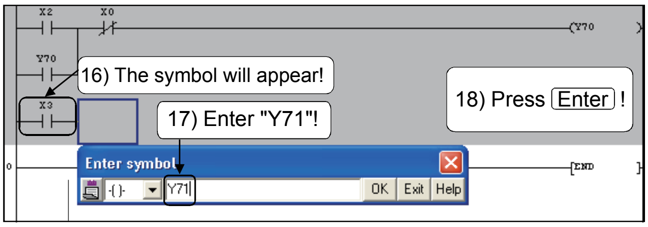


12) The symbol you entered (Y70 –| |–) will appear.

13) Move the cursor to the ladder under Y70 –| |–.

14) Press the F5 key, and enter "X3".

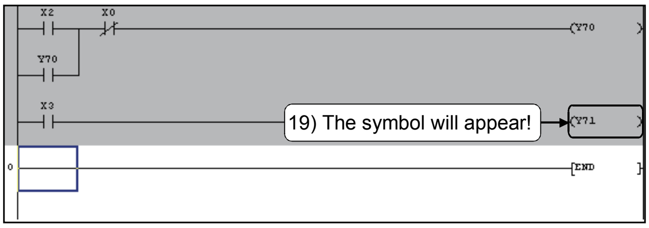
15) Press Enter to confirm the entry.



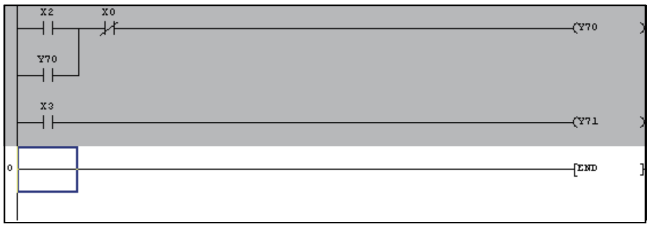
16) The symbol you entered (X3 –| |–) will appear.

17) Press the F7 key, and enter "Y71".

18) Press Enter to confirm the entry.



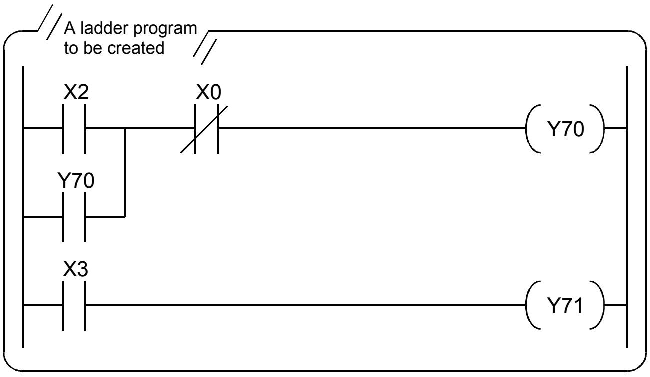
19) The symbol you entered (–(Y71)–) will appear.



20) This is the end of the procedure.

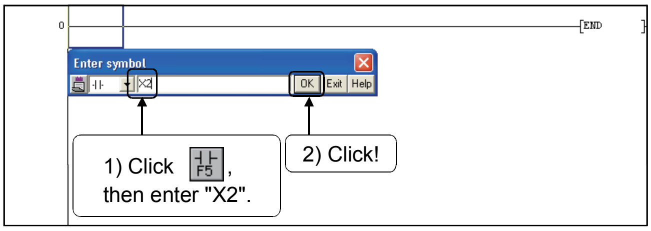
(Continued on the next page)

2.4.2 Creating a ladder program using the tool buttons



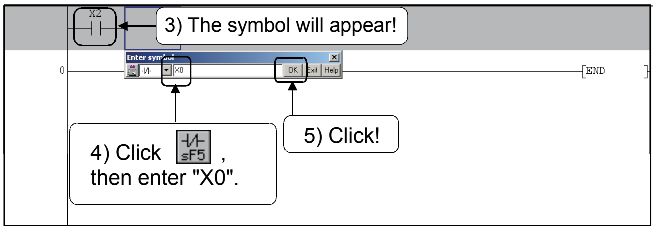
Follow the steps below to create the ladder program as shown on the left.

Make sure that the write mode is active before creating the ladder program.



1) Click on the tool bar to open the Enter symbol screen. Enter "X2".
If you clicked other key(s) by mistake, press the Exit button.

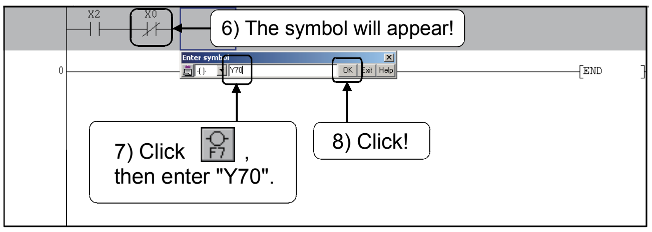
2) Click the OK button to confirm the entry.



3) The symbol you entered (X2) will appear.

4) Click on the tool bar, and enter "X0".

5) Click the OK button.



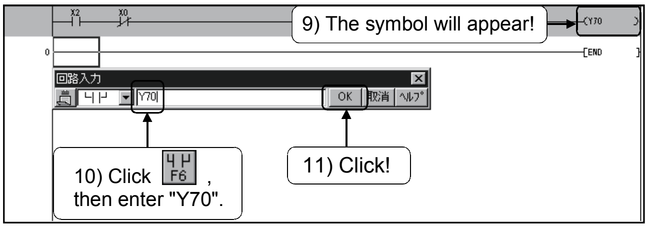
6) The symbol you entered (X0) will appear.

7) Click on the tool bar, and enter "Y70".

8) Click the [OK] button.

(Continued on the next page)

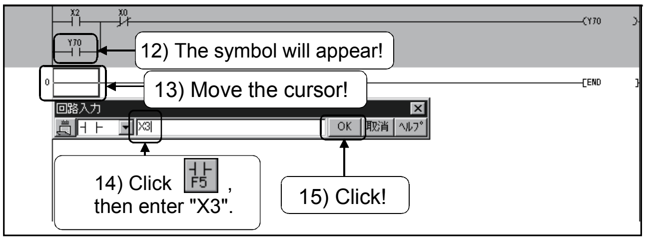
(Continued from the previous page)



9) The symbol you entered (-(Y70)-) will appear.

10) Click F6 on the tool bar, and enter "Y70".

11) Click the OK button.

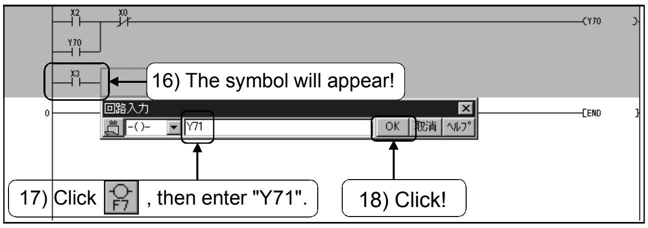


12) The symbol you entered (Y70 -| |-) will appear.

13) Move the cursor to the ladder under Y70 -| |-.

14) Click F5 on the tool bar, and enter "X3".

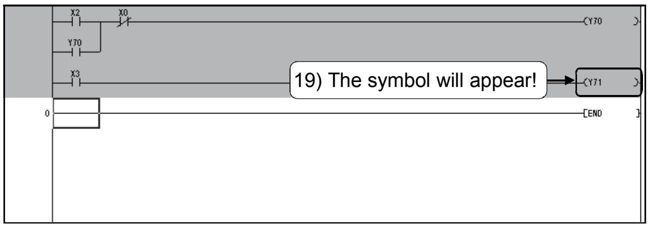
15) Click the OK button.



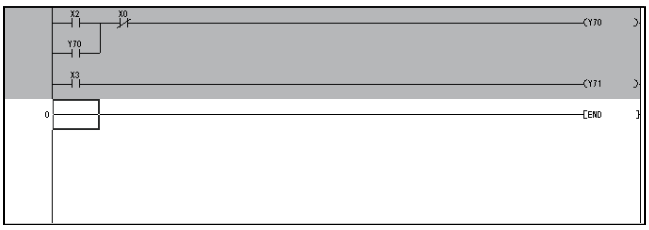
16) The symbol you entered (X3 -| |-) will appear.

17) Click F7 on the tool bar, and enter "Y71".

18) Click the OK button.

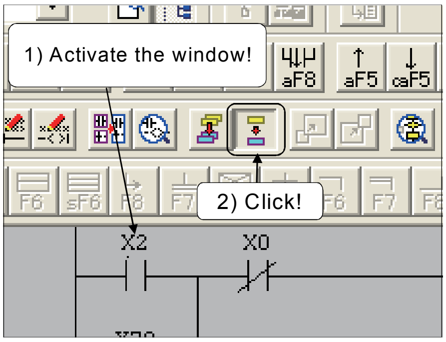


19) The symbol you entered (-(Y71)-) will appear.



20) This is the end of the procedure.

2.5 Converting the Program



1) Activate and select the window that displays the ladder you want to convert.

2) Click on the tool bar or select the [convert] → [convert] menu (F4).

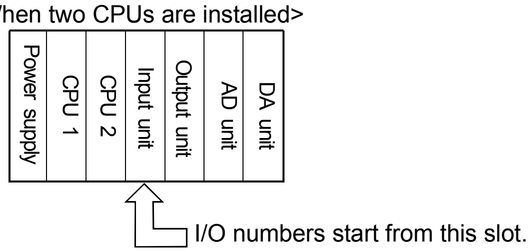
If an error occurs while converting, the cursor will automatically move to the defective point of the ladder program. Check the point and correct the program as necessary.

2.6 Writing to the PLC CPU

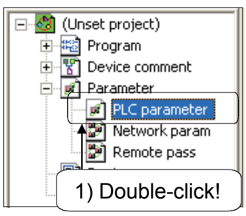
(1) Parameter setting for Multiple CPUs (if only one CPU is installed, omit this step.)
The Q-series Multiple-CPU version demonstration machines are equipped with two CPUs.
Those machines are not dealt with in this textbook, however, it is required to set the PLC parameters of each CPU for the reason mentioned below.

Each CPU should be informed where in the main base slot the I/O numbers begin

<When two CPUs are installed>



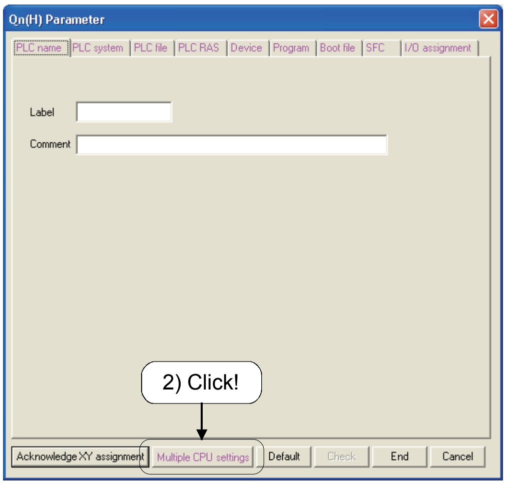
Follow the steps below to set the parameters. For details on parameters, refer to "3.2 Parameters".



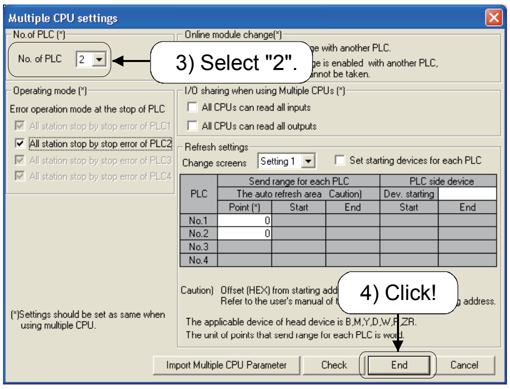
1) Double-click "PLC parameter" on the GX Developer project list.

(Continued on the next page)

(Continued from the previous page)



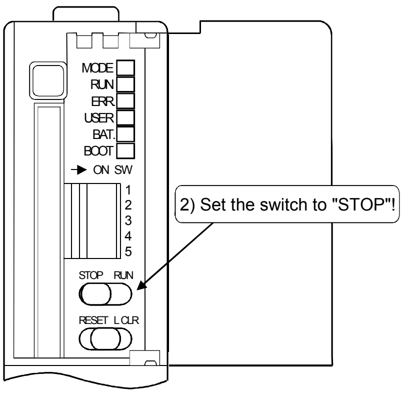
2) The Qn (H) parameter setting dialogue box will appear. Click the Multiple CPU settings button.



3) Select "2" in <Number of CPU> box in the Multiple CPU settings dialogue box.

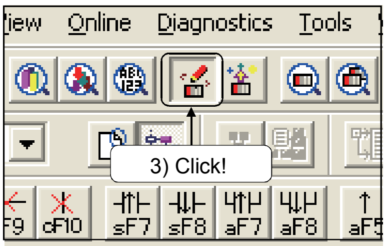
4) Click the End button.

(2) Writing to CPU



1) Suppose that the ladder program (sequence program) has been created with GX Developer to proceed to the next step.

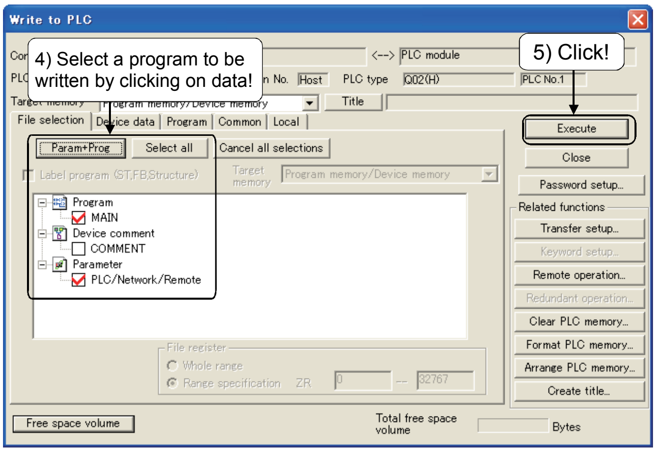
2) Set the RUN/STOP switch on the CPU to STOP.



3) Click [icon] on the tool bar or click [Online] → [Writ to PLC] menu.

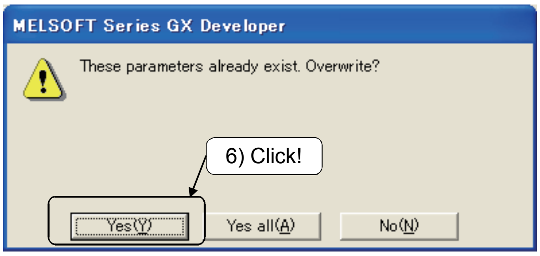
(Continued on the next page)

(Continued from the previous page)

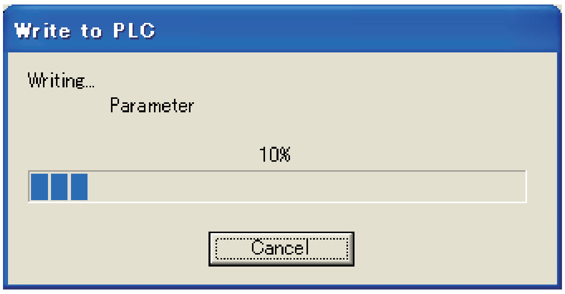


4) From the "File selection" tab, click to select the program and parameter that you want to write to the CPU. Or click Param+Prog to select them.

5) Click Execute to accept the selection.



6) If the parameter or program has already been written to the CPU, the confirmation appears asking if you want to overwrite the parameter/program. Click Yes.



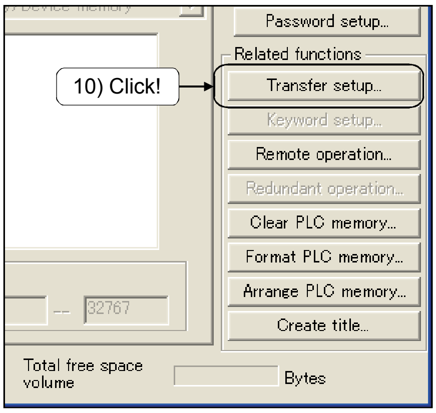
7) The progress bar will appear.

8) The completion pop-up window will appear when writing is complete. Click OK.

(Continued on the next page)

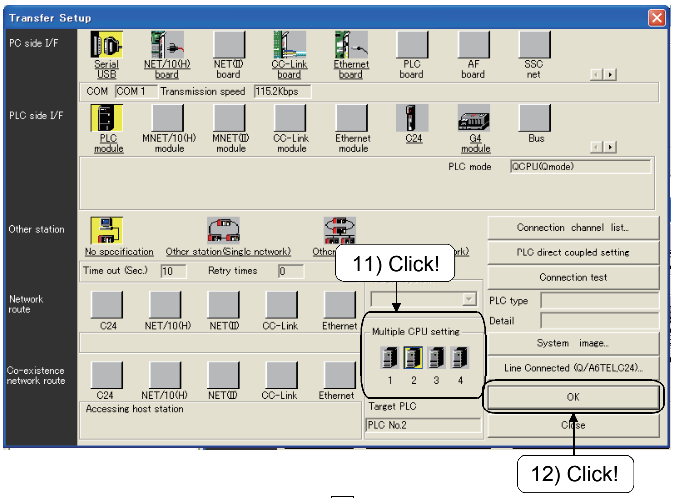
When two CPUs are installed, perform the procedure from step 9) to step 15) explained in the dotted lines on the next page in order to write a parameter into the CPU No.2 When only one CPU is installed, go on to step 17) on page 2-21 .

(Continued from the previous page)



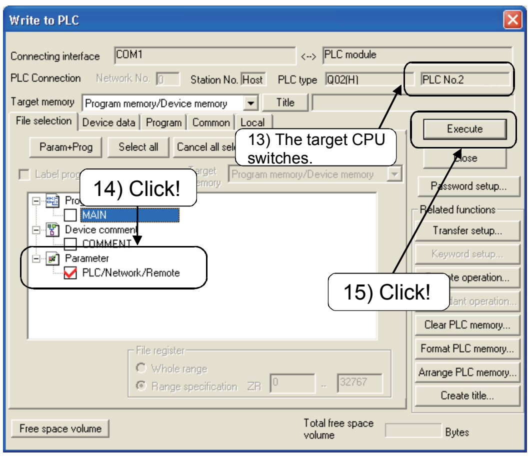
9) Set the RUN/STOP switch of the second PLC CPU to STOP.

10) Click the [Transfer setup] button on the "Write to PLC" dialogue box.



11) The "Transfer Setup" dialog box will appear. Click "2" in the "Multiple CPU setting" column.

12) Click the [OK] button.



13) The target CPU will switch to the second CPU.

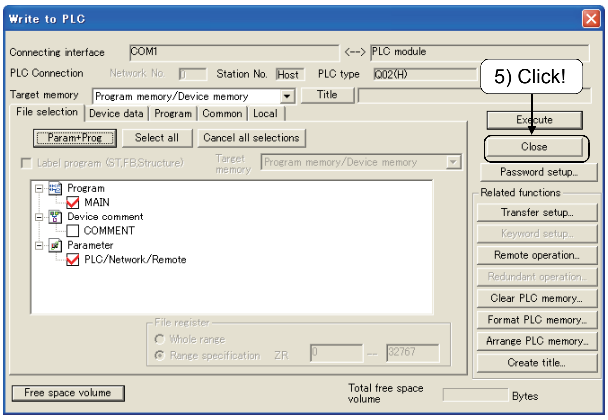
14) Mark the checkbox for "PLC/Network/Remote" placed below the "Parameter".

15) Click the [Execute] button to start writing the parameter to the CPU No.2.

(Continued on the next page)

16) When the writing is complete, follow the steps from 9) through 12) to select the first CPU again. (At Step 11), select "1" in the "Multiple CPU setting" column.)

(Continued from the previous page)



17) Click the Close button to close the dialog box.

IMPORTANT

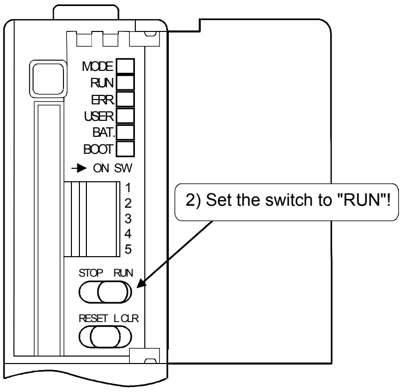
If you practice the operation by following the procedures described in this textbook, resetting/rewriting the parameter for the CPU No.2 is required only in Section 6.3 or in Chapter 7. (In the case where two CPUs are installed.)

<Reasons>

(1) Only the CPU No.1 drives the CPU program.

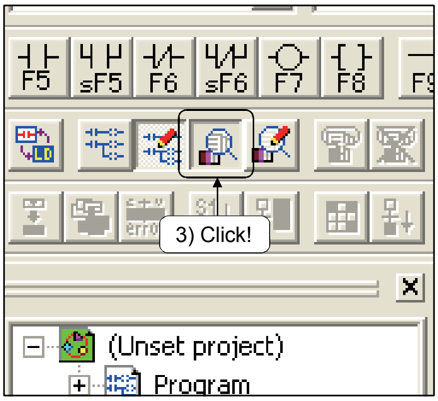
(2) No changes are made to the parameter items that correspond to the multiple CPU system.
(The number of CPUs, refresh, I/O assignment setting)

2.7 Monitoring the Condition of the Ladder Program

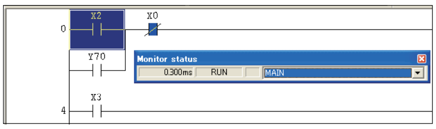


1) Suppose that the ladder program (sequence program) has been written into the PLC to proceed to the next step.

2) Reset with the RESET/L.CLR switch on the CPU, and set the RUN/STOP switch to RUN.



3) Click  on the tool bar or click the [Online] → [Monitor] → [Monitor mode] menu.



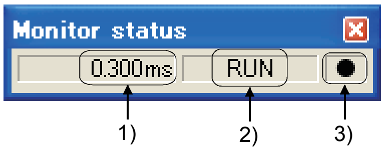
4) Selecting another menu ends the monitor mode.

Operation Practice

1) Confirm that the LED indicator Y70 lights up by turning the snap switch X2 on, and that the indicator remains lit after the snap switch is turned off.
2) Confirm that the LED indicator Y70 turns off by pressing push button X0, and that the indicator does not light up when releasing the button.
3) Turn the snap switch X3 on to turn on LED indicator Y71.

(1) In monitor mode, the monitor status dialogue box shown below will appear regardless of whether the monitoring is activated or not.

<QCPU (Q mode) or QnA series>



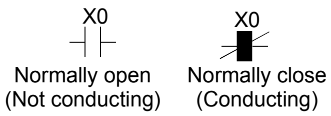
1) Scan time
Shows the maximum scan time of the monitored PLC CPU.
The Q-series device will display the scan time in 0.1-ms increments.

2) PLC CPU condition
Shows the operating condition of the PLC CPU.

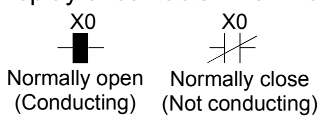
3) Monitor execution status
Flashes while the monitoring is active.

(2) The statuses of ladder are indicated as shown below.

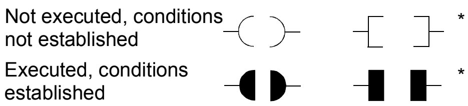
1) Display of contacts when X0 = OFF



Display of contacts when X0 = ON



2) Display of coil output instruction, contact-equivalent comparison instruction, and coil-equivalent instruction



*: Available contact-equivalent comparison and coil-equivalent instructions are; SET, RST, PLS, PLF, SFT, SFTP, MC, FF, DELTA, and DELTAP.

POINT

· The monitor of RST instruction shows on/off status of the device to be reset.

The device to be reset is off :

The device to be reset is on :

· By alternating to list mode, the ladder program can be displayed in list form.

```
0    LD        X2
       <X2         =    ON         >
1    OR        Y70
       <Y70        =    OFF        >
2    ANI       X0
       <X0         =    OFF        >
3    OUT       Y10
       <Y10        =    ON         >
4    LD        X3
       <X3         =    ON         >
5    OUT       Y20
       <Y20        =    OFF        >
6    END
```

In list mode, on/off status is shown as described below.

1) Bit device

Shows the device name and the monitor status under the row where the list instruction is displayed.

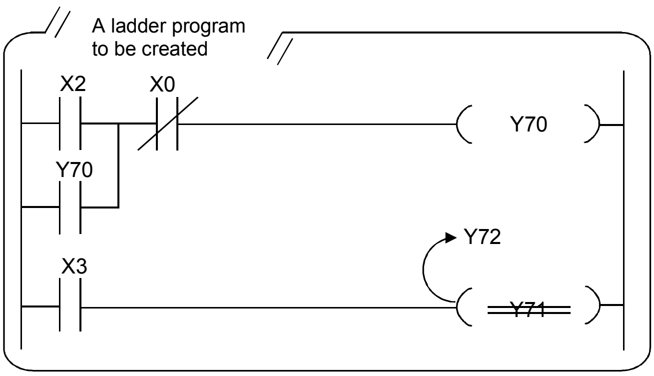
Off: [X0 = OFF], On: [X0 = ON]

2) Word device

Shows the current value.

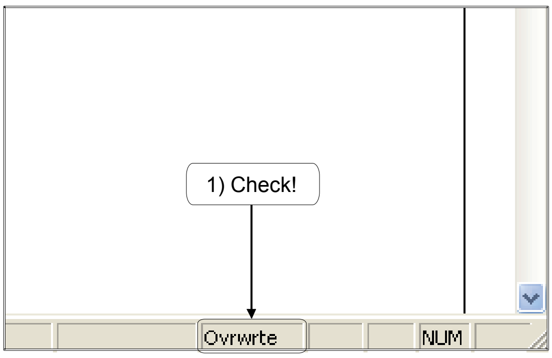
2.8 Editing Ladder Program

2.8.1 Making partial correction to the ladder program



Perform the following steps to make partial correction to the ladder as shown on the left.
(OUT Y71→OUT Y72)

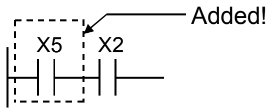
Make sure that the write mode is active before making changes to the ladder program.



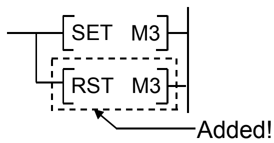
1) Confirm that "Overwrite" is shown in the lower-right portion of the screen.

If "Insert" is shown on the screen, press the Ins key to change the display to "Overwrite".
If "Insert" is shown on the screen, contacts or coils you enter will be added to the diagram.

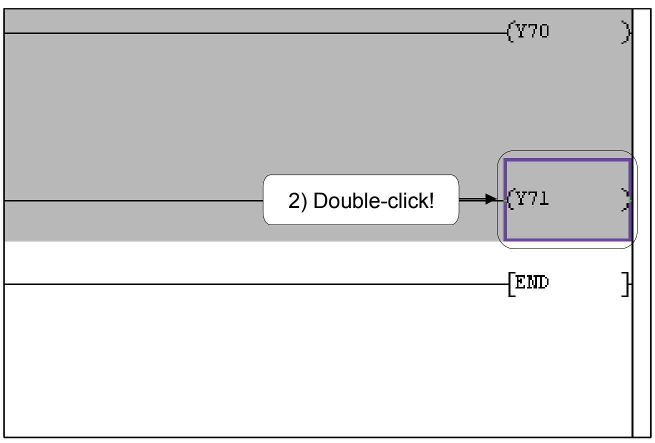
<When correcting from X2 to X5>



<When correcting from SET to RST>

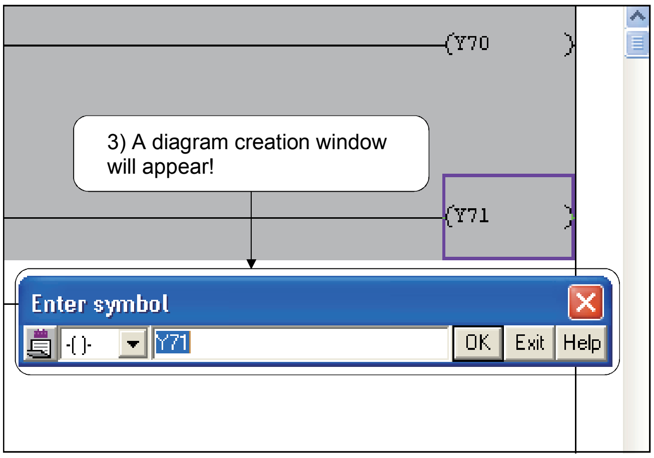


2) Double-click the point you want to correct.

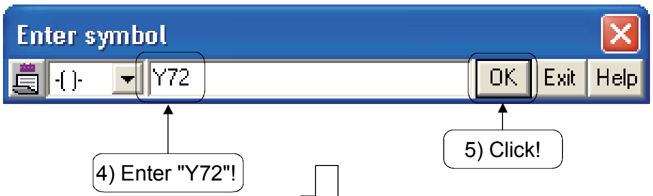


(Continued on the next page)

(Continued from the previous page)



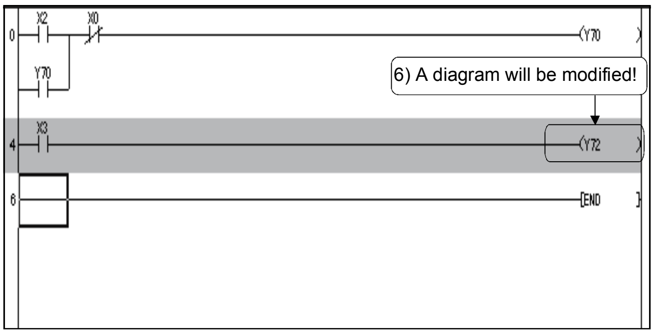
3) The Enter symbol screen will appear.



4) Click the edit box and enter "Y72".

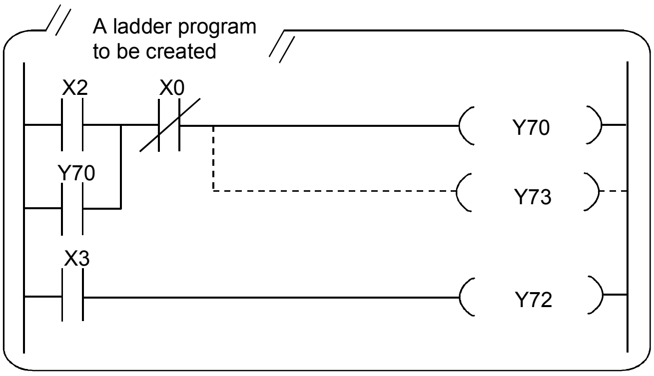
5) Click the OK button to accept the change.

6) The modified ladder program will appear.

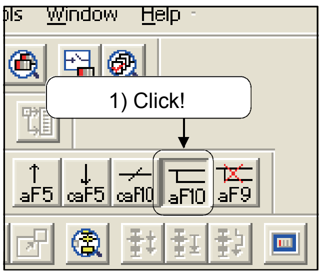


2.8.2 Drawing/Deleting lines

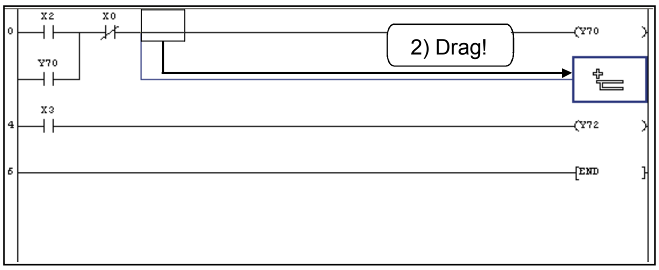
(1) Drawing lines



Perform the following steps to add a line to the ladder as shown on the left.

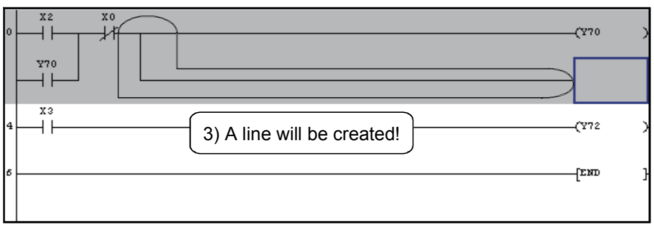


1) Click (Alt + F10) on the tool bar.



2) Drag the mouse from the start position to the end position.

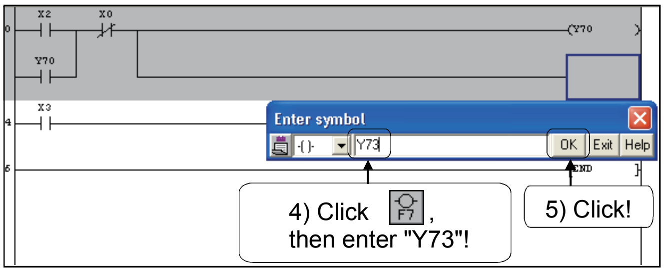
A vertical line will be created to the left of the cursor.



3) A line will be created when the left button of the mouse is released.

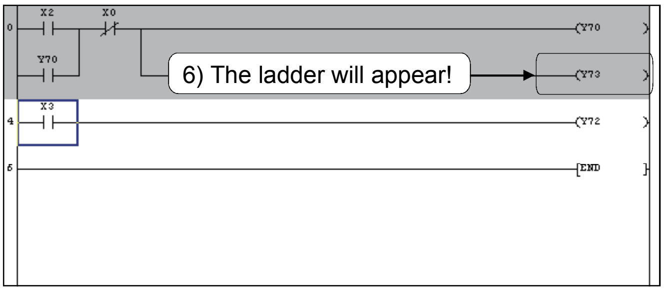
(Continued on the next page)

(Continued from the previous page)



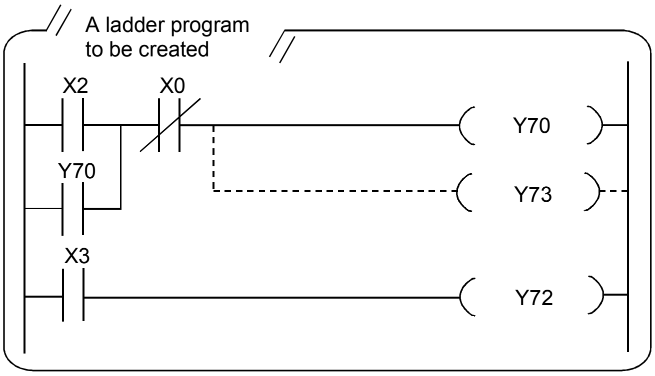
4) Click F7 on the tool bar, and enter "Y73".

5) Click the OK button.

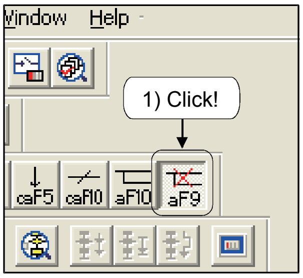


6) The ladder (–(Y73)–) you entered will appear.

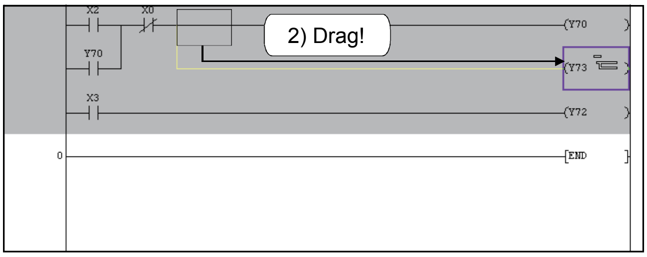
(2) Deleting lines



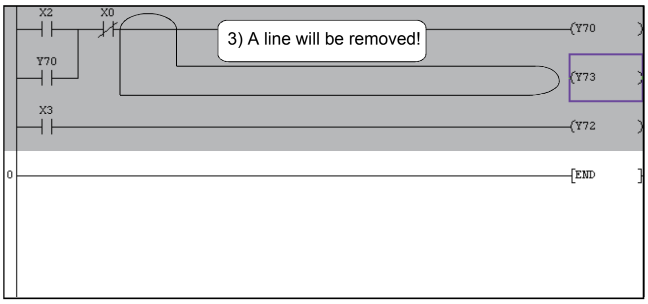
Perform the following steps to delete the line from the ladder shown on the left.



1) Click (Alt + F9) on the tool bar.

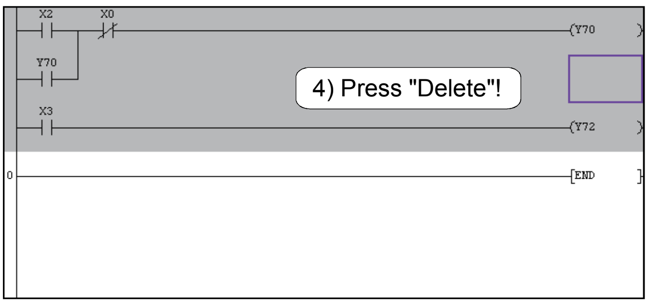


2) Drag the mouse from the start position to the end position that you want to delete.



3) The line will be deleted when the left button of the mouse is released.

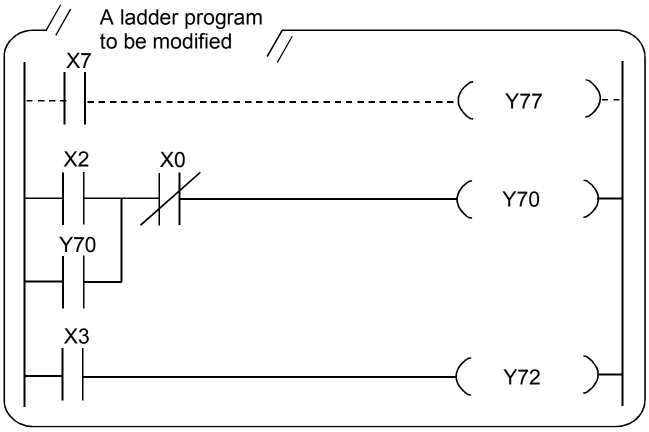
The line drawn for End instruction cannot be removed.



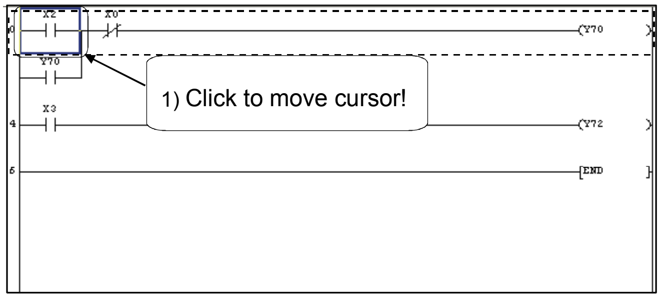
4) Press the Delete key to delete –(Y73)–.

2.8.3 Inserting/Deleting rows

(1) Inserting rows

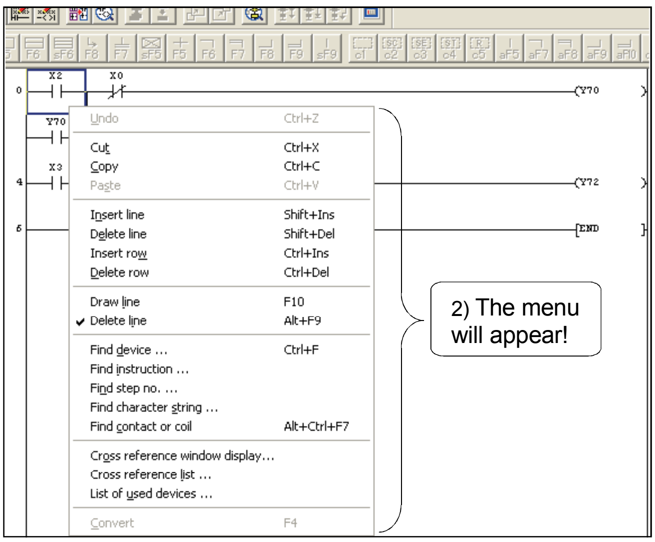


Perform the following steps to insert a row to the ladder shown on the left.



1) Click on any point of the row.

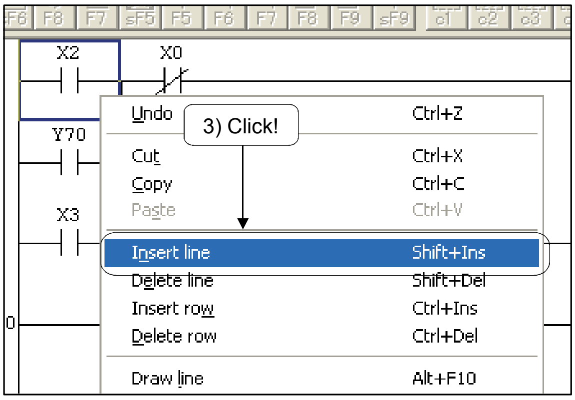
The new row will be inserted above the row selected with the cursor.



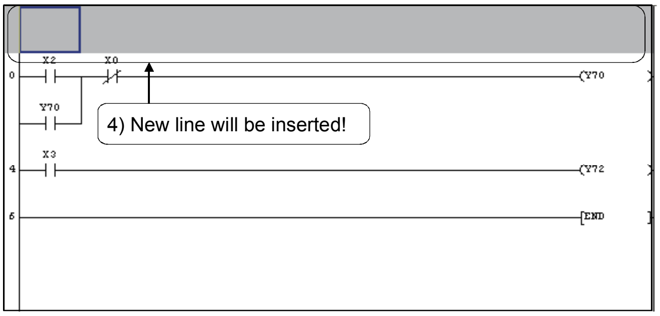
2) Click the right button of the mouse on any point on the ladder window except on the rows to display the menu.

(Continued on the next page)

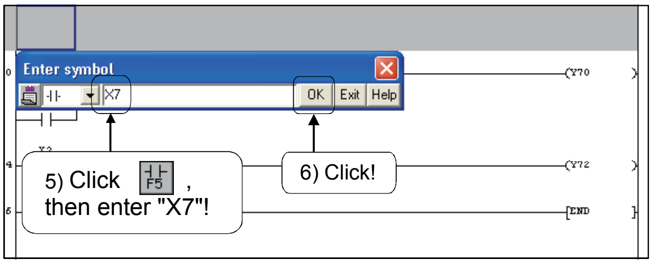
(Continued from the previous page)



3) Click the [Insert line] (Shift + Ins) menu.



4) The new row will be inserted above the selected row.

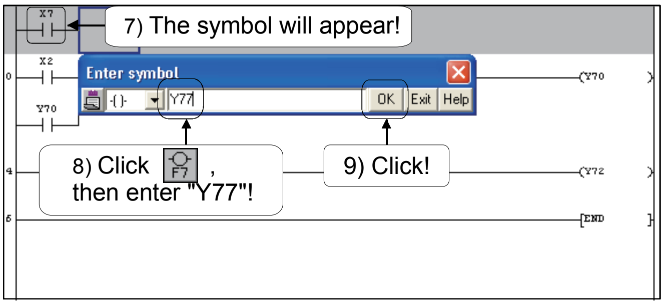


5) Click F5 on the tool bar to open the Enter Symbol screen. Enter "X7".

6) Click the OK button to confirm the entry.

(Continued on the next page)

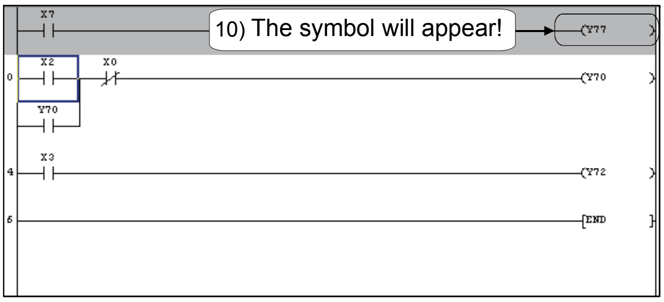
(Continued from the previous page)



7) The symbol (X7) you entered will appear.

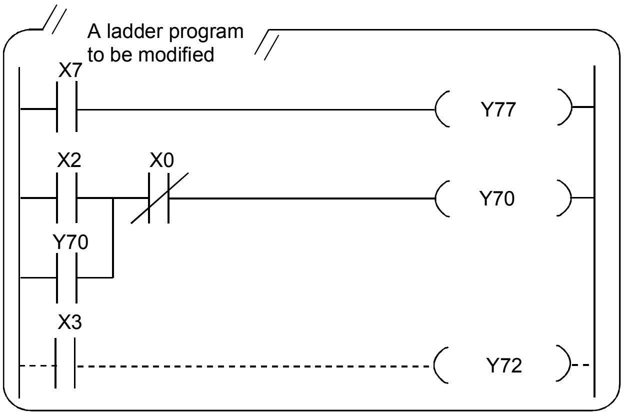
8) Click F7 on the tool bar, and input "Y77".

9) Click the OK button.

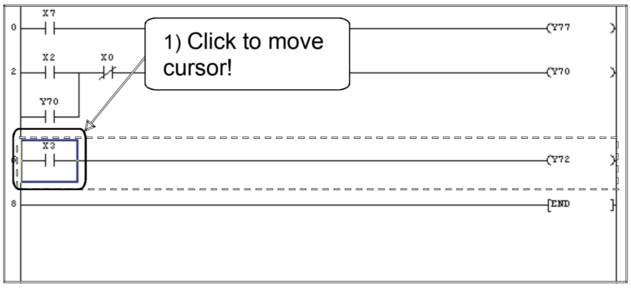


10) The symbol (Y77) you entered will appear.

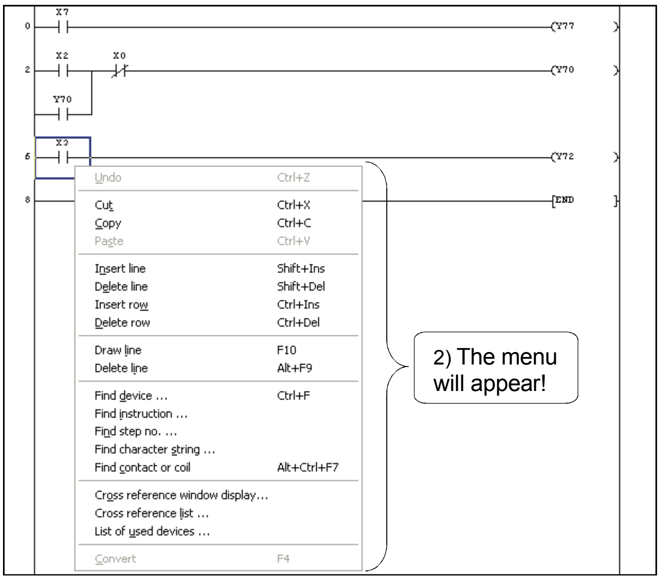
(2) Deleting rows



Perform the following steps to remove the row from the ladder shown on the left.



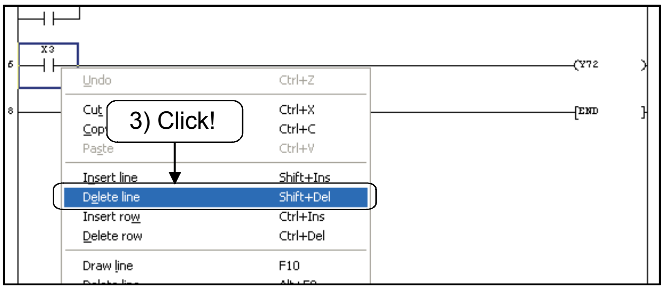
1) Click on any point of the row to be deleted.



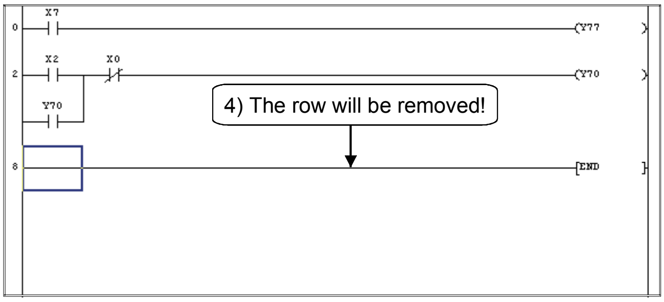
2) Click the right button of the mouse on any point on the ladder window except on the rows to display the menu.

(Continued on the next page)

(Continued from the previous page)

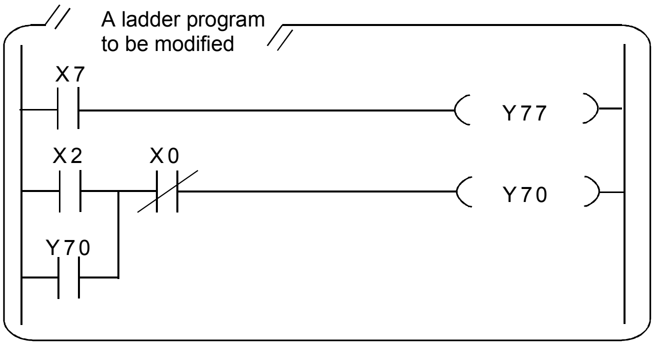


3) Click the [Delete line] (Shift + Del) menu.

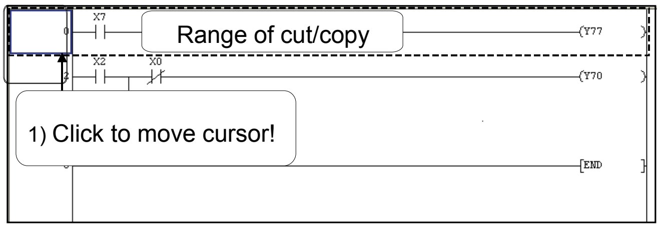


4) The selected row will be deleted.

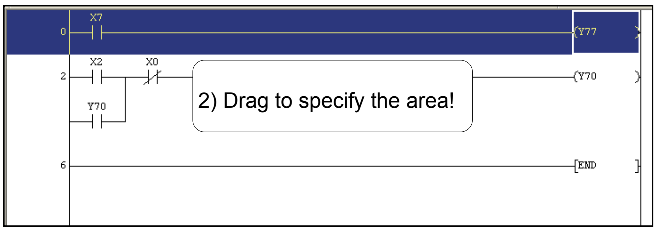
2.8.4 Cutting/Copying ladder program



Follow the steps below to cut and copy the ladder program shown on the left.

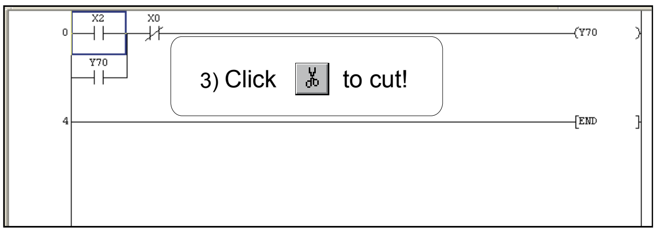


1) Click on the start point of the ladder program you want to cut.



2) Drag the mouse cursor over the ladder to specify the area.
The selected area will be highlighted.

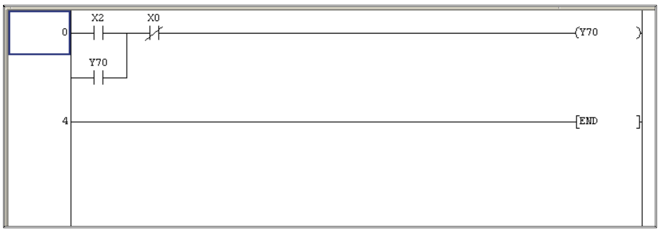
Click the step numbers and drag the mouse cursor vertically to specify the area in ladder block units.



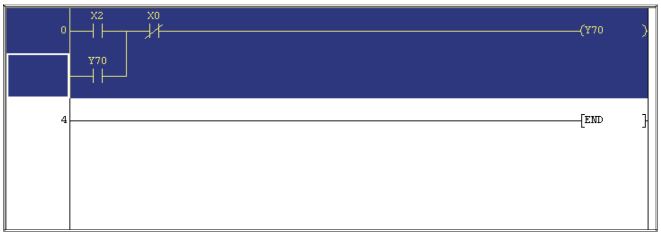
3) Click on the tool bar or select "Edit" → "cut" (Ctrl + X) to cut the specified area.

(Continued on the next page)

(Continued from the previous page)

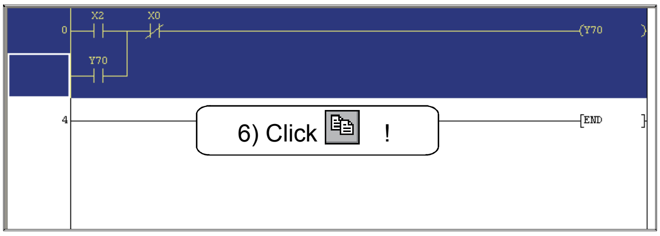


4) Click the start point of the ladder program you want to copy.

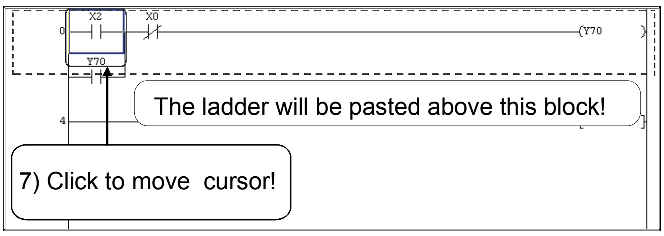


5) Drag the mouse cursor over the ladder to specify the area.
The selected area will be highlighted.

Click the step numbers and drag the mouse cursor vertically to specify the area in ladder block units.



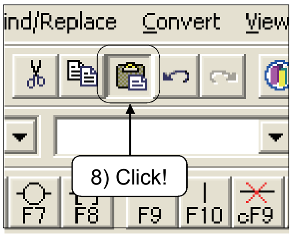
6) Click [icon] on the tool bar or select "Edit" → "copy" (Ctrl + C) to copy the specified area.



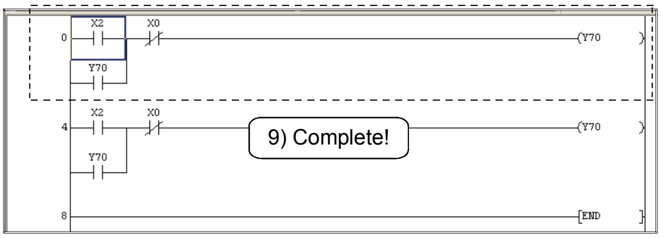
7) Click any ladder block to move the cursor to the ladder. The copied ladder will be pasted right above the row with a cursor.

(Continued on the next page)

(Continued from the previous page)



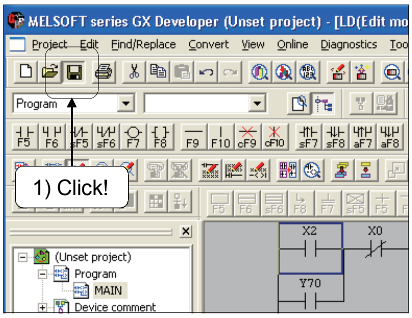
8) Click on the tool bar or select the "Edit" → "paste" menu (Ctrl + V) to paste the specified area.



9) The copied ladder will be pasted.

2.9 Saving Ladder Program

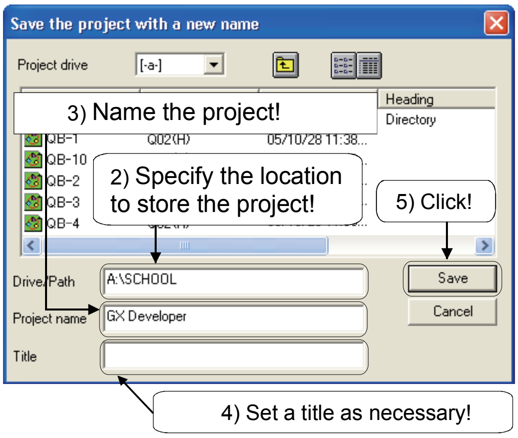
2.9.1 Saving newly-created or overwritten projects



1) Click [save icon] on the tool bar or select the [Project] → [Overwritten Project] menu (Ctrl + S).

Saving the overwritten project completes at this step.

(Saving newly-created project)

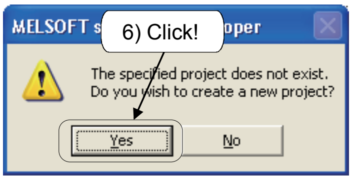


2) Specify the area to store the new project.

3) Name the project.

4) Set a title as necessary.

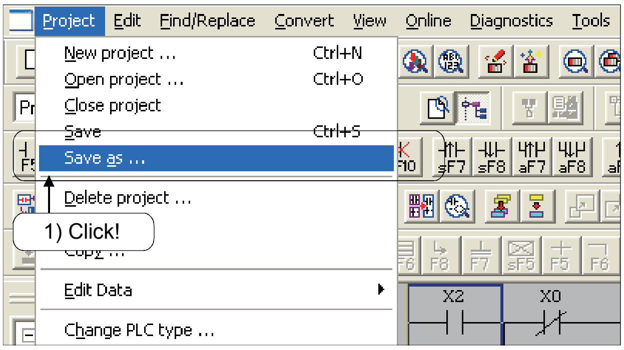
5) Click the Save button to confirm your entry.



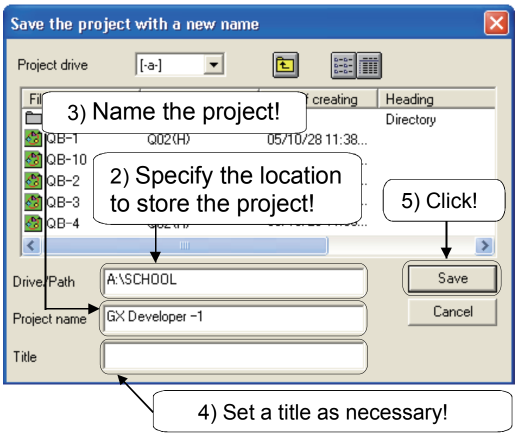
6) Click the Yes button.
The new project will be stored.

Reference
· The following characters cannot be used for the project name. /, \, >, <, *, ?," ", |, :, ; (: and \ can only be used to specify a drive.) Do not place a period (.) at the end of the name. · Eight or more characters can be used for the project name when operating with GX Developer (SW8D5-GPPW or later), however, only the first seven characters will be displayed if the project is read in GX Developer (SW2D5-GPPW or older). · Maximum 150 characters are allowed for the project path + project name. · Maximum 32 characters are allowed for the title. · If space(s) is included in the project path or the project name, GX Developer will not start up properly by double-clicking GPPW.gpj, ***.gps on the Explorer. To open such project, start GX Developer first and open the project by selecting the [Project] → [Open project] menu.

2.9.2 Saving a project under another name



1) Click [Project] → [Save as ...] on the menu bar.

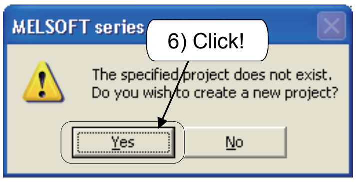


2) Specify the location to store the project.

3) Name the project.

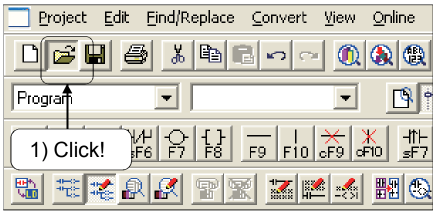
4) Set a title as necessary.

5) Click the [Save] button to confirm your entry.

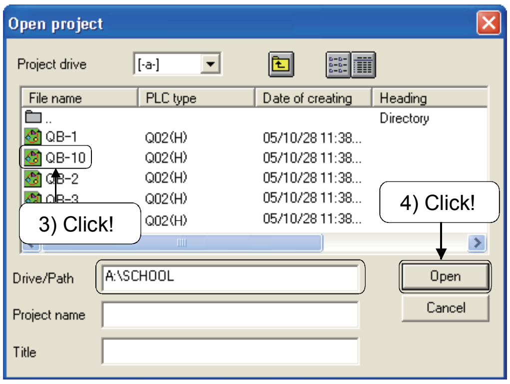


6) Click the [Yes] button.
The project will be stored under the new name.

2.10 Reading the Saved Project



1) Click on the tool bar or select the "Project" → "Open project" menu (Ctrl + O).

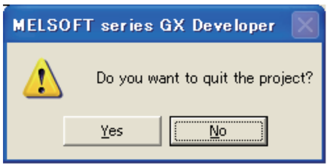


2) Specify the location where the project you want to open is stored.

3) Click on the project.

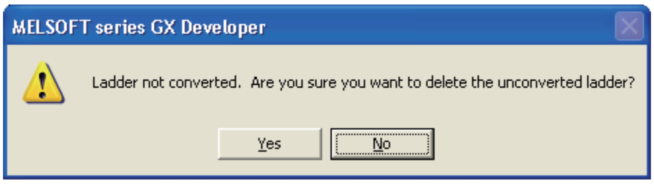
4) GX Developer starts to read the specified project.

The following dialogue box will appear depending on the condition (when another project has been open).



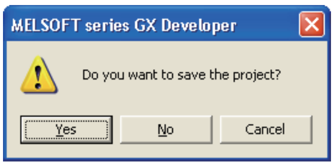
Yes : will terminate the project.
No : will keep the project activated.

When another project has been open without being converted.



Yes : will terminate the project without converting the project
No : will keep the project activated. (Continue editing the ladder program.)

When another project has been open without being saved.

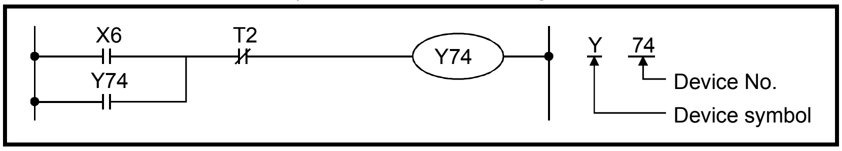


Yes : will save the project before terminating it.
No : will terminate the project without saving it.
Cancel : will keep the project activated.

CHAPTER 3 PLC DEVICES AND PARAMETERS

3.1 Devices

The devices are imaginary elements for programming in the PLC's CPU, as well as the components (such as contacts and coils) that compose a program.



Type		Description	Remark
X	Input	Transmits instructions or data to the PLC through the external devices such as the push buttons, selector switches, limit switches, and digital switches.	· Bit device · Mainly deals with the on/off signals.
Y	Output	Outputs to the solenoids, electromagnetic switches, signal lights or digital indicators as a result of control.	
M	Internal relay	Auxiliary relay inside the PLC that cannot output directly to the external devices.	
L	Latch relay	Uninterruptible auxiliary relay inside the PLC that cannot output directly to the external devices.	
S	Step relay	Auxiliary relay inside the PLC that cannot output directly to the external devices.	
B	Link relay	Internal relay for data link that cannot output directly to the external devices. The area not assigned by initial link information setting can be used as the internal relay.	
F	Annunciator	Used for failure detection. Create the failure detection program beforehand and turn on the program while the PLC is running to store the numerical values in the special register D.	
V	Edge relay	Internal relay that stores the operation result (on/off information) from the top of the circuit block.	
SM	Special relay	Internal relay that stores the CPU conditions.	
SB	Special link relay	Internal relay for data link that indicates the communication status and errors.	
FX	Function input	Internal relay that captures the on/off data specified by the subroutine call instructions with arguments in the subroutine program.	
FY	Function output	Internal relay that transmits the operation result (on/off data) in the subroutine program to the subroutine program call source.	
T(ST)	Timer	Accumulative timers of four types: low-speed timer, high-speed timer, low-speed integrator, and high-speed integrator.	· Word device · Mainly deals with a data. · One word consists of 16 bits. · Can be specified by entering "～ .*". (*=0 to F (hexadecimal).
C	Counter	Accumulative counters of two types: the counters for the sequence program and the counters for interruption sequence program.	
D	Data register	Memory that stores the data in the PLC.	
W	Link register	Data register for data link.	
R	File register	Register for the extensive use of data registers, which uses user memory area.	
SD	Special register	Register that stores the CPU conditions.	
SW	Link data register	Data register for data link that stores the communication status and failure information.	
FD	Function register	Register for the exchange data between the subroutine call source and the subroutine program.	
Z	Index register	Registers for modification to the devices (X, Y, M, L, B, F, T, C, D, W, R, K, H, and P).	

Type		Description	Remark
N	Nesting	Shows the nesting (nested structure) of the master control.	
P	Pointer	Locates the jump addresses of the branch instructions (CJ, SCJ, CALL and JMP).	
I	Interruption pointer	Locates the jump address that corresponds to the factor of the interruption when an interruption occurs.	
BL	SFC block device	Device that checks if the SFC program designated block is activated or not.	
TR	SFC transition device	Device to check if the designated transition condition of SFC program designated block is specified as forced transition or not.	
J	Network No. designation device	Used when designating Network No. by the instructions of data link.	
U	I/O No. designation device	Used when designating I/O No. by the instructions for the intelligent function module.	
K	Decimal constant	Used when designating the followings; timer counter set value, pointer number, interruption pointer number, number of digits of bit device, and basic/application instruction values.	
H	Hexadecimal constant	Used when designating the basic/application instruction values.	
E	Real constant	Used when specifying real numbers as instructions.	
"Character string"	Character-string constant	Used when specifying character strings as instructions.	
Jn\X Jn\Y Jn\S Jn\SB	Link direct device	Device that can access directly to the link device of the network module. (Establish the refresh parameter beforehand.)	· Bit device · Mainly deals with the on/off signals.
Jn\W Jn\SW Un\G			· Word device · Mainly deals with a data. · One word consists of 16 bits.
Un\G	Intelligent function module device	Device that can access directly to the buffer memory of the intelligent function module.	

3.2 Parameters

The parameters are basic settings applied to the PLC in order to control the object as planned.
The parameters are divided into the PLC parameter, network parameter, and remote password as shown below.

- PLC Parameter
  - PLC Name Setting
    - Label .... Sets the QCPU label (name and purpose)
    - Comment .... Sets the comment of QCPU label
  - PLC System Setting
    - Timer interval setting .... Sets either the low-speed/high-speed timer
    - RUN-PAUSE contact .... Sets contact that controls QCPU RUN/PAUSE
    - Remote reset .... Permits/Prohibits the remote reset operations from GX Developer
    - STOP-RUN output mode .... Sets the output (Y) status after switching from STOP to RUN
    - Computation on floating decimal point data .... Sets the execution of computing floating decimal point data in double accuracy
    - Intelligent function module setting .... Sets the following items: Interruption pointer assignment (I50 to I255), head I/O No. and head SI No. of intelligent function module
    - Common pointer .... Sets the head No. of the pointer used as a common pointer
    - Number of vacant slots .... Sets the number of vacant slots for the basic/expanded bases
    - System interruption setting .... Sets execution interval for head No. of interruption counters and interruption pointers (I28 to I31)
    - Interruption program/Fixed-cycle program setting .... Sets the execution of interruption program at high speeds
    - Unit synchronization setting .... Sets the synchronization of the start-up of QCPU with that of intelligent function module
    - A series CPU compatibility setting .... Sets whether or not to use the special relay/register for MELSEC-A series
  - PLC File Setting
    - File register .... Sets the file of the file register used throughout the program
    - Comment file used in instructions .... Sets the file of the comment used throughout the program
    - Device initial value .... Sets file of the device initial value used for QCPU
    - Local device file .... Sets the local device file used throughout the program
  - PLC RAS Setting
    - WDT setting .... Sets the QCPU watchdog timer
    - Operation mode in error condition .... Sets the QCPUs operation mode in the error status
    - Error check .... Sets the detection of specified errors
    - Constant scan .... Sets constant scan time
    - Low speed program execution time .... Sets the execution time for each scanning of the low speed program
    - Failure history .... Sets the area to store the QCPUs failure history
  - Device Setting
    - No. of devices .... Sets the number of devices according to the systems
    - Latch (1) start/last .... Sets the latch range (starting/last No. of devices) cleared by the latch clear key or remote latch clear operation
    - Latch (2) start/last .... Sets the latch range (starting/last No. of devices) cleared by the latch clear key or remote latch clear operation
    - Local device start/last .... Sets the device range (starting/last No. of devices) used at local device
  - Program Setting .... Sets the file name and execution type (execution condition) of the program when writing multiple programs for QCPU
  - Boot File Setting
    - Boot option .... Sets whether or not to clear the program memory at boot
    - Boot file setting .... Sets the type, data name, and transfer from drive of the program file that operate at boot
    - Standard ROM automatic refresh .... Sets the execution of automatic refresh on the standard ROM
  - SFC Setting .... Sets the start mode/condition of the SFC program when the program is in use, and sets the output mode when the block is at a stop
  - I/O Assignment
    - I/O assignment .... Sets the type, model, occupation numbers, and start input/output numbers of the installed module
    - Basic setting .... Sets the basic base module in use, expanded base module, power supply module, model of expanded cable, and the number of the slot of the base module
    - Switch setting .... Sets various switches of the intelligent function module
    - Detailed setting .... Sets the operations at an error and the response time of the input module
- Network Parameter
  - MELSECNET, Ethernet Setting .... Sets the network parameters for MELSECNET/H and Ethernet
  - CC-Link Setting .... Sets the parameters for CC-Link
- Remote Password .... Sets the password that limits the access via the Ethernet or serial communication modules

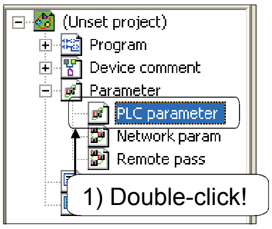
· When GX Developer starts, it employs the preset values as the parameters. These values are called the default (initial values).

· PLC can run with those values unchanged, however, modify them within a specified range as necessary.

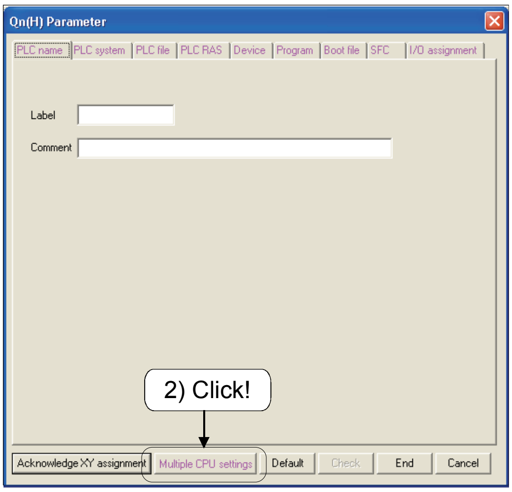
· The second CPU is installed on the demonstration machine for making up a Multiple CPU structure. In MELSEC-Q series, multiple CPU setting for the parameters is required if two to four CPUs are installed.

Operation Example: Changing Multiple-CPU Setting

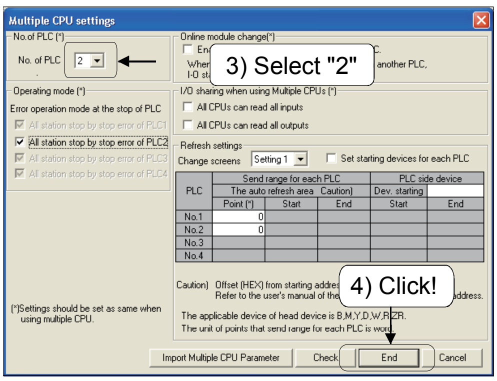
The number of CPUs of Multiple CPU setting is one by default.
Change the value to "two". (If only one CPU is installed, omit this step.)



1) Double-click "PLC parameter" on the GX Developer project list.

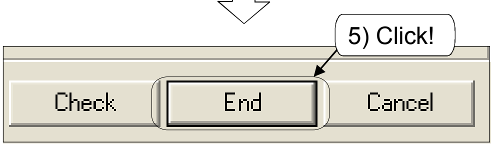


2) The Qn (H) parameter setting dialogue box appears. Click the Multiple CPU settings button.



3) Enter "2" in <Number of PLC> box in the multiple CPU setting dialogue box.

4) Click the End button.



5) Click the End button in the Qn (H) parameter setting dialogue box. The setting is completed.

CHAPTER 4 SEQUENCE & BASIC INSTRUCTIONS -Part 1-

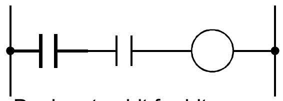
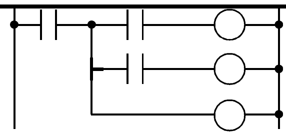
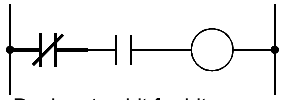
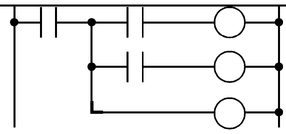
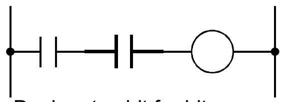
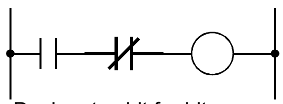
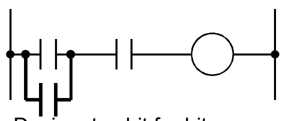
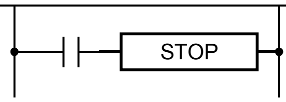
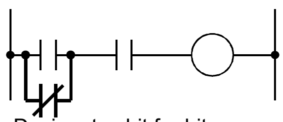
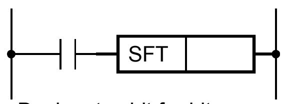
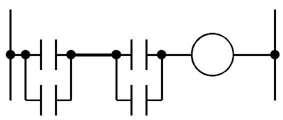
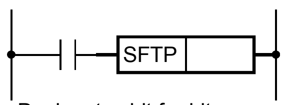
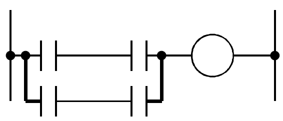
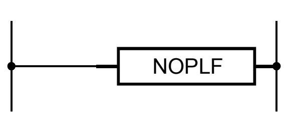
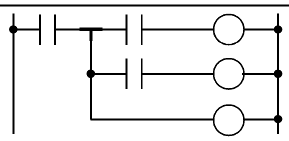
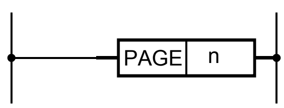
4.1 List of Instructions Described in this Chapter

The table below shows the sequence and basic instructions described in this chapter.

Instruction symbol (Name)	Functions	Drawing (devices to be used)	Instruction symbol (Name)	Functions	Drawing (devices to be used)
OUT Out	Coil output	Designates bit for bit devices and word devices.	CJ	Conditional jump (non-delay)	CJ Pn n = 0 to 4095 (pointer number)
MC Master control	Starts master control	Designates bit for bit devices and word devices. Nn Mc Nn n = 0 to 14 (nesting)	SCJ	Conditional jump (Jumps after one scan)	SCJ Pn n = 0 to 4095 (pointer number)
MCR Master control reset	Terminates master control	MCR Nn n = 0 to 14 (nesting)	CALL	Calls a subroutine program	CALL Pn n = 0 to 4095 (pointer number)
SET Set	Sets devices	SET Designates bit for bit devices and word devices.	CALLP	Calls a subroutine program (pulsing operation)	CALLP Pn n = 0 to 4095 (pointer number)
RST Reset	Resets devices	RST Designates bit for bit devices and word devices.	RET Return	Returns from a subroutine program	RET
PLS Pulse	Pulse (Generates the pulses for one program cycle during an input)	PLS Designates bit for bit devices and word devices.	FEND	Terminates a mainroutine program	FEND
PLF Pulf	Pulf (Generates the pulses for one program cycle during an input)	PLF Designates bit for bit devices and word devices.			

[List of Instructions Not Described in this Chapter: Part 1]

"Introduction: PLC Course" covers the instructions shown below. The conventional A series also support them. Refer to "QCPU (Q mode) / QnACPU Programming Manual (Common Instructions)" for more details.

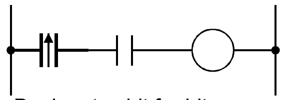
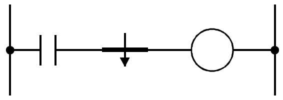
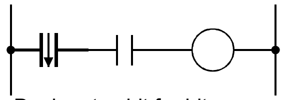
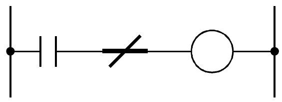
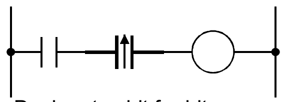
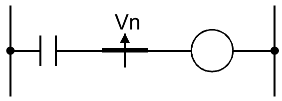
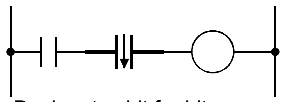
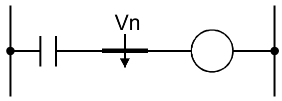
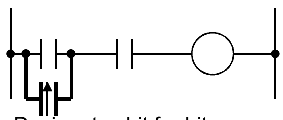
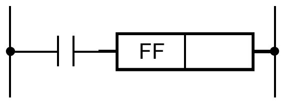
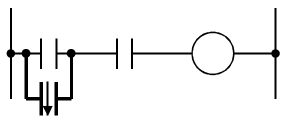
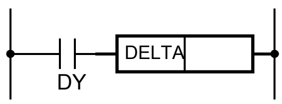
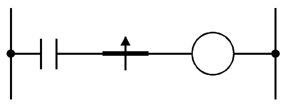
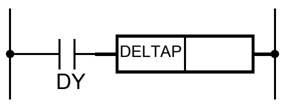
Instruction symbol (Name)	Functions	Drawing (devices to be used)	Instruction symbol (Name)	Functions	Drawing (devices to be used)
LD Load	Starts logical operation (Starts to operate normally open)	 Designates bit for bit devices and word devices.	MRD Lead	Intermediate branching	
LDI Load inverse	Starts logical inverse operation (Starts to operate normally open)	 Designates bit for bit devices and word devices.	MPP Pop	Terminates branching	
AND And	Logical AND operation (normally open series connection)	 Designates bit for bit devices and word devices.	NOP Nop	Ignored	For a space or deleting a program
ANI And inverse	Logical AND inverse operation (normally close series connection)	 Designates bit for bit devices and word devices.	END	Terminates a program END processing	Must be used as an end of a program
OR Or	Logical OR operation (normally open parallel connection)	 Designates bit for bit devices and word devices.	STOP	Stops operation	
ORI Or inverse	Logical OR inverse operation (normally close parallel connection)	 Designates bit for bit devices and word devices.	SFT Shift	1-bit shift for devices	 Designates bit for bit devices and word devices.
ANB And block	AND operation between logical blocks (Series connection between blocks)		SFTP Shift P	1-bit shift for devices (pulsing operation)	 Designates bit for bit devices and word devices.
ORB Or block	OR operation between logical blocks (Parallel connection between blocks)		NOPLF	Ignored (Inserts a page break when printing)	
MPS Push	Starts to cause a branch		PAGE	Ignored (Recognized as zero step of n-page)	

[List of Instructions Not Described in this Chapter: Part 2]

The instructions listed below are intended for the Q series and not supported by the A series.

Some of them are explained in "Q Programming Practice Course".

Refer to "QCPU (Q mode) / QnACPU Programming Manual (Common Instructions)" for more details.

Instruction symbol (Name)	Functions	Drawing (devices to be used)	Instruction symbol (Name)	Functions	Drawing (devices to be used)
LDP Load P	Starts to operate rising pulse	 Designates bit for bit devices and word devices.	MEF	Converts the results into a falling pulse	 Designates bit device and word device
LDF Load F	Starts to start a falling pulse	 Designates bit for bit devices and word devices.	INV Inverse	Inverts the operation results	 Designates bit device and word device
ANDP And P	Series connection of rising pulse	 Designates bit for bit devices and word devices.	EGP Edge P	Converts the results into a rising pulse (Memorized by Vn)	 Designates bit device and word device
ANDF And F	Series connection of falling pulse	 Designates bit for bit devices and word devices.	EGF Edge F	Converts the results into a falling pulse (Memorized by Vn)	 Designates bit device and word device
ORP Or P	Parallel connection of rising pulse	 Designates bit for bit devices and word devices.	FF	Inverts a device output	 Designates bit device and word device
ORF Or F	Parallel connection of falling pulse	 Designates bit for bit devices and word devices.	DELTA Delta	Converts a direct output to a pulse	
MEP	Converts the operation results into a rising pulse	 Designates bit device and word device	DELTAP Delta P	Converts a direct output to a pulse	

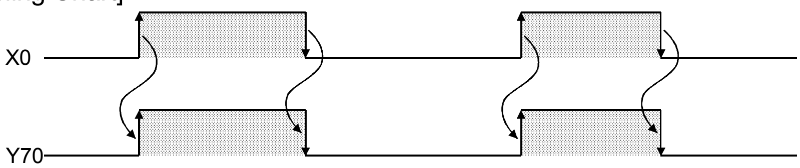
4.2 Differences between OUT and SET • RST

Path name	A:\SCHOOL
Project name	QB-1
Program name	MAIN

OUT Instruction

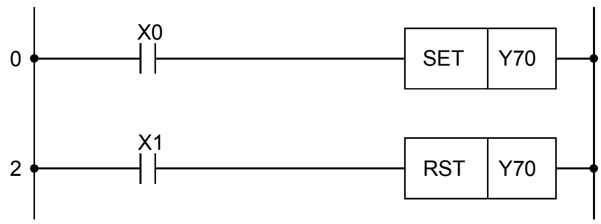
• The OUT instruction turns the specified device ON when receiving the input condition, and turns the device OFF when the condition becomes OFF.

[Timing Chart]



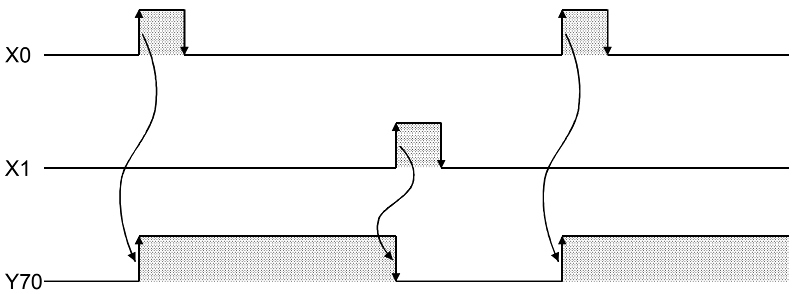
Path name	A:\SCHOOL
Project name	QB-2
Program name	MAIN

SET· RST instruction



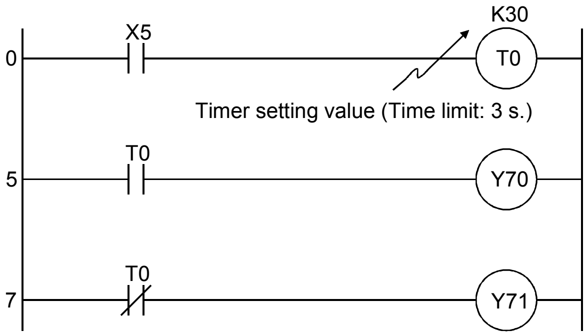
• The SET instruction turns the specified device ON when receiving the input condition, and maintains the device ON status even if the condition becomes OFF. To turn the device OFF, use the RST instruction.

[Timing Chart]



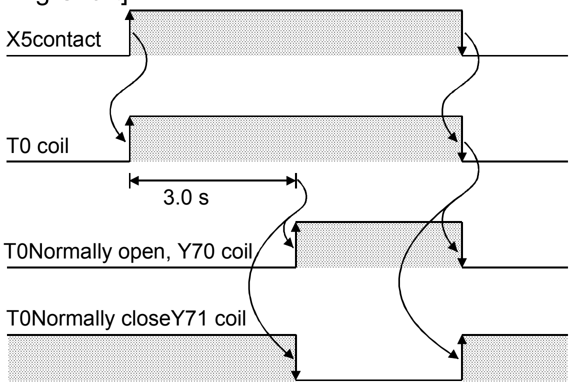
4.3 Measuring Timer

Path name	A:\SCHOOL
Project name	QB-3
Program name	MAIN



* OUT T is a 4-step instruction.

[Timing Chart]



- The timer contact operates when a given delay time elapses after the coil is energized. (Ondelay timer)
- The allowable range of timer setting is between K1 to K32767.
  - Low-speed (100 ms) timer 0.1 to 3276.7 s
  - High-speed (10 ms) timer 0.01 to 327.67 s
- When the timer setting value is set to zero, it turns ON (Time out) by the execution of the instruction.

The following four types of timer are available.

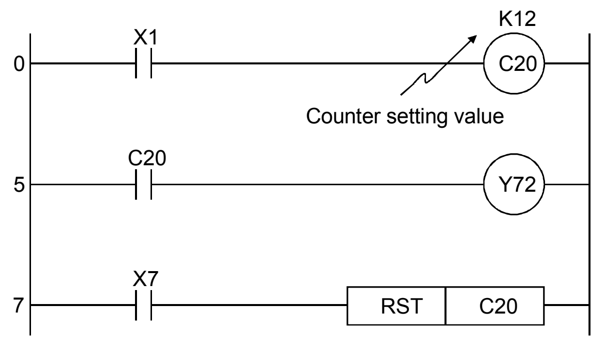
Type	Timer No. (Default)
Low-speed timer Counts time in increments of 100 ms.	T0 to T2047(2048)
High-speed timer Counts time in increments of 10 ms.	——
Low-speed retentive timer Accumulates time in increments of 100ms.	——
High-speed retentive timer Accumulates time in increments of 10ms.	——

- The number of each type of timer can be changed in units of 16 using parameters.
- Change the output instruction (OUT) to OUTH to select the high-speed timer or high-speed retentive timer.

Refer to Section 6.4 for explanation on the retentive timers.

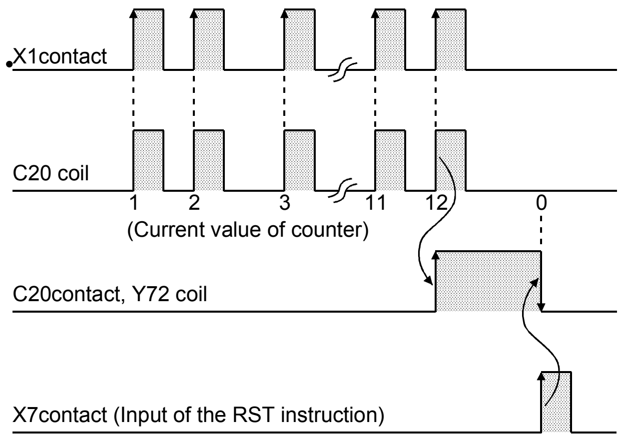
4.4 Counting by the Counter

Path name	A:\SCHOOL
Project name	QB-4
Program name	MAIN



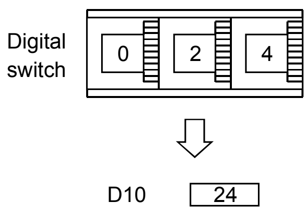
* OUT C is a 4-step instruction.

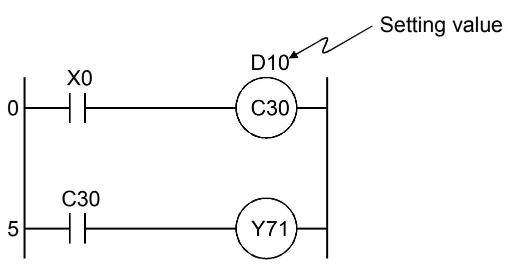
[Timing Chart]



- Counts when an input signal rises.
- After the count, the subsequent input signals are not counted.
- Once the counter counts, the contact status and the current counter value do not change until the RST instruction is performed.
- Perfoming the RST instruction before the count returns the counter to zero.
- The allowable range of the counter setting is between K0 and K32767. (K0 turns ON (Count up) by the execution of the instruction.)

In addition to the direct designation using K, indirect designation using D (Data register) is available.





- The counter C30 counts when the number of rising edges on the input signal X0 becomes the same as the number (e.g.; 24) specified by the data register D10.
- This indirect designation is useful for applying a value specified with an external digital switch to the counter.

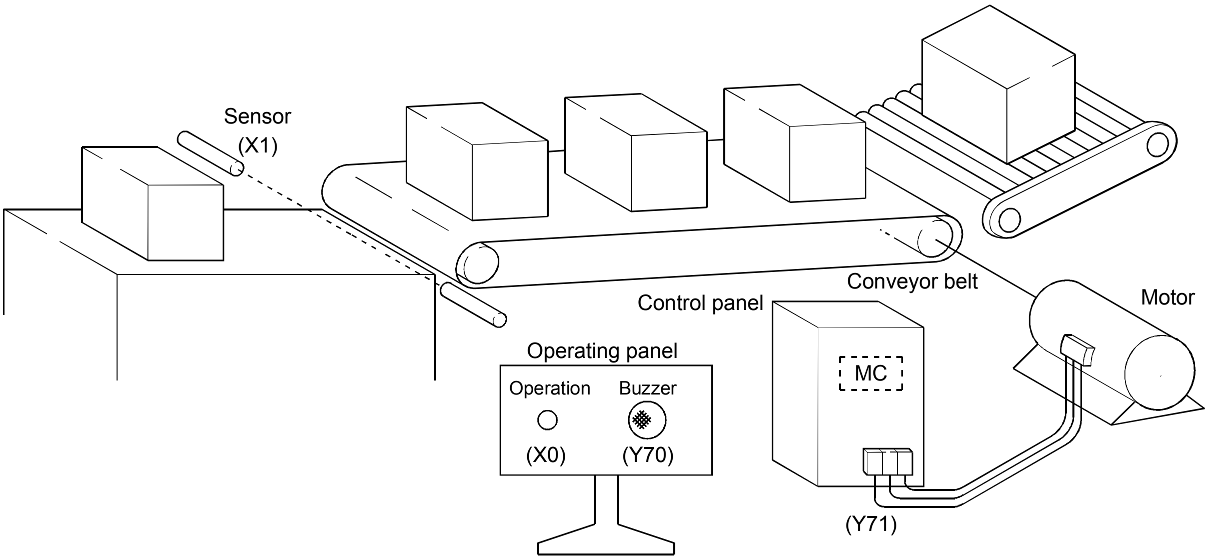
The indirect designation using data register D is also available for the timer.

Path name	A:\SCHOOL
Project name	QEX1
Program name	MAIN

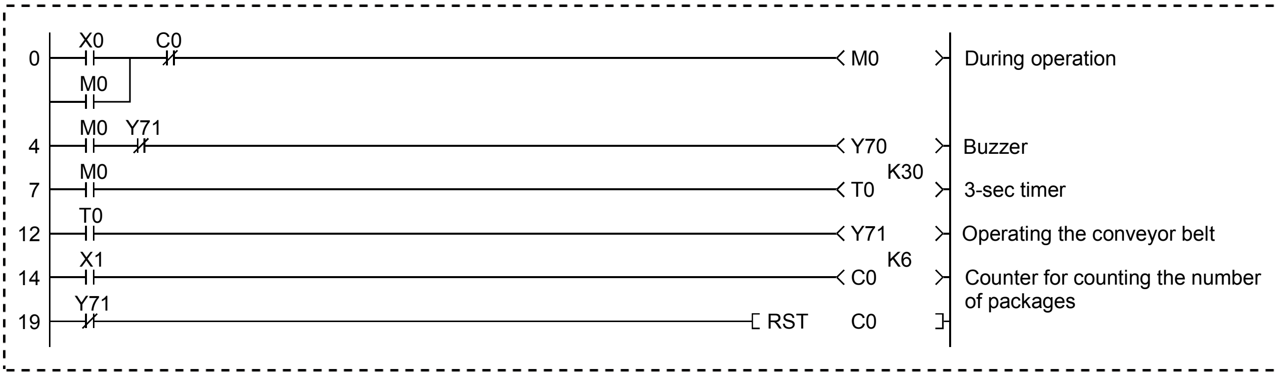
Ladder Example

When the conveyor belt operation start switch (X0) is turned ON, the buzzer (Y70) beeps for three seconds and then the conveyor belt (Y71) starts to operate.

The conveyor belt automatically stops when the sensor (X1) detects that six packages have passed through.



Create the following ladder and check if it works properly.

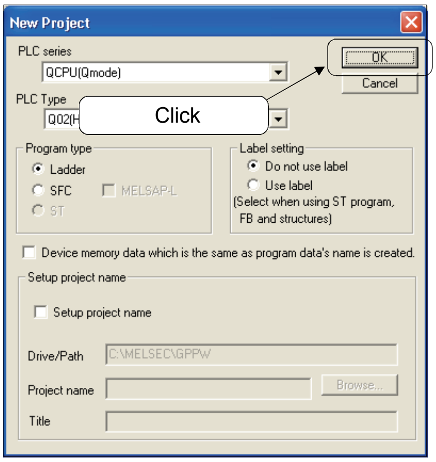


Operating Procedure

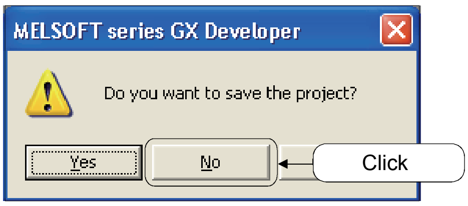
(1) Create a new project

1) Click on the tool bar.

2) The dialog box for creating a new project appears.
Confirm that the "PLC series" is QCPU and the "PLC type" is QO2H. Then click the OK button.



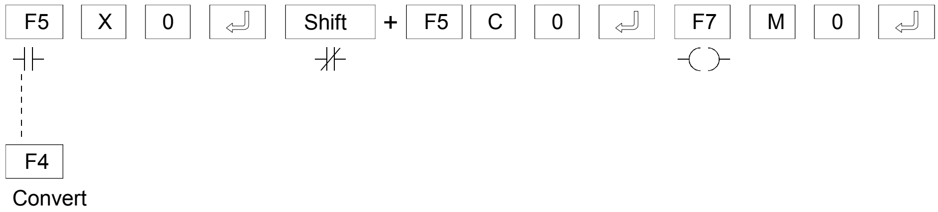
3) If the project in preparation exists, the dialog box appears asking if you want to save the product.
Click the No button.



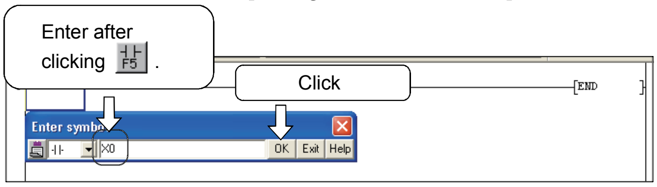
4) The screen shifts to the new project creation mode.

(2) Create a program

[Using the keyboard]

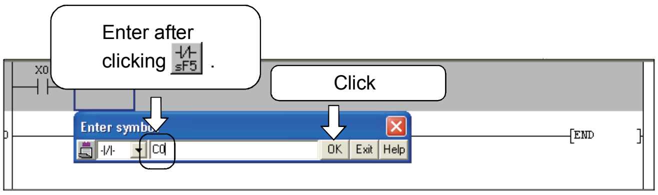


[Using the tool buttons]



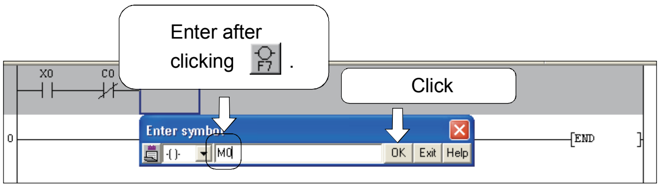
1) Click on the toolbar to open the ladder input window.

2) Enter "X0" with the keyboard and click the OK button.



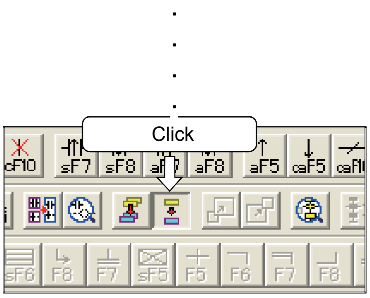
3) Click on the toolbar to open the ladder input window.

4) Enter "C0" with the keyboard and click the OK button.



5) Click on the toolbar to open the ladder input window.

6) Enter "M0" with the keyboard and click the OK button.



7) When creating the circuit is finished, click on the toolbar.

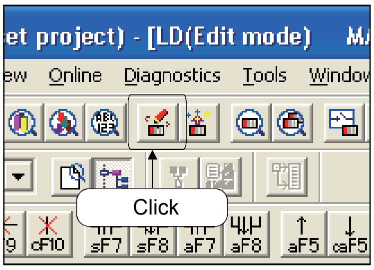
(3) Write to the PLC

As the demonstration machine has two CPUs, setting parameters for Multiple-CPU is required. (The setting is not necessary only for one CPU machine.)
Refer to Section 3.2 Parameters for the Multiple-CPU setting procedure.

1) Write the created ladder to the memory on the PLC.

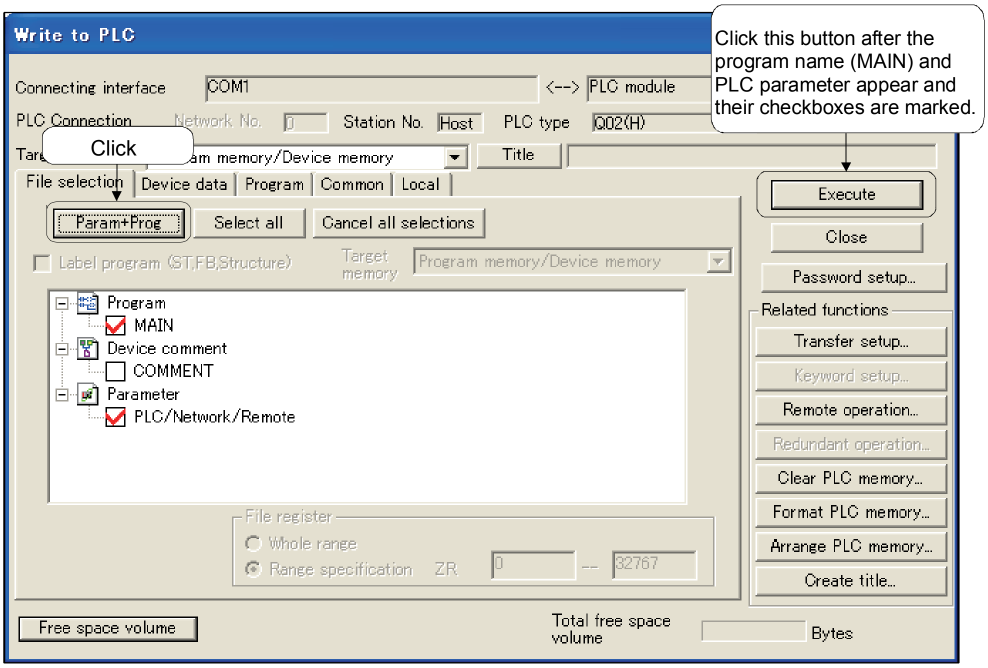
Set the RUN/STOP switch of the CPU to STOP

Click  on the tool bar.
The dialog box for writing to the PLC appears.



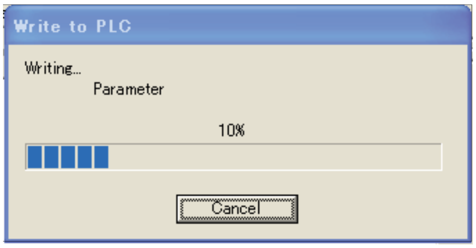
2) Click the Param+Prog button. Checkboxes for the program of the write data and the parameter displayed in the window are automatically marked.

3) Click the Execute button.

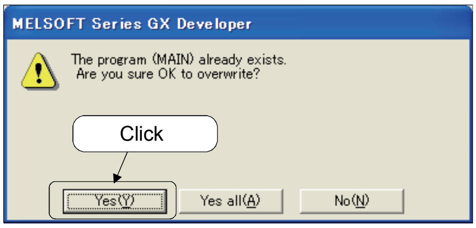


4) If the parameters have been already written, the dialog box appears asking if you want to overwrite them. Click the Yes button.

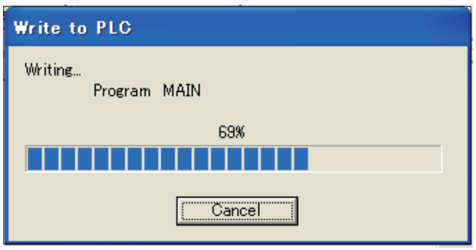
5) The progress bar dialog box appears to show how much the PLC parameter is written.



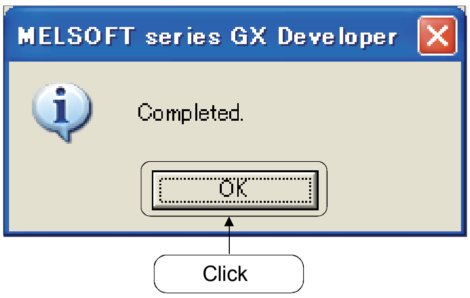
6) If the program has already been written, the dialog box appears asking if you want to overwrite it. Click the Yes button.



7) The progress bar dialog box appears to show how much the program MAIN is written.



8) Click the OK button.

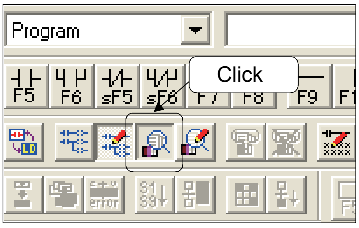


(4) Monitor the ladder

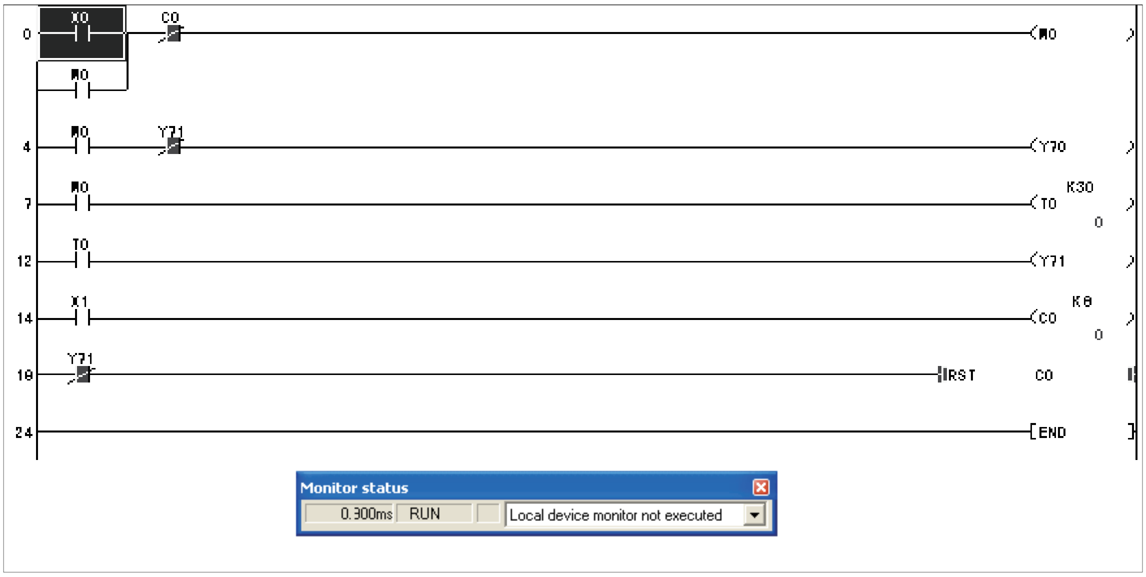
Reset the CPU

Set the RUN/STOP switch of the CPU to RUN

1) Click on the tool bar.



2) The ladder (write) screen is used to monitor the ladder.



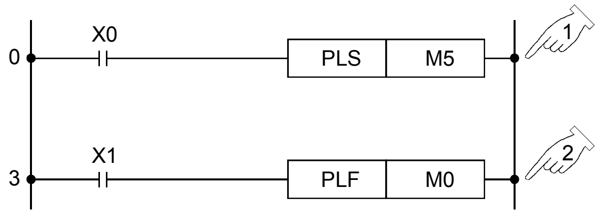
Operation Practice

1) Turn the push button switch (X0) ON. Y70 becomes ON, and T0 starts at the same time.
2) When the timer T0 counts three seconds (time is up), Y70 goes OFF and at the same time, Y71 becomes ON.
3) Turn the push button switch (X1) ON and OFF. The counter C0 counts the number of ON to turn Y71 OFF after counting ON for six times.

4.5 PLS Pulse (Turns the specified device ON for one scan at rising edge of an input condition.)

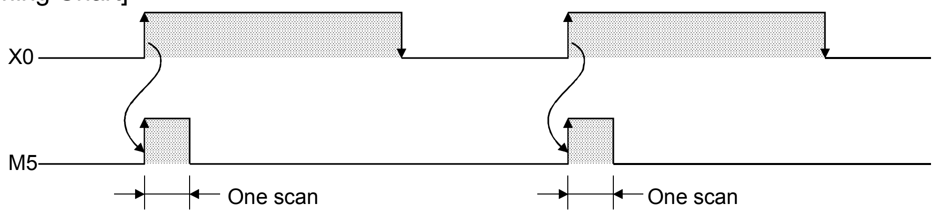
PLF Pulf (Turns the specified device ON for one scan at falling edge of an input condition.)

Path name	A:\SCHOOL
Project name	QB-5
Program name	MAIN



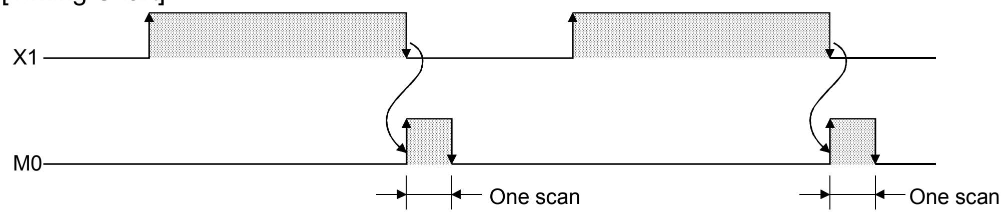
1 • The PLS instruction turns the specified device ON only for one scan at rising edge of the PLS command.

[Timing Chart]



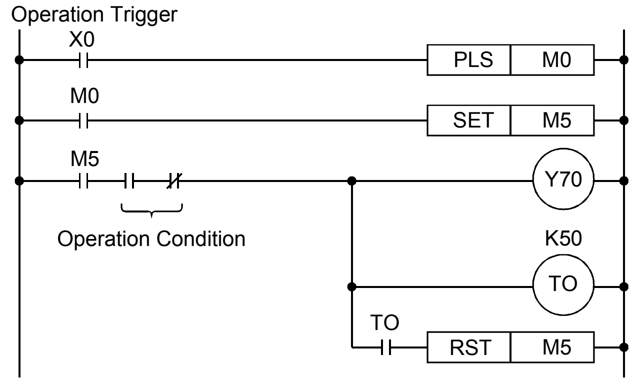
2 • The PLF instruction turns the specified device ON only for one scan at falling edge of the PLF command.

[Timing Chart]

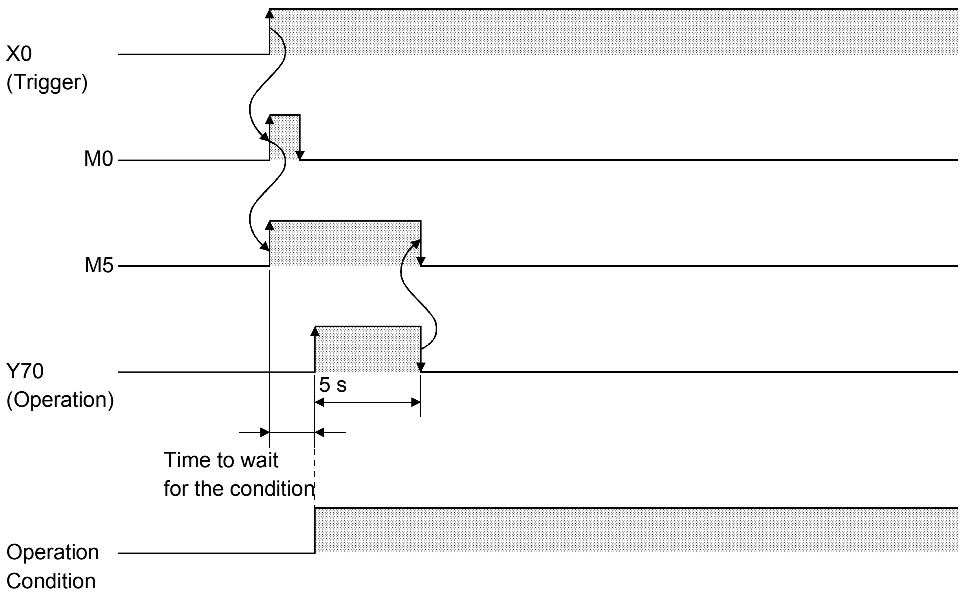


Application

• The instructions can be used in the standby program that waits for the operation condition.

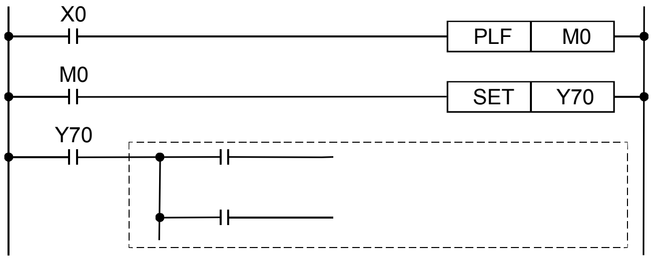


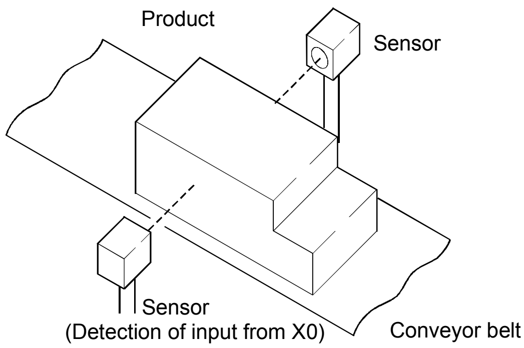
[Timing Chart]



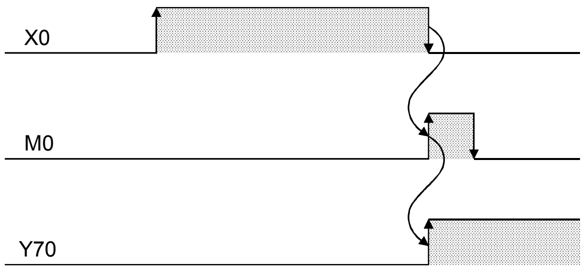
		Applicable Device												Digit	Basic number of steps
		Internal Device (system user)		File register	MELSECNET/10 (H) Direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level		
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
PLS Ⓓ PLF Ⓓ	Ⓓ	○	○	○	○	○	○								2

• The instructions can be used for the detection program that detects passage of moving objects.
After detecting the passage of a product, the next process for the product is started.





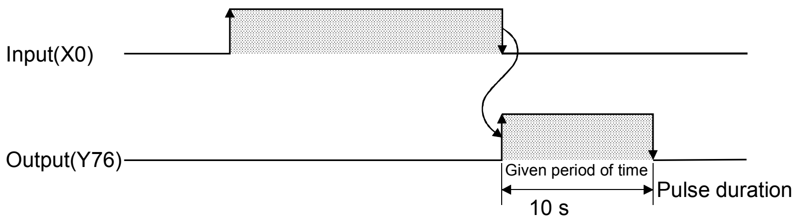
[Timing Chart]



Other Useful Ways of PLS & PLF Part 1

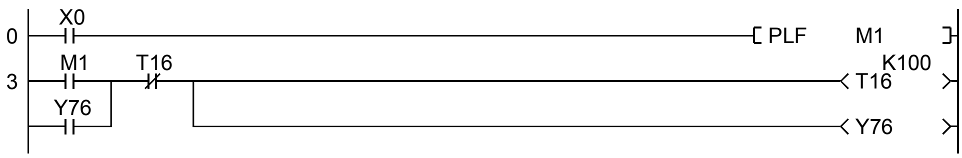
• They can be used to create the program that performs the output operation for a given period of time when the input signal changes ON to OFF.

[Timing Chart]



[Program example]

Path name	A:\SCHOOL
Project name	QB-6
Program name	MAIN

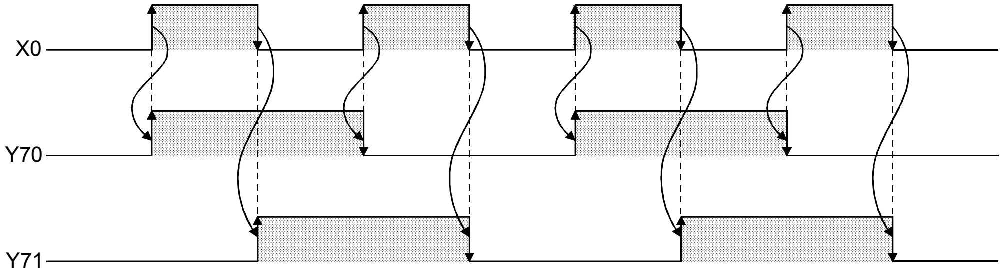


Other Useful Ways of PLS & PLF Part 2

- The program for the repeated operation such as switching ON/OFF status alternately by pressing the push button switch can be made using the instructions.

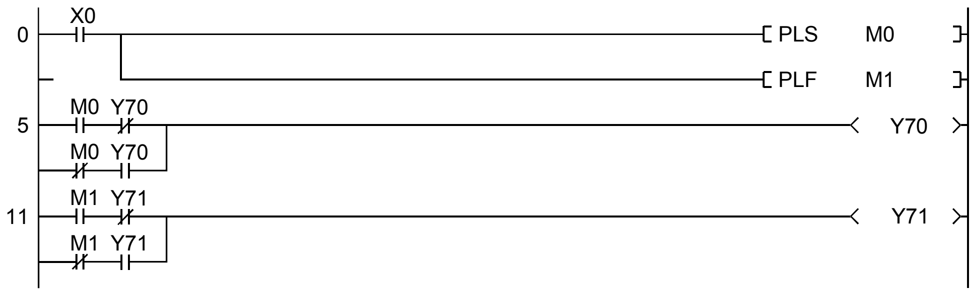
  If you use the PLS for the above program, the rising edge caused when the push button switch is pressed triggers the program. If you use the PLF, the falling edge caused when the switch is released becomes the trigger.

[Timing Chart]



[Program Example]

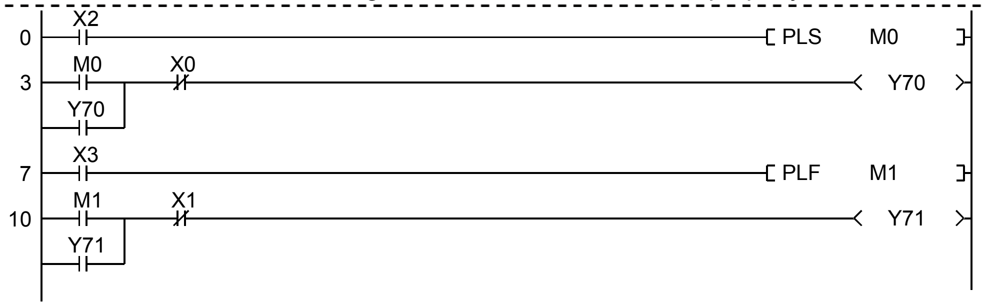
Path name	A:\SCHOOL
Project name	QB-7
Program name	MAIN



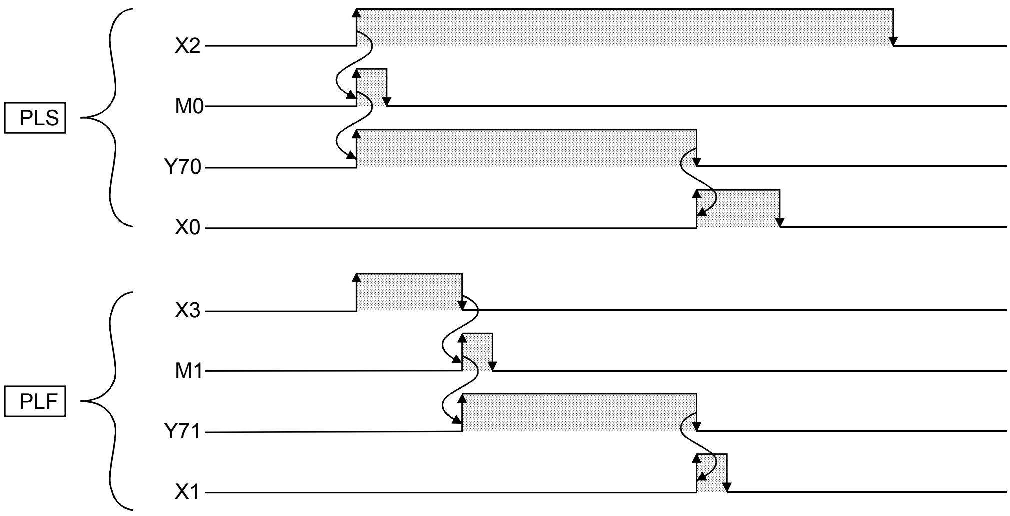
Path name	A:\SCHOOL
Project name	QEX2
Program name	MAIN

Ladder Example

Create the following ladder and check if it works properly.

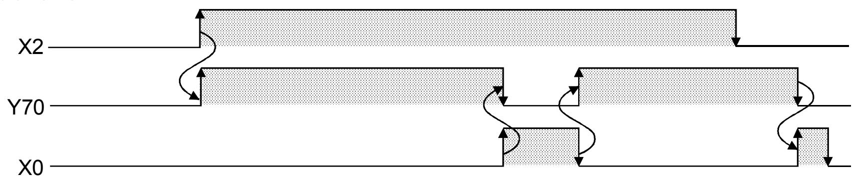


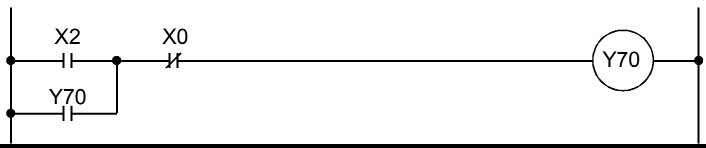
[Timing Chart]



REFERENCE

The following is a timing chart of a lockup ladder programmed using OUT instructions. Compare this with the lockup ladder created using the PLS instructions.





Operating Procedure

See Section 4.4 Operating Procedure for the procedure of the following operations.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

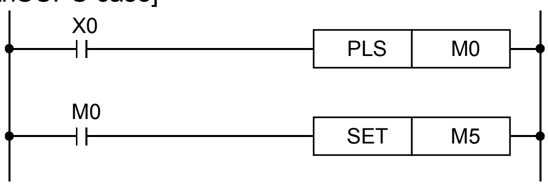
- Turning X2 switch ON makes Y70 turn ON, and Y70 switch OFF when X0 turns ON. (Even when X2 stays ON, Y70 is turned OFF when X0 turns ON.)
- Turning X3 switch OFF makes Y71 turn ON, and Y71 switch OFF when X1 turns ON.

Related Exercise — Exercise 3

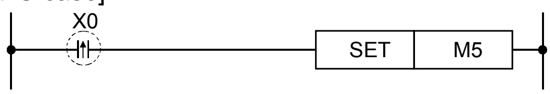
REMARK

Input pulse processing is not required for the QCPU as it uses a derivative contact. (|↑|/|↓|).

[A/AnSCPU case]



[QCPU case]

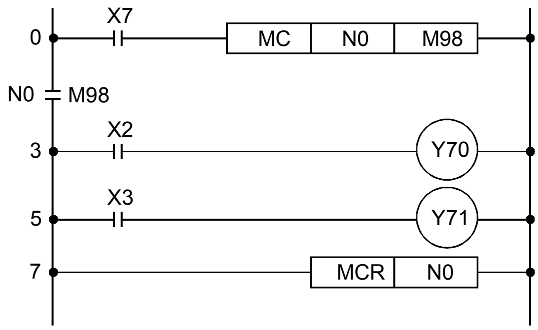


Supported instructions are; LDP, LDF, ANDP, ANDF, ORP, and ORF.

4.6 MC Master control (Start)

MCR Master control reset (End)

Path name	A:\SCHOOL
Project name	QB-8
Program name	MAIN

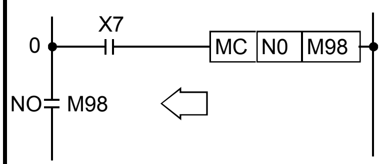


*N0 ≑M98 is displayed in the read mode.

- The above program is a basic one.
- MC N□ M□ to MCR N□ (indicated as "MC to MCR" hereafter.)
  The allowable nesting (N) numbers for the MC to MCR are between N0 and N14.
- The scan time skipped by the "MC to MCR" hardly changes.

The status of the devices which exist in the skipped area becomes as follows;
All the devices are turned OFF by the OUT instruction.
The SET, RST and SFT instructions make no change, and values of the counter and retentive timer remain unchanged.
The 100ms timer and 10ms timer are reset to zero.

Reference



You do not need to write contacts for the master control when creating the ladder.
After creating the ladder, convert it (F4) and enter into the read mode. The contacts are automatically inserted.

Application

- The instructions can be used to create the program for switching between manual and automatic operations. (Refer to the Ladder Example.)

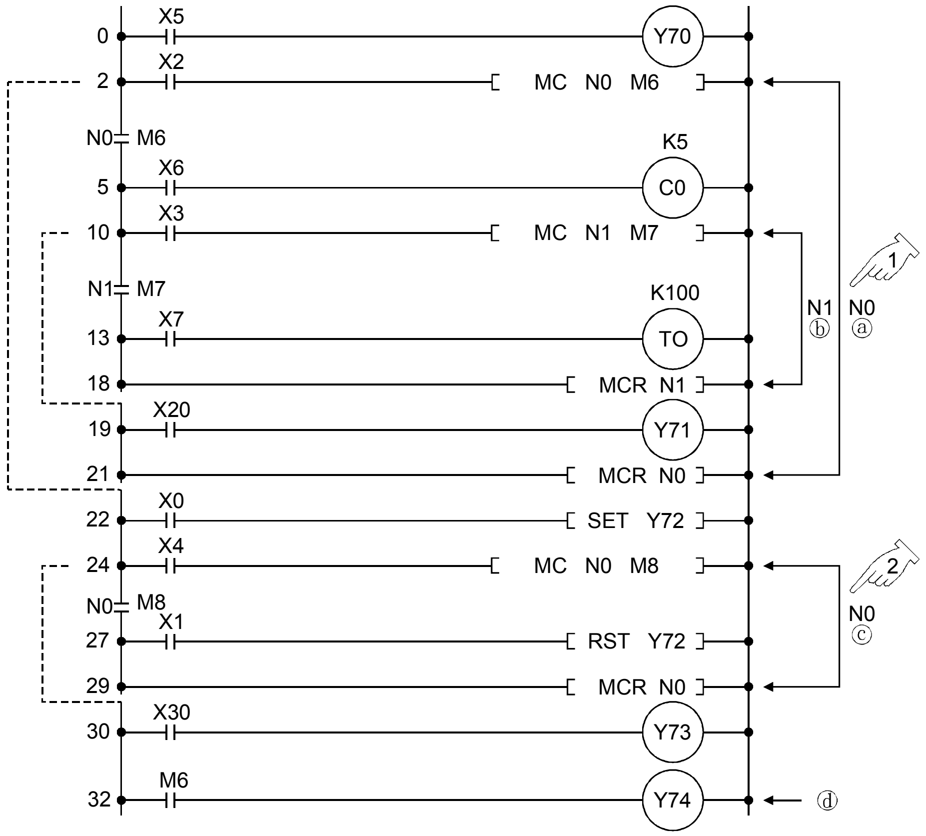
		Applicable Device: Internal Device (system user) Bit	Internal Device (system user) Word	File register R	MELSECNET/10 (H) Direct Jn\ Bit	MELSECNET/10 (H) Direct Jn\ Word	Intelligent function module Un\G	Index register Z	Constant K	Constant H	Pointer P	Pointer I	Level N	Digit	Basic number of steps
MC n Ⓓ	n												○		2
MCR n	Ⓓ	○	○	○	○	○	○								1

The basic number of steps of the MC instruction is two, and that of the MCR instruction is one.

Nested "MC to MCR" Program Example

• The MC and MCR instructions can be nested as shown below.

Path name	A:\SCHOOL
Project name	QB-9
Program name	MAIN



1 • The "MC to MCR" program ⓑ is nested under the "MC to MCR" program ⓐ. (It is called "nested structure".)
To do that;

1) Assign the nesting number (N) of MC instructions in ascending order.
2) For the MCR nesting number (N), assign the numbers used for the MC in descending order.

2 • The "MC to MCR" program ⓒ is independent from the ⓐ program. The nesting numbers (N) used in the ⓐ program can be used for the ⓒ program.
• The internal relay number (M) must be changed by MC.

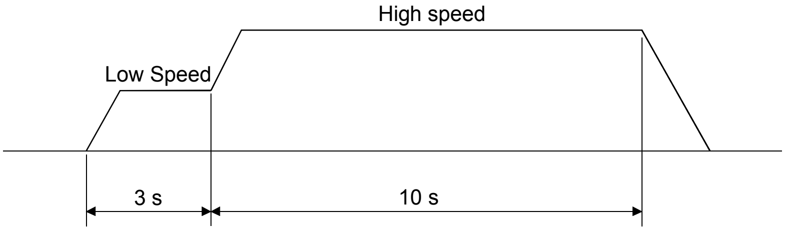
3 • As shown in the ⓓ program, the internal relay number M□ of MC can be used as a contact.

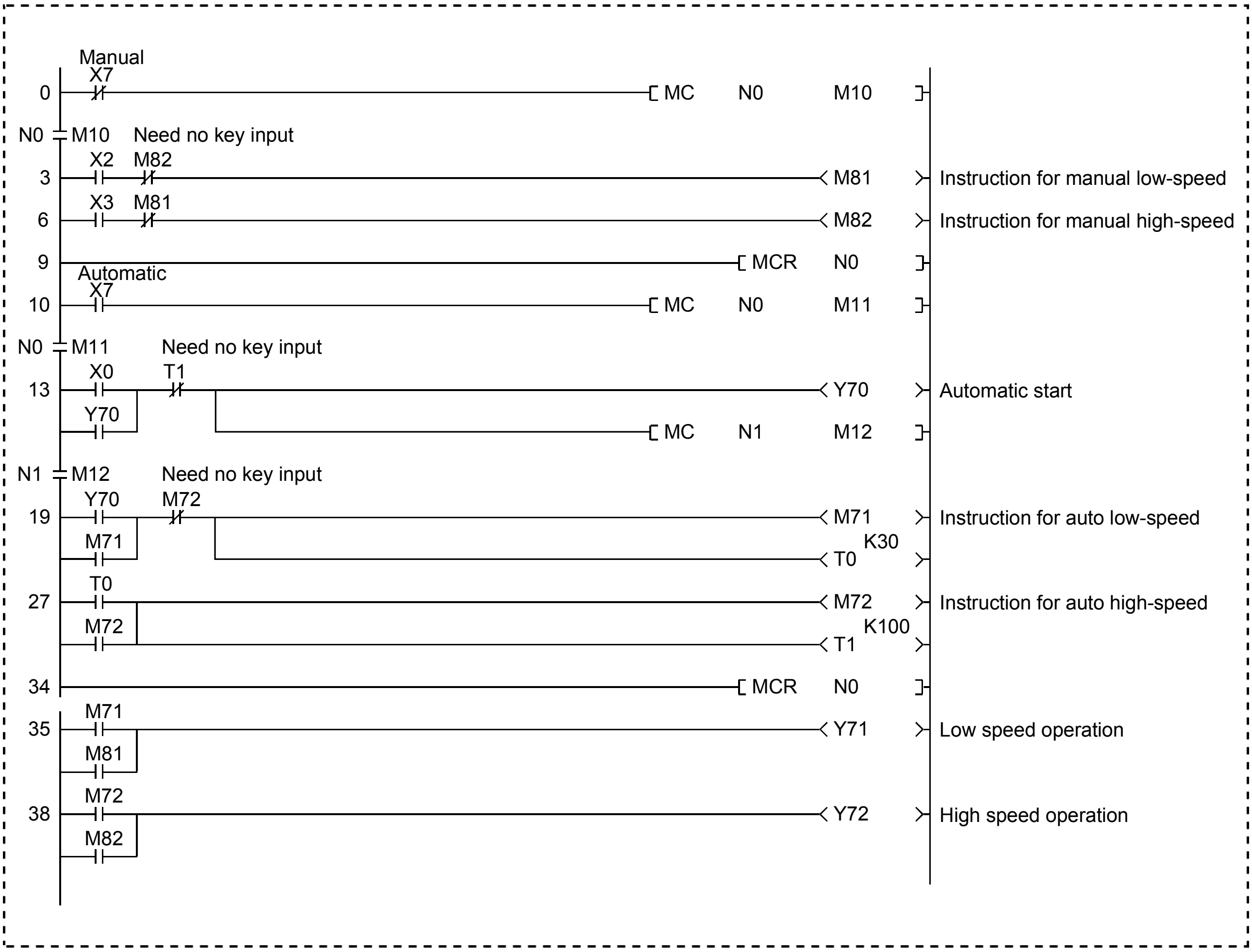
Path name	A:\SCHOOL
Project name	QEX3
Program name	MAIN

Ladder Example

A program for switching between manual and automatic operations can be made using the MC and MCR instructions.

- When selecting manual operation by turning X7 OFF
  1) The system goes into low-speed operation mode when X2 is turned ON.
  2) The system goes into high-speed mode when X3 is turned ON.
- When selecting automatic operation by turning X7 ON, the system operates in low-speed mode for 3 s after X0 is turned ON. Then it operates in high-speed mode for 10 s and stops.





Operating Procedure

See Section 4.4 Operating Procedure for the detailed procedure of the following operations.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

- The manual operation is selected by turning the X7 switch OFF.
  When the X2 switch is turned ON, Y71 lights and the low-speed operation is performed. To perform the high-speed operation, turn the X3 switch ON. Y72 lights and the high-speed operation starts.
- Automatic operation is selected by turning the X7 switch ON.
  When the X0 switch is turned ON, Y70 lights and indicates that it being automatically operated. Y71 also lights at the same instant and stays on for three seconds indicating the system is in low-speed mode. After the three seconds elapsed, Y72 lights and stays on for 10 s indicating in high-speed mode. Then the operation is stopped. (Y70, Y71, and Y72 have stopped lighting at the end.)

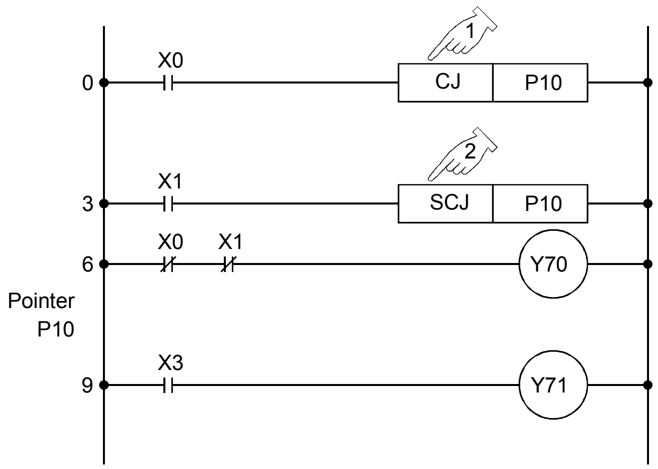
NOTE
For the MCR instructions in one nested program block, all master controls in the program can be terminated with the lowest nesting (N) number only.

4.7 CJ • SCJ • CALL • RET • FEND

Path name	A:\SCHOOL
Project name	QB-10
Program name	MAIN

4.7.1 CJ (Conditional jump: instataneous execution condition jump)

SCJ (S conditional jump: execution condition jump after one scan)

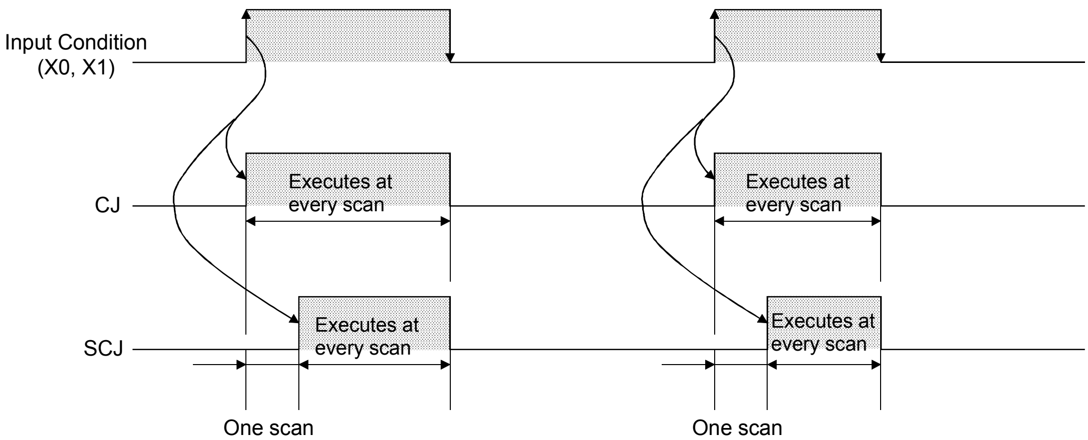


1 • The CJ instruction instantaneously executes a program with jumping it to the designated jump address (pointer number) when the input condition is ON.
When the condition is OFF, the program is not jumped.

2 • The SCJ instruction executes a program without jumping for the scan when the input condition is ON. From next scan, the instruction executes a program with jumping it to the designated jump address (pointer number).
When the input condition is OFF, the program is not executed.

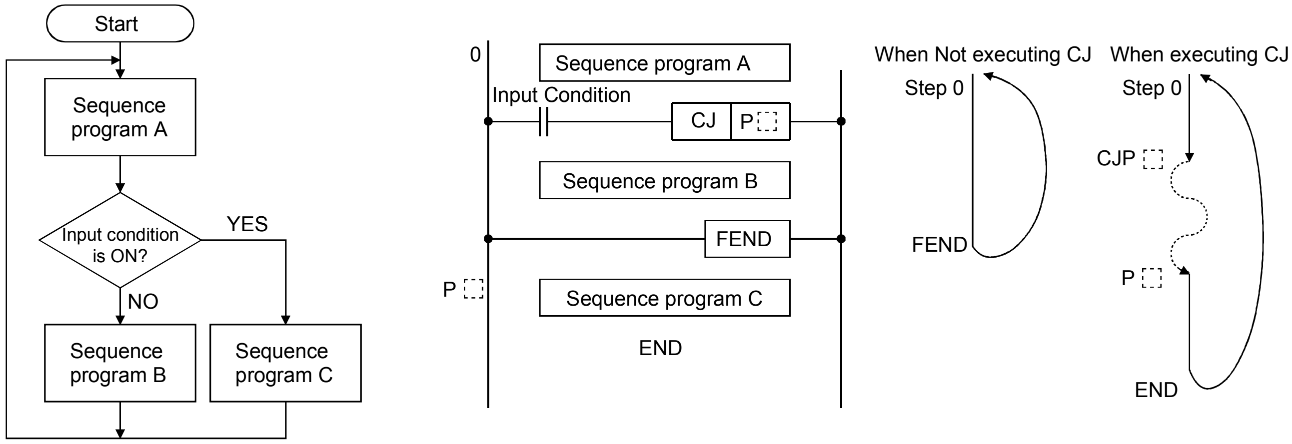
• The SCJ instruction is used when some operations must be executed before jumping the program.
For example, when the output needs to be ON or reset in advance.

[Timing Chart]

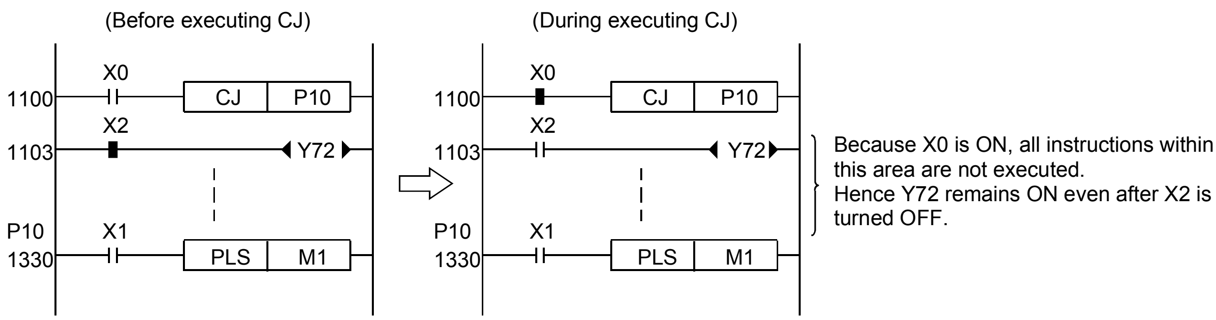


Caution

- The pointer numbers available for both CJ and SCJ are P0 to P4095.
- Use the FEND instruction as shown below when a program using the CJ and SCJ must be concluded in each program block. (See Section 4.7.3 for explanation on the FEND.)



- The status of ladders skipped by the CJ instruction remains unchanged.



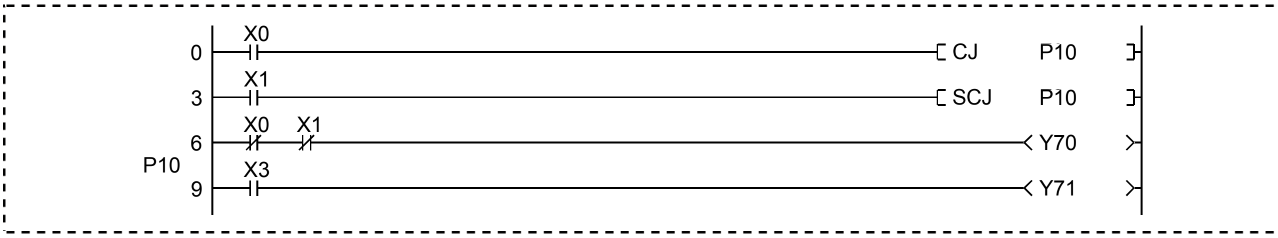
- Once the coil of the timer is energized, the timer updates even if the coil is skipped by the CJ or SCJ instruction.
  When time is up, the contact becomes ON. This is because the timer updates and turns the contact ON/OFF after each END.

		Applicable Device												Digit	Basic number of steps
		Internal Device (system user)		File register	MELSECNET/10 (H) Direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level		
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
CJ P** SCJ P**	P										○				2

Path name	A:\SCHOOL
Project name	QEX4
Program name	MAIN

Ladder Example

Create the following ladder with GX Developer and write it on the CPU of the demonstration machine. Then check the difference between CJ and SCJ instructions.



Operating Procedure

See Section 4.4 Operating Procedure for the detailed procedure of the following operations.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

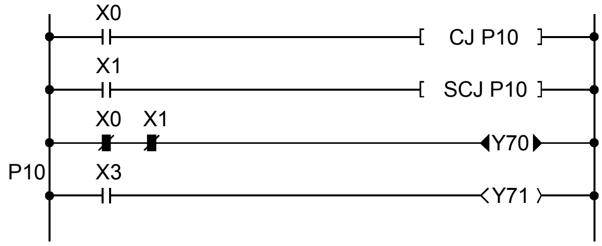
(4) Monitor the ladder

Operation Practice

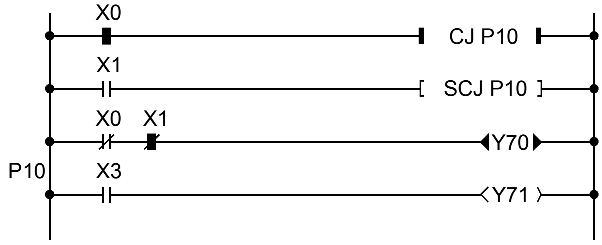
(1) When X0 and X1 are OFF, the CJ and SCJ instructions are not executed. Therefore, Y70 is ON.

(2) When X0 is turned ON, the CJ instruction is executed jumping to P10. Therefore, Y70 remains ON.

[Before executing CJ and SCJ]

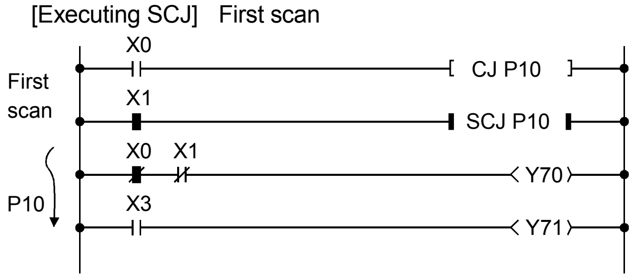


[Executing CJ] First scan and subsequent scans

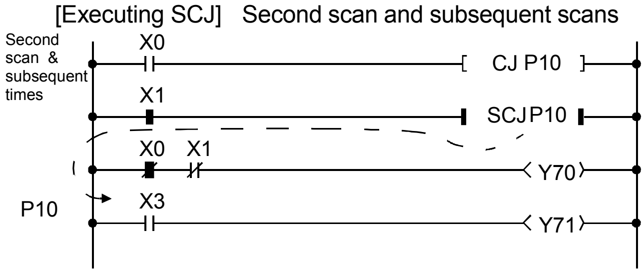


(3) The SCJ instruction is executed when X0 is turned OFF and X1 is turned ON.
The instruction is executed jumping to P10 from the next scan.
Therefore, Y70 becomes OFF.

[Executing SCJ] First scan



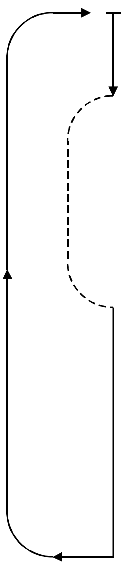
[Executing SCJ] Second scan and subsequent scans



(4) Y71 is turned ON/OFF by X3 regardless of the CJ or SCJ instructions.

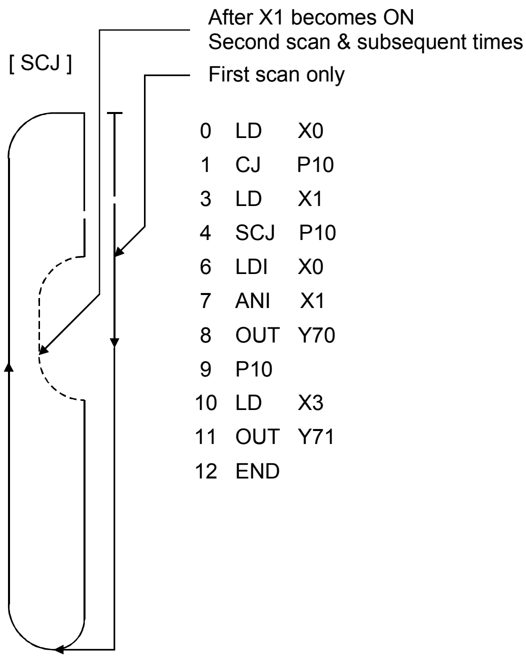
- The following lists explain the difference between the CJ and SCJ instructions.

[CJ]



```
0  LD   X0
1  CJ   P10
3  LD   X1
4  SCJ  P10
6  LDI  X0
7  ANI  X1
8  OUT  Y70
9  P10
10 LD   X3
11 OUT  Y71
12 END
```

[SCJ]

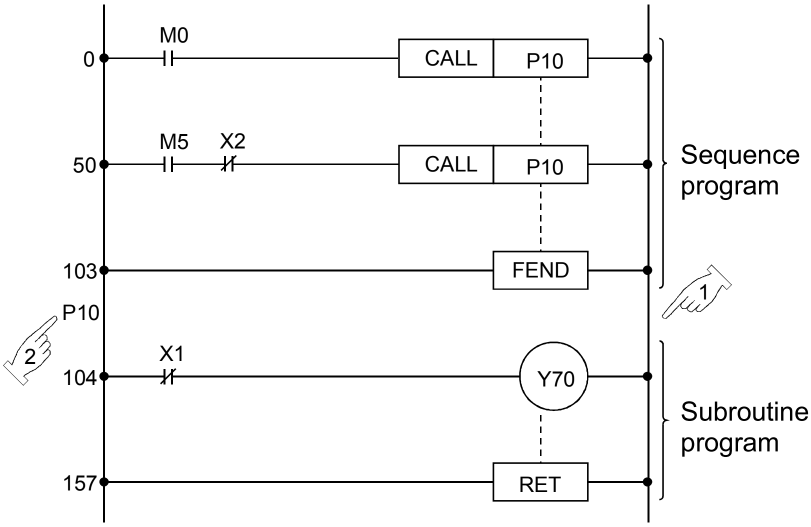



```
0  LD   X0
1  CJ   P10
3  LD   X1
4  SCJ  P10
6  LDI  X0
7  ANI  X1
8  OUT  Y70
9  P10
10 LD   X3
11 OUT  Y71
12 END
```

Related Practice Question —— Practice Question 4

4.7.2 CALL (P) Call
RET Return

Executes a subroutine program

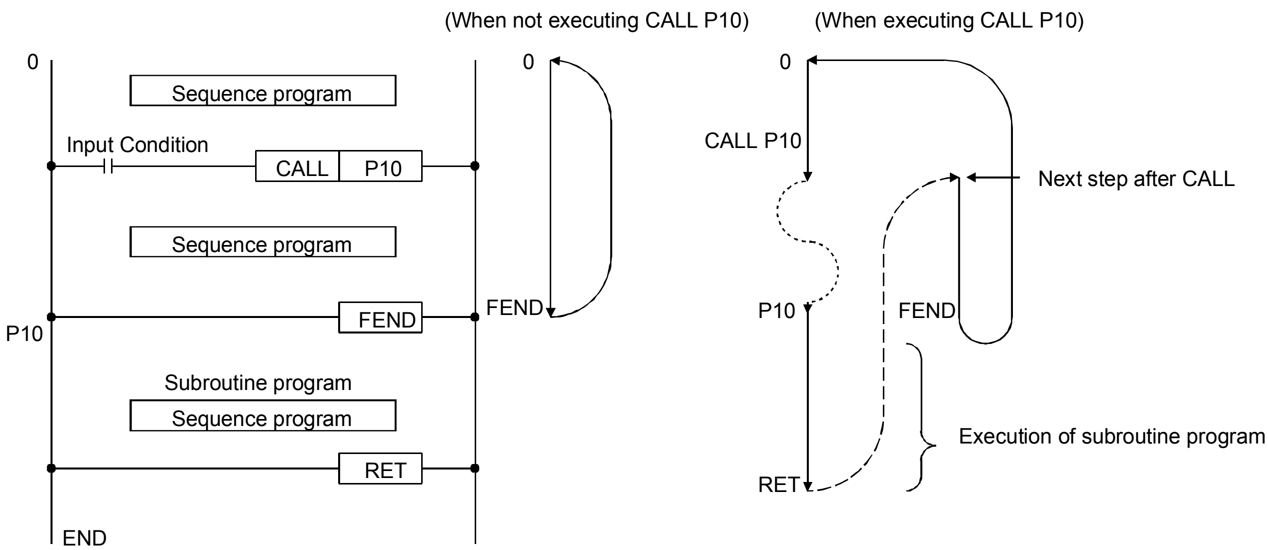


- The above program is a basic style to execute the subroutine program using the CALL and RET instructions.
  Keep this structure, otherwise an error occurs and PLC stops.

1 A subroutine program consists of the ladders for executing the same data many times in one program.
The subroutine starts at Pointer P □ and ends with the RET instruction.

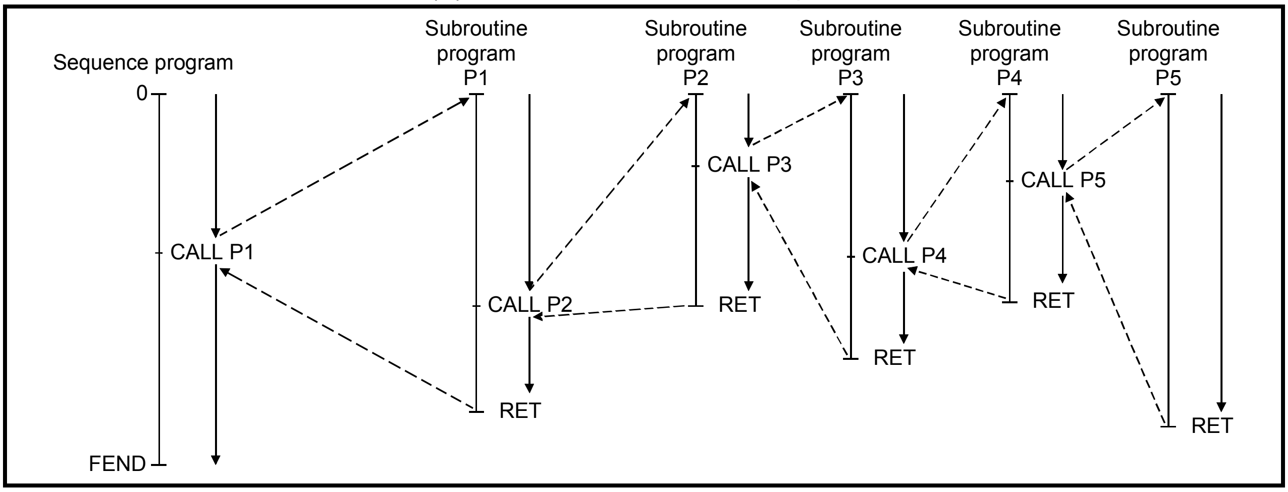
2 • 0 to 4095 can be used as the Pointer P number. (Same as the pointer numbers used for the CJ and SCJ instructions.)

- The subroutine program is executed as shown in the following diagrams.

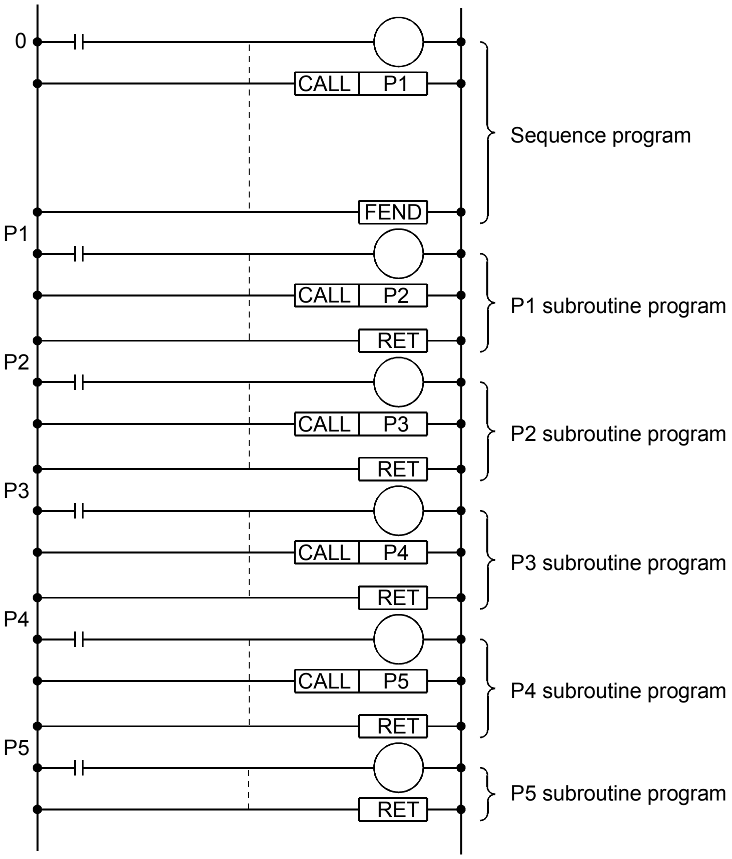


About Nesting

• The CALL (P) instructions can be nested up to 16 levels.



The following ladder circuit shows the above nested program.



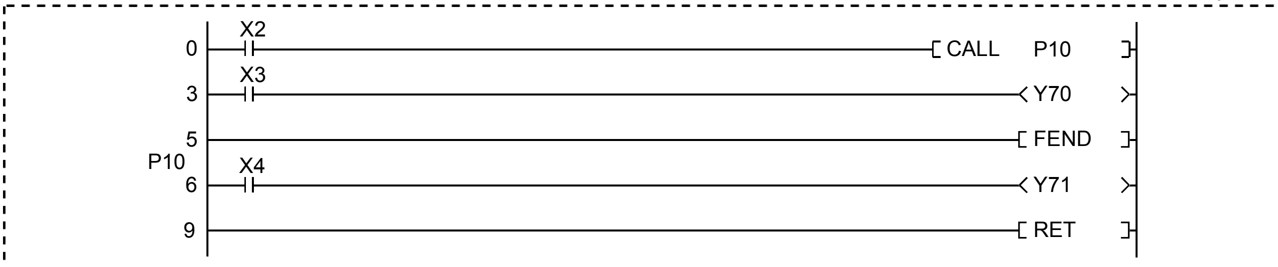
		Applicable Device: Internal Device (system user)		File register	MELSECNET/10 (H) Direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level	Digit	Basic number of steps
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
CALL(P) P**	P										○				2
RET															1

The basic number of steps of CALL (P) is 2 tn, and that of RET is one. ("n" is a argument passed to the subroutine.)

Path name	A:\SCHOOL
Project name	QEX5
Program name	MAIN

Ladder Example

Create the following ladder with the GX Developer and write it on the CPU of the demonstration machine to check if the CALL and RET instructions work properly.



Operating Procedure

Refer to Section 4.4 Operating Procedure for the detailed procedure of the following operations.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

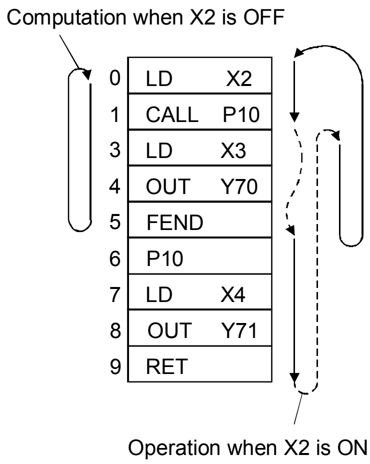
Verify the operation of the ladder, which was created with the GX Developer and written to the CPU of the demonstration machine, by monitoring the ladder on the screen.

(1) When X2 is OFF

1) Operates from 0 to FEND.
2) Y70 turns ON/OFF when turning X3 ON/OFF.
3) Y71 remains unchanged even when turning X4 ON/OFF.

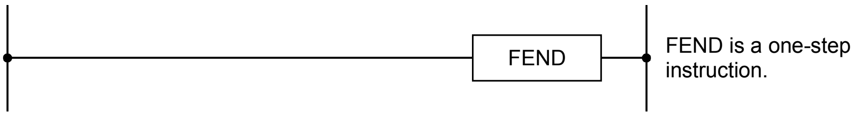
(2) When X2 is ON

1) After executing P10 subroutine, operate from Step 4 to FEND.
2) Y70 turns ON/OFF when turning X3 ON/OFF.
3) Y71 turns ON/OFF when turning X4 ON/OFF.

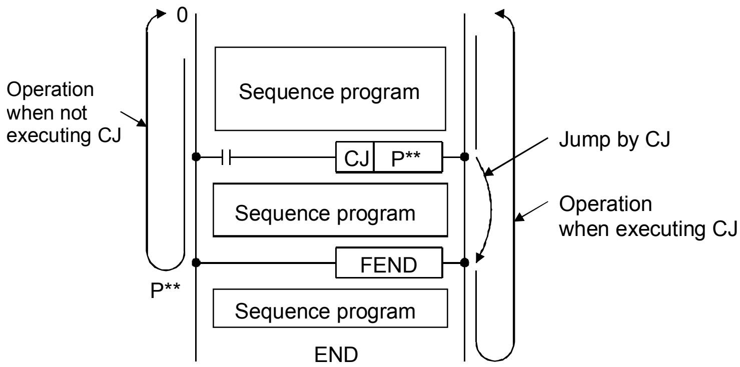


Related Practice Question —— Practice Question 4

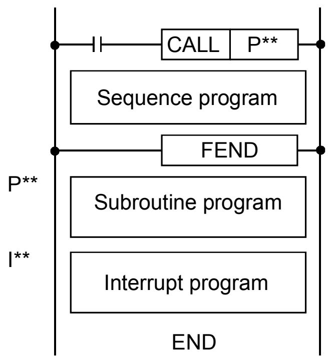
4.7.3 FEND F end



- Use the FEND instruction as the END instruction under the following conditions:
  1) When a sequence program must be executed and terminated in each program block.
     For example, use this instruction with CJ and SCJ instructions.
  2) When using the subroutine programs (CALL and RET instructions).
  3) When using an interrupt program.
- After each execution of the FEND, the PLC processes the current value of the timer and counter and makes self-diagnostic check, and then re-operates from 0 step.



(a) When operating in each program block by CJ instruction



(b) When using the subroutine and interrupt programs

Caution

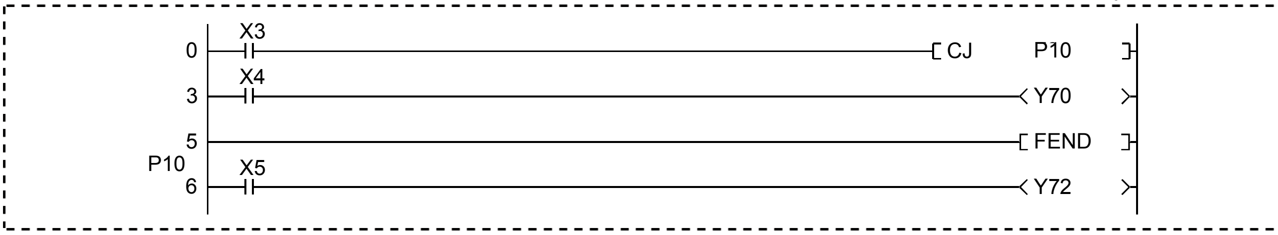
- There is no limit to the number of FEND instructions in a sequence program, however, it cannot be used in the subroutine and interrupt programs.
- The FEND instruction cannot be used to terminate the main or sub sequence program.
  Make sure to use an END instruction for the end of a whole program.

REFERENCE
The interrupt program allows you to stop the current process and processes an interrupt upon receiving an interrupt request in the middle of processing a normal program.

Path name	A:\SCHOOL
Project name	QEX6
Program name	MAIN

Ladder Example

Create the following ladder with the GX Developer and write it to the CPU of the demonstration machine to check if the FEND instruction work properly.



Operating Procedure

Refer to Section 4.4 Operating Procedure for the detailed procedure of the following operations.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

Verify the operation of the ladder, which was created with the GX Developer and written to the CPU of the demonstration machine, by monitoring the ladder on the screen.

(1) When X3 is OFF

1) Operates from 0 to FEND.

2) Y70 turns ON/OFF when turning X4 ON/OFF.

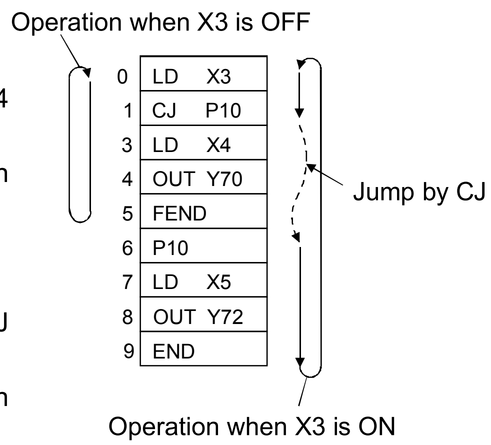
3) Y72 remains unchanged even when turning X5 ON/OFF.

(2) When X3 is ON

1) Jumps to P10 pointer by the CJ instruction.

2) Y70 remains unchanged when turning X4 ON/OFF.

3) Y72 turns ON/OFF when turning X5 ON/OFF.



Related Practice Question —— Practice Question 4

Path name	A:\SCHOOL
Project name	QTEST1
Program name	MAIN

4.8 Practice Questions

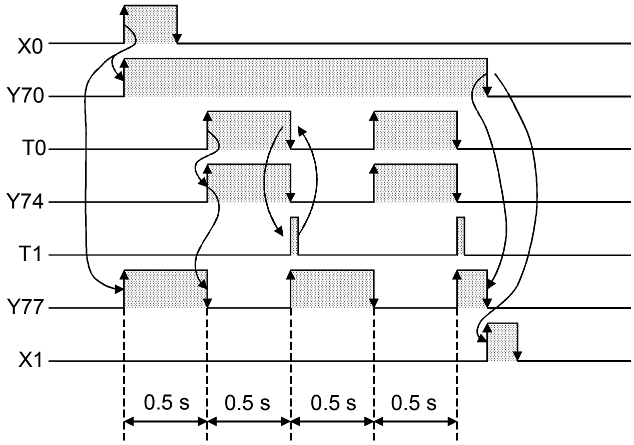
4.8.1 Practice Question (1)

LD to NOP

When X0 turns ON, Y70 is self-maintained, and Y74 and Y77 flicker alternately every 0.5 s.

When X1 turns ON, Y70 turns OFF and flickering of Y74 and Y77 also stops.

[Timing Chart]



Create the following program with the GX Developer filling in the blanks , and verify the operation using the demonstration machine.

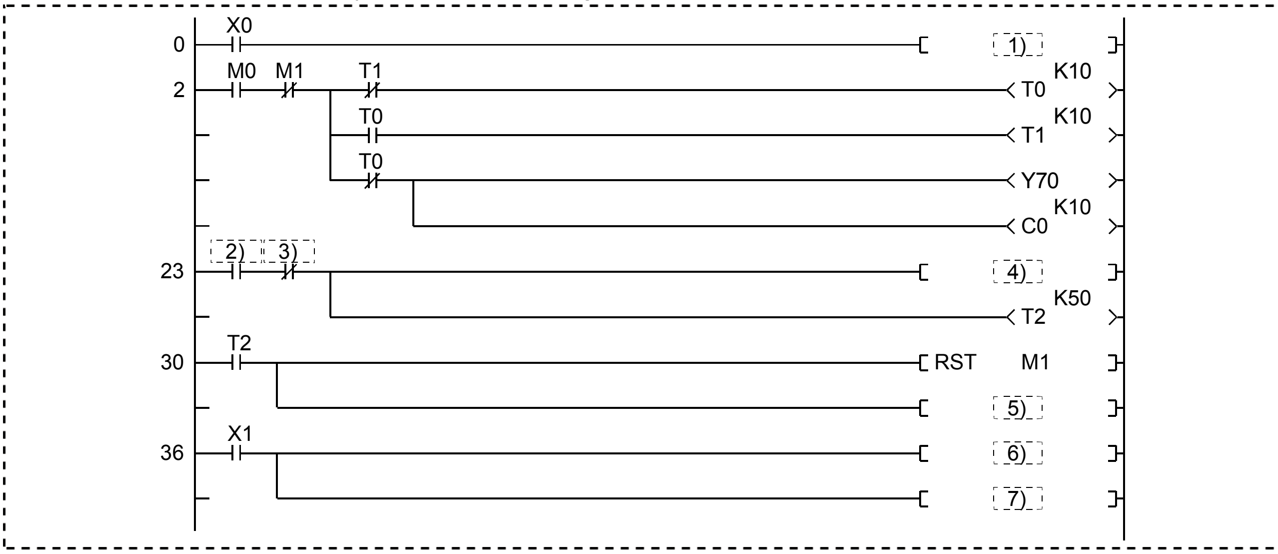
Path name	A:\SCHOOL
Project name	QTEST2
Program name	MAIN

4.8.2 Practice Question (2)

SET, RST

When turning X0 ON, Y70 starts to flicker at one-second intervals and stops the flickering for five seconds after flickering 10 times, then restart flickering.
The flickering of Y70 can be stopped by turning X1 ON.

Create the following program with the GX Developer filling in the blanks ☐, and verify the operation using the demonstration machine.



1) ______________________

2) ______________________

3) ______________________

4) ______________________

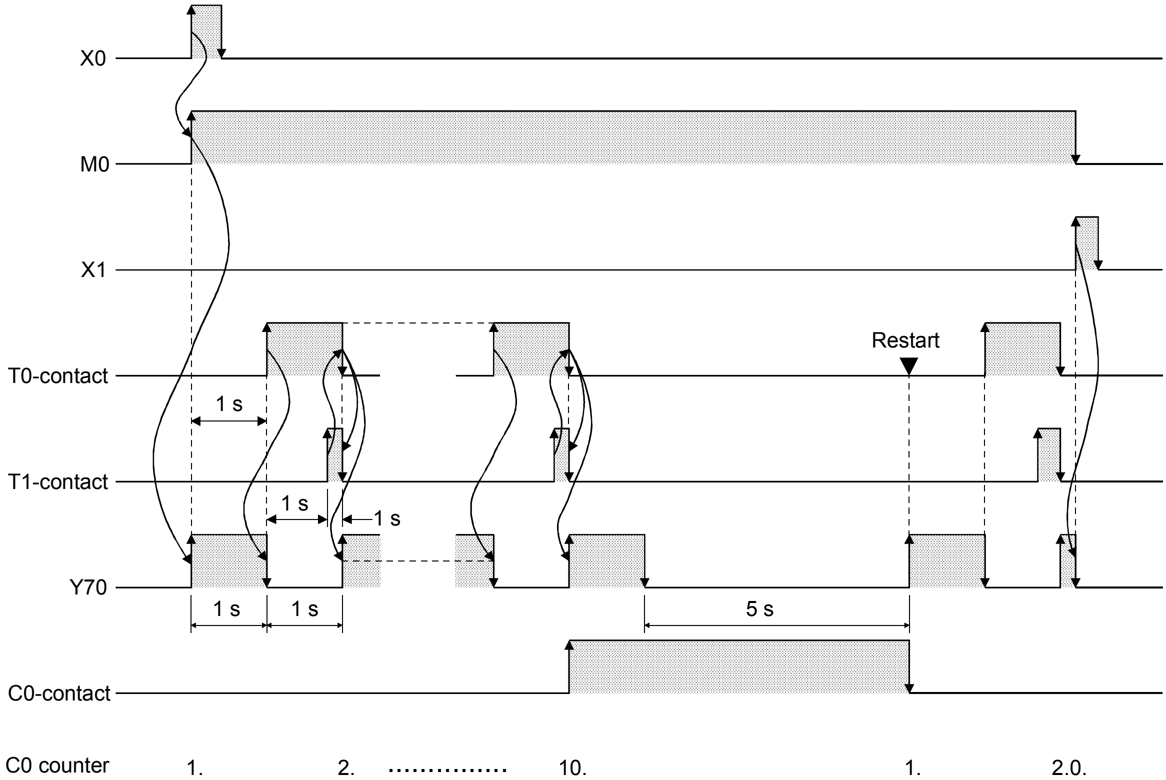
5) ______________________

6) ______________________

7) ______________________

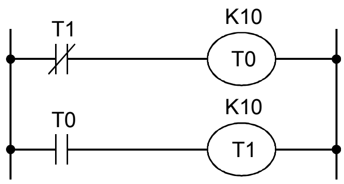
Hint

(1) The following shows the timing chart of the program.

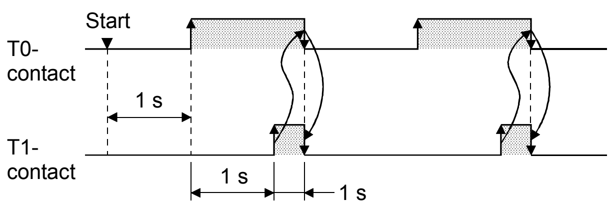


(2) The following shows the basic flickering ladder and its timing chart.

[Ladder]

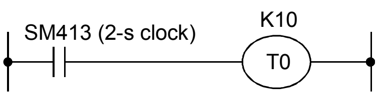


[Timing Chart]

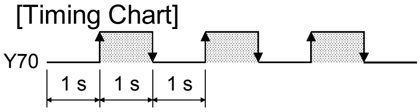


REFERENCE

The flickering ladder can be created using the special relay that generates clock as shown below.



[Timing Chart]



Other than the SM413 (2-s clock), the following can be used.

SM409 (0.01-s clock)
SM410 (0.1-s clock)
SM411 (0.2-s clock)
SM412 (1-s clock)
SM414 (2n-s clock)
SM415 (2-nms clock)

Starts from OFF when the PLC is reset or the power is turned ON.

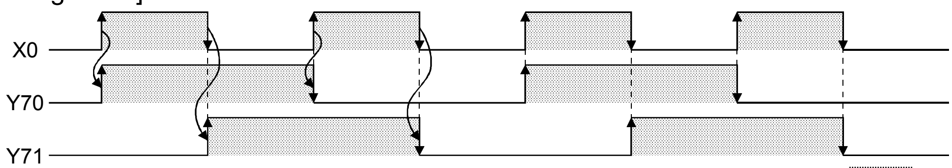
Path name	A:\SCHOOL
Project name	QTEST3
Program name	MAIN

4.8.3 Practice Question (3)

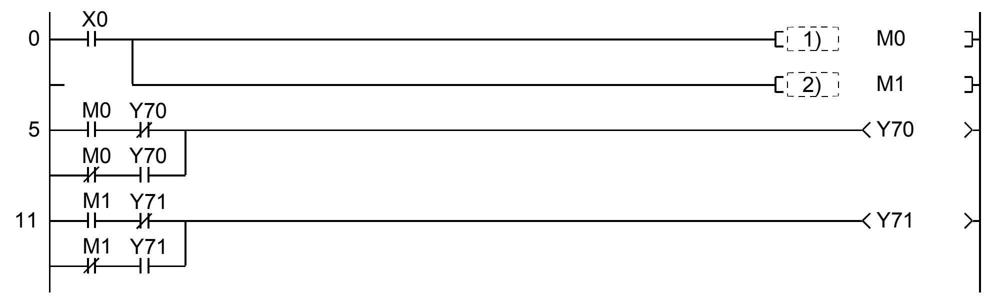
PLS, PLF

Y70 starts to switch between ON and OFF alternately upon detecting a rising edge of X0 signal, and a falling edge of the signal triggers Y71 to do the same operation as Y70 does.

[Timing Chart]



Create the following program with the GX Developer filling in the blanks ▭ , and verify the operation using the demonstration machine.



1) ______________________

2) ______________________

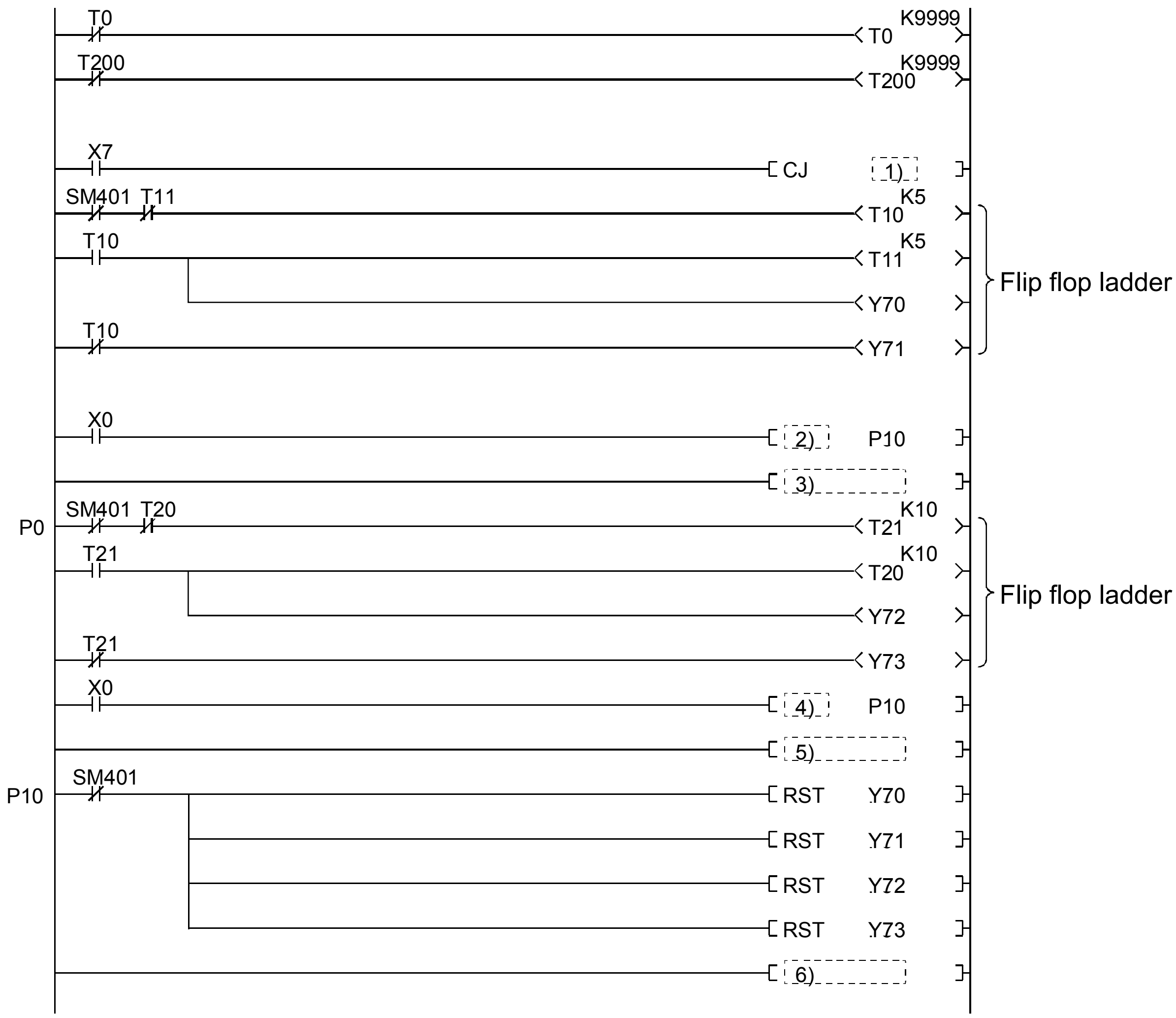
Path name	A:\SCHOOL
Project name	QTEST4
Program name	MAIN

4.8.4 Practice Question (4)

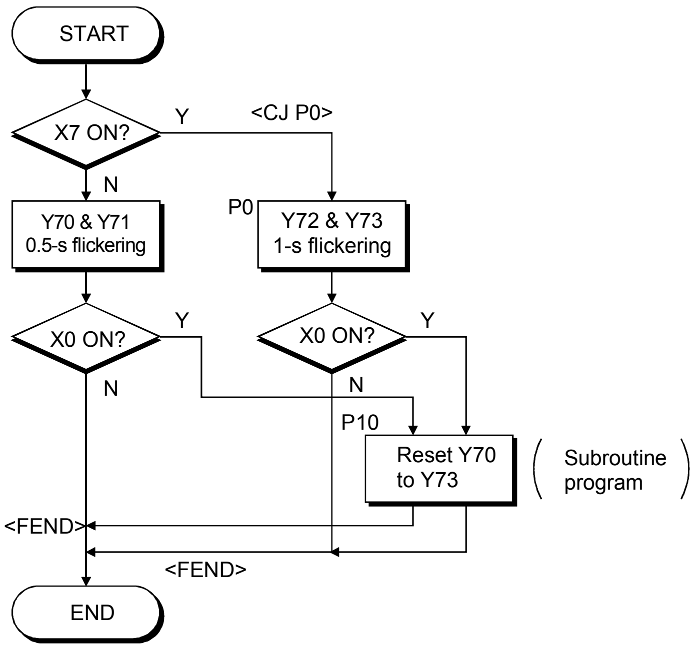
CJ, CALL, RET, FEND

Y70 and Y71 flicker for 0.5 s alternately when X7 is OFF, and when X7 is ON, Y72 and Y73 flicker for 1.0 s alternately. Turning X0 ON resets the currently flickering Y70 to Y73.

Create the following program filling in the blanks ☐, and verify the program with the demonstration machine.



Hint


START
X7 ON?
Y
<CJ P0>
N
Y70 & Y71
0.5-s flickering
P0
Y72 & Y73
1-s flickering
X0 ON?
Y
X0 ON?
Y
N
N
P10
Reset Y70
to Y73
Subroutine
program
<FEND>
<FEND>
END

1) ______________________
2) ______________________
3) ______________________
4) ______________________
5) ______________________
6) ______________________

Chapter 4 Practice Question Answers

Question No.		Answers
1	1)	Y70
	2)	X1
	3)	T1
	4)	T0
	5)	Y74
2	1)	SET M0
	2)	C0
	3)	Y70
	4)	SET M1
	5)	RST C0
	6)	RST M0
	7)	RST C0
3	1)	PLS
	2)	PLF
4	1)	P0
	2)	CALL
	3)	FEND
	4)	CALL
	5)	FEND
	6)	RET

CHAPTER 5 BASIC INSTRUCTION Part 2

5.1 Notation of Numbers (Data)

The PLC CPU converts all the input signals into ON or OFF signals (logical 1 or 0, respectively) to store and process them, and performs the numeric operation using the numeric value stored with the logical 1 or 0 (binary numbers = BIN).
In our everyday life, on the other hand, a decimal system is most commonly used because it is easier to understand. Therefore, decimal-to-binary conversion or the reverse is required whenever you write/read (monitor) numbers to/from the PLC. The programming system and some instructions have the function of performing the conversion.
This section explains how numbers (data) are expressed in decimal, binary, hexadecimal and binary-coded decimal notation (BCD), and how the conversion is made.

Decimal

- A decimal number system consists of ten single-digit numbers: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 which represent the order and size (amount). The number after 9 is 10. The number after 19 is 20 and so forth. Additional powers of 10 require the addition of another positional digit.
- The following shows how a base-ten number (in this case 153) is represented.

$$
\begin{aligned}
153 &= 100 + 50 + 3 \\
&= 1 \times 100 + 5 \times 10 + 3 \times 1 \\
&= \underline{1} \times \underline{10^2} + \underline{5} \times \underline{10^1} + \underline{3} \times \underline{10^0}
\end{aligned}
$$

Decimal (0 to 9) symbol

Base number 10 raised to the power of digit number

n ....Digit number (0,1,2…)
10 ....Decimal number

- The Q series PLC uses a symbol K when it expresses numbers in decimal notation.

Binary (BIN)

• The binary number is a base 2 method of counting in which only the digits 0 and 1 are used. When 1 is reached, counting begins at 0 again, with the digit to the left being incremented. The two digits 0 and 1 are referred to as bits.

Binary	Decimal
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
⋮	⋮

• Let's look at how the binary number shown below is converted to decimal.

"10011101"

The diagram below depicts the binary number with the powers of two written beneath it. The same incrementing pattern as the decimal system goes for the binary system except for the difference of the radix. Since the binary method uses only 0 and 1 digits, it has more carries than the decimal method.

7	6	5	4	3	2	1	0	← Bit number
1	0	0	1	1	1	0	1	← Binary symbol
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	← Base number 2 raised to the power of digit number = Bit value
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
128	64	32	16	8	4	2	1	

From the diagram, we see that the binary number can be broken down as:

$$\underline{1\times128}+0\times64+0\times32+\underline{1\times16}+\underline{1\times8}+\underline{1\times4}+0\times2+\underline{1\times1}$$

So, the equivalent decimal number is:

$$128+16+8+4+1=157$$

The decimal equivalent of a binary number can be calculated by adding together each digit 1 multiplied by its power of 2.

Hexadecimal

- The hexadecimal (often called hex for short) is a numeral system with a radix of 16 usually written using the symbols 0-9 and A-F. The A-F represent 10-15 respectively as shown in the table below. When F is reached, counting begins at 0 again, with the digit to the left being incremented just like the decimal and binary system.

Decimal	Hexadecimal	Binary
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111
16	10	10000
17	11	10001
18	12	10010
⋮	⋮	⋮
1 9 1 0 1	4 A 9 D	0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1

3	2	1	0	← Digit number
4	A	9	D	← Hex symbol

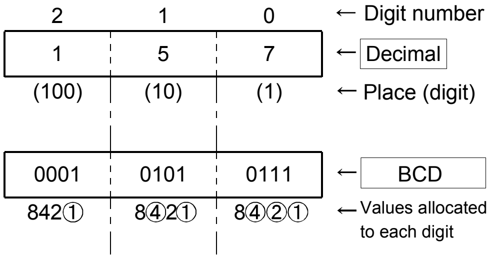
n ....Digit number
16 ....Hexadecimal

$$=(4)\times\underline{16}^{3}+(A)\times\underline{16}^{2}+(9)\times\underline{16}^{1}+(D)\times\underline{16}^{0}$$
$$=4\times4096+10\times256+9\times16+13\times1$$
$$=19101$$

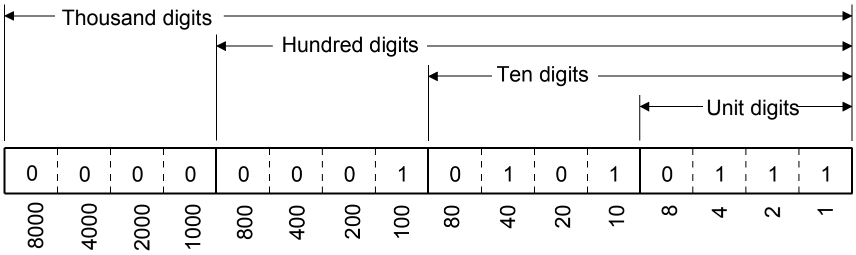
- One hex digit is equivalent to 4 bits of binary.
- The Q series PLC uses a symbol "H" when it expresses numbers in hex notation.
- The hex system is used to represent the specific numbers of the following devices.
  - Input and output (X, Y)
  - Input and output of function (FX, FY)
  - Link relay (B)
  - Link register (W)
  - Special relay for link (SM)
  - Special register for link (SW)
  - Link direct devices (Jn\X, Jn\Y, Jn\B, Jn\SB, Jn\W, Jn\SW)

Binary Coded Decimal (BCD)

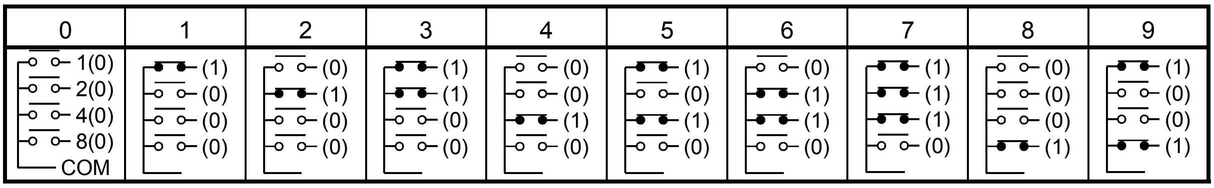
• The binary-coded decimal is a code in which a string of four binary digits represents a decimal number.
A decimal number 157, for example, is expressed as shown below in BCD.



• In BCD, decimal numbers (0 to the biggest 4-digit number; 9999) can be represented by 16 bits.
The diagram below shows the bit values allocated to each digit of BCD.



• The BCD is used for the following signals.
1) Output signals of digital switches
2) Signals of seven-element display (digital display)

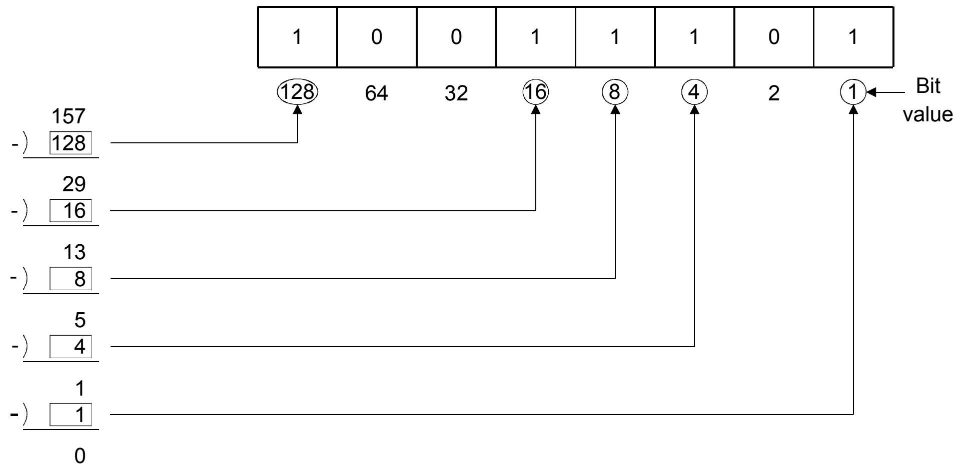


BCD Code Digital Switch

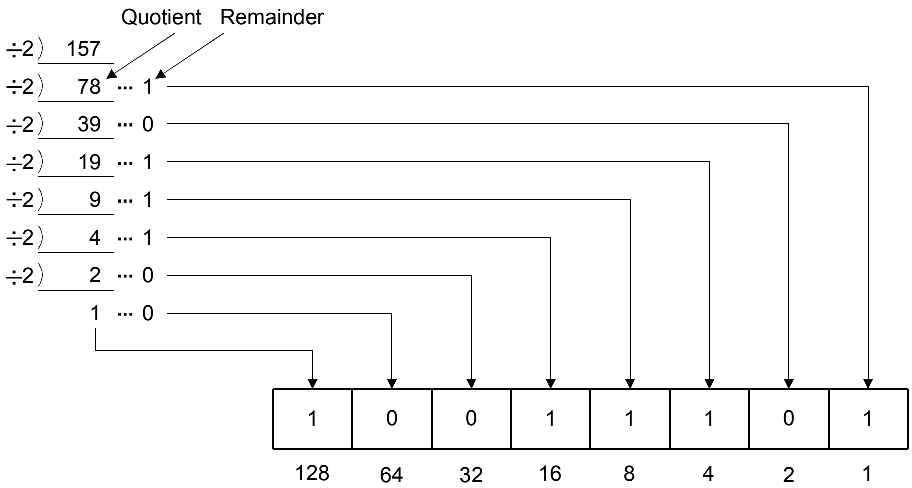
How to Convert Decimal to Binary

A decimal number 157, for example, is converted to binary as shown below.

1)



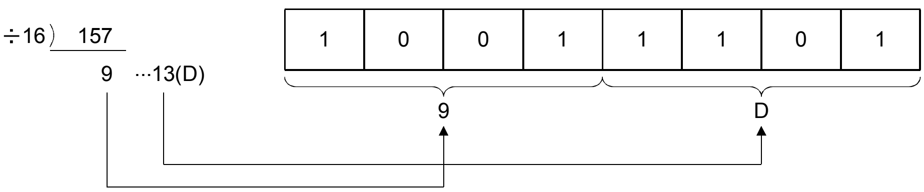
2)



How to Convert Decimal to Hex

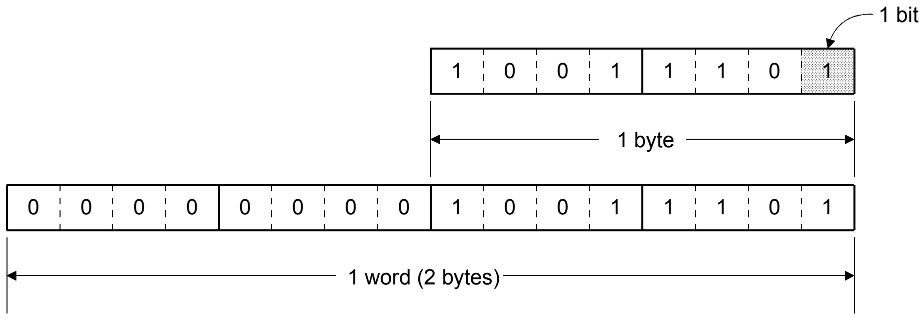
A decimal number 157, for example, is converted to hex as shown below.

1)

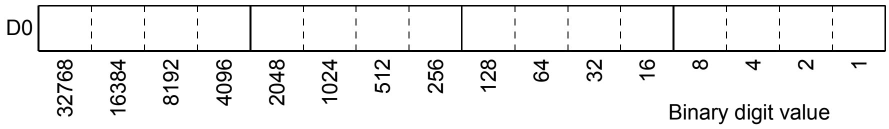


Valid Numbers for Q series PLC

• 8 bits is usually called 1 byte and 16 bits (2 bytes) is called 1 word.



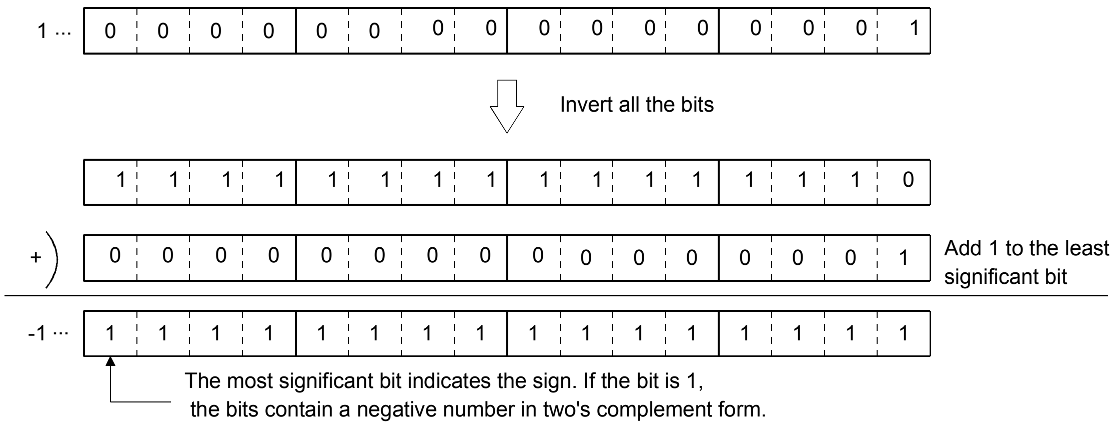
• Each register of word devices in the MELSEC-Q PLC consists of 16 bits.
  • Data register D
  • Current timer T value
  • Current counter C value
  • File register R
  • Link register W
  etc.



• The following two ranges of numbers can be processed using 16 bits (1 word).
  1) 0 to 65535
  2) -32768 to +32767

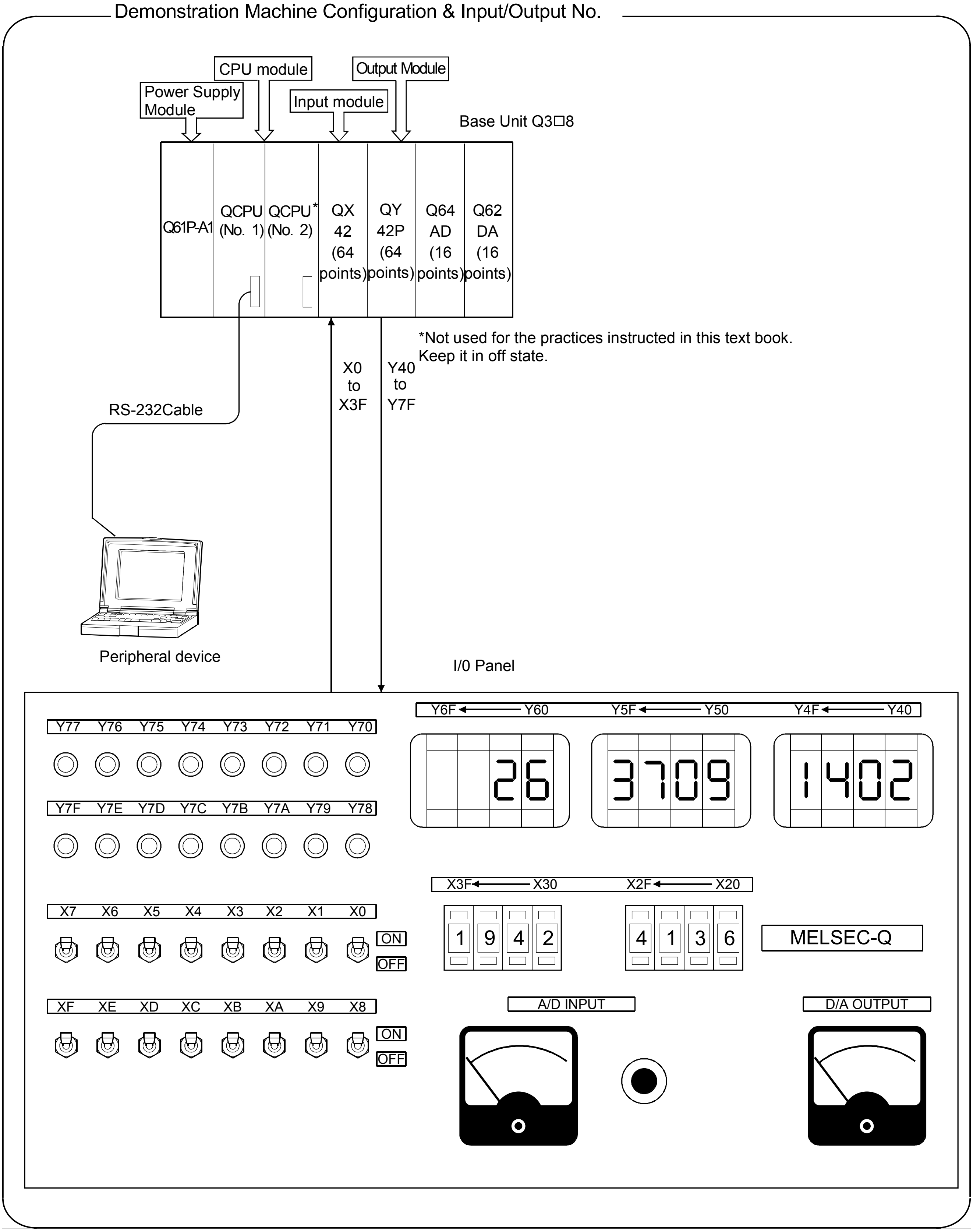
• The MELSEC-Q PLC uses the 2) range.
  The negative numbers adopt two's complement to positive numbers (1+0+32767).

• In the two's complement, each binary bit is inverted, and then 1 is added to the lower bits.
  Example)
  Negative 1 is represented as shown below using two's complement:



(Binary Coded Decimal) BCD	(Binary) BIN	(Decimal) K	(Hexadecimal) H
00000000 00000000	00000000 00000000	0	0000
00000000 00000001	00000000 00000001	1	0001
00000000 00000010	00000000 00000010	2	0002
00000000 00000011	00000000 00000011	3	0003
00000000 00000100	00000000 00000100	4	0004
00000000 00000101	00000000 00000101	5	0005
00000000 00000110	00000000 00000110	6	0006
00000000 00000111	00000000 00000111	7	0007
00000000 00001000	00000000 00001000	8	0008
00000000 00001001	00000000 00001001	9	0009
00000000 00010000	00000000 00001010	10	000A
00000000 00010001	00000000 00001011	11	000B
00000000 00010010	00000000 00001100	12	000C
00000000 00010011	00000000 00001101	13	000D
00000000 00010100	00000000 00001110	14	000E
00000000 00010101	00000000 00001111	15	000F
00000000 00010110	00000000 00010000	16	0010
00000000 00010111	00000000 00010001	17	0011
00000000 00011000	00000000 00010010	18	0012
00000000 00011001	00000000 00010011	19	0013
00000000 00100000	00000000 00010100	20	0014
00000000 00100001	00000000 00010101	21	0015
00000000 00100010	00000000 00010110	22	0016
00000000 00100011	00000000 00010111	23	0017
00000001 00000000	00000000 01100100	100	0064
00000001 00100111	00000000 01111111	127	007F
00000010 01010101	00000000 11111111	255	00FF
00010000 00000000	00000011 11101000	1000	03E8
00100000 01000111	00000111 11111111	2047	07FF
01000000 10010101	00001111 11111111	4095	0FFF
	00100111 00010000	10000	2710
	01111111 11111111	32767	7FFF
	11111111 11111111	—1	FFFF
	11111111 11111110	—2	FFFE
	10000000 00000000	—32768	8000

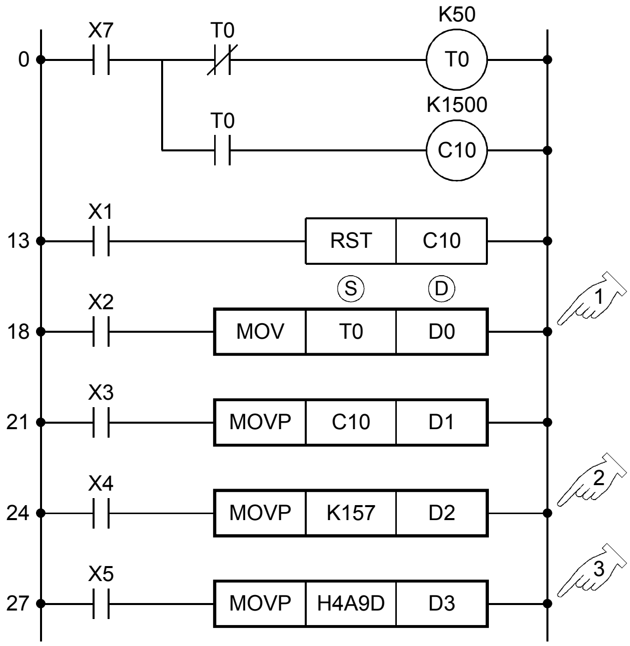
Demonstration Machine Configuration & Input/Output No.



5.2 Transfer Instruction

Path name	A:\SCHOOL
Project name	QB-11
Program name	MAIN

5.2.1 MOV (P) 16-bit data transfer



1 • As soon as the input condition goes active, the current value of timer T0 is transferred to the data register D0.
S…Source, D…Destination

• The current T0 value is stored in the register in binary form and transferred to the data register D0 without changing the form.

T0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	45
									128	64	32	16	8	4	2	1	

⬇

D0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1
									128	64	32	16	8	4	2	1

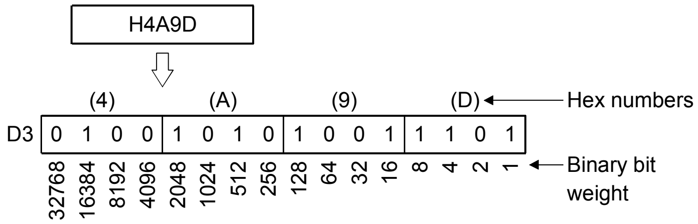
2 • When the input condition goes active, the decimal number 157 is transferred to the data register D2 and stored in the register binary form. The decimal number (K) is automatically converted to binary before it is transferred.

K157

⇩

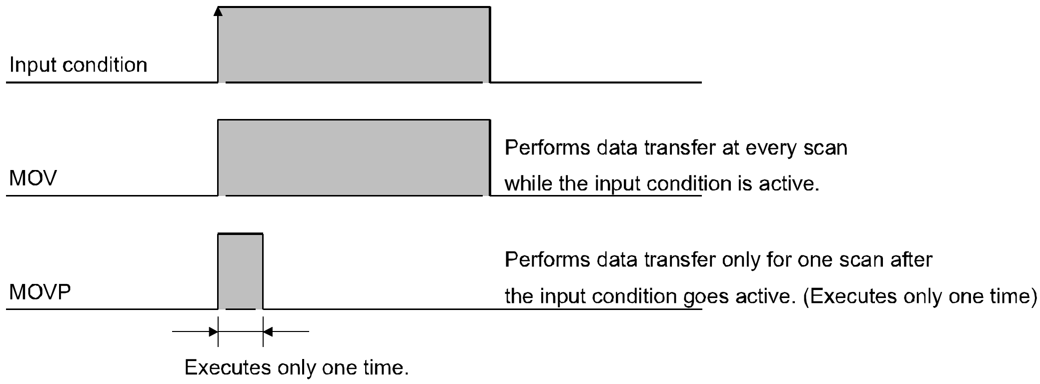
D2	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0	1
									128	64	32	16	8	4	2	1

• When the input condition goes active, the $4A9D_H$ is transferred to the data register D3.



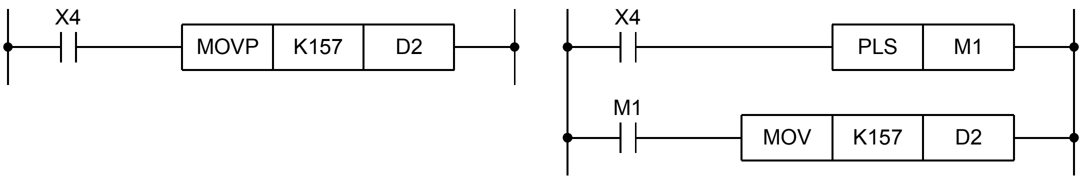
Difference Between MOV & MOVP

The P of MOVP stands for a pulse.



• Use the MOV instruction when the target data is changing and needs to be read frequently.
To read the fixed data such as setting data, or data required to identify the cause of abnormal condition, use the MOVP instruction.

• The following both of programs function the same.



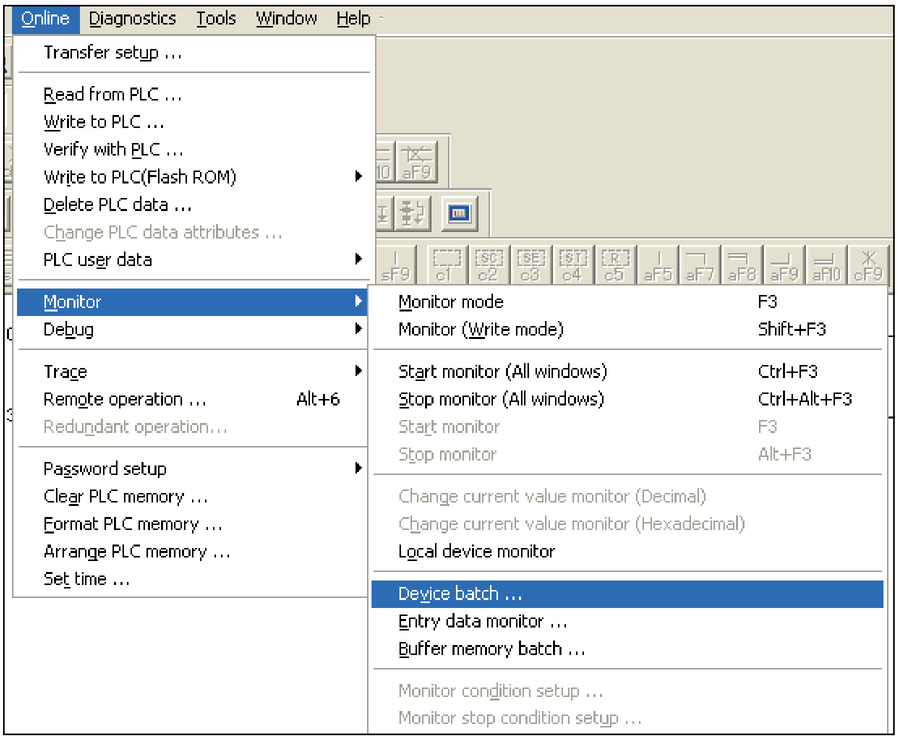
		Applicable Device												Digit	Basic number of steps
		Internal device (system user)		File register	MELSECNET/10 (H) Direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level		
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
MOV Ⓢ Ⓓ	S	○	○	○	○	○	○	○	○	○					*
	D	○	○	○	○	○	○	○							

* The number of steps varies by the device used.

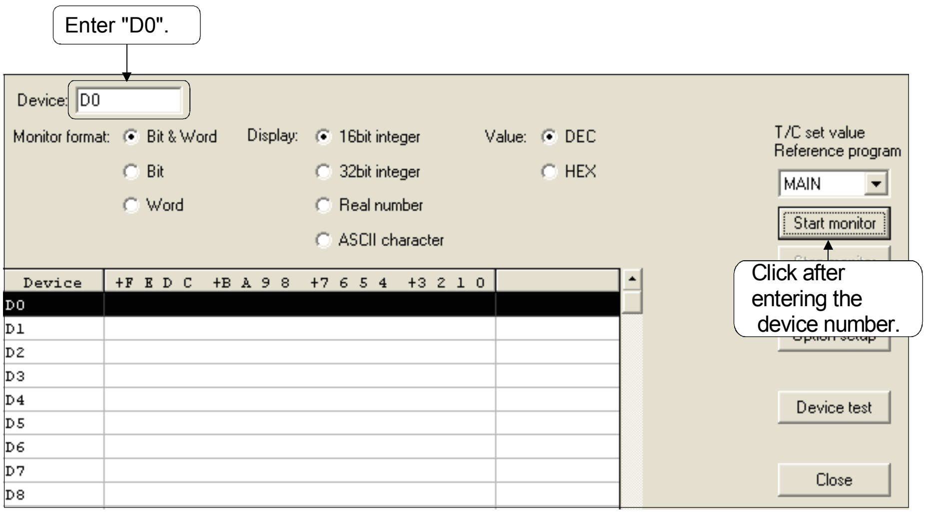
Check Operation

The CPU is in RUN
Input X2, X3, X4, X5, and X7 is ON.

• Monitor the data stored in the data register D0 to D3.
  • After writing the data to the PLC, select [Online]→[Monitor]→[Device batch].
    The Device batch monitor dialog box appears.



• Enter "D0" in the Device batch monitor dialog box and click the Start monitor button.



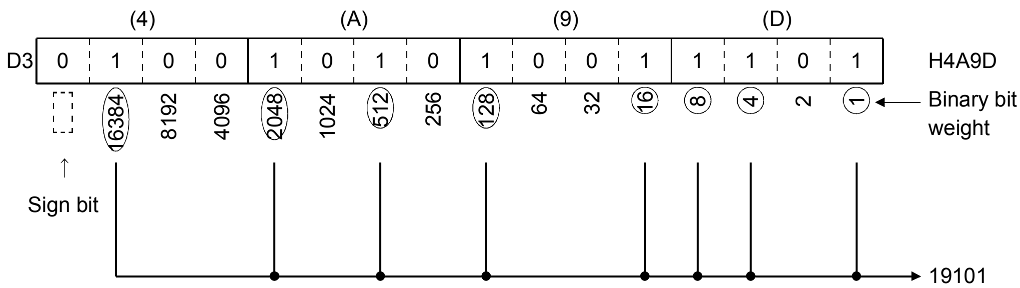
Device	+F E D C	+B A 9 8	+7 6 5 4	+3 2 1 0	
D0	0 0 0 0	0 0 0 0	0 0 0 0	1 1 1 1	15
D1	0 0 0 0	0 0 0 0	0 0 1 1	0 1 0 0	52
D2	0 0 0 0	0 0 0 0	1 0 0 1	1 1 0 1	157
D3	0 1 0 0	1 0 1 0	1 0 0 1	1 1 0 1	19101
D4	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0

Current values of timer and counter. (They changes.)

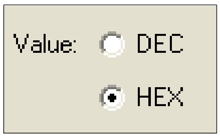
A decimal number 157 is stored.

A decimal number equivalent to a 4A9DH.

Indicates ON/OFF state of each bit in the word devices.
(Binary digit 0)
(Binary digit 1)



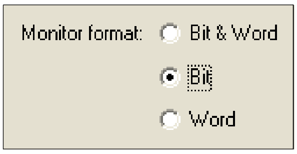
• Change the displayed decimal numbers into hex.
Select hex in the device batch monitoring dialog box.



[Device Batch Monitor Window]

Device	+F E D C	+B A 9 8	+7 6 5 4	+3 2 1 0	
D0	0 0 0 0	0 0 0 0	0 0 0 0	1 1 1 1	000F
D1	0 0 0 0	0 0 0 0	0 0 1 1	0 1 0 0	0034
D2	0 0 0 0	0 0 0 0	1 0 0 1	1 1 0 1	009D
D3	0 1 0 0	1 0 1 0	1 0 0 1	1 1 0 1	4A9D
D4	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0000

• Change the display format to the corresponding binary codes
Select Bit in the device batch monitoring dialog box.



[Device Batch Monitor Window]

Numbers in D1
Numbers in D0

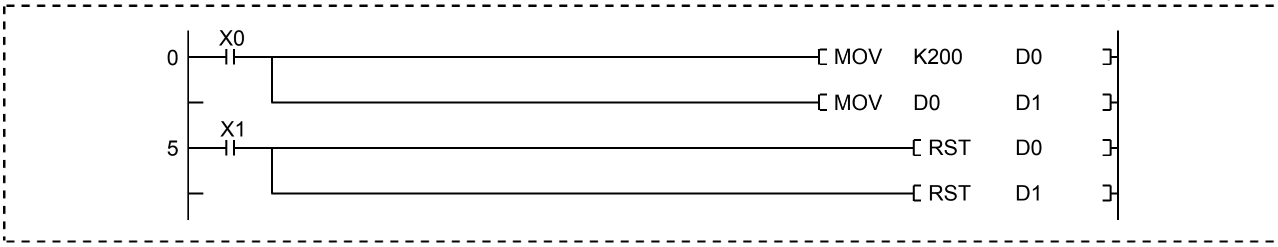
Device	+1 FEDC BA98 7654 3210	+0 FEDC BA98 7654 3210
D0	0000 0000 0011 0100	0000 0000 0000 1111
D2	0100 1010 1001 1101	0000 0000 1001 1101
D4	0000 0000 0000 0000	0000 0000 0000 0000

Numbers in D3
Numbers in D2

Path name	A:\SCHOOL
Project name	QEX7
Program name	MAIN

Ladder Example

Create the following ladder with the GX Developer and write it to the CPU of the demonstration machine to check if the MOV instruction works properly.



Operating Procedure

See Section 4.4 Operating Procedure for the detailed procedure of the following operations.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

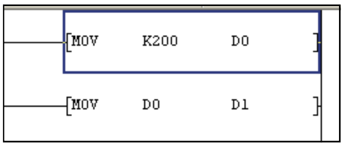
(4) Monitor the ladder

• How to alter the transfer instruction

Follow the procedure given below to alter the transfer instruction.

Example: Change the transfer data K200 of [MOV K200 D0] to K100

1) Double-click the instruction to be changed.



2) A ladder input window appears.

3) Move the cursor to "2" of "MOV K200 D0" and write "1" over the "2".

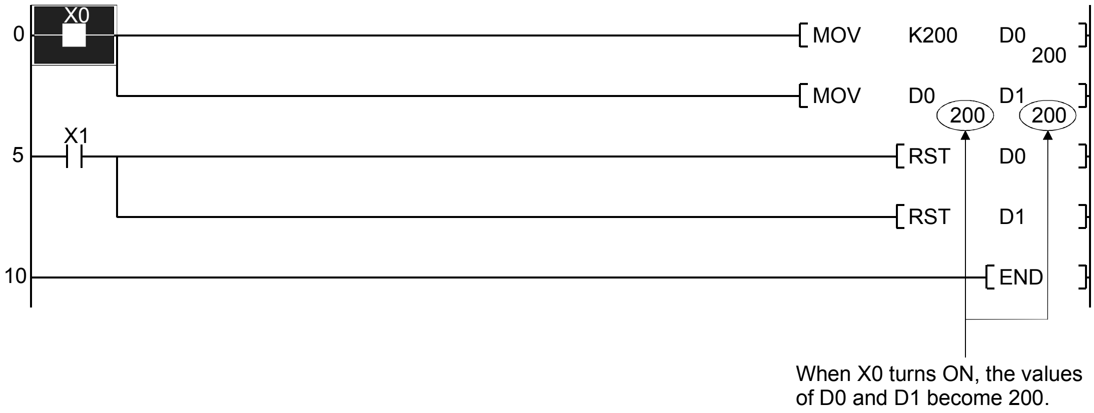
4) Click the OK button on the ladder input window.

All data exist in -[ ]- can be changed using the above method.
Be sure to make the changes in overwrite mode. If "Insert" is displayed in the lower right portion of the screen, press the Insert key to change it to overwrite.

5) When finished, click on the toolbar.

Operation Practice

Check that "200" is displayed under both D0 and D1 on the monitor screen when X0 on the control panel of the demonstration machine is turned ON.

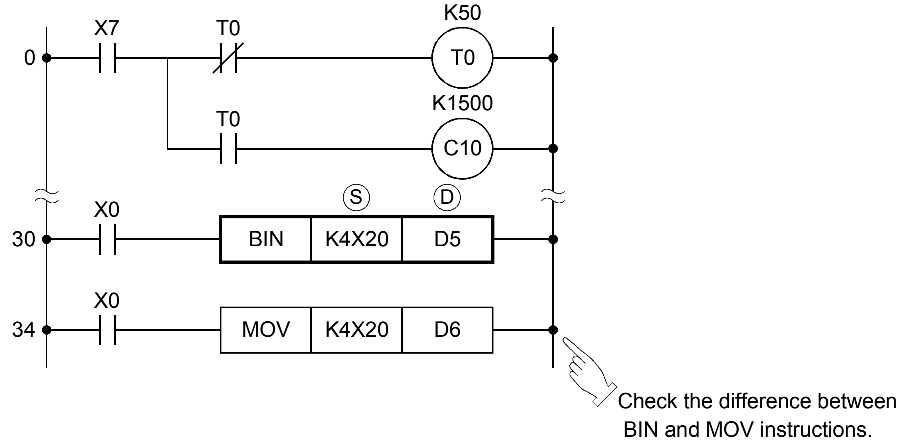


Related Practice Question Practice Question 5

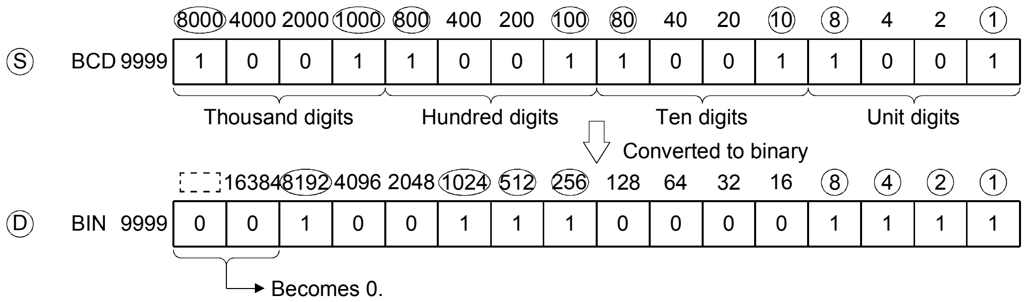
Path name	A:\SCHOOL
Project name	QB-12
Program name	MAIN

5.2.2 BIN (P) BCD→BIN data conversion instruction

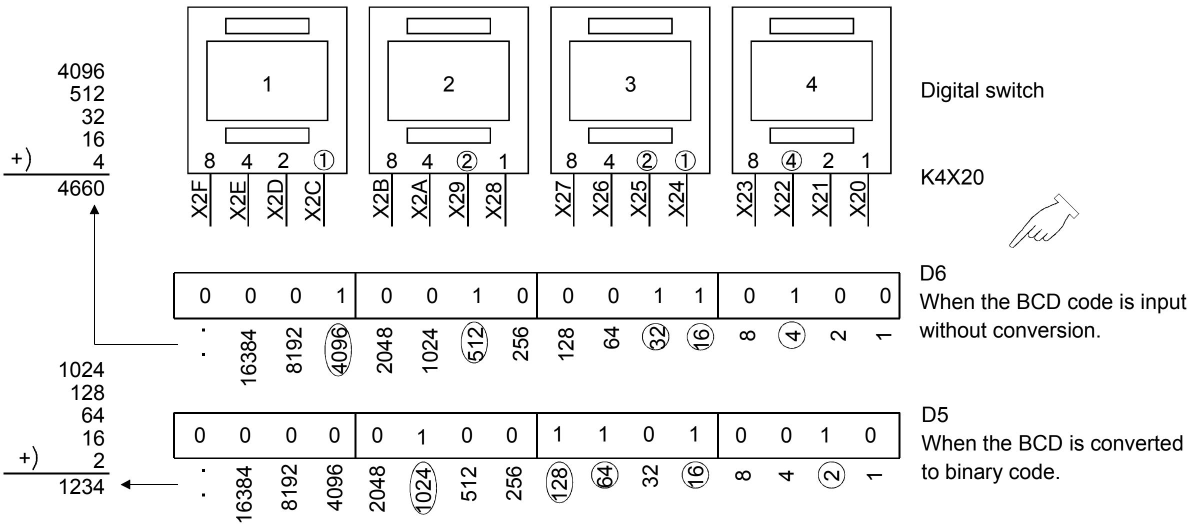
Operations to read and write data from 35 steps onward



• When the input condition is activated, the data in the device designated as Ⓢ is automatically judged as a BCD code, and converted into binary to be transferred to the device designated as Ⓓ.

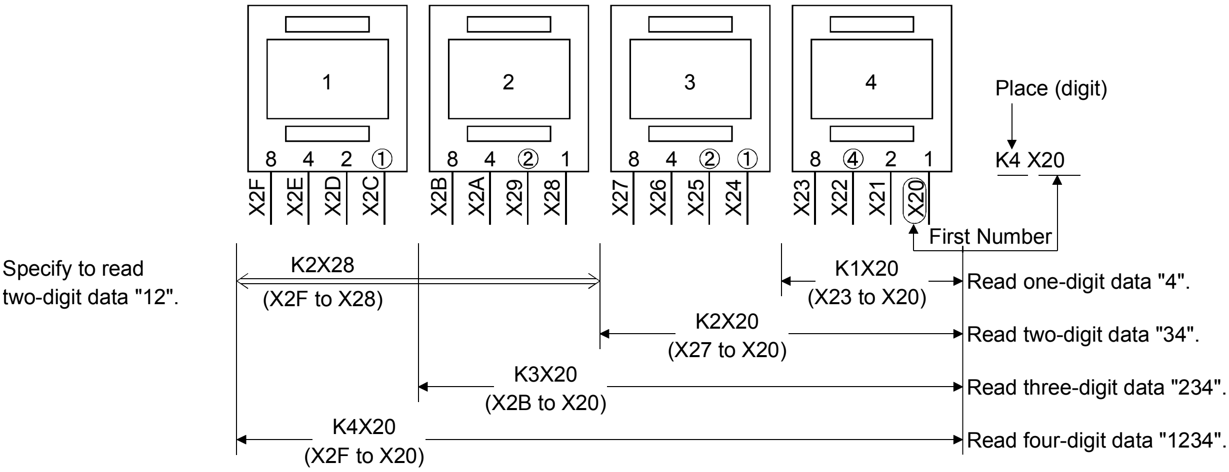


• As the ordinary digital switches generate BCD codes, the BIN instruction can be used to write data of the digital switches to the PLC.



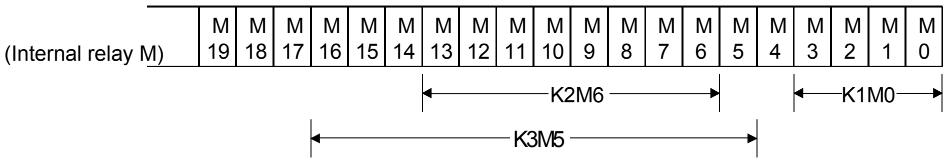
(K4X20)

- As each of the word devices D (data register), T (timer current value), and C (counter current value) consist of 16 bits (1 word), transferring the data is performed in units of device.
- Using 16 bit devices (such as X, Y, and M) allows you to handle 16-bit data. To do that, the numbers allocated to the 16 bit devices must be in consecutive order.
- Four bit devices allow you to handle data in units of 4 bits.



As long as four bit devices to read 4-digit data are in consecutive order, any bit device can be the first one.

- Other bit devices can be used in the same way as described above.

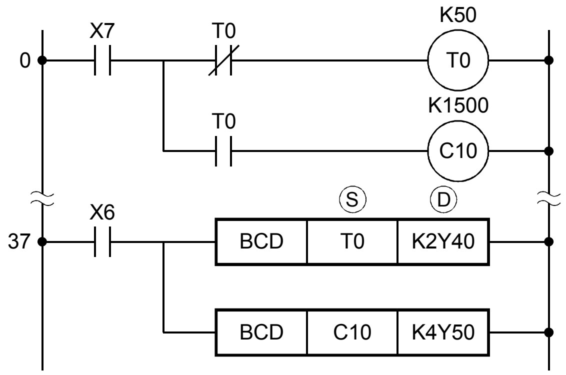


✻ A sample program that performs data input by digital switches is provided in Appendix 85.

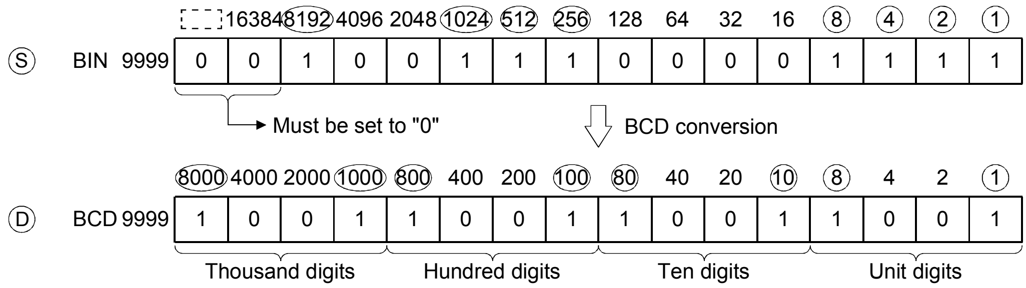
		Applicable Device														
		Internal device (system user)		File register	MELSECNET/10(H) Direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level	Digit	Basic number of steps
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
BIN (S) (D)	(S)	○	○	○	○	○	○	○						K1 to K4	3
	(D)	○	○	○	○	○	○	○							

Path name	A:\SCHOOL
Project name	QB-13
Program name	MAIN

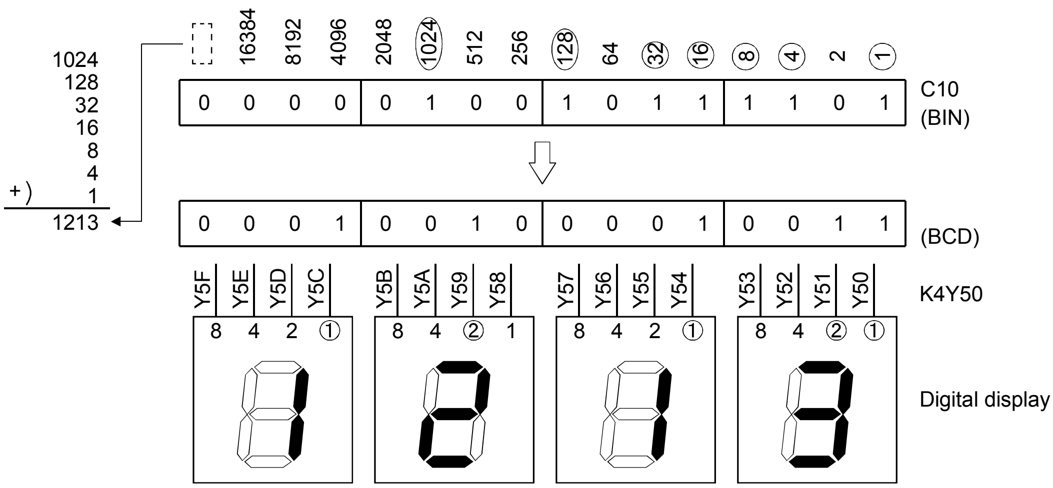
5.2.3 BCD (P) BIN→BCD data conversion instruction



- When the input condition is activated, data in the device designated as Ⓢ is automatically judged as a BIN code, and converted into BCD to be transferred to the device designated as Ⓓ.

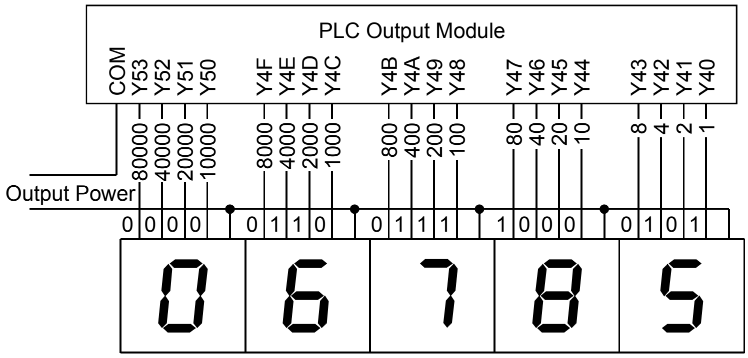


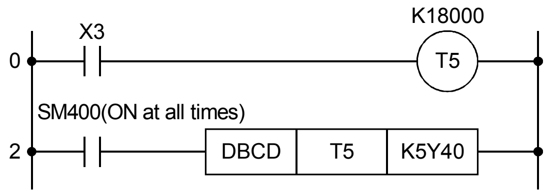
- As the ordinary digital displays display numbers using BCD code, the BCD instruction can be used to display data of the PLC (current timer and counter values, operation results).



Allowable Range with BCD Instruction

- The allowable range of data to be displayed by the BCD instruction (to be converted from BIN to BCD) is between 0 and 9999. Any data that falls outside the range cause an error.
  (Error code 50: OPERATION ERROR)
- When you want to display the current timer value that exceeds 9,999, use the DBCD instruction. The instruction allows you to handle 8-digit data up to 99,999,999.



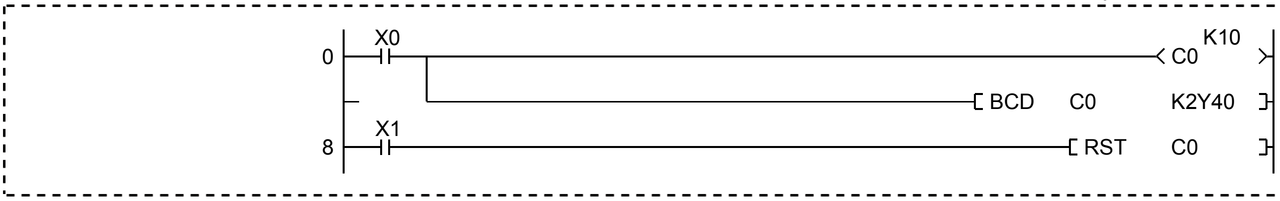


		Internal device (system user) Bit	Internal device (system user) Word	File register R	MELSECNET/10(H) Direct Jn\ Bit	MELSECNET/10(H) Direct Jn\ Word	Intelligent function module Un\G	Index register Z	Constant K	Constant H	Pointer P	Pointer I	Level N	Digit	Basic number of steps
BCD Ⓢ Ⓓ	Ⓢ	○	○	○	○	○	○	○						K1 to K4	3
	Ⓓ	○	○	○	○	○	○	○							

Path name	A:\SCHOOL
Project name	QEX8
Program name	MAIN

Ladder Example

Create the following ladder with GX Developer and write it to the CPU of the demonstration machine to check if the BCD instruction works properly.



Operating Procedure

Refer to Section 4.4 Operating Procedure for the detailed procedure of the following operations.

(1) Create a new project

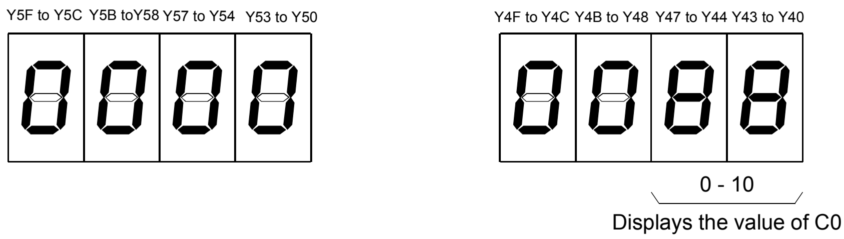
(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

Check that the current value of C0 is displayed on the Y40 to Y47 BCD digital displays when the X0 on the control panel is turned ON several times. The C0 is reset by turning the X1 ON.



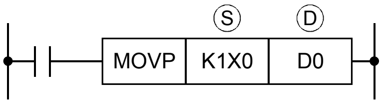
BCD Didital Display

Related Practice Question —— Practice Question 6

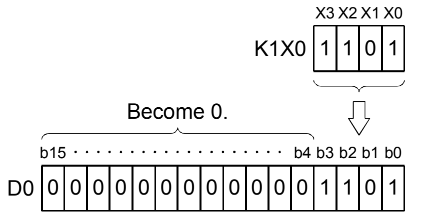
5.2.4 Example of specifying digit for bit devices and data transfer

Program Example	Process

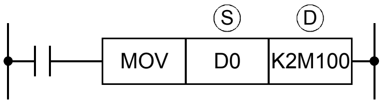
When the destination Ⓓ is a word device

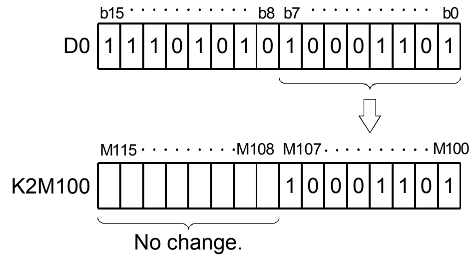


• Source: Source device
• Destination: Destination device

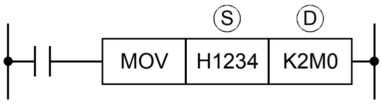


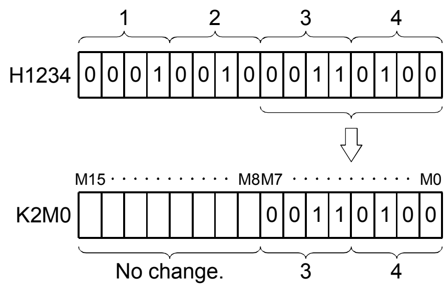
When the source Ⓢ is a word device



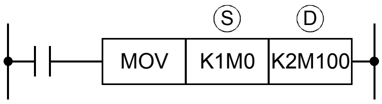


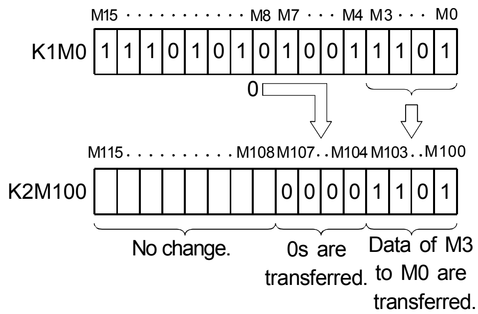
When the source Ⓢ is a constant





When the source Ⓢ is a bit device

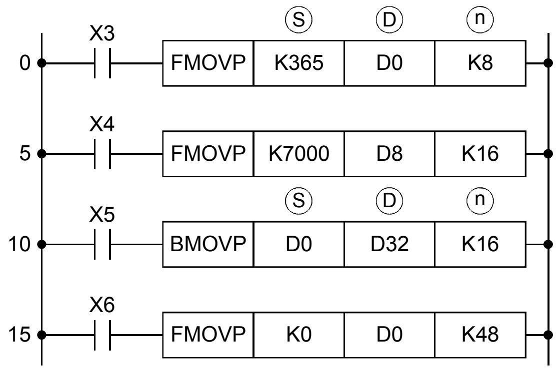




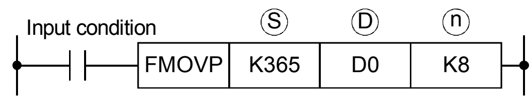
Path name	A:\SCHOOL
Project name	QB-14
Program name	MAIN

5.2.5 FMOV (P) FMOV (Batch transfer of the same data)

BMOV (P) BMOV (Batch transfer of the block data)



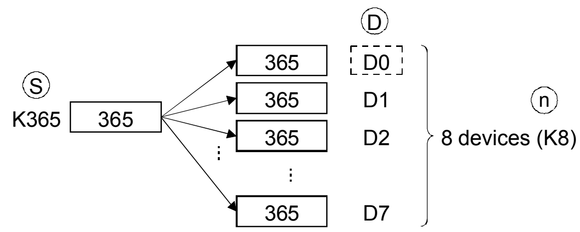
Operation Practice



FMOV

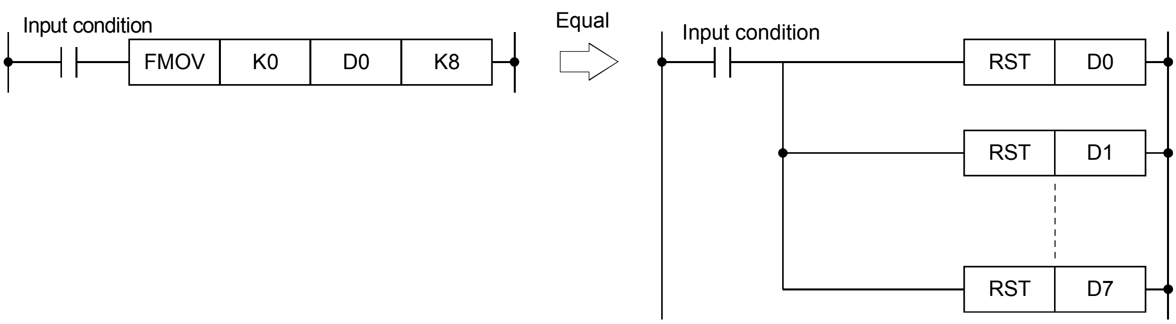
- When the input condition is activated, the FMOV instruction starts to transfer the data specified in S to the specified number (n) of devices starting from the device specified in D.

Example The FMOV instruction performs the following operation when X3 is turned ON.

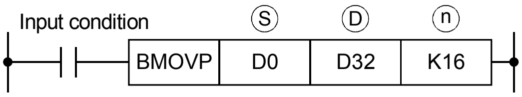


- FMOV instruction is useful when clearing many data all at once.

Example



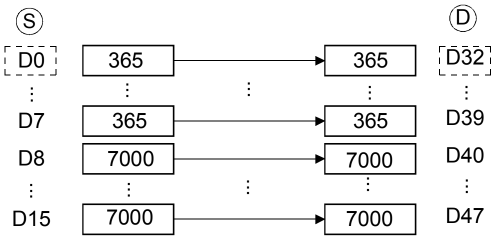
Only one FMOV instruction can be substituted for many RST instructions as shown above.



BMOV

- When the input condition goes active, the BMOV instruction performs the batch transfer of the data stored in the specified number (ⓝ) of devices starting from the device specified in Ⓢ, to the specified number (ⓝ) of devices starting from the device specified in Ⓓ.

Example The BMOV instruction performs the following operation when X5 is turned on.



- BMOV instruction is useful in the following situations:
  - When filing a logging data
  - For saving important data (e.g. data for auto-driving or measured data) into the latch area. This allows you to avoid data loss in the event of a power failure. Using the instruction, set to transfer the important data to data registers which have been set to live after the shut down.

		Available device												Digit	Basic number of steps
		Internal device (system, user)		File register	MELSECNET/10 (H) Direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer	Level			
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
FMOV Ⓢ Ⓓ ⓝ BMOV Ⓢ Ⓓ ⓝ	Ⓢ	○	○	○	○	○	○	(Note) ○	(Note) ○	(Note) ○				K1 to K4	4
	Ⓓ	○	○	○	○	○	○								
	ⓝ	○	○	○	○	○	○	○	○	○					

(Note) Not available in BMOV instruction.

Operation Practice

• Write the program on the previous page to the CPU, and run the CPU.

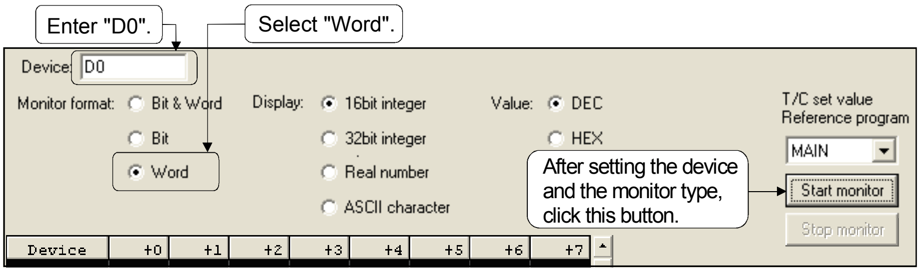
• Follow the following steps to perform device batch monitoring. The contents from D0 to D47 can be monitored.

After writing to PLC ⇨ Select [Online] → [Monitor] → [Device batch].

Enter the device "D0" in the device batch monitor dialogue box.

Select “Word” for the monitor type.

Click the Start monitor button.



[Monitor screen]

Device	+0	+1	+2	+3	+4	+5	+6	+7
D0	356	356	356	356	356	365	356	356
D8	0	0	0	0	0	0	0	0
D16	0	0	0	0	0	0	0	0
D24	0	0	0	0	0	0	0	0
D32	0	0	0	0	0	0	0	0
D40	0	0	0	0	0	0	0	0
D48	0	0	0	0	0	0	0	0

1) Turn X3 ON.
The numeric value 365 is sent to eight registers starting from D0 to D7 all at once.

Device	+0	+1	+2	+3	+4	+5	+6	+7
D0	356	356	356	356	356	365	356	356
D8	7000	7000	7000	7000	7000	7000	7000	7000
D16	7000	7000	7000	7000	7000	7000	7000	7000
D24	0	0	0	0	0	0	0	0
D32	0	0	0	0	0	0	0	0
D40	0	0	0	0	0	0	0	0
D48	0	0	0	0	0	0	0	0

2) Turn X4 ON.
The numeric value 7000 is sent to 16 registers starting from D8 to D23 all at once.

Device	+0	+1	+2	+3	+4	+5	+6	+7
D0	356	356	356	356	356	365	356	356
D8	7000	7000	7000	7000	7000	7000	7000	7000
D16	7000	7000	7000	7000	7000	7000	7000	7000
D24	0	0	0	0	0	0	0	0
D32	356	356	356	356	356	356	356	356
D40	7000	7000	7000	7000	7000	7000	7000	7000
D48	0	0	0	0	0	0	0	0

3) Turn X5 ON.
The contents of the 16 registers starting from D0 to D15 are sent to the 16 registers starting from D32 to D47 all at once.

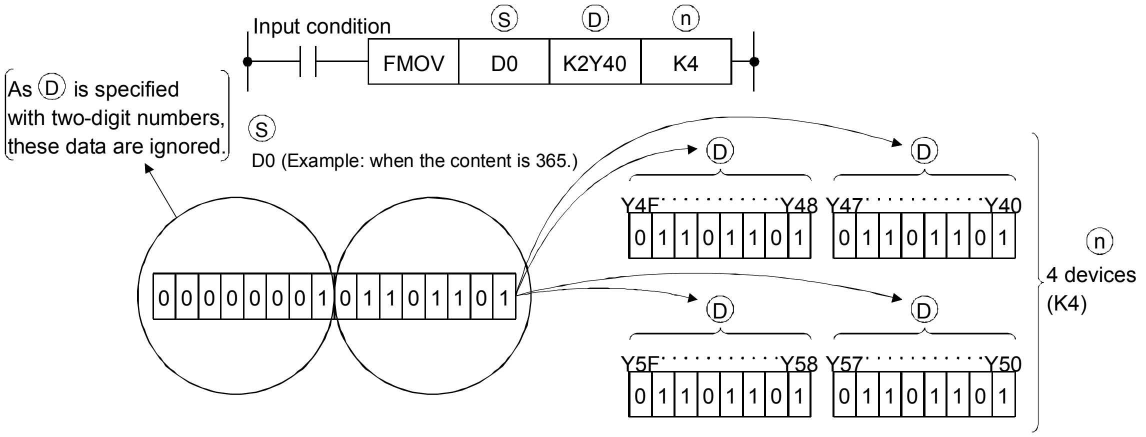
Device	+0	+1	+2	+3	+4	+5	+6	+7
D0	0	0	0	0	0	0	0	0
D8	0	0	0	0	0	0	0	0
D16	0	0	0	0	0	0	0	0
D24	0	0	0	0	0	0	0	0
D32	0	0	0	0	0	0	0	0
D40	0	0	0	0	0	0	0	0
D48	0	0	0	0	0	0	0	0

4) Turn X6 ON.
"0" is sent to the all 48 registers starting from D0 to D47 all at once. In other words, all the 48 registers are cleared to 0 at a time.

Reference

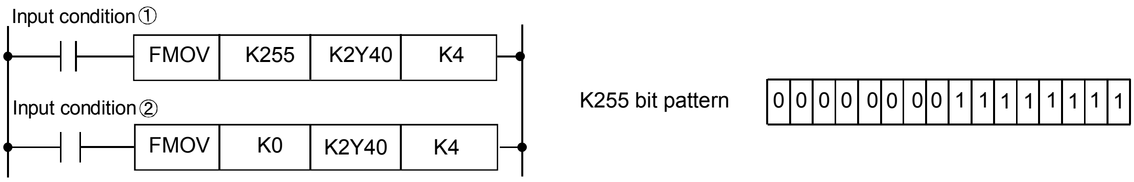
• If a bit device is specified as (D), the operation becomes as follows;

FMOV instruction



• Among the device from Y40 to Y5F, the devices specified as "1" are output first.

• Programming as shown below allows you to turn ON the output of Y40 to Y5F all at once by activating the input condition (1), or to turn OFF the output of Y40 to Y5F at a time by activating the input condition (2).

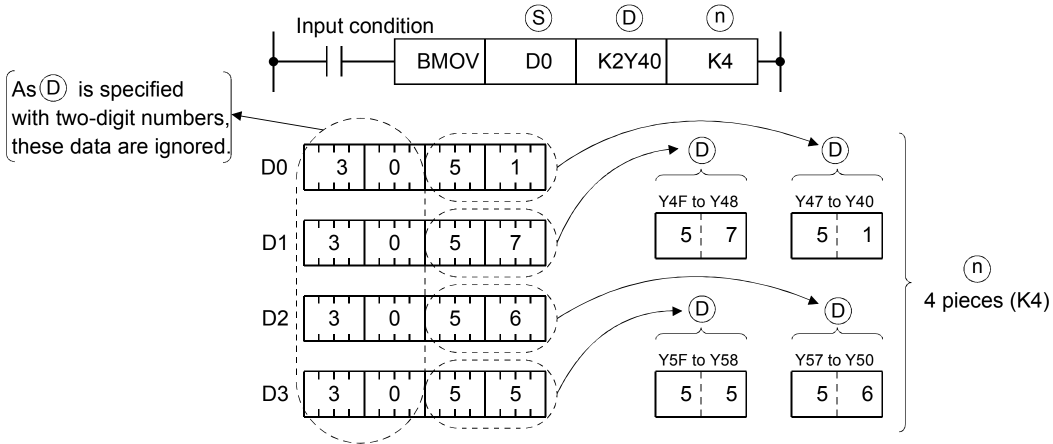


• When turning off bit devices in units of 4 bits;

16 bit devices or less	⇨ MOV instruction	e.g.	MOV K0 K4M0
32 bit devices or less	⇨ DMOV instruction	e.g.	DMOV K0 K8M0
More than 32 bit devices	⇨ FMOV instruction	e.g.	FMOV K0 K4M0 K4

(Turns OFF 64 bit device)

BMOV instruction

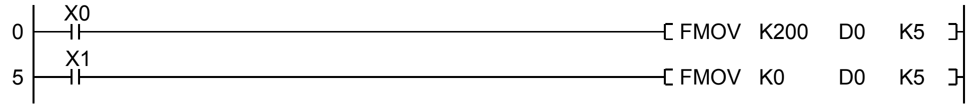


• The product codes (hexadecimal numbers) are stored in the devices from D0 to D3 as shown above. The instruction is useful for displaying and monitoring the last two digits that represent their types.

Path name	A:\SCHOOL
Project name	QEX9
Program name	MAIN

Ladder Example

Create the ladder chart shown below using GX Developer, write the chart into the demonstration machine, and confirm if the FMOV instruction is correctly performed.



(Operating Procedure)

The following procedures are the same as the (Operating Procedure) described in Section 4.4.

(1) Create a new project

(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

(Operation Practice)

Make sure that the contents of the devices from D0 to D4 become 200s on the batch monitor screen by turning X0 ON on the control panel of the demonstration machine. The data in the devices are cleared by turning X1 ON.

Device	+0	+1	+2	+3	+4	+5	+6	+7
D0	200	200	200	200	200	0	0	0
D8	0	0	0	0	0	0	0	0
D16	0	0	0	0	0	0	0	0
D24	0	0	0	0	0	0	0	0
D32	0	0	0	0	0	0	0	0
D40	0	0	0	0	0	0	0	0
D48	0	0	0	0	0	0	0	0

Change the setting of the device batch monitor as shown below to display the numbers in decimal, hexadecimal, or binary numbers.

Numeric value: decimal………..displays the numbers in decimal.
Numeric value: hexadecimal…..displays the numbers in hexadecimal.
Monitor type: bit…………………displays the numbers in binary.

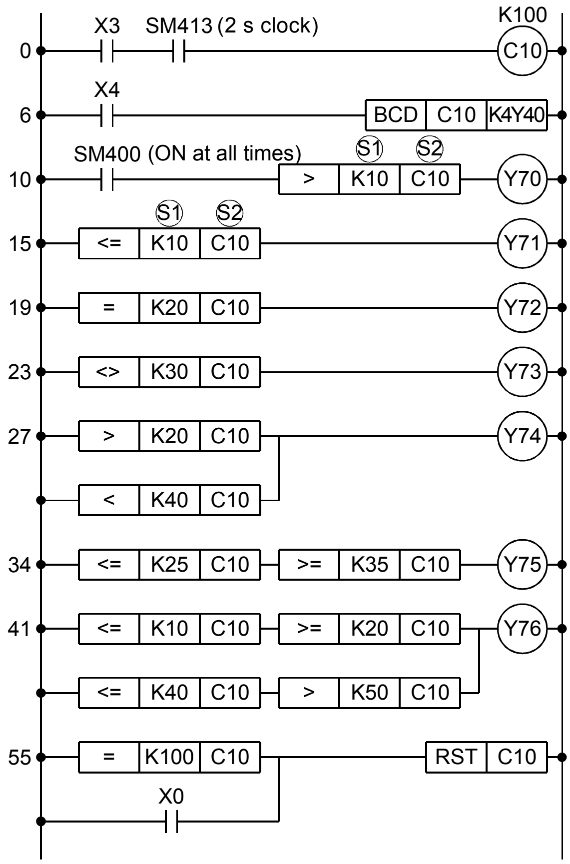
(Related Practice Question) Practice Question 7

Path name	A:\SCHOOL
Project name	QB-15
Program name	MAIN

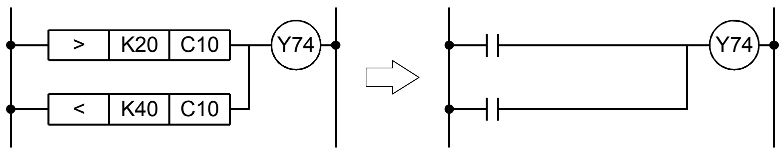
5.3 Comparison Operation Instruction

=	<>	>
>=	<	<=

Size comparison



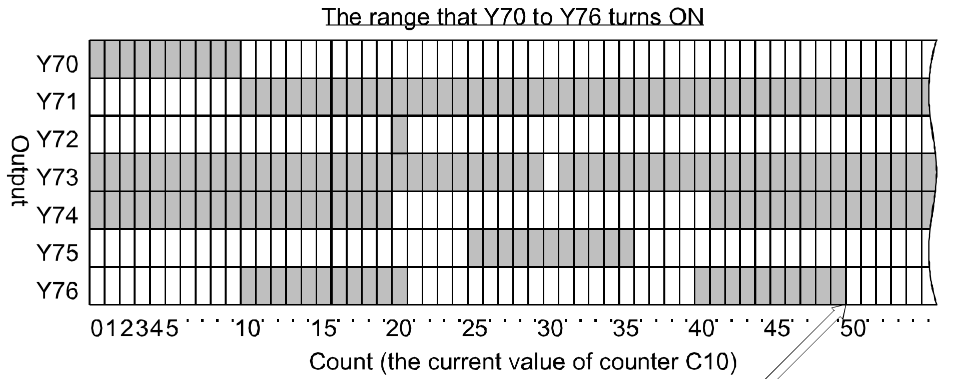
- The comparison operation instruction compares the data of source 1 (S1) and source 2 (S2), and brings the devices into conduction if the conditions are met.
- The instruction can be considered as one normally open (⊣⊢) as they are only conducted when the conditions are met.



- [= | S1 | S2] Becomes conducted when source 1 and source 2 agree.
- [< | S1 | S2] Becomes conducted when source 1 is smaller than source 2.
- [> | S1 | S2] Becomes conducted when source 1 is larger than source 2.
- [<= | S1 | S2] Becomes conducted when source 1 and source 2 agree or when source 1 is smaller than source 2.
- [>= | S1 | S2] Becomes conducted when source 1 and source 2 agree or when source 1 is larger than source 2.
- [<> | S1 | S2] Becomes conducted when source 1 and source 2 do not agree.

(Operation Practice)

• Write the program to the CPU.
• Turn ON both X3 and X4.
• C10 starts to count (one count every two seconds). The current counted value is displayed on the digital display (Y40 to Y4F).
• Make sure that the devices Y70 to Y76 turn ON as follows.



Difference between [>] and [>=] ⇨ [> | K50 | C10] equals 49.
[>= | K50 | C10] equals 50.

• The counter is designed to be reset every 200 s.
• In this way, the comparison instruction not only compares one data, but also specifies the range. This function is commonly used for the program that judges the passing status of products, etc.

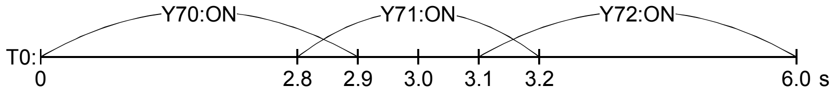
		Applicable device												Digit	Basic number of steps
		Internal device (system, user)		File register	MELSECNET/10 (H) direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level		
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
Comparison instruction S1 S2	S1	○	○	○	○	○	○	○	○	○				K1 to K4	3
	S2	○	○	○	○	○	○	○	○	○					

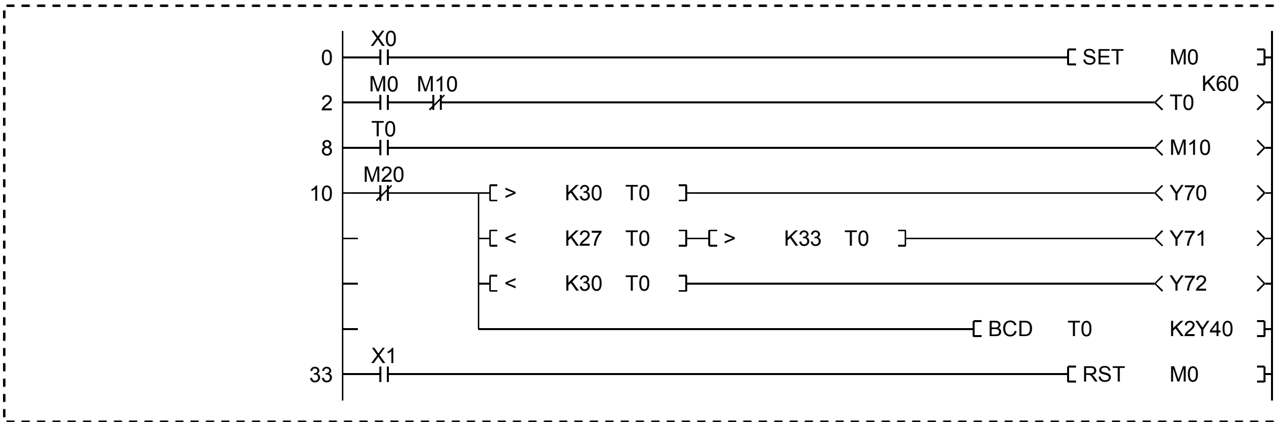
Path name	A:\SCHOOL
Project name	QEX10
Program name	MAIN

Ladder Example

Read the following ladder diagram from FD, write it to the demonstration machine, and make sure that >, < instructions are correctly performed.

0 s $\leqq$ T0 < 3 s → Y70: ON, 2.7 s < T0 < 3.3 s → Y71: ON,
3 s < T0 $\leqq$ 6 s → Y72: ON



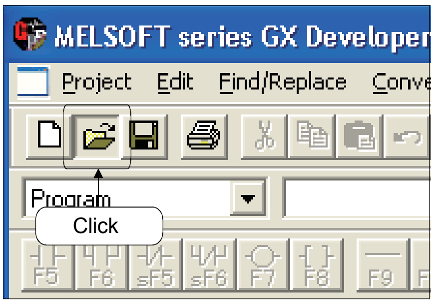


Operating Procedure

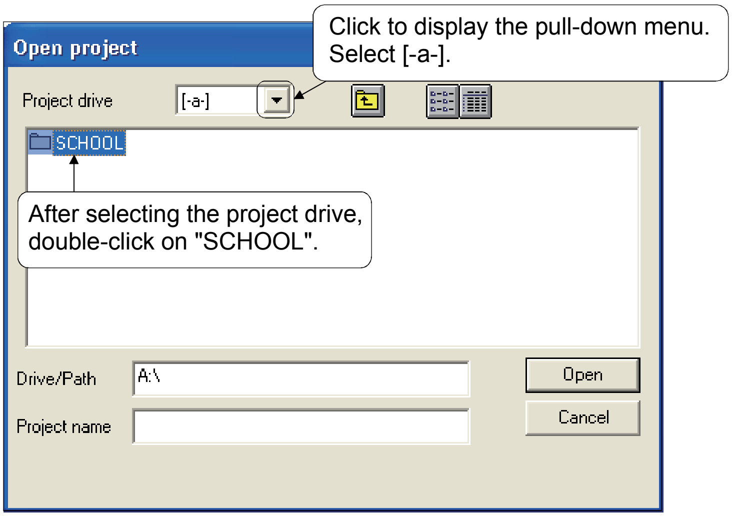
(1)Read data from FD

Read the project data stored in FD.

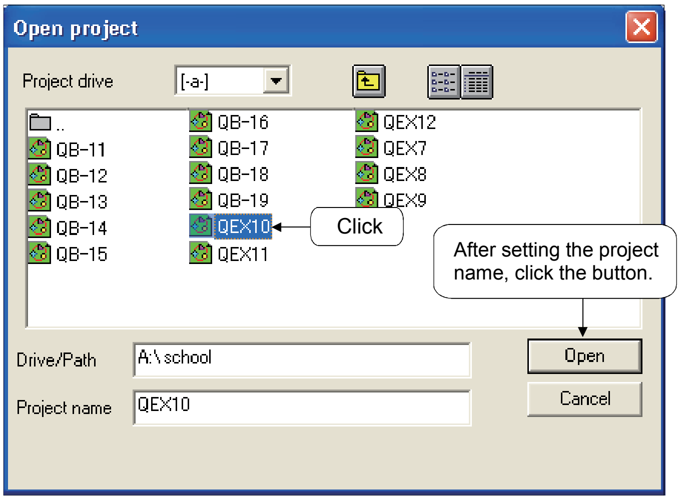
• Click  on the tool bar.



- Click [▼] beside "Project drive" and select [-a-] to display "SCHOOL" and the path name.
- Double-click on "SCHOOL".



- Double-click on "QEX10". Once "QEX10" is displayed on the project name box, click the Open button.



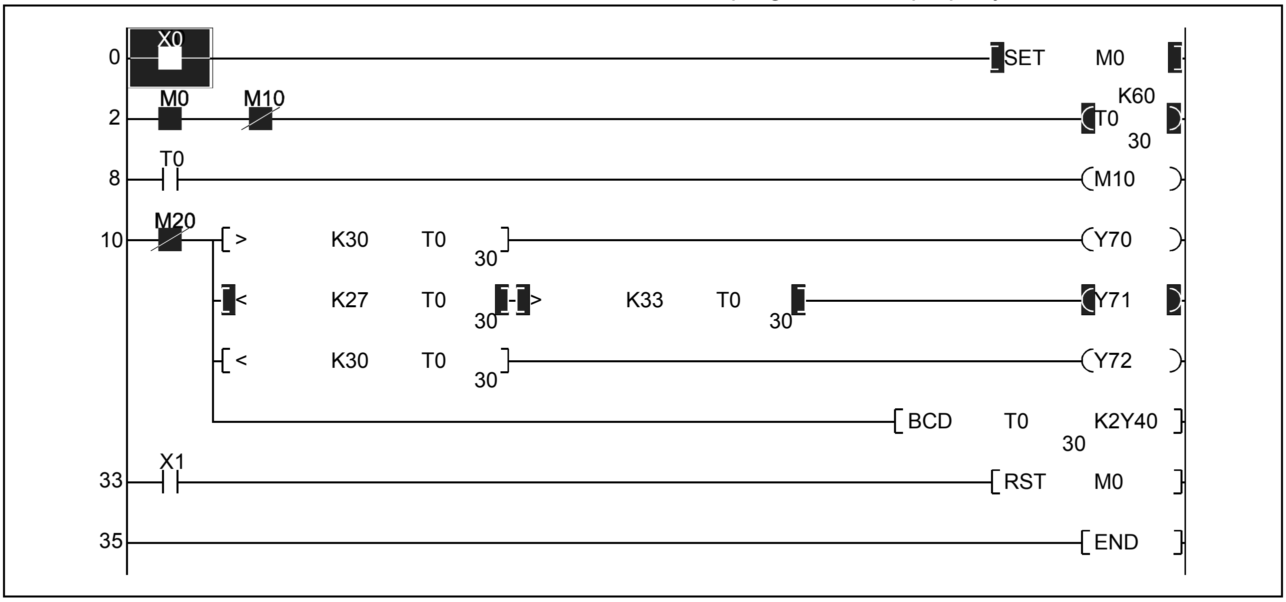
The following procedures are the same as the Operating Procedure in Section 4.4.

(2) Write to the PLC

(3) Monitor the ladder

Operation Practice

• Turn X0 ON, and make sure that the program works properly.

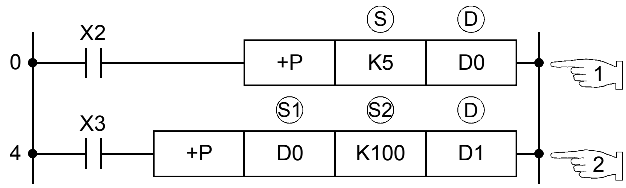


Related Practice Question —— Practice Question 8

Path name	A:\SCHOOL
Project name	QB-16
Program name	MAIN

5.4.1 +(P) BIN 16 bit data addition

-(P) BIN 16 bit data subtraction



1 • The device content specified as Ⓓ is added the device content specified as Ⓢ, and the result is stored in Ⓓ device every time the input condition is turned ON.

	Ⓓ		Ⓢ		Ⓓ
	D0 ☐	+	(5)	→	D0 ☐
(Input condition)					
First ON	0 (Assumption)	+	5	→	5
Second ON	5	+	5	→	10
Third ON	10	+	5	→	15

D0 contents are changed

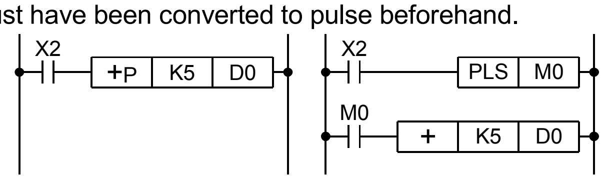
2 • The device content specified by Ⓢ1 is added the device content specified by Ⓢ2 and the result is stored in Ⓓ device when the input condition is turned ON.

	Ⓢ1		Ⓢ2		Ⓓ
	D0 ☐	+	(100)	→	D1 ☐
(Input condition)					
ON	15 (Assumption)	+	100	→	115

D0 contents are not changed.

CAUTION

- +P or -P should always be used for add-subtract instructions.
- When + or - is used for the instructions, add-subtract operation is performed upon every scanning. To use + or - for the instructions, operands must have been converted to pulse beforehand.

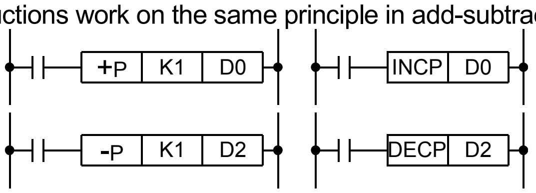


REFERENCE

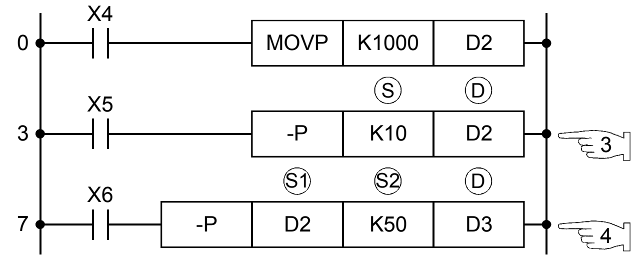
- The following two instructions work on the same principle in add-subtract operation.

(Addition)

(Subtraction)



Path name	A:\SCHOOL
Project name	QB-17
Program name	MAIN



3 • The device content specified as S is subtracted from the device content specified as D, and the result is stored in D device every time the input condition is turned ON.

	D		S		D
	D2	-	(10)	→	D2
(Input condition)					
First ON	1000	-	10	→	990
	(Assumption)	-	10	→	980
Second ON	990	-	10	→	970
Third ON	980				

D2 contents are changed.

4 • The device content specified as S2 is subtracted from the device content specified as S1 and the result is stored in D device when the input condition is turned ON.

	S1		S2		D
	D2	-	(50)	→	D3
(Input condition)					
ON	970	-	50	→	920

(Assumption)

D2 contents are not changed.

		Applicable device												Digit	Basic number of steps
		Internal device (system, user)		File register	MELSECNET/10 (H) direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level		
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
Add-substract instruction S D Add-substract instruction S1 S2 D	S S1 S2	○	○	○	○	○	○	○	○	○				K1 to K4	3 / 4
	D	○	○	○	○	○	○	○							

The basic number of steps becomes four steps for S1 S2 D type.

Path name	A:\SCHOOL
Project name	QEX11
Program name	MAIN

Ladder Example

Create the ladder diagram shown below with GX Developer, write it to the demonstration machine, and confirm if "+, - instructions" work properly.

Operating Procedure

The following procedures are the same as Operating Procedure described in Section 4.4.

(1) Create a new project

(2) Create a program

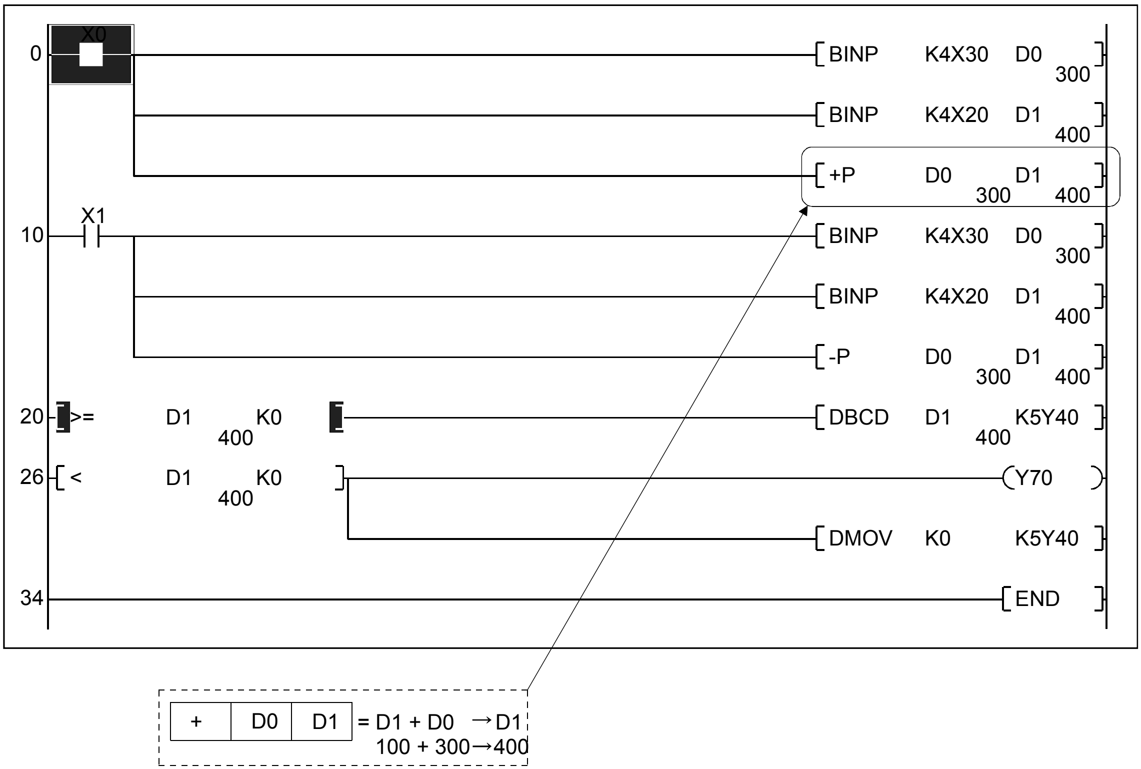
(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

(1) When X0 is turned ON, the addition of data in X30 to 3F and data in X20 to 2F is performed, and the result is output to Y40 to Y53.

(2) When X1 is turned ON, the subtraction of data of X30 to 3F and data of X20 to 2F is performed, and the result is output to Y40 to 53. When the result is negative value, Y70 is turned ON while Y40 to 53 are cleared to 0.

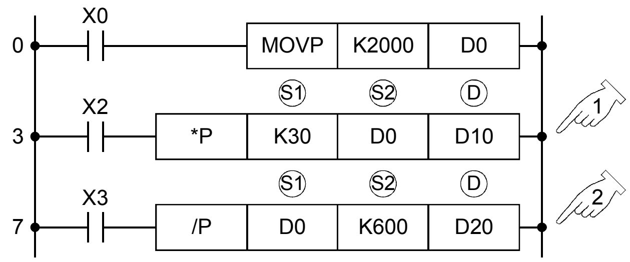


Related Practice Question Practice Question 9

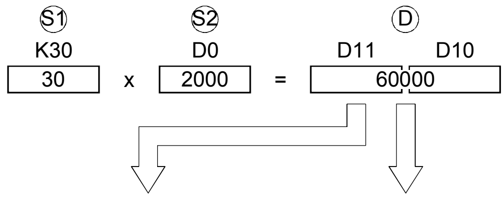
Path name	A:\SCHOOL
Project name	QB-18
Program name	MAIN

5.4.2 * (P) BIN 16 bit multiplication

/(P) BIN 16 bit division



1 • The device content specified by S1 is multiplied by the device content specified by S2, and the result is stored in the device specified by D when the input condition is turned ON.



16 bit (1 word) is not enough for storing the result of 16-bit data x 16-bit data.
Thus, D10 that is specified by the program and D11, which is the next device number, work as the holder of the result.

This device is handled as 32-bit register to hold the result. Left-most bit of D10 (b15) is not a sign bit.
It is handled as a part of the data.

The instructions should be handled as 32-bit when programming using the calculation result of the (P) instruction. (e.g. DMOV instruction, DBCD instruction)

☞2 • The device content specified by (S1) is divided by the device content specified by (S2) when the input condition is turned on, and the result is stored in the device specified by (D).

(S1)		(S2)		(D)	
D0		K600		D20	D21
2000	÷	600	=	3	200
				Quotient	Remainder

Quotient goes to D20 specified by the program. and ⇨ Remainder goes to D21, which is the next device number.

Digits after the decimal point of the operation result are ignored.

• If a bit device is specified as (D), the quotient is stored, however, the remainder is not saved.

• Examples when handling negative values are explained below.

Example -5 ÷ (-3) = 1, remainder -2
5 ÷ (-3) = -1, remainder 2

• Examples when dividing a number by 0, or dividing 0 by a number are explained below.

Example
0÷0 }
1÷0 } Error “OPERATION ERROR”
0÷1 Both quotient and remainder are 0

Operation Practice

• Write the program to the CPU and run the program.
• Turn X0 ON and store "2000" (BIN value) into D0.
• Turn X2 ON. The following operation is performed.
If the "60000" (operation result of D11 and D10) is handles as 16-bit integral number and only D10 is monitored, "-5536" is displayed. To avoid this, follow the procedures on the following pages.

(S1)		(S2)		(D)	
K30	×	D0	=	D11	D10
(30)		(2000)		(60000)	

• Turn X3 ON.

(S1)		(S2)		(D)	
D0	÷	K600	=	D21	D20
(2000)		(600)		(200)	(3)
				Remainder	Quotient

		Applicable device												Digit	Basic number of steps
		Internal device (system, user)		File register	MELSECNET/10 (H) direct Jn\		Intelligent function module Un\G	Index register	Constant		Pointer		Level		
		Bit	Word	R	Bit	Word		Z	K	H	P	I	N		
Multiply-division instruction (S1) (S2) (D)	(S1)	○	○	○	○	○	○	○	○	○				K1 to K4	* / 4
	(S2)	○	○	○	○	○	○	○	○	○					
	(D)	○	○	○	○	○	○	○							

The basic number of steps for multiplication instruction is three or four steps, and that for division instruction is four steps.

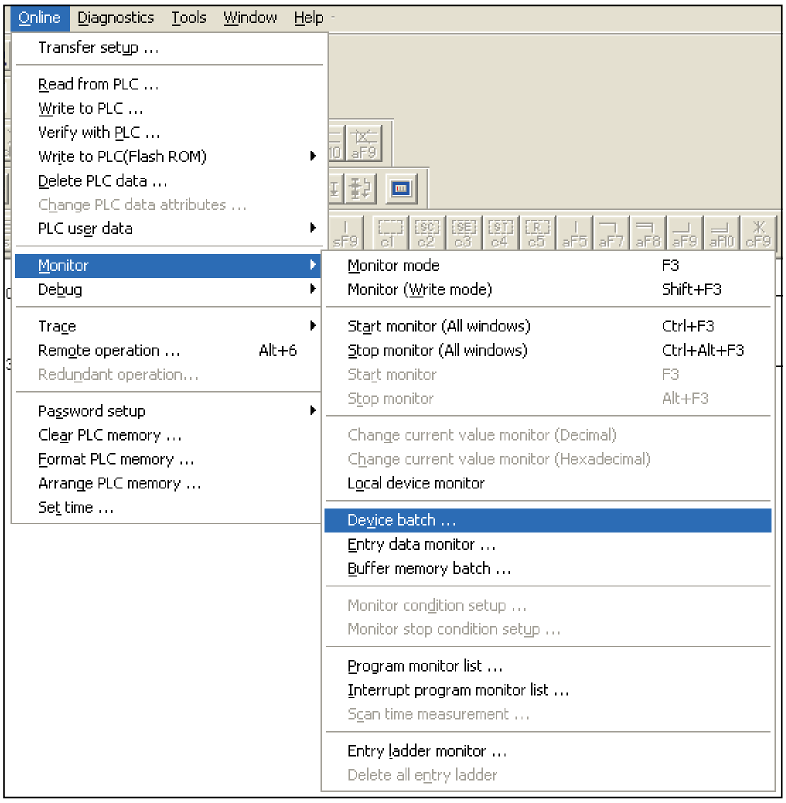
* The multiplication instruction varies depending on the device to be used.

• How to monitor 32-bit integral number data

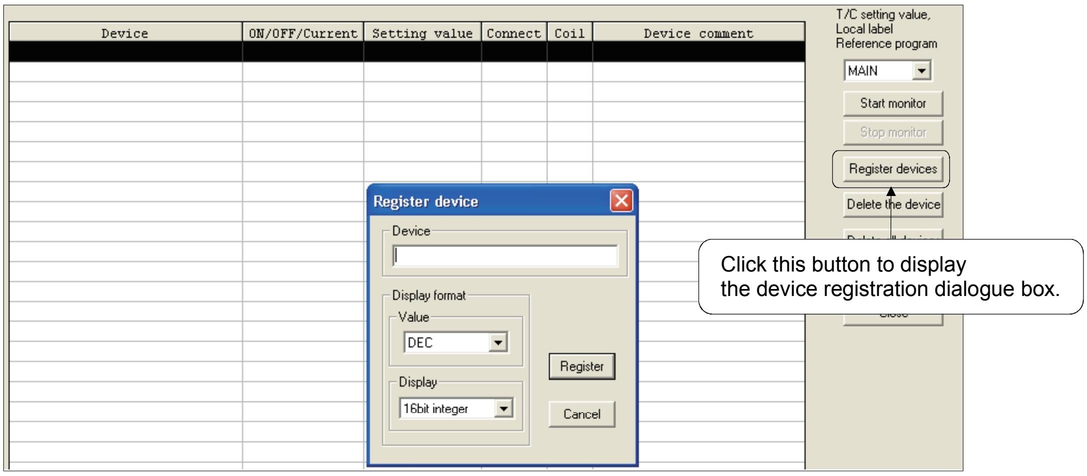
If the operation result of the multiplication instruction is out of the range from 0 to 32,767, the result cannot be displayed properly when the number is handled as 16-bit integral number and the contents of the lower register are monitored in ladder.

To monitor those numbers correctly, follow the steps below (device registration monitor operation).

• Select [Online] → [Monitor] → [Entry data monitor].



• Click [Register devices] to show the device registration dialogue box.



Continued on the next page

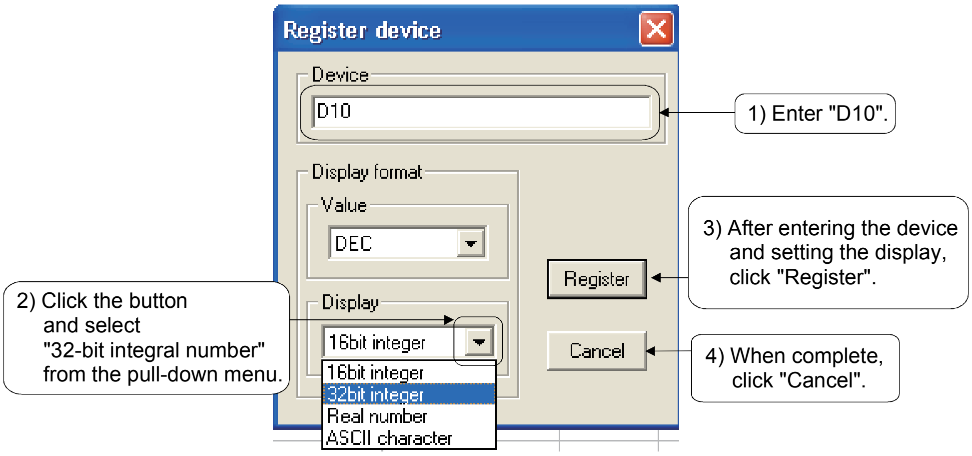
Continued from the previous page

• Enter "D10" in the device edit box on the device registration dialogue box.

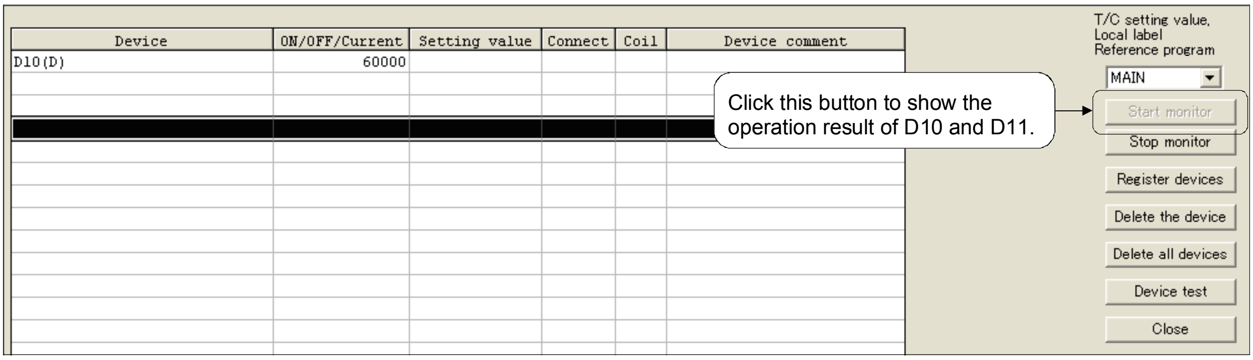
• Click ▼ on the display edit box, and select "32-bit integral number".

• Click the Register button. The items are displayed on the device registration monitor dialogue box.

• When complete, click the Cancel button to close the window.



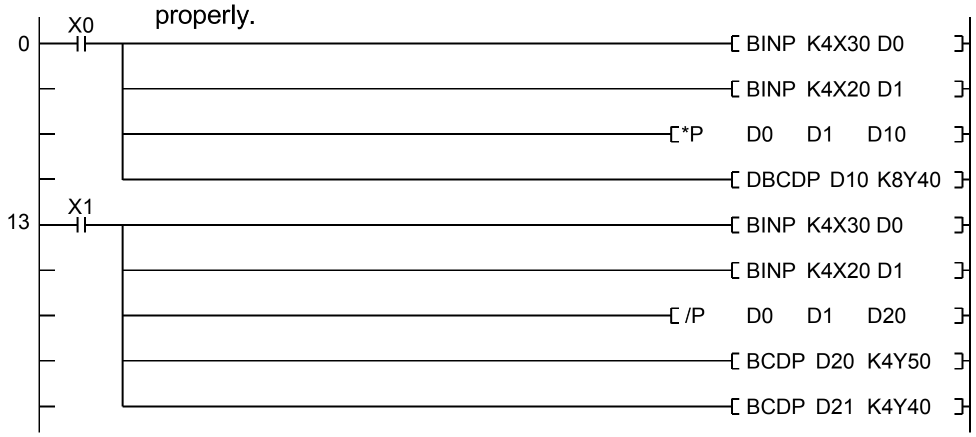
5) Click the Start monitor button.
Thus, 2 words of the operation result stored in D10 and D11 can be monitored.



Path name	A:\SCHOOL
Project name	QEX12
Program name	MAIN

Ladder Example

Create the ladder chart shown below with GX Developer, write it to the demonstration machine, and make sure that "*, / instructions" can be performed properly.



Operating Procedure

The following procedures are the same as the Operating Procedure described in Section 4.4.

(1) Create a new project

(2) Create a program

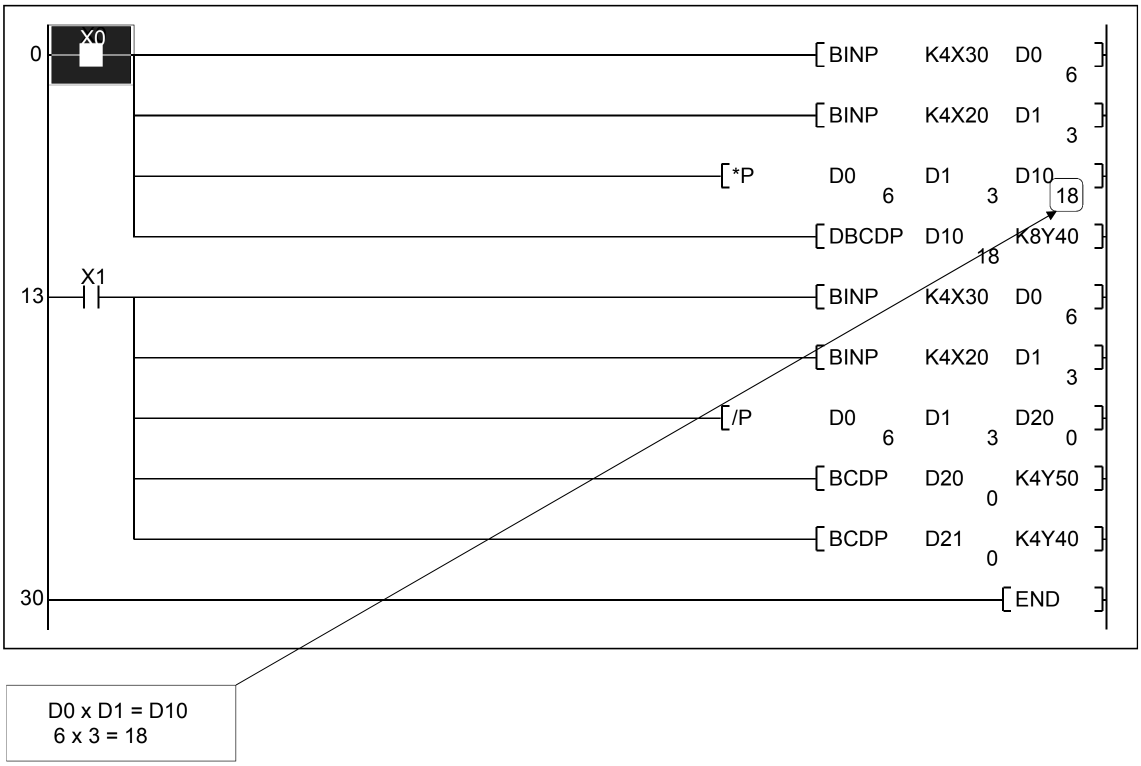
(3) Write to the PLC

(4) Monitor the ladder

(Operation Practice)

(1) When X0 is turned ON, the data of X20 to 2F is multiplied by that of X30 to 3F, and the result is output to Y40 to 5F.

(2) The data of X20 to 2F is divided by that of X30 to 3F when X1 is turned ON. Its quotient is output to Y50 to 5F, and the remainder is output to Y40 to 4F.



As the operation result of this example is in the range from 0 to 32767, the result is displayed properly even though the number is handled and monitored as 16-bit integral number.

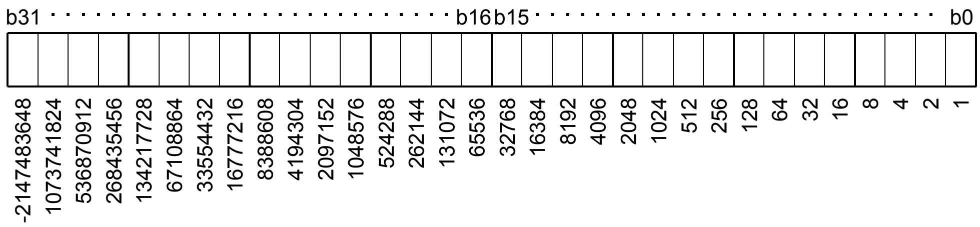
(Related Practice Question) Practice Question 10, Practice Question 11

5.4.3 32-bit data instructions and their necessity

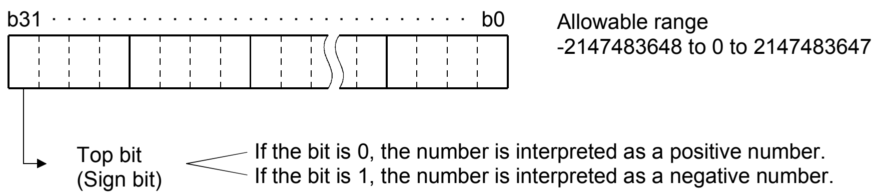
- The data memory of Q-series PLC, which consists of 16-bit, is a memory of 1-word unit. The memory generally processes transfer, comparison, and arithmetic operations in 1-word unit.
- Q-series PLC also supports 2-word (32-bit) unit. In this case, add "D" at the head of each instruction to indicate that the instruction is handled as 2-word. The following shows the examples:

Data / Instruction	1-word 16-bit	2-word 32-bit
Transfer	MOV(P)	DMOV(P)
	BIN(P)	DBIN(P)
	BCD(P)	DBCD(P)
Comparison	<, >, < = > =, =, < >	D <, D >, D < = D > =, D =, D< >
Four arithmetic operations	+ (P)	D + (P)
	- (P)	D - (P)
	* (P)	D * (P)
	/ (P)	D / (P)
Available range of numbers	-32768 to 32767 (0 to 9999) Numbers in parentheses are for BIN (P), BCD (P) instructions.	-2147483648 to 2147483647 (0 to 99999999) Numbers in parentheses are for BIN (P), DBCD (P) instructions.
Available range of digits	K1 to K4	K1 to K8

- The bit value of the 32-bit configuration is as follows:

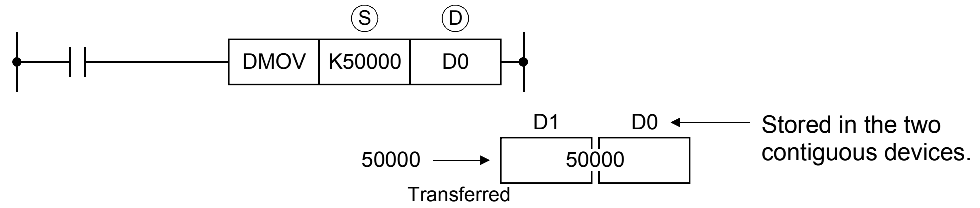


As in the case of handling 16-bit data, the PLC handles a 32-bit negative value in two's complement form. Therefore, the leftmost bit (B15 for 16-bit) is regarded as a sign bit.

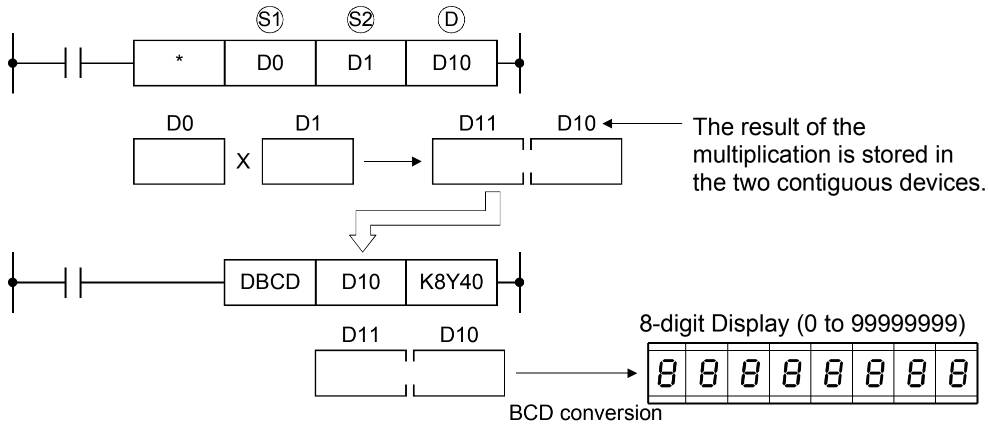


• Determine if the data should be handled in 2-word (32-bit) unit or not depending on the data size.
In the following conditions, 2-word instructions must be used.

1) When the data size exceeds the range (−32768 to 32767) that can be dealt with by 1 word.

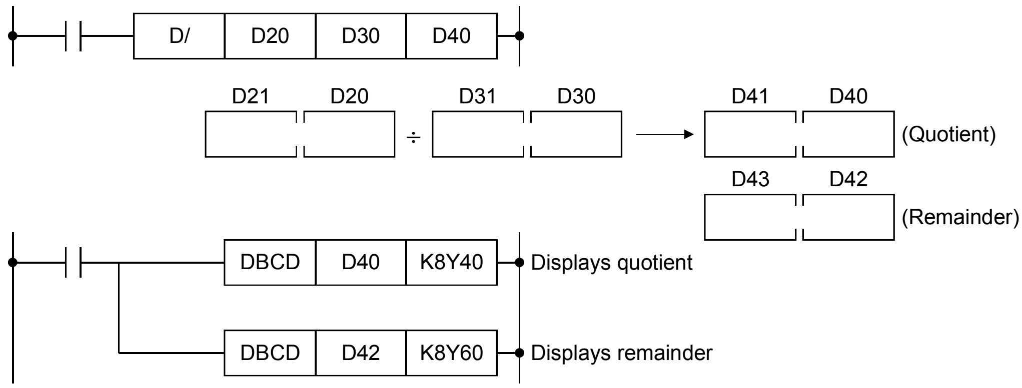


2) When transferring the result of 16-bit multiplication instruction (1-word instruction).



✻ The result of the 32-bit data multiplication will be 64 bits.

3) When utilizing the result of 32-bit division instruction.



Path name	A:\SCHOOL
Project name	QB-19
Program name	MAIN

5.4.4 Calculation examples of multiplication/division that include decimal points (in the case where an arithmetic operation instruction "*" or "/" is used)

(Example 1) Calculation example to determine a circle's perimeter

[Numeric value of the digital switch] × 3.14
(K4X30) (Circle ratio)

→ [Displays integral part] and [Displays decimal fraction part]
(K8Y50) (K2Y48)

• Programming method

Specify the circle ratio as 314 (3.14 x 100), and divide the result by 100 afterward.

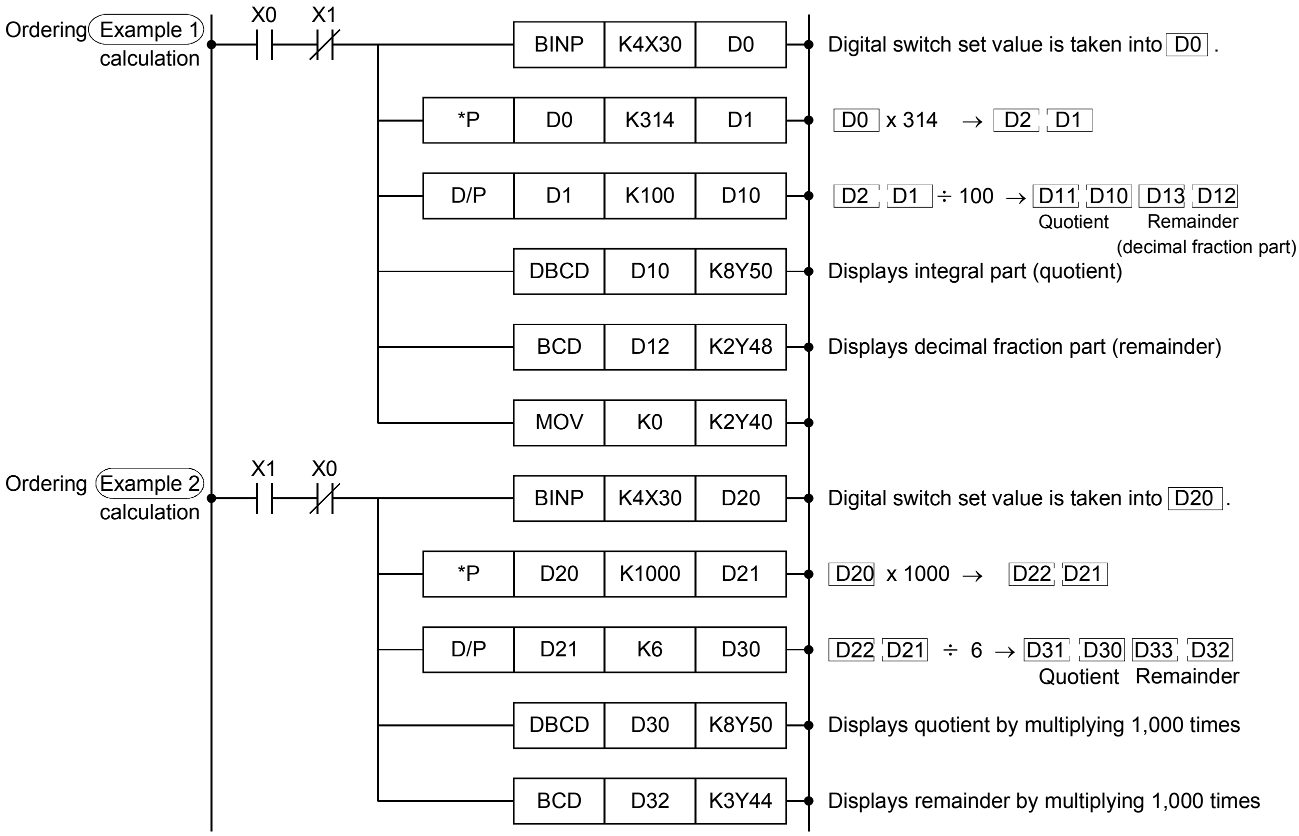
(Example 2) Calculation example to handle decimal places (division example)

[Numeric value of the digital switch] ÷ 0.006
(K4X30)

→ [Displays quotient] and [Displays remainder]
(K8Y50) (K4Y40)

• Programming method

In order to deal with 0.006 as an integer 6, both the dividend and divisor have to be multiplied 1,000 times.



REMARK

QCPU has instructions that can handle actual (floating point) operation data for highly accurate operations.
As long as you use the instructions, you don’t have to pay careful attention concerning the place of the decimal point as shown above.

5.5 Index Register, File Register

5.5.1 How to use index register Z

- The index register (Zn) is used to indirectly specify the device number. The result of an addition of data in the index register and the directly specified device number can be specified as the device number.

Example

D0Z0 → Can be interpreted as $\underbrace{D(0 + Z0)}_{\text{Device number.}}$

For example, if Z0 is 0, the device number becomes D0.
if Z0 is 50, the device number becomes D50.

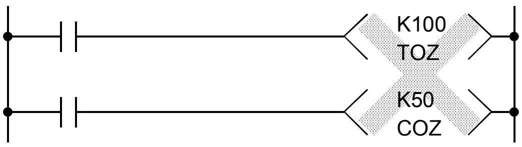
- Z0 to Z15 can be used as the index register.
- The index register (Zn) is a word device that consists of 16 bits. The allowable data size range is -32768 to +32767.
- Index modification can be applied to the following devices.

Bit device……… X, Y, M, L, S, B, F, Jn\X, Jn\Y, Jn\B, Jn\SB (e.g. K4Y40Z)
Word device …... T, C, D, R, W, Jn\W, Jn\SW, Jn\G (e.g. D0Z)
Constant………. K,H (e.g. K100Z)
Pointer………… P

CAUTION

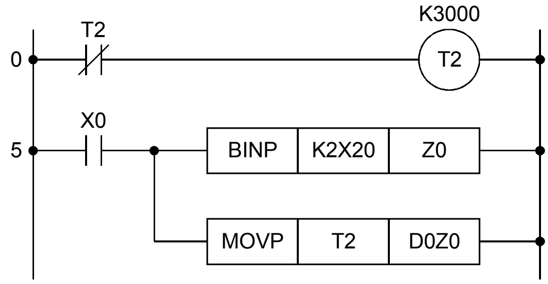
The index register cannot be used for indirectly specifying the timer or the counter coil.

Example

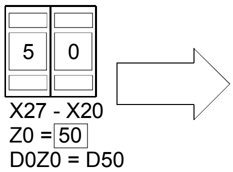


Application Example

- The data is written to the data register whose number is specified with the digital switch.



- Perform the device batch monitoring while conducting a check.
  The operating procedure is the same as the one explained in Section 5.2.1.
  Enter any two-digit number in the digital switch column (X27 to X20) and turn X0 ON.

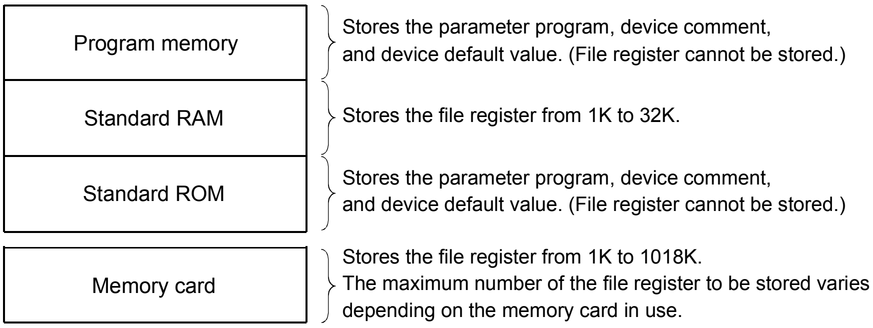


Device	+0	+1	+2	+3	+4	+5	+6	+7
D0	0	0	0	0	0	0	0	0
D8	0	0	0	0	0	0	0	0
D16	0	0	0	0	0	0	0	0
D24	0	0	0	0	0	0	0	0
D32	0	0	0	0	0	0	0	0
D40	0	0	0	0	0	0	0	0
D48	0	0	1124	0	0	0	0	0
D56	0	0	0	0	0	0	0	0

The current value of T2 is transferred to D50 column.

5.5.2 How to use file register R

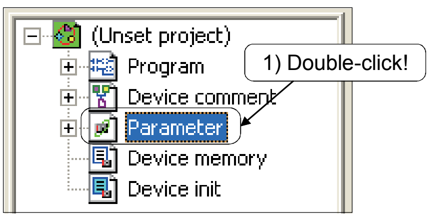
- The file register (R) is a register that consists of 16 bits as well as the data register (D).
- The file register applies to the standard RAM of the QCPU and the memory cards (SRAM card, Flash card).



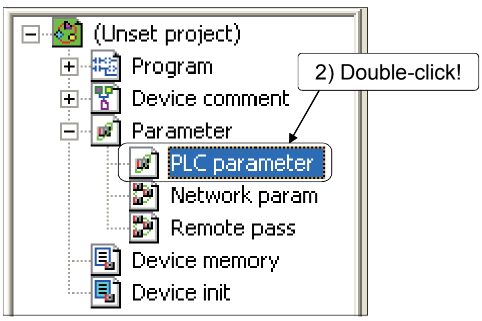
- The data in the file register remains after the reset operation or after the power is turned off. To clear the data, write 0 to the file register by using the MOV(P) instruction as such or write with GX Developer.
- Use [Write to PLC] of GX Developer or a sequence program to write to the standard RAM or to the SRAM card.
- Use [Write to PLC (Flash ROM)] of GX Developer to write to the Flash card.
- Specify the area of the file register in 1K-point (1,024-point) unit using the parameter.

Application Example

- Set 32-K points of the file register R0 to R32767, and use them with the program.
- Follow the steps below to register the parameter.



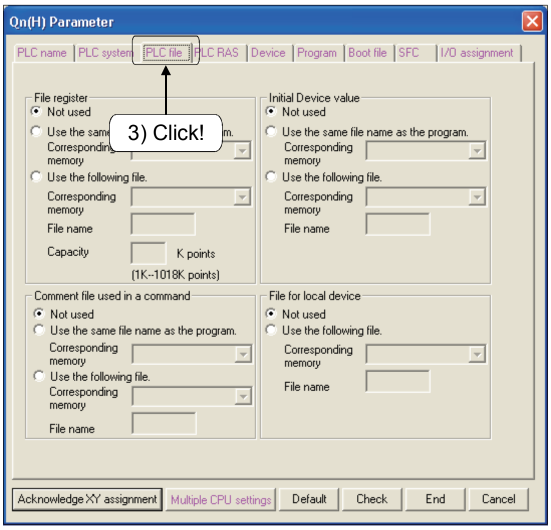
1) Double-click [Parameter] on the project list.



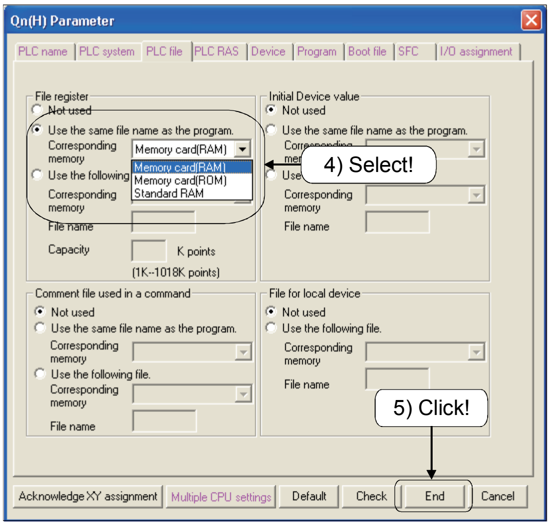
2) [PLC parameter], [Network param], and [Remote pass] appear. Double-click [PLC parameter].

(Continued on the next page)

(Continued from the previous page)

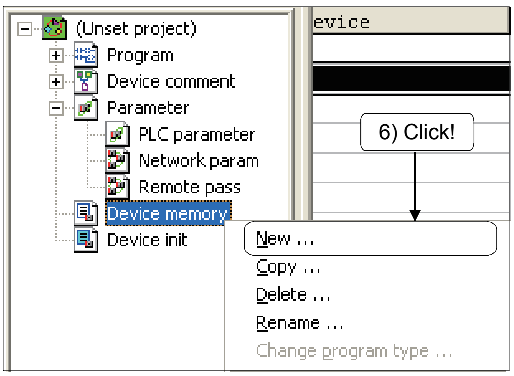


3) The Qn(H) Parameter dialogue box appears. Click the <PLC file> tab on the box.



4) Check the box beside "Use the same file name as the program." of the file register field, and select "Memory card(RAM)" for the corresponding memory.

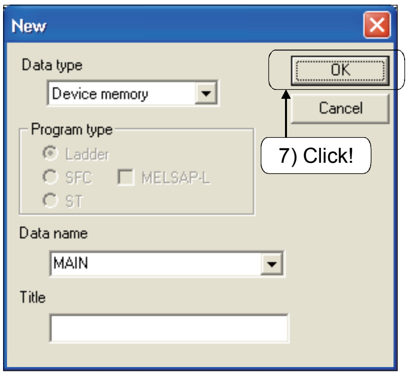
5) When complete, click the End button.



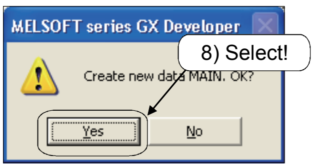
6) Right-click on the "Device memory" from the project data list, and click [New].

(Continued on the next page)

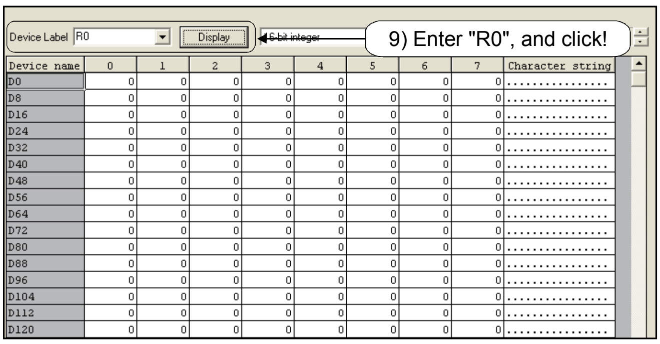
(Continued from the previous page)



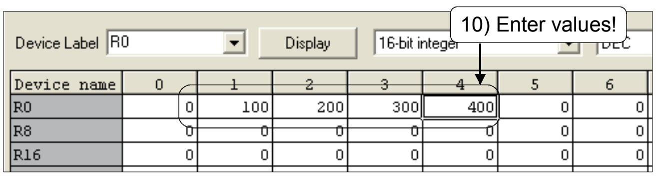
7) Click the OK button.



8) Click the Yes button.



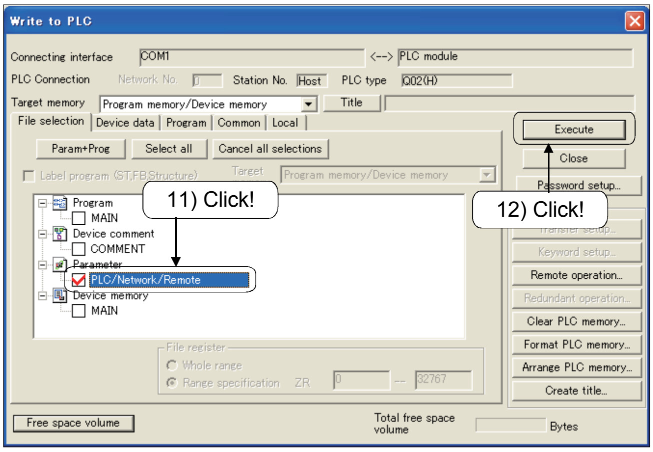
9) The device memory main screen appears.
Enter "R0" in the device edit box, and click the Display button.



10) Check to see that the device name is changed to "R". Enter "100", "200", "300", and "400" in R1, R2, R3, and R4 columns, respectively.
(To clear, enter "0".)

(Continued on the next page)

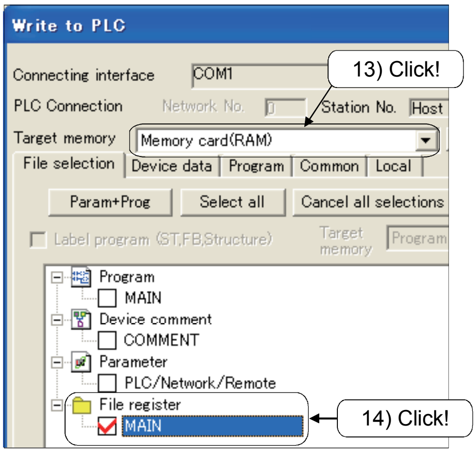
(Continued from the previous page)



11) Select [Online]→ [Write to PLC] to display the Write to PLC dialogue box, and select "parameter" on the File selection tab.

12) Click the Execute button to start writing.

As the demonstration machine has two CPUs installed, it is necessary to write the parameter after setting the parameter for the Multiple CPU. (This procedure is not required when only one CPU is installed.)
For details on the setting procedure, refer to Multiple CPU setting of the parameter in Section 3.2.

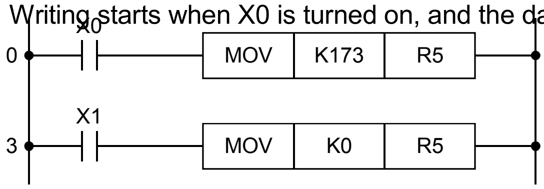


13) Select "memory card (RAM)" for "target memory" on the Write to PLC dialogue box.

14) Click the "MAIN" box under the file register on the File selection tab.
Click the Execute button to start writing.

- Check that 100 to 400 are written in the R1 to R4 columns by using the device batch monitor.
  The operating procedure is the same as the one described in Section 5.2.1.

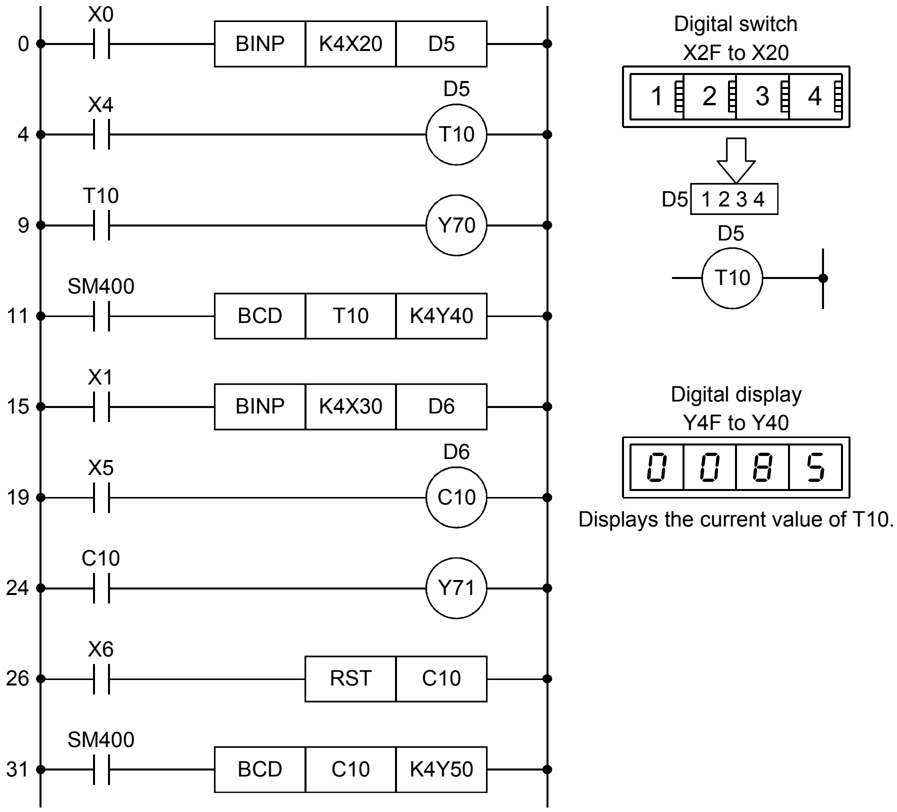
- To write or clear the data of the file register with the program, write the program below.
  For the operating procedure, refer to Section 4.4.
  Writing starts when X0 is turned on, and the data is cleared when X1 is turned on.



The file register data of the SRAM card is retained with the battery. Resetting or turning off the power does not clear the data. To clear, write "0".

5.6 External Setting of the Timer/Counter Set Value, and the External Display of the Current Value

The timer and the counter can be specified either by K (decimal constant) directly or by D (data register) indirectly. Programming as described below allows the set value to be changed with the external digital switch.



• This program is saved in the text FD as | Project name | QTC |
Read it to GX Developer, and write it to the PLC to see if it works.

(Operating Procedure)

The step (1) of the following procedure is the same as that of (Operating Procedure) described in Section 5.3.
The steps (2) to (4) of the following procedure are the same as that of (Operating Procedure) described in Section 4.4.

(1) Read data in FD

(2) Create a program

(3) Write to the PLC

(4) Monitor the ladder

Operation Practice

(1) External setting of the timer set value and display of the current value

- ○ Set the timer set value in the digital switch (X20 to 2F), and turn the X0 switch ON.
- ○ When the X4 switch is turned ON, Y70 turns ON after the set time specified with the digital switch elapses. (e.g. Y70 turns ON after 123.4 s elapsed if you entered $\boxed{1}\boxed{2}\boxed{.3}\boxed{4}$
- ○ The digital display (Y40 to 4F) shows the current value of the timer T10.

(2) External setting of the counter set value and display of the current value

- ○ Set the counter set value for the digital switch (X30 to 3F), and turn the X1 switch ON.
- ○ Turn the X5 switch ON and OFF repeatedly. Y71 turns ON when the number of times that X5 is turned ON reaches the number specified with the digital switch.
- ○ The digital display (Y50 to 5F) shows the current value of the timer C10 (the number of the times that X5 is turned on).
- ○ Turning the X6 switch on clears the counter C10 to 0. If the contact C10 is already turned on, the contact is released.

Path name	A:\SCHOOL
Project name	QTEST5
Program name	MAIN

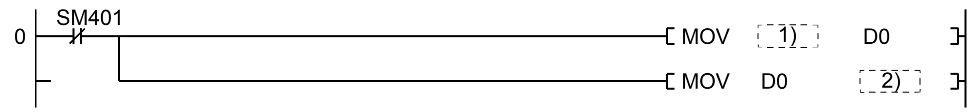
5.7 Practice Question

5.7.1 Practice Question (1) MOV

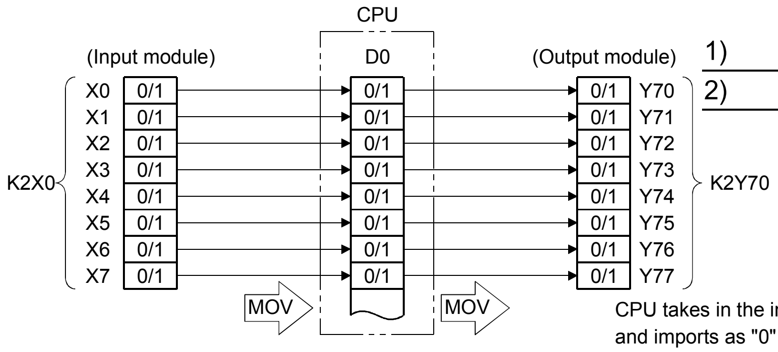
Temporarily send the eight input conditions (X0 to X7) to D0, and then output them toY70 to Y77.
(e.g. Y70 turns ON when X0 is turned ON)

X0 → Y70
X1 → Y71
X2 → Y72
X3 → Y73
X4 → Y74
X5 → Y75
X6 → Y76
X7 → Y77

Fill in the blank square of the program below, create a program with GX Developer and check to see if it works properly with the demonstration machine.



Hint



1) ____________________

2) ____________________

CPU takes in the input signal as "1" when the signal is ON, and imports as "0" when it is OFF.
The output module turns ON upon outputting "1", and turns OFF upon outputting "0".

Comparison

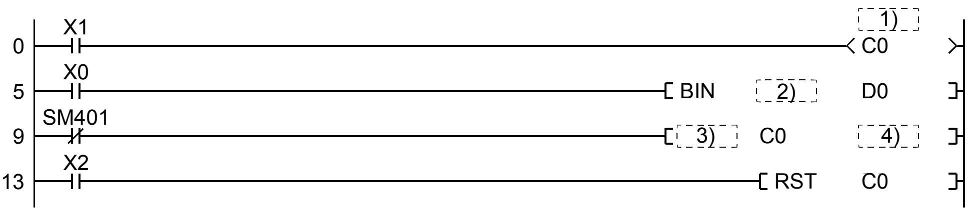
The program created by the sequence instruction without MOV instruction is shown on the next page.

Path name	A:\SCHOOL
Project name	QTEST6
Program name	MAIN

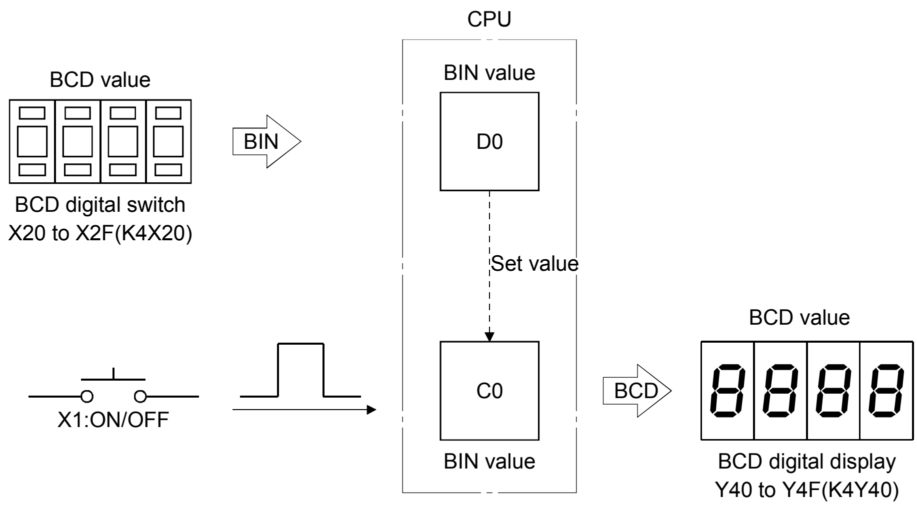
5.7.2 Practice Question (2) BIN,BCD conversion

Show the number of times that X1 is turned ON on the display connected to Y40 to Y4F in BCD. The counter (C0) set value should be input with the digital switch (X20 to X2F) after X0 is turned ON.

Fill in the blank square of the program below, create a program with GX Developer and check to see if it works properly with the demonstration machine.



Hint



1) ____________________

2) ____________________

3) ____________________

4) ____________________

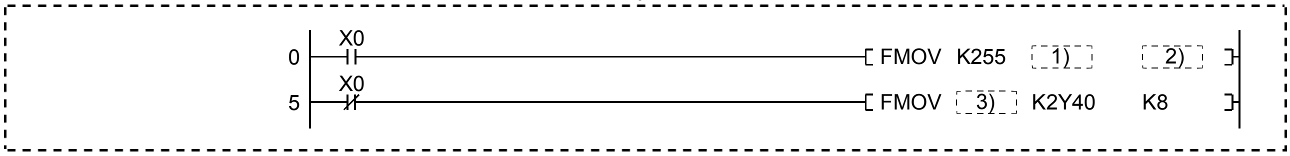
Path name	A:\SCHOOL
Project name	QTEST7
Program name	MAIN

5.7.3 Practice Question (3) FMOV

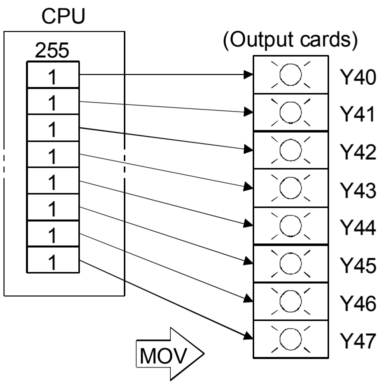
Create a program that works as follows:

1) 64 outputs (Y40 to Y7F) turn ON when X0 is turned ON.
2) 64 outputs (Y40 to Y7F) turn OFF when X0 is turned OFF.

Fill in the blank square of the program below, create a program with GX Developer and check to see if it works properly with the demonstration machine.



Hint



The constant should be output from the CPU in binary form.

Outputting 255 from Y40, for example.

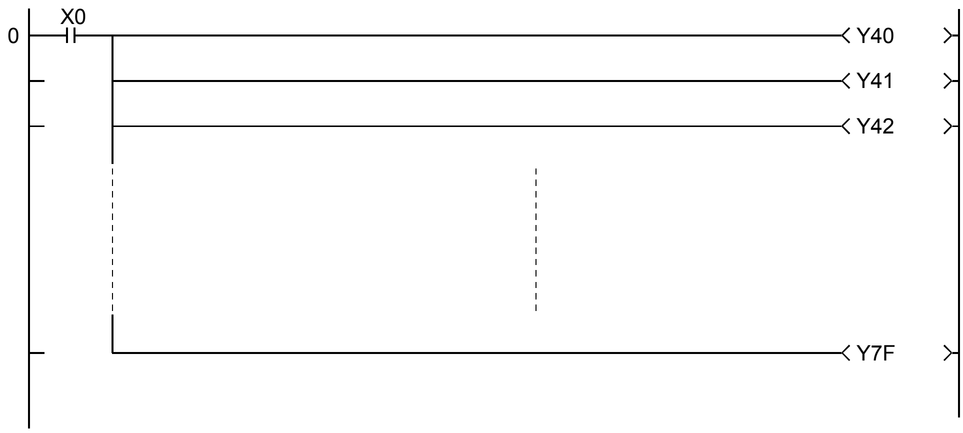
128	64	32	16	8	4	2	1	
1	1	1	1	1	1	1	1	= 1 + 2 + 4 + 8 + 16 + 32 + 64 + 128 = 255
Y47	Y46	Y45	Y44	Y43	Y42	Y41	Y40	

The question deals with 64-point (Y40 to Y7F) outputs.
How many blocks are required for 255 on an output unit basis?

1) ____________________
2) ____________________
3) ____________________

Comparison

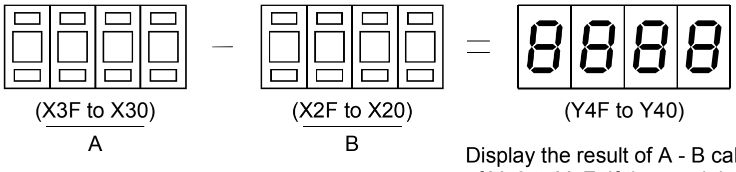
The program created by the sequence instruction without FMOV instruction is shown below. The number of steps used is 130.



Path name	A:\SCHOOL
Project name	QTEST8
Program name	MAIN

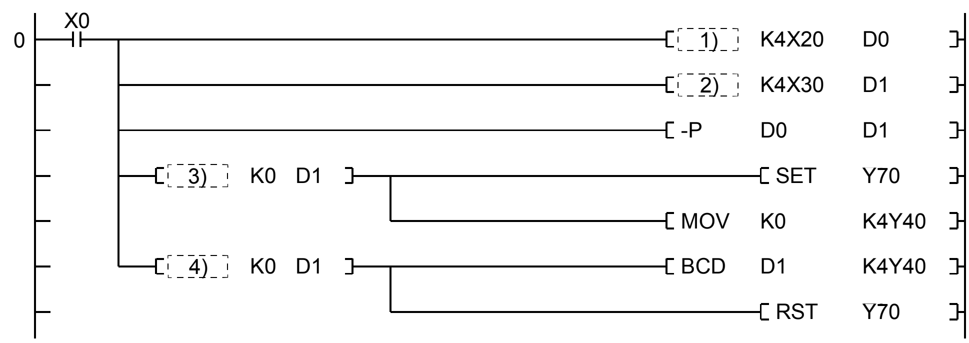
5.7.4 Practice Question (4) Comparison instruction

Use the two BCD digital switches, perform (A-B) operation and show the result on the BCD digital display (Y40 to Y4F).



Display the result of A - B calculation on the BCD display of Y40 to Y4F. If the result is a negative number, however, make sure that the display turns to 0 and the LED of Y70 turns on.

Fill in the blank square of the program below and check to see if it works properly with the demonstration machine.



Hint

The constant should be output from the CPU in binary form.

–	D0	D1	= D1 - D0 → D1

1) ____________________

2) ____________________

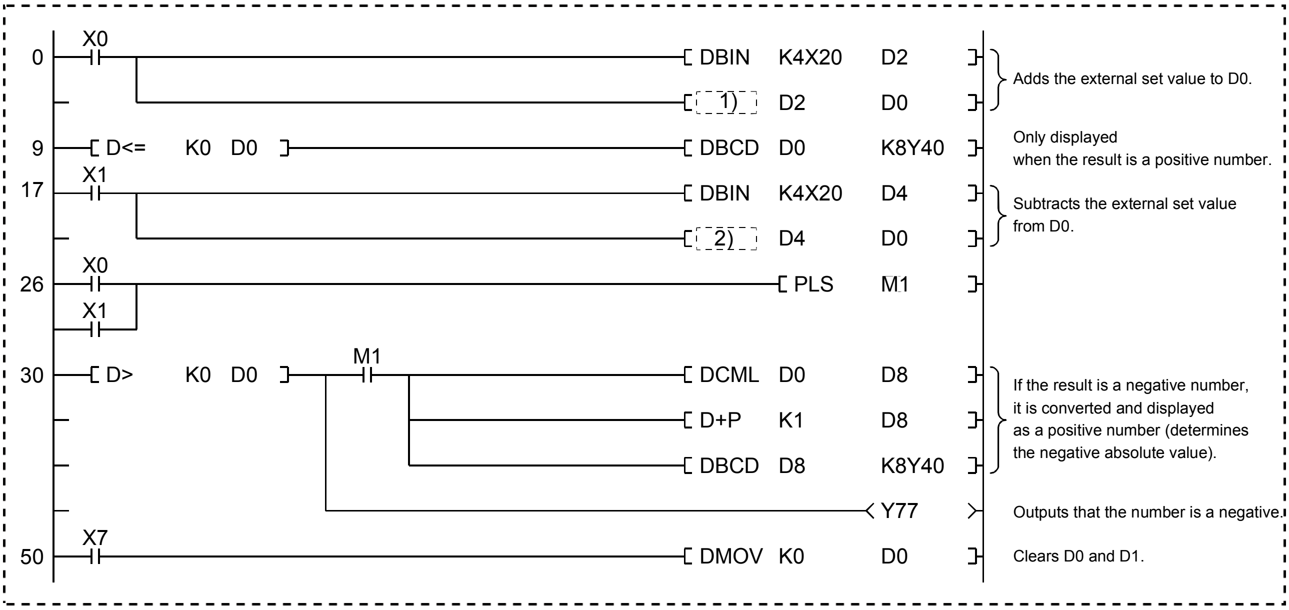
3) ____________________

4) ____________________

Path name	A:\SCHOOL
Project name	QTEST9
Program name	MAIN

5.7.5 Practice Question (5) +,-

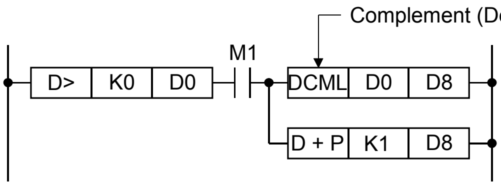
When X0 turns ON, create a program that imports the value specified by the digital switch (X20 to X2F) into D3 and D2 (32-bit data), adds them to D1 and D0 and shows the result on the display (Y40 to Y5F).
Also, when X1 turns ON, the program should import the value specified by the digital switch (X20 to X2F) into D5 and D4, subtract them from D1 and D0 and show the result on the display. If the result becomes a negative number, the program should turn Y77 ON and show the absolute value on the display in two's complement form.
Fill in the blank square of the program below and check to see if it works properly with the demonstration machine.



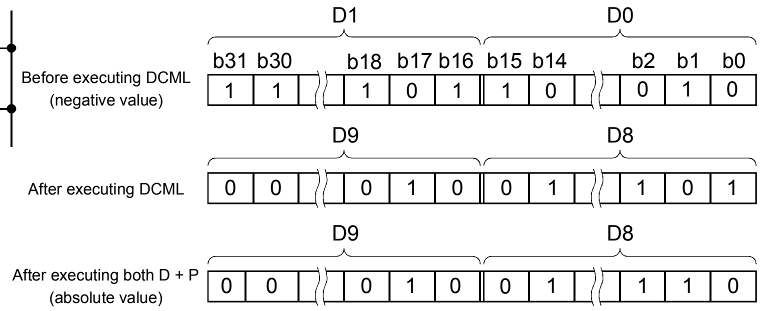
1) ______________________

2) ______________________

Reference



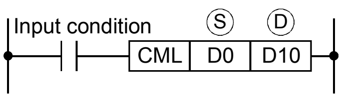
The absolute value is determined by calculating two's complement of D0 and D2 (32-bit data).



If 16-bit data is a negative number, it is changed to the absolute value using NEG (two's complement) instruction.

REMARK

The CML instruction inverts Ⓢ bit pattern and transfers it to Ⓓ when the input condition is turned ON.



Path name	A:\SCHOOL
Project name	QTEST10
Program name	MAIN

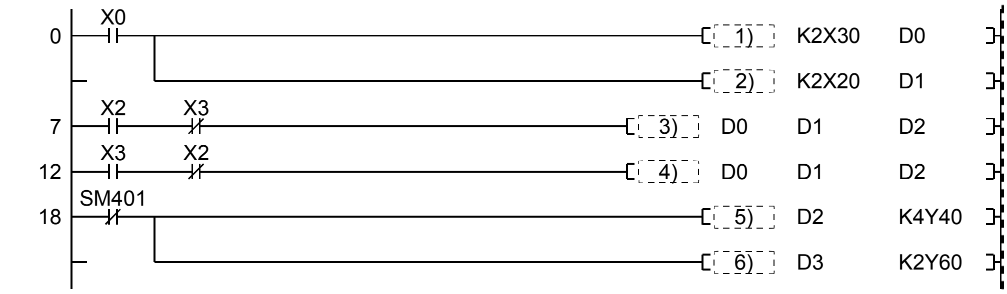
5.7.6 Practice Question (6) *, /

The multiplication/division data can be set by turning X0 ON. When X2 is turned ON, the BIN multiplication of the value specified with the digital switch X20 to X27 and X30 to X37 is performed, and a division is performed when X3 is turned ON. The result of the multiplication or the quotient of the division is displayed on the BCD display of Y40 to Y4F, and the remainder is displayed on the BCD display of Y60 to Y67.

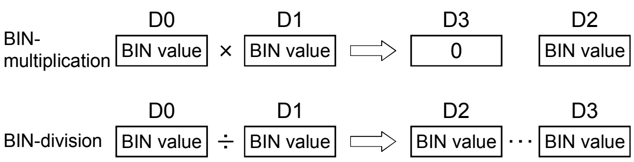
(X30 to X37) × (X20 to X27) ⇨ (Y40 to Y4F)

(X30 to X37) ÷ (X20 to X27) ⇨ (Y40 to Y4F)……(Y60 to Y67)

Fill in the blank square of the program below, create a program with GX Developer and check to see if it works properly with the demonstration machine.



Hint



1) ____________________

2) ____________________

3) ____________________

4) ____________________

5) ____________________

6) ____________________

Path name	A:\SCHOOL
Project name	QTEST11
Program name	MAIN

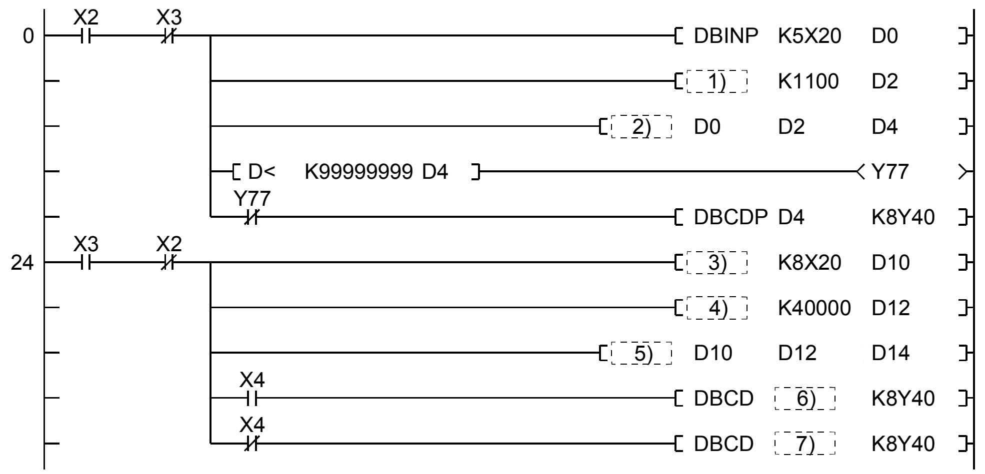
5.7.7 Practice Question (7) D*, D/

Create a program that performs the BIN-multiplication of the value specified with the 5-digit digital switch (X20 to X33) by 1100 when X2 is turned ON. If the result is less than 99999999, show the result on the 8-digit display (Y40 to Y5F).
The program should perform the BIN-division of the value specified with the 8-digit digital switch (X20 to X3F) by 40000 when X3 is turned ON. If X4 is ON, show the quotient of the result on the 8-digit display (Y40 to Y5F). If X4 is OFF, show the remainder on the same display.

(X20 to X33)×1100 ⇨ (Y40 to Y5F)

(X20 to X3F)÷40000 ⇨ Quotient (Y40 to Y5F) X4 ON
Remainder (Y40 to Y5F) X4 OFF

Fill in the blank square of the program below, create a program with GX Developer and check to see if it works properly with the demonstration machine.



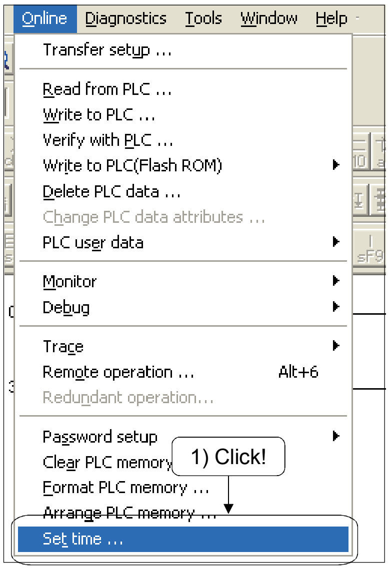
1) ____________________
2) ____________________
3) ____________________
4) ____________________
5) ____________________
6) ____________________
7) ____________________

Answers for the practice questions in Chapter 5

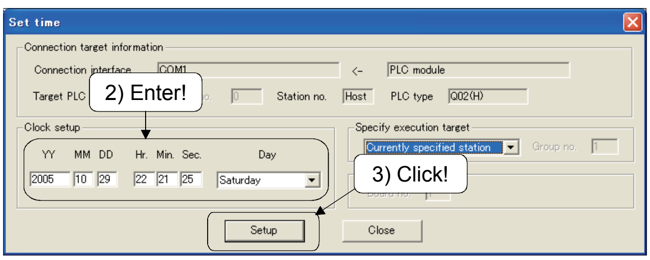
Question		Answer
5	1)	K2 X0
	2)	K2 Y70
6	1)	D0
	2)	K4 X20
	3)	BCD
	4)	K4 Y40
7	1)	K2 Y40
	2)	K8
	3)	K0
8	1)	BIN P
	2)	BIN P
	3)	<
	4)	<=
9	1)	D+P
	2)	D−P
10	1)	BIN
	2)	BIN
	3)	*P
	4)	/P
	5)	BCD
	6)	BCD
11	1)	DMOV P
	2)	D*P
	3)	DBIN P
	4)	DMOV P
	5)	D/P
	6)	D14
	7)	D16

6.1 Clock Function

With the clock function, the following items can be set in the clock elements incorporated in the CPU; year, month, day, hour, minute, second, day of the week.
To enable the clock function, use GX Developer or a program.
To set or read the clock data, use GX Developer.



1) Click [Online] and then [Set time] to display the Set time dialog box.



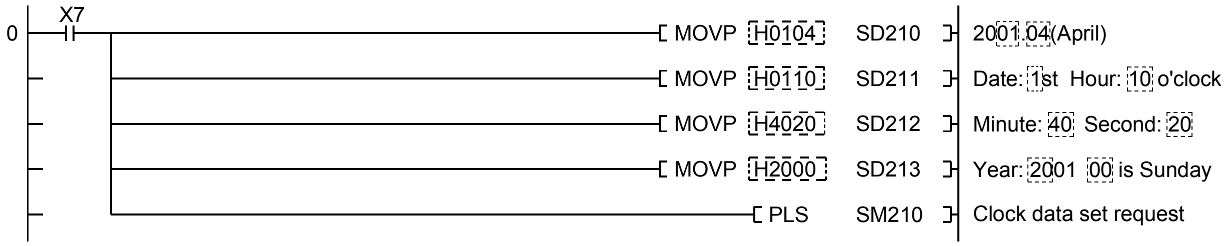
2) Input year, month, day, hour, minute and second, and select the day of the week in the Set time dialog box.

3) Click the $\boxed{\text{Setup}}$ button.

Reference

The following shows an example of using the clock function with a program. (Project name QEX13)

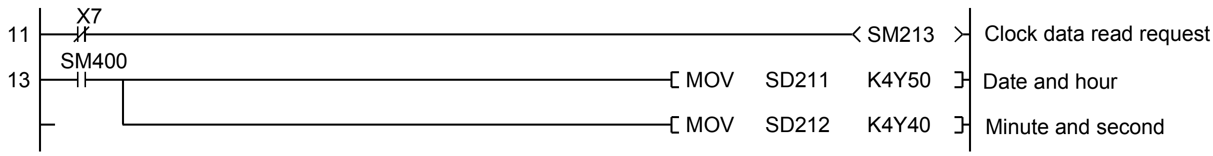
• Writing clock data



Settable items are items in

Set time a few minutes ahead of actual time, and then write it to the PLC. When actual time has reached the set time, turn on the input switch X7. Then the clock is activated.

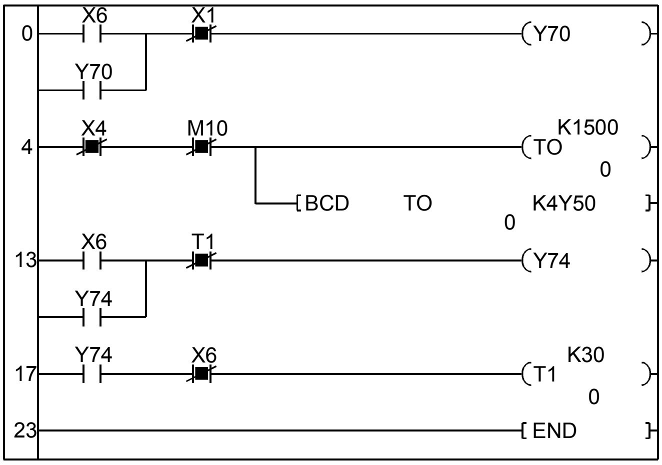
• Reading clock data



Turn off X07 to read time.

6.2 Test Function at Online

Exercise this function with the program | Project name | QEX14 |
As preparation, follow the procedure below.



For details on the operation method, refer to Chapter 2.

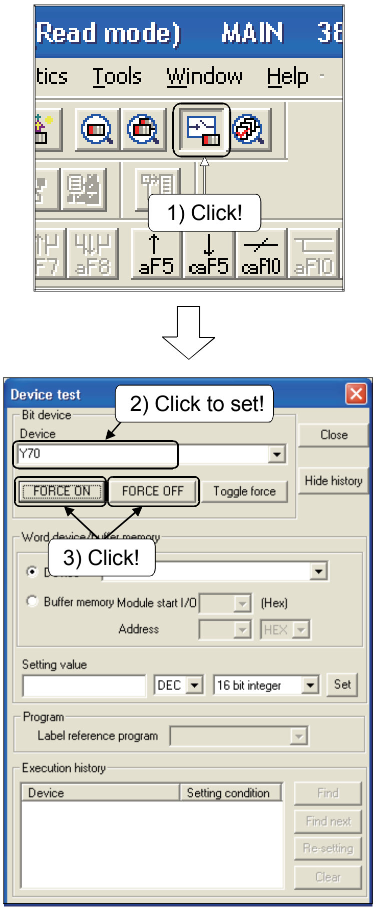
1) Read the project from the user floppy disk (FD) using GX Developer.

2) Write the parameter and program of the read project to the CPU (PLC).
(The CPU must be at stop to do this.)

3) Set GX Developer to the monitor mode.

4) Confirm a program displayed in the screen.

6.2.1 Turning device "Y" ON/OFF forcibly



Stop the CPU before proceeding with the procedure below.

1) Click the button on the toolbar.

2) The Device test dialog box appears. Click "Device" and input "Y70" in the list box.

3) Click the [FORCE ON]/[FORCE OFF] button to forcibly turn "Y70" on/off.

Check Using Demonstration Machine

1) Confirm that the content displayed on the execution history area switches between ON and OFF according to clicks of the [FORCE ON]/[FORCE OFF] button. Also, confirm Y70's LED on the demonstration machine switches between on and off according to the same operation.

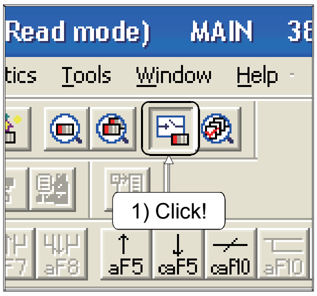
NOTE

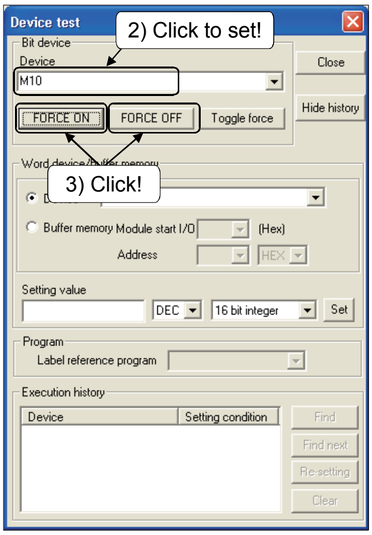
If the CPU is in a RUN state, operation results of programs are given priority. For this reason, stop the CPU first to make confirmation with the demonstration machine.

POINT

To execute setting/resetting of contacts, changing a current value of word devices, and forced outputs, it is also possible to use the test function while GX Developer is monitoring ladders.
In the ladder-monitoring screen of GX Developer, double-click the contact or press [Enter] key while holding the [Shift] key. This forcibly switches the contact between closed and open.
To display the dialog box for changing current values, double-click the word device or press [Enter] key while holding the [Shift] key in the ladder-monitoring screen of GX Developer.

6.2.2 Setting and resetting device "M"





Activate the CPU before proceeding with the procedure below.

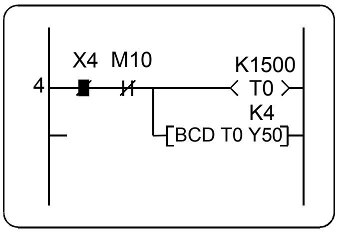
1) Click the button on the toolbar.

2) The Device test dialog box appears. Click "Device" and input "M10" in the list box.

3) Click the FORCE ON / FORCE OFF button to set or reset "M10".

Check Using Demonstration Machine

Turn X4 off and check the following items.



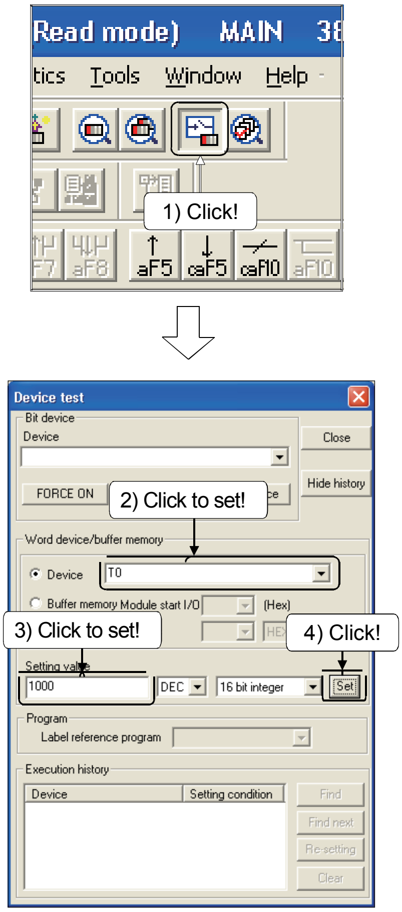
(Monitoring screen with M10 set)

1) When M10 is set, M10 is de-energized and the current value of the timer T0 is cleared to 0.
Confirm that the value on the digital display (Y50-Y5F) stops changing.

2) When M10 is reset, M10 is energized and the timer T0 starts counting from 0. This count value increases by 10 at a second.
Confirm that the value on the display (Y50-Y5F) increases by 10 at a second.

POINT
With the same procedure, bit devices other than internal relays (M) can also be set and reset forcibly.

6.2.3 Changing a current value of device "T"



Activate the CPU before proceeding with the procedure below.

1) Click the button on the toolbar.

2) The Device test dialog box appears. Click "Device" in "Word device/buffer memory" area and input "T0" in the list box.

3) Click "Setting value" and input "1000" in the list box.

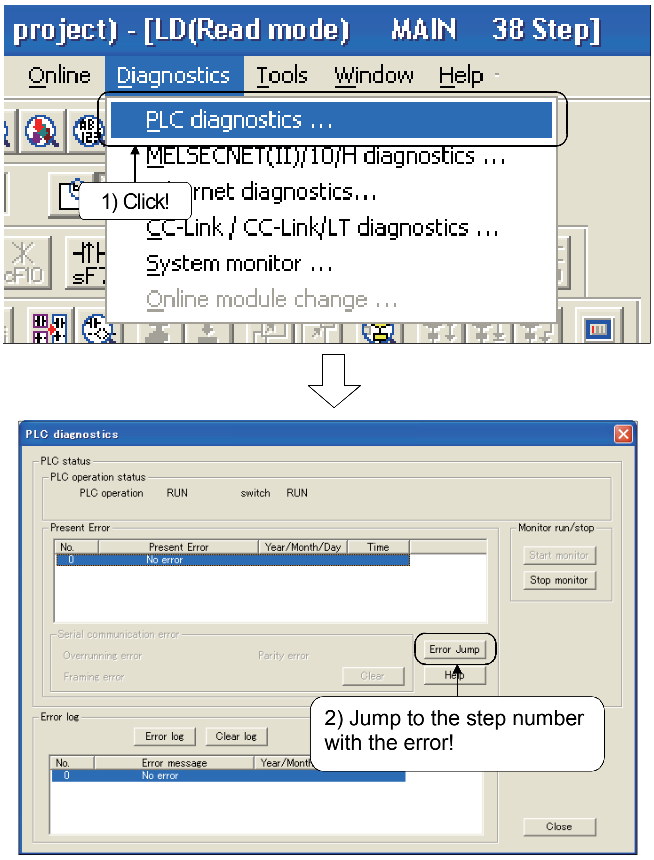
4) After the setting is completed, click the Set button to forcibly change the current value of T0 to 1000.

Check Using Demonstration Machine

1) Confirm that the value on the digital display (Y50-Y5F) changes to 1000 by pressing ⏎.

POINT
With the same procedure, word devices other than timers (T) can also be changed in their current values.

6.2.4 Reading error steps



Activate the CPU before proceeding with the procedure below.

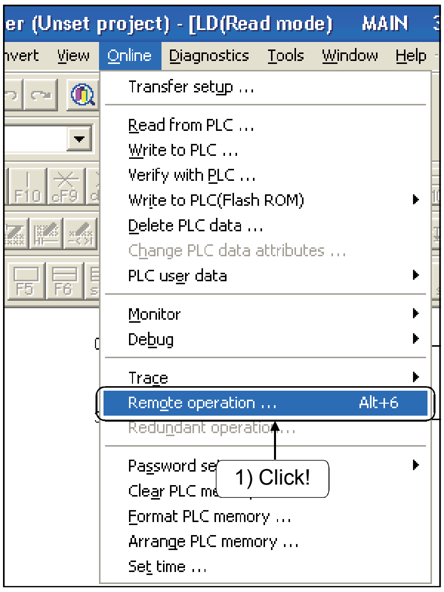
1) Click [Diagnostics] → [PLC diagnostics].

2) The PLC diagnostics dialog box appears.
Click the Error Jump button to jump to the sequence program step number where the highlighted error occurred.

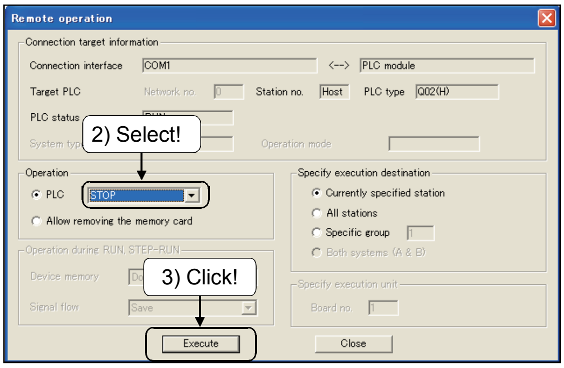
- An error number is displayed if an error was found.
- "No error" is displayed if no error was found.

6.2.5 Remote RUN/STOP

Activate the CPU before proceeding with the procedure below.

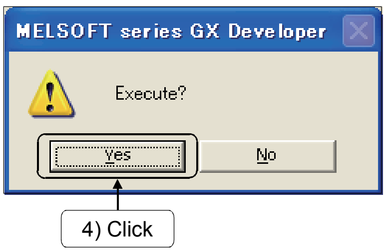


1) Click [Online] → [Remote operation].



2) The Remote operation dialog box appears. Select "STOP" in the list box in the Operation area.

3) After the setting is completed, click the Execute button.

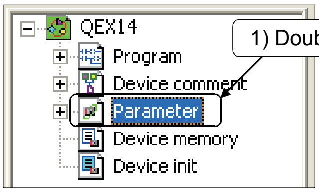


4) The message saying "Execute?" appears. Click the Yes button.

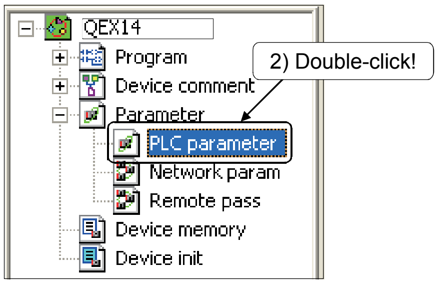
The CPU stops.

5) Select "RUN" in step 2), and perform step 2) to 4) again.

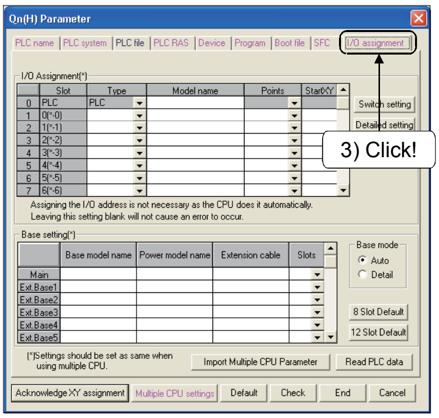
The CPU, which was set to STOP in the above procedure, enters a RUN state again.



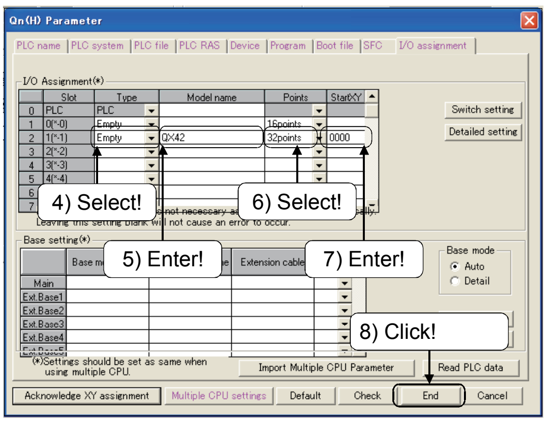
1) Double-click "Parameter" in the project list.



2) "PLC parameter", "Network param", and "Remote pass" are displayed. Double-click "PLC parameter".



3) The Qn(H) Parameter dialog box appears. Click the "I/O assignment" tab.



4) Select "Empty" in the list box of "Type" column.
5) Input "QX42" in the "Model name" column.
6) Select "32points" in the list box of "Points" column.
7) Input "0000" in the "StartXY" column.
8) After the setting is completed, click the END button.

After this exercise is finished, initialize the settings by the following procedure.
1) Click the Default button in the Qn(H) Parameter dialog box to initialize the parameter settings.
2) Click on the toolbar to open the Write to PLC dialog box, and then write only the parameters to the CPU. (Do the same for the second CPU.)

(Check Using Demonstration Machine)

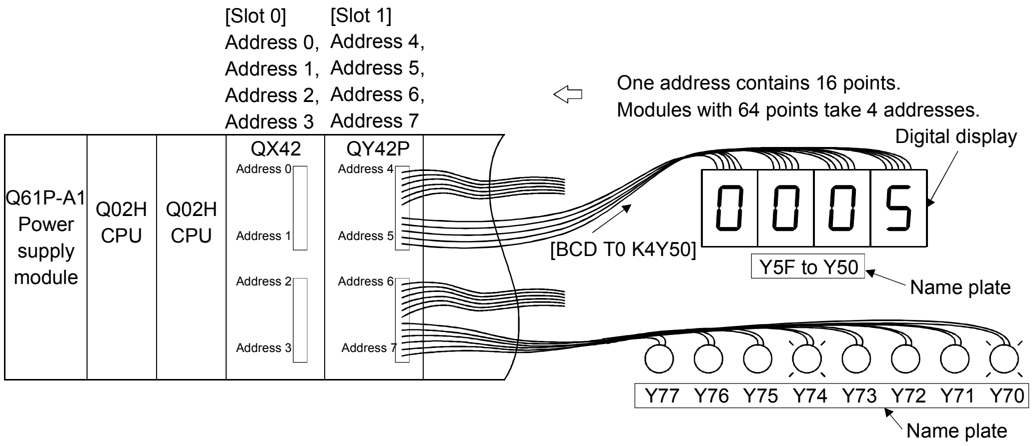
Stop the CPU and click [icon] on the toolbar.

The Write to PLC dialog box opens. Click the parameter of the target data, and then click the [Select] button → [Execute] button sequentially to write only the parameter to the CPU. After that, activate the CPU and confirm the following.

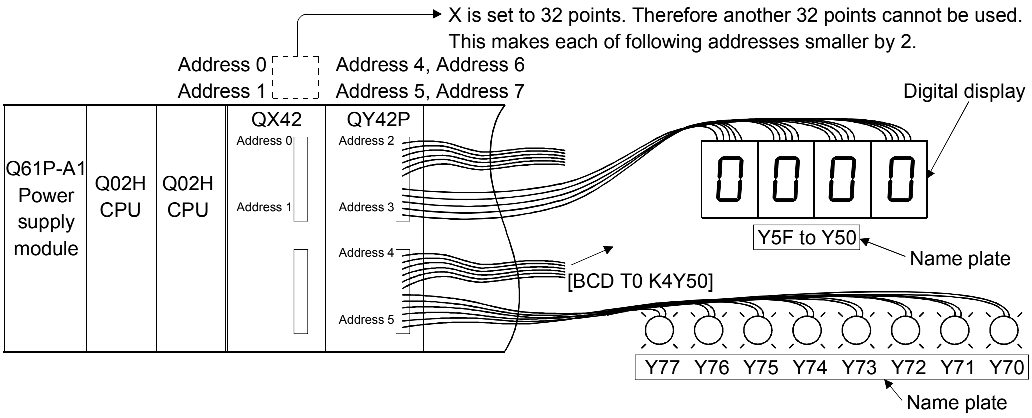
Perform the same parameter settings to second CPU.
(This is not necessary if there is only one CPU incorporated.)
For how to write parameters to second CPU, refer to Section 2.6.

1) A current value of the timer T0 disappears from the screen of the digital display (Y50 to Y5F). Instead, the LEDs of Y70 to Y77 start flashing. These flashes continue until the value reaches the set device value.
2) Outputting to Y70 and Y74 with X6 turned on does not m-+ake the LEDs of Y70 and Y74 flash.

[I/O numbers before forced assignment]



[I/O numbers after forced assignment]

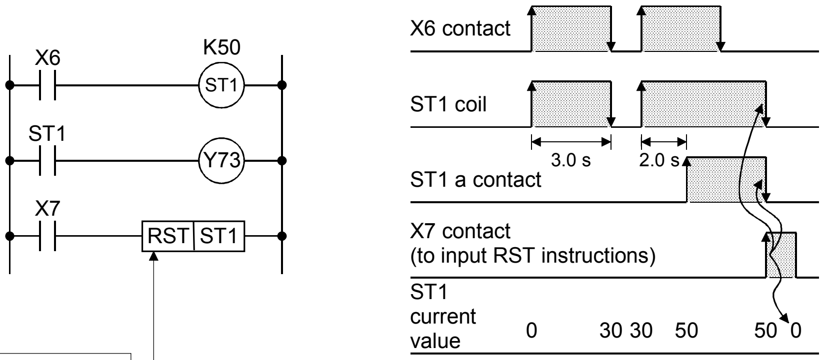


POINT

- Note that address 7 has been replaced with address 5. This means that a current value of the timer T0 is output to the newly assigned address 5, and LEDs of Y70 to Y77 flash as they are connected to the address 5.
- Results of outputting to Y70 or Y74 cannot be confirmed on any displays as address 7 for output modules no longer exists.
  To display normally, change the device number from K4Y50 to K4Y30 and from Y70-Y77 to Y50-Y57.

6.4 Using Retentive Timers

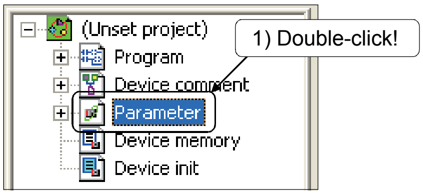
When an input condition is turned on, the coil is energized. Then the value of a retentive timer starts increasing. Once a current value has reached the set value, a retentive timer goes time out and its contact turns on. If the input condition is turned off during that increase, the coil is de-energized but the current value is kept. To restart the increase, which means to accumulate values, turn input conditions on again to re-energize the coil.



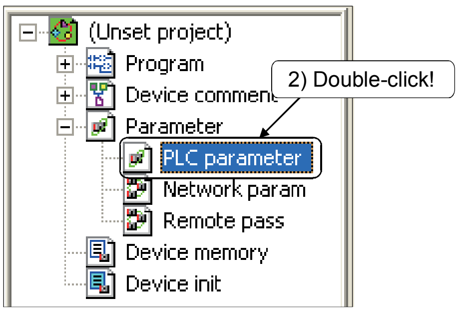
Before using as a retentive timer, specify the number of points in parameters.

An RTS instruction must be used to turn off the contact and clear the current value after the retentive timer goes time out.

In the example operation below, the retentive timer is set to ST0 to ST31.



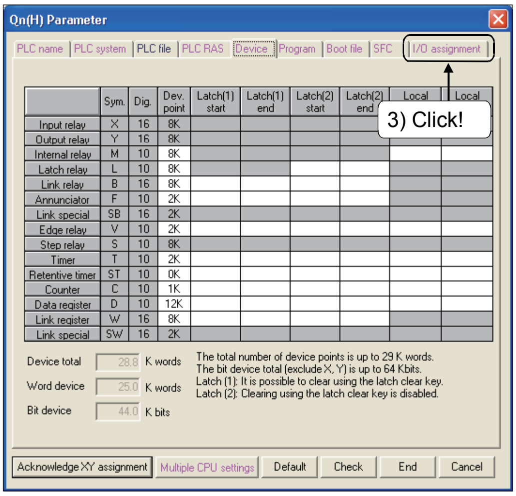
1) Double-click "Parameter" in the project list.



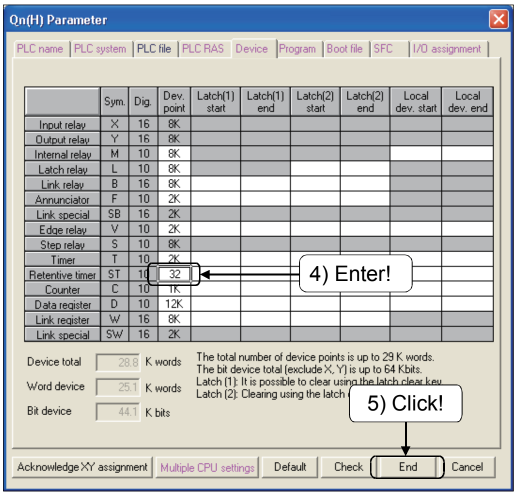
2) "PLC parameter", "Network param", and "Remote pass" are displayed. Double-click "PLC parameter".

(To next page)

(From previous page)



3) The Qn(H) Parameter dialog box appears.
Click the "I/O assignment" tab.



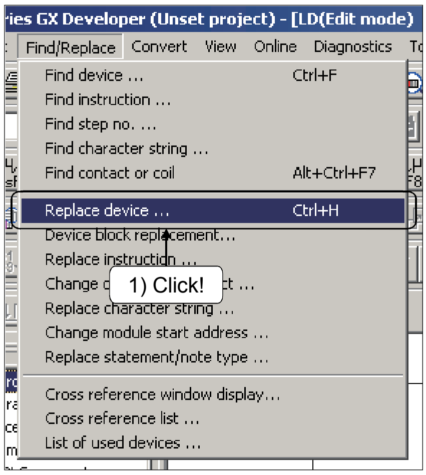
4) Click "Dev. point" in the "Retentive timer" row, and input "32" in there.

5) After the setting is completed, click the End button.

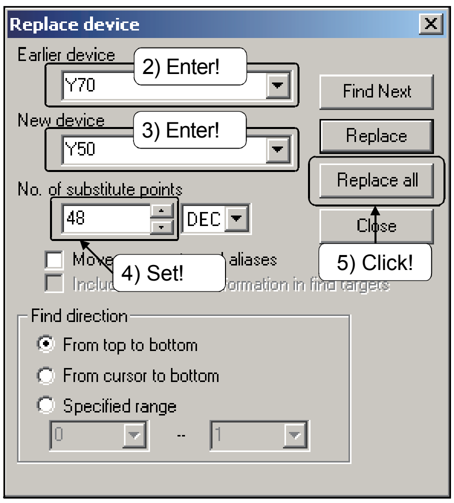
6.5 Batch Replacement of Devices

6.5.1 Batch replacement of device numbers

In the example operation below, Y70 to Y9F (48 in total) are replaced with Y50 to Y7F (48 in total) in batch.



1) Click [Find/Replace] → [Replace device].

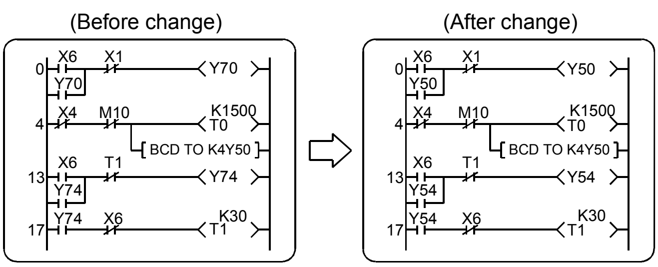


2) The Replace device dialog box appears. Click "Earlier device" and input "Y70" in the list box.

3) Click "New device" and input "Y50" in the list box.

4) Click "No. of substitute points" and set "48" in the spin box.

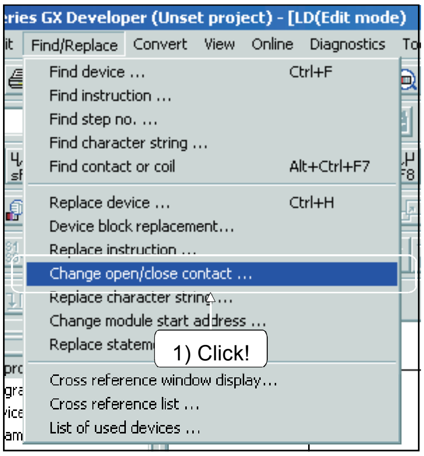
5) After the setting is completed, click the Replace button.



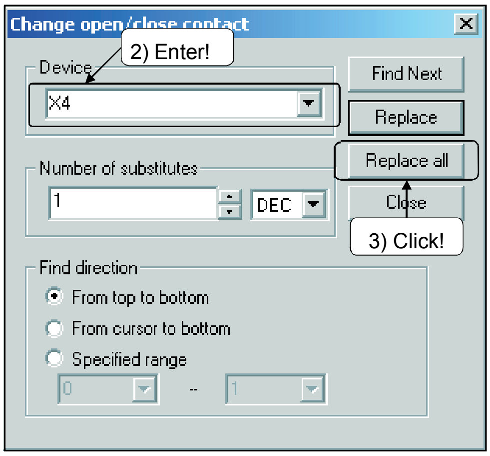
6) Confirm that the target device numbers have been properly replaced.

6.5.2 Batch switching of specified devices between normally open and normally close

Follow the procedure below to switch specified devices all at once between normally open and normally close.

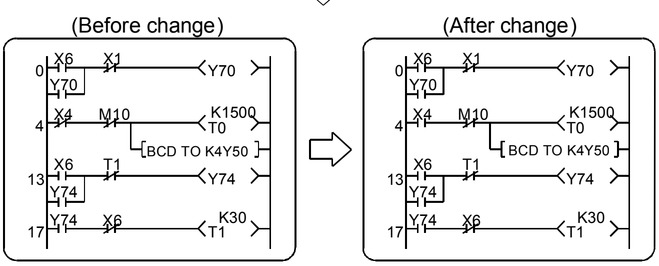


1) Click [Find/Replace] → [Change open/close contact] to select.



2) The Change open/close connect appears.
Click "Device" and input "X4" in the list box.

3) After the setting is completed, click the Replace all button.



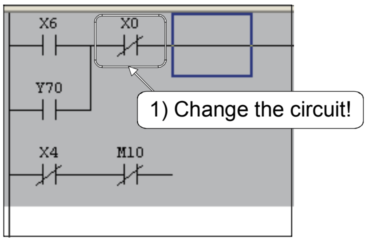
4) Confirm that normally open have been switched to B contacts and vice versa.

NOTE
Before exercising the contents in Section 6.6 after this section, be sure to store the program in a personal computer to a CPU. For how to write to a CPU, refer to Section 2.5.

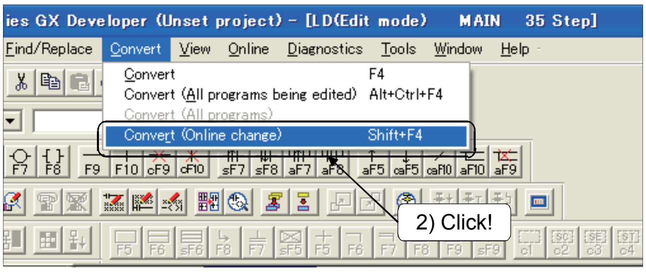
6.6 Write During RUN

This function allows programs to be written to CPUs that are currently running.

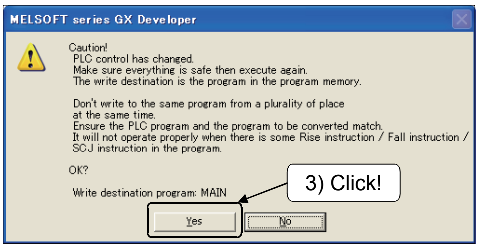
Activate the CPU before proceeding with the procedure below.



1) Change the ladder.
(In the example here, "X1" is changed to "X0".)



2) After the change is made, click "Convert" → "Convert (Online change)".



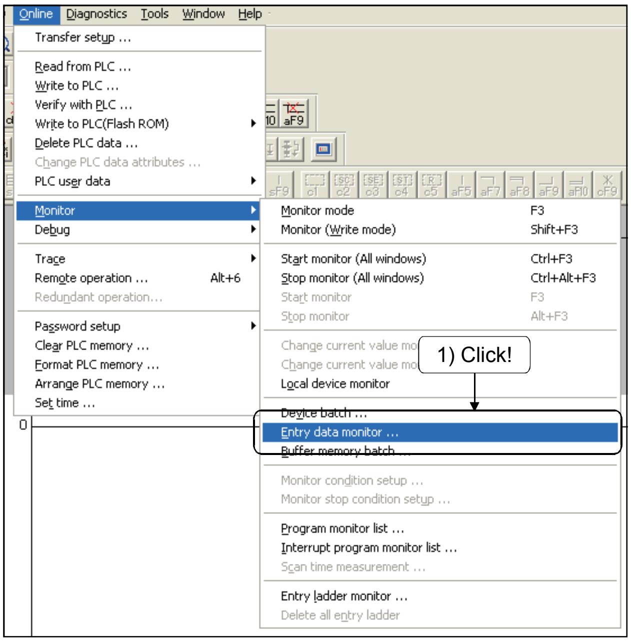
3) The "Caution" ears. Read through the message, and click the Yes button if you agree with it.

4) The message saying "RUN write processing has completed" appears. Click the OK button.

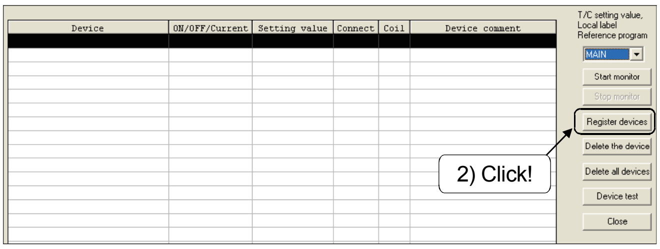
NOTE
Note that to execute the write during RUN operation, the PLC CPU and GX Developer must share an identical program before the modification. So, if it is not for sure that those two programs are identical, verify them before modifying with GX Developer and executing write during RUN.

6.7 Registering Devices

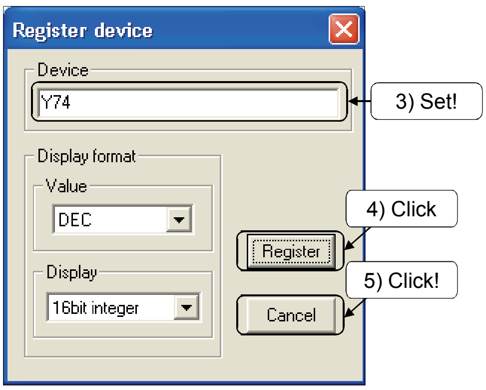
Various devices, including ones in separate locations on the ladder, can be monitored all together in one screen.



1) Click [Online], [Monitor], and [Entry data monitor] sequentially.



2) The monitor screen for device registration appears. Click the Register devices button.



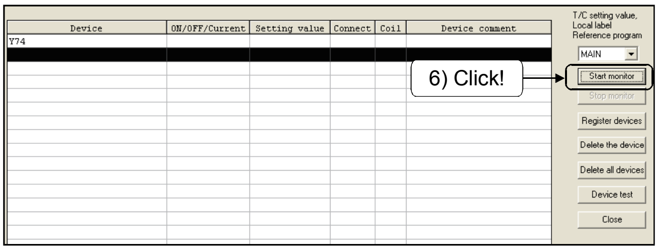
3) The Register device dialog box appears. Click "Device" and input "Y74" in the list box.

4) After the setting is completed, click the Register button.

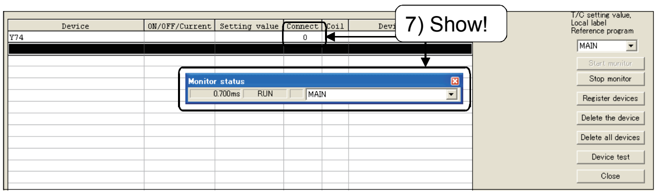
5) Click the Cancel button to close the dialog box.

(To next page)

(From previous page)



6) Click the Start monitor button.



7) The Monitor status dialog box appears and the ON/OFF/Current value of the device is displayed.

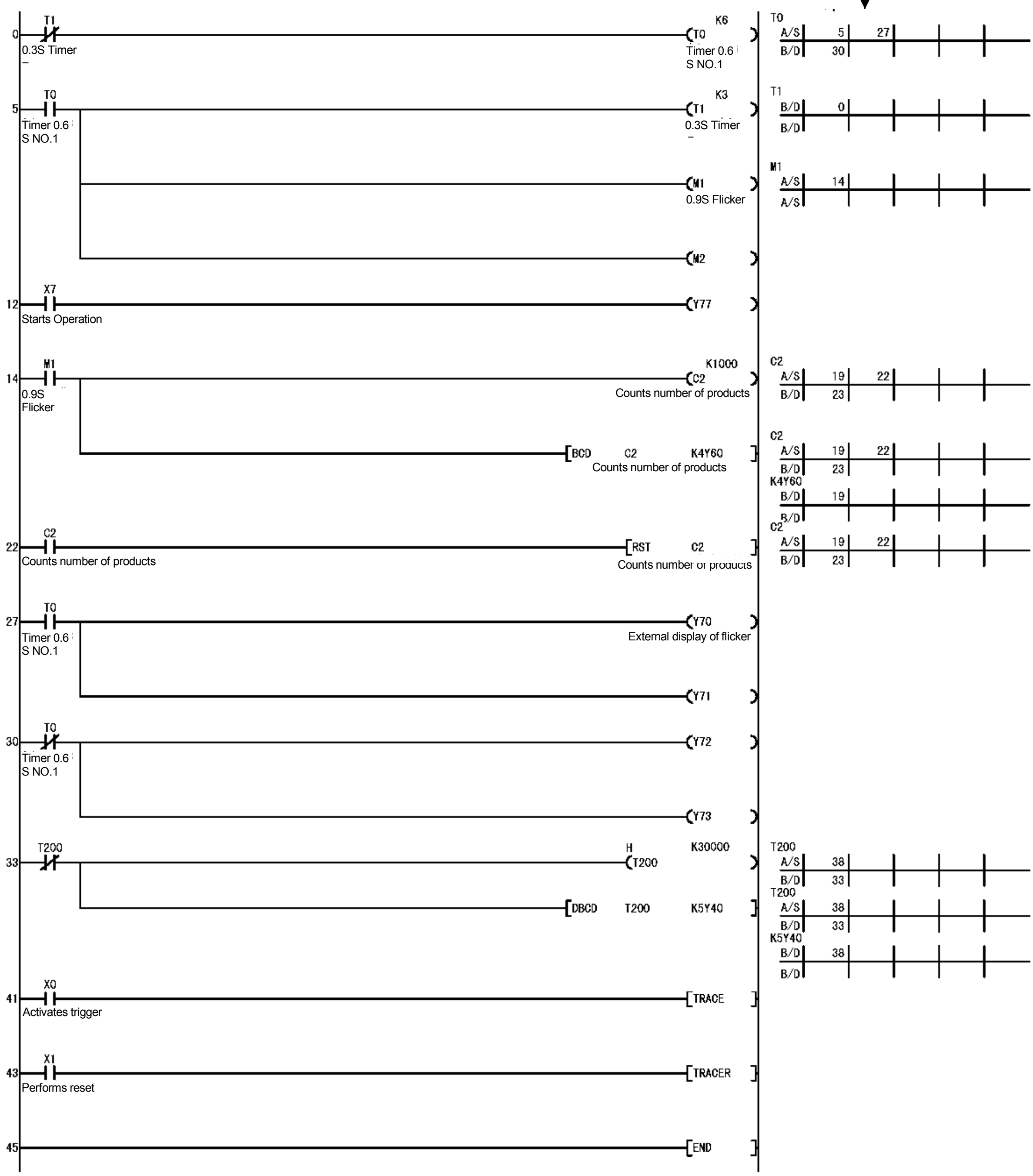
REMARK
To remove registered devices, click the Delete the device button.

6.8 Creating Comments

Path name	A:\SCHOOL
Project name	QEX15
Program name	MAIN

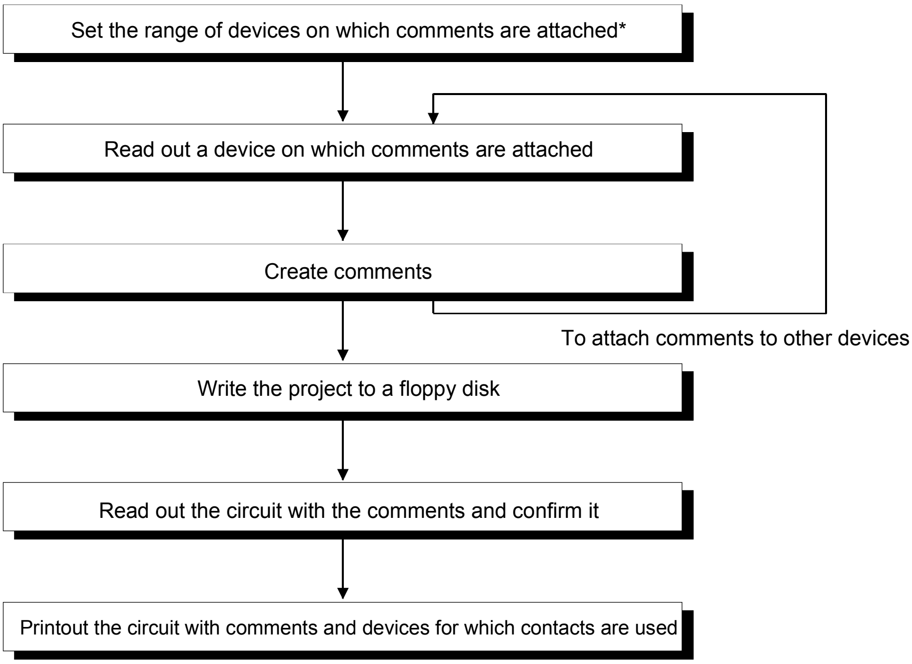
The following is an example of a printed out ladder with comments and devices for which contacts are used.

T0 a contact is in step 5 and 27.
b contact is in step 30.



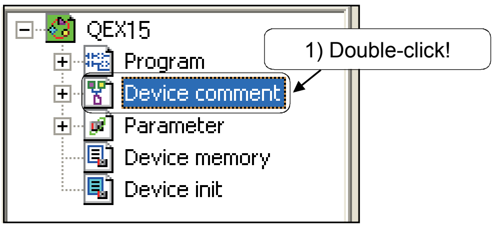
Use a keyboard to input the above program or read it from a FD.

(1) Flowchart of creating comments

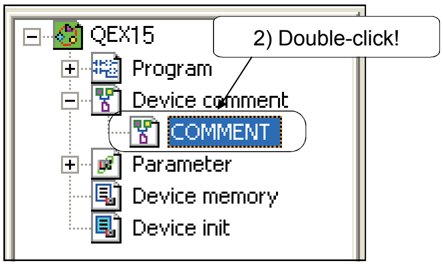


*: The above procedure must be taken to designate the device comment range and write it to a CPU.

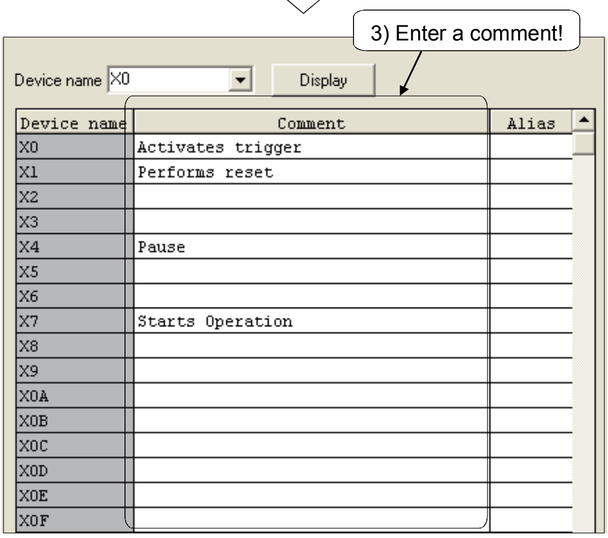
(2) Creating comments



1) Double-click "Device comment" in the project list.



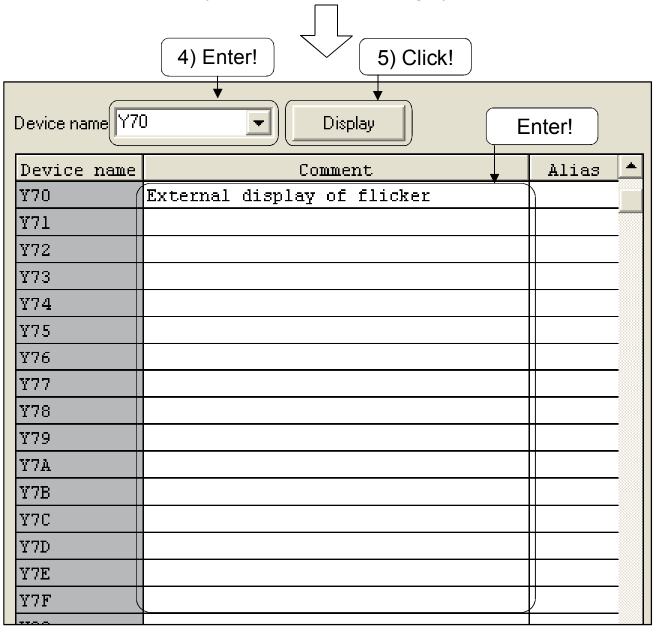
2) "COMMENT" and "MAIN" appear. Double-click "COMMENT" to display the COMMENT screen.



3) Click a desired comment area, and input desired character strings as shown on the left.

(To next page)

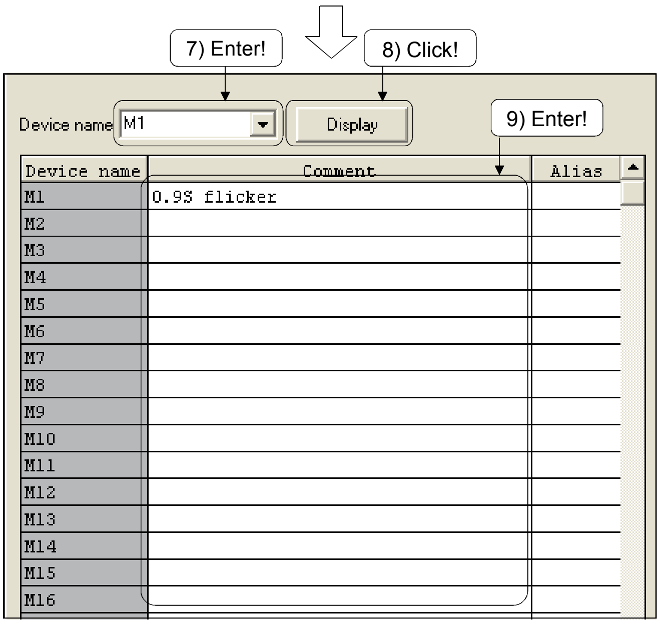
(From previous page)



4) Click "Device" and input "Y70" in the list box.

5) Click the Display button to display a comment list. In this list, the input device is shown at the top accompanied below by its following devices.

6) Click a desired comment area, and input desired character strings as shown on the left.



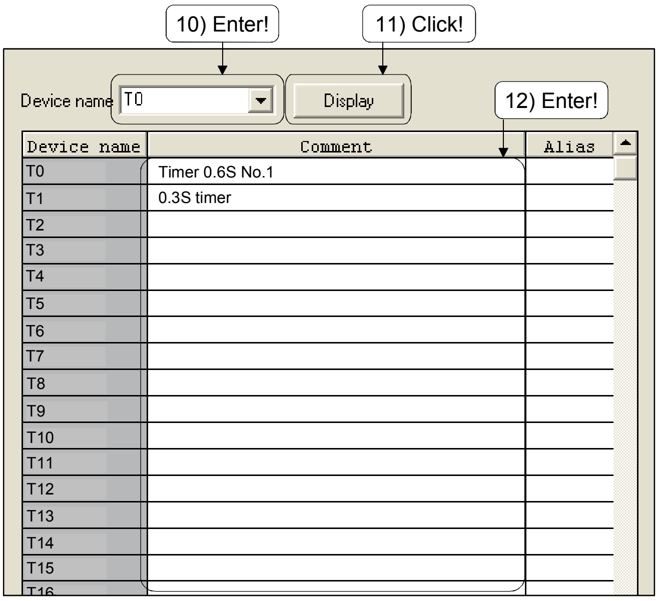
7) Click "Device" and input "M1" in the list box.

8) Click the Display button to display a comment list. In this list, the input device is shown at the top accompanied below by its following devices.

9) Click a desired comment area, and input desired character strings as shown on the left.

(To next page)

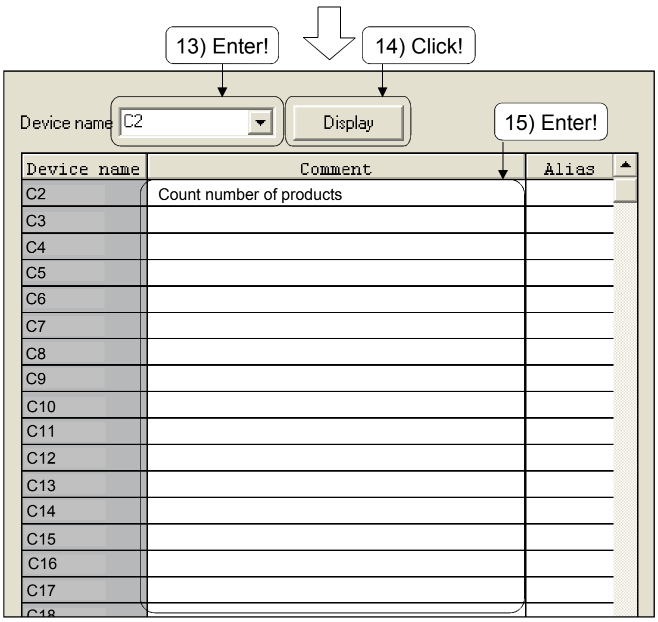
(From previous page)



10) Click "Device" and input "T0" in the list box.

11) Click the Display button to display a comment list. In this list, the input device is shown at the top accompanied below by its following devices.

12) Click a desired comment area, and input desired character strings as shown on the left.

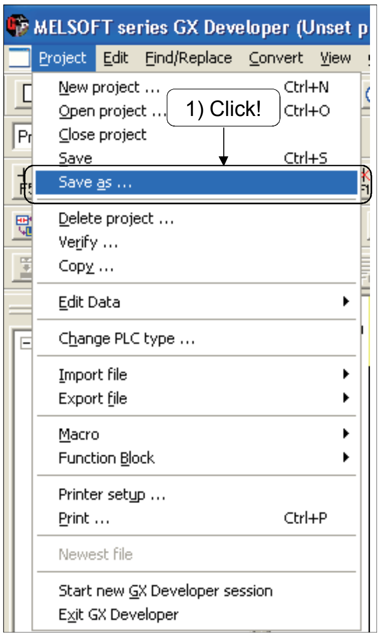


13) Click "Device" and input "C2" in the list box.

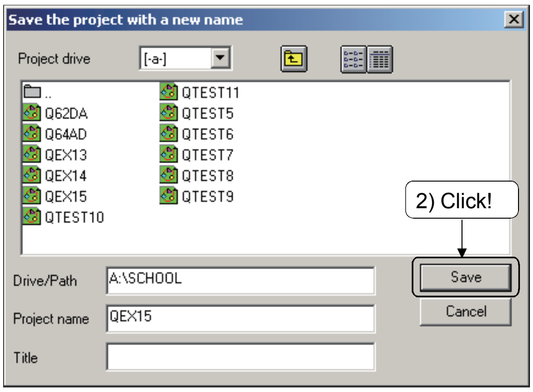
14) Click the Display button to display a comment list. In this list, the input device is shown at the top accompanied below by its following devices.

15) Click a desired comment area, and input desired character strings as shown on the left.

(3) Saving comments

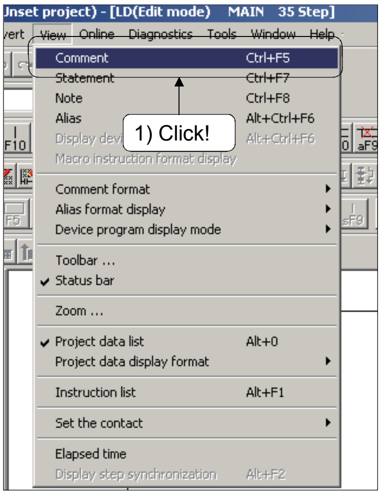


1) Click [Project] → [Save as] menu.

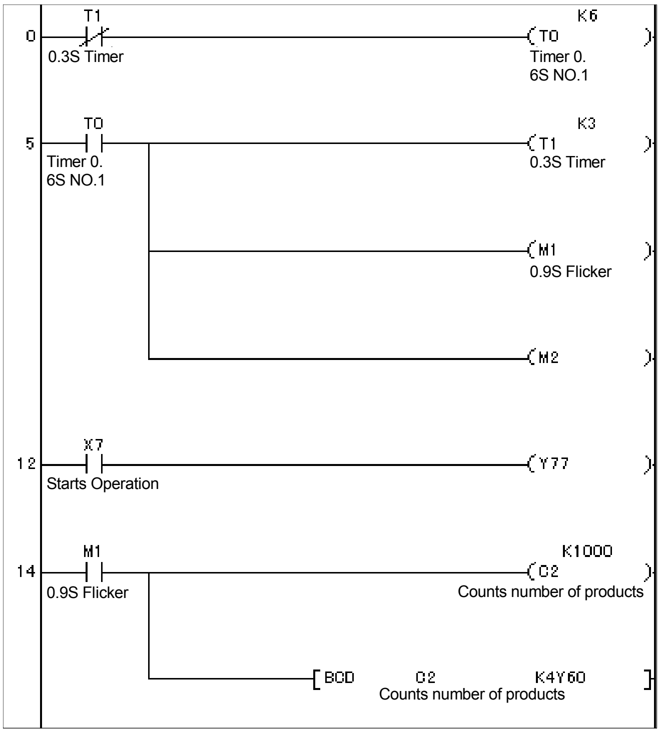


2) The Save the project with a new name dialog box appears. Click the Save button.

(4) Displaying a ladder with comments in GX Developer screens



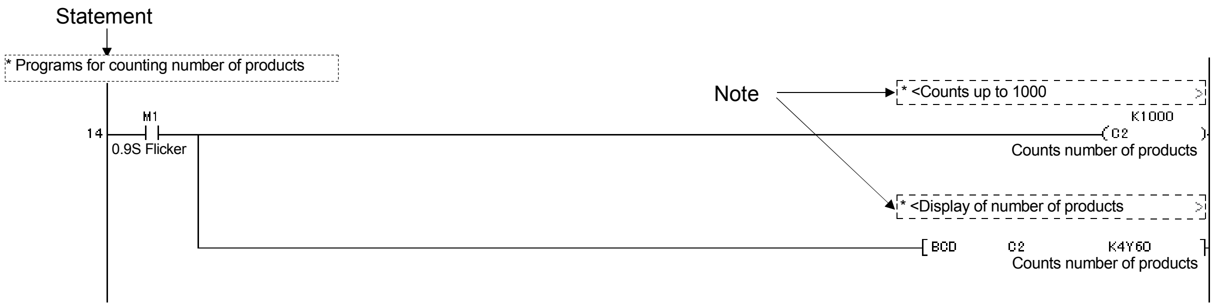
1) Click [View] → [Comment] menu.



2) Comments appear in the ladder screen.

POINT

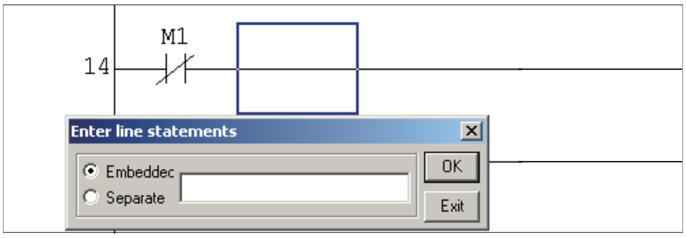
In addition to device comments, a ladder screen can be displayed with statements (comments on ladder blocks) and notes (comments on outputs or instructions).



• Creating statements

Click and then double-click a ladder block where comments are to be attached.

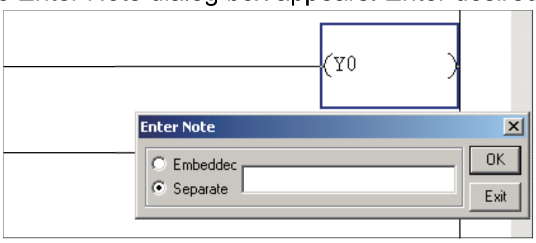
The Enter line statements dialog box appears. Enter desired character strings, and then click the OK button.



• Creating notes

Click and then double-click an output or instruction where comments are to be attached.

The Enter Note dialog box appears. Enter desired character strings, and then click the OK button.



• There are two types of statements and notes; "Embedded" and "Separate".

"Embedded" ...... The data of statements and notes is stored as a part of a program, so it is stored in a CPU used in a factory as well. Note, however, that this takes a lot of space in program memory of PLC CPUs.

"Separate" .... The data of statements and notes is stored not as a part of a program but separately in the peripheral device (personal computer). In this way, a program needs only one extra step at a location. This requires less space in program memory on PLC CPUs.

Note, however, that after taking this way, changing programs at an FA site breaks consistency between programs of GX Developer on the peripheral device (personal computer) and PLC CPUs.

MEMO

CHAPTER 7 PROGRAMMING INTELLIGENT FUNCTION MODULES

7.1 Intelligent Function Module

(1) Intelligent function module type
On PLC CPUs (hereinafter referred to as QCPUs), some functions are not supported or are limited in use. Intelligent function modules support those functions instead of QCPUs.
Therefore users need to select an intelligent function module that is appropriate for the purpose involved.
QCPUs are compatible with QCPU-compatible intelligent function modules and AnS-compatible special function modules.
The following list shows examples of the intelligent function modules.

Table 7.1 Example of intelligent function module

Name	Number of I/O occupied points	Functions	Module current consumption
Analog-digital conversion module (Q64AD)	16 points	Input module that converts 0 to 20mA→0 to 4000 (in standard resolution mode) 0 to ±10V→0 to ±4000 (in standard resolution mode)	5VDC 0.63A
Digital-analog conversion module (Q62DA)	16 points	Output module that converts 0 to 4000→0 to 20mA (in standard resolution mode) 0 to ±4000→0 to ±10V (in standard resolution mode)	5VDC 0.33A 24VDC 0.12A

(2) Using with CPUs
An intelligent function module can be installed on any I/O slots on a main base unit and extension base unit.

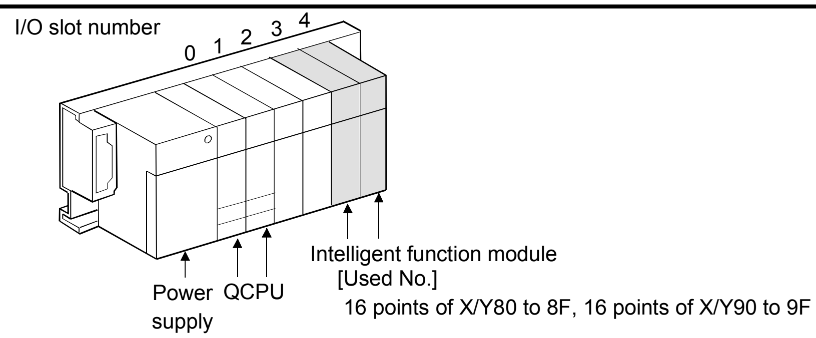
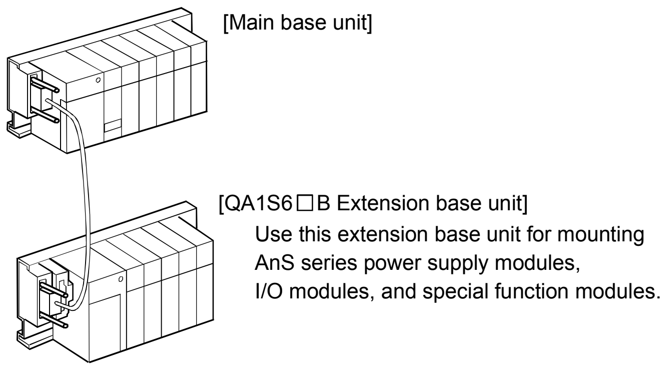


Figure 7.1 Installation of intelligent function module

REMARK

To use AnS series I/O modules or special function modules, install them on QA1S65B or QA168B extension base unit.



7.2 Data Communication between Intelligent Function Modules and CPUs

An intelligent function module and a CPU exchange mainly two formats of data.
Bit data ――――――Signals that use input Xs and output Ys
Word data ―――――16-bit data or 32-bit data

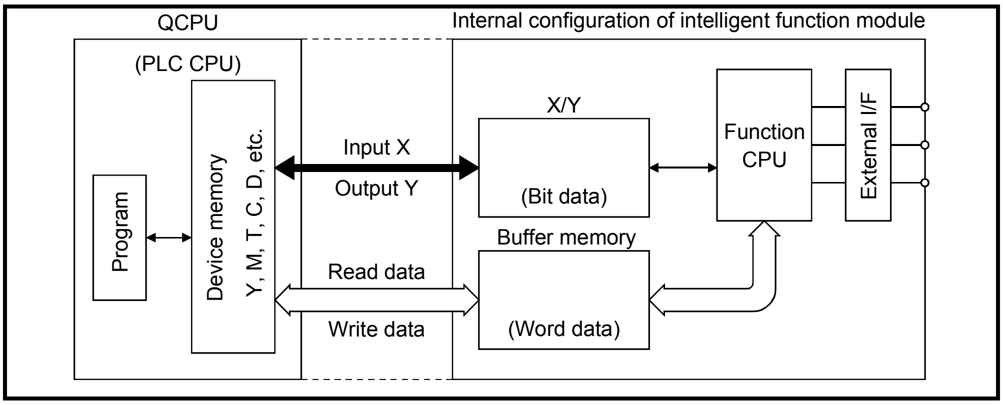


Figure 7.2 Internal configuration of intelligent function module

7.2.1 I/O signals to CPUs

For exchanging signals of 1-bit between a QCPU and an intelligent function module, input Xs and output Ys are used.
X/Y here does not mean external I/Os but symbols that are used in a sequence program to exclusively represent I/O signals of intelligent function modules. Also note that I/O numbers are assigned according to the slot where the intelligent function module is installed.

[X]

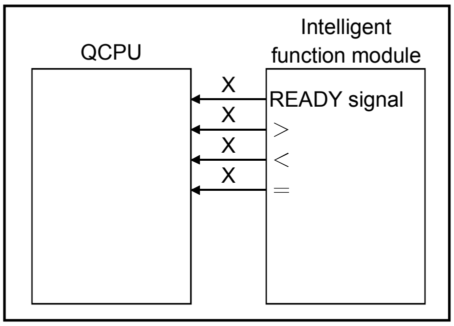


Figure 7.3 X from intelligent function module

Xs in a sequence program represent signals that are input to a QCPU by an intelligent function module. These signals are generated on an intelligent function module. Note that the Xs are used as contacts in a program. The followings are examples of the signals.

(1) READY signal
This signal notifies a QCPU that an intelligent function module started up normally at power-on and is ready for operation.

(2) Comparison result
This signal is used by high-speed counter modules. The modules compare an input count value with the set value to notify a QCPU the results of the comparison; larger (>), smaller (<) or match (=).

[Y]

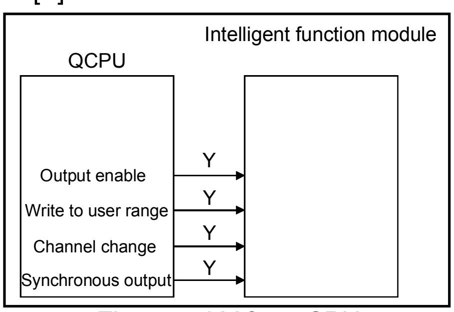


Figure 7.4 Y from CPU

SETs, RSTs, or OUT-Ys represent output signals transmitted from a QCPU to an intelligent function module. These signals are generated on a QCPU. Note that they are used as coils or contacts in a program.

(Ex.) D/A conversion modules output an enable instruction (output enable) before outputting analog values that were converted from digital values.

7.2.2 Data communication with intelligent function modules

Data is transmitted or received in 16-bit or 32-bit units. Intelligent function modules have buffer memory to store those data.

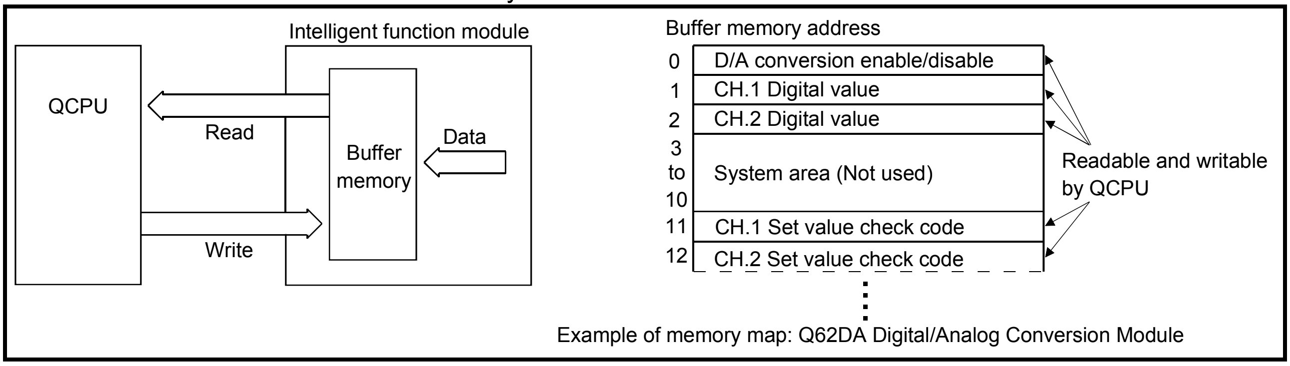


Figure 7.5 Buffer memory

(1) QCPUs can read and write buffer memory. Also note that some modules can write data to buffer memory from peripheral devices via an interface.

(2) In buffer memory, space of one word (16 bits) is reserved for each intelligent function module's unique address.
The smallest address is 0, and these addresses are used to specify a target module to read or write. The minimum unit is one word. 17- to 32-bit data is treated as 2-word (32-bit) data.

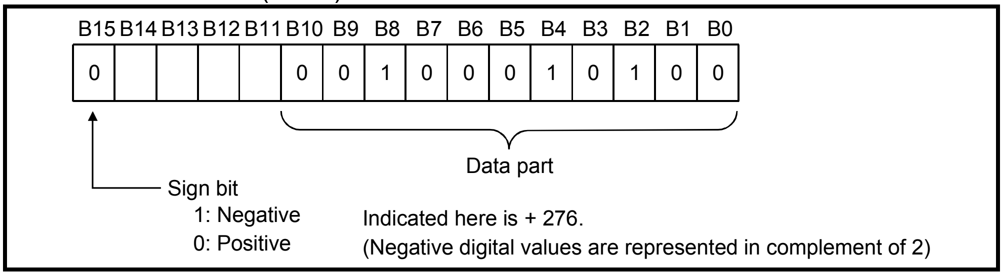


Figure 7.6 Example image of buffer memory content (D/A conversion module)

The buffer memory on Figure 7.6 shows an address of a D/A conversion module in 16 bits. The number is obtained from digital quantity that a QCPU wrote to the buffer memory. Digital values ranging from - 2048 to + 2047 can be set in signed binary (16 bits long).

(3) Buffer memory is a RAM.

7.3 Communicating with Intelligent Function Modules

7.3.1 Communication methods with intelligent function modules

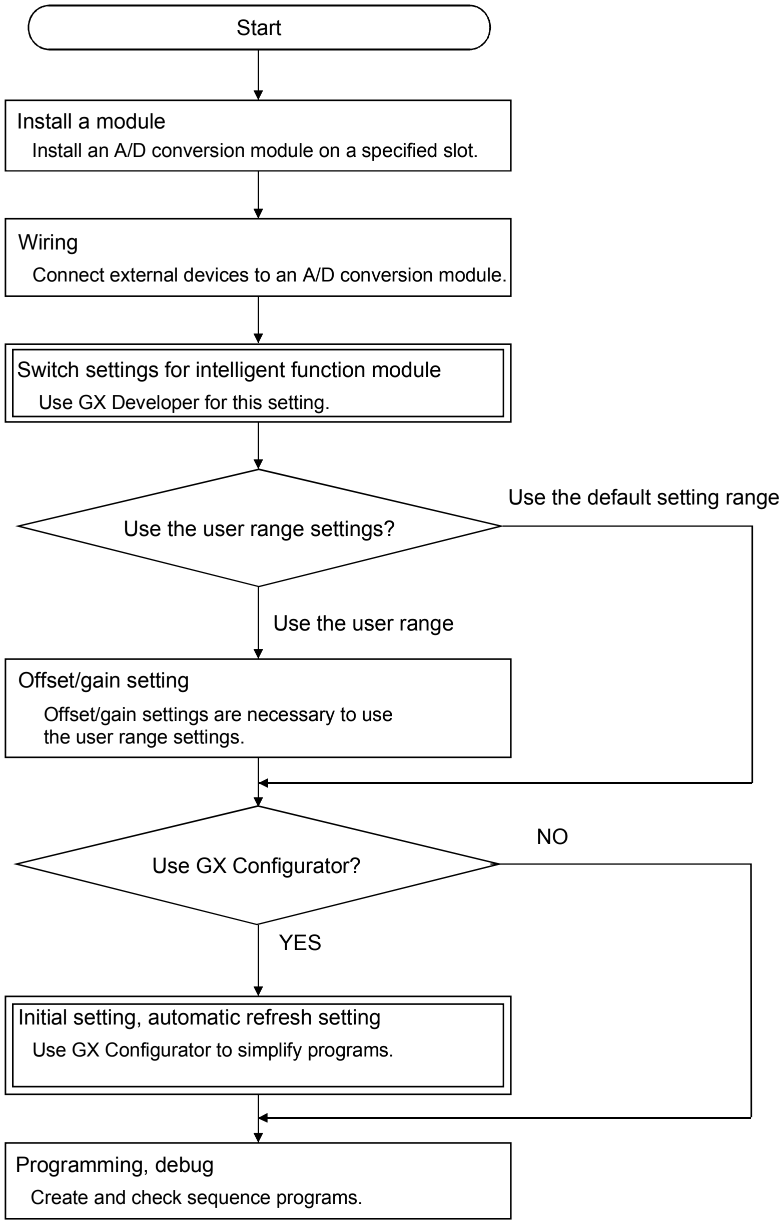
QCPUs provide the following methods for communicating with intelligent function modules.

Table 7.2 Type of communication with intelligent function modules

Communication method	Functions	Setting method
Initial setting, Automatic refresh setting	Performs initial settings and automatic refresh settings of intelligent function modules. These settings allow writing/reading data to/from intelligent function modules regardless of communication program creation or buffer memory address. Ex.) When A/D conversion module Q64AD is used. •Initial setting : •A/D conversion enable/disable setting •Sampling/averaging processing designation •Time average/number of times average designation •Average time/average number of times designation (Set data in initial settings is stored to the intelligent function module.) •Automatic refresh setting : Set a device on a QCPU to store the following data to. •Digital output from Q64AD •Max./min. value of Q64AD •Error code (Set data in automatic refresh settings is stored to the intelligent function module parameter on a QCPU.)	Use GX Configurator compatible with the intelligent function module.
Device initial value	Writes set data in device initial settings of intelligent function modules to the intelligent function modules at the following timings. •At power-on of a QCPU •At reset •At switch from STOP to RUN	Use GX Developer to specify the range for intelligent function module devices (U □\G□).
FROM/TO instruction	Executes data read/write to buffer memory on an intelligent function module.	Use this instruction in a sequence program.
Intelligent function module device (U □\G□)	Directly treats buffer memory on an intelligent function module as a device of a QCPU. Unlike "FROM/TO instruction", this requires only one instruction for processing data that is read from an intelligent function module.	Specify as a device in a sequence program.
Intelligent function module dedicated instruction	Dedicated instructions used to simplify programming for using the functions of intelligent function modules	Use this instruction in a sequence program.

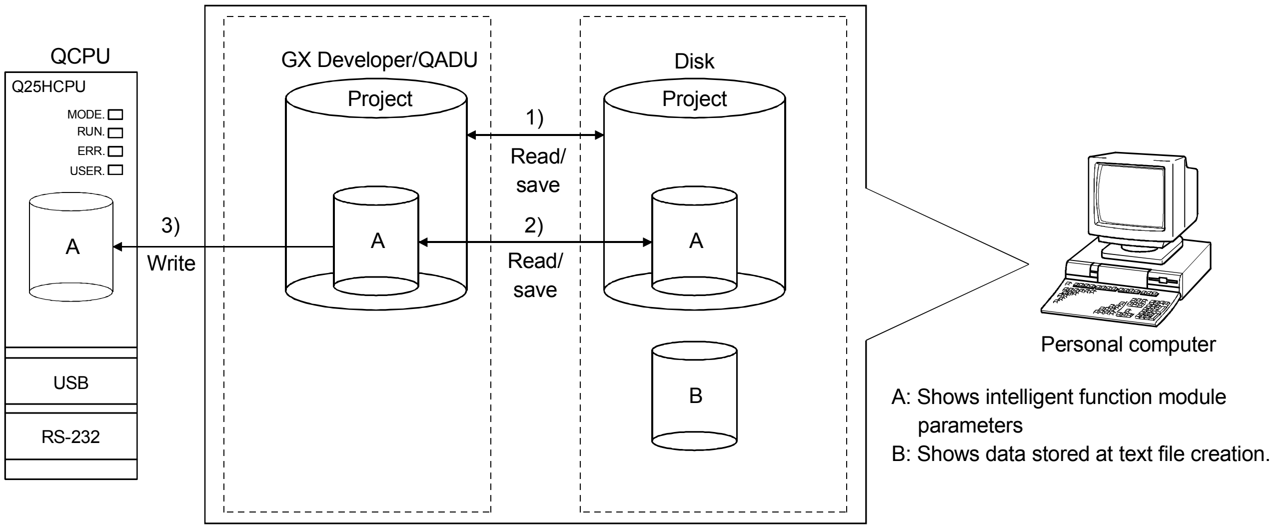
7.3.2 Using GX Configurator for communication

This section describes the procedure of using GX Configurator for communication with intelligent function modules.
In the example operation below, Q64AD module/SW□D5C-QADU is used.



7.3.3 Data created by GX Configurator

The following data/files can be created with the GX Configurator package and also used in GX Developer. Refer to the following figure to know in which operation each data/file is needed.



＜Parameter of intelligent function module＞

(a) This parameter data is created in the auto refresh settings and stored to the intelligent function module parameter file in a GX Developer-created project.

- Project
  - Program
  - Parameter
    - PLC parameter
    - Network parameter
    - Intelligent function module parameter

(b) For how to perform 1) to 3) on the above figure, refer to the followings.

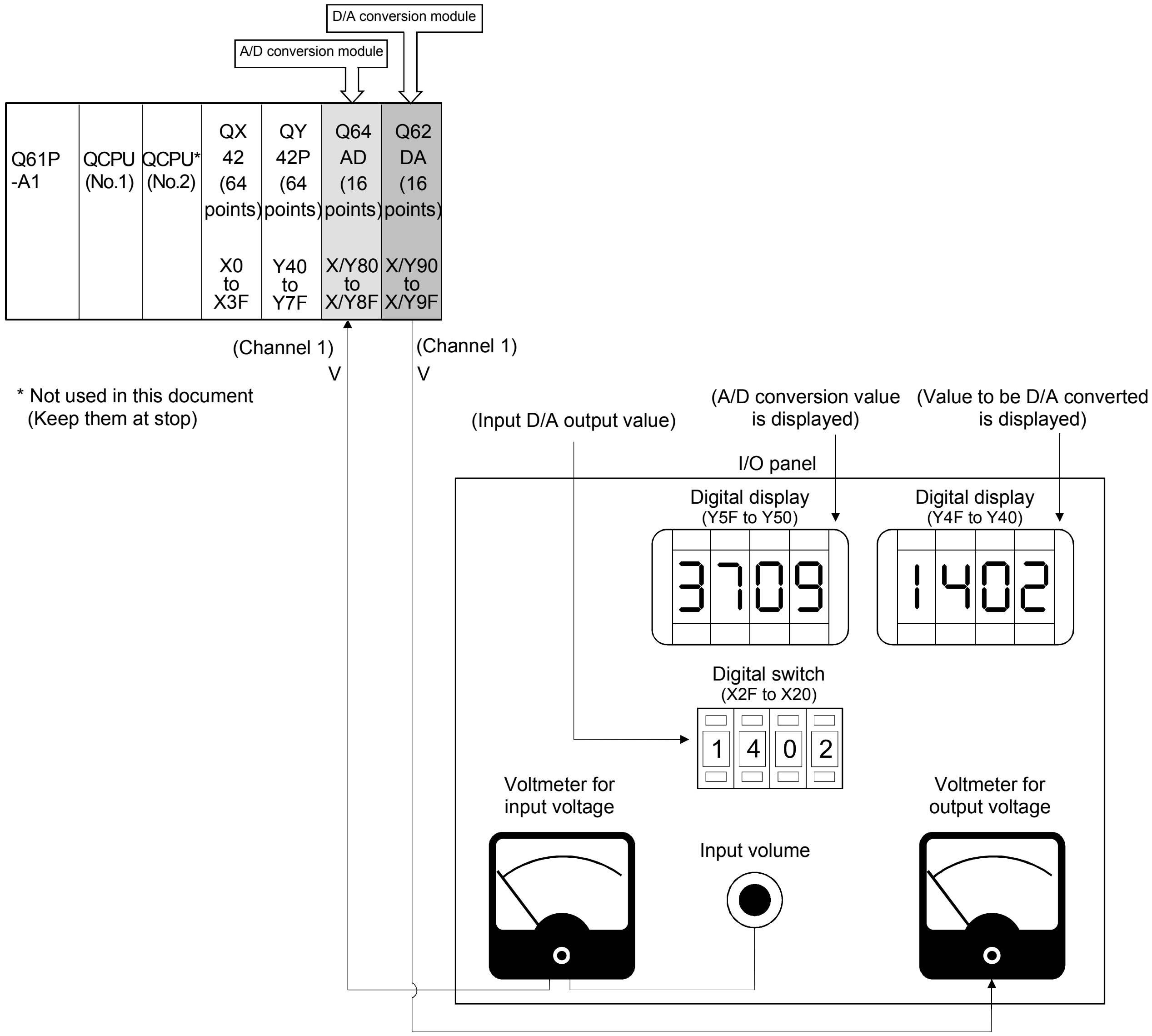
1) Use GX Developer as follows.
   [Project] → [Open project]/[Save]/[Save as]
2) Open the GX Configurator's screen for selecting a module on which parameter settings are to be made.
   [File] → [Open file]/[Save as]
3) Use GX Developer as follows.
   [Online] → [Read from PLC]/[Write to PLC] → "Intelligent function module parameters"
   Or, open the GX Configurator's screen for selecting a module on which parameter settings are to be made. [Online] → [Read from PLC]/[Write to PLC]

<Text files>

(a) These files are text format files created using $\boxed{\text{Text file creation}}$ operation in the initial settings, auto refresh settings and the monitor/test screen.
Use these files to create user's documents.

7.4 Exercise System of Intelligent Function Module

Use an A/D or D/A conversion module to convert analog signals/digital data that are input with the volume or digital switch on the demonstration machine.

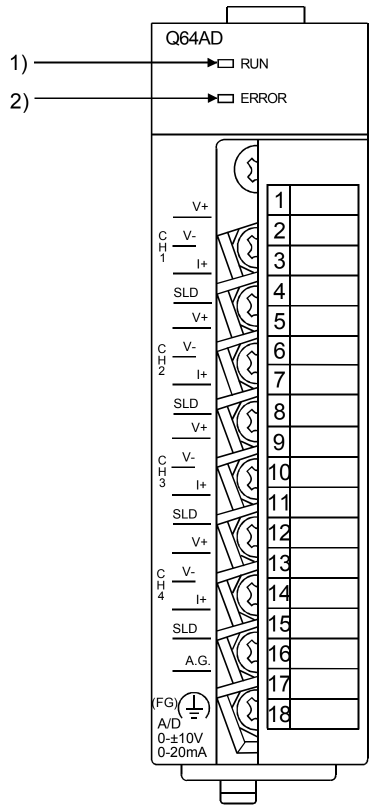


7.5 Q64AD Analog/Digital Conversion Module

7.5.1 Names of parts

Part names of Q64AD are given below together with descriptions.
For details, refer to the User's Manual.

Q64AD



No.	Name and appearance	Descriptions
1)	RUN LED	Indicates operation status of an A/D conversion module. ON: In normal operation Flicker: In offset/gain setting mode OFF: 5V power failure or watchdog timer error occurred
2)	ERROR LED	Indicates errors and status of an A/D conversion module. ON: Error occurred OFF: In normal operation Flicker: Switch settings error occurred Values other than 0 has been set to the switch 5 on an intelligent function module.

7.5.2 A/D conversion characteristics

(1) A/D conversion characteristics on voltage inputs
(when in a standard resolution mode with analog input range set to –10 to 10V)

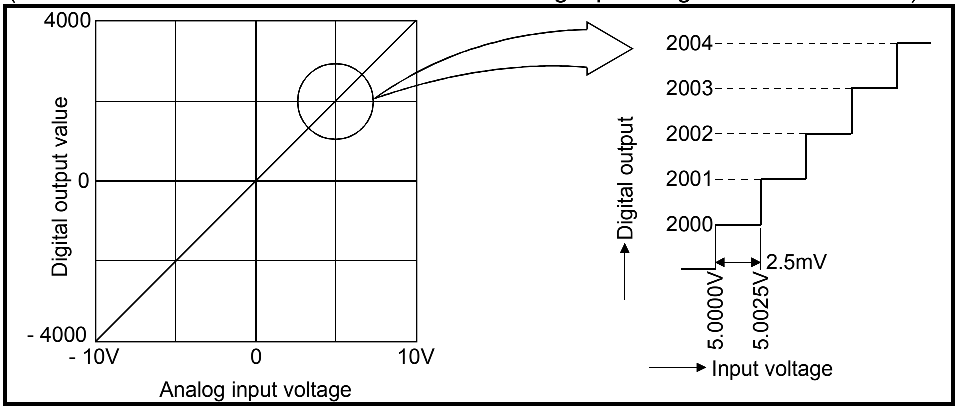


Figure 7.12 A/D conversion characteristics (voltage input)

Analog-digital conversion modules convert analog values input from other devices to digital quantity so that CPUs can operate those values. On voltage inputs, for example, they convert - 10V to digital quantity of – 4000 and 10V to 4000. This means that the modules convert input voltage of 2.5mV to digital quantity of 1, and abandon values smaller than 2.5mV.

(2) A/D conversion characteristics on current inputs
(when in a standard resolution mode with analog input range set to 0 to 20mA)

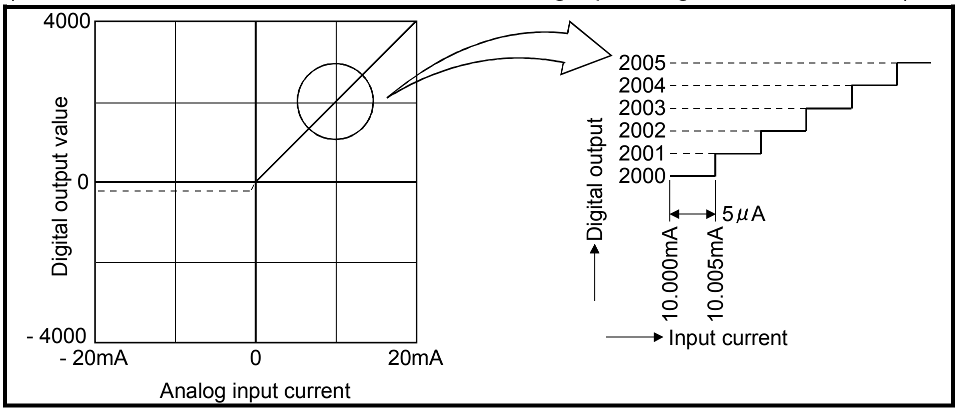


Figure 7.13 A/D conversion characteristics (current input)

The modules convert current input of 0mA to 0 for output, and 20mA to 4000. This means that the modules convert input current of 5μA to digital quantity of 1, and abandon values smaller than 5μA.

REMARK
A voltage/current value (max. resolution) that becomes digital value of 1 through A/D conversion differs depending on the settings of the resolution mode (1/4000, 1/20000, 1/60000) or the output range.

7.5.3 Intelligent function module switch settings

Q series uses the I/O assignment settings of GX Developer to configure the switch settings for intelligent function modules.
The intelligent function module switch settings provide switch 1 to 5, and use 16-bitdata.
When the intelligent function module switch settings are not configured, values of switch 1 to 5 are set to, which is 0.

(1) Setting items for module switches

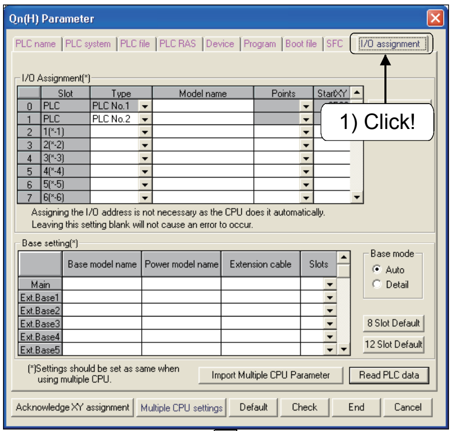
<table>
<tr><th></th><th colspan="2">Setting items</th></tr>
<tr><td>Switch 1</td><td>Input range setting
□□□□ H
CH4 CH3 CH2 CH1</td><td rowspan="2">
<table>
<tr><th>Analog output range</th><th>Setting value for output range</th></tr>
<tr><td>4 to 20mA</td><td>0_H</td></tr>
<tr><td>0 to 20mA</td><td>1_H</td></tr>
<tr><td>1 to 5V</td><td>2_H</td></tr>
<tr><td>0 to 5V</td><td>3_H</td></tr>
<tr><td>-10 to 10V</td><td>4_H</td></tr>
<tr><td>0 to 10V</td><td>5_H</td></tr>
<tr><td>User range setting</td><td>F_H</td></tr>
</table>
</td></tr>
<tr><td>Switch 2</td><td>Input range setting
□□□□ H
CH8 CH7 CH6 CH5</td></tr>
<tr><td>Switch 3</td><td colspan="2">Vacant</td></tr>
<tr><td>Switch 4</td><td colspan="2">□□□□ H
00_H : Temperature drift compensation
01 - FF_H : No temperature drift compensation
0_H : Standard resolution mode
1 - F_H : High resolution mode
0_H : Standard mode (A/D conversion)
1 - F_H : Offset/gain setting mode</td></tr>
<tr><td>Switch 5</td><td colspan="2">fixed to 0</td></tr>
</table>

REMARK

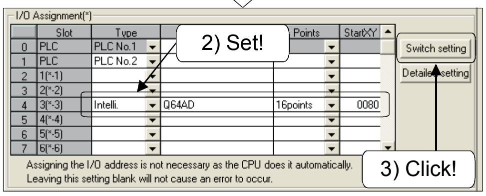
The settings for the offset/gain setting mode differ between the function version A and function version B.
Explanations in this section are made based on the function version B.
For details, refer to the User's Manual.

(2) Setting module switches

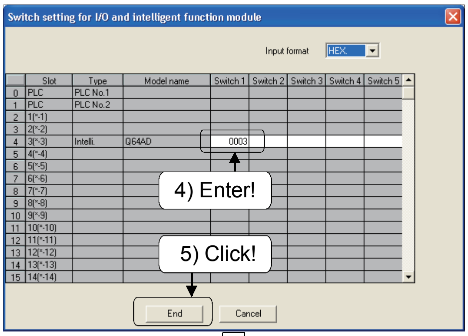
> The demonstration machine incorporates two CPUs, so it is necessary to perform the parameter settings for multi-CPU system. (if there is only one CPU incorporated, the parameter settings mentioned is not required.) For how to perform the parameter setting, refer to the multi-CPU setting of parameters in Section 3.2.



1) Click the "I/O assignment" tab in the PLC parameter settings.



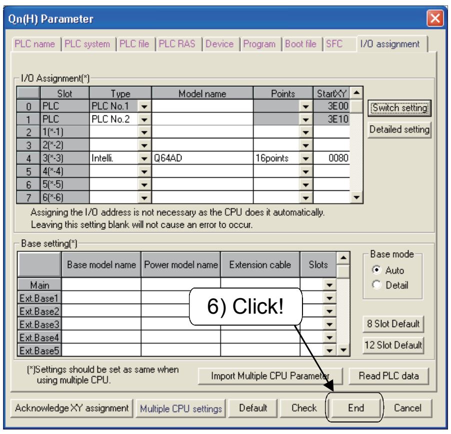
2) Perform I/O assignment on the slot "3(*-3)" where Q64AD is installed.
   Type: "Intelli" (required)
   Model name: "Q64AD"
   Points: "16points"
   StartXY: "80" (Hexadecimal)
3) Click the [Switch setting] button. The Switch setting for I/O and intelligent function module dialog box appears.



4) The following is an example of the intelligent function module switch settings of Q64AD.
   The default value of each switch is "0".

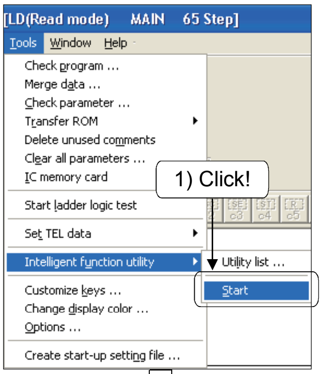
Switch	Set Value	Item	Description of setting
1	3	Input range	CH1: 0 to 5V
2	(Do not input)	Input range	Default value
3	(Do not input)	Vacant	Default value
4	(Do not input)	(Drift compensation mode selection)	Default value
5	(Do not input)	fixed to 0	Default value

5) Click the [End] button.

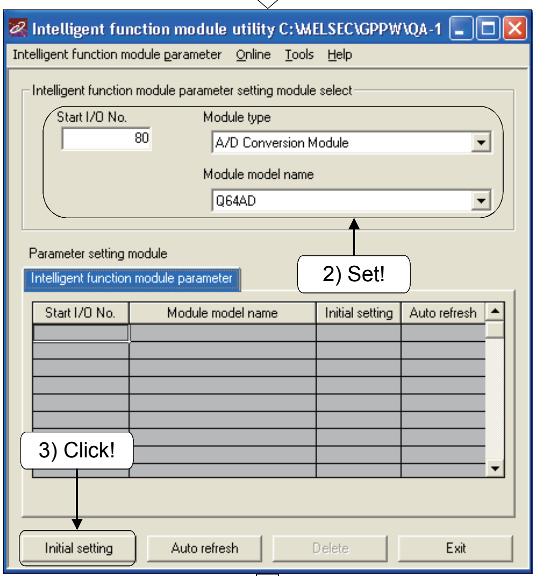


6) Click the [End] button on the Qn(H) Parameter dialog box to terminate the intelligent function switch settings.

7.5.4 Setting with GX Configurator

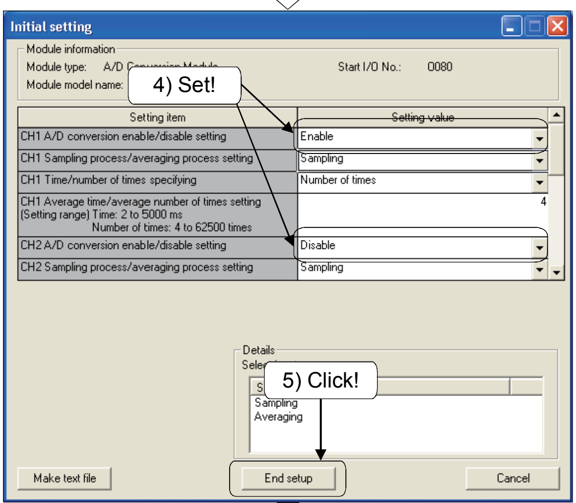


1) Activate GX Developer, and click [Tools], [Intelligent function utility], and [Start] sequentially.



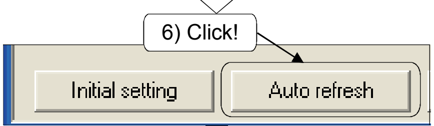
2) GX Configurator starts up.
Perform the settings of the A/D conversion module as follows.
Start I/O No.: "80"
Module type: "A/D Conversion Module"
Module model name: "Q64AD"

3) Click the Initial setting button.



4) The Initial setting screen appears.
In the following example, A/D conversion enable/disable settings from CH2 to CH4 are set to "Disable".
(this means using only CH1)

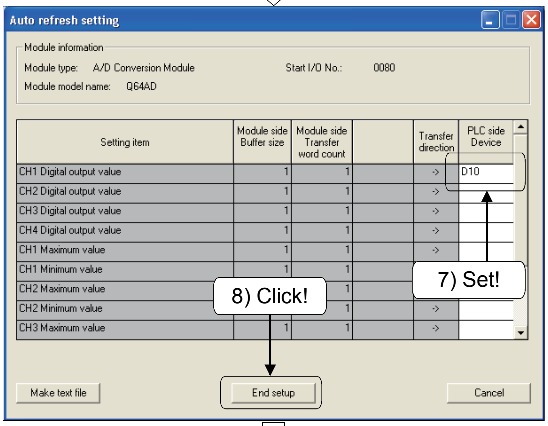
5) Click the End setup button.



6) Click the Auto refresh button in GX Configurator.

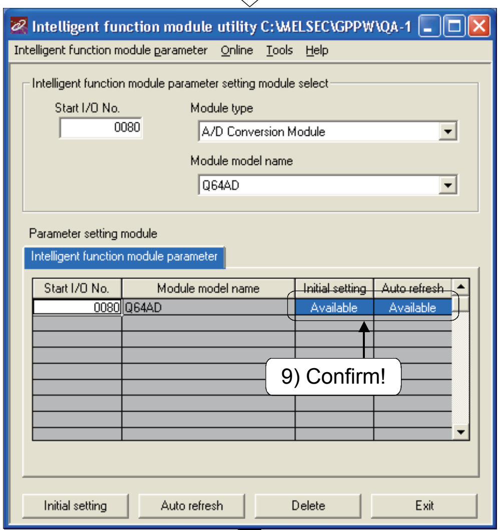
(To next page)

(From previous page)

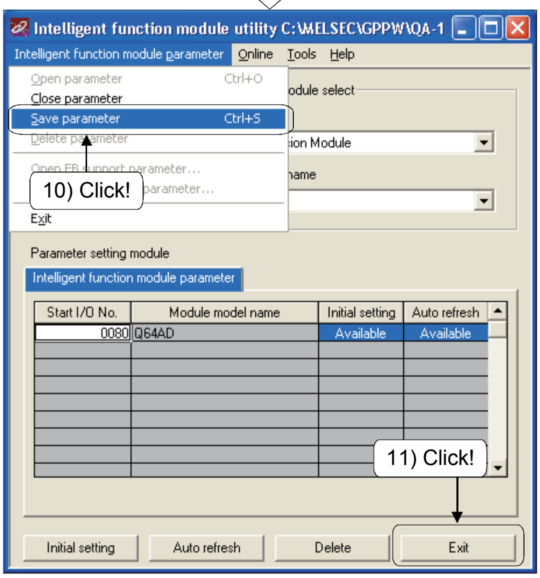


7) Set "D10" in the PLC side Device area on the "CH1 Digital output values" row of Setting item.

8) Click the End setup button.



9) Confirm that "Available" is set in the Initial setting and Auto refresh column of Q64AD.



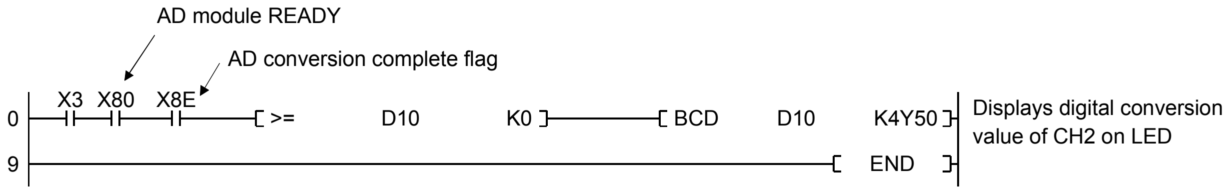
10) After the setting is completed, click [Intelligent function module parameter] and then click [Save parameter].

11) Click the Exit button.

7.5.5 Exercise with the demonstration machine

(1) Sequence program
The sequence program performs a sampling processing on analog voltages input to Q64AD from CH1, and then converts the analog values to digital values.

Path name	A:\SCHOOL
Project name	Q64AD
Program name	MAIN



X80: Module READY signal
X8E: A/D conversion complete flag
At power-on or reset of a PLC CPU, this flag turns on if A/D conversion is ready to be executed. A/D conversion is executed once this flag turned on.

(2) Operation of the demonstration machine
Stop the CPU and click  on the toolbar.
The Write to PLC dialog box opens. Click the Param + Prog button and then click the Execute button to write to a CPU. After that, activate a CPU and confirm the following items.

Write only the parameter settings to a second CPU in the same way.
(This is not necessary if there is only one CPU incorporated.) For how to how to write parameters to a second CPU, refer to Section 2.6.

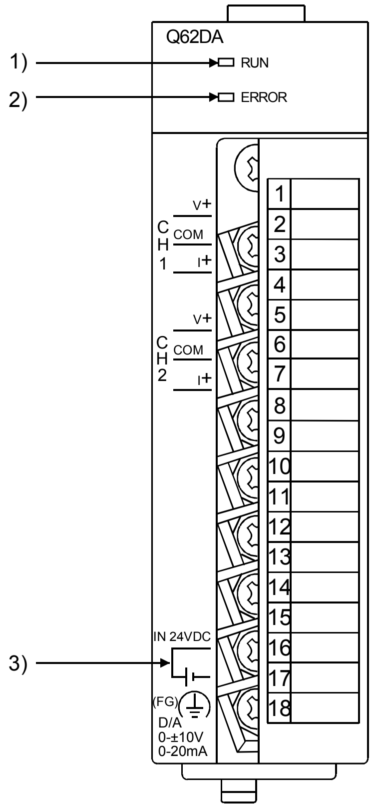
1) Turn on X3, and change input voltages for an A/D conversion module by the volume on the demonstration machine.
Analog values that were input to channel 1(CH1) on Q64AD are stored to buffer memory (in digital value). With the automatic refresh settings, a QCPU reads the stored digital values and stores them in its data register D10.
2) Whenever an analog value is smaller than "-1", 0 is set.
3) Analog-to-digital converted values appear on the digital display (Y50 to Y5F).

7.6 Q62DA Digital/Analog Conversion Module

7.6.1 Names of parts

Part names of Q62DA are given below together with descriptions.
For details, refer to the User's Manual.

Q62DA



No.	Name and appearance	Description
1)	RUN LED	Indicates operation status of a D/A conversion module. ON: In normal operation Flicker: In offset/gain setting mode OFF: 5V power failure or watchdog timer error occurred
2)	ERROR LED	Indicates errors and status of an D/A conversion module. ON: Error occurred OFF: In normal operation Flicker: Switch settings error occurred Values other than 0 has been set to the switch 5 on an intelligent function module.
3)	External power supply terminal	A terminal for connecting 24VDC external power supply

7.6.2 D/A conversion characteristics

(1) D/A conversion characteristics on voltage outputs
(when in a standard resolution mode with analog output range set to –10 to 10V)

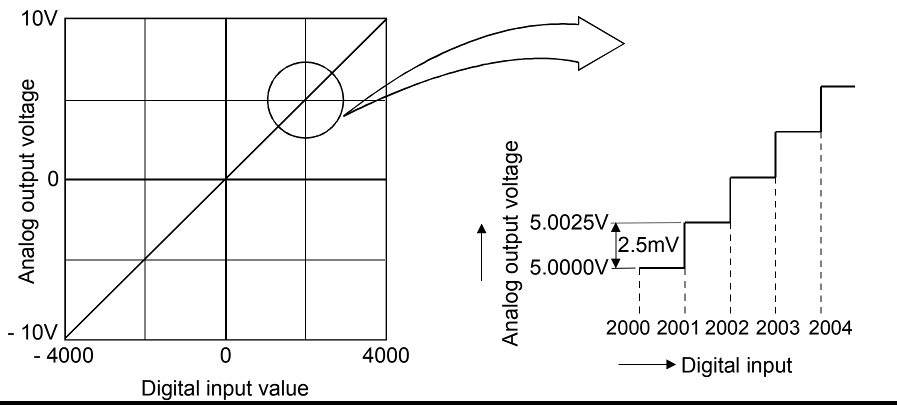


Figure 7.14 D/A conversion characteristics (current output)

Digital/analog conversion modules convert digital quantity that is input from a QCPU to analog values, and then output it to exterior. For example, the modules convert digital quantity of -4000 to analog quantity of -10V and 4000 to 10V before output. This means that the modules convert the digital input value of 1 to analog quantity of 2.5mA, and abandon digital input values in decimal places.

(2) D/A conversion characteristics on current outputs
(when in a standard resolution mode with analog output range set to 0 to 20mA)

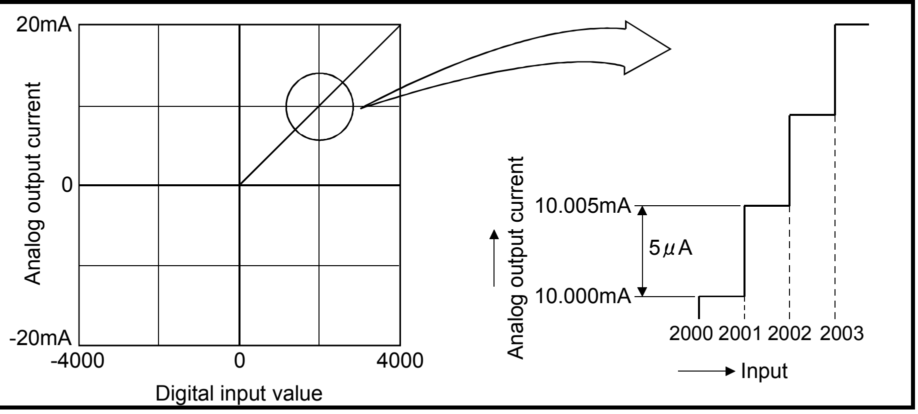


Figure 7.15 D/A conversion characteristics (current output)

For current outputs, the modules convert 0 to 0mA and 4,000 to 20mA. This means that the modules convert the digital input value of 1 to analog quantity of 5μA, and abandon digital input values in decimal places.

REMARK
A voltage/current value (max. resolution) that becomes digital value of 1 through D/A conversion differs depending on the settings of the resolution mode (1/4000, 1/20000, 1/60000) or the output range.

7.6.3 Intelligent function module switch settings

(1) Setting items for module switches

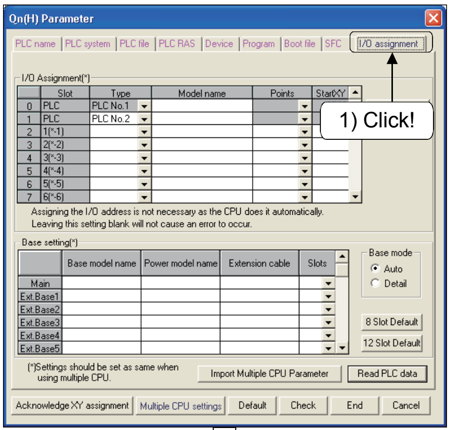
<table>
<tr><th></th><th colspan="2">Setting items</th></tr>
<tr><td>Switch 1</td><td>Input range setting
□□□□H
CH4 CH3 CH2 CH1</td><td rowspan="2">
<table>
<tr><th>Analog output range</th><th>Setting value for output range</th></tr>
<tr><td>4 to 20mA</td><td>0H</td></tr>
<tr><td>0 to 20mA</td><td>1H</td></tr>
<tr><td>1 to 5V</td><td>2H</td></tr>
<tr><td>0 to 5V</td><td>3H</td></tr>
<tr><td>-10 to 10V</td><td>4H</td></tr>
<tr><td>User range setting</td><td>FH</td></tr>
</table>
</td></tr>
<tr><td>Switch 2</td><td>Input range setting
□□□□H
CH8 CH7 CH6 CH5</td></tr>
<tr><td>Switch 3</td><td colspan="2">□□□□H
CH4 CH3 CH2 CH1

HOLD/CLEAR function setting
0H : CLEAR
1 to FH HOLD</td></tr>
<tr><td>Switch 4</td><td colspan="2">□ □ □□ H
0H :Standard mode (Asynchronous)
01-FFH :Synchronous output mode
0H :Standard resolution mode
1-FH :High resolution mode
0H :Standard mode (D/A conversion)
1-FH :Offset/gain setting mode</td></tr>
<tr><td>Switch 5</td><td colspan="2">Fixed to 0</td></tr>
</table>

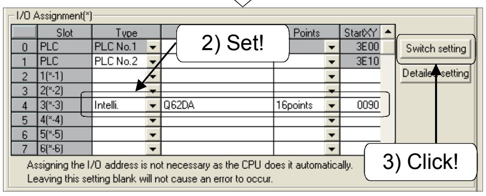
REMARK
The settings for the offset/gain setting mode differ between the function version A and function version B. Explanations in this section are made based on the function version B. For details, refer to the User's Manual.

(2) Setting module switches

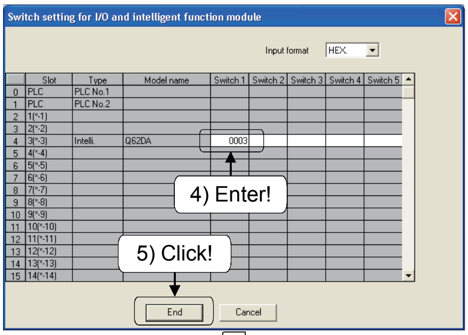
> The demonstration machine incorporates two CPUs, so it is necessary to perform the parameter settings for multi-CPU system. (if there is only one CPU incorporated, the parameter settings mentioned is not required.) For how to perform the parameter setting, refer to the multi-CPU setting of parameters in Section 3.2.



1) Click the "I/O assignment" tab in the PLC parameter settings.

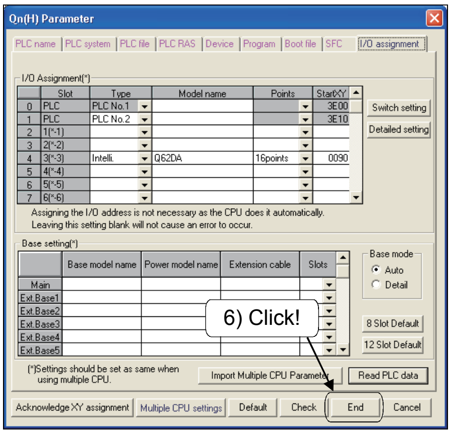


2) Perform I/O assignment on the slot "4(*-4)" on which Q62DA is installed.
   Type: "Intelli" (required)
   Model name: "Q62DA"
   Points: "16points"
   StartXY: "90" (Hexadecimal)
3) Click the [Switch setting] button. The Switch setting for I/O and intelligent function module dialog box appears.



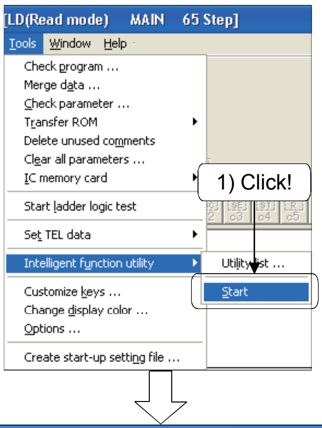
4) The following is an example of the intelligent function module switch settings of Q62DA.
   The default value of each switch is "0".

Switch	Set Value	Item	Description of setting
1	3	Output range	CH1: 0 to 5V
2	(Do not input)	Vacant	Default value
3	(Do not input)	HOLD/CLEAR	Default value
4	(Do not input)	(Synchronous/asynchronous mode selection)	Default value
5	(Do not input)	fixed to 0	Default value

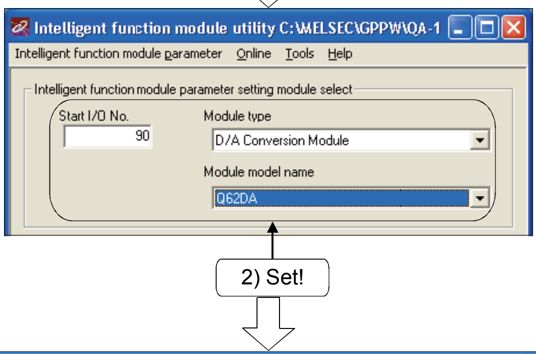


5) Click the [End] button.
6) Click the [End] button on the Qn(H) Parameter dialog box to terminate the intelligent function module switch settings.

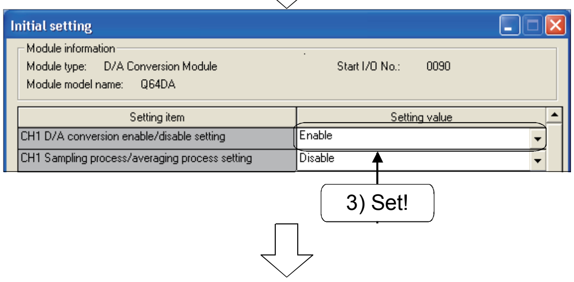
7.6.4 Setting with GX Configurator



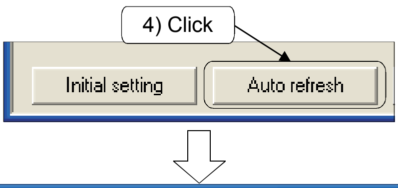
1) Activate GX Developer, and click [Tools], [Intelligent function utility], and [Start] sequentially.



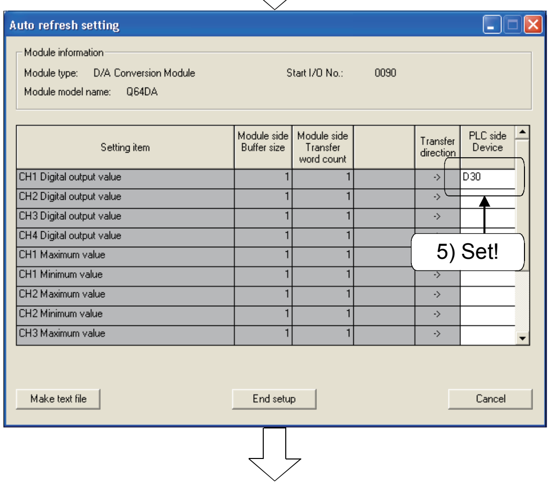
2) Set D/A conversion module as follows.
Start I/O No.: "90"
Module type: "D/A Conversion Module"
Module model name: "Q62DA"
After the settings are completed, click the Initial setting button.



3) In the following example, "CH1 D/A conversion enable/disable setting" is set to "Enable" .
(this means using only CH1)
After the setting is completed, click the End button.



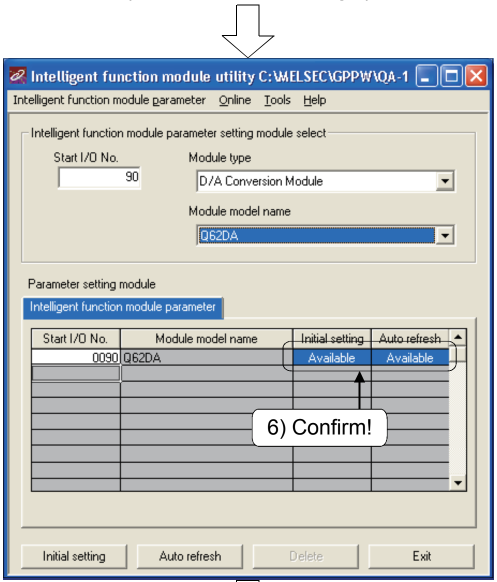
4) Click the Auto refresh button in GX Configurator.



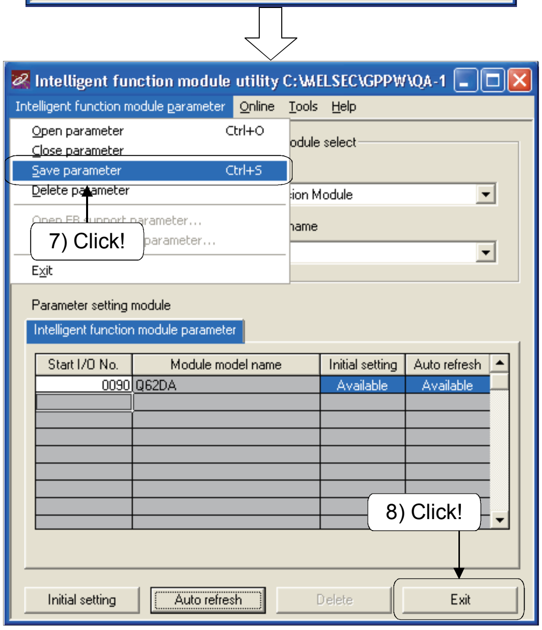
5) Set "D30" in the PLC side Device area on the "CH1 Digital output value" row of Setting item.
After the setting is completed, click the End setup button.

(To next page)

(From previous page)



6) Confirm that "Available" is set in the Initial setting and Auto refresh column of Q62DA.



7) After the setting is completed, click [Intelligent function module parameter] and then click [Save parameter].

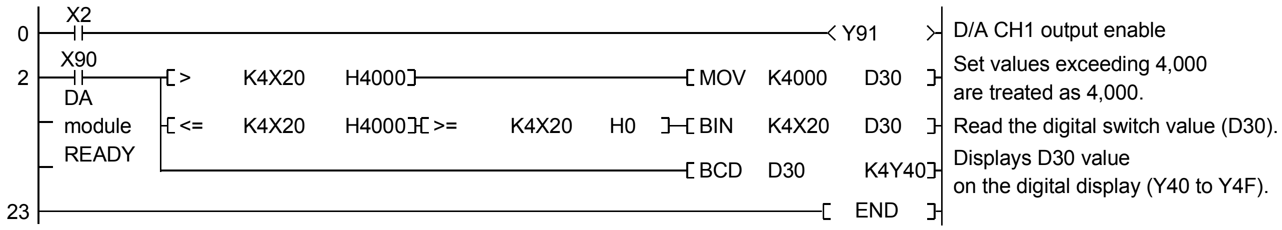
8) Click the Exit button.

7.6.5 Exercise with the demonstration machine

(1) Sequence program

The sequence program converts values of the digital switches to analog signals.

Path name	A:\SCHOOL
Project name	Q62DA
Program name	MAIN



X90: Module READY signal

At power-on or reset of a PLC CPU, this signal turns on if D/A conversion is ready to be executed. D/A conversion is executed once this signal turned on.

Y91: CH.1 output enable/disable flag

Turning this flag on or off selects on each channel whether to output D/A converted values or offset values.

ON: D/A converted value OFF: Offset value

(2) Operation of the demonstration machine

Stop the CPU and click  on the toolbar.

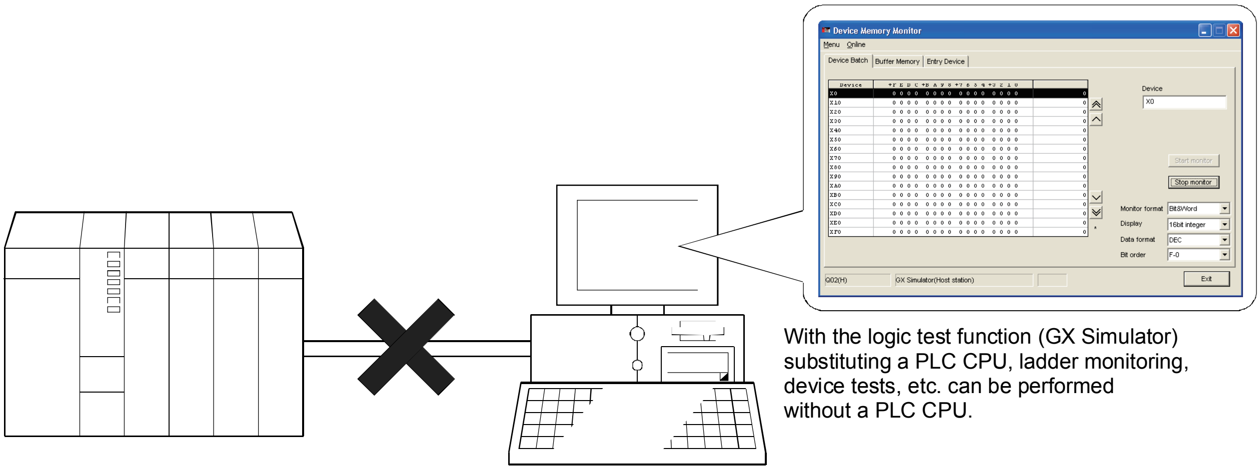
The Write to PLC dialog box opens. Click the Param + Prog button and then click the Execute button to write to a CPU. After that, activate a CPU and confirm the following items.

Write only the parameter settings to a second CPU in the same way.
(This is not necessary if there is only one CPU incorporated.) For how to write parameters to second CPU, refer to Section 2.6.

1) Turn on X2 to enable D/A CH1 outputs.
2) Convert digital values set in the digital switches (X20 to X2F) to analog values. Whenever the set value of the digital switch is outside the range of 0≤X≤4000, 4000 is D/A converted to be output.
   The digital value to be converted to an analog value appears on the digital display (Y40 to 4F).
3) The D/A OUTPUT voltmeter displays the voltage value that the D/A conversion module outputs.

MEMO

Offline debugging is possible by adding the logic test function (GX Simulator) to a computer in which GX Developer is installed.
With the logic test function (GX Simulator), which allows sequence programs to be developed and debugged on a single computer, checking a modified sequence program is easier and quicker.



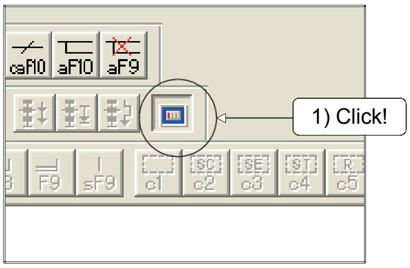
With the logic test function (GX Simulator) substituting a PLC CPU, ladder monitoring, device tests, etc. can be performed without a PLC CPU.

(1) Functions supported by the logic test function (GX Simulator)
The functions supported by the logic test function (GX Simulator) include functions executed from the logic test function (GX Simulator) menu and functions executed from the GX Developer menu. For functions from the GX Developer menu, it is necessary to use them together with the logic test function (GX Simulator).

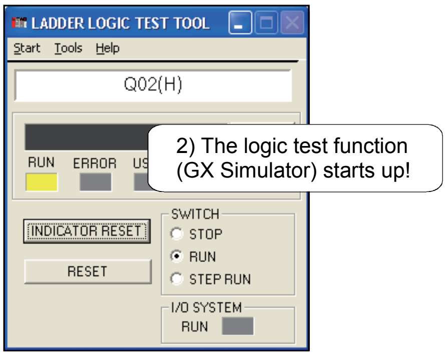
Functions		Meaning
Functions executed from the GX Developer menu	Ladder monitor, Device monitor	Monitors the operation status of the logic test function (GX Simulator).
	Device test	Forcibly rewrites device values of the logic test function (GX Simulator) during monitoring.
	Write to PLC	Writes parameter files and program files to the logic test function (GX Simulator).
	PLC diagnostics	Checks the logic test function (GX Simulator) status and errors.
	Skip execution	Skips (does not execute) program execution in the range between two designated steps.
	Partial execution	Executes the part of the program in a designated step or pointer range.
	Step execution	Executes the sequence program one step at a time.
	Remote operation	Operates the logic test function (GX Simulator) execution status.
	Program monitor list	Monitors the program execution status and number of executions as a table, and starts and stops the program execution in the table.
Functions executed from the logic test function (GX Simulator) menu	Monitor test	Conducts testing by monitoring the device memory status, forcing the devices ON/OFF, and changing present values.
	I/O system settings	Simulates the operation of external devices by simple settings.
	Tools	Saves and reads the device memory and buffer memory.
	Function equivalent to WDT	Issues a WDT error if a sequence program is written in such a way that it runs an infinite loop.
	Error detail display function	Displays detailed error information at occurrence of an error.
	Unsupported instruction list display function	Lists the instructions which are not supported by the logic test function (GX Simulator) if they are included in a sequence program.

8.1 Operating Procedure of Logic Test Function (GX Simulator)

This section describes how to use the logic test function (GX Simulator) for debugging.
Exercise this function with the program | Project name | QLLT |.

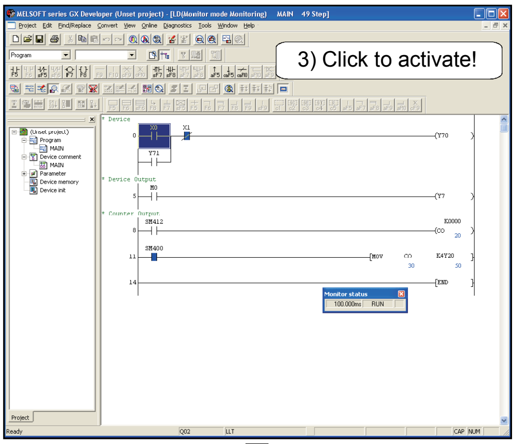


1) Confirm that the project is open, and then click the button.



2) The logic test function (GX Simulator) starts up.

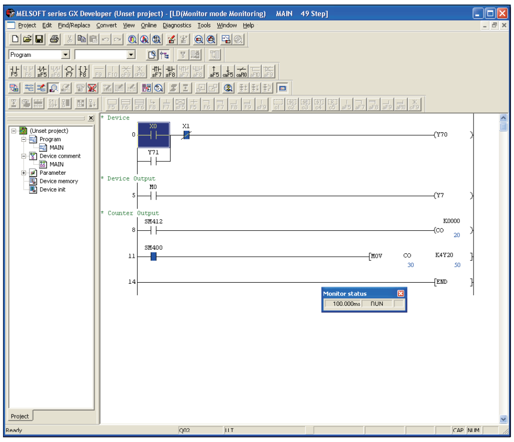
Once the logic test function (GX Simulator) has started up, the parameter and program is automatically written to the logic test function (GX Simulator).
(This is equivalent to the write to PLC function.)



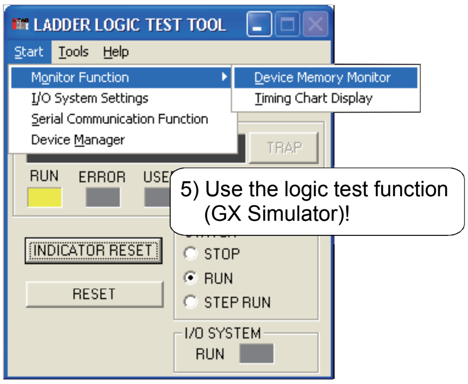
3) Click the window of GX Developer to activate it (select it). The selected screen in Windows switches from GX Simulator to GX Developer.

(To next page)

(From previous page)

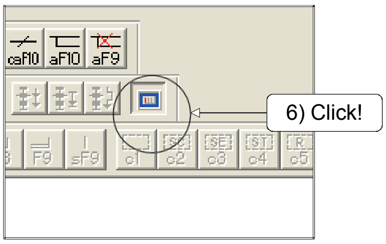


4) Ladder monitoring is executed without a PLC connected (with a PLC in an offline state).



5) Use the logic test function (GX Simulator) to monitor devices, change device values freely, simulate how I/O modules and special modules behave, etc.

Note that if you changed a sequence program according to debug results and intend to use the logic test function (GX Simulator) to do another debug, you need to stop the logic test function (GX Simulator) first and click the button, and then write the sequence program again.



6) After the debug is completed, click the button to terminate the logic test function (GX Simulator).

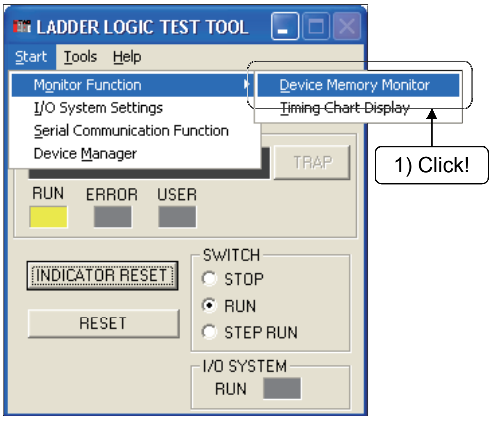
When monitoring is in operation, terminate the monitor mode before terminating the logic test function (GX Simulator).

This section describes how to monitor device status, turn bit devices on/off forcibly, change word device values, etc.

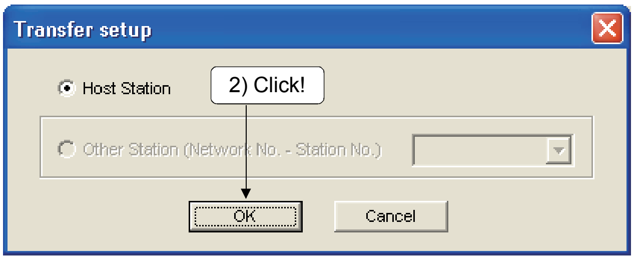
Exercise this function with the program | Project name | QLLT |.

(1) Turning bit devices on or off forcibly

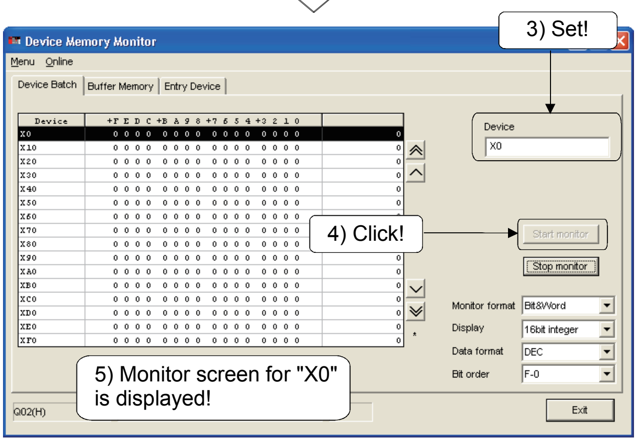
In the example operation below, bit device "X" is monitored and "X0" is forcibly turned on.



1) On the logic test function (GX Simulator) window, click [Start] → [Monitor Function] → [Device Memory Monitor] menu.



2) Check the Host Station and click the | OK | button on the Transfer setup window.



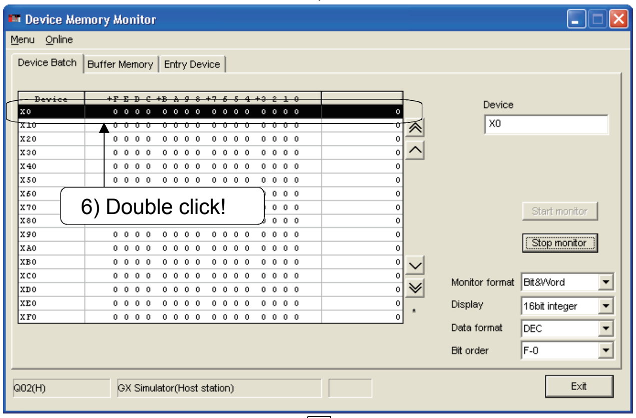
3) Set X0 in the Device column on the Device Memory Monitor window.

4) Click the | Start monitor | button.

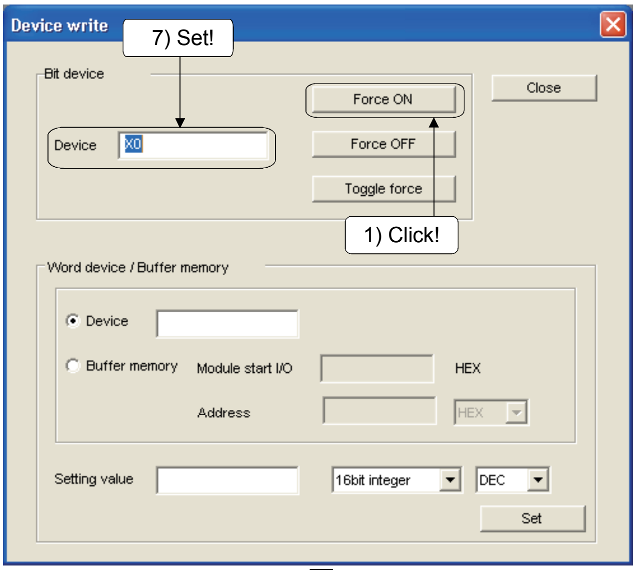
5) Status of bit device "X0" is displayed. Monitor the status of "X0" in this window.

(To next page)

(From previous page)

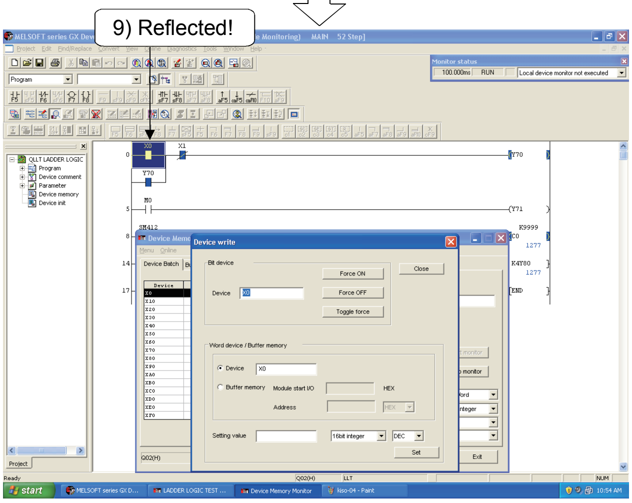


6) Double-click a device No. you wish to turn on/off forcibly.
(In this example, "X0" is double-clicked.)



7) Set X0 in the Device column of the Bit device column on the Device write window.

8) Click the Force ON button.

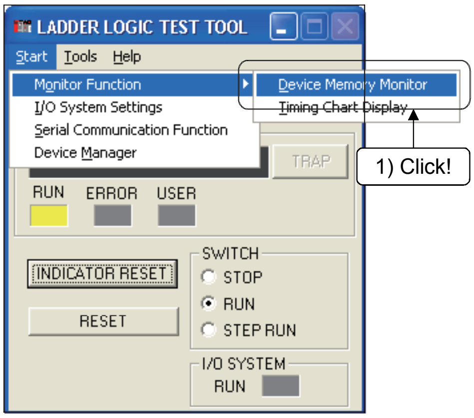


9) The result of the device being turned on is reflected on the ladder monitoring display.

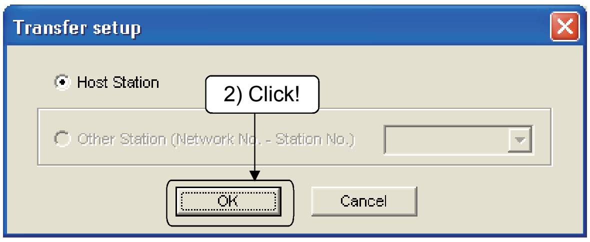
Click the Force OFF button on the Device write window to turn the device off.

(2) Changing word device values

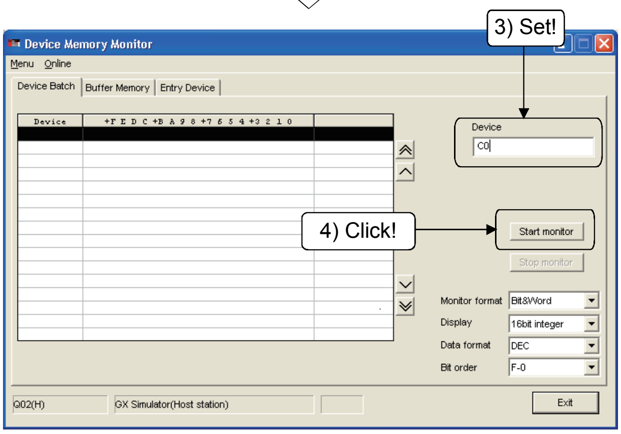
In the example operation below, word device "C (current value)" is monitored and "C0" is changed to "5".



1) On the logic test function (GX Simulator) window, click [Start] → [Monitor Function] → [Device Memory Monitor] menu.



2) Check the Host Station and click the OK button on the Transfer setup window.

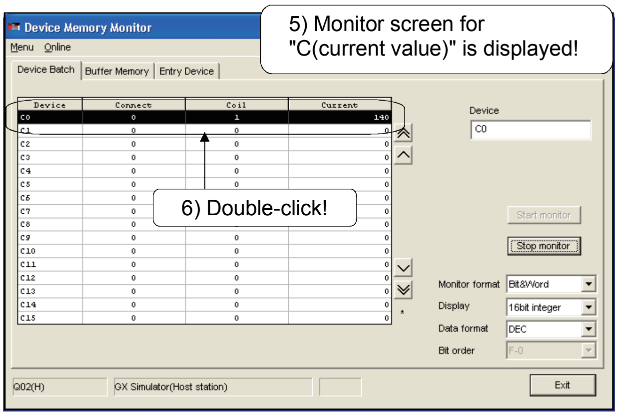


3) Set C0 in the Device column on the Device Memory Monitor window.

4) Click the Start monitor button.

(To next page)

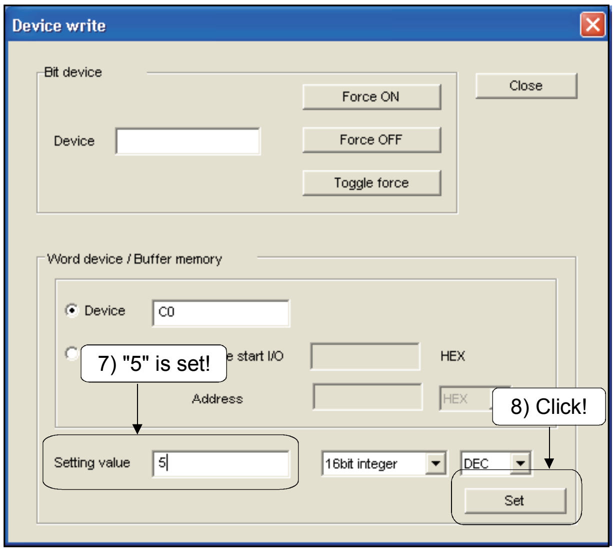
(From previous page)



5) Status of bit device "C(current value)" is displayed. Monitor the status of "C(current value)" in this window.

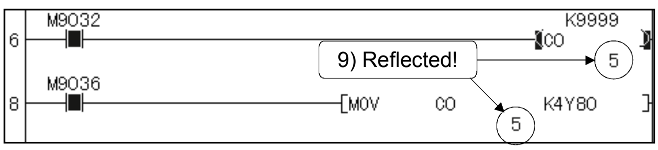
Note that the change on the value of "C0" is visibly recognizable in the logic test function (GX Simulator) window and ladder monitoring screen.

6) Double-click the device No. to change its device value.
(In this example, "C0" is double-clicked.)



7) Enter a value in the Setting value column of the Word device/Buffer memory column on the Device write window.

8) Click the Set button.



9) The result of the device value being set to "5" is reflected on the ladder monitoring display.

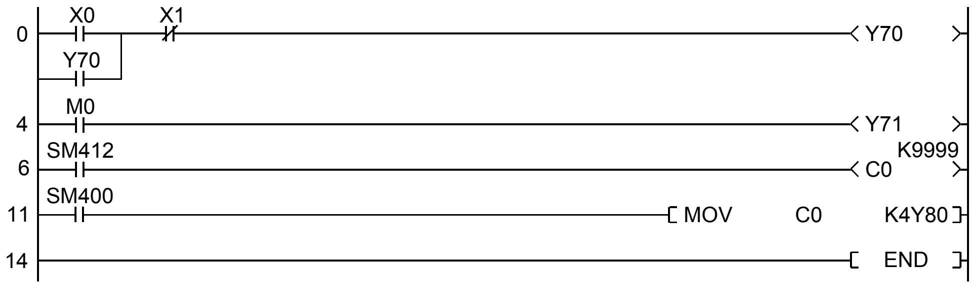
8.3 I/O System Settings Function

The I/O system settings function allows GX Simulator to simulate how external devices behave.
The following two setting methods are available for performing I/O system settings.

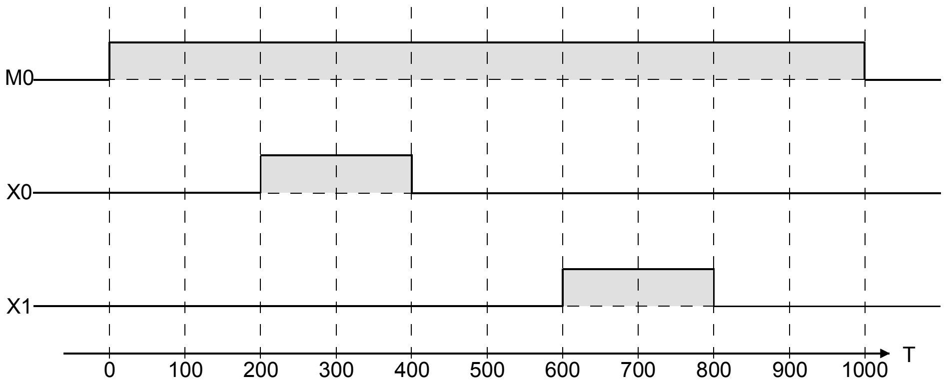
- Device value input: Specifies a device value that is to be set the set time after the conditions are fulfilled.
- Timing chart input: Sets a timing chart that is followed once the conditions are fulfilled.

Use the following example for exercise.

<<Example program>> Project name | QLLT



<<Signal timing>>



[Time unit: 10ms]

<<Operations to simulate>>

(1) The lamp Y71 on the demonstration machine lights when M0 turns on during the device test by GX Developer.

(2) An exterior person confirms the lamp Y71 lighting, turns on X0, and then turns it off. (T = 200) (T = 400)

(3) Confirm that the self-maintaining ladder in the sequence program is in operation, and then turn on X1 to reset the maintaining status. After that, turn off X1 again. (T = 600) (T = 800)

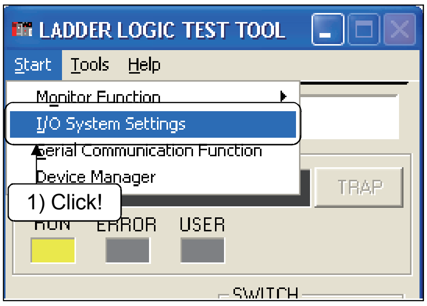
(4) Turn off M0. (T = 1000)

In the example operation below, settings listed in the following table are made.

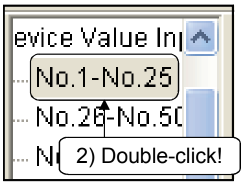
Signal name	Setting method
M0	Device value input
X0, X1	Timing chart input

8.3.1 Device value input

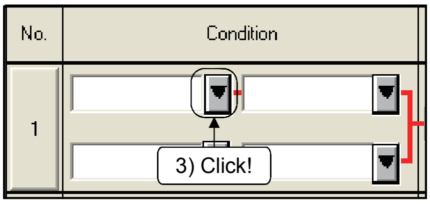
In the example operation below, a simulation where M0 turns off 10 s after M0 turns on is set.



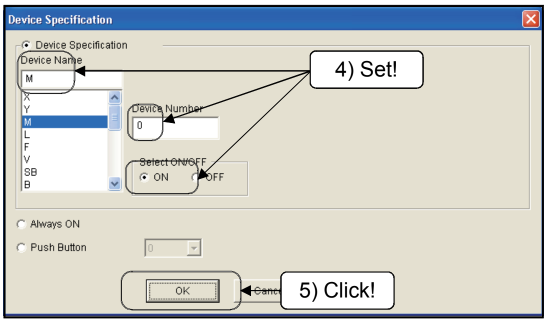
1) Click [Start] and then [I/O System Settings] in the window of the logic test function (GX Simulator).



2) Double-click "Device Value Input (No.1-No.25)" in the I/O System Settings tree.



3) Click the upper ▼ button in the No.1 area.

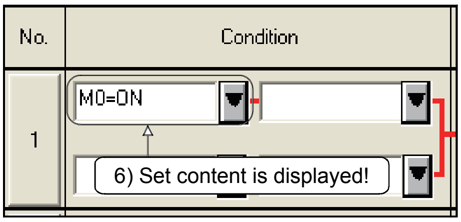


4) Set Device Name (M), Device Number (0), and Select ON/OFF (ON).

5) Click the OK button.

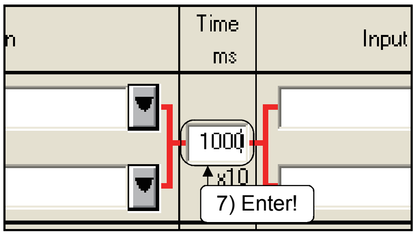
(To next page)

(From previous page)

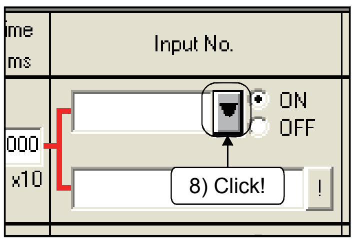


6) The set contents are displayed.

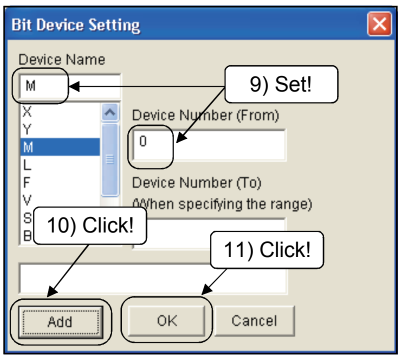
Note that this can be set by directly typing "M0=ON" on a keyboard.



7) Enter "1000" (10 s) in the text box of "Time".



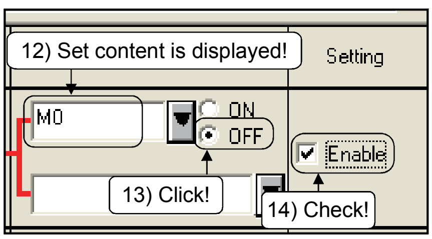
8) Click the upper ▼ button in the Input No. area.



9) Set Device Name (M) and Device Number (0).

10) Click the Add button.

11) Click the OK button.



12) The set contents are displayed.

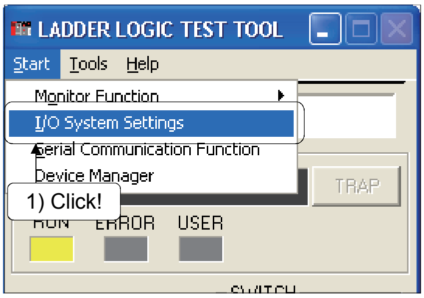
Note that this can be set by directly typing "M0" on a keyboard.

13) Click the "OFF" radio button.

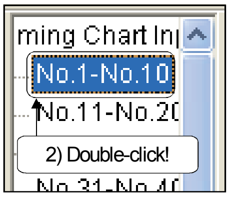
14) Click the Enable checkbox to enable the settings.

8.3.2 Timing chart value input

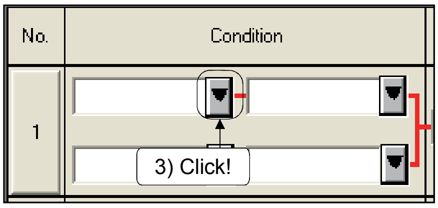
In the example operation below, the following simulation is set.
M0 turns ON → X0 turns ON 2 s later, and turns OFF 4 s later.
X1 turns ON 6 s later, and turns OFF 8 s later



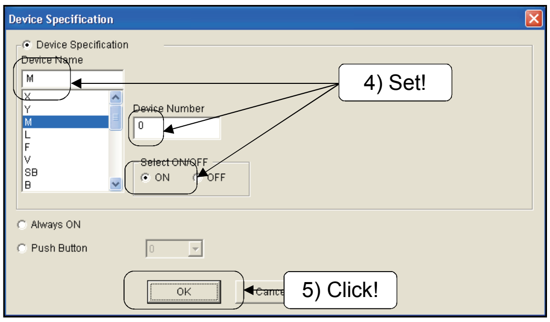
1) Click [Start] and then [I/O System Settings] in the window of the logic test function (GX Simulator).



2) Double-click "Timing Chart Input" in the I/O System Settings tree.

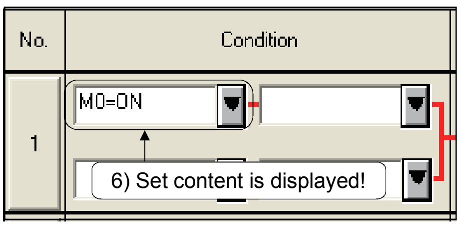


3) Click the upper ▼ button in the No. area.



4) Set Device Name (M), Device Number (0), and Select ON/OFF (ON).

5) Click the OK button.

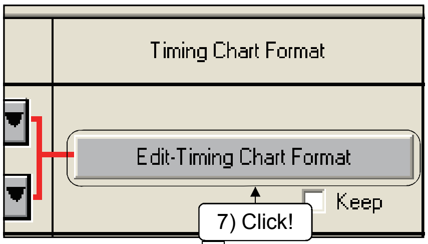


6) The set contents are displayed.

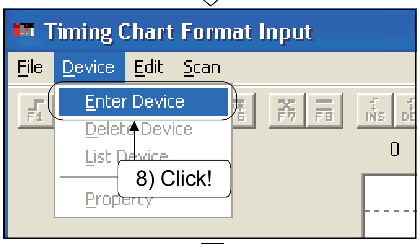
Note that this can be set by directly typing "M0=ON" on a keyboard.

(To next page)

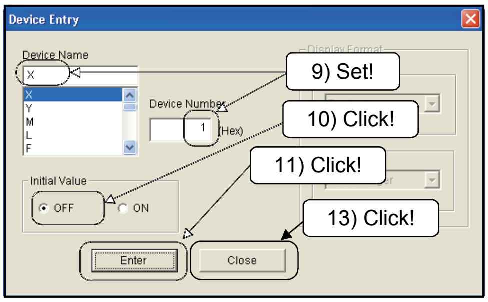
(From previous page)



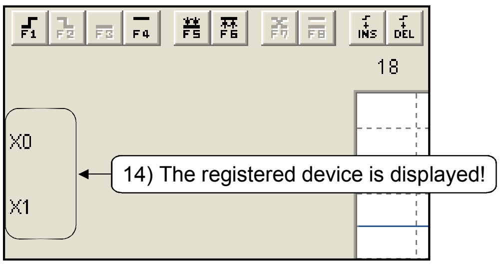
7) Click the [Edit-Timing Chart Format] button.
The Timing Chart Format Input dialog box appears.



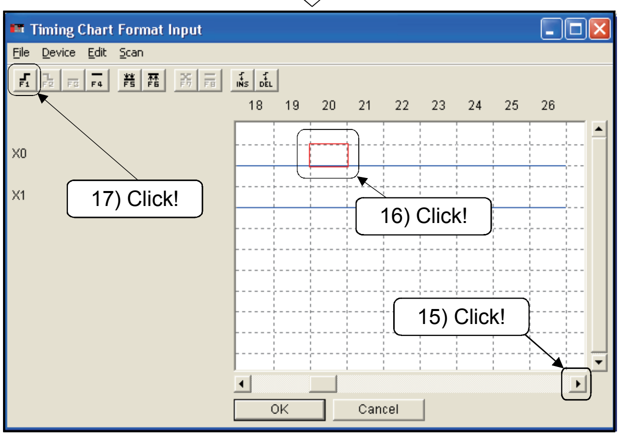
8) Click [Device] and then click [Enter Device].



9) Set Device Name (X) and Device Number (1).
10) Click the "OFF" radio button.
11) Click the [Enter] button.
X1 is registered.
12) Follow the procedures 9) to 11) again to register "X0" (default: OFF).
13) Click the [Close] button.



14) The registered device is displayed.



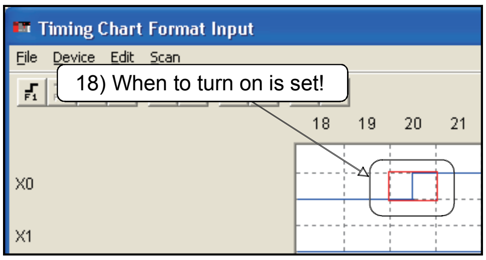
15) Click the [▶] button to display 20th scan.

16) Click 20th scan of the X0 line to specify it.

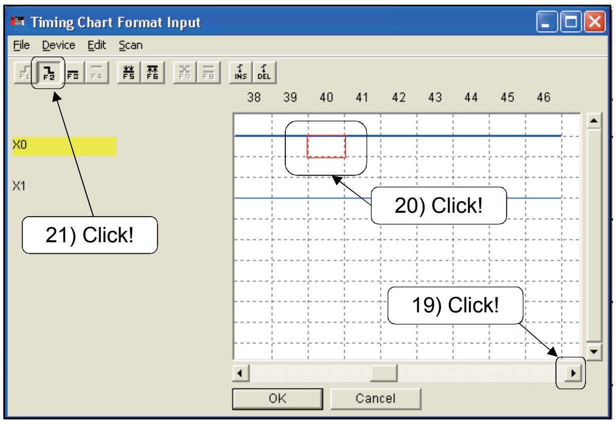
17) Click [F1], or click [Edit], [Bit Device], and [Device ON] sequentially.

(To next page)

(From previous page)



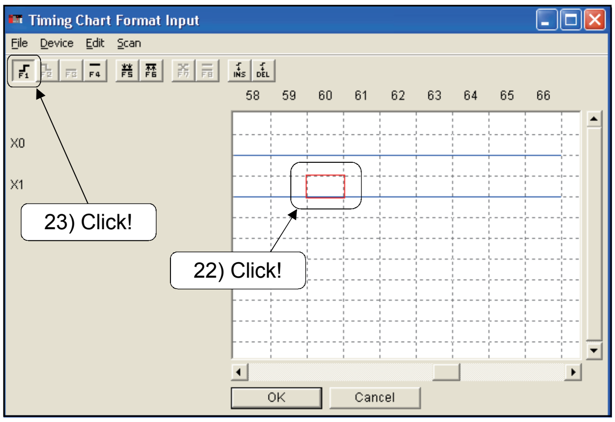
18) The timing at which X0 turns on is set.



19) Click the button to display 40th scan.

20) Click 40th scan of the X0 line to specify it.

21) Click , or click [Edit], [Bit Device], and [Device OFF] sequentially.



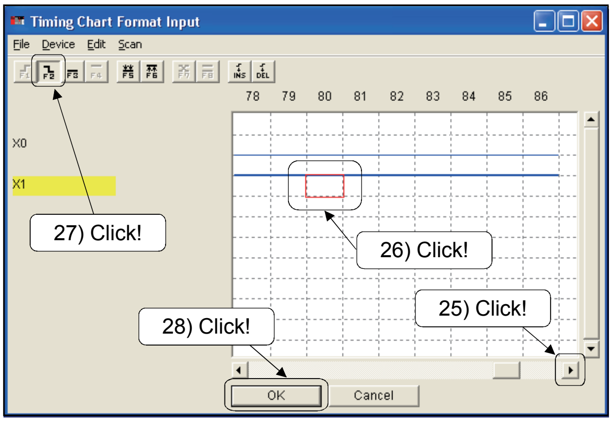
22) The timing at which X0 turns off is set.

23) Click 60th scan of the X1 line to specify it.

24) Click , or click [Edit], [Bit Device], and [Device ON] sequentially to set the timing at which X1 turns ON.

(To next page)

(From previous page)

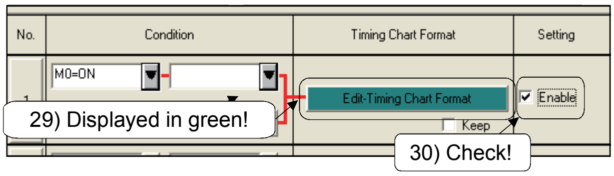


25) Click the [▶] button to display 80th scan.

26) Click 80th scan of the X1 line to specify it.

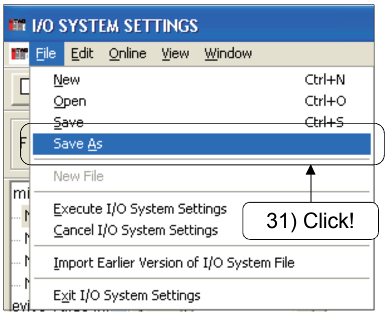
27) Click [F2 icon], or click [Edit], [Bit Device], and [Device OFF] sequentially to set the timing at which X1 turns off.

28) Click the OK button.

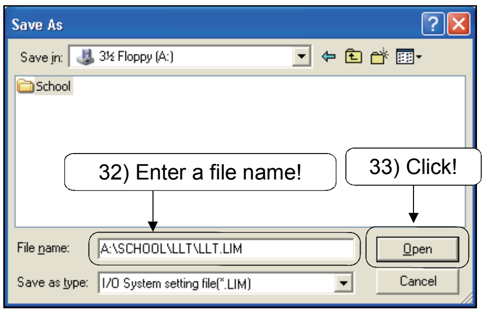


29) The set timing chart is displayed in green.

30) Click the Enable checkbox to enable the settings.



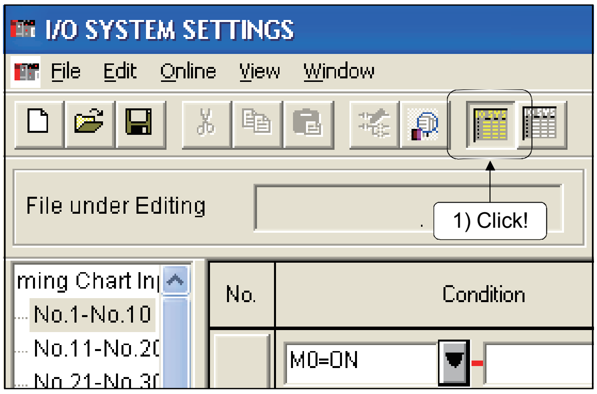
31) After the settings are complete, click [File] and then [Save as] to save the I/O system settings.



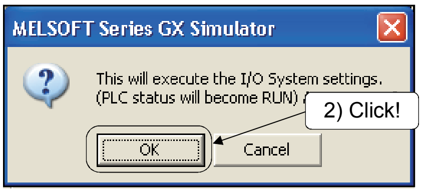
32) Enter a file name.
"A:\SCHOOL\LLT\LLT.LIM" is entered here as an example.

33) Click the Open button.

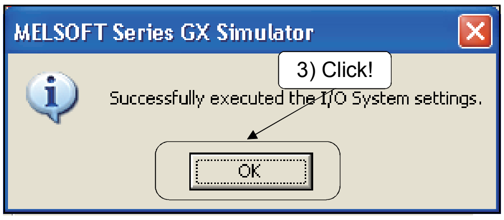
8.3.3 Executing the I/O system settings



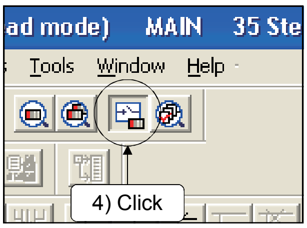
1) Click the icon on the icon line of the I/O SYSTEM SETTINGS dialog box or click [File] and [Execute I/O System Settings] sequentially.



2) Click the OK button.



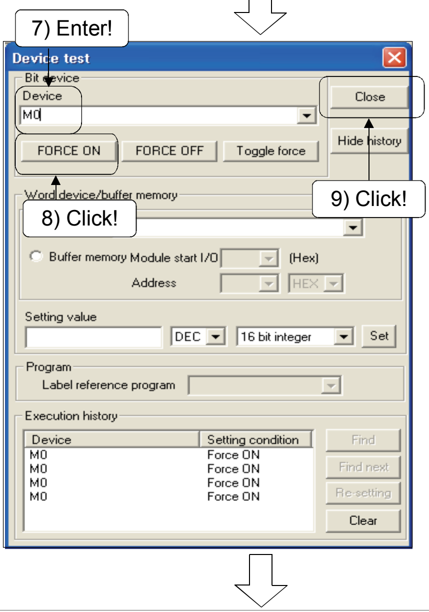
3) Click the OK button.



4) Click the button in the GX Developer window.

(To next page)

(From previous page)



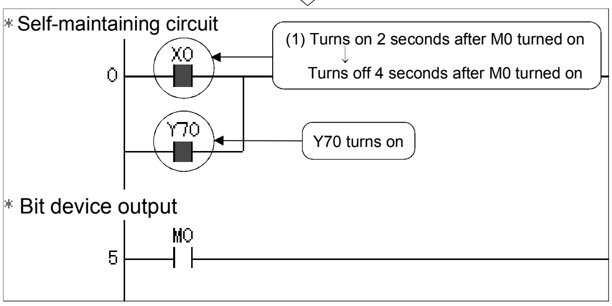
7) Enter "M0".

8) Click the FORCE ON button.

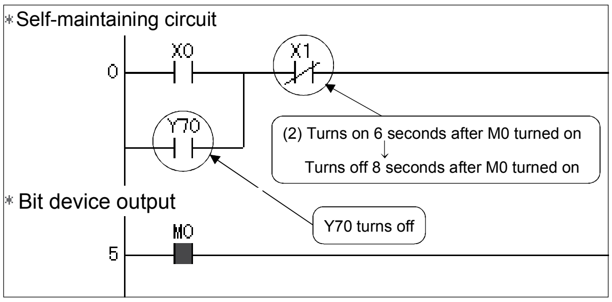
9) Click the Close button.

10) The simulation set in the I/O system settings starts. To do this, refer to <<Signal timing>> at the beginning of Section 8.3.

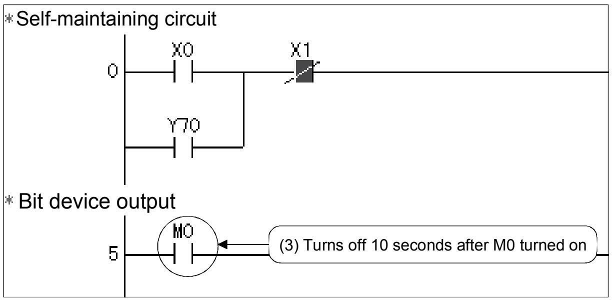
(1) X0 turns on 2 s after M0 turned ON.
X0 turns off 4 s after M0 turned ON.
(While X0 is ON, the self-maintaining ladder operates and Y70 stays ON.)



(2) X1 turns ON 6 s after M0 turned on.
X1 turns OFF 8 s after M0 turned on.
(While X1 is ON, Y70 in the self-maintaining ladder stays OFF.)



(3) M0 turns OFF 5 s after it turned ON.



REFERENCE

The logic test function (GX Simulator) takes 100ms to execute one scan. Time taken for one scan is the set time for constant scan. (Default is set to 100ms.)
This is designed to operate user-created sequence programs equally regardless of the performance of personal computers.
To change the default time of 100ms, configure the parameter setting in GX Developer and set any value as time for constant scan.
(QCPU must be operating to do this.)

CHAPTER 9 MAINTENANCE

9.1 Typical Troubles

The following bar graph shows the ratio of faulty parts and causes of PLC errors.
[Source: Inspection made by JEMA (The Japan Electrical Manufacture's Association)]

Figure 9.1 Faulty parts on PLCs (multiple answers allowed)

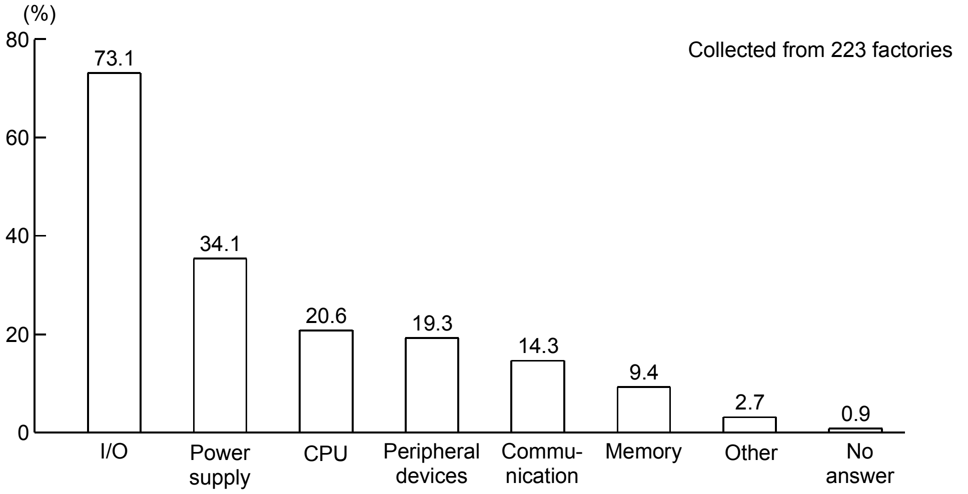
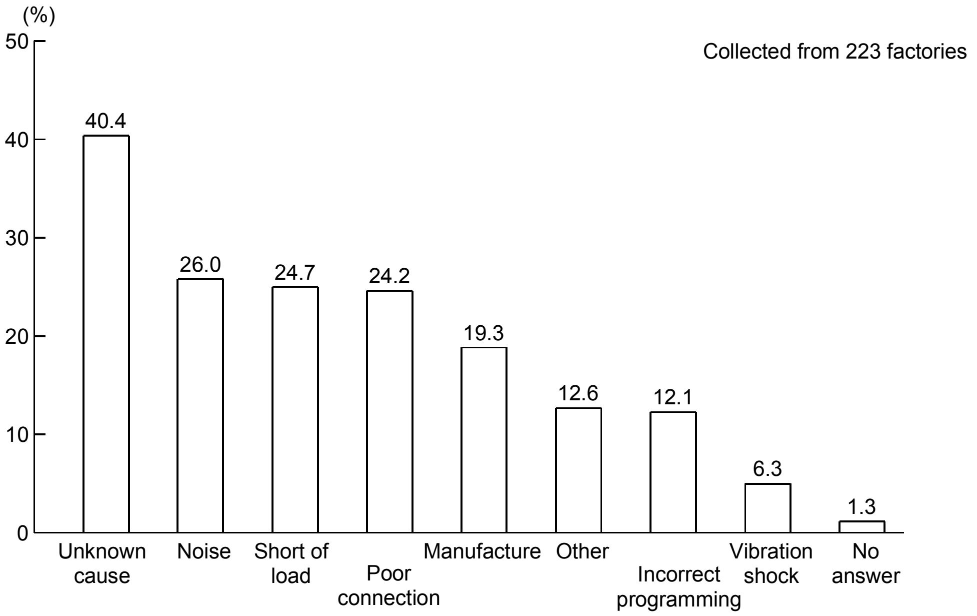


Figure 9.2 Causes of PLC faults (multiple answers allowed)



9.2 Maintenance

To keep PLCs in the best operating condition, conduct the following daily inspection and periodic inspection.

(1) Daily inspection
The items that must be inspected daily are listed in table 9.1.

Table 9.1 Daily inspection

Item	Inspection item		Inspection	Judgment criterion	Remedy
1	Installation of base unit		Check that fixing screws are not loose and the cover is not dislocated.	The module fixing hook must be engaged and installed securely.	Retighten the screws.
2	Installation of I/O module		Check that the module is not dislocated and the unit fixing hook is engaged securely.	The module fixing hook must be engaged and installed securely.	Securely engage the unit fixing hook. Or tighten with the screw.
3	Connecting conditions		Check for loose terminal screws.	Screws should not be loose.	Retighten the terminal screws.
			Check for distance between solderless terminals.	The proper clearance should be provided between Solderless terminals.	Correct.
			Check the connector part of the cable.	Connections should no be loose.	Retighten the connector fixing screws.
4	Module indication LED	Power supply module "POWER" LED*1	Check that the LED is ON.	The LED must be ON. (Abnormal if the LED is OFF)	Refer to QCPU (Q mode) User's Manual
		CPU module "RUN" LED	Check that the LED is ON in RUN status.	The LED must be ON. (Abnormal if the LED is OFF)	
		CPU "ERROR" LED	Check that the LED is OFF.	The LED must be Off. (Abnormal if the LED is ON or flickering.)	
		CPU module "BAT.ARM" LED	Check that the LED is OFF.	The LED must be Off. (Abnormal if the LED is ON)	
		Input LED	Check that the LED turns ON and OFF.	The LED must be ON when the input power is turned ON. The LED must be extinguished when the input power is turned OFF. (Abnormal if the LED does not turn ON or turn OFF as indicated above.)	
		Output LED	Check that the LED turns ON and OFF.	The LED must be ON when the input power is turned ON. The LED must be extinguished when the input power is turned OFF. (Abnormal if the LED does not turn ON or turn OFF as indicated above.)	

(2) Periodic inspection

The items that must be inspected one or two times every 6 months to 1 year are listed below. Also perform this inspection when following changes are made; the equipment is moved or modified, layout of the wiring is changed, the power supply module is changed, etc.

Table 9.2 Periodic Inspection

Item	Inspection item		Inspection	Judgment criterion	Remedy
1	Ambient environment	Ambient temperature	Measure with a thermometer and a hygrometer. Measure corrosive gas.	0 to 55 °C	When the sequencer is used in the board, the ambient temperature in the board becomes the ambient temperature.
		Ambient humidity		5 to 95%RH*1	
		Ambience		Corrosive gas must not be present.	
2	Power voltage		Measure a voltage across the terminals of 100/200VAC and 24VDC.	85 to 132VAC	Change the power supply.
				170 to 264VAC	
3	Installation	Looseness, rattling	Move the module to check for looseness and rattling.	The module must be installed fixedly.	Retighten the screws. If the CPU, I/O, or power supply module is loose, fix it with screws.
		Adhesion of dirt and foreign matter	Check visually.	Dirt and foreign matter must not be present.	Remove and clean.
4	Connection	Looseness of terminal screws.	Try to further tighten screws with a screwdriver.	Screws must not be loose.	Retighten the terminal screws.
		Proximity of solderless terminals to each other.	Check visually.	Solderless terminals must be positioned at proper intervals.	Correct.
		Looseness of connectors	Check visually.	Screws must not be loose.	Retighten the connector fixing screws.
5	Battery		Check on the monitor mode of the GX Developer that M51 or SM52 is turned OFF.	(Preventive maintenance)	Even if the lowering of a battery capacity is not shown, replace the battery with a new one if a specified service life of the battery is exceeded.
6	Spare product		Install the product on and the actual PLC.	Operation must meet the specifications.	Use the normal product on the actual PLC as a spare product.
7	Check on stored program		Compare the stored program with the running program.	The two programs must be identical.	Correct if there are differences.
8	Fan (heat exchanger) filter		Rotational status Rotational sound Clogging	Rotation must be without sounds or clogging.	Exchange and clean if any abnormality is found.
9	Analog I/O		Check the offset/gain value.	The value must be identical with the specifications (design value).	Correct if there are differences.

*1: When AnS Series Module is included in the system, the judgment criteria will be from 10 to 90 % RH.

9.3 Consumable Products

Backup batteries on PLCs are consumable products.

9.4 Service Life of Output Relay

The relays of modules relaying on contacts to output are subject to consumption due to the switching operation.
When a relay that is directly mounted on the print board of an output module is consumed, it is necessary to replace the output module itself.

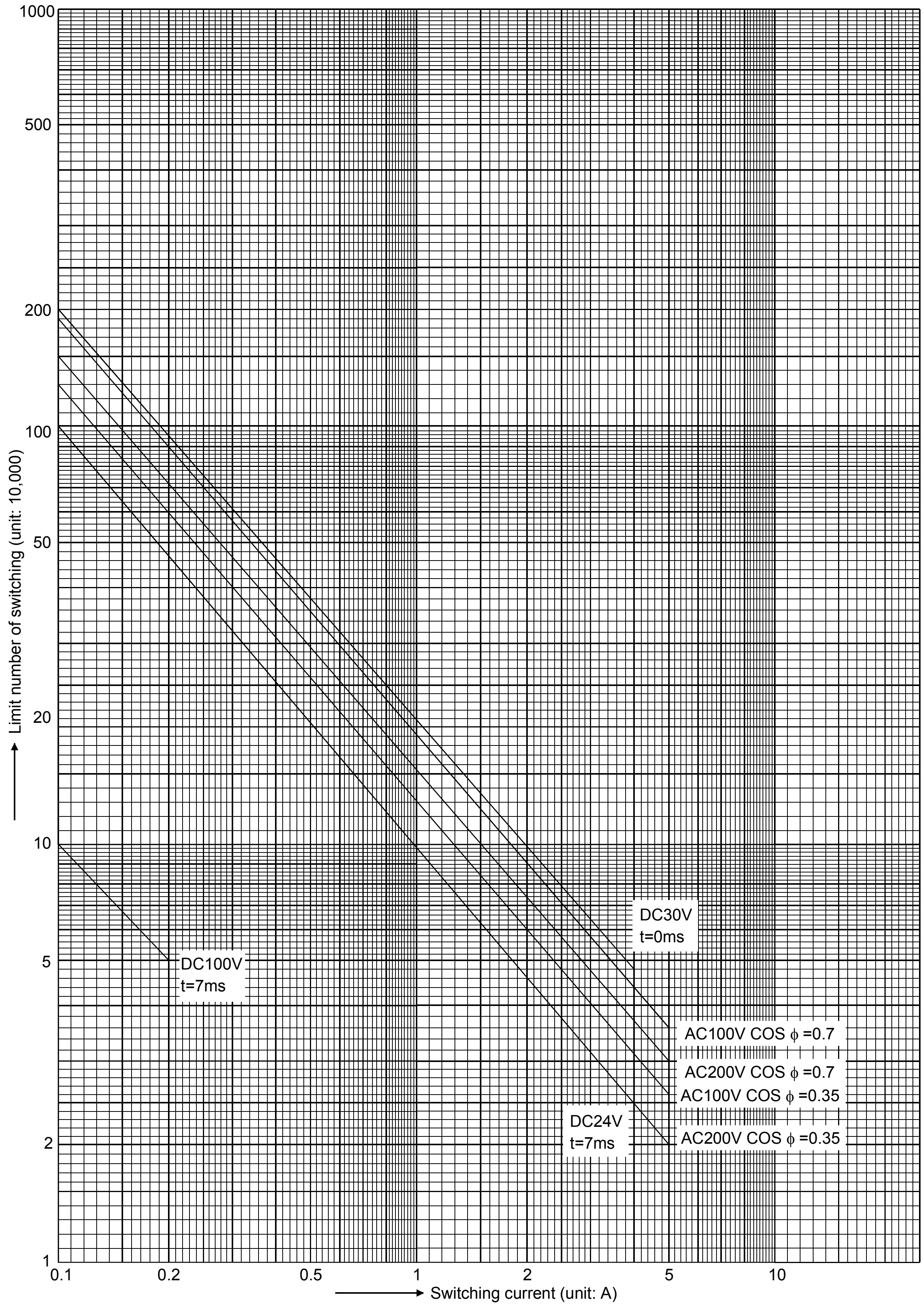


Figure 9.3 Life characteristics of output relay's contact

9.5 Spare Products

Alternative products are easily purchased through the Mitsubishi service centers or local Mitsubishi representatives in Japan. So the purchasing documents can be prepared even after an accident. Note, however, that for foreign-related products such as exported products it is necessary to send alternative products beforehand.
To ease maintenance, refer to the following tips at design work.

(1) Module type easily replaced
The building block type modules feature easy replacement. Their structures make it possible to replace just the faulty module to complete the replacement.

(2) Memory type
To use the standard RAMs or SRAM memory cards, the backup battery is required.
The standard ROMs, Flash cards, and ATA cards do not require the battery for use, and besides, they prevent unintentional program changes due to human-related mistakes. They are recommended to be employed especially in products for export.

(3) Reducing the number of module types
Reducing the number of module types is an efficient way for reducing the umber of spare product types.

(4) Reserving some I/O points
By not using all the I/O points on 16-, 32-, and 64-point I/O modules but reserving 10% to 20% of them, it is possible to just make changes on wiring and programs (I/O signals) instead of replacing the faulty module with a spare module. This is efficient when there are no spare modules.

(5) Creating a document
The fact that PLC programs are easily modified may lead to inconsistency between an operating program and documents (i.e. ladder diagram, program list). Keep updating the document.
To do this, using a printer is efficient.

(6) Being experienced in peripheral devices
Being experienced in DOS/V personal computers, PUs, printers, etc. helps quick recovery from an accident.

(7) Spare products

Table 9.3 Spare products

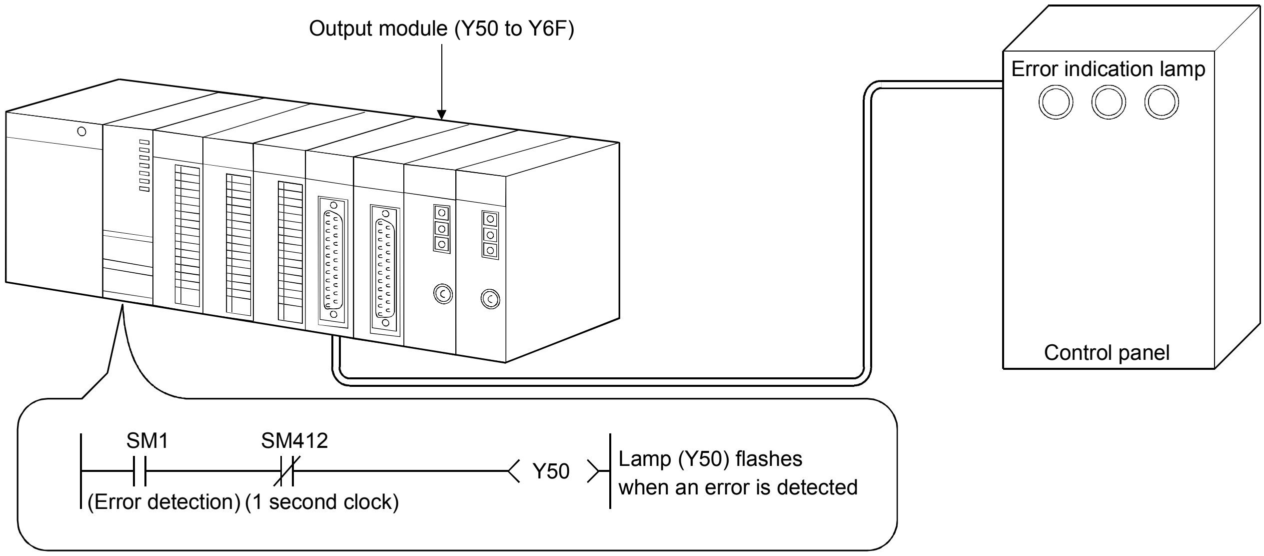
No.	Product name	Quantity	Remark
1	Battery	One or two	Storage lives of lithium batteries are about 5 years. So it is recommended not to keep the stock all the time but to purchase them when needed. However, keep stock of 1 or 2 for accidental situation.
2	Floppy disk	As required	Backup FDs for the startup software and FDs for user
3	I/O module	One per module type	Note that I/O modules tend to have an error during test operation. Also note that the contacts of relay output modules are subject to consumption in long-term use.
4	CPU module	One used model	CPU modules and memory cards are the core parts of a PLC. This means that an error on them causes the system to go down.
5	Memory card	One used model	
6	Power supply module	One used model	Same as above. As power supply modules are highly subject to temperature rise, their service lives tend to be shorter than supposed in a high ambient temperature environment.
7	Peripheral devices	As required	A PU is recommended as a spare of DOS/V personal computer.

The following shows examples of maintenance supporters, which are devices that automatically notify an operator or maintenance person faults or operation status of PLC-used systems or devices.

1. **Error indication by commercial lamp**

Connect an error LED lamp to the output module of a PLC so that the lamp flashes at an error occurrence.

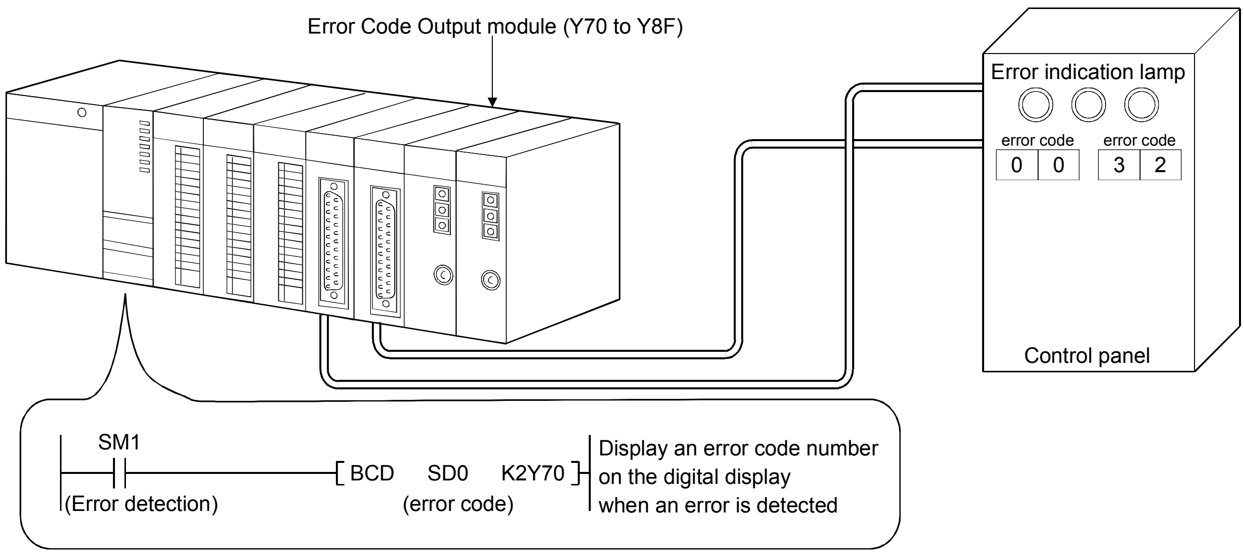
Lamp flicker



2. **Displaying an error code on a commercial digital display**

Connect the digital display to the output module of a PLC so that the error code number of the detected error is indicated on the digital display.

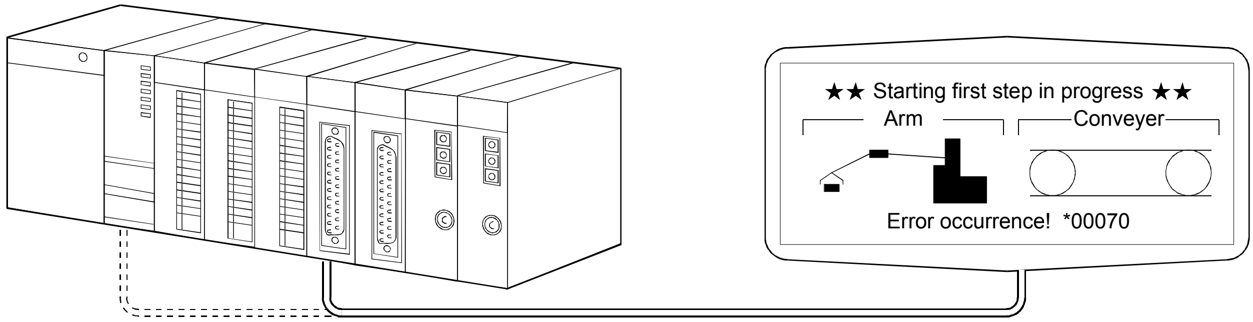
Display in numerical values



3. Displaying the description of a detected error on the screen

The details of an error occurred on a PLC can be displayed on an external CRT screen, plasma screen, liquid crystal screen, etc.

Displaying on a screen



MELSEC-Q supports a wide variety of GOTs (Graphic Operation Terminals).
In addition to the error display function, GOTs feature a lot of useful functions such as the graphic monitoring, ladder monitoring, device monitoring, touch-panel switch, and printing function.
(Refer to the catalogs for details.)

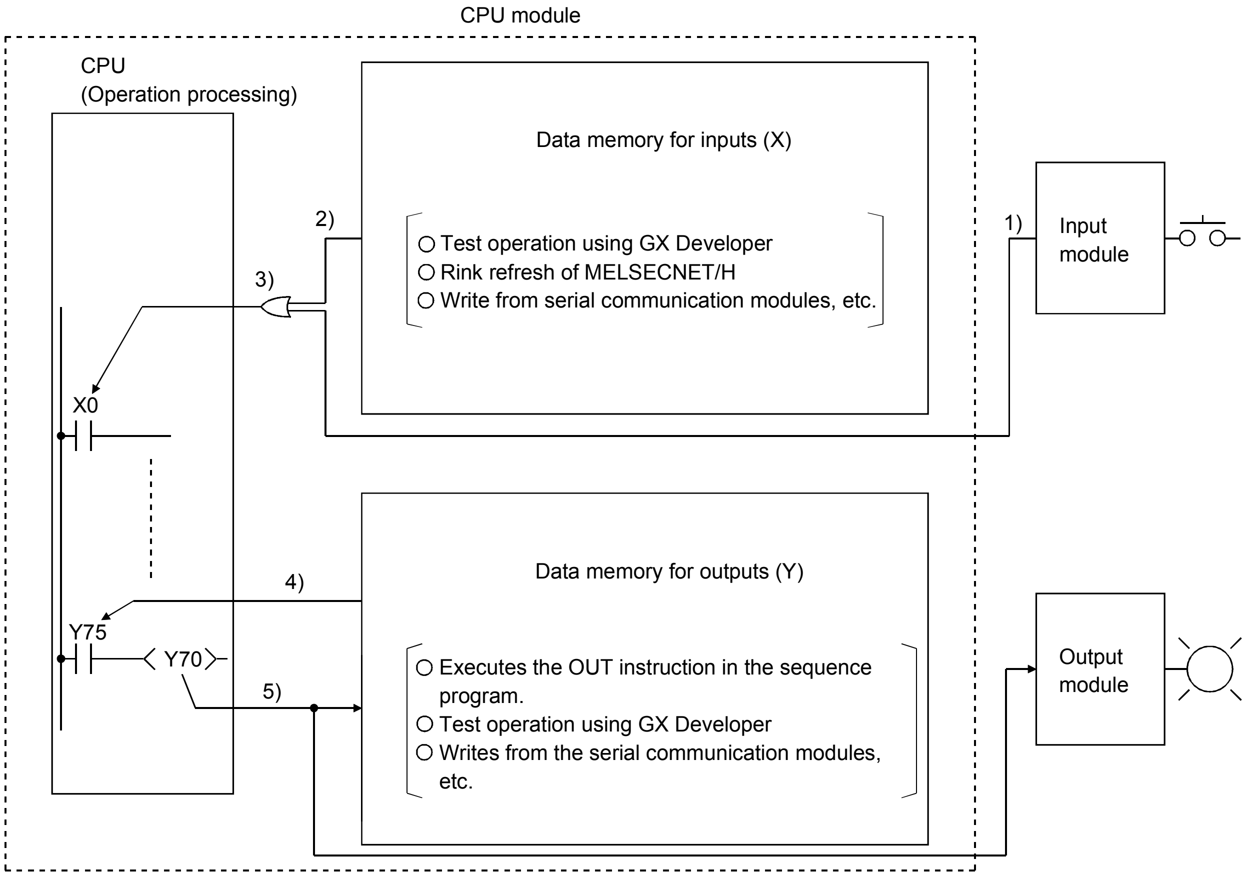
APPENDIX

Appendix 1 I/O Control Mode

The CPU supports two types of I/O control modes; the direct mode and refresh mode.

Appendix 1.1 Direct mode

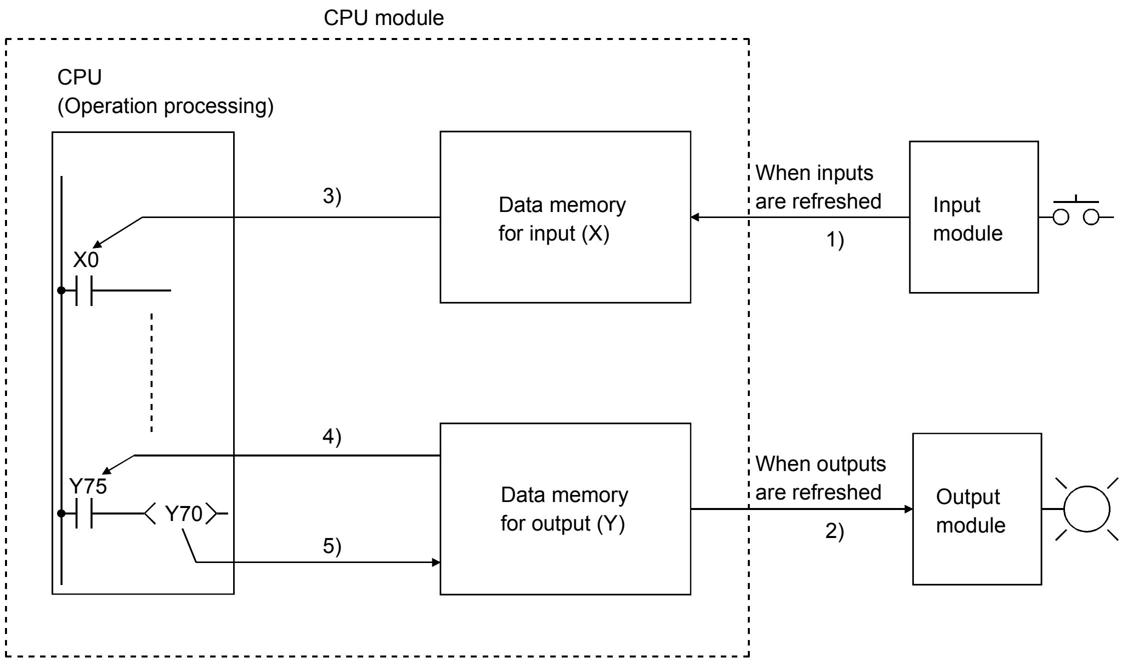
In the direct mode, input signals are imported to a PLC every time they are input and treated it as input information. The operation results of a program are output to the output data memory and the output modules. The following diagram shows the flow of I/O data in the direct mode.



- When the input contact instruction is performed:
  An OR operation is performed in the input information 1) from the input module and input information 2) in the data memory. Then this data is used as input information 3) at sequence program execution.
- When the output contact instruction is performed:
  Output information 4) is read from the output (Y) data memory, and a sequence program is executed.
- When the OUT instruction is performed:
  The sequence program's operation result 5) is output to the output module, and is stored in the output (Y) data memory.
- When the QCPU performs I/O in the direct mode, a sequence program uses DX for inputs and DY for outputs.

Appendix 1.2 Refresh mode

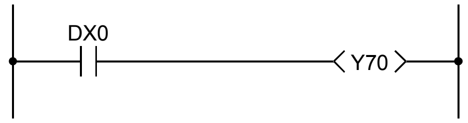
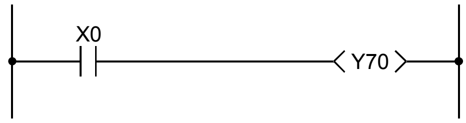
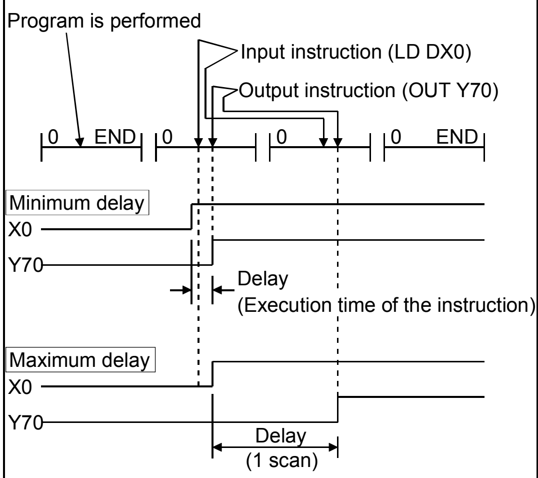
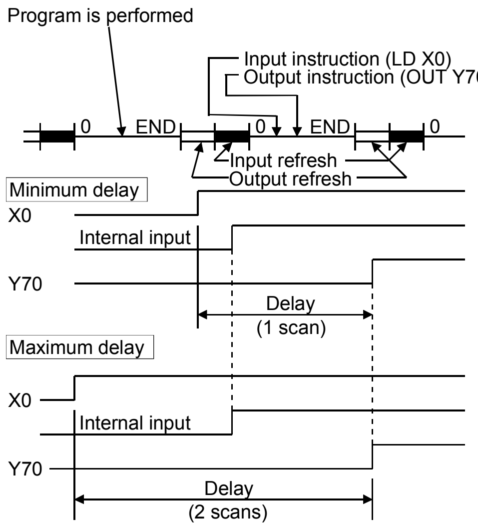
In the refresh mode, all changes occurred to an input module are imported to the input data memory in a PLC CPU at a time before each time a scan is performed. The data memory is used when performing the operation.
The operation results made in a program for output (Y) are stored to the output data memory each time the operation is made. All the data stored in the output data memory is batch-output to the output module after the execution of the END instruction.
The following diagram shows the flow of I/O data in the refresh mode.



- Input refresh
  Input data in the input module is batch-read 1) before the execution of 0 step, and stored to the data memory for input (X).
- Output refresh
  Data 2) in the data memory for output (Y) is batch-output to the output module before the execution of 0 step.
- When the input contact instruction is performed:
  Input data is read from the input (X) data memory 3), and a sequence program is performed.
- When the output contact instruction has been performed:
  Output data 4) is read from the output (Y) data memory, and a sequence program is performed.
- When the output OUT instruction is performed:
  The sequence program operation result 5) is stored in the output (Y) data memory.

Appendix 1.3 Comparisons between the direct mode and refresh mode

In the example ladder given below, input X0 turning on leads to output Y70 turning on.

Item	Direct mode	Refresh mode
1, Ladder example		
2. Response lag from when input is changed to when output is changed accordingly	 • The delay time ranges from 0 scan (only execution time of the instruction) to 1 scan. • The delay can be anywhere from 0 to 1 scan.	 • The delay time ranges from 1 scan to 2 scans. • The delay 1 to 2 scans.
3. Execution time of the I/O instructions	• Longer time than the refresh mode is needed as a PLC accesses I/O modules in the direct mode.	• Generally, only short time is needed as a PLC accesses data memory.
4. Scan time	• The longer of the execution time of the I/O instruction is, the longer the scan time is. • The actual scan time is the time of program execution.	• The shorter of the execution time of the I/O instruction is, the shorter the scan time is. • The actual scan time is the total time of program execution, input transfer, and output transfer.

Appendix 2 Instruction Table

For SFC-related instructions, refer to QCPU (Q Mode)/QnACPU Programming Manual (SFC) $\boxed{\text{SH-080023}}$.

Appendix 2.1 Sequence instructions

(1) Contact instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Contact	LD		• Starts logic operation (Starts a contact logic operation)		∗1	● ∗3
	LDI		• Starts logical NOT operation (Starts b contact logic operation)			
	AND		• Logical product (a contact series connection)			
	ANI		• Logical product NOT (b contact series connection)			
	OR		• Logical sum (a contact parallel connection)			
	ORI		• Logical sum NOT (b contact parallel connection)			
	LDP		• Starts leading edge pulse operation		∗2	● ∗3
	LDF		• Starts trailing edge pulse operation			
	ANDP		• Leading edge pulse series connection			
	ANDF		• Trailing edge pulse series connection			
	ORP		• Leading edge pulse parallel connection			
	ORF		• Trailing edge pulse parallel connection			

REMARK

1) ∗1: The number of steps may vary depending on the device being used.

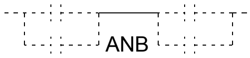
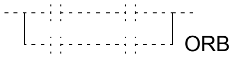
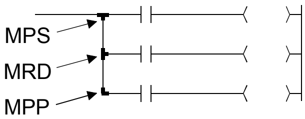
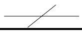
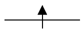
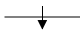
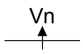
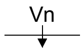
Device	Number of Steps
Internal device, file register (R0 to R32767)	1
Direct access input (DX)	2
Devices other than above	3

2) ∗2: The number of steps may vary depending on the device and type of CPU module being used.

Device	Number of Steps	
	QCPU	QnACPU
Internal device, file register (R0 to R32767)	1	1
Direct access input (DX)	2	2
Devices other than above	3	3

3) ∗3: The subset is effective only with QCPU.

(2) Connection instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Connection	ANB		• AND between logical blocks (Series connection between logical blocks)		1	
	ORB		• OR between logical blocks (Series connection between logical blocks)			
	MPS		• Memory storage of operation results		1	
	MRD		• Read of operation results stored with MPS instruction			
	MPP		• Read and reset of operation results stored with MPS instruction			
	INV		• Inversion of operation result		1	
	MEP		• Conversion of operation result to leading edge pulse		1	
	MEF		• Conversion of operation result to trailing edge pulse			
	EGP		• Conversion of operation result to leading edge pulse (Stored at Vn)		1	
	EGF		• Conversion of operation result to trailing edge pulse (Stored at Vn)			

(3) Output instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Output	OUT	—( )—	• Device output		*1	
	SET	[SET D]	• Set device	*2	*1	
	RST	[RST D]	• Reset device	*2	*1	
	PLS	[PLS D]	• Generates 1 cycle program pulse at leading edge of input signal		2	
	PLF	[PLF D]	• Generates 1 cycle program pulse at trailing edge of input signal			
	FF	[FF D]	• Reversal of device output		2	
	DELTA	[DELTA D]	• Pulse conversion of direct output		2	
	DELTAP	[DELTAP D]				

REMARK

1) *1: The number of steps may vary depending on the device being used.
For details of the number of steps, refer to a page having the explanation of each instruction.
2) *2: The execution condition applies only when an annunciator (F) is in use.

(4) Shift instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Shift	SFT	[SFT D]	• 1-bit shift of device		2	
	SFTP	[SFTP D]				

(5) Master control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Master control	MC	MC n D	• Starts master control		2	
	MCR	MCR n	• Resets master control		1	

(6) Termination instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Program end	FEND	FEND	• Termination of main program		1	
	END	END	• Termination of sequence program			

(7) Other Instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Stop	STOP	STOP	• Terminates sequence operation after input condition has been met • Sequence program is executed by placing the RUN/STOP key switch back in the RUN position		1	
Ignored	NOP		• Ignored (For program deletion or space)		1	
	NOPLF	NOPLF	• Ignored (To change pages during Ignored printouts)			
	PAGE	PAGE n	• Ignored (Subsequent programs will be controlled from step 0 of page n)			

(1) Comparison operation instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
16-bit data comparisons	LD=	= S1 S2	• Conductive status when (S1) = (S2) • Non-conductive status when (S1) ≠ (S2)		3	●
	AND=	= S1 S2				
	OR=	= S1 S2				
	LD<>	<> S1 S2	• Conductive status when (S1) ≠ (S2) • Non-conductive status when (S1) = (S2)		3	●
	AND<>	<> S1 S2				
	OR<>	<> S1 S2				
	LD>	> S1 S2	• Conductive status when (S1) > (S2) • Non-conductive status when (S1) ≤ (S2)		3	●
	AND>	> S1 S2				
	OR>	> S1 S2				
	LD<=	<= S1 S2	• Conductive status when (S1) ≤ (S2) • Non-conductive status when (S1) > (S2)		3	●
	AND<=	<= S1 S2				
	OR<=	<= S1 S2				
	LD<	< S1 S2	• Conductive status when (S1) < (S2) • Non-conductive status when (S1) ≥ (S2)		3	●
	AND<	< S1 S2				
	OR<	< S1 S2				
	LD>=	>= S1 S2	• Conductive status when (S1) ≥ (S2) • Non-conductive status when (S1) < (S2)		3	●
	AND>=	>= S1 S2				
	OR>=	>= S1 S2				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
32-bit data comparisons	LDD=	D= S1 S2	• Conductive status when (S1+1, S1) = (S2+1, S2) • Non-conductive status when (S1+1, S1) ≠ (S2+1, S2)		*1	●
	ANDD=	D= S1 S2				
	ORD=	D = S1 S2				
	LDD<>	D<> S1 S2	• Conductive status when (S1+1, S1) ≠ (S2+1, S2) • Non-conductive status when (S1+1, S1) = (S2+1, S2)		*1	●
	ANDD<>	D <> S1 S2				
	ORD<>	D<> S1 S2				
	LDD>	D> S1 S2	• Conductive status when (S1+1, S1) > (S2+1, S2) • Non-conductive status when (S1+1, S1) ≤ (S2+1, S2)		*1	●
	ANDD>	D> S1 S2				
	ORD>	D> S1 S2				
	LDD<=	D<= S1 S2	• Conductive status when (S1+1, S1) ≤ (S2+1, S2) • Non-conductive status when (S1+1, S1) > (S2+1, S2)		*1	●
	ANDD<=	D<= S1 S2				
	ORD<=	D<= S1 S2				
	LDD<	D< S1 S2	• Conductive status when (S1+1, S1) < (S2+1, S2) • Non-conductive status when (S1+1, S1) ≥ (S2+1, S2)		*1	●
	ANDD<	D< S1 S2				
	ORD<	D< S1 S2				
	LDD>=	D>= S1 S2	• Conductive status when (S1+1, S1) ≥ (S2+1, S2) • Non-conductive status when (S1+1, S1) < (S2+1, S2)		*1	●
	ANDD>=	D>= S1 S2				
	ORD>=	D>= S1 S2				

REMARK

*1: The number of steps may vary depending on the device and type of CPU module being used.

Device	Number of Steps	
	QCPU	QnACPU
• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no index modification. • Constant : No limitations	5	3
Devices other than above	3	

QCPUs need the increased number of steps but provide faster processing speed.

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Real number data comparisons	LDE=	E= S1 S2	• Conductive status when (S1+1, S1) = (S2+1, S2) • Non-conductive status when (S1+1, S1) ≠ (S2+1, S2)		3	
	ANDE=	E= S1 S2				
	ORE=	E= S1 S2				
	LDE<>	E<> S1 S2	• Conductive status when (S1+1, S1) ≠ (S2+1, S2) • Non-conductive status when (S1+1, S1) = (S2+1, S2)		3	
	ANDE<>	E<> S1 S2				
	ORE<>	E<> S1 S2				
	LDE>	E> S1 S2	• Conductive status when (S1+1, S1) > (S2+1, S2) • Non-conductive status when (S1+1, S1) ≤ (S2+1, S2)		3	
	ANDE>	E> S1 S2				
	ORE>	E> S1 S2				
	LDE<=	E<= S1 S2	• Conductive status when (S1+1, S1) ≤ (S2+1, S2) • Non-conductive status when (S1+1, S1) > (S2+1, S2)		3	
	ANDE<=	E<= S1 S2				
	ORE<=	E<= S1 S2				
	LDE<	E< S1 S2	• Conductive status when (S1+1, S1) < (S2+1, S2) • Non-conductive status when (S1+1, S1) ≥ (S2+1, S2)		3	
	ANDE<	E< S1 S2				
	ORE<	E< S1 S2				
	LDE>=	E>= S1 S2	• Conductive status when (S1+1, S1) ≥ (S2+1, S2) • Non-conductive status when (S1+1, S1) < (S2+1, S2)		3	
	ANDE>=	E>= S1 S2				
	ORE>=	E>= S1 S2				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Character string data comparisons	LD$=	$= S1 S2	• Compares character string S1 and character string S2 one character at a time. * • Conductive status when (character string S1) = (character string S2) • Non-Conductive status when (character string S1) ≠ (character string S2)		3	
	AND$=	$ = S1 S2				
	OR$=	$= S1 S2				
	LD$<>	$<> S1 S2	• Compares character string S1 and character string S2 one character at a time. * • Conductive status when (character string S1) ≠ (character string S2) • Non-Conductive status when (character string S1) = (character string S2)		3	
	AND$<>	$<> S1 S2				
	OR$<>	$<> S1 S2				
	LD$>	$> S1 S2	• Compares character string S1 and character string S2 one character at a time. * • Conductive status when (character string S1) > (character string S2) • Non-Conductive status when (character string S1) ≤ (character string S2)		3	
	AND$>	$> S1 S2				
	OR$>	$> S1 S2				
	LD$<=	$<= S1 S2	• Compares character string S1 and character string S2 one character at a time. * • Conductive status when (character string S1) ≤ (character string S2) • Non-Conductive status when (character string S1) > (character string S2)		3	
	AND$<=	$<= S1 S2				
	OR$<=	$<= S1 S2				
	LD$<	$< S1 S2	• Compares character string S1 and character string S2 one character at a time. * • Conductive status when (character string S1) < (character string S2) • Non-Conductive status when (character string S1) ≥ (character string S2)		3	
	AND$<	$< S1 S2				
	OR$<	$< S1 S2				
	LD$>=	$>= S1 S2	• Compares character string S1 and character string S2 one character at a time. * • Conductive status when (character string S1) ≥ (character string S2) • Non-Conductive status when (character string S1) < (character string S2)		3	
	AND$>=	$>= S1 S2				
	OR$>=	$>= S1 S2				

REMARK

1) *: The conditions under which character string comparisons can be made are as shown below.
   - Match: All characters in the strings must match
   - Larger string: If character strings are different, determines the string with the largest number of character codes
     If the lengths of the character strings are different, determines the longest character string
   - Smaller string: If the character strings are different, determines the string with the smallest number of character codes
     If the lengths of the character strings are different, determines the shortest character string

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Block data comparisons	BKCMP=	BKCMP= S1 S2 D n	• Compares n points of data from S1 with n points of data from S2 in 1-word units, and stores the results of the comparison at n points from the bit device designated by (D).		5	
	BKCMP<>	BKCMP<> S1 S2 D n				
	BKCMP>	BKCMP> S1 S2 D n				
	BKCMP<=	BKCMP<= S1 S2 D n				
	BKCMP<	BKCMP< S1 S2 D n				
	BKCMP>=	BKCMP>= S1 S2 D n				
	BKCMP=P	BKCMP=P S1 S2 D n				
	BKCMP<>P	BKCMP<>P S1 S2 D n				
	BKCMP>P	BKCMP>P S1 S2 D n				
	BKCMP<=P	BKCMP<=P S1 S2 D n				
	BKCMP<P	BKCMP<P S1 S2 D n				
	BKCMP>=P	BKCMP>=P S1 S2 D n				

(2) Arithmetic operation instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
BIN 16-bit addition and subtraction operations	+	[+ | S | D]	• (D)+(S)→(D)		3	●
	+P	[+P | S | D]				
	+	[+ | S1 | S2 | D]	• (S1)+(S2)→(D)		4	●
	+P	[+P | S1 | S2 | D]				
	−	[− | S | D]	• (D)−(S)→(D)		3	●
	−P	[−P | S | D]				
	−	[− | S1 | S2 | D]	• (S1)−(S2)→(D)		4	●
	−P	[−P | S1 | S2 | D]				
BIN 32-bit addition and subtraction operations	D+	[D+ | S | D]	• (D+1,D)+(S+1,S)→(D+1,D)		*1	●
	D+P	[D+P | S | D]				
	D+	[D+ | S1 | S2 | D]	• (S1+1,S1)+(S2+1,S2)→(D+1,D)		*2	●
	D+P	[D+P | S1 | S2 | D]				
	D−	[D− | S | D]	• (D+1,D)−(S+1,S)→(D+1,D)		*1	●
	D−P	[D−P | S | D]				
	D−	[D− | S1 | S2 | D]	• (S1+1,S1)−(S2+1,S2)→(D+1,D)		*2	●
	D−P	[D−P | S1 | S2 | D]				
BIN 16-bit multiplication and division operations	∗	[∗ | S1 | S2 | D]	• (S1) ∗ (S2)→(D+1,D)		*3	●
	∗P	[∗P | S1 | S2 | D]				
	/	[/ | S1 | S2 | D]	• (S1)/(S2)→Quotient (D), Remainder (D+1)		4	●
	/P	[/P | S1 | S2 | D]				
BIN 32-bit multiplication and division operations	D∗	[D∗ | S1 | S2 | D]	• (S1+1,S1) ∗ (S2+1,S2) →(D+3,D+2,D+1,D)		4	●
	D∗P	[D∗P | S1 | S2 | D]				
	D/	[D/ | S1 | S2 | D]	• (S1+1,S1)/(S2+1,S2) →Quotient (D+1,D), Remainder (D+3,D+2)		4	●
	D/P	[D/P | S1 | S2 | D]				

REMARK

1) *1: The number of steps may vary depending on the device and type of CPU module being used.

<table>
<tr><th rowspan="2" colspan="2">Device</th><th colspan="2">Number of Steps</th></tr>
<tr><th>QCPU</th><th>QnACPU</th></tr>
<tr><td>• Word device:
• Bit device:

• Constant :</td><td>Internal device (except for file register ZR)
Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no index modification.
No limitations</td><td>5</td><td rowspan="2">3</td></tr>
<tr><td colspan="2">Devices other than above</td><td>3</td></tr>
</table>

2) *2: The number of steps may vary depending on the device and type of CPU module being used.

<table>
<tr><th rowspan="2" colspan="2">Device</th><th colspan="2">Number of Steps</th></tr>
<tr><th>QCPU</th><th>QnACPU</th></tr>
<tr><td>• Word device:
• Bit device:

• Constant :</td><td>Internal device (except for file register ZR)
Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no index modification.
No limitations</td><td>6</td><td rowspan="2">4</td></tr>
<tr><td colspan="2">Devices other than above</td><td>4</td></tr>
</table>

3) *3: The number of steps may vary depending on the device and type of CPU module being used.

<table>
<tr><th rowspan="2" colspan="2">Device</th><th colspan="2">Number of Steps</th></tr>
<tr><th>QCPU</th><th>QnACPU</th></tr>
<tr><td>• Word device:
• Bit device:

• Constant :</td><td>Internal device (except for file register ZR)
Devices whose device Nos. are multiples of 16, whose digit designation is K4, and which use no index modification.
No limitations</td><td>3</td><td rowspan="2">4</td></tr>
<tr><td colspan="2">Devices other than above</td><td>4</td></tr>
</table>

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
BCD 4-digit addition and subtraction operations	B+	B+ S D	• (D)+(S)→(D)		3	●
	B+P	B+P S D				
	B+	B+ S1 S2 D	• (S1)+(S2)→(D)		4	
	B+P	B+P S1 S2 D				
	B−	B− S D	• (D)−(S)→(D)		3	●
	B−P	B−P S D				
	B−	B− S1 S2 D	• (S1)−(S2)→(D)		4	
	B−P	B−P S1 S2 D				
BCD 8-digit addition and subtraction operations	DB+	DB+ S D	• (D+1,D)+(S+1,S)→(D+1,D)		3	
	DB+P	DB+P S D				
	DB+	DB+ S1 S2 D	• (S1+1,S1)+(S2+1,S2)→(D+1,D)		4	
	DB+P	DB+P S1 S2 D				
	DB−	DB− S D	• (D+1,D)−(S+1,S)→(D+1,D)		3	
	DB−P	DB−P S D				
	DB−	DB− S1 S2 D	• (S1+1,S1)−(S2+1,S2)→(D+1,D)		4	
	DB−P	DB−P S1 S2 D				
BCD 4-digit multiplication and division operations	B∗	B∗ S1 S2 D	• (S1) ∗ (S2)→(D+1,D)		4	●
	B∗P	B∗P S1 S2 D				
	B/	B/ S1 S2 D	• (S1)/(S2)→Quotient (D), Remainder (D+1)		4	●
	B/P	B/P S1 S2 D				
BCD 8-digit multiplication and division operations	DB∗	DB∗ S1 S2 D	• (S1+1,S1) ∗ (S2+1,S2) →(D+3,D+2,D+1,D)		4	
	DB∗P	DB∗P S1 S2 D				
	DB/	DB/ S1 S2 D	• (S1+1,S1)/(S2+1,S2) →Quotient (D+1,D), Remainder (D+3,D+2)		4	●
	DB/P	DB/P S1 S2 D				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Floating decimal point data addition and subtrac-tion operations	E+	E+ S D	• (D+1,D)+(S+1,S)→(D+1,D)		3	
	E+P	E+P S D				
	E+	E+ S1 S2 D	• (S1+1,S1)+(S2+1,S2)→(D+1,D)		4	
	E+P	E+P S1 S2 D				
	E−	E− S D	• (D+1,D)−(S+1,S)→(D+1,D)		3	
	E−P	E−P S D				
	E−	E− S1 S2 D	• (S1+1,S1)−(S2+1,S2)→(D+1,D)		4	
	E−P	E−P S1 S2 D				
Floating decimal point data Multiplica-tion and division operations	E*	E* S1 S2 D	• (S1+1,S1) * (S2+1,S2)→(D+1,D)		3	
	E*P	E*P S1 S2 D				
	E/	E/ S1 S2 D	• (S1+1,S1)/(S2+1,S2)→Quotient (D+1,D)		4	
	E/P	E/P S1 S2 D				
BIN block addition and subtrac-tion operations	$+	$+ S D	• Links character string designated with (S) to character string designated with (D), and stores the result from (D) onward.		3	
	$+P	$+P S D				
	$+	$+ S1 S2 D	• Links character string designated with (S2) to character string designated with (S1), and stores the result from (D) onward.		4	
	$+P	$+P S1 S2 D				
Character string data combinations	BK+	BK+ S1 S2 D n	• Adds data of n points from (S1) and data of n points from (S2) in batch.		5	
	BK+P	BK+P S1 S2 D n				
	BK−	BK− S1 S2 D n	• Subtracts data of n points from (S1) and data of n points from (S2) in batch.		5	
	BK−P	BK−P S1 S2 D n				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
BIN data increment	INC	INC D	• (D)+1→(D)		2	●
	INCP	INCP D				
	DINC	DINC D	• (D+1,D)+1→(D+1,D)		*1	●
	DINCP	DINCP D				
	DEC	DEC D	• (D)−1→(D)		2	●
	DECP	DECP D				
	DDEC	DDEC D	• (D+1,D)−1→(D+1,D)		*1	●
	DDECP	DDECP D				

REMARK

*1: The number of steps may vary depending on the device and type of CPU module being used.

Device	Number of Steps QCPU	Number of Steps QnACPU
• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no index modification. • Constant : No limitations	3	2
Devices other than above	2	

(3) Data conversion instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
BCD conversion	BCD	BCD S D	· (S) $\xrightarrow{\text{BCD conversion}}$ (D) BIN(0 to 9999)		3	●
	BCDP	BCDP S D				
	DBCD	DBCD S D	· (S+1, S) $\xrightarrow{\text{BCD conversion}}$ (D+1, D) BIN(0 to 99999999)		3	●
	DBCDP	DBCDP S D				
BIN conversion	BIN	BIN S D	· (S) $\xrightarrow{\text{BIN conversion}}$ (D) BCD(0 to 9999)		3	●
	BINP	BINP S D				
	DBIN	DBIN S D	· (S+1, S) $\xrightarrow{\text{BIN conversion}}$ (D+1, D) BCD(0 to 99999999)		3	●
	DBINP	DBINP S D				
Conversion from BIN to floating decimal point	FLT	FLT S D	· (S+1, S) $\xrightarrow{\text{Conversion to floating point}}$ (D) BIN(-32768 to 32767)		3	
	FLTP	FLTP S D				
	DFLT	DFLT S D	· (S+1, S) $\xrightarrow{\text{Conversion to floating point}}$ (D+1, D) Real number(-2147483648 to 2147483647)		3	
	DFLTP	DFLTP S D				
Conversion from floating decimal point to BIN	INT	INT S D	· (S+1, S) $\xrightarrow{\text{BIN conversion}}$ (D) Real number(-32768 to 32767)		3	
	INTP	INTP S D				
	DINT	DINT S D	· (S+1, S) $\xrightarrow{\text{BIN conversion}}$ (D+1, D) Real number(-2147483648 to 2147483647)		3	
	DINTP	DINTP S D				
Conversion between BIN 16-bit and 32-bit	DBL	DBL S D	· (S) $\xrightarrow{\text{Conversion}}$ (D+, 1) BIN(-32768 to 32767)		3	
	DBLP	DBLP S D				
	WORD	WORD S D	· (S+1, S) $\xrightarrow{\text{Conversion}}$ (D) BIN(-32768 to 32767)		3	
	WORDP	WORDP S D				
Conversion from BIN to gray code	GRY	GRY S D	· (S) $\xrightarrow{\text{Conversion to gray code}}$ (D) BIN(-32768 to 32767)		3	
	GRYP	GRYP S D				
	DGRY	DGRY S D	· (S+1, S) $\xrightarrow{\text{Conversion to gray code}}$ (D+1, D) Real number(-2147483648 to 2147483647)		3	
	DGRYP	DGRYP S D				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Conversion from gray code to BIN	GBIN	GBIN S D	· (S) → (D): Conversion to gray code. (S): Gray code(-32768 to 32767)		3	
	GBINP	GBINP S D				
	DGBIN	DGBIN S D	· (S+1, S) → (D+1, D): Conversion to gray code. (S+1, S): Gray code(-2147483648 to 2147483647)		3	
	DGBINP	DGBINP S D				
Complement to 2	NEG	NEG D	· $\overline{(D)}$ → (D). (D): BIN data		2	
	NEGP	NEGP D				
	DNEG	DNEG D	· $\overline{(D+1, D)}$ → (D+1, D). (D+1, D): BIN data		2	
	DNEGP	DNEGP D				
	ENEG	ENEG D	· $\overline{(D+1, D)}$ → (D+1, D). (D+1, D): Real number data		2	
	ENEGP	ENEGP D				
Block conversions	BKBCD	BKBCD S D n	• Batch converts BIN data n points from (S) to BCD data and stores the result from (D) onward.		4	
	BKBCDP	BKBCDP S D n				
	BKBIN	BKBIN S D n	• Batch converts BCD data n points from (S) to BIN data and stores the result from (D) onward.		4	
	BKBINP	BKBINP S D n				

(4) Data conversion instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
16-bit data transfer	MOV	MOV S D	· (S) → (D)		∗1	●
	MOVP	MOVP S D				
32-bit data transfer	DMOV	DMOV S D	· (S+1,S) → (D+1,D)		3	●
	DMOVP	DMOVP S D				
Floating decimal point data transfer	EMOV	EMOV S D	· (S+1, S) → (D+1, D) (S+1, S): Real number data		3	● ∗2
	EMOVP	EMOVP S D				
Character string data transfer	$MOV	$MOV S D	• Transfers character string designated by (S) to device designated by (D) onward.		3	
	$MOVP	$MOVP S D				
16-bit data negation transfer	CML	CML S D	· $\overline{(S)}$ → (D)		∗1	●
	CMLP	CMLP S D				
32-bit data negation transfer	DCML	DCML S D	· $\overline{(S+1,S)}$ → (D+1,D)		3	●
	DCMLP	DCMLP S D				
Block transfer	BMOV	BMOV S D n	(S) → (D), n		4	●
	BMOVP	BMOVP S D n				
Multiple transfers of same block	FMOV	FMOV S D n	(S) → (D), n		4	●
	FMOVP	FMOVP S D n				
16-bit data exchange	XCH	XCH S D	· (S) ↔ (D)		3	●
	XCHP	XCHP S D				
32-bit data exchange	DXCH	DXCH S D	· (S+1,S) ↔ (D+1,D)		3	●
	DXCHP	DXCHP S D				
Block data exchange	BXCH	BXCH S D n	(S) ↔ (D), n		4	
	BXCHP	BXCHP S D n				
Exchange of upper and lower bytes	SWAP	SWAP D	(S) b15 to b8 b7 to b0: 8 bits, 8 bits (D) b15 to b8 b7 to b0: 8 bits, 8 bits		3	
	SWAPP	SWAPP D				

REMARK

*1: The number of steps may vary depending on the device and type of CPU module being used.

<table>
<tr><th rowspan="2">Device</th><th colspan="2">Number of Steps</th></tr>
<tr><th>QCPU</th><th>QnACPU</th></tr>
<tr><td>• Word device: Internal device (except for file register ZR)
• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K4, and which use no index modification.
• Constant : No limitations</td><td>2</td><td rowspan="2">3</td></tr>
<tr><td>Devices other than above</td><td>3</td></tr>
</table>

*2: The subset is effective only with QCPU.

(5) Program branch instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Jump	CJ	CJ Pn	• Jumps to Pn when input conditions are met		2	●
	SCJ	SCJ Pn	• Jumps to Pn from the scan after the meeting of input condition		2	●
	JMP	JMP Pn	• Jumps unconditionally to Pn		2	●
	GOEND	GOEND	• Jumps to END instruction when input condition is met		1	

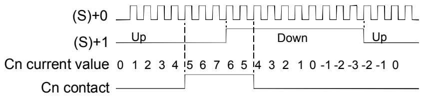
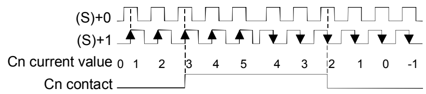
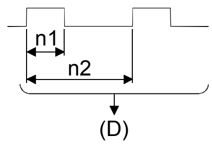
(6) Program execution control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Disable interrupts	DI	DI	• Prohibits the running of an interrupt program		1	
Enable interrupts	EI	EI	• Resets interrupt program execution prohibition		1	
Interrupt disable/ enable setting	IMASK	IMASK S	• Prohibits or permits interrupts for each interrupt program		2	
Return	IRET	IRET	• Returns to sequence program following an interrupt program		1	

(7) I/O refresh instructions

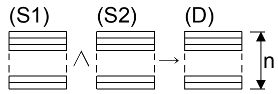
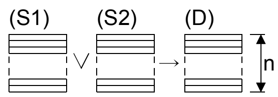
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
I/O refresh	RFS	RFS D n	• Refreshes the relevant I/O area during scan		3	

(8) Other convenient instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Up/down counter	UDCNT1	UDCNT1 S D n			4	
	UDCNT2	UDCNT2 S D n			4	
Teaching timer	TTMR	TTMR D n	· (ON time of TTMR)∗ n ⟶ (D) n=0:1, n=1:10, n=2:100		3	
Special timer	STMR	STMR S n D	• The 4 points from the bit device designated by (D) operate as shown below, depending. on the ON/OFF status of the input conditions for the STMR instruction: (D)+0: Off delay timer output (D)+1: One shot after off timer output (D)+2: One shot after on timer output (D)+3: On delay timer output		3	
Nearest path control	ROTC	ROTC S n1 n2 D	• Rotates a rotary table with n1 divisions from the stop position to the position designated by (S+1) by the nearest path.		5	
Ramp signal	RAMP	RAMP n1 n2 D1 n3 D2	• Changes device data designated by D1 from n1 to n2 in n3 scans.		6	
Pulse density	SPD	SPD S n D	• Counts the pulse input from the device designated by (S) for the duration of time designated by n, and stores the count in the device designated by (D).		4	
Pulse output	PLSY	PLSY n1 n2 D	· (n1)Hz ⟶ (D) Output n2 times		4	
Pulse width modulati on	PWM	PWM n1 n2 D			4	
Matrix input	MTR	MTR S D1 D2 n	• Store 16 times of n lows in the device specified by (S) to the device specified by (D2) in sequence.		5	

Appendix 2.3 Application instructions

(1) Logical operation instruction

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Logical product	WAND	WAND S D	· (D)∧(S)→(D)		3	●
	WANDP	WANDP S D				
	WAND	WAND S1 S2 D	· (S1)∧(S2)→(D)		4	● *2
	WANDP	WANDP S1 S2 D				
	DAND	DAND S D	· (D+1,D)∧(S+1,S)→(D+1,D)		*1	●
	DANDP	DANDP S D				
	DAND	DAND S1 S2 D	· (S1+1,S1)∧(S2+1,S2)→(D+1,D)		*1	● *2
	DANDP	DANDP S1 S2 D				
	BKAND	BKAND S1 S2 D n			5	
	BKANDP	BKANDP S1 S2 D n				
Logical sum	WOR	WOR S D	· (D)∨(S)→(D)		3	●
	WORP	WORP S D				
	WOR	WOR S1 S2 D	· (S1)∨(S2)→(D)		4	● *2
	WORP	WORP S1 S2 D				
	DOR	DOR S D	· (D+1,D)∨(S+1,S)→(D+1,D)		*1	●
	DORP	DORP S D				
	DOR	DOR S1 S2 D	· (S1+1,S1)∨(S2+1,S2)→(D+1,D)		*1	● *2
	DORP	DORP S1 S2 D				
	BKOR	BKOR S1 S2 D n			5	
	BKORP	BKORP S1 S2 D n				

REMARK

*1: The number of steps may vary depending on the device and type of CPU module being used.

Device	Number of Steps QCPU	Number of Steps QnACPU
• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no index modification. • Constant : No limitations	6	4
Devices other than above	4	

*2: The subset is effective only with QCPU.

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Exclusive OR	WXOR	WXOR S D	· $(D)\veebar(S)\rightarrow(D)$		3	●
	WXORP	WXORP S D				
	WXOR	WXOR S1 S2 D	· $(S1)\veebar(S2)\rightarrow(D)$		4	● *2
	WXORP	WXORP S1 S2 D				
	DXOR	DXOR S D	· $(D+1,D)\veebar(S+1,S)\rightarrow(D+1,D)$		*1	●
	DXORP	DXORP S D				
	DXOR	DXOR S1 S2 D	· $(S1+1,S1)\veebar(S2+1,S2)\rightarrow(D+1,D)$		*1	● *2
	DXORP	DXORP S1 S2 D				
	BKXOR	BKXOR S1 S2 D n	$(S1)\veebar(S2)\rightarrow(D)$ n		5	
	BKXORP	BKXORP S1 S2 D n				
NON exclusive logical sum	WXNR	WXNR S D	· $\overline{(D)\veebar(S)}\rightarrow(D)$		3	●
	WXNRP	WXNRP S D				
	WXNR	WXNR S1 S2 D	· $\overline{(S1)\veebar(S2)}\rightarrow(D)$		4	● *2
	WXNRP	WXNRP S1 S2 D				
	DXNR	DXNR S D	· $\overline{(D+1,D)\veebar(S+1,S)}\rightarrow(D+1,D)$		*1	●
	DXNRP	DXNRP S D				
	DXNR	DXNR S1 S2 D	· $\overline{(S1+1,S1)\veebar(S2+1,S2)}\rightarrow(D+1,D)$		*1	● *2
	DXNRP	DXNRP S1 S2 D				
	BKXNOR	BKXNOR S1 S2 D n	$\overline{(S1)\veebar(S2)}\rightarrow(D)$ n		5	
	BKXNORP	BKXNORP S1 S2 D n				

REMARK

*1: The number of steps may vary depending on the device and type of CPU module being used.

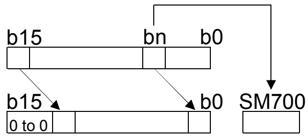
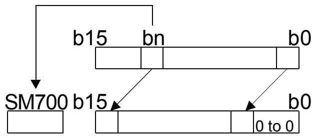
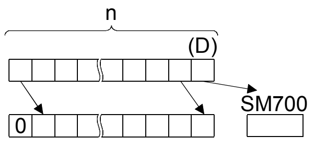
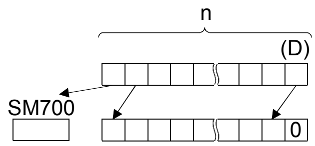
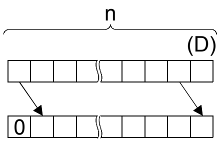
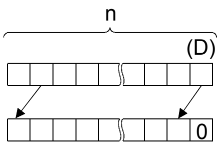
Device	Number of Steps	
	QCPU	QnACPU
• Word device: Internal device (except for file register ZR) • Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no index modification. • Constant : No limitations	6	4
Devices other than above	4	

*2: The subset is effective only with QCPU.

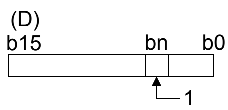
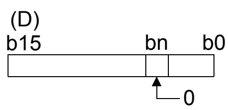
(2) Rotation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Right rotation	ROR	ROR D n	b15 (D) b0 SM700 Rotates n bits to the right		3	●
	RORP	RORP D n				
	RCR	RCR D n	b15 (D) b0 SM700 Rotates n bits to the right		3	●
	RCRP	RCRP D n				
Left rotation	ROL	ROL D n	SM700 b15 (D) b0 Rotates n bits to the left		3	●
	ROLP	ROLP D n				
	RCL	RCL D n	SM700 b15 (D) b0 Rotates n bits to the left		3	●
	RCLP	RCLP D n				
Right rotation	DROR	DROR D n	(D+1) (D) b31 to b16 b15 to b0 SM700 Rotates n bits to the right		3	●
	DRORP	DRORP D n				
	DRCR	DRCR D n	(D+1) (D) b31 to b16 b15 to b0 SM700 Rotates n bits to the right		3	●
	DRCRP	DRCRP D n				
Left rotation	DROL	DROL D n	(D+1) (D) SM700 b31 to b16 b15 to b0 Rotates n bits to the left		3	●
	DROLP	DROLP D n				
	DRCL	DRCL D n	(D+1) (D) SM700 b31 to b16 b15 to b0 Rotates n bits to the left		3	●
	DRCLP	DRCLP D n				

(3) Shift instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
n-bit shift	SFR	SFR D n			3	●
	SFRP	SFRP D n				
	SFL	SFL D n			3	●
	SFLP	SFLP D n				
1-bit shift	BSFR	BSFR D n			3	
	BSFRP	BSFRP D n				
	BSFL	BSFL D n			3	
	BSFLP	BSFLP D n				
1-word shift	DSFR	DSFR D n			3	●
	DSFRP	DSFRP D n				
	DSFL	DSFL D n			3	●
	DSFLP	DSFLP D n				

(4) Bit processing instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Bit set/ reset	BSET	BSET D n			3	●
	BSETP	BSETP D n				
	BRST	BRST D n			3	●
	BRSTP	BRSTP D n				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Bit tests	TEST	TEST S1 S2 D	(S1) b15 to b0 (D) Bit designated in (S2)		4	
	TESTP	TESTP S1 S2 D				
	DTEST	DTEST S1 S2 D	(S1) b31 to b0 (D) Bit designated in (S2)		4	
	DTESTP	DTESTP S1 S2 D				
Batch reset of bit devices	BKRST	BKRST S n	(S) ON OFF ON ON Reset (S) OFF OFF OFF OFF n		3	
	BKRSTP	BKRSTP S n				

(5) Data processing instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Data searches	SER	SER S1 S2 D n	(S1) (S2) n (D) : Match No. (D+1): No. of matches		5	
	SERP	SERP S1 S2 D n				
	DSER	DSER S1 S2 D n	32 bits (S1) (S2) n (D) : Match No. (D+1): No. of matches		5	
	DSERP	DSERP S1 S2 D n				
Bit checks	SUM	SUM S D	(S) b15 b0 (D): No. of 1s		3	●
	SUMP	SUMP S D				
	DSUM	DSUM S D	(S+1) (S) (D):No. of 1s		3	●
	DSUMP	DSUMP S D				
Decode	DECO	DECO S D n	Decode of 8 to 256 (S) n Decode (D) 2^n bits		4	
	DECOP	DECOP S D n				
Encode	ENCO	ENCO S D n	Decode of 256 to 8 (S) 2^n bits Encode (D) n		4	
	ENCOP	ENCOP S D n				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
7-segment decode	SEG	SEG S D	(S) b3 to b0 → 7SEG → (D)		3	●
	SEGP	SEGP S D				
Separating and linking	DIS	DIS S D n	• Separates 16-bit data designated by (S) into 4-bit units, and stores at the lower 4 bits of n points from (D). (n) ≤ (4)		4	
	DISP	DISP S D n				
	UNI	UNI S D n	• Links the lower 4 bits of n points from the device designated by (S) and stores at the device designated by (D). (n) ≤ (4)		4	
	UNIP	UNIP S D n				
	NDIS	NDIS S1 D S2	• Separates the data at the devices below that designated by (S1) into bits designated below (S2) and stores in sequence from the device designated by (D).		4	
	NDISP	NDISP S1 D S2				
	NUNI	NUNI S1 D S2	• Links the data at the devices below that designated by (S1) in the bits designated below (S2) and stores in sequence from the device designated by (D).			
	NUNIP	NUNIP S1 D S2				
	WTOB	WTOB S D n	• Breaks n-points of 16-bit data from the device designated by (S) into 8-bit units, and stores in sequence at the device designated by (D).		4	
	WTOBP	WTOBP S D n				
	BTOW	BTOW S D n	• Links the lower 8 bits of 16-bit data of n-points from the device designated by (S) into 16-bit units, and stores in sequence at the device designated by (D).			
	BTOWP	BTOWP S D n				
Search	MAX	MAX S D n	• Searches the data of n-points from the device designated by (S) in 16-bit units, and stores the maximum value at the device designated by (D).		4	
	MAXP	MAXP S D n				
	MIN	MIN S D n	• Searches the data of n-points from the device designated by (S) in 16-bit units, and stores the minimum value at the device designated by (D).			
	MINP	MINP S D n				
	DMAX	DMAX S D n	• Searches the data of 2∗n-points from the device designated by (S) in 32-bit units, and stores the maximum value at the device designated by (D).		4	
	DMAXP	DMAXP S D n				
	DMIN	DMIN S D n	• Searches the data of 2∗n-points from the device designated by (S) in 32-bit units, and stores the minimum value at the device designated by (D).			
	DMINP	DMINP S D n				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Sort	SORT	SORT S1 n S2 D1 D2 · S2: Number of comparisons made during one run · D1: Device to turn ON when sort is completed · D2: For system use	• Sorts data of n-points from device designated by (S1) in 16-bit units. (n×(n-1)/2 scans required)		6	
	DSORT	DSORT S1 n S2 D1 D2 · S2: Number of comparisons made during one run · D1: Device to turn ON when sort is completed · D2: For system use	• Sorts data of 2×n-points from device designated by (S1) in 32-bit units. (n×(n-1)/2 scans required)			
Total value calculations	WSUM	WSUM S D n	• Adds 16 bit BIN data of n-points from the device designated by (S), and stores it in the device designated by (D).		4	
	WSUMP	WSUMP S D n				
	DWSUM	DWSUM S D n	• Adds 32 bit BIN data of n-points from the device designated by (S), and stores it in the device designated by (D).			
	DWSUMP	DWSUMP S D n				

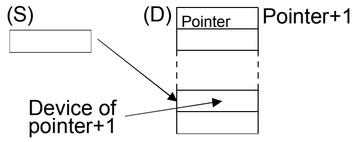
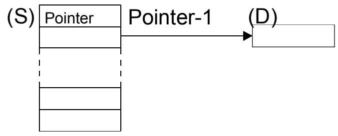
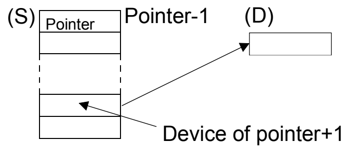
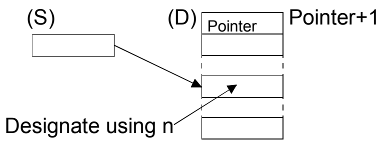
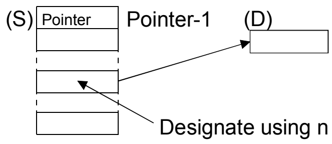
(6) Structure creation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Number of repeats	FOR	FOR | n	• Executes n times between [FOR] and [NEXT].		2	
	NEXT	NEXT			1	
	BREAK	BREAK | D | Pn	• Forcibly ends the execution of the [FOR] to [NEXT] cycle and jumps pointer to Pn.		3	
	BREAKP	BREAKP | D | Pn				
Sub routine program calls	CALL	CALL | Pn | S1 to Sn	• Executes sub-routine program Pn when input condition is met. (S1 to Sn are arguments sent to subroutine program. 0 ≤ (n) ≤ (5)		*1 2 + n	
	CALLP	CALLP | Pn | S1 to Sn				
	RET	RET	• Returns from sub-routine program		1	
	FCALL	FCALL | Pn | S1 to Sn	• Performs non-execution processing of sub-routine program Pn if input conditions have not been met.		*1 2 + n	
	FCALLP	FCALLP | Pn | S1 to Sn				
	ECALL	ECALL | ✻ | Pn | S1 to Sn ✻: Program name	• Executes sub-routine program Pn from within designated program name when input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. 0 ≤ (n) ≤ (5)		*2 3 + n	
	ECALLP	ECALLP | ✻ | Pn | S1 to Sn ✻: Program name				
	EFCALL	EFCALL | ✻ | Pn | S1 to Sn ✻: Program name	• Performs non-execution processing of sub-routine program Pn from within designated program name if input conditions have not been met.		*2 3 + n	
	EFCALLP	EFCALLP | ✻ | Pn | S1 to Sn ✻: Program name				
	COM	COM	• Performs link refresh and general data processing.		1	
Fixed index modification	IX	IX | S Device modification ladder	• Conducts index modification for individual devices used in device modification ladder.		2	
	IXEND	IXEND			1	
	IXDEV	IXDEV	• Stores modification value used for index modification performed between [IX] and [IXEND] in the device below that designated by (D).		1	
	IXSET	IXSET | Pn | D Modification value designation			3	

*1: n indicates number of arguments for sub-routine program.

*2: n indicates the total of the number of arguments used in the sub-routine program and the number of program name steps.
The number of program name steps is calculated as "number of characters in the program / 2". (decimal fraction is rounded up).

(7) Data table operation instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Data table processing	FIFW	FIFW S D			3	
	FIFWP	FIFWP S D				
	FIFR	FIFR S D			3	
	FIFRP	FIFRP S D				
	FPOP	FPOP S D			3	
	FPOPP	FPOPP S D				
	FINS	FINS S D n			4	
	FINSP	FINS S D n				
	FDEL	FDEL S D n			4	
	FDELP	FDELP S D n				

(8) Buffer memory access instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Data read	FROM	FROM n1 n2 D n3	• Reads data in 16-bit units from special function module.		5	
	FROMP	FROMP n1 n2 D n3				
	DFRO	DFRO n1 n2 D n3	• Reads data in 32-bit units from special function module.		5	
	DFROP	DFROP n1 n2 D n3				
Data write	TO	TO n1 n2 S n3	• Writes data in 16-bit units to special function module.		5	
	TOP	TOP n1 n2 S n3				
	DTO	DTO n1 n2 S n3	• Writes data in 32-bit units to special function module.		5	
	DTOP	DTOP n1 n2 S n3				

(9) Display instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
ASCII print	PR	* With SM701 off PR S D	• Outputs ASCII code of 8 points (16 characters) from device designated by (S) to output module.		3	
	PR	* With SM701 on PR S D	• Outputs ASCII code from device designated by (S) to 00H to output module.			
	PRC	PRC S D	• Converts comments from device designated by (S) to ASCII code and outputs to output module.			
Display	LED	LED S	• Displays ASCII code of 8 points (16 characters) from the device designated by (S) at the LED display device on the front of the CPU module.		2	
	LEDC	LEDC S	• Displays the comments from the device designated by (S) at the LED display device on the front of the CPU module.			
Reset	LEDR	LEDR	• Resets annunciator and display unit display.		1	

(10) Debugging and failure diagnosis instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Check	CHKST	CHKST	• CHK instruction is executed when CHKST is executable. • Jumps to the step following the CHK instruction when CHKST is in a non-executable status.		1	
	CHK	CHK Check condition	• During normal conditions→SM80: OFF, SD80: 0 • During abnormal conditions→SM80: ON, SD80: Failure No.			
	CHKCIR	CHKCIR	• Starts update in ladder pattern being checked by CHK instruction.		1	
	CHKEND	CHKEND	• Ends update in ladder pattern being checked by CHK instruction.			
Status latch	SLT	SLT	• Executes status latch.		1	
	SLTR	SLTR	• Resets status latch to enable re-execution.			
Sampling trace	STRA	STRA	• Applies trigger to sampling trace.		1	
	STRAR	STRAR	• Resets sampling trace to enable re-execution.			
Program trace	PTRA	PTRA	• Applies trigger to program trace.		1	
	PTRAR	PTRAR	• Resets program trace to enable re-execution.			
	PTRAEXE	PTRAEXE	• Executes program trace.		1	
	PTRAEXEP	PTRAEXEP				

(11) Character string processing instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
BIN to decimal ASCII	BINDA	BINDA S D	• Converts 1-word BIN value designated by (S) to a 5-digit, decimal ASCII value, and stores it at the word device designated by (D).		3	
	BINDAP	BINDAP S D				
	DBINDA	DBINDA S D	• Converts 2-word BIN value designated by (S) to a 10-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	
	DBINDAP	DBINDAP S D				
BIN to hexadecimal ASCII	BINHA	BINHA S D	• Converts 1-word BIN value designated by (S) to a 4-digit, hexadecimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	
	BINHAP	BINHAP S D				
	DBINHA	DBINHA S D	• Converts 2-word BIN value designated by (S) to a 8-digit, hexadecimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	
	DBINHAP	DBINHAP S D				
BCD to decimal ASCII	BCDDA	BCDDA S D	• Converts 1-word BCD value designated by (S) to a 4-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	
	BCDDAP	BCDDAP S D				
	DBCDDA	DBCDDA S D	• Converts 2-word BCD value designated by (S) to an 8-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	
	DBCDDAP	DBCDDAP S D				
Decimal ASCII to BIN	DABIN	DABIN S D	• Converts a 5-digit, decimal ASCII value designated by (S) to 1-word BIN value, and stores it at a word device number designated by (D).		3	
	DABINP	DABINP S D				
	DDABIN	DDABIN S D	• Converts a 10-digit, decimal ASCII value designated by (S) to 2-word BIN value, and stores it at a word device number designated by (D).		3	
	DDABINP	DDABINP S D				
Hexadecimal ASCII to BIN	HABIN	HABIN S D	• Converts a 4-digit, hexadecimal ASCII value designated by (S) to 1-word BIN value, and stores it at a word device number designated by (D).		3	
	HABINP	HABINP S D				
	DHABIN	DHABIN S D	• Converts an 8-digit, hexadecimal ASCII value designated by (S) to 2-word BIN value, and stores it at a word device number designated by (D).		3	
	DHABINP	DHABINP S D				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Decimal ASCII to BCD	DABCD	DABCD S D	• Converts a 4-digit, decimal ASCII value designated by (S) to 1-word BCD value, and stores it at a word device number designated by (D).		3	
	DABCDP	DABCDP S D				
	DDABCD	DDABCD S D	• Converts an 8-digit, decimal ASCII value designated by (S) to 2-word BCD value, and stores it at a word device number designated by (D).		3	
	DDABCDP	DDABCDP S D				
Device comment read operation	COMRD	COMRD S D	• Stores comment from device designated by (S) at a device designated by (D).		3	
	COMRDP	COMRDP S D				
Character string length detection	LEN	LEN S D	• Stores data length (number of characters) in character string designated by (S) at a device designated by (D).		3	
	LENP	LENP S D				
BIN to decimal character string	STR	STR S1 S2 D	• Converts a 1-word BIN value designated by (S2) to a decimal character string with the total number of digits and the number of decimal fraction digits designated by (S1) and stores them at a device designated by (D).		4	
	STRP	STRP S1 S2 D				
	DSTR	DSTR S1 S2 D	• Converts a 2-word BIN value designated by (S2) to a decimal character string with the total number of digits and the number of decimal fraction digits designated by (S1) and stores them at a device designated by (D).		4	
	DSTRP	DSTRP S1 S2 D				
Decimal character string to BIN	VAL	VAL S D1 D2	• Converts a character string including decimal point designated by (S) to a 1-word BIN value and the number of decimal fraction digits, and stores them into devices designated by (D1) and (D2).		4	
	VALP	VALP S D1 D2				
	DVAL	DVAL S D1 D2	• Converts a character string including decimal point designated by (S) to a 2-word BIN value and the number of decimal fraction digits, and stores them into devices designated by (D1) and (D2).		4	
	DVALP	DVALP S D1 D2				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Floating decimal point to character string	ESTR	ESTR S1 S2 D	• Converts floating decimal point data designated by (S1) to character string, and stores them in a device designated by (D).		4	
	ESTRP	ESTRP S1 S2 D				
Character string to floating decimal point	EVAL	EVAL S D	• Converts character string designated by (S1) to floating decimal point data, and stores them in a device designated by (D).		3	
	EVALP	EVALP S D				
Hexadecimal BIN to ASCII	ASC	ASC S D n	• Converts 1-word BIN values of the device number and later designated by (S) to ASCII, and stores only n characters of them at the device number designated by (D).		4	
	ASCP	ASCP S D n				
ASCII to Hexadecimal BIN	HEX	HEX S D n	• Converts only n ASCII characters of the device number and later designated by (S) to BIN values, and stores them at the device number designated by (D).		4	
	HEXP	HEXP S D n				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Character string processing	RIGHT	RIGHT S D n	• Stores n characters from the end of a character string designated by (S) at the device designated by (D).		4	
	RIGHTP	RIGHTP S D n				
	LEFT	LEFT S D n	• Stores n characters from the beginning of a character string designated by (S) at the device designated by (D).			
	LEFTP	LEFTP S D n				
	MIDR	MIDR S1 D S2	• Stores the designated number of characters in the character string designated by (S1) from the position designated by (S2) at the device designated by (D).		4	
	MIDRP	MIDRP S1 D S2				
	MIDW	MIDW S1 D S2	• Stores the designated number of characters in the character string designated by (S1) from the position designated by (S2) at the device designated by (D).			
	MIDWP	MIDWP S1 D S2				
	INSTR	INSTR S1 S2 D n	• Searches character string (S1) from the nth character of character string (S2), and stores matched positions at (D).		5	
	INSTRP	INSTRP S1 S2 D n				
Floating decimal point to BCD	EMOD	EMOD S1 S2 D	• Converts floating decimal point data (S1) to BCD data with number of decimal fraction digits designated by (S2), and stores at device designated by (D).		4	
	EMODP	EMODP S1 S2 D				
BCD to floating decimal point	EREXP	EREXP S1 S2 D	• Converts BCD data (S1) to floating decimal point data with the number of decimal fraction digits designated by (S2), and stores at device designated by (D).		4	
	EREXPP	EREXPP S1 S2 D				

(12) Special function instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Trigono-metric functions (Floating decimal point data)	SIN	SIN S D	· Sin(S+1,S) → (D+1,D)		3	
	SINP	SINP S D				
	COS	COS S D	· Cos(S+1,S) → (D+1,D)		3	
	COSP	COSP S D				
	TAN	TAN S D	· Tan(S+1,S) → (D+1,D)		3	
	TANP	TANP S D				
	ASIN	ASIN S D	· Sin^{-1}(S+1,S) → (D+1,D)		3	
	ASINP	ASINP S D				
	ACOS	ACOS S D	· Cos^{-1}(S+1,S) → (D+1,D)		3	
	ACOSP	ACOSP S D				
	ATAN	ATAN S D	· Tan^{-1}(S+1,S) → (D+1,D)		3	
	ATANP	ATANP S D				
Conversion between angles and radians	RAD	RAD S D	· (S+1,S) → (D+1,D) Conversion from angles to radians		3	
	RADP	RADP S D				
	DEG	DEG S D	· (S+1,S) → (D+1,D) Conversion from radians to angles		3	
	DEGP	DEGP S D				
Square root	SQR	SQR S D	· $\sqrt{(S+1,S)}$ → (D+1,D)		3	
	SQRP	SQRP S D				
Exponent operations	EXP	EXP S D	· $e^{(S+1,S)}$ → (D+1,D)		3	
	EXPP	EXPP S D				
Natural logarithm	LOG	LOG S D	· Log e(S+1,S) → (D+1,D)		3	
	LOGP	LOGP S D				
Random number generation	RND	RND D	• Generates a random number (from 0 to less than 32767) and stores it at the device designated by (D).		2	
	RNDP	RNDP D				
Random number series update	SRND	SRND D	• Updates random number series according to the 16-bit BIN data stored in the device designated by (S).			
	SRNDP	SRNDP D				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Square root	BSQR	BSQR S D	$\sqrt{(S)}$ → (D)+0 Integral part +1 Decimal fraction		3	
	BSQRP	BSQRP S D				
	BDSQR	BDSQR S D	$\sqrt{(S+1, S)}$ → (D)+0 Integral part +1 Decimal fraction		3	
	BDSQRP	BDSQRP S D				
Trigonometric function	BSIN	BSIN S D	Sin(S) → (D)+0 Sign +1 Integral part +2 Decimal fraction		3	
	BSINP	BSINP S D				
	BCOS	BCOS S D	Cos(S) → (D)+0 Sign +1 Integral part +2 Decimal fraction		3	
	BCOSP	BCOSP S D				
	BTAN	BTAN S D	Tan(S) → (D)+0 Sign +1 Integral part +2 Decimal fraction		3	
	BTANP	BTANP S D				
	BASIN	BASIN S D	Sin^{-1}(S) → (D)+0 Sign +1 Integral part +2 Decimal fraction		3	
	BASINP	BASINP S D				
	BACOS	BACOS S D	Cos^{-1}(S) → (D)+0 Sign +1 Integral part +2 Decimal fraction		3	
	BACOSP	BACOSP S D				
	BATAN	BATAN S D	Tan^{-1}(S) → (D)+0 Sign +1 Integral part +2 Decimal fraction		3	
	BATANP	BATANP S D				

(13) Data control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Upper and lower limit controls	LIMIT	LIMIT S1 S2 S3 D	• When (S3)<(S1) ....Store value of (S1) at (D) • When (S1) ≤ (S2) ≤ (S3) ....Store value of (S3) at (D) • When (S2)<(S3) ....Store value of (S2) at (D)		5	
	LIMITP	LIMITP S1 S2 S3 D				
	DLIMIT	DLIMIT S1 S2 S3 D	• When ((S3)+1, (S3))<((S1)+ 1, S1) .... Store value of ((S1)+1, (S1)) at ((D)+1, (D)) • When ((S1)+1, (S1)) ≤ ((S3)+1, (S3))<(S2+1, S2) .... Store value of ((S3)+1, (S3)) at ((D)+1, (D)) • When ((S2), (S2)+1)<((S3), (S3)+1) .... Store value of ((S2)+1, (S2)) at ((D)+1, (D))		5	
	DLIMITP	DLIMITP S1 S2 S3 D				
Dead band controls	BAND	BAND S1 S2 S3 D	• When (S1) ≤ (S3) ≤ (S2) ... 0→(D) • When (S3)<(S1) .... (S3)-(S1)→(D) • When (S2)<(S3) .... (S3)-(S2)→(D)		5	
	BANDP	BANDP S1 S2 S3 D				
	DBAND	DBAND S1 S2 S3 D	• When ((S1)+1, (S1)) ≤ ((S3)+1, (S3)) ≤ ((S2)+1, (S2)) .... 0→(D)+1, (D)) • When ((S3)+1, (S3))<((S1)+1, (S1)) ... ((S3)+1, (S3)) - ((S1)+1, (S1))→((D)+1, (D)) • When ((S2)+1, (S2))<((S3)+1, (S3)) ... ((S3)+1, (S3)) - ((S2)+1, (S2))→((D)+1, (D))		5	
	DBANDP	DBANDP S1 S2 S3 D				
Zone controls	ZONE	ZONE S1 S2 S3 D	• When (S3)=0...................... 0→(D) • When (S3)>0 .... (S3)+(S2)→(D) • When (S3)<0 .... (S3)-(S1)→(D)		5	
	ZONEP	ZONEP S1 S2 S3 D				
	DZONE	DZONE S1 S2 S3 D	• When ((S3)+1, (S3))=0 ...... 0→((D)+1, (D)) • When ((S3)+1, (S3))>0 ... ((S3)+1, (S3))+((S2)+1, (S2))→((D)+1, (D)) • When ((S3)+1, (S3))<0 ... ((S3)+1, (S3)) + ((S1)+1, (S1))→((D)+1, (D))		5	
	DZONEP	DZONEP S1 S2 S3 D				

(14) Switching instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Block number designations	RSET	RSET S	• Converts extension file register block number to number designated by (S).		2	
	RSETP	RSETP S				
File set	QDRSET	QDRSET File name	• Sets file names used as file registers.		*2 + n	
	QDRSETP	QDRSETP File name				
	QCDSET	QCDSET File name	• Sets file names used as file registers.		*2 + n	
	QCDSETP	QCDSETP File name				

* :n ([number of file name characters] / 2) indicates a step. (decimal fraction is rounded up)

(15) Clock instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Read/ write clock data	DATERD	DATERD D	(Clock element) → (D)+0 Year +1 Month +2 Day +3 Hour +4 Min. +5 Sec. +6 Day of week		2	
	DATERDP	DATERDP D				
	DATEWR	DATEWR S	(D)+0 Year → (Clock element) +1 Month +2 Day +3 Hour +4 Min. +5 Sec. +6 Day of week		2	
	DATEWRP	DATEWRP S				
Clock data addition/ subtraction	DATE+	DATE+ S1 S2 D	(S1) Hour Min. Sec. + (S2) Hour Min. Sec. → (D) Hour Min. Sec.		4	
	DATE+P	DATE+P S1 S2 D				
	DATE−	DATE− S1 S2 D	(S1) Hour Min. Sec. - (S2) Hour Min. Sec. → (D) Hour Min. Sec.		4	
	DATE−P	DATE−P S1 S2 D				
Clock data transaction	SECOND	SECOND S D	(S) Hour Min. Sec. → (D) Sec. (lower) Sec. (upper)		3	
	SECONDP	SECONDP S D				
	HOUR	HOUR S D	(S) Sec. (lower) Sec. (upper) → (D) Hour Min. Sec.			
	HOURP	HOURP S D				

(16) Peripheral device instructions

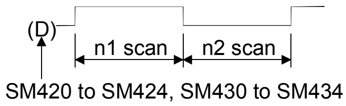
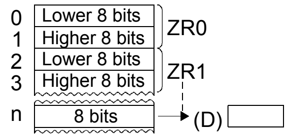
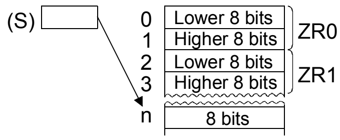
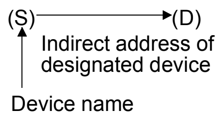
Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Input/ output to peripheral devices	MSG	MSG S	• Stores message designated by (S) at QnACPU. This message is displayed at the peripheral device.		2	
	PKEY	PKEY D	• Data input from the peripheral device is stored at device designated by (D).		2	

(17) Program control instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Program control instruc- tions	PSTOP	PSTOP Program name	• Places designated program in standby status.		*2 + n	
	PSTOPP	PSTOPP Program name				
	POFF	POFF Program name	• Turns OUT instruction coil of designated program OFF, and places program in standby status.		*2 + n	
	POFFP	POFFP Program name				
	PSCAN	PSCAN Program name	• Registers designated program as scan execution type.		*2 + n	
	PSCANP	PSCANP Program name				
	PLOW	PLOW Program name	• Registers designated program as low speed execution type.		*2 + n	
	PLOWP	PLOWP Program name				

*: n ([number of file name characters] / 2) indicates a step. (decimal fraction is rounded up).

(18) Other instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
WDT reset	WDT	WDT	• Resets watchdog timer during sequence program.		1	
	WDTP	WDTP				
Timing clock	DUTY	DUTY n1 n2 D			4	
Direct read/write operations in 1-byte units	ZRRDB	ZRRDB n D			3	
	ZRRDBP	ZRRDBP n D				
	ZRWRB	ZRWRB n S			3	
	ZRWRBP	ZRWRBP n S				
	ADRSET	ADRSET S D			3	
	ADRSETP	ADRSETP S D				
Numerical key input from keyboard	KEY	KEY S n D1 D2	• Takes in ASCII data for 8 points of input module designated by (S), converts to hexadecimal value following device number designated by (D1), and stores.		5	
Batch save of index register	ZPUSH	ZPUSH D	• Saves the contents of index registers Z0 to Z15 to a location starting from the device designated by (D).		2	
	ZPUSHP	ZPUSHP D				
Batch recovery of index register	ZPOP	ZPOP D	• Reads the data stored in the location starting from the device designated by (D) to index registers Z0 toZ15.			
	ZPOPP	ZPOPP D				
Batch write operation to E^2PROM file register	EROMWR	EROMWR S D1 n D2	• Writes a batch of data to E^2PROM file register.		5	
	EROM WRP	EROMWRP S D1 n D2				

Appendix 2.4 Instructions for data link

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Network refresh	ZCOM	J.ZCOM Jn	Refreshes the designated network.		5	
		JP.ZCOM Jn				
		G.ZCOM Un				
		GP.ZCOM Un				
QnA link instruction: Reading data from another station	READ	J.READ Jn (S1) (S2) (D1) (D2) G.READ Jn (S1) (S2) (D1) (D2)	Reads the word device data of another station to host station.		9	
		JP.READ Jn (S1) (S2) (D1) (D2) GP.READ Un (S1) (S2) (D1) (D2)				
	SREAD	J.SREAD Jn (S1) (S2) (D1) (D2) (D3) G.SREAD Un (S1) (S2) (D1) (D2) (D3)			10	
		JP.SREAD Jn (S1) (S2) (D1) (D2) (D3) GP.SREAD Un (S1) (S2) (D1) (D2) (D3)				
QnA link instruction: Writing data to other stations	WRITE	J.WRITE Jn (S1) (S2) (D1) (D2) G.WRITE Un (S1) (S2) (D1) (D2)	Writes the data of host station to the word device of other stations.		10	
		JP.WRITE Jn (S1) (S2) (D1) (D2) GP.WRITE Un (S1) (S2) (D1) (D2)				
	SWRITE	J.SWRITE Jn (S1) (S2) (D1) (D2) (D3) G.SWRITE Un (S1) (S2) (D1) (D2) (D3)			11	
		JP.SWRITE Jn (S1) (S2) (D1) (D2) (D3) GP.SWRITE Un (S1) (S2) (D1) (D2) (D3)				
QnA link instruction: Sending data	SEND	J.SEND Jn (S1) (S2) (D1) G.SEND Un (S1) (S2) (D1)	Sends data (message) to other stations.		8	
		JP.SEND Jn (S1) (S2) (D1) GP.SEND Un (S1) (S2) (D1)				
QnA link instruction: Receiving data	RECV	J.RECV Jn (S1) (S2) (D1) G.RECV Un (S1) (S2) (D1)	Receives data (message) sent to the host station.		8	
		JP.RECV Jn (S1) (S2) (D1) GP.RECV Un (S1) (S2) (D1)				
QnA link instruction: Transient requests from other stations	REQ	J.REQ Jn (S1) (S2) (D1) (D2) G.REQ Un (S1) (S2) (D1) (D2)	Sends a transient request to other stations and executes it.		8	
		JP.REQ Jn (S1) (S2) (D1) (D2) GP.REQ Un (S1) (S2) (D1) (D2)				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
QnA link instruction: Reading data from special function modules at remote I/O stations	ZNFR	JP.ZNFR Jn S1 S2 D1	Reads data from the special function modules at remote I/O stations.		8	
		GP.ZNFR Un S1 S2 D1				
QnA link instruction: Writing data to special function modules at remote I/O stations	ZNTO	J.ZNTO Jn S1 S2 D	Writes data from the special function modules at remote I/O stations.		8	
		JP.ZNTO Jn S1 S2 D				
		G.ZNTO Un S1 S2 D				
		GP.ZNTO Un S1 S2 D				
A-series compatible link instruction: Writing device data to other stations	ZNWR	J.ZNWR Jn n1 (D1) (S) n2 (D2)	Writes the data of host station to the word device of other stations.		32	
		JP.ZNWR Jn n1 (D1) (S) n2 (D2)				
A-series compatible link instruction: Reading device data from other stations	ZNRD	J.ZNRD Jn n1 (D1) (S) n2 (D2)	Reads the word device data of another station to host station.		32	
		JP.ZNRD Jn n1 (D1) (S) n2 (D2)				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
A-series compatible link instruction: Reading data from special function modules at remote I/O stations	RFRP	G.RFRP | Un | n1 | D1 | n2 | D2	Reads data from the special function modules at remote I/O stations.		11	
		GP.RFRP | Un | n1 | D1 | n2 | D2				
A-series compatible link instruction: Writing data to special function modules at remote I/O stations	RTOP	G.RTOP | Un | n1 | D1 | n2 | D	Writes data from the special function modules at remote I/O stations.		11	
		GP.RTOP | Un | n1 | D1 | n2 | D				
Reading routing information	RTREAD	Z.RTREAD | n | D	Reads data set at routing parameters.		7	
		ZP.RTREAD | n | D				
Registering routing information	RTWRITE	Z.RTWRITE | n | S	Writes routing data to the area designated by routing parameters.		8	
		ZP.RTWRITE | n | S				

Appendix 2.5 QCPU instructions

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Reading module information	UNIRD	UNIRD n1 D n2	• Reads the module information stored in the area starting from the I/O No. designated by (n) by the points designated by (n2), and stores it in the area starting from the device designated by (d).		4	
	UNIRDP	UNIRDP n1 D n2				
Trace set	TRACE	TRACE	• Stores the trace data set with peripheral device, by the number of times set when SM800, SM801and SM802 turn on, to the sampling trace file in the IC memory card.		1	
Trace reset	TRACER	TRACER	• Resets the data set by TRACE instruction.		1	
Writing data to designated file	SP.FW RITE	SP.FWRITE U0 S0 D0 S1 S2 D1	• Writes data to the designated file.		11	
Reading data from designated file	SP.FREAD	SP.FREAD U0 S0 D0 S1 S2 D1	• Reads data from the designated file.		11	
Loading program from memory	PLOADP	PLOADP S D	• Transfers the program stored in a memory card or standard memory (other than drive 0) to drive 0 and places the program in standby status.		3	
Unloading program from program memory	PUNLO ADP	PUNLOADP S D	• Deletes the standby program stored in standard memory (drive 0).		3	
Load + unload	PSWAPP	PSWAPP S1 S2 D	• Deletes standby program stored in standard memory (drive 0) designated by (S1). Then, transfers the program stored in a memory card or standard memory (other than drive 0) designated by (S2) to drive 0 and places it in standby status.		4	
High-speed block transfer of file register	RBMOV	RBMOV S D n	• Transfers n points of 16-bit data from the device designated by (S) to the location starting from the device designated by (D).		4	
	RBMOVP	RBMOVP S D n				

Category	Instruction symbol	Symbol	Processing details	Execution condition	Number of basic steps	Subset
Write to host station CPU shared memory	S.TO	S.TO n1 n2 n3 n4 D	• Writes the device data of the host station to the shared memory area of the host station CPU module.		5	
	S.TOP	S.TOP n1 n2 n3 n4 D				
Read from another station CPU shared memory	FROM	FROM n1 n2 D n3	• Reads device data from the CPU shared memory area of another station CPU module to the host station.		5	
	FROMP	FROMP n1 n2 D n3				
Automatic refresh of CPU shared memory	COM	COM	• Performs the automatic refresh of the intelligent function module, general data processing, and the automatic refresh of the CPU shared memory.		1	

Appendix 3 Special Relay List

Special relays, SM, are internal relays whose applications are fixed in the PLC. For this reason, they cannot be used by sequence programs in the same way as the normal internal relays. However, they can be turned ON or OFF as needed in order to control the CPU and remote I/O modules.

The headings in the list that follows have the following meanings.

Item	Function of item
No.	• Indicates the number of the special relay.
name	• Indicates the name of the special relay.
Meaning	• Indicates the contents of the special relay.
Explanation	• Contains detailed information about the contents of the special relay.
Set by (When set)	• Indicates whether the relay is set by the system or user and when setting is performed if it is set by the system. <Set by> 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. 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 occurrence: Set when error is generated Instruction execution: Set when instruction is executed Request: Set only when there is request from a user(through SM, etc.)
Corresponding ACPU M9□□□	• Indicates special relay (M9□□□) corresponding to the ACPU. (Indicates as "Change" when there has been a change in contents) • Items indicated as "New" have been newly added for Q/QnACPU
Applicable CPU	Indicates the applicable CPU type name. ○+ Rem: Indicates all the CPU and MELSECNET/H remote I/O modules. ○: Indicates all types of CPU QCPU: Indicates the Q-series CPU QnA: Indicates the QnA series and Q2ASCPU. Remote: Indicates the MELSECNET/H remote I/O modules. Each CPU type name: Indicates the relevant specific CPU module. (Example: Q4ARCPU, Q3ACPU)

For details on the following items, refer to these manuals:

- Networks → • Q corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
  - • Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)
  - • For QnA/Q4AR MELSECNET/10 Network System Reference Manual
- SFC → QCPU (Q Mode)/QnACPU Programming Manual (SFC)

POINT

(1) SD1200 to SD1255 are used for QnACPU.
These relays are vacant with QCPU.
(2) SM1500 or later is exclusively used for Q4ARCPU.

Special Relay List

(1) Diagnostic information

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM0	Diagnostic errors	OFF: No error ON: Error	• ON if diagnosis results show error occurrence (Includes the annunciator being ON and detecting an error by CHK instruction) • Stays ON subsequently even if normal operations restored	S (Error occurrence)	New	○+ Rem
SM1	Self-diagnostic error	OFF: No self-diagnosis errors ON: Self-diagnosis	• Comes ON when an error occurs as a result of self-diagnosis. (excludes the annunciator being ON and detecting an error by CHK instruction) • Remains ON if the condition is restored to normal thereafter.	S (Error occurrence)	M9008	
SM5	Error common information	OFF: No error common information ON: Error common information	• When SM0 is ON, ON if there is error common information	S (Error occurrence)	New	
SM16	Error individual information	OFF: No error common information ON: Error common information	• When SM0 is ON, ON if there is error individual information	S (Error occurrence)	New	
SM50	Error reset	OFF → ON: Error reset	• Performs error reset operation	U	New	
SM51	Battery low latch	OFF: Normal ON: Battery low	• ON if battery voltage at CPU module or memory card drops below rated value. Remains ON if the battery voltage returns to normal thereafter. • Synchronizes with the BAT. ALARM/BAT. LED.	S (Error occurrence)	M9007	○
SM52	Battery low	OFF: Normal ON: Battery low	• Same as SM51, but goes OFF subsequently when battery voltage returns to normal.	S (Error occurrence)	M9006	
SM53	AC DOWN detection	OFF: AC DOWN not detected ON: AC DOWN detected	• Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module. Reset when power is switched OFF, then ON.	S (Error occurrence)	M9005	
			• Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module. Reset when power is switched OFF, then ON.			QCPU
			• Turns ON if an instantaneous power failure of within 1ms occurs during use of the DC power supply module. Reset when power is switched OFF, then ON.			QnA
SM54	MINI link errors	OFF: Normal ON: Error	• Goes ON if MINI (S3) link error is detected at even one of the installed MELSECNET/MINI-S3 master modules. Remains ON if the condition is restored to normal thereafter.	S (Error occurrence)	M9004	QnA
SM56	Operation errors	OFF: Normal ON: Operation error	• ON when operation error is generated • Remains ON if the condition is restored to normal thereafter.	S (Error occurrence)	M9011	○
SM60	Fuse blown detection	OFF: Normal ON: Module with fuse blown	• Comes ON even if there is only one output module with a fuse blown, and remains ON even after return to normal • Fuse blown state is checked even for remote I/O station output modules.	S (Error occurrence)	M9000	○+ Rem
SM61	I/O module verification error	OFF: Normal ON: Error	• Comes ON if there is a discrepancy between the actual I/O modules and the registered information when the power is turned on, and remains ON even after return to normal. • I/O module verification is also performed for remote I/O station modules.	S (Error occurrence)	M9002	
SM62	Annunciator detection	OFF: Not detected ON: Detected	• Goes ON if even one annunciator F goes ON.	S (Instruction execution)	M9009	○
SM80	CHK detection	OFF: Not detected ON: Detected	• Goes ON if error is detected by CHK instruction. • Remains ON if the condition is restored to normal thereafter.	S (Instruction execution)	New	
SM90	Startup of watchdog timer for step transition (Enabled only when SFC program exists)	OFF: Not started (watchdog timer reset) ON: Started (watchdog timer started)	Corresponds to SD90 • Goes ON when measurement of step transition watchdog timer is commenced. • Resets step transition watchdog timer when it goes OFF.	U	M9108	
SM91			Corresponds to SD91		M9109	
SM92			Corresponds to SD92		M9110	
SM93			Corresponds to SD93		M9111	
SM94			Corresponds to SD94		M9112	
SM95			Corresponds to SD95		M9113	
SM96			Corresponds to SD96		M9114	
SM97			Corresponds to SD97		New	
SM98			Corresponds to SD98		New	
SM99			Corresponds to SD99		New	
SM120	Detection of external power supply OFF	OFF: Normal ON: There is a module whose external power supply is OFF.	• Goes ON when at least one module is in the status where the external power supply is OFF. Remains ON even after return to normal. *Applicable only for Q-series module. (For future use)	S (Error occurrence)	New	QCPU remote

Special Relay List

(2) System information

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM202	LED OFF command	OFF → ON: LED OFF	• When this relay goes from OFF to ON, the LEDs corresponding to the individual bits at SD202 go off	U	New	
SM203	STOP contact	STOP status	• Goes ON at STOP status	S (Status change)	M9042	
SM204	PAUSE contact	PAUSE status	• Goes ON at PAUSE status	S (Status change)	M9041	○
SM205	STEP-RUN contact	STEP-RUN status	• Goes ON at STEP-RUN status	S (Status change)	M9054	
SM206	PAUSE enable coil	OFF: PAUSE disabled ON: PAUSE enabled	• If this relay is ON when the remote PAUSE contact goes ON, the PAUSE state is entered.	U	M9040	
	Device test request acceptance status	OFF: Device test not yet executed ON: Device test executed	• Comes ON when the device test mode is executed on GX Developer.	S (Request)	New	remote
SM210	Clock data set request	OFF: Ignored ON: Set request	• When this relay goes from OFF to ON and after END instruction execution of subsequent scan, clock data stored in SD210 to SD213 are written to the CPU module.	U	M9025	○
SM211	Clock data error	OFF: No error ON: Error	• ON when error is generated in clock data (SD210 to SD213) value, and OFF if no error is detected.	S (Request)	M9026	
SM212	Time data display	OFF: Ignored ON: Display	• Displays clock data as month, day, hour, minute, and second at the LED display at front of CPU module.(Enabled only for Q3ACPU and Q4ACPU)	U	M9027	Q3A Q4A Q4AR
SM213	Clock data read request	OFF: Ignored ON: Read request	• When this relay is ON, clock data is read to SD210 through SD213 as BCD values.	U	M9028	○+ Rem
SM240	No. 1 CPU reset flag	OFF: No. 1 CPU reset cancel ON: No. 1 CPU resetting	• Goes OFF when reset of the No. 1 CPU is canceled. • Comes ON when the No. 1 CPU is resetting (including the case where the PLC is removed from the base). The other PLCs are also put in reset status.			
SM241	No. 2 CPU reset flag	OFF: No. 2 CPU reset cancel ON: No. 2 CPU resetting	• Goes OFF when reset of the No. 2 CPU is canceled. • Comes ON when the No. 2 CPU is resetting (including the case where the PLC is removed from the base). The other PLCs result in "MULTI CPU DOWN" (error code: 7000).			
SM242	No. 3 CPU reset flag	OFF: No. 3 CPU reset cancel ON: No. 3 CPU resetting	• Goes OFF when reset of the No. 3 CPU is canceled. • Comes ON when the No. 3 CPU is resetting (including the case where the PLC is removed from the base). The other PLCs result in "MULTI CPU DOWN" (error code: 7000).			
SM243	No. 4 CPU reset flag	OFF: No. 4 CPU reset cancel ON: No. 4 CPU resetting	• Goes OFF when reset of the No. 4 CPU is canceled. • Comes ON when the No. 4 CPU is resetting (including the case where the PLC is removed from the base). The other PLCs result in "MULTI CPU DOWN" (error code: 7000).	S (Status change)	New	QCPU function Ver. B
SM244	No. 1 CPU error flag	OFF: No. 1 CPU normal ON: No. 1 CPU during stop error	• Goes OFF when the No. 1 CPU is normal (including a continuation error). • Comes ON when the No. 1 CPU is during a stop error.			
SM245	No. 2 CPU error flag	OFF: No. 2 CPU normal ON: No. 2 CPU during stop error	• Goes OFF when the No. 2 CPU is normal (including a continuation error). • Comes ON when the No. 2 CPU is during a stop error.			
SM246	No. 3 CPU error flag	OFF: No. 3 CPU normal ON: No. 3 CPU during stop error	• Goes OFF when the No. 3 CPU is normal (including a continuation error). • Comes ON when the No. 3 CPU is during a stop error.			
SM247	No. 4 CPU error flag	OFF: No. 4 CPU normal ON: No. 4 CPU during stop error	• Goes OFF when the No. 4 CPU is normal (including a continuation error). • Comes ON when the No. 4 CPU is during a stop error.			
SM250	Max. loaded I/O read	OFF: Ignored ON: Read	• When this relay goes from OFF to ON, maximum loaded I/O number is read to SD250.	U	New	○+ Rem
SM251	I/O change flag	OFF: No replacement ON: Replacement	• By turning this relay ON after setting the head I/O number of the replaced I/O module to SD251, the I/O module can be replaced online (with power on). (Only one module can be replaced for each setting.) • Turn this relay ON in the test mode of the program or peripheral device for an I/O module change during RUN, or in the test mode of the peripheral device for an I/O change during STOP. • Do not execute a RUN/STOP mode change until I/O module change is finished.	U (END)	M9094	Q2A(S1) Q3A Q4A Q4AR
SM252	I/O change OK	OFF: Replacement prohibited ON: Replacement enabled	• Goes ON when I/O replacement is OK.	S (END)	New	
SM254	All stations refresh command	OFF: Refresh arrival station ON: Refresh all stations	• Effective for the batch refresh (also effective for the low speed cyclic) • Designate whether to receive arrival stations only or to receive all slave stations.	U (Every END processing)	New	QCPU

Special Relay List (Continued)

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM255	MELSECNET/10 module 1 information	OFF: Operative network ON: Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	○
SM256		OFF: Reads ON: Does not read	• For refresh from link to CPU module (B, W, etc.), designate whether to read from the link module.	U	New	○
SM257		OFF: Writes ON: Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	○
SM260	MELSECNET/10 module 2 information	OFF: Operative network ON: Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	○
SM261		OFF: Reads ON: Does not read	• For refresh from link to CPU module (B, W, etc.), designate whether to read from the link module.	U	New	○
SM262		OFF: Writes ON: Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	○
SM265	MELSECNET/10 module 3 information	OFF: Operative network ON: Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	○
SM266		OFF: Reads ON: Does not read	• For refresh from link to CPU module (B, W, etc.), designate whether to read from the link module.	U	New	○
SM267		OFF: Writes ON: Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	○
SM270	MELSECNET/10 module 4 information	OFF: Operative network ON: Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.))	S (Initial)	New	○
SM271		OFF: Reads ON: Does not read	• For refresh from link to CPU module (B, W, etc.), designate whether to read from the link module.	U	New	○
SM272		OFF: Writes ON: Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	○
SM280	CC-Link error	OFF: Normal ON: Error	• Goes ON when a CC-Link error is detected in any of the installed QJ61QBT11. Goes OFF when normal operation is restored.	S (Status change)	New	QCPU remote
			• Goes ON when a CC-Link error is detected in any of the installed A(1S)J61QBT11. Stays ON even after normal operation is restored.	S (Error occurrence)	New	QnA
SM320	Presence/absence of SFC program	OFF: SFC programs not used ON: SFC programs used	• Turns ON when an SFC program is registered • OFF when an SFC program is not registered.	S (Initial)	M9100	○
SM321	Start/stop SFC program	OFF: SFC programs stop ON: SFC programs start	• Initial value is set at the same value as SM320. (Goes ON automatically if SFC program is present.) • Turn this relay OFF before the SFC program processing to suspend SFC program. • Turn this relay from OFF to ON to start program execution. • Turn this relay from ON to OFF to stop program execution.	S (Initial) U	M9101 format change	○
SM322	SFC program start status	OFF: Initial Start ON: Continue	• Initial value can be ON or OFF by setting the parameter. • Turns this relay off to clear execution status of SFC program when SFC program is stopped. The block which received a start request starts from its initial step. • Turn this relay on to make the execution block restart from the execution step that was suspended when SFC program was stopped. (ON is valid only when Continue is specified in the parameter.) • SM902 is not automatically latch-specified.	S (Initial) U	M9102 format change	○
SM323	Presence/absence of continuous transition for entire block	OFF: Continuous transition not effective ON: Continuous transition effective	• Turn this relay off to make all the blocks take 1 step per 1 scan. • Turn this relay on to make all the blocks take steps continuously at 1 scan. • For specifying blocks, the continuous transition bit takes priority. (Specification is checked when a block is started.)	U	M9103	○
SM324	Continuous transition prevention flag	OFF: When transition is executed ON: When no transition	• OFF during operation in the continuous transition mode, and ON during continuous transition or when continuous transition is not executed. • Always ON during operation in the no continuous transition mode.	S (Instruction execution)	M9104	○
SM325	Output mode at block stop	OFF: OFF ON: Preserves	Selects the operational outputs of the active steps at the time of a block stop. • All coil outputs go OFF when this relay is OFF. • Coil outputs are preserved when this relay is ON.	S (Initial) U	M9196	○
SM326	SFC device clear mode	OFF: Clear device ON: Preserves device	Selects the device status when the stopped CPU restarts running after the sequence program or SFC program has been modified when the SFC program exists.	U	New	○
SM327	Output during end step execution	OFF: OFF ON: Hold	Selects operation output of held steps when terminating a block by end step execution. • All coil outputs go OFF when this relay is OFF. • Coil outputs are preserved when this relay is ON.	S (Initial) U	New	○

Special Relay List (Continued)

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM330	Operation mode for low speed execution type program	OFF: Asynchronous mode ON: Synchronous mode	• Asynchronous mode Mode in which the operation of the low speed execution type program is performed continuously within the excess time. • Synchronous mode Mode in which the operation of the low speed execution type program is not performed continuously and operation is performed from the next scan even if there is excess time.	U (END)	New	○
SM390	Access execution flag	ON indicates completion of intelligent function module access	• The status of the intelligent function module access instruction executed immediately before is stored. (This data is overwritten when the intelligent function module access instruction is executed again.) • Used by the user in a program as a completion bit.	S (Status change)	New	QCPU

(3) System clocks/counters

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM400	Always ON	ON OFF	• Always ON	S(Every END processing)	M9036	○
SM401	Always OFF	ON OFF	• Always OFF	S(Every END processing)	M9037	
SM402	After RUN, ON for 1 scan only	ON OFF 1 scan	• After RUN, ON for 1 scan only • This connection can be used for scan execution type programs only.	S(Every END processing)	M9038	
SM403	OFF for 1 scan only after RUN	ON OFF 1 scan	• After RUN, OFF for 1 scan only • This connection can be used for scan execution type programs only.	S(Every END processing)	M9039	
SM404	ON for 1 scan only after low speed execution type program RUN	ON OFF 1 scan	• After RUN, ON for 1 scan only • This connection can be used for low speed execution type programs only.	S(Every END processing)	New	
SM405	OFF for 1 scan only after low speed execution type program RUN	ON OFF 1 scan	• After RUN, OFF for 1 scan only • This connection can be used for low speed execution type programs only.	S(Every END processing)	New	
SM409	0.01 second clock	0.005 sec. 0.005 sec.	• Repeatedly changes between ON and OFF at 5-ms interval. • When turned OFF or reset, goes from OFF to start.	S (Status change)	New	QCPU
SM410	0.1 second clock	0.05 sec. 0.05 sec.	• Repeatedly changes between ON and OFF at each designated time interval. • When turned OFF or reset, goes from OFF to start. * Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)	S (Status change)	M9030	○
SM411	0.2 second clock	0.1 sec. 0.1 sec.			M9031	
SM412	1 second clock	0.5 sec. 0.5 sec.			M9032	
SM413	2 second clock	1 sec. 1 sec.			M9033	
SM414	2n second clock	n sec. n sec.	• Alternates between ON and OFF at intervals of seconds specified in SD414.	S (Status change)	M9034 format change	
SM415	2n (ms) clock	n(ms) n(ms)	• Alternates between ON and OFF at intervals of the time (unit: ms) specified in SD415.	S (Status change)	New	QCPU
SM420	User timing clock No. 0	n2 scan n2 scan n1 scan	• Relay repeats ON/OFF switching at fixed scan intervals. • When turned ON or reset, goes from OFF to start. • The ON/OFF intervals are set with the DUTY instruction. [DUTY | n1 | n2 | SM420]	S (Every END processing)	M9020	○
SM421	User timing clock No. 1				M9021	
SM422	User timing clock No. 2				M9022	
SM423	User timing clock No. 3				M9023	
SM424	User timing clock No. 4				M9024	
SM430	User timing clock No. 5		• For use with SM420 to SM424 low speed programs	S (Every END processing)	New	
SM431	User timing clock No. 6					
SM432	User timing clock No. 7					
SM433	User timing clock No. 8					
SM434	User timing clock No. 9					

(4) Scan information

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM510	Low speed program execution flag	OFF: Completed or not executed ON: Execution under way	• Goes ON when low speed execution type program is executed.	S(Every END processing)	New	○
SM551	Reads module service interval	OFF: Ignored ON: Read	• When this relay goes from OFF to ON, the module service interval designated by SD550 is read to SD551 through 552.	U	New	○+ Rem

(5) Memory cards

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM600	Memory card usable flag	OFF: Not available ON: Available	• ON when memory card is ready for use by user.	S (Initial)	New	○
SM601	Memory card protect flag	OFF: No protect ON: Protect	• Goes ON when memory card protect switch is ON.	S (Initial)	New	
SM602	Drive 1 flag	OFF: No drive 1 ON: Drive 1	• Turns ON when the mounted memory card is RAM.	S (Initial)	New	
SM603	Drive 2 flag	OFF: No drive 2 ON: Drive 2	• Turns ON when the mounted memory card is ROM.	S (Initial)	New	
SM604	Memory card in-use flag	OFF: Not used ON: In use	• Goes ON when memory card is in use.	S (Initial)	New	
SM605	Memory card remove/insert prohibit flag	OFF: Remove/insert enabled ON: Remove/insert prohibited	• Goes ON when memory card cannot be inserted or removed.	U	New	
SM609	Memory card remove/insert enable flag	OFF: Remove/insert disabled ON: Remove/insert enabled	• Turned ON by user to enable the removal/insertion of memory card. • Turned OFF by the system after the memory card is removed.	U/S	New	
SM620	Memory card B usable flag	OFF: Not available ON: Available	• Always ON	S (Initial)	New	QCPU
			• ON when memory card B is ready for use by user.	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR
SM621	Memory card B protect flag	OFF: No protect ON: Protect	• Always ON	S (Initial)	New	QCPU
			• Goes ON when memory card B protect switch is ON.	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR
SM622	Drive 3 flag	OFF: No drive 3 ON: Drive 3	• Always ON	S (Initial)	New	QCPU
			• Goes ON when drive 3 (card 2 RAM area) is present.	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR
SM623	Drive 4 flag	OFF: No drive 4 ON: Drive 4	• Always ON	S (Initial)	New	QCPU
			• Goes ON when drive 4 (card 2 ROM area) is present.	S (Initial)	New	Q2A (S1) Q3A Q4A Q4AR
SM624	Memory card B in-use flag	OFF: Not used ON: In use	• Goes ON when memory card B is in use.	S (Initial)	New	
SM625	Memory card B remove/insert prohibit flag	OFF: Remove/insert enabled ON: Remove/insert prohibited	• Goes ON when memory card B cannot be inserted or removed.	U	New	

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM640	File register use	OFF: File register not used ON: File register in use	• Goes ON when file register is in use.	S (Status change)	New	○
SM650	Comment use	OFF: Comment not used ON: Comment in use	• Goes ON when comment file is in use.	S (Status change)	New	
SM660	Boot operation	OFF: Internal memory execution ON: Boot operation in progress	• Goes ON while boot operation is in process. • Goes OFF if boot designation switch is OFF.	S (Status change)	New	○
SM672	Memory card A file register access range flag	OFF: Within access range ON: Outside access range	• Goes ON when access is made to area outside the range of file register R of memory card A. (Set within END processing.) • Reset at user program.	S/U	New	
SM673	Memory card B file register access range flag	OFF: Within access range ON: Outside access range	• Goes ON when access is made to area outside the range of file register R of memory card B. (Set within END processing.) • Reset at user program.	S/U	New	Q2A (S1) Q3A Q4A Q4AR

(6) Instruction-related special relays

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM700	Carry flag	OFF: Carry OFF ON: Carry ON	• Carry flag used in application instruction	S (Instruction execution)	M9012	○
SM701	Number of output characters selection	OFF: Outputs until NUL ON: 16 characters output	• When SM701 is OFF, output conducted until NUL (00H) code is encountered. • When SM701 is OFF, 16 characters of ASCII code are output.	U	M9049	
SM702	Search method	OFF: Search next ON: 2-part search	• Designates method to be used by search instruction. • Data must be arranged for 2-part search.	U	New	
SM703	Sort order	OFF: Ascending order ON: Descending order	• The sort instruction is used to designate whether data should be sorted in ascending order or in descending order.	U	New	
SM704	Block comparisons	OFF: Non-match found ON: All match	• Goes ON when all data conditions have been met for the BKCMP instruction.	S (Instruction execution)	New	
SM707	Selection of real number instructions processing type	OFF: Speed oriented ON: Accuracy oriented	• When SM707 is OFF, real number instructions are processed at high speed. • When it is ON, real number instructions are processed with high accuracy.	U	New	Q4AR
SM710	CHK instruction priority ranking flag	OFF: Conditions priority ON: Pattern priority	• Remains as originally set when OFF. • CHK priorities updated when ON.	S (Instruction execution)	New	○
SM711	Divided transmission status	OFF: Other than during divided processing ON: During divided processing	• In processing of AD57(S1), goes ON when canvas screen is divided for transfer, and goes OFF when split processing is completed.	S (Instruction execution)	M9065	QnA
SM712	Transmission processing selection	OFF: Batch processing ON: Divided processing	• In processing of AD57(S1), goes ON when canvas screen is divided for transfer.	S (Instruction execution)	M9066	
SM714	Communication request registration area BUSY signal	OFF: Communication request to remote terminal module enabled ON: Communication request to remote terminal module disabled	• Used to determine whether communications requests to remote terminal modules connected to the AJ71PT32-S3 can be executed or not.	S (Instruction execution)	M9081	
SM715	EI flag	0: During DI 1: During EI	• ON when EI instruction is being executed.	S (Instruction execution)	New	○
SM720	Comment read completion flag	OFF: Comment read not completed ON: Comment read completed	• Turns on only during one scan when the processing of the COMRD or PRC instruction is completed.	S (Status change)	New	QCPU
SM721	File being accessed	OFF: File not accessed ON: File being accessed	• Switches ON while a file is being accessed by the S.FWRITE, S. FREAD, COMRD, PRC, or LEDC instruction.	S (Status change)	New	
SM722	BIN/DBIN instruction error disabling flag	OFF: Error OK ON: Error NG	• Turned ON when "OPERATION ERROR" is suppressed for BIN or DBIN instruction.	U	New	

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM730	BUSY signal for CCLink communication request registration area	OFF: Request for communication wit intelligent device station enabled ON: Request for communication wit intelligent device station disabled	• Used for determination whether to enable or disable the communication request for the intelligent device station connected with A(1S)J61QBT11.	S (Instruction execution)	New	QnA
SM736	PKEY instruction execution in progress flag	OFF: Instruction not executed ON: Instruction execution	• ON when PKEY instruction is being executed. Goes OFF when CR is input, or when input character string reaches 32 characters.	S (Instruction execution)	New	○
SM737	Keyboard input reception flag for PKEY instruction	OFF: Keyboard input reception enabled ON: Keyboard input reception disabled	• Goes ON when keyboard input is being conducted. Goes OFF when keyboard input has been stored at the CPU.	S (Instruction execution)	New	
SM738	MSG instruction reception flag	OFF: Instruction not executed ON: Instruction execution	• Goes ON when MSG instruction is executed.	S (Instruction execution)	New	○
SM774	PID bumpless processing	OFF: Forces match ON: Does not force match	• Specify whether the set value (SV) will be matched with the process value (PV) in the manual mode.	U	New	
SM775	Selection of link refresh processing during COM instruction execution	OFF: Performs link refresh ON: Performs no link refresh	• Select whether link refresh processing will be performed or not when only general data is processed at the execution of the COM instruction.	U	New	
SM776	Enable/disable local device at CALL	OFF: Local device disabled ON: Local device enabled	• Set whether the local device of the subroutine program called at execution of the CALL instruction is valid or invalid.	U (Status change)	New	
SM777	Enable/disable local device in interrupt program	OFF: Local device disabled ON: Local device enabled	• Set whether the local device at execution of the interrupt program is valid or invalid.	U (Status change)	New	
SM780	CC-Link dedicated instruction executable	OFF: CC-Link dedicated instruction executable ON: CC-Link dedicated instruction not executable	• Switches ON when the number of the CC-Link dedicated instructions that can be executed simultaneously reaches 32. Switches OFF when the number goes below 32.	U (Status change)	New	QnA

(7) Debug

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM800	Trace preparation	OFF: Not ready ON: READY	• Switches ON when the trace preparation is completed.	S (Status change)	New	QCPU
	Sampling trace preparation		• Switches ON when the sampling trace preparation is completed.	S (Status change)	New	QnA
SM801	Trace start	OFF: Suspend ON: Start	• Trace started when this goes ON. • Suspended when OFF (Related special M all OFF)	U	M9047	QCPU
	Sampling trace start		• Sampling trace started when this goes ON. • Suspended when OFF (Related special M all OFF)	U	M9047	QnA
SM802	Trace execution in progress	OFF: Suspend ON: Start	• Switches ON during execution of trace.	S (Status change)	M9046	QCPU
	Sampling trace execution in progress		• Switches ON during execution of sampling trace.	S (Status change)	M9046	QnA
SM803	Trace trigger	OFF →ON: Start	• Trace is triggered when this relay switches from OFF to ON. (Identical to TRACE instruction execution status)	U	M9044	QCPU
	Sampling trace trigger		• Sampling trace is triggered when this relay switches from OFF to ON. (Identical to STRA instruction execution status)	U	M9044	QnA
SM804	After trace trigger	OFF: Not after trigger ON: After trigger	• Switches After trace is triggered.	S (Status change)	New	QCPU
	After sampling trace trigger		• Switches After sampling trace is triggered.	S (Status change)	New	QnA

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM805	Trace completed	OFF: Not completed ON: End	• Switches ON at completion of trace.	S (Status change)	M9043	QCPU
	Sampling trace completed		• Switches ON at completion of sampling trace.	S (Status change)	M9043	QnA
SM806	Status latch preparation	OFF: Not ready ON: READY	• Goes ON when status latch is ready.	S (Status change)	New	QnA
SM807	Status latch command	OFF →ON: Latch	• Runs status latch command	U	New	
SM808	Status latch completion	OFF: Not ready ON: Ready	• Goes ON when program trace is ready.	S (Status change)	M9055	
SM809	Status latch clear	OFF →ON: Clear	• Enable next status latch	U	New	
SM810	Program trace preparation	OFF: Not ready ON: READY	• Goes ON when program trace is ready.	S (Status change)	New	
SM811	Start program trace	OFF: Suspend ON: Start	• Program trace started when this goes ON. • Suspended when OFF (Related special M all OFF)	S (Status change)	New	
SM812	Program trace execution under way	OFF: Suspend ON: Start	• ON when program trace execution is underway.	U	New	
SM813	Program trace trigger	OFF →ON: Start	• Program trace trigger goes ON when this goes from OFF to ON. (Identical to PTRA instruction execution status)	S (Status change)	New	
SM814	After program trace trigger	OFF: Not after trigger ON: After trigger	• Goes ON after program trace trigger.	S (Status change)	New	
SM815	Program trace completion	OFF: Not completed ON: End	• Goes ON at completion of program trace.	S (Status change)	New	
SM820	Step trace preparation	OFF: Not ready ON: READY	• Goes ON after step trace is registered and ready.	U	New	○
SM821	Step trace starts	OFF: Suspend ON: Start	• When this goes ON, step trace is started. • Suspended when OFF (Related special M all OFF)	S (Status change)	M9182 format change	
SM822	Step trace execution underway	OFF: Suspend ON: Start	• Goes ON when step trace execution is underway. • Goes OFF at completion or suspension.	S (Status change)	M9181	
SM823	After step trace trigger	OFF: Not after trigger ON: Is after first trigger	• Goes ON if even 1 block within the step trace being executed is triggered. • Goes OFF when step trace is started.	S (Status change)	New	
SM824	After step trace trigger	OFF: Is not after all triggers ON: Is after all triggers	• Goes ON if all blocks within the step trace being executed are triggered. • Goes OFF when step trace is started.	S (Status change)	New	
SM825	Step trace completed	OFF: Not completed ON: End	• Goes ON at step trace completion. • Goes OFF when step trace is started.	S (Status change)	M9180	
SM826	Trace error	OFF: Normal ON: Errors	• Switches ON if error occurs during execution of trace.	S (Status change)	New	QCPU
	Sampling trace error		• Switches ON if error occurs during execution of sampling trace.	S (Status change)	New	QnA
SM827	Status latch error	OFF: Normal ON: Errors	• Switches ON if error occurs during execution of status latch.	S (Status change)	New	
SM828	Program trace error	OFF: Normal ON: Errors	• Switches ON if error occurs during execution of program trace.	S (Status change)	New	

(8) Latch area

No.	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU M9□□□	Corresponding CPU
SM900	Power cut file OFF	OFF: No power cut file ON: Power cut file present	• Goes ON if a file being accessed is present when power is disconnected.	S/U (Status change)	New	QnA
SM910	RKEY registration flag	OFF: Keyboard input not registered ON: Keyboard input registered	• Goes ON at registration of keyboard input. OFF if keyboard input is not registered.	S (Instruction execution)	New	○

(9) A to Q/QnA conversion correspondences

Special relays SM1000 to SM1255 are the relays which correspond to ACPU special relays M9000 to M9255 after A to Q/QnA conversion.
These special relays are all set by the system, and cannot be set by the user program.
To turn them ON/OFF by the user program, change the special relays in the program into those of QCPU/QnACPU.
However, some of SM1084 and SM1200 to SM1255 (corresponding to M9084 and M9200 to M9255 before conversion) can be turned ON/OFF by the user program, if they could be turned ON/OFF by the user program before conversion.
For details on the ACPU special relays, see the user's manuals for the individual CPUs, and MELSECNET or MELSECNET/B Data Link System Reference Manuals.

POINT

The processing time may be longer when converted special relays are used with the QCPU. Uncheck "A-series CPU compatibility setting" within the PC system setting in GX Developer parameters when converted special relays are not used.

REMARK

The following are additional explanations about the Special Relay for Modification column.

1) When a special relay for modification is provided, the device number should be changed to the provided QCPU/QnACPU special relay.
2) When ⊟ is provided, the converted special relay can be used for the device number.
3) When ⊠ is provided, the device number does not work with QCPU/QnACPU.

Special Relay List

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9000	SM1000	——	Fuse blown	OFF: Normal ON: Module with blown fuse	• Turned on when there is one or more output modules of which fuse has been blown, and remains ON if the condition is restored to normal thereafter. Output modules of remote I/O stations are also checked fore fuse condition.	○
M9002	SM1002	——	I/O module verification error	OFF: Normal ON: Error	• Turned on if the status of I/O module is different form entered status when power is turned on, and remains ON if the condition is restored to normal thereafter. I/O module verification is also performed for remote I/O station modules. [Reset is enabled only when special registers SD1116 to SD1123 are reset.]	
M9004	SM1004	——	MINI link error	OFF: Normal ON: Error	• Goes ON if MINI (S3) link error is detected at even one of the installed MELSECNET/MINI-S3 master modules, and remains ON if the condition is restored to normal thereafter.	QnA
M9005	SM1005	——	AC DOWN detection	OFF: AC DOWN not detected ON: AC DOWN detected	• Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module. Reset when power is switched OFF, then ON. • Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module. Reset when power is switched OFF, then ON. Turns ON if an instantaneous power failure of within 1ms occurs during use of the DC power supply module. Reset when power is switched OFF, then ON.	○
M9006	SM1006	——	Battery low	OFF: Normal ON: Battery low	• Turns ON when the battery voltage drops to or below the specified, and turns OFF when the battery voltage returns to normal thereafter.	
M9007	SM1007	——	Battery low latch	OFF: Normal ON: Battery low	• Turns ON when the battery voltage drops to or below the specified, and remains ON if the battery voltage returns to normal thereafter.	
M9008	SM1008	SM1	Self-diagnostic error	OFF: No error ON: Error	• Turned on when error is found as a result of self-diagnosis.	

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9009	SM1009	SM62	Annunciator detection	OFF: No F number detected ON: F number Detected	• Turned on when OUT F of SET F instruction is executed. Switched off when SD1124 data is zeroed.	
M9011	SM1011	SM56	Operation error flag	OFF: No error ON: Error	• Turns on when operation error occurs during execution of application instruction, and remains ON even if the condition is restored to normal thereafter.	
M9012	SM1012	SM700	Carry flag	OFF: Carry OFF ON: Carry ON	• Carry flag used in application instruction	
M9016	SM1016		Data memory clear flag	OFF: Ignored ON: Output cleared	• Clears the data memory including the latch range (other than special relays and special registers) in remote run mode from computer, etc. when SM1016 is on.	
M9017	SM1017		Data memory clear flag	OFF: Ignored ON: Output cleared	• Clears the unlatched data memory (other than special relays and special registers) in remote run mode from computer, etc. when SM1017 is on.	
M9020	SM1020	——	User timing clock No. 0	n2 scan n2 scan n1 scan	• Relay which repeats on/off at intervals of predetermined scan. • When power is turned on or reset is per-formed, the clock starts with off. • Set the intervals of on/off by $\boxed{\text{DUTY}}$ instruction. DUTY n1 n2 M9020	
M9021	SM1021	——	User timing clock No. 1			
M9022	SM1022	——	User timing clock No. 2			
M9023	SM1023	——	User timing clock No. 3			
M9024	SM1024	——	User timing clock No. 4			
M9025	SM1025	——	Clock data set request	OFF: Ignored ON: Set request present used	• Writes the clock data stored in SD1025 to SD1028 to the CPU module after the END instruction is executed in the scan in which SM1025 turned from OFF to ON.	
M9026	SM1026	——	Clock data error	OFF: No error ON: Error	• Switched ON by clock data (SD1025 to SD1028) error, and OFF if no error is detected.	
M9027	SM1027	——	Time data display	OFF: Ignored ON: Display	• Clock data is read from SD1025 to SD1028 and month, day, hour, minute and minute are indicated on the CPU module front LED display.	
M9028	SM1028	——	Clock data read request	OFF: Ignored ON: Read request	• Reads clock data to SD1025 to SD1028 in BCD when SD1028 is on.	○
M9029	SM1029		Batch processing of data communication s requests	OFF: Batch processing not conducted ON: Batch processing conducted	• The SM1029 relay is turned on using a sequence program to process all data communication requests accepted during one scan in the END processing of that scan. • The batch processing of the data communication requests can be turned on and off during running. • The default is OFF (processed one at a time for each END processing in the order in which data communication requests are accepted).)	
M9030	SM1030	——	0.1 second clock	0.05 sec. 0.05 sec.	• 0.1 second, 0.2 second, 1 second and 2 second, clocks are generated. • Not turned on or off per scan but turned on and off even during scan if corresponding time has elapsed. • Starts with off when power supply is turned on or CPU module reset is performed.	
M9031	SM1031	——	0.2 second clock	0.1 sec. 0.1 sec.		
M9032	SM1032	——	1 second clock	0.5 sec. 0.5 sec.		
M9033	SM1033	——	2 second clock	1 sec. 1 sec.		
M9034	SM1034	——	1 minute clock	30 sec. 30 sec.		
M9036	SM1036	——	Always ON	ON OFF	• Used as dummy contacts of initialization and application instruction in sequence program. • SM1038 and SM1037 are turned on and off without regard to position of key switch on CPU module front. SM1038 and SM1039 are under the same condition as RUN status except when the key switch is at STOP position, and turned off and on. Switched off if the key switch is in STOP position. SM1038 is on for one scan only and SM1039 is off for one scan only if the key switch is not in STOP position.	
M9037	SM1037	——	Always OFF	ON OFF		
M9038	SM1038	——	ON for 1 scan only after RUN	ON OFF 1 scan		
M9039	SM1039	——	RUN flag (After RUN, OFF for 1 scan only)	ON OFF 1 scan		
M9040	SM1040	SM206	PAUSE enable coil	OFF: PAUSE disabled ON: PAUSE enabled	• When RUN key switch is at PAUSE position or pause contact has turned on and if SM204 is on, PAUSE mode is set and SM206 is turned on.	
M9041	SM1041	SM204	PAUSE status contact	OFF: PAUSE not in effect ON: PAUSE in effect		

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9042	SM1042	SM203	STOP status contact	OFF: STOP not in effect ON: STOP in effect	• Switched on when the RUN key switch is in STOP position.	○
M9043	SM1043	SM805	Sampling trace completed	OFF: Sampling trace in progress ON: Sampling trace completed	• Turned on upon completion of sampling trace performed the number of times preset by parameter after STRA instruction is executed. Reset when STRAR instruction is executed.	○
M9044	SM1044	SM803	Sampling trace	OFF→ON: Same as STRA execution ON→OFF: Same as STRAR execution	• Turning on/off SM803 can execute STRA / STRAR instruction. (SM803 is forcibly turned on/off by a peripheral device.) When switched from OFF to ON: STRA instruction When switched from ON to OFF: STRAR instruction The value stored in SD1044 is used as the condition for the sampling trace. At scanning, at time → Time (10 ms unit)	○
M9045	SM1045		Watchdog timer (WDT) reset	OFF: Does not reset WDT ON: Resets WDT	• The SM1045 relay is turned on to reset the WDT when the ZCOM instruction and data communication request batch processing are executed. (used when the scan time exceeds 200 ms)	○
M9046	SM1046	SM802	Sampling trace	OFF: Trace not in progress ON: Trace in progress	• Switched on during sampling trace.	○
M9047	SM1047	SM801	Sampling trace preparations	OFF: Sampling trace suspended ON: Sampling trace started	• Sampling trace is not executed unless SM801 is turned ON. Sampling trace is suspended when SM801 goes OFF.	○
M9049	SM1049	SM701	Selection of number of characters output	OFF: Output until NUL encountered ON: 16 characters output	• When SM701 is OFF, characters up to NULL (00H) code are output. • When SM701 is ON, ASCII codes of 16 characters are output.	○
M9051	SM1051		CHG instruction execution disable	OFF: Enabled ON: Disable	• Switched ON to disable the CHG instruction. • Switched ON when program transfer is requested. Automatically switched OFF when transfer is complete.	○
M9052	SM1052		SEG instruction switch	OFF: 7SEG segment display ON: I/O partial refresh	• When SM1052 is ON, the SEG instruction is executed as an I/O partial refresh instruction. When SM1052 is OFF, the SEG instruction is executed as a 7-SEG display instruction.	○
M9054	SM1054	SM205	STEP RUN flag	OFF: STEP RUN not in effect ON: STEP RUN in effect	• Switched on when the RUN key switch is in STEP RUN position.	QnA
M9055	SM1055	SM808	Status latch completion flag	OFF: Not completed ON: Completed	• Turned on when status latch is completed. Turned off by reset instruction.	QnA
M9056	SM1056		Main side P, I set request	OFF: Other than when P, I set being requested ON: P, I set being requested	• Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	○
M9057	SM1057		Sub side P, I set request	OFF: Other than when P, I set being requested ON: P, I set being requested		○
M9058	SM1058		Main side P, I set request	Momentarily ON at P, I set completion	• Turned ON once when the P, I set has been completed, and then turned OFF again.	○
M9059	SM1059		Sub program P, I set completion	Momentarily ON at P, I set completion		○
M9060	SM1060		Sub program 2 P, I set request	OFF: Other than when P, I set being requested ON: P, I set being requested	• Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	○
M9061	SM1061		Sub program 3 P, I set request	OFF: Other than when P, I set being requested ON: P, I set being requested		○
M9065	SM1065	SM711	Divided execution detection	OFF: Divided processing not underway ON: During divided processing	• Turned on when canvas screen transfer to AD57(S1)/AD58 is done by divided processing, and turned off at completion of divided processing.	QnA
M9066	SM1066	SM712	Divided processing request flag	OFF: Batch processing ON: Divided processing	• Turned on when canvas screen transfer to AD57(S1)/AD58 is done by divided processing.	QnA
M9070	SM1070		A8UPU/A8PUJ required search time	OFF: Read time not shortened ON: Read time shortened	• Turned ON to shorten the search time in the A8UPU/A8PUJ. (In this case, the scan time is extended by 10 %.) * A8UPU and A8PUJ are not used with QCPU/QnACPU.	○

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9081	SM1081	SM714	Communication request registration area BUSY signal	OFF: Empty spaces in communication request registration area ON: No empty spaces in communication request registration area	• Indication of communication enable/disable to remote terminal modules connected to the MELSECNET/MINI-S3 master module, A2C or A52G.	QnA
M9084	SM1084		Error check	OFF: Error check executed ON: No error check	• It is set whether the error checks below are performed or not when the END instruction is processed (to set the END instruction processing time). • Check for breakage of fuse • Collation check of I/O module • Check of battery	○
M9091	SM1091		Operation error details flag	OFF: No error ON: Error	• Turns ON when the detail factor of the operation error is stored into SD1091. Remains ON if the condition is restored to normal thereafter.	
M9094	SM1094	SM251	I/O change flag	OFF: Replacement ON: No replacement	• The I/O module can be changed online (with power on) when SM251 is turned ON after the head I/O number of the I/O module is set to SD251. (One module only is allowed to be changed by one setting.) • To be switched on in the program or peripheral device test mode to change the module during CPU RUN. To be switched on in peripheral device test mode to change the module during CPU STOP. • RUN/STOP mode must not be changed until I/O module change is complete.	QnA
M9100	SM1100	SM320	Presence/ absence of SFC program	OFF: SFC programs not used ON: SFC programs used	• Turned on if the SFC program is registered. Turned off if it is not.	○
M9101	SM1101	SM321	Start/stop SFC program	OFF: SFC programs stop ON: SFC programs start	• Turned on by user to start SFC program. Turned OFF to stop SFC program by disabling operational outputs of execution steps.	
M9102	SM1102	SM322	SFC program start status	OFF: Initial Start ON: Continue	• Selects a start step of restarting SFC program with SM322. ON: Makes the execution block restart from the execution step that was suspended when SFC program was stopped. OFF: Clears execution status of SFC program when SFC program is stopped. The block 0 starts from its initial step. • Once turned on, this relay remains on even if power supply is cut by latch with system. To start with the initial step of blocks at power-on, turn this relay off with the sequence block.	
M9103	SM1103	SM323	Presence/ absence of continuous transition	OFF: Continuous transition not effective ON: Continuous transition effective	• Set whether continuous transition will be performed in 1 scan for all the steps on which transition conditions are met. ON: Transits continuously (continuous transition enabled). OFF: Transits 1 step at 1 scan (continuous transition disabled).	
M9104	SM1104	SM324	Continuous transition suspension flag	OFF: When transition is completed ON: When no transition	• ON when continuous transition is not executed during operation in the continuous transition mode, and OFF when transition of 1 step is completed. By writing AND condition as the transition condition of SM324, continuous transition can be prevented from being performed on the corresponding step.	
M9108	SM1108	SM90	Step transition watchdog timer start (equivalent of SD90)	OFF: Watchdog timer reset ON: Watchdog timer reset start	• Turns ON when the measurement of the step transition watchdog timer is started. Turning this relay OFF resets the step transition watchdog timer.	
M9109	SM1109	SM91	Step transition watchdog timer start (equivalent of SD91)			
M9110	SM1110	SM92	Step transition watchdog timer start (equivalent of SD92)			
M9111	SM1111	SM93	Step transition watchdog timer start (equivalent of SD93)			

Special Relay List (Continued)

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9112	SM1112	SM94	Step transition watchdog timer start (equivalent of SD94)	OFF: Watchdog timer reset ON: Watchdog timer reset start	• Turns ON when the measurement of the step transition watchdog timer is started. Turning this relay OFF resets the step transition watchdog timer.	○
M9113	SM1113	SM95	Step transition watchdog timer start (equivalent of SD95)	OFF: Watchdog timer reset ON: Watchdog timer reset start	• Turns ON when the measurement of the step transition watchdog timer is started. Turning this relay OFF resets the step transition watchdog timer.	○
M9114	SM1114	SM96	Step transition watchdog timer start (equivalent of SD96)	OFF: Watchdog timer reset ON: Watchdog timer reset start	• Turns ON when the measurement of the step transition watchdog timer is started. Turning this relay OFF resets the step transition watchdog timer.	○
M9180	SM1180	SM825	Active step sampling trace completion flag	OFF: Trace start ON: Trace completed	• Set when sampling trace of all specified blocks is completed. Reset when sampling trace is started.	○
M9181	SM1181	SM822	Active step sampling trace execution flag	OFF: Trace not being executed ON: Trace execution underway	• Set when sampling trace is being executed. Reset when sampling trace is completed or suspended.	○
M9182	SM1182	SM821	Active step sampling trace permission	OFF: Trace disable/suspend ON: Trace enable	• Selects sampling trace execution enable/disable. ON: Sampling trace execution is enabled. OFF: Sampling trace execution is disabled. If turned off during sampling trace execution, trace is suspended.	○
M9196	SM1196	SM325	Operation output at block stop	OFF: Coil output OFF ON: Coil output ON	• Selects the operation output when block stop is executed. ON: Retains the ON/OFF status of the coil being used by using operation output of the step being executed at block stop. OFF: All coil outputs are turned off. (Operation output by the SET instruction is retained regardless of the ON/OFF status of M9196.)	○
M9197	SM1197		Switch between blown fuse and I/O verification error display	SM1197 / SM1198 / I/O numbers to be displayed OFF / OFF / X/Y 0 to 7F0 ON / OFF / X/Y 800 to FF0 OFF / ON / X/Y 1000 to 17F0 ON / ON / X/Y 1800 to 1FF0	Switches I/O numbers in the fuse blow module storage registers (SD1100 to SD1107) and I/O module verify error storage registers (SD1116 to SD1123) according to the combination of ON/OFF of the SM1197 and SM1198.	○
M9198	SM1198		Switch between blown fuse and I/O verification error display	(see M9197)	(see M9197)	○
M9199	SM1199		Data recovery of online sampling trace/status latch	OFF: Data recovery disabled ON: Data recovery enabled	• Recovers the setting data stored in the CPU module at restart when sampling trace/status latch is executed. • SM1199 should be ON to execute again. (Unnecessary when writing the data again from peripheral devices.)	○
M9200	SM1200	——	ZNRD instruction (LRDP instruction for ACPU) completion	OFF: Not accepted ON: Accepted	• Depends on whether or not the ZNRD (word device read) instruction has been received. • Used in the program as an interlock for the ZNRD instruction. • Use the RST instruction to reset.	QnA
M9201	SM1201	——	ZNRD instruction (LRDP instruction for ACPU) completion	OFF: Not completed ON: End	• Depends on whether or not the ZNRD (word device read) instruction execution is complete. • Used as a condition contact for resetting SM1202 and SM1203 after the ZNRD instruction is complete. • Use the RST instruction to reset.	QnA

Appendix 4 Special Register List

Special registers, SD, are internal relays with fixed applications in the PLC. For this reason, it is not possible to use these registers in sequence programs in the same way that normal registers are used. Data stored in the special registers are stored as BIN values if no special designation has been made to the contrary.
Data stored in the special registers are stored as BIN values if no special designation has been made to the contrary.

The heading descriptions in the following special register lists are shown in the following table.

Item	Function of item
Number	• Indicates the number of special register.
Name	• Indicates the name of the special register.
Meaning	• Indicates the contents of the special register.
Explanation	• Indicates the detailed contents of the special register.
Set by (When set)	• Indicates whether the relay is set by the system or user, and if it is set by the system, when setting is performed. <Set by> S : Set by system U : Set by user (Sequence program or test operations at a peripheral device) S/U : Set by both system and user <When set> indicated only for registers set by system. Each END : Set during each END processing Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN) Status change : Set only when there is a change in status Error : Set when error occurs Instruction execution: Set when instruction is executed Request : Set only when there is a user request (through SM, etc.)
Corresponding ACPU D9□□□	• Indicates corresponding special register in ACPU (When the contents are changed, the special register is represented as changed) • New indicates the special register newly added to the QnACPU or Q series CPU module.

For details on the following items, refer to the following manuals:

• Networks → • Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)
• Q Corresponding MELSECNET/H
• QnA/Q4AR MELSECNET/10 Network System Reference Manual

• SFC → QCPU (Q Mode)/QnACPU Programming Manual (SFC)

POINT

(1) SD1200 to SD1255 are used for QnACPU.
These relays are vacant with QCPU.
(2) SM1500 or later is exclusively used for Q4ARCPU.

(1) Diagnostic information

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD0	Diagnostic errors	Diagnosis error code	• Error codes for errors found by diagnosis are stored as BIN data. • Contents identical to latest fault history information.	S (Error)	D9008 format change	○+Rem
SD1	Clock time for diagnosis error occurrence	Clock time for diagnosis error occurrence	• Year (last two digits) and month that SD0 data is updated is stored as BCD 2-digit code. B15 to B8 | B7 to B0 Year (0 to 99) | Month (1 to 12) (e.g.) Oct. 1995 H9510	S (Error)	New	
SD2			• The day and hour that SD0 is updated is stored as BCD 2-digit code. B15 to B8 | B7 to B0 Day (1 to 31) | Hour (0 to 23) (e.g.) 25th, 10 o'clock H2510			
SD3			• The minute and second that SD0 data was updated is stored as BCD 2-digit code. B15 to B8 | B7 to B0 Min. (1 to 59) | Sec. (0 to 59) (e.g.) 35, 48 H3548			
SD4	Error information categories	Error information category code	Category codes which help indicate what type of error information is being stored in the common information areas (SD5 through SD15) and the individual information areas (SD16 through SD26) are stored here. B15 to B8 | B7 to B0 Individual info category code | Common info category code • The common information category codes store the following codes: 0 : No error 1 : Unit/module No./ PLC No. * 2 : File name/Drive name 3 : Time (value set) 4 : Program error location 5 : Switch cause (for Q4AR only) For a multiple CPU system, the module number or PLC number is stored depending on the error that occurred. (Refer to the corresponding error code for which number has been stored.) PLC No. 1: 1, PLC No. 2: 2, PLC No. 3: 3, PLC No. 4: 4 • The individual information category codes store the following codes: 0 : No error 1 : (Vacancy) 2 : File name/Drive name 3 : Time (value actually measured) 4 : Program error location 5 : Parameter number 6 : Annunciator F number 7 : CHK instruction malfunction number	S (Error)	New	

Special Register List (Continued)

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When set)</th><th>Corresponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr>
<td>SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14

SD15</td>
<td></td>
<td></td>
<td>

• Common information corresponding to the error codes (SD0) is stored here.
• The following four types of information are stored here:
1) Unit/module No.

<table>
<tr><th>Number</th><th>Contents</th></tr>
<tr><td>SD5</td><td>Slot No./PLC No.*1 *2</td></tr>
<tr><td>SD6</td><td>I/O No.</td></tr>
<tr><td>SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td><td>(Vacancy)</td></tr>
</table>

*1 : For a multiple CPU system, the slot number or PLC number is stored depending on the error that occurred.
Slot 0 in the multiple CPU system is the one on the slot on the right of the rightmost CPU module. (Refer to the corresponding error code for which number has been stored.)
PLC No. 1: 1, PLC No. 2: 2, PLC No. 3: 3, PLC No. 4: 4
*2 : If a fuse blown or I/O verify error occurred in the module loaded in the MELSECNET/H remote I/O station, the network number is stored into the upper 8 bits and the station number into the lower 8 bits.
Use the I/O No. to check the module where the fuse blown or I/O verify error occurred.

2) File name/Drive name

<table>
<tr><th>Number</th><th colspan="2">Contents</th></tr>
<tr><td>SD5</td><td colspan="2">Drive</td></tr>
<tr><td>SD6
SD7
SD8
SD9</td><td colspan="2">File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD10</td><td>Extension*3</td><td>2EH(.)</td></tr>
<tr><td>SD11</td><td colspan="2">(ASCII code: 3 characters)</td></tr>
<tr><td>SD12
SD13
SD14
SD15</td><td colspan="2">(Vacancy)</td></tr>
</table>

(e.g.) File name = ABCDEFGH. IJK

<table>
<tr><th>B15 to B8</th><th>B7 to B0</th></tr>
<tr><td>42H(B)</td><td>41H(A)</td></tr>
<tr><td>44H(D)</td><td>43H(C)</td></tr>
<tr><td>46H(F)</td><td>45H(E)</td></tr>
<tr><td>48H(H)</td><td>47H(G)</td></tr>
<tr><td>49H(I)</td><td>2DH(.)</td></tr>
<tr><td>4BH(K)</td><td>4AH(B)</td></tr>
</table>

</td>
<td>S (Error)</td>
<td>New</td>
<td>○+Rem</td>
</tr>
</table>

*3 For extensions, refer to REMARK at Appendix 66.

Special Register List (Continued)

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When set)</th><th>Corresponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr>
<td rowspan="2">SD5
SD6
SD7
SD8
SD9
SD10
SD11
SD12
SD13
SD14
SD15</td>
<td rowspan="2">Error common information</td>
<td rowspan="2">Error common information</td>
<td>
(Continued)
3) Time (value set)
<table>
<tr><th>Number</th><th>Contents</th></tr>
<tr><td>SD5</td><td>Time: In 1 µs units (0 to 999 µs)</td></tr>
<tr><td>SD6</td><td>Time: In 1 ms units (0 to 65535 ms)</td></tr>
<tr><td>SD7</td><td rowspan="9">(Vacancy)</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>
4) Program error location
<table>
<tr><th>Number</th><th colspan="2">Contents</th></tr>
<tr><td>SD5</td><td colspan="2" rowspan="4">File name
(ASCII code: 8 characters)</td></tr>
<tr><td>SD6</td></tr>
<tr><td>SD7</td></tr>
<tr><td>SD8</td></tr>
<tr><td>SD9</td><td>Extension *3</td><td>2EH(.)</td></tr>
<tr><td>SD10</td><td colspan="2">(ASCII code: 3 characters)</td></tr>
<tr><td>SD11</td><td colspan="2">Pattern *4</td></tr>
<tr><td>SD12</td><td colspan="2">Block No.</td></tr>
<tr><td>SD13</td><td colspan="2">Step No./Transition No.</td></tr>
<tr><td>SD14</td><td colspan="2">Sequence step No. (L)</td></tr>
<tr><td>SD15</td><td colspan="2">Sequence step No. (H)</td></tr>
</table>
*4: Contents of pattern data
<table>
<tr><td>15</td><td>14</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td>← (Bit No.)</td></tr>
<tr><td>0</td><td>0</td><td>to</td><td>0</td><td>0</td><td>*</td><td>*</td><td>*</td><td></td></tr>
</table>
(Not used)
SFC block designated (1)/not designated (0)
SFC step designated (1)/not designated (0)
SFC transition designated (1)/not designated (0)
</td>
<td>S (Error)</td><td>New</td><td>○+Rem</td>
</tr>
<tr>
<td>
5) Switch cause
<table>
<tr><th>Number</th><th>Contents</th></tr>
<tr><td>SD5</td><td>Switching factor (0: Auto/1: Manual)</td></tr>
<tr><td>SD6</td><td>Switching direction (0: Standby to control/1: Control to standby)</td></tr>
<tr><td>SD7</td><td>Tracking flag *6</td></tr>
<tr><td>SD8</td><td rowspan="8">(Vacancy)</td></tr>
<tr><td>SD9</td></tr>
<tr><td>SD10</td></tr>
<tr><td>SD11</td></tr>
<tr><td>SD12</td></tr>
<tr><td>SD13</td></tr>
<tr><td>SD14</td></tr>
<tr><td>SD15</td></tr>
</table>
*6: Tracking flag contents
Shows whether or not the tracking data is valid.
<table>
<tr><td>15</td><td>14</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td>← (Bit No.)</td></tr>
<tr><td>0</td><td>0</td><td>to</td><td>0</td><td>0</td><td>*</td><td>*</td><td>*</td><td></td></tr>
</table>
(Not used)
Initial work data Disable (0)/Enable (1)
System data (SFC active step data) Disable (0)/Enable (1)
Switching factor Disable (0)/Enable (1)
</td>
<td>S (Error)</td><td>New</td><td>Q4AR</td>
</tr>
</table>

*3 For extensions, refer to REMARK at Appendix 66.

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD16 SD17 SD18 SD19 SD20 SD21 SD22 SD23 SD24 SD25 SD26	Error individual information	Error individual information	• Individual information corresponding to error codes (SD0) is stored here. • The following six types of information are stored here:	S (Error)	New	○+Rem

1) File name/Drive name

Number	Contents	
SD16	Drive	
SD17	File name (ASCII code: 8 characters)	
SD18		
SD19		
SD20		
SD21	Extension*3	2EH(.)
SD22	(ASCII code: 3 characters)	
SD23	(Vacancy)	
SD24		
SD25		
SD26		

(e.g.) File name = ABCDEFGH. IJK

B15 to B8	B7 to B0
42H(B)	41H(A)
44H(D)	43H(C)
46H(F)	45H(E)
48H(H)	47H(G)
49H(I)	2DH(.)
4BH(K)	4AH(B)

2) Time (value actually measured)

Number	Contents
SD16	Time: In 1 μs units (0 to 999 μs)
SD17	Time: In 1 ms units (0 to 65535 ms)
SD18	(Vacancy)
SD19	
SD20	
SD21	
SD22	
SD23	
SD24	
SD25	
SD26	

3) Program error location

Number	Contents	
SD16	File name (ASCII code: 8 characters)	
SD17		
SD18		
SD19		
SD20	Extension*3	2EH(.)
SD21	(ASCII code: 3 characters)	
SD22	Pattern*4	
SD23	Block No.	
SD24	Step No./Transition No.	
SD25	Sequence step No. (L)	
SD26	Sequence step No. (H)	

*4: Contents of pattern data

15	14	to	4	3	2	1	0	← (Bit No.)
0	0	to	0	0	*	*	*	

(Not used)

- Bit 0: SFC block designated (1)/not designated (0)
- Bit 1: SFC step designated (1)/not designated (0)
- Bit 2: SFC transition designated (1)/not designated (0)

4) Parameter No.

Number	Contents
SD16	Parameter No.*5
SD17	(Vacancy)
SD18	
SD19	
SD20	
SD21	
SD22	
SD23	
SD24	
SD25	
SD26	

5) Annunciator number /CHK instruction malfunction number

Number	Contents
SD16	No.
SD17	(Vacancy)
SD18	
SD19	
SD20	
SD21	
SD22	
SD23	
SD24	
SD25	
SD26	

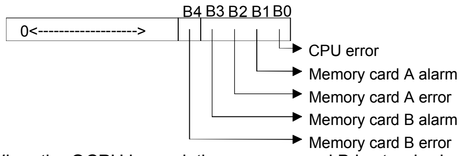
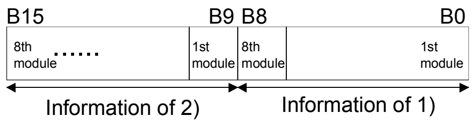
6) Intelligent function module parameter error (for QCPU only)

Number	Contents
SD16	Parameter No.*5
SD17	Error code of intelligent function module
SD18	(Vacancy)
SD19	
SD20	
SD21	
SD22	
SD23	
SD24	
SD25	
SD26	

*5: For details of the parameter numbers, refer to the user's manual of the CPU used.

*3 For extensions, refer to REMARK at Appendix 66.

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD50	Error reset	Error number that performs error reset	• Stores error number that performs error reset	U	New	○+Rem
SD51	Battery low latch	Bit pattern indicating where battery voltage drop occurred	• All corresponding bits go 1(ON) when battery voltage drops. • Subsequently, these remain 1(ON) even after battery voltage has been returned to normal.  • When the QCPU is used, the memory card B is standard and therefore the corresponding bits always remain OFF.	S (Error)	New	○
SD52	Battery low	Bit pattern indicating where battery voltage drop occurred	• Same configuration as SD51 above • Turns to 0 (OFF) when the battery voltage returns to normal thereafter. • When the QCPU is used, the memory card B is standard and therefore the corresponding bits always remain OFF.	S (Error)	New	
SD53	AC DOWN detection	Number of time for AC DOWN detection	• Every time the input voltage falls to or below 85% (AC power)/65% (DC power) of the rating during operation of the CPU module, the value is incremented by 1 and stored in BIN code.	S (Error)	D9005	○+Rem
SD54	MINI link errors	Error detection status	1) When any of X(n+0)/X(n+20), X(n+6)/X(n+26), X(n+7)/X(n+27) and X(n+8)/X(n+28) of the mounted MINI(-S3) turns ON, the bit of the corresponding station turns to 1 (ON). 2) Turns to 1 (ON) when communication between the mounted MINI (-S3) and CPU module cannot be made. 	S (Error)	D9004 format change	QnA
SD60	Number of module with blown fuse	Number of module with fuse blown	• Value stored here is the lowest station I/O number of the module with the blown fuse.	S (Error)	D9000	○+Rem
SD61	I/O module verification error number	I/O module verification error module number	• The lowest I/O number of the module where the I/O module verification error took place.	S (Error)	D9002	
SD62	Annunciator number	Annunciator number	• The first annunciator number (F number) to be detected is stored here.	S (Instruction execution)	D9009	○
SD63	Number of annunciators	Number of annunciators	• Stores the number of annunciators searched.	S (Instruction execution)	D9124	

REMARK

Extensions are shown below.

SD10	SD11		Extension name	File type
Higher 8 bits	Lower 8 bits	Higher 8 bits		
51_H	50_H	41_H	QPA	parameters
51_H	50_H	47_H	QPG	Sequence program/SFC program
51_H	43_H	44_H	QCD	Device comment
51_H	44_H	49_H	QDI	Device initial value
51_H	44_H	52_H	QDR	File register
51_H	44_H	53_H	QDS	Simulation data
51_H	44_H	$4C_H$	QDL	Local device
51_H	54_H	53_H	QTS	Sampling trace data (For QnA)
51_H	54_H	$4C_H$	QTL	Status latch data (For QnA)
51_H	54_H	50_H	QTP	Program trace data (For QnA)
51_H	54_H	52_H	QTR	SFC trace file
51_H	46_H	44_H	QFD	Trouble history data

Special Register List (Continued)

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th colspan="2">Explanation</th><th>Set by (When set)</th><th>Corresponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD64</td><td rowspan="16">Table of detected annunciator numbers</td><td rowspan="16">Annunciator detection numbers</td><td colspan="2" rowspan="16">When F goes ON due to OUT F or SET F , the F numbers which go progressively ON from SD64 through SD79 are registered.
The F numbers turned OFF by RST F are deleted from SD64 - SD79, and the F numbers stored after the deleted F numbers are shifted to the preceding registers. Execution of the LEDR instruction shifts the contents of SD64 to SD79 up by one. (This can also be done by using the INDICATOR RESET switch on the Q3A/Q4ACPU.) After 16 annunciators have been detected, detection of the 17th will not be stored from SD64 through SD79.
<table>
<tr><th></th><th></th><th>SET F50</th><th>SET F25</th><th>SET F99</th><th>RET F25</th><th>SET F15</th><th>SET F70</th><th>SET F65</th><th>SET F38</th><th>SET F110</th><th>SET F151</th><th>SET F210</th><th>LEDR</th><th></th></tr>
<tr><td>SD62</td><td>0</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>99</td><td>... (Detected No.)</td></tr>
<tr><td>SD63</td><td>0</td><td>1</td><td>2</td><td>3</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>8</td><td>... (No. of detected annunciators)</td></tr>
<tr><td>SD64</td><td>0</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>99</td><td rowspan="16">(Detected No.)</td></tr>
<tr><td>SD65</td><td>0</td><td>0</td><td>25</td><td>25</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>15</td></tr>
<tr><td>SD66</td><td>0</td><td>0</td><td>0</td><td>99</td><td>0</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>70</td></tr>
<tr><td>SD67</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>65</td></tr>
<tr><td>SD68</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>65</td><td>65</td><td>65</td><td>65</td><td>65</td><td>38</td></tr>
<tr><td>SD69</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>38</td><td>38</td><td>38</td><td>38</td><td>110</td></tr>
<tr><td>SD70</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>110</td><td>110</td><td>110</td><td>151</td></tr>
<tr><td>SD71</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>151</td><td>151</td><td>210</td></tr>
<tr><td>SD72</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>210</td><td>0</td></tr>
<tr><td>SD73</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD74</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD75</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD76</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD77</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD78</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>SD79</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table></td><td rowspan="16">S (Instruction execution)</td><td>D9125</td><td rowspan="27">○</td></tr>
<tr><td>SD65</td><td>D9126</td></tr>
<tr><td>SD66</td><td>D9127</td></tr>
<tr><td>SD67</td><td>D9128</td></tr>
<tr><td>SD68</td><td>D9129</td></tr>
<tr><td>SD69</td><td>D9130</td></tr>
<tr><td>SD70</td><td>D9131</td></tr>
<tr><td>SD71</td><td>D9132</td></tr>
<tr><td>SD72</td><td>New</td></tr>
<tr><td>SD73</td><td>New</td></tr>
<tr><td>SD74</td><td>New</td></tr>
<tr><td>SD75</td><td>New</td></tr>
<tr><td>SD76</td><td>New</td></tr>
<tr><td>SD77</td><td>New</td></tr>
<tr><td>SD78</td><td>New</td></tr>
<tr><td>SD79</td><td>New</td></tr>
<tr><td>SD80</td><td>CHK number</td><td>CHK number</td><td colspan="2">• Error codes detected by the CHK instruction are stored as BCD code.</td><td>S (Instruction execution)</td><td>New</td></tr>
<tr><td>SD90</td><td rowspan="10">Step transition watchdog timer setting value (Enabled only when SFC program exists)</td><td rowspan="10">F number for timer set value and time over error</td><td>Corresponds to SM90</td><td rowspan="10">• F number that will be turned ON when the step transition watchdog timer setting or watchdog timer time limit error occurs.
B15 B8 B7 B0
Set F number (0 to 255)
Set time limit of timer (1 to 255 sec.; (In 1 sec units))
• Turning ON any of SM90 to SM99 during an active step starts the timer, and if the transition condition next to the corresponding step is not met within the timer time limit, the set annunciator (F) turns ON.</td><td rowspan="10">U</td><td>D9108</td></tr>
<tr><td>SD91</td><td>Corresponds to SM91</td><td>D9109</td></tr>
<tr><td>SD92</td><td>Corresponds to SM92</td><td>D9110</td></tr>
<tr><td>SD93</td><td>Corresponds to SM93</td><td>D9111</td></tr>
<tr><td>SD94</td><td>Corresponds to SM94</td><td>D9112</td></tr>
<tr><td>SD95</td><td>Corresponds to SM95</td><td>D9113</td></tr>
<tr><td>SD96</td><td>Corresponds to SM96</td><td>D9114</td></tr>
<tr><td>SD97</td><td>Corresponds to SM97</td><td>New</td></tr>
<tr><td>SD98</td><td>Corresponds to SM98</td><td>New</td></tr>
<tr><td>SD99</td><td>Corresponds to SM99</td><td>New</td></tr>
<tr><td>SD105</td><td>xCH1 transmission speed setting (RS232)</td><td>Stores the preset transmission speed when GX Developer is used.</td><td colspan="2">K3: 300bps, K6: 600bps, K24: 2400bps, K48: 4800bps
K96: 9600bps, K192: 19.2kbps, K384: 38.4kbps
K576: 57.6kbps, K1152: 115.2kbps</td><td>S</td><td>New</td><td rowspan="2">QCPU remote</td></tr>
<tr><td>SD120</td><td>Error No. for external power supply OFF</td><td>Module No. which has external power supply error</td><td colspan="2">• Stores the lowest head No. of the module whose external power supply is OFF.
* Applicable only for Q-series modules (For future use)</td><td>S (Error occurrence)</td><td>New</td></tr>
</table>

Special Register List

(2) System information

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD200	Status of switch	Status of CPU switch	• The switch status of the remote I/O module is stored in the following format. B15 … B4 B3 … B0 Vacant (B15–B4), 1) (B3–B0) 1) Remote I/O module switch status Always 1: STOP	S (Always)	New	Remote
			• The CPU switch status is stored in the following format: B15 B12 B11 B8 B7 B4 B3 B0 3) (B15–B8), Vacant (B7–B6), 2) (B5–B4), 1) (B3–B0) 1) CPU switch status: 0: RUN, 1: STOP, 2: L.CLR 2) Memory card switch: Always OFF 3) DIP switch: B8 through BC correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON. BD through BF are vacant.	S (Every END processing)	New	QCPU
			• The CPU switch status is stored in the following format: B15 B12 B11 B8 B7 B4 B3 B0 3) (B15–B8), Vacant (B7–B6), 2) (B5–B4), 1) (B3–B0) 1) CPU switch key status: 0: RUN, 1: STOP, 2: L.CLR 2) Memory card switch: B4 corresponds to card A, and B5 corresponds to card B. 0: OFF, 1: ON 3) DIP switch: B8 through B12 correspond to SW1 through SW5 of system setting switch 1. B14 through B15 correspond to SW1 through SW2 of system setting switch 2. 0: OFF, 1: ON	S (Every END processing)	New	QnA
SD201	LED Status	Status of CPU-LED	• The following bit patterns are used to store the statuses of the LEDs on the CPU module: • 0 is off, 1 is on, and 2 is flicker. B15 B12 B11 B8 B7 B4 B3 B0 8) 7) 6) 5) 4) 3) 2) 1) 1): RUN 5): BOOT 2): ERROR 6): Vacant 3): USER 7): Vacant 4): BAT.ALARM 8): MODE Mode bit patter 0: OFF 1: Green 2: Orange	S (Status change)	New	QCPU
			• The following bit patterns are used to store the statuses of the LEDs on the CPU module: • 0 is off, 1 is on, and 2 is flicker. B15 B12 B11 B8 B7 B4 B3 B0 8) 7) 6) 5) 4) 3) 2) 1) 1): RUN 5): BOOT 2): ERROR 6): CARD A (memory card) 3): USER 7): CARD B (memory card) 4): BAT.ALARM 8): Vacant	S (Status change)	New	QnA
SD202	LED off	Bit pattern of LED that is turned off	• Stores bit patterns of LEDs turned off (Only USER and BOOT enabled) • Turned off at 1, not turned off at 0	U	New	

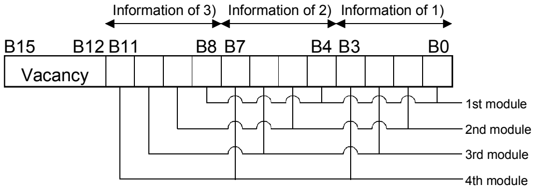
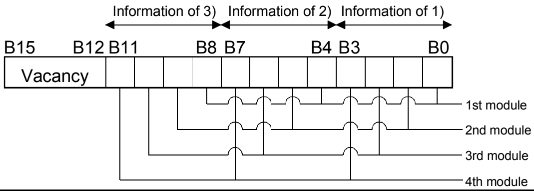
Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD203	Operating state of CPU	Operating state of CPU	• The operating status of the remote I/O module is stored in the following format. B15 … B4 B3 … B0 (1): B3–B0) 1) Remote I/O module operating status Always 2: STOP	S (Always)	New	Remote
			• The CPU operating state is stored as indicated in the following figure: B15 B12 B11 B8 B7 B4 B3 B0 (2): B7–B4, 1): B3–B0) 1) Operating status of CPU: 0: RUN, 1: STEP-RUN, 2: STOP, 3: PAUSE 2) STOP/PAUSE cause: 0: Key switch, 1: Remote contact, 2: Peripheral, computer link, or operation from some other remote source, 3: Internal program instruction, 4: Error Note: Priority is earliest first	S(Every END processing)	D9015 format change	○
SD206	Device test execution type	0: Test not yet executed 1: During X device test 2: During Y device test 3: During X/Y device test	• Set when the device test mode is executed on GX Developer.	S (Request)	New	Remote
SD207	LED display priority ranking	Priorities 1 to 4	• When error is generated, the LED display (flicker) is made according to the error number setting priorities. • The setting areas for priorities are as follows: B15–B12 / B11–B8 / B7–B4 / B3–B0 SD207: 4th priority / 3rd priority / 2nd priority / 1st priority SD208: 8th priority / 7th priority / 6th priority / 5th priority SD209: — / — / 10th priority / 9th priority Default value SD207 = H4321, SD208 = H8765, SD209 = H00A9 • No display is made if "0" is set. However, even if "0" has been set, information concerning CPU module operation stop (including parameter settings) errors will be indicated by the LEDs without conditions.	U	D9038	○
SD208		Priorities 5 to 8			D3039 format change	
SD209		Priorities 9 to 10			New	
SD210	Time data	Time data (year, month)	• The year (last two digits) and month are stored as BCD code at SD210 as shown below: B15 ••••• B12 B11 ••••• B8 B7 ••••• B4 B3 ••••• B0 Year (B15–B8), Month (B7–B0) (e.g.) July. 1993 H9307	S/U (Request)	D9025	○+Rem
SD211	Time data	Time data (day, hour)	• The day and hour are stored as BCD code at SD211 as shown below: B15 ••••• B12 B11 ••••• B8 B7 ••••• B4 B3 ••••• B0 Day (B15–B8), Hour (B7–B0) (e.g.) 31st, 10 o'clock H3110		D9026	
SD212	Time data	Time data (minute, second)	• The minutes and seconds (after the hour) are stored as BCD code at SD212 as shown below: B15 ••••• B12 B11 ••••• B8 B7 ••••• B4 B3 ••••• B0 Min. (B15–B8), Sec. (B7–B0) (e.g.) 35, 48 H3548		D9027	
SD213	Time data	Time data (Higher digits of year, day of week)	• The day of the week is stored as BCD code at SD213 as shown below: B15 ••••• B12 B11 ••••• B8 B7 ••••• B4 B3 ••••• B0 (e.g.) Friday H0005 Later 2 digits of year (0 to 99) (B15–B8) Day of week (B3–B0): 0 Sun, 1 Mon, 2 Tues, 3 Wed, 4 Thur, 5 Fri, 6 Sat		D9028	QCPU remote

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD213	Time data	Time data (, day of week)	• The day of the week is stored as BCD code at SD213 as shown below: B15 ····· B12 B11 ····· B8 B7 ····· B4 B3 ····· B0 (e.g.) Friday H0005 B15 to B4: "0" must be set B3 to B0: Day of week 0: Sun 1: Mon 2: Tues 3: Wed 4: Thur 5: Fri 6: Sat	S/U (Request)	D9028	QnA
SD220 SD221 SD222 SD223 SD224 SD225 SD226 SD227	LED display data	LED display data	• LED display ASCII data (16 characters) stored here. B15 to B8 / B7 to B0 SD220: 15th character from right / 16th character from right SD221: 13th character from right / 14th character from right SD222: 11th character from right / 12th character from right SD223: 9th character from right / 10th character from right SD224: 7th character from right / 8th character from right SD225: 5th character from right / 6th character from right SD226: 3rd character from right / 4th character from right SD226: 1st character from right / 2nd character from right	S (When changed)	New	○
SD240	Base mode	0: Automatic mode 1: Detail mode	The base mode is stored.	S (Initial)	New	QCPU remote
SD241	No. of extension bases	0: Main only 1 to 7: Number of extension bases	• Stores the maximum number of the extension bases being installed.	S (Initial)	New	
SD242	A/Q base differentiation	Base type differentiation 0: QA**B is installed (A mode) 1: Q**B is installed (Q mode)	B7 to B2 B1 B0 Fixed to 0 (upper bits) B0: Main base B1: 1st extension B2: 2nd extension to B7: 7th extension Fixed to 0 if no extension	S (Initial)	New	
SD243 SD244	No. of base slots	No. of base slots	B15 B12 B11 B8 B7 B4 B3 B0 SM243: Extension 3 / Extension 2 / Extension 1 / Main SM244: Extension 7 / Extension 6 / Extension 5 / Extension 4 • As shown above, each area stores the number of slots being installed.	S (Initial)	New	
SD250	Loaded maximum I/O	Loaded maximum I/O No.	• When SM250 goes from OFF to ON, the upper 2 digits of the final I/O number plus 1 of the modules loaded are stored as BIN values.	S (Request END)	New	○+Rem
SD251	Head I/O No. for replacement	Head I/O No. for replacement	• Stores the upper two digits of the head I/O number of an I/O module that is removed/replaced in the online status. (Default value: 100_H)	U	D9094	Q2A(S1) Q3A Q4A Q4AR
SD253	RS422 baud rate	RS422 baud rate	• Stores the baud rate of RS422. 0: 9600bps 1: 19.2kbps 2: 38.4kbps	S (When changed)	New	QnA
SD254	NET/10 information	Number of modules mounted	• Indicates the number of mounted NET/10 modules.	S (Initial)	New	○
SD255		Information of 1st module: I/O No.	• Indicates I/O No. of the 1st NET/10 module mounted.			
SD256		Information of 1st module: Network No.	• Indicates network No. of the 1st NET/10 module mounted.			
SD257		Information of 1st module: Group No.	• Indicates group No. of the 1st NET/10 module mounted.			
SD258		Information of 1st module: Station No.	• Indicates station No. of the 1st NET/10 module mounted.			
SD259		Information of 1st module: Standby information	• In the case of standby stations, the module number of the standby station is stored.(1 to 4)			
SD260 to SD264		Information of 2nd module	• Configuration is identical to that for the first module.			
SD265 to SD269		Information of 3rd module	• Configuration is identical to that for the first module.			
SD270 to SD274		Information of 4th module	• Configuration is identical to that for the first module.			

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD280	CC-Link error	Error detection status	1) When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON). 2) When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of corresponding station turns to 1 (ON). 3) Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.  The above module Nos. n are in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)	S (Error)	New	QCPU remote
			1) When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON). 2) When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of corresponding station turns to 1 (ON). 3) Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made. 	S (Error)	New	QnA
SD290	Device allocation (Same as parameter contents)	Number of points allocated for X	• Stores the number of points currently set for X devices	S (Initial)	New	○+Rem
SD291		Number of points allocated for Y	• Stores the number of points currently set for Y devices			
SD292		Number of points allocated for M	• Stores the number of points currently set for M devices			
SD293		Number of points allocated for L	• Stores the number of points currently set for L devices			○
SD294		Number of points allocated for B	• Stores the number of points currently set for B devices			○+Rem
SD295		Number of points allocated for F	• Stores the number of points currently set for F devices			○
SD296		Number of points allocated for SB	• Stores the number of points currently set for SB devices			○+Rem
SD297		Number of points allocated for V	• Stores the number of points currently set for V devices			○
SD298		Number of points allocated for S	• Stores the number of points currently set for S devices			
SD299		Number of points allocated for T	• Stores the number of points currently set for T devices			
SD300		Number of points allocated for ST	• Stores the number of points currently set for ST devices			
SD301		Number of points allocated for G	• Stores the number of points currently set for G devices			
SD302		Number of points allocated for D	• Stores the number of points currently set for D devices			○+Rem
SD303		Number of points allocated for W	• Stores the number of points currently set for W devices			
SD304		Number of points allocated for SW	• Stores the number of points currently set for SW devices			
SD315	Time reserved for communication processing	Time reserved for communication processing	Reserves the designated time for communication processing with GX Developer or other units The greater the value is designated, the shorter the response time for communication with other devices (GX Developer, serial communication units) becomes. Setting range: 1 to 100 ms If the designated value is out of the range above, it is assumed to no setting. The scan time becomes longer by the designated time.	U(END processing)	New	QCPU remote
SD340	Ethernet information	Number of modules mounted	• Number of modules mounted on Ethernet	S (Initial)	New	
SD341		Information of 1st module: I/O No.	• Indicates Ethernet I/O No. of the 1st module mounted			
SD342		Network No.	• Indicates Ethernet network No. of the 1st module mounted			
SD343		Group No.	• Indicates Ethernet group No. of the 1st module mounted			
SD344		Station No.	• Indicates Ethernet station No. of the 1st module mounted			
SD345 to SD346		Vacant	• Vacant (With QCPU, the Ethernet IP address of the 1st module is stored in buffer memory.)			
SD347		Vacant	• Vacant (With QCPU, the Ethernet error code of the 1st module is read with the ERRORRD instruction.)			

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD348 to SD354	Ethernet information	Information from 2nd module	• Configuration is identical to that for the first module.	S (Initial)	New	QCPU remote
SD355 to SD361		Information of 3rd module	• Configuration is identical to that for the first module.			
SD362 to SD368		Information of 4th module	• Configuration is identical to that for the first module.			
SD340	Ethernet information	Number of modules mounted	• Number of modules installed on Ethernet	S (Initial)	New	QnA
SD341		Information of 1st module: I/O No.	• Indicates Ethernet I/O No. of the 1st module mounted			
SD342		Information of 1st module: Network No.	• Indicates Ethernet network No. of the 1st module mounted			
SD343		Information of 1st module: Group No.	• Indicates Ethernet group No. of the 1st module mounted.			
SD344		Information of 1st module: Station No.	• Indicates Ethernet station No. of the 1st module mounted.			
SD345 to SD346		Information of 1st module: IP address	• Indicates Ethernet station No. of the 1st module mounted.			
SD347		Information of 1st module: Error Code	• Indicates error code of the 1st module mounted.			
SD348 to SD354		Information of 2nd module	• Configuration is identical to that for the first module.			
SD355 to SD361		Information of 3rd module	• Configuration is identical to that for the first module.			
SD362 to SD368		Information of 4th module	• Configuration is identical to that for the first module.			
SD380	Ethernet instruction reception status	Instruction reception status of 1st module	B15 to B8: 0 to 0 (Not used) B0: Reception status of channel 1 B1: Reception status of channel 2 B2: Reception status of channel 3 B3: Reception status of channel 4 B4: Reception status of channel 5 B5: Reception status of channel 6 B6: Reception status of channel 7 B7: Reception status of channel 8 ON : Received (channel in use) OFF: Not received (channel not in use)	S (Initial)	New	QnA
SD381		Instruction reception status of 2nd module	• Configuration is identical to that for the first module.			
SD382		Instruction reception status of 3rd module	• Configuration is identical to that for the first module.			
SD383		Instruction reception status of 4th module	• Configuration is identical to that for the first module.			
SD392	Software version	Internal system software version	• Stores the internal system software version in ASCII code. High byte | Low byte — Stored in low byte; Inconsistent value in high byte "41H" is stored when using version "A" Note: The internal system software version may differ from the version indicated by the version symbol printed on the case.	S (Initial)	D9060	
SD395	Multi CPU number	Multi CPU number	• In a multiple CPU system configuration, the CPU number of the host CPU is stored. PLC No. 1: 1, PLC No. 2: 2, PLC No. 3: 3, PLC No. 4: 4	S (Error)	New	QCPU function Ver. B

(3) System clocks/counters

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD412	1 second counter	Number of counts in 1-second units	• Incremented by 1 for each second execution after the CPU module is set to RUN • Count repeats from 0 to 32767 to -32768 to 0	S (Status change)	D9022	○
SD414	2n second clock setting	2n second clock units	• Stores value n of 2n second clock (Default is 30) • Setting can be made between 1 and 32767	U	New	
SD415	2nms clock setting	2nms clock units	• Stores value n of 2nms clock (Default is 30) • Setting can be made between 1 and 32767	U	New	QCPU
SD420	Scan counter	Number of counts in each scan	• Incremented by 1 for each scan execution after the CPU module is set to RUN.* • Count repeats from 0 to 32767 to -32768 to 0	S (Every END processing)	New	○
SD430	Low speed scan counter	Number of counts in each scan	• Incremented by 1 for each scan execution after the CPU module is set to RUN. • Count repeats from 0 to 32767 to -32768 to 0 • Used only for low speed execution type programs	S (Every END processing)	New	

*: Counting is not executed for scans by initial execution type program.

(4) Scan information

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD500	Execution program No.	Program No. in execution	• Program number of program currently being executed is stored as BIN value.	S (Status change)	New	○
SD510	Low speed program No.	Low speed program in execution	• Program number of low speed execution type program No. currently being executed is stored as BIN value. • Enabled only when SM510 is ON.	S (Every END processing)	New	
SD520	Current scan time	Current scan time (In 1 ms units)	• The current scan time is stored. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	D9017 format change	
SD521		Current scan time (In 100 μs units)	• The current scan time is stored. (In 100 μs units) • Range from 000 to 900 (Example) When the current scan time is 23.6ms, the following values are stored. D520=23 D521=600	S (Every END processing)	New	
SD522	Initial scan time	Initial scan time (In 1 ms units)	• Stores the scan time of an initial execution type program. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	New	
SD523		Initial scan time (In 100 μs units)	• Stores the scan time of an initial execution type program. (In 100 μs units) • Range from 000 to 900			
SD524	Minimum scan time	Minimum scan time (In 1 ms units)	• Stores the minimum value of the scan time. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	D9018 format change	
SD525	Minimum scan time	Minimum scan time (In 100 μs units)	• Stores the minimum value of the scan time. (In 100 μs units) • Range from 000 to 900	S (Every END processing)	New	
SD526	Maximum scan time	Maximum scan time (In 1 ms units)	• Stores the maximum value of the scan time except that of 1st scan. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	D9019 format change	
SD527		Maximum scan time (In 100 μs units)	• Stores the maximum value of the scan time except that of 1st scan. (In 100 μs units) • Range from 000 to 900		New	
SD528	Current scan time for low speed program	Current scan time (In 1 ms units)	• Stores the current scan time of a low speed program. (In 1 ms units)	S (Every END processing)	New	
SD529		Current scan time (In 100 μs units)	• Stores the current scan time of a low speed program. (In 100 μs units) • Range from 000 to 900			
SD532	Minimum scan time for low speed program	Minimum scan time (In 1 ms units)	• Stores the minimum value of the scan time of a low speed program. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	New	
SD533		Minimum scan time (In 100 μs units)	• Stores the minimum value of the scan time of a low speed program. (In 100 μs units) • Range from 000 to 900			
SD534	Maximum scan time for low speed program	Maximum scan time (In 1 ms units)	• Stores the maximum value of the scan time except that of 1st scan of a low speed program.. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	New	
SD535		Maximum scan time (In 100 μs units)	• Stores the maximum value of the scan time except that of 1st scan of a low speed program.. (In 100 μs units) • Range from 000 to 900			
SD540	END processing time	END processing time (In 1ms units)	• Stores the time from the end of a scan program to the start of the next scan. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	New	
SD541		END processing time (In 100 μs units)	• Stores the time from the end of a scan program to the start of the next scan. (In 100 μs units) • Range from 000 to 900			
SD542	Constant scan wait time	Constant scan wait time (In 1ms units)	• Stores the wait time for constant scan setting. (In 1 ms units) • Range from 0 to 65535	S (Every END processing)	New	
SD543		Constant scan wait time (In 100 μs units)	• Stores the wait time for constant scan setting. (In 100 μs units) • Range from 000 to 900			
SD544	Cumulative execution time for low speed programs	Cumulative execution time for low speed programs (In 1ms units)	• Stores the cumulative execution time of a low speed program. (In 1 ms units) • Range from 0 to 65535 • Cleared to 0 after the end of one low speed scan.	S (Every END processing)	New	
SD545		Cumulative execution time for low speed programs (In 100 μs units)	• Stores the cumulative execution time of a low speed program. (In 100 μs units) • Range from 000 to 900 • Cleared to 0 after the end of one low speed scan.			
SD546	Execution time for low speed programs	Execution time for low speed programs (In 1ms units)	• Stores the execution time of a low speed program during one scan. (In 1 ms units) • Range from 0 to 65535 • Stored every scan.	S (Every END processing)	New	
SD547		Execution time for low speed programs (In 100 μs units)	• Stores the execution time of a low speed program during one scan. (In 100 μs units) • Range from 000 to 900 • Stored every scan.			

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD548	Scan program execution time	Scan program execution time (In 1 ms units)	• Stores the execution time of a scan program during one scan. (In 1 ms units) • Range from 0 to 65535 • Stored every scan.	S(Every END processing)	New	○
SD549		Scan program execution time (In 100 µs units)	• Stores the execution time of a scan program during one scan. (In 100 µs units) • Range from 000 to 900 • Stored every scan.			
SD550	Service interval measurement module	Unit/module No.	• Sets I/O number for module that measures service interval	U	New	○+Rem
SD551	Service interval time	Module service interval (In 1 ms units)	• When SM551 is ON, stores service interval for module designated by SD550. (In 1 ms units) • Range from 0 to 65535	S (Request)	New	
SD552		Module service interval (In 100 µs units)	• When SM551 is ON, stores service interval for module designated by SD550. (in 100 µs units) • Range from 000 to 900			

Special Register List

(5) Memory card

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD600	Memory card A models	Memory card A models	• Indicates memory card A model installed B15 B8 B7 B4 B3 B0 0<---------->0 Drive 1 (RAM) type: 0: Does not exist, 1: SRAM Drive 2 (ROM) type: 0: Does not exist, (1: SRAM), 2: ATA FRASH, 3: FLASH ROM	S (Initial and card removal)	New	QCPU
			• Indicates memory card A model installed B15 B8 B7 B4 B3 B0 0<---------->0 Drive 1 (RAM) type: 0: Does not exist, 1: SRAM Drive 2 (ROM) type: 0: Does not exist, 2: EEPROM, 3: FLASH ROM	S (Initial and card removal)	New	QnA
SD602	Drive 1 (Standard RAM) capacity	Drive 1 capacity	• Drive 1 capacity is stored in 1 K byte units.	S (Initial and card removal)	New	QCPU
				S (Initial and card removal)	New	QnA
SD603	Drive 2 (Standard ROM) capacity	Drive 2 capacity	• Drive 2 capacity is stored in 1 K byte units.	S (Initial and card removal)	New	QCPU
				S (Initial and card removal)	New	QnA
SD604	Memory card A use conditions	Memory card A use conditions	• The use conditions for memory card (A) are stored as bit patterns. (In use when ON) • The significance of there bit patterns is indicated below: B0: Boot operation (QBT) B1: Parameters (QPA) B2: Device comments (QCD) B3: Device initial value (QDI) B4: File register R (QDR) B5: Trace (QTS) B6: B7: B8: — B9: CPU fault history (QFD) BA: SFC trace (QTS) BB: Local device (QDL) BC: BD: BE: BF:	S (Status change)	New	QCPU
			• The use conditions for memory card (A) are stored as bit patterns. (In use when ON) • The significance of there bit patterns is indicated below: B0: Boot operation (QBT) B1: Parameters (QPA) B2: Device comments (QCD) B3: Device initial value (QDI) B4: File R (QDR) B5: Sampling trace (QTS) B6: Status latch (QTL) B7: Program trace (QTP) B8: Simulation data (QDS) B9: CPU fault history (QFD) B10: SFC trace (QTS) B11: Local device (QDL) B12: B13: B14: B15:	S (Status change)	New	QnA
SD620	Memory card B models	Memory card B models	• Indicates memory card B model installed B15 B8 B7 B4 B3 B0 0<---------->0 Drive 3 (RAM) type: 0: Does not exist, 1: SRAM Drive 4 (ROM) type: 0: Does not exist, (1: SRAM), 2: E2PROM, 3: FLASH ROM Drive 4 is fixed for "3" since it incorporates a flash ROM.	S (Initial)	New	QCPU

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD620	Memory card B models	Memory card B models	• Indicates memory card B model installed B15 B8 B7 B4 B3 B0 0<----------->0 Drive 1 (RAM) type: 0: Does not exist / 1: SRAM Drive 2 (ROM) type: 0: Does not exist / 2: EEPROM / 3: FLASH ROM	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SD622	Drive 3 (Standard RAM) capacity	Drive 3 capacity	• Drive 3 capacity is stored in 1 K byte units. (Fixed for "64" since Drive 3 incorporates 64 K bytes RAM.)	S (Initial)	New	QCPU
			• Drive 3 capacity is stored in 1 K byte units.	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SD623	Drive 4 (Standard ROM) capacity	Drive 4 capacity	• Drive 4 capacity is stored in 1 K byte units.	S (Initial)	New	QCPU
			• Drive 4 capacity is stored in 1 K byte units.	S (Initial)	New	Q2A(S1) Q3A Q4A Q4AR
SD624	Drive 3/4 use conditions	Drive 3/4 use conditions	• The use conditions for drive 3/4 are stored as bit patterns. (In use when ON) • The significance of there bit patterns is indicated below: B0: Boot operation (QBT) B1: Parameters (QPA) B2: Device comments (QCD) B3: Device initial value (QDI) B4: File R (QDR) B5: Trace (QTS) B6: B7: B8: — B9: CPU fault history (QFD) B10: SFC trace (QTS) B11: Local device (QDL) B12: B13: B14: B15:	S (Status change)	New	QCPU
	Memory card B use conditions	Memory card B use conditions	• The use conditions for memory card B are stored as bit patterns. (ON when in use) • The significance of there bit patterns is indicated below: B0: Boot operation (QBT) B1: Parameters (QPA) B2: Device comments (QCD) B3: Device initial value (QDI) B4: File R (QDR) B5: Sampling trace (QTS) B6: Status latch (QTL) B7: Program trace (QTP) B8: Simulation data (QDS) B9: CPU fault history (QFD) B10: SFC trace (QTS) B11: Local device (QDL) B12: B13: B14: B15:	S (Status change)	New	Q2A(S1) Q3A Q4A Q4AR
SD640	File register drive	Drive number	• Stores drive number being used by file register.	S (Status change)	New	○
SD641 SD642 SD643 SD644 SD645 SD646	File register file name	File register file name	• Stores file register file name (with extension) selected at parameters or by use of QDRSET Instruction as ASCII code. B15 to B8 / B7 to B0 SD641: 2nd character / 1st character SD642: 4th character / 3rd character SD643: 6th character / 5th character SD644: 8th character / 7th character SD645: 1st character of extension / $2E_H$(.) SD646: 3rd character of extension / 2nd character of extension	S (Status change)	New	
SD647	File register capacity	File register capacity	• Stores the data capacity of the currently selected file register in 1 k word units.	S (Status change)	New	
SD648	File register block number	File register block number	• Stores the currently selected file register block number.	S (Status change)	D9035	
SD650	Comment drive	Comment drive number	• Stores the comment drive number selected at the parameters or by the QCDSET Instruction in ASCII code.	S (Status change)	New	

Special Register List (Continued)

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD651 SD652 SD653 SD654 SD655 SD656	Comment file name	Comment file name	• Stores file register file name (with extension) selected at the parameters or by use of QDRSET Instruction as ASCII code. B15 to B8 / B7 to B0 SD651: 2nd character / 1st character SD652: 4th character / 3rd character SD653: 6th character / 5th character SD654: 8th character / 7th character SD655: 1st character of extension / 2EH(.) SD656: 3rd character of extension / 2nd character of extension	S (Status change)	New	○
SD660	Boot operation designation file	Boot designation file drive number	• Stores the drive number where the boot designation file (*.QBT) is being stored.	S (Initial)	New	
SD661 SD662 SD663 SD664 SD665 SD666		File name of boot designation file	• Stores the file name of the boot designation file (*.QBT). B15 to B8 / B7 to B0 SD661: 2nd character / 1st character SD662: 4th character / 3rd character SD663: 6th character / 5th character SD664: 8th character / 7th character SD665: 1st character of extension / 2EH(.) SD666: 3rd character of extension / 2nd character of extension	S (Initial)	New	

(6) Instruction-related special registers-

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD705 SD706	Mask pattern	Mask pattern	• During block operations, turning SM705 ON makes it possible to use the mask pattern being stored at SD705 (or at SD705 and SD706 if double words are being used) to operate on all data in the block with the masked values.	U	New	○
SD714	Number of vacant communication request registration areas	0 to 32	• Stores the number of vacant blocks in the communications request area for remote terminal modules connected to the AJ71PT32-S3.	S (During execution)	D9081	QnA
SD715 SD716 SD717	IMASK instruction mask pattern	Mask pattern	• Patterns masked by use of the IMASK instruction are stored in the following manner: B15 / B11 / B0 SD715: I15 to I1 I0 SD716: I31 to I17 I16 SD717: I47 to I33 I32	S (During execution)	New	○
SD718 SD719	Accumulator	Accumulator	• For use as replacement for accumulators used in A-series programs.	S/U	New	
SD720	Program No. designation for PLOAD instruction	Program No. designation for PLOAD instruction	Stores the program number of the program to be loaded by the PLOAD instruction when designated. Range: 1 to 124	U	New	QCPU
SD730	Number of vacant CC-Link communication request registration areas	0 to 32	• Stores the number of vacant registration area for the request for communication with the intelligent device station connected to A(1S)J61QBT11.	S (During execution)	New	QnA
SD736	PKEY Input	PKEY Input	• Special register that temporarily stores keyboard data input by means of the PKEY instruction.	S (During execution)	New	○

Special Register List (Continued)

<table>
<tr><th>Number</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Set by (When set)</th><th>Corresponding ACPU D9□□□</th><th>Corresponding CPU</th></tr>
<tr><td>SD738
SD739
SD740
SD741
SD742
SD743
SD744
SD745
SD746
SD747
SD748
SD749
SD750
SD751
SD752
SD753
SD754
SD755
SD756
SD757
SD758
SD759
SD760
SD761
SD762
SD763
SD764
SD765
SD766
SD767
SD768
SD769</td><td>Message storage</td><td>Message storage</td><td>• Stores the message designated by the MSG instruction.
<table>
<tr><th></th><th>B15 to B8</th><th>B7 to B0</th></tr>
<tr><td>SD738</td><td>2nd character</td><td>1st character</td></tr>
<tr><td>SD739</td><td>4th character</td><td>3rd character</td></tr>
<tr><td>SD740</td><td>6th character</td><td>5th character</td></tr>
<tr><td>SD741</td><td>8th character</td><td>7th character</td></tr>
<tr><td>SD742</td><td>10th character</td><td>9th character</td></tr>
<tr><td>SD743</td><td>12th character</td><td>11th character</td></tr>
<tr><td>SD744</td><td>14th character</td><td>13th character</td></tr>
<tr><td>SD745</td><td>16th character</td><td>15th character</td></tr>
<tr><td>SD746</td><td>18th character</td><td>17th character</td></tr>
<tr><td>SD747</td><td>20th character</td><td>19th character</td></tr>
<tr><td>SD748</td><td>22th character</td><td>21th character</td></tr>
<tr><td>SD749</td><td>24th character</td><td>23th character</td></tr>
<tr><td>SD750</td><td>26th character</td><td>25th character</td></tr>
<tr><td>SD751</td><td>28th character</td><td>27th character</td></tr>
<tr><td>SD752</td><td>30th character</td><td>29th character</td></tr>
<tr><td>SD753</td><td>32th character</td><td>31th character</td></tr>
<tr><td>SD754</td><td>34th character</td><td>33th character</td></tr>
<tr><td>SD755</td><td>36th character</td><td>35th character</td></tr>
<tr><td>SD756</td><td>38th character</td><td>37th character</td></tr>
<tr><td>SD757</td><td>40th character</td><td>39th character</td></tr>
<tr><td>SD758</td><td>42th character</td><td>41th character</td></tr>
<tr><td>SD759</td><td>44th character</td><td>43th character</td></tr>
<tr><td>SD760</td><td>46th character</td><td>45th character</td></tr>
<tr><td>SD761</td><td>48th character</td><td>47th character</td></tr>
<tr><td>SD762</td><td>50th character</td><td>49th character</td></tr>
<tr><td>SD763</td><td>52th character</td><td>51th character</td></tr>
<tr><td>SD764</td><td>54th character</td><td>53th character</td></tr>
<tr><td>SD765</td><td>56th character</td><td>55th character</td></tr>
<tr><td>SD766</td><td>58th character</td><td>57th character</td></tr>
<tr><td>SD767</td><td>60th character</td><td>59th character</td></tr>
<tr><td>SD768</td><td>62th character</td><td>61th character</td></tr>
<tr><td>SD769</td><td>64th character</td><td>63th character</td></tr>
</table></td><td>S (During execution)</td><td>New</td><td>○</td></tr>
<tr><td>SD774
to
SD775</td><td>PID Limit setting</td><td>0: Limit set
1: Limit not set</td><td>Designate the limit for each PID loop as follows:
<table>
<tr><th></th><th>B15</th><th></th><th>B1</th><th>B0</th></tr>
<tr><td>SD774</td><td>Loop 16</td><td>to</td><td>Loop 2</td><td>Loop 1</td></tr>
<tr><td>SD775</td><td>Loop 32</td><td>to</td><td>Loop 18</td><td>Loop 17</td></tr>
</table></td><td>U</td><td>New</td><td>QCPU</td></tr>
<tr><td>SD780</td><td>Remaining No. of simultaneous execution of CC-Link dedicated instruction</td><td>0 to 32</td><td>• Stores the remaining number of simultaneous execution of the CC-Link dedicated instructions.</td><td>U</td><td>New</td><td>QnA</td></tr>
<tr><td>SD781
to
SD793</td><td>IMASK instruction mask pattern</td><td>Mask pattern</td><td>• Stores the mask patterns masked by the IMASK instruction as follows:
<table>
<tr><th></th><th>B15</th><th></th><th>B11</th><th>B0</th></tr>
<tr><td>SD781</td><td>I63</td><td>to</td><td>I59</td><td>I48</td></tr>
<tr><td>SD782</td><td>I79</td><td>to</td><td>I65</td><td>I64</td></tr>
<tr><td></td><td colspan="4">to</td></tr>
<tr><td>SD793</td><td>I255</td><td>to</td><td>I241</td><td>I240</td></tr>
</table></td><td>S (During execution)</td><td>New</td><td>QCPU</td></tr>
</table>

(7) A to Q/QnA conversion

ACPU special registers D9000 to D9255 correspond to Q/QnA special registers SD1000 to SD1255 after A to Q/QnA conversion.

These special registers are all set by the system, and cannot be turned ON or OFF by the user program.

To set data by the user program, correct the program for use of the Q/QnACPU special registers.

However, some of SD1200 to SD1255 (corresponding to D9200 to 9255 before conversion) can be set by the user program if they could be set by the user program before conversion.

For details on the ACPU special registers, refer to the user's manual for the corresponding CPU, and MELSECNET or MELSECNET/B Data Link System Reference Manuals.

REMARK

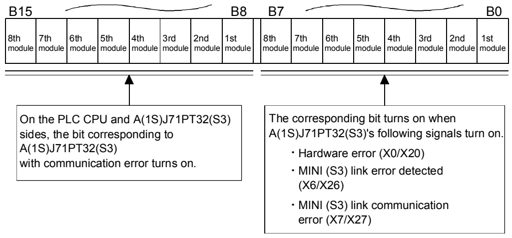
Supplemental explanation on "Special Register for Modification" column

1) For the device numbers for which a special register for modification is specified, modify it to the special register for Q/QnACPU.
2) For the device numbers for which $\boxed{-}$ is specified, special register after conversion can be used.
3) Device numbers for which $\boxtimes$ is specified do not function for Q/QnACPU.

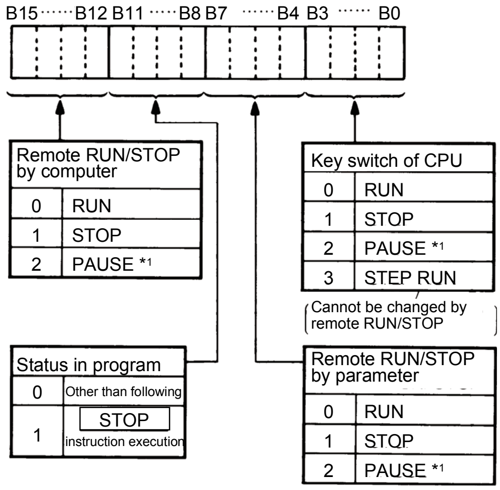
Special Register List

<table>
<tr><th>ACPU Special Register</th><th>Special Register after Conversion</th><th>Special Register for Modification</th><th>Name</th><th>Meaning</th><th>Explanation</th><th>Corresponding CPU</th></tr>
<tr><td>D9000</td><td>SD1000</td><td>—</td><td>Fuse blown</td><td>Number of module with blown fuse</td><td>When fuse blown modules are detected, the first I/O number of the lowest number of the detected modules is stored in hexadecimal. (Example: When fuses of Y50 to 6F output modules have blown, "50" is stored in hexadecimal) To monitor the number by peripheral devices, perform monitor operation given in hexadecimal.
(Cleared when all contents of SD1100 to SD1107 are reset to 0.)
• Fuse blow check is executed also to the output modules of remote I/O stations.</td><td rowspan="2">○</td></tr>
<tr><td>D9001</td><td>SD1001</td><td>—</td><td>Fuse blown</td><td>Number of module with blown fuse</td><td>• Stores the module numbers corresponding to setting switch numbers or base slot numbers when fuse blow occurred.
<table>
<tr><th colspan="2">A0J2 I/O module</th><th colspan="2">Extension base unit</th></tr>
<tr><th>Setting switch</th><th>Stored data</th><th>Slot No. of base unit</th><th>Stored data</th></tr>
<tr><td>0</td><td>1</td><td>0</td><td>5</td></tr>
<tr><td>1</td><td>2</td><td>1</td><td>6</td></tr>
<tr><td>2</td><td>3</td><td>2</td><td>7</td></tr>
<tr><td>3</td><td>4</td><td>3</td><td>8</td></tr>
<tr><td>4</td><td>5</td><td></td><td></td></tr>
<tr><td>5</td><td>6</td><td></td><td></td></tr>
<tr><td>6</td><td>7</td><td></td><td></td></tr>
<tr><td>7</td><td>8</td><td></td><td></td></tr>
</table>
• For the remote I/O station, the value of (module I/O No./10H) + 1 is stored.</td></tr>
</table>

Special Register List (Continued)

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Explanation	Corresponding CPU
D9002	SD1002	—	I/O module verification error	I/O module verification error module number	If I/O modules, of which data are different from data entered, are detected when the power is turned on, the first I/O number of the lowest number unit among the detected units is stored in hexadecimal. (Storing method is the same as that of SD1000.) To monitor the number by peripheral devices, perform monitor operation given in hexadecimal. (Cleared when all contents of SD1116 to SD1123 are reset to 0.) • I/O module verify check is executed also to the modules of remote I/O stations.	○
D9004	SD1004	—	NIMI link error	Error detection status	• Error status of the MINI(S3) link detected on loaded MELSECNET/MINI-S3 master module is stored. 	QnA
D9005	SD1005	—	AC DOWN counter	Number of time for AC DOWN	• When the AC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 20ms.(The value is stored in BIN code.) It is reset when power is switched from OFF to ON.	○
					• When the DC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 10ms. (The value is stored in BIN code.) It is reset when power is switched from OFF to ON.	QCPU
					• When the DC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 1ms. (The value is stored in BIN code.) It is reset when power is switched from OFF to ON.	QnA
D9008	SD1008	SD0	Self-diagnostic error	Self-diag-nosis error number	• When error is found as a result of self-diagnosis, error number is stored in BIN code.	○
D9009	SD1009	SD62	Annunciator detection	F number at which external failure has occurred	• When one of F0 to 2047 is turned on by $\boxed{\text{OUT F}}$ or $\boxed{\text{SET F}}$, the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code. • SD62 can be cleared by $\boxed{\text{RST F}}$ or $\boxed{\text{LEDR}}$ instruction. If another F number has been detected, clearing of SD62 causes the next number to be stored in SD62.	
					• When one of F0 to 2047 is turned on by $\boxed{\text{OUT F}}$ or $\boxed{\text{SET F}}$, the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code. • SD62 can be cleared by executing $\boxed{\text{RST F}}$ or $\boxed{\text{LEDR}}$ instruction or moving INDICATOR RESET switch on CPU module front to ON position. If another F number has been detected, clearing of SD62 causes the next number to be stored in SD62.	
D9010	SD1010		Error step	Step number at which operation error has occurred.	• When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Thereafter, each time operation error occurs, the contents of SD1010 are renewed.	
D9011	SD1011		Error step	Step number at which operation error has occurred.	• When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Since the step number is stored into SD1011 when SM1011 turns from OFF to ON, the data of SD1011 is not updated unless SM1011 is cleared by a user program.	
D9014	SD1014		I/O control mode	I/O control mode number	• The I/O control mode set is returned in any of the following numbers: 0: Both input and output in direct mode 1: Input in refresh mode, output in direct mode 3: Both input and output in refresh mode	

Special Register List (Continued)

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Explanation	Corresponding CPU
D9015	SD1015	SD203	Operatin g state of CPU	Operating state of CPU	• The operation status of CPU as shown below are stored in SD203.  *1: When the CPU module is in RUN mode and SM1040 is off, the CPU module remains in RUN mode if changed to PAUSE mode.	
D9016	SD1016		Program number	0: Main program (ROM) 1: Main program (RAM) 2: Subprogram 1 (RAM) 3: Subprogram 2 (RAM) 4: Subprogram 3 (RAM) 5: Subprogram 1 (ROM) 6: Subprogram 2 (ROM) 7: Subprogram 3 (ROM) 8: Main program (E2PROM) 9: Subprogram 1 (E2PROM) A: Subprogram 2 (E2PROM) B:Subprogram 3 (E2PROM)	• Indicates which sequence program is run presently. One value of 0 to B is stored in BIN code.	○
D9017	SD1017	SD520	Scan time	Minimum scan time (In 10 ms units)	• If scan time is smaller than the content of SD520, the value is newly stored at each END. Namely, the minimum value of scan time is stored into SD520 in BIN code.	
D9018	SD1018	SD524	Scan time	Scan time (In 10 ms units)	• At every END, the scan time is stored in BIN code and always rewritten.	
D9019	SD1019	SD526	Scan time	Maximum scan time (In 10 ms units)	• If scan time is larger than the content of SD526, the value is newly stored at each END. Namely, the maximum value of scan time is stored into SD526 in BIN code.	
D9020	SD1020		Constant scan	Constant scan time (User sets in 10 ms units)	• Sets the interval between consecutive program starts in multiples of 10 ms. 0 : No setting 1 to 200 : Set. Program is executed at intervals of (set value) × 10 ms.	
D9021	SD1021	—	Scan time	Scan time (In 1 ms units)	• At every END, the scan time is stored in BIN code and always rewritten.	

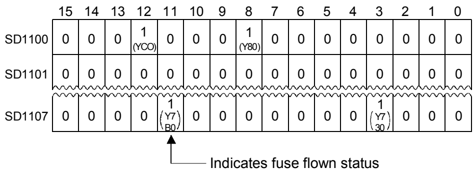
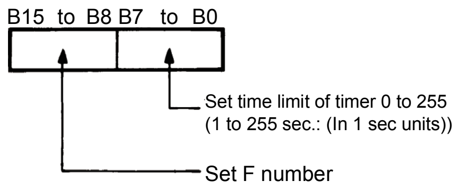
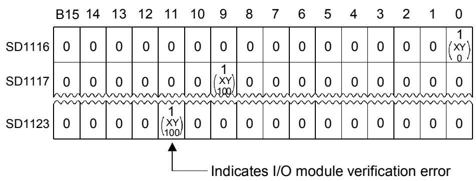
Special Register List (Continued)

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Explanation	Corresponding CPU
D9022	SD1022	SD412	1 second counter	Count in units of 1s.	• When the PLC CPU starts running, it starts counting 1 every second. • It starts counting up from 0 to 32767, then down to -32768 and then again up to 0. Counting repeats this routine.	
D9025	SD1025	—	Time data	Time data (year, month)	• Stores the year (2 lower digits) and month in BCD. B15 ⋯ B12 B11 ⋯ B8 B7 ⋯ B4 B3 ⋯ B0 Year / Month (e.g.) July. 1987 H8707	
D9026	SD1026	—	Time data	Time data (day, hour)	• Stores the day and hour in BCD. B15 ⋯ B12 B11 ⋯ B8 B7 ⋯ B4 B3 ⋯ B0 Day / Hour (e.g.) 31st, 10 o'clock H3110	
D9027	SD1027	—	Time data	Time data (minute, second)	• Stores the minute and second in BCD. B15 ⋯ B12 B11 ⋯ B8 B7 ⋯ B4 B3 ⋯ B0 Min. / Sec. (e.g.) 35, 48 H3548	
D9028	SD1028	—	Time data	Time data (, day of week)	• Stores the day of the week in BCD. B15 ⋯ B12 B11 ⋯ B8 B7 ⋯ B4 B3 ⋯ B0 (e.g.) Friday H0005 "0" must be set Day of week: 0 Sun, 1 Mon, 2 Tues, 3 Wed, 4 Thur, 5 Fri, 6 Sat	○
D9035	SD1035	SD648	Extension file register	Use block No.	• Stores the block No. of the extension file register being used in BCD code.	
D9036	SD1036		Extension file register for designation of device number	Device number when individual devices from extension file register are directly accessed	• Designate the device number for the extension file register for direct read and write in 2 words at SD1036 and SD1037 in BIN data. Use consecutive numbers beginning with R0 of block No. 1 to designate device numbers. Extension file register 0 to 16383: Block No.1 area 16384 to: Block No.2 areaa SD1036,SD1037 Device No. (BIN) to	
D9037	SD1037					
D9038	SD1038	SD207	LED display priority ranking	Priorities 1 to 4	• Sets priority of ERROR LEDs which illuminate (or flicker) to indicate errors with error code numbers. • Configuration of the priority setting areas is as shown below. B15 to B12 B11 to B8 B7 to B4 B3 to B0 SD207: 4th priority, 3rd priority, 2nd priority, 1st priority SD208: —, 7th priority, 6th priority, 5th priority • For details, refer to the applicable CPUs User's Manual and the ACPU Programming manual (Fundamentals) SH-3435 (version I or later).	
D9039	SD1039	SD208		Priorities 5 to 7		

Special Register List (Continued)

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Explanation	Corresponding CPU
D9044	SD1044		For sampling trance	Step or time during sampling trace	Turned on/off with a peripheral device. When STRA or STRAR is executed, the value stored in SD1044 is used as the sampling trace condition. At scanning 0 At time Time (In 10 ms units) The value is stored into SD1044 in BIN code.	○
D9049	SD1049		Work area for SFC	Block number of extension file register	• Stores the block number of the expansion file register which is used as the work area for the execution of a SFC program in a binary value. • Stores "0" if an empty area of 16K bytes or smaller, which cannot be expansion file register No. 1, is used or if SM320 is OFF.	○
D9050	SD1050		SFC program error number	Error code generated by SFC program	• Stores error code of errors occurred in the SFC program in BIN code. 0 : No error 80: SFC program parameter error 81: SFC code error 82: Number of steps of simultaneous execution exceeded 83: Block start error 84: SFC program operation error	○
D9051	SD1051		Error block	Block number where error occurred	• Stores the block number in which an error occurred in the SFC program in BIN code. In the case of error 83 the starting block number is stored.	○
D9052	SD1052		Error step	Step number where error occurred	• Stores the step number, where error code 84 occurred in an SFC program, in BIN code. Stores "0" when error code 80, 81 or 82 occurred. Stores the block stating step number when error code 83 occurs.	○
D9053	SD1053		Error transition	Transition condition number where error occurred	• Stores the transition condition number, where error code 84 occurred in an SFC program, in BIN code. Stores "0" when error code 80, 81, 82 or 82 occurred.	○
D9054	SD1054		Error sequence step	Sequence step number where error occurred	• Stores the sequence step number of transfer condition and operation output in which error 84 occurred in the SFC program in BIN code.	○
D9055	SD1055	SD812	Status latch	Status latch step	• Stores the step number when status latch is executed. • Stores the step number in a binary value if status latch is executed in a main sequence program. • Stores the block number and the step number if status latch is executed in a SFC program. Block No. (BIN) | Step No. (BIN) ← Higher 8 bits → ← Lower 8 bits →	○
D9060	SD1060	SD392	Software version	Software version of internal system	• Stores the software version of internal system in ASCII code. High byte | Low byte — Stored in low byte; Inconsistent value in high byte "41H" is stored when using version "A" Note: The software version of the initial system may differ from the version indicated by the version information printed on the rear of the case.	QnA
D9072	SD1072		PLC communication check	Serial communication module data check	• In the self-loopback test of the serial communication module, the serial communication module writes/reads data automatically to make communication checks.	○
D9081	SD1081	SD714	Number of empty blocks in communications request registration area	0 to 32	• Stores the number of empty blocks in the communication request registration area to the remote terminal module connected to the MELSECNET/MINI-S3 master unit, A2CCPU or A52GCPU.	QnA

Special Register List (Continued)

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Explanation	Corresponding CPU
D9085	SD1085		Register for setting time check value	1S to 65535S	• Sets the time check time of the data link instructions (ZNRD, ZNWR) for the MELSECNET/10. • Setting range : 1 s to 65535 s (1 to 65535) • Setting unit : S • Default value : 10 s (If 0 has been set, default 10 s is applied)	
D9090	SD1090		Head device No. in data area of microcomputer sub routine INPUT	See each micro computer package	• For details, refer to the manual of each microcomputer package.	
D9091	SD1091		Instruction error	Detailed error code	• Stores the detail code of cause of an instruction error.	
D9094	SD1094	SD251	Head I/O number of I/O module to be replaced	Head I/O number of I/O module to be replaced	• Stores the first two digits of the head I/O number of the I/O module, which will be dismounted/mounted online (with power on), in BIN value. Example) Input module X2F0 → H2F	
D9100	SD1100	—	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown	• Output module numbers (in units of 16 points), of which fuses have blown, are entered in bit pattern (Preset output module numbers when parameter setting has been performed.).  • Fuse blow check is executed also to the output modules of remote I/O stations. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)	
D9101	SD1101					
D9102	SD1102					
D9103	SD1103					
D9104	SD1104					
D9105	SD1105					
D9106	SD1106					
D9107	SD1107					○
D9108	SD1108	—	Step transfer monitoring timer setting	Time setting value and the F number at time out	• Set the set value of the step transition watchdog timer and the annunciator number (F number) that will turn on when the watchdog timer times out.  (By turning on any of MS1108 to SM1114, the monitoring timer starts. If the transfer condition following a step which corresponds to the timer is not established within set time, set annunciator (F) is tuned on.)	
D9109	SD1109					
D9110	SD1110					
D9111	SD1111					
D9112	SD1112					
D9113	SD1113					
D9114	SD1114					
D9116	SD1116	—	I/O module verification error	Bit pattern, in units of 16 points, indicating the modules with verification errors.	• When I/O modules, of which data are different from those entered at power-on, have been detected, the I/O module numbers (in units of 16 points) are entered in bit pattern. (Preset I/O module numbers set in parameters when parameter setting has been performed.).  • I/O module verify check is executed also to the modules of remote I/O stations. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)	
D9117	SD1117					
D9118	SD1118					
D9119	SD1119					
D9120	SD1120					
D9121	SD1121					
D9122	SD1122					
D9123	SD1123					

Special Register List (Continued)

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Explanation	Corresponding CPU
D9124	SD1124	SD63	Annunciator detection quantity	Annunciator detection quantity	• When one of F0 to 2047 is turned on by SET F, 1 is added to the contents of SD63. When RST F or LEDR instruction is executed, 1 is subtracted from the contents of SD63. (If the INDICATOR RESET switch is provided to the CPU module, pressing the switch can execute the same processing.) Quantity, which has been turned on by SET F is stored up to 8.	○
D9125	SD1125	SD64	Annunciator detection number	Annunciator detection numbers	• When any of F0 to 2047 is turned on by SET F, the annunciator numbers (F numbers) that are turned on in order are registered into D9125 to D9132. The F number turned off by RST F is erased from any of D9125 to D9132, and the F numbers stored after the erased F number are shifted to the preceding registers. By executing LEDR instruction, the contents of SD64 to SD71 are shifted upward by one. (If the INDICATOR RESET switch is provided to the CPU module, pressing the switch can execute the same processing.) When there are 8 annunciator detections, the 9th one is not stored into SD64 to SD71 even if detected.	○
D9126	SD1126	SD65				
D9127	SD1127	SD66				
D9128	SD1128	SD67				
D9129	SD1129	SD68				
D9130	SD1130	SD69				
D9131	SD1131	SD70				
D9132	SD1132	SD71				

		SET F50	SET F25	SET F99	RST F25	SET F15	SET F70	SET F65	SET F38	SET F110	SET F151	SET F210	LEDR
SD62	0	50	50	50	50	50	50	50	50	50	50	50	99
SD63	0	1	2	3	2	3	4	5	6	7	8	8	8
SD64	0	50	50	50	50	50	50	50	50	50	50	50	99
SD65	0	0	25	25	99	99	99	99	99	99	99	99	15
SD66	0	0	0	99	0	15	15	15	15	15	15	15	70
SD67	0	0	0	0	0	0	70	70	70	70	70	70	65
SD68	0	0	0	0	0	0	0	65	65	65	65	65	38
SD69	0	0	0	0	0	0	0	0	38	38	38	38	110
SD70	0	0	0	0	0	0	0	0	0	110	110	110	151
SD71	0	0	0	0	0	0	0	0	0	0	151	151	210

(8) Fuse blown module

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1300	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown 0: No blown fuse 1: Fuse blown present	• The numbers of output modules whose fuses have blown are input as a bit pattern (in units of 16 points). (If the module numbers are set by parameter, the parameter-set numbers are stored.) • Also detects blown fuse state at remote station output modules B15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 SD1300: 0 0 0 1(YC0) 0 0 0 1(Y80) 0 0 0 0 0 0 0 0 SD1301: 1(Y1F0) 0 0 0 0 1(Y1A) 0 0 0 0 0 0 0 0 0 0 SD1331: 0 0 0 0 1(Y1FB0) 0 0 0 0 0 0 0 1(Y1F30) 0 0 0 Indicates fuse flown status • Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.	S (Error)	D9100	○+Rem
SD1301					D9101	
SD1302					D9102	
SD1303					D9103	
SD1304					D9104	
SD1305					D9105	
SD1306					D9106	
SD1307					D9107	
SD1308					New	
SD1309 to SD1330					New to New	
SD1331					New	
SD1350 to SD1381	External power supply disconnected module (For future extension)	Bit pattern in units of 16 points, indicating the modules whose external power supply has been disconnected 0: External power supply disconnected 1: External power supply not disconnected	• The module number (in units of 16 points) whose external power supply has been disconnected is input as a bit pattern. (If the module numbers are set by parameter, the parameter-set numbers are used.) B15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 SD1350: 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 SD1351: 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 SD1381: 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 Indicates fuse flown status	S (Error)	New	QCPU remote

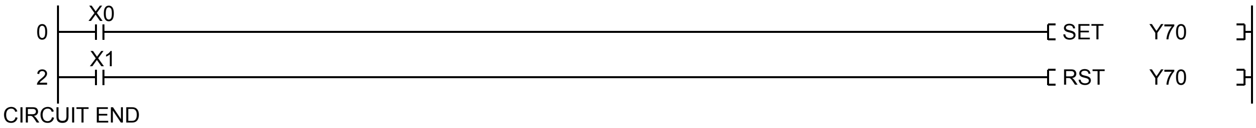
(9) I/O module verification

Number	Name	Meaning	Explanation	Set by (When set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1400	I/O module verification error	Bit pattern, in units of 16 points, indicating the modules with verification 0: No I/O verification errors 1: I/O verification error present	• When the I/O modules whose I/O module information differs from that registered at power on are detected, the numbers of those I/O modules (in units of 16 points) are entered in bit pattern. (If the I/O numbers are set by parameter, the parameter-set numbers are stored.) • Also detects I/O module information of remote station B15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 D9116: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1(XY0) D9117: 0 0 0 0 0 0 1(XY190) 0 0 0 0 0 0 0 0 0 D9123: 0 1(XY1FE0) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 Indicates I/O module verification error • Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.	S (Error)	D9116	○+Rem
SD1401					D9117	
SD1402					D9118	
SD1403					D9119	
SD1404					D9120	
SD1405					D9121	
SD1406					D9122	
SD1407					D9123	
SD1408					New	
SD1409 to SD1430					New to New	
SD1431					New	

Appendix 5 Application Program Examples

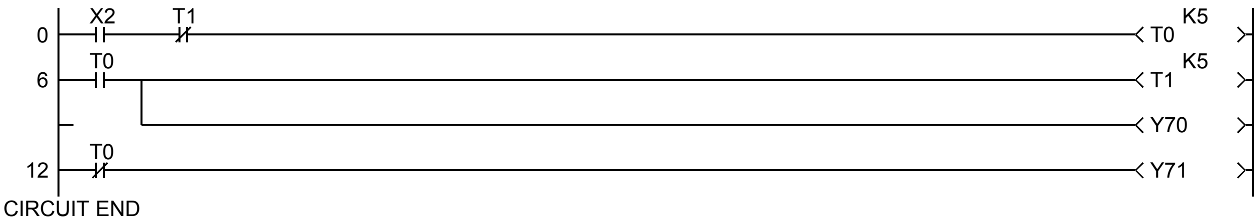
Appendix 5.1 Flip-flop ladder

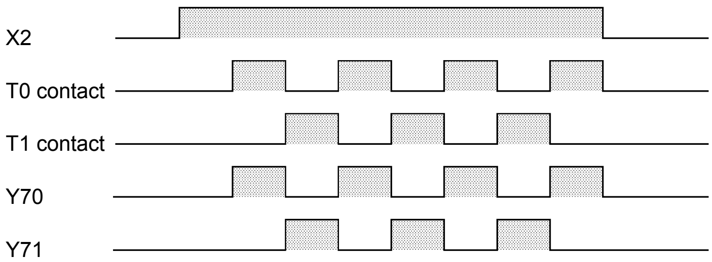
(1) Y70 turns ON when X00 is turned ON, and turns OFF when X01 is turned ON.



(2) When X02 is turned ON, Y71 turns OFF if Y70 is ON, and turns on if Y70 is OFF.
This flip-flop operation is repeated.

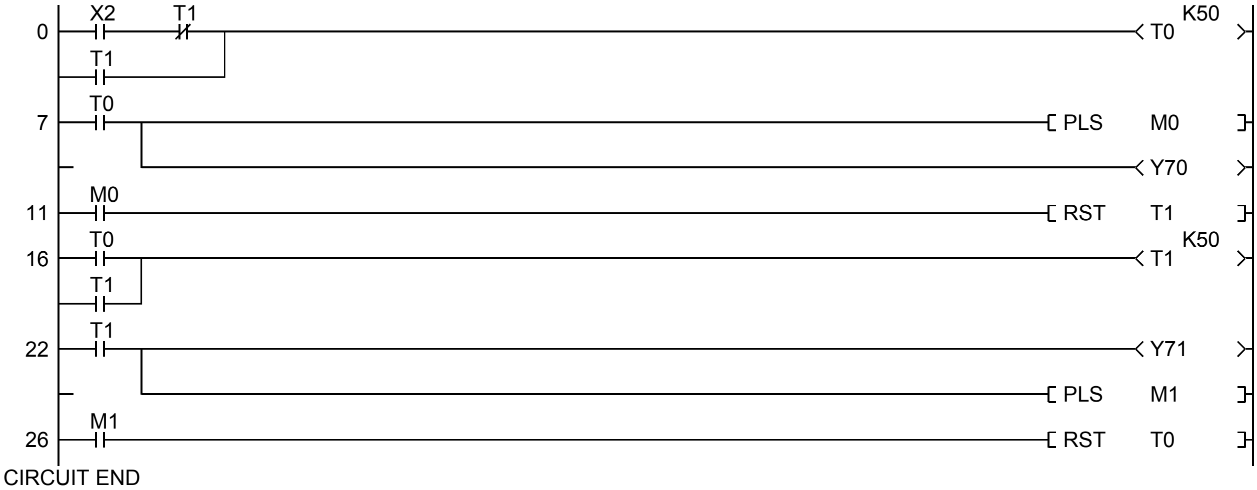
Path name	A:\SCHOOL
Project name	QA-16
Program name	MAIN

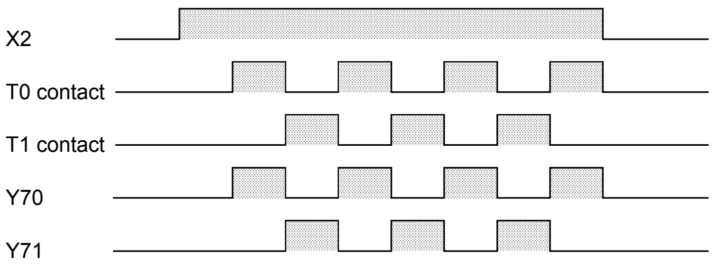




(3) The flip-flop operation starts when X2 is turn-ed on. In this operation, Y70 turns on if the timer T0 is on, and Y71 turns on if the timer T1 is on. (Cycle: 10 s)

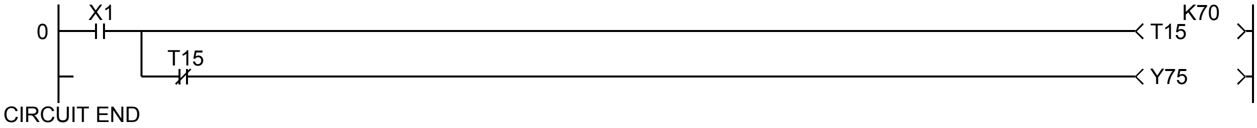
Path name	A:\SCHOOL
Project name	QA-17
Program name	MAIN

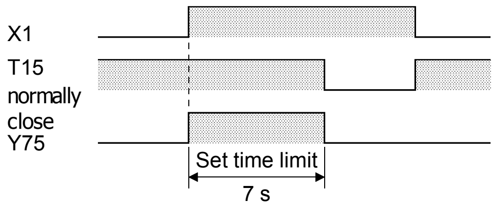




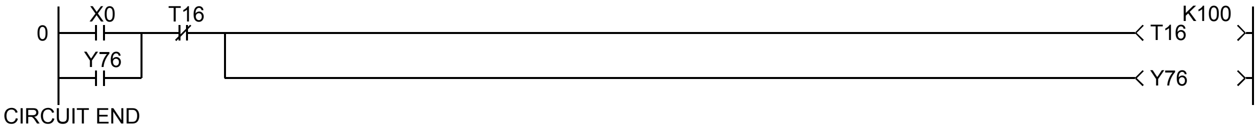
Appendix 5.2 One shot ladder

(1) Output starts and continues for a certain time after input X1 is turned on.
(Time for the input being on must be longer than the set time limit.)

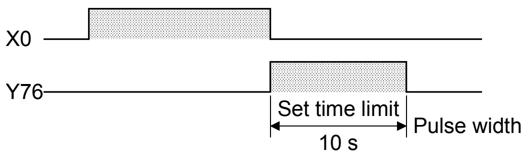




(2) When the input X0 is turned on momentarily, Y76 turns on for a certain time.

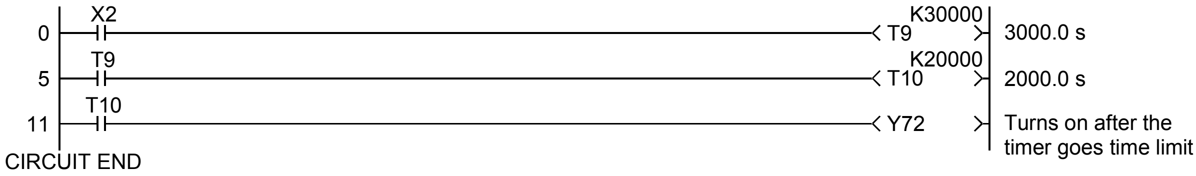


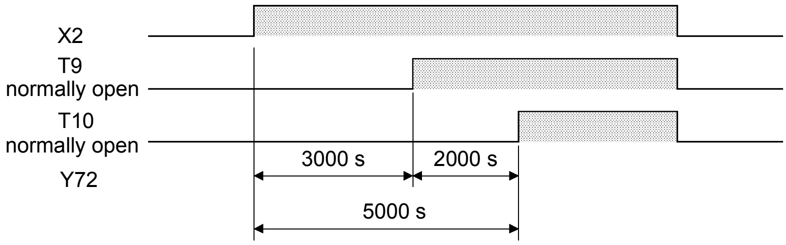
(3) Output starts and continues for a certain time when the input X0 is switched from on to off.



Appendix 5.3 Long time timer

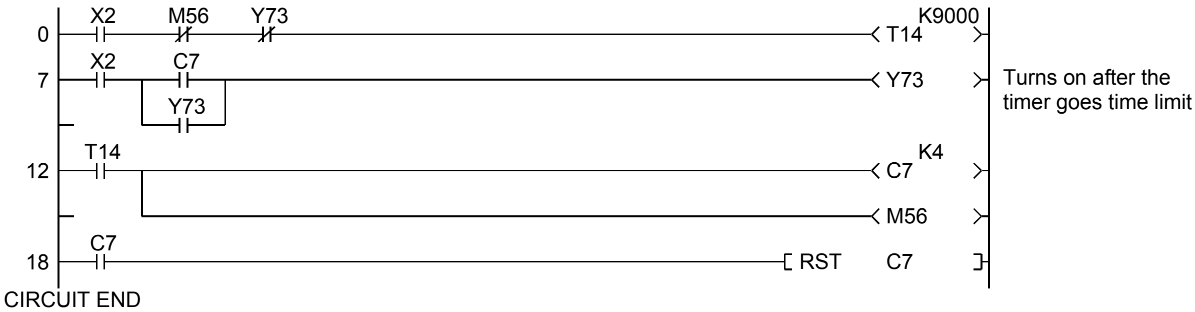
(1) Necessary time is obtained by connecting timers in serial.

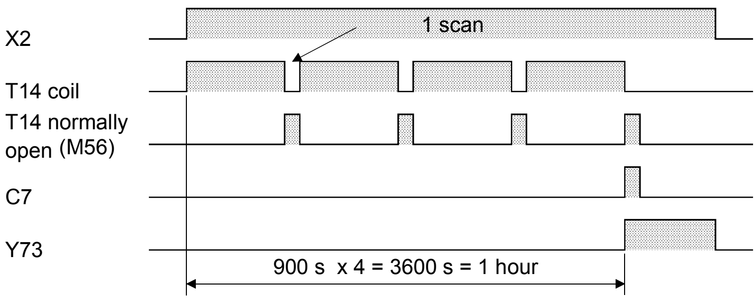




(2) Necessary time is obtained by using timers and counters.
Timer time limit X Counter's set value = Long time timer (note that accuracy of timers are accumulated.)

Path name	A:\SCHOOL
Project name	QA-18
Program name	MAIN



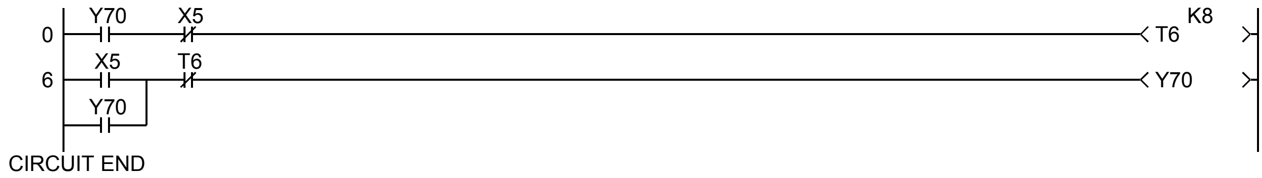


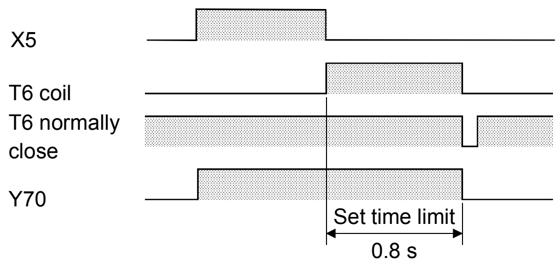
(Note) Sufficient time is obtained with the counter C7, which counts the number of time-outs of the timer T14.
With M56, T14 is reset after it goes time-out. With C7, the output Y73 is self-energized while count up is in progress. With Y73, T14 is reset and following time-limit actions are stopped.

Appendix 5.4 Off delay timer

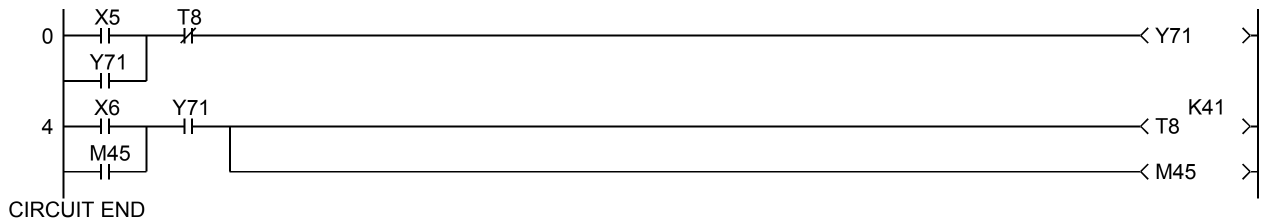
MELSEC-Q PLCs do not provide off delay timers. Make it as follows.

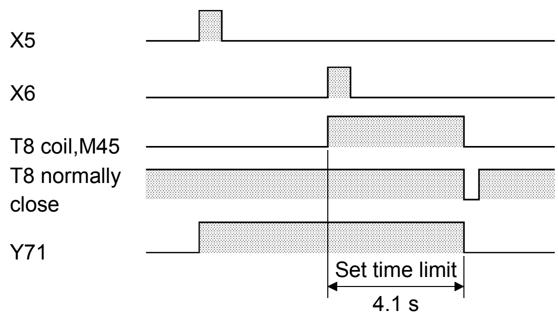
(1) T6 starts operating at the timing when X5 is turned OFF.





(2) Turn on X5 momentarily.
After that, the timer T8 starts operating at the timing when X6 is momentarily turned ON.



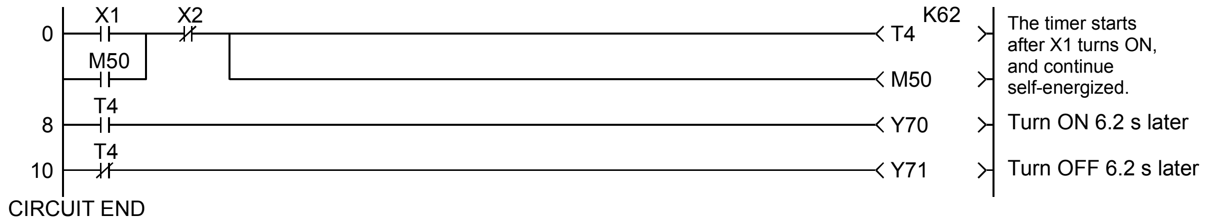


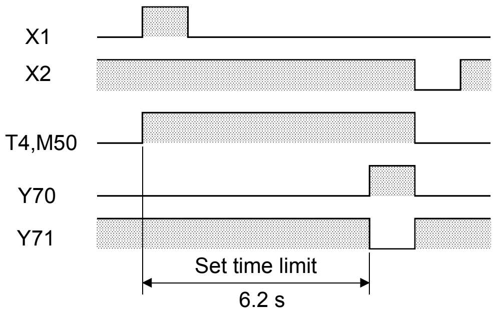
(Note) The above ladder behaves as an off delay ladder by momentarily turning ON input X5 and X6.
M45 is equivalent to a momentary contact of T8.

Appendix 5.5 On delay timer (momentary input)

PLC's timers operate with the on delay system, which allows easy continuous inputs but requires relays M for momentary inputs.

Path name	A:\SCHOOL
Project name	QA-19
Program name	MAIN

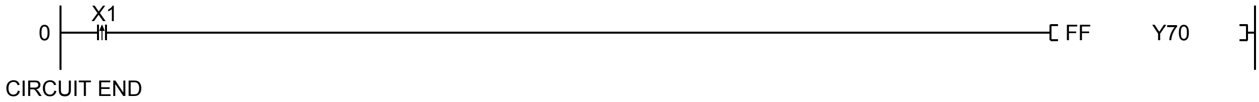




(Note) The above ladder behaves as an on delay ladder by momentarily turning on input X1 and X2.

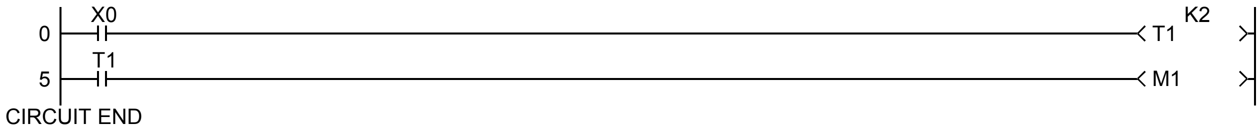
Appendix 5.6 ON-OFF repeat ladder

In an ON-OFF repeat ladder, Y70 turns ON when X0 is turned ON, and turns OFF when X0 is turned ON again.



Appendix 5.7 Preventing chattering input

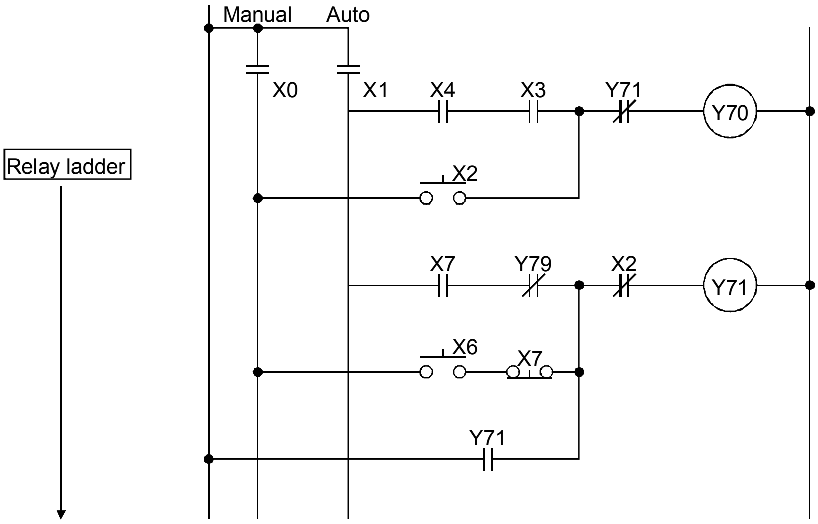
The timer is set so that it starts output when the input keeps being on for 0.2 s.



M1 turns ON when X0 keeps being ON for 0.2 s or longer. Therefore, this is used instead of X0 when writing a program.

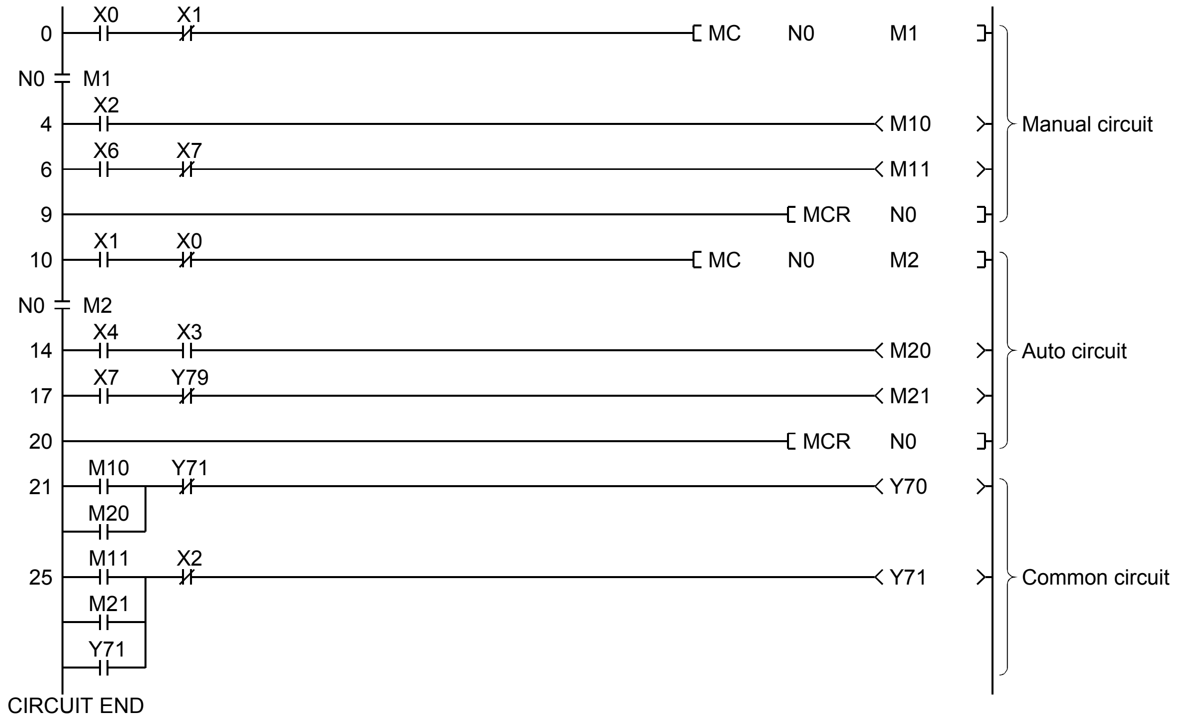
Appendix 5.8 Ladders with a common line

The following ladder cannot be operated by PLC programs. To make such ladders controllable, use master control instructions (MC, MCR) in a program.



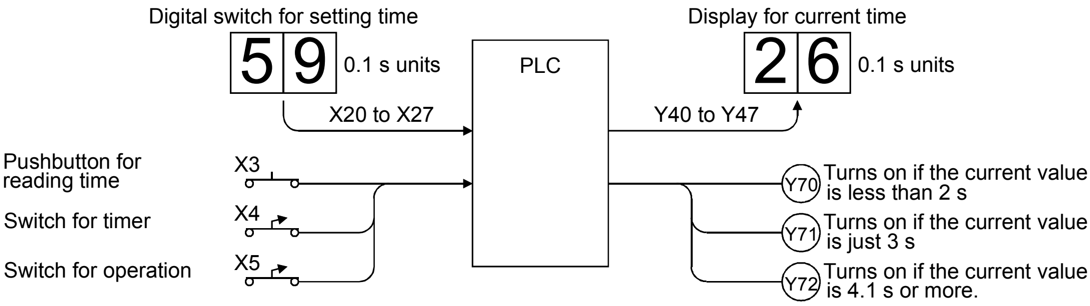
Path name	A:\SCHOOL
Project name	QA-1
Program name	MAIN

A sequence program with master controls

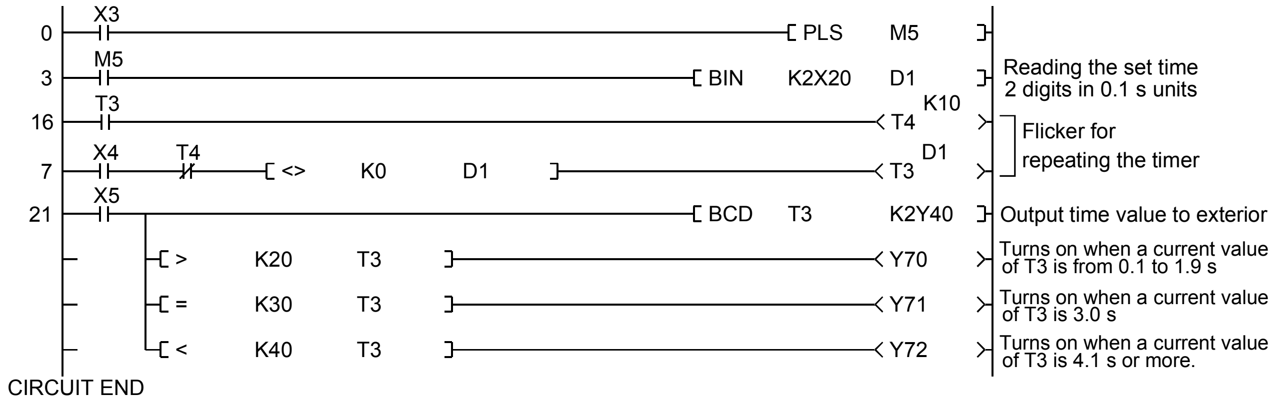


Appendix 5.9 Time control program

With the time control program, time value is set in the two digits of a digital switch, according to which the outputs Y70 to Y72 turn on after the set time limit has elapsed. Currently elapsed time is displayed on Y40 to Y47.
This operation is repeated.



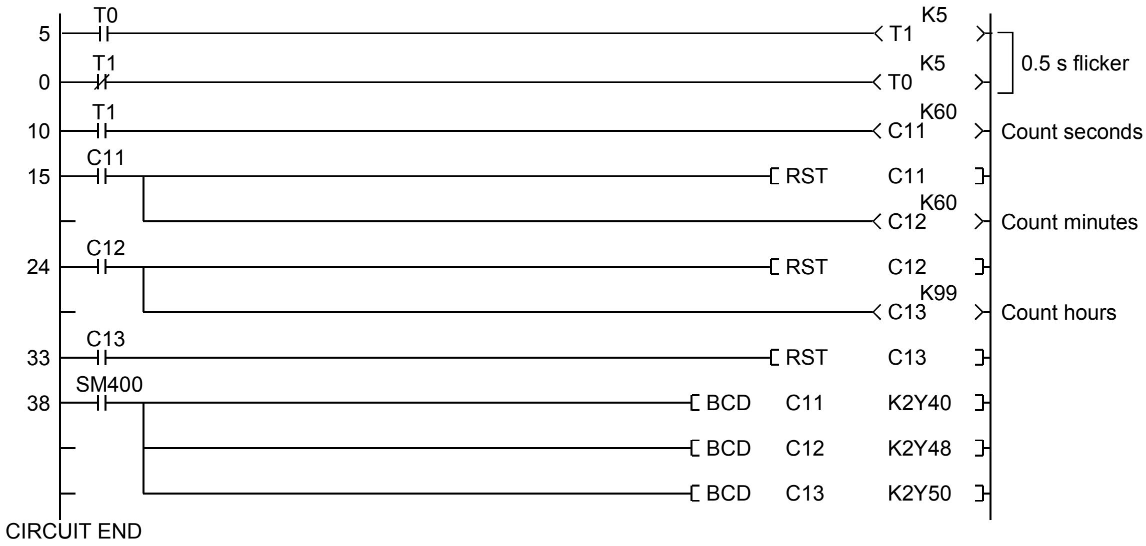
Path name	A:\SCHOOL
Project name	QA-2
Program name	MAIN

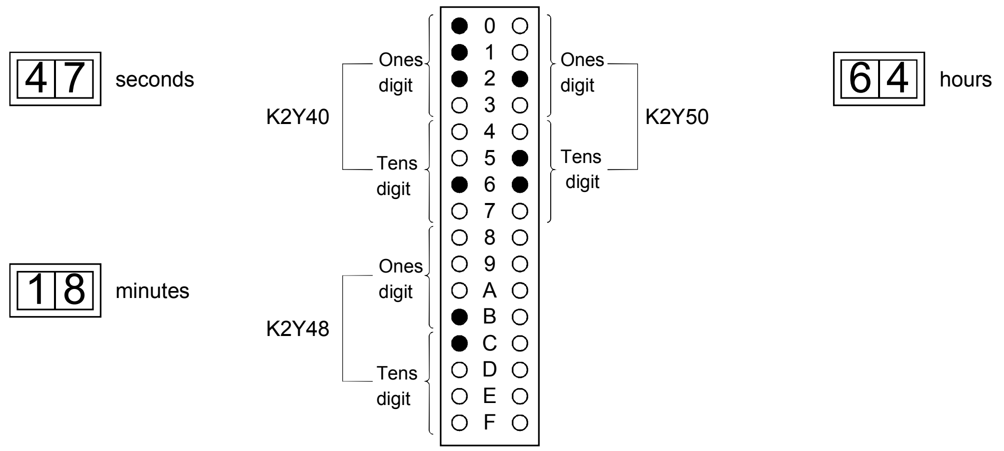


Appendix 5.10 Clock ladder

With a clock ladder, the clock data such as hour, minute and second is output to a digital display.

Path name	A:\SCHOOL
Project name	QA-3
Program name	MAIN





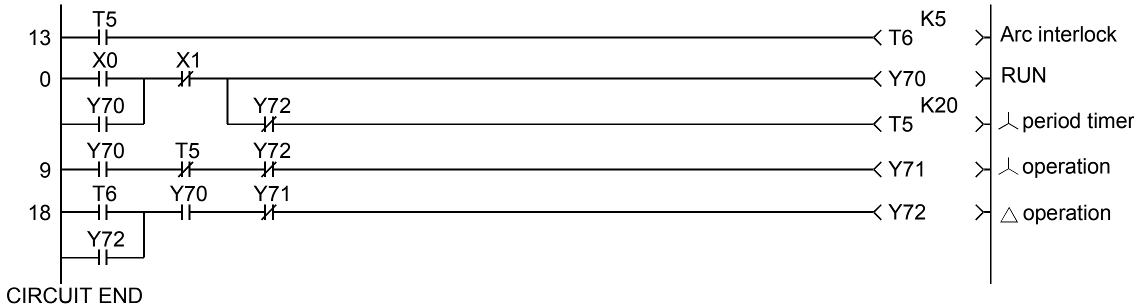
LED of output module

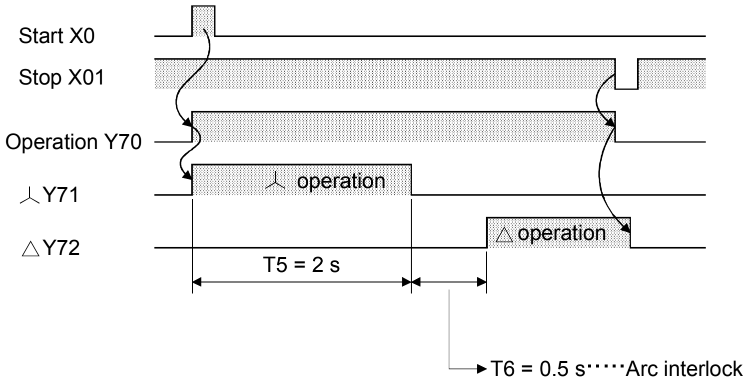
ON	ON				ON						ON	ON					ON				ON	ON	ON
Y57	Y56	Y55	Y54	Y53	Y52	Y51	Y50	Y4F	Y4E	Y4D	Y4C	Y4B	Y4A	Y49	Y48	Y47	Y46	Y45	Y44	Y43	Y42	Y41	Y40
8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
Tens digit				Ones digit				Tens digit				Ones digit				Tens digit				Ones digit			
K2Y50 hour, Output								K2Y48 minute, Output								K2Y40 second, Output							

Appendix 5.11 Starting ⅄ - △operation of electrical machinery

By turning the start switch on, machinery starts ⅄ operation. After ⅄ operation time has elapsed, the machinery enters an arc interlock state and then △ operation mode.

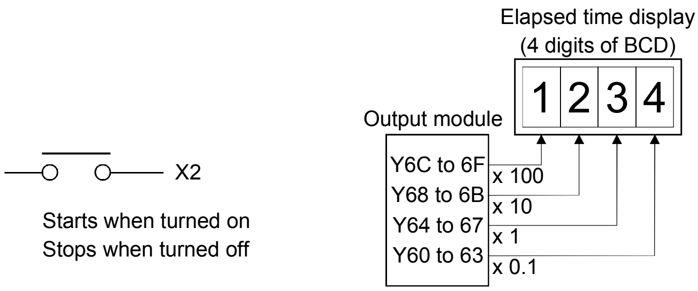
Path name	A:\SCHOOL
Project name	QA-20
Program name	MAIN

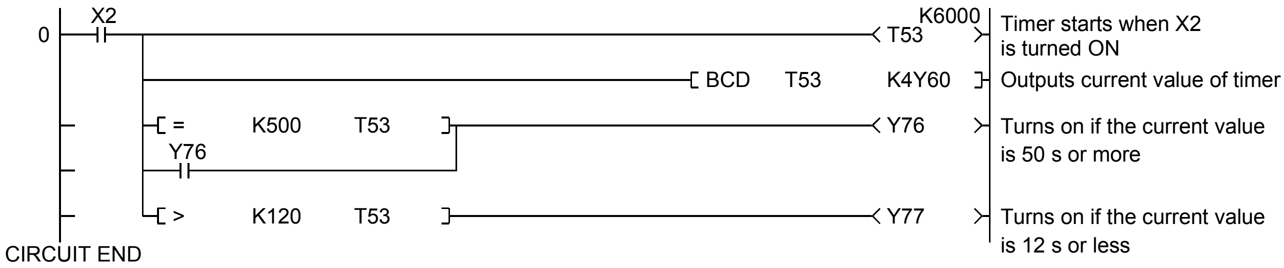


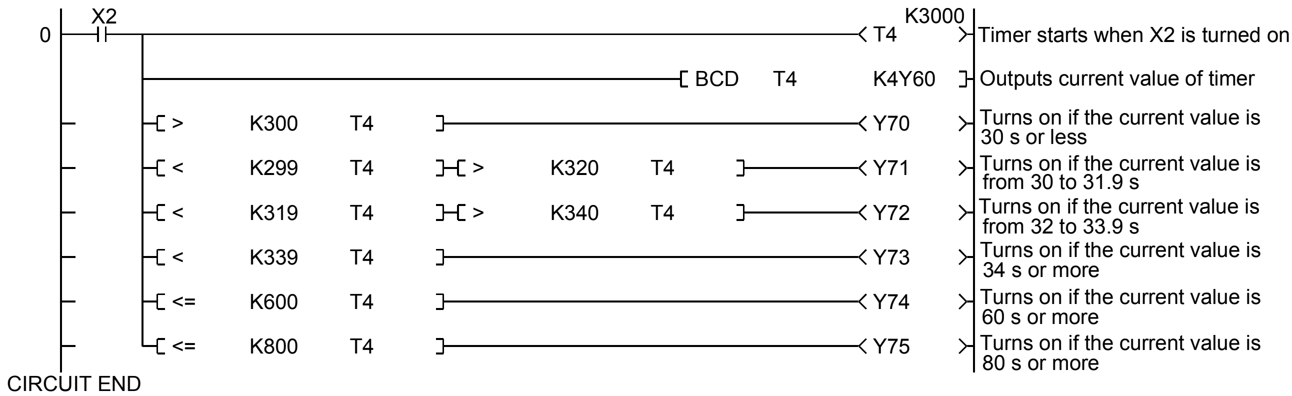


Appendix 5.12 Displaying elapsed time and outputting before time limit

With the following ladder, time elapsed in the timer is displayed on the LED, and output is performed before the set time limit has been reached. This system can also be applied to counters.





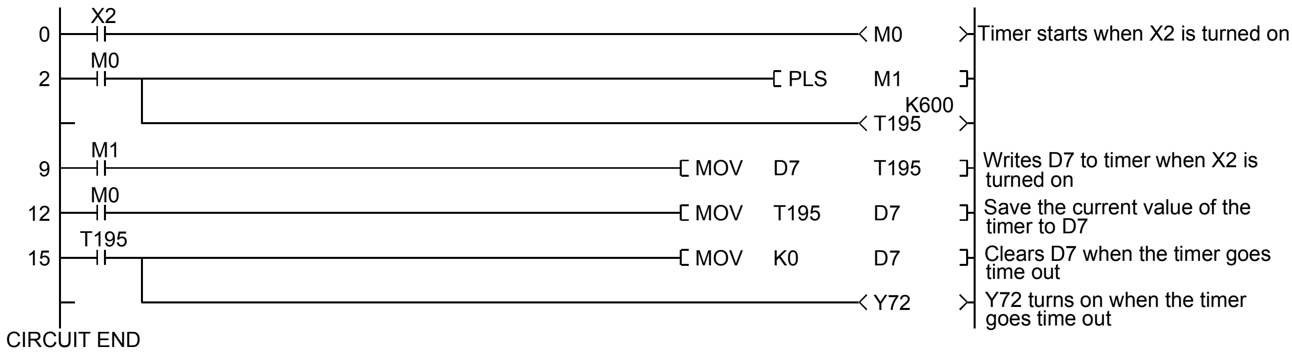


Appendix 5.13 Retentive timer

Input X2 switches between on and off continuously. The time of X2 being on is accumulated and Y72 turns on according to this accumulated value n.

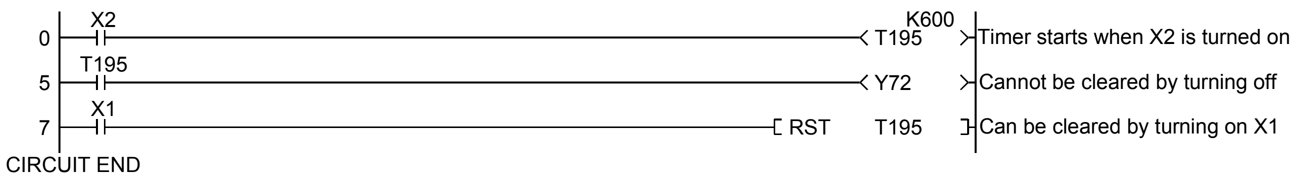
(1) When using a ladder that accumulates value without a retentive timer.

Path name	A:\SCHOOL
Project name	QA-21
Program name	MAIN



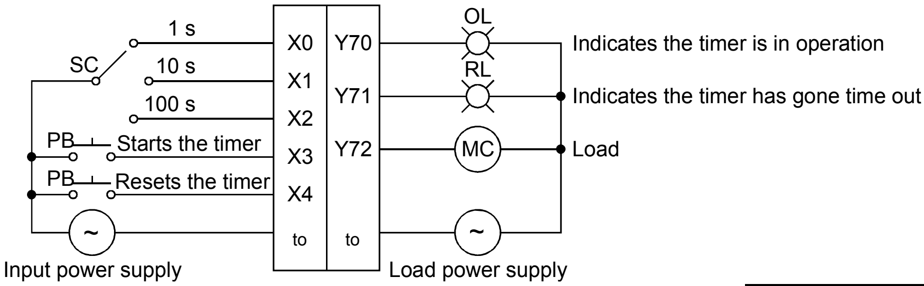
(2) When retentive timers are allocated in the device settings of PLC parameters.
Retentive timer (ST): 224 points (ST0 to ST223)

Path name	A:\SCHOOL
Project name	QA-8
Program name	MAIN

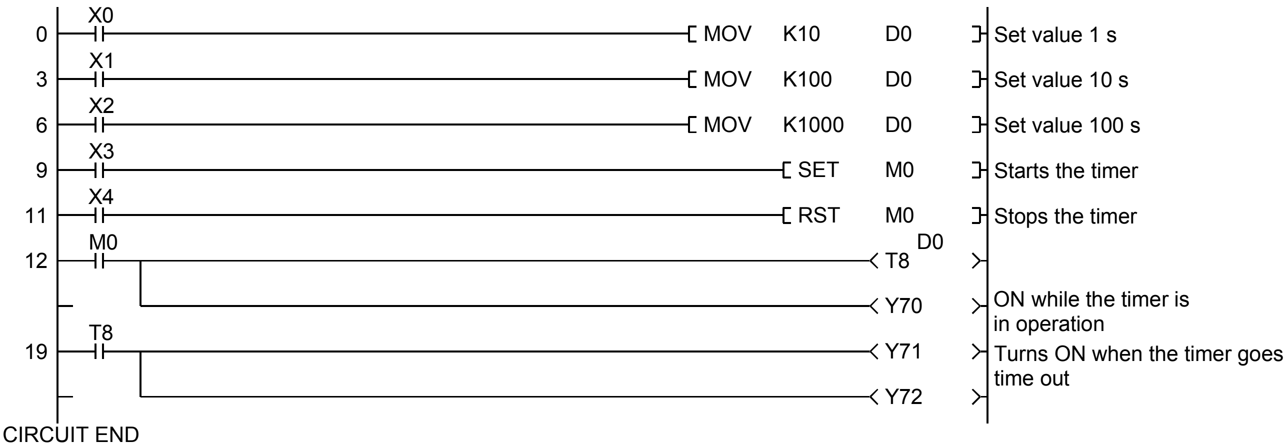


Appendix 5.14 Switching timer set value externally

(1) With an external switch, a value to be set in one timer can be selected from three patterns; 1 s, 10 s, and 100 s
A timer is activated and reset by a pushbutton switch.

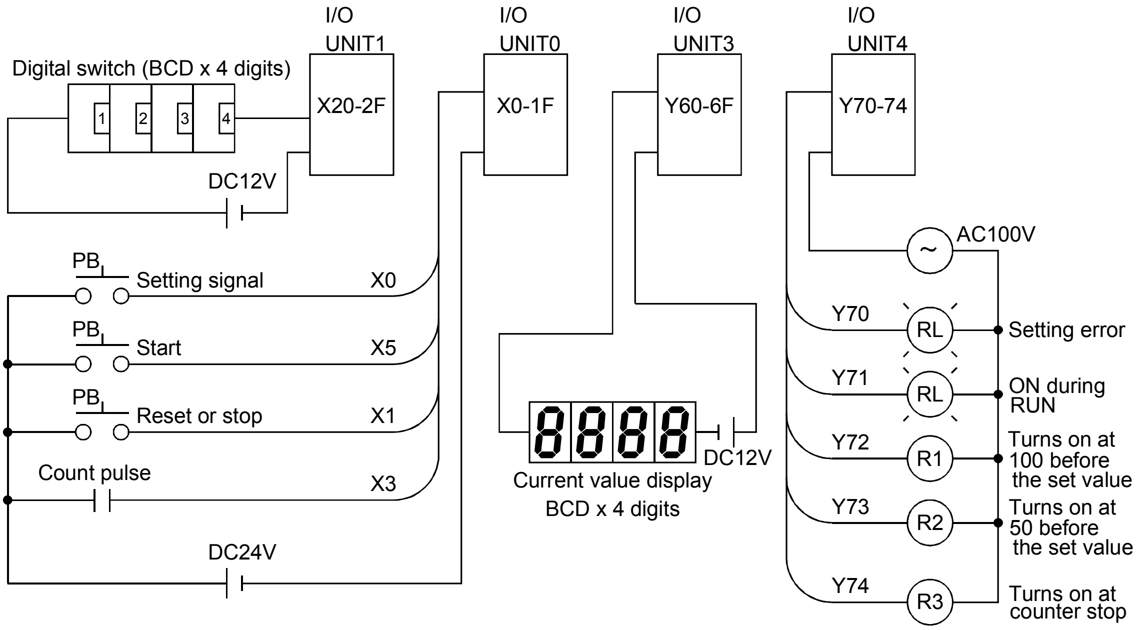


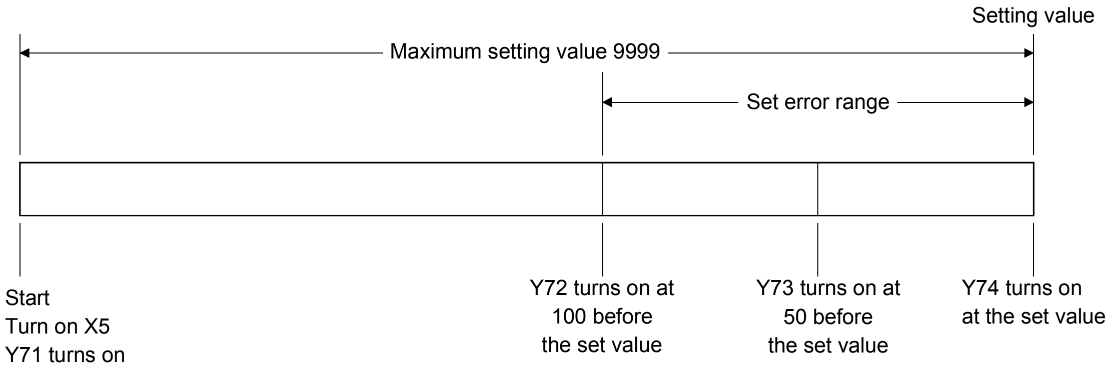
Path name	A:\SCHO
Project name	QA-22
Program name	MAIN



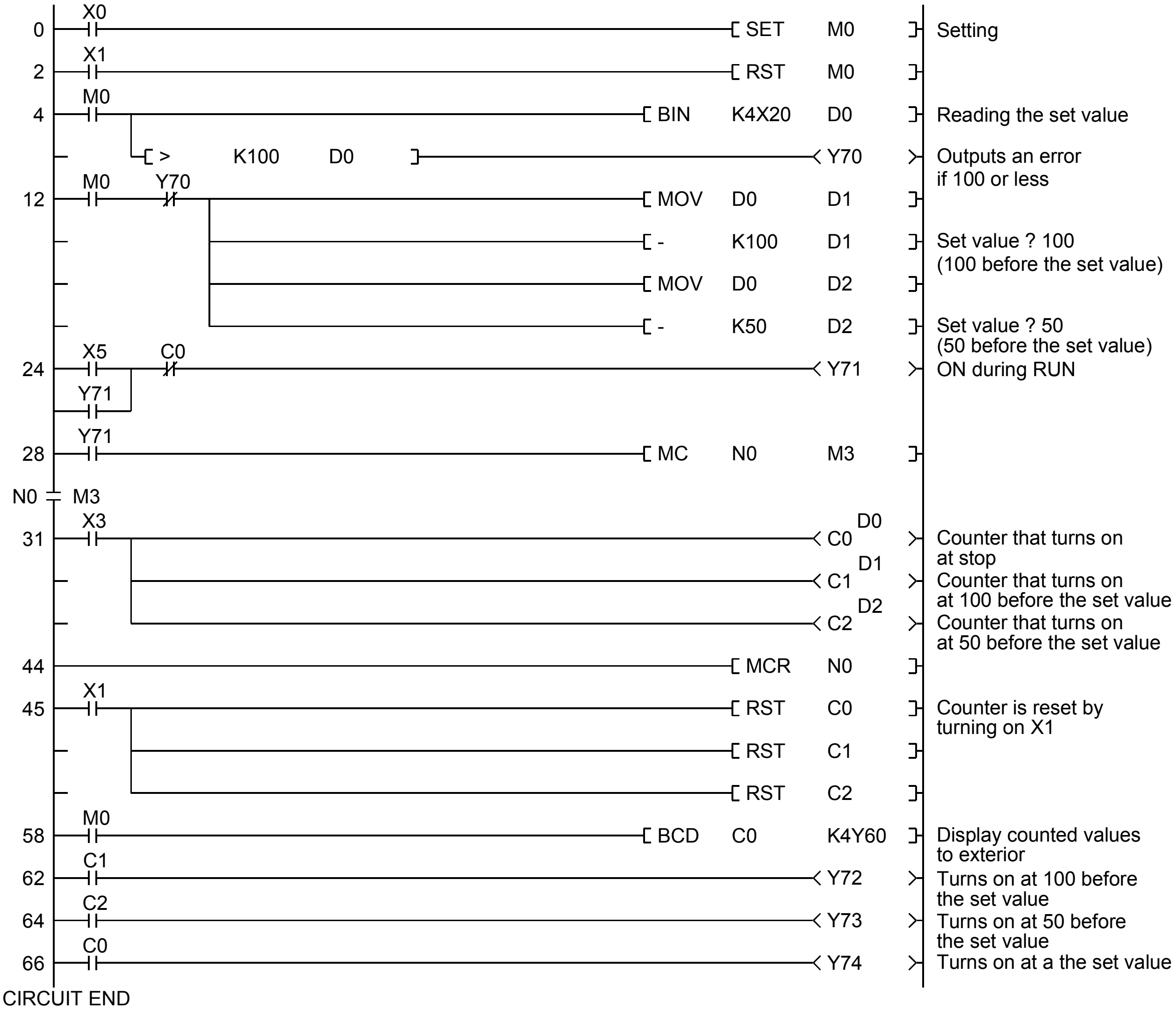
Appendix 5.15 Setting counters externally

With an external digital switch having 4 digits, counters can be set remotely and their current values are displayed in 4 digits. In addition to every count-up, the timer outputs data when it reaches a value of 100 and 50 before the set limit.
Note that a setting error is indicated if the set limit of counters is less than 100.





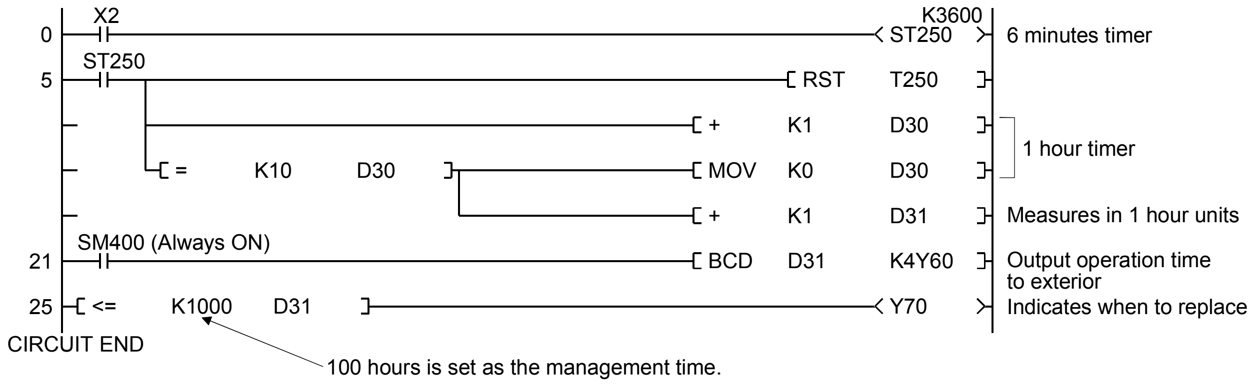
Path name	A:\SCHOOL
Project name	QA-4
Program name	MAIN



Appendix 5.16 Measuring operation time

Setting operation time to a control target is useful for judging when to replace its components, do lubrication to it, etc. The timer ST and data register D must have back-up power source so that they can continue operating at power failure. With the contents of D31 (in one hour units) displayed externally, it can work as an operation timer.

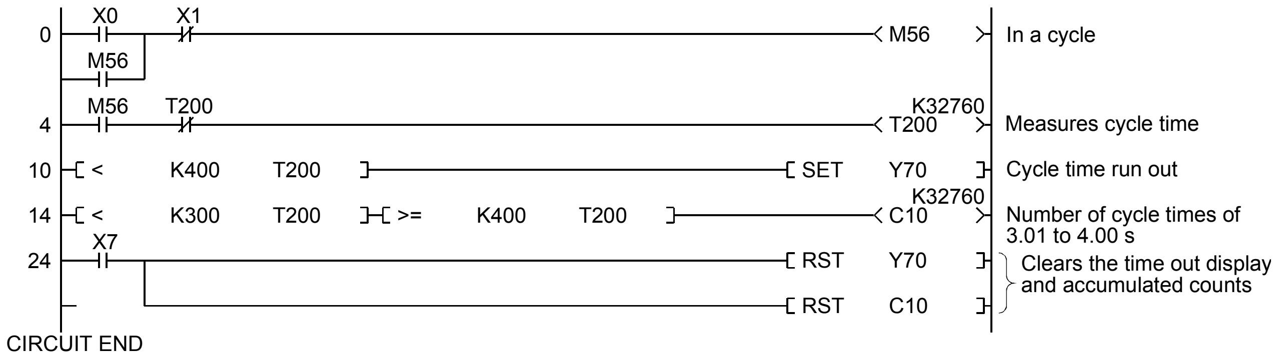
Path name	A:\SCHOOL
Project name	QA-23
Program name	MAIN



Appendix 5.17 Measuring cycle time

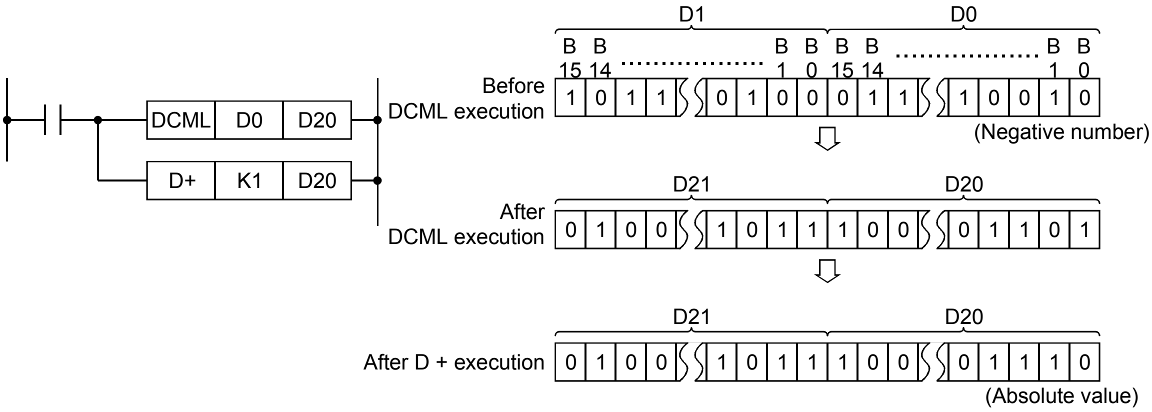
By measuring operation time of a control target (from its start to end), it is possible to display cycle time out, control time lag, etc.
The following ladder can indicate cycle time out. To measure time lag, use the <, >, and = instructions to judge the state of T200, and turn on a counter.

Path name	A:\SCHOOL
Project name	QA-24
Program name	MAIN

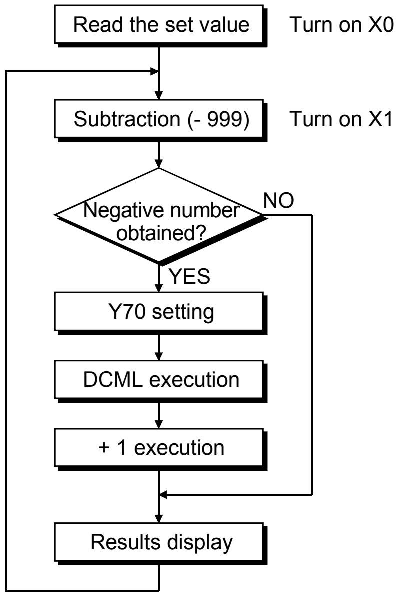


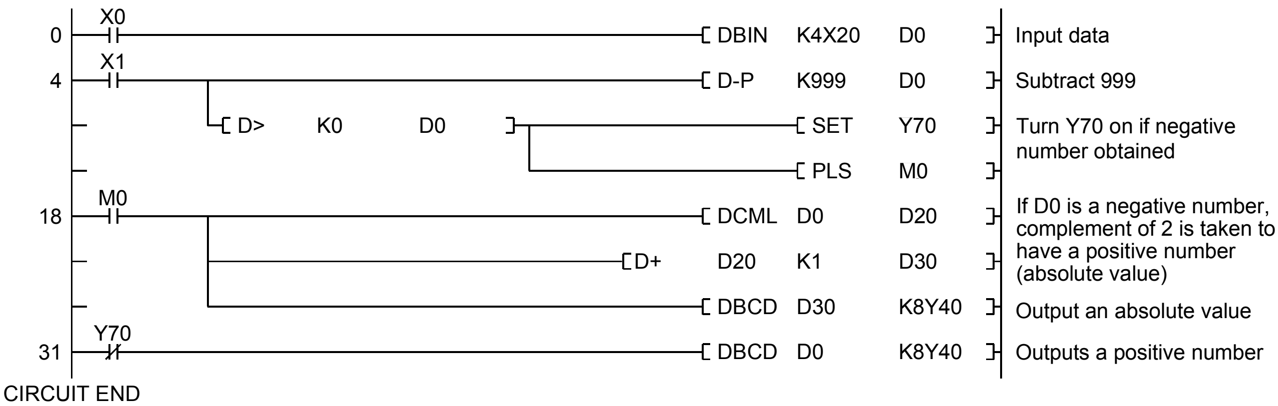
Appendix 5.18 Application example of (D) CML (P)

Obtain absolute values of negative values – 32768 or smaller (to –2147483648 ...... 32 bit data).



(Example)
999 is subtracted from a set value each time X1 is turned on, and the resulting value is displayed.
When the resulting value goes down below zero, the output Y70 turns on, and the absolute value of the resulting value is displayed.

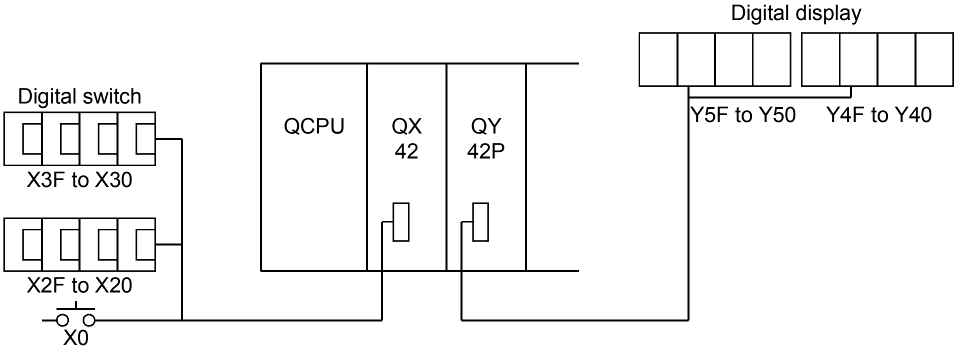


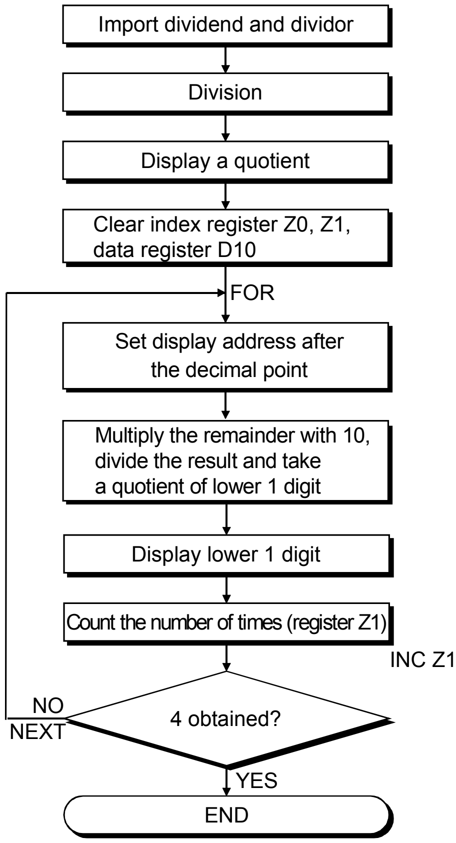


Appendix 5.19 Program showing divided value of 4-digit BIN value to 4 places of decimals

(1) Example 1

Two digital switches are provided, one of which contains a dividend, and the other of which contains a divisor. The results of operation using this dividend and divisor are displayed in its 4 integral parts and 4 decimal parts.





Dividend Digital switch X30 to X3F→D0

Divisor Digital switch X20 to X2F→D1

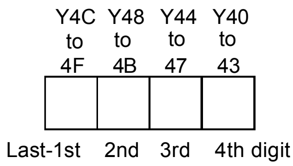
(D0)÷(D1)= (D2)……(D3)

Quotient Remainder

	4×(Z1)→(D10)		HC-(D10)→(Z0)
1st time	4×0 →	0	HC- K0 →HC
2nd time	4×1 →	4	HC- K4 →H8
3rd time	4×2 →	8	HC- K8 →H4
4th time	4×3 →	12	HC- K12 →H0

(D3)×10→(D3)

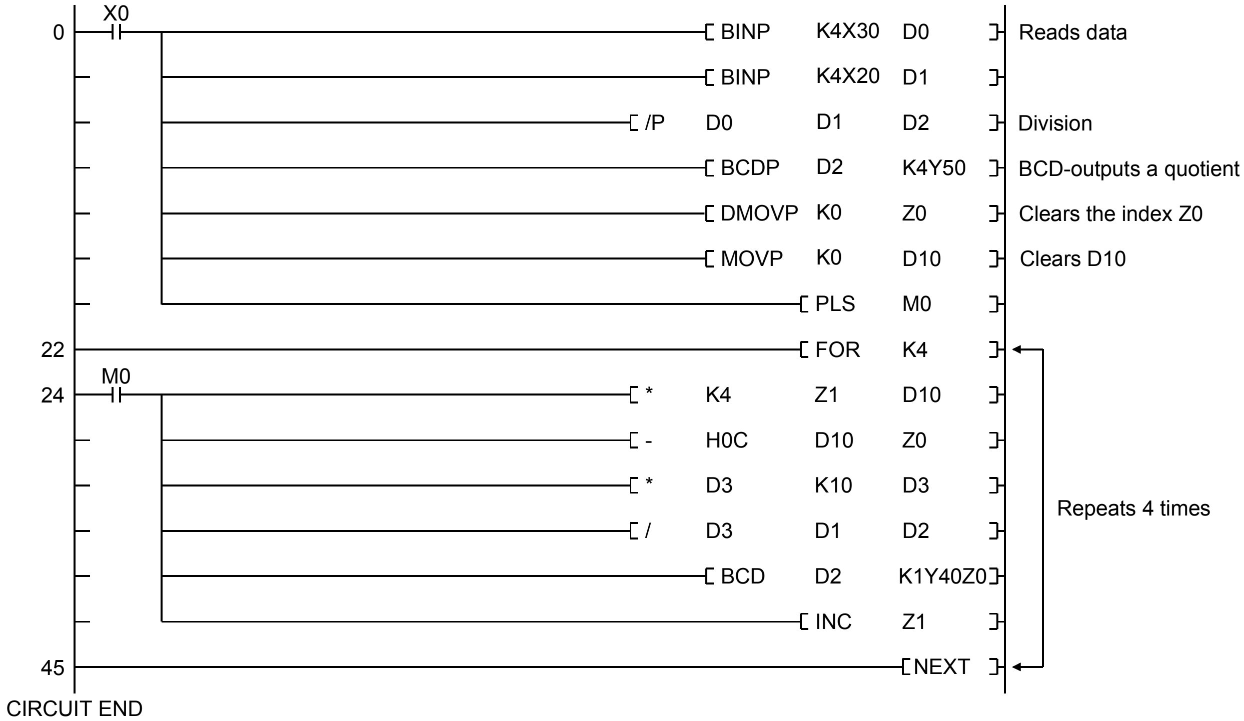
(D3)÷(D1)= (D2)……(D3)



The sequence program of example 1.

The FOR-NEXT instruction is issued to divide each decimal place individually and display 4 decimal places in K4Y40.

Path name	A:\SCHOOL
Project name	QA-5
Program name	MAIN

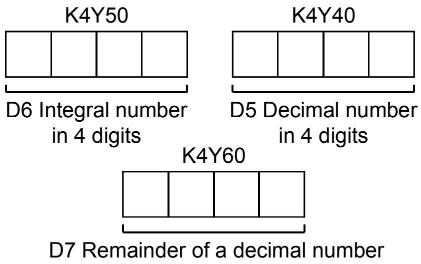


The value of Z1 is added one every $\boxed{\text{INC Z1}}$ instruction.

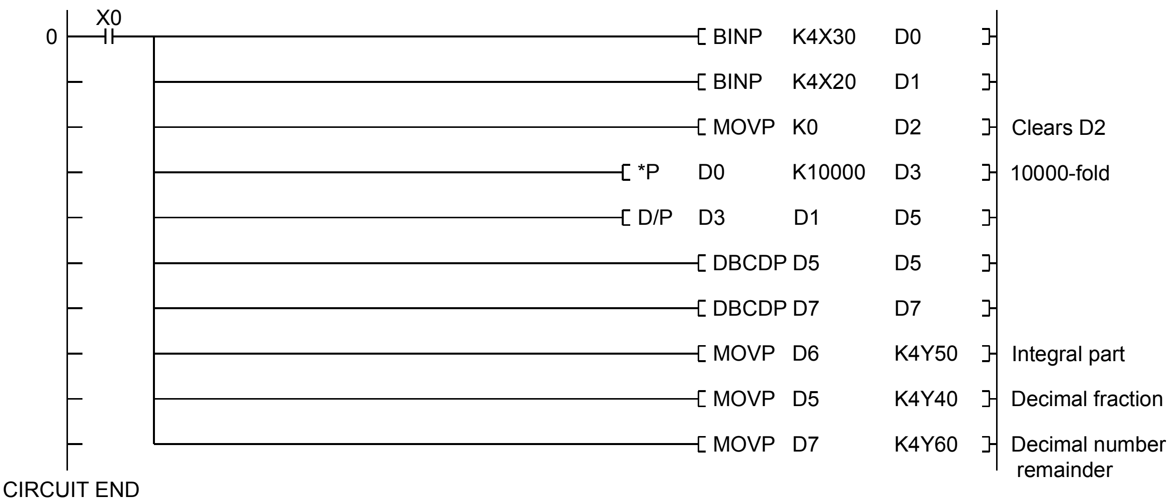
(2) Example 2

D0 is divided by D1 to obtain D5 in 4 decimal places.

The dividend D0 is multiplied with 10,000. The result of dividing calculation using this multiplied value is converted to a BCD value and output to an external digital display.

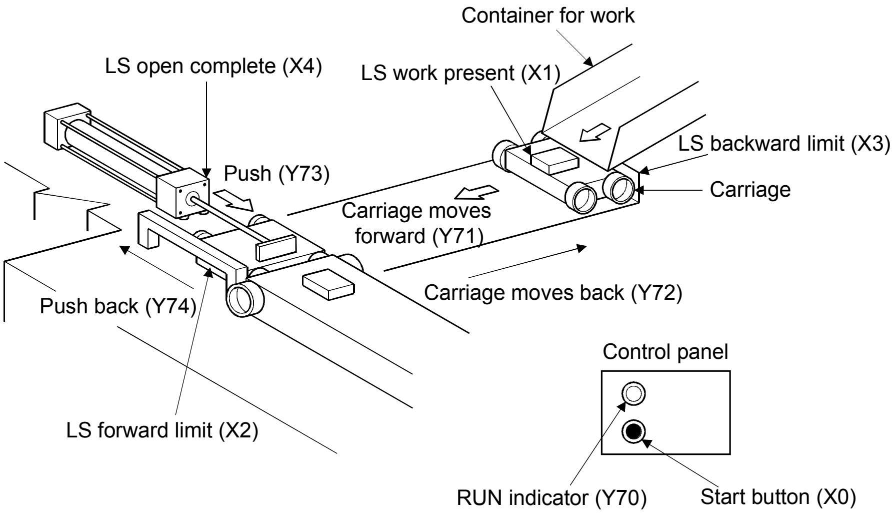


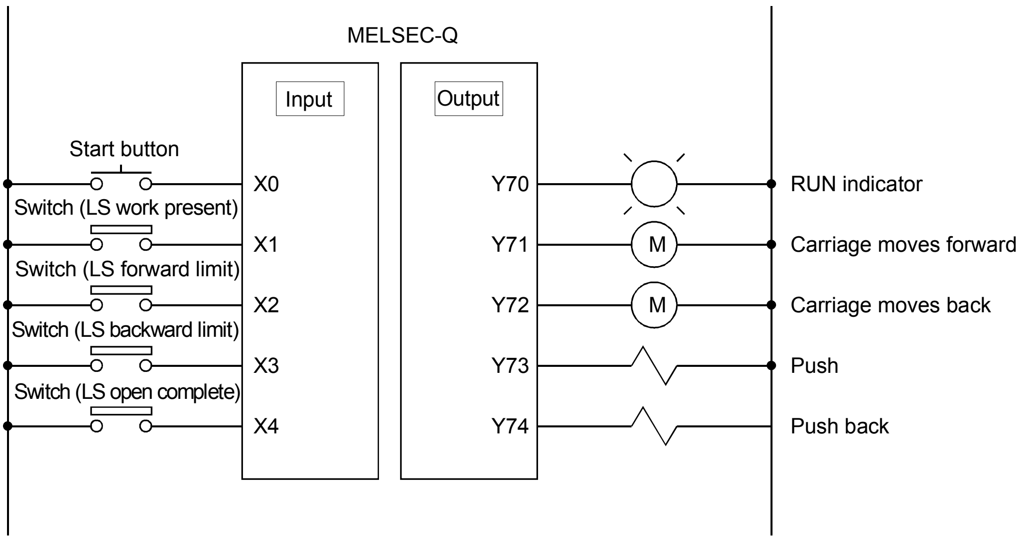
Path name	A:\SCHOOL
Project name	QA-6
Program name	MAIN



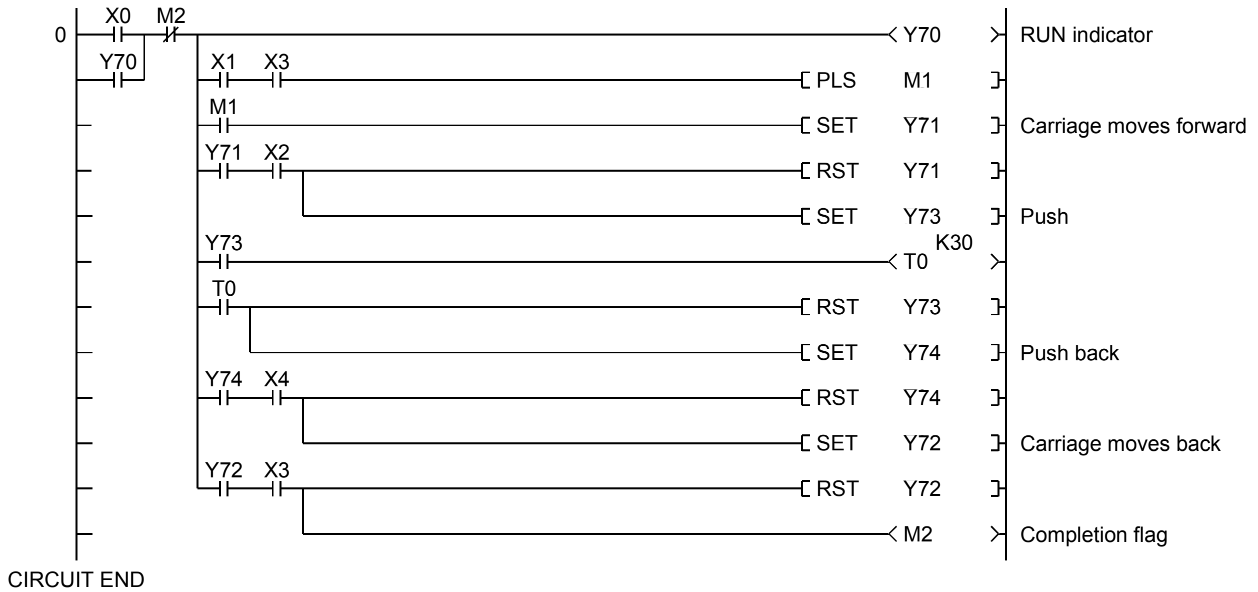
Appendix 5.20 Carriage line control

The following is an example of sequence control using a carriage to convey works (materials).
Series of operations performed in one cycle is as follows; A work is set on the carriage, the carriage moves forward, the carriage stops at forward limit, the pushing arm pushes the work to the other conveyer side, and the carriage moves back to the backward limit.

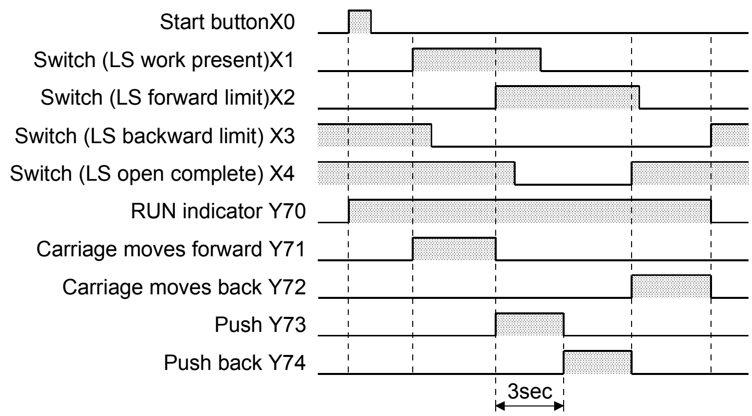




Path name	A:\SCHO
Project name	QA-10
Program name	MAIN

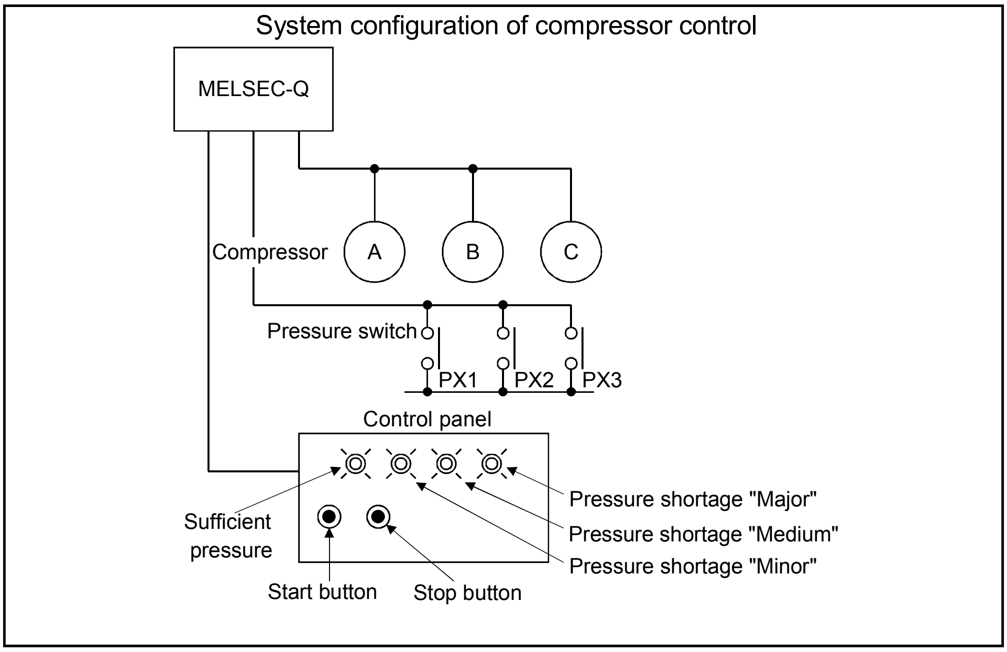


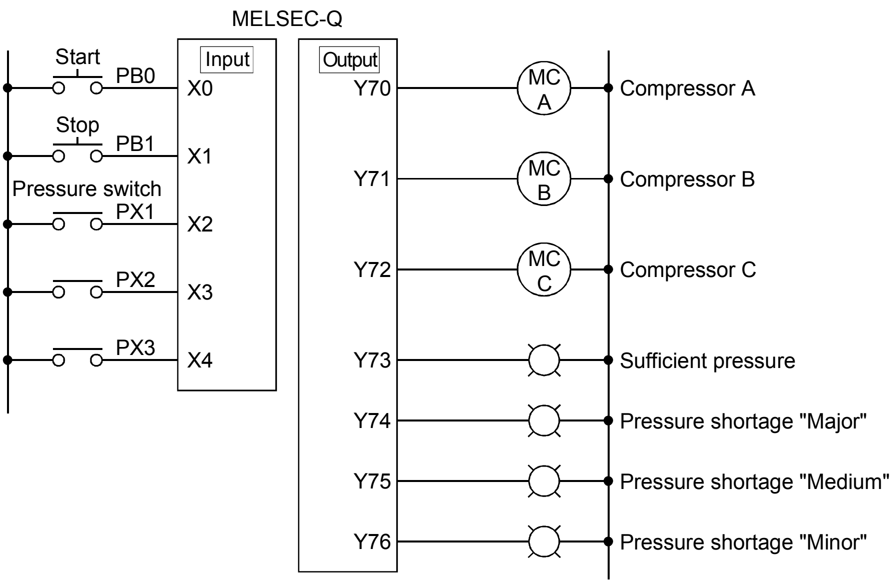
Timing chart



Appendix 5.21 Starting compressors in order using ring counters

This system provides pressure control using three compressors.
Pressure shortage is detected by the three pressure switches. The less pressure is provided, the more compressors are activated. To equal the number of usages of each compressor, compressors are activated according to the set order.





Operation explanation

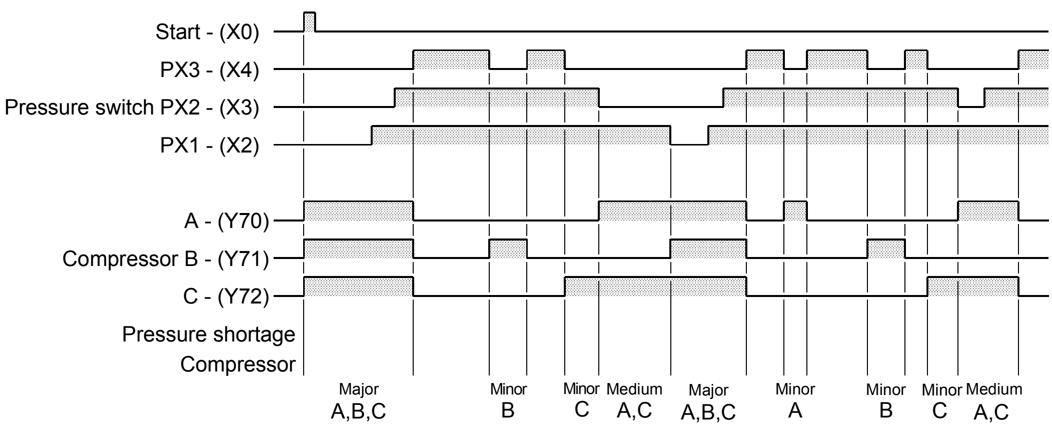
(1) The pressure switches (X2, X3, X4) are initially off. In this state, turning on the start switch (X0) activates the three compressors all together, and when sufficient pressure is obtained (X2, X3, and X4 turn on), the three compressors stop. This is the basic operation of this system.
If all compressors are at stop with sufficient pressure provided and the pressure shortage "Minor" is detected (X4 turns off), one compressor is activated and supplies pressure until sufficient pressure is obtained.
The compressor activated as such is decided in order from A to C each time compressors are reactivated in reaction to pressure shortage.
Note that the stop switch (X1) is available for stopping compressors at any time.

(2) If one compressor could not supply sufficient pressure, the pressure shortage level goes up to "Medium" (X3 turns off) and the second compressor is activated to support the first compressor. This second compressor will be compressor C if compressor A has been in operation, A if B has been in operation, and B if C has been in operation.

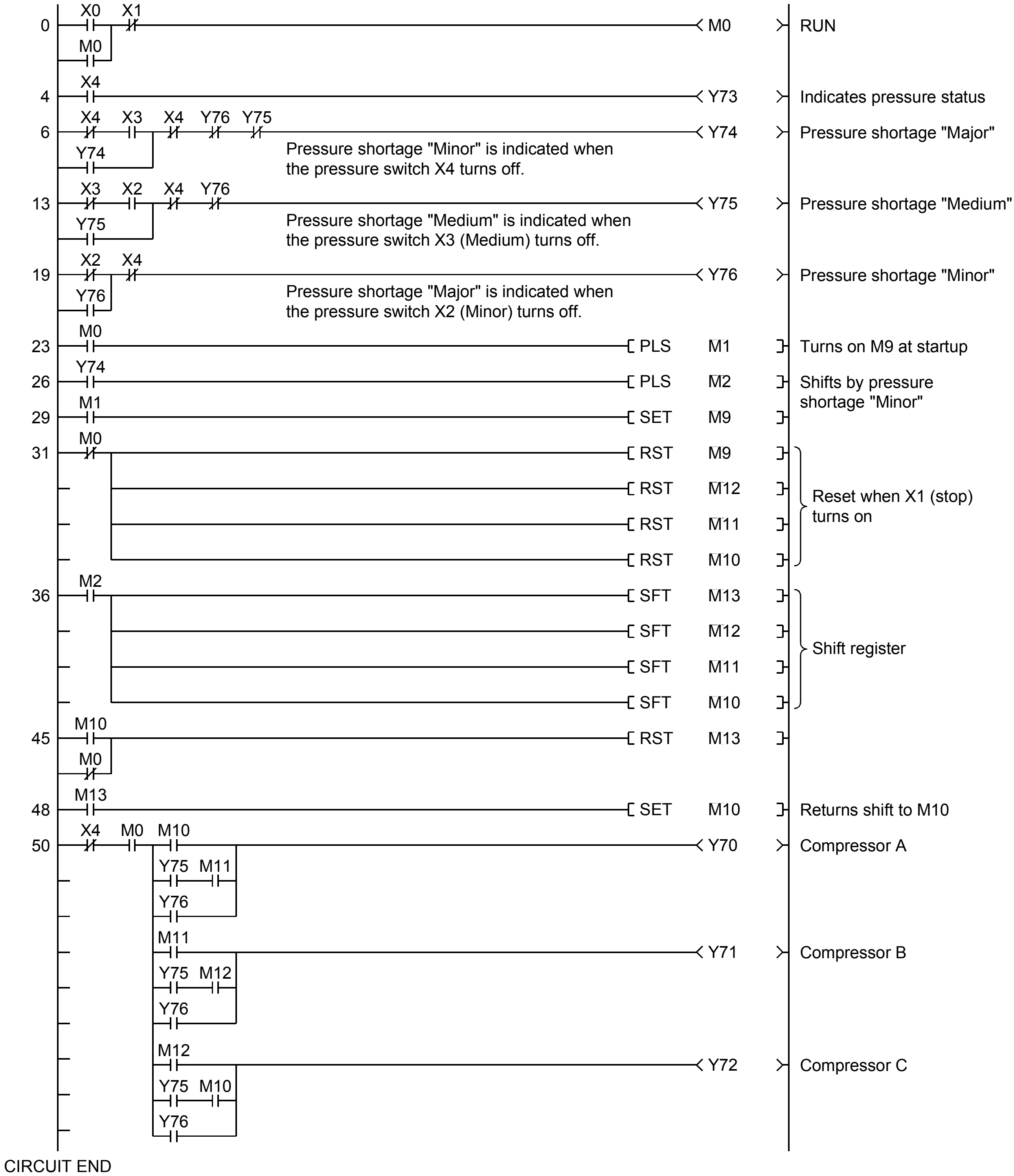
(3) If two compressors could not supply sufficient pressure, pressure shortage level goes up to "Major" (X2 turns off), and the last compressor is activated.
When only one compressor is in operation and pressure shortage level goes from "Minor" to "Major" directly, the rest two compressors are activated at once.

(4) When two or three compressors are in operation, they continue operating together until sufficient pressure is obtained and stop together when obtained (X4 turns on).

Timing chart

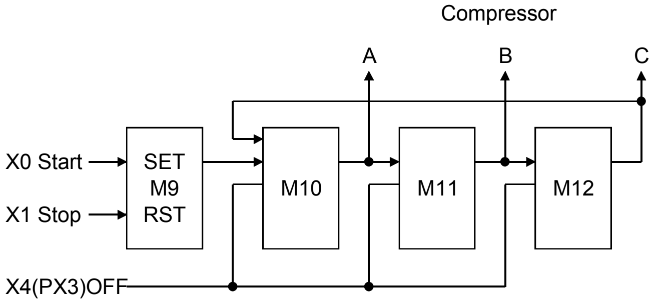


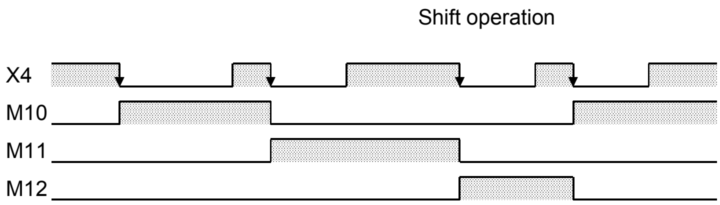
Path name	A:\SCHO
Project name	QA-11
Program name	MAIN



After the basic operation, one compressor is activated in reaction to pressure shortage detected. To use the three compressors equally, there is ordering control available. This control is enabled by the 3-stage ring counter (ring-shaped shift registers) M10 to M12.

A shift signal is generated when pressure shortage is detected (X04 switches from on to off).

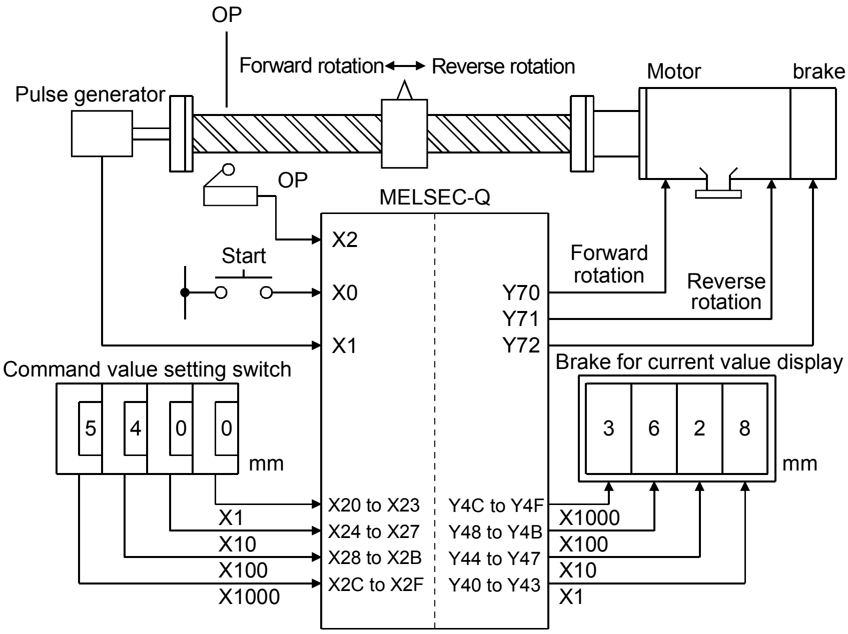




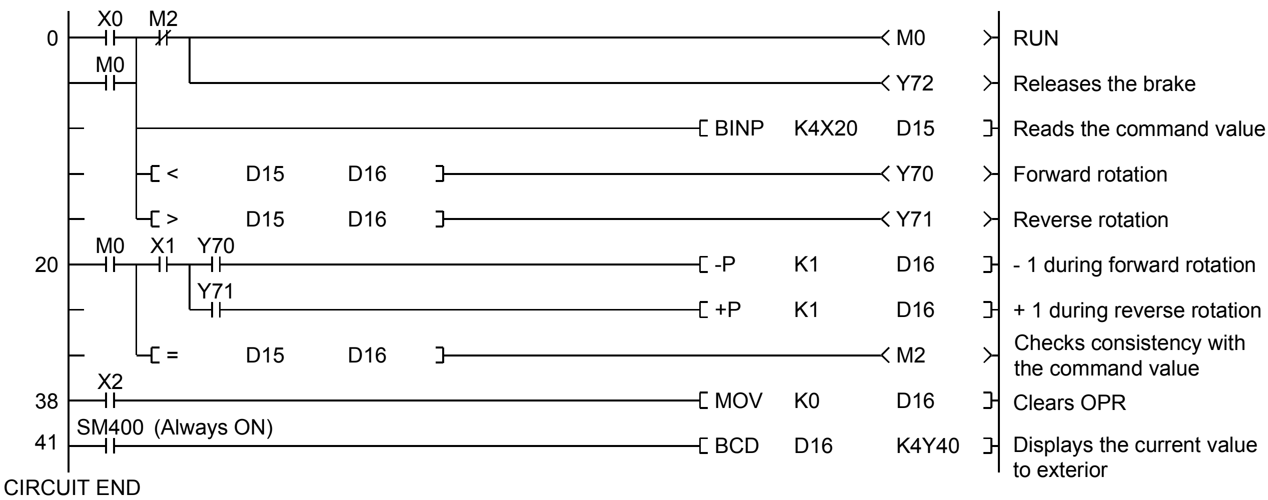
Appendix 5.22 Application example of positioning control

The following is an example of a positioning system with a pulse generator that outputs pulses per motor, brake, and unit of distance.

In this system, a command value is set with the digital switch, and this set command value is compared with the current value at start-up to decide in which way, forward or reverse, the motor rotates. The current value in the register D16 is subtracted by 1 when in forward direction, and incremented by 1 when in reverse direction. Positioning completes when the command value matches the current value. The current value is converted to a BCD value so that current position is represented in 4-digit decimal numbers.



Path name	A:\SCHO
Project name	QA-26
Program name	MAIN



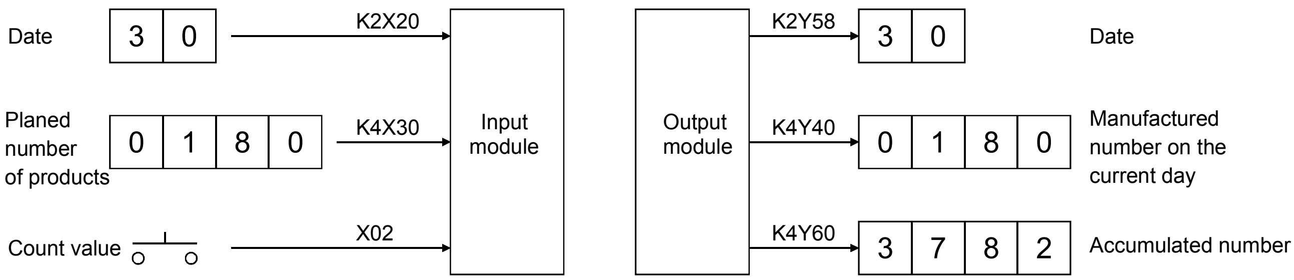
Appendix 5.23 Application example using index Z

(1) Counts the number of manufactured products every day in one month cycle, and stores the resulting number to the corresponding register of the date (D1 to D31).

(2) Inputs the planed number of products to manufacture using the external digital switch. Production stops when this number is accomplished.

(3) Inputs the date using the external digital switch.

(4) Displays to exterior how many products have been manufactured in the current month as well as the number manufactured on the current day.

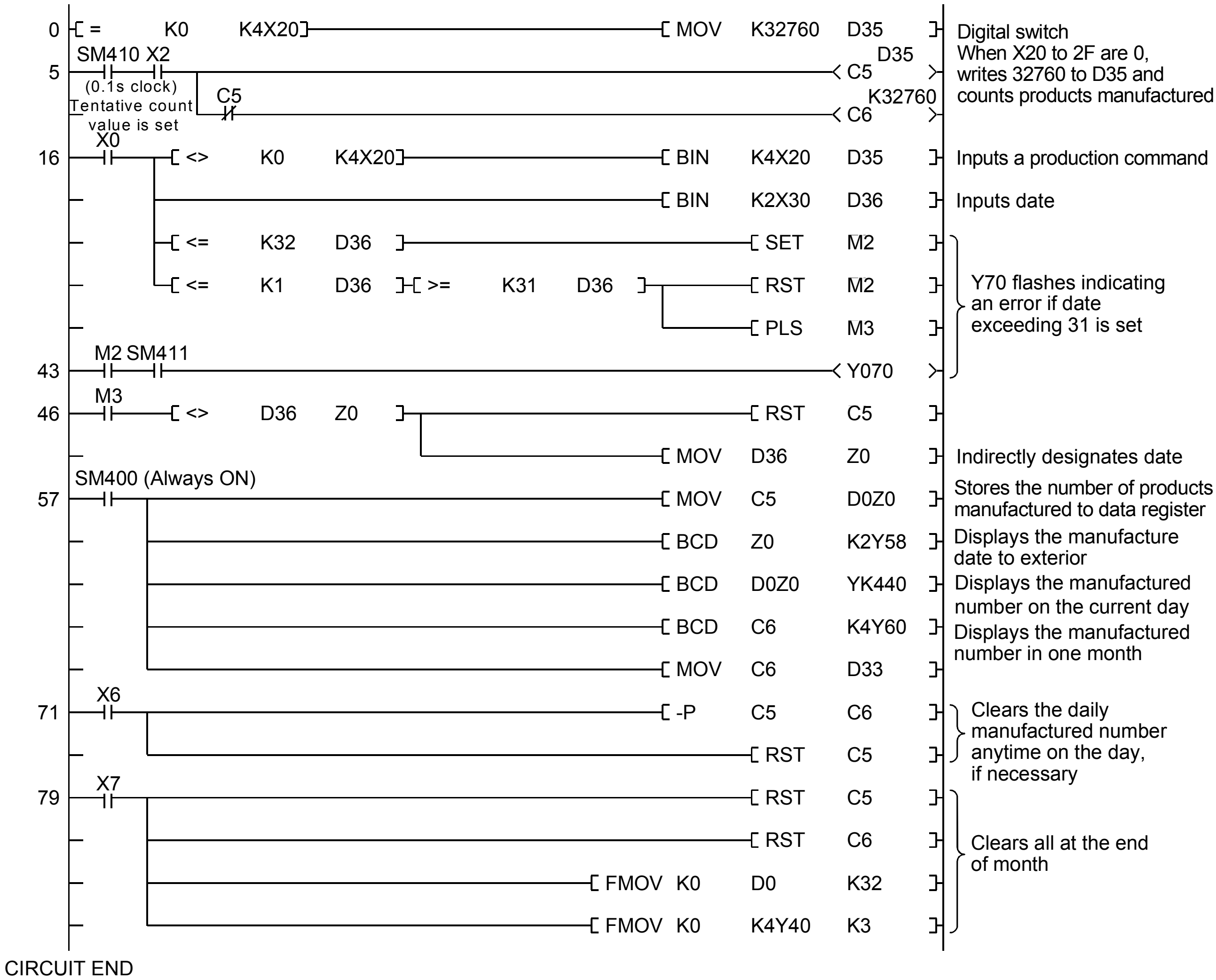


How many products manufactured on the current day is counted with C5.
Accumulated number of products manufactured is counted with C6.
The date is entered in the index Z to indirectly designate the data register corresponding to the date using D0Z0.
When Z0 is 30, D0Z0 becomes 0 + Z, designating D30.

D	0	0	D	8	124	D	16	263	D	24	170	D	32	0	
D	1	159	D	9	129	D	17	241	D	25	194	D	33	3782	Accumulated number
D	2	145	D	10	169	D	18	181	D	26	219	D	34	0	
D	3	168	D	11	119	D	19	179	D	27	0	D	35	180	Planed number of products
D	4	144	D	12	247	D	20	0	D	28	0	D	36	30	Date
D	5	130	D	13	0	D	21	0	D	29	213	D	37	0	
D	6	0	D	14	0	D	22	163	D	30	180	D	38	0	
D	7	0	D	15	124	D	23	129	D	31	0	D	39	0	

Manufacturing results of each day ranging from 1 to 31 are stored in D1 to D31, being available as production data.

Path name	A:\SCHO
Project name	QA-7
Program name	MAIN



FMOV	K0	D0	K32

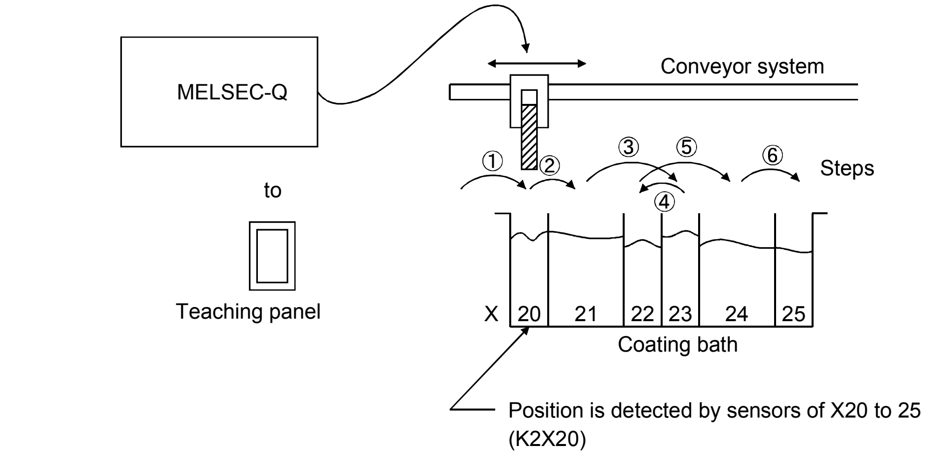
Simultaneously transfers data 0 to D0 to D31.

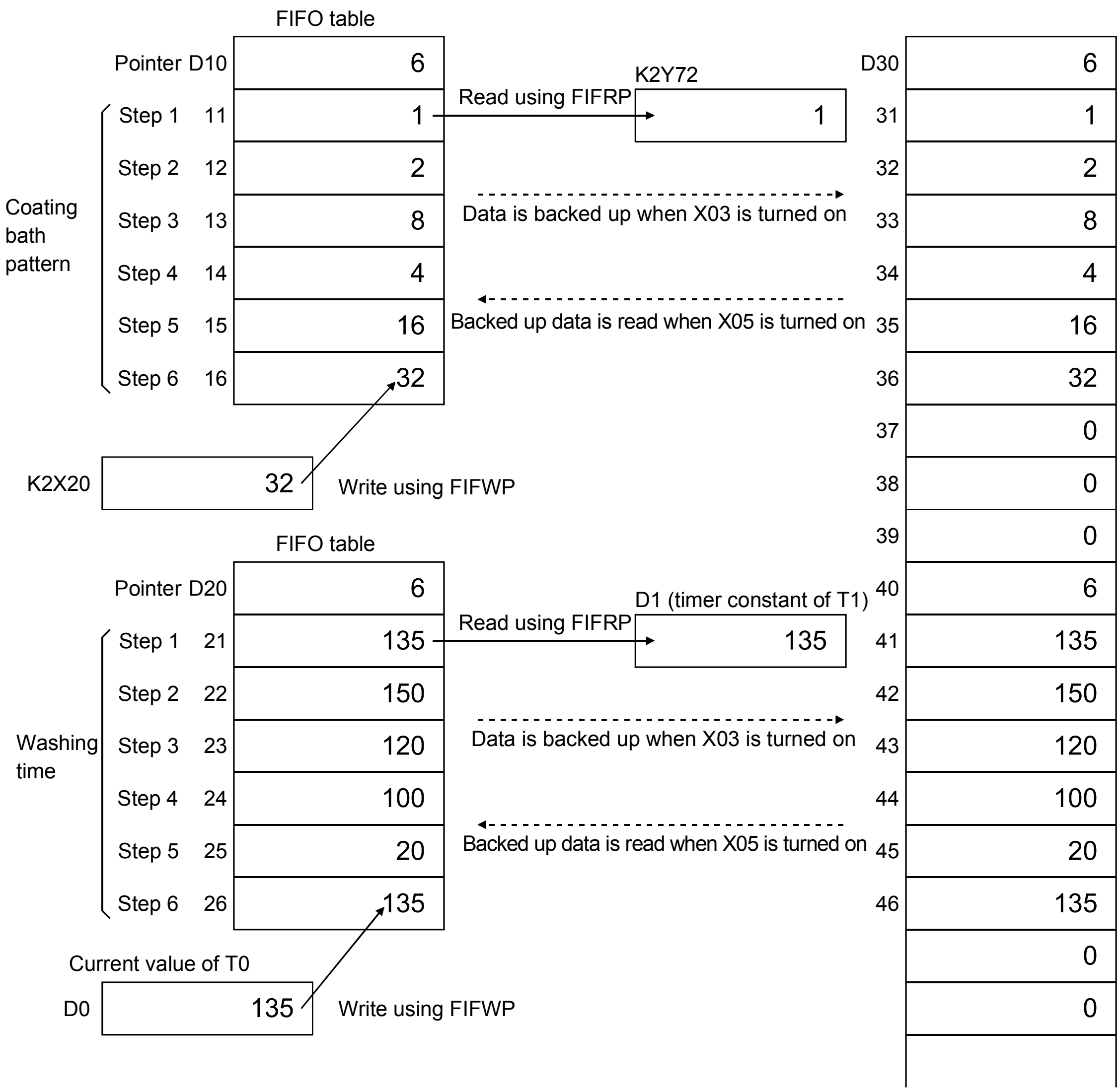
FMOV	K0	K4Y40	K3

Simultaneously transfers data 0 to D0 to D31.

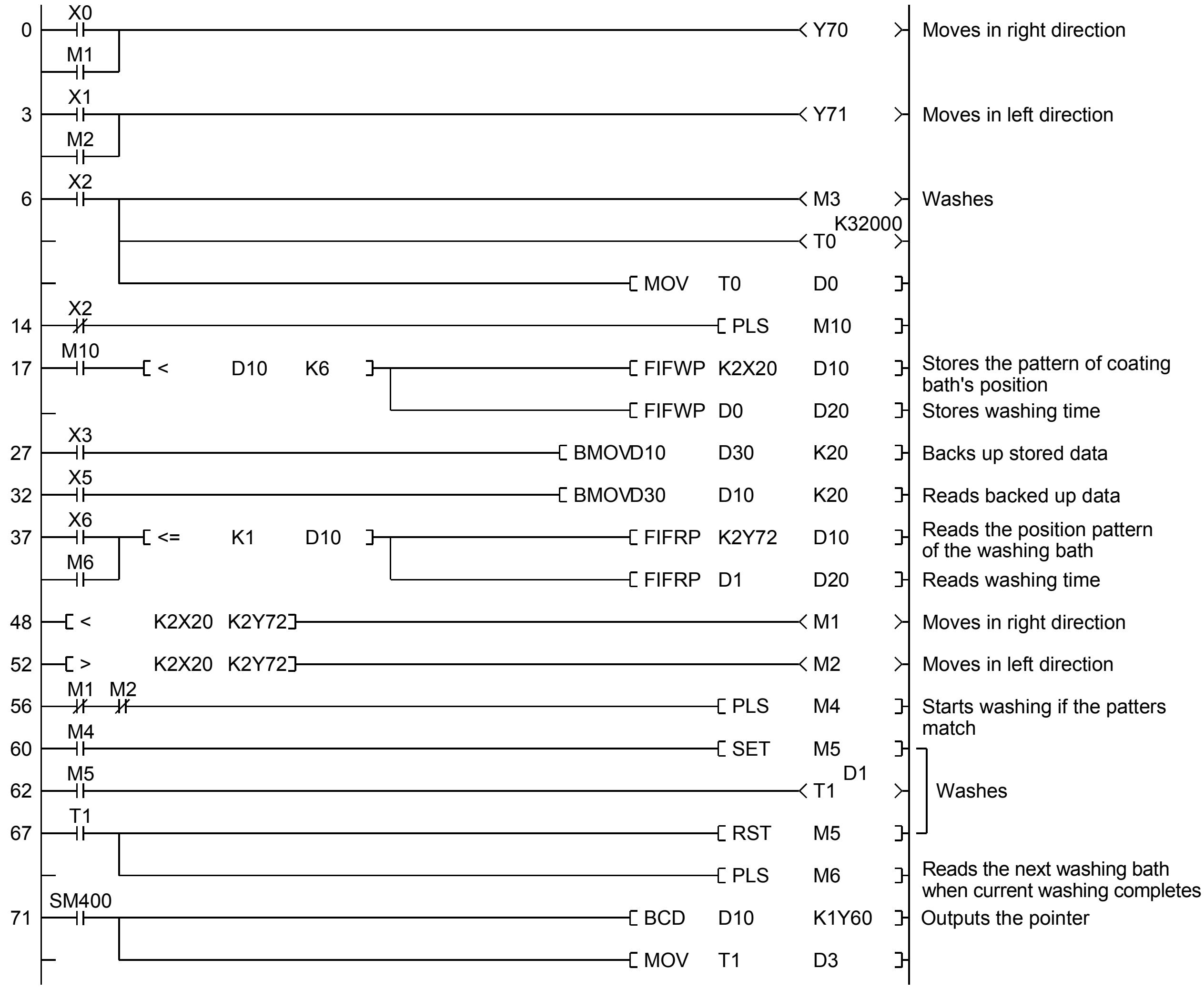
Appendix 5.24 Application example of FIFO instruction

Manual coating work and its working time can be stored and duplicated by machinery later.





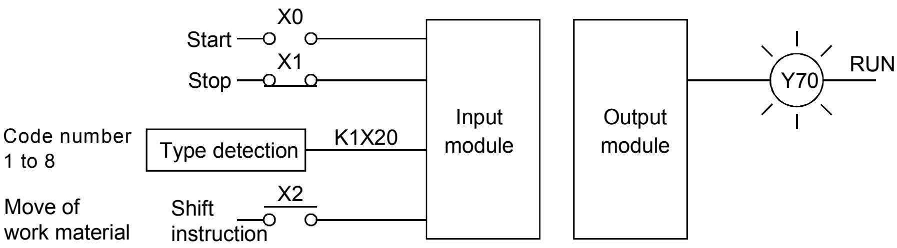
Path name	A:\SCHO
Project name	QA-9
Program name	MAIN



CIRCUIT END

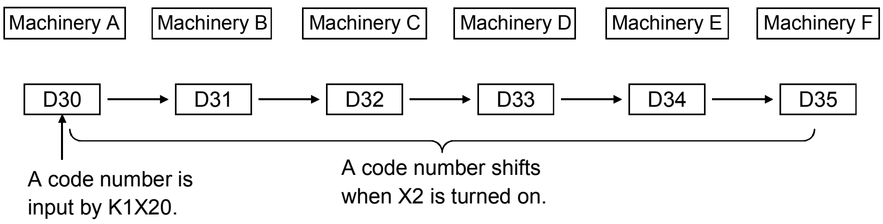
Appendix 5.25 Application example of data shift

Working materials are conveyed along with their code numbers, and the data register of the processing machinery is analyzed to machine the work material according to its code number.

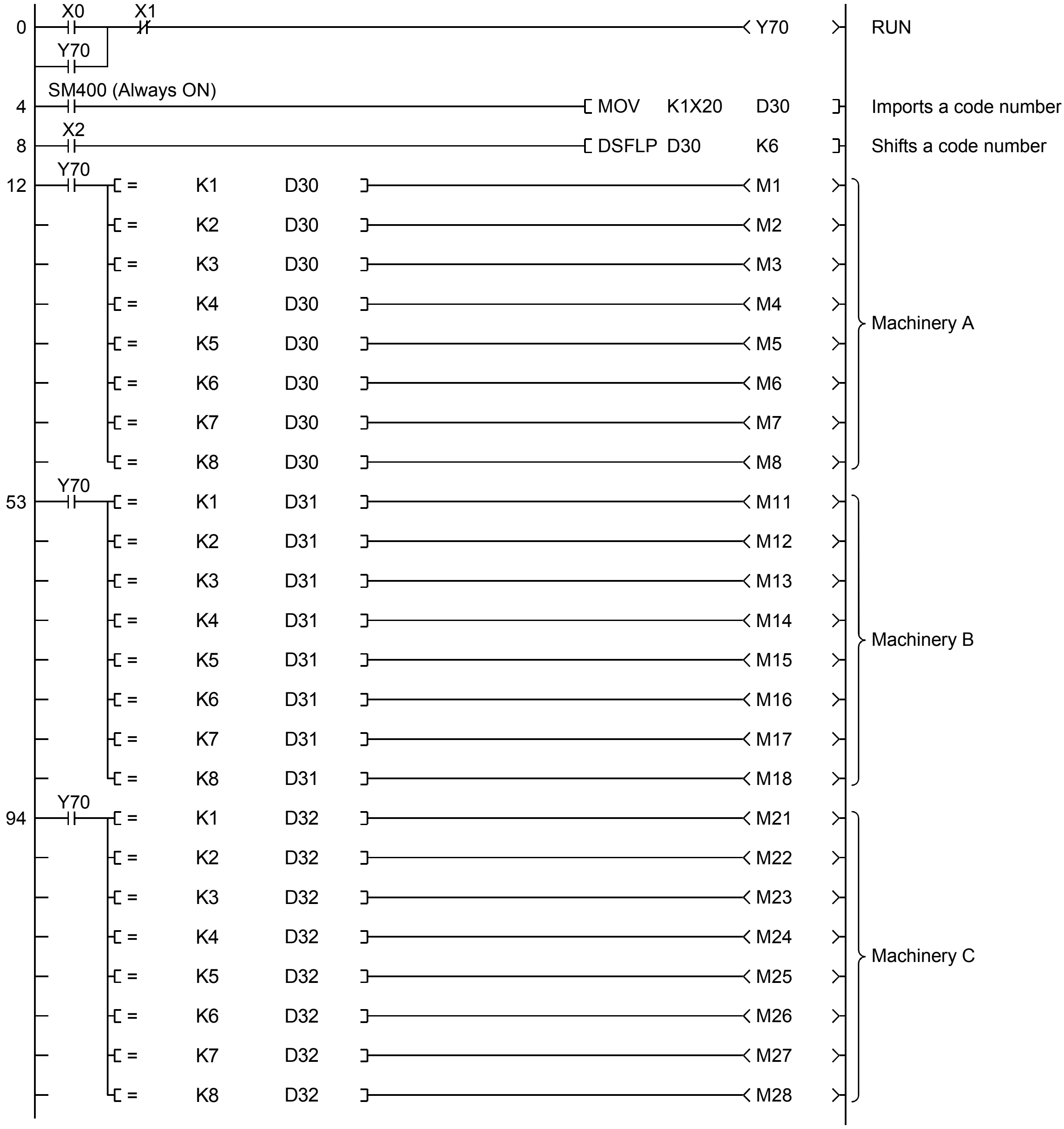


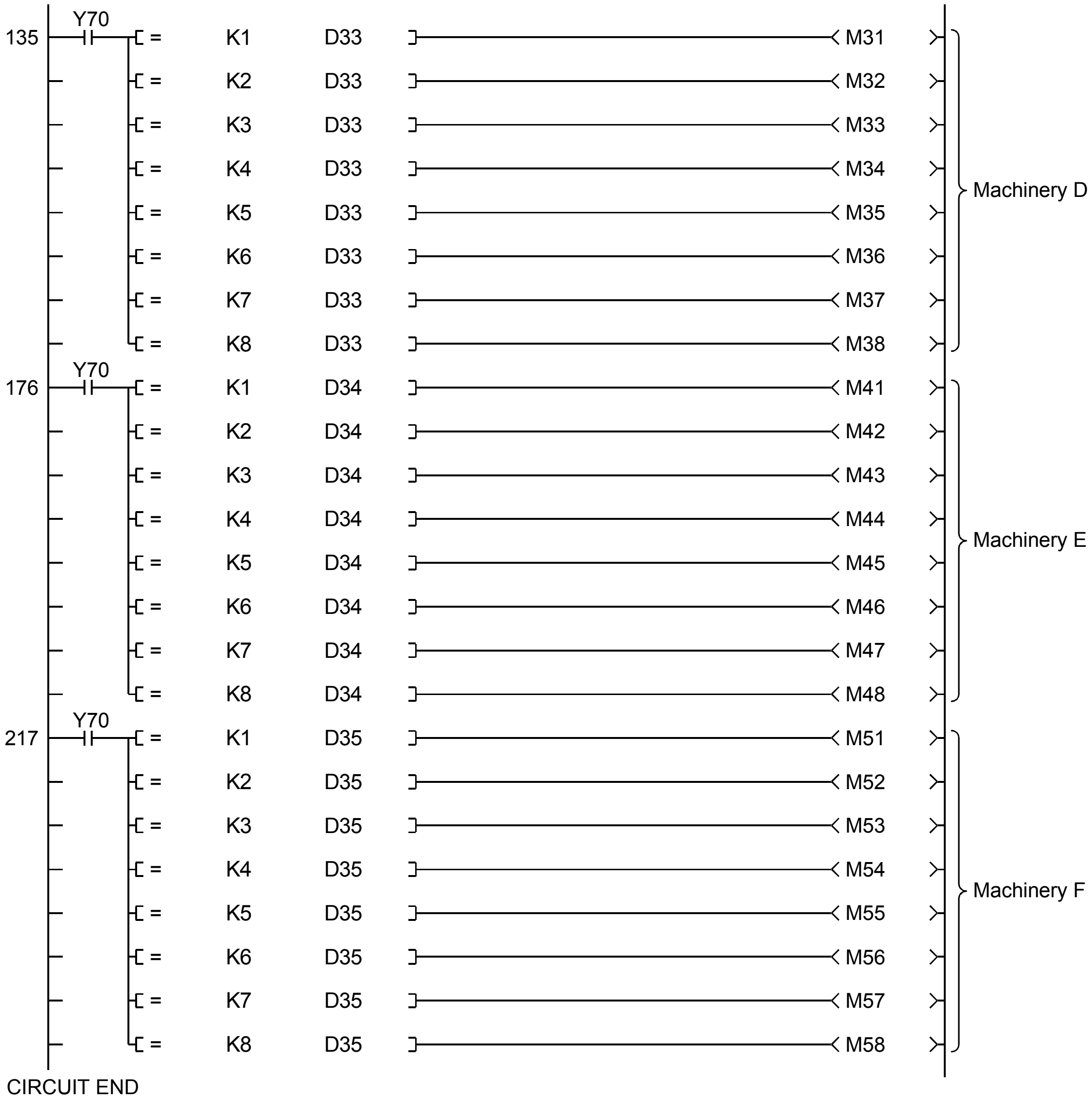
Machinery	Data register	Code 1	Code 2	Code 3	Code 4	Code 5	Code 6	Code 7	Code 8
A	D30	M1	M2	M3	M4	M5	M6	M7	M8
B	D31	M11	M12	M13	M14	M15	M16	M17	M18
C	D32	M21	M22	M23	M24	M25	M26	M27	M28
D	D33	M31	M32	M33	M34	M35	M36	M37	M38
E	D34	M41	M42	M43	M44	M45	M46	M47	M48
F	D35	M51	M52	M53	M54	M55	M56	M57	M58

A code number is stored in the data register, and the M corresponding to the stored number is activated to perform machining.



Path name	A:\SCHO
Project name	QA-12
Program name	MAIN

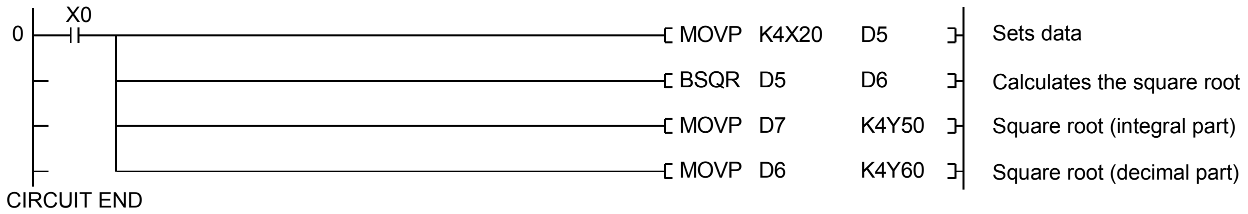



135 Y70 [= K1 D33] (M31)
[= K2 D33] (M32)
[= K3 D33] (M33)
[= K4 D33] (M34)
[= K5 D33] (M35)
[= K6 D33] (M36)
[= K7 D33] (M37)
[= K8 D33] (M38)
Machinery D
176 Y70 [= K1 D34] (M41)
[= K2 D34] (M42)
[= K3 D34] (M43)
[= K4 D34] (M44)
[= K5 D34] (M45)
[= K6 D34] (M46)
[= K7 D34] (M47)
[= K8 D34] (M48)
Machinery E
217 Y70 [= K1 D35] (M51)
[= K2 D35] (M52)
[= K3 D35] (M53)
[= K4 D35] (M54)
[= K5 D35] (M55)
[= K6 D35] (M56)
[= K7 D35] (M57)
[= K8 D35] (M58)
Machinery F
CIRCUIT END

Path name	A:\SCHO
Project name	QA-14
Program name	MAIN

Appendix 5.26 Example of operation program calculating square root of data

The data stored in D5 is calculated to its square root and the result is stored in D6 and D7.



Results of square root operation are stored as follows.

$$\frac{\sqrt{D5}}{\begin{array}{c}\text{0 to 9999}\\\text{(BCD value)}\end{array}} = \underset{\text{(BCD value)}}{\overset{\boxed{\begin{array}{c}\text{Square root}\\\text{(integral part)}\end{array}}}{\frac{D6}{\text{0 to 9999}}}} \cdot \underset{\text{(BCD value)}}{\overset{\boxed{\begin{array}{c}\text{Square root}\\\text{(decimal part)}\end{array}}}{\frac{D7}{\text{0 to 9999}}}} \ldots$$

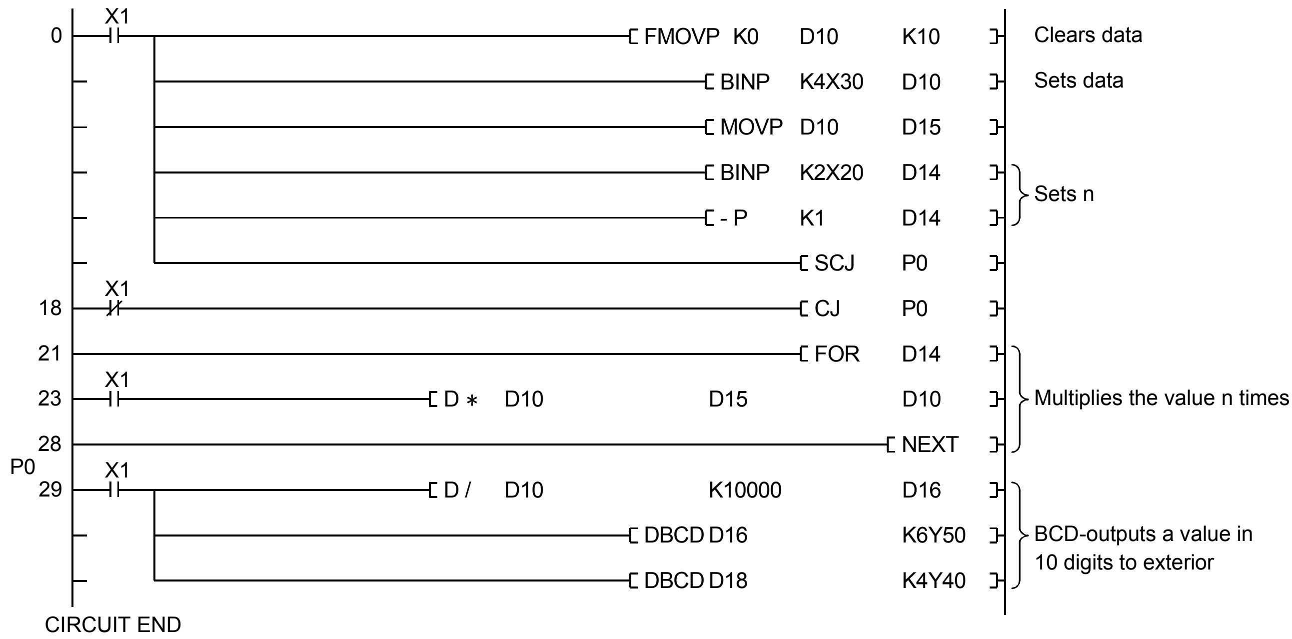
A value in 5th decimal pace is rounded off.Therefore a value in 4th decimal place has error of ±1.

REMARK
QCPUs provide square root operation instructions for data in a real number (floating point) format.

Path name	A:\SCHO
Project name	QA-15
Program name	MAIN

Appendix 5.27 Example of operation program calculating n-th power of data

A value stored in D10 is calculated to its n-th power (n is a value stored in D14) and the result is stored in D10.

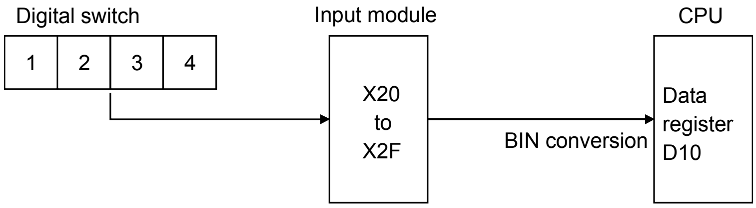


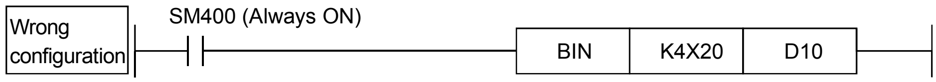
NOTE

An operation error occurs if a value in D10 exceeds 2147483647.

Appendix 5.28 Program using digital switch to input data

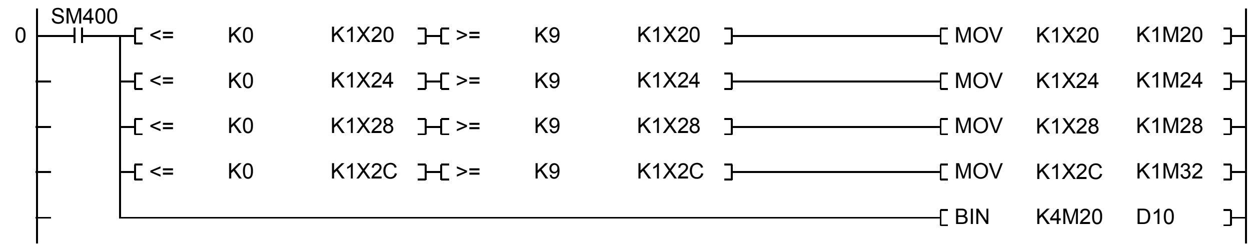
When always inputting and storing a set value of the digital switch to D10 of the PLC



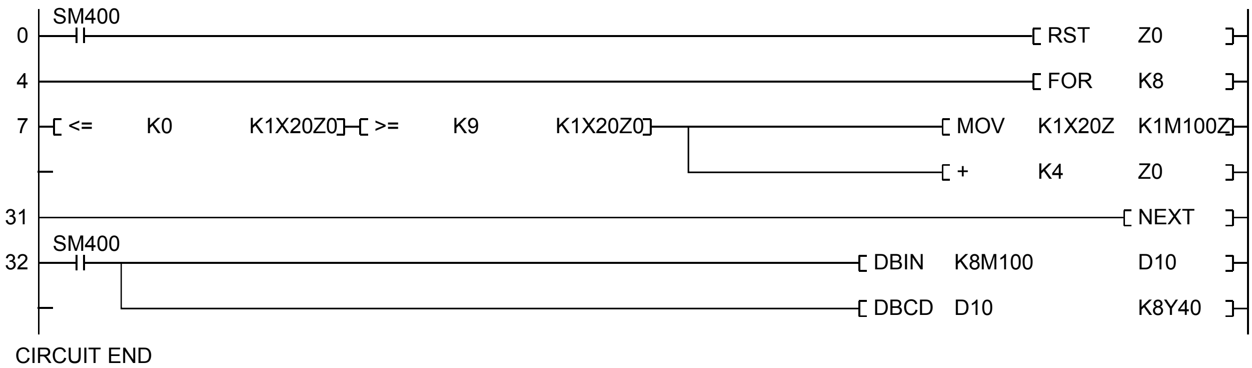


In the above program, changing a value of the digital switch with the PLC in RUN mode may cause codes other than 0 to 9 to occur at the timing of the change. This generates "OPERATION ERROR" of the CPU.
To avoid this, write a program as follows.

(Example 1) When 4 digits of X20 to X2F are used.



(Example 2) When 8 digits of X20 to X3F are used.

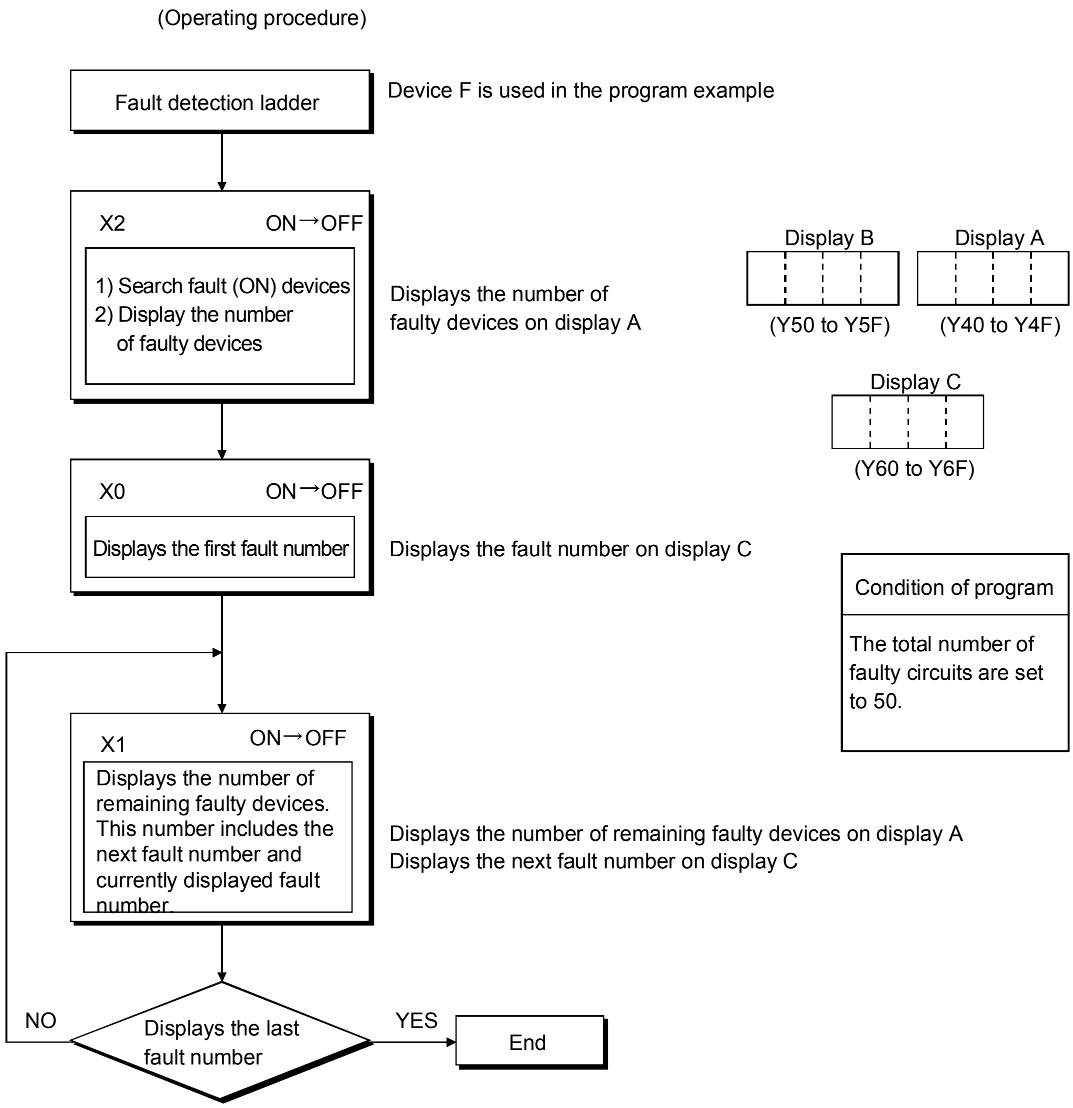


Appendix 5.29 Displaying number of faults with fault numbers using fault detection program

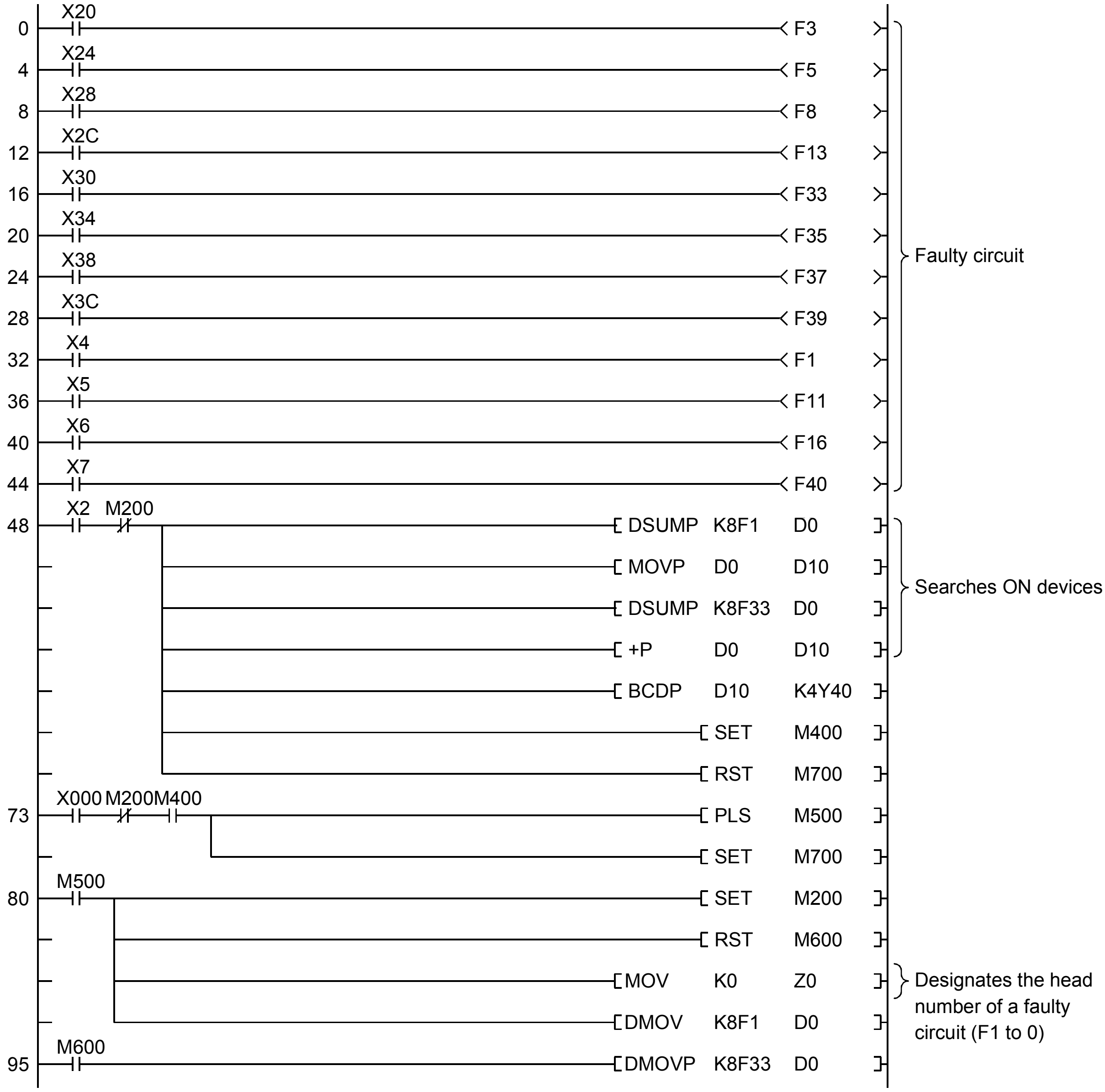
The following program sequentially displays the number of turned-on bit devices (X, M. F, etc.) among many bit devices being used continuously, together with their device numbers.

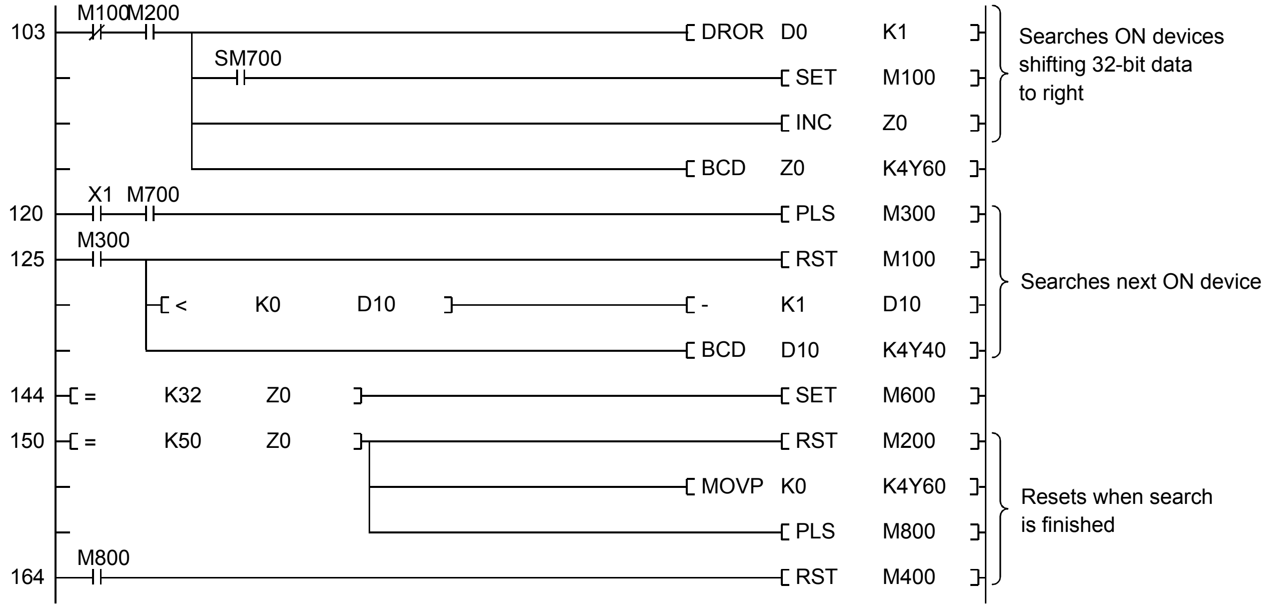
[Application example]
When M or F is used as an output device of a fault detection program, use the following program to know how many faults were occurred and fault code numbers of the faults occurred.

[Sequence program flow]



Path name	A:\SCHO
Project name	QA-31
Program name	MAIN



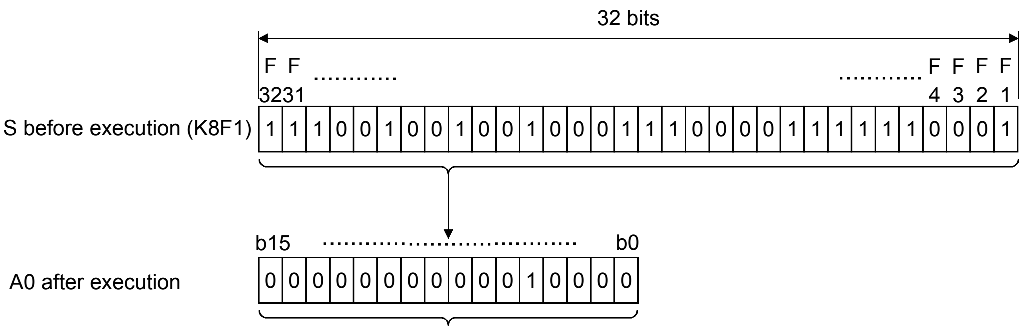


CIRCUIT END

(1) Searching ON devices

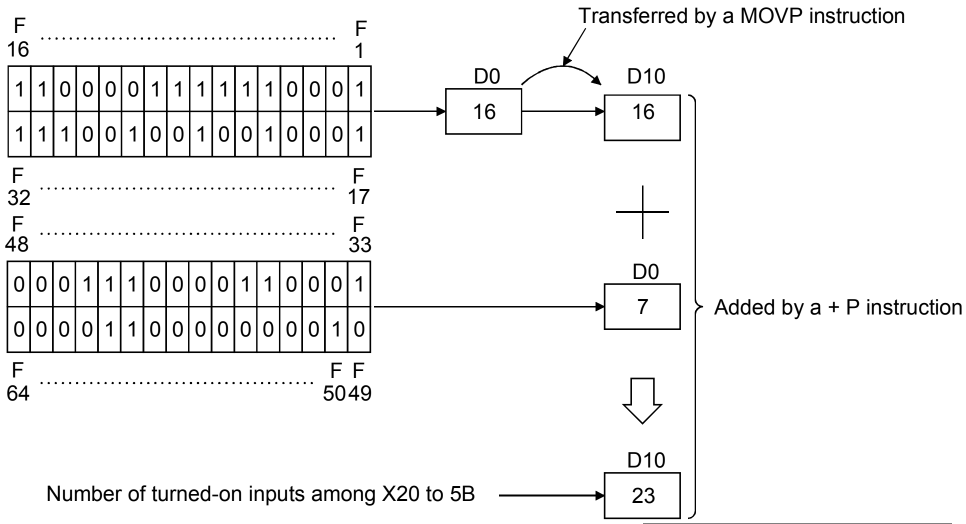
DSUMP	K8F1	D0

DSUMP	K8F33	D0



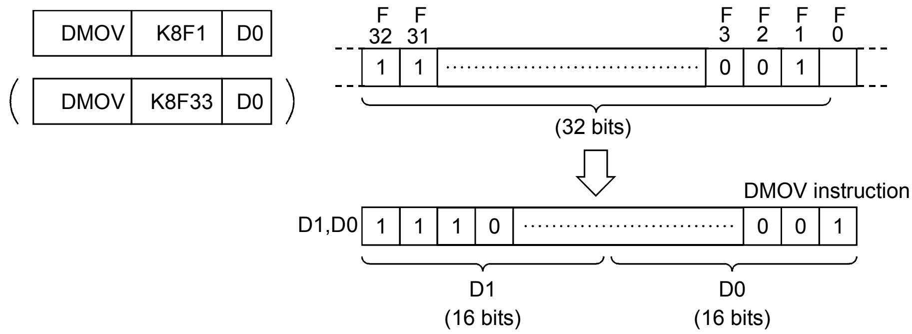
The total number of bits with 1 is stored in BIN.
(16 in this example)

When X2 is turned on, the number of turned-on bits among F1 to F64 is stored to D10 and displayed.

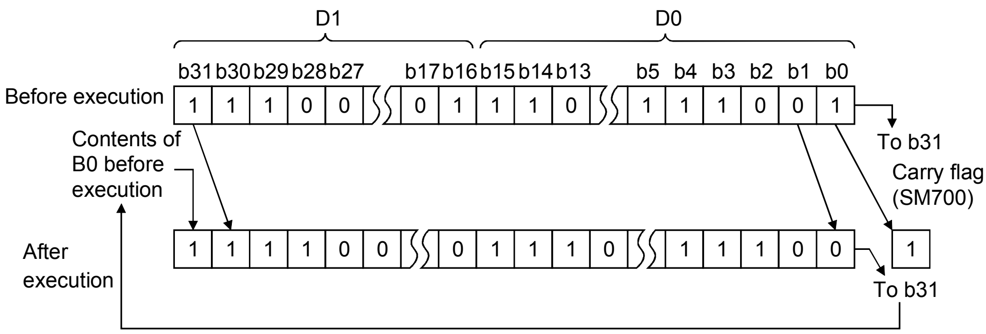


(2) Searching ON devices shifting 32-bit data to left

DROR	D0	K1



DRDR	D0	K1



(1) When X0 is turned on, the above shift data (D0, D1) is set. After that, the data is shifted in right direction by 1 bit at each scan until a turned on bit is detected.
When a turned-on bit is detected, shifting stops in that scan (SM700 turns on), and the accumulated number of shifts (equivalent to a device number) is displayed.

(2) Detecting the next turned-on bit takes place each time X1 is turned on, and the detected device number is displayed in the same manner. At the same time, 1 is subtracted from the number of turned-on bits, which was obtained in advance, to display the remaining number of turned-on bits.

APPENDIX 6 Keys of GX Developer

The following table lists keys used in GX Developer with their applications.

(1) Names and applications of keys

	Names	Application	Names	Application
JIS compliant keys	Esc	Closes windows, stops execution, opens/closes instruction selection windows	Alt	Selects menus
	Tab	Inputs TAB codes, moves a cursor fast	Back Space	Deletes a character before a cursor
	Ctrl	Enables various operation in combination with alphanumerical keys or function keys	Enter	Inputs a carriage return
	Shift	Selects characters at the shift position		
	Shift + Caps Lock	Switches alphabet characters between upper and lower case		
Special keys	Page Up	Goes up in a displayed page such as ladders, HELP, etc. (scrolls 1 screen in - direction)	Print Screen	Captures a screen shot
	Page Down	Goes down in a displayed page such as ladders, HELP, etc. (scrolls 1 screen in + direction)	Scroll Lock	Disables scroll-up and scroll down
	Insert	Switches between overwrite and insert in ladder screens	Num Lock	Makes the ten keys function only for inputting numbers
	Delete	Deletes a character after a cursor (clears all set contents)		
	Home	Moves a cursor to the home position		
	↑ ↓ ← →	Moves a cursor, or scrolls a line in screens such as ladders, lists, etc.(↑ ↓)		

(2) Function keys in ladder mode

Combination	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	C	V	X
None	—	Write	Monitor	Conversion	┤├	└┤├┘	-( )-	-[ ]-	——	│	—	—	—
Alt	Ladder/ list switch	—	Monitor stop	—	—	—	—	—	Border delete	Border write	—	—	—
Ctrl	—	—	Monitor start (all windows)	—	Comment display	—	Statement display	Note display	Horizontal stripe delete	Vertical stripe delete	Copy	Paste	Cut
Shift	—	Read	Monitor write mode	Conversion (write during RUN)	┤/├	└┤/├┘	—	—	—	—	—	—	—
Alt + Ctrl	—	—	Monitor stop (all windows)	Conversion (all program being edited)	—	Device name display	—	—	—	—	—	—	—

1) Press function key F2 (write), Shift + F2 (read), or F3 (monitor) to switch between each mode. To convert, press F4 (conversion) key.

2) Edit ladders using Ctrl + C (copy), Ctrl + V (paste), and Ctrl + X (cut).

3) Press F4 to convert.
Press Shift + F4 to perform write during RUN.
Press Alt + Ctrl + F4 to perform write during RUN to all programs currently being edited.

Mitsubishi Programmable Logic Controller Training Manual
Q-series basic course(for GX Developer)

MODEL	SCHOOL-Q-BASIC-WIN-E
MODEL CODE	13JW50
SH(NA)-080617ENG-A(0601)MEE	

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN